Add a permission의

AWS App Studio

AWS App Studio: Add a permission의

Table of Contents

AWS App Studio란 무엇입니까? 1
App Studio를 처음 사용하시나요? 1
개념 2
관리자 역할 2
애플리케이션(앱) 2
자동화 3
자동화 작업 3
Builder 역할 3
구성 요소 3
커넥터 3
개발 환경 4
개체 4
Instance 4
Page 4
트리거 4
App Studio 작동 방식 5
애플리케이션을 다른 서비스에 연결 6
애플리케이션의 데이터 모델 구성 7
애플리케이션의 UI 빌드 8
애플리케이션의 로직 또는 동작 구현 10
애플리케이션의 개발 수명 주기 12
자세히 알아보기 12
App Studio 설정 및 로그인 14
App Studio 인스턴스 최초 생성 및 설정 14
AWS 계정 가입 14
AWS 리소스 관리를 위한 관리 사용자 생성 15
에서 App Studio 인스턴스 생성 AWS Management Console 15
App Studio 가입 초대 수락 19
시작하기 21
자습서: AI를 사용하여 앱 생성 21
사전 조건 22
1단계: 애플리케이션 생성 22
2단계: 새 애플리케이션 탐색 23
3단계: 애플리케이션 미리 보기 25

다음 단계 25
자습서: 빈 앱에서 빌드 시작 26
사전 조건 28
1단계: 애플리케이션 생성 28
2단계: 앱의 데이터를 정의하는 엔터티 생성 29
3단계: 사용자 인터페이스(UI) 및 로직 설계 31
4단계: 애플리케이션 미리 보기 34
5단계: 애플리케이션을 테스트 환경에 게시 34
다음 단계 35
관리자 설명서 36
그룹 및 역할을 사용하여 사용자 액세스 관리 36
역할 및 권한 36
그룹 보기 37
사용자 또는 그룹 추가 37
그룹의 역할 변경 38
사용자 또는 그룹 제거 39
커넥터를 사용하여 다른 서비스에 연결 40
AWS 서비스에 연결 40
타사 서비스에 연결 83
커넥터 보기, 편집 및 삭제 91
App Studio 인스턴스 삭제 92
Builder 설명서 93
자습서 93
Amazon Bedrock을 사용하여 텍스트 요약기 앱 빌드 93
Amazon S3와 상호 작용 101
Lambda 함수 호출 110
생성형 AI로 앱 빌드 112
앱 생성 112
앱 빌드 또는 편집 112
데이터 모델 생성 113
샘플 데이터 생성 113
AWS 서비스에 대한 작업 구성 113
응답 모의 114
빌드 중 AI에 도움 요청 114
애플리케이션 생성, 편집 및 삭제 114
애플리케이션 생성 114

애플리케이션 가져오기 ... 115
애플리케이션 복제 ... 120
애플리케이션 편집 또는 빌드 ... 120
이전에 게시한 앱 버전 편집 ... 121
애플리케이션 이름 바꾸기 ... 122
애플리케이션 삭제 ... 122
애플리케이션 미리 보기, 게시 및 공유 ... 123
애플리케이션 미리 보기 ... 123
애플리케이션 게시 ... 124
게시된 애플리케이션 공유 ... 128
이전에 게시된 버전으로 롤백 ... 129
애플리케이션 내보내기 ... 130
페이지 및 구성 요소: 앱의 사용자 인터페이스 빌드 ... 131
페이지 관리 ... 131
구성 요소 관리 ... 133
페이지의 역할 기반 가시성 구성 ... 135
앱 탐색에서 페이지 정렬 및 구성 ... 137
앱 테마를 사용하여 앱의 색상 변경 ... 137
구성 요소 참조 ... 138
자동화 및 작업: 앱의 비즈니스 로직 정의 ... 182
자동화 개념 ... 183
자동화 생성, 편집 및 삭제 ... 184
자동화 작업 추가, 편집 및 삭제 ... 186
Automation 작업 참조 ... 187
개체 및 데이터 작업: 앱의 데이터 모델 구성 ... 205
데이터 모델 설계 모범 사례 ... 206
개체 생성 ... 207
개체 구성 ... 210
개체 삭제 ... 217
관리형 데이터 엔터티 ... 218
페이지 및 자동화 파라미터 ... 219
페이지 파라미터 ... 219
자동화 파라미터 ... 220
JavaScript를 사용하여 표현식 작성 ... 225
기본 구문 ... 226
보간 ... 226

연결 226
날짜 및 시간 227
코드 블록 227
글로벌 변수 및 함수 228
UI 구성 요소 값 참조 또는 업데이트 228
테이블 데이터 작업 230
자동화 액세스 231
데이터 종속성 및 타이밍 고려 사항 233
예: 주문 세부 정보 및 고객 정보 233
데이터 종속성 및 타이밍 모범 사례 233
여러 사용자로 앱 빌드 235
빌더가 앱을 편집하도록 초대 235
다른 사용자가 편집 중인 앱 편집 시도 235
앱의 콘텐츠 보안 설정 업데이트 236
문제 해결 및 디버깅 239
설정, 권한 및 온보딩 239
계정 인스턴스 생성 옵션을 선택할 때 App Studio 설정이 실패했습니다. 239
설정 후 App Studio에 액세스할 수 없음 239
App Studio에 로그인할 때 사용할 사용자 이름 또는 암호가 확실하지 않음 240
App Studio를 설정할 때 시스템 오류가 발생합니다. 240
App Studio 인스턴스 URL을 찾을 수 없음 240
App Studio에서 그룹 또는 역할을 수정할 수 없음 240
App Studio에서 오프보딩하려면 어떻게 해야 합니까? 239
앱 문제 해결 및 디버깅 241
AI 빌더 어시스턴트 241
앱 스튜디오에서 242
앱 미리 보기 242
테스트 환경에서 243
CloudWatch에서 로그 사용 245
커넥터 247
앱 게시 및 공유 250
공유 대화 상자에 새로 생성된 앱 역할이 표시되지 않습니다. 250
앱 게시가 완료될 때 이메일을 받지 못했습니다. 250
앱의 최종 사용자가 게시된 앱에 액세스할 수 없음 250
보안 251
보안 고려 사항 및 완화 조치 252

보안 고려 사항 252
보안 위험 완화 권장 사항 252
데이터 보호 253
데이터 암호화 254
전송 중 암호화 254
키 관리 254
인터네트워크 트래픽 개인 정보 보호 255
App Studio 및 Identity and Access Management 255
자격 증명 기반 정책 257
리소스 기반 정책 257
정책 작업 258
정책 리소스 258
정책 조건 키 258
ACL 259
ABAC 259
임시 자격 증명 259
엔터티 권한 259
서비스 역할 259
서비스 연결 역할 260
AWS 관리형 정책 260
서비스 연결 역할 264
ID 기반 정책 예시 267
규정 준수 확인 270
복원력 271
인프라 보안 271
구성 및 취약성 분석 271
교차 서비스 혼동된 대리인 방지 271
리전 간 데이터 전송 272
지원되는 브라우저 274
애플리케이션 구축을 위한 지원 및 권장 브라우저 274
애플리케이션 최종 사용자를 위한 지원 및 권장 브라우저 274
브라우저 설정을 업데이트하여 App Studio에서 앱 빌드 274
할당량 276
문서 이력 277
..... cclxxxvi

AWS App Studio란 무엇입니까?

AWS App Studio는 자연어를 사용하여 엔터프라이즈급 애플리케이션을 생성하는 데 도움이 되는 생성형 AI 기반 서비스입니다. App Studio는 IT 프로젝트 관리자, 데이터 엔지니어, 엔터프라이즈 아키텍트와 같은 소프트웨어 개발 기술 없이 기술 전문가에게 애플리케이션 개발을 시작합니다. App Studio를 사용하면 운영 전문 지식 AWS없이에서 안전하고 완벽하게 관리하는 애플리케이션을 빠르게 구축할 수 있습니다.

빌더는 App Studio를 사용하여 앱을 생성하고 배포하여 내부 비즈니스 프로세스를 현대화할 수 있습니다. 일부 사용 사례 예로는 재고 관리 및 추적, 클레임 처리, 직원 생산성과 고객 성과를 개선하기 위한 복잡한 승인 등이 있습니다.

주제

- App Studio를 처음 사용하시나요?

App Studio를 처음 사용하시나요?

App Studio를 처음 사용하는 경우 먼저 다음 섹션을 읽는 것이 좋습니다.

- App Studio를 설정하고, 사용자 및 액세스를 관리하고, 다른 AWS 또는 타사 서비스로 커넥터를 구성할 관리자 역할이 있는 사용자의 경우 AWS App Studio 개념 및 섹션을 참조하세요AWS App Studio 설정 및 로그인.
- 애플리케이션을 생성하고 개발할 빌더는 AWS App Studio 개념 및 단원을 참조하십시오AWS App Studio 시작하기.

AWS App Studio 개념

주요 App Studio 개념을 숙지하여 팀을 위한 애플리케이션 생성 및 프로세스 자동화 속도를 높이세요. 이러한 개념에는 관리자 및 빌더 모두에 대해 App Studio 전체에서 사용되는 용어가 포함됩니다.

주제

- 관리자 역할
- 애플리케이션(앱)
- 자동화
- 자동화 작업
- Builder 역할
- 구성 요소
- 커넥터
- 개발 환경
- 개체
- Instance
- Page
- 트리거

관리자 역할

Admin은 App Studio에서 그룹에 할당할 수 있는 역할입니다. 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.

관리자 역할을 가진 사용자만 역할, 데이터 소스 및 애플리케이션을 관리하는 도구가 포함된 Admin Hub에 액세스할 수 있습니다.

애플리케이션(앱)

애플리케이션(앱)은 최종 사용자가 특정 작업을 수행할 수 있도록 개발된 단일 소프트웨어 프로그램입니다. App Studio의 앱에는 UI 페이지 및 구성 요소, 자동화, 사용자가 상호 작용할 수 있는 데이터 소스와 같은 자산이 포함됩니다.

자동화

자동화는 애플리케이션의 비즈니스 로직을 정의하는 방법입니다. 자동화의 주요 구성 요소는 자동화를 시작하는 트리거, 하나 이상의 작업 시퀀스, 자동화에 데이터를 전달하는 데 사용되는 입력 파라미터, 출력입니다.

자동화 작업

일반적으로 작업이라고 하는 자동화 작업은 자동화를 구성하는 로직의 개별 단계입니다. 각 작업은 이메일 전송, 데이터 레코드 생성, Lambda 함수 호출 또는 APIs 호출 등 특정 작업을 수행합니다. 작업은 작업 라이브러리의 자동화에 추가되며 조건문 또는 루프로 그룹화할 수 있습니다.

Builder 역할

Builder는 App Studio의 그룹에 할당할 수 있는 역할입니다. 빌더는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.

Builder 역할이 있는 사용자는 Builder Hub에 액세스할 수 있습니다. 여기에는 빌더가 액세스할 수 있는 애플리케이션과 같은 리소스에 대한 세부 정보와 학습 리소스와 같은 유용한 정보가 포함되어 있습니다.

구성 요소

구성 요소는 애플리케이션의 UI 내에 있는 개별 기능 항목입니다. 구성 요소는 페이지에 포함되어 있으며 일부 구성 요소는 다른 구성 요소의 컨테이너 역할을 할 수 있습니다. 구성 요소는 UI 요소를 해당 UI 요소가 수행할 비즈니스 로직과 결합합니다. 예를 들어, 한 가지 유형의 구성 요소는 사용자가 필드에 정보를 입력할 수 있는 양식이며, 제출되면 해당 정보가 데이터베이스 레코드로 추가됩니다.

커넥터

커넥터는 App Studio와 및 AWS Lambda Amazon Redshift 또는 타사 서비스와 같은 다른 AWS 서비스 간의 연결입니다. 커넥터가 생성되고 구성되면 빌더는 커넥터와 해당 애플리케이션에서 App Studio에 연결되는 리소스를 사용할 수 있습니다.

관리자 역할을 가진 사용자만 커넥터를 생성, 관리 또는 삭제할 수 있습니다.

개발 환경

개발 환경은 애플리케이션을 빌드하기 위한 시각적 도구입니다. 이 환경에는 앱을 빌드하기 위한 다음 탭이 포함되어 있습니다.

- 페이지: 빌더가 페이지와 구성 요소를 사용하여 애플리케이션을 설계하는 위치입니다.
- 자동화: 빌더가 자동화를 통해 애플리케이션의 비즈니스 로직을 설계하는 위치입니다.
- 데이터: 빌더가 엔터티를 사용하여 애플리케이션의 데이터 모델을 설계하는 위치입니다.

개발 환경에는 디버그 콘솔과 빌드 중에 컨텍스트 지원을 받을 수 있는 AI 채팅 창도 포함되어 있습니다. 빌더는 개발 환경에서 진행 중인 애플리케이션을 미리 볼 수 있습니다.

개체

개체는 App Studio의 데이터 테이블입니다. 개체는 데이터 소스의 테이블과 직접 상호 작용합니다. 개체에는 데이터를 설명하는 필드, 데이터를 찾아 반환하는 쿼리, 개체의 필드를 데이터 소스의 열에 연결하는 매핑이 포함됩니다.

Instance

인스턴스는 모든 App Studio 리소스에 대한 논리적 컨테이너입니다. 사용자, 회사, 팀 또는 조직을 나타내며 사용자 및 그룹에 대한 애플리케이션, 커넥터 및 역할 할당과 같은 모든 App Studio 리소스를 포함합니다. 대규모 조직 또는 엔터프라이즈에는 일반적으로 샌드박스, 테스트 및 프로덕션 인스턴스와 같은 여러 App Studio 인스턴스가 있습니다. App Studio 설정의 일부로 인스턴스를 생성합니다.

Page

페이지는 App Studio에서 애플리케이션의 UI를 구성하는 구성 요소의 컨테이너입니다. 각 페이지는 사용자가 상호 작용할 애플리케이션의 사용자 인터페이스(UI) 화면을 나타냅니다. 페이지는 애플리케이션 스튜디오의 페이지 탭에서 생성 및 편집됩니다.

트리거

트리거는 자동화를 실행할 시기와 조건을 결정합니다. 트리거의 몇 가지 예는 `On click` 버튼과 텍스트 입력`On select`입니다. 구성 요소 유형에 따라 해당 구성 요소에 사용 가능한 트리거 목록이 결정됩니다. 트리거는 구성 요소에 추가되고 애플리케이션 스튜디오에서 구성됩니다.

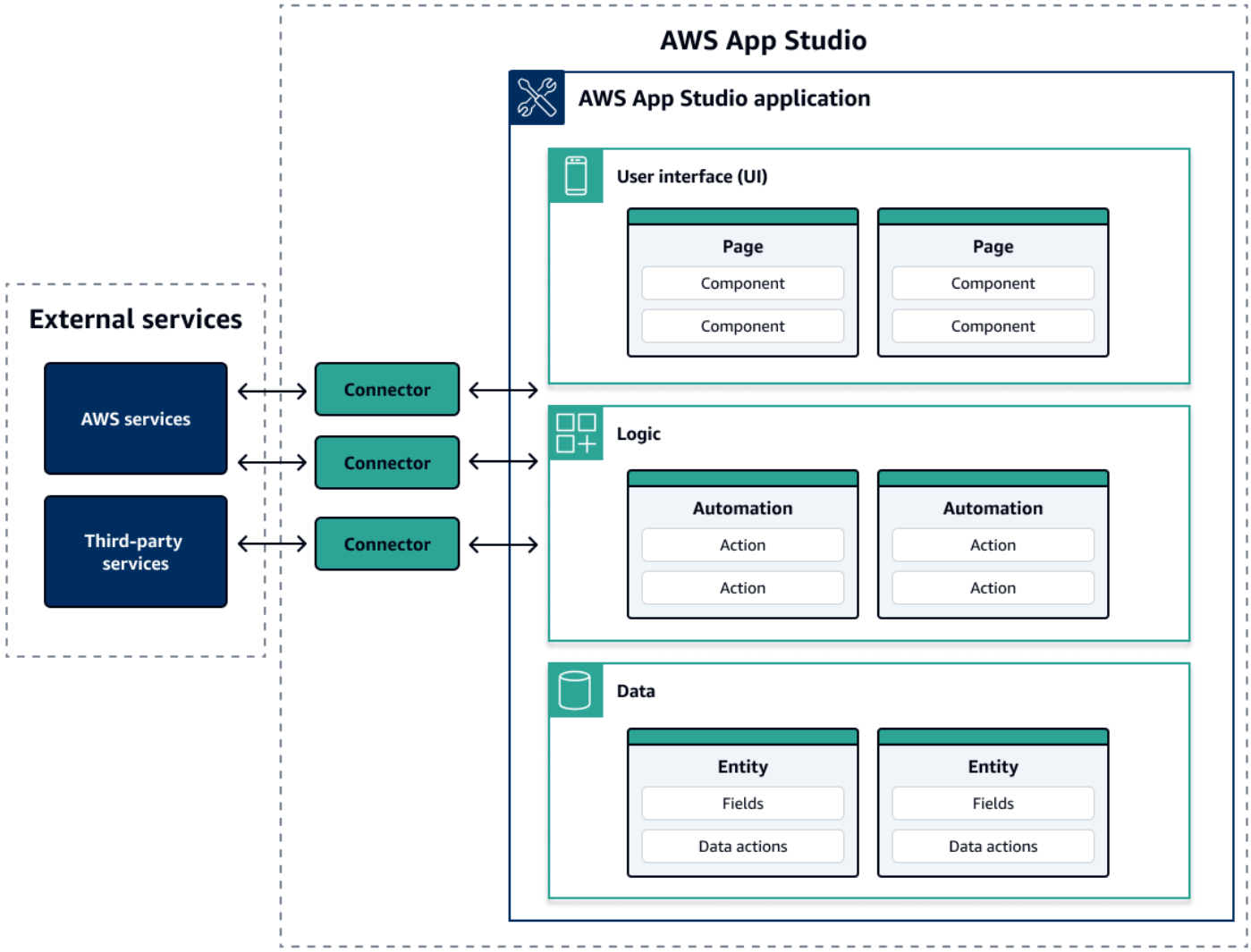
AWS App Studio 작동 방식

AWS App Studio를 사용하여 애플리케이션을 빌드할 때 이해해야 할 몇 가지 주요 개념이 있습니다. 이 주제에서는 다음 개념 또는 리소스의 기본 사항을 다룹니다.

- 커넥터를 사용하여 다른 서비스에 연결하여 애플리케이션에서 리소스 또는 API 호출을 사용합니다. 예를 들어 커넥터를 사용하여 데이터를 저장 및 액세스하거나 앱에서 알림을 보낼 수 있습니다.
- 엔터티를 사용하여 애플리케이션의 데이터 모델을 구성하여 애플리케이션을 외부 데이터 소스와 연결합니다.
- 페이지 및 구성 요소를 사용하여 애플리케이션의 사용자 인터페이스(UI)를 빌드합니다.
- 자동화 및 작업을 사용하여 애플리케이션의 로직 또는 동작을 구현합니다.
- App Studio의 애플리케이션 개발 수명 주기: 빌드, 테스트 및 게시.

App Studio 개념에 대한 자세한 내용은 섹션을 참조하세요AWS App Studio 개념.

다음 이미지는 App Studio와 해당 리소스가 구성되는 방법을 보여주는 간단한 다이어그램입니다.



App Studio의 앱 내에서 페이지, 자동화 및 엔터티는 모두 서로 상호 작용합니다. 커넥터를 사용하여 이러한 리소스를 데이터, 스토리지 또는 알림 공급자와 같은 외부 서비스에 연결합니다. 앱을 성공적으로 빌드하려면 이러한 모든 개념과 리소스가 서로 어떻게 상호 작용하는지 이해하는 것이 중요합니다.

애플리케이션을 다른 서비스에 연결

App Studio를 사용하여 애플리케이션을 구축할 때 가장 큰 이점 중 하나는 앱을 다른 서비스와 쉽게 통합할 수 있다는 것입니다. App Studio에서는 애플리케이션에 사용하려는 서비스 및 리소스 또는 API 호출과 관련된 커넥터를 사용하여 다른 서비스에 연결합니다.

개별 앱이 아닌 App Studio 인스턴스 수준에서 커넥터를 생성합니다. 커넥터를 생성한 후에는 연결된 서비스 및 애플리케이션에 따라 애플리케이션의 다양한 부분에서 사용할 수 있습니다.

다음은 커넥터를 사용하여 다른 서비스에 연결하는 애플리케이션의 기능 예제입니다.

- 거의 모든 애플리케이션에서 사용되는 가장 일반적인 사용 사례는 Amazon Redshift, Amazon DynamoDB 또는 Amazon Aurora와 같은 데이터 서비스에 연결하여 애플리케이션에 사용되는 AWS 데이터를 저장하고 액세스하는 것입니다.
- 영수증과 같은 이미지를 업로드하고 볼 수 있는 애플리케이션은 Amazon S3를 사용하여 이미지 파일을 저장하고 액세스할 수 있습니다.
- 텍스트 요약기 앱은 Amazon Bedrock에 텍스트 입력을 보내고 반환된 요약을 표시할 수 있습니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다. 커넥터를 생성할 때 사용할 리소스 또는 API 호출에 대한 적절한 자격 증명과 정보를 포함해야 합니다.

애플리케이션의 데이터 모델 구성

애플리케이션의 데이터는 애플리케이션을 구동하는 정보입니다. App Studio에서는 저장하고 작업하는 다양한 유형의 데이터를 나타내는 엔터티를 생성하고 사용합니다. 예를 들어 고객 회의를 위한 추적 애플리케이션에는 고객 회의, 의제 및 참석자를 나타내는 세 개의 엔터티가 있을 수 있습니다.

개체에는 저장 중인 데이터를 설명하는 정수 또는 문자열과 같은 유형이 있는 필드가 포함되어 있습니다. 엔터티를 사용하여 데이터 모델을 정의하더라도 엔터티를 Amazon Redshift 또는 Amazon DynamoDB와 같은 외부 데이터 스토리지 서비스에 연결하여 데이터를 저장해야 합니다. 개체는 App Studio 애플리케이션과 외부 서비스의 데이터 간의 중개자라고 생각할 수 있습니다.

데이터 작업을 사용하여 구성 요소 및 자동화에서 애플리케이션의 데이터와 상호 작용할 수 있습니다. 사용할 가장 일반적인 두 가지 데이터 작업은 `getAll` 작업과 `getByID` 작업입니다. 예를 들어 애플리케이션은 `getAll` 데이터 작업을 사용하여 테이블을 데이터로 채우고 세부 정보 구성 요소를 특정 데이터 항목에 대한 자세한 정보로 채우는 `getByID` 작업을 사용할 수 있습니다.

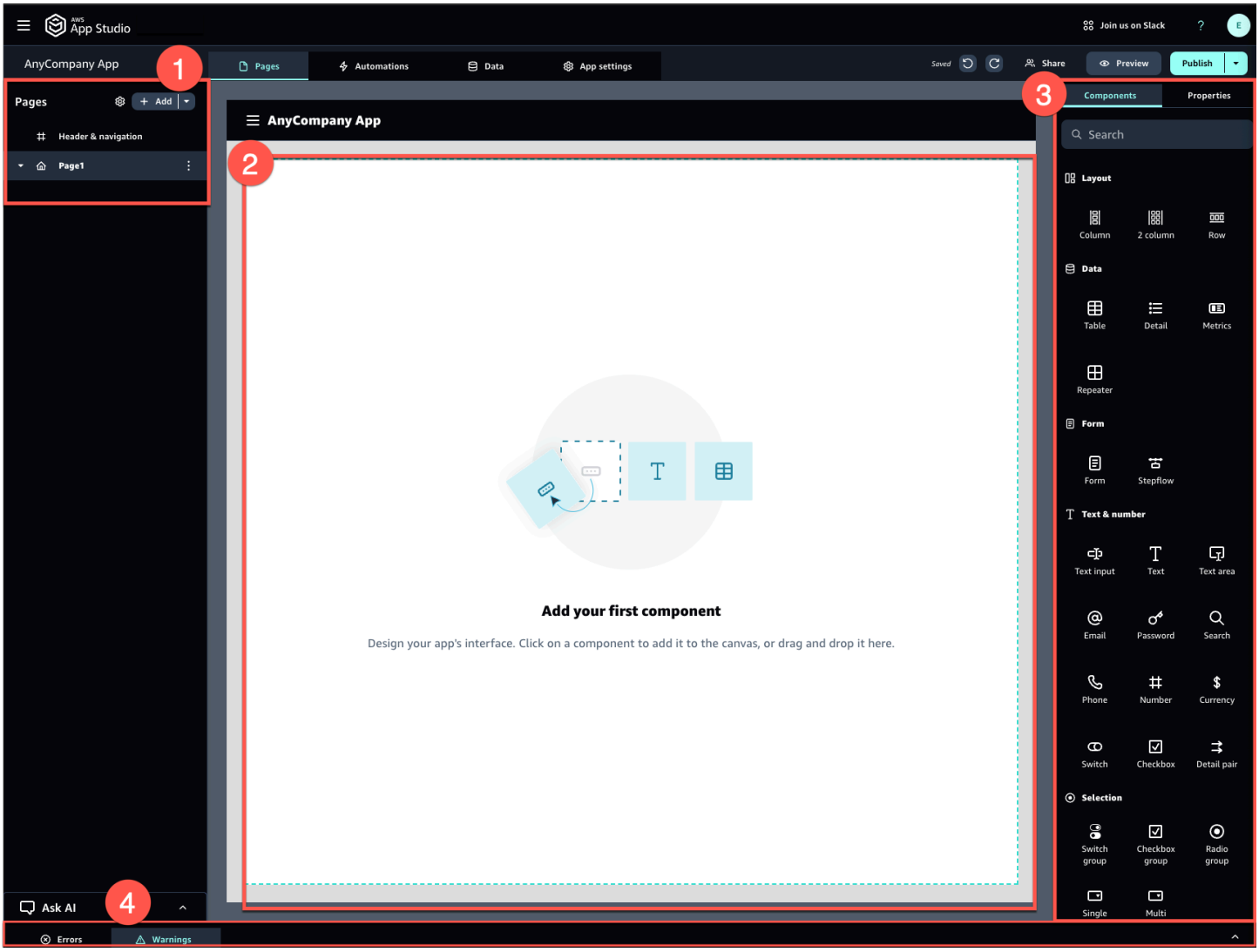
또한 외부 서비스를 호출할 필요 없이 엔터티에 샘플 데이터를 추가하여 애플리케이션을 더 쉽게 테스트할 수 있습니다.

애플리케이션의 UI 빌드

App Studio에서는 페이지 및 구성 요소를 사용하여 애플리케이션의 UI를 빌드합니다. 페이지는 애플리케이션의 개별 화면이며 구성 요소의 컨테이너입니다. 구성 요소는 애플리케이션 UI의 구성 요소입니다. 테이블, 양식, 이미지 뷰어, 버튼 등 다양한 유형의 구성 요소가 있습니다.

다음 이미지는 애플리케이션에서 페이지와 구성 요소를 추가하거나 구성하는 애플리케이션 스튜디오의 페이지 탭을 보여줍니다. 다음 키 영역이 강조 표시되고 번호가 매겨집니다.

1. 왼쪽 페이지 패널입니다. 여기에서 페이지, 애플리케이션 헤더 및 탐색 설정을 관리합니다. 애플리케이션의 모든 페이지와 구성 요소를 볼 수 있습니다.
2. 캔버스 - 현재 페이지의 구성 요소를 표시합니다. 캔버스에서 구성 요소를 선택하여 해당 속성을 구성할 수 있습니다.
3. 오른쪽 구성 요소 또는 속성 패널입니다. 아무 것도 선택하지 않으면 페이지에 추가할 수 있는 구성 요소 목록이 표시되는 구성 요소 패널이 표시됩니다. 페이지 또는 구성 요소를 선택하면 페이지 또는 구성 요소를 구성하는 속성 패널이 표시됩니다.
4. 하단 오류 및 경고 패널. 이러한 패널에는 가장 일반적으로 구성 문제로 인한 애플리케이션의 오류 또는 경고가 표시됩니다. 패널을 선택하여 확장하고 메시지를 볼 수 있습니다.



예를 들어 사용자가 정보를 입력해야 하는 애플리케이션에는 다음 페이지와 구성 요소가 있을 수 있습니다.

- 사용자가 정보를 작성하고 제출하는 데 사용하는 양식 구성 요소가 포함된 입력 페이지입니다.
- 각 입력에 대한 정보가 포함된 테이블 구성 요소가 포함된 목록 보기 페이지입니다.
- 각 입력에 대한 자세한 정보가 포함된 세부 정보 구성 요소가 포함된 세부 보기 페이지입니다.

구성 요소에는 정의된 필드가 있는 양식과 같은 정적 정보 또는 데이터가 포함될 수 있습니다. Amazon S3 버킷에서 이미지를 검색하여 사용자에게 표시하는 이미지 뷰어와 같은 자동화를 사용하여 동적 정보를 포함할 수도 있습니다.

페이지 파라미터의 개념을 이해하는 것이 중요합니다. 페이지 파라미터를 사용하여 한 페이지에서 다른 페이지로 정보를 전송합니다. 페이지 파라미터 사용 사례의 일반적인 예는 검색 및 필터링입니다.

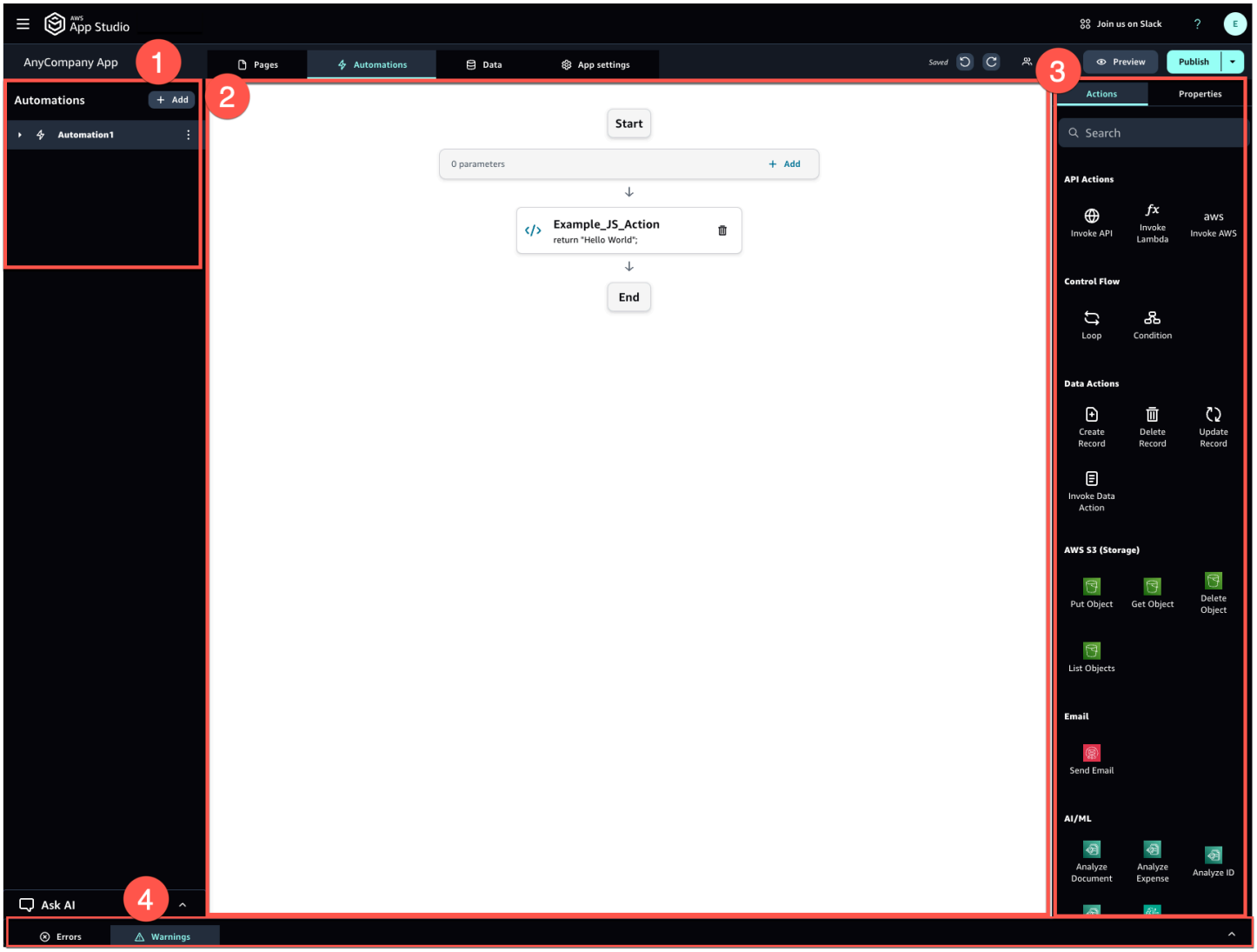
여기서 한 페이지의 검색어는 다른 페이지의 필터링할 항목 목록 또는 테이블로 전송됩니다. 또 다른 사용 사례 예제는 항목 식별자가 세부 뷰어 페이지로 전송되는 항목 세부 정보를 보는 것입니다.

애플리케이션의 로직 또는 동작 구현

애플리케이션의 로직 또는 동작을 애플리케이션의 기능으로 생각할 수 있습니다. 사용자가 버튼을 선택하거나, 정보를 제출하거나, 새 페이지로 이동하거나, 다른 방식으로 상호 작용할 때 발생하는 상황을 정의할 수 있습니다. App Studio에서는 자동화 및 작업을 사용하여 애플리케이션의 로직을 정의합니다. 자동화는 자동화 기능의 구성 요소인 작업의 컨테이너입니다.

다음 이미지는 애플리케이션에서 자동화 및 해당 작업을 추가하거나 구성하는 애플리케이션 스튜디오의 자동화 탭을 보여줍니다. 다음 키 영역이 강조 표시되고 번호가 매겨집니다.

- 왼쪽 자동화 패널입니다. 여기에서 자동화를 관리합니다. 애플리케이션의 모든 자동화 및 작업을 볼 수 있습니다.
- 캔버스 - 현재 자동화를 표시합니다. 구성된 자동화 파라미터(이 섹션의 뒷부분에서 설명)와 작업이 표시됩니다. 캔버스에서 구성 요소를 선택하여 해당 속성을 구성할 수 있습니다.
- 오른쪽 작업 및 속성 패널입니다. 아무 것도 선택하지 않으면 작업 패널이 표시됩니다. 자동화에 추가할 수 있는 작업 목록이 표시됩니다. 자동화를 선택하면 자동화의 입력 및 출력과 같은 속성을 보고 구성할 수 있습니다. 작업을 선택하면 작업의 속성을 보고 구성할 수 있습니다.
- 하단 오류 및 경고 패널. 이 패널에는 애플리케이션의 오류 또는 경고가 표시됩니다(가장 일반적으로 구성 문제로 인해). 패널을 선택하여 확장하고 메시지를 볼 수 있습니다.



자동화는 간단하거나(예: 숫자 추가 및 결과 반환) 더 강력할 수 있습니다(예: 다른 서비스에 입력 전송 및 결과 반환). 자동화의 주요 구성 요소는 다음과 같습니다.

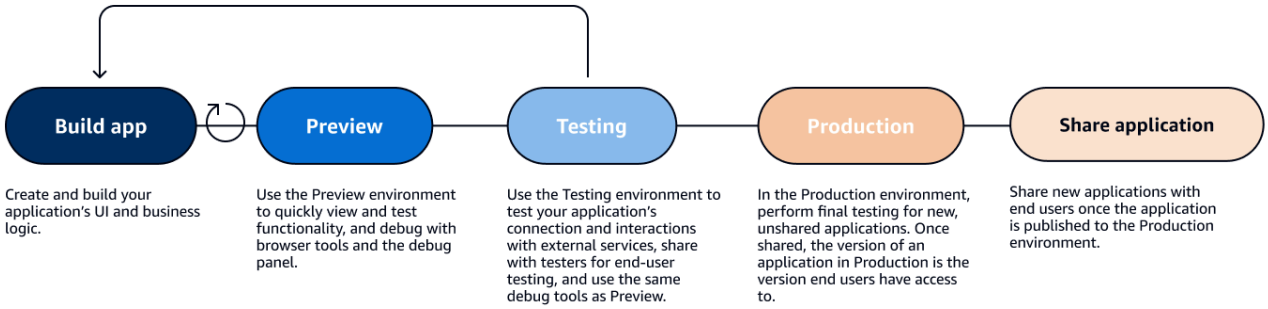
- 자동화가 실행되는 시기를 정의하는 트리거입니다. 사용자가 UI에서 버튼을 누르는 경우를 예로 들 수 있습니다.
- 자동화에 정보를 전송하는 자동화 입력입니다. 자동화 파라미터를 사용하여 자동화 입력을 정의합니다. 예를 들어 Amazon Bedrock을 사용하여 사용자에게 텍스트 요약을 반환하려는 경우 자동화 파라미터로 요약되도록 텍스트를 구성합니다.
- 자동화 기능의 구성 요소입니다. 각 작업을 자동화의 단계로 생각할 수 있습니다. 작업은 APIs 호출하고, 사용자 지정 JavaScript를 호출하고, 데이터 레코드를 생성하고, 다른 함수를 수행할 수 있습니다. 작업을 루프 또는 조건으로 그룹화하여 기능을 추가로 사용자 지정할 수도 있습니다. 작업을 사용하여 다른 자동화를 호출할 수도 있습니다.

- 구성 요소 또는 기타 자동화에 사용할 수 있는 자동화 출력입니다. 예를 들어 자동화 출력은 텍스트 구성 요소에 표시되는 텍스트, 이미지 뷰어 구성 요소에 표시되는 이미지 또는 다른 자동화에 대한 입력일 수 있습니다.

애플리케이션의 개발 수명 주기

애플리케이션의 개발 수명 주기에는 구축, 테스트 및 게시 단계가 포함됩니다. 애플리케이션을 생성하고 반복할 때 이러한 단계를 통해 그리고 단계 간에 반복할 가능성이 높기 때문에 이를 주기라고 합니다.

다음 이미지는 App Studio에서 애플리케이션 개발 수명 주기의 간소화된 타임라인을 보여줍니다.



App Studio는 애플리케이션의 수명 주기를 지원하는 다양한 도구를 제공합니다. 이러한 도구에는 이전 다이어그램에 표시된 다음과 같은 세 가지 개별 환경이 포함됩니다.

- 애플리케이션을 미리 보고 최종 사용자에게 어떻게 보이는지 확인하고 특정 기능을 테스트할 수 있는 미리 보기 환경입니다. 미리 보기 환경을 사용하면 애플리케이션을 게시할 필요 없이 애플리케이션을 빠르게 테스트하고 반복할 수 있습니다. 미리 보기 환경의 애플리케이션은 외부 서비스와 통신하거나 데이터를 전송하지 않습니다. 즉, 미리 보기 환경에서 외부 서비스에 의존하는 상호 작용 및 기능을 테스트할 수 없습니다.
- 애플리케이션의 외부 서비스와의 연결 및 상호 작용을 테스트할 수 있는 테스트 환경입니다. 또한 테스트 환경에 게시된 버전을 테스터 그룹에 공유하여 최종 사용자 테스트를 수행할 수 있습니다.
- 새 앱을 최종 사용자와 공유하기 전에 최종 테스트를 수행할 수 있는 프로덕션 환경입니다. 앱을 공유한 후 프로덕션 환경에 게시되는 애플리케이션 버전은 최종 사용자가 보고 사용할 버전입니다.

자세히 알아보기

이제 App Studio에서 애플리케이션 개발이 작동하는 방식에 대한 기본 사항을 알았으므로 직접 애플리케이션을 빌드하거나 개념 및 리소스에 대해 자세히 알아볼 수 있습니다.

빌드를 시작하려면 시작하기 자습서 중 하나를 시도하는 것이 좋습니다.

- AI 자습서: AI를 사용하여 앱 생성 빌더 어시스턴트를 사용하여 앱 빌드를 시작하는 방법을 알아보세요.
- 기본 자습서: 빈 앱에서 빌드 시작 사항을 학습하면서 처음부터 앱을 빌드하는 방법을 알아봅니다.

이 주제에서 언급한 리소스 또는 개념에 대해 자세히 알아보려면 다음 주제를 참조하세요.

- 커넥터를 사용하여 App Studio를 다른 서비스에 연결
- 개체 및 데이터 작업: 앱의 데이터 모델 구성
- 페이지 및 구성 요소: 앱의 사용자 인터페이스 빌드
- 자동화 및 작업: 앱의 비즈니스 로직 정의
- 애플리케이션 미리 보기, 게시 및 공유

AWS App Studio 설정 및 로그인

AWS App Studio 설정은 역할에 따라 다릅니다.

- AWS 또는 조직 관리자로 처음 설정: 관리자로 App Studio를 처음 설정하려면 계정이 없는 경우 AWS 계정을 생성하고, App Studio 인스턴스를 생성하고, IAM Identity Center 그룹을 사용하여 사용자 액세스를 구성합니다. 인스턴스가 생성된 후 App Studio에서 관리자 역할을 가진 사람은 누구나 다른 서비스(예: 데이터 소스)를 App Studio 인스턴스에 연결하도록 커넥터를 구성하는 등의 작업을 추가로 설정할 수 있습니다. 최초 설정에 대한 자세한 내용은 섹션을 참조하세요App Studio 인스턴스 최초 생성 및 설정.
- 빌더로 시작하기: App Studio에 빌더로 가입하라는 초대를 받으면 초대를 수락하고 암호를 제공하여 IAM Identity Center 사용자 자격 증명을 활성화해야 합니다. 그런 다음 App Studio에 로그인하여 애플리케이션 빌드를 시작할 수 있습니다. 초대 수락 및 App Studio 인스턴스 가입에 대한 자세한 내용은 섹션을 참조하세요App Studio 가입 초대 수락.

App Studio 인스턴스 최초 생성 및 설정

AWS 계정 가입

App Studio를 설정하려면 AWS 계정이 필요합니다. App Studio를 사용하려면 하나의 AWS 계정만 필요합니다. AWS IAM Identity Center에서 액세스를 관리하므로 빌더와 관리자는 App Studio를 사용하는 데 AWS 계정이 필요하지 않습니다.

를 생성하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따르세요.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   에 가입하면 AWS 계정AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 리소스 관리를 위한 관리 사용자 생성

AWS 계정을 처음 생성할 때 계정의 모든 AWS 리소스에 대한 전체 액세스 권한이 있는 기본 자격 증명 세트로 시작합니다. 이 자격 증명을 AWS 계정 루트 사용자라고 합니다. App Studio에서 사용할 AWS 역할 및 리소스를 생성하려면 AWS 계정 루트 사용자를 사용하지 않는 것이 좋습니다. 대신 관리 사용자를 생성하고 사용하는 것이 좋습니다.

다음 주제를 사용하여 App Studio에서 사용할 AWS 역할 및 리소스를 관리하기 위한 관리 사용자를 생성합니다.

- 단일 독립 실행형 AWS 계정은 IAM 사용 설명서의 첫 번째 IAM 사용자 생성을 참조하세요. 모든 사용자 이름을 제공할 수 있지만 `AdministratorAccess` 권한 정책이 있어야 합니다.
- 를 통해 관리되는 여러 AWS 계정의 경우 AWS IAM Identity Center 사용 설명서의 IAM Identity Center 관리 사용자에 대한 AWS 계정 액세스 설정을 AWS Organizations참조하세요.

에서 App Studio 인스턴스 생성 AWS Management Console

App Studio를 사용하려면의 App Studio 랜딩 페이지에서 인스턴스를 생성해야 합니다 AWS Management Console. App Studio 인스턴스를 생성하는 데 사용할 수 있는 두 가지 방법이 있습니다.

1. 간편한 생성:이 간소화된 방법을 사용하면 설정의 일부로 App Studio에 액세스하고 사용하도록 한 명의 사용자만 설정할 수 있습니다. 조직 또는 팀에 대해 App Studio를 평가하거나 App Studio만 직접 사용할 계획인 경우이 방법을 사용해야 합니다. 설정 후 App Studio에 사용자 또는 그룹을 더 추가할 수 있습니다. IAM Identity Center의 조직 인스턴스가 있는 경우이 방법을 사용할 수 없습니다.
2. 표준 생성:이 방법을 사용하면 설정의 일부로 App Studio에서 사용자 또는 그룹을 추가하고 역할을 할당할 수 있습니다. 설정할 때 App Studio에 사용자를 두 명 이상 추가하려면이 방법을 사용해야 합니다.

> **Note**
>
> 모든 AWS 리전에서 App Studio 인스턴스를 하나만 생성할 수 있습니다. 기존 인스턴스가 있는 경우 다른 인스턴스를 생성하기 전에 삭제해야 합니다. 자세한 내용은 App Studio 인스턴스 삭제 단원을 참조하십시오.

Easy create

에서 쉽게 생성할 수 AWS Management Console 있는 App Studio 인스턴스를 생성하려면

1. https://console.aws.amazon.com/appstudio/ App Studio 콘솔을 엽니다.
2. App Studio 인스턴스를 생성하려는 AWS 리전으로 이동합니다.
3. Get started를 선택합니다.
4. 간편 생성을 선택하고 다음을 선택합니다.
5. App Studio를 설정하는 다음 단계는 IAM Identity Center 계정 인스턴스가 있는지 여부에 따라 결정됩니다. 다양한 유형과 사용 중인 유형을 찾는 방법을 포함하여 IAM Identity Center 인스턴스에 대한 자세한 내용은 IAM Identity Center 사용 설명서의 IAM Identity Center의 조직 및 계정 인스턴스 관리를 참조하세요. AWS
   - IAM Identity Center의 계정 인스턴스가 있는 경우:
     a. 계정 권한에서 App Studio를 활성화하는 데 필요한 권한을 검토합니다. 계정에 필요한 권한이 없는 경우 App Studio를 활성화할 수 없습니다. 계정에 필요한 권한을 추가하거나 권한이 있는 계정으로 전환해야 합니다.
     b. 사용자 추가에서 App Studio에 액세스할 IAM Identity Center 계정 인스턴스에서 사용자의 이메일 주소를 검색하고 선택합니다. 이 사용자는 App Studio 인스턴스에서 관리자 역할을 갖게 됩니다. App Studio에 대한 액세스를 제공하려는 사용자가 표시되지 않으면 IAM Identity Center 인스턴스에 추가해야 할 수 있습니다.
   - IAM Identity Center의 계정 인스턴스가 없는 경우:

     > **Note**
     >
     > App Studio를 설정하면 설정 프로세스 중에 구성한 사용자와 함께 IAM Identity Center 계정 인스턴스가 자동으로 생성됩니다. 설정이 완료되면 https://console.aws.amazon.com/singlesignon/ IAM Identity Center 콘솔에서 사용자 및 그룹을 추가하거나 관리할 수 있습니다.

     a. 계정 권한에서 App Studio를 활성화하는 데 필요한 권한을 검토합니다. 계정에 필요한 권한이 없는 경우 App Studio를 활성화할 수 없습니다. 계정에 필요한 권한을 추가하거나 권한이 있는 계정으로 전환해야 합니다.

b. 사용자 추가에서 App Studio에 액세스하는 사용자의 이메일 주소, 이름, 성 및 사용자 이름을 입력합니다. 이 사용자는 App Studio 인스턴스에서 관리자 역할을 갖게 됩니다.

6. 서비스 액세스 및 역할에서 서비스에 필요한 권한을 제공하도록 App Studio를 설정할 때 자동으로 생성되는 서비스 역할 및 서비스 연결 역할을 검토합니다. 권한 보기를 선택하여 서비스 역할에 부여된 정확한 권한을 보거나 정책 보기를 선택하여 서비스 연결 역할에 연결된 권한 정책을 확인합니다.
7. 승인에서 확인란을 선택하여 문을 확인합니다.
8. 설정을 선택하여 인스턴스를 생성합니다.

> **Note**
>
> 설정 후 App Studio 인스턴스에 사용자 또는 그룹을 더 추가하려면 IAM Identity Center 인스턴스에 추가해야 합니다.

Standard create

표준 메서드를 AWS Management Console 사용하여에서 App Studio 인스턴스를 생성하려면

1. https://console.aws.amazon.com/appstudio/ App Studio 콘솔을 엽니다.
2. App Studio 인스턴스를 생성하려는 AWS 리전으로 이동합니다.
3. Get started를 선택합니다.
4. 표준 생성을 선택하고 다음을 선택합니다.
5. App Studio를 설정하는 단계는 IAM Identity Center 인스턴스가 있는지 여부와 인스턴스 유형에 따라 결정됩니다. 다양한 유형과 사용 중인 유형을 찾는 방법을 포함하여 IAM Identity Center 인스턴스에 대한 자세한 내용은 IAM Identity Center 사용 설명서의 IAM Identity Center의 조직 및 계정 인스턴스 관리를 참조하세요. AWS
   - IAM Identity Center의 조직 인스턴스가 있는 경우:
     - Single Sign-On을 사용하여 App Studio에 대한 액세스 구성에서 기존 IAM Identity Center 그룹을 선택하여 App Studio에 대한 액세스 권한을 제공합니다. App Studio 그룹은 지정된 구성을 기반으로 생성됩니다. 관리자 그룹에 추가된 그룹의 구성원은 관리자 역할을 갖게 되고, Builder 그룹에 추가된 그룹의 구성원은 App Studio에서 Builder 역할을 갖게 됩니다. 역할은 다음과 같이 정의됩니다.

- 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
- 빌더는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.

- IAM Identity Center 인스턴스의 계정 인스턴스가 있는 경우:

  a. 계정 권한에서 App Studio를 활성화하는 데 필요한 권한을 검토합니다. 계정에 필요한 권한이 없는 경우 App Studio를 활성화할 수 없습니다. 계정에 필요한 권한을 추가하거나 권한이 있는 계정으로 전환해야 합니다.

  b. Single Sign-On을 사용하여 App Studio에 대한 액세스 구성의 IAM Identity Center 계정에서 기존 계정 인스턴스 사용을 선택합니다.

  c. AWS 리전에서 IAM Identity Center 계정 인스턴스가 위치한 리전을 선택합니다.

  d. 기존 IAM Identity Center 그룹을 선택하여 App Studio에 대한 액세스 권한을 제공합니다. App Studio 그룹은 지정된 구성을 기반으로 생성됩니다. 관리자 그룹에 추가된 그룹의 구성원은 관리자 역할을 갖게 되고, Builder 그룹에 추가된 그룹의 구성원은 App Studio에서 Builder 역할을 갖게 됩니다. 역할은 다음과 같이 정의됩니다.

  - 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
  - 빌더는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.

- IAM Identity Center 인스턴스가 없는 경우:

> **Note**
>
> App Studio 설정은 설정 프로세스 중에 구성한 그룹으로 IAM Identity Center 계정 인스턴스를 자동으로 생성합니다. 설정이 완료되면 https://console.aws.amazon.com/singlesignon/ IAM Identity Center 콘솔에서 사용자 및 그룹을 추가하거나 관리할 수 있습니다.

a. 계정 권한에서 App Studio를 활성화하는 데 필요한 권한을 검토합니다. 계정에 필요한 권한이 없는 경우 App Studio를 활성화할 수 없습니다. 계정에 필요한 권한을 추가하거나 권한이 있는 계정으로 전환해야 합니다.

b. Single Sign-On을 사용하여 App Studio에 대한 액세스 구성의 IAM Identity Center 계정에서 계정 인스턴스 생성을 선택합니다.

c. 사용자 및 그룹 생성 및 App Studio에 추가에서 이름을 제공하고 관리자 그룹 및 빌더 그룹에 사용자를 추가합니다. 관리자 그룹에 추가된 사용자는 App Studio에서 관리자 역할을 갖게 되고, 빌더 그룹에 추가된 사용자는 빌더 역할을 갖게 됩니다. 역할은 다음과 같이 정의됩니다.

- 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
- 빌더는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.

> **Important**
>
> App Studio를 설정하고 설정 후 관리자 액세스 권한을 가지려면 관리자 그룹의 사용자로 자신을 추가해야 합니다.

6. 서비스 액세스 및 역할에서 서비스에 필요한 권한을 제공하도록 App Studio를 설정할 때 자동으로 생성되는 서비스 역할 및 서비스 연결 역할을 검토합니다. 권한 보기를 선택하여 서비스 역할에 부여된 정확한 권한을 보거나 정책 보기를 선택하여 서비스 연결 역할에 연결된 권한 정책을 확인합니다.
7. 승인에서 확인란을 선택하여 문을 확인합니다.
8. 설정을 선택하여 인스턴스를 생성합니다.

App Studio 가입 초대 수락

App Studio에 대한 액세스는 IAM Identity Center에서 관리합니다. 즉, App Studio를 사용하려는 각 사용자는 IAM Identity Center에서 사용자를 구성하고 관리자가 App Studio에 추가한 그룹에 속해야 합니다. 관리자가 IAM Identity Center에 가입하도록 초대하면 초대를 수락하고 사용자 자격 증명을 활성화

하라는 이메일을 받게 됩니다. 활성화되면 해당 자격 증명을 사용하여 App Studio에 로그인할 수 있습니다.

App Studio에 액세스하기 위한 IAM Identity Center 초대를 수락하려면

1. 초대 이메일을 받으면 단계에 따라 IAM Identity Center에서 암호를 제공하고 사용자 자격 증명을 활성화합니다. 자세한 내용은 IAM Identity Center 가입 초대 수락을 참조하세요.
2. 사용자 자격 증명을 활성화한 후 이를 사용하여 App Studio 인스턴스에 로그인합니다.

AWS App Studio 시작하기

다음 시작하기 자습서에서는 App Studio에서 첫 번째 애플리케이션을 빌드하는 방법을 안내합니다.

- 권장 사항: 생성형 AI를 사용하여 생성하려는 앱을 설명하고 자동으로 생성하려면 단원을 참조하십시오자습서: AI를 사용하여 앱 생성.
- 빈 앱에서 빌드를 시작하려면 섹션을 참조하세요자습서: 빈 앱에서 빌드 시작.

자습서: AI를 사용하여 앱 생성

AWS App Studio에는 애플리케이션 구축 속도를 높이는 데 도움이 되는 서비스 전반의 생성형 AI 기능이 포함되어 있습니다. 이 자습서에서는 자연어를 사용하여 앱을 설명하여 AI를 사용하여 앱을 생성하는 방법을 알아봅니다.

AI를 사용하여 앱을 생성하는 것은 앱의 많은 리소스가 자동으로 생성되므로 빌드를 시작하는 좋은 방법입니다. 일반적으로 빈 앱에서 시작하는 것보다 기존 리소스로 생성된 앱에서 빌드를 시작하는 것이 훨씬 쉽습니다.

> **Note**
>
> 블로그 게시물 AWS App Studio(미리 보기)를 사용하여 자연어로 엔터프라이즈급 애플리케이션 구축을 보고 이미지가 포함된 유사한 연습을 볼 수 있습니다. 블로그 게시물에는 관리자 관련 리소스 설정 및 구성에 대한 정보도 포함되어 있지만, 원하는 경우 애플리케이션 구축에 대한 부분으로 건너뛸 수 있습니다.

App Studio는 AI로 앱을 생성할 때 사용자가 설명한 앱에 맞게 조정된 다음 리소스로 앱을 생성합니다.

- 페이지 및 구성 요소: 구성 요소는 애플리케이션 사용자 인터페이스의 구성 요소입니다. 테이블, 양식 및 버튼과 같은 시각적 요소를 나타냅니다. 각 구성 요소에는 고유한 속성 세트가 있으며 특정 요구 사항에 맞게 구성 요소를 사용자 지정할 수 있습니다. 페이지는 구성 요소의 컨테이너입니다.
- 자동화: 자동화를 사용하여 애플리케이션의 작동 방식을 제어하는 로직과 워크플로를 정의합니다. 예를 들어 자동화를 사용하여 데이터 테이블의 행을 생성, 업데이트, 읽기 또는 삭제하거나 Amazon S3 버킷의 객체와 상호 작용할 수 있습니다. 또한 이를 사용하여 데이터 검증, 알림 또는 다른 시스템과의 통합과 같은 작업을 처리할 수 있습니다.
- 개체: 데이터는 애플리케이션을 구동하는 정보입니다. 생성된 앱은 테이블과 유사한 개체를 생성합니다. 개체는 고객, 제품 또는 주문과 같이 저장하고 작업해야 하는 다양한 유형의 데이터를 나타냅

니다. App Studio 커넥터를 사용하여 이러한 데이터 모델을 AWS 서비스 및 외부 APIs를 비롯한 다양한 데이터 소스에 연결할 수 있습니다.

목차

- 사전 조건
- 1단계: 애플리케이션 생성
- 2단계: 새 애플리케이션 탐색
  - 페이지 및 구성 요소 탐색
  - 자동화 및 작업 살펴보기
  - 엔터티를 사용하여 데이터 탐색
- 3단계: 애플리케이션 미리 보기
- 다음 단계

사전 조건

시작하기 전에 다음 사전 조건을 검토하고 완료하세요.

- AWS App Studio에 대한 액세스. 자세한 내용은 AWS App Studio 설정 및 로그인 단원을 참조하십시오.
- 선택 사항: AWS App Studio 개념를 검토하여 중요한 App Studio 개념을 숙지합니다.

1단계: 애플리케이션 생성

앱 생성의 첫 번째 단계는 App Studio의 AI 어시스턴트에 생성하려는 앱을 설명하는 것입니다. 생성할 애플리케이션을 검토하고 생성 전에 원하는 대로 반복할 수 있습니다.

AI를 사용하여 앱을 생성하려면

1. App Studio에 로그인합니다.
2. 왼쪽 탐색 창에서 Builder 허브를 선택하고 + 앱 생성을 선택합니다.
3. AI를 사용하여 앱 생성을 선택합니다.
4. 앱 이름 필드에 앱의 이름을 입력합니다.
5. 데이터 소스 선택 대화 상자에서 건너뛰기를 선택합니다.

6. 텍스트 상자에 설명하거나 샘플 프롬프트에서 사용자 지정을 선택하여 생성하려는 앱을 정의할 수 있습니다. 앱을 설명하면 App Studio는 검토할 앱 요구 사항과 세부 정보를 생성합니다. 여기에는 사용 사례, 사용자 흐름 및 데이터 모델이 포함됩니다.
7. 요구 사항 및 세부 정보가 충족될 때까지 텍스트 상자를 사용하여 필요에 따라 앱과 반복합니다.
8. 앱을 생성하고 빌드를 시작할 준비가 되면 앱 생성을 선택합니다.
9. 선택적으로 새 앱을 탐색하는 방법을 자세히 설명하는 짧은 비디오를 볼 수 있습니다.
10. 앱 편집을 선택하여 앱의 개발 환경에 들어갑니다.

2단계: 새 애플리케이션 탐색

개발 환경에서는 다음 리소스를 찾을 수 있습니다.

- 애플리케이션을 보거나 편집하는 데 사용하는 캔버스입니다. 캔버스는 선택한 리소스에 따라 변경됩니다.
- 캔버스 상단의 탐색 탭. 탭은 다음 목록에 설명되어 있습니다.
  - 페이지: 페이지와 구성 요소를 사용하여 앱의 UI를 설계하는 위치입니다.
  - 자동화: 자동화에서 작업을 사용하여 앱의 비즈니스 로직을 정의하는 위치입니다.
  - 데이터: 엔터티, 필드, 샘플 데이터 및 데이터 작업을 정의하여 앱의 데이터 모델을 정의하는 위치입니다.
  - 앱 설정: 최종 사용자의 페이지에 대한 역할 기반 가시성을 정의하는 데 사용하는 앱 역할을 포함하여 앱에 대한 설정을 정의하는 위치입니다.
- 보려는 탭에 따라 리소스가 포함된 왼쪽 탐색 메뉴입니다.
- 페이지 및 자동화 탭에서 선택한 리소스의 리소스와 속성을 나열하는 오른쪽 메뉴입니다.
- 빌더 하단에 경고 및 오류를 표시하는 디버그 콘솔입니다. 생성된 앱에 오류가 있을 수 있습니다. 이는 Amazon Simple Email Service로 이메일 전송과 같은 작업을 수행하기 위해 구성된 커넥터가 필요한 자동화 때문일 수 있습니다.
- AI 빌더 어시스턴트로부터 상황별 도움을 받을 수 있는 AI 채팅 요청 창입니다.

페이지, 자동화 및 데이터 탭을 자세히 살펴보겠습니다.

페이지 및 구성 요소 탐색

페이지 탭에는 사용자를 위해 생성된 페이지와 해당 구성 요소가 표시됩니다.

각 페이지는 사용자가 상호 작용할 애플리케이션의 사용자 인터페이스(UI) 화면을 나타냅니다. 이 페이지에서는 다양한 구성 요소(예: 테이블, 양식 및 버튼)를 찾아 원하는 레이아웃과 기능을 생성할 수 있습니다.

왼쪽 탐색 메뉴를 사용하여 페이지와 해당 구성 요소를 볼 수 있습니다. 페이지 또는 구성 요소를 선택할 때 오른쪽 메뉴에서 속성을 선택할 수 있습니다.

자동화 및 작업 살펴보기

자동화 탭에는 자동화와 자동으로 생성된 작업이 표시됩니다.

자동화는 데이터 항목 생성, 보기, 업데이트 또는 삭제, 이메일 전송, API 또는 Lambda 함수 호출APIs 정의합니다.

왼쪽 탐색 메뉴를 사용하여 잠시 시간을 내어 자동화를 확인하세요. 자동화를 선택하면 오른쪽 속성 메뉴에서 해당 속성을 볼 수 있습니다. 자동화에는 다음 리소스가 포함됩니다.

- 자동화는 앱 비즈니스 로직의 구성 요소인 개별 작업으로 구성됩니다. 왼쪽 탐색 메뉴 또는 선택한 자동화의 캔버스에서 자동화 작업을 볼 수 있습니다. 작업을 선택하면 오른쪽 속성 메뉴에서 해당 속성을 볼 수 있습니다.
- 자동화 파라미터는 데이터가 자동화로 전달되는 방법입니다. 파라미터는 자동화가 실행될 때 실제 값으로 대체되는 자리 표시자 역할을 합니다. 이렇게 하면 매번 다른 입력으로 동일한 자동화를 사용할 수 있습니다.
- 자동화 출력은 자동화 결과를 구성하는 곳입니다. 기본적으로 자동화에는 출력이 없으므로 자동화 결과를 구성 요소 또는 기타 자동화에 사용하려면 여기에서 정의해야 합니다.

자세한 내용은 자동화 개념 단원을 참조하십시오.

엔터티를 사용하여 데이터 탐색

데이터 탭에는 자동으로 생성된 엔터티가 표시됩니다.

개체는 데이터베이스의 테이블과 마찬가지로 애플리케이션 데이터를 포함하는 테이블을 나타냅니다. 애플리케이션의 사용자 인터페이스(UI)와 자동화를 데이터 소스에 직접 연결하는 대신 먼저 엔터티에 연결합니다. 개체는 실제 데이터 소스와 App Studio 앱 간의 중개자 역할을 합니다. 이를 통해 한 곳에서 데이터를 관리하고 액세스할 수 있습니다.

왼쪽 탐색 메뉴에서 엔터티를 선택하여 생성된 엔터티를 볼 수 있습니다. 다음 세부 정보를 검토할 수 있습니다.

- 구성 탭에는 개체 이름과 개체의 열을 나타내는 해당 필드가 표시됩니다.
- 데이터 작업 탭에는 엔터티로 생성된 데이터 작업이 표시됩니다. 구성 요소 및 자동화는 데이터 작업을 사용하여 개체에서 데이터를 가져올 수 있습니다.
- 샘플 데이터 탭에는 개발 환경(외부 서비스와 통신하지 않음)에서 앱을 테스트하는 데 사용할 수 있는 샘플 데이터가 표시됩니다. 환경에 대한 자세한 내용은 애플리케이션 환경 섹션을 참조하세요.
- 연결 탭에는 엔터티가 연결된 외부 데이터 소스에 대한 정보가 표시됩니다. App Studio는 DynamoDB 테이블을 사용하는 관리형 데이터 스토리지 솔루션을 제공합니다. 자세한 내용은 AWS App Studio의 관리형 데이터 엔터티 단원을 참조하십시오.

3단계: 애플리케이션 미리 보기

App Studio에서 애플리케이션을 미리 보고 사용자에게 어떻게 표시되는지 확인할 수 있습니다. 기능을 사용하고 디버그 패널에서 로그를 확인하여 기능을 테스트할 수도 있습니다.

애플리케이션 미리 보기 환경은 라이브 데이터 표시 또는 데이터 소스와 같은 커넥터를 사용한 외부 리소스와의 연결을 지원하지 않습니다. 대신 샘플 데이터와 모의 출력을 사용하여 기능을 테스트할 수 있습니다.

테스트를 위해 앱을 미리 보려면

1. 앱 빌더의 오른쪽 상단 모서리에서 미리 보기를 선택합니다.
2. 앱의 페이지와 상호 작용합니다.

다음 단계

이제 첫 번째 앱을 생성했으므로 몇 가지 다음 단계는 다음과 같습니다.

- 이미지가 포함된 또 다른 시작 안내서는 블로그 게시물 Build enterprise-grade applications with natural language using AWS App Studio (preview)를 참조하세요.
- 앱은 커넥터를 사용하여 데이터를 전송 및 수신하거나 외부 서비스(AWS 서비스 및 타사 서비스 모두)와 통신합니다. 커넥터에 대해 자세히 알아보고 앱을 빌드하도록 구성하는 방법을 알아야 합니다. 커넥터를 관리하려면 관리자 역할이 있어야 합니다. 자세한 내용은 커넥터를 사용하여 App Studio를 다른 서비스에 연결를 참조하세요.
- 앱을 미리 보고, 게시하고, 최종 사용자에게 공유하는 방법에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 미리 보기, 게시 및 공유.
- 몇 가지 실습 경험을 위해 생성한 앱을 계속 탐색하고 업데이트합니다.

- 앱 빌드에 대한 자세한 내용은 단원을 참조하십시오Builder 설명서. 특히 다음 주제는 탐색하는 데 유용할 수 있습니다.
  - Automation 작업 참조
  - 구성 요소 참조
  - Amazon Simple Storage Service와 구성 요소 및 자동화의 상호 작용
  - 보안 고려 사항 및 완화 조치

자습서: 빈 앱에서 빌드 시작

이 자습서에서는 AWS App Studio를 사용하여 내부 고객 회의 요청 애플리케이션을 빌드합니다. App Studio에서 앱을 빌드하는 방법과 실제 사용 사례 및 실습 예제에 대해 알아봅니다. 또한 데이터 구조, UI 설계 및 앱 배포 정의에 대해 알아봅니다.

