A

PROJECT REPORT

ON

AWS Serverless Voice Recipe Assistant

Academic Year
2025-2026

Under the Guidance of

Associate Professor. Dr.NARESH VURUKONDA

Submitted by

Akula Srinithya (L024. & 70572200013)

Department of STME, B.Tech Data Science , 3rd Year
SVKM's NMIMS (Deemed-to-be-University)
Mukesh Patel School of Technology Management & Engineering, Hyderabad Campus

Index

1. Introduction
About Project:

The Voice Recipe Assistant is a modern, serverless web application designed to simulate a conversational AI kitchen helper. The system leverages a powerful combination of AWS services to create an intelligent and interactive user experience.

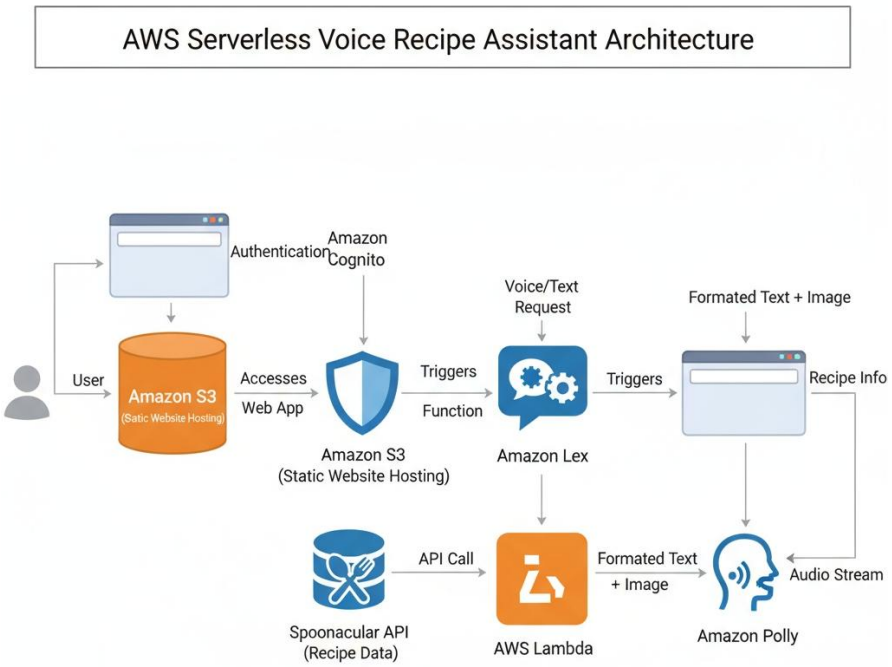
At its core, the project uses Amazon Lex V2 for Natural Language Understanding (NLU), allowing it to interpret user requests for recipes, ingredients, or cooking steps. It uses Amazon Polly to convert the bot's text-based answers into natural, lifelike speech, making the application accessible and hands-free.

The entire application is serverless, with the frontend hosted on Amazon S3, backend logic on AWS Lambda, and security managed by Amazon Cognito and IAM.

OBJECTIVE:

The primary objective of this project is to build a web-based, voice-enabled conversational assistant using AWS services. The goal is to allow users to input text (or in the future, voice) and receive a helpful, context-aware audio-visual response in real-time, ensuring a smooth, scalable, and intelligent serverless experience.

2. Project Architecture & Flowchart
Voice Recipe Assistant Architecture



The user accesses the web app hosted on Amazon S3, authenticates via Amazon Cognito, and sends a voice or text recipe request. Amazon Lex processes the request and invokes an AWS Lambda function, which queries the Spoonacular API for recipe data. Lambda then formats the recipe title, image, and instructions. This formatted response is returned to Lex, which sends it back to the web app for display. At the same time, the text is sent to Amazon Polly, which synthesizes it into audio so users hear the recipe aloud.

The entire application is built on a 100% serverless "event-driven" architecture. This model uses managed AWS services and a third-party API (Spoonacular) to handle all functionality, from the user interface to the AI logic and data retrieval.

Workflow Description (Features in Action)
The data flows through the system in a secure and logical sequence:

1.User Interface (Amazon S3): The user opens the web application in their browser. This static website (HTML, CSS, and JavaScript) is loaded from an Amazon S3 bucket configured for static website hosting.

2.Authentication (Amazon Cognito): As the page loads, the JavaScript application communicates with an Amazon Cognito Identity Pool. Cognito provides the application with secure, temporary AWS credentials, allowing the browser to directly (and safely) access specific AWS services, namely Amazon Lex and Amazon Polly.

3.Natural Language Understanding (Amazon Lex V2):

The user types a request, such as "Find me a recipe for chicken pasta," and hits "Send."

The frontend JavaScript sends this text directly to the Amazon Lex V2 bot.

Lex processes this text, identifies the user's Intent (e.g., FindRecipe), and extracts the key Slots (e.g., ingredient: "chicken", dish: "pasta").

4.Backend Fulfillment (AWS Lambda & Spoonacular):

The Lex bot is configured to use an AWS Lambda function for fulfillment.

Lex triggers the Lambda function, passing it the FindRecipe intent and the "chicken pasta" slots.

The Lambda function's code then makes a secure HTTP request to the Spoonacular API. This request includes the recipe query and a private API key (securely stored as a Lambda environment variable).

5.Data Retrieval (Spoonacular API): The Spoonacular API receives the request, searches its vast database, and returns the recipe data (including title, image URL, and instructions) to the Lambda function in JSON format.

6.Response Processing (AWS Lambda): The Lambda function parses the JSON from Spoonacular. It extracts the most relevant recipe title and image URL and formats a user-friendly text response (e.g., "I found a great recipe for Chicken Alfredo Pasta!").

7.Response to Frontend (Amazon Lex): Lambda passes this formatted text and the recipe image URL back to Lex. Lex, in turn, relays this complete response package to the frontend JavaScript in the browser.

8.Render Text & Images (S3/Browser): The JavaScript code receives the response from Lex. It instantly updates the chat window to display the bot's text message and uses the image URL to display the recipe's picture.

9.Speech Synthesis (Amazon Polly):

Simultaneously, the JavaScript takes only the text part of the bot's response ("I found a great recipe...") and sends it to the Amazon Polly service.

Polly synthesizes this text into a natural-sounding, lifelike MP3 audio stream.

