User Guide

Amazon Connect Health

Amazon Connect Health: User Guide

Table of Contents

What is Amazon Connect Health?

Amazon Connect Health is an AI-powered healthcare service built on Amazon Connect. It provides pre-built agents that automate patient engagement workflows and support clinical documentation at the point of care. You can use Amazon Connect Health to streamline voice-based patient interactions and give clinicians real-time insights during patient visits.

Amazon Connect Health integrates with electronic health record (EHR) systems through FHIR R4 APIs. This integration enables agents to access and update patient data directly from your EHR. Currently, Amazon Connect Health supports Epic EHR systems.

> **Important**
>
> Amazon Connect Health is not a medical device. Information generated by AI may contain mistakes and should be reviewed. Healthcare providers retain full responsibility for the accuracy of clinical documentation and the coordination of patient care.

Topics

Features of Amazon Connect Health

Amazon Connect Health provides the following features.

Patient engagement agents

Patient engagement agents handle inbound and outbound voice interactions with patients through Amazon Connect.

- **Patient verification** – Verifies patient identity at the start of a call by matching caller information against EHR records.
- **Appointment management** – Helps patients schedule, reschedule, and cancel appointments through automated voice conversations.

Point of care agents

Point of care agents assist clinicians during patient visits by providing relevant information and automating documentation tasks.

- **patient insights** – Generates pre-visit summaries that consolidate patient history, medications, and recent lab results from the EHR. This feature is currently available as a preview.
- **ambient documentation** – Captures patient-clinician conversations in real time and generates structured clinical documentation for provider review.

Patient profile in Amazon Connect Agent Workspace

Amazon Connect Health surfaces a unified patient profile within the Amazon Connect Agent Workspace. This profile gives agents a consolidated view of patient demographics, visit history, and relevant clinical data during live interactions.

EHR integration

Amazon Connect Health connects to your EHR system through FHIR R4 APIs. This integration enables bidirectional data exchange for patient verification, appointment management, and clinical data retrieval. Amazon Connect Health currently supports integration with Epic EHR systems.

Are you a first-time user?

If you are a first-time user of Amazon Connect Health, we recommend that you begin by reading the following sections:

- **the section called "Features of Amazon Connect Health"** – Learn about the features that Amazon Connect Health provides.
- ***Setting up Amazon Connect Health*** – Understand the prerequisites and configure your environment.

Supported Regions

Amazon Connect Health is available in the following AWS Regions:

Region name	Region code	Feature availability
US East (N. Virginia)	us-east-1	All features
US West (Oregon)	us-west-2	All features

> **Note**
>
> Patient insights is available as a preview in both supported Regions. Ambient documentation is generally available in both supported Regions.

Accessing Amazon Connect Health

You can access Amazon Connect Health in the following ways:

- **AWS Management Console** – Sign in to the AWS Management Console and open the Amazon Connect Health console. The console provides a web-based interface for configuring and managing your Amazon Connect Health agents.
- **AWS CLI** – Use the AWS Command Line Interface to interact with Amazon Connect Health from the command line. For more information about installing and configuring the AWS CLI, see the AWS CLI User Guide.
- **AWS API** – Use the Amazon Connect Health API to programmatically manage resources. For more information, see the Amazon Connect Health API Reference.
- **AWS SDKs** – Use the AWS SDKs to access Amazon Connect Health from your preferred programming language. For more information, see Developing with the AWS SDKs.

Related services

Amazon Connect Health works with the following AWS services:

- Amazon Connect – Cloud-based contact center service that powers the voice channel for patient engagement agents.
- AWS HealthLake – FHIR-compliant data store that patient insights uses to retrieve clinical records.
- AWS Lambda – Serverless compute service used for insurance verification integration.
- Amazon S3 – Object storage used for patient insights input documents and output delivery.
- AWS KMS – Key management service used for encryption of data at rest.

Pricing

Amazon Connect Health pricing is based on usage. For current pricing information, see Amazon Connect Health pricing.

Setting up Amazon Connect Health

This section is intended for IT administrators who are responsible for setting up and configuring Amazon Connect Health.

Before you use Amazon Connect Health, complete the following tasks.

Topics

Sign up for an AWS account

If you do not have an AWS account, complete the following steps to create one.

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call and entering a verification code on the phone keypad.

   When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to a user, and use only the root user to perform tasks that require root user access.

Create a user with administrative access

After you sign up for an AWS account, secure your AWS account root user, enable AWS IAM Identity Center, and create an administrative user so that you don't use the root user for everyday tasks.

Secure your AWS account root user

1. Sign in to the AWS Management Console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   For help signing in by using root user, see Signing in as the root user in the *AWS Sign-In User Guide*.

2. Turn on multi-factor authentication (MFA) for your root user.

   For instructions, see Enable a virtual MFA device for your AWS account root user (console) in the *IAM User Guide*.

Create a user with administrative access

1. Enable IAM Identity Center.

   For instructions, see Enabling AWS IAM Identity Center in the *AWS IAM Identity Center User Guide*.

2. In IAM Identity Center, grant administrative access to a user.

   For a tutorial about using the IAM Identity Center directory as your identity source, see Configure user access with the default IAM Identity Center directory in the *AWS IAM Identity Center User Guide*.

Sign in as the user with administrative access

1. To sign in with your IAM Identity Center user, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user.

   For help signing in using an IAM Identity Center user, see Signing in to the AWS access portal in the *AWS Sign-In User Guide*.

Create an Amazon Connect Health domain

A domain is a top-level container of resources and service configurations for Amazon Connect Health. You can have up to 10 domains in each account.

To create a domain, complete the following steps:

1. Sign in to the AWS Management Console and open the Amazon Connect Health console.
2. Choose **Create domain**.
3. Choose the scope of AI capabilities for the domain:
   - **Agents for patient engagement** — Enables AI agents for automated administrative support for patients and EHR integration, with testing and agent customization provided in an application.
   - **Agents for point of care** — Provides agents for use by healthcare professionals and office staff to support clinical workflows with a unified SDK.
   - **For both** — Enables all patient engagement and point of care capabilities simultaneously.
4. Enter domain details:
   - **Name** — Enter a domain name (for example, your EHR or health system name). Valid characters are a–z, A–Z, 0–9, underscore (_), and hyphen (-), up to 100 characters.
   - **Customize Encryption Settings** — Data is encrypted by default using an AWS managed key. Optionally, select **Customize encryption settings (advanced)** to use a customer managed key.

   > **Note**
   >
   > The remaining steps don't apply to domains for point-of-care agents only. For point-of-care setup, see the section called "Patient insights" and the section called "Ambient documentation".

5. Add users through AWS IAM Identity Center to provide access to the Amazon Connect Health application.
6. (Optional) Configure an integration function. Set up an AWS Lambda function for the AI agent to perform insurance verification using your own insurance RTE vendor. See sample-healthcare-realtime-eligibility on GitHub for a reference implementation. Choose **Create function** to build a new function, then enter the Lambda ARN in the provided field.
7. (Optional) Deploy a sample agent flow. Set up an Amazon Connect instance to deploy a sample contact flow and test the agent in an end-to-end patient conversation:
   - **Skip for now** — Defer this setup to a later time.
   - **Create and use a new Amazon Connect instance** (selected by default) — Recommended for most users. The access URL is auto-populated based on the domain name.

- **Use an existing Amazon Connect instance** — For organizations with an existing Amazon Connect instance.

> **Important**
>
> Before you create a new Amazon Connect instance, check your service quota. By default, each account can have 2 Amazon Connect instances. To request a quota increase, see Amazon Connect service quotas in the *Amazon Connect Administrator Guide*.

> **Important**
>
> Inputs on the domain creation page cannot be changed after domain creation, except for fields marked as recommended.

Manage user access

User access to the Amazon Connect Health application is managed through AWS IAM Identity Center. You can manage users in two ways:

- Use the IAM Identity Center widget in the domain setup page to directly add users. This approach is ideal for quick testing.
- Use the IAM Identity Center CLI or API to manage users, groups, and application assignments. This approach supports enterprise identity sources such as Active Directory and external identity providers. For more information, see Users, groups, and provisioning in IAM Identity Center.

> **Important**
>
> Amazon Connect Health must be in the same AWS Region as your AWS IAM Identity Center instance. If they are in different Regions, you can replicate your IAM Identity Center instance to an additional Region or change your Amazon Connect Health Region.

Enable single sign-on with Amazon Connect

To enable single sign-on (SSO) between Amazon Connect and Amazon Connect Health, assign the same IAM Identity Center user or user group to both applications. With SSO enabled, workforce users authenticate once and can access both Amazon Connect and Amazon Connect Health based on their enterprise identity.

Amazon Connect is available directly from the IAM Identity Center application catalog. See Step-by-step instructions to integrate Amazon Connect with IAM Identity Center using SAML 2.0.

Access the Amazon Connect Health application

After creating the domain, in the **Agents for patient engagement** section, choose **Open Application**. This launches the Amazon Connect Health application in your browser. If you don't have a valid session, you are prompted to sign in with your configured identity provider.

You can bookmark and share the application URL for direct access by authorized users.

Patient engagement agents

Amazon Connect Health provides two AI agents for patient engagement: the Patient verification agent and the Appointment management agent. These agents handle routine patient interactions through voice calls via Amazon Connect. They integrate with Epic EHR systems in real time through FHIR R4 APIs and support intelligent escalation to human agents with full context preservation.

Communication channels

Amazon Connect Health patient engagement agents currently support the voice channel only through Amazon Connect. Web chat, SMS, and other digital channels are not supported in the current release.

Configuring and testing patient engagement agents

This section is intended for administrative and clinical staff who configure and test patient engagement agents.

After your administrator completes the steps in *Setting up Amazon Connect Health*, you can explore the AI agents, test them with your own EHR environment, and customize their behavior in the Amazon Connect Health application.

Topics

Agent demo and testing

The Amazon Connect Health application home page provides two ways to experience the patient engagement AI agents:

- **Hear Healthcare Agents in action** – Play a recorded conversation between a patient and the Amazon Connect Health agents. The demo shows how the agents verify a patient's identity and

schedule an appointment through natural conversation with contextual understanding. You can play the audio directly from the home page without prior configuration.

- **Test your agent** – Experience a live conversation with the Amazon Connect Health agents integrated with your own EHR and Amazon Connect environment by calling your Amazon Connect provisioned number. This option becomes available after you complete EHR integration. Until then, you are prompted to configure your EHR setup.

Agent customization

After exploring the demo, you can tailor agent behavior to match your organization's workflows.

1. On the home page, choose **Customize** on the Agent setup card.
2. Enable or disable scheduling capabilities – **Schedule**, **Reschedule**, **Cancel Appointments**, and **Verify Insurance**. Unchecked tasks are routed to a representative.
3. Configure three sequential identity verification steps by selecting required patient inputs, such as phone number or medical record number (MRN), date of birth, and zip code or last four Social Security number (SSN) digits.
4. Choose **Publish** to apply the updates.

For more information about customization options, see the section called "Agent customization".

Sample contact flow for testing

Amazon Connect Health provides a pre-built sample contact flow (`connect-health-{DOMAIN_ID}-inbound-flow`) for testing and operationalizing appointment management conversations powered by AI agents. The flow simulates a typical appointment scheduling conversation, including patient identity verification. You can use this flow to validate end-to-end agent behavior – including EHR data lookup, insurance verification, and escalation routing – before moving to production.