> **Note**
>
> 이 자습서에서는 빈 앱부터 시작하여 처음부터 앱을 빌드하는 방법을 자세히 설명합니다. 일반적으로 AI를 사용하여 생성하려는 앱에 대한 설명을 제공하여 앱과 해당 리소스를 생성하는 것이 훨씬 빠르고 쉽습니다. 자세한 내용은 자습서: AI를 사용하여 앱 생성 단원을 참조하십시오.

App Studio를 사용하여 애플리케이션을 구축하는 방법을 이해하는 핵심은 구성 요소, 자동화, 데이터 및 커넥터라는 네 가지 핵심 개념과 이러한 개념이 함께 작동하는 방식을 이해하는 것입니다.

- 구성 요소: 구성 요소는 애플리케이션 사용자 인터페이스의 구성 요소입니다. 테이블, 양식 및 버튼과 같은 시각적 요소를 나타냅니다. 각 구성 요소에는 특정 요구 사항에 맞게 사용자 지정할 수 있는 고유한 속성 세트가 있습니다.
- 자동화: 자동화를 사용하면 애플리케이션의 작동 방식을 제어하는 로직과 워크플로를 정의할 수 있습니다. 자동화를 사용하여 데이터 테이블의 행을 생성, 업데이트, 읽기 또는 삭제하거나 Amazon S3 버킷의 객체와 상호 작용할 수 있습니다. 또한 이를 사용하여 데이터 검증, 알림 또는 다른 시스템과의 통합과 같은 작업을 처리할 수 있습니다.
- 데이터: 데이터는 애플리케이션을 구동하는 정보입니다. App Studio에서 엔터티라고 하는 데이터 모델을 정의할 수 있습니다. 개체는 고객 회의 요청, 의제 또는 참석자와 같이 저장하고 작업해야 하는 다양한 유형의 데이터를 나타냅니다. App Studio 커넥터를 사용하여 이러한 데이터 모델을 AWS 서비스 및 외부 APIs를 비롯한 다양한 데이터 소스에 연결할 수 있습니다.
- 커넥터: App Studio는 Aurora, DynamoDB 및 Amazon Redshift와 같은 AWS 서비스를 포함하는 다양한 데이터 소스와의 연결을 제공합니다. 데이터 소스에는 Salesforce와 같은 타사 서비스 또는

OpenAPI 또는 일반 API 커넥터를 사용하는 다른 많은 서비스도 포함됩니다. App Studio 커넥터를 사용하여 이러한 엔터프라이즈급 서비스 및 외부 애플리케이션의 데이터와 기능을 애플리케이션에 쉽게 통합할 수 있습니다.

자습서를 진행하면서 구성 요소, 데이터 및 자동화의 주요 개념을 결합하여 내부 고객 회의 요청 애플리케이션을 구축하는 방법을 살펴보겠습니다.

다음은이 자습서에서 수행할 작업을 설명하는 상위 수준 단계입니다.

1. 데이터로 시작: 많은 애플리케이션이 데이터 모델로 시작하므로이 자습서는 데이터로도 시작합니다. 고객 회의 요청 앱을 빌드하려면 먼저 `MeetingRequests`엔터티를 생성해야 합니다. 이 엔터티는 고객 이름, 회의 날짜, 의제 및 참석자와 같은 모든 관련 회의 요청 정보를 저장하기 위한 데이터 구조를 나타냅니다. 이 데이터 모델은 애플리케이션의 기반 역할을 하며 빌드할 다양한 구성 요소와 자동화를 지원합니다.
2. 사용자 인터페이스(UI) 생성: 데이터 모델을 마련하면 자습서에서 사용자 인터페이스(UI)를 구축하는 방법을 안내합니다. App Studio에서는 페이지를 추가하고 여기에 구성 요소를 추가하여 UI를 빌드합니다. 회의 요청 대시보드 페이지에 테이블, 세부 정보 보기 및 일정과 같은 구성 요소를 추가합니다. 이러한 구성 요소는 `MeetingRequests` 개체에 저장된 데이터를 표시하고 상호 작용하도록 설계되었습니다. 이를 통해 사용자는 고객 회의를 보고, 관리하고, 예약할 수 있습니다. 또한 회의 요청 생성 페이지를 생성합니다. 이 페이지에는 데이터를 수집하는 양식 구성 요소와 데이터를 제출하는 버튼 구성 요소가 포함되어 있습니다.
3. 자동화를 사용하여 비즈니스 로직 추가: 애플리케이션의 기능을 개선하기 위해 사용자 상호 작용을 활성화하도록 일부 구성 요소를 구성합니다. 일부 예는 페이지로 이동하거나 `MeetingRequests`엔터티에서 새 회의 요청 레코드를 생성하는 것입니다.
4. 검증 및 표현식으로 개선: 데이터의 무결성과 정확성을 보장하기 위해 양식 구성 요소에 검증 규칙을 추가합니다. 이렇게 하면 사용자가 새 회의 요청 레코드를 생성할 때 완전하고 유효한 정보를 제공할 수 있습니다. 또한 표현식을 사용하여 애플리케이션 내의 데이터를 참조하고 조작하므로 사용자 인터페이스 전체에 동적 및 컨텍스트 정보를 표시할 수 있습니다.
5. 미리 보기 및 테스트: 애플리케이션을 배포하기 전에 애플리케이션을 미리 보고 철저히 테스트할 수 있습니다. 이렇게 하면 구성 요소, 데이터 및 자동화가 모두 원활하게 작동하는지 확인할 수 있습니다. 이를 통해 사용자는 원활하고 직관적인 경험을 할 수 있습니다.
6. 애플리케이션 게시: 마지막으로 완료된 내부 고객 회의 요청 애플리케이션을 배포하고 사용자가 액세스할 수 있도록 합니다. App Studio의 로우 코드 접근 방식의 기능을 사용하면 광범위한 프로그래밍 전문 지식 없이도 조직의 특정 요구 사항을 충족하는 사용자 지정 애플리케이션을 구축할 수 있습니다.

목차

- 사전 조건
- 1단계: 애플리케이션 생성
- 2단계: 앱의 데이터를 정의하는 엔터티 생성
  - 관리형 엔터티 생성
  - 엔터티에 필드 추가
- 3단계: 사용자 인터페이스(UI) 및 로직 설계
  - 회의 요청 대시보드 페이지 추가
  - 회의 요청 생성 페이지 추가
- 4단계: 애플리케이션 미리 보기
- 5단계: 애플리케이션을 테스트 환경에 게시
- 다음 단계

사전 조건

시작하기 전에 다음 사전 조건을 검토하고 완료하세요.

- AWS App Studio에 대한 액세스. 자세한 내용은 AWS App Studio 설정 및 로그인 단원을 참조하십시오.
- 선택 사항: AWS App Studio 개념를 검토하여 중요한 App Studio 개념을 숙지합니다.
- 선택 사항: JavaScript 구문과 같은 기본 웹 개발 개념에 대한 이해.
- 선택 사항: AWS 서비스에 대한 지식.

1단계: 애플리케이션 생성

1. App Studio에 로그인합니다.
2. 왼쪽 탐색에서 Builder 허브를 선택하고 + 앱 생성을 선택합니다.
3. 처음부터 시작을 선택합니다.
4. 앱 이름 필드에와 같은 앱 이름을 입력합니다**`Customer Meeting Requests`**.
5. 데이터 소스 또는 커넥터를 선택하라는 메시지가 표시되면이 자습서에서 건너뛰기를 선택합니다.
6. 다음을 선택하여 계속 진행합니다.

7. (선택 사항): App Studio에서 앱을 빌드하는 방법에 대한 간략한 개요를 보려면 비디오 자습서를 시청하세요.
8. 앱 편집을 선택하면 App Studio 앱 빌더로 이동합니다.

2단계: 앱의 데이터를 정의하는 엔터티 생성

개체는 데이터베이스의 테이블과 마찬가지로 애플리케이션 데이터를 포함하는 테이블을 나타냅니다. 애플리케이션의 사용자 인터페이스(UI)와 데이터 소스에 직접 연결하는 자동화 대신 먼저 엔터티에 연결합니다. 개체는 실제 데이터 소스와 App Studio 앱 간의 중개자 역할을 하며 데이터를 관리하고 액세스할 수 있는 단일 위치를 제공합니다.

개체를 생성하는 방법에는 네 가지가 있습니다. 이 자습서에서는 App Studio 관리형 엔터티를 사용합니다.

관리형 엔터티 생성

관리형 엔터티를 생성하면 App Studio가 관리하는 해당 DynamoDB 테이블도 생성됩니다. App Studio 앱의 개체가 변경되면 DynamoDB 테이블이 자동으로 업데이트됩니다. 이 옵션을 사용하면 타사 데이터 소스를 수동으로 생성, 관리 또는 연결하거나 엔터티 필드에서 테이블 열로의 매핑을 지정할 필요가 없습니다.

개체를 생성할 때 기본 키 필드를 정의해야 합니다. 기본 키는 개체의 각 레코드 또는 행에 대한 고유 식별자 역할을 합니다. 이렇게 하면 각 레코드를 모호하지 않게 쉽게 식별하고 검색할 수 있습니다. 기본 키는 다음 속성으로 구성됩니다.

- 기본 키 이름: 개체의 기본 키 필드 이름입니다.
- 기본 키 데이터 유형: 기본 키 필드의 유형입니다. App Studio에서 지원되는 기본 키 유형은 텍스트의 경우 문자열이고 숫자의 경우 부동 소수점입니다. 텍스트 기본 키(예: *meetingName*)에는 문자열 유형이 있고 숫자 기본 키(예: *meetingId*)에는 부동 소수점 유형이 있습니다.

기본 키는 데이터 무결성을 적용하고, 데이터 중복을 방지하고, 효율적인 데이터 검색 및 쿼리를 가능하게 하므로 개체의 중요한 구성 요소입니다.

관리형 엔터티를 생성하려면

1. 상단 표시줄 메뉴에서 데이터를 선택합니다.
2. + 개체 생성을 선택합니다.

3. App Studio 관리형 엔터티 생성을 선택합니다.
4. 엔터티 이름 필드에 엔터티의 이름을 입력합니다. 이 자습서에서는 **MeetingRequests**을 입력합니다.
5. 기본 키 필드에 개체의 기본 키 열에 부여할 기본 키 이름 레이블을 입력합니다. 이 자습서에서는 **requestID**을 입력합니다.
6. 기본 키 데이터 유형에서 부동 소수점을 선택합니다.
7. 개체 생성을 선택합니다.

엔터티에 필드 추가

각 필드에 앱 사용자에게 표시되는 레이블인 표시 이름을 지정합니다. 표시 이름에는 공백과 특수 문자가 포함될 수 있지만 개체 내에서 고유해야 합니다. 표시 이름은 필드에 대한 사용자 친화적 레이블 역할을 하며 사용자가 용도를 쉽게 식별하고 이해하는 데 도움이 됩니다.

다음으로 애플리케이션에서 필드를 참조하기 위해 내부적으로 사용하는 고유 식별자인 시스템 이름을 제공합니다. 시스템 이름은 공백이나 특수 문자 없이 간결해야 합니다. 시스템 이름을 사용하면 애플리케이션이 필드 데이터를 변경할 수 있습니다. 애플리케이션 내의 필드에 대한 고유한 참조 지점 역할을 합니다.

마지막으로 문자열(텍스트), 부울(true/false), 날짜, 소수점, 부동 소수점, 정수 또는 DateTime과 같이 필드에 저장하려는 데이터의 종류를 가장 잘 나타내는 데이터 유형을 선택합니다. 적절한 데이터 유형을 정의하면 데이터 무결성이 보장되고 필드 값을 적절하게 처리하고 처리할 수 있습니다. 예를 들어 회의 요청에 고객 이름을 저장하는 경우 텍스트 값을 수용할 `String` 데이터 유형을 선택합니다.

`MeetingRequests` 엔터티에 필드를 추가하려면

- + 필드 추가를 선택하여 다음 네 필드를 추가합니다.

  a. 다음 정보와 함께 고객의 이름을 나타내는 필드를 추가합니다.

     - 표시 이름: **`Customer name`**
     - 시스템 이름: **`customerName`**
     - 데이터 유형: **`String`**

  b. 다음 정보와 함께 회의 날짜를 나타내는 필드를 추가합니다.

     - 표시 이름: **`Meeting date`**
     - 시스템 이름: **`meetingDate`**

- 데이터 유형: **DateTime**

c. 다음 정보와 함께 회의 의제를 나타내는 필드를 추가합니다.

- 표시 이름: **Agenda**
- 시스템 이름: **agenda**
- 데이터 유형: **String**

d. 다음 정보와 함께 회의 참석자를 나타내는 필드를 추가합니다.

- 표시 이름: **Attendees**
- 시스템 이름: **attendees**
- 데이터 유형: **String**

게시하기 전에 애플리케이션을 테스트하고 미리 보는 데 사용할 수 있는 샘플 데이터를 엔터티에 추가할 수 있습니다. 최대 500행의 모의 데이터를 추가하여 실제 시나리오를 시뮬레이션하고 실제 데이터에 의존하거나 외부 서비스에 연결하지 않고도 애플리케이션이 다양한 유형의 데이터를 처리하고 표시하는 방법을 검사할 수 있습니다. 이를 통해 개발 프로세스 초기에 문제나 불일치를 식별하고 해결할 수 있습니다. 이렇게 하면 실제 데이터를 처리할 때 애플리케이션이 의도한 대로 작동합니다.

개체에 샘플 데이터를 추가하려면

1. 배너에서 샘플 데이터 탭을 선택합니다.
2. 더 많은 샘플 데이터 생성을 선택합니다.
3. 저장을 선택합니다.

필요에 따라 배너에서 연결을 선택하여 커넥터 및 사용자를 위해 생성된 DynamoDB 테이블에 대한 세부 정보를 검토합니다.

3단계: 사용자 인터페이스(UI) 및 로직 설계

회의 요청 대시보드 페이지 추가

App Studio에서 각 페이지는 사용자가 상호 작용할 애플리케이션의 사용자 인터페이스(UI) 화면을 나타냅니다. 이러한 페이지 내에서 테이블, 양식 및 버튼과 같은 다양한 구성 요소를 추가하여 원하는 레이아웃과 기능을 생성할 수 있습니다.

새로 생성된 애플리케이션은 기본 페이지와 함께 제공되므로 간단한 회의 요청 대시보드 페이지로 사용할 새 애플리케이션을 추가하는 대신 이름을 바꿉니다.

기본 페이지의 이름을 바꾸려면

1. 상단 표시줄 탐색 메뉴에서 페이지를 선택합니다.
2. 왼쪽 패널에서 Page1을 두 번 클릭하고 이름을 로 변경**MeetingRequestsDashboard**한 다음 Enter 키를 누릅니다.

이제 회의 요청을 표시하는 데 사용할 테이블에 테이블 구성 요소를 추가합니다.

회의 요청 대시보드 페이지에 테이블 구성 요소를 추가하려면

1. 오른쪽 구성 요소 패널에서 테이블 구성 요소를 찾아 캔버스로 끕니다.
2. 캔버스에서 테이블을 선택하여 선택합니다.
3. 오른쪽 속성 패널에서 다음 설정을 업데이트합니다.
   a. 연필 아이콘을 선택하여 테이블의 이름을 로 바꿉니다**meetingRequestsTable**.
   b. 소스 드롭다운에서 엔터티를 선택합니다.
   c. 데이터 작업 드롭다운에서 생성한 엔터티(**MeetingRequests**)를 선택하고 + 데이터 작업 추가를 선택합니다.
4. 메시지가 표시되면 getAll을 선택합니다.

> **Note**
>
> getAll 데이터 작업은 지정된 개체에서 모든 레코드(행)를 검색하는 특정 유형의 데이터 작업입니다. 예를 들어 getAll 데이터 작업을 테이블 구성 요소와 연결하면 테이블이 연결된 개체의 모든 데이터로 자동으로 채워지고 각 레코드가 테이블에 행으로 표시됩니다.

회의 요청 생성 페이지 추가

그런 다음 최종 사용자가 회의 요청을 생성하는 데 사용할 양식이 포함된 페이지를 생성합니다. 또한 `MeetingRequests` 개체에 레코드를 생성한 다음 최종 사용자를 `MeetingRequestsDashboard` 페이지로 다시 탐색하는 제출 버튼을 추가합니다.

회의 요청 생성 페이지를 추가하려면

1. 상단 배너에서 페이지를 선택합니다.
2. 왼쪽 패널에서 + 추가를 선택합니다.

3. 오른쪽 속성 패널에서 연필 아이콘을 선택하고 페이지 이름을 로 변경합니다**CreateMeetingRequest**.

이제 페이지가 추가되었으므로 최종 사용자가 MeetingRequests엔터티에서 회의 요청을 생성하는 데 정보를 입력하는 데 사용할 페이지에 양식을 추가합니다. App Studio는 기존 엔터티에서 양식을 생성하는 방법을 제공합니다.이 방법은 엔터티의 필드를 기반으로 양식 필드를 자동으로 채우고 양식 입력으로 엔터티에 레코드를 생성하기 위한 제출 버튼을 생성합니다.

회의 요청 생성 페이지에서 엔터티로부터 양식을 자동으로 생성하려면

1. 오른쪽 구성 요소 메뉴에서 양식 구성 요소를 찾아 캔버스로 끕니다.
2. 양식 생성을 선택합니다.
3. 드롭다운에서 MeetingRequests개체를 선택합니다.
4. 생성을 선택합니다.
5. 캔버스에서 제출 버튼을 선택하여 선택합니다.
6. 오른쪽 속성 패널의 트리거 섹션에서 + 추가를 선택합니다.
7. 탐색을 선택합니다.
8. 오른쪽 속성 패널에서 작업 이름을와 같이 설명이 포함된 이름으로 변경합니다**Navigate to MeetingRequestsDashboard**.
9. 탐색 유형을 페이지로 변경합니다. 탐색 드롭다운에서를 선택합니다**MeetingRequestsDashboard**.

이제 회의 요청 생성 페이지와 양식이 있으므로 대시보드를 검토하는 최종 사용자가 회의 요청을 쉽게 생성할 수 있도록 페이지에서이 MeetingRequestsDashboard 페이지로 쉽게 이동할 수 있도록 하려고 합니다. 다음 절차에 따라 MeetingRequestsDashboard 페이지로 이동하는 CreateMeetingRequest 페이지에 버튼을 생성합니다.

에서 로 이동하는 버튼을 추가하려면 **MeetingRequestsDashboard** **CreateMeetingRequest**

1. 상단 배너에서 페이지를 선택합니다.
2. MeetingRequestsDashboard 페이지를 선택합니다.
3. 오른쪽 구성 요소 패널에서 버튼 구성 요소를 찾아 캔버스로 끌어 테이블 위에 놓습니다.
4. 새로 추가된 버튼을 선택하여 선택합니다.
5. 오른쪽 속성 패널에서 다음 설정을 업데이트합니다.

a. 연필 아이콘을 선택하여 버튼의 이름을 로 변경합니다**createMeetingRequestButton**.

b. 버튼 레이블: **Create Meeting Request**. 최종 사용자에게 표시되는 이름입니다.

c. 아이콘 드롭다운에서 + Plus를 선택합니다.

d. 최종 사용자를 MeetingRequestsDashboard 페이지로 이동하는 트리거를 생성합니다.

1. 트리거 섹션에서 + 추가를 선택합니다.
2. 작업 유형에서 탐색을 선택합니다.
3. 방금 생성한 트리거를 선택하여 구성합니다.
4. 작업 이름에와 같은 설명이 포함된 이름을 입력합니다**NavigateToCreateMeetingRequest**.
5. 탐색 유형 드롭다운에서 페이지를 선택합니다.
6. 탐색 드롭다운에서 CreateMeetingRequest 페이지를 선택합니다.

4단계: 애플리케이션 미리 보기

App Studio에서 애플리케이션을 미리 보고 사용자에게 어떻게 표시되는지 확인할 수 있습니다. 또한 기능을 사용하고 디버그 패널에서 로그를 확인하여 기능을 테스트할 수 있습니다.

애플리케이션 미리 보기 환경은 라이브 데이터 표시를 지원하지 않습니다. 또한 데이터 소스와 같은 커넥터를 사용한 외부 리소스와의 연결도 지원하지 않습니다. 대신 샘플 데이터와 모의 출력을 사용하여 기능을 테스트할 수 있습니다.

테스트를 위해 앱을 미리 보려면

1. 앱 빌더의 오른쪽 상단 모서리에서 미리 보기를 선택합니다.
2. MeetingRequestsDashboard 페이지와 상호 작용하고 테이블, 양식 및 버튼을 테스트합니다.

5단계: 애플리케이션을 테스트 환경에 게시

이제 애플리케이션 생성, 구성 및 테스트를 마쳤으므로 테스트 환경에 게시하여 최종 테스트를 수행한 다음 사용자와 공유할 차례입니다.

앱을 테스트 환경에 게시하려면

1. 앱 빌더의 오른쪽 상단 모서리에서 게시를 선택합니다.

2. 테스트 환경에 대한 버전 설명을 추가합니다.
3. SLA와 관련된 확인란을 검토하고 선택합니다.
4. 시작을 선택합니다. 게시에는 최대 15분이 걸릴 수 있습니다.
5. (선택 사항) 준비가 되면 공유를 선택하고 프롬프트에 따라 다른 사용자에게 액세스 권한을 부여할 수 있습니다.

> **Note**
>
> 앱을 공유하려면 관리자가 최종 사용자 그룹을 생성해야 합니다.

테스트 후 게시를 다시 선택하여 애플리케이션을 프로덕션 환경으로 승격합니다. 다양한 애플리케이션 환경에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 환경.

다음 단계

이제 첫 번째 앱을 생성했으므로 몇 가지 다음 단계는 다음과 같습니다.

1. 자습서 앱을 계속 빌드합니다. 이제 데이터, 일부 페이지 및 자동화가 구성되었으므로 추가 페이지를 추가하고 구성 요소를 추가하여 앱 빌드에 대해 자세히 알아볼 수 있습니다.
2. 앱 빌드에 대한 자세한 내용은 섹션을 참조하세요Builder 설명서. 특히 다음 주제는 탐색하는 데 유용할 수 있습니다.
   - Automation 작업 참조
   - 구성 요소 참조
   - Amazon Simple Storage Service와 구성 요소 및 자동화의 상호 작용
   - 보안 고려 사항 및 완화 조치

   또한 다음 주제에는 자습서에서 설명하는 개념에 대한 자세한 정보가 포함되어 있습니다.
   - 애플리케이션 미리 보기, 게시 및 공유
   - App Studio 앱에서 개체 생성

관리자 설명서

다음 주제에는 App Studio에서 타사 서비스 연결 및 액세스, 사용자 및 역할을 관리하는 사용자에게 도움이 되는 정보가 포함되어 있습니다.

주제

- App Studio에서 액세스 및 역할 관리
- 커넥터를 사용하여 App Studio를 다른 서비스에 연결
- App Studio 인스턴스 삭제

App Studio에서 액세스 및 역할 관리

App Studio에서 관리자의 책임 중 하나는 액세스, 역할 및 권한을 관리하는 것입니다. 다음 주제에는 App Studio의 역할에 대한 정보와 사용자를 추가하거나, 사용자를 제거하거나, 역할을 변경하는 방법이 포함되어 있습니다.

AWS App Studio에 대한 액세스는 IAM Identity Center 그룹을 사용하여 관리됩니다. App Studio 인스턴스에 사용자를 추가하려면 다음 중 하나를 수행해야 합니다.

- App Studio에 추가된 기존 IAM Identity Center 그룹에 추가합니다.
- App Studio에 추가되지 않은 신규 또는 기존 IAM Identity Center 그룹에 추가한 다음 App Studio에 추가합니다.

역할은 그룹에 적용되므로 IAM Identity Center 그룹은 그룹 구성원에게 할당하려는 액세스 권한(또는 역할)을 나타내야 합니다. 사용자 및 그룹 관리에 대한 정보를 포함하여 IAM Identity Center에 대한 자세한 내용은 IAM Identity Center 사용 설명서를 참조하세요.

역할 및 권한

App Studio에는 세 가지 역할이 있습니다. 다음 목록에는 각 역할과 해당 설명이 포함되어 있습니다.

- 관리자: 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
- Builder: Builder는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.

- 앱 사용자: 앱 사용자는 게시된 앱에 액세스하고 사용할 수 있지만 App Studio 인스턴스에 액세스하여 앱을 빌드하거나 리소스를 관리할 수는 없습니다.

App Studio에서 역할은 그룹에 할당되므로 추가된 IAM Identity Center 그룹의 각 구성원에게는 그룹에 할당된 역할이 할당됩니다.

그룹 보기

App Studio 인스턴스에 추가된 그룹을 보려면 다음 단계를 수행합니다.

> **Note**
>
> App Studio 인스턴스에서 그룹을 보려면 관리자여야 합니다.

App Studio 인스턴스에 추가된 그룹을 보려면

- 탐색 창의 관리 섹션에서 역할을 선택합니다. 기존 그룹 목록과 각 그룹의 할당된 역할을 표시하는 페이지로 이동합니다.

  그룹 관리에 대한 자세한 내용은 그룹의 역할 변경, 또는 사용자 또는 그룹 추가단원을 참조하십시오App Studio에서 사용자 또는 그룹 제거.

사용자 또는 그룹 추가

App Studio에 사용자를 추가하려면 사용자를 IAM Identity Center 그룹에 추가하고 해당 그룹을 App Studio에 추가해야 합니다. 다음 단계를 수행하여 IAM Identity Center 그룹을 추가하고 역할을 할당하여 App Studio에 사용자를 추가합니다.

> **Note**
>
> App Studio 인스턴스에 사용자를 추가하려면 관리자여야 합니다.

App Studio 인스턴스에 사용자 또는 그룹을 추가하려면

1. App Studio 인스턴스에 사용자를 추가하려면 App Studio에 추가된 기존 IAM Identity Center 그룹에 사용자를 추가하거나 새 IAM Identity Center 그룹을 생성하고 새 사용자를 추가한 다음 새 그룹을 App Studio에 추가해야 합니다.

IAM Identity Center 사용자 및 그룹 관리에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 IAM Identity Center에서 ID 관리를 참조하세요.

2. App Studio에 이미 추가된 기존 IAM Identity Center 그룹에 사용자를 추가한 경우 새 사용자는 IAM Identity Center 권한 설정을 완료한 후 지정된 권한으로 App Studio에 액세스할 수 있습니다. 새 IAM Identity Center 그룹을 생성한 경우 다음 단계를 수행하여 그룹을 App Studio에 추가하고 그룹 구성원의 역할을 지정합니다.
3. 탐색 창의 관리 섹션에서 역할을 선택합니다.
4. 역할 페이지에서 + 그룹 추가를 선택합니다. 그러면 그룹에 대한 정보를 입력할 수 있는 그룹 추가 대화 상자가 열립니다.
5. 그룹 추가 대화 상자에 다음 정보를 입력합니다.
   - 드롭다운에서 기존 IAM Identity Center 그룹을 선택합니다.
   - 그룹의 역할을 선택합니다.
     - 관리자: 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
     - Builder: Builder는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.
     - 앱 사용자: 앱 사용자는 게시된 앱에 액세스하고 사용할 수 있지만 App Studio 인스턴스에 액세스하여 앱을 빌드하거나 리소스를 관리할 수는 없습니다.
6. 할당을 선택하여 App Studio에 그룹을 추가하고 멤버에게 구성된 역할을 제공합니다.

그룹의 역할 변경

다음 단계에 따라 App Studio에서 그룹에 할당된 역할을 변경합니다. 그룹의 역할을 변경하면 해당 그룹의 모든 구성원의 역할이 변경됩니다.

> **Note**
>
> App Studio에서 그룹의 역할을 변경하려면 관리자여야 합니다.

그룹의 역할을 변경하려면

1. 탐색 창의 관리 섹션에서 역할을 선택합니다. 기존 그룹 목록과 각 그룹의 할당된 역할을 표시하는 페이지로 이동합니다.
2. 줄임표 아이콘(...)을 선택하고 역할 변경을 선택합니다.
3. 역할 변경 대화 상자에서 그룹의 새 역할을 선택합니다.

   - 관리자: 관리자는 App Studio 내에서 사용자 및 그룹을 관리하고, 커넥터를 추가 및 관리하고, 빌더가 생성한 애플리케이션을 관리할 수 있습니다. 또한 관리자 역할이 있는 사용자는 Builder 역할에 포함된 모든 권한을 가집니다.
   - Builder: Builder는 애플리케이션을 생성하고 빌드할 수 있습니다. 빌더는 사용자 또는 그룹을 관리하거나, 커넥터 인스턴스를 추가 또는 편집하거나, 다른 빌더의 애플리케이션을 관리할 수 없습니다.
   - 앱 사용자: 앱 사용자는 게시된 앱에 액세스하고 사용할 수 있지만 App Studio 인스턴스에 액세스하여 앱을 빌드하거나 리소스를 관리할 수는 없습니다.
4. 그룹 역할 변경을 선택합니다.

App Studio에서 사용자 또는 그룹 제거

App Studio에서 IAM Identity Center 그룹을 제거할 수 없습니다. 대신 다음 지침을 수행하면 그룹의 역할이 앱 사용자로 다운그레이드됩니다. 그룹의 멤버는 여전히 게시된 App Studio 앱에 액세스할 수 있습니다.

App Studio 및 해당 앱에 대한 모든 액세스를 제거하려면 AWS IAM Identity Center 콘솔에서 IAM Identity Center 그룹 또는 사용자를 삭제해야 합니다. IAM Identity Center 사용자 및 그룹 관리에 대한 자세한 내용은 AWS IAM Identity Center 사용 설명서의 IAM Identity Center에서 ID 관리를 참조하세요.

> **Note**
>
> App Studio에서 그룹의 액세스를 다운그레이드하려면 관리자여야 합니다.

그룹을 제거하려면

1. 탐색 창의 관리 섹션에서 역할을 선택합니다. 기존 그룹 목록과 각 그룹의 할당된 역할을 표시하는 페이지로 이동합니다.

2. 줄임표 아이콘(...)을 선택하고 역할 취소를 선택합니다.
3. 역할 취소 대화 상자에서 취소를 선택하여 그룹의 역할을 앱 사용자로 다운그레이드합니다.

커넥터를 사용하여 App Studio를 다른 서비스에 연결

커넥터는 App Studio와 및 AWS Lambda Amazon Redshift 또는 타사 서비스와 같은 다른 AWS 서비스 간의 연결입니다. 커넥터가 생성되고 구성되면 빌더는 커넥터와 해당 애플리케이션에서 App Studio에 연결되는 리소스를 사용할 수 있습니다.

관리자 역할을 가진 사용자만 커넥터를 생성, 관리 또는 삭제할 수 있습니다.

주제

- AWS 서비스에 연결
- 타사 서비스에 연결
- 커넥터 보기, 편집 및 삭제

AWS 서비스에 연결

주제

- Amazon Redshift에 연결
- Amazon DynamoDB에 연결
- 에 연결 AWS Lambda
- Amazon Simple Storage Service(Amazon S3)에 연결
- Amazon Aurora에 연결
- Amazon Bedrock에 연결
- Amazon Simple Email Service에 연결
- 기타 AWS 서비스 커넥터를 사용하여 AWS 서비스에 연결
- CMKs에서 암호화된 데이터 소스 사용

Amazon Redshift에 연결

App Studio를 Amazon Redshift와 연결하여 빌더가 애플리케이션에서 Amazon Redshift 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. 1단계: Amazon Redshift 리소스 생성 및 구성
2. 2단계: 적절한 Amazon Redshift 권한을 사용하여 IAM 정책 및 역할 생성
3. 3단계: Amazon Redshift 커넥터 생성

1단계: Amazon Redshift 리소스 생성 및 구성

다음 절차에 따라 App Studio와 함께 사용할 Amazon Redshift 리소스를 생성하고 구성합니다.

App Studio와 함께 사용할 Amazon Redshift를 설정하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/redshiftv2/ Amazon Redshift 콘솔을 엽니다.

   에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. Redshift Serverless 데이터 웨어하우스 또는 프로비저닝된 클러스터를 생성합니다. 자세한 내용은 Amazon Redshift 사용 설명서의 Redshift Serverless를 사용하여 데이터 웨어하우스 생성 또는 클러스터 생성을 참조하세요.
3. 프로비저닝이 완료되면 쿼리 데이터를 선택하여 쿼리 편집기를 엽니다. 데이터베이스에 연결합니다.
4. 다음 설정을 변경합니다.

   1. 격리된 세션 토글을 로 설정합니다OFF. 이는 실행 중인 App Studio 애플리케이션과 같은 다른 사용자의 데이터 변경 사항을 볼 수 있도록 하기 위해 필요합니다.
   2. “톱니” 아이콘을 선택합니다. 계정 설정(Account settings)을 선택합니다. 에 대한 최대 동시 연결을 늘립니다10. 이는 Amazon Redshift 데이터베이스에 연결할 수 있는 쿼리 편집기 세션 수에 대한 제한입니다. App Studio 애플리케이션과 같은 다른 클라이언트에는 적용되지 않습니다.
5. `public` 스키마로 데이터 테이블을 생성합니다. `INSERT` 초기 데이터는 이러한 테이블로 들어갑니다.
6. 쿼리 편집기에서 다음 명령을 실행합니다.

   다음 명령은 데이터베이스 사용자를 생성하고 App Studio에서 사용하는 *AppBuilderDataAccessRole*이라는 IAM 역할에 연결합니다. 이후 단계에서 IAM 역할을 생성하며, 여기에 있는 이름은 해당 역할에 지정된 이름과 일치해야 합니다.

   ```
   CREATE USER "IAMR:AppBuilderDataAccessRole" WITH PASSWORD DISABLE;
   ```

다음 명령은 모든 테이블에 대한 모든 권한을 App Studio에 부여합니다.

> **Note**
>
> 보안 모범 사례를 위해 여기에 있는 권한의 범위를 적절한 테이블에 필요한 최소 권한으로 좁혀야 합니다. GRANT 명령에 대한 자세한 내용은 Amazon Redshift 데이터베이스 개발자 안내서의 GRANT를 참조하세요.

```
GRANT ALL ON ALL TABLES IN SCHEMA public to "IAMR:AppBuilderDataAccessRole";
```

2단계: 적절한 Amazon Redshift 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 Amazon Redshift 리소스를 사용하려면 관리자가 IAM 정책 및 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 정책은 빌더가 사용할 수 있는 데이터의 범위와 생성, 읽기, 업데이트 또는 삭제와 같이 해당 데이터에 대해 호출할 수 있는 작업을 제어합니다. 그런 다음 IAM 정책은 App Studio에서 사용하는 IAM 역할에 연결됩니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다. 예를 들어 빌더가 Amazon Redshift의 여러 테이블에서 지원하는 두 개의 애플리케이션을 생성하는 경우 관리자는 Amazon Redshift의 각 테이블에 대해 하나씩 두 개의 IAM 정책 및 역할을 생성해야 합니다.

2a단계: 적절한 Amazon Redshift 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 Amazon Redshift 권한을 사용하여 IAM 정책을 생성하려면

1. IAM 정책을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. JSON 정책 문서를 입력하거나 붙여 넣습니다. 다음 탭에는 프로비저닝된 Amazon Redshift와 서버리스 Amazon Redshift 모두에 대한 예제 정책이 포함되어 있습니다.

> **Note**
>
> 다음 정책은 와일드카드()를 사용하는 모든 Amazon Redshift 리소스에 적용됩니다*. 보안 모범 사례를 위해 와일드카드를 App Studio와 함께 사용할 리소스의 Amazon 리소스 이름(ARN)으로 바꿔야 합니다.

Provisioned

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
      "Sid": "ProvisionedRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift:DescribeClusters",
            "redshift:GetClusterCredentialsWithIAM",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
            "redshift-data:ExecuteStatement",
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

Serverless

JSON

```
{
   "Version":"2012-10-17",
```

```
   "Statement": [
      {
      "Sid": "ServerlessRedshiftForAppStudio",
      "Effect": "Allow",
         "Action": [
            "redshift-serverless:ListNamespaces",
            "redshift-serverless:GetCredentials",
            "redshift-serverless:ListWorkgroups",
            "redshift-data:ListDatabases",
            "redshift-data:ListTables",
            "redshift-data:DescribeTable",
            "redshift-data:DescribeStatement",
            "redshift-data:ExecuteStatement",
            "redshift-data:GetStatementResult"
         ],
         "Resource": "*"
      }
   ]
}
```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **RedshiftServerlessForAppStudio** 또는 , 설명(선택 사항)**RedshiftProvisionedForAppStudio**과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 Amazon Redshift 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성

이제 이전에 생성한 정책을 사용하는 IAM 역할을 생성합니다. App Studio는이 정책을 사용하여 구성된 Amazon Redshift 리소스에 액세스할 수 있습니다.

App Studio에 Amazon Redshift 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
5. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

- *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
- *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

6. 권한 추가에서 이전 단계(**RedshiftServerlessForAppStudio** 또는)에서 생성한 정책을 검색하고 선택합니다**RedshiftProvisionedForAppStudio**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

7. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.

> **Important**
>
> 여기서 역할 이름은 1단계: Amazon Redshift 리소스 생성 및 구성 (*AppBuilderDataAccessRole*)의 GRANT 명령에 사용된 역할 이름과 일치해야 합니다.

8. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.
   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`
9. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 Amazon Redshift 커넥터를 생성할 때 필요합니다.

3단계: Amazon Redshift 커넥터 생성

이제 Amazon Redshift 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 Amazon Redshift에 연결하는 데 사용할 수 있는 커넥터를 App Studio에 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Amazon Redshift용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. Amazon Redshift 커넥터를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.
   - 이름: 커넥터의 이름을 입력합니다.
   - 설명: 커넥터에 대한 설명을 제공합니다.

- IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 Amazon Redshift 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
- 리전: Amazon Redshift 리소스가 있는 AWS 리전을 선택합니다.
- 컴퓨팅 유형: Amazon Redshift Serverless 또는 프로비저닝된 클러스터를 사용하고 있는지 선택합니다.
- 클러스터 또는 작업 그룹 선택: 프로비저닝됨을 선택한 경우 App Studio에 연결할 클러스터를 선택합니다. 서버리스를 선택한 경우 작업 그룹을 선택합니다.
- 데이터베이스 선택: App Studio에 연결할 데이터베이스를 선택합니다.
- 사용 가능한 테이블: App Studio에 연결할 테이블을 선택합니다.

6. 다음을 선택합니다. 연결 정보를 검토하고 생성을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

Amazon DynamoDB에 연결

App Studio를 DynamoDB와 연결하여 빌더가 애플리케이션에서 DynamoDB 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. 1단계: DynamoDB 리소스 생성 및 구성
2. 2단계: 적절한 DynamoDB 권한을 사용하여 IAM 정책 및 역할 생성
3. DynamoDB 커넥터 생성

1단계: DynamoDB 리소스 생성 및 구성

다음 절차에 따라 App Studio와 함께 사용할 DynamoDB 리소스를 생성하고 구성합니다.

App Studio와 함께 사용할 DynamoDB를 설정하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/dynamodb/ DynamoDB 콘솔을 엽니다.

   에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 테이블을 선택합니다.
3. 테이블 생성을 선택합니다.
4. 테이블의 이름과 키를 입력합니다.

5. 테이블 생성을 선택합니다.
6. 테이블이 생성된 후 테이블이 App Studio에 연결되면 표시되도록 일부 항목을 추가합니다.

    a. 테이블을 선택하고 작업을 선택한 다음 항목 탐색을 선택합니다.

    b. 반환된 항목에서 항목 생성을 선택합니다.

    c. (선택 사항): 새 속성 추가를 선택하여 테이블에 속성을 더 추가합니다.

    d. 각 속성의 값을 입력하고 항목 생성을 선택합니다.

2단계: 적절한 DynamoDB 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 DynamoDB 리소스를 사용하려면 관리자가 IAM 정책 및 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 정책은 빌더가 사용할 수 있는 데이터의 범위와 생성, 읽기, 업데이트 또는 삭제와 같이 해당 데이터에 대해 호출할 수 있는 작업을 제어합니다. 그런 다음 IAM 정책은 App Studio에서 사용하는 IAM 역할에 연결됩니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다. 예를 들어 빌더가 DynamoDB의 동일한 테이블에 의해 지원되는 두 개의 애플리케이션을 생성하는 경우, 하나는 읽기 액세스만 필요하고 다른 하나는 읽기, 생성, 업데이트 및 삭제가 필요합니다. 관리자는 DynamoDB의 해당 테이블에 대한 전체 CRUD 권한이 있는 두 개의 IAM 역할을 생성해야 합니다.

2a단계: 적절한 DynamoDB 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 DynamoDB 권한을 사용하여 IAM 정책을 생성하려면

1. IAM 정책을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. JSON 정책 문서를 입력하거나 붙여 넣습니다. 다음 탭에는 DynamoDB 테이블에 대한 읽기 전용 및 전체 액세스 정책 예제와 AWS KMS 고객 관리형 키(CMK)로 암호화된 DynamoDB 테이블에 대한 AWS KMS 권한이 포함된 정책 예제가 포함되어 있습니다.

> **Note**
>
> 다음 정책은 와일드카드()를 사용하는 모든 DynamoDB 리소스에 적용됩니다*. 보안 모범 사례를 위해 와일드카드를 App Studio와 함께 사용할 리소스의 Amazon 리소스 이름(ARN)으로 바꿔야 합니다.

Read only

다음 정책은 구성된 DynamoDB 리소스에 대한 읽기 액세스 권한을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
      "Sid": "ReadOnlyDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
            "dynamodb:ListTables",
            "dynamodb:DescribeTable",
            "dynamodb:PartiQLSelect"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

다음 정책은 구성된 DynamoDB 리소스에 대한 생성, 읽기, 업데이트 및 삭제 액세스 권한을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
```

```
   "Statement": [
      {
      "Sid": "FullAccessDDBForAppStudio",
      "Effect": "Allow",
         "Action": [
           "dynamodb:ListTables",
           "dynamodb:DescribeTable",
           "dynamodb:PartiQLSelect",
           "dynamodb:PartiQLInsert",
           "dynamodb:PartiQLUpdate",
           "dynamodb:PartiQLDelete"
         ],
         "Resource": "*"
      }
   ]
}
```

Read only - KMS encrypted

다음 정책은 AWS KMS 권한을 제공하여 구성된 암호화된 DynamoDB 리소스에 대한 읽기 액세스 권한을 부여합니다. ARN을 AWS KMS 키의 ARN으로 바꿔야 합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyDDBForAppStudio",
            "Effect": "Allow",
            "Action": [
                "dynamodb:ListTables",
                "dynamodb:DescribeTable",
                "dynamodb:PartiQLSelect"
            ],
            "Resource": "*"
        },
        {
            "Sid": "KMSPermissionsForEncryptedTable",
            "Effect": "Allow",
            "Action": [
```

```
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

Full access - KMS encrypted

다음 정책은 AWS KMS 권한을 제공하여 구성된 암호화된 DynamoDB 리소스에 대한 읽기 액세스 권한을 부여합니다. ARN을 AWS KMS 키의 ARN으로 바꿔야 합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "ReadOnlyDDBForAppStudio",
            "Effect": "Allow",
            "Action": [
                "dynamodb:ListTables",
                "dynamodb:DescribeTable",
                "dynamodb:PartiQLSelect",
                "dynamodb:PartiQLInsert",
                "dynamodb:PartiQLUpdate",
                "dynamodb:PartiQLDelete"
            ],
            "Resource": "*"
        },
        {
            "Sid": "KMSPermissionsForEncryptedTable",
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "kms:DescribeKey"
            ],
            "Resource": "arn:aws:kms:us-
west-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
```

```
        }
    ]
}
```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **ReadOnlyDDBForAppStudio** 또는 , 설명(선택 사항)**FullAccessDDBForAppStudio**과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 DynamoDB 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성

이제 이전에 생성한 정책을 사용하는 IAM 역할을 생성합니다. App Studio는이 정책을 사용하여 구성된 DynamoDB 리소스에 액세스합니다.

App Studio에 DynamoDB 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 이전 단계(**ReadOnlyDDBForAppStudio** 또는)에서 생성한 정책을 검색하고 선택합니다**FullAccessDDBForAppStudio**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

6. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 DynamoDB 커넥터를 생성할 때 필요합니다.

DynamoDB 커넥터 생성

이제 DynamoDB 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 DynamoDB에 연결하는 데 사용할 수 있는 커넥터를 App Studio에서 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

DynamoDB용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. 커넥터 유형 목록에서 Amazon DynamoDB를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.
   - 이름: DynamoDB 커넥터의 이름을 입력합니다.
   - 설명: DynamoDB 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 DynamoDB 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 리전: DynamoDB 리소스가 위치한 AWS 리전을 선택합니다.
   - 사용 가능한 테이블: App Studio에 연결할 테이블을 선택합니다.
6. 다음을 선택합니다. 연결 정보를 검토하고 생성을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

에 연결 AWS Lambda

App Studio를 Lambda와 연결하여 빌더가 애플리케이션에서 Lambda 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. 1단계: Lambda 함수 생성 및 구성
2. 2단계: App Studio에 Lambda 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성
3. 3단계: Lambda 커넥터 생성

1단계: Lambda 함수 생성 및 구성

기존 Lambda 함수가 없는 경우 먼저 함수를 생성해야 합니다. Lambda 함수 생성에 대한 자세한 내용은 AWS Lambda 개발자 안내서를 참조하세요.

2단계: App Studio에 Lambda 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성

App Studio에서 Lambda 리소스를 사용하려면 관리자가 IAM 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 역할은 애플리케이션이 Lambda에서 액세스할 수 있는 리소스 또는 작업을 제어합니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

App Studio에 Lambda 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
```

```
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 역할에 적절한 권한을 부여하는 정책을 검색하고 선택합니다. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다. Lambda의 경우 Lambda 함수를 호출할 수 있는 권한을 부여하는 `AWSLambdaRole` 정책을 추가하는 것이 좋습니다.

   관리형 정책 목록 및 설명을 포함하여 Lambda에서 IAM 정책을 사용하는 방법에 대한 자세한 내용은 AWS Lambda 개발자 안내서의 Identity and Access Management for AWS Lambda 섹션을 참조하세요.

   다음을 선택합니다.

6. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 Lambda 커넥터를 생성할 때 필요합니다.

3단계: Lambda 커넥터 생성

이제 Lambda 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 Lambda에 연결하는 데 사용할 수 있는 커넥터를 App Studio에 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Lambda용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. 커넥터 유형 목록에서 기타 AWS 서비스를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.
   - 이름: Lambda 커넥터의 이름을 입력합니다.
   - 설명: Lambda 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2단계: App Studio에 Lambda 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 서비스: Lambda를 선택합니다.
   - 리전: Lambda 리소스가 있는 AWS 리전을 선택합니다.
6. 생성(Create)을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

Amazon Simple Storage Service(Amazon S3)에 연결

App Studio를 Amazon S3와 연결하여 빌더가 애플리케이션에서 Amazon S3 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행합니다.

1. 1단계: Amazon S3 리소스 생성 및 구성
2. 2단계: 적절한 Amazon S3 권한을 사용하여 IAM 정책 및 역할 생성
3. 3단계: Amazon S3 커넥터 생성

단계를 완료하고 적절한 권한이 있는 커넥터를 생성한 후 빌더는 커넥터를 사용하여 Amazon S3 리소스와 상호 작용하는 앱을 생성할 수 있습니다. App Studio 앱에서 Amazon S3와 상호 작용하는 방법에

대한 자세한 내용은 섹션을 참조하세요Amazon Simple Storage Service와 구성 요소 및 자동화의 상호 작용.

1단계: Amazon S3 리소스 생성 및 구성

앱의 요구 사항과 기존 리소스에 따라 앱이 쓰고 읽을 Amazon S3 버킷을 생성해야 할 수 있습니다. 버킷을 포함한 Amazon S3 리소스 생성에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 Amazon Amazon S3 시작하기를 참조하세요.

앱에서 S3 업로드 구성 요소를 사용하려면 업로드하려는 Amazon S3 버킷에 CORS(Cross-Origin Resource Sharing) 구성을 추가해야 합니다. CORS 구성은 객체를 버킷으로 푸시할 수 있는 권한을 App Studio에 부여합니다. 다음 절차에서는 콘솔을 사용하여 Amazon S3 버킷에 CORS 구성을 추가하는 방법을 자세히 설명합니다. CORS 및 구성에 대한 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 CORS(Cross-Origin 리소스 공유) 사용을 참조하세요.

콘솔에서 Amazon S3 버킷에 CORS 구성을 추가하려면

1. https://console.aws.amazon.com/s3/ 버킷으로 이동합니다.
2. 권한 탭을 선택합니다.
3. CORS(Cross-Origin Resource Sharing)에서 편집을 선택합니다.
4. 다음 코드 조각을 추가합니다.

```
[
    {
        "AllowedHeaders": [
            "*"
        ],
        "AllowedMethods": [
            "PUT",
            "POST"
        ],
        "AllowedOrigins": [
            "*"
        ],
        "ExposeHeaders": []
    }
]
```

5. 변경 사항 저장을 선택합니다.

2단계: 적절한 Amazon S3 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 Amazon S3 리소스를 사용하려면 관리자가 IAM 정책 및 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 정책은 빌더가 사용할 수 있는 데이터의 범위와 생성, 읽기, 업데이트 또는 삭제와 같이 해당 데이터에 대해 호출할 수 있는 작업을 제어합니다. 그런 다음 IAM 정책은 App Studio에서 사용하는 IAM 역할에 연결됩니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다. 예를 들어 빌더가 Amazon S3의 서로 다른 버킷이 지원하는 두 개의 애플리케이션을 생성하는 경우 관리자는 각 버킷에 대해 하나씩 두 개의 IAM 정책 및 역할을 생성해야 합니다.

2a단계: 적절한 Amazon S3 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 Amazon S3 권한을 사용하여 IAM 정책을 생성하려면

1. IAM 정책을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. JSON 정책 문서를 입력하거나 붙여 넣습니다. 다음 탭에는 Amazon S3 리소스에 대한 읽기 전용 및 전체 액세스에 대한 정책 예제가 포함되어 있습니다.

> **Note**
>
> 다음 정책은 와일드카드()를 사용하는 모든 Amazon S3 리소스에 적용됩니다*. 모범 보안 사례를 위해 와일드카드를 App Studio와 함께 사용할 버킷 또는 폴더와 같은 리소스의 Amazon 리소스 이름(ARN)으로 바꿔야 합니다.

Read only

다음 정책은 구성된 Amazon S3 버킷 또는 폴더에 대한 읽기 전용 액세스(가져오기 및 나열)를 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "S3ReadOnlyForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:GetObject",
            "s3:ListBucket"
         ],
         "Resource": "*"
      }
   ]
}
```

Full access

다음 정책은 구성된 Amazon S3 버킷 또는 폴더에 대한 전체 액세스 권한(입력, 가져오기, 나열 및 삭제)을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "S3FullAccessForAppStudio",
         "Effect": "Allow",
         "Action": [
            "s3:PutObject",
            "s3:GetObject",
            "s3:ListBucket",
            "s3:DeleteObject"
         ],
         "Resource": "*"
      }
   ]
```

```
}
```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **AWSAppStudioS3FullAccess**, 설명(선택 사항)과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 Amazon S3 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성

App Studio에서 Amazon S3 리소스를 사용하려면 관리자가 IAM 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 역할은 빌더가 사용할 수 있는 데이터의 범위와 생성, 읽기, 업데이트 또는 삭제와 같이 해당 데이터에 대해 호출할 수 있는 작업을 제어합니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

App Studio에 Amazon S3 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 이전 단계(**S3ReadOnlyForAppStudio** 또는)에서 생성한 정책을 검색하고 선택합니다**S3FullAccessForAppStudio**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

6. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. 다음 단계에서 App Studio에서 Amazon S3 커넥터를 생성하는 데 필요합니다.

3단계: Amazon S3 커넥터 생성

이제 Amazon S3 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 Amazon S3에 연결하는 데 사용할 수 있는 커넥터를 App Studio에 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Amazon S3용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. Amazon S3 커넥터를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.
   - 이름: Amazon S3 커넥터의 이름을 입력합니다.
   - 설명: Amazon S3 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 Amazon S3 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 리전: Amazon S3 리소스가 위치한 AWS 리전을 선택합니다.
6. 생성(Create)을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

Amazon Aurora에 연결

App Studio를 Aurora와 연결하여 빌더가 애플리케이션에서 Aurora 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. 1단계: Aurora 리소스 생성 및 구성
2. 2단계: 적절한 Aurora 권한을 사용하여 IAM 정책 및 역할 생성
3. 3단계: App Studio에서 Aurora 커넥터 생성

App Studio는 다음 Aurora 버전을 지원합니다.

- Aurora MySQL Serverless V1: 5.72

- Aurora PostgreSQL Serverless V1: 11.18, 13.9
- Aurora MySQL Serverless V2: 13.11 이상, 14.8 이상 및 15.3 이상
- Aurora PostgreSQL Serverless V2: 13.11 이상, 14.8 이상 및 15.3 이상

1단계: Aurora 리소스 생성 및 구성

App Studio에서 Aurora 데이터베이스를 사용하려면 먼저 데이터베이스를 생성하고 적절하게 구성해야 합니다. App Studio에서 지원하는 Aurora 데이터베이스 유형은 Aurora PostgreSQL과 Aurora MySQL의 두 가지입니다. 유형을 비교하려면 MySQL과 PostgreSQL의 차이점은 무엇인가요?를 참조하세요. 적절한 탭을 선택하고 절차에 따라 App Studio 앱에 사용할 Aurora를 설정합니다.

Aurora PostgreSQL

다음 절차에 따라 App Studio와 함께 사용할 Aurora PostgreSQL 데이터베이스 클러스터를 생성하고 구성합니다.

App Studio와 함께 사용할 Aurora를 설정하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/rds/ Amazon RDS 콘솔을 엽니다.
2. 데이터베이스 생성을 선택합니다.
3. Aurora(PostgreSQL 호환)를 선택합니다.
4. 사용 가능한 버전에서 버전 , 13.11 14.8및 이상의 버전을 선택합니다15.3.
5. 설정에서 DB 클러스터 식별자를 입력합니다.
6. 인스턴스 구성에서 서버리스 v2를 선택하고 적절한 용량을 선택합니다.
7. 연결에서 RDS 데이터 API 활성화를 선택합니다.
8. 데이터베이스 인증에서 IAM 데이터베이스 인증을 선택합니다.
9. 추가 구성의 초기 데이터베이스 이름에 데이터베이스의 초기 데이터베이스 이름을 입력합니다.

Aurora MySQL

다음 절차에 따라 App Studio와 함께 사용할 Aurora MySQL 데이터베이스 클러스터를 생성하고 구성합니다.

Aurora MySQL은 데이터 API 또는 서버리스 v1을 지원하는 버전에 대해 UI에서의 생성을 지원하지 않습니다. 데이터 API를 지원하는 Aurora MySQL 클러스터를 생성하려면를 사용해야 합니다 AWS CLI.

> **Note**
>
> App Studio에서 Aurora MySQL 데이터베이스를 사용하려면 Virtual Private Cloud(VPC)에 있어야 합니다. 자세한 내용은 Amazon Aurora 사용 설명서의 VPC에서 DB 클러스터 작업을 참조하세요.

App Studio와 함께 사용할 Aurora MySQL을 설정하려면

1. 필요한 경우 AWS Command Line Interface 사용 설명서의 설치 또는 최신 버전의 로 업데이트 AWS CLI의 지침에 AWS CLI 따라를 설치합니다.
2. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/rds/ Amazon RDS 콘솔을 엽니다.
3. 왼쪽 탐색에서 서브넷 그룹을 선택합니다.
4. DB 서브넷 그룹 생성을 선택합니다.
5. 정보를 입력하고 Sunbnet 그룹을 생성합니다. 서브넷 그룹 및 VPCs 사용에 대한 자세한 내용은 Amazon Aurora 사용 설명서의 VPC에서 DB 클러스터 작업을 참조하세요.
6. 다음 AWS CLI 명령을 실행합니다.

```
aws rds create-db-cluster --database-name db_name \
    --db-cluster-identifier db_cluster_identifier \
    --engine aurora-mysql \
    --engine-version 5.7.mysql_aurora.2.08.3 \
    --engine-mode serverless \
    --scaling-configuration
 MinCapacity=4,MaxCapacity=32,SecondsUntilAutoPause=1000,AutoPause=true \
    --master-username userName \
    --master-user-password userPass \
    --availability-zones us-west-2b us-west-2c \
    --db-subnet-group-name subnet-group-name
```

다음 필드를 교체합니다.

- *db_name*을 원하는 데이터베이스 이름으로 바꿉니다.

- *db_cluster_identifier*를 원하는 데이터베이스 클러스터 식별자로 바꿉니다.
- (선택 사항) scaling-configuration 필드의 숫자를 원하는 대로 바꿉니다.
- *userName*을 원하는 사용자 이름으로 바꿉니다.
- *userPass*를 원하는 암호로 바꿉니다.
- 에서 생성한 서브넷 그룹의 가용 영역을 availability-zones추가합니다.
- *subnet-group-name*을 생성한 서브넷 그룹의 이름으로 바꿉니다.

2단계: 적절한 Aurora 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 Aurora 리소스를 사용하려면 관리자가 IAM 정책을 생성하여 App Studio에 구성된 리소스에 액세스할 수 있는 권한을 부여하는 데 사용되는 IAM 역할에 연결해야 합니다. IAM 정책 및 역할은 빌더가 사용할 수 있는 데이터의 범위와 생성, 읽기, 업데이트 또는 삭제와 같이 해당 데이터에 대해 호출할 수 있는 작업을 제어합니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

2a단계: 적절한 Aurora 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 Aurora 권한으로 IAM 정책을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. 기존 코드 조각을 다음 코드 조각으로 바꾸고 *111122223333*을 AWS Amazon Redshift 및 Aurora 리소스가 포함된 계정 번호로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Sid": "BaselineAuroraForAppStudio",
            "Effect": "Allow",
            "Action": [
                "rds-data:ExecuteStatement",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:rds:*:111122223333:cluster:*",
                "arn:aws:secretsmanager:*:111122223333:secret:rds*"
            ]
        }
    ]
}
```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **Aurora_AppStudio** 및 설명(선택 사항)과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 Aurora 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성

이제 이전에 생성한 정책을 사용하는 IAM 역할을 생성합니다. App Studio는이 정책을 사용하여 구성된 Aurora 리소스에 액세스합니다.

App Studio에 Aurora 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.

- *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 이전에 생성한 정책()을 검색하고 선택합니다**Aurora_AppStudio**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

6. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 Aurora 커넥터를 생성할 때 필요합니다.

3단계: App Studio에서 Aurora 커넥터 생성

이제 Aurora 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 Aurora에 연결하는 데 사용할 수 있는 커넥터를 App Studio에 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Aurora용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. Amazon Aurora 커넥터를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.

   - 이름: Aurora 커넥터의 이름을 입력합니다.
   - 설명: Aurora 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 Aurora 리소스에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 보안 암호 ARN: 데이터베이스 클러스터의 보안 암호 ARN을 입력합니다. 보안 암호 ARN을 찾을 수 있는 위치에 대한 자세한 내용은 Amazon Aurora 사용 설명서의 DB cluser의 보안 암호에 대한 세부 정보 보기를 참조하세요.
   - 리전: Aurora 리소스가 있는 AWS 리전을 선택합니다.
   - 데이터베이스 ARN: 데이터베이스 클러스터의 ARN을 입력합니다. ARN은 보안 암호 ARN과 마찬가지로 데이터베이스 클러스터의 구성 탭에서 찾을 수 있습니다.
   - 데이터베이스 유형:에서 생성된 데이터베이스 유형과 일치하는 데이터베이스 유형인 MySQL 또는 PostgreSQL을 선택합니다1단계: Aurora 리소스 생성 및 구성.

- 데이터베이스 이름: 데이터베이스 이름을 입력합니다. 데이터베이스 이름은 데이터베이스 클러스터의 구성 탭에서도 찾을 수 있습니다.
- 사용 가능한 테이블:이 커넥터를 사용하여 App Studio와 함께 사용할 테이블을 선택합니다.

6. 다음을 선택하여 개체 매핑을 검토하거나 정의합니다.
7. 생성을 선택하여 Aurora 커넥터를 생성합니다. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

Amazon Bedrock에 연결

빌더가 애플리케이션에서 Amazon Bedrock에 액세스하고 사용할 수 있도록 App Studio를 Amazon Bedrock과 연결하려면 다음 단계를 수행해야 합니다.

1. 1단계: Amazon Bedrock 모델 활성화
2. 2단계: 적절한 Amazon Bedrock 권한을 사용하여 IAM 정책 및 역할 생성
3. 3단계: Amazon Bedrock 커넥터 생성

1단계: Amazon Bedrock 모델 활성화

다음 절차에 따라 Amazon Bedrock 모델을 활성화합니다.

Amazon Bedrock 모델을 활성화하려면

1. 에 로그인 AWS Management Console 하고 https://console.aws.amazon.com/bedrock/ Amazon Bedrock 콘솔을 엽니다.
2. 왼쪽 탐색 창에서 모델 액세스를 선택합니다.
3. 사용하려는 모델을 활성화합니다. 자세한 내용은 Amazon Bedrock 파운데이션 모델에 대한 액세스 관리를 참조하세요.

2단계: 적절한 Amazon Bedrock 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 Amazon Bedrock 리소스를 사용하려면 관리자가 IAM 정책 및 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 정책은와 같은 리소스에 대해 호출할 수 있는 리소스와 작업을 제어합니다`InvokeModel`. 그런 다음 IAM 정책은 App Studio에서 사용하는 IAM 역할에 연결됩니다.

2a단계: 적절한 Amazon Bedrock 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 Amazon Bedrock 권한을 사용하여 IAM 정책을 생성하려면

1. IAM 정책을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. JSON 정책 문서를 입력하거나 붙여 넣습니다. 다음 예제 정책은 와일드카드()를 사용하여 모든 Amazon Bedrock 리소스`InvokeModel`에를 제공합니다*.

   보안 모범 사례를 위해 와일드카드를 App Studio와 함께 사용할 리소스의 Amazon 리소스 이름(ARN)으로 바꿔야 합니다.

   JSON

   ```
   {
      "Version":"2012-10-17",
      "Statement": [
         {
            "Sid": "BedrockAccessForAppStudio",
            "Effect": "Allow",
            "Action": [
               "bedrock:InvokeModel"
            ],
            "Resource": "*"
         }
      ]
   }
   ```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **BedrockAccessForAppStudio**, 설명(선택 사항)과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 Amazon Bedrock에 대한 액세스 권한을 부여하는 IAM 역할 생성

Amazon Bedrock을 App Studio와 함께 사용하려면 관리자가 IAM 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 역할은 App Studio 앱이 사용할 권한 범위를 제어하며 커넥터를 생성할 때 사용됩니다. 서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

App Studio에 Amazon Bedrock에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
```

```
                    "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 이전 단계()에서 생성한 정책을 검색하고 선택합니다**BedrockAccessForAppStudio**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

6. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 App Studio 액세스를 제공하는 다음 태그를 추가합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. 다음 단계에서 App Studio에서 Amazon Bedrock 커넥터를 생성할 때 필요합니다.

3단계: Amazon Bedrock 커넥터 생성

이제 Amazon Bedrock 리소스와 IAM 정책 및 역할을 구성했으므로 해당 정보를 사용하여 빌더가 앱을 Amazon Bedrock에 연결하는 데 사용할 수 있는 커넥터를 App Studio에 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Amazon Bedrock용 커넥터를 생성하려면

1. App Studio로 이동합니다.

2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. 커넥터 유형 목록에서 기타 AWS 서비스를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.

   - 이름: Amazon Bedrock 커넥터의 이름을 입력합니다.
   - 설명: Amazon Bedrock 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 Amazon Bedrock에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 서비스: Bedrock 런타임을 선택합니다.

     > **Note**
     >
     > Bedrock 런타임은 Amazon Bedrock에서 호스팅되는 모델에 대한 추론 요청을 하는 데 사용되는 반면 Bedrock은 모델을 관리, 훈련 및 배포하는 데 사용됩니다.

   - 리전: Amazon Bedrock 리소스가 있는 AWS 리전을 선택합니다.
6. 생성(Create)을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

Amazon Simple Email Service에 연결

App Studio를 Amazon SES와 연결하여 빌더가 이를 사용하여 앱에서 이메일 알림을 보낼 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. 1단계: Amazon SES 리소스 구성
2. 2단계: 적절한 Amazon SES 권한을 사용하여 IAM 정책 및 역할 생성
3. 3단계: Amazon SES 커넥터 생성

1단계: Amazon SES 리소스 구성

그렇지 않은 경우 먼저 이메일을 보내는 데 사용하도록 Amazon SES를 구성해야 합니다. Amazon SES 설정에 대한 자세한 내용은 Amazon Simple Email Service 개발자 안내서의 Amazon Simple Email Service 시작하기를 참조하세요.

2단계: 적절한 Amazon SES 권한을 사용하여 IAM 정책 및 역할 생성

App Studio에서 Amazon SES 리소스를 사용하려면 관리자가 IAM 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 역할은 App Studio 앱에서 사용할 수 있는 Amazon SES 함수 또는 리소스를 제어합니다.

서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

2a단계: 적절한 Amazon SES 권한을 사용하여 IAM 정책 생성

App Studio에서 생성하고 사용하는 IAM 정책에는 애플리케이션이 모범 보안 사례를 따르는 데 필요한 적절한 리소스에 대한 최소 권한만 포함되어야 합니다.

적절한 Amazon SES 권한을 사용하여 IAM 정책을 생성하려면

1. IAM 정책을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 정책을 선택합니다.
3. 정책 생성을 선택합니다.
4. 정책 편집기 섹션에서 JSON 옵션을 선택합니다.
5. 다음 JSON 정책 문서를 입력하거나 붙여 넣습니다.

> **Note**
>
> 다음 정책은 와일드카드()를 사용하는 모든 Amazon SES 리소스에 적용됩니다*. 보안 모범 사례를 위해 와일드카드를 App Studio와 함께 사용할 리소스의 Amazon 리소스 이름(ARN)으로 바꿔야 합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "ses:SendEmail",
            "Resource": "*"
```

```
        }
    ]
}
```

6. 다음을 선택합니다.
7. 검토 및 생성 페이지에서 **SESForAppStudioPolicy**, 설명(선택 사항)과 같은 정책 이름을 제공합니다.
8. 정책 생성을 선택하여 정책을 생성합니다.