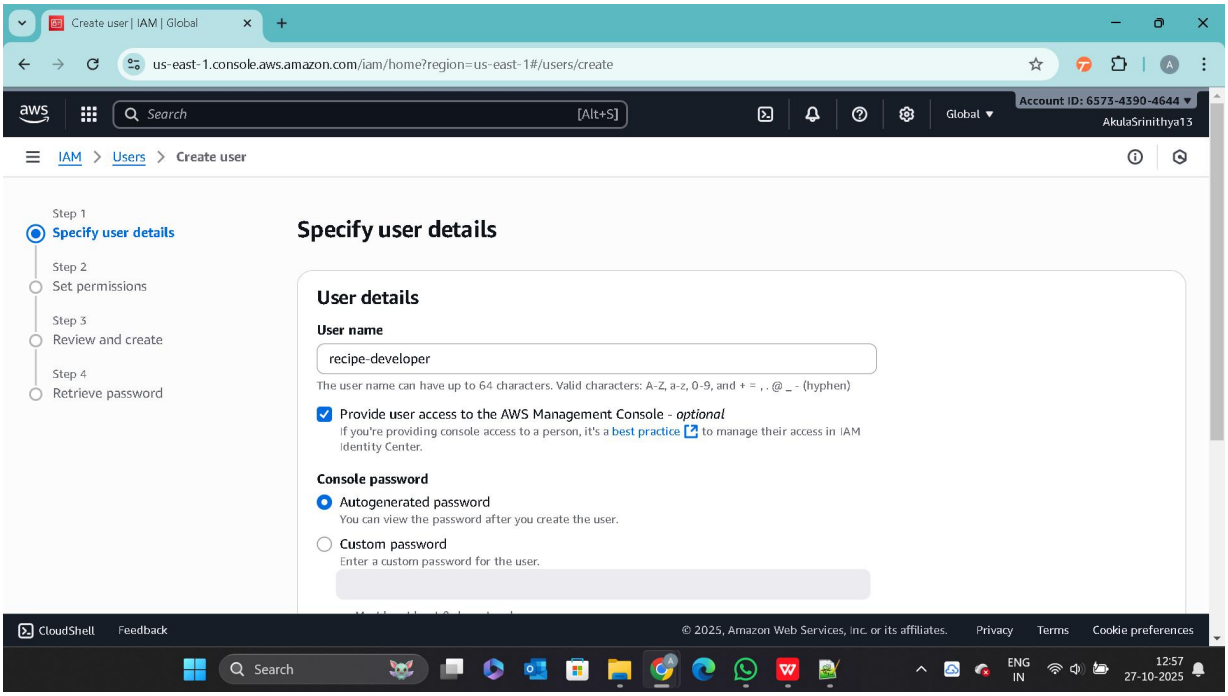
10.**Play Audio (Browser):** Polly streams the audio back to the browser, which plays it immediately, giving the user both a visual and audible response

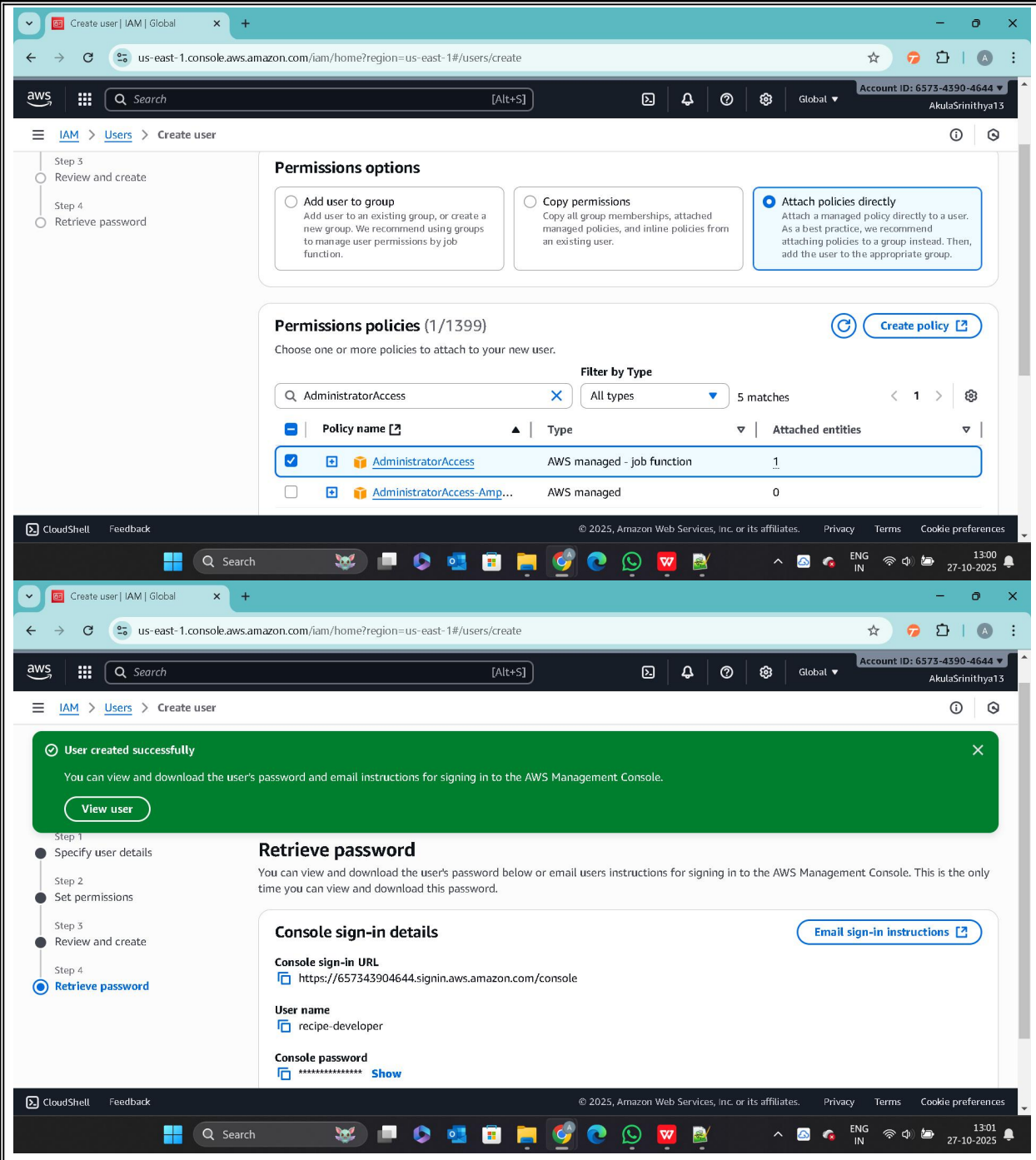
3.List of Cloud Services & APIs Used

This project integrates several key technologies to function:

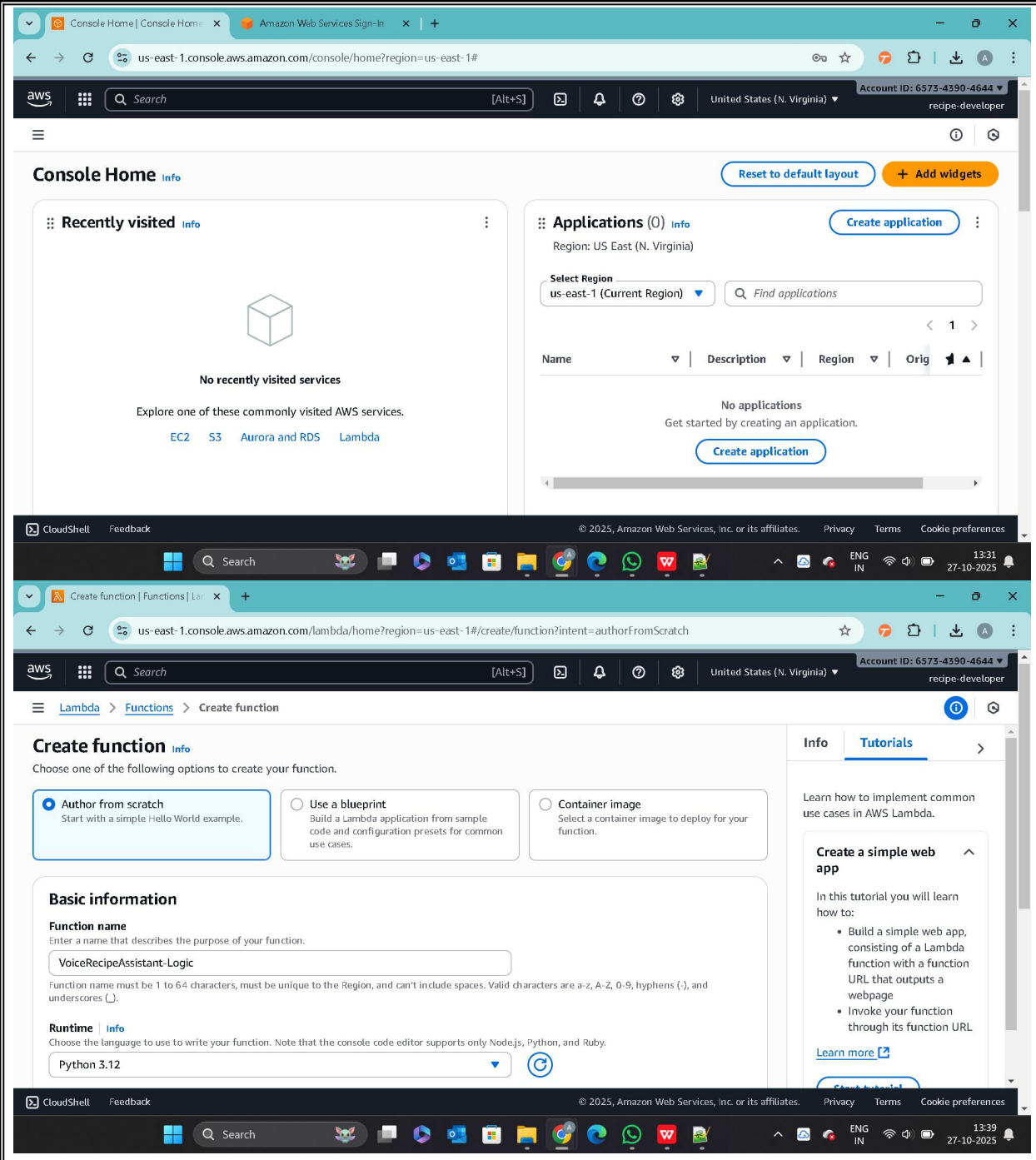
Sr. No.	Name of Service / API	Specification	Purpose (Usage in Project)
1	**Amazon Lex V2**	AWS AI Service	Provides Natural Language Understanding (NLU) to identify user intents and slots; manages the conversational flow.
2	**Amazon Polly**	AWS AI Service	Provides Text-to-Speech (TTS) synthesis, converting the bot's text replies into a playable audio voice.
3	**AWS Lambda**	Serverless Compute	The backend "brain." It runs the code to fulfill the Lex intent, call the Spoonacular API, and process the results.
4	**Spoonacular API**	3rd Party REST API	The project's data source. Provides all recipe data, including titles, instructions, and image URLs, via HTTP requests.
5	**Amazon S3**	Cloud Storage	Hosts the static website (HTML, CSS, JavaScript files) and makes it accessible to the public internet.
6	**Amazon Cognito**	Security & Identity	Provides secure, temporary AWS credentials to the frontend web application so it can access Lex and Polly without exposing secret keys.
7	**IAM Roles**	Security	Grants specific permissions for AWS services to communicate with each other (e.g., allowing Lex to invoke Lambda, allowing Lambda to write logs).