The sample flow is deployed automatically at domain creation, whether Amazon Connect Health creates a new Amazon Connect instance or you use an existing one.

For instructions on locating, using, and customizing the sample contact flow, see the section called "Contact flow".

Communication channel support

Amazon Connect Health currently supports the voice channel through Amazon Connect. Web chat, SMS, and other digital channels are not supported in the current release.

Next steps

After you configure and test your agents, explore the following topics:

- *Patient engagement* – Learn about the patient verification and appointment management agents.
- *Point of care* – Learn about agents for clinical workflows.

Patient verification agent

The Patient verification agent provides secure, conversational identity verification through real-time EHR integration and configurable multi-attribute authentication such as date of birth and phone number. It provides patient profile lookup based on incoming call number and eliminates manual lookup by contact center staff. This agent collects and verifies patient identity through self service for caregivers on behalf of a patient.

The Patient verification agent eliminates time-consuming manual EHR lookups by automating the identity verification process. It queries Epic FHIR APIs in real time to match and verify patient records during the call.

Topics

- Capabilities
- Customization options
- Consent and patient notification

Capabilities

The Patient verification agent provides the following capabilities:

- **Real-time patient identity verification** – Verifies patient identity through Epic FHIR APIs.
- **Configurable multi-factor authentication** – Supports verification using phone number, medical record number (MRN), date of birth, zip code, and Social Security number (SSN).

- **LLM-based safety guardrails** – Detects safety concerns, frustrated patients, and complex requests.
- **Intelligent escalation to human agents** – Escalates to human agents with full verification context.
- **24/7 availability** – Provides around-the-clock availability for routine verification tasks.

Customization options

You can configure the Patient verification agent to match your organization's verification requirements. To customize the agent, choose **Customize** on the Agent setup card in the Amazon Connect Health console.

You can configure the following settings.

Verification attributes

Choose which factors are required for authentication, such as MRN, date of birth, zip code, or last four digits of SSN.

After you configure the settings, choose **Publish** to apply the changes.

Consent and patient notification

Amazon Connect Health patient engagement uses AI to capture and transcribe conversations in real time. Because this feature records spoken communications that may contain protected health information (PHI), customers and their downstream integrators are responsible for complying with all applicable consent, recording, and privacy laws. This includes obtaining all legally required consents before enabling ambient documentation for any patient encounter.

AWS's patient engagement agent provides the following language at the outset of each interaction:

"Hi, I'm your AI assistant. This call may be monitored and recorded by your health care provider and its service providers to improve their services."

Appointment management agent

The appointment management agent handles appointment scheduling, rescheduling, cancellations, and lookup through natural conversation integrated with real-time provider availability in EHR.

The appointment management agent handles full scheduling tasks from finding available slots across a provider's locations to booking, rescheduling, and canceling appointments within a single intelligent conversation. With optional real-time insurance eligibility verification and copay transparency built into the booking flow, the agent reduces administrative friction for both patients and staff while ensuring care continuity. When specialized needs are detected, the agent escalates immediately to human staff with a complete summary including appointment details, patient preferences, and verification status, so no context is lost in the handoff.

Topics

Capabilities

The Appointment management agent provides the following capabilities.

- **New appointment scheduling with real-time provider availability lookup** – Finds available slots across a provider's locations and books appointments within a single conversation.
- **Appointment rescheduling and cancellation** – Retrieves the current appointment, presents available alternatives for rescheduling, or processes cancellations and provides confirmation to the patient.
- **Appointment lookup and status confirmation** – Retrieves and confirms upcoming appointment details for the patient.
- **Optional real-time insurance eligibility (RTE) verification** – Verifies insurance eligibility before confirming a booking. This feature requires a customer-managed AWS Lambda function.
- **Copay transparency prior to appointment confirmation** – Provides copay information to the patient before the appointment is confirmed, so there are no surprises at the time of the visit.

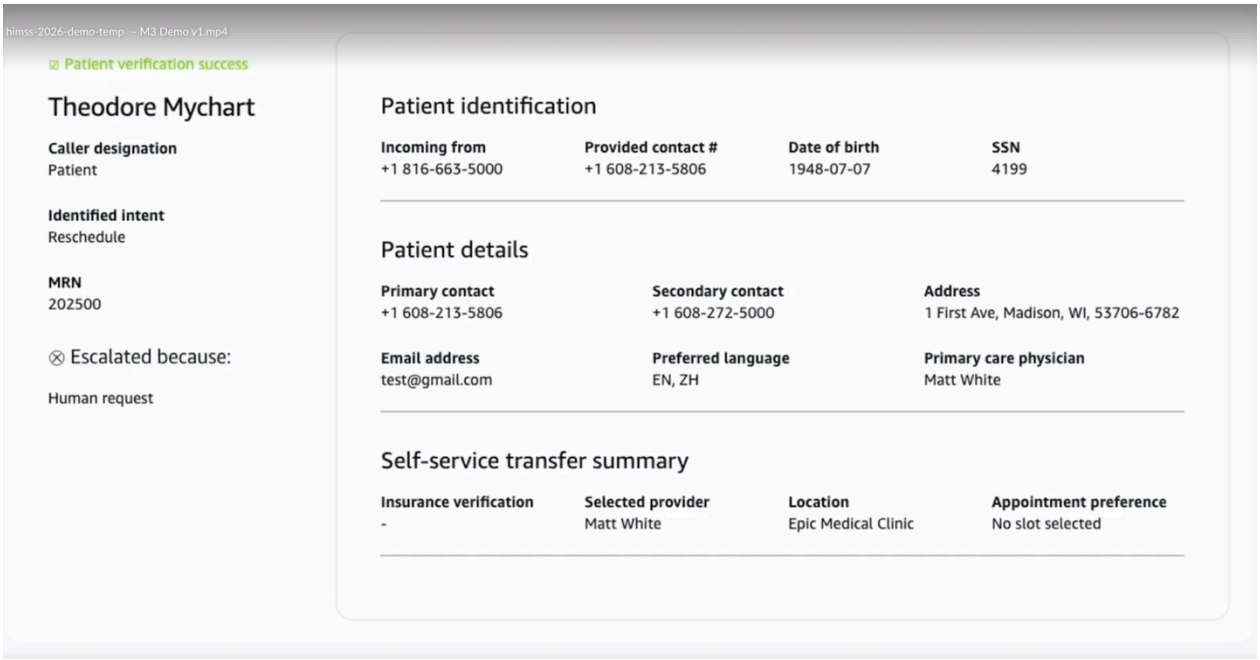
Configuration options

You can configure the following settings for the Appointment management agent.

Setting	Description
Enabled capabilities	Choose which appointment actions are available to patients: schedule, reschedule, cancel, and lookup. Unchecked tasks are routed to a human representative.
Insurance verification	Enable or disable real-time insurance eligibility verificat ion. When enabled, the agent invokes a customer-managed Lambda function to check eligibility and retrieve copay information. For more information, see the section called "Insurance verification integration".
AI Autonomy	Configure whether the appointment is fully self-serviced or preferences are collected and shared with staff for final action.

Patient profile

The patient profile is a unified view displayed in the Amazon Connect Agent Workspace that provides human agents with complete context when they are connected with a caller. Whether the call is transferred after successful patient verification or escalated during appointment management, this profile reduces the need for patients to repeat information and reduces cognitive burden on contact center staff.



Topics

Patient information displayed

The patient profile displays the following demographic and clinical information:

- Full name, date of birth, and medical record number (MRN)
- Contact information, including phone number and address
- Primary care provider and care team

Appointment intent and details

When a call involves appointment management, the profile displays:

- Appointment type, date, time, and location
- Provider name and department
- Scheduling intent captured during the AI interaction

Verification status

The profile shows the outcome of the identity verification process:

- Verification outcome: verified, partially verified, or failed
- Factors that were successfully verified
- Number of verification attempts

Escalation information

When a call is escalated from an AI agent, the profile includes:

- Reason for escalation, such as a safety concern, complex request, patient frustration, or verification failure
- Escalation timestamp and originating flow

Self-service summary

The profile provides a summary of actions completed by the AI agent before the escalation:

- Actions completed or attempted during the self-service interaction
- Appointment changes made or confirmed
- Insurance verification outcome, if applicable

Error scenarios

When patient verification was incomplete or failed, the patient profile displays all information collected from the caller during the verification attempt. This ensures that human agents can see which verification factors were already provided and which ones failed or were missing, eliminating the need to ask patients to repeat information they have already shared.

Agent customization

After exploring the demo, you can tailor agent behavior to match your organization's workflows. The customization page lets you configure scheduling capabilities, insurance verification, and identity verification steps.

The following table summarizes the customization areas available for each agent.

Setting	Description
Scheduling capabilities	Select which appointment actions (schedule, reschedule, cancel, lookup) are available to patients. For more information, see the section called "Appointment management agent".
Insurance verification	Enable or disable real-time insurance eligibility verificat ion through customer-owned Lambda functions. For more information, see the section called "Insurance verification integration".
AI autonomy	Configure whether the appointment is fully self-serviced or preferences are collected and shared with staff for final action. For more information, see the section called "Appointment management agent".
Verification attributes	Select which factors (for example, medical record number (MRN), date of birth, zip code) are required for authentication. For more information, see the section called "Patient verificat ion agent".

To customize agent settings

1. Open the Amazon Connect Health console and choose **Customize** on the Agent setup card.
2. In the customization page, enable or disable scheduling capabilities:
   - Schedule appointments
   - Reschedule appointments
   - Cancel appointments

- Verify insurance

  Unchecked tasks are routed to a representative.

3. Configure three sequential identity verification steps by selecting the required patient inputs:
   - **Step 1** – Phone number or MRN
   - **Step 2** – Date of birth
   - **Step 3** – Zip code or last four digits of Social Security number (SSN)
4. Choose **Publish** to apply the updates.

Expected results

After you choose **Publish**, the updated configuration takes effect. The agent uses your selected scheduling capabilities and verification steps when handling patient interactions.

Sample contact flow

Amazon Connect Health provides a pre-built sample contact flow for testing and operationalizing appointment management conversations powered by AI agents. The flow simulates a typical appointment scheduling conversation including patient identity verification. You can use it to validate end-to-end agent behavior before moving to production.

The sample flow is deployed automatically at domain creation, whether Amazon Connect Health creates a new Amazon Connect instance or you use an existing one.