2b단계: App Studio에 Amazon SES에 대한 액세스 권한을 부여하는 IAM 역할 생성

이제 이전에 생성한 정책을 사용하는 IAM 역할을 생성합니다. App Studio는이 정책을 사용하여 Amazon SES에 액세스합니다.

App Studio에 Amazon SES에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 왼쪽 탐색 창에서 역할을 선택합니다.
3. 역할 생성을 선택합니다.
4. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
5. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-55555555555555*를 App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
```

```
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

6. 권한 추가에서 이전 단계()에서 생성한 정책을 검색하고 선택합니다**SESForAppStudioPolicy**. 정책 옆에 있는 +를 선택하면 정책이 확장되어 정책이 부여한 권한이 표시되고 확인란을 선택하면 정책이 선택됩니다.

   다음을 선택합니다.

7. 이름, 검토 및 생성 페이지에서 역할 이름 및 설명을 제공합니다.
8. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

9. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 Amazon SES 커넥터를 생성할 때 필요합니다.

3단계: Amazon SES 커넥터 생성

이제 Amazon SES와 IAM 정책 및 역할이 구성되었으므로 해당 정보를 사용하여 빌더가 앱에서 Amazon SES를 사용하는 데 사용할 수 있는 커넥터를 App Studio에서 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

Amazon SES용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. 커넥터 유형 목록에서 기타 AWS 서비스를 선택합니다.
5. 다음 필드를 작성하여 커넥터를 구성합니다.
   - 이름: Amazon SES 커넥터의 이름을 입력합니다.
   - 설명: Amazon SES 커넥터에 대한 설명을 입력합니다.
   - IAM 역할:에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다2b단계: App Studio에 Amazon SES에 대한 액세스 권한을 부여하는 IAM 역할 생성. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.
   - 서비스: Simple Email Service를 선택합니다.
   - 리전: Amazon SES 리소스가 위치한 AWS 리전을 선택합니다.
6. 생성(Create)을 선택합니다.
7. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

기타 AWS 서비스 커넥터를 사용하여 AWS 서비스에 연결

App Studio는 특정 AWS 서비스에 특정한 일부 커넥터를 제공하지만 기타 AWS 서비스 커넥터를 사용하여 다른 AWS 서비스에 연결할 수도 있습니다.

> **Note**
>
> 사용 가능한 경우 AWS 서비스별 커넥터를 사용하는 것이 좋습니다.

App Studio를 AWS 서비스와 연결하여 빌더가 애플리케이션에서 서비스의 리소스에 액세스하고 사용할 수 있도록 하려면 다음 단계를 수행해야 합니다.

1. IAM 역할을 생성하여 App Studio에 AWS 리소스에 대한 액세스 권한 부여
2. 기타 AWS 서비스 커넥터 생성

IAM 역할을 생성하여 App Studio에 AWS 리소스에 대한 액세스 권한 부여

App Studio에서 AWS 서비스 및 리소스를 사용하려면 관리자가 IAM 역할을 생성하여 App Studio에 리소스에 액세스할 수 있는 권한을 부여해야 합니다. IAM 역할은 빌더가 액세스할 수 있는 리소스의 범위와 리소스에 대해 호출할 수 있는 작업을 제어합니다. 서비스 및 정책당 하나 이상의 IAM 역할을 생성하는 것이 좋습니다.

App Studio에 AWS 리소스에 대한 액세스 권한을 부여하는 IAM 역할을 생성하려면

1. IAM 역할을 생성할 권한이 있는 사용자로 IAM 콘솔에 로그인합니다. 에서 생성된 관리 사용자를 사용하는 것이 좋습니다AWS 리소스 관리를 위한 관리 사용자 생성.
2. 콘솔의 탐색 창에서 역할을 선택한 후 역할 생성을 선택합니다.
3. 신뢰할 수 있는 엔터티 유형에서 사용자 지정 신뢰 정책을 선택합니다.
4. App Studio 애플리케이션이 계정에서이 역할을 수임하도록 허용하려면 기본 정책을 다음 정책으로 바꿉니다.

   정책에서 다음 자리 표시자를 바꿔야 합니다. 사용할 값은 App Studio의 계정 설정 페이지에서 찾을 수 있습니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *1111111-2222-3333-4444-5555555555555#* App Studio 인스턴스의 계정 설정에 인스턴스 ID로 나열된 App Studio 인스턴스 ID로 바꿉니다.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "AWS": "arn:aws:iam::111122223333:root"
            },
```

```
            "Action": "sts:AssumeRole",
            "Condition": {
                "StringEquals": {
                        "aws:PrincipalTag/IsAppStudioAccessRole": "true",
                        "sts:ExternalId":
 "11111111-2222-3333-4444-555555555555"
                }
            }
        }
    ]
}
```

다음을 선택합니다.

5. 권한 추가에서 역할에 적절한 권한을 부여하는 정책을 검색하고 선택합니다. 정책 옆의 +를 선택하면 정책이 부여된 권한을 표시하도록 정책이 확장되고 확인란을 선택하면 정책이 선택됩니다. IAM에 대한 자세한 내용은 IAM 사용 설명서를 참조하세요.

   다음을 선택합니다.

6. 역할 세부 정보에서 이름과 설명을 입력합니다.
7. 3단계: 태그 추가에서 새 태그 추가를 선택하여 다음 태그를 추가하여 App Studio 액세스를 제공합니다.

   - 키: `IsAppStudioDataAccessRole`
   - 값: `true`

8. 역할 생성을 선택하고 생성된 Amazon 리소스 이름(ARN)을 기록해 둡니다. App Studio에서 기타 AWS 서비스 커넥터를 생성할 때 필요합니다.

기타 AWS 서비스 커넥터 생성

이제 IAM 역할을 구성했으므로 해당 정보를 사용하여 App Studio에서 빌더가 앱을 서비스 및 리소스에 연결하는 데 사용할 수 있는 커넥터를 생성합니다.

> **Note**
>
> 커넥터를 생성하려면 App Studio에 관리자 역할이 있어야 합니다.

기타 AWS 서비스 커넥터를 사용하여 서비스에 연결하려면 AWS

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다.
3. + 커넥터 생성을 선택합니다.
4. 지원되는 AWS 서비스 목록의 커넥터 섹션에서 기타 서비스를 선택합니다. AWS
5. 다음 필드를 작성하여 AWS 서비스 커넥터를 구성합니다.
   - 이름: 커넥터의 이름을 입력합니다.
   - 설명: 커넥터에 대한 설명을 제공합니다.
   - IAM 역할: 에서 생성된 IAM 역할의 Amazon 리소스 이름(ARN)을 입력합니다IAM 역할을 생성하여 App Studio에 AWS 리소스에 대한 액세스 권한 부여.
   - 서비스: App Studio에 연결할 AWS 서비스를 선택합니다.
   - 리전: AWS 리소스가 위치한 AWS 리전을 선택합니다.
6. 생성(Create)을 선택합니다. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

CMKs에서 암호화된 데이터 소스 사용

이 주제에는 AWS KMS 고객 관리형 키(CMK)를 사용하여 암호화된 데이터 소스에 App Studio를 설정하고 연결하는 방법에 대한 정보가 포함되어 있습니다.

목차

- 암호화된 관리형 데이터 스토리지 테이블 사용
- 암호화된 DynamoDB 테이블 사용

암호화된 관리형 데이터 스토리지 테이블 사용

다음 절차에 따라 App Studio 앱의 관리형 스토리지 엔터티에서 사용하는 DynamoDB 테이블을 암호화합니다. 관리형 데이터 엔터티에 대한 자세한 내용은 섹션을 참조하세요AWS App Studio의 관리형 데이터 엔터티.

암호화된 관리형 데이터 스토리지 테이블을 사용하려면

1. 필요한 경우 App Studio의 애플리케이션에서 관리형 데이터 엔터티를 생성합니다. 자세한 내용은 App Studio 관리형 데이터 소스를 사용하여 엔터티 생성 단원을 참조하십시오.

2. 다음 단계를 수행하여 CMK로 테이블 데이터를 암호화하고 복호화할 수 있는 권한이 있는 정책 설명을 `AppStudioManagedStorageDDBAccess` IAM 역할에 추가합니다.

   a. IAM 콘솔(https://console.aws.amazon.com/iam/)을 엽니다.

   > **Important**
   >
   > App Studio 인스턴스를 생성하는 데 사용된 것과 동일한 계정을 사용해야 합니다.

   b. IAM 콘솔의 탐색 창에서 역할을 선택합니다.

   c. `AppStudioManagedStorageDDBAccess`을 선택합니다.

   d. 권한 정책에서 권한 추가를 선택한 다음 인라인 정책 생성을 선택합니다.

   e. JSON을 선택하고 콘텐츠를 다음 정책으로 바꾸고 다음을 바꿉니다.

   - *111122223333*을 App Studio 인스턴스를 설정하는 데 사용된 계정의 AWS 계정 번호로 바꿉니다. App Studio 인스턴스의 계정 설정에 AWS 계정 ID로 나열됩니다.
   - *CMK_id*를 CMK ID로 바꿉니다. 찾으려면 키 ID 및 키 ARN 찾기를 참조하세요.

3. 다음 단계를 수행하여 App Studio 관리형 데이터 엔터티에서 사용하는 DynamoDB 테이블을 암호화합니다.

   a. https://console.aws.amazon.com/dynamodbv2/ Amazon DynamoDB 콘솔을 엽니다.

   b. 암호화할 테이블을 선택합니다. App Studio에서 해당 개체의 연결 탭에서 테이블 이름을 찾을 수 있습니다.

   c. 부가 설정을 선택합니다.

   d. 암호화에서 암호화 관리를 선택합니다.

   e. 계정에 저장됨, 사용자가 소유 및 관리하는를 선택하고 CMK를 선택합니다.

4. 앱을 다시 게시하고 데이터 읽기 및 쓰기가 테스트 및 프로덕션 환경 모두에서 작동하고 다른 엔터티에서이 테이블을 사용하는 것이 예상대로 작동하는지 확인하여 변경 사항을 테스트합니다.

> **Note**
>
> 새로 추가된 관리형 데이터 엔터티는 기본적으로 DynamoDB 관리형 키를 사용하며 이전 단계에 따라 CMK를 사용하여 로 업데이트해야 합니다.

암호화된 DynamoDB 테이블 사용

다음 절차에 따라 App Studio 앱에서 사용할 암호화된 DynamoDB 테이블을 구성합니다.

암호화된 DynamoDB 테이블을 사용하려면

1. 다음 변경 사항1단계: DynamoDB 리소스 생성 및 구성과 함께의 지침을 따릅니다.
   - 암호화할 테이블을 구성합니다. 자세한 내용은 Amazon DynamoDB 개발자 안내서의 새 테이블에 대한 암호화 키 지정을 참조하세요.
2. 의 지침에 따라 다음 단계를 수행하여 CMK를 사용하여 테이블 데이터를 암호화하고 복호화할 수 있도록 허용하는 새 정책 설명을 추가하여 새 역할에 대한 권한 정책을 2단계: 적절한 DynamoDB 권한을 사용하여 IAM 정책 및 역할 생성업데이트합니다.
   a. 필요한 경우 IAM 콘솔에서 역할로 이동합니다.
   b. 권한 정책에서 권한 추가를 선택한 다음 인라인 정책 생성을 선택합니다.
   c. JSON을 선택하고 콘텐츠를 다음 정책으로 바꾸고 다음을 바꿉니다.
      - *team_account_id*를 계정 설정에서 찾을 수 있는 App Studio 팀 ID로 바꿉니다.
      - *CMK_id*를 CMK ID로 바꿉니다. 찾으려면 키 ID 및 키 ARN 찾기를 참조하세요.
3. 의 지침에 따라 이전에 생성한 역할을 DynamoDB 커넥터 생성 사용하여 커넥터를 생성합니다.
4. DynamoDB 커넥터와 테이블을 사용하는 앱을 테스트 또는 프로덕션에 게시하여 구성을 테스트합니다. 데이터 읽기 및 쓰기가 제대로 작동하는지 확인하고이 테이블을 사용하여 다른 개체도 생성합니다.

> **Note**
>
> 새 DynamoDB 테이블이 생성되면 이전 단계에 따라 CMK를 사용하여 암호화되도록 구성해야 합니다.

타사 서비스에 연결

주제

- OpenAPI 커넥터와 API 커넥터 비교
- 타사 서비스 및 APIs에 연결(일반)
- OpenAPI를 사용하여 서비스에 연결

- Salesforce에 연결

OpenAPI 커넥터와 API 커넥터 비교

App Studio 애플리케이션에서 타사 서비스로 API 요청을 보내려면 애플리케이션이 서비스로 인증하고 API 호출을 구성하는 데 사용하는 커넥터를 생성하고 구성해야 합니다. App Studio는 이를 위해 `API Connector` 및 `OpenAPI Connector` 커넥터 유형을 모두 제공하며, 이에 대한 설명은 다음과 같습니다.

- API 커넥터: 모든 유형의 REST API에 대한 인증 및 요청 정보를 구성하는 데 사용됩니다.
- OpenAPI 커넥터: OpenAPI 사양(OAS)을 채택한 APIs에 대한 인증 및 요청 정보를 구성하는 데 사용됩니다. OAS를 준수하는 APIs 표준화, 보안, 거버넌스, 설명서 등 여러 가지 이점을 제공합니다.

App Studio는 OAS를 준수하고 OpenAPI 사양 파일을 제공하는 모든 APIs에 `OpenAPI Connector` 대해를 사용할 것을 권장합니다. OpenAPI에 대한 자세한 내용은 Swagger 설명서의 OpenAPI란 무엇입니까?를 참조하세요.

타사 서비스 및 APIs에 연결(일반)

다음 절차에 따라 App Studio에서 일반 API 커넥터를 생성합니다. API Connector는 App Studio 앱에 타사 서비스, 리소스 또는 작업에 대한 액세스 권한을 제공하는 데 사용됩니다.

API Connector를 사용하여 타사 서비스에 연결하려면

1. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
2. + 커넥터 생성을 선택합니다.
3. API 커넥터를 선택합니다. 이제 다음 필드를 작성하여 커넥터를 구성합니다.
4. 커넥터 이름: 커넥터의 이름을 입력합니다.
5. 커넥터 설명: 커넥터에 대한 설명을 제공합니다.
6. 기본 URL: 타사 연결의 웹 사이트 또는 호스트입니다. 예를 들어 `www.slack.com`입니다.
7. 인증 방법: 대상 서비스로 인증할 방법을 선택합니다.
    - 없음: 인증 없이 대상 서비스에 액세스합니다.
    - 기본: 연결 중인 서비스에서 가져온 사용자 이름 및 암호를 사용하여 대상 서비스에 액세스합니다.

- 베어러 토큰: 서비스의 사용자 계정 또는 API 설정에서 얻은 인증 토큰의 토큰 값을 사용하여 대상 서비스에 액세스합니다.
- OAuth 2.0: 자격 증명이나 자격 증명을 공유하지 않고 App Studio에 서비스 및 리소스에 대한 액세스 권한을 부여하는 OAuth 2.0 프로토콜을 사용하여 대상 서비스에 액세스합니다. OAuth 2.0 인증 방법을 사용하려면 먼저 App Studio를 나타내는에 연결된 서비스에서 애플리케이션을 생성하여 필요한 정보를 얻어야 합니다. 해당 정보를 사용하여 다음 필드를 작성합니다.

  a. 클라이언트 자격 증명 흐름: 애플리케이션이 사용자 상호 작용 없이 자체적으로 작동하는 system-to-system 상호 작용에 적합합니다. 예를 들어 사용자가 추가한 새 레코드를 기반으로 Salesforce 레코드를 자동으로 업데이트하는 CRM 앱 또는 트랜잭션 데이터를 검색하여 보고서에 표시하는 앱이 있습니다.

     1. 클라이언트 ID에 대상 서비스에서 생성된 OAuth 앱에서 가져온 ID를 입력합니다.
     2. 클라이언트 보안 암호에 대상 서비스에서 생성된 OAuth 앱에서 가져온 보안 암호를 입력합니다.
     3. 액세스 토큰 URL에 대상 서비스에서 생성된 OAuth 앱에서 가져온 토큰 URL을 입력합니다.
     4. 필요에 따라 범위에 애플리케이션의 범위를 입력합니다. 범위는 애플리케이션에 필요한 권한 또는 액세스 수준입니다. 범위를 이해하려면 대상 서비스의 API 설명서를 참조하고 App Studio 앱에 필요한 범위만 구성합니다.

     연결 확인을 선택하여 인증 및 연결을 테스트합니다.

  b. 권한 부여 코드 흐름: 사용자를 대신하여 작업해야 하는 애플리케이션에 적합합니다. 예를 들어, 사용자가 로그인하여 지원 티켓을 보고 업데이트하는 고객 지원 앱 또는 각 팀원이 로그인하여 판매 데이터를 보고 관리하는 영업 앱입니다.

     1. 클라이언트 ID에 대상 서비스에서 생성된 OAuth 앱에서 가져온 ID를 입력합니다.
     2. 클라이언트 보안 암호에 대상 서비스에서 생성된 OAuth 앱에서 가져온 보안 암호를 입력합니다.
     3. 권한 부여 URL에 대상 서비스의 권한 부여 URL을 입력합니다.
     4. 액세스 토큰 URL에 대상 서비스에서 생성된 OAuth 앱에서 가져온 토큰 URL을 입력합니다.

5. 필요에 따라 범위에 애플리케이션의 범위를 입력합니다. 범위는 애플리케이션에 필요한 권한 또는 액세스 수준입니다. 범위를 이해하려면 대상 서비스의 API 설명서를 참조하고 App Studio 앱에 필요한 범위만 구성합니다.

8. 헤더: 요청 또는 응답에 대한 메타데이터를 제공하는 데 사용되는 HTTP 헤더를 추가합니다. 키와 값을 모두 추가하거나 빌더가 애플리케이션에 값을 제공할 수 있는 키만 제공할 수 있습니다.
9. 쿼리 파라미터: 요청 URL의 일부로 옵션, 필터 또는 데이터를 전달하는 데 사용되는 쿼리 파라미터를 추가합니다. 헤더와 마찬가지로 키와 값을 모두 제공하거나 빌더가 애플리케이션에 값을 제공할 수 있는 키만 제공할 수 있습니다.
10. 생성(Create)을 선택합니다. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

이제 커넥터가 생성되었으므로 빌더는 앱에서 커넥터를 사용할 수 있습니다.

OpenAPI를 사용하여 서비스에 연결

빌더가 요청을 보내고 서비스로부터 응답을 받는 애플리케이션을 빌드할 수 있도록 OpenAPI를 사용하여 App Studio를 서비스와 연결하려면 다음 단계를 수행합니다.

1. OpenAPI 사양 파일 가져오기 및 서비스 정보 수집
2. OpenAPI 커넥터 생성

OpenAPI 사양 파일 가져오기 및 서비스 정보 수집

OpenAPI를 사용하여 App Studio에 서비스를 연결하려면 다음 단계를 수행합니다.

1. App Studio에 연결하려는 서비스로 이동하여 OpenAPI 사양 JSON 파일을 찾습니다.

> **Note**
>
> App Studio는 OpenAPI 사양 버전 3.0.0 이상을 준수하는 OpenAPI 사양 파일을 지원합니다.

2. 다음을 포함하여 OpenAPI 커넥터를 구성하는 데 필요한 데이터를 수집합니다.

- 서비스에 연결하기 위한 기본 URL입니다.
- 토큰 또는 사용자 이름/암호와 같은 인증 자격 증명.
- 해당하는 경우 모든 헤더입니다.
- 해당하는 경우 모든 쿼리 파라미터입니다.

OpenAPI 커넥터 생성

OpenAPI용 커넥터를 생성하려면

1. App Studio로 이동합니다.
2. 왼쪽 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
3. + 커넥터 생성을 선택합니다.
4. 커넥터 유형 목록에서 OpenAPI 커넥터를 선택합니다. 이제 다음 필드를 작성하여 커넥터를 구성합니다.
5. 이름: OpenAPI 커넥터의 이름을 입력합니다.
6. 설명: OpenAPI 커넥터에 대한 설명을 입력합니다.
7. 기본 URL: 서비스에 연결하기 위한 기본 URL을 입력합니다.
8. 인증 방법: 대상 서비스로 인증할 방법을 선택합니다.
    - 없음: 인증 없이 대상 서비스에 액세스합니다.
    - 기본: 연결 중인 서비스에서 가져온 사용자 이름 및 암호를 사용하여 대상 서비스에 액세스합니다.
    - 베어러 토큰: 서비스의 사용자 계정 또는 API 설정에서 가져온 인증 토큰의 토큰 값을 사용하여 대상 서비스에 액세스합니다.
    - OAuth 2.0: 자격 증명이나 자격 증명을 공유하지 않고 App Studio에 서비스 및 리소스에 대한 액세스 권한을 부여하는 OAuth 2.0 프로토콜을 사용하여 대상 서비스에 액세스합니다. OAuth 2.0 인증 방법을 사용하려면 먼저 App Studio를 나타내는에 연결된 서비스에서 애플리케이션을 생성하여 필요한 정보를 얻어야 합니다. 해당 정보를 사용하여 다음 필드를 작성합니다.

        a. 클라이언트 자격 증명 흐름:

            1. 클라이언트 ID에 대상 서비스의 ID를 입력합니다.
            2. 클라이언트 보안 암호에 대상 서비스의 보안 암호를 입력합니다.
            3. 액세스 토큰 URL에 대상 서비스의 토큰 URL을 입력합니다.
            4. 필요에 따라 범위에 애플리케이션의 범위를 입력합니다. 범위는 애플리케이션에 필요한 권한 또는 액세스 수준입니다. 대상 서비스의 API 설명서를 참조하여 범위를 이해하고 App Studio 앱에 필요한 범위만 구성합니다.

호출할 때마다 서비스와 함께 전송할 변수를 추가하고 연결 확인을 선택하여 인증 및 연결을 테스트합니다.

b. 권한 부여 코드 흐름:

1. 클라이언트 ID에 대상 서비스의 ID를 입력합니다.
2. 클라이언트 보안 암호에 대상 서비스의 보안 암호를 입력합니다.
3. 권한 부여 URL에 대상 서비스의 권한 부여 URL을 입력합니다.
4. 액세스 토큰 URL에 대상 서비스의 토큰 URL을 입력합니다.
5. 필요에 따라 범위에 애플리케이션의 범위를 입력합니다. 범위는 애플리케이션에 필요한 권한 또는 액세스 수준입니다. 대상 서비스의 API 설명서를 참조하여 범위를 이해하고 App Studio 앱에 필요한 범위만 구성합니다.

9. 변수: 호출할 때마다 서비스에 전송할 변수를 추가합니다. 구성 중에 추가된 변수는 안전하게 저장되며 연결을 사용하는 애플리케이션의 런타임 중에만 액세스됩니다.
10. 헤더: 요청 또는 응답에 대한 메타데이터를 제공하는 데 사용되는 HTTP 헤더를 추가합니다. 키와 값을 모두 추가하거나 빌더가 애플리케이션에 값을 제공할 수 있는 키만 제공할 수 있습니다.
11. 쿼리 파라미터: 요청 URL의 일부로 옵션, 필터 또는 데이터를 전달하는 데 사용되는 쿼리 파라미터를 추가합니다. 헤더와 마찬가지로 키와 값을 모두 제공하거나 빌더가 애플리케이션에 값을 제공할 수 있는 키만 제공할 수 있습니다.
12. OpenAPI 사양 파일: 끌어서 놓거나 파일 선택을 선택하여 로컬 파일 시스템을 탐색하고 업로드할 파일을 선택하여 OpenAPI 사양 JSON 파일을 업로드합니다.

    추가되면 파일이 처리되고 사용 가능한 옵션 목록이 표시됩니다. 커넥터에 필요한 작업을 선택합니다.
13. 생성(Create)을 선택합니다. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

이제 커넥터가 생성되었으므로 빌더는 앱에서 커넥터를 사용할 수 있습니다.

Salesforce에 연결

빌더가 애플리케이션에서 Salesforce 리소스에 액세스하고 사용할 수 있도록 App Studio를 Salesforce와 연결하려면 Salesforce에서 연결된 앱을 생성 및 구성하고 App Studio에서 Salesforce 커넥터를 생성해야 합니다.

Salesforce를 App Studio에 연결하려면

1. App Studio의 탐색 창에서 관리 섹션에서 커넥터를 선택합니다. 기존 커넥터 목록과 각 커넥터에 대한 몇 가지 세부 정보가 표시된 페이지로 이동합니다.
2. + 커넥터 생성을 선택합니다.
3. 커넥터 유형 목록에서 Salesforce를 선택하여 커넥터 생성 페이지를 엽니다.
4. 다음 단계에서 Salesforce를 구성하는 데 사용할 리디렉션 URL을 기록해 둡니다.
5. 다음 단계는 Salesforce에서 연결된 앱을 생성하는 것입니다. 다른 탭 또는 창에서 Salesforce 인스턴스로 이동합니다.
6. 빠른 찾기 상자에서 App Manager를 검색**App Manager**한 다음 선택합니다.
7. 새 연결된 앱을 선택합니다.
8. 연결된 앱 이름 및 API 이름에 앱 이름을 입력합니다. App Studio 앱 이름과 일치할 필요는 없습니다.
9. 필요에 따라 연락처 정보를 제공합니다.
10. API(OAuth 설정 활성화) 섹션에서 OAuth 설정 활성화를 활성화합니다.
11. 콜백 URL에 App Studio에서 앞서 기록한 리디렉션 URL을 입력합니다.
12. 선택한 OAuth 범위의 목록에서 필요한 권한 범위를 추가합니다. App Studio는 Salesforce REST APIs와 상호 작용하여 계정, 사례, 연락처, 리드 및 기회의 5가지 객체에 대해 CRUD 작업을 수행할 수 있습니다. App Studio 앱에 모든 관련 권한 또는 범위가 있는지 확인하려면 전체 액세스(전체)를 추가하는 것이 좋습니다.
13. 지원되는 권한 부여 흐름에 대한 PKCE(Proof Key for Code Exchange) 확장 필요 옵션을 비활성화합니다. PKCE는 App Studio에서 지원되지 않습니다.
14. 웹 서버 흐름에 보안 암호 필요 및 토큰 새로 고침 흐름에 보안 암호 필요를 활성화하여 보안 모범 사례를 따릅니다.
15. App Studio는 다음 인증 흐름을 모두 지원합니다.

    - 클라이언트 자격 증명 흐름: 애플리케이션이 사용자 상호 작용 없이 자체적으로 작동하는 server-to-server 상호 작용에 적합합니다. 예를 들어 Salesforce 액세스 권한이 없는 임시 직원 팀에 대한 모든 리드 정보를 나열합니다.
    - 권한 부여 코드 흐름: 개인 데이터 액세스 또는 작업과 같이 사용자를 대신하여 작동하는 애플리케이션에 적합합니다. 예를 들어,이 앱을 통해 다른 작업을 수행하기 위해 자신이 소싱하거나 소유한 각 영업 관리자의 리드를 나열합니다.

    - 클라이언트 자격 증명 흐름의 경우:

a. 클라이언트 자격 증명 흐름 활성화를 활성화합니다. 메시지를 검토하고 확인합니다.

b. 앱을 저장합니다.

c. 흐름에는 사용자 상호 작용이 없지만 실행 사용자를 선택해야 합니다. Salesforce는 실행 사용자를 선택하여 사용자를 대신하여 액세스 토큰을 반환합니다.

1. App Manager의 앱 목록에서 App Studio 앱의 화살표를 선택하고 관리를 선택합니다.
2. 정책 편집을 선택합니다.
3. 클라이언트 자격 증명 흐름에서 적절한 사용자를 추가합니다.

- 권한 부여 코드 흐름의 경우 권한 부여 코드 및 자격 증명 흐름 활성화를 활성화합니다.

16. Salesforce는 다음 단계에서 App Studio에서 커넥터를 구성하는 데 사용해야 하는 클라이언트 ID와 클라이언트 보안 암호를 제공합니다.

a. App Manager에서 App Studio 앱의 화살표를 선택하고 보기를 선택합니다.

b. API(OAuth 설정 활성화) 섹션에서 소비자 세부 정보 관리를 선택합니다. 이렇게 하면 확인을 위해 입력해야 하는 확인 키에 대한 이메일이 전송될 수 있습니다.

c. 소비자 키(클라이언트 ID)와 소비자 보안 암호(클라이언트 보안 암호)를 기록해 둡니다.

17. App Studio로 돌아가서 다음 필드를 채워 커넥터를 구성하고 생성합니다.

18. 이름에 Salesforce 커넥터의 이름을 입력합니다.

19. 설명에 Salesforce 커넥터에 대한 설명을 입력합니다.

20. 기본 URL에 Salesforce 인스턴스의 기본 URL을 입력합니다. 호스트 *###* Salesforce 인스턴스 이름으로 `https://hostname.salesforce.com/services/data/v60.0`바꾸면 다음과 같습니다.

21. 인증 방법에서 OAuth 2.0이 선택되어 있는지 확인합니다.

22. OAuth 2.0 흐름에서 OAuth 인증 방법을 선택하고 관련 필드를 작성합니다.

- system-to-system 통합을 위해 자체적으로 작동하는 애플리케이션에서 사용할 클라이언트 자격 증명 흐름을 선택합니다.

a. 클라이언트 ID에 Salesforce에서 이전에 얻은 소비자 키를 입력합니다.

b. 클라이언트 보안 암호에 이전에 Salesforce에서 얻은 소비자 보안 암호를 입력합니다.

c. 액세스 토큰 URL에 OAuth 2.0 토큰 엔드포인트를 입력합니다. 호스트 *###* Salesforce 인스턴스 이름으로 `https://hostname/services/oauth2/token`바꾸면 다음과 같습니다. 자세한 내용은 Salesforce OAuth 엔드포인트 설명서를 참조하세요.

d. 연결 확인을 선택하여 인증 및 연결을 테스트합니다.

- 사용자를 대신하여 작동하는 애플리케이션에서 사용할 권한 부여 코드 흐름을 선택합니다.

  a. 클라이언트 ID에 Salesforce에서 이전에 얻은 소비자 키를 입력합니다.

  b. 클라이언트 보안 암호에 이전에 Salesforce에서 얻은 소비자 보안 암호를 입력합니다.

  c. 권한 부여 URL에 권한 부여 엔드포인트를 입력합니다. 호스트 *###* Salesforce 인스턴스 이름으로 `https://hostname/services/oauth2/authorize`바꾸면 다음과 같습니다. 자세한 내용은 Salesforce OAuth 엔드포인트 설명서를 참조하세요.

  d. 액세스 토큰 URL에 OAuth 2.0 토큰 엔드포인트를 입력합니다. 호스트 *###* Salesforce 인스턴스 이름으로 `https://hostname/services/oauth2/token`바꾸면 다음과 같습니다. 자세한 내용은 Salesforce OAuth 엔드포인트 설명서를 참조하세요.

23. 작업에서 커넥터가 지원할 Salesforce 작업을 선택합니다. 이 목록의 작업은 미리 정의되어 있으며 공통 객체에서 레코드 생성, 검색, 업데이트 또는 삭제와 같은 Salesforce 내의 일반적인 작업을 나타냅니다.
24. 생성(Create)을 선택합니다. 새로 생성된 커넥터가 커넥터 목록에 나타납니다.

커넥터 보기, 편집 및 삭제

기존 커넥터를 보거나 편집하거나 삭제하려면

1. 탐색 창의 관리 섹션에서 커넥터를 선택합니다. 각 커넥터에 대한 다음 세부 정보가 포함된 기존 커넥터 목록이 표시된 페이지로 이동합니다.

   - 이름: 생성 중에 제공된 커넥터의 이름입니다.
   - 설명: 생성 중에 제공된 커넥터에 대한 설명입니다.
   - 연결 대상: 커넥터가 App Studio에 연결하는 서비스입니다. API 값은 타사 서비스에 대한 연결을 나타냅니다.
   - 작성자: 커넥터를 생성한 사용자입니다.
   - 생성 날짜: 커넥터가 생성된 날짜입니다.

2. 커넥터에 대한 자세한 내용을 보거나 커넥터를 편집 또는 삭제하려면 다음 지침을 사용합니다.

   - 특정 커넥터에 대한 자세한 내용을 보려면 해당 커넥터에 대해 보기를 선택합니다.
   - 커넥터를 편집하려면 보기 옆의 드롭다운 메뉴를 선택하고 편집을 선택합니다.
   - 커넥터를 삭제하려면 보기 옆의 드롭다운 메뉴를 선택하고 삭제를 선택합니다.

App Studio 인스턴스 삭제

이 주제의 절차를 사용하여 App Studio 인스턴스를 삭제합니다. App Studio와 함께 사용할 리소스를 다른 서비스에서 생성한 경우 요금이 부과되지 않도록 필요에 따라 리소스를 검토하고 삭제합니다.

다음과 같은 이유로 App Studio 인스턴스를 삭제할 수 있습니다.

- 더 이상 App Studio를 사용하고 싶지 않습니다.
- 다른 AWS 리전에서 App Studio 인스턴스를 생성하려고 합니다. App Studio는 한 번에 한 리전의 인스턴스만 지원하므로 다른 인스턴스를 생성하려면 기존 인스턴스를 삭제해야 합니다.

> **Warning**
>
> App Studio 인스턴스를 삭제하면 애플리케이션 및 커넥터와 같은 모든 App Studio 리소스도 삭제됩니다. 인스턴스 삭제는 실행 취소할 수 없습니다.

App Studio 인스턴스를 삭제하려면

1. https://console.aws.amazon.com/appstudio/ App Studio 콘솔을 엽니다.
2. App Studio 인스턴스가 있는 리전을 선택합니다.
3. 탐색 창에서 인스턴스를 선택합니다.
4. 작업을 선택하여 추가 인스턴스 작업이 포함된 드롭다운을 엽니다.
5. App Studio 인스턴스 삭제를 선택합니다.
6. **`confirm`**를 입력한 다음 삭제를 선택합니다.
7. 인스턴스 삭제를 처리하는 데 시간이 걸릴 수 있습니다. 삭제되면 확인 이메일을 받게 됩니다. 이메일을 받으면 원하는 경우 다른 인스턴스를 생성할 수 있습니다.

Builder 설명서

다음 주제에는 App Studio에서 애플리케이션을 생성, 편집 및 게시하는 사용자에게 도움이 되는 정보가 포함되어 있습니다.

주제

- 자습서
- 생성형 AI를 사용하여 App Studio 앱 빌드
- 애플리케이션 생성, 편집 및 삭제
- 애플리케이션 미리 보기, 게시 및 공유
- 페이지 및 구성 요소: 앱의 사용자 인터페이스 빌드
- 자동화 및 작업: 앱의 비즈니스 로직 정의
- 개체 및 데이터 작업: 앱의 데이터 모델 구성
- 페이지 및 자동화 파라미터
- JavaScript를 사용하여 App Studio에서 표현식 작성
- 데이터 종속성 및 타이밍 고려 사항
- 여러 사용자로 앱 빌드
- 앱의 콘텐츠 보안 설정 보기 또는 업데이트

자습서

주제

- Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 구축
- Amazon Simple Storage Service와 구성 요소 및 자동화의 상호 작용
- App Studio 앱에서 Lambda 함수 호출

Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 구축

이 자습서에서는 Amazon Bedrock을 사용하여 최종 사용자의 텍스트 입력에 대한 간결한 요약을 제공하는 애플리케이션을 App Studio에서 빌드합니다. 애플리케이션에는 사용자가 요약하려는 텍스트를 입력할 수 있는 간단한 사용자 인터페이스가 포함되어 있습니다. 회의 기록, 기사 콘텐츠, 조사 결과 또는 기타 텍스트 정보일 수 있습니다. 사용자가 텍스트를 입력한 후 버튼을 눌러 Amazon Bedrock으로

텍스트를 전송할 수 있습니다. 그러면 Claude 3 Sonnet 모델을 사용하여 텍스트를 처리하고 요약된 버전을 반환합니다.

목차

- 사전 조건
- 1단계: IAM 역할 및 App Studio 커넥터 생성 및 구성
- 2단계: 애플리케이션 생성
- 3단계: 자동화 생성 및 구성
- 4단계: 페이지 및 구성 요소 생성
  - 기본 페이지 이름 바꾸기
  - 페이지에 구성 요소 추가
  - 페이지 구성 요소 구성
- 5단계: 애플리케이션을 테스트 환경에 게시
- (선택 사항) 정리

사전 조건

시작하기 전에 다음 사전 조건을 검토하고 완료하세요.

- AWS App Studio에 대한 액세스. 이 자습서에서는 커넥터를 생성하려면 관리자 역할이 있어야 합니다.
- 선택 사항: AWS App Studio 개념 및를 검토하여 중요한 App Studio 개념자습서: 빈 앱에서 빌드 시작을 숙지합니다.

1단계: IAM 역할 및 App Studio 커넥터 생성 및 구성

Amazon Bedrock 모델에 App Studio 액세스 권한을 제공하려면 다음을 수행해야 합니다.

1. 앱에서 사용하려는 Amazon Bedrock 모델을 활성화합니다. 이 자습서에서는 Claude 3 Sonnet을 사용하므로 해당 모델을 활성화해야 합니다.
2. Amazon Bedrock에 대한 적절한 권한이 있는 IAM 역할을 생성합니다.
3. 앱에서 사용할 IAM 역할을 사용하여 App Studio 커넥터를 생성합니다.

자세한 지침을 Amazon Bedrock에 연결 보려면 로 이동하여 단계를 따르고 커넥터를 생성한 후이 자습서로 돌아갑니다.

2단계: 애플리케이션 생성

다음 절차에 따라 App Studio에서 텍스트 요약기 앱에 빌드할 빈 앱을 생성합니다.

1. App Studio에 로그인합니다.
2. 빌더 허브로 이동하여 + 앱 생성을 선택합니다.
3. 처음부터 시작을 선택합니다.
4. 앱 이름 필드에와 같은 앱 이름을 입력합니다**`Text Summarizer`**.
5. 데이터 소스 또는 커넥터를 선택하라는 메시지가 표시되면이 자습서에서 건너뛰기를 선택합니다.
6. 다음을 선택하여 계속 진행합니다.
7. (선택 사항): App Studio에서 앱을 빌드하는 방법에 대한 간략한 개요를 보려면 비디오 자습서를 시청하세요.
8. 앱 편집을 선택하면 애플리케이션 스튜디오로 이동합니다.

3단계: 자동화 생성 및 구성

자동화에서 App Studio 앱의 로직과 동작을 정의합니다. 자동화는 작업, 다른 리소스의 작업에 데이터를 전달하는 데 사용되는 파라미터, 다른 자동화 또는 구성 요소에서 사용할 수 있는 출력이라고 하는 개별 단계로 구성됩니다. 이 단계에서는 다음을 사용하여 Amazon Bedrock과의 상호 작용을 처리하는 자동화를 생성합니다.

- 입력: 사용자로부터 자동화로 텍스트 입력을 전달하는 파라미터입니다.
- 작업: 텍스트 입력을 Amazon Bedrock으로 보내고 출력 텍스트 요약을 반환하는 하나의 GenAI 프롬프트 작업입니다.
- 출력: 앱에서 사용할 수 있는 Amazon Bedrock에서 처리된 요약으로 구성된 자동화 출력입니다.

Amazon Bedrock에 프롬프트를 보내고 요약을 처리하고 반환하는 자동화를 생성 및 구성하려면

1. 캔버스 상단의 자동화 탭을 선택합니다.
2. + 자동화 추가를 선택합니다.
3. 오른쪽 패널에서 속성을 선택합니다.

4. 연필 아이콘을 선택하여 자동화 이름을 업데이트합니다. **InvokeBedrock**를 입력하고 Enter 키를 누릅니다.
5. 다음 단계를 수행하여 사용자의 텍스트 프롬프트 입력을 요청에 사용할 자동화에 전달하는 데 사용할 파라미터를 Amazon Bedrock에 추가합니다.

   a. 캔버스의 파라미터 상자에서 + 추가를 선택합니다.
   b. 이름에 **input**를 입력합니다.
   c. 설명에와 같은 설명을 입력합니다**Text to be sent to Amazon Bedrock**.
   d. 유형에서 문자열을 선택합니다.
   e. 추가를 선택하여 파라미터를 생성합니다.
6. 다음 단계를 수행하여 GenAI 프롬프트 작업을 추가합니다.

   a. 오른쪽 패널에서 작업을 선택합니다.
   b. GenAI 프롬프트를 선택하여 작업을 추가합니다.
7. 다음 단계를 수행하여 작업을 구성합니다.

   a. 캔버스에서 작업을 선택하여 오른쪽 속성 메뉴를 엽니다.
   b. 연필 아이콘을 선택하고 이름을 입력한 다음 Enter 키를 눌러 작업의 이름을 **PromptBedrock**로 바꿉니다.
   c. 커넥터에서에서 생성된 커넥터를 선택합니다1단계: IAM 역할 및 App Studio 커넥터 생성 및 구성.
   d. 모델에서 프롬프트를 처리하는 데 사용할 Amazon Bedrock 모델을 선택합니다. 이 자습서에서는 Claude 3.5 Sonnet을 선택합니다.
   e. 사용자 프롬프트에를 입력합니다`{{params.input}}`. 이는 이전에 생성한 `input` 파라미터를 나타내며 앱 사용자의 텍스트 입력을 포함합니다.
   f. 시스템 프롬프트에 Amazon Bedrock으로 전송할 시스템 프롬프트 지침을 입력합니다. 이 자습서에서는 다음을 입력합니다.

      ```
      You are a highly efficient text summarizer. Provide a concise summary of the
       prompted text, capturing the key points and main ideas.
      ```

   g. 설정을 확장하려면 요청 설정을 선택하고 다음 필드를 업데이트합니다.

      - 온도에를 입력합니다`0`. tempearture는 출력의 무작위성 또는 창의성을 0~10의 척도로 결정합니다. 숫자가 높을수록 응답의 창의성이 높아집니다.
      - 최대 토큰에서 `4096`를 입력하여 응답 길이를 제한합니다.

8. 이 자동화의 출력은 요약된 텍스트이지만 기본적으로 자동화는 출력을 생성하지 않습니다. 다음 단계를 수행하여 자동화 출력을 생성하도록 자동화를 구성합니다.

   a. 왼쪽 탐색에서 InvokeBedrock 자동화를 선택합니다.

   b. 오른쪽 속성 메뉴의 출력에서 + 추가를 선택합니다.

   c. 출력에를 입력합니다**{{results.PromptBedrock.text}}**. 이 표현식은 `processResults` 작업의 내용을 반환합니다.

4단계: 페이지 및 구성 요소 생성

App Studio에서 각 페이지는 사용자가 상호 작용할 애플리케이션의 사용자 인터페이스(UI) 화면을 나타냅니다. 이러한 페이지 내에서 테이블, 양식, 버튼 등과 같은 다양한 구성 요소를 추가하여 원하는 레이아웃과 기능을 생성할 수 있습니다.

기본 페이지 이름 바꾸기

이 자습서의 텍스트 요약기 앱에는 한 페이지만 포함됩니다. 새로 생성된 애플리케이션은 기본 페이지와 함께 제공되므로 이름을 추가하는 대신 이름을 바꿉니다.

기본 페이지의 이름을 바꾸려면

1. 상단 표시줄 탐색 메뉴에서 페이지를 선택합니다.
2. 왼쪽 패널에서 Page1을 선택하고 오른쪽 패널에서 속성 패널을 선택합니다.
3. 연필 아이콘을 선택하고를 입력한 다음 Enter **TextSummarizationTool**키를 누릅니다.
4. 탐색 레이블에를 입력합니다**TextSummarizationTool**.

페이지에 구성 요소 추가

이 자습서에서는 텍스트 요약기 앱에 다음 구성 요소가 포함된 페이지가 하나 있습니다.

- 최종 사용자가 요약할 프롬프트를 입력하는 데 사용하는 텍스트 입력 구성 요소입니다.
- 프롬프트를 Amazon Bedrock으로 전송하는 데 사용되는 버튼 구성 요소입니다.
- Amazon Bedrock의 요약을 표시하는 텍스트 영역 구성 요소입니다.

사용자가 요약할 텍스트 프롬프트를 입력하는 데 사용할 페이지에 텍스트 입력 구성 요소를 추가합니다.

텍스트 입력 구성 요소를 추가하려면

1. 오른쪽 구성 요소 패널에서 텍스트 입력 구성 요소를 찾아 캔버스로 끕니다.
2. 캔버스에서 텍스트 입력을 선택하여 선택합니다.
3. 오른쪽 속성 패널에서 다음 설정을 업데이트합니다.
    a. 연필 아이콘을 선택하여 텍스트 입력의 이름을 로 변경합니다**inputPrompt**.
    b. 레이블에를 입력합니다**Prompt**.
    c. 자리 표시자에를 입력합니다**Enter text to be summarized**.

이제 사용자가 Amazon Bedrock에 프롬프트를 전송하도록 선택할 버튼 구성 요소를 추가합니다.

버튼 구성 요소를 추가하려면

1. 오른쪽 구성 요소 패널에서 버튼 구성 요소를 찾아 캔버스로 끕니다.
2. 캔버스에서 버튼을 선택하여 선택합니다.
3. 오른쪽 속성 패널에서 다음 설정을 업데이트합니다.
    a. 연필 아이콘을 선택하여 버튼의 이름을 로 바꿉니다**sendButton**.
    b. 버튼 레이블에를 입력합니다**Send**.

이제 Amazon Bedrock에서 반환한 요약을 표시할 텍스트 영역 구성 요소를 추가합니다.

텍스트 영역 구성 요소를 추가하려면

1. 오른쪽 구성 요소 패널에서 텍스트 영역 구성 요소를 찾아 캔버스로 끕니다.
2. 캔버스에서 텍스트 영역을 선택하여 선택합니다.
3. 오른쪽 속성 패널에서 다음 설정을 업데이트합니다.
    a. 연필 아이콘을 선택하여 버튼의 이름을 로 변경합니다**textSummary**.
    b. 레이블에를 입력합니다**Summary**.

페이지 구성 요소 구성

이제 앱에 구성 요소가 포함된 페이지가 포함되어 있으므로 다음 단계는 적절한 동작을 수행하도록 구성 요소를 구성하는 것입니다. 버튼과 같은 구성 요소가 상호 작용할 때 작업을 수행하도록 구성하려면

트리거를 추가해야 합니다. 이 자습서의 앱의 경우 sendButton 버튼에 트리거 2개를 추가하여 다음을 수행합니다.

- 첫 번째 트리거는 구성 textPrompt 요소의 텍스트를 Amazon Bedrock으로 전송하여 분석합니다.
- 두 번째 트리거는 Amazon Bedrock에서 반환된 요약을 textSummary 구성 요소에 표시합니다.

Amazon Bedrock으로 프롬프트를 보내는 트리거를 추가하려면

1. 캔버스에서 버튼을 선택하여 선택합니다.
2. 오른쪽 속성 패널의 트리거 섹션에서 + 추가를 선택합니다.
3. 자동화 호출을 선택합니다.
4. 생성된 InvokeAutomation1 트리거를 선택하여 구성합니다.
5. 작업 이름에를 입력합니다**invokeBedrockAutomation**.
6. 자동화 호출에서 이전에 생성된 InvokeBedrock 자동화를 선택합니다.
7. 파라미터 상자의 이전에 생성된 입력 파라미터에 inputPrompt 텍스트 입력 구성 요소의 콘텐츠를 **{{ui.inputPrompt.value}}**전달하는를 입력합니다.
8. 패널 상단의 왼쪽 화살표를 선택하여 구성 요소 속성 메뉴로 돌아갑니다.

이제 버튼을 클릭할 때 Amazon Bedrock에 요청을 보내도록 자동화를 호출하는 트리거를 구성했습니다. 다음 단계는 textSummary 구성 요소에 결과를 표시하는 두 번째 트리거를 구성하는 것입니다.

텍스트 영역 구성 요소에 Amazon Bedrock 결과를 표시하는 트리거를 추가하려면

1. 버튼의 오른쪽 속성 패널에서 트리거 섹션에서 + 추가를 선택합니다.
2. 구성 요소 작업 실행을 선택합니다.
3. 생성된 Runcomponentaction1 트리거를 선택하여 구성합니다.
4. 작업 이름에를 입력합니다**setTextSummary**.
5. 구성 요소에서 textSummary 구성 요소를 선택합니다.
6. 작업에서 값 설정을 선택합니다.
7. 값을 로 설정에서를 입력합니다**{{results.invokeBedrockAutomation}}**.

5단계: 애플리케이션을 테스트 환경에 게시

일반적으로 앱을 빌드하는 동안 앱을 미리 보고 모양을 확인하고 기능에 대한 초기 테스트를 수행하는 것이 좋습니다. 그러나 애플리케이션이 미리 보기 환경에서 외부 서비스와 상호 작용하지 않으므로 대신 테스트 환경에 앱을 게시하여 Amazon Bedrock으로부터 전송 요청 및 수신 응답을 테스트할 수 있습니다.

앱을 테스트 환경에 게시하려면

1. 앱 빌더의 오른쪽 상단 모서리에서 게시를 선택합니다.
2. 테스트 환경에 대한 버전 설명을 추가합니다.
3. SLA와 관련된 확인란을 검토하고 선택합니다.
4. 시작을 선택합니다. 게시에는 최대 15분이 걸릴 수 있습니다.
5. (선택 사항) 준비가 되면 공유를 선택하고 프롬프트에 따라 다른 사용자에게 액세스 권한을 부여할 수 있습니다. App Studio 앱 공유에 대한 자세한 내용은 섹션을 참조하세요게시된 애플리케이션 공유.

애플리케이션을 테스트한 후 게시를 다시 선택하여 애플리케이션을 프로덕션 환경으로 승격합니다. 프로덕션 환경의 앱은 공유될 때까지 최종 사용자가 사용할 수 없습니다. 다양한 애플리케이션 환경에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 환경.

(선택 사항) 정리

이제 자습서를 성공적으로 완료하고 Amazon Bedrock을 사용하여 App Studio에서 텍스트 요약 앱을 구축했습니다. 앱을 계속 사용하거나이 자습서에서 생성된 리소스를 정리할 수 있습니다. 다음 목록에는 정리할 리소스 목록이 포함되어 있습니다.

- App Studio에서 생성된 Amazon Bedrock 커넥터입니다. 자세한 내용은 커넥터 보기, 편집 및 삭제 단원을 참조하십시오.
- App Studio의 텍스트 요약기 앱입니다. 자세한 내용은 애플리케이션 삭제 단원을 참조하십시오.
- IAM 콘솔에서 생성된 IAM 역할입니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 역할 또는 인스턴스 프로파일 삭제를 참조하세요.
- Claude 3 Sonnet을 사용하도록 모델 액세스를 요청하고 액세스를 되돌리려면 Amazon Bedrock 사용 설명서의 Amazon Bedrock 파운데이션 모델에 대한 액세스 관리를 참조하세요.

Amazon Simple Storage Service와 구성 요소 및 자동화의 상호 작용

App Studio 앱에서 다양한 Amazon S3 작업을 호출할 수 있습니다. 예를 들어 간단한 관리자 패널을 생성하여 사용자 및 주문을 관리하고 Amazon S3에서 미디어를 표시할 수 있습니다. 간접 호출 작업을 사용하여 모든 Amazon S3 작업을 간접 호출 AWS할 수 있지만, Amazon S3 버킷 및 객체에서 공통 작업을 수행하기 위해 앱의 자동화에 추가할 수 있는 4개의 전용 Amazon S3 작업이 있습니다. 네 가지 작업과 해당 작업은 다음과 같습니다.

- Put Object: `Amazon S3 PutObject` 작업을 사용하여 Amazon S3 버킷에 객체를 추가합니다.
- 객체 가져오기: `Amazon S3 GetObject` 작업을 사용하여 Amazon S3 버킷에서 객체를 검색합니다.
- 객체 나열: `Amazon S3 ListObjects` 작업을 사용하여 Amazon S3 버킷의 객체를 나열합니다.
- 객체 삭제: `Amazon S3 DeleteObject` 작업을 사용하여 Amazon S3 버킷에서 객체를 삭제합니다.

작업 외에도 애플리케이션의 페이지에 추가할 수 있는 S3 업로드 구성 요소가 있습니다. 사용자는이 구성 요소를 사용하여 업로드할 파일을 선택하고 구성 요소가를 호출`Amazon S3 PutObject`하여 구성된 버킷 및 폴더에 파일을 업로드할 수 있습니다. 이 자습서에서는 독립 실행형 Put Object 자동화 작업 대신이 구성 요소를 사용합니다. (독립 실행형 작업은 업로드 전후에 수행할 추가 로직 또는 작업이 포함된 보다 복잡한 시나리오에서 사용해야 합니다.)

사전 조건

이 안내서에서는 다음 사전 조건을 완료했다고 가정합니다.

1. Amazon S3를 App Studio와 성공적으로 통합하기 위해 Amazon S3 버킷, IAM 역할 및 정책, Amazon S3 커넥터를 생성하고 구성했습니다. 커넥터를 생성하려면 관리자 역할이 있어야 합니다. 자세한 내용은 Amazon Simple Storage Service(Amazon S3)에 연결 단원을 참조하십시오.

빈 애플리케이션 생성

다음 단계를 수행하여이 가이드 전체에서 사용할 빈 애플리케이션을 생성합니다.

빈 애플리케이션을 생성하려면

1. 탐색 창에서 내 애플리케이션을 선택합니다.
2. + 앱 생성을 선택합니다.

3. 앱 생성 대화 상자에서 애플리케이션에 이름을 지정하고 처음부터 시작을 선택한 다음 다음을 선택합니다.
4. 기존 데이터에 연결 대화 상자에서 건너뛰기를 선택하여 애플리케이션을 생성합니다.
5. 구성 요소, 자동화 및 데이터를 사용하여 애플리케이션의 모양과 기능을 구성할 수 있는 새 앱의 캔버스로 이동하려면 앱 편집을 선택합니다.

페이지 생성

애플리케이션에서 3페이지를 생성하여 정보를 수집하거나 표시합니다.

페이지를 생성하려면

1. 필요한 경우 캔버스 상단의 페이지 탭을 선택합니다.
2. 왼쪽 탐색에는 앱으로 생성된 단일 페이지가 있습니다. + 추가를 두 번 선택하여 두 페이지를 더 생성합니다. 탐색 창에 총 3개의 페이지가 표시되어야 합니다.
3. 다음 단계를 수행하여 Page1 페이지의 이름을 업데이트합니다.
   a. 줄임표 아이콘을 선택하고 페이지 속성을 선택합니다.
   b. 오른쪽 속성 메뉴에서 연필 아이콘을 선택하여 이름을 편집합니다.
   c. **`FileList`**를 입력하고 Enter 키를 누릅니다.
4. 이전 단계를 반복하여 다음과 같이 두 번째 및 세 번째 페이지를 업데이트합니다.
   - Page2의 이름을 로 바꿉니다**`UploadFile`**.
   - Page3의 이름을 로 바꿉니다**`FailUpload`**.

이제 앱에는 왼쪽 페이지 패널에 표시된 FileList, UploadFile 및 FailUpload라는 세 페이지가 있어야 합니다.

다음으로 Amazon S3와 상호 작용하는 자동화를 생성하고 구성합니다.

자동화 생성 및 구성

Amazon S3와 상호 작용하는 애플리케이션의 자동화를 생성합니다. 다음 절차에 따라 다음 자동화를 생성합니다.

- 테이블 구성 요소를 채우는 데 사용되는 Amazon S3 버킷의 객체를 나열하는 getFiles 자동화입니다.

- 테이블 구성 요소에 삭제 버튼을 추가하는 데 사용되는 Amazon S3 버킷에서 객체를 삭제하는 deleteFile 자동화입니다.
- Amazon S3 버킷에서 객체를 가져와 표시하는 viewFile 자동화로, 테이블 구성 요소에서 선택한 단일 객체에 대한 세부 정보를 표시하는 데 사용됩니다.

getFiles 자동화 생성

지정된 Amazon S3 버킷의 파일을 나열하는 자동화를 생성합니다.

1. 캔버스 상단의 자동화 탭을 선택합니다.
2. + 자동화 추가를 선택합니다.
3. 오른쪽 패널에서 속성을 선택합니다.
4. 연필 아이콘을 선택하여 자동화 이름을 업데이트합니다. **getFiles**를 입력하고 Enter 키를 누릅니다.
5. 다음 단계를 수행하여 객체 나열 작업을 추가합니다.
   a. 오른쪽 패널에서 작업을 선택합니다.
   b. 객체 나열을 선택하여 작업을 추가합니다. 작업의 이름은 이어야 합니다`ListObjects1`.
6. 다음 단계를 수행하여 작업을 구성합니다.
   a. 캔버스에서 작업을 선택하여 오른쪽 속성 메뉴를 엽니다.
   b. 커넥터에서 사전 요구 사항에서 생성한 Amazon S3 커넥터를 선택합니다.
   c. 구성에 다음 텍스트를 입력하고 *bucket_name*을 사전 조건에서 생성한 버킷으로 바꿉니다.

```
{
  "Bucket": "bucket_name",
  "Prefix": ""
}
```

> **Note**
>
> `Prefix` 필드를 사용하여 지정된 문자열로 시작하는 객체에 대한 응답을 제한할 수 있습니다.

7. 이 자동화의 출력은 테이블 구성 요소를 Amazon S3 버킷의 객체로 채우는 데 사용됩니다. 그러나 자동화는 기본적으로 출력을 생성하지 않습니다. 다음 단계를 수행하여 자동화 출력을 생성하도록 자동화를 구성합니다.

   a. 왼쪽 탐색에서 getFiles 자동화를 선택합니다.

   b. 오른쪽 속성 메뉴의 자동화 출력에서 + 출력 추가를 선택합니다.

   c. 출력에를 입력합니다**{{results.ListObjects1.Contents}}**. 이 표현식은 작업의 내용을 반환하며 이제 테이블 구성 요소를 채우는 데 사용할 수 있습니다.

deleteFile 자동화 생성

지정된 Amazon S3 버킷에서 객체를 삭제하는 자동화를 생성합니다.

1. 왼쪽 자동화 패널에서 + 추가를 선택합니다.
2. + 자동화 추가를 선택합니다.
3. 오른쪽 패널에서 속성을 선택합니다.
4. 연필 아이콘을 선택하여 자동화 이름을 업데이트합니다. **deleteFile**를 입력하고 Enter 키를 누릅니다.
5. 다음 단계를 수행하여 자동화에 데이터를 전달하는 데 사용되는 자동화 파라미터를 추가합니다.

   a. 오른쪽 속성 메뉴의 자동화 파라미터에서 + 추가를 선택합니다.

   b. 연필 아이콘을 선택하여 자동화 파라미터를 편집합니다. 파라미터 이름을 로 업데이트**fileName**하고 Enter 키를 누릅니다.

6. 다음 단계를 수행하여 객체 삭제 작업을 추가합니다.

   a. 오른쪽 패널에서 작업을 선택합니다.

   b. 객체 삭제를 선택하여 작업을 추가합니다. 작업의 이름은 이어야 합니다`DeleteObject1`.

7. 다음 단계를 수행하여 작업을 구성합니다.

   a. 캔버스에서 작업을 선택하여 오른쪽 속성 메뉴를 엽니다.

   b. 커넥터에서 사전 요구 사항에서 생성한 Amazon S3 커넥터를 선택합니다.

   c. 구성에 다음 텍스트를 입력하고 *bucket_name*을 사전 조건에서 생성한 버킷으로 바꿉니다.

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
```

```
}
```

viewFile 자동화 생성

지정된 Amazon S3 버킷에서 단일 객체를 검색하는 자동화를 생성합니다. 나중에 객체를 표시하도록 파일 뷰어 구성 요소로이 자동화를 구성합니다.

1. 왼쪽 자동화 패널에서 + 추가를 선택합니다.
2. + 자동화 추가를 선택합니다.
3. 오른쪽 패널에서 속성을 선택합니다.
4. 연필 아이콘을 선택하여 자동화 이름을 업데이트합니다. **viewFile**를 입력하고 Enter 키를 누릅니다.
5. 다음 단계를 수행하여 자동화에 데이터를 전달하는 데 사용되는 자동화 파라미터를 추가합니다.
   a. 오른쪽 속성 메뉴의 자동화 파라미터에서 + 추가를 선택합니다.
   b. 연필 아이콘을 선택하여 자동화 파라미터를 편집합니다. 파라미터 이름을 로 업데이트**fileName**하고 Enter 키를 누릅니다.
6. 다음 단계를 수행하여 객체 가져오기 작업을 추가합니다.
   a. 오른쪽 패널에서 작업을 선택합니다.
   b. 객체 가져오기를 선택하여 작업을 추가합니다. 작업의 이름은 이어야 합니다`GetObject1`.
7. 다음 단계를 수행하여 작업을 구성합니다.
   a. 캔버스에서 작업을 선택하여 오른쪽 속성 메뉴를 엽니다.
   b. 커넥터에서 사전 요구 사항에서 생성한 Amazon S3 커넥터를 선택합니다.
   c. 구성에 다음 텍스트를 입력하고 *bucket_name*을 사전 조건에서 생성한 버킷으로 바꿉니다.

```
{
  "Bucket": "bucket_name",
  "Key": params.fileName
}
```

8. 기본적으로 자동화는 출력을 생성하지 않습니다. 다음 단계를 수행하여 자동화 출력을 생성하도록 자동화를 구성합니다.
   a. 왼쪽 탐색에서 viewFile 자동화를 선택합니다.
   b. 오른쪽 속성 메뉴의 자동화 출력에서 + 출력 추가를 선택합니다.

c. 출력에를 입력합니다**{{results.GetObject1.Body.transformToWebStream()}}**. 이 표현식은 작업의 내용을 반환합니다.

> **Note**
>
> 다음과 같은 S3 GetObject 방법으로의 응답을 읽을 수 있습니다.
>
> - transformToWebStream: 데이터를 검색하는 데 사용해야 하는 스트림을 반환합니다. 자동화 출력으로 사용되는 경우 자동화가 이를 처리하고 출력을 이미지 또는 PDF 뷰어 구성 요소의 데이터 소스로 사용할 수 있습니다. 와 같은 다른 작업에 대한 입력으로도 사용할 수 있습니다S3 PutObject.
> - transformToString: 자동화의 원시 데이터를 반환하며, 파일에 JSON 데이터와 같은 텍스트 콘텐츠가 포함된 경우 JavaScript 작업에 사용해야 합니다. 다음과 같이 대기해야 합니다. `await results.GetObject1.Body.transformToString();`
> - transformToByteArray: 부호 없는 8비트 정수의 배열을 반환합니다. 이 응답은 바이너리 데이터의 저장 및 조작을 허용하는 바이트 배열의 목적을 수행합니다. 다음과 같이 대기해야 합니다. `await results.GetObject1.Body.transformToByteArray();`

다음으로 이전에 생성한 페이지에 구성 요소를 추가하고 사용자가 앱을 사용하여 파일을 보고 삭제할 수 있도록 자동화로 구성해야 합니다.

페이지 구성 요소 추가 및 구성

이제 앱의 비즈니스 로직과 기능을 정의하는 자동화를 생성했으므로 구성 요소를 생성하고 둘 다 연결합니다.

FileList 페이지에 구성 요소 추가

이전에 생성한 FileList 페이지는 구성된 Amazon S3 버킷의 파일 목록과 목록에서 선택한 파일에 대한 자세한 정보를 표시하는 데 사용됩니다. 이렇게 하려면 다음을 수행합니다.

1. 테이블 구성 요소를 생성하여 파일 목록을 표시합니다. 이전에 생성한 getFiles 자동화의 출력으로 채워지도록 테이블의 행을 구성합니다.
2. PDF 뷰어 구성 요소를 생성하여 단일 PDF를 표시합니다. 버킷에서 파일을 가져오기 위해 이전에 생성한 viewFile 자동화를 사용하여 테이블에서 선택한 파일을 보도록 구성 요소를 구성합니다.

FileList 페이지에 구성 요소를 추가하려면

1. 캔버스 상단의 페이지 탭을 선택합니다.
2. 왼쪽 페이지 패널에서 FileList 페이지를 선택합니다.
3. 오른쪽 구성 요소 페이지에서 테이블 구성 요소를 찾아 캔버스 중앙으로 끕니다.
4. 페이지에 방금 추가한 테이블 구성 요소를 선택합니다.
5. 오른쪽 속성 메뉴에서 소스 드롭다운을 선택하고 자동화를 선택합니다.
6. 자동화 드롭다운을 선택하고 getFiles 자동화를 선택합니다. 테이블은 getFiles 자동화의 출력을 콘텐츠로 사용합니다.
7. 파일 이름으로 채울 열을 추가합니다.

    a. 오른쪽 속성 메뉴의 열 옆에 있는 + 추가를 선택합니다.

    b. 방금 추가된 Column1 열 오른쪽에 있는 화살표 아이콘을 선택합니다.

    c. 열 레이블에서 열의 이름을 로 바꿉니다**Filename**.

    d. 값에 **{{currentRow.Key}}**를 입력합니다.

    e. 패널 상단의 화살표 아이콘을 선택하여 기본 속성 패널로 돌아갑니다.
8. 테이블 작업을 추가하여 행에서 파일을 삭제합니다.

    a. 오른쪽 속성 메뉴의 작업 옆에 있는 + 추가를 선택합니다.

    b. 작업에서 버튼의 이름을 로 변경합니다**Delete**.

    c. 방금 이름이 변경된 삭제 작업 오른쪽에 있는 화살표 아이콘을 선택합니다.

    d. 클릭 시 + 작업 추가를 선택하고 자동화 호출을 선택합니다.

    e. 추가한 작업을 선택하여 구성합니다.

    f. 함수 이름에 **DeleteRecord**를 입력합니다.

    g. 자동화 호출에서를 선택합니다**deleteFile**.

    h. 파라미터 텍스트 상자에를 입력합니다**{{currentRow.Key}}**.

    i. 값에 **{{currentRow.Key}}**를 입력합니다.
9. 오른쪽 패널에서 구성 요소를 선택하여 구성 요소 메뉴를 봅니다. 파일을 표시하는 두 가지 옵션이 있습니다.