4.Step-by-Step Execution with screenshots in detailed

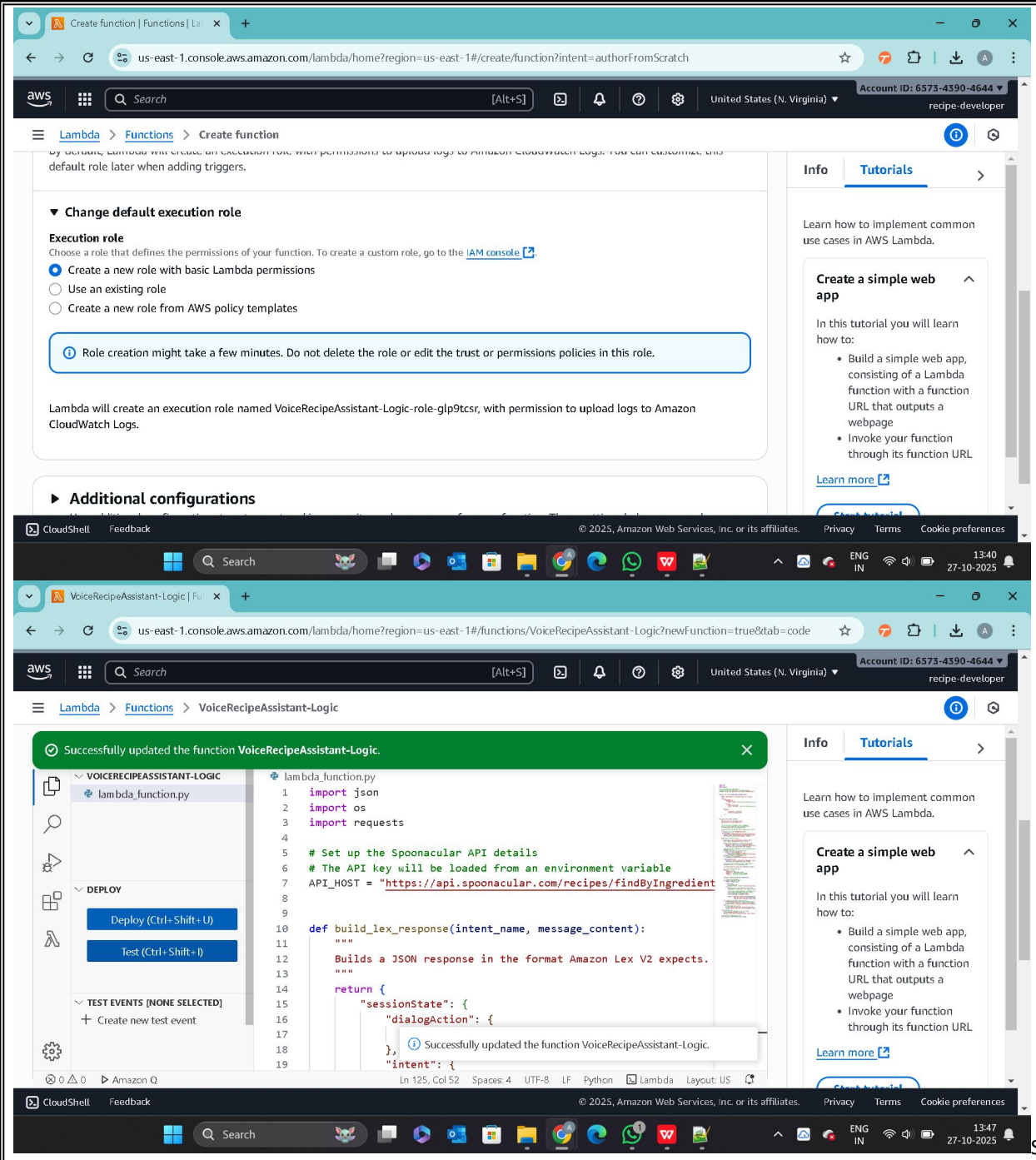



Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create
Search [Alt+S]
Global
Account ID: 6573-4390-4644
AkulaSrinithya13
IAM > Users > Create user
Step 3
Review and create
Step 4
Retrieve password
Permissions options
Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.
Permissions policies (1/1399)
Create policy
Choose one or more policies to attach to your new user.
Filter by Type
AdministratorAccess
All types
5 matches
Policy name
Type
Attached entities
AdministratorAccess
AWS managed - job function
1
AdministratorAccess-Amp...
AWS managed
0
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
13:00
27-10-2025

Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/create
Search [Alt+S]
Global
Account ID: 6573-4390-4644
AkulaSrinithya13
IAM > Users > Create user
User created successfully
You can view and download the user's password and email instructions for signing in to the AWS Management Console.
View user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
Retrieve password
You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.
Console sign-in details
Email sign-in instructions
Console sign-in URL
https://657343904644.signin.aws.amazon.com/console
User name
recipe-developer
Console password
********** Show
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
13:01
27-10-2025


Console Home | Console Home
Amazon Web Services Sign-In
us-east-1.console.aws.amazon.com/console/home?region=us-east-1#
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Console Home Info
Reset to default layout
+ Add widgets
Recently visited Info
No recently visited services
Explore one of these commonly visited AWS services.
EC2
S3
Aurora and RDS
Lambda
Applications (0) Info
Create application
Region: US East (N. Virginia)
Select Region
us-east-1 (Current Region)
Find applications
Name
Description
Region
Orig
No applications
Get started by creating an application.
Create application
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
13:31
27-10-2025

Create function | Functions | La
us-east-1.console.aws.amazon.com/lambda/home?region=us-east-1#/create/function?intent=authorFromScratch
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Lambda
Functions
Create function
Create function Info
Choose one of the following options to create your function.
Author from scratch
Start with a simple Hello World example.
Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.
Container image
Select a container image to deploy for your function.
Basic information
Function name
Enter a name that describes the purpose of your function.
VoiceRecipeAssistant-Logic
Function name must be 1 to 64 characters, must be unique to the Region, and can't include spaces. Valid characters are a-z, A-Z, 0-9, hyphens (-), and underscores (_).
Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12
Info
Tutorials
Learn how to implement common use cases in AWS Lambda.
Create a simple web app
In this tutorial you will learn how to:
Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
Invoke your function through its function URL
Learn more
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
13:39
27-10-2025



Securely Add Your Spoonacular API Key

We must never paste the API key directly into the code. We use Environment Variables, which is the secure, professional way.

Go to the Configuration Tab:

On your Lambda function's page, click the "Configuration" tab (it's next to the "Code" tab you are on).

Open Environment Variables:

On the left-hand menu, click "Environment variables".

Add Your Key:

Click the "Edit" button.