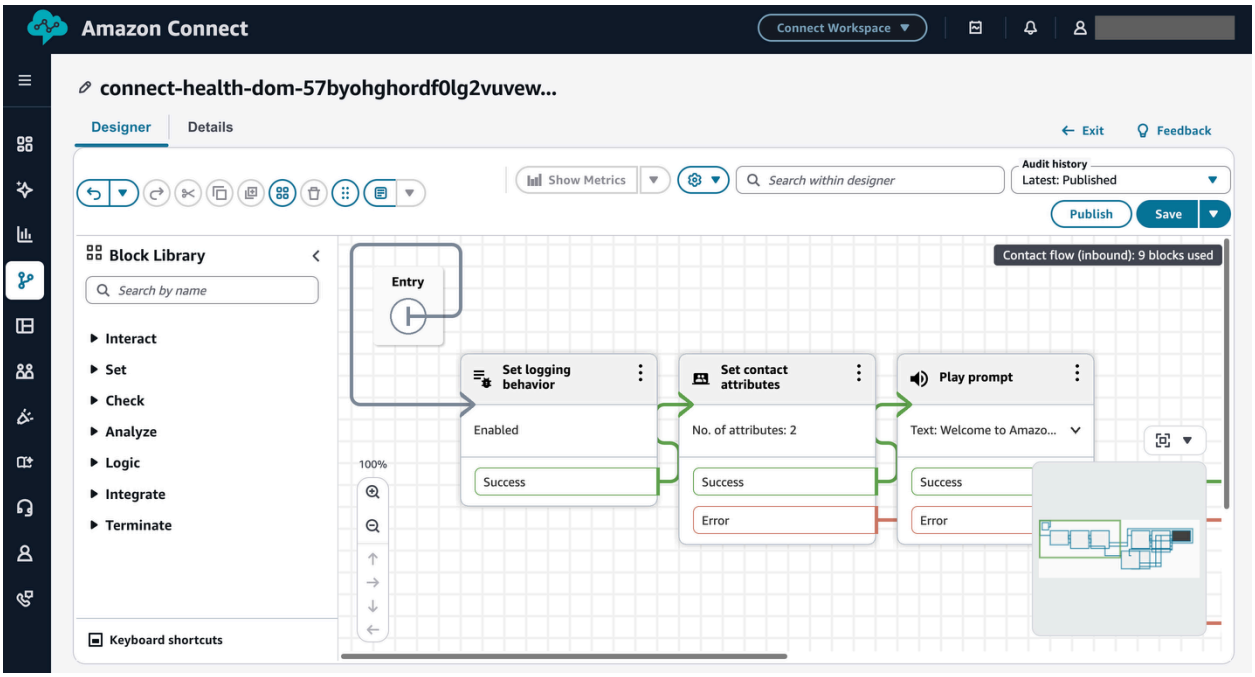
Topics

Locate the sample flow

To find the sample contact flow:

1. Sign in to your Amazon Connect instance.
2. In the navigation pane, choose **Routing**, then choose **Contact flows**.
3. Search for `connect-health-{DOMAIN_ID}-inbound-flow`, where `{DOMAIN_ID}` is your Amazon Connect Health domain identifier.

You can assign this flow to your preferred phone number if the provisioned number doesn't fit your workflows.



Provisioned resources

Sample Connect Health contact flow

Amazon Connect Health provisions the following resources as part of the sample contact flow deployment:

Resource	Naming pattern
Amazon Connect instance (if created by Amazon Connect Health)	`connect-health-{DOMAIN_ID}`
Sample contact flow	`connect-health-{DOMAIN_ID}-inbound-flow`

Resource	Naming pattern
Lambda function	`connect-health-{DOMAIN_ID}-lambda`
Lambda function role	`connect-health-{DOMAIN_ID}-lambda-role`
Amazon Lex bot	`connect-health-{DOMAIN_ID}-bot`
Amazon Lex bot role	`connect-health-{DOMAIN_ID}-lex-bot-role`

Session attributes

Session attributes are key-value pairs passed between contact flow blocks and Lambda functions during a voice call. Amazon Connect administrators configure session attributes to carry patient context — such as verification status, appointment intent, and escalation reason — across flow transitions.

The sample contact flow includes the following session attributes for the AI agent identifiers:

- `patient_agent_id` – The identifier for the Patient verification agent.
- `appointment_agent_id` – The identifier for the Appointment management agent.

To view or modify session attributes, open the sample flow in the Amazon Connect flow editor and inspect the **Set contact attributes** block.

Lambda function

The pre-built Lambda function (`connect-health-{DOMAIN_ID}-lambda`) handles EHR lookups, insurance verification, and session attribute processing during each patient call. The associated IAM execution role (`connect-health-{DOMAIN_ID}-lambda-role`) provides the required permissions. The Lambda function currently includes `"appointmentType": {"id": "1004", "type": "External"}` and can be updated to handle other appointment types you have.

You can extend or replace this function with your own implementation to support custom integrations or business logic.

Amazon Lex bot

The Amazon Lex bot (`connect-health-{DOMAIN_ID}-bot`) enables natural language understanding and supports patient conversation segments beyond the scope of the Amazon Connect Health agents, such as initial greetings and intent routing.

Customize the contact flow

Amazon Connect Health provisions a sample Amazon Connect contact flow that demonstrates how Amazon Connect Health agents are invoked within a call routing sequence, but it is intentionally generic. Customize the sample contact flow to align with your operational processes, queue structures, escalation policies, and patient communication standards (including privacy and consent practices) before production deployment.

Customization typically includes the following tasks:

- **Queue assignments and routing profiles** — Reassign contact flow branches to your existing queues and agent routing profiles configured in your Amazon Connect instance.
- **Call flow logic** — Update branching conditions and agent handoff sequences to reflect your department processes.
- **IVR integration** — Integrate the sample flow with existing IVR menus or self-service flows so that Amazon Connect Health agents are reachable within your broader telephony environment.
- **Localization** — Replace default audio with organization-approved recordings or text-to-speech content that reflects your communication standards.

For guidance on building and modifying contact flows, see Use the flow designer in Amazon Connect to create flows in the *Amazon Connect Administrator Guide*.

Insurance verification integration

Real-time insurance eligibility (RTE) verification is an optional feature of the Appointment management agent. When enabled, the agent verifies a patient's insurance eligibility and retrieves copay information prior to confirming an appointment. This feature is recommended for organizations that want to provide copay transparency before scheduling or rescheduling appointments.

Topics

Overview

Customers are responsible for creating and maintaining their Lambda function. AWS provides sample Lambda code that customers must update with their vendor-specific API integration details and authentication logic. Customers must deploy the Lambda to their AWS account, configure it with vendor credentials (using AWS Secrets Manager recommended), and add a resource policy allowing health-agent.amazonaws.com service principal to invoke the function.

When to use insurance verification

Use insurance verification integration when you want to:

- Provide copay transparency to patients before confirming appointments
- Integrate with your preferred RTE vendor (such as Experian Health or Waystar)

You don’t need RTE Lambda if:

- Post-booking insurance verification through Epic’s private APIs is sufficient for your workflow
- You prefer to handle insurance verification through existing staff processes

How it works

Insurance verification involves a setup phase and a runtime phase.

Setup phase

The customer deploys a Lambda function that connects to their preferred RTE vendor (for example, Experian Health or Waystar). The Lambda function is registered with Amazon Connect Health and granted invocation permissions.

Runtime phase

When a patient schedules or reschedules an appointment, the Appointment management agent invokes the customer's Lambda function with patient and appointment details. The Lambda function queries the RTE vendor and returns eligibility status and copay information to the agent, which presents the results to the patient before confirming the appointment.

Lambda input and output schema

AWS provides sample Lambda code for patient insurance verification. You update the sample code with your vendor-specific integration details and deploy to your production account.

For the sample Lambda code, see sample-healthcare-realtime-eligibility on GitHub.

Input schema

The Appointment management agent provides the following information to your Lambda function.

Field	Description
CoverageDetails.identifier	Payer code identifying the insurance company
CoverageDetails.groupNumber	Insurance group number
CoverageDetails.insuranceName	Free-text insurance name
CoverageDetails.memberNumber	Subscriber's member ID
subscriber.identifier	System identifier for the subscriber (optional)
subscriber.name	Subscriber name (optional)
subscriber.dateOfBirth	Subscriber date of birth (optional)
subscriber.relationshipToPatient	Relationship to the patient, for example "Self" (optional)
patientIdentifier	Patient identifier

Field	Description
requestPeriodStart	Service date start
requestPeriodEnd	Service date end
providerNPI	Provider's 10-digit NPI
providerLastName	Provider last name (optional)
departmentNPI	Department NPI (optional)

Output schema

Your Lambda function must return the following information to the Appointment management agent.

Field	Type	Description
eligibilityStatus	String	One of: `eligible`, `ineligible` , or unknown
copayAmount	Number (optional)	Estimated copay in USD
coverageDetails	String (optional)	Free-text summary of coverage
errorMessage	String (optional)	Error description if verification failed

Lambda resource policy

The customer must attach a resource-based policy to their Lambda function granting the Amazon Connect Health service principal invocation permissions. The policy must include:

- **Effect**: Allow
- **Principal Service**: `health-agent.amazonaws.com`
- **Action**: `lambda:InvokeFunction`
- **Resource**: Your Lambda function ARN

- **Condition**: `ArnLike` with `AWS:SourceArn` matching `arn:aws:health-agent:<region>:<aws-account-id>:*`

See Granting Lambda function access to AWS services for reference.

For the complete policy template, see the sample Lambda code at https://github.com/aws-samples/sample-healthcare-realtime-eligibility.

Setup steps

1. Download the sample Lambda code from https://github.com/aws-samples/sample-healthcare-realtime-eligibility.
2. Update the sample code with your RTE vendor's API integration details and authentication logic.
3. Deploy the Lambda function to your AWS account.
4. Configure the Lambda function with vendor credentials using AWS Secrets Manager (recommended).
5. Attach the resource policy to grant Amazon Connect Health permission to invoke the function. For more information, see Granting Lambda function access to AWS services in the *AWS Lambda Developer Guide*.
6. In the Amazon Connect Health console, configure the Lambda function ARN in the domain settings under **Integration function**.
7. Test the integration in a non-production environment before enabling in production.

Point of care agents

Amazon Connect Health point of care features are AI-powered capabilities that streamline administrative workflows in outpatient clinical settings. Point of care agentic capabilities combine speech recognition, generative AI, and reasoning to reduce documentation burden for clinicians and back-office staff.

Point of care features operate within a domain that you provision in your AWS account. Each feature accepts inputs — such as a real-time audio stream of a patient-clinician conversation, patient context from the EHR, and a clinical note template — and produces structured outputs for provider review including clinical documentation, evidence mappings, and after-visit summaries.

At launch, point of care capabilities include patient insights and ambient documentation.

Topics

Key concepts

Concept	Description
Domain	An isolated environment within your AWS account where you provision point of care agents.
Subscription	A configuration resource that associates a provider or session with an agent. Required for ambient documentation.
Template	A structured definition of the clinical note format. Used by ambient documentation to generate documentation.
Vended artifacts	The output files written to your Amazon S3 bucket, such as pre-visit summaries, transcripts, and clinical notes.

Regional availability

Point of care agents are available in the following AWS Regions:

- US East (N. Virginia) - `us-east-1`
- US West (Oregon) - `us-west-2`

> **Note**
>
> Patient insights is available as a preview in both supported Regions. Ambient documentation is generally available in both supported Regions.

Patient insights

Patient insights synthesizes longitudinal patient data into a concise, actionable pre-visit summary by retrieving structured clinical records, clinical documents, and ad-hoc files, then generating a focused summary for clinicians to review before each encounter.

Patient insights provides the following capabilities:

- **Longitudinal data synthesis** – Retrieves and synthesizes patient data scattered across structured FHIR resources, FHIR DocumentReference/Binary resources, and S3 documents into a single unified pre-visit summary.
- **Evidence-linked output** – Every clinical statement in the summary includes evidence references linking back to the specific FHIR resources or source documents that support it.
- **Focus summaries by encounter reason** – Tailors insights to encounter reason or chief complaint. For example, a wellness visit summary emphasizes preventive screenings and immunization status, while a surgical follow-up summary prioritizes procedure history and post-operative notes.
- **Seamless workflow integration** – Delivers a structured JSON document that gives full control over rendering within your existing workflows.
- **AWS ecosystem integration** – Integrates with AWS HealthLake as a FHIR-compliant data store and Amazon S3 for both document ingestion and summary delivery.