    - `.png`, `.jpeg`또는 `.jpg` 확장자가 있는 파일을 볼 수 있는 이미지 뷰어입니다.
    - PDF 파일을 볼 수 있는 PDF 뷰어 구성 요소입니다.

이 자습서에서는 PDF 뷰어 구성 요소를 추가하고 구성합니다.

10. PDF 뷰어 구성 요소를 추가합니다.

    a. 오른쪽 구성 요소 페이지에서 PDF 뷰어 구성 요소를 찾아 테이블 구성 요소 아래의 캔버스로 끕니다.

    b. 방금 추가된 PDF 뷰어 구성 요소를 선택합니다.

    c. 오른쪽 속성 메뉴에서 소스 드롭다운을 선택하고 자동화를 선택합니다.

    d. 자동화 드롭다운을 선택하고 viewFile 자동화를 선택합니다. 테이블은 viewFile 자동화의 출력을 콘텐츠로 사용합니다.

    e. 파라미터 텍스트 상자에를 입력합니다**{{ui.table1.selectedRow["Filename"]}}**.

    f. 오른쪽 패널에는 파일 이름 필드도 있습니다. 이 필드의 값은 PDF 뷰어 구성 요소의 헤더로 사용됩니다. 이전 단계와 동일한 텍스트를 입력합니다**{{ui.table1.selectedRow["Filename"]}}**.

UploadFile 페이지에 구성 요소 추가

UploadFile 페이지에는 구성된 Amazon S3 버킷에 파일을 선택하고 업로드하는 데 사용할 수 있는 파일 선택기가 포함되어 있습니다. 사용자가 파일을 선택하고 업로드하는 데 사용할 수 있는 S3 업로드 구성 요소를 페이지에 추가합니다.

1. 왼쪽 페이지 패널에서 UploadFile 페이지를 선택합니다.
2. 오른쪽 구성 요소 페이지에서 S3 업로드 구성 요소를 찾아 캔버스 중앙으로 끕니다.
3. 페이지에 방금 추가한 S3 업로드 구성 요소를 선택합니다.
4. 오른쪽 속성 메뉴에서 구성 요소를 구성합니다.

    a. 커넥터 드롭다운에서 사전 요구 사항에서 생성된 Amazon S3 커넥터를 선택합니다.

    b. 버킷에 Amazon S3 버킷의 이름을 입력합니다.

    c. 파일 이름에 **{{ui.s3Upload1.files[0]?.nameWithExtension}}**를 입력합니다.

    d. 최대 파일 크기**5**에 텍스트 상자에를 입력하고 드롭다운에서 **MB**가 선택되어 있는지 확인합니다.

    e. 트리거 섹션에서 다음 단계를 수행하여 업로드 성공 또는 실패 후 실행되는 작업을 추가합니다.

    업로드 성공 후 실행되는 작업을 추가하려면:

1. 성공 시 + 작업 추가를 선택하고 탐색을 선택합니다.
2. 추가한 작업을 선택하여 구성합니다.
3. 탐색 유형에서 페이지를 선택합니다.
4. 이동에서를 선택합니다**`FileList`**.
5. 패널 상단의 화살표 아이콘을 선택하여 기본 속성 패널로 돌아갑니다.

업로드 실패 후 실행되는 작업을 추가하려면:

1. 실패 시 + 작업 추가를 선택하고 탐색을 선택합니다.
2. 추가한 작업을 선택하여 구성합니다.
3. 탐색 유형에서 페이지를 선택합니다.
4. 이동에서를 선택합니다**`FailUpload`**.
5. 패널 상단의 화살표 아이콘을 선택하여 기본 속성 패널로 돌아갑니다.

FailUpload 페이지에 구성 요소 추가

FailUpload 페이지는 사용자에게 업로드 실패를 알리는 텍스트 상자가 포함된 간단한 페이지입니다.

1. 왼쪽 페이지 패널에서 FailUpload 페이지를 선택합니다.
2. 오른쪽 구성 요소 페이지에서 텍스트 구성 요소를 찾아 캔버스 중앙으로 끕니다.
3. 페이지에 방금 추가한 텍스트 구성 요소를 선택합니다.
4. 오른쪽 속성 메뉴의 값에를 입력합니다**`Failed to upload, try again`**.

앱 보안 설정 업데이트

App Studio의 모든 애플리케이션에는 외부 미디어 또는 리소스를 제한하는 데 사용할 수 있는 콘텐츠 보안 설정 또는 객체를 업로드할 수 있는 Amazon S3의 도메인이 있습니다. 기본 설정은 모든 도메인을 차단하는 것입니다. 애플리케이션에서 Amazon S3에 객체를 업로드하려면 객체를 업로드할 도메인을 허용하도록 설정을 업데이트해야 합니다.

도메인이 Amazon S3에 객체를 업로드하도록 허용하려면

1. 앱 설정 탭을 선택합니다.
2. 콘텐츠 보안 설정 탭을 선택합니다.

3. Connect 소스에서 모든 연결 허용을 선택합니다.
4. 저장을 선택합니다.

다음 단계: 테스트용 애플리케이션 미리 보기 및 게시

이제 애플리케이션을 테스트할 준비가 되었습니다. 애플리케이션 미리 보기 및 게시에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 미리 보기, 게시 및 공유.

App Studio 앱에서 Lambda 함수 호출

이 자습서에서는 App Studio를 Lambda에 연결하고 앱에서 Lambda 함수를 호출하는 방법을 보여줍니다.

사전 조건

이 안내서에서는 다음 사전 조건을 완료했다고 가정합니다.

1. App Studio 앱을 생성했습니다. 없는 경우 자습서에서 사용할 빈 앱을 생성할 수 있습니다. 자세한 내용은 애플리케이션 생성 단원을 참조하십시오.

> **Note**
>
> 이 자습서를 따르고 구성하는 방법을 배우는 데 Lambda 함수가 필요하지 않지만 앱을 올바르게 구성했는지 확인하는 데 Lambda 함수가 있으면 도움이 될 수 있습니다. 이 자습서에는 Lambda 함수 생성에 대한 정보가 포함되어 있지 않습니다. 자세한 내용은 AWS Lambda 개발자 안내서를 참조하세요.

Lambda 커넥터 생성

App Studio 앱에서 Lambda 함수를 사용하려면 커넥터를 사용하여 App Studio를 Lambda에 연결하여 함수에 대한 액세스를 제공해야 합니다. App Studio에서 커넥터를 생성하려면 관리자여야 합니다. Lambda 커넥터 생성 단계를 포함하여 Lambda 커넥터 생성에 대한 자세한 내용은 섹션을 참조하세요에 연결 AWS Lambda.

자동화 생성 및 구성

자동화는 애플리케이션의 로직을 정의하는 데 사용되며 작업으로 구성됩니다. 앱에서 Lambda 함수를 호출하려면 먼저 자동화에 Lambda 호출 작업을 추가하고 구성합니다. 다음 단계에 따라 자동화를 생성하고 Lambda 호출 작업을 추가합니다.

1. 앱을 편집하는 동안 자동화 탭을 선택합니다.
2. + 자동화 추가를 선택합니다.
3. 오른쪽 작업 메뉴에서 Lambda 호출을 선택하여 자동화에 단계를 추가합니다.
4. 캔버스에서 새 Lambda 단계를 선택하여 속성을 보고 구성합니다.
5. 오른쪽 속성 메뉴에서 다음 단계를 수행하여 단계를 구성합니다.

    a. 커넥터에서 App Studio를 Lambda 함수에 연결하기 위해 생성된 커넥터를 선택합니다.

    b. 함수 이름에 Lambda 함수의 이름을 입력합니다.

    c. 함수에서 Lambda 함수에 전달할 이벤트를 입력합니다. 몇 가지 일반적인 사용 사례 예제는 다음 목록에 나와 있습니다.

    - 파일 이름 또는 기타 문자열과 같은 자동화 파라미터의 값 전달: *varName*: params.*paramName*
    - 이전 작업의 결과 전달: *varName*: results.*actionName1*.data[0].*fieldName*
    - 루프 작업 내에 Lambda 호출 작업을 추가하는 경우 파라미터와 유사한 각 반복 항목에서 필드를 보낼 수 있습니다. varName: *currentItem*.*fieldName*

    d. 모의 출력 필드는 커넥터가 활성화되지 않은 미리 보기 중에 앱을 테스트하기 위한 모의 출력을 제공하는 데 사용할 수 있습니다.

자동화를 실행하도록 UI 요소 구성

이제 Lambda 함수를 호출하는 작업으로 구성된 자동화가 있으므로 자동화를 실행하도록 UI 요소를 구성할 수 있습니다. 이 자습서에서는 클릭 시 자동화를 실행하는 버튼을 생성합니다.

> **Tip**
>
> 자동화 호출 작업을 사용하여 다른 자동화에서 자동화를 실행할 수도 있습니다.

버튼에서 자동화를 실행하려면

1. 앱을 편집하는 동안 페이지 탭을 선택합니다.
2. 오른쪽 메뉴에서 버튼 구성 요소를 선택하여 페이지에 버튼을 추가합니다.
3. 새 버튼을 선택하여 구성합니다.
4. 오른쪽 속성 메뉴의 트리거에서 + 추가를 선택하고 자동화 호출을 선택합니다.
5. 새 자동화 호출 트리거를 선택하여 구성합니다.
6. 자동화 호출에서 Lambda 함수를 호출하는 자동화를 선택하고 자동화에 전송할 파라미터를 구성합니다.

이제 앱에서이 버튼을 선택하면 구성된 자동화가 실행됩니다.

다음 단계: 테스트용 애플리케이션 미리 보기 및 게시

이제 애플리케이션을 테스트할 준비가 되었습니다. 개발 환경에서 앱을 미리 볼 때 커넥터는 활성화되지 않으므로 커넥터로 연결할 수 있으므로 미리 보는 동안 자동화를 테스트할 수 없습니다 AWS Lambda. 커넥터에 의존하는 앱의 기능을 테스트하려면 앱을 테스트 환경에 게시해야 합니다. 애플리케이션 미리 보기 및 게시에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 미리 보기, 게시 및 공유.

생성형 AI를 사용하여 App Studio 앱 빌드

AWS App Studio는 개발을 가속화하고 일반적인 작업을 간소화하는 통합된 생성형 AI 기능을 제공합니다. 생성형 AI를 활용하여 앱, 데이터 모델, 샘플 데이터를 생성 및 편집하고 앱을 구축하는 동안 컨텍스트 지원을 받을 수도 있습니다.

앱 생성

가속화된 시작을 위해 AI 기반 자연어 프롬프트를 사용하여 전체 애플리케이션을 생성할 수 있습니다. 이 기능을 사용하면 원하는 앱 기능을 설명할 수 있으며 AI는 데이터 모델, 사용자 인터페이스, 워크플로 및 커넥터를 자동으로 빌드합니다. AI를 사용하여 앱을 생성하는 방법에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 생성.

앱 빌드 또는 편집

애플리케이션을 편집하는 동안 채팅을 사용하여 변경하려는 사항을 설명할 수 있으며 앱이 자동으로 업데이트됩니다. 기존 샘플 프롬프트 중에서 선택하거나 자체 프롬프트를 입력할 수 있습니다. 채팅을

사용하여 지원되는 구성 요소를 추가, 편집 및 제거하고 자동화 및 작업을 생성 및 구성할 수도 있습니다. AI를 사용하여 애플리케이션을 편집하거나 빌드하려면 다음 절차를 따르세요.

AI를 사용하여 앱을 편집하려면

1. 필요한 경우 앱을 편집하여 애플리케이션 스튜디오로 이동합니다.
2. (선택 사항) AI를 사용하여 편집하려는 페이지 또는 구성 요소를 선택합니다.
3. 왼쪽 하단 모서리에서 AI로 빌드를 선택하여 채팅을 엽니다.
4. 변경하려는 내용을 입력하거나 샘플 프롬프트에서 선택합니다.
5. 수행할 변경 사항을 검토합니다. 변경 사항을 적용하려면 확인을 선택합니다. 그렇지 않으면 다른 프롬프트를 입력합니다.
6. 변경 사항 요약을 검토합니다.

데이터 모델 생성

제공된 엔터티 이름을 기반으로 필드, 데이터 유형 및 데이터 작업을 사용하여 엔터티를 자동으로 생성할 수 있습니다. GenAi를 사용하여 엔터티를 생성하는 등 엔터티를 생성하는 방법에 대한 자세한 내용은 섹션을 참조하세요App Studio 앱에서 개체 생성.

다음과 같은 방법으로 기존 개체를 업데이트할 수도 있습니다.

- 엔터티에 필드를 더 추가합니다. 자세한 내용은 개체 필드 추가, 편집 또는 삭제 단원을 참조하십시오.
- 개체에 데이터 작업을 추가합니다. 자세한 내용은 데이터 작업 생성 단원을 참조하십시오.

샘플 데이터 생성

개체의 필드를 기반으로 개체에 대한 샘플 데이터를 생성할 수 있습니다. 이는 외부 데이터 소스를 연결하거나 외부 데이터 소스와 통신하지 않는 개발 환경에서 애플리케이션을 테스트하기 전에 애플리케이션을 테스트하는 데 유용합니다. 자세한 내용은 샘플 데이터 추가 또는 삭제 단원을 참조하십시오.

앱을 테스트 또는 프로덕션에 게시하면 해당 환경에서 라이브 데이터 소스와 커넥터가 사용됩니다.

AWS 서비스에 대한 작업 구성

Amazon Simple Email Service와 같은 AWS 서비스와 통합할 때 AI를 사용하여 선택한 서비스를 기반으로 미리 채워진 필드가 있는 예제 구성을 생성할 수 있습니다. 시도하려면 AWS 자동화 호출 작업의

속성 메뉴에서 양면 화살표를 선택하여 구성 필드를 확장합니다. 그런 다음 샘플 구성 생성을 선택합니다.

응답 모의

AWS 서비스 작업에 대해 모의 응답을 생성할 수 있습니다. 이는 외부 데이터 소스와 통신하지 않는 개발 환경에서 애플리케이션을 테스트하는 데 유용합니다.

빌드 중 AI에 도움 요청

애플리케이션 스튜디오 내에는 지원되는 리소스 또는 속성에 대한 도움 요청 버튼이 있습니다. 이를 사용하여 현재 보기 또는 선택한 구성 요소와 관련된 컨텍스트 제안, 설명서 및 지침을 얻을 수 있습니다. App Studio, 앱 구축 모범 사례 또는 특정 애플리케이션 사용 사례에 대한 일반적인 질문을 통해 맞춤형 정보와 권장 사항을 받으세요.

애플리케이션 생성, 편집 및 삭제

목차

- 애플리케이션 생성
- 애플리케이션 가져오기
  - App Studio에서 제공하는 가져오기 가능한 앱
- 애플리케이션 복제
- 애플리케이션 편집 또는 빌드
- 이전에 게시한 앱 버전 편집
- 애플리케이션 이름 바꾸기
- 애플리케이션 삭제

애플리케이션 생성

다음 절차에 따라 App Studio에서 애플리케이션을 생성합니다.

애플리케이션을 생성하는 방법

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택하여 애플리케이션 목록으로 이동합니다.
2. + 앱 생성을 선택합니다.
3. 앱 생성 대화 상자에서 애플리케이션에 이름을 지정하고 다음 앱 생성 방법 중 하나를 선택합니다.

- AI를 사용하여 앱 생성: 자연어로 앱을 설명하고 AI가 앱과 해당 리소스를 생성하도록 하려면 이 옵션을 선택합니다.
- 처음부터 시작: 빈 앱에서 빌드를 시작하려면이 옵션을 선택합니다.

4. 다음을 선택합니다.
5. AI를 사용하여 앱 생성을 선택한 경우:

   a. 기존 데이터에 연결 대화 상자에서 App Studio에 데이터 소스에 대한 액세스를 제공하는 커넥터를 선택한 다음 Tablse를 선택하고 다음을 선택하여 기존 데이터 소스를 앱에 추가합니다. 여기에 데이터 소스를 추가하면 AI가 최적화된 앱을 생성하는 데 도움이 됩니다. 건너뛰기를 선택하여이 단계를 건너뛰고 나중에 데이터 소스를 추가할 수 있습니다.

   b. 짧은 지연(몇 분) 후 AI를 사용하여 앱 생성 페이지로 이동합니다. 여기서 생성하려는 앱을 설명할 수 있습니다.

   c. 채팅에서 앱 설명을 시작하거나 제공된 샘플 프롬프트를 선택하고 사용자 지정할 수 있습니다.

   d. 프롬프트를 분석한 후 앱 요구 사항 및 개요를 검토합니다. 채팅을 사용하여 변경 사항을 요청하거나 다시 시작을 선택하여 빈 프롬프트에서 시작합니다.

   e. 준비가 되면 앱 생성을 선택합니다.

   f. 앱이 생성되면 앱 미리 보기를 선택하여 다른 탭에서 앱을 미리 봅니다. 편집을 시작할 준비가 되면 앱 편집을 선택할 수 있습니다. 애플리케이션의 페이지, 자동화 및 데이터를 탐색하여 익숙해지세요. 하단 디버그 패널에서 오류 또는 경고를 검토합니다. AI를 사용하여 앱을 생성하는 방법에 대한 자세한 내용은 섹션을 참조하세요자습서: AI를 사용하여 앱 생성. App Studio의 빌드 작동 방식에 대한 일반적인 정보는 섹션을 참조하세요AWS App Studio 작동 방식.

6. 처음부터 시작을 선택한 경우:

   a. 기존 데이터에 연결 대화 상자에서 App Studio에 데이터 소스에 대한 액세스 권한을 제공하는 커넥터를 선택하고 Tablse를 선택한 후 다음을 선택하여 기존 데이터 소스를 앱에 추가합니다. 건너뛰기를 선택하여이 단계를 건너뛰고 나중에 데이터 소스를 추가할 수 있습니다.

   b. 앱이 생성되면 앱 편집을 선택하여 앱 편집을 시작합니다. 빈 앱에서 빌드하는 방법에 대한 자세한 내용은 섹션을 참조하세요자습서: 빈 앱에서 빌드 시작. App Studio의 빌드 작동 방식에 대한 일반적인 정보는 섹션을 참조하세요AWS App Studio 작동 방식.

애플리케이션 가져오기

내보낸 애플리케이션의 사본을 App Studio 인스턴스로 가져올 수 있습니다. 다른 App Studio 인스턴스에서 내보낸 앱 또는 App Studio에서 제공하는 카탈로그에서 앱을 가져올 수 있습니다. App Studio 앱

카탈로그에서 앱을 가져오면 유사한 기능으로 앱을 시작하거나 가져온 앱을 탐색하여 App Studio에서 앱 빌드에 대해 알아볼 수 있습니다.

앱을 인스턴스로 가져오면 원본 앱의 사본이 인스턴스에 생성됩니다. 새 앱이 생성되면 앱의 개발 환경으로 이동하여 앱을 미리 보고 앱의 기능을 찾아볼 수 있습니다.

> **Warning**
>
> 앱을 가져올 때 애플리케이션에서 모든 로직을 가져오므로 원치 않거나 예기치 않은 동작이 발생할 수 있습니다. 예를 들어 애플리케이션에 연결하는 데이터베이스에서 데이터를 삭제하는 파괴적인 쿼리가 있을 수 있습니다. 프로덕션 데이터를 연결하기 전에 애플리케이션과 해당 구성을 철저히 검토하고 비프로덕션 자산에서 테스트하는 것이 좋습니다.

애플리케이션을 가져오려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택하여 애플리케이션 목록으로 이동합니다.
2. + 앱 생성 옆의 드롭다운 화살표를 선택합니다.
3. 앱 가져오기를 선택합니다.
4. 앱 가져오기 대화 상자의 가져오기 코드에 가져오려는 애플리케이션의 가져오기 코드를 입력합니다. 앱 설명 및 가져오기 코드를 포함하여 가져올 수 있는 App Studio에서 제공하는 앱 목록은 섹션을 참조하세요App Studio에서 제공하는 가져오기 가능한 앱.
5. 가져오기를 선택하여 앱을 가져오고 가져온 앱의 개발 환경으로 이동하여 앱을 보거나 편집합니다. App Studio에서 앱을 빌드하는 방법에 대한 자세한 내용은 섹션을 참조하세요. AWS App Studio 작동 방식

App Studio에서 제공하는 가져오기 가능한 앱

App Studio는 앱 빌드에 대해 알아볼 수 있도록 인스턴스로 가져올 수 있는 앱을 제공합니다. App Studio에서 특정 앱 기능이 구현되는 방식을 확인하려면 애플리케이션을 미리 본 다음 개발 환경에서 해당 구성을 찾아볼 수 있습니다.

다음 표에는 애플리케이션 목록, 가져오기 코드 및 앱에 대한 간략한 설명이 나와 있습니다. 각 앱에는 앱을 가져온 후 볼 수 있는 앱에 대한 정보가 포함된 `README` 페이지가 포함되어 있습니다.

앱 이름	설명	코드 가져오기
Swag 요청 설문 조사	직원이 브랜드 회사 상품을 주문할 수 있도록 설계된 내부 스웨그 요청 애플리케이션입니다. 직원은 항목을 선택하고 크기를 지정하고 간단한 양식을 통해 요청을 제출할 수 있습니다. 이 애플리케이션은 내장 스토리지를 통해 모든 데이터를 처리하므로 외부 연결이 필요하지 않습니다.	Swag 요청 Survey/ec4f5faf-e2f8-42ee-ab8d-6723d8ca21b2
스프린트 추적	팀이 소프트웨어 개발 작업을 구성하고 추적하는 데 사용할 수 있는 스프린트 관리 애플리케이션입니다. 사용자는 전용 스프린트, 트랙 및 작업 보기를 통해 스프린트를 생성하고, 작업을 추가하고, 작업을 할당하고, 진행 상황을 모니터링할 수 있습니다. 이 애플리케이션은 내장 스토리지를 통해 모든 데이터를 처리하므로 외부 연결이 필요하지 않습니다.	스프린트 Tracking/8f31e160-771f-48d7-87b0-374e285e2fbc
Amazon Review 감정 트래커	이 애플리케이션은 제품 리뷰에서 감정 점수를 생성하여 기업이 고객 만족도를 이해하는 데 도움이 되는 고객 피드백 분석 도구입니다. 애플리케이션에는 빠른 테스트를 위한 샘플 데이터 생성 유틸리티가 포함되어 있으며, 기본 제공 스토리지 시스템 내의 다른 모든 데이터를 유지하면서 AI 기반 인사	Amazon Review Sentiment Tracker/60f0dae4-f8e2-4c20-9583-fa456f5ebfab

앱 이름	설명	코드 가져오기
	이트를 위한 Amazon Bedrock 커넥터가 필요합니다.	
인보이스 및 수신 처리	이 수신 처리 애플리케이션은 수동 데이터 입력을 자동화하고 문서 승인 워크플로를 간소화하여 시간을 절약하고 오류를 줄입니다. 솔루션에는 Amazon Textract, Amazon S3 및 Amazon SES 커넥터가 필요합니다. Amazon Textract를 사용하여 Amazon S3에 저장된 수신에서 데이터를 분석하고 추출한 다음 Amazon SES를 사용하여 추출된 정보를 처리하고 승인자에게 이메일로 보냅니다.	인보이스 및 수신 Processing/98bde3ae-e454-4b18-a1e6-6f23e8b2a4f1
검사 및 인벤토리 감사	창고 검사 및 장비 추적을 관리하기 위한 애플리케이션입니다. 사용자는 룸 위치별로 합격/불합격 장비 평가를 수행하고, 인벤토리 수준을 모니터링하고, 검사 기록을 볼 수 있습니다. 애플리케이션은 시설 검사 및 장비 상태를 모두 추적하기 위한 중앙 집중식 시스템을 제공합니다. 이 애플리케이션은 내장 스토리지를 통해 모든 데이터를 처리하므로 외부 연결이 필요하지 않습니다.	검사 및 인벤토리 Audit/cf570a06-1c5e-4dd7-9ea8-5c04723d687f

앱 이름	설명	코드 가져오기
제품 채택 트래커	고객 피드백, 기능 요청 및 고객 회의 메모를 중앙 집중화하는 제품 개발을 관리하기 위한 포괄적인 애플리케이션입니다. 팀은 고객 상호 작용을 추적하고, 요구 사항을 구성하고, 분기별 로드맵 계획을 위한 AI 기반 보고서를 생성할 수 있습니다. 애플리케이션에는 샘플 데이터 유틸리티가 포함되어 있으며 AI 인사이트를 위해 Amazon Bedrock을, 데이터 관리를 위해 Amazon Aurora PostgreSQL을 활용합니다.	제품 채택 Tracker/9b3a4437-bb50-467f-ae9e-d108776b7ca1
빠른 임베딩	사용자가 기본 데이터로 작업하는 동안 분석을 볼 수 있게 해주는 데모 애플리케이션입니다. 앱에는 App Studio에 Amazon Quick 대시보드를 임베딩하는 두 가지 방법, 즉 등록된 사용자 및 익명 사용자를 위한 API 기반 접근 방식(Quick 커넥터 필요)과 퍼블릭 대시보드를 위한 iFrame 통합이 포함되어 있습니다.	Quicksight Embedding /0cdc15fc-ca8b-41b7-869e-ed13c9072bc8

앱 이름	설명	코드 가져오기
주방 싱크	고급 App Studio 개발 팁과 모범 사례를 보여주는 참조 애플리케이션입니다. 빌더가 자체 애플리케이션에서 연구하고 구현할 수 있는 상태 관리, CSV 데이터 처리, 브라우저 API 통합 및 UI 패턴의 실제 예제가 포함되어 있습니다. 외부 연결이 필요한 예제는 없습니다.	App Studio Kitchen Sink/1cfe6b2f-544c-4611-b82c-80eadc76a0c8

애플리케이션 복제

애플리케이션 소유자와 공동 소유자는 앱을 복제하여 정확한 앱 사본을 생성할 수 있습니다. 테스트 목적으로 현재 상태를 유지하거나 복제된 앱을 스타터로 사용하여 새 앱을 생성하려는 경우 앱 복제가 유용합니다.

애플리케이션을 복제하려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 액세스 권한이 있는 애플리케이션 목록이 표시된 페이지로 이동합니다.
2. 복제하려는 애플리케이션의 작업 열에서 드롭다운을 선택합니다.
3. Duplicate(복제)를 선택합니다. 중복 옵션을 사용할 수 없는 경우 애플리케이션의 소유자 또는 공동 소유자가 아닐 수 있습니다.
4. 선택적으로 복제된 앱의 이름을 입력합니다. 기본 이름은 *Current_App_Name COPY*입니다.
5. Duplicate(복제)를 선택합니다.

애플리케이션 편집 또는 빌드

다음 절차에 따라 App Studio에서 애플리케이션을 편집합니다.

애플리케이션을 편집(구축)하려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 액세스 권한이 있는 애플리케이션 목록이 표시된 페이지로 이동합니다.

2. 편집하려는 애플리케이션의 작업 열에서 편집을 선택합니다. 그러면 구성 요소, 자동화 및 데이터를 사용하여 애플리케이션의 모양과 기능을 구성할 수 있는 개발 환경으로 이동합니다. 애플리케이션 빌드에 대한 자세한 내용은 섹션을 참조하세요AWS App Studio 시작하기.

이전에 게시한 앱 버전 편집

다음 절차에 따라 이전에 게시한 App Studio 애플리케이션 버전을 편집합니다. 이전에 게시한 버전을 편집하도록 선택한 후 개발 환경에서 앱을 편집하거나 테스트 및 프로덕션에 게시할 수 있습니다.

> **Warning**
>
> 이전에 게시된 버전은 개발 환경에서 진행 중인 앱 버전을 대체합니다. 앱에 대한 게시되지 않은 변경 사항은 모두 손실됩니다.

이전에 게시한 버전을 편집하면 원치 않는 변경 사항이나 사용자의 애플리케이션을 손상시키는 변경 사항을 실수로 게시하고 이전 앱 버전에서 추가 빌드 또는 편집하려는 경우에 유용합니다.

> **Note**
>
> 게시된 앱과 관련된 문제를 감지하고 이전에 작동하던 버전을 즉시 게시해야 하거나 이전 버전을 게시하고 개발 환경에서 앱에 대한 최신 업데이트를 유지하려는 경우 대신 앱을 롤백해야 합니다. 자세한 내용은 이전에 게시된 버전으로 롤백 단원을 참조하십시오.

이전에 게시한 앱 버전을 편집하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 게시 버튼 옆의 드롭다운 화살표를 선택한 다음 센터 게시를 선택합니다.
3. 이전에 게시된 애플리케이션 버전 목록을 보려면 버전 기록을 선택합니다.
4. 편집하려는 버전을 찾아 편집을 선택합니다.
5. 정보를 검토하고 되돌리기를 선택합니다.
6. 편집하도록 선택한 버전은 이제 개발 환경의 현재 버전입니다. 이를 변경하거나 게시를 선택하여 테스트 환경에 그대로 게시할 수 있습니다. 테스트에 게시된 후에는 원하는 경우 프로덕션 환경에 다시 게시할 수 있습니다.

애플리케이션 이름 바꾸기

다음 절차에 따라 App Studio에서 애플리케이션의 이름을 바꿉니다. 앱을 빌드하는 동안 애플리케이션 목록 또는 개발 환경에서 애플리케이션의 이름을 바꿀 수 있습니다.

애플리케이션 이름을 바꾸려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 액세스 권한이 있는 애플리케이션 목록이 표시된 페이지로 이동합니다.
2. 편집하는 동안이 목록 또는 개발 환경에서 애플리케이션의 이름을 바꿀 수 있습니다.
   - 이 목록에서 이름을 바꾸려면:
     a. 이름을 바꾸려는 애플리케이션의 작업 열에서 드롭다운을 선택한 다음 이름 바꾸기를 선택합니다.
     b. 애플리케이션에 새 이름을 지정하고 이름 바꾸기를 선택합니다.
   - 개발 환경에서 이름을 바꾸려면:
     a. 편집하려는 애플리케이션의 작업 열에서 편집을 선택합니다.
     b. 개발 환경에서 애플리케이션 이름을 선택하고 업데이트한 다음 Enter 키를 누르거나 텍스트 필드 밖으로 이동하여 변경 사항을 저장합니다.

애플리케이션 삭제

다음 절차에 따라 App Studio에서 애플리케이션을 삭제합니다.

애플리케이션을 삭제하려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 액세스 권한이 있는 애플리케이션 목록이 표시된 페이지로 이동합니다.
2. 삭제할 애플리케이션의 작업 열에서 드롭다운을 선택합니다.
3. 삭제를 선택합니다.
4. 애플리케이션 삭제 대화 상자에서 애플리케이션 삭제에 대한 정보를 주의 깊게 검토합니다. 애플리케이션을 삭제하려면 삭제를 선택합니다.

애플리케이션 미리 보기, 게시 및 공유

주제

- 애플리케이션 미리 보기
- 애플리케이션 게시
- 게시된 애플리케이션 공유
- 이전에 게시된 버전으로 롤백
- 애플리케이션 내보내기

애플리케이션 미리 보기

App Studio에서 애플리케이션을 미리 보고 사용자에게 표시되는 방식을 확인하고, 애플리케이션을 사용하고 디버그 패널에서 로그를 확인하여 기능을 테스트할 수 있습니다.

애플리케이션 미리 보기 환경은 라이브 데이터 표시 또는 데이터 소스와 같은 커넥터를 사용한 외부 리소스와의 연결을 지원하지 않습니다. 미리 보기 환경에서 기능을 테스트하려면 자동화에서 모의 출력을 사용하고 개체에서 샘플 데이터를 사용할 수 있습니다. 실시간 데이터로 애플리케이션을 보려면 앱을 게시해야 합니다. 자세한 내용은 애플리케이션 게시 단원을 참조하십시오.

미리 보기 또는 개발 환경은 다른 환경에 게시된 애플리케이션을 업데이트하지 않습니다. 애플리케이션이 게시되지 않은 경우 사용자는 게시되고 공유될 때까지 애플리케이션에 액세스할 수 없습니다. 애플리케이션이 이미 게시되고 공유된 경우에도 사용자는 미리 보기 환경에서 사용되는 버전이 아니라 게시된 버전에 계속 액세스할 수 있습니다.

애플리케이션을 미리 보려면

1. 필요한 경우 미리 보려는 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
   a. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다.
   b. 애플리케이션에 대해 편집을 선택합니다.
2. 미리 보기를 선택하여 애플리케이션의 미리 보기 환경을 엽니다.
3. (선택 사항) 화면 하단 근처에서 헤더를 선택하여 디버그 패널을 확장합니다. 로그 필터링 섹션에서 메시지 유형을 선택하여 메시지 유형별로 패널을 필터링할 수 있습니다. 콘솔 지우기를 선택하여 패널의 로그를 지울 수 있습니다.
4. 미리 보기 환경에서 페이지를 탐색하고, 구성 요소를 사용하고, 버튼을 선택하여 데이터를 전송하는 자동화를 시작하여 애플리케이션을 테스트할 수 있습니다. 미리 보기 환경은 라이브 데이터 또

는 외부 소스에 대한 연결을 지원하지 않으므로 디버그 패널에서 전송 중인 데이터의 예를 볼 수 있습니다.

애플리케이션 게시

애플리케이션 생성 및 구성을 마치면 다음 단계는 데이터 전송을 테스트하거나 최종 사용자와 공유하기 위해 애플리케이션을 게시하는 것입니다. App Studio의 애플리케이션 게시를 이해하려면 사용 가능한 환경을 이해하는 것이 중요합니다. App Studio는 다음 목록에 설명된 세 가지 개별 환경을 제공합니다.

1. 개발: 애플리케이션을 빌드하고 미리 보는 위치입니다. 최신 버전의 애플리케이션이 자동으로 호스팅되므로 개발 환경에 게시할 필요가 없습니다. 이 환경에서는 라이브 데이터나 타사 서비스 또는 리소스를 사용할 수 없습니다.
2. 테스트: 애플리케이션에 대한 포괄적인 테스트를 수행할 수 있는 곳입니다. 테스트 환경에서는 다른 서비스에 연결하고, 데이터를 전송하고, 다른 서비스에서 데이터를 수신할 수 있습니다.
3. 프로덕션: 최종 사용자 소비를 위한 실시간 운영 환경입니다.

모든 앱 빌드는 개발 환경에서 이루어집니다. 그런 다음 테스트 환경에 게시하여 최종 사용자에게 액세스 URL을 제공하여 다른 서비스 간의 데이터 전송과 사용자 수락 테스트(UAT)를 테스트합니다. 그런 다음 앱을 프로덕션 환경에 게시하여 사용자와 공유하기 전에 최종 테스트를 수행합니다. 애플리케이션 환경에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 환경.

애플리케이션을 게시하면 공유될 때까지 사용자가 애플리케이션을 사용할 수 없습니다. 이를 통해 사용자가 애플리케이션에 액세스하기 전에 테스트 및 프로덕션 환경에서 애플리케이션을 사용하고 테스트할 수 있습니다. 이전에 게시되고 공유된 애플리케이션을 프로덕션에 게시하면 사용자가 사용할 수 있는 버전이 업데이트됩니다.

애플리케이션 게시

다음 절차에 따라 테스트 또는 프로덕션 환경에 App Studio 애플리케이션을 게시합니다.

애플리케이션을 테스트 또는 프로덕션 환경에 게시하려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 액세스 권한이 있는 애플리케이션 목록이 표시된 페이지로 이동합니다.
2. 게시하려는 애플리케이션에 대해 편집을 선택합니다.
3. 오른쪽 상단 모서리에서 게시를 선택합니다.

4. 업데이트 게시 대화 상자에서:

   a. 애플리케이션 게시에 대한 정보를 검토합니다.

   b. (선택 사항) 버전 설명에이 애플리케이션 버전에 대한 설명을 포함합니다.

   c. 상자를 선택하여 환경에 대한 정보를 확인합니다.

   d. 시작을 선택합니다. 라이브 환경에서 애플리케이션을 업데이트하는 데 최대 15분이 걸릴 수 있습니다.

5. 테스트 또는 프로덕션 환경에서 애플리케이션을 보는 방법에 대한 자세한 내용은 섹션을 참조하세요게시된 애플리케이션 보기.

> **Note**
>
> 테스트 또는 프로덕션 환경에서 애플리케이션을 사용하면 커넥터와 연결된 데이터 소스 테이블에 레코드를 생성하는 등 실시간 데이터가 전송됩니다.

공유된 적이 없는 게시된 애플리케이션은 사용자 또는 다른 빌더가 사용할 수 없습니다. 사용자가 애플리케이션을 사용할 수 있도록 하려면 게시 후 애플리케이션을 공유해야 합니다. 자세한 내용은 게시된 애플리케이션 공유 단원을 참조하십시오.

게시된 애플리케이션 보기

테스트 및 프로덕션 환경에 게시된 애플리케이션을 보고 최종 사용자 또는 다른 빌더와 공유하기 전에 애플리케이션을 테스트할 수 있습니다.

테스트 또는 프로덕션 환경에서 게시된 애플리케이션을 보려면

1. 필요한 경우 미리 보려는 애플리케이션의 애플리케이션 스튜디오로 이동합니다.

   a. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다.

   b. 애플리케이션에 대해 편집을 선택합니다.

2. 오른쪽 상단 모서리의 게시 옆에 있는 드롭다운 화살표를 선택하고 센터 게시를 선택합니다.

3. 게시 센터에서 애플리케이션이 게시되는 환경을 볼 수 있습니다. 애플리케이션이 테스트 또는 프로덕션 환경에 게시된 경우 각 환경의 URL 링크를 사용하여 앱을 볼 수 있습니다.

> **Note**
>
> 테스트 또는 프로덕션 환경에서 애플리케이션을 사용하면 커넥터와 연결된 데이터 소스 테이블에 레코드를 생성하는 등 실시간 데이터가 전송됩니다.

애플리케이션 환경

AWS App Studio는 개발, 테스트 및 프로덕션이라는 세 가지 개별 환경을 통해 애플리케이션 수명 주기 관리(ALM) 기능을 제공합니다. 이를 통해 전체 앱 수명 주기에서 별도의 환경 유지 관리, 버전 관리, 공유 및 모니터링과 같은 모범 사례를 보다 쉽게 수행할 수 있습니다.

개발 환경

개발 환경은 애플리케이션 스튜디오 및 샘플 데이터를 사용하여 라이브 데이터 소스 또는 서비스에 연결하지 않고도 앱을 빌드할 수 있는 격리된 샌드박스입니다. 개발 환경에서 앱을 미리 보고 프로덕션 데이터를 손상시키지 않고 앱을 보고 테스트할 수 있습니다.

앱이 개발 환경의 다른 서비스에 연결되지는 않지만 라이브 데이터 커넥터 및 자동화를 모방하도록 앱의 다양한 리소스를 구성할 수 있습니다.

개발 환경의 애플리케이션 스튜디오 하단에는 빌드 시 앱을 검사하고 디버깅하는 데 도움이 되는 오류와 경고가 포함된 축소 가능한 디버그 패널이 있습니다. 앱 문제 해결 및 디버깅에 대한 자세한 내용은 섹션을 참조하세요App Studio 문제 해결 및 디버깅.

테스트 환경

초기 앱 개발이 완료되면 다음 단계는 테스트 환경에 게시하는 것입니다. 테스트 환경에 있는 동안 앱은 다른 서비스에 연결하고, 데이터를 전송하고, 다른 서비스에서 데이터를 수신할 수 있습니다. 따라서이 환경을 사용하여 최종 사용자에게 액세스 URL을 제공하여 사용자 수락 테스트(UAT)를 포함한 포괄적인 테스트를 수행할 수 있습니다.

> **Note**
>
> 테스트 환경에 처음 게시하는 데 최대 15분이 걸릴 수 있습니다.

테스트 환경에 게시된 앱의 버전은 최종 사용자가 3시간 동안 사용하지 않으면 제거됩니다. 그러나 모든 버전은 유지되며 버전 기록 탭에서 복원할 수 있습니다.

테스트 환경의 주요 기능은 다음과 같습니다.

- 라이브 데이터 소스 및 APIs와의 통합 테스트
- 제어된 액세스를 통해 제공되는 사용자 수락 테스트(UAT)
- 피드백을 수집하고 문제를 해결하기 위한 환경
- 브라우저 콘솔 및 개발자 도구를 사용하여 클라이언트 측 및 서버 측 활동을 모두 검사하고 디버깅할 수 있습니다.

앱 문제 해결 및 디버깅에 대한 자세한 내용은 섹션을 참조하세요App Studio 문제 해결 및 디버깅.

프로덕션 환경

문제를 테스트하고 수정한 후에는 애플리케이션의 버전을 테스트 환경에서 프로덕션 환경으로 승격하여 실시간으로 운영할 수 있습니다. 프로덕션 환경은 최종 사용자 사용을 위한 라이브 운영 환경이지만 사용자와 공유하기 전에 게시된 버전을 테스트할 수 있습니다.

최종 사용자가 14일 동안 사용하지 않으면 프로덕션 환경에서 게시된 버전이 제거됩니다. 그러나 모든 버전은 유지되며 버전 기록 탭에서 복원할 수 있습니다.

프로덕션 환경의 주요 기능은 다음과 같습니다.

- 최종 사용자 소비를 위한 실시간 운영 환경
- 세분화된 역할 기반 액세스 제어
- 버전 관리 및 롤백 기능
- 클라이언트 측 활동만 검사하고 디버깅할 수 있는 기능
- 라이브 커넥터, 데이터, 자동화 및 APIs 사용

버전 관리 및 릴리스 관리

App Studio는 게시 센터의 버전 관리 시스템을 통해 버전 관리 및 릴리스 관리 기능을 제공합니다.

주요 버전 관리 기능:

- 테스트 환경에 게시하면 새 버전 번호(1.0, 2.0, 3.0...)가 생성됩니다.
- 테스트 환경에서 프로덕션 환경으로 승격할 때 버전 번호는 변경되지 않습니다.
- 버전 기록에서 이전 버전으로 롤백할 수 있습니다.

- 테스트 환경에 게시된 애플리케이션은 3시간 동안 활동이 없으면 일시 중지됩니다. 버전은 유지되며 버전 기록에서 복원할 수 있습니다.
- 프로덕션 환경에 게시된 애플리케이션은 14일 동안 활동이 없으면 제거됩니다. 버전은 유지되며 버전 기록에서 복원할 수 있습니다.

이 버전 관리 모델을 사용하면 앱 개발 및 테스트 주기 전반에 걸쳐 추적성, 롤백 기능 및 최적의 성능을 유지하면서 빠르게 반복할 수 있습니다.

유지 관리 및 운영

App Studio는 특정 유지 관리 작업, 운영 활동을 해결하고 새 소프트웨어 라이브러리를 통합하기 위해 애플리케이션을 자동으로 다시 게시해야 할 수 있습니다. 빌더인 사용자의 작업은 필요하지 않지만 최종 사용자는 애플리케이션에 다시 로그인해야 할 수 있습니다. 특정 상황에서는 애플리케이션을 다시 게시하여 자동으로 추가할 수 없는 새로운 기능과 라이브러리를 통합해야 할 수 있습니다. 다시 게시하기 전에 오류를 해결하고 경고를 검토해야 합니다.

게시된 애플리케이션 공유

아직 게시되지 않은 애플리케이션을 게시하면 공유될 때까지 사용자가 애플리케이션을 사용할 수 없습니다. 게시된 애플리케이션이 공유되면 사용자가 사용할 수 있으며 다른 버전이 게시된 경우 다시 공유할 필요가 없습니다.

> **Note**
>
> 이 섹션에서는 최종 사용자 또는 테스터와 게시된 애플리케이션을 공유하는 방법에 대해 설명합니다. 앱을 빌드하도록 다른 사용자를 초대하는 방법에 대한 자세한 내용은 섹션을 참조하세요여러 사용자로 앱 빌드.

게시된 애플리케이션을 공유하려면

1. 다음 지침에 따라 애플리케이션 목록 또는 앱의 애플리케이션 스튜디오에서 공유 대화 상자에 액세스합니다.
   - 애플리케이션 목록에서 공유 대화 상자에 액세스하려면 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택합니다. 공유하려는 애플리케이션의 작업 열에서 드롭다운을 선택하고 공유를 선택합니다.
   - 애플리케이션 스튜디오에서 공유 대화 상자에 액세스하려면: 앱의 애플리케이션 스튜디오에서 상단 헤더에서 공유를 선택합니다.

2. 공유 대화 상자에서 공유하려는 환경의 탭을 선택합니다. 테스트 또는 프로덕션 탭이 표시되지 않으면 앱이 해당 환경에 게시되지 않을 수 있습니다. 게시에 대한 자세한 내용은 애플리케이션 게시 단원을 참조하십시오.
3. 해당 탭의 드롭다운 메뉴에서 그룹을 선택하여 환경을 공유합니다.
4. (선택 사항) 조건부 페이지 가시성을 테스트하거나 구성하기 위해 그룹에 앱 수준 역할을 할당합니다. 자세한 내용은 페이지의 역할 기반 가시성 구성 단원을 참조하십시오.
5. 공유를 선택합니다.
6. (선택 사항) 링크를 복사하여 사용자와 공유합니다. 애플리케이션 및 환경을 공유한 사용자만 해당 환경의 애플리케이션에 액세스할 수 있습니다.

이전에 게시된 버전으로 롤백

다음 절차에 따라 App Studio 앱의 프로덕션 환경을 이전에 게시된 버전으로 롤백합니다. 애플리케이션 최종 사용자에게 영향을 미치고 배포 후 앱의 롤백 버전이 표시됩니다. 애플리케이션을 롤백하면 이전 게시 시간의 버전으로 구성 요소 코드를 롤백하고 전체 앱 배포 스택(사용자 코드, 구성 요소 구성 상태)에 영향을 미칩니다. 즉, 필드 또는 기타 구성 변경과 같이 App Studio가 구성 요소 코드에 수행한 모든 업데이트는 롤백 애플리케이션 버전이 원래 게시되었을 때와 동일하게 작동하도록 롤백됩니다.

게시된 버전을 롤백해도 개발 환경에서 진행 중인 애플리케이션 버전은 영향을 받지 않습니다.

게시된 버전의 애플리케이션을 롤백하면 게시된 앱과 관련된 문제를 감지하고 이전에 작동한 버전을 즉시 게시해야 하거나 이전 버전을 게시하고 개발 환경에서 앱에 대한 최신 업데이트를 보존하려는 경우에 유용합니다.

> **Note**
>
> 앱의 개발 환경을 이전에 게시된 버전으로 되돌리려면 애플리케이션을 되돌려야 합니다. 자세한 내용은 이전에 게시한 앱 버전 편집 단원을 참조하십시오.

프로덕션 환경 버전을 이전에 게시된 앱 버전으로 롤백하려면

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다. 자세한 내용은 애플리케이션 편집 또는 빌드 단원을 참조하십시오.
2. 프로덕션 환경 타일 상단의 버전 드롭다운 화살표를 선택하여 롤백에 사용할 수 있는 버전을 확인합니다. 드롭다운에는 지난 30일 이내에 게시된 버전이 포함되어 있습니다. 이 드롭다운이 비활성화된 경우 앱 게시가 이미 진행 중이고 한 번에 하나의 게시만 발생할 수 있기 때문일 수 있습니다.

3. 롤백할 버전을 선택합니다.
4. 롤백 이유를 입력하고 롤백을 선택합니다. 롤백 게시가 시작되고 완료되면 애플리케이션의 프로덕션 환경이 선택한 버전으로 업데이트됩니다.

> **Note**
>
> 롤백한 후 이전에 게시된 앱 버전으로 롤포워드할 수도 있습니다.

애플리케이션 내보내기

애플리케이션의 스냅샷을 내보내 다른 App Studio 인스턴스와 공유할 수 있습니다. 앱을 내보내면 앱의 개발 환경에서 스냅샷이 생성되고 가져오기 코드가 생성됩니다. 그런 다음 가져오기 코드를 사용하여 애플리케이션을 보고 빌드할 수 있는 다른 App Studio 인스턴스로 가져올 수 있습니다.

내보낸 앱은 App Studio에서 AWS 리전 지원하는의 인스턴스로 가져올 수 있습니다.

애플리케이션을 내보내려면

1. 탐색 창의 빌드 섹션에서 내 애플리케이션을 선택하여 애플리케이션 목록으로 이동합니다.
2. 내보내려는 애플리케이션의 작업 열에서 드롭다운을 선택합니다.
3. 내보내기를 선택합니다.
4. 가져오기 코드를 생성하고 공유하는 절차는 앱에 대한 가져오기 코드가 이미 생성되었는지 여부에 따라 달라집니다.
   - 가져오기 코드가 생성되지 않은 경우:
     a. 애플리케이션 가져오기 권한에서 내보낸 앱을 가져올 수 있는 인스턴스를 지정합니다. 모든 인스턴스에 가져오기 권한을 부여하거나 인스턴스 IDs를 입력하여 특정 App Studio 인스턴스를 추가할 수 있습니다. 여러 인스턴스 IDs.

        인스턴스 ID를 찾으려면 App Studio 콘솔에서 계정 설정을 선택하여 인스턴스의 계정 설정으로 이동합니다.
     b. 가져오기 코드 생성을 선택합니다.
     c. 생성된 가져오기 코드를 복사하고 공유합니다.
   - 가져오기 코드가 이미 생성된 경우:

- 현재 내보낸 앱을 공유하려면 기존 가져오기 코드를 복사하고 공유합니다. 앱의 최신 변경 사항을 사용하여 내보낸 새 앱을 생성하려면 새 코드 생성을 선택합니다. 필요한 경우 가져오기 권한을 업데이트할 수도 있습니다.

페이지 및 구성 요소: 앱의 사용자 인터페이스 빌드

주제

- 페이지 관리
- 구성 요소 관리
- 페이지의 역할 기반 가시성 구성
- 앱 탐색에서 페이지 정렬 및 구성
- 앱 테마를 사용하여 앱의 색상 변경
- 구성 요소 참조

페이지 관리

다음 절차에 따라 AWS App Studio 애플리케이션에서 페이지를 생성, 편집 또는 삭제합니다.

페이지는 App Studio에서 애플리케이션의 UI를 구성하는 구성 요소의 컨테이너입니다. 각 페이지는 사용자가 상호 작용할 애플리케이션의 사용자 인터페이스(UI) 화면을 나타냅니다. 페이지는 애플리케이션 스튜디오의 페이지 탭에서 생성 및 편집됩니다.

페이지 생성

다음 절차에 따라 App Studio의 애플리케이션에서 페이지를 생성합니다. 기존 페이지 복제에 대한 자세한 내용은 섹션을 참조하세요페이지 복제.

페이지를 생성하려면

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 + 추가를 선택합니다.

페이지 복제

다음 절차에 따라 App Studio의 애플리케이션에서 페이지를 복제합니다.

페이지를 복제하려면

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 복제하려는 페이지 이름 옆에 있는 줄임표 메뉴를 선택하고 복제를 선택합니다. 복제된 페이지는 원래 페이지 바로 뒤에 추가됩니다.

페이지 속성 보기 및 편집

다음 절차에 따라 App Studio의 애플리케이션에서 페이지를 편집합니다. 페이지 이름, 파라미터 및 레이아웃과 같은 속성을 편집할 수 있습니다.

페이지 속성을 보거나 편집하려면

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 편집하려는 페이지 이름 옆에 있는 줄임표 메뉴를 선택하고 페이지 속성을 선택합니다. 그러면 오른쪽 속성 메뉴가 열립니다.
4. 페이지 이름을 편집하려면:

   > **Note**
   >
   > 유효한 페이지 이름 문자: A-Z, a-z, 0-9, _, $

   a. 속성 메뉴 상단 근처의 이름 옆에 있는 연필 아이콘을 선택합니다.
   b. 페이지의 새 이름을 입력하고 Enter 키를 누릅니다.
5. 페이지 파라미터를 생성, 편집 또는 삭제하려면:

   a. 페이지 파라미터를 생성하려면 페이지 파라미터 섹션에서 + 새로 추가를 선택합니다.
   b. 페이지 파라미터의 키 또는 설명 값을 편집하려면 변경하려는 속성의 입력 필드를 선택하고 새 값을 입력합니다. 편집 시 변경 사항이 저장됩니다.
   c. 페이지 파라미터를 삭제하려면 삭제하려는 페이지 파라미터의 휴지통 아이콘을 선택합니다.

6. 페이지의 로고 또는 배너를 추가, 편집 또는 제거하려면:

   a. 페이지 로고 또는 배너를 추가하려면 스타일 섹션에서 해당 옵션을 활성화합니다. 이미지의 소스를 구성하고 선택적으로 대체 텍스트를 제공합니다.

   b. 페이지 로고 또는 배너를 편집하려면 스타일 섹션에서 필드를 업데이트합니다.

   c. 페이지 로고 또는 배너를 제거하려면 스타일 섹션에서 해당 옵션을 비활성화합니다.

7. 페이지의 레이아웃을 편집하려면:

   - 레이아웃 섹션의 필드를 업데이트합니다.

페이지 삭제

다음 절차에 따라 App Studio의 애플리케이션에서 페이지를 삭제합니다.

페이지를 삭제하려면

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 삭제할 페이지 이름 옆에 있는 줄임표 메뉴를 선택하고 삭제를 선택합니다.

구성 요소 관리

다음 절차에 따라 App Studio 애플리케이션 스튜디오의 페이지에서 구성 요소를 추가, 편집 및 삭제하여 애플리케이션에 원하는 사용자 인터페이스를 만듭니다.

페이지에 구성 요소 추가

다음 절차에 따라 App Studio의 페이지에 구성 요소를 추가합니다. 기존 구성 요소 복제에 대한 자세한 내용은 섹션을 참조하세요구성 요소 복제.

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 구성 요소 패널은 사용 가능한 구성 요소가 포함된 오른쪽 메뉴에 있습니다.
4. 패널에서 원하는 구성 요소를 캔버스로 끌어서 놓습니다. 또는 패널에서 구성 요소를 두 번 클릭하여 현재 페이지의 중앙에 자동으로 추가할 수 있습니다.

5. 이제 구성 요소를 추가했으므로 오른쪽 속성 패널을 사용하여 데이터 소스, 레이아웃 및 동작과 같은 설정을 조정합니다. 각 구성 요소 유형 구성에 대한 자세한 내용은 섹션을 참조하세요구성 요소 참조.

구성 요소 복제

다음 절차에 따라 App Studio 앱에서 구성 요소를 복제합니다. 복제된 구성 요소에는 원래 구성 요소의 연결된 자동화 또는 엔터티가 포함됩니다.

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 구성 요소를 복제하는 방법에는 두 가지가 있습니다.
   - 왼쪽 페이지 메뉴에서 복제하려는 구성 요소가 포함된 페이지를 확장합니다. 복제하려는 구성 요소의 이름 옆에 있는 줄임표 메뉴를 선택하고 복제를 선택합니다.
   - 복제하려는 구성 요소를 선택하고 중복 아이콘을 선택합니다.

   복제된 구성 요소는 원래 구성 요소 바로 뒤에 추가됩니다.

   > **Tip**
   >
   > CTRL+Z 또는 CMD+Z 키보드 바로 가기를 사용하여 개발 환경의 다른 많은 작업과 함께 구성 요소 복제를 실행 취소할 수 있습니다.

구성 요소 속성 보기 및 편집

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 구성 요소가 포함된 페이지를 확장하고 보거나 편집할 구성 요소를 선택합니다. 또는 페이지를 선택한 다음 캔버스에서 구성 요소를 선택할 수 있습니다.
4. 오른쪽 속성 패널에는 선택한 구성 요소에 대해 구성 가능한 설정이 표시됩니다.
5. 사용 가능한 다양한 속성과 옵션을 살펴보고 필요에 따라 업데이트하여 구성 요소의 모양과 동작을 구성합니다. 예를 들어 데이터 소스를 변경하거나 레이아웃을 구성하거나 추가 기능을 활성화할 수 있습니다.

   각 구성 요소 유형 구성에 대한 자세한 내용은 섹션을 참조하세요구성 요소 참조.

구성 요소 삭제

1. 필요한 경우 애플리케이션을 편집하여 애플리케이션의 개발 환경으로 이동합니다.
2. 페이지 탭으로 이동합니다.
3. 왼쪽 페이지 메뉴에서 삭제할 구성 요소를 선택하여 선택합니다.
4. 오른쪽 속성 메뉴에서 휴지통 아이콘을 선택합니다.
5. 확인 대화 상자에서 삭제를 선택합니다.

페이지의 역할 기반 가시성 구성

App Studio 앱 내에서 역할을 생성하고 해당 역할에 따라 페이지의 가시성을 구성할 수 있습니다. 예를 들어 프로젝트 승인 또는 클레임 처리와 같은 기능을 제공하고 특정 페이지를 특정 역할에 표시하는 앱의 관리자, 관리자 또는 사용자와 같은 사용자 요구 사항 또는 액세스 수준에 따라 역할을 생성할 수 있습니다. 이 예에서 관리자는 전체 액세스 권한을 가질 수 있고, 관리자는 보고 대시보드를 볼 수 있으며, 사용자는 입력 양식이 있는 작업 페이지에 액세스할 수 있습니다.

다음 절차에 따라 App Studio 앱에서 페이지의 역할 기반 가시성을 구성합니다.

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다. 왼쪽 탐색 메뉴에서 내 애플리케이션을 선택하고 애플리케이션을 찾은 다음 편집을 선택합니다.
2. 애플리케이션 스튜디오에서 앱 수준 역할을 생성합니다.
    a. 애플리케이션 스튜디오 상단의 앱 설정 탭을 선택합니다.
    b. + 역할 추가를 선택합니다.
    c. 역할 이름에 역할을 식별할 이름을 입력합니다. 이름을 사용하여 페이지 가시성을 설정하므로 그룹의 액세스 수준 또는 의무를 설명하는 이름을 사용하는 것이 좋습니다.
    d. 선택적으로 설명에 역할에 대한 설명을 추가합니다.
    e. 이 단계를 반복하여 필요한 만큼 역할을 생성합니다.
3. 페이지의 가시성 구성
    a. 애플리케이션 스튜디오 상단의 페이지 탭을 선택합니다.
    b. 왼쪽 페이지 메뉴에서 역할 기반 가시성을 구성할 페이지를 선택합니다.
    c. 오른쪽 메뉴에서 속성 탭을 선택합니다.
    d. 가시성에서 모든 최종 사용자에게 열기를 비활성화합니다.

e. 역할을 선택한 상태로 유지하여 이전 단계에서 생성한 역할 목록에서 선택합니다. 보다 복잡한 가시성 구성을 위해 JavaScript 표현식을 작성하려면 사용자 지정을 선택합니다.

   1. 역할을 선택한 상태에서 페이지가 표시될 앱 역할의 확인란을 선택합니다.
   2. 사용자 지정을 선택한 상태에서 true 또는 false로 확인되는 JavaScript 표현식을 입력합니다. 다음 예제를 사용하여 현재 사용자에게 관리자 역할이 있는지 확인합니다`{{currentUser.roles.includes('manager')}}`.

4. 이제 가시성이 구성되었으므로 앱을 미리 보고 페이지 가시성을 테스트할 수 있습니다.

   a. 미리 보기를 선택하여 앱 미리 보기를 엽니다.

   b. 미리 보기 오른쪽 상단에서 미리 보기 메뉴를 선택하고 테스트하려는 역할의 확인란을 선택합니다. 표시되는 페이지에는 선택한 역할이 반영되어야 합니다.

5. 이제 게시된 앱의 앱 역할에 그룹을 할당합니다. 그룹 및 역할 할당은 각 환경에 대해 별도로 구성해야 합니다. 앱 환경에 대한 자세한 내용은 섹션을 참조하세요애플리케이션 환경.

   > **Note**
   >
   > App Studio 그룹을 생성 및 구성한 역할에 할당하려면 앱을 테스트 또는 프로덕션 환경에 게시해야 합니다. 필요한 경우 앱을 게시하여 역할에 그룹을 할당합니다. 게시에 대한 자세한 내용은 애플리케이션 게시 단원을 참조하십시오.

   a. 애플리케이션 스튜디오의 오른쪽 상단에서 공유를 선택합니다.

   b. 페이지 가시성을 구성하려는 환경의 탭을 선택합니다.

   c. 그룹 검색 입력 상자를 선택하고 앱 버전을 공유할 그룹을 선택합니다. 텍스트를 입력하여 그룹을 검색할 수 있습니다.

   d. 드롭다운 메뉴에서 그룹에 할당할 역할을 선택합니다. 역할 없음을 선택하여 앱 버전을 공유하고 그룹에 역할을 할당할 수 없습니다. 모든 사용자에게 표시되는 페이지만 역할이 없는 그룹에 표시됩니다.

   e. 공유를 선택합니다. 이 단계를 반복하여 필요한 만큼 그룹을 추가합니다.

앱 탐색에서 페이지 정렬 및 구성

이 주제에는 App Studio 애플리케이션에서 페이지 재정렬 및 구성에 대한 정보가 포함되어 있습니다. 제품에는 앱 페이지가 표시되는 두 가지 영역이 있습니다. 애플리케이션 스튜디오에서 앱을 편집하는 동안 왼쪽 페이지 메뉴와 게시된 앱의 미리 보기 왼쪽 탐색이 있습니다.

앱을 편집하는 동안 왼쪽 페이지 메뉴에서 페이지 순서 지정

애플리케이션 스튜디오에서 앱을 편집하는 동안 페이지는 왼쪽 페이지 메뉴에서 생성 시간을 기준으로 정렬됩니다. 이 메뉴의 페이지는 재정렬할 수 없습니다.

미리 보기 또는 게시된 앱의 탐색에서 페이지 정렬, 표시 또는 숨기기

미리 보기 또는 게시된 앱의 왼쪽 탐색에서 다음 설정을 편집할 수 있습니다.

- 전체 탐색의 가시성
- 탐색에서 특정 페이지의 표시 여부
- 탐색의 페이지 순서

미리 보기 또는 게시된 앱의 왼쪽 탐색을 편집하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동하여 편집합니다.
2. 왼쪽 페이지 메뉴에서 헤더 및 탐색을 선택합니다.
3. 오른쪽 헤더 및 탐색 메뉴에서 다음을 보거나 편집합니다.

    a. 앱에서 탐색을 숨기거나 표시하려면 앱 탐색 토글을 사용합니다.
    b. 앱 탐색에서 페이지를 숨기려면 페이지를 연결 해제된 페이지 섹션으로 드래그합니다.
    c. 앱 탐색에서 페이지를 재정렬하려면 연결된 페이지 섹션에서 원하는 순서로 페이지를 드래그합니다.

앱 테마를 사용하여 앱의 색상 변경

다음 절차에 따라 앱 테마를 구성하여 애플리케이션의 색상을 업데이트합니다.

1. 필요한 경우 앱의 애플리케이션 스튜디오로 이동하여 편집합니다.
2. 애플리케이션 스튜디오에서 페이지 탭으로 이동합니다.
3. 왼쪽 탐색에서 앱 테마를 선택하여 오른쪽 앱 테마 설정을 엽니다.

4. 기본 테마에서 조명 모드 또는 다크 모드를 선택합니다.
5. 애플리케이션에 사용자 지정 색상을 추가하려면 사용자 지정 토글을 활성화하고 다음 설정을 업데이트합니다.

   a. 기본 색상에서 특정 구성 요소에 적용되는 색상과 앱의 탐색을 선택합니다. 색상 선택기, RGB, HSL 또는 HEX 코드를 사용하여 색상을 선택할 수 있습니다.

   > **Note**
   >
   > App Studio는 자동으로 색상에 액세스할 수 있도록 합니다. 예를 들어 조명 모드에서 조명 색상을 선택하면 더 쉽게 액세스할 수 있도록 업데이트됩니다.

   b. 헤더 색상에서 앱의 헤더에 적용되는 색상을 선택합니다. 색상 선택기, RGB, HSL 또는 HEX 코드를 사용하여 색상을 선택할 수 있습니다.

   c. 미리 정의된 테마를 보고 선택할 기본 테마를 선택하거나 무작위화를 선택하여 무작위 기본 및 헤더 색상을 생성합니다.
6. 변경 사항 저장을 선택하여 앱 테마를 업데이트합니다.

구성 요소 참조

이 주제에서는 각 App Studio의 구성 요소, 속성에 대해 자세히 설명하고 구성 예제를 포함합니다.

공통 구성 요소 속성

이 섹션에서는 애플리케이션 스튜디오의 여러 구성 요소에서 공유되는 일반적인 속성과 기능을 간략하게 설명합니다. 각 속성 유형에 대한 특정 구현 세부 정보 및 사용 사례는 구성 요소에 따라 다를 수 있지만 이러한 속성의 일반적인 개념은 App Studio 전체에서 일관되게 유지됩니다.

이름

각 구성 요소에 대해 기본 이름이 생성되지만를 편집하여 각 구성 요소의 고유한 이름으로 변경할 수 있습니다. 이 이름을 사용하여 동일한 페이지 내의 다른 구성 요소 또는 표현식에서 구성 요소 및 해당 데이터를 참조합니다. 제한: 구성 요소 이름에 공백을 포함하지 마세요. 문자, 숫자, 밑줄 및 달러 기호만 포함할 수 있습니다. 예: `userNameInput`, `ordersTable`, `metricCard1`.

기본 값, 보조 값 및 값

애플리케이션 스튜디오의 많은 구성 요소는 구성 요소 내에 표시되는 콘텐츠 또는 데이터를 결정하는 값 또는 표현식을 지정하기 위한 필드를 제공합니다. 이러한 필드는 구성 요소 유형 및 용도에 따라 `Value`, `Primary value` `Secondary value`또는 단순히 로 레이블이 지정되는 경우가 많습니다.

`Primary value` 필드는 일반적으로 구성 요소 내에서 눈에 띄게 표시되어야 하는 기본 값, 데이터 포인트 또는 콘텐츠를 정의하는 데 사용됩니다.

사용 가능한 경우 `Secondary value` 필드는 기본 값과 함께 추가 또는 지원 값 또는 정보를 표시하는 데 사용됩니다.

`Value` 필드를 사용하면 구성 요소에 표시해야 하는 값 또는 표현식을 지정할 수 있습니다.

이러한 필드는 정적 텍스트 입력과 동적 표현식을 모두 지원합니다. 표현식을 사용하면 애플리케이션 내의 다른 구성 요소, 데이터 소스 또는 변수의 데이터를 참조하여 동적 및 데이터 기반 콘텐츠 표시를 활성화할 수 있습니다.