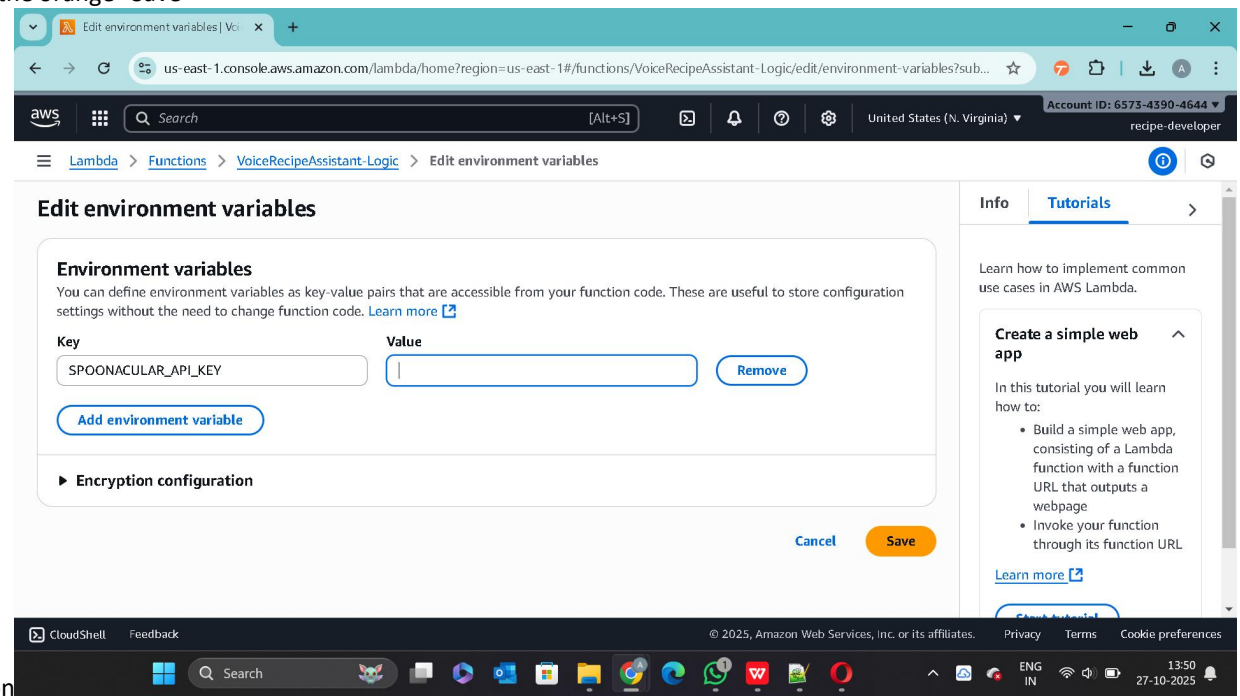
Click "Add environment variable".

You will see two boxes, "Key" and "Value".

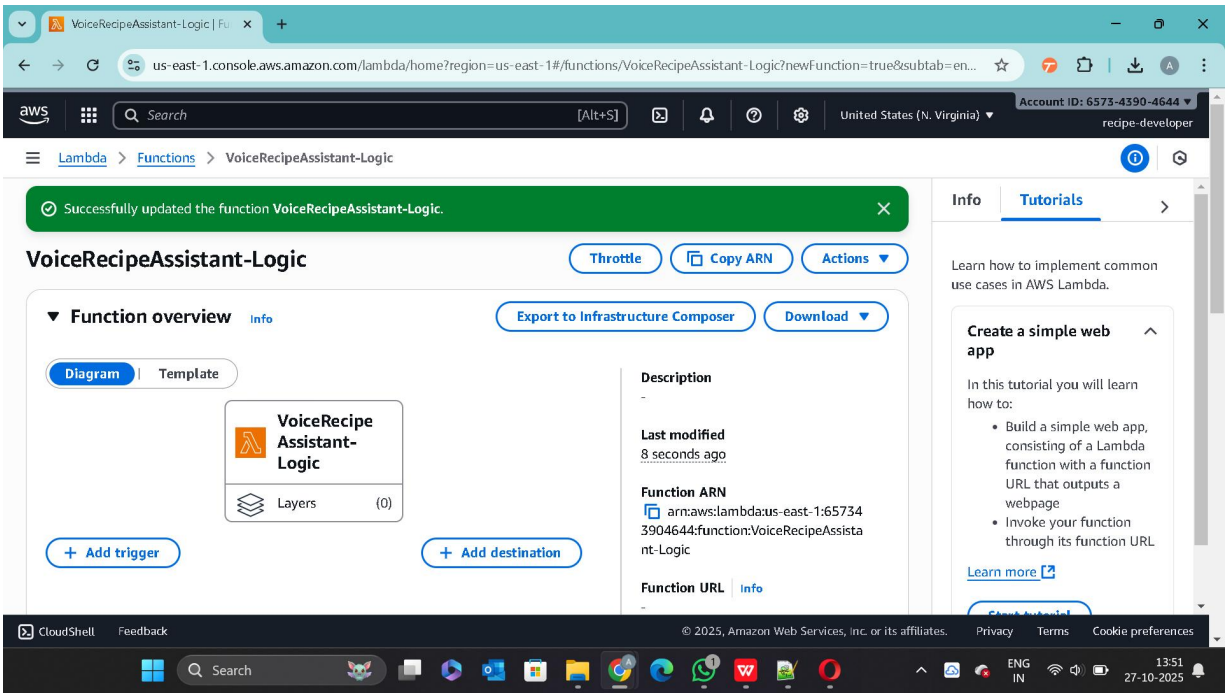
Key: SPOONACULAR_API_KEY (You must type this exactly as shown, all caps. Our code looks for this exact name.)

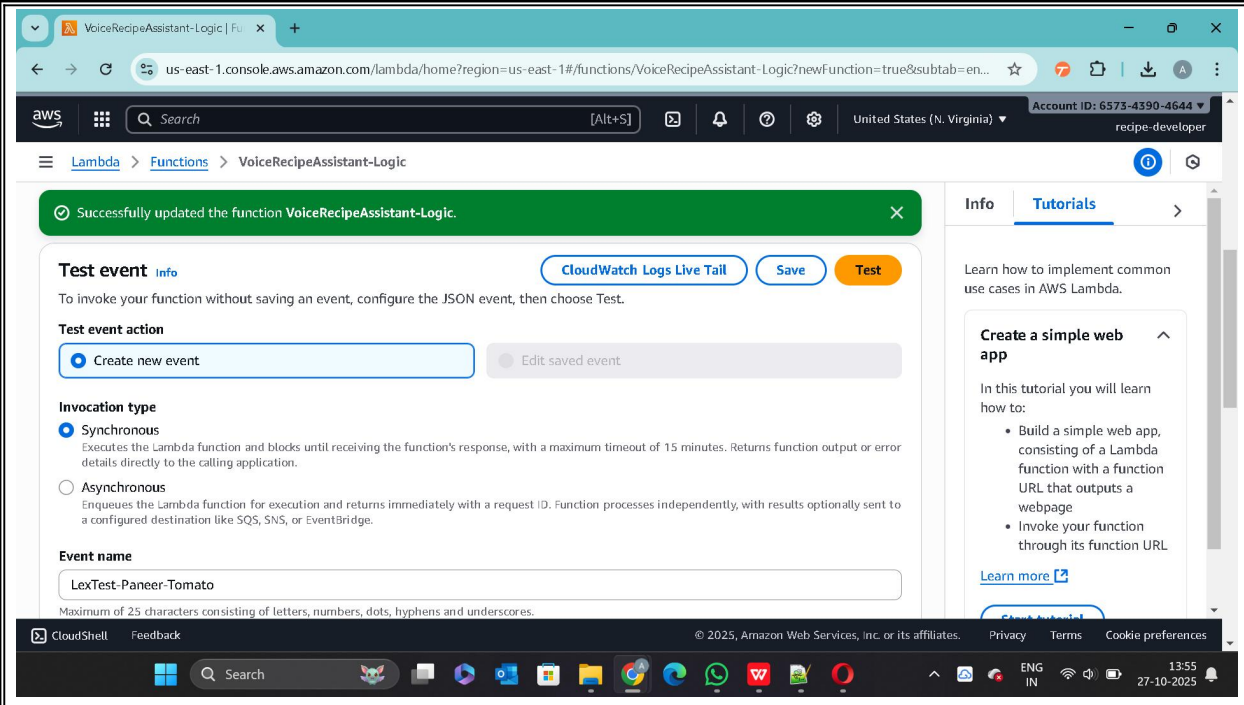
Value: Paste your Spoonacular API key (the long string you saved from their website).