Patient insights is available as a preview in US East (N. Virginia) (`us-east-1`) and US West (Oregon) (`us-west-2`) Regions.

> **Important**
>
> Patient insights is available as a preview feature and is subject to change. Do not use preview features in production environments.

Topics

How patient insights works

Patient insights uses an asynchronous job-based workflow with four stages:

1. **Job submission** – Your application submits a patient insights job by calling the StartPatientInsightsJob API. The request specifies the patient, encounter type, requesting clinician, data sources, and output location. The API returns a unique job identifier.
2. **Data retrieval and synthesis** – Patient insights retrieves the patient's clinical data from your configured data sources. The service connects to your FHIR-compliant data store (such as AWS HealthLake) to pull structured clinical records. It also retrieves clinical documents from FHIR DocumentReference and Binary resources, and can ingest ad-hoc documents from Amazon S3.
3. **Polling for completion** – Your application polls for completion using the GetPatientInsightsJob API. A typical job completes in 2 to 5 minutes, depending on the volume and complexity of the patient's clinical history.
4. **Output retrieval** – When the job status reaches SUCCEEDED, the generated pre-visit summary is available at the S3 output path you specified. Your application retrieves the summary and presents it to the clinician.

Inputs

Patient context

Provide a unique patient identifier that corresponds to the patient's FHIR ID in your FHIR server. This identifier is the key the service uses to query the FHIR endpoint for that patient's clinical history.

Encounter reason

The encounter reason tells the service what type of visit is planned and why the patient is coming in. This helps the service prioritize which clinical information is most relevant. For example, a wellness visit summary emphasizes preventive screenings and immunization status, while a surgical follow-up summary prioritizes procedure history and post-operative notes.

User context

Identify the clinician requesting the summary, their role, and their specialty. This information helps the service tailor the summary to the clinician's needs.

Input data configuration

Specify where the service should look for the patient's clinical data:

- **FHIR server** – Provide a FHIR endpoint URL pointing to your FHIR-compliant data store (such as AWS HealthLake) along with an OAuth token for authentication. The service supports a configurable lookback period ranging from 1 to 24 months.
- **FHIR DocumentReference and Binary resources** – The service retrieves clinical documents such as discharge summaries, consultation notes, and diagnostic reports from the same FHIR endpoint.
- **S3 sources** – Specify documents stored in Amazon S3 that the service should include, such as faxed specialist referral letters, outside hospital records, or scanned imaging reports.

Supported file formats

File format	Maximum size
PDF (application/pdf)	500 MB or 3,000 pages

File format	Maximum size
JPEG (image/jpeg)	10 MB
PNG (image/png)	10 MB

Note

The service handles unsupported or oversized files differently depending on the source. S3 sources cause the job to fail if any file is unsupported or oversized. FHIR DocumentReference and Binary resources that are unsupported or oversized are silently skipped.

Document processing limits

Patient insights accepts a maximum of 10 documents per job across all document sources combined. The service processes documents in this order:

1. S3 source documents are processed first.
2. FHIR DocumentReference and Binary documents are processed in sequential order.

If the total number of documents exceeds 10, only the first 10 are processed according to this ordering.

Output data configuration

Specify an S3 output path where the service delivers the completed pre-visit summary. You manage the retention and lifecycle of patient insights output through your S3 bucket lifecycle policies.

Submitting and tracking jobs

After you submit a patient insights job, you can track its progress by polling the GetPatientInsightsJob API. The following table describes the possible job statuses:

Status	Description
SUBMITTED	The job has been accepted and is queued for processing.
IN_PROGRESS	The service is retrieving clinical data, processing source documents, and generating the patient summary.
SUCCEEDED	The job has completed successfully. The generated summary is available at the specified S3 output path.
FAILED	The job did not complete successfully. Check the error details in the GetPatientInsightsJob response for the specific failure reason.

Output

When a patient insights job completes successfully, Amazon Connect Health delivers a structured JSON document to your configured S3 output path. This document contains the pre-visit summary organized into clearly defined sections, with every clinical statement linked back to its supporting evidence.

Summary sections

The following tree shows the structure of the output schema:

```
PatientSummaries[] (array)
### PatientSummary
    ### JobId
    ### SummaryType (e.g., "Pre-visit")
    ### PatientId
    ### GeneratedAt (ISO 8601 timestamp)
    ### Sections[] (array)
        ### SectionName
        ### ClinicalNarrative[] (array)
            ### Text (clinical content with markdown formatting)
            ### Evidence[] (array of FHIR resource references)
TotalCount
GeneratedAt (ISO 8601 timestamp)
HccIncluded (boolean)
```

The pre-visit summary is organized into five sections:

Section	What it contains
PATIENT_AND_ENCOUNTER_OVERVIEW	Medical history, active conditions, current medications, allergies, past surgeries, and family history.
SINCE_LAST_VISIT	New labs, specialist notes, medication changes, symptoms, and interventions that have occurred since the patient's last encounter with this provider.
TRENDS	Patterns over time for key clinical metrics such as A1c, blood pressure, weight, and lab values.
CMS_HCC_CODING_ANALYSIS	Conditions previously documented that are relevant to CMS Hierarchical Condition Category (HCC) risk adjustment. The clinician confirms whether each condition is still present.
HHS_HCC_CODING_ANALYSIS	Conditions previously documented that are relevant to HHS-HCC risk adjustment. The clinician confirms whether each condition is still present.

Each section contains an array of clinical narrative objects. A clinical narrative is a discrete unit of clinical content that the service generates from the patient's data. The text within clinical narratives might include markdown formatting (such as bullet markers, headers, and emphasis) that your application can parse and render according to your display context.

The first three sections answer the questions clinicians typically need answered before walking into the exam room: Who is this patient? What's happened since I last saw them? How have things changed? The final two sections support Condition Review, which helps organizations capture appropriate revenue across both Fee-for-Service and Value-Based Care models. These sections surface previously documented conditions that carry HCC risk adjustment weight, along with the date each condition was last assessed and a prompt to confirm whether the condition is still clinically present. The CMS and HHS sections might surface overlapping conditions where both risk adjustment models apply, or might surface different conditions depending on each model's category definitions.

Evidence references

Every clinical narrative in the summary includes an evidence array that links the generated statement back to the specific FHIR resources or source documents that support it. Evidence references use the format `ResourceType/ResourceId`, such as `Condition/d68e8fc6-2e5f-4a3e-a2a7-942a435e78ec`.

Structural elements such as section headers or summary statements that synthesize across multiple sources might have an empty evidence array. Clinical statements derived from specific patient data — a particular lab result, a documented allergy, a past encounter, a previously assessed condition — include the corresponding FHIR resource references.

HccIncluded flag

The output includes a top-level `HccIncluded` flag indicating whether the Condition Review sections (CMS and HHS HCC Coding Analysis) were generated for the job.

Rendering the summary

To present the summary to clinicians in your application:

1. Iterate through each section in order.
2. Concatenate the text values from each clinical narrative within the section.
3. Parse the markdown formatting for your display context.
4. Use the evidence references to provide drill-down links to source FHIR resources or documents.

For the Condition Review sections, render each surfaced condition as an interactive element that allows the clinician to confirm, dismiss, or annotate each condition directly within the workflow. The structured output format gives you full control over how the summary is rendered.

For complete API parameter details and request/response schemas, see the Amazon Connect Health API Reference.

Ambient documentation

Ambient documentation captures patient-clinician conversations in real time and generates structured clinical documentation for provider review. The service combines speech recognition with generative AI to produce clinical notes, extract medical terminology, identify speaker roles, and classify dialogue segments.

> **Important**
>
> Amazon Connect Health ambient documentation produces probabilistic results. Output accuracy varies based on audio quality, background noise, speaker clarity, medical terminology complexity, and context-specific language. All output must be reviewed for accuracy by a trained medical professional before use in patient care.

Amazon Connect Health ambient documentation is available in the US East (N. Virginia) (`us-east-1`) and US West (Oregon) (`us-west-2`) Regions.

Topics

How ambient documentation works

Ambient documentation uses HTTP/2 streaming for real-time audio processing. The workflow includes:

1. **Create a subscription** — Associate a provider with the ambient documentation agent. Subscriptions are automatically created in activated mode.
2. **Stream audio** — Your application streams audio from the patient-clinician conversation to Amazon Connect Health using HTTP/2. The service transcribes the audio in real time and identifies speakers.

3. **Generate documentation** — After the conversation ends, the service generates structured clinical notes, evidence mappings, and an after-visit summary based on the configured template.
4. **Retrieve outputs** — The service writes the transcript, clinical documentation, and after-visit summary to your configured Amazon S3 bucket.

Technical requirements

- **Supported language** — US English (en-US)
- **Supported audio formats** — FLAC, PCM
- **Encoding** — PCM 16-bit
- **Sample rate** — 16,000 Hz or higher

Supported medical specialties

Ambient documentation currently supports the following specialties:

- Allergy Immunology
- Cardiology
- Dermatology
- Endocrinology
- Gastroenterology
- Hematology/Oncology
- Infectious Disease
- Nephrology
- Neurology
- OBGYN
- Oncology
- Ophthalmology
- Orthopedics
- Otolaryngology
- Pain Medicine
- Pediatrics

- Primary Care
- Psychiatry
- Pulmonology
- Rheumatology
- Surgery
- Urology

Consent and patient notification

Amazon Connect Health ambient documentation uses AI to capture and transcribe clinical conversations in real time. Because this feature records spoken communications that may contain protected health information (PHI), customers and their downstream integrators are responsible for complying with all applicable consent, recording, and privacy laws. This includes obtaining all legally required consents before enabling ambient documentation for any patient encounter. AWS does not collect consent from patients on your behalf.

Appropriate consent must be obtained from each patient and anyone present in the room when ambient documentation is used. As part of obtaining consent, patients should be informed that the visit will be recorded and used by an AI service provider to create clinical notes, that their information may be shared with service providers, and that they can decline without any impact on their care. Customers and integrators should maintain records of patient consent, in accordance with applicable state law and internal retention policies. Customers should ensure that consent is obtained in accordance with their organization's privacy practices.

Sample language:

"Before we begin, I want to let you know that today's visit will be recorded and monitored by an AI service provider to help with documentation. Do you consent to proceed?"

Subscription management

Create a subscription

To create a subscription, call the `CreateSubscription` API operation. This generates a unique `subscriptionId` that you use to authorize the user and start streaming sessions. Subscriptions are automatically created in activated mode.

Tip

Create the subscription on the user's first use to align subscription start with actual usage.

Deactivate a subscription

To temporarily stop a subscription from accepting new streaming sessions, call the `DeactivateSubscription` API operation. A deactivated subscription retains its configuration and can be reactivated later. In-progress streams complete normally.

Important

Deactivating a subscription immediately forfeits any remaining free trial period. If a subscription that was deactivated during a free trial is reactivated later, it is reactivated as a paid subscription.

Reactivate a subscription

Deactivated subscriptions can be reactivated by calling the `ActivateSubscription` API operation with the `subscriptionId`. Paid metering begins immediately upon reactivation.

Streaming audio

Ambient documentation uses HTTP/2 streaming to process audio in real time. Your application establishes an HTTP/2 connection, sends audio chunks as event-encoded messages, and receives transcription results as the conversation progresses.