표현식에 대한 구문

이러한 필드에 표현식을 입력하는 구문은 일관된 패턴을 따릅니다.

```
{{expression}}
```

여기서 `##`식은 표시하려는 값 또는 데이터로 평가되는 유효한 표현식입니다.

예: 정적 텍스트

- 기본 값: `"123"` 또는와 같은 정적 숫자 또는 값을 직접 입력할 수 있습니다`"$1,999.99"`.
- 보조 값: `"Goal"` 또는와 같은 정적 텍스트 레이블을 입력할 수 있습니다`"Projected Revenue"`.
- 값: `"since last month"` 또는와 같은 정적 문자열을 입력할 수 있습니다`"Total Quantity"`.

예: 표현식

- `Hello, {{currentUser.firstName}}`: 현재 로그인한 사용자의 이름이 인 인사말을 표시합니다.
- `{{currentUser.role === 'Admin' ? 'Admin Dashboard' : 'User Dashboard'}}`: 사용자의 역할에 따라 다른 대시보드 제목을 조건부로 표시합니다.

- {{ui.componentName.data?.[0]?.fieldName}}: ID가 인 구성 요소의 데이터에 있는 첫 번째 항목에서 fieldName 필드 값을 검색합니다componentName.
- {{ui.componentName.value * 100}}: ID가 인 구성 요소의 값에 대한 계산을 수행합니다componentName.
- {{ui.componentName.value + ' items'}}: 구성 요소의 값을 ID componentName 및 문자열과 연결합니다' items'.
- {{ui.ordersTable.data?.[0]?.orderNumber}}: ordersTable 구성 요소의 첫 번째 데이터 행에서 주문 번호를 검색합니다.
- {{ui.salesMetrics.data?.[0]?.totalRevenue * 1.15}}: salesMetrics 구성 요소의 첫 번째 데이터 행에서 총 수익을 15% 늘려 예상 수익을 계산합니다.
- {{ui.customerProfile.data?.[0]?.firstName + ' ' + ui.customerProfile.data?.lastName}}: customerProfile 구성 요소의 데이터에서 이름과 성을 연결합니다.
- {{new Date(ui.orderDetails.data?.orderDate).toLocaleDateString()}}: orderDetails 구성 요소의 주문 날짜를 보다 읽기 쉬운 날짜 문자열로 포맷합니다.
- {{ui.productList.data?.length}}: productList 구성 요소에 연결된 데이터의 총 제품 수를 표시합니다.
- {{ui.discountPercentage.value * ui.orderTotal.value}}: 할인율과 주문 합계를 기준으로 할인 금액을 계산합니다.
- {{ui.cartItemCount.value + ' items in cart'}}: 레이블과 함께 장바구니의 항목 수를 표시합니다items in cart.

이러한 표현식 필드를 사용하면 애플리케이션 내에 동적 데이터 기반 콘텐츠를 생성하여 사용자의 컨텍스트 또는 애플리케이션 상태에 맞는 정보를 표시할 수 있습니다. 이를 통해 보다 개인화된 대화형 사용자 경험을 제공할 수 있습니다.

Label

레이블 속성을 사용하면 구성 요소의 캡션 또는 제목을 지정할 수 있습니다. 이 레이블은 일반적으로 구성 요소와 함께 또는 구성 요소 위에 표시되므로 사용자가 목적을 이해하는 데 도움이 됩니다.

정적 텍스트와 표현식을 모두 사용하여 레이블을 정의할 수 있습니다.

예: 정적 텍스트

레이블 필드에 "이름" 텍스트를 입력하면 구성 요소에 "이름"이 레이블로 표시됩니다.

예: 표현식

예: 소매점

다음 예제에서는 각 사용자에 대해 레이블을 개인 설정하므로 인터페이스가 개인에 맞게 조정된 느낌을 줍니다.

```
{{currentUser.firstName}} {{currentUser.lastName}}'s Account
```

예: SaaS 프로젝트 관리

다음 예제에서는 선택한 프로젝트에서 데이터를 가져와 컨텍스트별 레이블을 제공하므로 사용자가 애플리케이션 내에서 방향을 유지하는 데 도움이 됩니다.

```
Project {{ui.projectsTable.selectedRow.id}} - {{ui.projectsTable.selectedRow.name}}
```

예: 의료 클리닉

다음 예제에서는 현재 사용자의 프로필과 의사 정보를 참조하여 환자에게 보다 개인화된 경험을 제공합니다.

```
Dr. {{ui.doctorProfileTable.data.firstName}}
       {{ui.doctorProfileTable.data.lastName}}
```

Placeholder

자리 표시자 속성을 사용하면 구성 요소가 비어 있을 때 구성 요소 내에 표시되는 힌트 또는 지침 텍스트를 지정할 수 있습니다. 이를 통해 사용자는 예상 입력 형식을 이해하거나 추가 컨텍스트를 제공할 수 있습니다.

정적 텍스트와 표현식을 모두 사용하여 자리 표시자를 정의할 수 있습니다.

예: 정적 텍스트

`Enter your name` 자리 표시자 필드에 텍스트를 입력하면 구성 요소가 자리 표시자 텍스트`Enter your name`로 표시됩니다.

예: 표현식

예: 금융 서비스

`Enter the amount you'd like to deposit into your {{ui.accountsTable.selectedRow.balance}} account`이 예제에서는 선택한 계정에서 데이터를 가져와 관련 프롬프트를 표시하므로 은행 고객이 인터페이스를 직관적으로 사용할 수 있습니다.

예: 전자 상거래

`Enter the coupon code for {{ui.cartTable.data.currency}} total` 여기의 자리 표시자는 사용자의 장바구니 콘텐츠에 따라 동적으로 업데이트되므로 원활한 체크아웃 환경을 제공합니다.

예: 의료 클리닉

`Enter your {{ui.patientProfile.data.age}}-year-old patient's symptoms`이 애플리케이션은 환자의 연령을 참조하는 표현식을 사용하여 보다 개인화되고 유용한 자리 표시자를 만들 수 있습니다.

소스

소스 속성을 사용하면 구성 요소의 데이터 소스를 선택할 수 있습니다. 선택하면 , `entity` `expression`또는 데이터 소스 유형 중에서 선택할 수 있습니다`automation`.

개체

엔터티를 데이터 소스로 선택하면 구성 요소를 애플리케이션의 기존 데이터 엔터티 또는 모델에 연결할 수 있습니다. 이는 애플리케이션 전체에서 활용하려는 데이터 구조 또는 스키마가 잘 정의된 경우에 유용합니다.

엔터티 데이터 소스를 사용하는 경우:

- 구성 요소에 표시할 정보가 포함된 데이터 모델 또는 엔터티가 있는 경우(예: "Name", "Description", "Price"와 같은 필드가 있는 "Products" 엔터티).
- 데이터베이스, API 또는 기타 외부 데이터 소스에서 데이터를 동적으로 가져와 구성 요소에 표시해야 하는 경우.
- 애플리케이션의 데이터 모델에 정의된 관계 및 연결을 활용하려는 경우.

개체에 대한 쿼리 선택

구성 요소를 전체 개체가 아닌 개체에서 데이터를 검색하는 특정 쿼리에 연결하는 것이 좋습니다. 엔터티 데이터 소스에는 기존 쿼리에서 선택하거나 새 쿼리를 생성할 수 있는 옵션이 있습니다.

쿼리를 선택하면 다음을 수행할 수 있습니다.

- 특정 기준에 따라 구성 요소에 표시되는 데이터를 필터링합니다.
- 쿼리에 파라미터를 전달하여 데이터를 동적으로 필터링하거나 정렬합니다.
- 쿼리에 정의된 복잡한 조인, 집계 또는 기타 데이터 조작 기술을 활용합니다.

예를 들어, 애플리케이션에 , Name Email및와 같은 필드가 있는 Customers엔터티가 있는 경우. PhoneNumber 테이블 구성 요소를이 엔터티에 연결하고 상태에 따라 고객을 필터링하는 사전 정의된 ActiveCustomers 데이터 작업을 선택할 수 있습니다. 이렇게 하면 전체 고객 데이터베이스가 아닌 테이블의 활성 고객만 표시할 수 있습니다.

개체 데이터 소스에 파라미터 추가

개체를 데이터 소스로 사용하는 경우 구성 요소에 파라미터를 추가할 수도 있습니다. 이러한 파라미터를 사용하여 구성 요소에 표시된 데이터를 필터링, 정렬 또는 변환할 수 있습니다.

예를 들어 , Name, Description Price및 같은 필드가 있는 Products 개체가 있는 경우 Category 제품 목록을 표시하는 category 테이블 구성 요소에 라는 파라미터를 추가할 수 있습니다. 사용자가 드롭다운에서 범주를 선택하면 테이블은 데이터 작업의 {{params.category}} 표현식을 사용하여 선택한 범주에 속하는 제품만 표시하도록 자동으로 업데이트됩니다.

표현식

표현식을 데이터 소스로 선택하여 사용자 지정 표현식 또는 계산을 입력하여 구성 요소에 대한 데이터를 동적으로 생성합니다. 이는 변환을 수행하거나, 여러 소스의 데이터를 결합하거나, 특정 비즈니스 로직을 기반으로 데이터를 생성해야 할 때 유용합니다.

표현식 데이터 소스를 사용해야 하는 경우:

- 데이터 모델에서 직접 사용할 수 없는 데이터를 계산하거나 도출해야 하는 경우(예: 수량 및 가격을 기준으로 총 주문 금액 계산).
- 여러 엔터티 또는 데이터 소스의 데이터를 결합하여 복합 보기를 생성하려는 경우(예: 고객 연락처 정보와 함께 고객의 주문 내역 표시).
- 특정 규칙 또는 조건을 기반으로 데이터를 생성해야 하는 경우(예: 사용자의 검색 기록을 기반으로 "권장 제품" 목록 표시).

예를 들어 이번 달의 총 수익을 표시해야 하는 ## 구성 요소가 있는 경우 다음과 같은 표현식을 사용하여 월별 수익을 계산하고 표시할 수 있습니다.

```
{{ui.table1.orders.concat(ui.table1.orderDetails).filter(o => o.orderDate.getMonth()
 === new Date().getMonth()).reduce((a, b) => a + (b.quantity * b.unitPrice), 0)}}
```

자동화

구성 요소를 애플리케이션의 기존 자동화 또는 워크플로에 연결하려면 자동화를 데이터 소스로 선택합니다. 이는 구성 요소의 데이터 또는 기능이 특정 프로세스 또는 워크플로의 일부로 생성되거나 업데이트될 때 유용합니다.

자동화 데이터 소스를 사용해야 하는 경우:

- 구성 요소에 표시된 데이터가 특정 자동화 또는 워크플로의 결과인 경우(예: 승인 프로세스의 일부로 업데이트된 "대기 중인 승인" 테이블).
- 자동화 내의 이벤트 또는 조건을 기반으로 구성 요소에 대한 작업 또는 업데이트를 트리거하려는 경우(예: 지표를 SKU의 최신 판매 수치로 업데이트).
- 자동화를 통해 애플리케이션의 다른 서비스 또는 시스템과 구성 요소를 통합해야 하는 경우(예: 타사 API에서 데이터를 가져와 테이블에 표시).

예를 들어 작업 애플리케이션 프로세스를 통해 사용자를 안내하는 단계 흐름 구성 요소가 있는 경우입니다. Stepflow 구성 요소는 작업 애플리케이션 제출, 신원 조회 및 제안 생성을 처리하는 자동화에 연결할 수 있습니다. 자동화가 이러한 단계를 진행함에 따라 단계 흐름 구성 요소는 애플리케이션의 현재 상태를 반영하도록 동적으로 업데이트할 수 있습니다.

각 구성 요소에 적합한 데이터 소스를 신중하게 선택하면 애플리케이션의 사용자 인터페이스가 올바른 데이터 및 로직으로 구동되도록 하여 사용자에게 원활하고 매력적인 경험을 제공할 수 있습니다.

다음과 같은 경우 표시됩니다.

Visible if 속성을 사용하여 특정 조건 또는 데이터 값을 기반으로 구성 요소 또는 요소를 표시하거나 숨깁니다. 이는 애플리케이션 사용자 인터페이스의 특정 부분에 대한 가시성을 동적으로 제어하려는 경우에 유용합니다.

속성이 다음 구문을 사용하는 경우 표시:

```
{{expression ? true : false}}
```

또는

```
{{expression}}
```

여기서 ##식은 true 또는 로 평가되는 부울 표현식입니다false.

표현식이 로 평가되면 true구성 요소가 표시됩니다. 표현식이 로 평가되면 false구성 요소가 숨겨집니다. 표현식은 애플리케이션 내의 다른 구성 요소, 데이터 소스 또는 변수의 값을 참조할 수 있습니다.

표현식 예제인 경우 표시

예: 이메일 입력을 기반으로 암호 입력 필드 표시 또는 숨기기

이메일 입력 필드와 암호 입력 필드가 있는 로그인 양식이 있다고 가정해 보겠습니다. 사용자가 이메일 주소를 입력한 경우에만 암호 입력 필드를 표시하려고 합니다. 표현식인 경우 다음 Visible을 사용할 수 있습니다.

```
{{ui.emailInput.value !== ""}}
```

이 표현식은 emailInput 구성 요소의 값이 빈 문자열이 아닌지 확인합니다. 사용자가 이메일 주소를 입력하면 표현식이 로 평가true되고 암호 입력 필드가 표시됩니다. 이메일 필드가 비어 있으면 표현식이 로 평가false되고 암호 입력 필드가 숨겨집니다.

예: 드롭다운 선택에 따라 추가 양식 필드 표시

사용자가 드롭다운 목록에서 범주를 선택할 수 있는 양식이 있다고 가정해 보겠습니다. 선택한 범주에 따라 더 구체적인 정보를 수집하기 위해 추가 양식 필드를 표시하거나 숨기려고 합니다.

예를 들어 사용자가 ## 범주를 선택하는 경우 다음 표현식을 사용하여 추가 ## ## ## 필드를 표시할 수 있습니다.

```
{{ui.categoryDropdown.value === "Products"}}
```

사용자가 ### 또는 ### 범주를 선택하는 경우이 표현식을 사용하여 다른 추가 필드 세트를 표시할 수 있습니다.

```
{{ui.categoryDropdown.value === "Services" || ui.categoryDropdown.value ===
 "Consulting"}}
```

예: 기타

구성 요소의 값이 빈 문자열이 아닌 경우 구성 `textInput1` 요소를 표시하려면:

```
{{ui.textInput1.value === "" ? false : true}}
```

구성 요소를 항상 표시하려면:

```
{{true}}
```

구성 요소의 값이 빈 문자열이 아닌 경우 구성 `emailInput` 요소를 표시하려면:

```
{{ui.emailInput.value !== ""}}
```

다음과 같은 경우 비활성화됨

Disabled if 기능을 사용하면 특정 조건 또는 데이터 값에 따라 구성 요소를 조건부로 활성화하거나 비활성화할 수 있습니다. 이는 구성 요소의 활성화 여부를 결정하는 부울 표현식을 수락하는 Disabled if 속성을 사용하여 수행됩니다.

속성이 다음 구문을 사용하는 경우 비활성화됨:

```
{{expression ? true : false}}
```

또는

```
{{expression}}
```

표현식 예제인 경우 비활성화됨

예: 양식 검증을 기반으로 제출 버튼 비활성화

입력 필드가 여러 개인 양식이 있고 모든 필수 필드가 올바르게 채워질 때까지 제출 버튼을 비활성화하려는 경우 다음 Disabled If 표현식을 사용할 수 있습니다.

```
{{ui.nameInput.value === "" || ui.emailInput.value === "" || ui.passwordInput.value ===
 ""}}
```

이 표현식은 필수 입력 필드(nameInput, emailInput, passwordInput)가 비어 있는지 확인합니다. 필드가 비어 있으면 표현식이 로 평가true되고 제출 버튼이 비활성화됩니다. 필수 필드를 모두 입력하면 표현식이 로 평가false되고 제출 버튼이 활성화됩니다.

Visible if 및 Disabled ff 속성에서 이러한 유형의 조건 표현식을 사용하면 사용자 입력에 적응하는 동적 및 응답형 사용자 인터페이스를 생성하여 애플리케이션 사용자에게 보다 간소화되고 관련성이 높은 환경을 제공할 수 있습니다.

여기서 ##식은 true 또는 false로 평가되는 부울 표현식입니다.

예제:

```
{{ui.textInput1.value === "" ? true : false}}: The component will be Disabled if the
 textInput1 component's value is an empty string.
{{!ui.nameInput.isValid || !ui.emailInput.isValid || !ui.passwordInput.isValid}}: The
 component will be Disabled if any of the named input fields are invalid.
```

컨테이너 레이아웃

레이아웃 속성은 구성 요소 내의 콘텐츠 또는 요소가 정렬되고 배치되는 방식을 결정합니다. 여러 레이아웃 옵션을 사용할 수 있으며, 각 레이아웃 옵션은 아이콘으로 표시됩니다.

- 열 레이아웃:이 레이아웃은 콘텐츠 또는 요소를 단일 열에 세로로 정렬합니다.
- 2개 열 레이아웃:이 레이아웃은 구성 요소를 동일한 너비의 2개 열로 나누므로 콘텐츠 또는 요소를 나란히 배치할 수 있습니다.
- 행 레이아웃:이 레이아웃은 콘텐츠 또는 요소를 단일 행에 가로로 정렬합니다.

Orientation(방향)

- 수평:이 레이아웃은 콘텐츠 또는 요소를 단일 행에 수평으로 정렬합니다.
- 세로:이 레이아웃은 콘텐츠 또는 요소를 단일 열에 세로로 정렬합니다.
- 인라인 래핑:이 레이아웃은 콘텐츠 또는 요소를 가로로 정렬하지만 요소가 사용 가능한 너비를 초과하면 다음 줄로 래핑됩니다.

정렬

- 왼쪽: 콘텐츠 또는 요소를 구성 요소의 왼쪽에 정렬합니다.
- Center: 구성 요소 내에서 콘텐츠 또는 요소를 가로로 중앙에 배치합니다.

- 오른쪽: 콘텐츠 또는 요소를 구성 요소의 오른쪽에 정렬합니다.

너비

Width 속성은 구성 요소의 가로 크기를 지정합니다. 상위 컨테이너 또는 사용 가능한 공간을 기준으로 구성 요소의 너비를 나타내는 0%~100% 사이의 백분율 값을 입력할 수 있습니다.

Height

Height 속성은 구성 요소의 세로 크기를 지정합니다. "auto" 값은 콘텐츠 또는 사용 가능한 공간에 따라 구성 요소의 높이를 자동으로 조정합니다.

사이의 공간

속성 간 공간에 따라 구성 요소 내의 콘텐츠 또는 요소 간 간격이 결정됩니다. 0px(간격 없음)에서 64px 사이의 값을 4px 단위로 선택할 수 있습니다(예: 4px, 8px, 12px 등).

패딩

패딩 속성은 콘텐츠 또는 요소와 구성 요소의 엣지 사이의 공간을 제어합니다. 0px(패딩 없음)에서 64px 사이의 값을 4px 단위로 선택할 수 있습니다(예: 4px, 8px, 12px 등).

배경

백그라운드는 구성 요소의 백그라운드 색상 또는 스타일을 활성화하거나 비활성화합니다.

이러한 레이아웃 속성은 구성 요소 내에서 콘텐츠를 정렬하고 위치를 지정할 뿐만 아니라 구성 요소 자체의 크기, 간격 및 시각적 모양을 제어할 수 있는 유연성을 제공합니다.

데이터 구성 요소

이 섹션에서는 테이블, 세부 정보, 지표, 양식 및 반복기 구성 요소를 포함하여 애플리케이션 스튜디오에서 사용할 수 있는 다양한 데이터 구성 요소를 다룹니다. 이러한 구성 요소는 애플리케이션 내에서 데이터를 표시, 수집 및 조작하는 데 사용됩니다.

표

테이블 구성 요소에는 행과 열이 포함된 테이블 형식의 데이터가 표시됩니다. 데이터베이스의 항목 또는 레코드 목록과 같은 구조화된 데이터를 체계적이고 easy-to-read 방식으로 표시하는 데 사용됩니다.

테이블 속성

테이블 구성 요소는 , Name Source및와 같은 다른 구성 요소와 몇 가지 일반적인 속성을 공유합니다Actions. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

테이블 구성 요소에는 공통 속성 외에도 , 및를 비롯한 특정 속성 Columns Search and export및 구성 옵션이 있습니다Expressions.

열

이 섹션에서는 테이블에 표시할 열을 정의할 수 있습니다. 각 열은 다음 속성으로 구성할 수 있습니다.

- 형식: 필드의 데이터 유형입니다. 예: text, number, date.
- 열 레이블: 열의 헤더 텍스트입니다.
- 값:이 열에 표시되어야 하는 데이터 소스의 필드입니다.

  이 필드를 사용하면 열 셀에 표시해야 하는 값 또는 표현식을 지정할 수 있습니다. 표현식을 사용하여 연결된 소스 또는 기타 구성 요소의 데이터를 참조할 수 있습니다.

  예: {{currentRow.*title*}} -이 표현식은 열 셀의 현재 행에 있는 ## 필드의 값을 표시합니다.
- 정렬 활성화:이 토글을 사용하면 특정 열에 대한 정렬 기능을 활성화하거나 비활성화할 수 있습니다. 활성화하면 사용자는이 열의 값을 기반으로 테이블 데이터를 정렬할 수 있습니다.

검색 및 내보내기

테이블 구성 요소는 검색 및 내보내기 기능을 활성화하거나 비활성화하는 다음과 같은 토글을 제공합니다.

- 검색 표시이 토글을 활성화하면 테이블에 검색 입력 필드가 추가되어 사용자가 표시된 데이터를 검색하고 필터링할 수 있습니다.
- 내보내기 표시이 토글을 활성화하면 테이블에 내보내기 옵션이 추가되어 사용자가 CSV와 같은 다양한 형식으로 테이블 데이터를 다운로드할 수 있습니다.

> **Note**
>
> 기본적으로 검색 기능은 테이블에 로드된 데이터로 제한됩니다. 검색을 완전히 사용하려면 모든 데이터 페이지를 로드해야 합니다.

페이지당 행 수

테이블의 페이지당 표시할 행 수를 지정할 수 있습니다. 그런 다음 사용자는 페이지 간에 이동하여 전체 데이터 세트를 볼 수 있습니다.

미리 가져오기 제한

각 쿼리 요청에서 미리 가져올 최대 레코드 수를 지정합니다. 최대값은 3000입니다.

작업

작업 섹션에서 다음 속성을 구성합니다.

- 작업 위치: 오른쪽으로 고정이 활성화되면 사용자 스크롤과 관계없이 추가된 모든 작업이 항상 테이블 오른쪽에 표시됩니다.
- 작업: 테이블에 작업 버튼을 추가합니다. 사용자가 클릭할 때 다음과 같이 지정된 작업을 수행하도록 이러한 버튼을 구성할 수 있습니다.
  - 구성 요소 작업 실행
  - 다른 페이지로 이동
  - 데이터 작업 호출
  - 사용자 지정 JavaScript 실행
  - 자동화 간접 호출

Expressions

테이블 구성 요소는 테이블의 기능과 상호 작용을 사용자 지정하고 향상시킬 수 있는 표현식과 행 수준 작업 기능을 사용할 수 있는 여러 영역을 제공합니다. 이를 통해 테이블 내에서 데이터를 동적으로 참조하고 표시할 수 있습니다. 이러한 표현식 필드를 활용하여 동적 열을 생성하고, 행 수준 작업에 데이터를 전달하고, 애플리케이션 내의 다른 구성 요소 또는 표현식에서 테이블 데이터를 참조할 수 있습니다.

예: 행 값 참조

`{{currentRow.columnName}}` 또는 `{{currentRow["Column Name"]}}` 이러한 표현식을 사용하면 렌더링되는 현재 행에 대한 특정 열의 값을 참조할 수 있습니다. `columnName` 또는 `# ###` 참조하려는 열의 실제 이름으로 바꿉니다.

예시:

- {{currentRow.productName}} 현재 행의 제품 이름을 표시합니다.
- {{currentRow["*Supplier Name*"]}} 현재 행의 공급업체 이름을 표시합니다. 여기서 열 헤더는 *#### ##*입니다.
- {{currentRow.orderDate}} 현재 행의 주문 날짜를 표시합니다.

예: 선택한 행 참조

{{ui.*table1*.selectedRow["columnName"]}}이 표현식을 사용하면 ID table*1# #### ### ## ## ### ## ## ## ## ## ### # ####*. *table1*을 테이블 구성 요소의 실제 ID로 바꾸고 *columnName*을 참조하려는 열의 이름으로 바꿉니다.

예시:

- {{ui.*ordersTable*.selectedRow["totalAmount"]}} ID *ordersTable*과 함께 테이블에서 현재 선택한 행의 총 금액을 표시합니다.
- {{ui.*customersTable*.selectedRow["email"]}} ID *customersTable*과 함께 테이블에서 현재 선택한 행의 이메일 주소를 표시합니다.
- {{ui.*employeesTable*.selectedRow["department"]}} 테이블에서 현재 선택한 행의 부서를 ID *employeesTable*과 함께 표시합니다.

예: 사용자 지정 열 생성

기본 데이터 작업, 자동화 또는 표현식에서 반환된 데이터를 기반으로 테이블에 사용자 지정 열을 추가할 수 있습니다. 기존 열 값과 JavaScript 표현식을 사용하여 새 열을 생성할 수 있습니다.

예시:

- {{currentRow.quantity * currentRow.unitPrice}} 수량 및 단위 가격 열을 곱하여 총 가격을 표시하는 새 열을 생성합니다.
- {{new Date(currentRow.orderDate).toLocaleDateString()}} 주문 날짜를 더 읽기 쉬운 형식으로 표시하는 새 열을 생성합니다.
- {{currentRow.firstName + ' ' + currentRow.lastName + ' (' + currentRow.email + ')' }} 각 행의 전체 이름과 이메일 주소를 표시하는 새 열을 생성합니다.

예: 열 표시 값 사용자 지정:

열 매핑의 필드를 설정하여 테이블 열 내 Value 필드의 표시 값을 사용자 지정할 수 있습니다. 이렇게 하면 표시된 데이터에 사용자 지정 형식 지정 또는 변환을 적용할 수 있습니다.

예시:

- {{ currentRow.rating >= 4 ? '##'.repeat(currentRow.rating) : currentRow.rating }} 각 행의 등급 값을 기반으로 별 이모티콘을 표시합니다.
- {{ currentRow.category.toLowerCase().replace(/\b\w/g, c => c.toUpperCase()) }} 각 행에 대해 대문자로 표시된 각 단어와 함께 범주 값을 표시합니다.
- {{ currentRow.status === 'Active' ? '# Active' : '# Inactive' }}: 각 행의 상태 값을 기반으로 색상이 지정된 원 이모티콘과 텍스트를 표시합니다.

행 수준 버튼 작업

{{currentRow.columnName}} 또는 이러한 표현식을 사용하여 선택한 행의 데이터로 다른 페이지로 이동하거나 행의 데이터로 자동화를 트리거하는 등 행 수준 작업 내에서 참조된 행의 컨텍스트를 전달할 {{currentRow["Column Name"]}} 수 있습니다.

예시:

- 행 작업 열에 편집 버튼이 있는 경우 파라미터{{currentRow.orderId}}로를 전달하여 선택한 주문의 ID가 있는 주문 편집 페이지로 이동할 수 있습니다.
- 행 작업 열에 삭제 버튼이 있는 경우 주문을 삭제{{currentRow.customerName}}하기 전에 고객에게 확인 이메일을 보내는 자동화에 전달할 수 있습니다.
- 행 작업 열에 세부 정보 보기 버튼이 있는 경우 주문 세부 정보를 본 직원을 로깅하는 {{currentRow.employeeId}} 자동화로 전달할 수 있습니다.

이러한 표현식 필드와 행 수준 작업 기능을 활용하여 특정 요구 사항에 따라 데이터를 표시하고 조작하는 고도로 사용자 지정된 대화형 테이블을 생성할 수 있습니다. 또한 행 수준 작업을 애플리케이션 내의 다른 구성 요소 또는 자동화와 연결하여 원활한 데이터 흐름과 기능을 구현할 수 있습니다.

세부 정보

세부 정보 구성 요소는 특정 레코드 또는 항목에 대한 세부 정보를 표시하도록 설계되었습니다. 단일 개체 또는 행과 관련된 포괄적인 데이터를 제공하는 전용 공간을 제공하므로 심층적인 세부 정보를 표시하거나 데이터 입력 및 편집 작업을 용이하게 하는 데 적합합니다.

세부 정보 속성

Detail 구성 요소는 , Name Source및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다Actions. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

세부 정보 구성 요소에는 , 및 FieldsLayout를 비롯한 특정 속성 및 구성 옵션도 있습니다Expressions.

레이아웃

레이아웃 섹션에서는 세부 정보 구성 요소 내의 필드 배열 및 표시를 사용자 지정할 수 있습니다. 다음과 같은 옵션을 구성할 수 있습니다.

- 열 수: 필드를 표시할 열 수를 지정합니다.
- 필드 순서 지정: 필드를 끌어서 놓아 모양을 재정렬합니다.
- 간격 및 정렬: 구성 요소 내의 필드 간격 및 정렬을 조정합니다.

표현식 및 예제

세부 정보 구성 요소는 구성 요소 내의 데이터를 동적으로 참조하고 표시할 수 있는 다양한 표현식 필드를 제공합니다. 이러한 표현식을 사용하면 애플리케이션의 데이터 및 로직과 원활하게 연결되는 사용자 지정 및 대화형 세부 정보 구성 요소를 생성할 수 있습니다.

예: 데이터 참조

`{{ui.details.data[0]?.["colName"]}}`:이 표현식을 사용하면 ID가 "details"인 Detail 구성 요소에 연결된 데이터 배열의 첫 번째 항목(인덱스 0)에 대해 "colName"이라는 열의 값을 참조할 수 있습니다. "colName"을 참조하려는 열의 실제 이름으로 바꿉니다. 예를 들어 다음 표현식은 "details" 구성 요소에 연결된 데이터 배열의 첫 번째 항목에 대한 "customerName" 열의 값을 표시합니다.

```
{{ui.details.data[0]?.["customerName"]}}
```

> **Note**
>
> 이 표현식은 세부 정보 구성 요소가 참조되는 테이블과 동일한 페이지에 있고 세부 정보 구성 요소에서 테이블의 첫 번째 행에 있는 데이터를 표시하려는 경우에 유용합니다.

예: 조건부 렌더링

`{{ui.table1.selectedRow["colName"]}}`:이 표현식은 ID `table1`이 있는 테이블에서 선택한 행에 `colName`이라는 열에 대한 데이터가 있는 경우 true를 반환합니다. 테이블에서 선택한 행이 비어 있는지 여부에 따라 세부 정보 구성 요소를 조건부로 표시하거나 숨기는 데 사용할 수 있습니다.

예제:

세부 정보 구성 요소의 `Visible if` 속성에서이 표현식을 사용하여 테이블에서 선택한 행을 기반으로 조건부로 표현식을 표시하거나 숨길 수 있습니다.

```
{{ui.table1.selectedRow["customerName"]}}
```

이 표현식이 true로 평가되면(`table1` 구성 요소에서 선택한 행에 `customerName` 열 값이 있음) 세부 정보 구성 요소가 표시됩니다. 표현식이 false로 평가되면(즉, 선택한 행이 비어 있거나 "customerName" 값이 없는 경우) 세부 정보 구성 요소가 숨겨집니다.

예: 조건부 표시

`{{(ui.Component.value === "green" ? "#" : ui.Component.value === "yellow" ? "#" : ui.detail1.data?.[0]?.CustomerStatus)}}`:이 표현식은 구성 요소 또는 데이터 필드의 값을 기반으로 이모티콘을 조건부로 표시합니다.

분류:

- `ui.Component.value`: ID 구성 요소가 있는 `## ###` 값을 참조합니다.
- `=== "green"`: 구성 요소의 값이 문자열 "green"과 같은지 확인합니다.
- `? "#"`: 조건이 true이면에 녹색 원 이모티콘이 표시됩니다.
- `: ui.Component.value === "yellow" ? "#"`: 첫 번째 조건이 false인 경우는 구성 요소의 값이 문자열 "yellow"와 같은지 확인합니다.
- `? "#"`: 두 번째 조건이 true이면에 노란색 사각형 이모티콘이 표시됩니다.
- `: ui.detail1.data?.[0]?.CustomerStatus`: 두 조건이 모두 false인 경우 세부 정보 구성 요소에 연결된 데이터 배열의 첫 번째 항목의 "CustomerStatus" 값을 ID "detail1"로 참조합니다.

이 표현식은 세부 정보 구성 요소 내의 구성 요소 또는 데이터 필드 값을 기반으로 이모티콘 또는 특정 값을 표시하는 데 사용할 수 있습니다.

Metrics

지표 구성 요소는 주요 지표 또는 데이터 포인트를 카드와 유사한 형식으로 표시하는 시각적 요소입니다. 중요한 정보 또는 성과 지표를 제시하는 간결하고 시각적으로 매력적인 방법을 제공하도록 설계되었습니다.

지표 속성

지표 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 일반적인 속성을 공유합니다`Actions`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

추세

지표의 추세 기능을 사용하면 표시되는 지표의 성능에 대한 시각적 지표를 표시하거나 시간 경과에 따라 변경할 수 있습니다.

추세 값

이 필드를 사용하면 추세 방향과 크기를 결정하는 데 사용해야 하는 값 또는 표현식을 지정할 수 있습니다. 일반적으로 이는 특정 기간 동안의 변경 또는 성능을 나타내는 값입니다.

예제:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}
```

이 표현식은 "salesMetrics" 지표에 연결된 데이터의 첫 번째 항목에서 month-over-month 수익 값을 검색합니다.

긍정적 추세

이 필드를 사용하면 양의 추세에 대한 조건을 정의하는 표현식을 입력할 수 있습니다. 표현식은 true 또는 false로 평가되어야 합니다.

예제:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}
```

이 표현식은 month-over-month 수익 값이 0보다 큰지 확인하여 긍정적인 추세를 나타냅니다.

부정적 추세

이 필드를 사용하면 부정적인 추세에 대한 조건을 정의하는 표현식을 입력할 수 있습니다. 표현식은 true 또는 false로 평가되어야 합니다.

예제:

```
{{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}
```

이 표현식은 month-over-month 수익 값이 0보다 작은지 확인하여 부정적인 추세를 나타냅니다.

색상 막대

이 토글을 사용하면 색상 막대 표시를 활성화하거나 비활성화하여 추세 상태를 시각적으로 표시할 수 있습니다.

색상 막대 예제:

예: 판매 지표 추세

- 추세 값: {{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue}}
- 긍정적 추세: {{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue > 0}}
- 부정적 추세: {{ui.salesMetrics.data?.[0]?.monthOverMonthRevenue < 0}}
- 색상 막대: 활성화됨

예: 인벤토리 지표 추세

- 추세 값: {{ui.inventoryMetrics.data?.[0]?.currentInventory - ui.inventoryMetrics.data?.[1]?.currentInventory}}
- 긍정적 추세: {{ui.inventoryMetrics.data?.[0]?.currentInventory > ui.inventoryMetrics.data?.[1]?.currentInventory}}
- 부정적 추세: {{ui.inventoryMetrics.data?.[0]?.currentInventory < ui.inventoryMetrics.data?.[1]?.currentInventory}}
- 색상 표시줄: 활성화됨

예: 고객 만족도 추세

- 추세 값: {{ui.customerSatisfactionMetrics.data?.[0]?.npsScore}}
- 긍정적 추세: {{ui.customerSatisfactionMetrics.data?.[0]?.npsScore >= 8}}
- 부정적 추세: {{ui.customerSatisfactionMetrics.data?.[0]?.npsScore < 7}}
- 색상 막대: 활성화됨

이러한 추세 관련 속성을 구성하여 표시되는 지표에 대한 성능을 시각적으로 표현하거나 시간 경과에 따라 변경하는 지표 구성 요소를 생성할 수 있습니다.

이러한 표현식을 활용하여 데이터를 동적으로 참조하고 표시하는 고도로 사용자 지정되고 대화형인 지표 구성 요소를 생성하여 애플리케이션 내에서 주요 지표, 성능 지표 및 데이터 기반 시각화를 표시할 수 있습니다.

지표 표현식 예제

속성 패널에서 표현식을 입력하여 제목, 기본 값, 보조 값 및 값 캡션을 표시하여 값을 동적으로 표시할 수 있습니다.

예: 기본 값 참조

`{{ui.metric1.primaryValue}}`:이 표현식을 사용하면 동일한 페이지 내의 다른 구성 요소 또는 표현식의 ID `metric1`을 사용하여 지표 구성 요소의 기본 값을 참조할 수 있습니다.

예: `{{ui.salesMetrics.primaryValue}}`는 `salesMetrics` 지표 구성 요소의 기본 값을 표시합니다.

예: 보조 값 참조

`{{ui.metric1.secondaryValue}}`:이 표현식을 사용하면 동일한 페이지 내의 다른 구성 요소 또는 표현식의 ID `metric1`을 사용하여 지표 구성 요소의 보조 값을 참조할 수 있습니다.

예: `{{ui.revenueMetrics.secondaryValue}}`는 `revenueMetrics` 지표 구성 요소의 보조 값을 표시합니다.

예: 데이터 참조

`{{ui.metric1.data}}`:이 표현식을 사용하면 동일한 페이지 내의 다른 구성 요소 또는 표현식에서 ID `metric1`을 사용하여 지표 구성 요소의 데이터를 참조할 수 있습니다.

예: `{{ui.kpiMetrics.data}}`는 `kpiMetrics` 지표 구성 요소에 연결된 데이터를 참조합니다.

예: 특정 데이터 값 표시:

`{{ui.metric1.data?.[0]?.id}}`:이 표현식은 ID metric1을 사용하여 지표 구성 요소에 연결된 데이터의 특정 정보를 표시하는 방법의 예입니다. 데이터에서 첫 번째 항목의 특정 속성을 표시하려는 경우에 유용합니다.

분류:

- `ui.metric1`: ID metric1을 사용하여 지표 구성 요소를 참조합니다.

- `data`: 해당 구성 요소에 연결된 정보 또는 데이터 세트를 나타냅니다.
- `?.[0]`: 해당 데이터 세트의 첫 번째 항목 또는 항목을 의미합니다.
- `?.id`: 첫 번째 항목 또는 항목의 *ID* 값 또는 식별자를 표시합니다.

예: `{{ui.`*orderMetrics*`.data?.[0]?.`*orderId*`}}`는 *orderMetrics* 지표 구성 요소에 연결된 데이터에 첫 번째 항목의 orderId *orderId* 값을 표시합니다.

예: 데이터 길이 표시

`{{ui.`*metric1*`.data?.length}}`:이 표현식은 ID metric*1*을 사용하여 지표 구성 요소에 연결된 데이터의 길이(항목 수)를 표시하는 방법을 보여줍니다. 데이터의 항목 수를 표시하려는 경우에 유용합니다.

분류:

- `ui.`*metric1*`.data`: 구성 요소에 연결된 데이터 세트를 참조합니다.
- `?.length`: 해당 데이터 세트의 총 항목 수 또는 항목 수에 액세스합니다.

예: `{{ui.`*productMetrics*`.data?.length}}`는 *productMetrics* 지표 구성 요소에 연결된 데이터의 항목 수를 표시합니다.

리피터

Repeater 구성 요소는 제공된 데이터 소스를 기반으로 요소 컬렉션을 생성하고 표시할 수 있는 동적 구성 요소입니다. 애플리케이션의 사용자 인터페이스 내에서 목록, 그리드 또는 반복 패턴을 쉽게 생성할 수 있도록 설계되었습니다. 몇 가지 사용 사례는 다음과 같습니다.

- 계정의 각 사용자에 대한 카드 표시
- 이미지와 장바구니에 추가할 버튼이 포함된 제품 목록 표시
- 사용자가 액세스할 수 있는 파일 목록 표시

Repeater 구성 요소는 콘텐츠가 풍부한 테이블 구성 요소와 구별됩니다. 테이블 구성 요소에는 엄격한 행 및 열 형식이 있습니다. 리피터는 데이터를 보다 유연하게 표시할 수 있습니다.

리피터 속성

Repeater 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다`Actions`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

일반적인 속성 외에도 Repeater 구성 요소에는 다음과 같은 추가 속성 및 구성 옵션이 있습니다.

항목 템플릿

항목 템플릿은 데이터 소스의 각 항목에 대해 반복될 구조와 구성 요소를 정의할 수 있는 컨테이너입니다. 텍스트, 이미지, 버튼 또는 각 항목을 나타내는 데 필요한 기타 구성 요소와 같은 다른 구성 요소를 이 컨테이너로 끌어서 놓을 수 있습니다.

항목 템플릿 내에서 형식의 표현식을 사용하여 현재 항목의 속성 또는 값을 참조할 수 있습니다`{{currentItem.propertyName}}`.

예를 들어 데이터 소스에 `itemName` 속성이 포함된 경우 `{{currentItem.itemName}}`를 사용하여 현재 항목의 항목 이름(들)을 표시할 수 있습니다.

레이아웃

레이아웃 섹션에서는 반복기 구성 요소 내에서 반복되는 요소의 배열을 구성할 수 있습니다.

Orientation(방향)

- 목록: 반복되는 요소를 단일 열에 세로로 정렬합니다.
- 그리드: 반복되는 요소를 여러 열이 있는 그리드 레이아웃에 정렬합니다.

페이지당 행

목록 레이아웃의 페이지당 표시할 행 수를 지정합니다. 지정된 수의 행을 오버플로하는 항목에 대해 페이지 매김이 제공됩니다.

페이지당 열 및 행(그리드)

- 열: 그리드 레이아웃의 열 수를 지정합니다.
- 페이지당 행: 그리드 레이아웃의 페이지당 표시할 행 수를 지정합니다. 지정된 그리드 차원을 오버플로하는 항목에 대해 페이지 매김이 제공됩니다.

표현식 및 예제

Repeater 구성 요소는 구성 요소 내의 데이터를 동적으로 참조하고 표시할 수 있는 다양한 표현식 필드를 제공합니다. 이러한 표현식을 사용하면 애플리케이션의 데이터 및 로직과 원활하게 연결되는 사용자 지정 대화형 리피터 구성 요소를 생성할 수 있습니다.

예: 항목 참조

- {{currentItem.propertyName}}: 항목 템플릿 내에서 현재 항목의 속성 또는 값을 참조합니다.
- {{ui.repeaterID[index]}}: 인덱스별로 리피터 구성 요소의 특정 항목을 참조합니다.

예: 제품 목록 렌더링

- 소스: ## 개체를 데이터 소스로 선택합니다.
- 항목 템플릿: 안에 텍스트 구성 요소가 있는 컨테이너 구성 요소를 추가하여 제품 이름({{currentItem.productName}})을 표시하고 이미지 구성 요소를 추가하여 제품 이미지를 표시합니다({{currentItem.productImageUrl}}).
- 레이아웃:를 Orientation 로 설정하고 원하는 Rows per Page 대로를 List 조정합니다.

예: 사용자 아바타 그리드 생성

- 소스: 표현식을 사용하여 사용자 데이터 배열(예: [{name: 'John', avatarUrl: '...'}, {...}, {...}])을 생성합니다.
- 항목 템플릿: 이미지 구성 요소를 추가하고 Source 속성을 로 설정합니다{{currentItem.avatarUrl}}.
- 레이아웃:를 Orientation로 설정하고Grid, Columns 및의 수를 지정하고Rows per Page, 필요에 Padding 따라 Space Between 및를 조정합니다.

Repeater 구성 요소를 사용하면 동적 데이터 기반 사용자 인터페이스를 생성하여 요소 컬렉션을 렌더링하는 프로세스를 간소화하고 수동 반복 또는 하드 코딩의 필요성을 줄일 수 있습니다.

양식

양식 구성 요소는 사용자 입력을 캡처하고 애플리케이션 내에서 데이터 입력 작업을 용이하게 하도록 설계되었습니다. 입력 필드, 드롭다운, 확인란 및 기타 양식 제어를 표시하는 구조화된 레이아웃을 제공하므로 사용자가 데이터를 원활하게 입력하거나 수정할 수 있습니다. 테이블과 같은 양식 구성 요소 내부에 다른 구성 요소를 중첩할 수 있습니다.

양식 속성

양식 구성 요소는 , Name Source및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다Actions. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

양식 생성

양식 생성 기능을 사용하면 선택한 데이터 소스를 기반으로 양식 필드를 자동으로 채워 빠르게 생성할 수 있습니다. 이렇게 하면 많은 수의 필드를 표시해야 하는 양식을 작성할 때 시간과 노력을 절약할 수 있습니다.

양식 생성 기능을 사용하려면:

1. 양식 구성 요소의 속성에서 양식 생성 섹션을 찾습니다.
2. 양식 필드를 생성하는 데 사용할 데이터 소스를 선택합니다. 이는 애플리케이션에서 사용할 수 있는 엔터티, 워크플로 또는 기타 데이터 소스일 수 있습니다.
3. 양식 필드는 필드 레이블, 유형 및 데이터 매핑을 포함하여 선택한 데이터 소스를 기반으로 자동으로 생성됩니다.
4. 생성된 필드를 검토하고 검증 규칙 추가 또는 필드 순서 변경과 같은 필요한 사용자 지정을 수행합니다.
5. 양식 구성에 만족하면 제출을 선택하여 생성된 필드를 양식 구성 요소에 적용합니다.

양식 생성 기능은 애플리케이션에 사용자 입력을 캡처해야 하는 잘 정의된 데이터 모델 또는 엔터티 세트가 있는 경우 특히 유용합니다. 양식 필드를 자동으로 생성하면 시간을 절약하고 애플리케이션 양식 간의 일관성을 보장할 수 있습니다.

양식 생성 기능을 사용한 후 특정 요구 사항에 맞게 양식 구성 요소의 레이아웃, 작업 및 표현식을 추가로 사용자 지정할 수 있습니다.

표현식 및 예제

다른 구성 요소와 마찬가지로 표현식을 사용하여 양식 구성 요소 내의 데이터를 참조하고 표시할 수 있습니다. 예제:

- `{{ui.userForm.data.email}}`: ID가 인 양식 구성 요소에 연결된 데이터 소스의 `email` 필드 값을 참조합니다`userForm`.

> **Note**
>
> 공통 속성에 공통 구성 요소 속성 대한 자세한 표현식 예제는 섹션을 참조하세요.

이러한 속성을 구성하고 표현식을 활용하면 애플리케이션의 데이터 소스 및 로직과 원활하게 통합되는 사용자 지정 대화형 양식 구성 요소를 생성할 수 있습니다. 이러한 구성 요소를 사용하여 사용자 입력을 캡처하고, 미리 채워진 데이터를 표시하고, 양식 제출 또는 사용자 상호 작용을 기반으로 작업을 트리거할 수 있습니다.

Stepflow

Stepflow 구성 요소는 애플리케이션 내의 다단계 프로세스 또는 워크플로를 통해 사용자를 안내하도록 설계되었습니다. 각 단계마다 고유한 입력, 검증 및 작업 세트를 제공하는 구조화되고 직관적인 인터페이스를 제공합니다.

Stepflow 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다`Actions`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

Stepflow 구성 요소에는 , 및와 같은 추가 속성 `Step` `Navigation` `Validation`및 구성 옵션이 있습니다`Expressions`.

AI 구성 요소

AI 생성

Gen AI 구성 요소는 애플리케이션 스튜디오 내에서 채팅을 사용하여 AI를 사용하여 구성 요소와 함께 제공되는 로직을 그룹화하는 데 사용되는 그룹화 컨테이너입니다. 채팅을 사용하여 구성 요소를 생성하면 구성 요소가 Gen AI 컨테이너로 그룹화됩니다. 이 구성 요소의 편집 또는 사용에 대한 자세한 내용은 섹션을 참조하세요앱 빌드 또는 편집.

텍스트 및 숫자 구성 요소

텍스트 입력

텍스트 입력 구성 요소를 사용하면 사용자가 애플리케이션 내에서 텍스트 데이터를 입력하고 제출할 수 있습니다. 이름, 주소 또는 기타 텍스트 정보와 같은 사용자 입력을 캡처하는 간단하고 직관적인 방법을 제공합니다.

- `{{ui.inputTextID.value}}`: 입력 필드에 제공된 값을 반환합니다.
- `{{ui.inputTextID.isValid}}`: 입력 필드에 제공된 값의 유효성을 반환합니다.

텍스트

텍스트 구성 요소는 애플리케이션 내에서 텍스트 정보를 표시하는 데 사용됩니다. 표현식에서 생성된 정적 텍스트, 동적 값 또는 콘텐츠를 표시하는 데 사용할 수 있습니다.

텍스트 영역

텍스트 영역 구성 요소는 사용자의 여러 줄 텍스트 입력을 캡처하도록 설계되었습니다. 사용자가 설명, 메모 또는 설명과 같은 더 긴 텍스트 항목을 입력할 수 있도록 더 큰 입력 필드 영역을 제공합니다.

- {{ui.*textAreaID*.value}}: 텍스트 영역에 제공된 값을 반환합니다.
- {{ui.*textAreaID*.isValid}}: 텍스트 영역에서 제공된 값의 유효성을 반환합니다.

이메일

이메일 구성 요소는 사용자의 이메일 주소를 캡처하도록 설계된 특수 입력 필드입니다. 입력한 값이 올바른 이메일 형식을 준수하는지 확인하기 위해 특정 검증 규칙을 적용할 수 있습니다.

- {{ui.*emailID*.value}}: 이메일 입력 필드에 제공된 값을 반환합니다.
- {{ui.*emailID*.isValid}}: 이메일 입력 필드에 제공된 값의 유효성을 반환합니다.

암호

암호 구성 요소는 사용자가 암호 또는 PIN 코드와 같은 민감한 정보를 입력하도록 특별히 설계된 입력 필드입니다. 입력한 문자를 마스킹 처리하여 프라이버시와 보안을 유지합니다.

- {{ui.*passwordID*.value}}: 암호 입력 필드에 제공된 값을 반환합니다.
- {{ui.*passwordID*.isValid}}: 암호 입력 필드에 제공된 값의 유효성을 반환합니다.

검색

검색 구성 요소는 애플리케이션 내에서 채워진 데이터 내에 검색 쿼리를 수행하거나 검색어를 입력하기 위한 전용 입력 필드를 사용자에게 제공합니다.

- {{ui.*searchID*.value}}: 검색 필드에 제공된 값을 반환합니다.

전화번호

전화 구성 요소는 사용자의 전화번호 또는 기타 연락처 정보를 캡처하도록 조정된 입력 필드입니다. 입력한 값이 올바른 전화번호 형식을 준수하는지 확인하기 위해 특정 검증 규칙 및 형식 지정 옵션을 포함할 수 있습니다.

- {{ui.*phoneID*.value}}: 전화 입력 필드에 제공된 값을 반환합니다.

- {{ui.*phoneID*.isValid}}: 전화 입력 필드에 제공된 값의 유효성을 반환합니다.

숫자

숫자 구성 요소는 사용자가 숫자 값을 입력할 수 있도록 특별히 설계된 입력 필드입니다. 입력한 값이 지정된 범위 또는 형식 내의 유효한 숫자인지 확인하기 위해 검증 규칙을 적용할 수 있습니다.

- {{ui.*numberID*.value}}: 숫자 입력 필드에 제공된 값을 반환합니다.
- {{ui.*numberID*.isValid}}: 숫자 입력 필드에 제공된 값의 유효성을 반환합니다.

통화

통화 구성 요소는 통화 값 또는 금액을 캡처하기 위한 특수 입력 필드입니다. 통화 기호, 십진수 구분자를 표시하고 통화 입력과 관련된 검증 규칙을 적용하는 형식 지정 옵션을 포함할 수 있습니다.

- {{ui.*currencyID*.value}}: 통화 입력 필드에 제공된 값을 반환합니다.
- {{ui.*currencyID*.isValid}}: 통화 입력 필드에 제공된 값의 유효성을 반환합니다.

세부 정보 페어

세부 정보 페어 구성 요소는 키-값 페어 또는 관련 정보 페어를 구조화되고 읽기 가능한 형식으로 표시하는 데 사용됩니다. 일반적으로 특정 항목 또는 개체와 연결된 세부 정보 또는 메타데이터를 제공하는 데 사용됩니다.

선택 구성 요소

스위치

전환 구성 요소는 사용자가 켜기/끄기, true/false 또는 활성화/비활성화와 같은 두 상태 또는 옵션을 전환할 수 있는 사용자 인터페이스 제어입니다. 현재 상태를 시각적으로 표시하며 사용자가 클릭 또는 탭 한 번으로 변경할 수 있습니다.

그룹 전환

스위치 그룹 구성 요소는 사용자가 사전 정의된 세트에서 하나 이상의 옵션을 선택할 수 있는 개별 스위치 제어 모음입니다. 선택한 옵션과 선택하지 않은 옵션을 시각적으로 표시하므로 사용자가 사용 가능한 옵션을 더 쉽게 이해하고 상호 작용할 수 있습니다.

그룹 표현식 필드 전환

- {{ui.switchGroupID.value}}: 앱 사용자가 활성화한 각 스위치의 값을 포함하는 문자열 배열을 반환합니다.

확인란 그룹

확인란 그룹 구성 요소는 사용자에게 여러 옵션을 동시에 선택할 수 있는 확인란 그룹을 제공합니다. 사용자에게 옵션 목록에서 하나 이상의 항목을 선택할 수 있는 기능을 제공하려는 경우에 유용합니다.

확인란 그룹 표현식 필드

- {{ui.checkboxGroupID.value}}: 앱 사용자가 선택한 각 확인란의 값을 포함하는 문자열 배열을 반환합니다.

라디오 그룹

라디오 그룹 구성 요소는 사용자가 여러 상호 배타적인 선택 항목 중에서 단일 옵션을 선택할 수 있는 라디오 버튼 세트입니다. 한 번에 하나의 옵션만 선택할 수 있으므로 사용자가 명확하고 모호하지 않게 선택할 수 있습니다.

라디오 그룹 표현식 필드

다음 필드를 표현식에 사용할 수 있습니다.

- {{ui.radioGroupID.value}}: 앱 사용자가 선택한 라디오 버튼의 값을 반환합니다.

단일 선택

단일 선택 구성 요소는 사용자에게 단일 항목을 선택할 수 있는 옵션 목록을 제공합니다. 일반적으로 사용자가 범주, 위치 또는 기본 설정 선택과 같은 사전 정의된 옵션 세트 중에서 선택해야 하는 시나리오에서 사용됩니다.

단일 선택 표현식 필드

- {{ui.singleSelectID.value}}: 앱 사용자가 선택한 목록 항목의 값을 반환합니다.

다중 선택

다중 선택 구성 요소는 단일 선택 구성 요소와 유사하지만 사용자가 선택 목록에서 여러 옵션을 동시에 선택할 수 있습니다. 사용자가 여러 태그, 관심사 또는 기본 설정 선택과 같이 미리 정의된 옵션 세트에서 여러 항목을 선택해야 하는 경우에 유용합니다.

다중 선택 표현식 필드

- `{{ui.multiSelectID.value}}`: 앱 사용자가 선택한 각 목록 항목의 값을 포함하는 문자열 배열을 반환합니다.

버튼 및 탐색 구성 요소

애플리케이션 스튜디오는 사용자가 작업을 트리거하고 애플리케이션 내에서 탐색할 수 있도록 다양한 버튼 및 탐색 구성 요소를 제공합니다.

버튼 구성 요소

사용 가능한 버튼 구성 요소는 다음과 같습니다.

- Button
- 개요 버튼
- 아이콘 버튼
- 텍스트 버튼

이러한 버튼 구성 요소는 다음과 같은 공통 속성을 공유합니다.

내용

- 버튼 레이블: 버튼에 표시할 텍스트입니다.

Type

- 버튼: 표준 버튼입니다.
- 개요: 요약된 스타일의 버튼입니다.
- 아이콘: 아이콘이 있는 버튼입니다.
- 텍스트: 텍스트 전용 버튼입니다.

Size:

버튼의 크기입니다. 가능한 값은 `Small`, `Medium` 및 `Large`입니다.

아이콘

버튼에 표시할 다음과 같은 다양한 아이콘 중에서 선택할 수 있습니다.

- 봉투 닫힘
- Bell
- Person
- 햄버거 메뉴
- 검색
- 동그라미로 표시된 정보
- 기어
- 왼쪽 셰브론
- 오른쪽 셰브론
- 가로 점
- Trash
- 편집
- Check]를 선택합니다
- Close
- 홈
- Plus

트리거

버튼을 클릭하면 트리거할 작업을 하나 이상 구성할 수 있습니다. 사용 가능한 작업 유형은 다음과 같습니다.

- 기본
  - 구성 요소 작업 실행: 구성 요소 내에서 특정 작업을 실행합니다.
  - 탐색: 다른 페이지 또는 보기로 이동합니다.
  - 데이터 작업 호출: 레코드 생성, 업데이트 또는 삭제와 같은 데이터 관련 작업을 트리거합니다.

- 고급
  - JavaScript: 사용자 지정 JavaScript 코드를 실행합니다.
  - 자동화 호출: 기존 자동화 또는 워크플로를 시작합니다.

JavaScript 작업 버튼 속성

버튼을 클릭할 때 사용자 지정 JavaScript 코드를 실행할 `JavaScript` 작업 유형을 선택합니다.

소스 코드

`Source code` 필드에 JavaScript 표현식 또는 함수를 입력할 수 있습니다. 예제:

```
return "Hello World";
```

이렇게 하면 버튼을 클릭할 `Hello World` 때 문자열만 반환됩니다.

조건: 다음 경우에 실행

JavaScript 작업을 실행할지 여부를 결정하는 부울 표현식을 제공할 수도 있습니다. 이 방식은 다음 구문을 사용합니다.

```
{{ui.textinput1.value !== ""}}
```

이 예제에서 JavaScript 작업은 `textinput1` 구성 요소의 값이 빈 문자열이 아닌 경우에만 실행됩니다.

이러한 고급 트리거 옵션을 사용하면 애플리케이션의 로직 및 데이터와 직접 통합되는 고도로 사용자 지정된 버튼 동작을 생성할 수 있습니다. 이를 통해 버튼의 내장 기능을 확장하고 사용자 경험을 특정 요구 사항에 맞게 조정할 수 있습니다.

> **Note**
>
> JavaScript 작업이 예상대로 작동하는지 항상 철저하게 테스트합니다.

하이퍼링크

Hyperlink 구성 요소는 외부 URLs 또는 내부 애플리케이션 경로로 이동할 수 있는 클릭 가능한 링크를 제공합니다.

하이퍼링크 속성

내용

- 하이퍼링크 레이블: 하이퍼링크 레이블로 표시할 텍스트입니다.

URL

외부 웹 사이트 또는 내부 애플리케이션 경로일 수 있는 하이퍼링크의 대상 URL입니다.

트리거

하이퍼링크를 클릭하면 트리거할 작업을 하나 이상 구성할 수 있습니다. 사용 가능한 작업 유형은 버튼 구성 요소의 작업 유형과 동일합니다.

날짜 및 시간 구성 요소

Date

날짜 구성 요소를 사용하면 사용자가 날짜를 선택하고 입력할 수 있습니다.

날짜 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다`Validation`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

공통 속성 외에도 날짜 구성 요소에는 다음과 같은 특정 속성이 있습니다.

날짜 속성

형식

- YYYY/MM/DD, DD/MM/YYYY, YYYY/MM/DD, YYYY/DD/MM, MM/DD, DD/MM: 날짜를 표시해야 하는 형식입니다.

값

- YYYY-MM-DD: 날짜 값이 내부적으로 저장되는 형식입니다.

최소 날짜

- YYYY-MM-DD: 선택할 수 있는 최소 날짜입니다.

> **Note**
>
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

최대 날짜

- YYYY-MM-DD: 선택할 수 있는 최대 날짜입니다.

> **Note**
>
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

일정 유형

- 1개월, 2개월: 표시할 일정 UI의 유형입니다.

비활성화된 날짜

- 소스: 비활성화해야 하는 날짜의 데이터 소스입니다. 예: 없음, 표현식.
- 비활성화된 날짜: 다음과 같이 비활성화할 날짜를 결정하는 표현식입니다.
  - `{{currentRow.column}}`:이 표현식이 평가되는 것과 일치하는 날짜를 비활성화합니다.
  - `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}`: 2023년 1월 1일 이전의 날짜 비활성화
  - `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}`: 주말을 비활성화합니다.

동작

- Visible if: 날짜 구성 요소의 가시성을 결정하는 표현식입니다.
- Disable if: 날짜 구성 요소를 비활성화할지 여부를 결정하는 표현식입니다.

검증

검증 섹션에서는 날짜 입력에 대한 추가 규칙 및 제약 조건을 정의할 수 있습니다. 이러한 검증 규칙을 구성하면 사용자가 입력한 날짜 값이 애플리케이션의 특정 요구 사항을 충족하는지 확인할 수 있습니다. 다음과 같은 유형의 검증을 추가할 수 있습니다.

- 필수:이 토글을 사용하면 사용자가 양식을 제출하기 전에 날짜 값을 입력해야 합니다.
- 사용자 지정: JavaScript 표현식을 사용하여 사용자 지정 검증 규칙을 생성할 수 있습니다. 예제:

  ```
  {{new Date(ui.dateInput.value) < new Date("2023-01-01")}}
  ```

  이 표현식은 입력한 날짜가 2023년 1월 1일 이전인지 확인합니다. 조건이 true이면 검증이 실패합니다.

  검증이 충족되지 않을 때 표시할 사용자 지정 검증 메시지를 제공할 수도 있습니다.

  ```
  "Validation not met. The date must be on or after January 1, 2023."
  ```

이러한 검증 규칙을 구성하면 사용자가 입력한 날짜 값이 애플리케이션의 특정 요구 사항을 충족하는지 확인할 수 있습니다.

표현식 및 예제

날짜 구성 요소는 다음 표현식 필드를 제공합니다.

- `{{ui.dateID.value}}`: 사용자가 입력한 날짜 값을 형식으로 반환합니다`YYYY-MM-DD`.

Time

시간 구성 요소를 사용하면 사용자가 시간 값을 선택하고 입력할 수 있습니다. 시간 구성 요소의 다양한 속성을 구성하면 선택 가능한 시간 범위 제한, 특정 시간 비활성화, 구성 요소의 가시성 및 상호 작용 제어와 같은 애플리케이션의 특정 요구 사항을 충족하는 시간 입력 필드를 생성할 수 있습니다.

시간 속성

시간 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다`Validation`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

일반 속성 외에도 시간 구성 요소에는 다음과 같은 특정 속성이 있습니다.

시간 간격

- 5분, 10분, 15분, 20분, 25분, 30분, 60분: 시간을 선택하는 데 사용할 수 있는 간격입니다.

값

- HH:MM AA: 시간 값이 내부적으로 저장되는 형식입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`HH:MM AA`.

Placeholder

- 일정 설정: 시간 필드가 비어 있을 때 표시되는 자리 표시자 텍스트입니다.

최소 시간

- HH:MM AA: 선택할 수 있는 최소 시간입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`HH:MM AA`.

최대 시간

- HH:MM AA: 선택할 수 있는 최대 시간입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`HH:MM AA`.

비활성화된 시간

- 소스: 비활성화해야 하는 시간(예: 없음, 표현식)의 데이터 소스입니다.

- 비활성화된 시간:와 같이 비활성화해야 하는 시간을 결정하는 표현식입니
  다{{currentRow.column}}.

비활성화된 시간 구성

비활성화된 시간 섹션을 사용하여 선택할 수 없는 시간 값을 지정할 수 있습니다.

소스

- 없음: 비활성화된 시간은 없습니다.
- 표현식: JavaScript 표현식을 사용하여와 같이 비활성화해야 하는 시간을 결정할 수 있습니
  다{{currentRow.column}}.

표현식 예:

```
{{currentRow.column === "Lunch Break"}}
```

이 표현식은 현재 행에 대해 "런치 브레이크" 열이 true인 경우 언제든지 비활성화됩니다.

이러한 검증 규칙과 비활성화된 시간 표현식을 구성하면 사용자가 입력한 시간 값이 애플리케이션의 특정 요구 사항을 충족하는지 확인할 수 있습니다.

동작

- Visible if: 시간 구성 요소의 가시성을 결정하는 표현식입니다.
- Disable if: 시간 구성 요소를 비활성화할지 여부를 결정하는 표현식입니다.

검증

- 필수: 사용자가 양식을 제출하기 전에 시간 값을 입력해야 하는 토글입니다.
- 사용자 지정: JavaScript 표현식을 사용하여 사용자 지정 검증 규칙을 생성할 수 있습니다.

  사용자 지정 검증 메시지: 사용자 지정 검증이 충족되지 않을 때 표시되는 메시지입니다.

예제:

```
{{ui.timeInput.value === "09:00 AM" || ui.timeInput.value === "09:30 AM"}}
```

이 표현식은 입력한 시간이 오전 9시 또는 오전 9시 30분인지 확인합니다. 조건이 true이면 검증이 실패합니다.

검증이 충족되지 않을 때 표시할 사용자 지정 검증 메시지를 제공할 수도 있습니다.

```
Validation not met. The time must be 9:00 AM or 9:30 AM.
```

표현식 및 예제

시간 구성 요소는 다음 표현식 필드를 제공합니다.

- `{{ui.timeID.value}}`: 사용자가 입력한 시간 값을 HH:MM AA 형식으로 반환합니다.

예: 시간 값

- `{{ui.timeID.value}}`: 사용자가 입력한 시간 값을 형식으로 반환합니다`HH:MM AA`.

예: 시간 비교

- `{{ui.timeInput.value > "10:00 AM"}}`: 시간 값이 오전 10시보다 큰지 확인합니다.
- `{{ui.timeInput.value < "05:00 pM"}}`: 시간 값이 오후 5시 미만인지 확인합니다.

날짜 범위

날짜 범위 구성 요소를 사용하면 사용자가 날짜 범위를 선택하고 입력할 수 있습니다. 날짜 범위 구성 요소의 다양한 속성을 구성하면 선택 가능한 날짜 범위 제한, 특정 날짜 비활성화, 구성 요소의 가시성 및 상호 작용 제어와 같은 애플리케이션의 특정 요구 사항을 충족하는 날짜 범위 입력 필드를 생성할 수 있습니다.