Click the orange "Save"



button





Event JSON code:

```
{
  "sessionState": {
    "intent": {
      "name": "SearchRecipes",
      "slots": {
        "Ingredient": {
          "values": [
            {
              "value": {
                "originalValue": "paneer",
                "resolvedValues": [
                  "paneer"
                ],
                "interpretedValue": "paneer"
              }
            },
            {
              "value": {
                "originalValue": "tomato",
                "resolvedValues": [
                  "tomato"
                ],
                "interpretedValue": "tomato"
              }
            }
          ]
        }
      },
      "state": "ReadyForFulfillment"
    },
    "invocationSource": "FulfillmentCodeHook"
  }
}
```

Lambda code:

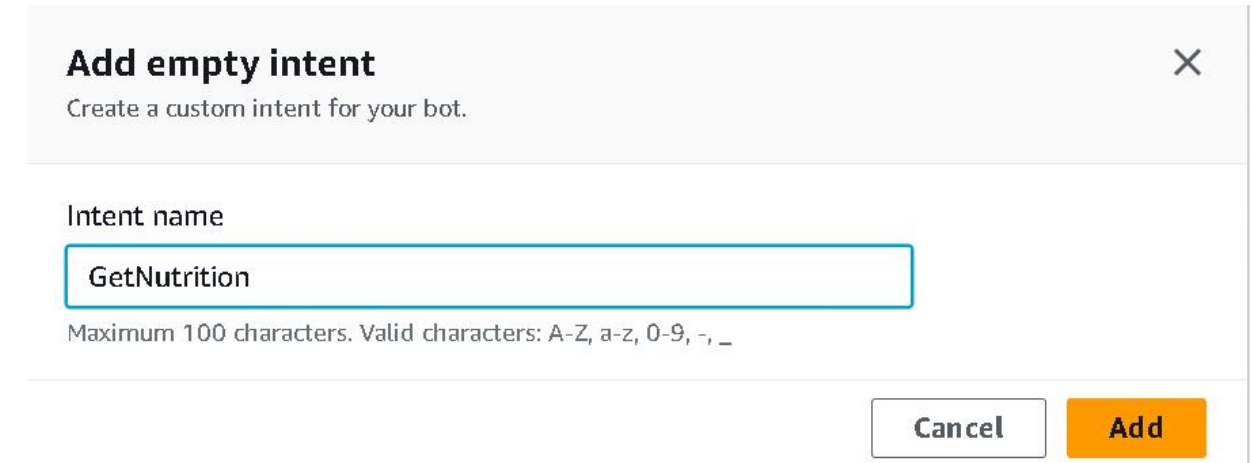
```
import json
import os
import urllib.request
import urllib.parse
from urllib.error import HTTPError
import boto3
```