If your audio has two channels, you can use channel identification to transcribe the speech from each channel separately. Ambient documentation currently supports audio with up to two channels. In your transcript, channels are assigned the labels `ch_0` and `ch_1`.

In addition to the standard transcript sections (transcripts and items), requests with channel identification enabled include a channel_labels section. This section contains each utterance or punctuation mark, grouped by channel, and its associated channel label, timestamps, and confidence score. Note that if a person on one channel speaks at the same time as a person on a separate channel, timestamps for each channel overlap while the individuals are speaking over each other.

For detailed information about HTTP/2 streaming setup, authentication, and event stream encoding, see the Amazon Connect Health API Reference.

Using the AWS SDKs

The following code example shows how to set up an Amazon Connect Health ambient documentation streaming session using the AWS SDK for Java 2.x.

```
package com.example.connecthealth;

import io.reactivex.rxjava3.core.BackpressureStrategy;
import io.reactivex.rxjava3.core.Flowable;
import org.reactivestreams.Publisher;
import org.reactivestreams.Subscriber;
import software.amazon.awssdk.auth.credentials.AwsCredentialsProvider;
import software.amazon.awssdk.auth.credentials.DefaultCredentialsProvider;
import software.amazon.awssdk.core.SdkBytes;
import software.amazon.awssdk.http.nio.netty.NettyNioAsyncHttpClient;
import software.amazon.awssdk.regions.Region;
import software.amazon.awssdk.services.connecthealth.ConnectHealthAsyncClient;
import software.amazon.awssdk.services.connecthealth.model.*;

import javax.sound.sampled.AudioFormat;
import javax.sound.sampled.AudioInputStream;
import javax.sound.sampled.AudioSystem;
import javax.sound.sampled.DataLine;
import javax.sound.sampled.LineUnavailableException;
import javax.sound.sampled.TargetDataLine;
import java.io.BufferedInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.io.UncheckedIOException;
import java.util.Arrays;
import java.util.UUID;
import java.util.concurrent.CompletableFuture;

public class MedicalScribeStreamingApp {
    private static final int CHUNK_SIZE_IN_BYTES = 6400;
    private static final int SAMPLE_RATE = 16000;
    private static final Region REGION = Region.US_WEST_2;
    private static final String SESSION_ID = UUID.randomUUID().toString();
    private static final String DOMAIN_ID = "your-domain-id";
    private static final String SUBSCRIPTION_ID = "your-subscription-id";
```

```
    private static final String OUTPUT_S3_URI = "s3://your-bucket/output/";
    private static ConnectHealthAsyncClient client;

    public static void main(String[] args) {
        client = ConnectHealthAsyncClient.builder()
                .credentialsProvider(getCredentials())
                .httpClientBuilder(NettyNioAsyncHttpClient.builder())
                .region(REGION)
                .build();
        try {
            StartMedicalScribeListeningSessionRequest request =
                    StartMedicalScribeListeningSessionRequest.builder()
                            .sessionId(SESSION_ID)
                            .domainId(DOMAIN_ID)
                            .subscriptionId(SUBSCRIPTION_ID)
                            .languageCode(MedicalScribeLanguageCode.EN_US)
                            .mediaSampleRateHertz(SAMPLE_RATE)
                            .mediaEncoding(MedicalScribeMediaEncoding.PCM)
                            .build();

            MedicalScribeInputStream endSessionEvent =
                    MedicalScribeInputStream.sessionControlEventBuilder()
                            .type(MedicalScribeSessionControlEventType.END_OF_SESSION)
                            .build();

            CompletableFuture<Void> result = client.startMedicalScribeListeningSession(
                    request,
                    new AudioStreamPublisher(
                            getStreamFromMic(),
                            getConfigurationEvent(),
                            endSessionEvent
                    ),
                    getResponseHandler()
            );
            result.get();
            client.close();
        } catch (Exception e) {
            System.err.println("Error occurred: " + e.getMessage());
            e.printStackTrace();
        }
    }

    private static AudioInputStream getStreamFromMic() throws LineUnavailableException
{
```

```
        AudioFormat format = new AudioFormat(SAMPLE_RATE, 16, 1, true, false);
        DataLine.Info info = new DataLine.Info(TargetDataLine.class, format);
        if (!AudioSystem.isLineSupported(info)) {
            throw new LineUnavailableException("Microphone line not supported");
        }
        TargetDataLine line = (TargetDataLine) AudioSystem.getLine(info);
        line.open(format);
        line.start();
        System.out.println("Recording... Press Enter to stop");
        Thread monitorThread = new Thread(() -> {
            try {
                System.in.read();
                line.stop();
                line.close();
            } catch (IOException e) {
                e.printStackTrace();
            }
        });
        monitorThread.setDaemon(true);
        monitorThread.start();
        return new AudioInputStream(
                new BufferedInputStream(new AudioInputStream(line)),
                format,
                AudioSystem.NOT_SPECIFIED
        );
    }

    private static AwsCredentialsProvider getCredentials() {
        return DefaultCredentialsProvider.create();
    }

    private static StartMedicalScribeListeningSessionResponseHandler
getResponseHandler() {
        return StartMedicalScribeListeningSessionResponseHandler.builder()
                .onResponse(r -> {
                    System.out.println("Session started: " + r.sessionId());
                    System.out.println("Domain ID: " + r.domainId());
                    System.out.println("Subscription ID: " + r.subscriptionId());
                    System.out.println("Request ID: " + r.requestId());
                })
                .onError(e -> {
                    System.err.println("Stream error: " + e.getMessage());
                    e.printStackTrace();
                })
```

```
                .onComplete(() -> {
                    System.out.println("=== Stream completed successfully ===");
                })
                .subscriber(event -> {
                    if (event instanceof MedicalScribeTranscriptEvent) {
                        MedicalScribeTranscriptSegment segment =
                                ((MedicalScribeTranscriptEvent)
event).transcriptSegment();
                        if (segment != null && segment.content() != null) {
                            System.out.printf("[%s][Channel %s] %s%n",
                                    segment.isPartial() ? "PARTIAL" : "FINAL",
                                    segment.channelId(),
                                    segment.content()
                            );
                        }
                    }
                })
                .build();
    }

    private static MedicalScribeConfigurationEvent getConfigurationEvent() {
        return MedicalScribeConfigurationEvent.builder()
                .postStreamActionSettings(
                        MedicalScribePostStreamActionSettings.builder()
                                .outputS3Uri(OUTPUT_S3_URI)
                                .clinicalNoteGenerationSettings(
                                        ClinicalNoteGenerationSettings.builder()
                                                .noteTemplateSettings(

NoteTemplateSettings.fromManagedTemplate(

ManagedTemplate.builder()

.templateType(ManagedNoteTemplate.SOAP)
                                                                                .build()
                                                        )
                                                )
                                                .build()
                                )
                                .build()
                )
                .channelDefinitions(Arrays.asList(
                        MedicalScribeChannelDefinition.builder()
                                .channelId(0)
```

```
.participantRole(MedicalScribeParticipantRole.CLINICIAN)
                                .build(),
                        MedicalScribeChannelDefinition.builder()
                                .channelId(1)
                                .participantRole(MedicalScribeParticipantRole.PATIENT)
                                .build()
                ))
                .build();
    }

    private static class AudioStreamPublisher implements
Publisher<MedicalScribeInputStream> {
        private final InputStream audioInputStream;
        private final MedicalScribeConfigurationEvent configEvent;
        private final MedicalScribeInputStream endSessionEvent;

        private AudioStreamPublisher(
                AudioInputStream audioInputStream,
                MedicalScribeConfigurationEvent configEvent,
                MedicalScribeInputStream endSessionEvent) {
            this.audioInputStream = audioInputStream;
            this.configEvent = configEvent;
            this.endSessionEvent = endSessionEvent;
        }

        @Override
        public void subscribe(Subscriber<? super MedicalScribeInputStream> subscriber)
{
            createAudioFlowable()
                    .doOnComplete(() -> {
                        try {
                            audioInputStream.close();
                        } catch (IOException e) {
                            throw new UncheckedIOException(e);
                        }
                    })
                    .subscribe(subscriber);
        }

        private Flowable<MedicalScribeInputStream> createAudioFlowable() {
            Flowable<MedicalScribeInputStream> configFlow = Flowable.just(
                    MedicalScribeInputStream.fromConfigurationEvent(configEvent)
            );
```

```
            Flowable<MedicalScribeInputStream> audioFlow = Flowable.create(emitter -> {
                byte[] buffer = new byte[CHUNK_SIZE_IN_BYTES];
                int bytesRead;
                try {
                    while (!emitter.isCancelled() && (bytesRead =
 audioInputStream.read(buffer)) > 0) {
                        byte[] audioData = bytesRead < buffer.length
                                ? Arrays.copyOfRange(buffer, 0, bytesRead)
                                : buffer;
                        MedicalScribeInputStream audioEvent =
                                MedicalScribeInputStream.fromAudioEvent(
                                        MedicalScribeAudioEvent.builder()

 .audioChunk(SdkBytes.fromByteArray(audioData))
                                                .build()
                                );
                        emitter.onNext(audioEvent);
                    }
                    emitter.onComplete();
                } catch (IOException e) {
                    emitter.onError(e);
                }
            }, BackpressureStrategy.BUFFER);
            Flowable<MedicalScribeInputStream> endFlow =
 Flowable.just(endSessionEvent);
            return Flowable.concat(configFlow, audioFlow, endFlow);
        }
    }
}
```

Storage

An S3 storage location must be specified at the start of a session. Output artifacts are stored at the following base location:

`s3://{customer-provided-uri}/health-agent-listening-session/{domainId}/{subscriptionId}/{sessionId}/post-stream-action/`

Clinical notes are stored in a `clinical-notes` folder at this base location.

Patient context

Patient context provides the agent with clinical history before the conversation through the `encounterContext` API parameter. The `encounterContext` object contains the field `unstructuredContext`, which accepts up to 10 KB of text data for each session. When you include patient context, the agent uses it to enrich the generated documentation with background information that was not explicitly discussed during the visit.

Patient context can include:

- Prior encounter notes and visit summaries
- Active medication lists
- Problem lists and diagnoses
- Allergies and immunizations
- Relevant lab and imaging reports
- Surgical and family history
- Patient's preferred pronouns, used when referring to the patient in generated clinical output

The context is used throughout the generated note to improve specificity and accuracy when the necessary information is not present in the transcript alone, with evidence mapping to source materials for clinician review.

> **Note**
>
> The following example uses fictional patient data for illustration purposes only.