날짜 범위 속성

날짜 범위 구성 요소는 , `Name` `Source`및와 같은 다른 구성 요소와 몇 가지 공통 속성을 공유합니다`Validation`. 이러한 속성에 대한 자세한 내용은 섹션을 참조하세요공통 구성 요소 속성.

공통 속성 외에도 날짜 범위 구성 요소에는 다음과 같은 특정 속성이 있습니다.

형식

- MM/DD/YYYY: 날짜 범위를 표시해야 하는 형식입니다.

시작일

- YYYY-MM-DD: 범위의 시작으로 선택할 수 있는 최소 날짜입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

종료일

- YYYY-MM-DD: 범위의 끝으로 선택할 수 있는 최대 날짜입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

Placeholder

- 일정 설정: 날짜 범위 필드가 비어 있을 때 표시되는 자리 표시자 텍스트입니다.

최소 날짜

- YYYY-MM-DD: 선택할 수 있는 최소 날짜입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

최대 날짜

- YYYY-MM-DD: 선택할 수 있는 최대 날짜입니다.

> **Note**
> 이 값은 형식과 일치해야 합니다`YYYY-MM-DD`.

일정 유형

- 1개월: 표시할 일정 UI의 유형입니다. 예를 들어 한 달입니다.
- 2개월: 표시할 일정 UI의 유형입니다. 예: 2개월.

필수 일 선택됨

- 0: 날짜 범위 내에서 선택해야 하는 필수 일수입니다.

비활성화된 날짜

- 소스: 비활성화해야 하는 날짜의 데이터 소스입니다(예: 없음, 표현식, 개체 또는 자동화).
- 비활성화된 날짜:와 같이 비활성화해야 하는 날짜를 결정하는 표현식입니다`{{currentRow.column}}`.

검증

검증 섹션에서는 날짜 범위 입력에 대한 추가 규칙 및 제약 조건을 정의할 수 있습니다.

표현식 및 예제

날짜 범위 구성 요소는 다음 표현식 필드를 제공합니다.

- `{{ui.dateRangeID.startDate}}`: 선택한 범위의 시작 날짜를 형식으로 반환합니다`YYYY-MM-DD`.
- `{{ui.dateRangeID.endDate}}`: 선택한 범위의 종료 날짜를 형식으로 반환합니다`YYYY-MM-DD`.

예: 날짜 차이 계산

- `{(new Date(ui.dateRangeID.endDate) - new Date(ui.dateRangeID.startDate)) / (1000 * 60 * 60 * 24)}}` 시작 날짜와 종료 날짜 사이의 일수를 계산합니다.

예: 날짜 범위에 따른 조건부 가시성

- `{{new Date(ui.dateRangeID.startDate) < new Date("2023-01-01") || new Date(ui.dateRangeID.endDate) > new Date("2023-12-31")}}` 선택한 날짜 범위가 2023년을 벗어났는지 확인합니다.

예: 현재 행 데이터를 기반으로 비활성화된 날짜

- `{{currentRow.isHoliday}}` 현재 행의 "isHoliday" 열이 true인 날짜를 비활성화합니다.
- `{{new Date(currentRow.dateColumn) < new Date("2023-01-01")}}` 현재 행의 "dateColumn"을 기반으로 2023년 1월 1일 이전의 날짜를 비활성화합니다.
- `{{new Date(currentRow.dateColumn).getDay() === 0 || new Date(currentRow.dateColumn).getDay() === 6}}` 현재 행의 "dateColumn"을 기반으로 주말을 비활성화합니다.

사용자 지정 검증

- `{{new Date(ui.dateRangeID.startDate) > new Date(ui.dateRangeID.endDate)}}` 시작 날짜가 종료 날짜보다 이후인지 확인하여 사용자 지정 검증에 실패합니다.

미디어 구성 요소

애플리케이션 스튜디오는 애플리케이션 내에 다양한 미디어 유형을 임베딩하고 표시하기 위한 여러 구성 요소를 제공합니다.

iFrame 임베드

iFrame 임베드 구성 요소를 사용하면 iFrame을 사용하여 애플리케이션 내에 외부 웹 콘텐츠 또는 애플리케이션을 임베드할 수 있습니다.

iFrame 임베드 속성

URL

> **Note**
>
> 이 구성 요소에 표시되는 미디어의 소스는 애플리케이션의 콘텐츠 보안 설정에서 허용되어야 합니다. 자세한 내용은 앱의 콘텐츠 보안 설정 보기 또는 업데이트 단원을 참조하십시오.

임베드하려는 외부 콘텐츠 또는 애플리케이션의 URL입니다.

레이아웃

- 너비: 백분율(%) 또는 고정 픽셀 값(예: 300px)으로 지정된 iFrame의 너비입니다.

- 높이: 백분율(%) 또는 고정 픽셀 값으로 지정된 iFrame의 높이입니다.

S3 업로드

S3 업로드 구성 요소를 사용하면 사용자가 Amazon S3 버킷에 파일을 업로드할 수 있습니다. S3 업로드 구성 요소를 구성하면 사용자가 애플리케이션의 Amazon S3 스토리지에 파일을 쉽게 업로드한 다음 애플리케이션의 로직 및 사용자 인터페이스 내에서 업로드된 파일 정보를 활용할 수 있습니다.

> **Note**
>
> 애플리케이션의 파일 업로드 및 스토리지 요구 사항을 지원하는 데 필요한 권한과 Amazon S3 버킷 구성이 마련되어 있는지 확인해야 합니다.

S3 업로드 속성

S3 구성

- 커넥터: 파일 업로드에 사용할 사전 구성된 Amazon S3 커넥터를 선택합니다.
- 버킷: 파일이 업로드될 Amazon S3 버킷입니다.
- 폴더: 파일이 저장될 Amazon S3 버킷 내의 폴더입니다.
- 파일 이름: 업로드된 파일의 이름 지정 규칙입니다.

파일 업로드 구성

- 레이블: 파일 업로드 영역 위에 표시되는 레이블 또는 지침입니다.
- 설명: 파일 업로드에 대한 추가 지침 또는 정보입니다.
- 파일 유형: 업로드할 수 있는 파일 유형입니다. 예: 이미지, 문서 또는 비디오.
- 크기: 업로드할 수 있는 개별 파일의 최대 크기입니다.
- 버튼 레이블: 파일 선택 버튼에 표시되는 텍스트입니다.
- 버튼 스타일: 파일 선택 버튼의 스타일입니다. 예를 들어, 개괄적으로 설명되거나 채워집니다.
- 버튼 크기: 파일 선택 버튼의 크기입니다.

검증

- 최대 파일 수: 한 번에 업로드할 수 있는 최대 파일 수입니다.

- 최대 파일 크기: 각 개별 파일에 허용되는 최대 크기입니다.

트리거

- 성공 시: 파일 업로드가 성공할 때 트리거되는 작업입니다.
- 실패 시: 파일 업로드가 실패할 때 트리거되는 작업입니다.

S3 업로드 표현식 필드

S3 업로드 구성 요소는 다음 표현식 필드를 제공합니다.

- `{{ui.s3uploadID.files}}`: 업로드된 파일의 배열을 반환합니다.
- `{{ui.s3uploadID.files[0]?.size}}`: 지정된 인덱스에서 파일의 크기를 반환합니다.
- `{{ui.s3uploadID.files[0]?.type}}`: 지정된 인덱스에서 파일 유형을 반환합니다.
- `{{ui.s3uploadID.files[0]?.nameOnly}}`: 지정된 인덱스에서 확장 접미사가 없는 파일 이름을 반환합니다.
- `{{ui.s3uploadID.files[0]?.nameWithExtension}}`: 지정된 인덱스에 확장 접미사가 있는 파일의 이름을 반환합니다.

표현식 및 예제

예: 업로드된 파일 액세스

- `{{ui.s3uploadID.files.length}}`: 업로드된 파일 수를 반환합니다.
- `{{ui.s3uploadID.files.map(f => f.name).join(', ')}}`: 업로드된 파일 이름의 쉼표로 구분된 목록을 반환합니다.
- `{{ui.s3uploadID.files.filter(f => f.type.startsWith('image/'))}}`: 업로드된 이미지 파일의 배열만 반환합니다.

예: 파일 업로드 검증

- `{{ui.s3uploadID.files.some(f => f.size > 5 * 1024 * 1024)}}`: 업로드된 파일의 크기가 5MB를 초과하는지 확인합니다.
- `{{ui.s3uploadID.files.every(f => f.type === 'image/png')}}`: 업로드된 모든 파일이 PNG 이미지인지 확인합니다.
- `{{ui.s3uploadID.files.length > 3}}`: 3개 이상의 파일이 업로드되었는지 확인합니다.

예: 작업 트리거

- `{{ui.s3uploadID.files.length > 0 ? 'Upload Successful' : 'No files uploaded'}}`: 하나 이상의 파일이 업로드된 경우 성공 메시지를 표시합니다.
- `{{ui.s3uploadID.files.some(f => f.type.startsWith('video/')) ? triggerVideoProcessing() : null}}`: 비디오 파일이 업로드된 경우 비디오 처리 자동화를 트리거합니다.
- `{{ui.s3uploadID.files.map(f => f.url)}}`: 업로드된 파일의 URLs을 검색하여 파일을 표시하거나 추가로 처리하는 데 사용할 수 있습니다.

이러한 표현식을 사용하면 업로드된 파일에 액세스하고, 파일 업로드를 검증하고, 파일 업로드 결과를 기반으로 작업을 트리거할 수 있습니다. 이러한 표현식을 활용하면 애플리케이션의 파일 업로드 기능 내에서 더 역동적이고 지능적인 동작을 만들 수 있습니다.

> **Note**
>
> *s3uploadID*를 S3 업로드 구성 요소의 ID로 바꿉니다.

PDF 뷰어 구성 요소

PDF 뷰어 구성 요소를 사용하면 사용자가 애플리케이션 내에서 PDF 문서를 보고 상호 작용할 수 있습니다. App Studio는 PDF 소스에 대해 이러한 다양한 입력 유형을 지원하며, PDF 뷰어 구성 요소는 정적 URL, 인라인 데이터 URI 또는 동적으로 생성된 콘텐츠에서 PDF 문서를 애플리케이션에 통합하는 방법에 유연성을 제공합니다.

PDF 뷰어 속성

소스

> **Note**
>
> 이 구성 요소에 표시되는 미디어의 소스는 애플리케이션의 콘텐츠 보안 설정에서 허용되어야 합니다. 자세한 내용은 앱의 콘텐츠 보안 설정 보기 또는 업데이트 단원을 참조하십시오.

표현식, 개체, URL 또는 자동화일 수 있는 PDF 문서의 소스입니다.

표현식

표현식을 사용하여 PDF 소스를 동적으로 생성합니다.

개체

PDF 뷰어 구성 요소를 PDF 문서가 포함된 데이터 엔터티에 연결합니다.

URL

PDF 문서의 URL을 지정합니다.

URL

표시하려는 PDF 문서를 가리키는 URL을 입력할 수 있습니다. 이는 퍼블릭 웹 URL이거나 자체 애플리케이션 내의 URL일 수 있습니다.

예시: `https://example.com/document.pdf`

데이터 URI

데이터 URI는 애플리케이션 내에 작은 데이터 파일(예: 이미지 또는 PDFs 인라인으로 포함하는 간단한 방법입니다. PDF 문서는 base64 문자열로 인코딩되며 구성 요소의 구성에 직접 포함됩니다.

Blob 또는 ArrayBuffer

또한 PDF 문서를 Blob 또는 ArrayBuffer 객체로 제공하여 애플리케이션 내의 다양한 소스에서 PDF 데이터를 동적으로 생성하거나 검색할 수 있습니다.

자동화

PDF 뷰어 구성 요소를 PDF 문서를 제공하는 자동화에 연결합니다.

작업

- 다운로드: 사용자가 PDF 문서를 다운로드할 수 있는 버튼 또는 링크를 추가합니다.

레이아웃

- 너비: 백분율(%) 또는 고정 픽셀 값(예: 600px)으로 지정된 PDF 뷰어의 너비입니다.
- 높이: 고정 픽셀 값으로 지정된 PDF 뷰어의 높이입니다.

이미지 뷰어

이미지 뷰어 구성 요소를 사용하면 사용자가 애플리케이션 내의 이미지 파일을 보고 상호 작용할 수 있습니다.

이미지 뷰어 속성

소스

> **Note**
>
> 이 구성 요소에 표시되는 미디어의 소스는 애플리케이션의 콘텐츠 보안 설정에서 허용되어야 합니다. 자세한 내용은 앱의 콘텐츠 보안 설정 보기 또는 업데이트 단원을 참조하십시오.

- 개체: 이미지 뷰어 구성 요소를 이미지 파일이 포함된 데이터 개체에 연결합니다.
- URL: 이미지 파일의 URL을 지정합니다.
- 표현식: 표현식을 사용하여 이미지 소스를 동적으로 생성합니다.
- 자동화: 이미지 뷰어 구성 요소를 이미지 파일을 제공하는 자동화에 연결합니다.

대체 텍스트

접근성 목적으로 사용되는 이미지의 대체 텍스트 설명입니다.

레이아웃

- 이미지 맞춤: 이미지의 크기를 조정하고 구성 요소 내에 표시하는 방법을 결정합니다. 예: `Contain`, `Cover` 또는 `Fill`.
- 너비: 이미지 뷰어 구성 요소의 너비로, 백분율(%) 또는 고정 픽셀 값(예: 300px)으로 지정됩니다.
- 높이: 고정 픽셀 값으로 지정된 이미지 뷰어 구성 요소의 높이입니다.
- 배경: 이미지 뷰어 구성 요소의 배경색 또는 이미지를 설정할 수 있습니다.

자동화 및 작업: 앱의 비즈니스 로직 정의

자동화는 애플리케이션의 비즈니스 로직을 정의하는 방법입니다. 자동화의 주요 구성 요소는 자동화를 시작하는 트리거, 하나 이상의 작업 시퀀스, 자동화에 데이터를 전달하는 데 사용되는 입력 파라미터, 출력입니다.

주제

- 자동화 개념
- 자동화 생성, 편집 및 삭제
- 자동화 작업 추가, 편집 및 삭제
- Automation 작업 참조

자동화 개념

다음은 App Studio에서 자동화를 사용하여 앱의 비즈니스 로직을 정의하고 구성할 때 알아야 할 몇 가지 개념과 용어입니다.

자동화

자동화는 애플리케이션의 비즈니스 로직을 정의하는 방법입니다. 자동화의 주요 구성 요소는 자동화를 시작하는 트리거, 하나 이상의 작업 시퀀스, 자동화에 데이터를 전달하는 데 사용되는 입력 파라미터, 출력입니다.

작업

일반적으로 작업이라고 하는 자동화 작업은 자동화를 구성하는 로직의 개별 단계입니다. 각 작업은 이메일 전송, 데이터 레코드 생성, Lambda 함수 호출 또는 APIs 호출 등 특정 작업을 수행합니다. 작업은 작업 라이브러리의 자동화에 추가되며 조건문 또는 루프로 그룹화할 수 있습니다.

자동화 입력 파라미터

자동화 입력 파라미터는 구성 요소에서 자동화로 전달하여 유연하고 재사용 가능한 동적 입력 값입니다. 파라미터를 자동화의 변수로 생각하면 값을 자동화로 하드 코딩하는 대신 파라미터를 정의하고 필요할 때 다른 값을 제공할 수 있습니다. 파라미터를 사용하면 실행할 때마다 서로 다른 입력으로 동일한 자동화를 사용할 수 있습니다.

모의 출력

일부 작업은 커넥터를 사용하여 외부 리소스 또는 서비스와 상호 작용합니다. 미리 보기 환경을 사용할 때 애플리케이션은 외부 서비스와 상호 작용하지 않습니다. 미리 보기 환경에서 커넥터를 사용하는 작업을 테스트하려면 모의 출력을 사용하여 커넥터의 동작과 출력을 시뮬레이션할 수 있습니다. 모의 출력은 JavaScript를 사용하여 구성되며, 커넥터의 응답이 게시된 앱에 저장되는 것처럼 결과는 작업의 결과에 저장됩니다.

모의를 사용하면 커넥터를 통해 외부 서비스를 호출하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불행한 경로와 같은 자동화를 통해 다른 작업에 미치는 영향을 테스트할 수 있습니다.

자동화 출력

자동화 출력은 한 자동화의 값을 구성 요소 또는 기타 자동화와 같은 앱의 다른 리소스로 전달하는 데 사용됩니다. 자동화 출력은 표현식으로 구성되며 표현식은 자동화 파라미터 및 작업에서 계산된 정적 값 또는 동적 값을 반환할 수 있습니다. 기본적으로 자동화는 자동화 내의 작업 결과를 포함하여 어떤 데이터도 반환하지 않습니다.

자동화 출력을 사용하는 방법의 몇 가지 예는 다음과 같습니다.

- 배열을 반환하도록 자동화 출력을 구성하고 해당 배열을 전달하여 데이터 구성 요소를 채울 수 있습니다.
- 자동화를 사용하여 값을 계산하고 비즈니스 로직을 중앙 집중화하고 재사용하는 방법으로 해당 값을 다른 여러 자동화에 전달할 수 있습니다.

트리거

트리거는 자동화를 실행할 시기와 조건을 결정합니다. 트리거의 몇 가지 예는 `On click` 버튼과 텍스트 입력`On select`입니다. 구성 요소 유형에 따라 해당 구성 요소에 사용 가능한 트리거 목록이 결정됩니다. 트리거는 구성 요소에 추가되고 애플리케이션 스튜디오에서 구성됩니다.

자동화 생성, 편집 및 삭제

목차

- 자동화 생성
- 자동화 속성 보기 또는 편집
- 자동화 삭제

자동화 생성

다음 절차에 따라 App Studio 애플리케이션에서 자동화를 생성합니다. 일단 생성되면 속성을 편집하고 해당 속성에 작업을 추가하여 자동화를 구성해야 합니다.

자동화를 생성하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 자동화가 없는 경우 캔버스에서 + 자동화 추가를 선택합니다. 그렇지 않으면 왼쪽 자동화 메뉴에서 + 추가를 선택합니다.
4. 새 자동화가 생성되고 속성을 편집하거나 애플리케이션의 비즈니스 로직을 정의하는 작업을 추가 및 구성할 수 있습니다.

자동화 속성 보기 또는 편집

다음 절차에 따라 자동화 속성을 보거나 편집합니다.

자동화 속성을 보거나 편집하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 왼쪽 자동화 메뉴에서 속성을 보거나 편집하려는 자동화를 선택하여 오른쪽 속성 메뉴를 엽니다.
4. 속성 메뉴에서 다음 속성을 볼 수 있습니다.

   - 자동화 식별자: 자동화의 고유한 이름입니다. 편집하려면 텍스트 필드에 새 식별자를 입력합니다.
   - 자동화 파라미터: 자동화 파라미터는 앱의 UI에서 자동화 및 데이터 작업으로 동적 값을 전달하는 데 사용됩니다. 파라미터를 추가하려면 + 추가를 선택합니다. 연필 아이콘을 선택하여 파라미터의 이름, 설명 또는 유형을 변경합니다. 파라미터를 제거하려면 휴지통 아이콘을 선택합니다.

     > **Tip**
     >
     > 캔버스에서 직접 자동화 파라미터를 추가할 수도 있습니다.

   - 자동화 출력: 자동화 출력은 다른 자동화 또는 구성 요소에서 참조할 수 있는 자동화의 데이터를 구성하는 데 사용됩니다. 기본적으로 자동화는 출력을 생성하지 않습니다. 자동화 출력을 추가하려면 + 추가를 선택합니다. 출력을 제거하려면 휴지통 아이콘을 선택합니다.
5. 작업을 추가하고 구성하여 자동화가 수행하는 작업을 정의합니다. 작업에 대한 자세한 내용은 자동화 작업 추가, 편집 및 삭제 및 단원을 참조하십시오Automation 작업 참조.

자동화 삭제

다음 절차에 따라 App Studio 애플리케이션에서 자동화를 삭제합니다.

자동화를 삭제하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 왼쪽 자동화 메뉴에서 삭제할 자동화의 줄임표 메뉴를 선택하고 삭제를 선택합니다. 또는 자동화의 오른쪽 속성 메뉴에서 휴지통 아이콘을 선택할 수 있습니다.
4. 확인 대화 상자에서 삭제를 선택합니다.

자동화 작업 추가, 편집 및 삭제

일반적으로 작업이라고 하는 자동화 작업은 자동화를 구성하는 로직의 개별 단계입니다. 각 작업은 이메일 전송, 데이터 레코드 생성, Lambda 함수 호출 또는 APIs 호출 등 특정 작업을 수행합니다. 작업은 작업 라이브러리의 자동화에 추가되며 조건문 또는 루프로 그룹화할 수 있습니다.

목차

- 자동화 작업 추가
- 자동화 작업 속성 보기 및 편집
- 자동화 작업 삭제

자동화 작업 추가

다음 절차에 따라 App Studio 애플리케이션의 자동화에 작업을 추가합니다.

자동화 작업을 추가하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 왼쪽 자동화 메뉴에서 작업을 추가할 자동화를 선택합니다.
4. 오른쪽 작업 메뉴에서 추가할 작업을 선택하거나 작업을 캔버스로 끌어서 놓습니다. 작업이 생성된 후 작업을 선택하여 작업 속성을 구성하여 작업의 기능을 정의할 수 있습니다. 작업 속성 및 구성에 대한 자세한 내용은 섹션을 참조하세요Automation 작업 참조.

자동화 작업 속성 보기 및 편집

다음 절차에 따라 App Studio 애플리케이션에서 자동화 작업의 속성을 보거나 편집합니다.

자동화 작업 속성을 보거나 편집하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 왼쪽 자동화 메뉴에서 속성을 보거나 편집할 작업을 선택합니다. 또는 캔버스가 포함된 자동화를 볼 때 캔버스에서 작업을 선택할 수 있습니다.
4. 오른쪽 속성 메뉴에서 작업 속성을 보거나 편집할 수 있습니다. 작업의 속성은 작업 유형마다 다릅니다. 작업 속성 및 구성에 대한 자세한 내용은 섹션을 참조하세요Automation 작업 참조.

자동화 작업 삭제

다음 절차에 따라 App Studio 애플리케이션의 자동화에서 작업을 삭제합니다.

자동화 작업을 삭제하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 자동화 탭을 선택합니다.
3. 왼쪽 자동화 메뉴에서 삭제하려는 작업이 포함된 자동화를 선택합니다.
4. 캔버스에서 삭제하려는 작업의 휴지통 아이콘을 선택하고 삭제를 선택합니다.

Automation 작업 참조

다음은 App Studio에서 사용되는 자동화 작업에 대한 참조 설명서입니다.

일반적으로 작업이라고 하는 자동화 작업은 자동화를 구성하는 로직의 개별 단계입니다. 각 작업은 이메일 전송, 데이터 레코드 생성, Lambda 함수 호출 또는 APIs 호출 등 특정 작업을 수행합니다. 작업은 작업 라이브러리의 자동화에 추가되며 조건문 또는 루프로 그룹화할 수 있습니다.

자동화 및 해당 작업을 생성하고 구성하는 방법에 대한 자세한 내용은의 주제를 참조하세요자동화 및 작업: 앱의 비즈니스 로직 정의.

API 호출

HTTP REST API 요청을 호출합니다. 빌더는이 작업을 사용하여 App Studio에서 APIs. 예를 들어 이를 사용하여 타사 시스템 또는 자체 개발 애플리케이션에 연결하여 비즈니스 크리티컬 데이터에 액세스하거나 전용 App Studio 작업에서 호출할 수 없는 API 엔드포인트를 호출할 수 있습니다.

REST APIs에 대한 자세한 내용은 RESTful API란 무엇입니까?를 참조하세요.

속성

커넥터

이 작업에서 이루어진 API 요청에 사용할 커넥터입니다. 커넥터 드롭다운에는 API Connector 및 유형의 커넥터만 포함됩니다OpenAPI Connector. 커넥터 구성 방식에 따라 자격 증명, 기본 헤더 또는 쿼리 파라미터와 같은 중요한 정보가 포함될 수 있습니다.

API Connector 사용 및 간의 비교를 포함하여 API 커넥터에 대한 자세한 내용은 섹션을 OpenAPI Connector참조하세요타사 서비스에 연결.

API 요청 구성 속성

속성 패널에서 API 요청 구성을 선택하여 요청 구성 대화 상자를 엽니다. API 커넥터를 선택하면 대화 상자에 커넥터 정보가 포함됩니다.

메서드: API 호출의 메서드입니다. 가능한 값은 다음과 같습니다.

- DELETE: 지정된 리소스를 삭제합니다.
- GET: 정보 또는 데이터를 검색합니다.
- HEAD: 본문이 없는 응답의 헤더만 검색합니다.
- POST: 처리할 데이터를 제출합니다.
- PUSH: 처리할 데이터를 제출합니다.
- PATCH: 지정된 리소스를 부분적으로 업데이트합니다.

경로: 리소스의 상대 경로입니다.

헤더: API 요청과 함께 전송할 키-값 페어 형태의 모든 헤더입니다. 커넥터를 선택하면 구성된 헤더가 자동으로 추가되고 제거할 수 없습니다. 구성된 헤더는 편집할 수 없지만 동일한 이름의 다른 헤더를 추가하여 재정의할 수 있습니다.

쿼리 파라미터: API 요청과 함께 전송할 키-값 페어 형태의 모든 쿼리 파라미터입니다. 커넥터를 선택하면 구성된 쿼리 파라미터가 자동으로 추가되고 편집하거나 제거할 수 없습니다.

본문: API 요청과 함께 JSON 형식으로 전송할 정보입니다. GET 요청에 대한 본문은 없습니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

간접 호출 AWS

AWS 서비스에서 작업을 호출합니다. 이는 AWS 서비스 또는 작업을 호출하기 위한 일반적인 작업이며, 원하는 AWS 서비스 또는 작업에 대한 전용 작업이 없는 경우 사용해야 합니다.

속성

서비스

실행할 작업이 포함된 AWS 서비스입니다.

연산

실행할 작업입니다.

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

지정된 작업을 실행할 때가 될 JSON 입력입니다. AWS 작업에 대한 입력 구성에 대한 자세한 내용은 섹션을 참조하세요AWS SDK for JavaScript.

Lambda 간접 호출

기존 Lambda 함수를 호출합니다.

속성

커넥터

이 작업에서 실행되는 Lambda 함수에 사용할 커넥터입니다. 구성된 커넥터는 Lambda 함수에 액세스하기 위한 적절한 자격 증명과 Lambda 함수가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다. Lambda용 커넥터 구성에 대한 자세한 내용은 섹션을 참조하세요3단계: Lambda 커넥터 생성.

함수 이름

실행할 Lambda 함수의 이름입니다. 이는 함수 ARN(Amazon 리소스 이름)이 아닌 함수 이름입니다.

함수 이벤트

Lambda 함수에 이벤트 페이로드로 전달할 키-값 페어입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Loop

중첩된 작업을 반복적으로 실행하여 항목 목록을 한 번에 한 항목씩 반복합니다. 예를 들어 작업을 루프 레코드 생성 작업에 추가하여 여러 레코드를 생성합니다.

루프 작업은 다른 루프 또는 조건 작업 내에 중첩될 수 있습니다. 루프 작업은 병렬이 아닌 순차적으로 실행됩니다. 루프 내의 각 작업 결과는 동일한 루프 반복 내의 후속 작업에만 액세스할 수 있습니다. 루프 외부에서 또는 루프의 다른 반복으로 액세스할 수 없습니다.

속성

소스

한 번에 한 항목씩 반복할 항목 목록입니다. 소스는 이전 작업의 결과이거나 JavaScript 표현식을 사용하여 제공할 수 있는 문자열, 숫자 또는 객체의 정적 목록일 수 있습니다.

예제

다음 목록에는 소스 입력의 예가 포함되어 있습니다.

- 이전 작업의 결과: {{results.*actionName*.data}}
- 숫자 목록: {{[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]}}
- 문자열 목록: {{["apple", "banana", "orange", "grape", "kiwi"]}}
- 계산된 값: {{params.*actionName*.split("\n")}}

현재 항목 이름

반복되는 현재 항목을 참조하는 데 사용할 수 있는 변수의 이름입니다. 현재 항목 이름은 두 개 이상의 루프를 중첩하고 각 루프에서 변수에 액세스할 수 있도록 구성할 수 있습니다. 예를 들어 두 개의 루프가 있는 국가 및 도시를 반복하는 경우 currentCountry 및를 구성하고 참조할 수 있습니다currentCity.

조건

자동화가 실행될 때 평가되는 하나 이상의 지정된 논리적 조건의 결과를 기반으로 작업을 실행합니다. 조건 작업은 다음 구성 요소로 구성됩니다.

- true 또는 로 평가되는 JavaScript 표현식을 제공하는 데 사용되는 조건 필드입니다false.
- 조건이 로 평가될 경우 실행되는 작업이 포함된 true 브랜치입니다true.
- 조건이 로 평가될 경우 실행되는 작업이 포함된 false 브랜치입니다false.

true 및 false 브랜치를 조건 작업으로 드래그하여 해당 브랜치에 작업을 추가합니다.

속성

조건

작업이 실행될 때 평가할 JavaScript 표현식입니다.

레코드 생성

기존 App Studio 엔터티에 하나의 레코드를 생성합니다.

속성

개체

레코드를 생성할 엔터티입니다. 개체를 선택한 후에는 레코드를 생성하려면 개체의 필드에 값을 추가해야 합니다. 필드의 유형과 필드가 필수 또는 선택 사항인지 여부는 엔터티에 정의됩니다.

레코드 업데이트

App Studio 엔터티의 기존 레코드를 업데이트합니다.

속성

개체

업데이트할 레코드가 포함된 엔터티입니다.

조건

작업에 의해 업데이트되는 레코드를 정의하는 기준입니다. 조건을 그룹화하여 하나의 논리적 문을 생성할 수 있습니다. 그룹 또는 조건을 AND 또는 OR 문과 결합할 수 있습니다.

필드

조건에 지정된 레코드에서 업데이트할 필드입니다.

값

지정된 필드에서 업데이트할 값입니다.

레코드 삭제

App Studio 엔터티에서 레코드를 삭제합니다.

속성

개체

삭제할 레코드가 포함된 엔터티입니다.

조건

작업에 의해 삭제되는 레코드를 정의하는 기준입니다. 조건을 그룹화하여 하나의 로직 문을 생성할 수 있습니다. 그룹 또는 조건을 AND 또는 OR 문과 결합할 수 있습니다.

데이터 작업 호출

선택적 파라미터를 사용하여 데이터 작업을 실행합니다.

속성

데이터 작업

작업에서 실행할 데이터 작업입니다.

파라미터

데이터 작업에서 사용할 데이터 작업 파라미터입니다. 데이터 작업 파라미터는 데이터 작업의 입력으로 사용되는 값을 전송하는 데 사용됩니다. 자동화 작업을 구성할 때 데이터 작업 파라미터를 추가할 수 있지만 데이터 탭에서 편집해야 합니다.

고급 설정

`Invoke data action` 작업에는 다음과 같은 고급 설정이 포함됩니다.

- 페이지 크기: 각 쿼리에서 가져올 최대 레코드 수입니다. 기본값은 500이고 최대값은 3000입니다.
- 페이지 매김 토큰: 쿼리에서 추가 레코드를 가져오는 데 사용되는 토큰입니다. 예를 들어 `Page size`가 500으로 설정되었지만 레코드가 500개 이상인 경우 페이지 매김 토큰을 후속 쿼리에 전달하면 다음 500개가 가져옵니다. 레코드나 페이지가 더 이상 없는 경우 토큰이 정의되지 않습니다.

Amazon S3: 객체 넣기

`Amazon S3 PutObject` 작업을 사용하여 키(파일 경로)로 식별되는 객체를 지정된 Amazon S3 버킷에 추가합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`PutObject` 명령에 사용할 필수 옵션입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon S3 PutObject` 작업에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 PutObject를 참조하세요.

- 버킷: 객체를 넣을 Amazon S3 버킷의 이름입니다.
- 키: Amazon S3 버킷에 넣을 객체의 고유한 이름입니다.
- 본문: Amazon S3 버킷에 넣을 객체의 콘텐츠입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon S3: 객체 삭제

`Amazon S3 DeleteObject` 작업을 사용하여 지정된 Amazon S3 버킷에서 키(파일 경로)로 식별되는 객체를 삭제합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`DeleteObject` 명령에 사용할 필수 옵션입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon S3 DeleteObject` 작업에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 DeleteObject를 참조하세요.

- 버킷: 객체를 삭제할 Amazon S3 버킷의 이름입니다.
- 키: Amazon S3 버킷에서 삭제할 객체의 고유한 이름입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon S3: 객체 가져오기

`Amazon S3 GetObject` 작업을 사용하여 지정된 Amazon S3 버킷에서 키(파일 경로)로 식별되는 객체를 검색합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`GetObject` 명령에 사용할 필수 옵션입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon S3 GetObject` 작업에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 GetObject를 참조하세요.

- 버킷: 객체를 검색할 Amazon S3 버킷의 이름입니다.
- 키: Amazon S3 버킷에서 검색할 객체의 고유한 이름입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon S3: 객체 나열

`Amazon S3 ListObjects` 작업을 사용하여 지정된 Amazon S3 버킷의 객체를 나열합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`ListObjects` 명령에 사용할 필수 옵션입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon S3 ListObjects` 작업에 대한 자세한 내용은 Amazon Simple Storage Service API 참조의 ListObjects를 참조하세요.

- 버킷: 객체를 나열할 Amazon S3 버킷의 이름입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩

니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Textract: 문서 분석

`Amazon Textract AnalyzeDocument` 작업을 사용하여 입력 문서에서 감지된 항목 간의 관계를 분석합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`AnalyzeDocument` 명령에 사용할 요청의 내용입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon Textract AnalyzeDocument` 작업에 대한 자세한 내용은 Amazon Textract 개발자 안내서의 AnalyzeDocument를 참조하세요.

- 문서/S3Object/버킷: Amazon S3 버킷의 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/이름: 입력 문서의 파일 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/버전: Amazon S3 버킷에 버전 관리가 활성화된 경우 객체의 버전을 지정할 수 있습니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- FeatureTypes: 수행할 분석 유형의 목록입니다. 유효한 값은 `TABLES`, `FORMS`, `QUERIES`, `SIGNATURES`, `LAYOUT`입니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Textract: 비용 분석

`Amazon Textract AnalyzeExpense` 작업을 사용하여 텍스트 간의 재무 관련 관계에 대한 입력 문서를 분석합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`AnalyzeExpense` 명령에 사용할 요청의 내용입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon Textract AnalyzeExpense` 작업에 대한 자세한 내용은 Amazon Textract 개발자 안내서의 AnalyzeExpense를 참조하세요.

- 문서/S3Object/버킷: Amazon S3 버킷의 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/이름: 입력 문서의 파일 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/버전: Amazon S3 버킷에 버전 관리가 활성화된 경우 객체의 버전을 지정할 수 있습니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Textract: ID 분석

`Amazon Textract AnalyzeID` 작업을 사용하여 자격 증명 문서에서 관련 정보를 분석합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`AnalyzeID` 명령에 사용할 요청의 내용입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon Textract AnalyzeID` 작업에 대한 자세한 내용은 Amazon Textract 개발자 안내서의 AnalyzeID를 참조하세요.

- 문서/S3Object/버킷: Amazon S3 버킷의 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/이름: 입력 문서의 파일 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/버전: Amazon S3 버킷에 버전 관리가 활성화된 경우 객체의 버전을 지정할 수 있습니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Textract: 문서 텍스트 감지

`Amazon Textract DetectDocumentText` 작업을 사용하여 입력 문서에서 텍스트 줄과 텍스트 줄을 구성하는 단어를 감지합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`DetectDocumentText` 명령에 사용할 요청의 내용입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon Textract DetectDocumentText` 작업에 대한 자세한 내용은 Amazon Textract 개발자 안내서의 DetectDocumentText를 참조하세요.

- 문서/S3Object/버킷: Amazon S3 버킷의 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되면이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/이름: 입력 문서의 파일 이름입니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.
- 문서/S3Object/버전: Amazon S3 버킷에 버전 관리가 활성화된 경우 객체의 버전을 지정할 수 있습니다. S3 업로드 구성 요소가 있는 작업에 파일이 전달되는 경우이 파라미터를 비워 둘 수 있습니다.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Bedrock: GenAI 프롬프트

Amazon Bedrock InvokeModel 작업을 사용하여 작업 속성에 제공된 프롬프트 및 추론 파라미터를 사용하여 추론을 실행합니다. 작업은 텍스트, 이미지 및 임베딩을 생성할 수 있습니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 이 작업을 성공적으로 사용하려면 Amazon Bedrock 런타임을 서비스로 사용하여 커넥터를 구성해야 합니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

모델

Amazon Bedrock에서 요청을 처리하는 데 사용할 파운데이션 모델입니다. Amazon Bedrock의 모델에 대한 자세한 내용은 Amazon Bedrock 사용 설명서의 Amazon Bedrock 파운데이션 모델 정보를 참조하세요.

입력 유형

Amazon Bedrock 모델로 전송하는 입력의 입력 유형입니다. 가능한 값은 텍스트, 문서 및 이미지입니다. 입력 유형을 선택할 수 없는 경우 구성된 모델에서 지원되지 않을 수 있습니다.

사용자 프롬프트

응답을 생성하기 위해 처리할 Amazon Bedrock 모델로 전송할 프롬프트입니다. 파라미터를 사용하는 구성 요소, 자동화의 이전 작업 또는 다른 자동화와 같이 애플리케이션의 다른 부분에서 정적 텍스트를

입력하거나 입력을 전달할 수 있습니다. 다음 예제에서는 구성 요소 또는 이전 작업의 값을 전달하는 방법을 보여줍니다.

- 파라미터로 구성 요소의 값을 전달하려면: `{{params.paramName}}`
- 이전 작업의 값을 전달하려면: `{{results.actionName}}`

시스템 프롬프트(Claude 모델)

요청을 처리할 때 Amazon Bedrock 모델에서 사용할 시스템 프롬프트입니다. 시스템 프롬프트는 Claude 모델에 컨텍스트, 지침 또는 지침을 제공하는 데 사용됩니다.

요청 설정

다양한 요청 설정 및 모델 추론 파라미터를 구성합니다. 다음과 같은 설정을 구성할 수 있습니다.

- 온도: 요청을 처리할 때 Amazon Bedrock 모델에서 사용할 온도입니다. 온도에 따라 Bedrock 모델 출력의 무작위성 또는 창의성이 결정됩니다. 온도가 높을수록 응답이 더 창의적이고 덜 분석적입니다. 가능한 값은 입니다`[0-10]`.
- 최대 토큰: Amazon Bedrock 모델의 출력 길이를 제한합니다.
- TopP: nucleus 샘플링에서 모델은 각 후속 토큰의 모든 옵션에 대한 누적 분포를 확률 내림차순으로 계산하고 TopP에서 지정한 특정 확률에 도달하면 잘라냅니다. 온도 또는 TopP를 변경해야 하지만 둘 다 변경해서는 안 됩니다.
- 중지 시퀀스: 모델이 요청 처리 및 출력 생성을 중지하는 시퀀스입니다.

자세한 내용은 Amazon Bedrock 사용 설명서의 파운데이션 모델에 대한 추론 요청 파라미터 및 응답 필드를 참조하세요.

시퀀스 중지

Amazon Bedrock Guardrail ID 및 버전을 입력합니다. 가드레일은 사용 사례와 책임 있는 AI 정책에 따라 보호 기능을 구현하는 데 사용됩니다. 자세한 내용은 Amazon Bedrock 사용 설명서의 Amazon Bedrock Guardrails를 사용하여 모델에서 유해한 콘텐츠 중지를 참조하세요.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

Amazon Bedrock: 모델 간접 호출

Amazon Bedrock InvokeModel 작업을 사용하여 요청 본문에 제공된 프롬프트 및 추론 파라미터를 사용하여 추론을 실행합니다. 모델 추론을 사용하여 텍스트, 이미지 및 임베딩을 생성합니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 이 작업을 성공적으로 사용하려면 Amazon Bedrock 런타임을 서비스로 사용하여 커넥터를 구성해야 합니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`InvokeModel` 명령에 사용할 요청의 내용입니다.

> **Note**
>
> 예제 명령을 포함하여 `Amazon Bedrock InvokeModel` 작업에 대한 자세한 내용은 Amazon Bedrock API 참조의 InvokeModel을 참조하세요.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

JavaScript

사용자 지정 JavaScript 함수를 실행하여 지정된 값을 반환합니다.

> **Important**
>
> App Studio는 타사 또는 사용자 지정 JavaScript 라이브러리 사용을 지원하지 않습니다.

속성

소스 코드

작업에서 실행할 JavaScript 코드 조각입니다.

> **Tip**
>
> AI를 사용하여 다음 단계를 수행하여 JavaScript를 생성할 수 있습니다.
>
> 1. 확장 아이콘을 선택하여 확장된 JavaScript 편집기를 엽니다.
> 2. (선택 사항): 코드 수정 토글을 활성화하여 기존 JavaScript를 수정합니다. 그렇지 않으면 AI가 기존 JavaScript를 대체합니다.
> 3. JavaScript 생성에서 JavaScript로 수행할 작업을 설명합니다. 예: **`Add two numbers`**.
> 4. 전송 아이콘을 선택하여 JavaScript를 생성합니다.

자동화 호출

지정된 자동화를 실행합니다.

속성

자동화 호출

작업에서 실행할 자동화입니다.

이메일 보내기

`Amazon SES SendEmail` 작업을 사용하여 이메일을 보냅니다.

속성

커넥터

이 작업에서 실행되는 작업에 사용할 커넥터입니다. 구성된 커넥터는 작업을 실행하기 위한 적절한 자격 증명과 작업에서 참조되는 리소스가 포함된 AWS 리전과 같은 기타 구성 정보로 설정해야 합니다.

구성

`SendEmail` 명령에 사용할 요청의 내용입니다. 옵션은 다음과 같습니다.

> **Note**
>
> `Amazon SES SendEmail` 작업에 대한 자세한 내용은 Amazon Simple Email Service API 참조의 SendEmail을 참조하세요.

모의 출력

작업은 미리 보기 환경의 외부 서비스 또는 리소스와 상호 작용하지 않습니다. 모의 출력 필드는 테스트 목적으로 미리 보기 환경에서 커넥터의 동작을 시뮬레이션하는 JSON 표현식을 제공하는 데 사용됩니다. 이 코드 조각은 라이브 환경의 게시된 앱에 대한 커넥터 응답과 마찬가지로 작업의 `results` 맵에 저장됩니다.

이 필드를 사용하면 커넥터를 통해 외부 서비스와 통신하지 않고도 다양한 시나리오와 다양한 결과 값 시뮬레이션, 오류 시나리오, 엣지 케이스 또는 불만족스러운 경로와 같은 자동화 내의 다른 작업에 미치는 영향을 테스트할 수 있습니다.

개체 및 데이터 작업: 앱의 데이터 모델 구성

개체는 App Studio의 데이터 테이블입니다. 개체는 데이터 소스의 테이블과 직접 상호 작용합니다. 개체에는 데이터를 설명하는 필드, 데이터를 찾아 반환하는 쿼리, 개체의 필드를 데이터 소스의 열에 연결하는 매핑이 포함됩니다.

주제

- 데이터 모델 설계 모범 사례
- App Studio 앱에서 개체 생성
- App Studio 앱에서 개체 구성 또는 편집
- 개체 삭제
- AWS App Studio의 관리형 데이터 엔터티

데이터 모델 설계 모범 사례

다음 모범 사례를 사용하여 애플리케이션의 요구 사항을 충족하고 데이터 인프라의 장기 안정성과 성능을 보장하는 App Studio 애플리케이션에서 AWS 사용할 수 있도록에서 강력하고 확장 가능하며 안전한 관계형 데이터 모델을 생성합니다.

- 올바른 AWS 데이터 서비스 선택: 요구 사항에 따라 적절한 AWS 데이터 서비스를 선택합니다. 예를 들어 온라인 트랜잭션 처리(OLTP) 애플리케이션의 경우 MySQL 및 PostgreSQL과 같은 다양한 데이터베이스 엔진을 지원하는 클라우드 네이티브, 관계형 및 완전 관리형 데이터베이스 서비스인 Amazon Aurora와 같은 데이터베이스(DB)를 고려할 수 있습니다. App Studio에서 지원하는 Aurora 버전의 전체 목록은 섹션을 참조하세요Amazon Aurora에 연결. 반면 온라인 분석 처리(OLAP) 사용 사례의 경우 매우 큰 데이터 세트에 대해 복잡한 쿼리를 실행할 수 있는 클라우드 데이터 웨어하우스인 Amazon Redshift를 사용하는 것이 좋습니다. 이러한 쿼리는 완료하는 데 종종 시간(몇 초)이 걸릴 수 있으므로 지연 시간이 짧은 데이터 액세스가 필요한 OLTP 애플리케이션에는 Amazon Redshift가 적합하지 않습니다.
- 확장성을 위한 설계: 향후 성장과 확장성을 염두에 두고 데이터 모델을 계획합니다. 적절한 데이터 서비스, 데이터베이스 인스턴스 유형 및 구성(예: 프로비저닝된 용량)을 선택할 때 예상 데이터 볼륨, 액세스 패턴 및 성능 요구 사항과 같은 요소를 고려합니다.
  - Aurora 서버리스를 사용한 크기 조정에 대한 자세한 내용은 Aurora 서버리스 V2의 성능 및 크기 조정을 참조하세요.
- 데이터 정규화: 데이터베이스 정규화 원칙에 따라 데이터 중복을 최소화하고 데이터 무결성을 개선합니다. 여기에는 적절한 테이블 생성, 기본 및 외래 키 정의, 엔터티 간 관계 수립이 포함됩니다. App Studio에서 한 엔터티의 데이터를 쿼리할 때 쿼리에 `join` 절을 지정하여 다른 엔터티에서 관련 데이터를 검색할 수 있습니다.
- 적절한 인덱싱 구현: 가장 중요한 쿼리 및 액세스 패턴을 식별하고 성능을 최적화하기 위한 적절한 인덱스를 생성합니다.
- AWS 데이터 서비스 기능 활용: 자동 백업, 다중 AZ 배포, 자동 소프트웨어 업데이트 등 선택한 AWS 데이터 서비스에서 제공하는 기능을 활용합니다.
- 데이터 보안: IAM(AWS Identity and Access Management) 정책과 같은 강력한 보안 조치를 구현하고, 테이블 및 스키마에 대한 제한된 권한이 있는 데이터베이스 사용자를 생성하고, 저장 및 전송 중 암호화를 적용합니다.
- 성능 모니터링 및 최적화: 데이터베이스의 성능을 지속적으로 모니터링하고 필요에 따라 리소스 조정, 쿼리 최적화 또는 데이터베이스 구성 조정과 같은 조정을 수행합니다.

- 데이터베이스 관리 자동화: Aurora Autoscaling, Aurora용 성능 개선 도우미 및 AWS Database Migration Service와 같은 AWS 서비스를 활용하여 데이터베이스 관리 작업을 자동화하고 운영 오버헤드를 줄입니다.
- 재해 복구 및 백업 전략 구현: Aurora 자동 백업, point-in-time 복구, 교차 리전 복제본 구성과 같은 기능을 활용하여 잘 정의된 백업 및 복구 계획이 있는지 확인합니다.
- AWS 모범 사례 및 설명서 준수: 선택한 데이터 서비스에 대한 최신 AWS 모범 사례, 지침 및 설명서를 up-to-date 데이터 모델 및 구현이 AWS 권장 사항에 부합하는지 확인합니다.

각 AWS 데이터 서비스의 자세한 지침은 다음 주제를 참조하세요.

- Amazon Aurora의 모범 사례
- Amazon Aurora MySQL 모범 사례
- Amazon Redshift 쿼리 성능 튜닝
- Amazon DynamoDB에서 데이터 쿼리 및 스캔 모범 사례

App Studio 앱에서 개체 생성

App Studio 앱에서 개체를 생성하는 방법에는 네 가지가 있습니다. 다음 목록에는 각 메서드, 그 이점, 해당 메서드를 사용하여 개체를 생성 및 구성하기 위한 지침 링크가 포함되어 있습니다.

- 기존 데이터 소스에서 개체 생성: 기존 데이터 소스 테이블에서 개체와 해당 필드를 자동으로 생성하고 필드를 데이터 소스 테이블 열에 매핑합니다. App Studio 앱에서 사용하려는 기존 데이터 소스가 있는 경우이 옵션을 사용하는 것이 좋습니다.
- App Studio 관리형 데이터 소스를 사용하여 엔터티 생성: App Studio가 관리하는 엔터티와 DynamoDB 테이블을 생성합니다. 개체를 업데이트하면 DynamoDB 테이블이 자동으로 업데이트됩니다. 이 옵션을 사용하면 타사 데이터 소스를 수동으로 생성, 관리 또는 연결하거나 엔터티 필드에서 테이블 열로의 매핑을 지정할 필요가 없습니다. 앱의 모든 데이터 모델링 및 구성은 App Studio에서 수행됩니다. 이 옵션은 자체 데이터 소스와 DynamoDB 테이블을 관리하지 않으려는 경우에 선호되며, 해당 기능으로 앱에 충분합니다.
- 빈 개체 생성: 완전히 처음부터 빈 개체를 생성합니다. 관리자가 생성한 기존 데이터 소스 또는 커넥터가 없고 외부 데이터 소스의 제약 없이 앱의 데이터 모델을 유연하게 설계하려는 경우이 옵션을 사용하는 것이 좋습니다. 생성 후 개체를 데이터 소스에 연결할 수 있습니다.
- AI를 사용하여 개체 생성: 지정된 개체 이름을 기반으로 개체, 필드, 데이터 작업 및 샘플 데이터를 생성합니다. 앱의 데이터 모델을 알고 있지만 엔터티로 변환하는 데 도움이 필요한 경우이 옵션을 사용하는 것이 좋습니다.

기존 데이터 소스에서 개체 생성

데이터 소스의 테이블을 사용하여 엔터티와 해당 필드를 자동으로 생성하고 엔터티 필드를 테이블의 열에 매핑합니다. App Studio 앱에서 사용하려는 기존 데이터 소스가 있는 경우이 옵션을 사용하는 것이 좋습니다.

1. 필요한 경우 애플리케이션으로 이동합니다.
2. 캔버스 상단의 데이터 탭을 선택합니다.
3. 앱에 개체가 없는 경우 + 개체 생성을 선택합니다. 그렇지 않으면 왼쪽 엔터티 메뉴에서 + 추가를 선택합니다.
4. 기존 데이터 소스에서 테이블 사용을 선택합니다.
5. 커넥터에서 개체를 생성하는 데 사용할 테이블이 포함된 커넥터를 선택합니다.
6. 테이블에서 개체를 생성하는 데 사용할 테이블을 선택합니다.
7. 데이터 작업 생성 확인란을 선택하여 데이터 작업을 생성합니다.
8. 개체 생성을 선택합니다. 이제 개체가 생성되고 왼쪽 개체 패널에서 해당 개체를 볼 수 있습니다.
9. 의 절차에 따라 새 개체를 구성합니다App Studio 앱에서 개체 구성 또는 편집. 개체가 기존 데이터 소스로 생성되었으므로 필드, 연결된 데이터 소스 및 필드 매핑과 같은 일부 속성 또는 리소스가 이미 생성되었습니다. 또한 생성 중에 데이터 작업 생성 확인란을 선택한 경우 엔터티에 데이터 작업이 포함됩니다.

App Studio 관리형 데이터 소스를 사용하여 엔터티 생성

App Studio에서 관리하는 관리형 엔터티와 해당 DynamoDB 테이블을 생성합니다. 연결된 AWS 계정에 DynamoDB 테이블이 있는 동안 App Studio 앱의 개체가 변경되면 DynamoDB 테이블이 자동으로 업데이트됩니다. 이 옵션을 사용하면 타사 데이터 소스를 수동으로 생성, 관리 또는 연결하거나 엔터티 필드에서 테이블 열로의 매핑을 지정할 필요가 없습니다. 이 옵션은 자체 데이터 소스와 DynamoDB 테이블을 관리하지 않으려는 경우에 선호되며, 해당 기능으로 앱에 충분합니다. 관리형 엔터티에 대한 자세한 내용은 섹션을 참조하세요AWS App Studio의 관리형 데이터 엔터티.

여러 애플리케이션에서 동일한 관리형 엔터티를 사용할 수 있습니다. 지침은 기존 데이터 소스에서 개체 생성 섹션을 참조하세요.

1. 필요한 경우 애플리케이션으로 이동합니다.
2. 캔버스 상단의 데이터 탭을 선택합니다.
3. 앱에 개체가 없는 경우 + 개체 생성을 선택합니다. 그렇지 않으면 왼쪽 엔터티 메뉴에서 + 추가를 선택합니다.

4. App Studio 관리형 엔터티 생성을 선택합니다.
5. 엔터티 이름에 엔터티의 이름을 입력합니다.
6. 프라이머리 키에서 엔터티의 프라이머리 키 이름을 입력합니다. 기본 키는 엔터티의 고유 식별자이며 엔터티가 생성된 후에는 변경할 수 없습니다.
7. 기본 키 데이터 유형에서 개체의 기본 키 데이터 유형을 선택합니다. 개체가 생성된 후에는 데이터 유형을 변경할 수 없습니다.
8. 개체 생성을 선택합니다. 이제 개체가 생성되고 왼쪽 개체 패널에서 해당 개체를 볼 수 있습니다.
9. 의 절차에 따라 새 개체를 구성합니다App Studio 앱에서 개체 구성 또는 편집. 개체가 관리형 데이터로 생성되었으므로 기본 키 필드 및 연결된 데이터 소스와 같은 일부 속성 또는 리소스가 이미 생성되었습니다.

빈 개체 생성

완전히 처음부터 빈 개체를 생성합니다. 관리자가 생성한 기존 데이터 소스 또는 커넥터가 없는 경우 이 옵션을 사용하는 것이 좋습니다. 외부 데이터 소스의 제약 없이 App Studio 앱 내에서 개체를 설계할 수 있으므로 빈 개체를 생성하면 유연성이 제공됩니다. 앱의 데이터 모델을 설계하고 그에 따라 개체를 구성한 후에도 나중에 외부 데이터 소스에 연결할 수 있습니다.

1. 필요한 경우 애플리케이션으로 이동합니다.
2. 캔버스 상단의 데이터 탭을 선택합니다.
3. 앱에 개체가 없는 경우 + 개체 생성을 선택합니다. 그렇지 않으면 왼쪽 엔터티 메뉴에서 + 추가를 선택합니다.
4. 개체 생성을 선택합니다.
5. 개체 생성을 선택합니다. 이제 개체가 생성되고 왼쪽 개체 패널에서 해당 개체를 볼 수 있습니다.
6. 의 절차에 따라 새 개체를 구성합니다App Studio 앱에서 개체 구성 또는 편집.

AI를 사용하여 개체 생성

지정된 개체 이름을 기반으로 개체, 필드, 데이터 작업 및 샘플 데이터를 생성합니다. 앱의 데이터 모델을 알고 있지만 엔터티로 변환하는 데 도움이 필요한 경우이 옵션을 사용하는 것이 좋습니다.

1. 필요한 경우 애플리케이션으로 이동합니다.
2. 캔버스 상단의 데이터 탭을 선택합니다.
3. 앱에 개체가 없는 경우 + 개체 생성을 선택합니다. 그렇지 않으면 왼쪽 엔터티 메뉴에서 + 추가를 선택합니다.

4. AI를 사용하여 개체 생성을 선택합니다.
5. 개체 이름에 개체의 이름을 입력합니다. 이 이름은 개체의 필드, 데이터 작업 및 샘플 데이터를 생성하는 데 사용됩니다.
6. 데이터 작업 생성 확인란을 선택하여 데이터 작업을 생성합니다.
7. 개체 생성을 선택합니다. 이제 개체가 생성되고 왼쪽 개체 패널에서 해당 개체를 볼 수 있습니다.
8. 의 절차에 따라 새 개체를 구성합니다App Studio 앱에서 개체 구성 또는 편집. 개체가 AI로 생성되었으므로 개체에 이미 생성된 필드가 포함됩니다. 또한 생성 중에 데이터 작업 생성 확인란을 선택한 경우 엔터티에 데이터 작업이 포함됩니다.

App Studio 앱에서 개체 구성 또는 편집

다음 주제를 사용하여 App Studio 애플리케이션에서 개체를 구성합니다.

주제

- 개체 이름 편집
- 개체 필드 추가, 편집 또는 삭제
- 데이터 작업 생성, 편집 또는 삭제
- 샘플 데이터 추가 또는 삭제
- 연결된 데이터 소스 및 맵 필드 추가 또는 편집

개체 이름 편집

1. 필요한 경우 편집하려는 엔터티로 이동합니다.
2. 구성 탭의 개체 이름에서 개체 이름을 업데이트하고 텍스트 상자 외부를 선택하여 변경 사항을 저장합니다.

개체 필드 추가, 편집 또는 삭제

> **Tip**
>
> CTRL+Z를 눌러 엔터티에 대한 최신 변경 사항을 실행 취소할 수 있습니다.

1. 필요한 경우 편집하려는 엔터티로 이동합니다.

2. 구성 탭의 필드에서 개체의 필드 테이블을 볼 수 있습니다. 개체 필드에는 다음 열이 있습니다.

   - 표시 이름: 표시 이름은 테이블 헤더 또는 양식 필드와 유사하며 애플리케이션 사용자가 볼 수 있습니다. 공백과 특수 문자를 포함할 수 있지만 개체 내에서 고유해야 합니다.
   - 시스템 이름: 시스템 이름은 코드에서 필드를 참조하는 데 사용되는 고유 식별자입니다. Amazon Redshift 테이블의 열에 매핑할 때 Amazon Redshift 테이블 열 이름과 일치해야 합니다.
   - 데이터 유형: , `Integer` `Boolean`또는와 같이이 필드 내에 저장될 데이터의 유형입니다`String`.

3. 필드를 추가하려면:

   a. AI를 사용하여 엔터티 이름 및 연결된 데이터 소스를 기반으로 필드를 생성하려면 추가 필드 생성을 선택합니다.

   b. 단일 필드를 추가하려면 + 필드 추가를 선택합니다.

4. 필드를 편집하려면:

   a. 표시 이름을 편집하려면 표시 이름 텍스트 상자에 원하는 값을 입력합니다. 필드의 시스템 이름이 편집되지 않은 경우 표시 이름의 새 값으로 업데이트됩니다.

   b. 시스템 이름을 편집하려면 시스템 이름 텍스트 상자에 원하는 값을 입력합니다.

   c. 데이터 유형을 편집하려면 데이터 유형 드롭다운 메뉴를 선택하고 목록에서 원하는 유형을 선택합니다.

   d. 필드의 속성을 편집하려면 필드의 기어 아이콘을 선택합니다. 다음 목록은 필드 속성을 자세히 설명합니다.

      - 필수: 데이터 소스에 필드가 필요한 경우이 옵션을 활성화합니다.
      - 프라이머리 키: 필드가 데이터 소스의 프라이머리 키에 매핑된 경우이 옵션을 활성화합니다.
      - 고유:이 필드의 값이 고유해야 하는 경우이 옵션을 활성화합니다.
      - 데이터 소스 기본값 사용: 자동 증가 또는 이벤트 타임스탬프를 사용하는 등 데이터 소스에서 필드 값을 제공하는 경우이 옵션을 활성화합니다.
      - 데이터 형식 옵션: 특정 데이터 형식의 필드는 최소값 또는 최대값과 같은 데이터 형식 옵션으로 구성할 수 있습니다.

5. 필드를 삭제하려면 삭제하려는 필드의 휴지통 아이콘을 선택합니다.

데이터 작업 생성, 편집 또는 삭제

데이터 작업은 애플리케이션에서 모든 레코드 가져오기 또는 ID로 레코드 가져오기와 같은 개체의 데이터에 대한 작업을 실행하는 데 사용됩니다. 데이터 작업은 테이블 또는 세부 정보 보기와 같은 구성 요소에서 볼 수 있도록 지정된 조건과 일치하는 데이터를 찾고 반환하는 데 사용할 수 있습니다.

목차

- 데이터 작업 생성
- 데이터 작업 편집 또는 구성
- 데이터 작업 조건 연산자 및 예제
  - 데이터베이스의 조건 연산자 지원
  - 데이터 작업 조건 예제
- 데이터 작업 삭제

데이터 작업 생성

> **Tip**
>
> CTRL+Z를 눌러 엔터티에 대한 최신 변경 사항을 실행 취소할 수 있습니다.

1. 필요한 경우 데이터 작업을 생성하려는 엔터티로 이동합니다.
2. 데이터 작업 탭을 선택합니다.
3. 데이터 작업을 생성하는 방법에는 두 가지가 있습니다.
   - (권장) AI를 사용하여 엔터티 이름, 필드 및 연결된 데이터 소스에 따라 데이터 작업을 생성하려면 데이터 작업 생성을 선택합니다. 다음 작업이 생성됩니다.
     1. `getAll`: 개체에서 모든 레코드를 검색합니다. 이 작업은 레코드 목록을 표시하거나 한 번에 여러 레코드에 대해 작업을 수행해야 하는 경우에 유용합니다.
     2. `getByID`: 고유 식별자(ID 또는 기본 키)를 기반으로 개체에서 단일 레코드를 검색합니다. 이 작업은 특정 레코드에 대한 작업을 표시하거나 수행해야 하는 경우에 유용합니다.
   - 단일 데이터 작업을 추가하려면 + 데이터 작업 추가를 선택합니다.
4. 새 데이터 작업을 보거나 구성하려면 단원을 참조하십시오데이터 작업 편집 또는 구성.

데이터 작업 편집 또는 구성

1. 필요한 경우 데이터 작업을 생성하려는 엔터티로 이동합니다.
2. 데이터 작업 탭을 선택합니다.
3. 필드에서 쿼리에서 반환할 필드를 구성합니다. 기본적으로 개체에 구성된 모든 필드가 선택됩니다.

   다음 단계를 수행하여 데이터 작업에 조인을 추가할 수도 있습니다.

   1. + 조인 추가를 선택하여 대화 상자를 엽니다.
   2. 관련 엔터티에서 현재 엔터티와 조인할 엔터티를 선택합니다.
   3. 별칭에 선택적으로 관련 엔터티의 임시 별칭 이름을 입력합니다.
   4. 조인 유형에서 원하는 조인 유형을 선택합니다.
   5. 각 개체의 필드를 선택하여 조인 절을 정의합니다.
   6. 추가를 선택하여 조인을 생성합니다.

   생성된 조인은 조인 섹션에 표시되므로 반환할 필드 드롭다운에서 추가 필드를 사용할 수 있습니다. 엔터티 간에 연결된 조인을 포함하여 여러 조인을 추가할 수 있습니다. 조인된 엔터티의 필드를 기준으로 필터링하고 정렬할 수도 있습니다.

   조인을 삭제하려면 조인 옆에 있는 휴지통 아이콘을 선택합니다. 그러면 해당 필드를 사용하여 해당 조인에서 모든 필드가 제거되고 종속 조인 또는 제약 조건이 해제됩니다.
4. 조건에서 쿼리 출력을 필터링하는 규칙을 추가, 편집 또는 제거합니다. 규칙을 그룹으로 구성하고 여러 규칙을 AND 또는 OR 문과 함께 연결할 수 있습니다. 사용할 수 있는 연산자에 대한 자세한 내용은 섹션을 참조하세요데이터 작업 조건 연산자 및 예제.
5. 정렬에서 속성을 선택하고 오름차순 또는 내림차순을 선택하여 쿼리 결과를 정렬하는 방법을 구성합니다. 정렬 규칙 옆에 있는 휴지통 아이콘을 선택하여 정렬 구성을 제거할 수 있습니다.
6. 변환 결과에서 사용자 지정 JavaScript를 입력하여 결과를 표시하거나 자동화로 전송하기 전에 수정하거나 형식을 지정할 수 있습니다.
7. 출력 미리 보기에서 구성된 필드, 필터, 정렬 및 JavaScript를 기반으로 쿼리 출력의 미리 보기 테이블을 봅니다.

데이터 작업 조건 연산자 및 예제

조건 연산자를 사용하여 구성된 표현식 값을 개체 열과 비교하여 데이터베이스 객체의 하위 집합을 반환할 수 있습니다. 사용할 수 있는 연산자는 열의 데이터 유형과 Amazon Redshift, Amazon Aurora 또는 Amazon DynamoDB와 같이 개체가 연결된 데이터베이스 유형에 따라 달라집니다.

다음 조건 연산자는 모든 데이터베이스 서비스에 사용할 수 있습니다.

- = 및 !=: 모든 데이터 유형에 사용할 수 있습니다(기본 키 열 제외).
- <=, >=<, 및 >=: 숫자 열에만 사용할 수 있습니다.
- `IS NULL` 및 `IS NOT NULL`: null이거나 빈 값이 있는 열을 일치시키는 데 사용됩니다. Null 값은 종종 각 데이터베이스에서 다르게 해석되지만 App Studio에서는 `NULL` 연산자가 연결된 데이터베이스 테이블에 null 값이 있는 레코드를 일치시키고 반환합니다.

다음 조건 연산자는 이를 지원하는 데이터베이스 서비스에 연결된 엔터티에서만 사용할 수 있습니다.

- `LIKE` 및 `NOT LIKE`(Redshift, Aurora): 연결된 데이터베이스에서 패턴 기반 쿼리를 수행하는 데 사용됩니다. `LIKE` 연산자는 지정된 패턴에 맞는 레코드를 찾아 반환하기 때문에 검색 기능에 유연성을 제공합니다. 패턴 내의 문자 또는 문자 시퀀스와 일치하는 와일드카드 문자를 사용하여 패턴을 정의합니다. 각 데이터베이스 관리 시스템에는 고유한 와일드카드 문자 집합이 있지만 가장 많이 사용되는 두 가지는 원하는 수의 문자(0 포함)_를 % 나타내고 단일 문자를 나타내는 것입니다.
- `Contains` 및 `Not Contains` (DynamoDB): 대소문자를 구분하는 검색을 수행하여 열 값 내에서 지정된 텍스트를 찾을 수 있는지 확인하는 데 사용됩니다.
- `Starts With` 및 `Not Starts With` (DynamoDB): 대/소문자를 구분하는 검색을 수행하여 지정된 텍스트가 열 값의 시작 부분에 있는지 확인하는 데 사용됩니다.