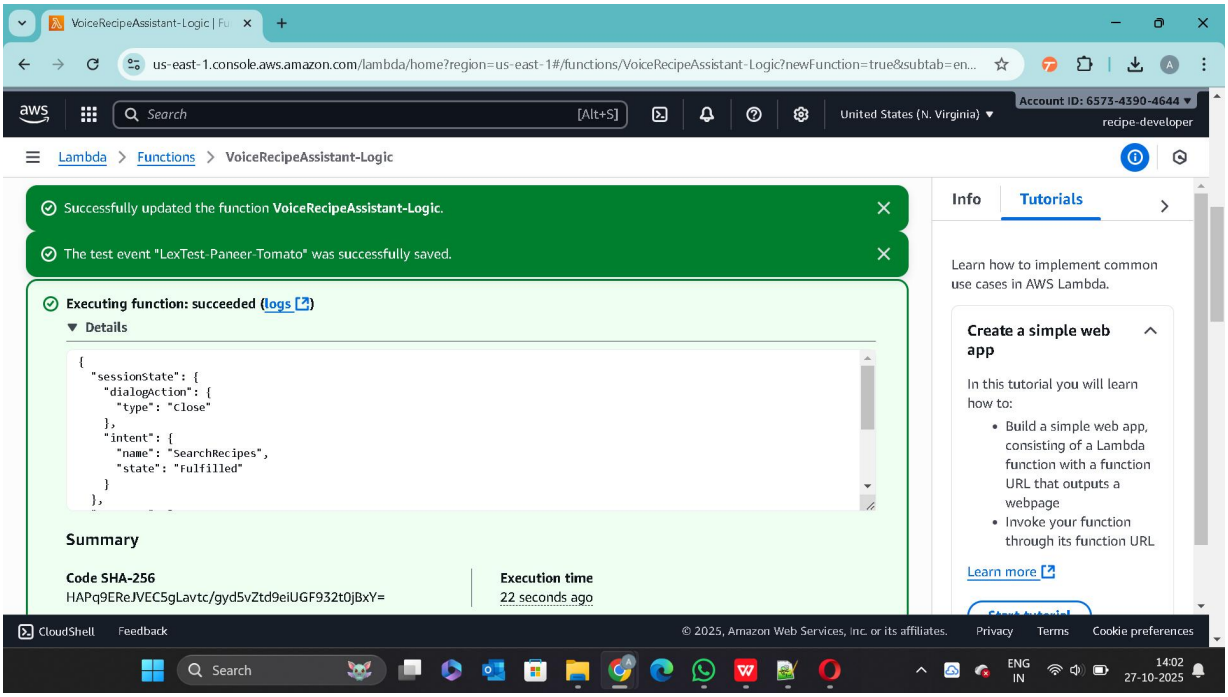
```
import re
import html # For escaping

# --- Initialize Services ---
try:
    dynamodb = boto3.resource('dynamodb')
    USER_TABLE_NAME = 'VoiceRecipeAssistant-Users'
    user_table = dynamodb.Table(USER_TABLE_NAME)
    print("DynamoDB connected.")
except Exception as e:
    print(f"ERROR connecting DynamoDB: {e}"); user_table = None

API_HOST = "https://api.spoonacular.com"
SEARCH_PATH = "/recipes/findByIngredients"
INSTRUCTIONS_PATH = "/recipes/{id}/analyzedInstructions"
NUTRITION_PATH = "/recipes/{id}/nutritionWidget.json"

NON_VEGAN_KEYWORDS = ['beef', 'pork', 'lamb', 'chicken', 'turkey', 'fish', 'salmon', 'tuna', 'shrimp', 'cheese', 'milk', 'yogurt',
'butter', 'cream', 'paneer', 'egg', 'honey']
NON_VEGAN_PATTERN = re.compile(r'\b(?:' + '|'.join(NON_VEGAN_KEYWORDS) + r')\b', re.IGNORECASE)

def escape_ssml(text):
    """ Escapes characters invalid in SSML using html.escape """
    return html.escape(text or "", quote=True)

# ================= ROUTER =====================
def lambda_handler(event, context):
    print(f"Event: {json.dumps(event)}")
    # Safely get intent name, handle potential missing keys
    intent_name = event.get('sessionState', {}).get('intent', {}).get('name', 'FallbackIntent')
    print(f"Intent: {intent_name}")

    if intent_name == 'UpdateProfile': return handle_update_profile(event)
    elif intent_name == 'SearchRecipes': return handle_search_recipes(event)
    elif intent_name == 'StartCooking': return handle_start_cooking(event)
    elif intent_name == 'NextStep': return handle_next_step(event)
    elif intent_name == 'GetNutrition': return handle_get_nutrition(event)
    else: # Default to FallbackIntent
        ssml_msg = f"<speak>Sorry, I'm not sure how to handle that command.</speak>"
        return build_lex_response({},'FallbackIntent', {"ssmlMessage": ssml_msg}, "SSML")


# ================= SEARCH RECIPES ==================
def handle_search_recipes(event):
    print("Handler: search_recipes")
    try: API_KEY = os.environ['SPOONACULAR_API_KEY']
    except KeyError: return build_lex_response({}, "SearchRecipes", {"ssmlMessage": "<speak>Assistant not configured: Missing API
key.</speak>"}, "SSML")

    session_attrs = event.get('sessionState', {}).get('sessionAttributes', {})
    user_id = event.get('sessionId', 'test-user'); user_profile = get_user_profile(user_id)
    intent_name = event['sessionState']['intent']['name']; slots = event['sessionState']['intent']['slots']

    # --- Robust Ingredient Extraction ---
    ingredients = []
    ingredient_slot = slots.get('Ingredient') # Get the slot safely
    if ingredient_slot:
        if 'values' in ingredient_slot and ingredient_slot['values']: # Check if values list exists and is not empty
            ingredients.extend(v['value']['interpretedValue'] for v in ingredient_slot['values'] if v.get('value') and
v['value'].get('interpretedValue'))
        elif 'value' in ingredient_slot and ingredient_slot['value'] and ingredient_slot['value'].get('interpretedValue'): # Check if
single value exists
             ingredients.append(ingredient_slot['value']['interpretedValue'])

    # --- Check for ingredients AFTER attempting extraction ---
    if not ingredients:
        print("No valid ingredients found in slots.")
        return build_lex_response(session_attrs, intent_name, {"ssmlMessage": "<speak>I didn't catch any valid ingredients. Please try
again.</speak>"}, "SSML")
    # --- End Check ---

    ingredients_str = ",".join(ingredients)
    print(f"Found ingredients: {ingredients_str}")

    params = {"apiKey": API_KEY, "ingredients": ingredients_str, "number": 10, "ranking": 1}
    user_diet = user_profile.get('diet'); allergies = set(user_profile.get('allergies', [])); is_vegan = False
    if user_diet:
        if user_diet.lower() == 'vegan': is_vegan = True; allergies.update(["dairy", "eggs"])
```

```
        else: params['diet'] = user_diet
    if allergies: params['intolerances'] = ",".join(list(allergies))
    url = f"{API_HOST}{SEARCH_PATH}?{urllib.parse.urlencode(params)}"
    print(f"API Call: {url}")
    try:
        req = urllib.request.Request(url);
        with urllib.request.urlopen(req) as resp: api_recipes = json.loads(resp.read().decode('utf-8'))
        recipes = [r for r in api_recipes if not (is_vegan and NON_VEGAN_PATTERN.search(r.get('title','')))] if is_vegan else
api_recipes
        if not recipes:
            filter_msg_text = f" for your {user_diet} diet" if user_diet else ""; filter_msg_text += f" avoiding
{','.join(list(allergies))}" if allergies else ""
            plain_msg = f"Sorry, I couldn't find recipes matching {ingredients_str}{filter_msg_text}."
            ssml_msg = f"<speak>{escape_ssml(plain_msg)}</speak>"
            msg = {"ssmlMessage": ssml_msg}; session_attrs = {}
        else:
            top_recipe = recipes[0]; recipe_id, title, img = str(top_recipe['id']), top_recipe['title'], top_recipe.get('image', '')
            session_attrs.update({'currentRecipeId': recipe_id, 'currentRecipeTitle': title}); session_attrs.pop('cookingSteps', None);
session_attrs.pop('currentStep', None)
            filter_msg_text = f" for your {user_diet} diet" if user_diet else ""; filter_msg_text += f" avoiding
{','.join(list(allergies))}" if allergies else ""
            plain_msg = f"Success! I found {len(recipes[:1])} valid recipe(s){filter_msg_text}. The top result is {title}. You can ask
me to 'start cooking' or 'get nutrition'."
            ssml_msg = f"<speak>{escape_ssml(plain_msg)}</speak>"
            msg = {"ssmlMessage": ssml_msg, "recipeInfo": {"title": title, "imageUrl": img}}
    except HTTPError as e:
        print(f"API Error: {e.code}");
        error_detail = ""
        try: # Try to read error body for more info
           error_body = e.read().decode('utf-8')
           print(f"API Error Body: {error_body}")
           if e.code == 401: error_detail = " Check the API key."
           elif e.code == 402: error_detail = " The API quota might be exceeded."
        except Exception: pass # Ignore if reading body fails