Example patient context

```
## Patient Information
Name: Patricia Underwood
Age: 28 years
Sex: female
Pronouns: she/her
MEDICATIONS:
Ondansetron 4mg PO PRN - Nausea/vomiting
Dicyclomine 20mg PO BID PRN - Abdominal cramping
Sertraline 50mg PO daily - Depression
```

```
Ferrous sulfate 325mg PO TID - Iron deficiency anemia
ALLERGIES: NKDA (No Known Drug Allergies)
PAST MEDICAL HISTORY:
Brainstem pilocytic astrocytoma diagnosed 04/2010
Posterior fossa craniotomy with gross total resection 05/12/2010
Hyperprolactinemia diagnosed 08/2013
Autoimmune hemolytic anemia diagnosed 12/2012
Splenomegaly secondary to autoimmune hemolytic anemia 01/2013
Major depressive disorder diagnosed 08/2012
Substance use disorder (heroin) in sustained remission since 04/2011
Tobacco use disorder, quit 09/15/2013 (10 pack-year history)
Iron deficiency anemia diagnosed 01/2013
Gastroesophageal reflux disease diagnosed 05/2013
Appendectomy 07/23/2009 (uncomplicated laparoscopic)
Wisdom teeth extraction 11/08/2008
FAMILY HISTORY:
Maternal grandmother: Cervical cancer, ovarian cancer, dementia/Alzheimer's
Maternal aunt: Cervical cancer
Paternal grandfather: Type 2 diabetes, hypertension, kidney transplant
Maternal great-grandfather: Colon cancer
Multiple maternal great-uncles: Colon cancer
No family history of breast, uterine, or cardiac disease
SOCIAL HISTORY:
Tobacco: Former smoker, quit 09/15/2013 (10 pack-year history)
Alcohol: Not documented
Illicit drugs: History of IV heroin use, abstinent since 04/2011, occasional marijuana
 use
Sexual history: Single, sexually active, monogamous relationship since 05/2012
Previous relationship with military personnel ended 03/2012
Employment: Lives independently, employed
Contraception: Not currently using
Former plasma donor, discontinued 10/2013 due to positive syphilis screening
PROBLEM LIST:
History of brainstem pilocytic astrocytoma (C71.7) - Diagnosed 04/2010 with posterior
 fossa craniotomy and gross total resection 05/12/2010. Annual MRI surveillance shows
 post-surgical changes, no residual tumor. Most recent MRI 10/18/2013 normal. Followed
 by neurology with annual visits.
Hyperprolactinemia (E22.1) - Diagnosed 08/2013 with prolactin 45.2 ng/mL (normal <25).
 Referred to neurology for pituitary evaluation. Recent MRI 10/18/2013 shows normal
 pituitary gland. Not currently on treatment. Neurology follow-up scheduled.
Autoimmune hemolytic anemia with splenomegaly (D59.1) - Diagnosed 12/2012-01/2013.
 Positive direct Coombs test, elevated LDH 420 U/L, low haptoglobin <10 mg/dL,
 reticulocyte count 8.2%. Associated splenomegaly. Managed by primary care physician.
 On iron supplementation for concurrent iron deficiency.
```

```
Major depressive disorder, mild (F32.0) - Diagnosed 08/2012. Started sertraline 50mg
 daily 09/08/2012 with good response. Stable mood, no current suicidal ideation.
 Continues on current regimen.
Substance use disorder (heroin), in sustained remission (F11.21) - History of IV drug
 use, abstinent since 04/2011. Not currently in formal treatment program. Maintains
 abstinence, occasional marijuana use.
Iron deficiency anemia (D50.9) - Diagnosed 01/2013 concurrent with autoimmune hemolytic
 anemia. Started ferrous sulfate 325mg TID 01/14/2013. Recent Hgb 9.8 g/dL, Hct 29%,
 MCV 102 fL.
Gastroesophageal reflux disease (K21.9) - Diagnosed 05/2013. Managed with lifestyle
 modifications. Symptoms of nausea and vomiting, taking ondansetron and dicyclomine
 PRN.
RECENT LABS (Various dates 2013):
Prolactin: 45.2 ng/mL (08/22/2013, elevated)
CBC: Hgb 9.8, Hct 29%, MCV 102 fL (10/28/2013)
Direct Coombs: Positive (01/14/2013)
LDH: 420 U/L, Haptoglobin <10 mg/dL (01/14/2013)
MRI brain with contrast: Normal pituitary, post-surgical changes only (10/18/2013)
Pap smear: Normal cytology, HPV negative (11/15/2012)
Mammogram: BI-RADS 1, normal (02/28/2013)
```

Clinical note templates

Templates define the structure, sections, and formatting rules that the agent follows when generating clinical documentation. You must specify an output format by passing a configuration object to the `noteTemplateSettings` API parameter with every session. Ambient documentation supports two methods to specify the output format — managed templates or custom templates.

Managed templates

Ambient documentation provides seven pre-built note templates. The configuration object for managed templates, `managedTemplate`, specifies the template through the `templateType` parameter. The default template is HISTORY_AND_PHYSICAL.

Template	Description	Use case
HISTORY_AND_PHYSICAL (default)	Summaries for key clinical documentation sections	General physical health encounters

Template	Description	Use case
PHYSICAL_SOAP	Physical health focused SOAP format	Physical health encounters using SOAP structure
BEHAVIORAL_SOAP	Behavioral health focused SOAP format	Behavioral health encounters using SOAP structure
GIRPP	Progress-toward-goals format	Behavioral health — tracking patient progress
BIRP	Behavioral patterns and responses format	Behavioral health — documenting behavioral patterns
SIRP	Situational context of therapy format	Behavioral health — emphasizing situational context
DAP	Simplified clinical documentation format	Brief or focused encounters

HISTORY_AND_PHYSICAL sections

Section	Description
CHIEF COMPLAINT	Brief description of the patient's reason for visiting the clinician
HISTORY OF PRESENT ILLNESS	Information on the patient's illness, including severity, onset, timing, current treatments, and affected areas
REVIEW OF SYSTEMS	Patient-reported evaluation of symptoms across different body systems
PAST MEDICAL HISTORY	Previous medical conditions, surgeries, and treatments
PAST FAMILY HISTORY	Health conditions that run in the patient's family
PAST SOCIAL HISTORY	Social life, habits, occupation, and environmental factors affecting health

Section	Description
PHYSICAL EXAMINATION	Clinician's findings from physical examination of body systems and vital signs
DIAGNOSTIC TESTING	Results and interpretations of laboratory tests, imaging studies, and other diagnostic procedures
ASSESSMENT	Clinician's assessment of patient's health
PLAN	Clinician-recommended medical treatments, lifestyle adjustments, and further appointments

PHYSICAL_SOAP and BEHAVIORAL_SOAP sections

Section	Description
Subjective	The patient's goals, experiences, and existing and past issues
Objective	Data and facts about the patient
Assessment	The clinician's diagnosis of the patient's situation
Plan	Clinician-recommended next steps in treatment, including future interventions and referrals

Note

PHYSICAL_SOAP is optimized for physical health documentation. BEHAVIORAL_SOAP is optimized for behavioral health documentation. Both share the same section structure.

GIRPP sections

Section	Description
Goal	The identified problem, challenge, or behavior to address through treatment
Intervention	The specific treatment, method, or technique used by the clinician
Response	How the patient responded to the intervention, including participation level and feedback
Progress	The clinician's assessment of movement toward treatment goals
Plan	Clinician-recommended next steps in treatment, including future interventions, homework, and referrals

BIRP sections

Section	Description
Behavior	The problems the patient presents and their response to treatment
Intervention	The specific treatment, method, or technique used by the clinician
Response	How the patient responded to the intervention
Plan	Next steps in treatment

SIRP sections

Section	Description
Situation	The problem the patient presents and their goal for seeking therapy
Intervention	The specific treatment, method, or technique used by the clinician
Response	How the patient responded to the intervention
Plan	Clinician-recommended next steps in treatment

DAP sections

Section	Description
Data	The patient's reasons for seeking treatment and information about the patient
Assessment	The clinician's diagnosis of the patient's situation
Plan	Clinician-recommended next steps in treatment

Custom templates

Ambient documentation uses a two-layer customization model: Base and Output Specification. These two layers are managed in a configuration object, `customTemplate`. The `customTemplate` configuration object contains two parameters: `templateType` sets the Base template and `templateInstructions` contains the Output Specification.

The Base (`templateType`) sets the organization structure of the clinical facts detected during the conversation. The following base templates are supported:

Base	Description	Use case
HISTORY_AND_PHYSICAL	Summaries for key clinical documentation sections	General physical health encounters
BEHAVIORAL_SOAP	Behavioral health focused SOAP format	Behavioral health encounters using SOAP structure
GIRPP	Progress-toward-goals format	Behavioral health — tracking patient progress
BIRP	Behavioral patterns and responses format	Behavioral health — documenting behavioral patterns
SIRP	Situational context of therapy format	Behavioral health — emphasizing situational context
DAP	Simplified clinical documentation format	Brief or focused encounters

The Output Specification object (`templateInstructions`) is organized as an array of instructions, with a `sectionHeader` that defines the section name and `sectionInstructions` that combines instructions and a template for that section.

Customization instructions can include three types of directives:

- **Verbosity instructions** — Control content conciseness or elaboration. Example: "Describe the chief complaint in 1 sentence or less."
- **Template usage instructions** — Direct how the agent handles misalignment between the template and the encounter content. Example: "Follow the template exactly: if requested data is unavailable, write INFORMATION NOT FOUND."
- **Style instructions** — Specify formatting, terminology, and reasoning requirements. Example: "Use numbered problems in the Assessment section."

Templates can be provided as text with placeholders, structured JSON schemas, or example previous notes.

Method	Description	Use when
Text template with placeholders	A template with section headers and placeholder fields (such as <chief_complaint>) that the agent fills from the encounter	You want precise control over section layout and content placement
Structured JSON template	A JSON schema defining fields, nesting, and per-field formatting rules	Your EHR requires structured data output rather than prose
Example previous note	A prior clinical note provided as a reference for the desired format and style	A provider wants notes that match their existing documentation patterns

Note

The service is stateless. To use a previous note as a style reference, your application must include it in the instructions for each session. The agent does not retain provider preferences across sessions.

Example customization instruction

The following example shows a customization instruction for a SOAP note using the `customTemplate` configuration object.

```
{
  "clinicalNoteGenerationSettings": {
    "noteTemplateSettings": {
      "customTemplate": {
        "templateType": "HISTORY_AND_PHYSICAL",
        "templateInstructions": [
          {
            "sectionHeader": "Subjective",
            "sectionInstruction": "You will be generating a SOAP note one section
 at a time, starting with the `S` section. Please use this template when generating
 the `S` section:\n<template>\nSUBJECTIVE:\nChief Complaint: <Brief statement, in
```

```
 patient's own words, if available>\nHistory of Present Illness: <Narrative description
 of current symptoms, onset, duration, quality, severity, timing, context, modifying
 factors, associated symptoms.>\nReview of Systems:\n• Constitutional: <fever, chills,
 weight changes, fatigue>\n• Cardiovascular: <chest pain, palpitations, shortness of
 breath>\n• Respiratory: <cough, dyspnea, wheezing>\n• GI: <nausea, vomiting, diarrhea,
 constipation, abdominal pain>\n• GU: <dysuria, frequency, urgency, hematuria>
\n• Musculoskeletal: <joint pain, muscle weakness, back pain>\n• Neurological:
 <headache, dizziness, numbness, weakness>\n• Psychiatric: <mood changes, anxiety,
 sleep disturbances>\n• All other systems negative unless noted above Past Medical
 History: <List chronic conditions>\nPast Surgical History: <List previous surgeries
 with dates>\nMedications: <Current medications with doses>\nAllergies: <Drug allergies
 and reactions, or NKDA>\nSocial History: <Tobacco, alcohol, drugs, occupation, living
 situation>\nFamily History: <Relevant family medical history>\n</template>"
          }
        ]
      }
    }
  }
}
```

Template best practices

The agent achieves 97.7% average layout adherence with well-designed templates.