데이터베이스의 조건 연산자 지원

다음 표에는 App Studio에 연결할 수 있는 각 데이터베이스에서 지원되는 데이터 작업 조건 연산자가 나와 있습니다.

	=, !=, <, >, <=, >=	LIKE, NOT LIKE	포함, 포함되지 않음	로 시작하고 로 시작하지 않음	IS NULL, IS NOT NULL
DynamoDB	예	아니요	예	예	예

	=, !=, <, >, <=, >=	LIKE, NOT LIKE	포함, 포함되지 않음	로 시작하고 로 시작하지 않음	IS NULL, IS NOT NULL
Aurora	예	예	아니요	아니요	예
Redshift	예	예	아니요	아니요	예

데이터 작업 조건 예제

, `name` `city`및 `hireDate` 필드가 있는 여러 항목이 포함된 다음 데이터베이스 테이블을 고려하세요.

이름	city	hireDate
Adam	시애틀	2025-03-01
아드리엔	보스턴	2025-03-05
Bob	앨버커키	2025-03-06
Carlos	시카고	2025-03-10
캐롤라인	NULL	2025-03-12
리타	Miami	2025-03-15

이제 App Studio에서 지정된 조건과 일치하는 항목의 `name` 필드를 반환하는 데이터 작업을 생성하는 것이 좋습니다. 다음 목록에는 조건 예제와 테이블이 각각에 대해 반환하는 값이 포함되어 있습니다.

> **Note**
>
> 예제는 SQL 예제로 형식이 지정됩니다. App Studio에서처럼 표시되지 않을 수 있지만 연산자의 동작을 설명하는 데 사용됩니다.

- `WHERE name LIKE 'Adam'`:를 반환합니다`Adam`.
- `WHERE name LIKE 'A%'`: `Adam` 및를 반환합니다`Adrienne`.
- `WHERE name NOT LIKE 'B_B'`: `Adam`, `Adrienne`, `Carlos`, 및 `Caroline`를 반환합니다`Rita`.

- `WHERE contains(name, 'ita')`:를 반환합니다Rita.
- `WHERE begins_with(name, 'Car')`: Carlos 및를 반환합니다Caroline.
- `WHERE city IS NULL`:를 반환합니다Caroline.
- `WHERE hireDate < "2025-03-06"`: Adam 및를 반환합니다Adrienne.
- `WHERE hireDate >= DateTime.now().toISODate()`:는 현재 날짜를 `DateTime.now().toISODate()` 반환합니다. 현재 날짜가 2025-03-10인 시나리오에서 표현식은 , Caroline및 Carlos를 반환합니다Rita.

> **Tip**
>
> 표현식의 날짜 및 시간 비교에 대한 자세한 내용은 섹션을 참조하세요날짜 및 시간.

데이터 작업 삭제

다음 절차에 따라 App Studio 엔터티에서 데이터 작업을 삭제합니다.

1. 필요한 경우 데이터 작업을 삭제할 엔터티로 이동합니다.
2. 데이터 작업 탭을 선택합니다.
3. 삭제하려는 각 데이터 작업에 대해 편집 옆의 드롭다운 메뉴를 선택하고 삭제를 선택합니다.
4. 대화 상자에서 확인을 선택합니다.

샘플 데이터 추가 또는 삭제

App Studio 애플리케이션의 개체에 샘플 데이터를 추가할 수 있습니다. 애플리케이션은 게시될 때까지 외부 서비스와 통신하지 않으므로 샘플 데이터를 사용하여 미리 보기 환경에서 애플리케이션 및 엔터티를 테스트할 수 있습니다.

1. 필요한 경우 편집하려는 엔터티로 이동합니다.
2. 샘플 데이터 탭을 선택합니다.
3. 샘플 데이터를 생성하려면 더 많은 샘플 데이터 생성을 선택합니다.
4. 샘플 데이터를 삭제하려면 삭제하려는 데이터의 확인란을 선택하고 삭제 또는 백스페이스 키를 누릅니다. 저장을 선택하여 변경 사항을 저장합니다.

연결된 데이터 소스 및 맵 필드 추가 또는 편집

> **Tip**
>
> CTRL+Z를 눌러 엔터티에 대한 최신 변경 사항을 실행 취소할 수 있습니다.

1. 필요한 경우 편집하려는 엔터티로 이동합니다.
2. 연결 탭을 선택하여 애플리케이션이 게시될 때 데이터가 저장되는 데이터 소스 테이블과 개체 간의 연결을 보거나 관리합니다. 데이터 소스 테이블이 연결되면 개체 필드를 테이블의 열에 매핑할 수 있습니다.
3. 커넥터에서 원하는 데이터 소스 테이블에 대한 연결이 포함된 커넥터를 선택합니다. 커넥터에 대한 자세한 내용은 단원을 참조하십시오커넥터를 사용하여 App Studio를 다른 서비스에 연결.
4. 테이블에서 개체의 데이터 소스로 사용할 테이블을 선택합니다.
5. 이 표에는 개체의 필드와 개체가 매핑된 데이터 소스 열이 나와 있습니다. 자동 매핑을 선택하여 엔터티 필드를 데이터 소스 열에 자동으로 매핑합니다. 각 개체 필드의 드롭다운에서 데이터 소스 열을 선택하여 테이블의 필드를 수동으로 매핑할 수도 있습니다.

개체 삭제

다음 절차에 따라 App Studio 애플리케이션에서 개체를 삭제합니다.

> **Note**
>
> App Studio 앱에서 엔터티를 삭제해도 관리형 엔터티의 해당 DynamoDB 테이블을 포함하여 연결된 데이터 소스 테이블은 삭제되지 않습니다. 데이터 소스 테이블은 연결된 AWS 계정에 남아 있으며 원하는 경우 해당 서비스에서 삭제해야 합니다.

개체를 삭제하려면

1. 필요한 경우 애플리케이션으로 이동합니다.
2. 데이터 탭을 선택합니다.
3. 왼쪽 엔터티 메뉴에서 삭제하려는 엔터티 옆에 있는 줄임표 메뉴를 선택하고 삭제를 선택합니다.
4. 대화 상자의 정보를 검토하고를 **`confirm`** 입력한 다음 삭제를 선택하여 개체를 삭제합니다.

AWS App Studio의 관리형 데이터 엔터티

일반적으로 외부 데이터베이스 테이블에 연결하여 App Studio에서 엔터티를 구성하며, 연결된 데이터베이스 테이블의 열을 사용하여 각 엔터티 필드를 생성하고 매핑해야 합니다. 데이터 모델을 변경할 때 외부 데이터베이스 테이블과 개체를 모두 업데이트하고 변경된 필드를 다시 매핑해야 합니다. 이 방법은 유연하며 다양한 유형의 데이터 소스를 사용할 수 있지만 더 많은 사전 계획과 지속적인 유지 관리가 필요합니다.

관리형 엔터티는 App Studio가 전체 데이터 스토리지 및 구성 프로세스를 관리하는 엔터티 유형입니다. 관리형 엔터티를 생성하면 연결된 AWS 계정에 해당 DynamoDB 테이블이 생성됩니다. 이를 통해 내에서 안전하고 투명한 데이터 관리를 보장할 수 있습니다 AWS. 관리형 엔터티를 사용하면 App Studio에서 엔터티의 스키마를 구성하고 해당 DynamoDB 테이블도 자동으로 업데이트됩니다.

여러 애플리케이션에서 관리형 엔터티 사용

App Studio 앱에서 관리형 엔터티를 생성하면 해당 엔터티를 다른 App Studio 앱에서 사용할 수 있습니다. 이는 유지할 단일 기본 리소스를 제공하여 동일한 데이터 모델 및 스키마가 있는 앱에 대한 데이터 스토리지를 구성하는 데 유용합니다.

여러 애플리케이션에서 관리형 엔터티를 사용하는 경우 해당 DynamoDB 테이블에 대한 모든 스키마 업데이트는 관리형 엔터티가 생성된 원래 애플리케이션을 사용하여 이루어져야 합니다. 다른 애플리케이션의 엔터티에 대한 스키마 변경 사항은 해당 DynamoDB 테이블을 업데이트하지 않습니다.

관리형 엔터티 제한 사항

기본 키 업데이트 제한: 엔터티가 생성된 후에는 엔터티의 기본 키 이름 또는 유형을 변경할 수 없습니다. 이는 DynamoDB의 파괴적인 변경이며 기존 데이터가 손실될 수 있기 때문입니다.

열 이름 바꾸기: DynamoDB에서 열 이름을 바꾸면 원래 열이 원래 데이터와 함께 유지되는 동안 실제로 새 열을 생성합니다. 원본 데이터는 새 열에 자동으로 복사되거나 원본 열에서 삭제되지 않습니다. 시스템 이름이라고 하는 관리형 엔터티 필드의 이름을 바꿀 수 있지만 원래 열과 해당 데이터에 대한 액세스 권한을 잃게 됩니다. 표시 이름의 이름 변경에는 제한이 없습니다.

데이터 형식 변경: DynamoDB는 테이블 생성 후 열 데이터 형식을 유연하게 수정할 수 있지만 이러한 변경은 쿼리 로직 및 정확도뿐만 아니라 기존 데이터에 심각한 영향을 미칠 수 있습니다. 데이터 형식을 변경하려면 모든 기존 데이터를 새 형식에 맞게 변환해야 합니다.이 형식은 대규모 활성 테이블에서 복잡합니다. 또한 데이터 마이그레이션이 완료될 때까지 데이터 작업은 예상치 못한 결과를 반환할 수 있습니다. 필드의 데이터 유형을 전환할 수 있지만 기존 데이터는 새 데이터 유형으로 마이그레이션되지 않습니다.

정렬 열: DynamoDB는 정렬 키를 통해 정렬된 데이터 검색을 활성화합니다. 정렬 키는 파티션 키와 함께 복합 기본 키의 일부로 정의되어야 합니다. 제한 사항에는 필수 정렬 키, 한 파티션 내에 제한된 정렬, 파티션 간 전역 정렬이 포함되지 않습니다. 핫 파티션을 방지하려면 정렬 키의 신중한 데이터 모델링이 필요합니다. 미리 보기 마일스톤에 대한 정렬은 지원되지 않습니다.

조인: DynamoDB에서는 조인이 지원되지 않습니다. 테이블은 비용이 많이 드는 조인 작업을 방지하기 위해 설계에 따라 비정규화됩니다. one-to-many 관계를 모델링하기 위해 하위 테이블에는 상위 테이블의 기본 키를 참조하는 속성이 포함되어 있습니다. 다중 테이블 데이터 쿼리에는 세부 정보를 검색할 상위 테이블의 항목을 찾는 작업이 포함됩니다. 미리 보기 마일스톤의 일부로 관리형 엔터티에 대한 기본 조인을 지원하지 않습니다. 해결 방법으로 두 개체의 데이터 병합을 수행할 수 있는 자동화 단계를 소개합니다. 이는 한 수준 조회와 매우 유사합니다. 미리 보기 마일스톤에 대한 정렬은 지원되지 않습니다.

Env 단계: 게시는 테스트할 수 있지만 두 환경 모두에서 동일한 관리형 스토어를 사용합니다.

페이지 및 자동화 파라미터

파라미터는 애플리케이션 내의 다양한 구성 요소, 페이지 및 자동화 간에 동적 값을 전달하는 데 사용되는 AWS App Studio의 강력한 기능입니다. 파라미터를 사용하면 유연하고 컨텍스트 인식 환경을 만들어 애플리케이션의 응답성과 개인화를 높일 수 있습니다. 이 문서에서는 페이지 파라미터와 자동화 파라미터라는 두 가지 유형의 파라미터에 대해 다룹니다.

주제

- 페이지 파라미터
- 자동화 파라미터

페이지 파라미터

페이지 파라미터는 페이지 간에 정보를 전송하는 방법이며 App Studio 앱 내에서 한 페이지에서 다른 페이지로 이동하여 컨텍스트를 유지하거나 데이터를 전달할 때 자주 사용됩니다. 페이지 파라미터는 일반적으로 이름과 값으로 구성됩니다.

페이지 파라미터 사용 사례

페이지 파라미터는 App Studio 애플리케이션 내의 여러 페이지와 구성 요소 간에 데이터를 전달하는 데 사용됩니다. 특히 다음과 같은 사용 사례에 유용합니다.

1. 검색 및 필터링: 사용자가 앱의 홈페이지에서 검색할 때 검색어를 결과 페이지에 파라미터로 전달하여 필터링된 관련 항목만 표시할 수 있습니다. 예를 들어 사용자가 `### ## ###`을 검색하는 경우 `### ## ###` 값이 있는 파라미터를 제품 목록 페이지로 전달할 수 있습니다.
2. 항목 세부 정보 보기: 사용자가 제품과 같은 목록을 클릭하면 해당 항목의 고유 식별자를 세부 정보 페이지에 파라미터로 전달할 수 있습니다. 이렇게 하면 세부 정보 페이지에 특정 항목에 대한 모든 정보가 표시됩니다. 예를 들어 사용자가 헤드폰 제품을 클릭하면 제품의 고유 ID가 제품 세부 정보 페이지에 파라미터로 전달됩니다.
3. 페이지 탐색에서 사용자 컨텍스트 전달: 사용자가 페이지 사이를 탐색할 때 파라미터는 사용자의 위치, 기본 제품 범주, 장바구니 콘텐츠 및 기타 설정과 같은 중요한 컨텍스트를 전달할 수 있습니다. 예를 들어 사용자가 앱에서 다양한 제품 범주를 탐색하면 위치 및 기본 범주가 파라미터로 유지되어 개인화되고 일관된 경험을 제공합니다.
4. 딥 링크: 페이지 파라미터를 사용하여 앱 내의 특정 페이지에 대한 링크를 공유하거나 북마크합니다.
5. 데이터 작업: 파라미터 값을 수락하는 데이터 작업을 생성하여 전달된 파라미터를 기반으로 데이터 소스를 필터링하고 쿼리할 수 있습니다. 예를 들어 제품 목록 페이지에서 `category` 파라미터를 수락하여 관련 제품을 가져오는 데이터 작업을 생성할 수 있습니다.

페이지 파라미터 보안 고려 사항

페이지 파라미터는 페이지 간에 데이터를 전달하는 강력한 방법을 제공하지만 올바르게 사용하지 않으면 민감한 정보가 노출될 수 있으므로 신중하게 사용해야 합니다. 다음은 유의해야 할 중요한 보안 고려 사항입니다.

1. URLs.
   a. 위험: 데이터 작업 파라미터를 포함한 URLs은 종종 서버 로그, 브라우저 기록 및 기타 위치에 표시됩니다. 따라서 페이지 파라미터 값에 사용자 자격 증명, 개인 식별 정보(PII) 또는 기타 기밀 데이터와 같은 민감한 데이터가 노출되지 않도록 해야 합니다.
   b. 완화: 민감한 데이터에 안전하게 매핑할 수 있는 식별자를 사용하는 것이 좋습니다. 예를 들어 사용자의 이름이나 이메일 주소를 파라미터로 전달하는 대신 사용자 이름이나 이메일을 가져오는 데 사용할 수 있는 임의의 고유 식별자를 전달할 수 있습니다.

자동화 파라미터

자동화 파라미터는 UI, 기타 자동화 또는 데이터 작업과 같은 다양한 소스에서 동적 값을 전달하여 유연하고 재사용 가능한 자동화를 생성하는 데 사용할 수 있는 App Studio의 강력한 기능입니다. 자동화

가 실행될 때 실제 값으로 대체되는 자리 표시자 역할을 하므로 매번 다른 입력으로 동일한 자동화를 사용할 수 있습니다.

자동화 내에서 파라미터는 고유한 이름을 가지며 파라미터 변수 뒤에와 같은 파라미터 이름을 사용하여 파라미터 값을 참조할 수 있습니다`{{params.customerId}}`.

이 문서에서는 기본 개념, 사용 및 모범 사례를 포함하여 자동화 파라미터에 대한 심층적인 이해를 제공합니다.

자동화 파라미터의 이점

자동화 파라미터는 다음 목록을 포함하여 몇 가지 이점을 제공합니다.

1. 재사용성: 파라미터를 사용하면 서로 다른 입력 값으로 사용자 지정할 수 있는 재사용 가능한 자동화를 생성하여 동일한 자동화 로직을 서로 다른 입력으로 재사용할 수 있습니다.
2. 유연성: 값을 자동화로 하드 코딩하는 대신 파라미터를 정의하고 필요한 경우 다른 값을 제공하여 자동화를 더 역동적이고 적응 가능하게 만들 수 있습니다.
3. 우려 사항 분리: 파라미터는 자동화 로직을 사용된 특정 값과 분리하여 코드 구성 및 유지 관리를 촉진합니다.
4. 검증: 각 파라미터에는 런타임 시 검증되는 문자열, 숫자 또는 부울과 같은 데이터 형식이 있습니다. 이렇게 하면 사용자 지정 검증 코드 없이 잘못된 데이터 형식의 요청이 거부됩니다.
5. 선택적 및 필수 파라미터: 자동화 파라미터를 선택적 또는 필수로 지정할 수 있습니다. 자동화를 실행할 때 필수 파라미터를 제공해야 하지만 선택적 파라미터에는 기본값이 있거나 생략할 수 있습니다. 이러한 유연성을 통해 제공된 파라미터를 기반으로 다양한 시나리오를 처리할 수 있는 보다 다재다능한 자동화를 생성할 수 있습니다.

시나리오 및 사용 사례

시나리오: 제품 세부 정보 검색

제품 ID를 기반으로 데이터베이스에서 제품 세부 정보를 검색하는 자동화가 있다고 가정해 보겠습니다. 이 자동화에는 라는 파라미터가 있을 수 있습니다`productId`.

`productId` 파라미터는 자동화를 실행할 때 실제 제품 ID 값으로 채울 수 있는 자리 표시자 역할을 합니다. 자동화에 특정 제품 ID를 하드 코딩하는 대신 파라미터를 정의하고 자동화를 실행할 때마다 다른 제품 ID 값을 `productId` 전달할 수 있습니다.

이중 중괄호 구문를 사용하여 선택한 제품의 ID를 productId 파라미터로 전달하여 구성 요소의 데이터 소스에서이 자동화를 호출할 수 있습니다{{ui.productsTable.selectedRow.id}}. 이렇게 하면 사용자가 테이블(ui.productsTable)에서 제품을 선택하면 자동화는 선택한 행의 ID를 productId 파라미터로 전달하여 선택한 제품의 세부 정보를 검색합니다.

또는 제품 목록을 반복하고 제품의 ID를 productId 파라미터로 전달하여 각 제품의 세부 정보를 검색하는 다른 자동화에서이 자동화를 호출할 수 있습니다. 이 시나리오에서는 각 루프 반복의 {{product.id}} 표현식에서 productId 파라미터 값이 동적으로 제공됩니다.

productId 파라미터와 이중 중괄호 구문을 사용하면이 자동화를 보다 유연하고 재사용 가능하게 만들 수 있습니다. 각 제품에 대해 별도의 자동화를 생성하는 대신 UI 구성 요소 또는 기타 자동화와 같은 다양한 소스의 파라미터 값으로 적절한 제품 ID를 제공하면 모든 제품의 세부 정보를 검색할 수 있는 단일 자동화를 사용할 수 있습니다.

시나리오: 대체 값이 있는 선택적 파라미터 처리

필수 "소유자" 열이 있는 "작업" 엔터티가 있지만 자동화에서이 필드를 선택 사항으로 설정하고 소유자가 선택되지 않은 경우 대체 값을 제공하는 시나리오를 살펴보겠습니다.

1. Task 엔터티의 Owner 필드에 매핑Owner되는 라는 파라미터로 자동화를 생성합니다.
2. 엔터티에 Owner 필드가 필요하므로 Owner 파라미터는 필수 설정과 동기화됩니다.
3. 자동화에서 Owner 파라미터를 선택 사항으로 설정하려면이 파라미터의 required 설정을 끕니다.
4. 자동화 로직에서와 같은 표현식을 사용할 수 있습니다{{params.Owner || currentUser.userId}}. 이 표현식은 Owner 파라미터가 제공되는지 확인합니다. 제공되지 않으면 현재 사용자의 ID가 소유자로 대체됩니다.
5. 이렇게 하면 사용자가 양식이나 구성 요소에서 소유자를 선택하지 않으면 자동화가 자동으로 현재 사용자를 작업의 소유자로 할당합니다.

Owner 파라미터에 대한 required 설정을 전환하고 폴백 표현식을 사용하면 개체 필드 요구 사항과 분리하여 자동화에서 선택 사항으로 설정하고 파라미터가 제공되지 않을 때 기본값을 제공할 수 있습니다.

자동화 파라미터 유형 정의

파라미터 유형을 사용하여 데이터 유형을 지정하고 요구 사항을 설정하면 자동화에 대한 입력을 제어할 수 있습니다. 이렇게 하면 예상 입력으로 자동화를 안정적으로 실행할 수 있습니다.

개체의 유형 동기화

개체 필드 정의의 파라미터 유형 및 요구 사항을 동적으로 동기화하면 개체 데이터와 상호 작용하는 자동화 구축이 간소화되어 파라미터가 항상 최신 개체 필드 유형 및 요구 사항을 반영할 수 있습니다.

다음 절차에서는 개체의 파라미터 유형을 동기화하기 위한 일반적인 단계를 자세히 설명합니다.

1. 입력 필드(예: 부울, 숫자 등)가 있는 개체를 생성하고 필요에 따라 필드를 표시합니다.
2. 새 자동화를 생성합니다.
3. 자동화에 파라미터를 추가하고 유형을 선택할 때 동기화할 엔터티 필드를 선택합니다. 데이터 유형과 필수 설정은 매핑된 개체 필드에서 자동으로 동기화됩니다.
4. 필요한 경우 각 파라미터에 대해 켜기/끄기로 전환하여 "필수" 설정을 재정의할 수 있습니다. 즉, 필수 상태는 엔터티 필드와 동기화되지 않지만 그렇지 않으면 동기화된 상태로 유지됩니다.

수동으로 유형 정의

개체에서 동기화하지 않고 파라미터 유형을 수동으로 정의할 수도 있습니다.

사용자 지정 파라미터 유형을 정의하면 엔터티 필드 매핑에 의존하지 않고도 특정 입력 유형을 수락하고 필요에 따라 선택적 또는 필수 파라미터를 처리하는 자동화를 생성할 수 있습니다.

1. 입력 필드(예: 부울, 숫자 등)가 있는 개체를 생성하고 필요에 따라 필드를 표시합니다.
2. 새 자동화를 생성합니다.
3. 자동화에 파라미터를 추가하고 유형을 선택할 때 원하는 유형을 선택합니다.

자동화 파라미터에 전달할 동적 값 구성

자동화에 대한 파라미터를 정의한 후에는 자동화를 호출할 때 해당 파라미터에 값을 전달할 수 있습니다. 파라미터 값은 다음 두 가지 방법으로 전달할 수 있습니다.

1. 구성 요소 트리거: 버튼 클릭과 같은 구성 요소 트리거에서 자동화를 호출하는 경우 JavaScript 표현식을 사용하여 구성 요소 컨텍스트의 값을 전달할 수 있습니다. 예를 들어 라는 텍스트 입력 필드가 있는 경우 라는 표현식을 사용하여 해당 값을 이메일 파라미터에 전달할 `emailInput`수 있습니다`ui.emailInput.value`.
2. 기타 자동화: 다른 자동화에서 자동화를 호출하는 경우 JavaScript 표현식을 사용하여 자동화 컨텍스트의 값을 전달할 수 있습니다. 예를 들어 다른 파라미터의 값 또는 이전 작업 단계의 결과를 전달할 수 있습니다.

유형 안전

문자열, 숫자 또는 부울과 같은 특정 데이터 유형으로 파라미터를 정의하면 자동화에 전달되는 값이 예상 유형인지 확인할 수 있습니다.

> **Note**
>
> App Studio에서 date(s)는 ISO 문자열 날짜이며 이러한 날짜도 검증됩니다.

이러한 유형의 안전은 자동화 로직에서 오류 또는 예기치 않은 동작으로 이어질 수 있는 유형 불일치를 방지하는 데 도움이 됩니다. 예를 들어 파라미터를 로 정의하면 해당 파라미터에 전달된 값이 숫자가 될 것이라고 확신할 `Number`수 있으며 자동화 내에서 추가 유형 확인 또는 변환을 수행할 필요가 없습니다.

검증

파라미터에 검증 규칙을 추가하여 자동화에 전달된 값이 특정 기준을 충족하는지 확인할 수 있습니다.

App Studio는 파라미터에 대한 기본 제공 검증 설정을 제공하지 않지만 특정 제약 조건이 위반될 경우 오류가 발생하는 JavaScript 작업을 자동화에 추가하여 사용자 지정 검증을 구현할 수 있습니다.

개체 필드의 경우 최소값/최대값과 같은 검증 규칙의 하위 집합이 지원됩니다. 그러나 레코드 Create/Update/Delete 작업을 실행할 때는 자동화 수준에서 검증되지 않으며 데이터 계층에서만 검증됩니다.

자동화 파라미터 모범 사례

자동화 파라미터가 잘 설계되고 유지 관리 가능하며 사용하기 쉬운지 확인하려면 다음 모범 사례를 따르세요.

1. 설명 파라미터 이름 사용: 파라미터의 용도 또는 컨텍스트를 명확하게 설명하는 파라미터 이름을 선택합니다.
2. 파라미터 설명 제공: 파라미터를 정의할 때 설명 필드를 활용하여 용도, 제약 조건 및 기대치를 설명합니다. 이러한 설명은 파라미터를 참조할 때 JSDoc 설명과 사용자가 자동화를 호출할 때 파라미터 값을 제공해야 하는 모든 사용자 인터페이스에 표시됩니다.
3. 적절한 데이터 형식 사용: 문자열, 숫자, 부울, 객체와 같은 예상 입력 값을 기반으로 각 파라미터의 데이터 형식을 신중하게 고려합니다.
4. 파라미터 값 검증: 추가 작업을 진행하기 전에 자동화 내에서 적절한 검증 검사를 구현하여 파라미터 값이 특정 요구 사항을 충족하는지 확인합니다.

5. 대체 또는 기본값 사용: App Studio는 현재 파라미터의 기본값 설정을 지원하지 않지만 자동화 로직에서 파라미터를 사용할 때 대체 또는 기본값을 구현할 수 있습니다. 예를 들어 `param1` 파라미터가 `{{ params.param1 || "default value" }}` 제공되지 않았거나 값이 false인 경우와 같은 표현식을 사용하여 기본값을 제공할 수 있습니다.
6. 파라미터 일관성 유지: 유사한 파라미터가 필요한 자동화가 여러 개 있는 경우 해당 자동화 전반에서 파라미터 이름 및 데이터 형식의 일관성을 유지하십시오.
7. 문서 파라미터 사용: 각 파라미터에 대한 설명, 용도, 예상 값 및 관련 예제 또는 엣지 케이스를 포함하여 자동화에 대한 명확한 문서를 유지 관리합니다.
8. 자주 검토 및 리팩터링: 자동화 및 파라미터를 정기적으로 검토하여 필요에 따라 파라미터를 리팩터링하거나 통합하여 명확성, 유지 관리 가능성 및 재사용성을 개선합니다.
9. 파라미터 수 제한: 파라미터는 유연성을 제공하지만 파라미터가 너무 많으면 자동화가 복잡해지고 사용하기 어려울 수 있습니다. 파라미터 수를 필요한 것으로만 제한하여 유연성과 단순성 간의 균형을 맞추는 것을 목표로 합니다.
10. 파라미터 그룹화 고려: 여러 관련 파라미터를 정의하는 경우 단일 `##` 파라미터로 그룹화하는 것이 좋습니다.
11. 별도의 우려 사항: 단일 파라미터를 여러 용도로 사용하거나 관련 없는 값을 단일 파라미터로 결합하지 마세요. 각 파라미터는 고유한 문제 또는 데이터 조각을 나타내야 합니다.
12. 파라미터 별칭 사용: 이름이 길거나 복잡한 파라미터가 있는 경우 더 나은 가독성과 유지 관리를 위해 자동화 로직 내에서 별칭 또는 간편 버전을 사용하는 것이 좋습니다.

이러한 모범 사례를 따르면 자동화 파라미터가 잘 설계되고 유지 관리 가능하며 사용하기 쉽고 궁극적으로 자동화의 전반적인 품질과 효율성을 개선할 수 있습니다.

JavaScript를 사용하여 App Studio에서 표현식 작성

AWS App Studio에서 JavaScript 표현식을 사용하여 애플리케이션의 동작과 모양을 동적으로 제어할 수 있습니다. 한 줄 JavaScript 표현식은 이중 중괄호, 내에 작성되며 자동화`{{ }}`, UI 구성 요소 및 데이터 쿼리와 같은 다양한 컨텍스트에서 사용할 수 있습니다. 이러한 표현식은 런타임에 평가되며 계산을 수행하고, 데이터를 조작하고, 애플리케이션 로직을 제어하는 데 사용할 수 있습니다.

App Studio는 앱 구성 내에서 JavaScript 구문 및 유형 확인 오류를 감지하기 위한 SDK 통합뿐만 아니라 Luxon, UUID, Lodash의 세 가지 JavaScript 오픈 소스 라이브러리에 대한 기본 지원을 제공합니다.

> **Important**
>
> App Studio는 타사 또는 사용자 지정 JavaScript 라이브러리 사용을 지원하지 않습니다.

기본 구문

JavaScript 표현식에는 변수, 리터럴, 연산자 및 함수 호출이 포함될 수 있습니다. 표현식은 일반적으로 계산을 수행하거나 조건을 평가하는 데 사용됩니다.

다음 예를 참조하세요.

- `{{ 2 + 3 }}`는 5로 평가됩니다.
- `{{ "Hello, " + "World!" }}`는 "Hello, World!"로 평가됩니다.
- `{{ Math.max(5, 10) }}`는 10으로 평가됩니다.
- `{{ Math.random() * 10 }}`는 [0~10) 사이의 난수(십진수 포함)를 반환합니다.

보간

JavaScript를 사용하여 정적 텍스트 내에서 동적 값을 보간할 수도 있습니다. 이는 다음 예제와 같이 이중 중괄호 안에 JavaScript 표현식을 묶어 달성합니다.

```
Hello {{ currentUser.firstName }}, welcome to App Studio!
```

이 예제에서 `currentUser.firstName`는 현재 사용자의 이름을 검색한 다음 인사말 메시지에 동적으로 삽입되는 JavaScript 표현식입니다.

연결

다음 예제와 같이 JavaScript에서 + 연산자를 사용하여 문자열과 변수를 연결할 수 있습니다.

```
{{ currentRow.FirstName + " " + currentRow.LastName }}
```

이 표현식은 `currentRow.FirstName` 및의 값을 사이의 공백`currentRow.LastName`과 결합하여 현재 행의 전체 이름을 생성합니다. 예를 들어 `currentRow.FirstName`가 `John`이고 `currentRow.LastName`가 `Doe`인 경우 표현식은 로 확인됩니다`John Doe`.

날짜 및 시간

JavaScript는 날짜 및 시간 작업을 위한 다양한 함수와 객체를 제공합니다. 예제:

- `{{ new Date().toLocaleDateString() }}`: 현재 날짜를 현지화된 형식으로 반환합니다.
- `{{ DateTime.now().toISODate() }}`: 날짜 구성 요소에 사용할 수 있도록 현재 날짜를 YYYY-MM-DD 형식으로 반환합니다.

날짜 및 시간 비교

=, >, <>=, 또는 등의 연산자를 사용하여 날짜 또는 시간 값을 비교<=합니다. 예제:

- `{{ui.timeInput.value > "10:00 AM"}}`: 시간이 오전 10시 이후인지 확인합니다.
- `{{ui.timeInput.value <= "5:00 PM"}}`: 시간이 오후 5시 이전인지 확인합니다.
- `{{ui.timeInput.value > DateTime.now().toISOTime()}}`: 시간이 현재 시간 이후인지 확인합니다.
- `{{ui.dateInput.value > DateTime.now().toISODate()}}`: 날짜가 현재 날짜 이전인지 확인합니다.
- `{{ DateTime.fromISO(ui.dateInput.value).diff(DateTime.now(), "days").days >= 5 }}`: 날짜가 현재 날짜로부터 최소 5일인지 확인합니다.

코드 블록

표현식 외에도 여러 줄의 JavaScript 코드 블록을 작성할 수도 있습니다. 표현식과 달리 코드 블록에는 중괄호가 필요하지 않습니다. 대신 코드 블록 편집기 내에서 JavaScript 코드를 직접 작성할 수 있습니다.

> **Note**
>
> 표현식이 평가되고 해당 값이 표시되는 동안 코드 블록이 실행되고 해당 출력(있는 경우)이 표시됩니다.

글로벌 변수 및 함수

App Studio는 JavaScript 표현식 및 코드 블록 내에서 사용할 수 있는 특정 전역 변수 및 함수에 대한 액세스를 제공합니다. 예를 들어 `currentUser`는 현재 로그인한 사용자를 나타내는 글로벌 변수이며와 같은 속성에 액세스`currentUser.role`하여 사용자의 역할을 검색할 수 있습니다.

UI 구성 요소 값 참조 또는 업데이트

구성 요소 및 자동화 작업에서 표현식을 사용하여 UI 구성 요소 값을 참조하고 업데이트할 수 있습니다. 구성 요소 값을 프로그래밍 방식으로 참조하고 업데이트하면 사용자 입력 및 데이터 변경에 응답하는 동적 대화형 사용자 인터페이스를 생성할 수 있습니다.

UI 구성 요소 값 참조

UI 구성 요소의 값에 액세스하여 동적 동작을 구현하여 대화형 데이터 기반 애플리케이션을 생성할 수 있습니다.

표현식의 `ui` 네임스페이스를 사용하여 동일한 페이지에서 UI 구성 요소의 값 및 속성에 액세스할 수 있습니다. 구성 요소의 이름을 참조하여 해당 값을 검색하거나 상태에 따라 작업을 수행할 수 있습니다.

> **Note**
>
> `ui` 구성 요소의 범위가 해당 페이지로 지정되므로 네임스페이스에는 현재 페이지에만 구성 요소가 표시됩니다.

App Studio 앱에서 구성 요소를 참조하기 위한 기본 구문은 입니다`{{ui.componentName}}`.

다음 목록에는 `ui` 네임스페이스를 사용하여 UI 구성 요소 값에 액세스하는 예제가 포함되어 있습니다.

- `{{ui.textInputName.value}}`: *textInputName*이라는 텍스트 입력 구성 요소의 값을 나타냅니다.
- `{{ui.formName.isValid}}`: 제공된 검증 기준에 따라 *formName* 형식의 모든 필드가 유효한지 확인합니다.
- `{{ui.tableName.currentRow.columnName}}`: *tableName*이라는 테이블 구성 요소의 현재 행에 있는 특정 열의 값을 나타냅니다.
- `{{ui.tableName.selectedRowData.fieldName}}`: *tableName*이라는 테이블 구성 요소에서 선택한 행의 지정된 필드 값을 나타냅니다. 그런 다음 `ID`

({{ui.*tableName*.selectedRowData.*ID*}})와 같은 필드 이름을 추가하여 선택한 행에서 해당 필드의 값을 참조할 수 있습니다.

다음 목록에는 구성 요소 값을 참조하는 보다 구체적인 예제가 포함되어 있습니다.

- {{ui.*inputText1*.value.trim().length > 0}}: 선행 또는 후행 공백을 잘라낸 후 *inputText1* 구성 요소의 값에 비어 있지 않은 문자열이 있는지 확인합니다. 이는 사용자 입력을 검증하거나 입력 텍스트 필드의 값을 기반으로 다른 구성 요소를 활성화/비활성화하는 데 유용할 수 있습니다.
- {{ui.*multiSelect1*.value.join(", ")}}: *multiSelect1*이라는 다중 선택 구성 요소의 경우이 표현식은 선택한 옵션 값의 배열을 쉼표로 구분된 문자열로 변환합니다. 이는 선택한 옵션을 사용자에게 친숙한 형식으로 표시하거나 선택 항목을 다른 구성 요소 또는 자동화에 전달하는 데 도움이 될 수 있습니다.
- {{ui.*multiSelect1*.value.includes("*option1*")}}:이 표현식은 값 *option1*이 *multiSelect1* 구성 요소에 대해 선택한 옵션의 배열에 포함되어 있는지 확인합니다. *option1*을 선택하면 true를 반환하고, 그렇지 않으면 false를 반환합니다. 이는 구성 요소를 조건부로 렌더링하거나 특정 옵션 선택에 따라 작업을 수행하는 데 유용할 수 있습니다.
- {{ui.*s3Upload1*.files.length > 0}}: s3Upload1이라는 Amazon S3 파일 업로드 구성 요소의 경우이 표현식은 파일 배열의 길이를 확인하여 파일이 업로드되었는지 확인합니다. *s3Upload1* 파일이 업로드되었는지 여부에 따라 다른 구성 요소 또는 작업을 활성화/비활성화하는 데 유용할 수 있습니다.
- {{ui.*s3Upload1*.files.filter(file => file.type === "*image/png*").length}}:이 표현식은 PNG 이미지 파일만 포함하도록 *s3Upload1* 구성 요소에 업로드된 파일 목록을 필터링하고 해당 파일의 수를 반환합니다. 이는 업로드된 파일 유형에 대한 정보를 검증하거나 표시하는 데 도움이 될 수 있습니다.

UI 구성 요소 값 업데이트

구성 요소의 값을 업데이트하거나 조작하려면 자동화 RunComponentAction 내에서를 사용합니다. 다음은 RunComponentAction 작업을 사용하여 *myInput*이라는 텍스트 입력 구성 요소의 값을 업데이트하는 데 사용할 수 있는 구문의 예입니다.

```
RunComponentAction(ui.myInput, "setValue", "New Value")
```

이 예제에서 RunComponentAction 단계는 *myInput* 구성 요소에 대한 setValue 작업을 호출하여 새 값인 *# ##* 전달합니다.

테이블 데이터 작업

테이블 데이터 및 값에 액세스하여 작업을 수행할 수 있습니다. 다음 표현식을 사용하여 테이블 데이터에 액세스할 수 있습니다.

- currentRow: 테이블 내의 현재 행에서 테이블 데이터에 액세스하는 데 사용됩니다. 예를 들어 테이블 작업의 이름을 설정하거나, 작업에서 시작된 자동화로 행의 값을 보내거나, 테이블의 기존 열에서 값을 사용하여 새 열을 생성합니다.
- ui.*tableName*.selectedRow 및 ui.*tableName*.selectedRowData는 모두 페이지의 다른 구성 요소에서 테이블 데이터에 액세스하는 데 사용됩니다. 예를 들어 선택한 행을 기반으로 테이블 외부에서 버튼 이름을 설정합니다. 반환되는 값은 동일하지만 selectedRow와 간의 차이점selectedRowData은 다음과 같습니다.
  - selectedRow:이 네임스페이스에는 각 필드의 열 헤더에 표시된 이름이 포함됩니다. 테이블에 표시되는 열의 값을 참조할 selectedRow 때를 사용해야 합니다. 예를 들어 테이블에 개체의 필드로 존재하지 않는 사용자 지정 또는 계산된 열이 있는 경우입니다.
  - selectedRowData:이 네임스페이스에는 테이블의 소스로 사용되는 개체의 필드가 포함됩니다. selectedRowData를 사용하여 테이블에 표시되지 않지만 앱의 다른 구성 요소 또는 자동화에 유용한 개체의 값을 참조해야 합니다.

다음 목록에는 표현식의 테이블 데이터에 액세스하는 예제가 포함되어 있습니다.

- {{ui.*tableName*.selectedRow.*columnNameWithNoSpace*}}: 테이블에서 선택한 행의 *columnNameWithNoSpace* 열 값을 반환합니다.
- {{ui.*tableName*.selectedRow.['*Column Name With Space*']}}: 테이블에서 선택한 행의 *### ## # ##* 열의 값을 반환합니다.
- {{ui.*tableName*.selectedRowData.*fieldName*}}: 테이블에서 선택한 행의 *fieldName* 엔터티 필드 값을 반환합니다.
- {{ui.*tableName*.selectedRows[0].*columnMappingName*}}: 동일한 페이지의 다른 구성 요소 또는 표현식에서 선택한 행의 열 이름을 참조합니다.
- {{currentRow.*firstName* + ' ' + currentRow.*lastNamecolumnMapping*}}: 여러 열의 값을 연결하여 테이블에 새 열을 생성합니다.
- {{ { "Blocked": "#", "Delayed": "#", "On track": "#" } [currentRow.*statuscolumnMapping*] + " " + currentRow.*statuscolumnMapping*}}: 저장된 상태 값을 기반으로 테이블 내 필드의 표시 값을 사용자 지정합니다.

- {{currentRow.*colName*}}, {{currentRow["*First Name*"]}}{{currentRow}}, 또는 {{ui.*tableName*.selectedRows[0]}}: 행 작업 내에서 참조된 행의 컨텍스트를 전달합니다.

자동화 액세스

자동화를 사용하여 App Studio에서 서버 측 로직 및 작업을 실행할 수 있습니다. 자동화 작업 내에서 표현식을 사용하여 데이터를 처리하고, 동적 값을 생성하고, 이전 작업의 결과를 통합할 수 있습니다.

자동화 파라미터 액세스

UI 구성 요소 및 기타 자동화의 동적 값을 자동화에 전달하여 재사용 가능하고 유연하게 만들 수 있습니다. 이는 다음과 같이 params 네임스페이스가 있는 자동화 파라미터를 사용하여 수행됩니다.

{{params.*parameterName*}}: UI 구성 요소 또는 기타 소스에서 자동화로 전달된 값을 참조합니다. 예를 들어는 *ID*라는 파라미터를 참조{{params.*ID*}}합니다.

자동화 파라미터 조작

JavaScript를 사용하여 자동화 파라미터를 조작할 수 있습니다. 다음 예를 참조하세요.

- {{params.*firstName*}} {{params.*lastName*}}: 파라미터로 전달된 값을 연결합니다.
- {{params.*numberParam1* + params.*numberParam2*}}: 두 개의 숫자 파라미터를 추가합니다.
- {{params.*valueProvided*?.length > 0 ? params.*valueProvided* : '*Default*'}}: 파라미터가 null이거나 정의되지 않았으며 길이가 0이 아닌지 확인합니다. true인 경우 제공된 값을 사용하고, 그렇지 않으면 기본값을 설정합니다.
- {{params.*rootCause* || "*No root cause provided*"}}: params.*rootCause* 파라미터가 false(null, undefined 또는 빈 문자열)인 경우 제공된 기본값을 사용합니다.
- {{Math.min(params.numberOfProducts, 100)}}: 파라미터 값을 최대값(이 경우)으로 제한합니다100.
- {{ DateTime.fromISO(params.startDate).plus({ days: 7 }).toISO() }}: params.*startDate* 파라미터가 인 경우 "2023-06-15T10:30:00.000Z"이 표현식은 시작 날짜 1주일 후의 날짜"2023-06-22T10:30:00.000Z"인 로 평가됩니다.

이전 작업의 자동화 결과에 액세스

자동화를 통해 애플리케이션은 데이터베이스 쿼리, APIs와의 상호 작용 또는 데이터 변환 수행과 같은 서버 측 로직 및 작업을 실행할 수 있습니다. results 네임스페이스는 동일한 자동화 내에서 이전 작

업에서 반환된 출력 및 데이터에 대한 액세스를 제공합니다. 자동화 결과 액세스에 대한 다음 사항에 유의하세요.

1. 동일한 자동화 내에서 이전 자동화 단계의 결과에만 액세스할 수 있습니다.
2. *action1* 및 *action2*라는 작업이 순서대로 있는 경우 *action1*은 결과를 참조할 수 없으며 *action2*는 에만 액세스할 수 있습니다`results.action1`.
3. 이는 클라이언트 측 작업에서도 작동합니다. 예를 들어 `InvokeAutomation` 작업을 사용하여 자동화를 트리거하는 버튼이 있는 경우입니다. 그런 다음 자동화에서 파일이 PDF로 표시되는 경우와 같은 `Run If` 조건으로 탐색 단계를 수행하여 PDF 뷰어가 있는 페이지로 `results.myInvokeAutomation1.fileType === "pdf"` 이동할 수 있습니다.

다음 목록에는 `results` 네임스페이스를 사용하여 이전 작업의 자동화 결과에 액세스하는 구문이 포함되어 있습니다.

- `{{results.stepName.data}}`: *stepName*이라는 자동화 단계에서 데이터 배열을 검색합니다.
- `{{results.stepName.output}}`: *stepName*이라는 자동화 단계의 출력을 검색합니다.

자동화 단계의 결과에 액세스하는 방법은 작업 유형과 반환되는 데이터에 따라 달라집니다. 작업마다 속성 또는 데이터 구조가 다를 수 있습니다. 다음은 몇 가지 일반적인 예제입니다.

- 데이터 작업의 경우를 사용하여 반환된 데이터 배열에 액세스할 수 있습니다`results.stepName.data`.
- API 호출 작업의 경우를 사용하여 응답 본문에 액세스할 수 있습니다`results.stepName.body`.
- Amazon S3 작업의 경우를 사용하여 파일 콘텐츠에 액세스할 수 있습니다`results.stepName.Body.transformToWebStream()`.

반환되는 데이터의 형태와 `results` 네임스페이스 내에서 액세스하는 방법을 이해하려면 사용 중인 특정 작업 유형에 대한 설명서를 참조하세요. 다음 목록에는 몇 가지 예가 포함되어 있습니다.

- `{{results.getDataStep.data.filter(row => row.status === "pending").length}}`: *getDataStep*이 데이터 행의 배열을 반환하는 `Invoke Data Action` 자동화 작업이라고 가정하면이 표현식은 상태 필드가와 같은 행만 포함하도록 데이터 배열을 필터링`pending`하고 필터링된 배열의 길이(개수)를 반환합니다. 이는 특정 조건에 따라 데이터를 쿼리하거나 처리하는 데 유용할 수 있습니다.

- {{params.*email*.split("@")[0]}}: params.*email* 파라미터에 이메일 주소가 포함된 경우 이 표현식은 @ 기호에서 문자열을 분할하고 @ 기호 앞에 있는 부분을 반환하여 이메일 주소의 사용자 이름 부분을 효과적으로 추출합니다.
- {{new Date(params.*timestamp* * 1000)}}:이 표현식은 Unix 타임스탬프 파라미터(params.*timestamp*)를 가져와 JavaScript Date 객체로 변환합니다. 타임스탬프가 초 단위라고 가정하므로 타임스탬프에 1000을 곱하여 Date 생성자가 예상하는 형식인 밀리초로 변환합니다. 이는 자동화에서 날짜 및 시간 값을 사용하는 데 유용할 수 있습니다.
- {{results.*stepName*.Body}}: *stepName*이라는 Amazon S3 GetObject 자동화 작업의 경우 이 표현식은 파일 콘텐츠를 검색하며,이 내용은 이미지 또는 PDF 뷰어와 같은 UI 구성 요소에서 검색된 파일을 표시하는 데 사용할 수 있습니다. 이 표현식은 구성 요소에 사용할 자동화의 자동화 출력에서 구성해야 합니다.

데이터 종속성 및 타이밍 고려 사항

App Studio에서 복잡한 애플리케이션을 구축할 때는 양식, 세부 정보 보기, 자동화 기반 구성 요소와 같은 다양한 데이터 구성 요소 간의 데이터 종속성을 이해하고 관리하는 것이 중요합니다. 데이터 구성 요소와 자동화가 동시에 데이터 검색 또는 실행을 완료하지 못하여 타이밍 문제, 오류 및 예상치 못한 동작이 발생할 수 있습니다. 잠재적인 타이밍 문제를 파악하고 모범 사례를 따르면 App Studio 애플리케이션에서 보다 안정적이고 일관된 사용자 경험을 만들 수 있습니다.

몇 가지 잠재적 문제는 다음과 같습니다.

1. 렌더링 타이밍 충돌: 데이터 구성 요소가 데이터 종속성과 일치하지 않는 순서로 렌더링되어 시각적 불일치 또는 오류가 발생할 수 있습니다.
2. 자동화 실행 타이밍: 구성 요소가 완전히 로드되기 전에 자동화 작업이 완료되어 런타임 실행 오류가 발생할 수 있습니다.
3. 구성 요소 충돌: 자동화로 구동되는 구성 요소는 잘못된 응답 또는 자동화 실행이 완료되지 않은 경우 충돌할 수 있습니다.

예: 주문 세부 정보 및 고객 정보

이 예제에서는 데이터 구성 요소 간의 종속성으로 인해 타이밍 문제가 발생하고 데이터 표시에 잠재적 오류가 발생할 수 있는 방법을 보여줍니다.

동일한 페이지에 다음 두 가지 데이터 구성 요소가 있는 애플리케이션을 생각해 보세요.

- 주문 데이터를 가져오는 세부 정보 구성 요소(orderDetails)입니다.

- 주문과 관련된 고객 세부 정보를 표시하는 세부 정보 구성 요소(customerDetails)입니다.

이 애플리케이션에는 orderDetails 세부 정보 구성 요소에 다음 값으로 구성된 두 개의 필드가 있습니다.

```
// 2 text fields within the orderDetails detail component

// Info from orderDetails Component
{{ui.orderDetails.data[0].name}}

// Info from customerDetails component
{{ui.customerDetails.data[0].name}} // Problematic reference
```

이 예제에서는 orderDetails 구성 요소가 customerDetails 구성 요소의 데이터를 참조하여 고객 이름을 표시하려고 시도합니다. 이는 구성 orderDetails 요소가 데이터를 customerDetails 가져오기 전에 렌더링될 수 있기 때문에 문제가 됩니다. customerDetails 구성 요소 데이터 가져오기가 지연되거나 실패하면 구성 orderDetails 요소에 불완전하거나 잘못된 정보가 표시됩니다.

데이터 종속성 및 타이밍 모범 사례

다음 모범 사례를 사용하여 App Studio 앱의 데이터 종속성 및 타이밍 문제를 완화합니다.

1. 조건부 렌더링 사용: 구성 요소를 렌더링하거나 사용 가능한지 확인한 경우에만 데이터를 표시합니다. 조건문을 사용하여 데이터를 표시하기 전에 데이터 존재 여부를 확인합니다. 다음 코드 조각은 조건문의 예를 보여줍니다.

   ```
   {{ui.someComponent.data ? ui.someComponent.data.fieldName : "Loading..."}}
   ```

2. 하위 구성 요소 가시성 관리: 데이터가 로드되기 전에 하위 구성 요소를 렌더링하는 Stepflow, Form 또는 Detail과 같은 구성 요소의 경우 하위 구성 요소의 가시성을 수동으로 설정합니다. 다음 코드 조각은 상위 구성 요소 데이터 가용성을 기반으로 가시성을 설정하는 예를 보여줍니다.

   ```
   {{ui.parentComponent.data ? true : false}}
   ```

3. 조인 쿼리 사용: 가능하면 조인 쿼리를 사용하여 단일 쿼리에서 관련 데이터를 가져옵니다. 이렇게 하면 별도의 데이터 가져오기 수가 줄어들고 데이터 구성 요소 간의 타이밍 문제가 최소화됩니다.
4. 자동화에서 오류 처리 구현: 자동화에서 강력한 오류 처리를 구현하여 예상 데이터를 사용할 수 없거나 잘못된 응답이 수신되는 시나리오를 정상적으로 관리합니다.

5. 선택적 연결 사용: 중첩 속성에 액세스할 때 상위 속성이 정의되지 않은 경우 오류를 방지하기 위해 선택적 연결 기능을 사용합니다. 다음 코드 조각은 선택적 연결의 예를 보여줍니다.

```
{{ui.component.data?.[0]?.fieldSystemName}}
```

여러 사용자로 앱 빌드

여러 사용자가 단일 App Studio 앱에서 작업할 수 있지만 한 번에 한 명의 사용자만 앱을 편집할 수 있습니다. 앱을 편집하도록 다른 사용자를 초대하는 방법과 여러 사용자가 앱을 동시에 편집하려고 할 때의 동작에 대한 자세한 내용은 다음 섹션을 참조하세요.

빌더가 앱을 편집하도록 초대

다음 지침에 따라 App Studio 앱을 편집하도록 다른 빌더를 초대합니다.

앱을 편집하도록 다른 빌더를 초대하려면

1. 필요한 경우 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 공유를 선택합니다.
3. 개발 탭에서 텍스트 상자를 사용하여 앱 편집에 초대하려는 그룹 또는 개별 사용자를 검색하고 선택합니다.
4. 각 사용자 또는 그룹에 대해 드롭다운을 선택하고 해당 사용자 또는 그룹에 부여할 권한을 선택합니다.

    - 공동 소유자: 공동 소유자는 앱 소유자와 동일한 권한을 갖습니다.
    - 편집 전용: 편집 전용 역할이 있는 사용자는 다음을 제외하고 소유자 및 공동 소유자와 동일한 권한을 갖습니다.
        - 앱을 편집하도록 다른 사용자를 초대할 수 없습니다.
        - 테스트 또는 프로덕션 환경에는 앱을 게시할 수 없습니다.
        - 앱에 데이터 소스를 추가할 수 없습니다.
        - 앱을 삭제하거나 복제할 수 없습니다.

다른 사용자가 편집 중인 앱 편집 시도

App Studio 앱은 한 번에 한 명의 사용자만 편집할 수 있습니다. 여러 사용자가 앱을 동시에 편집하려고 하면 어떻게 되는지 알아보려면 다음 예제를 참조하세요.

이 예제에서 User A는 현재 앱을 편집하고 있으며와 공유했습니다User B. 그런 User B 다음는에서 편집 중인 앱을 편집하려고 시도합니다User A.

User B가 앱을 편집하려고 하면 현재 앱을 편집 중User A이고 계속 진행하면 애플리케이션 스튜디오User A에서 시작되며 모든 변경 사항이 저장된다는 대화 상자가 표시됩니다.는 취소하고 User A 계속하도록 선택하거나 계속 진행하고 애플리케이션 스튜디오에 들어가 앱을 편집하도록 선택할 User B 수 있습니다. 이 예제에서는 앱을 편집하도록 선택합니다.

가 앱을 편집하기로 User B 선택하면 User B는 앱 편집을 시작했고 세션이 종료되었다는 알림을 User A 수신합니다. 앱이 비활성 브라우저 탭에서 열려 User A 있는 경우 알림을 받지 못할 수 있습니다. 이 경우 앱으로 돌아가서 편집하려고 하면 오류 메시지가 표시되고 페이지를 새로 고치도록 안내되어 애플리케이션 목록으로 돌아갑니다.

앱의 콘텐츠 보안 설정 보기 또는 업데이트

App Studio의 모든 애플리케이션에는 이미지, iFrames 및 PDFs와 같은 외부 미디어 또는 리소스가 로드되지 않도록 제한하거나 지정된 도메인 또는 URLs(Amazon S3 버킷 포함)에서만 허용되는 콘텐츠 보안 설정이 있습니다. 앱이 Amazon S3에 객체를 업로드할 수 있는 도메인을 지정할 수도 있습니다.

모든 앱의 기본 콘텐츠 보안 설정은 Amazon S3 버킷을 포함한 외부 소스에서 모든 미디어 로드를 차단하고 Amazon S3에 객체 업로드를 차단하는 것입니다. 따라서 이미지, iFrames, PDFs 또는 유사한 미디어를 로드하려면 미디어 소스를 허용하도록 설정을 편집해야 합니다. 또한 Amazon S3에 객체 업로드를 허용하려면 업로드할 수 있는 도메인을 허용하도록 설정을 편집해야 합니다.

> **Note**
>
> 콘텐츠 보안 설정은 애플리케이션에서 콘텐츠 보안 정책(CSP) 헤더를 구성하는 데 사용됩니다. CSP는 교차 사이트 스크립팅(XSS), 클릭재킹 및 기타 코드 삽입 공격으로부터 앱을 보호하는 데 도움이 되는 보안 표준입니다. CSP에 대한 자세한 내용은 MDN 웹 문서의 콘텐츠 보안 정책(CSP)을 참조하세요.

앱의 콘텐츠 보안 설정을 업데이트하려면

1. 필요한 경우 애플리케이션 목록에서 편집하도록 선택하여 애플리케이션의 애플리케이션 스튜디오로 이동합니다.
2. 앱 설정을 선택합니다.
3. 콘텐츠 보안 설정 탭을 선택하여 다음 설정을 확인합니다.

- 프레임 소스: 앱이 프레임 및 iframe(예: 대화형 콘텐츠 또는 PDFs)을 로드할 수 있는 도메인을 관리하는 데 사용됩니다. 이 설정은 다음 구성 요소 또는 앱 리소스에 영향을 줍니다.
  - iFrame 임베드 구성 요소
  - PDF 뷰어 구성 요소
- 이미지 소스: 앱이 이미지를 로드할 수 있는 도메인을 관리하는 데 사용됩니다. 이 설정은 다음 구성 요소 또는 앱 리소스에 영향을 줍니다.
  - 앱 로고 및 배너
  - 이미지 뷰어 구성 요소
- 소스 연결: 앱이 Amazon S3 객체를 업로드할 수 있는 도메인을 관리하는 데 사용됩니다.

4. 각 설정에 대해 드롭다운에서 원하는 설정을 선택합니다.

- 모든 frames/images/connections 차단: 미디어(이미지, 프레임, PDFs) 또는 Amazon S3에 업로드할 객체를 로드하지 마세요.
- 모든 frames/images/connections 허용: 모든 도메인의 모든 미디어(이미지, 프레임, PDFs)를 로드하거나 모든 도메인에 대해 Amazon S3에 객체 업로드를 허용합니다.
- 특정 도메인 허용:에서 미디어를 로드하거나 지정된 도메인에 미디어를 업로드할 수 있습니다. 도메인 또는 URLs은 공백으로 구분된 표현식 목록으로 지정됩니다. 여기서 와일드카드(*)를 하위 도메인, 호스트 주소 또는 포트 번호에 사용하여 각의 모든 법적 값이 유효함을 나타낼 수 있습니다. 를 지정하면 `http`도 일치합니다`https`. 다음 목록에는 유효한 항목의 예가 포함되어 있습니다.
  - `blob:`: Amazon S3 버킷에서 항목 반환 또는 Amazon Bedrock에서 생성된 이미지와 같이 자동화 작업에서 `GetObject` 반환된 파일 데이터를 포함하는 모든 BLOB과 일치합니다.

    > **Important**
    >
    > 제공된 표현`blob:`식에를 포함하여 작업에서 반환되는 파일 데이터를 허용해야 합니다. 표현식이 인 경우에도 로 업데이트*해야 합니다. `* blob:`

  - `http://*.example.com`:의 하위 도메인에서 로드하려는 모든 시도와 일치합니다`example.com`. `https` 리소스와도 일치합니다.
  - `https://source1.example.com https//source2.example.com`: `https://source1.example.com` 및 모두에서 로드하려는 모든 시도와 일치합니다. `https://source2.example.com`

- `https://example.com/subdirectory/`: 하위 디렉터리 디렉터리에서 파일을 로드하려는 모든 시도와 일치합니다. 예를 들어 `https://example.com/subdirectory/path/to/file.jpeg`입니다. 와 일치하지 않습니다`https://example.com/path/to/file.jpeg`.

5. 저장을 선택하여 변경 사항을 저장합니다.

App Studio 문제 해결 및 디버깅

주제

- App Studio 설정, 권한 및 온보딩 문제 해결
- 앱 문제 해결 및 디버깅
- 애플리케이션 게시 및 공유 문제 해결

App Studio 설정, 권한 및 온보딩 문제 해결

이 주제에는 App Studio를 설정하거나 온보딩할 때 발생하는 일반적인 문제 해결 및 권한 관리에 대한 정보가 포함되어 있습니다.

계정 인스턴스 생성 옵션을 선택할 때 App Studio 설정이 실패했습니다.

문제: IAM Identity Center는 하나의 인스턴스만 지원하므로 모든 리전에 계정 수준 IAM Identity Center 인스턴스가 있는 경우 계정 인스턴스 생성을 사용하여 App Studio를 설정하면 실패합니다. AWS

해결 방법: https://console.aws.amazon.com/singlesignon/ IAM Identity Center 콘솔로 이동하여 IAM Identity Center 인스턴스가 있는지 확인합니다. 인스턴스를 찾을 때까지 지원되는 모든 AWS 리전을 확인합니다. App Studio를 설정할 때 해당 인스턴스를 사용하거나 IAM Identity Center 인스턴스를 삭제하고 계정 인스턴스 생성 옵션을 사용하여 다시 시도할 수 있습니다.

> **Warning**
>
> IAM Identity Center 인스턴스를 삭제하면 기존 사용 사례에 영향을 미칩니다. 삭제하기 전에 인스턴스가 사용되고 있지 않은지 확인하거나 인스턴스를 사용하여 App Studio를 설정합니다.

설정 후 App Studio에 액세스할 수 없음

문제: App Studio를 설정할 때 멤버가 아닌 IAM Identity Center 그룹을 제공했을 수 있습니다. App Studio에 액세스하려면 하나 이상의 그룹에 속해야 합니다.

해결 방법: https://console.aws.amazon.com/singlesignon/ IAM Identity Center 콘솔로 이동하여 설정 시 App Studio에 추가된 그룹에 자신을 추가합니다.

App Studio에 로그인할 때 사용할 사용자 이름 또는 암호가 확실하지 않음

문제: IAM Identity Center 자격 증명을 설정하지 않았거나 IAM Identity Center 사용자 이름 또는 암호를 잊어버렸기 때문에 App Studio에 로그인하는 방법을 잘 모를 수 있습니다.

해결 방법: IAM Identity Center 인스턴스 없이 App Studio를 설정할 때 IAM Identity Center 사용자를 생성하는 데 사용할 이메일과 사용자 이름이 각 사용자에게 제공되었습니다. 제공된 각 이메일 주소에 IAM Identity Center 가입 초대장이 포함된 이메일이 전송되었습니다. 각 사용자는 초대를 수락하고 IAM Identity Center 사용자 자격 증명에 대한 암호를 생성해야 합니다. 그런 다음 각 사용자는 IAM Identity Center 사용자 이름과 암호를 사용하여 App Studio에 로그인할 수 있습니다.

자격 증명을 이미 설정하고 사용자 이름이나 암호를 잊어버린 경우 관리자에게 IAM Identity Center 콘솔을 사용하여 사용자 이름을 보고 제공하거나 암호를 재설정하도록 요청해야 합니다.

App Studio를 설정할 때 시스템 오류가 발생합니다.

문제: App Studio를 설정할 때 다음 오류가 발생합니다.

```
System error. We encountered a problem. Report the issue and the App Studio service
 team will get back to you.
```

이 오류는 서비스에 알 수 없는 오류가 발생한 경우에 발생합니다.

해결 방법: 왼쪽 탐색의 학습 섹션에서 또는 앱을 편집하는 동안 상단 배너에서 Slack에 가입을 선택하여 커뮤니티 Slack에 가입하여 지원 팀에 문의하세요.

App Studio 인스턴스 URL을 찾을 수 없음

App Studio 인스턴스에 액세스할 URL을 찾을 수 없는 경우 App Studio를 설정한 관리자에게 문의하세요. 관리자는의 App Studio 콘솔에서 URL을 볼 수 있습니다 AWS Management Console.

App Studio에서 그룹 또는 역할을 수정할 수 없음

문제: 왼쪽 탐색에는 역할 링크가 표시되지 않습니다. 관리자 역할을 가진 사용자만 App Studio에서 그룹 및 역할을 수정할 수 있기 때문입니다.

해결 방법: 관리자 역할을 가진 사용자에게 문의하여 그룹 또는 역할을 변경하거나 관리자에게 문의하여 관리자 그룹에 추가합니다.

App Studio에서 오프보딩하려면 어떻게 해야 합니까?

현재 App Studio에서 오프보딩할 수 없습니다. 앱 및 커넥터와 같은 모든 리소스를 제거하고 그룹의 역할을 앱 사용자로 변경하여 액세스 또는 사용을 방지하는 것이 좋습니다. 또한 IAM 역할 또는 데이터베이스 테이블과 같이 App Studio에만 사용되는 타사 리소스도 삭제해야 합니다.

앱 문제 해결 및 디버깅

다음 주제에는 App Studio 앱의 문제 해결 및 디버깅에 대한 정보가 포함되어 있습니다.

주제

- AI 빌더 어시스턴트 및 채팅 문제 해결
- 애플리케이션 스튜디오의 문제 해결
- 앱 미리 보기 문제 해결
- 테스트 환경의 문제 해결
- Amazon CloudWatch Logs에서 게시된 앱의 로그를 사용하여 디버깅
- 커넥터 문제 해결

AI 빌더 어시스턴트 및 채팅 문제 해결

이 주제에는 AI 빌더 어시스턴트를 사용할 때 발생하는 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다.

AI로 앱을 생성할 때 오류 발생

AI 프롬프트를 사용하여 앱을 생성할 때 다음 오류가 발생할 수 있습니다.

```
We apologize, but we cannot proceed with your request. The request may contain content
 that violates our policies and guidelines. Please revise your prompt before trying
 again.
```

문제: 잠재적으로 유해한 콘텐츠로 인해 요청이 차단되었습니다.

해결 방법: 프롬프트의 문구를 바꾸고 다시 시도하세요.

AI를 사용하여 생성된 앱이 비어 있거나 구성 요소가 없습니다.

문제: 예기치 않은 서비스 오류로 인해 발생할 수 있습니다.

해결 방법: AI를 사용하여 앱 생성을 다시 시도하거나 생성된 앱에서 구성 요소를 수동으로 생성합니다.

애플리케이션 스튜디오의 문제 해결

이 주제에는 애플리케이션 빌드 시 문제에 대한 문제 해결 및 디버깅 지침이 포함되어 있습니다.

디버그 패널 사용

앱을 빌드하는 동안 라이브 디버깅을 지원하기 위해 App Studio는 애플리케이션 스튜디오의 페이지, 자동화 및 데이터 탭에 걸쳐 있는 축소 가능한 빌더 디버그 패널을 제공합니다. 이 패널에는 오류와 경고가 모두 표시됩니다. 경고는 구성되지 않은 리소스와 같은 실행 가능한 제안 역할을 하지만 앱을 성공적으로 미리 보거나 게시하려면 오류를 해결해야 합니다. 각 오류 또는 경고에는 문제의 위치를 탐색하는 데 사용할 수 있는 보기 링크가 포함되어 있습니다.