- **Define your note structure with named section headers.** List each section of your desired note by name, using a consistent delimiter. The model uses these headers as structural anchors to place content in the correct location.
- **Use descriptive placeholders** that explain what content belongs in each section. For example, `Chief Complaint: <Brief statement in patient's own words, if available>`.
- **Enumerate expected subsections** for multi-part fields. For sections that span multiple categories (such as body systems or problem lists), list them explicitly with representative values to indicate scope.
- **Handle missing information gracefully.** Use phrasing like "if available" or "if applicable" within placeholders to signal that a section can be omitted when the encounter does not produce relevant content.
- **Test templates across visit types.** A template that works for follow-up visits may not suit new patient encounters or wellness exams. Validate your templates against a representative sample of encounters before deploying broadly.

Outputs

Ambient documentation generates three output files:

Transcript file

The transcript file contains turn-by-turn transcription with word-level timestamps. Amazon Connect Health adds participant role detection, labeling each speaker as CLINICIAN or PATIENT. If a conversation has more than one participant in each category, each participant is assigned a number (for example, `CLINICIAN_0`, `CLINICIAN_1`).

Clinical documentation and evidence mapping file

The clinical documentation file contains the structured clinical note generated from the patient-clinician conversation and a section linking each generated statement back to its source in the conversation transcript or patient context input.

The file follows the managed template or customization instructions provided at the start of the session. Each section can contain visit-derived content (from the conversation) and context-derived content (from patient context input). Multiple output formats are supported: prose/free-text and structured JSON depending on template configuration. The evidence mapping section enables clinicians to verify the origin of any AI-generated content. Each mapping entry contains the generated sentence and the source transcript/context reference.

After-visit summary file

The after-visit summary file contains a patient-facing summary written in accessible language for a clinician's review and finalization. It includes a plain-language description of the visit, current medications with dosage and frequency, clinician's follow-up instructions, and action items for the patient.

The summary is generated from the clinical documentation file, not directly from the transcript, ensuring consistency between the clinical note and the patient-facing summary.

For complete API parameter details, request/response schemas, and streaming setup instructions, see the Amazon Connect Health API Reference.

Security in Amazon Connect Health

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from a data center and network architecture that is built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security of the cloud and security in the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Connect Health, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Amazon Connect Health. The following topics show you how to configure Amazon Connect Health to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon Connect Health resources.

Topics

Security topics

- the section called "Data protection" – Learn about encryption, zero-persistence architecture, and PHI handling.

- the section called "Identity and access management" – Learn about IAM policies, service roles, and fine-grained permissions.
- the section called "CloudTrail logs" – Learn about logging Amazon Connect Health API calls with AWS CloudTrail. For more information, see *Monitoring*.
- the section called "Compliance validation" – Learn about HIPAA eligibility and cross-region inference disclosure.
- the section called "Resilience" – Learn about resilience and Availability Zone support.
- the section called "Infrastructure security" – Learn about network isolation and infrastructure protection.

Data protection in Amazon Connect Health

The AWS shared responsibility model applies to data protection in Amazon Connect Health. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Amazon Connect Health or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names might be used for billing or diagnostic logs.

Data handled by Amazon Connect Health

Amazon Connect Health handles a variety of data related to healthcare interactions, including but not limited to the following categories:

- **Patient data** — Patient information retrieved in real time from EHR systems or other external data sources during each interaction.
- **Call transcripts** — Retained in the customer's Amazon Connect instance for the customer-configured retention period.
- **Contact Trace Records (CTRs)** — Retained in Amazon Connect for up to 24 months.
- **CloudWatch Logs** — Retained per the customer-configured log group retention policy.
- **Domain and agent configurations** — Resources and configurations including domains, agents, agent versions, and integrations.

Encryption at rest

All data stored by Amazon Connect Health is encrypted at rest using AWS Key Management Service (AWS KMS). Contact data classified as PII, or data that represents customer content being stored by Amazon Connect Health, is encrypted at rest using AWS KMS encryption keys. For more information about AWS KMS keys, see What is AWS Key Management Service? in the *AWS Key Management Service Developer Guide*.

Customers can use AWS managed keys or provide their own customer managed keys for additional control over encryption. AWS KMS charges apply for a customer managed key. For more information about pricing, see AWS KMS pricing.

Encryption in transit

All data exchanged with Amazon Connect Health is protected in transit using industry-standard TLS 1.2 or higher encryption. This includes:

- Communications between the user's web browser and Amazon Connect Health

- Patient calls and agent workspace communications
- EHR API requests and responses
- Integrations with AWS services such as AWS Lambda, Amazon Bedrock, and Amazon Connect

When Amazon Connect Health integrates with AWS services, data is always encrypted in transit using TLS.

PHI handling

Amazon Connect Health does not store Patient Health Info (PHI) outside of an active call session — once a patient interaction ends, no PHI is retained within the service layer, minimizing compliance exposure for healthcare customers.

For Amazon Connect Health patient engagement workflows integrated with Amazon Connect contact center features, customers can rely on Connect's native PHI handling. Amazon Connect provides proactive PHI redaction in chats and transcripts, encrypts data in transit, and manages customer profiles securely. For example, customers can enable Connect Contact Lens sensitive data blocking to prevent PHI from being stored in plain text.

Both services are HIPAA eligible. Together, the two services form a layered approach: Amazon Connect Health handles session-scoped PHI securely at the agentic layer, while Amazon Connect provides PHI protection controls at the contact center infrastructure level.

Cross-Region Inference (CRIS) disclosure

Amazon Connect Health uses Amazon Bedrock foundation models to power AI capabilities. Customers need to be aware of the following:

- **Cross-Region Inference** — Depending on your AWS Region and model availability, inference requests might be processed across AWS Regions to optimize performance and availability. All data transmission occurs within the AWS secure infrastructure with end-to-end encryption. For detailed information about Amazon Bedrock's cross-region inference capabilities and data handling practices, refer to the Amazon Bedrock documentation.
- **Compliance considerations** — Organizations using Amazon Connect Health agents need to review their data residency requirements and ensure alignment with their HIPAA Business Associate Agreement (BAA) with AWS. The service maintains HIPAA eligibility across all supported Regions.

- **Model usage** — The agents use Amazon Bedrock models in accordance with provider license restrictions. All usage complies with the AWS Responsible AI Policy and provider-specific restrictions.

Service improvement and how to opt out

You can opt out of the use of your content for service improvement by contacting the Amazon Connect Health team at amazon-connecthealth-ai-optout@amazon.com.

Identity and access management for Amazon Connect Health

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon Connect Health resources. IAM is an AWS service that you can use with no additional charge.

Audience

How you use AWS Identity and Access Management (IAM) differs, depending on the work that you do in Amazon Connect Health.

Service user — If you use the Amazon Connect Health service to do your job, then your administrator provides you with the credentials and permissions that you need. As you use more Amazon Connect Health features for your work, you might need additional permissions. Understanding how access is managed can help you request the right permissions from your administrator.

Service administrator — If you're in charge of Amazon Connect Health resources at your company, you probably have full access to Amazon Connect Health. It's your job to determine which Amazon Connect Health features and resources your service users should access. You must then submit requests to your IAM administrator to change the permissions of your service users. Review the information on this page to understand the basic concepts of IAM.

IAM administrator — If you're an IAM administrator, you might want to learn details about how you can write policies to manage access to Amazon Connect Health.

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated (signed in to AWS) as the AWS account root user, as an IAM user, or by assuming an IAM role.

For programmatic access, AWS provides an SDK and CLI that cryptographically sign your requests using your credentials. If you don't use AWS tools, you must sign requests yourself. Amazon Connect Health supports Signature Version 4, a protocol for authenticating inbound API requests. For more information about authenticating requests, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

Regardless of the authentication method that you use, you might also be required to provide additional security information. For example, AWS recommends that you use multi-factor authentication (MFA) to increase the security of your account.

AWS account root user — When you create an AWS account, you begin with one sign-in identity that has complete access to all AWS services and resources in the account. This identity is called the AWS account root user. Do not use the root user for your everyday tasks. Safeguard your root user credentials and use them to perform the tasks that only the root user can perform.

IAM users and groups — An IAM user is an identity within your AWS account that has specific permissions for a single person or application. Where possible, we recommend relying on temporary credentials instead of creating IAM users who have long-term credentials such as passwords and access keys.

IAM roles — An IAM role is an identity within your AWS account that has specific permissions. It is similar to an IAM user, but is not associated with a specific person. You can temporarily assume an IAM role in the AWS Management Console by switching roles.

Federated identities — As a best practice, require human users, including users that require administrator access, to use federation with an identity provider to access AWS services by using temporary credentials. A federated identity is a user from your enterprise user directory, a web identity provider, the AWS Directory Service, the Identity Center directory, or any user that accesses AWS services by using credentials provided through an identity source.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy is an object in AWS that, when associated with an identity or resource, defines their

permissions. AWS evaluates these policies when a principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents.

Identity-based policies — Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions.

Resource-based policies — Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM role trust policies and Amazon S3 bucket policies.

Other policy types — AWS supports additional, less-common policy types, including permissions boundaries, service control policies (SCPs), resource control policies (RCPs), and session policies.

How Amazon Connect Health works with IAM

Amazon Connect Health uses IAM identity-based policies to control access to Connect Health operations and resources. Administrators should apply least-privilege principles, granting only the permissions required for each role.

Amazon Connect Health supports the following IAM features:

- **Identity-based policies**: Yes
- **Resource-based policies**: No
- **Policy actions**: Yes (`health-agent:*`)
- **Policy resources**: Yes (domain, agent, integration ARNs)
- **Policy condition keys**: Yes (`aws:RequestTag`, `aws:ResourceTag`, `aws:TagKeys`)
- **ABAC (attribute-based access control)**: Yes

Amazon Connect Health supports attribute-based access control (ABAC) through IAM tags for domain and agent related operations. Resource-level permissions can be applied to restrict access to specific domains, agents, and integrations. There are minimum IAM permissions required for administrators to access Amazon Connect Health through the AWS console.

Read-only console policy:

```
{
```

```
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:ListDomains",
        "health-agent:GetDomain",
        "health-agent:ListIntegrations",
        "health-agent:ListAgents"
      ],
      "Resource": "*"
    }
  ]
}
```

Full access console policy:

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:CreateDomain",
        "health-agent:DeleteDomain",
        "health-agent:ListDomains",
        "health-agent:GetDomain",
        "health-agent:StartPatientInsightsJob",
        "health-agent:GetPatientInsightsJob",
        "health-agent:CreateIntegration",
        "health-agent:GetIntegration",
        "health-agent:UpdateIntegration",
        "health-agent:DeleteIntegration",
        "health-agent:ListIntegrations",
        "health-agent:ListAgents",
        "health-agent:CreateAgent",
        "health-agent:UpdateAgent",
        "health-agent:GetAgent",
        "health-agent:PublishAgent",
        "health-agent:DeleteAgent",
        "healthlake:ReadResource",
        "healthlake:SearchWithGet",
        "healthlake:SearchWithPost",
```

```
        "healthlake:SearchEverything",
        "organizations:DescribeOrganization",
        "organizations:ListAccounts",
        "iam:CreateRole",
        "iam:PutRolePolicy",
        "iam:CreatePolicy",
        "iam:AttachRolePolicy",
        "iam:PassRole",
        "iam:CreatePolicyVersion",
        "iam:DeletePolicyVersion",
        "cloudformation:CreateStack",
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackEvents",
        "cloudformation:GetTemplate",
        "cloudformation:ListStacks",
        "connect:DescribeInstance",
        "connect:ListInstances",
        "connect:AssociatePhoneNumberContactFlow",
        "connect:ListPhoneNumbersV2",
        "kms:ListKeys",
        "kms:DescribeKey",
        "kms:ListAliases",
        "sso:CreateApplication",
        "sso:CreateInstance",
        "sso:ListInstances",
        "sso:PutApplicationAuthenticationMethod",
        "sso:PutApplicationGrant",
        "sso:CreateApplicationAssignment",
        "sso:ListDirectoryAssociations",
        "sso-directory:SearchUsers",
        "sso-directory:CreateUser",
        "s3:putObject",
        "s3:getObject",
        "s3:ListBucket",
        "s3:ListAllMyBuckets"
      ],
      "Resource": "*"
    }
  ]
}
```

Identity-based policy examples for Amazon Connect Health

By default, users and roles don't have permission to create or modify Amazon Connect Health resources. They also can't perform tasks by using the AWS Management Console, AWS CLI, or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Creating policies on the JSON tab in the *IAM User Guide*.

Policy best practices:

- Get started with AWS managed policies and move toward least-privilege permissions.
- Apply least-privilege permissions — When you set permissions with IAM policies, grant only the permissions required to perform a task.
- Use conditions in IAM policies to further restrict access.
- Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions.
- Require multi-factor authentication (MFA).

Example 1: Allow full access to a specific domain

The following policy grants full access to a specific Amazon Connect Health domain and its integrations.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:*"
      ],
      "Resource": [
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>",
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>/integration/*"
      ]
    }
```

```
  ]
}
```

Example 2: Allow users to start and view ambient documentation sessions

The following policy allows users to start and retrieve ambient documentation sessions.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:StartMedicalScribeListeningSession",
        "health-agent:GetMedicalScribeListeningSession"
      ],
      "Resource": "*"
    }
  ]
}
```

Example 3: Read-only access to ambient documentation

The following policy grants read-only access to ambient documentation sessions.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:GetMedicalScribeListeningSession",
        "health-agent:ListMedicalScribeListeningSessions"
      ],
      "Resource": "*"
    }
  ]
}
```

Example 4: Full access to all Connect Health resources

The following policy grants full access to all Amazon Connect Health resources.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "health-agent:*",
      "Resource": "*"
    }
  ]
}
```

Example 5: Read-only access with deny on destructive actions

The following policy grants read-only access to domains and agents while explicitly denying delete operations.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "health-agent:GetDomain",
        "health-agent:ListAgents",
        "health-agent:GetAgent"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Deny",
      "Action": [
        "health-agent:DeleteDomain",
        "health-agent:DeleteAgent"
      ],
      "Resource": "*"
    }
  ]
}
```

Example 6: Allow IAM users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity.

```
{
  "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": "*"
    }
  ]
}
```

Service roles and trust policies

Amazon Connect Health uses an IAM service role to perform actions on your behalf. Each AWS service component that Amazon Connect Health interacts with operates under a dedicated IAM service role with a trust policy scoped to the specific service principal.

The following example shows a service role policy for Amazon Connect Health that grants permissions for domain, integration, and agent management.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Sid": "DescribeUserWithSourceIdentity",
      "Effect": "Allow",
      "Action": [
        "identitystore:DescribeUser"
      ],
      "Resource": "arn:aws:identitystore:::user/${aws:SourceIdentity}"
    },
    {
      "Sid": "DomainReadAccess",
      "Effect": "Allow",
      "Action": [
        "health-agent:GetDomain"
      ],
      "Resource": "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>"
    },
    {
      "Sid": "ListOperations",
      "Effect": "Allow",
      "Action": [
        "health-agent:ListIntegrations",
        "health-agent:ListAgents",
        "health-agent:ListAgentVersions"
      ],
      "Resource": "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>"
    },
    {
      "Sid": "IntegrationManagement",
      "Effect": "Allow",
      "Action": [
        "health-agent:CreateIntegration",
```

```
        "health-agent:GetIntegration",
        "health-agent:UpdateIntegration",
        "health-agent:DeleteIntegration"
      ],
      "Resource": [
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>",
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>/integration/*"
      ]
    },
    {
      "Sid": "AgentManagement",
      "Effect": "Allow",
      "Action": [
        "health-agent:CreateAgent",
        "health-agent:UpdateAgent",
        "health-agent:GetAgent",
        "health-agent:PublishAgent",
        "health-agent:DeleteAgent"
      ],
      "Resource": [
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>",
        "arn:aws:health-agent:<region>:<account-id>:domain/<domain-id>/agent/*"
      ]
    }
  ]
}
```

The following example shows a trust policy that allows the Amazon Connect Health service principal to assume the role.

```
{
  "Version": "2012-10-17" ,
  "Statement": [
    {
      "Sid": "AllowHealthAgentServicePrincipal",
      "Effect": "Allow",
      "Principal": {
        "Service": "health-agent.amazonaws.com"
      },
      "Action": [
        "sts:AssumeRole",
        "sts:SetContext",
        "sts:SetSourceIdentity"
```

```
      ],
      "Condition": {
        "ArnLike": {
          "aws:SourceArn": "arn:aws:health-agent:<region>:<account-id>:domain/<domain-
id>/*"
        },
        "StringEquals": {
          "aws:SourceAccount": "<account-id>"
        }
      }
    }
  ]
}
```

To help prevent the confused deputy problem, we recommend using the `aws:SourceArn` and `aws:SourceAccount` condition keys in the trust policy. The source ARN should be scoped to the specific Amazon Connect Health domain that the role is intended to serve.

Compliance validation for Amazon Connect Health

To learn whether an AWS service is within the scope of specific compliance programs, see AWS Services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations.

HIPAA eligibility

Amazon Connect Health is a HIPAA-eligible AWS service. Customers subject to the Health Insurance Portability and Accountability Act of 1996 (HIPAA) must execute a Business Associate Addendum (BAA) with AWS prior to processing protected health information (PHI).

The zero-persistence architecture of Amazon Connect Health minimizes PHI exposure. However, customers remain responsible for ensuring their configuration meets HIPAA requirements.

Customer responsibilities for compliance

When using Amazon Connect Health, you are responsible for the following:

Data classification

- Identify which data constitutes PHI in your environment.
- Implement appropriate controls based on data sensitivity.
- Document data flows and storage locations.

Access controls

- Implement least-privilege access for all users, including workforce users provisioned through AWS IAM Identity Center.
- Enforce multi-factor authentication (MFA) for administrative access.
- Conduct regular access reviews and remove unused accounts.

Incident response

- Establish procedures for detecting and reporting security incidents.
- Amazon Connect Health provides audit logs through AWS CloudTrail to support incident investigations.
- Customers are responsible for notifying affected individuals per HIPAA breach notification rules.

Risk assessment

- Conduct regular HIPAA Security Rule risk assessments.
- Document risks and mitigation strategies.
- Update security policies based on assessment findings.

Resilience in Amazon Connect Health

The AWS global infrastructure is built around AWS Regions and Availability Zones. Regions provide multiple physically separated and isolated Availability Zones, which are connected through low-latency, high-throughput, and highly redundant networking. With Availability Zones, you

can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

Infrastructure security in Amazon Connect Health

As a managed service, Amazon Connect Health is protected by AWS global network security. For information about AWS security services and how AWS protects infrastructure, see AWS Cloud Security. To design your AWS environment using the best practices for infrastructure security, see Infrastructure Protection in *Security Pillar — AWS Well-Architected Framework*.

You use AWS published API calls to access Amazon Connect Health through the network. Clients must support the following:

- Transport Layer Security (TLS). We require TLS 1.2 and recommend TLS 1.3.
- Cipher suites with perfect forward secrecy (PFS) such as DHE (Ephemeral Diffie-Hellman) or ECDHE (Elliptic Curve Ephemeral Diffie-Hellman). Most modern systems such as Java 7 and later support these modes.

Additionally, requests must be signed by using an access key ID and a secret access key that is associated with an IAM principal. Or you can use the AWS Security Token Service (AWS STS) to generate temporary security credentials to sign requests.

Monitoring Amazon Connect Health

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon Connect Health and your other AWS solutions. AWS provides monitoring tools to watch Amazon Connect Health, report when something is wrong, and take automatic actions when appropriate.

Amazon Connect Health is integrated with AWS CloudTrail for logging and monitoring API activity. For more information, see the section called “CloudTrail logs”.

Logging and monitoring in Amazon Connect Health

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon Connect Health and your other AWS solutions. AWS provides the following monitoring tools to watch Amazon Connect Health, report when something is wrong, and take automatic actions when appropriate.

AWS CloudTrail

Amazon Connect Health is integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service in Amazon Connect Health. CloudTrail captures all API calls for Amazon Connect Health as events. The calls captured include calls from the Amazon Connect Health console and code calls to the Amazon Connect Health API operations.

If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon S3 bucket, including events for Amazon Connect Health. If you don’t configure a trail, you can still view the most recent events in the CloudTrail console in **Event history**. Using the information collected by CloudTrail, you can determine the request that was made to Amazon Connect Health, the IP address from which the request was made, who made the request, when it was made, and additional details.

For more information about CloudTrail, see the AWS CloudTrail User Guide.

The event source for Amazon Connect Health is `health-agent.amazonaws.com`.

Management events

All Amazon Connect Health control plane API calls are logged as management events by default. You don’t need to configure anything to receive management events.

Data events

Amazon Connect Health logs data plane API calls as data events. Data events are not logged by default. To log data events, you must create a trail or event data store and configure it to log data events. For more information about configuring data events, see Logging data events in the *AWS CloudTrail User Guide*.

Understanding Amazon Connect Health log file entries

CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on.

The following example shows a CloudTrail log entry that demonstrates the `CreateSubscription` action.


```
{
  "eventVersion": "1.09",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA123456789EXAMPLE:session-name",
    "arn": "arn:aws:sts::123456789012:assumed-role/ExampleRole/session-name",
    "accountId": "123456789012"
  },
  "eventTime": "2026-03-12T19:54:27Z",
  "eventSource": "health-agent.amazonaws.com",
  "eventName": "CreateSubscription",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "192.0.2.1",
  "userAgent": "aws-cli/2.18.6",
  "requestParameters": {
    "domainId": "dom-EXAMPLE1234567890"
  },
  "responseElements": {
    "subscriptionId": "sub-EXAMPLE1234567890",
    "domainId": "dom-EXAMPLE1234567890",
    "status": "ACTIVE"
  },
  "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
  "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
  "readOnly": false,
  "resources": [
```

```
    {
      "accountId": "123456789012",
      "type": "AWS::HealthAgent::Subscription",
      "ARN": "arn:aws:health-agent:us-west-2:123456789012:domain/dom-EXAMPLE1234567890/
subscription/*"
    }
  ],
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management"
}
```

Quotas for Amazon Connect Health

Amazon Connect Health enforces service quotas that govern how AI agents operate within your deployment. These include limits for agent invocation rates, the number of conversation turns per session, and the number of tokens processed per turn. These quotas are designed to ensure service stability and consistent performance across all customers.

If your operational scale requires higher limits — for example, health systems with high concurrent call volumes — you can submit limit increase requests through AWS Support.

In addition to Amazon Connect Health quotas, you are also subject to the default service quotas of Amazon Connect. If you are deploying at scale, review and request increases for Amazon Connect service quotas as needed.

The following table describes the documentation releases for Amazon Connect Health.

Change	Description	Date
Initial release	Initial release of the Amazon Connect Health User Guide.	March 5, 2026