디버그 패널은 새 오류 또는 경고가 발생하면 자동으로 업데이트되고 해결되면 오류 또는 경고가 자동으로 사라집니다. 빌더에서 나가면 이러한 경고 및 오류 메시지의 상태가 유지됩니다.

JavaScript 표현식 구문 및 데이터 유형 처리

App Studio는 코드를 빨간색 선으로 강조 표시하여 오류를 강조 표시하는 JavaScript 오류 감지 기능을 제공합니다. 이러한 컴파일 오류는 앱이 성공적으로 빌드되지 못하게 하여 오타, 잘못된 참조, 잘못된 작업 및 필요한 데이터 유형에 대한 잘못된 출력과 같은 문제를 나타냅니다. 일반적인 문제는 다음 목록을 참조하세요.

1. 리소스 이름 변경으로 인한 오류: JavaScript 표현식이 App Studio에서 리소스 이름을 참조하면 해당 이름을 변경하면 표현식이 잘못되어 오류가 발생합니다. 디버그 패널에서 이러한 오류를 볼 수 있습니다.
2. 데이터 형식 문제: 데이터 형식 불일치로 인해 앱에 오류가 발생합니다. 예를 들어 자동화가 유형의 파라미터를 수락하도록 구성`String`되었지만 구성 요소가 유형의 값을 전송하도록 구성된 경우 `Integer`오류가 발생합니다. 데이터 형식이 구성 요소, 자동화, 데이터 엔터티 및 작업을 포함한 적절한 리소스 간에 일치하는지 확인합니다. JavaScript 표현식에서 값의 유형을 변경해야 할 수 있습니다.

앱 미리 보기 문제 해결

이 주제에는 앱을 미리 볼 때 발생하는 문제 해결에 대한 정보가 포함되어 있습니다.

다음 오류와 함께 미리 보기가 로드되지 않습니다. **Your app failed to build and cannot be previewed**

문제: 앱을 성공적으로 빌드해야 미리 볼 수 있습니다. 이 오류는 앱이 성공적으로 빌드되지 않는 컴파일 오류가 있을 때 발생합니다.

해결 방법: 애플리케이션 스튜디오의 디버그 패널을 사용하여 오류를 검토하고 해결합니다.

미리 보기를 로드하는 데 시간이 오래 걸립니다.

문제: 특정 유형의 앱 업데이트를 컴파일하고 빌드하는 데 시간이 오래 걸립니다.

해결 방법: 탭을 열어 두고 업데이트가 빌드될 때까지 기다립니다. 앱의 애플리케이션 스튜디오 오른쪽 상단에 저장됨이 표시되고 미리 보기가 다시 로드됩니다.

미리 보기에 최신 변경 사항이 반영되지 않음

문제: 다른 사용자가 앱 편집 세션을 인계했지만 알림을 받지 못한 경우이 문제가 발생할 수 있습니다. 이로 인해 앱이 미리 보기 환경과 일치하지 않을 수 있습니다.

해결 방법: 애플리케이션 스튜디오 브라우저 탭을 새로 고치고 필요한 경우 편집 세션을 인계합니다.

테스트 환경의 문제 해결

이 주제에는 테스트 환경에 게시된 앱 문제 해결에 대한 정보가 포함되어 있습니다.

> **Note**
>
> 자동화 또는 데이터 작업의 HTTP 500 응답은 표현식의 런타임 충돌, 커넥터 장애 또는 애플리케이션에 연결된 데이터 소스의 제한으로 인해 발생할 수 있습니다. 의 지침을 사용하여 기본 오류 세부 정보를 표시할 디버그 로그를 브라우저 콘솔을 사용하여 디버깅 봅니다.

디버그 패널 사용

앱을 빌드할 때 사용되는 빌드 디버그 패널과 마찬가지로 App Studio는 테스트 환경에서 축소 가능한 디버그 패널을 제공합니다. 이 패널에는 페이지 로드 시간, 사용자 탐색 및 앱 이벤트와 같은 정보 메시지가 표시됩니다. 여기에는 오류 및 경고도 포함됩니다. 디버그 패널은 이벤트가 발생할 때 새 메시지로 자동으로 업데이트됩니다.

브라우저 콘솔을 사용하여 디버깅

앱을 미리 보는 동안 작업이 호출되지 않으므로 호출 및 응답 처리를 테스트하려면 앱을 테스트 환경에 게시해야 합니다. 자동화 실행 중에 오류가 발생하거나 애플리케이션이 특정 방식으로 작동하는 이유를 이해하려는 경우 브라우저의 콘솔을 사용하여 실시간으로 디버깅할 수 있습니다.

브라우저 콘솔을 사용하여 테스트 환경에서 앱을 디버깅하려면

1. URL `?debug=true` 끝에를 추가하고 Enter 키를 누릅니다. URL에 이미 쿼리 문자열(포함?)이 있는 경우 대신 URL `&debug=true` 끝에를 추가합니다.
2. 브라우저 콘솔을 열어 작업 또는 API 입력 및 출력을 탐색하여 디버깅을 시작합니다.

    - Chrome에서: 브라우저에서 마우스 오른쪽 버튼을 클릭하고 검사를 선택합니다. Chrome DevTools를 사용한 디버깅에 대한 자세한 내용은 Chrome DevTools 설명서를 참조하세요.
    - Firefox: 웹 페이지 요소를 길게 누르거나 마우스 오른쪽 버튼으로 클릭한 다음 요소 검사를 선택합니다. Firefox DevTools를 사용한 디버깅에 대한 자세한 내용은 Firefox DevTools 사용자 문서를 참조하세요.

다음 목록에는 오류를 발생시키는 몇 가지 일반적인 문제가 포함되어 있습니다.

- 런타임 오류
    - 문제: 자동화 또는 표현식이 잘못 구성된 경우 자동화가 실행될 때 오류가 발생할 수 있습니다. 일반적인 오류는 자산 이름을 변경하여 잘못된 표현식, 기타 JavaScript 컴파일 오류 또는 인 데이터 또는 자산을 사용하려는 시도를 초래하는 것입니다`undefined`.
    - 해결 방법: 사용자 지정 코드 입력(식, JavaScript 및 JSON)의 각 사용량을 확인하고 코드 편집기 또는 디버그 패널에 컴파일 오류가 없는지 확인합니다.
- 커넥터 문제
    - 문제: App Studio 앱은 게시될 때까지 커넥터와 외부 서비스와 통신하지 않으므로 테스트 환경에서 미리 보기 중에 발생하지 않은 오류가 발생할 수 있습니다. 커넥터를 사용하는 자동화의 작업이 실패하면 요청을 커넥터로 보내는 작업의 잘못된 구성 또는 커넥터 구성 자체로 인한 것일 수 있습니다.
    - 해결 방법: 이러한 오류를 방지하려면 모의 출력을 사용하여 미리 보기 환경 초기에 자동화를 테스트해야 합니다. 커넥터가 올바르게 구성되었는지 확인합니다. 자세한 내용은 섹션을 참조하세요커넥터 문제 해결. 마지막으로 CloudWatch를 사용하여 로그를 검토할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs에서 게시된 앱의 로그를 사용하여 디버깅 단원을 참조하십시오.

ConnectorService 네임스페이스 로그에는 커넥터에서 시작된 오류 메시지 또는 메타데이터가 있어야 합니다.

Amazon CloudWatch Logs에서 게시된 앱의 로그를 사용하여 디버깅

Amazon CloudWatch Logs는 AWS 리소스와에서 AWS 실시간으로 실행하는 애플리케이션을 모니터링합니다. CloudWatch Logs를 사용하여 리소스 및 애플리케이션에 대해 측정할 수 있는 변수인 지표를 수집하고 추적할 수 있습니다.

App Studio 앱 디버깅의 경우 CloudWatch Logs는 앱 실행 중에 발생하는 오류를 추적하고, 정보를 감사하고, 사용자 작업 및 독점 상호 작용에 대한 컨텍스트를 제공하는 데 유용합니다. 로그는 애플리케이션 사용량 및 액세스 패턴을 감사하고 사용자가 발견한 오류를 검토하는 데 사용할 수 있는 기록 데이터를 제공합니다.

> **Note**
>
> CloudWatch Logs는 애플리케이션의 UI에서 전달된 파라미터 값의 실시간 추적을 제공하지 않습니다.

다음 절차에 따라 CloudWatch Logs의 App Studio 앱에서 로그에 액세스합니다.

1. 앱의 App Studio 애플리케이션 스튜디오에서 URL에서를 확인하여 앱 ID를 찾고 기록해 둡니다. 앱 ID는 다음과 같을 수 있습니다`802a3bd6-ed4d-424c-9f6b-405aa42a62c5`.
2. https://console.aws.amazon.com/cloudwatch/에서 CloudWatch 콘솔을 엽니다.
3. 탐색 창에서 로그 그룹을 선택합니다.
4. 여기에서 애플리케이션당 5개의 로그 그룹을 찾을 수 있습니다. 관심 있는 정보 유형에 따라 그룹을 선택하고 검색하려는 데이터에 대한 쿼리를 작성합니다.

   다음 목록에는 로그 그룹과 각 로그 그룹 사용 시기에 대한 정보가 포함되어 있습니다.

   1. `/aws/appstudio/`*`teamId`*`/`*`appId`*`/TEST/app`: 테스트 환경에 현재 게시된 앱 버전과 관련된 자동화 응답, 구성 요소 오류 또는 JavaScript 코드를 디버깅하는 데 사용합니다.
   2. `/aws/appstudio/`*`teamId`*`/`*`appId`*`/TEST/audit`: 조건부 가시성 또는 변환, 쿼리 실패, 현재 테스트 환경에 게시된 앱 버전과 관련된 로그인 또는 권한 사용자 오류와 같은 JavaScript 코드 오류를 디버깅하는 데 사용합니다.
   3. `/aws/appstudio/`*`teamId`*`/setup`:를 사용하여 빌더 또는 관리자 작업을 모니터링합니다.

4. /aws/appstudio/*teamId*/*appId*/PRODUCTION/app: 현재 프로덕션 환경에 게시된 앱 버전과 관련된 자동화 응답, 쿼리 실패, 구성 요소 오류 또는 JavaScript 코드를 디버깅하는 데 사용합니다.
5. /aws/appstudio/*teamId*/*appId*/PRODUCTION/audit: 조건부 가시성 또는 변환과 같은 JavaScript 코드 오류와 현재 프로덕션 환경에 게시된 앱 버전과 관련된 로그인 또는 권한 사용자 오류를 디버깅하는 데 사용합니다.

> **Note**
>
> 디버깅에 사용할 대부분의 로그는 DebugLogClient 네임스페이스로 분류됩니다.

5. 로그 그룹에 속하면 최신 로그 스트림을 선택하거나 마지막 이벤트 시간이 관심 시간에 가장 가까운 로그 스트림을 선택하거나 모든 로그 스트림을 검색하여 해당 로그 그룹의 모든 이벤트를 검색하도록 선택할 수 있습니다. CloudWatch Logs에서 로그 데이터를 보는 방법에 대한 자세한 내용은 CloudWatch Logs로 전송된 로그 데이터 보기를 참조하세요.

CloudWatch Logs Insights 쿼리를 사용하여 로그 필터링 및 정렬

CloudWatch Logs Insights를 사용하여 한 번에 여러 로그 그룹을 쿼리할 수 있습니다. 세션 정보가 포함된 로그 그룹 목록을 식별하면 CloudWatch Logs Insights로 이동하여 로그 그룹을 선택합니다. 그런 다음 쿼리를 사용자 지정하여 대상 로그 항목의 범위를 좁힙니다. 다음은 몇 가지 샘플 쿼리입니다.

키워드가 포함된 로그 목록: ##

```
fields @timestamp, @message
| filter @message like 'error'
| sort @timestamp desc
```

테스트 환경에서 로그를 디버깅합니다.

```
fields @timestamp, @message
| filter namespace = "DebugLogClient"
| sort @timestamp desc
```

5분 간격 동안 전체 504/404/500 오류 수:

```
filter @message like '/api/automation' and (@message like ': 404' or @message like ':
 500' or @message like ': 504')
```

```
| fields @timestamp, method, path, statusCode
| stats count(*) as errorCount by bin(5m)
```

CloudWatch Logs Insights에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 CloudWatch Logs Insights를 사용하여 로그 데이터 분석을 참조하세요. Amazon CloudWatch

커넥터 문제 해결

이 주제에는 일반적인 커넥터 문제에 대한 문제 해결 지침이 포함되어 있습니다. 커넥터를 보거나 편집하려면 관리자 그룹의 구성원이어야 합니다.

IAM 역할에 올바른 사용자 지정 신뢰 정책 및 태그가 있는지 확인

커넥터에 대한 IAM 역할을 설정하는 동안 App Studio에 대한 액세스를 제공하도록 사용자 지정 신뢰 정책이 올바르게 구성되어 있는지 확인합니다. AWS 리소스가 App Studio를 설정하는 데 사용된 것과 동일한 AWS 계정에 있는 경우에도이 사용자 지정 신뢰 정책이 필요합니다.

- `Principal` 섹션의 AWS 계정 번호가 App Studio를 설정하는 데 사용되는 AWS 계정의 계정 ID인지 확인합니다. 이 계정 번호가 항상 리소스가 위치한 계정인 것은 아닙니다.
- `sts:AssumeRole` 단원에 `"aws:PrincipalTag/IsAppStudioAccessRole": "true"`가 올바르게 추가되었는지 확인합니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::111122223333:root"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "aws:PrincipalTag/IsAppStudioAccessRole": "true"
      }
    }
  }
```

```
  ]
}
```

또한 다음 키와 값이 있는 태그가 IAM 역할에 추가되었는지 확인합니다. 태그 추가에 대한 자세한 내용은 IAM 역할 태그 지정을 참조하세요.

> **Note**
>
> 태그의 값은 사용자 지정 신뢰 정책()의 값과 `IsAppStudioDataAccessRole`약간 다른 입니다`IsAppStudioAccessRole`.

- 키: `IsAppStudioDataAccessRole`
- 값: `true`

커넥터가 연결 중인 제품 또는 서비스의 리소스 구성을 확인합니다. Amazon Redshift 테이블과 같은 일부 리소스는 App Studio와 함께 사용하기 위해 추가 구성이 필요합니다.

커넥터 구성을 확인합니다. AWS 서비스의 경우 App Studio의 커넥터로 이동하여 올바른 Amazon 리소스 이름(ARN)이 포함되어 있고 지정된 AWS 리전이 리소스가 포함된 리전인지 확인합니다.

IAM 역할에 올바른 권한이 있는지 확인

App Studio에 AWS 리소스에 대한 액세스 권한을 제공하려면 커넥터에서 사용하는 IAM 역할에 적절한 권한을 할당해야 합니다. 필요한 권한은 수행할 서비스, 리소스 및 작업에 고유합니다. 예를 들어 Amazon Redshift 테이블에서 데이터를 읽으려면 Amazon S3 버킷에 객체를 업로드하는 것과 다른 권한이 필요합니다. 자세한 내용은의 해당 주제를 참조AWS 서비스에 연결하세요.

Amazon Redshift 커넥터 문제 해결

이 섹션에는 Amazon Redshift 커넥터의 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다. Amazon Redshift 커넥터 및 리소스 구성에 대한 자세한 내용은 섹션을 참조하세요Amazon Redshift에 연결.

1. Amazon Redshift 편집기에서 `Isolated Session` 토글이 `OFF` 로 설정되어 있는지 확인합니다. 이 설정은 App Studio 앱과 같은 다른 사용자가 변경한 데이터를 볼 수 있도록 하기 위해 필요합니다.
2. Amazon Redshift 테이블에 적절한 권한이 부여되었는지 확인합니다.

3. 커넥터 구성에서 Amazon Redshift 테이블 유형과 일치하도록 적절한 컴퓨팅 유형(`Provisioned` 또는 `Serverless`)이 선택되어 있는지 확인합니다.

Aurora 커넥터 문제 해결

이 섹션에는 Aurora 커넥터의 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다. Aurora 커넥터 및 리소스 구성에 대한 자세한 내용은 섹션을 참조하세요Amazon Aurora에 연결.

1. 테이블을 생성할 때 적절하고 지원되는 Aurora 버전이 선택되어 있는지 확인합니다.
2. App Studio가 Aurora 테이블에서 작업을 수행하도록 허용하는 요구 사항이므로 Amazon RDS 데이터 API가 활성화되어 있는지 확인합니다. 자세한 내용은 Amazon RDS 데이터 API 활성화를 참조하세요.
3. AWS Secrets Manager 권한이 제공되었는지 확인합니다.

DynamoDB 커넥터 문제 해결

이 섹션에는 DynamoDB 커넥터의 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다. DynamoDB 커넥터 및 리소스 구성에 대한 자세한 내용은 섹션을 참조하세요Amazon DynamoDB에 연결.

커넥터를 생성할 때 DynamoDB 테이블 스키마가 표시되지 않는 경우 DynamoDB 테이블이 고객 관리형 키(CMK)로 암호화되고 키를 설명하고 테이블을 복호화할 권한이 없으면 테이블 데이터에 액세스할 수 없기 때문일 수 있습니다. CMK로 암호화된 테이블로 DynamoDB 커넥터를 생성하려면 IAM 역할에 `kms:decrypt` 및 `kms:describeKey` 권한을 추가해야 합니다.

Amazon S3 커넥터 문제 해결

이 섹션에는 Amazon S3 커넥터의 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다. Amazon S3 커넥터 및 리소스 구성에 대한 자세한 내용은 섹션을 참조하세요Amazon Simple Storage Service(Amazon S3)에 연결.

일반적인 문제 해결 지침에는 다음이 포함됩니다.

1. Amazon S3 커넥터가 Amazon S3 리소스가 있는 AWS 리전으로 구성되어 있는지 확인합니다.
2. IAM 역할이 올바르게 구성되었는지 확인합니다.
3. Amazon S3 버킷에서 CORS 구성이 적절한 권한을 부여하는지 확인합니다. 자세한 내용은 1단계: Amazon S3 리소스 생성 및 구성 단원을 참조하십시오.

Amazon S3 파일 업로드 오류: 미리 서명된 URL을 계산하지 못함

S3 업로드 구성 요소를 사용하여 Amazon S3 버킷에 파일을 업로드하려고 할 때 다음 오류가 발생할 수 있습니다.

```
Error while uploading file to S3: Failed to calculate presigned URL.
```

이 오류는 일반적으로 Amazon S3 버킷의 잘못된 IAM 역할 구성 또는 잘못된 CORS 구성으로 인해 발생하며의 정보로 해당 구성을 수정하여 해결할 수 있습니다Amazon Simple Storage Service(Amazon S3)에 연결.

애플리케이션 게시 및 공유 문제 해결

이 주제에는 App Studio 애플리케이션을 게시하거나 공유할 때 발생하는 일반적인 문제에 대한 문제 해결 지침이 포함되어 있습니다.

공유 대화 상자에 새로 생성된 앱 역할이 표시되지 않습니다.

새로 생성된 앱 수준 역할은 앱이 다시 게시된 후에만 공유 대화 상자에 표시됩니다. 새 역할이 생성된 후 앱을 게시하여 사용합니다.

앱 게시가 완료될 때 이메일을 받지 못했습니다.

앱이 게시되면 앱 소유자만 이메일을 받습니다.

앱의 최종 사용자가 게시된 앱에 액세스할 수 없음

최종 사용자가 게시된 앱에 액세스할 수 없고 액세스하려고 할 때 `Forbidden` 메시지가 표시되는 경우 게시된 앱이 액세스를 시도하는 사용자와 공유되지 않을 수 있습니다. 그룹의 사용자에게 액세스 권한을 부여하려면 게시된 앱을 그룹과 공유해야 합니다.

애플리케이션 공유에 대한 자세한 내용은 게시된 애플리케이션 공유 단원을 참조하십시오.

AWS App Studio의 보안

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 보안에 가장 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이를 클라우드의 보안과 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는 AWS 클라우드에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 서드 파티 감사자는 정기적으로 AWS 규정 준수 프로그램의 일환으로 보안 효과를 테스트하고 검증합니다. App Studio에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 AWS 규정 준수 프로그램 제공 범위 내 서비스를 참조하세요.
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 데이터의 민감도, 조직의 요건 및 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 App Studio를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목표에 맞게 App Studio를 구성하는 방법을 보여줍니다. 또한 App Studio 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법도 알아봅니다.

주제

- 보안 고려 사항 및 완화 조치
- AWS App Studio의 데이터 보호
- AWS App Studio 및 AWS Identity and Access Management (IAM)
- AWS App Studio에 대한 규정 준수 검증
- AWS App Studio의 복원력
- AWS App Studio의 인프라 보안
- AWS App Studio의 구성 및 취약성 분석
- 교차 서비스 혼동된 대리인 방지
- AWS App Studio에서 리전 간 데이터 전송

보안 고려 사항 및 완화 조치

보안 고려 사항

데이터 커넥터, 데이터 모델 및 게시된 애플리케이션을 처리할 때 데이터 노출, 액세스 제어 및 잠재적 취약성과 관련된 몇 가지 보안 문제가 발생합니다. 다음 목록에는 주요 보안 문제가 포함되어 있습니다.

IAM 역할의 부적절한 구성

데이터 커넥터에 대한 IAM 역할을 잘못 구성하면 무단 액세스 및 데이터 유출이 발생할 수 있습니다. 데이터 커넥터의 IAM 역할에 지나치게 허용적인 액세스 권한을 부여하면 권한이 없는 사용자가 민감한 데이터에 액세스하고 수정할 수 있습니다.

IAM 역할을 사용하여 데이터 작업 수행

App Studio 앱의 최종 사용자는 작업을 수행하기 위해 커넥터 구성에 제공된 IAM 역할을 수임하므로 해당 최종 사용자는 일반적으로 액세스할 수 없는 데이터에 액세스할 수 있습니다.

게시된 애플리케이션의 데이터 커넥터 삭제

데이터 커넥터가 삭제되면 이미 해당 커넥터를 사용하고 있는 게시된 애플리케이션에서 연결된 보안 암호 자격 증명이 자동으로 제거되지 않습니다. 이 시나리오에서 애플리케이션이 특정 커넥터와 함께 게시되고 해당 커넥터 중 하나가 App Studio에서 삭제된 경우 게시된 애플리케이션은 이전에 저장된 커넥터 자격 증명을 사용하여 계속 작동합니다. 커넥터 삭제에도 불구하고 게시된 앱은 영향을 받지 않고 작동합니다.

게시된 애플리케이션에서 데이터 커넥터 편집

데이터 커넥터를 편집하면 해당 커넥터를 사용하는 게시된 애플리케이션에 변경 사항이 자동으로 반영되지 않습니다. 애플리케이션이 특정 커넥터와 함께 게시되고 이러한 커넥터 중 하나가 App Studio에서 수정된 경우 게시된 애플리케이션은 이전에 저장된 커넥터 구성 및 자격 증명을 계속 사용합니다. 업데이트된 커넥터 변경 사항을 통합하려면 애플리케이션을 다시 게시해야 합니다. 앱이 다시 게시될 때까지는 올바르지 않고 작동하지 않거나 영향을 받지 않고 작동하지만 최신 커넥터 수정 사항은 반영되지 않습니다.

보안 위험 완화 권장 사항

이 섹션에는 이전 보안 고려 사항 섹션에 자세히 설명된 보안 위험을 방지하기 위한 완화 권장 사항이 나열되어 있습니다.

1. 적절한 IAM 역할 구성: 무단 액세스 및 데이터 유출을 방지하기 위해 데이터 커넥터에 대한 IAM 역할이 최소 권한 원칙으로 올바르게 구성되어 있는지 확인합니다.
2. 제한된 앱 액세스: 애플리케이션 데이터를 보거나 작업을 수행할 권한이 있는 사용자와만 앱을 공유합니다.
3. 앱 게시: 커넥터가 업데이트되거나 삭제될 때마다 앱이 다시 게시되는지 확인합니다.

AWS App Studio의 데이터 보호

AWS 공동 책임 모델 AWS App Studio의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 이 인프라에 호스팅되는 콘텐츠에 대한 통제 권한을 유지할 책임이 있습니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 관한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내부의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 연방 정보 처리 표준(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 AWS App Studio 또는 기타 AWS 서비스 에서 콘솔 AWS CLI, API 또는 AWS SDKs를 사용하여 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유

형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 자격 증명을 URL에 포함해서는 안 됩니다.

데이터 암호화

App Studio는 저장 데이터와 전송 중 데이터를 암호화하여 데이터를 안전하게 저장하고 전송합니다.

저장 시 암호화

유휴 데이터 암호화는 저장된 데이터를 암호화하여 무단 액세스로부터 데이터를 보호하는 것을 의미합니다. App Studio는 기본적으로 AWS KMS 키를 사용하여 저장 데이터 암호화를 제공하며 저장 데이터 암호화를 위한 추가 구성을 수행할 필요가 없습니다.

App Studio는 애플리케이션에 대해 소스 코드, 빌드 아티팩트, 메타데이터 및 권한 정보 등의 데이터를 안전하게 저장합니다.

AWS KMS 고객 관리형 키(CMK)로 암호화된 데이터 소스를 사용하는 경우 App Studio 리소스는 AWS 관리형 키를 사용하여 계속 암호화되는 반면 암호화된 데이터 소스의 데이터는 CMK에 의해 암호화됩니다. App Studio 앱에서 암호화된 데이터 소스를 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요CMKs에서 암호화된 데이터 소스 사용.

App Studio는 Amazon CloudFront를 사용하여 사용자에게 앱을 제공합니다. CloudFront는 엣지 로케이션 POP(상호 접속 위치)용으로 암호화된 SSD 및 Regional Edge Caches(REC)용으로 암호화된 EBS 볼륨을 사용합니다. CloudFront Functions의 함수 코드 및 구성은 항상 엣지 로케이션 POP의 암호화된 SSD와 CloudFront에서 사용하는 다른 스토리지 위치에 암호화된 형식으로 저장됩니다.

전송 중 암호화

전송 중 데이터 암호화는 데이터가 통신 엔드포인트 간을 이동하는 동안 데이터를 가로채기에서 보호하는 것을 의미합니다. App Studio는 기본적으로 전송 중 데이터에 대한 암호화를 제공합니다. 고객과 App Studio 간의 모든 통신과 App Studio와 다운스트림 종속성 간의 모든 통신은 서명 버전 4 서명 프로세스를 사용하여 서명된 TLS 연결을 사용하여 보호됩니다. 모든 App Studio 엔드포인트는 AWS Certificate Manager Private Certificate Authority에서 관리하는 SHA-256 인증서를 사용합니다.

키 관리

App Studio는 암호화 키 관리를 지원하지 않습니다.

인터네트워크 트래픽 개인 정보 보호

App Studio에서 인스턴스를 생성할 때 해당 인스턴스에 대한 데이터 및 리소스가 저장될 AWS 리전을 선택합니다. 애플리케이션 빌드 아티팩트와 메타데이터는 해당 AWS 리전을 벗어나지 않습니다.

그러나 다음 정보를 기록해 둡니다.

- App Studio는 Amazon CloudFront를 사용하여 애플리케이션을 제공하고 Lambda@Edge를 사용하여 애플리케이션에 대한 인증을 관리하기 때문에 제한된 인증 데이터 세트, 권한 부여 데이터, 애플리케이션 메타데이터는 다른 리전에 있을 수 있는 CloudFront 엣지 로케이션에서 액세스할 수 있습니다.
- AWS App Studio는 AWS 리전 간에 데이터를 전송하여 서비스에서 특정 생성형 AI 기능을 활성화합니다. 리전 간 데이터 전송이 지원하는 기능, 리전 간에 이동하는 데이터 유형, 옵트아웃 방법에 대한 자세한 내용은 섹션을 참조하세요AWS App Studio에서 리전 간 데이터 전송.

AWS App Studio 및 AWS Identity and Access Management (IAM)

AWS App Studio에서는 IAM Identity Center의 그룹을 App Studio의 적절한 역할에 할당하여 서비스의 액세스 및 권한을 관리합니다. 그룹 멤버의 권한은 (AWS Identity and Access Management IAM)에서 사용자, 역할 또는 권한을 직접 구성하는 것이 아니라 할당된 역할에 따라 결정됩니다. App Studio의 액세스 및 권한 관리에 대한 자세한 내용은 섹션을 참조하세요App Studio에서 액세스 및 역할 관리.

App Studio는 결제 목적으로 인스턴스를 확인할 때, 그리고 계정에 연결되어 해당 AWS 계정에서 리소스를 생성하고 사용할 때 IAM과 통합 AWS 됩니다. 애플리케이션에서 사용할 다른 AWS 서비스에 App Studio를 연결하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS 서비스에 연결.

App Studio에서 인스턴스를 생성할 때 AWS 계정을 인스턴스의 결제 및 관리 계정으로 연결해야 합니다. 주요 기능을 활성화하기 위해 App Studio는 사용자를 대신하여 작업을 수행하는 데 필요한 권한을 서비스에 제공하는 IAM 서비스 역할도 생성합니다.

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 AWS 서비스 되는 입니다. IAM 관리자는 App Studio 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는 사용자를 제어합니다. IAM은 추가 비용 없이 사용할 수 AWS 서비스 있는 입니다.

주제

- App Studio의 자격 증명 기반 정책

- App Studio 내 리소스 기반 정책
- App Studio에 대한 정책 작업
- App Studio에 대한 정책 리소스
- App Studio의 정책 조건 키
- App StudioACLs
- App Studio를 사용한 ABAC
- App Studio에서 임시 자격 증명 사용
- App Studio에 대한 교차 서비스 보안 주체 권한
- App Studio의 서비스 역할
- App Studio의 서비스 연결 역할
- AWS AWS App Studio에 대한 관리형 정책
- App Studio의 서비스 연결 역할
- AWS App Studio의 자격 증명 기반 정책 예제

IAM을 사용하여 App Studio에 대한 액세스를 관리하기 전에 App Studio에서 사용할 수 있는 IAM 기능을 알아봅니다.

AWS App Studio와 함께 사용할 수 있는 IAM 기능

IAM 특성	App Studio 지원
자격 증명 기반 정책	예
리소스 기반 정책	아니요
정책 작업	예
정책 리소스	예
정책 조건 키	아니요
ACL	아니요
ABAC(정책 내 태그)	아니요
임시 보안 인증	예

IAM 특성	App Studio 지원
엔터티 권한	예
서비스 역할	예
서비스 연결 역할	예

App Studio 및 기타 AWS 서비스가 대부분의 IAM 기능과 작동하는 방식을 전체적으로 알아보려면 IAM 사용 설명서의 AWS IAM으로 작업하는 서비스를 참조하세요.

App Studio의 자격 증명 기반 정책

ID 기반 정책 지원: 예

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 ID에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자 및 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. JSON 정책에서 사용할 수 있는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

App Studio의 자격 증명 기반 정책 예제

App Studio 자격 증명 기반 정책의 예를 보려면 섹션을 참조하세요AWS App Studio의 자격 증명 기반 정책 예제.

App Studio 내 리소스 기반 정책

리소스 기반 정책 지원: 아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예제는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 페더레이션 사용자 또는가 포함될 수 있습니다 AWS 서비스.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM에서 교차 계정 리소스 액세스를 참조하세요.

App Studio에 대한 정책 작업

정책 작업 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

App Studio 작업 목록을 보려면 서비스 승인 참조의 AWS App Studio에서 정의한 작업을 참조하세요.

App Studio의 정책 작업은 작업 앞에 다음 접두사를 사용합니다.

```
appstudio
```

단일 문에서 여러 작업을 지정하려면 쉼표로 구분합니다.

```
"Action": [
      "appstudio:action1",
      "appstudio:action2"
         ]
```

다음 문은 App Studio의 모든 작업을 나열합니다.

App Studio에 대한 정책 리소스

정책 리소스 지원: 예

App Studio 권한은 정책의 `Resource` 요소에 와일드카드(*)만 지원합니다.

App Studio의 정책 조건 키

서비스별 정책 조건 키 지원: 아니요

App Studio는 정책 조건 키를 지원하지 않습니다.

App StudioACLs

ACL 지원: 아니요

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

App Studio를 사용한 ABAC

ABAC 지원(정책의 태그): 아니요

App Studio는 속성 기반 액세스 제어(ABAC)를 지원하지 않습니다.

App Studio에서 임시 자격 증명 사용

임시 자격 증명 지원: 예

임시 자격 증명은 AWS 리소스에 대한 단기 액세스를 제공하며 페더레이션 또는 전환 역할을 사용할 때 자동으로 생성됩니다. 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성하는 것이 AWS 좋습니다. 자세한 내용은 IAM 사용 설명서의 IAM의 임시 보안 자격 증명 및 IAM으로 작업하는 AWS 서비스 섹션을 참조하세요.

App Studio에 대한 교차 서비스 보안 주체 권한

전달 액세스 세션(FAS) 지원: 예

전달 액세스 세션(FAS)은를 호출하는 보안 주체의 권한을 다운스트림 서비스에 AWS 서비스 대한 요청과 AWS 서비스함께 사용합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

App Studio의 서비스 역할

서비스 역할 지원: 예

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 AWS 서비스 AWS에 권한을 위임할 역할 생성을 참조하세요.

AWS App Studio는 일부 기능에 IAM 서비스 역할을 사용하여 사용자를 대신하여 작업을 수행할 수 있는 권한을 App Studio에 부여합니다. 콘솔은 App Studio를 설정할 때 지원되는 기능에 대한 서비스 역할을 자동으로 생성합니다.

> **Warning**
>
> 서비스 역할에 대한 권한을 변경하면 App Studio 기능이 중단될 수 있습니다. App Studio가 관련 지침을 제공하는 경우에만 서비스 역할을 편집합니다.

App Studio의 서비스 연결 역할

서비스 연결 역할 지원: 예

서비스 연결 역할은에 연결된 서비스 역할의 한 유형입니다 AWS 서비스. 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은에 나타나 AWS 계정 며 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 IAM으로 작업하는AWS 서비스를 참조하세요. 서비스 연결 역할 열에서 Yes가 포함된 서비스를 테이블에서 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예(Yes) 링크를 선택합니다.

AWS AWS App Studio에 대한 관리형 정책

사용자, 그룹 및 역할에 권한을 추가하려면 직접 정책을 작성하는 것보다 AWS 관리형 정책을 사용하는 것이 더 쉽습니다. 팀에 필요한 권한만 제공하는 IAM 고객 관리형 정책을 생성하기 위해서는 시간과 전문 지식이 필요합니다. 빠르게 시작하려면 AWS 관리형 정책을 사용할 수 있습니다. 이 정책은 일반적인 사용 사례를 다루며 사용자의 AWS 계정에서 사용할 수 있습니다. AWS 관리형 정책에 대한 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책을 참조하세요.

AWS 서비스는 AWS 관리형 정책을 유지 관리하고 업데이트합니다. AWS 관리형 정책에서는 권한을 변경할 수 없습니다. 서비스에서 때때로 추가 권한을 AWS 관리형 정책에 추가하여 새로운 기능을 지원합니다. 이 유형의 업데이트는 정책이 연결된 모든 ID(사용자, 그룹 및 역할)에 적용됩니다. 서비스는 새로운 기능이 시작되거나 새 작업을 사용할 수 있을 때 AWS 관리형 정책에 업데이트됩니다. 서비스는 AWS 관리형 정책에서 권한을 제거하지 않으므로 정책 업데이트로 인해 기존 권한이 손상되지 않습니다.

또한는 여러 서비스에 걸쳐 있는 직무에 대한 관리형 정책을 AWS 지원합니다. 예를 들어 ReadOnlyAccess AWS 관리형 정책은 모든 AWS 서비스 및 리소스에 대한 읽기 전용 액세스를 제공합니다. 서비스가 새 기능을 시작하면는 새 작업 및 리소스에 대한 읽기 전용 권한을 AWS 추가합니다. 직무 정책의 목록과 설명은 IAM 사용 설명서의 직무에 관한AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AppStudioServiceRolePolicy

`AppStudioServiceRolePolicy`를 IAM 엔티티에 연결할 수 없습니다. 이 정책은 App Studio가 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 App Studio의 서비스 연결 역할 단원을 참조하십시오.

이 정책은 서비스 연결 역할이 AWS 리소스를 관리할 수 있는 권한을 부여합니다.

권한 세부 정보

이 정책에는 다음을 할 수 있는 권한이 포함되어 있습니다.

- `logs` - CloudWatch 로그 그룹 및 로그 스트림을 생성합니다. 또한 해당 로그 그룹 및 스트림에서 로그 이벤트를 생성할 수 있는 권한을 부여합니다.
- `secretsmanager` - App Studio에서 관리하는 관리형 보안 암호를 생성, 읽기, 업데이트 및 삭제합니다.
- `sso` - 애플리케이션 인스턴스를 검색합니다.
- `sso-directory` - 사용자에 대한 정보를 검색하고 그룹의 멤버 목록을 검색합니다.

JSON

```
{
 "Version":"2012-10-17",
 "Statement": [
     {
         "Sid": "AppStudioResourcePermissionsForCloudWatch",
         "Effect": "Allow",
         "Action": [
             "logs:CreateLogGroup",
             "logs:CreateLogStream",
             "logs:PutLogEvents"
         ],
         "Resource": [
             "arn:aws:logs:*:*:log-group:/aws/appstudio/*"
         ],
         "Condition": {
             "StringEquals": {
                 "aws:ResourceAccount": "${aws:PrincipalAccount}"
```

```
            }
        }
    },
    {
        "Sid": "AppStudioResourcePermissionsForSecretsManager",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret",
            "secretsmanager:DeleteSecret",
            "secretsmanager:DescribeSecret",
            "secretsmanager:GetSecretValue",
            "secretsmanager:PutSecretValue",
            "secretsmanager:UpdateSecret",
            "secretsmanager:TagResource"
        ],
        "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio-*",
        "Condition": {
            "ForAllValues:StringEquals": {
                "aws:TagKeys": [
                    "IsAppStudioSecret"
                ]
            },
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}",
                "aws:ResourceTag/IsAppStudioSecret": "true"
            }
        }
    },
    {
        "Sid": "AppStudioResourcePermissionsForManagedSecrets",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:DeleteSecret",
            "secretsmanager:DescribeSecret",
            "secretsmanager:GetSecretValue",
            "secretsmanager:PutSecretValue",
            "secretsmanager:UpdateSecret"
        ],
        "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio!*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}",
                "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
"appstudio"
```

```
            }
        }
    },
    {
        "Sid": "AppStudioResourceWritePermissionsForManagedSecrets",
        "Effect": "Allow",
        "Action": [
            "secretsmanager:CreateSecret"
        ],
        "Resource": "arn:aws:secretsmanager:*:*:secret:appstudio!*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}"
            }
        }
    },
    {
        "Sid": "AppStudioResourcePermissionsForSSO",
        "Effect": "Allow",
        "Action": [
            "sso:GetManagedApplicationInstance",
            "sso-directory:DescribeUsers",
            "sso-directory:ListMembersInGroup"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:ResourceAccount": "${aws:PrincipalAccount}"
            }
        }
    }
 ]
}
```

AWS 관리형 정책에 대한 App Studio 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후부터 App Studio의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다.

변경	설명	Date
AppStudioServiceRolePolicy – 기존 정책에 대한 업데이트	App Studio는에서 App Studio 관리형 보안 암호를 관리할 수 있는 새 권한을 추가했습니다 AWS Secrets Manager.	2025년 3월 14일
App Studio에서 변경 사항 추적 시작	App Studio가 AWS 관리형 정책에 대한 변경 사항 추적을 시작했습니다.	2024년 6월 28일

App Studio의 서비스 연결 역할

App Studio는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 App Studio에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 App Studio에서 사전 정의하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

필요한 권한을 수동으로 추가할 필요가 없으므로 서비스 연결 역할을 사용하면 App Studio를 더 쉽게 설정할 수 있습니다. App Studio는 서비스 연결 역할의 권한을 정의하며, 달리 정의되지 않은 한 App Studio만 해당 역할을 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔터티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제한 후에만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 실수로 제거할 수 없기 때문에 App Studio 리소스가 보호됩니다.

내용

- App Studio에 대한 서비스 연결 역할 권한
- App Studio에 대한 서비스 연결 역할 생성
- App Studio에 대한 서비스 연결 역할 편집
- App Studio에 대한 서비스 연결 역할 삭제

App Studio에 대한 서비스 연결 역할 권한

App Studio는 라는 서비스 연결 역할을 사용합니다`AWSServiceRoleForAppStudio`. App Studio가 서비스를 지속적으로 관리하고 애플리케이션 구축 환경을 유지하는 데 필요한 AWS 서비스 연결 역할입니다.

`AWSServiceRoleForAppStudio` 서비스 연결 역할은 `appstudio-service.amazonaws.com` 서비스만 신뢰하는 다음 신뢰 정책을 사용합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": "appstudio-service.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

권한의 경우 `AWSServiceRoleForAppStudio` 서비스 연결 역할은 다음 서비스에 대한 권한을 제공합니다.

- Amazon CloudWatch: App Studio 사용에 대한 로그 및 지표를 전송합니다.
- AWS Secrets Manager: App Studio에서 커넥터의 자격 증명을 관리하기 위해 앱을 다른 서비스에 연결하는 데 사용됩니다.
- IAM Identity Center: 사용자 액세스를 관리하기 위한 읽기 전용 액세스입니다.

특히에 부여된 권한`AWSServiceRoleForAppStudio`은 연결된 `AppStudioServiceRolePolicy` 관리형 정책에 의해 정의됩니다. 포함된 권한을 포함하여 관리형 정책에 대한 자세한 내용은 섹션을 참조하세요AWS 관리형 정책: AppStudioServiceRolePolicy.

App Studio에 대한 서비스 연결 역할 생성

서비스 연결 역할은 수동으로 생성할 필요가 없습니다. App Studio 인스턴스를 생성하면 App Studio가 서비스 연결 역할을 생성합니다.

이 서비스 연결 역할을 삭제하는 경우 App Studio 인스턴스를 생성하여 다른 인스턴스를 자동으로 생성하는 것이 좋습니다.

필수는 아니지만 앞서 설명한 신뢰 정책 코드 조각과 같이 IAM 콘솔 또는 AWS CLI 를 사용하여 서비스 이름으로 서비스 연결 역할을 생성하여 `appstudio-service.amazonaws.com` 서비스 연결 역할을 생성할 수도 있습니다. 자세한 내용은 IAM 사용자 설명서의 서비스 연결 역할 생성 섹션을 참조하세요.

App Studio에 대한 서비스 연결 역할 편집

App Studio에서는 `AWSServiceRoleForAppStudio` 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 엔터티가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 그러나 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

App Studio에 대한 서비스 연결 역할 삭제

`AWSServiceRoleForAppStudio` 역할을 삭제할 필요가 없습니다. App Studio 인스턴스를 삭제하면 App Studio가 리소스를 정리하고 서비스 연결 역할을 자동으로 삭제합니다.

권장되지는 않지만 IAM 콘솔 또는를 사용하여 서비스 연결 역할을 AWS CLI 삭제할 수 있습니다. 이렇게 하려면 먼저 서비스 연결 역할의 리소스를 정리한 다음 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려고 할 때 App Studio에서 역할을 사용하는 경우 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면 다음을 수행하세요.

1. App Studio 인스턴스에서 애플리케이션 및 커넥터를 삭제합니다.
2. IAM 콘솔, IAM CLI 또는 IAM API를 사용하여 `AWSServiceRoleForAppStudio` 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

AWS App Studio의 자격 증명 기반 정책 예제

기본적으로 사용자 및 역할에는 App Studio 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성(콘솔)을 참조하세요.

각 리소스 유형에 대한 ARNs 형식을 포함하여 App Studio에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 서비스 승인 참조의 AWS App Studio에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

주제

- 정책 모범 사례
- App Studio 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용
- 예제 1: 사용자가 App Studio 인스턴스를 설정하도록 허용
- 예제 2: 사용자가 App Studio 인스턴스를 설정하지 못하도록 거부

정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 App Studio 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정

책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특정를 통해 사용되는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.

- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.
- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정켭니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

App Studio 콘솔 사용

AWS App Studio 콘솔에 액세스하려면 최소 권한 집합이 있어야 합니다. 이러한 권한은에서 App Studio 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다 AWS 계정. 최소 필수 권한보다 더 제한적인 ID 기반 정책을 생성하는 경우, 콘솔이 해당 정책에 연결된 엔티티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게는 최소 콘솔 권한을 허용할 필요가 없습니다. 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

사용자와 역할이 App Studio 콘솔을 계속 사용할 수 있도록 하려면 App Studio *ConsoleAccess* 또는 *ReadOnly* AWS 관리형 정책도 엔터티에 연결합니다. 자세한 내용은 IAM 사용 설명서의 사용자에게 권한 추가를 참조하세요.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

예제 1: 사용자가 App Studio 인스턴스를 설정하도록 허용

다음 예제에서는 역할이 App Studio 인스턴스를 설정하도록 허용하는 자격 증명 기반 정책을 보여줍니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "appstudio:GetAccountStatus",
```

```
            "appstudio:GetEnablementJobStatus",
            "appstudio:StartEnablementJob",
            "appstudio:StartRollbackEnablementJob",
            "appstudio:StartTeamDeployment"
        ],
        "Resource": "*"
    }]
}
```

예제 2: 사용자가 App Studio 인스턴스를 설정하지 못하도록 거부

다음 예제에서는 역할이 App Studio 인스턴스를 설정하지 못하도록 거부하는 자격 증명 기반 정책을 보여줍니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Deny",
        "Action": [
            "appstudio:*"
        ],
        "Resource": "*"
    }]
}
```

AWS App Studio에 대한 규정 준수 검증

AWS 서비스 가 특정 규정 준수 프로그램의 범위 내에 있는지 알아보려면 AWS 서비스 규정 준수 프로그램 범위 내를 참조하고 관심 있는 규정 준수 프로그램을 선택합니다. 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 에서 보고서 다운로드 AWS Artifact에서 .

사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 AWS 서비스 결정됩니다. 사용 시 규정 준수 책임에 대한 자세한 내용은 AWS 보안 설명서를 AWS 서비스참조하세요.

AWS App Studio의 복원력

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다.는 물리적으로 분리되고 격리된 여러 가용 영역을 AWS 리전 제공하며,이 가용 영역은 지연 시간이 짧고 처리량이 높으며 중복성이 높은 네트워킹과 연결됩니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

AWS 글로벌 인프라 외에도 AWS App Studio는 데이터 복원력 및 백업 요구 사항을 지원하는 데 도움이 되는 여러 기능을 제공합니다.

AWS App Studio의 인프라 보안

관리형 서비스인 AWS App Studio는 Amazon Web Services: 보안 프로세스 개요 백서에 설명된 AWS 글로벌 네트워크 보안 절차로 보호됩니다.

AWS 에서 게시한 API 호출을 사용하여 네트워크를 통해 App Studio에 액세스합니다. 클라이언트는 최소 TLS(전송 계층 보안) 1.2를 지원해야 하지만 TLS 1.3이 권장됩니다. 클라이언트는 DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군도 지원해야 합니다. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

또한 요청은 액세스 키 ID 및 IAM 위탁자와 관련된 시크릿 액세스 키를 사용하여 서명해야 합니다. 또는 AWS Security Token Service(AWS STS)를 사용하여 임시 보안 자격 증명을 생성하여 요청에 서명할 수 있습니다.

AWS App Studio의 구성 및 취약성 분석

구성 및 IT 제어는 AWS 와 고객 간의 공동 책임입니다. 자세한 내용은 AWS 공동 책임 모델을 참조하세요.

교차 서비스 혼동된 대리인 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에게 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS교차 서비스 가장은 혼동된 대리자 문제를 초래할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)

를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 위탁자를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

리소스 정책에서 aws:SourceArn 및 aws:SourceAccount 전역 조건 컨텍스트 키를 사용하여 리소스에 다른 서비스를 제공하는 권한을 제한하는 것이 좋습니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 aws:SourceArn을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 aws:SourceAccount을(를) 사용합니다.

혼동된 대리자 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 aws:SourceArn 전역 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 arn:aws:*servicename*:*:*123456789012*:*입니다.

만약 aws:SourceArn 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우, 권한을 제한하려면 두 글로벌 조건 컨텍스트 키를 모두 사용해야 합니다.

aws:SourceArn의 값은 ResourceDescription이어야 합니다.

다음 예는 aws:SourceArn 및 aws:SourceAccount 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

AWS App Studio에서 리전 간 데이터 전송

AWS App Studio는 AWS 리전 간에 데이터를 전송하여 서비스에서 특정 생성형 AI 기능을 활성화합니다. 이 주제에는 리전 간 데이터 전송에서 활성화된 기능, 리전 간에 이동하는 데이터 유형, 옵트아웃 방법에 대한 정보가 포함되어 있습니다.

다음 기능은 리전 간 데이터 전송을 통해 활성화되며 옵트아웃하면 인스턴스에서 액세스할 수 없습니다.

1. 자연어로 앱을 설명하고 리소스를 생성하여 앱 빌드를 시작하는 데 사용되는 AI를 사용하여 앱을 생성합니다.
2. 애플리케이션 스튜디오의 AI 채팅으로, 앱 구축, 게시 및 공유에 대한 질문을 하는 데 사용됩니다.

다음 데이터는 리전 간에 전송됩니다.

1. 앞서 설명한 기능의 프롬프트 또는 사용자 입력입니다.

리전 간 데이터 전송 및 이를 통해 활성화된 기능을 옵트아웃하려면 다음 절차에 따라 콘솔에서 옵트아웃 요청 양식을 작성합니다.

1. https://console.aws.amazon.com/appstudio/ App Studio 콘솔을 엽니다.
2. 데이터 전송 옵트아웃을 선택합니다.
3. AWS 계정 ID를 입력하고 이메일 주소를 입력합니다.
4. 제출을 선택합니다.
5. 제출되면 리전 간 데이터 전송 옵트아웃 요청이 처리되며 최대 60일이 걸릴 수 있습니다.

AWS App Studio에서 지원되는 브라우저

이 주제에는 게시된 애플리케이션에 액세스하는 최종 사용자와 애플리케이션 빌더 모두에 대한 브라우저 지원을 포함하여 AWS App Studio에서 지원 및 권장되는 브라우저에 대한 정보가 포함되어 있습니다.

애플리케이션 구축을 위한 지원 및 권장 브라우저

최적의 애플리케이션 구축 경험을 위해 App Studio는 Google Chrome 사용을 지원하고 적극 권장합니다.

> **Note**
>
> 권장되지는 않지만 Mozilla Firefox, Microsoft Edge 또는 MacOS용 Apple Safari와 같은 널리 사용되는 다른 웹 브라우저를 사용하여 애플리케이션을 빌드할 수도 있지만, 이러한 브라우저는 공식적으로 지원되거나 검증되지 않으며 일부 빌더 기능에 액세스하려면 설정을 업데이트해야 할 수 있습니다. 자세한 내용은 브라우저 설정을 업데이트하여 App Studio에서 앱 빌드 단원을 참조하십시오.

App Studio는 모바일 플랫폼에서 애플리케이션 빌드를 지원하지 않습니다.

애플리케이션 최종 사용자를 위한 지원 및 권장 브라우저

게시된 애플리케이션에 액세스하는 최종 사용자의 경우 App Studio는 Google Chrome 또는 Mozilla Firefox 사용을 적극 권장합니다. 권장 브라우저이지만 최종 사용자는 Microsoft Edge 또는 MacOS용 Apple Safari와 같은 다른 인기 웹 브라우저를 사용하여 게시된 앱에 액세스할 수도 있습니다.

최종 사용자는 모바일 플랫폼에서 게시된 애플리케이션에 액세스할 수도 있습니다.

브라우저 설정을 업데이트하여 App Studio에서 앱 빌드

App Studio는 Google Chrome을 사용하여 애플리케이션을 빌드하는 것을 공식적으로 지원하고 권장합니다. 그러나 다른 브라우저를 사용하여 애플리케이션을 빌드하려는 경우 App Studio의 특정 페이지에 액세스하려면 교차 사이트 추적과 관련된 특정 설정 또는 쿠키를 업데이트해야 할 수 있습니다.

Mozilla Firefox: 애플리케이션을 미리 보려면 설정을 `Firefox Settings > Privacy & Security > Enhanced Tracking Protection`로 업데이트합니다`Custom > Cookies > Cross-site tracking cookies`.

MacOS용 Apple Safari: 애플리케이션을 빌드하거나 미리 보려면 설정을 비활성화합니다`Settings > Privacy > Prevent cross-site tracking`.

AWS App Studio 할당량

다음 표에서는 AWS App Studio의 할당량 및 제한에 대해 설명합니다.

App Studio 인스턴스의 최대 앱 수	20
App Studio 인스턴스의 테스트 또는 프로덕션 환경에 게시된 최대 애플리케이션 수입니다. 테스트 및 프로덕션 모두에 게시된 단일 애플리케이션은 2개의 게시된 애플리케이션으로 계산됩니다.	6
앱당 최대 관리형 엔터티 수	20
쿼리당 반환되는 최대 행 수	3000
개체당 샘플 데이터의 최대 행 수	500
자동화의 최대 실행 시간	2분. 2분 이상 실행되는 자동화는 실패합니다.
최대 자동화 입력 및 출력 크기	입력 또는 출력당 5GB.
자동화 또는 데이터 작업에서 사용하는 최대 데이터 크기	자동화 또는 데이터 작업 실행당 450MB.
페이지 이름 및 구성 요소 이름	비어 있지 않고 고유해야 합니다. 문자, 숫자 밑줄(_) 및 달러 기호($)만 포함해야 합니다. 공백을 포함할 수 없습니다.

AWS App Studio 사용 설명서의 문서 기록

다음 표에서는 AWS App Studio의 설명서 릴리스를 설명합니다.

변경 사항	설명	날짜
새 주제: 앱의 프로덕션 환경 버전 롤백	문제가 감지되면 게시된 앱 버전을 이전에 게시된 버전으로 롤백하는 방법에 대한 정보가 추가되었습니다. 자세한 내용은 이전에 게시된 버전으로 롤백을 참조하세요.	2025년 4월 10일
새 주제: 앱, 페이지 및 구성 요소 복제	App Studio에서 앱, 페이지 및 구성 요소를 복제하는 방법에 대한 정보가 포함된 새 주제가 추가되었습니다. 자세한 내용은 애플리케이션 복제, 페이지 복제 및 구성 요소 복제를 참조하세요.	2025년 4월 7일
새 주제: App Studio 인스턴스 간에 애플리케이션 가져오기 및 내보내기	앱 빌드 개념을 학습하는 데 사용할 수 있는 App Studio에서 제공하는 가져오기 가능한 애플리케이션 목록을 포함하여 App Studio 인스턴스 간 애플리케이션 가져오기 및 내보내기에 대한 정보가 포함된 새 주제가 추가되었습니다. 자세한 내용은 애플리케이션 가져오기 및 애플리케이션 내보내기를 참조하세요.	2025년 3월 30일
업데이트된 주제: AWS 관리형 정책: AppStudioServiceRolePolicy	`AppStudioServiceRolePolicy` 권한을 업데이트하고 각 서비스에 대한 정책 설명 정보를 추가했습니다. 자세	2025년 3월 14일

	한 내용은 AWS 관리형 정책: AppStudioServiceRolePolicy를 참조하세요.	
업데이트된 주제: 데이터 작업 편집 또는 구성	조건과 일치하는 데이터베이스 테이블에서 데이터 하위 집합을 검색하는 데 사용되는 데이터 작업 조건에 사용되는 기존 및 새 연산자에 대한 정보가 추가되었습니다. 자세한 내용은 데이터 작업 편집 또는 구성을 참조하세요.	2025년 2월 27일
업데이트된 주제: JavaScript를 사용하여 표현식 작성	애플리케이션에서 표현식을 사용하여 UI 구성 요소 및 테이블 데이터를 참조하거나 업데이트하는 방법에 대한 정보를 재구성하고 추가했습니다. 자세한 내용은 JavaScript를 사용하여 App Studio에서 표현식 작성을 참조하세요.	2025년 2월 18일
업데이트된 주제: 앱 콘텐츠 보안 설정	콘텐츠 소스 앱 콘텐츠 보안 설정에 대한 정보가 추가되었습니다. 이 설정을 사용하여 앱이 Amazon S3에 객체를 업로드할 수 있는 도메인을 제한할 수 있습니다. 자세한 내용은 앱의 콘텐츠 보안 설정 보기 또는 업데이트를 참조하세요.	2025년 2월 14일
새 주제: App Studio 앱에서 Lambda 함수 호출	App Studio 앱에서 Lambda 함수를 호출하는 방법을 자세히 설명하는 짧은 자습서를 추가했습니다. 자세한 내용은 Lambda 함수 호출을 참조하세요.	2025년 1월 24일

새 주제: Amazon SES에 연결	App Studio 앱에서 서비스를 사용하기 위해 Amazon SES 커넥터를 생성하는 지침을 추가했습니다. 자세한 내용은 Amazon Simple Email Service에 연결을 참조하세요.	2025년 1월 16일
업데이트된 주제: App Studio 인스턴스 최초 생성 및 설정	보다 빠르게 시작할 수 있도록 App Studio 인스턴스를 생성하는 간편한 생성 방법을 사용하는 방법에 대한 지침이 추가되었습니다. 자세한 내용은 App Studio 인스턴스 최초 생성 및 설정을 참조하세요.	2024년 12월 13일
새로운 주제: 데이터 종속성 및 타이밍 문제 관리 모범 사례	App Studio 앱에서 데이터 종속성 및 타이밍 문제를 정상적으로 관리하는 방법에 대한 설명서가 추가되었습니다. 자세한 내용은 데이터 종속성 및 타이밍 고려 사항을 참조하세요.	2024년 11월 20일
업데이트된 주제: AI를 사용하여 앱 편집	애플리케이션 스튜디오에서 AI 채팅을 사용하여 앱을 편집하는 방법에 대한 정보가 포함된 설명서가 추가되었습니다. 자세한 내용은 생성형 AI로 App Studio 앱 빌드를 참조하세요.	2024년 11월 18일
업데이트된 주제: AI를 사용하여 JavaScript 생성	AI를 사용하여 JavaScript를 생성하는 방법에 대한 정보를 포함하도록 JavaScript 자동화 작업 참조를 업데이트했습니다. 자세한 내용은 JavaScript 자동화 작업을 참조하세요.	2024년 11월 18일

업데이트된 주제: Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 구축	새로 릴리스된 GenAI 프롬프트 작업을 사용하도록 Amazon Bedrock 프롬프트 자습서를 업데이트했습니다. 자세한 내용은 Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 빌드를 참조하세요.	2024년 11월 18일
새 주제: 앱 테마로 앱 색상 변경	앱 테마를 사용하여 앱의 색상을 변경하는 방법에 대한 정보가 포함된 주제를 추가했습니다. 자세한 내용은 앱 테마로 앱의 색상 변경을 참조하세요.	2024년 11월 18일
새로운 주제: 데이터 모델 모범 사례	App Studio 앱에서 사용할 안전하고 강력하며 확장 가능한 데이터 모델을 생성하는 모범 사례가 포함된 주제가 추가되었습니다. 자세한 내용은 데이터 모델 설계 시 모범 사례를 참조하세요.	2024년 11월 15일
업데이트된 주제: AWS 서비스에 연결	AWS 서비스에 대한 커넥터를 생성하는 데 사용되는 IAM 역할에 필요한를 포함하도록 신뢰 정책을 업데이트`sts:ExternalId` 했습니다. 자세한 내용은 AWS 서비스에 연결을 참조하세요.	2024년 11월 13일
새 주제: 이전에 게시된 앱 버전으로 롤백 또는 되돌리기	애플리케이션을 이전에 게시된 버전으로 롤백하거나 되돌리는 방법에 대한 정보가 포함된 주제를 추가했습니다. 자세한 내용은 이전에 게시된 버전으로 롤백을 참조하세요.	2024년 11월 13일

새 주제: App Studio 인스턴스 삭제	App Studio 인스턴스 삭제 방법에 대한 지침을 포함하여 App Studio 인스턴스 삭제에 대한 정보가 포함된 주제를 추가했습니다. 자세한 내용은 App Studio 인스턴스 삭제를 참조하세요.	2024년 11월 12일
새 주제: 앱 콘텐츠 보안 설정 업데이트	업데이트 방법을 포함하여 App Studio의 앱 콘텐츠 보안 설정에 대한 정보가 포함된 주제를 추가했습니다. 자세한 내용은 앱의 콘텐츠 보안 설정 보기 또는 업데이트를 참조하세요.	2024년 11월 8일
업데이트된 주제: AWS App Studio의 보안	데이터 보호 및 App Studio와 IAM의 상호 작용 방식에 대한 정보를 포함하여 보안 설명서를 확장했습니다. 자세한 내용은 AWS App Studio의 보안을 참조하세요.	2024년 11월 6일
업데이트된 주제: AWS App Studio의 할당량	잘못된 값을 수정하고 일부 할당량을 제거하도록 App Studio 서비스 할당량 및 제한 설명서를 업데이트했습니다. 자세한 내용은 AWS App Studio의 할당량을 참조하세요.	2024년 10월 21일
업데이트된 주제: App Studio를 다른 AWS 서비스에 연결	App Studio에 AWS 서비스 또는 리소스에 필요한 최소 권한을 부여하는 지침과 예제를 제공하여 모범 보안 사례를 더 잘 준수하도록 서비스에 연결하기 위한 설명서를 업데이트했습니다. 자세한 내용은 AWS 서비스에 연결을 참조하세요.	2024년 10월 18일

업데이트된 주제: Aurora 커넥터 설명서에 버전 지원 추가	Aurora 커넥터 설명서에 지원되는 버전 목록을 추가했습니다. 자세한 내용은 Amazon Aurora에 연결을 참조하세요.	2024년 10월 16일
새 주제: App Studio에서 지원되는 브라우저	App Studio 사용에 대한 브라우저 지원 및 권장 사항이 포함된 주제가 추가되었습니다. 자세한 내용은 지원되는 브라우저를 참조하세요.	2024년 10월 10일
새 주제: AWS App Studio 작동 방식	다이어그램 및 스크린샷을 포함하여 App Studio에서 앱 개발의 주요 개념을 안내하는 주제를 추가했습니다. 자세한 내용은 App Studio 작동 방식을 참조하세요.	2024년 10월 10일
새 주제: 페이지 정렬 및 구성	미리 보기 또는 게시된 앱의 탐색에서 페이지 재정렬 및 숨기기 또는 표시에 대한 정보가 포함된 주제를 추가했습니다. 자세한 내용은 페이지 순서 지정 및 구성을 참조하세요.	2024년 9월 24일
새 주제: AWS App Studio의 할당량	App Studio와 관련된 서비스 할당량 및 제한이 포함된 주제를 추가했습니다. 자세한 내용은 AWS App Studio의 할당량을 참조하세요.	2024년 9월 11일

업데이트된 주제: 암호화된 DynamoDB 테이블에 연결	App Studio에서 AWS KMS 고객 관리형 키(CMKs)로 암호화된 DynamoDB 테이블을 사용하는 데 필요한 권한과 같은 정보가 추가되었습니다. 자세한 내용은 DynamoDB에 연결을 참조하세요.	2024년 9월 6일
업데이트된 주제: DynamoDB, Amazon Redshift 및 Aurora에 연결	App Studio 앱에서 DynamoDB, Amazon Redshift 및 Aurora 리소스를 사용하기 위해 IAM 역할에 추가하는 데 필요한 최소 권한을 추가했습니다. 자세한 내용은 AWS 서비스에 연결을 참조하세요.	2024년 9월 5일
업데이트된 주제: Amazon Aurora에 연결	App Studio 앱과 함께 사용할 Amazon Aurora 데이터베이스 및 테이블을 생성하고 구성하기 위한 설명서를 업데이트했습니다. 자세한 내용은 Amazon Aurora에 연결을 참조하세요.	2024년 9월 5일
신규 및 업데이트된 주제: 문제 해결 및 디버깅	애플리케이션 구축을 위한 디버깅 정보를 포함하여 App Studio의 일반적인 문제를 해결하는 데 도움이 되도록 문제 해결 및 디버깅 설명서를 확장했습니다. 자세한 내용은 App Studio 문제 해결 및 디버깅을 참조하세요.	2024년 8월 26일

새로운 주제: 자습서: Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 구축	자습서의 단계에 따라 최종 사용자의 입력 프롬프트를 받아 Amazon Bedrock으로 전송하고 요약된 버전을 반환하고 표시하는 앱을 빌드합니다. 자세한 내용은 Amazon Bedrock을 사용하여 AI 텍스트 요약기 앱 빌드를 참조하세요.	2024년 8월 20일
업데이트된 주제: App Studio 앱 미리 보기, 게시 및 공유	설명서 미리 보기, 게시 및 공유를 확장하여 명확성을 높이고, 서비스 경험을 일치시키고, 게시 환경 및 게시 환경 내 앱 보기에 대한 추가 정보를 제공했습니다. 자세한 내용은 애플리케이션 미리 보기, 게시 및 공유를 참조하세요.	2024년 8월 2일
새 주제: 여러 사용자로 앱 빌드	설명서 미리 보기, 게시 및 공유를 확장하여 명확성을 높이고, 서비스 경험을 일치시키고, 게시 환경 및 게시 환경 내 앱 보기에 대한 추가 정보를 제공했습니다. 자세한 내용은 여러 사용자로 앱 빌드를 참조하세요.	2024년 8월 2일
업데이트된 주제: App Studio를 AWS 서비스에 연결	기타 AWS 서비스 커넥터를 생성할 때 AWS 리소스에 대한 액세스를 제공하기 위한 IAM 역할을 생성하고 제공하는 방법에 대한 정보가 추가되었습니다. 자세한 내용은 기타 AWS 서비스 커넥터를 사용하여 AWS 서비스에 연결을 참조하세요.	2024년 7월 29일

업데이트된 주제: 설정의 일부로 AWS 관리 사용자를 생성하기 위한 지침 추가	App Studio 설명서 설정에 AWS 리소스 관리를 위한 관리 사용자를 생성하는 지침이 추가되었습니다. 또한 커넥터 설명서 전체에서 해당 사용자 사용을 권장하도록 업데이트했습니다.	2024년 7월 24일
새 주제: Amazon Bedrock에 연결	Amazon Bedrock용 커넥터 생성 지침이 포함된 주제가 추가되었습니다. 빌더는 커넥터를 사용하여 Amazon Bedrock을 사용하는 앱을 빌드할 수 있습니다. 자세한 내용은 Amazon Bedrock에 연결을 참조하세요.	2024년 7월 24일
최초 릴리스	AWS App Studio 사용 설명서의 최초 릴리스	2024년 7월 10일

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.