        msg = {"ssmlMessage": f"<speak>Sorry, there was an error searching for recipes.{error_detail}</speak>"};
        session_attrs = {}
    except Exception as e:
        print(f"Search Exception: {str(e)}")
        msg = {"ssmlMessage": "<speak>Sorry, an unexpected problem occurred while searching.</speak>"};
        session_attrs = {}

    return build_lex_response(session_attrs, intent_name, msg, "SSML", "Close") # Send SSML


# ================ START COOKING ==================
def handle_start_cooking(event):
    print("Handler: start_cooking")
    session_attrs = event.get('sessionState', {}).get('sessionAttributes', {})
    recipe_id = session_attrs.get('currentRecipeId'); title = session_attrs.get('currentRecipeTitle', 'recipe')
    if not recipe_id: return build_lex_response(session_attrs, "StartCooking", {"ssmlMessage": "<speak>No recipe loaded.</speak>"},
"SSML")
    try:
        API_KEY = os.environ['SPOONACULAR_API_KEY']; url = f"{API_HOST}{INSTRUCTIONS_PATH.format(id=recipe_id)}?apiKey={API_KEY}"
        print(f"Fetching instructions: {url}"); req = urllib.request.Request(url)
        with urllib.request.urlopen(req) as resp: instructions = json.loads(resp.read().decode('utf-8'))
        if not instructions or not isinstance(instructions, list) or not instructions[0].get('steps'): return
build_lex_response(session_attrs, "StartCooking", {"ssmlMessage": f"<speak>Couldn't find steps for {escape_ssml(title)}.</speak>"},
"SSML")
        steps_data = instructions[0]['steps']
        if not steps_data: return build_lex_response(session_attrs, "StartCooking", {"ssmlMessage": f"<speak>No steps listed for
{escape_ssml(title)}.</speak>"}, "SSML")
        steps = [s['step'] for s in steps_data]; session_attrs.update({'cookingSteps': json.dumps(steps), 'currentStep': "0"})
        safe_title = escape_ssml(title); safe_step = escape_ssml(steps[0])
        ssml = f"<speak>OK, let's cook {safe_title}. <break time='500ms'/> Step 1: {safe_step} <break time='1s'/> Say 'next'.</speak>"
        return build_lex_response(session_attrs, "StartCooking", {"ssmlMessage": ssml}, "SSML", "ElicitIntent")
    except HTTPError as e: print(f"API Error instructions: {e.code}"); return build_lex_response(session_attrs, "StartCooking",
{"ssmlMessage": "<speak>Error getting instructions from service.</speak>"}, "SSML")
    except Exception as e: print(f"Parse Error instructions: {str(e)}"); return build_lex_response(session_attrs, "StartCooking",
{"ssmlMessage": "<speak>Error processing instructions.</speak>"}, "SSML")

# ================ NEXT STEP ======================
def handle_next_step(event):
    print("Handler: next_step")
    session_attrs = event.get('sessionState', {}).get('sessionAttributes', {})
    if 'cookingSteps' not in session_attrs or 'currentStep' not in session_attrs: return build_lex_response(session_attrs, "NextStep",
{"ssmlMessage": "<speak>No steps loaded.</speak>"}, "SSML")
    try:
        steps = json.loads(session_attrs['cookingSteps']); current_idx = int(session_attrs['currentStep']); next_idx = current_idx + 1
        if next_idx >= len(steps):
            session_attrs = {}; ssml = "<speak>You're all done! Enjoy.</speak>"
```

```
            return build_lex_response(session_attrs, "NextStep", {"ssmlMessage": ssml}, "SSML", "Close")
        else:
            session_attrs['currentStep'] = str(next_idx)
            safe_step = escape_ssml(steps[next_idx])
            ssml = f"<speak>Step {next_idx + 1}: {safe_step} <break time='1s'/> Say 'next'.</speak>"
            return build_lex_response(session_attrs, "NextStep", {"ssmlMessage": ssml}, "SSML", "ElicitIntent")
    except Exception as e: print(f"Error in next_step: {str(e)}"); return build_lex_response(session_attrs, "NextStep", {"ssmlMessage":
"<speak>Sorry, I lost my place.</speak>"}, "SSML")

# ================ UPDATE PROFILE ==================
def handle_update_profile(event):
    print("Handler: update_profile")
    session_attrs = event.get('sessionState', {}).get('sessionAttributes', {});
    if not user_table: return build_lex_response(session_attrs, "UpdateProfile", {"ssmlMessage": "<speak>Can't connect user
database.</speak>"}, "SSML")
    intent_name, slots, user_id = event['sessionState']['intent']['name'], event['sessionState']['intent']['slots'],
event.get('sessionId')
    if not user_id: return build_lex_response(session_attrs, intent_name, {"ssmlMessage": "<speak>Can't find session ID.</speak>"},
"SSML")
    diet_slot, allergy_slot = slots.get('Diet'), slots.get('Allergy'); parts, vals, names, saved = [], {}, {}, []
    current_allergies = set(get_user_profile(user_id).get('allergies', [])); new_allergies = set()
    if diet_slot and diet_slot.get('value'): user_diet = diet_slot['value']['interpretedValue']; parts.append("#d = :d"); names['#d'],
vals[':d'] = 'diet', user_diet; saved.append(f"diet as {user_diet}")
    if allergy_slot and allergy_slot.get('values'): new_allergies.update(v['value']['interpretedValue'] for v in allergy_slot['values']
if v.get('value') and v['value'].get('interpretedValue'))
    elif allergy_slot and allergy_slot.get('value') and allergy_slot['value'].get('interpretedValue'):
new_allergies.add(allergy_slot['value']['interpretedValue'])
    if new_allergies: combined = current_allergies.union(new_allergies); parts.append("#a = :a"); names['#a'], vals[':a'] = 'allergies',
list(combined); saved.append(f"allergies as {', '.join(vals[':a'])}")
    if not saved: return build_lex_response(session_attrs, intent_name, {"ssmlMessage": "<speak>Didn't catch what profile information
to save.</speak>"}, "SSML")
    try:
        user_table.update_item(Key={'UserId': user_id}, UpdateExpression="SET " + ", ".join(parts), ExpressionAttributeNames=names,
ExpressionAttributeValues=vals)
        plain_msg = f"OK! Updated profile with { ' and '.join(saved) }."
    except Exception as e: print(f"DB Error: {str(e)}"); plain_msg = "Problem saving profile update."
    ssml_msg = f"<speak>{escape_ssml(plain_msg)}</speak>"
    return build_lex_response(session_attrs, intent_name, {"ssmlMessage": ssml_msg}, "SSML") # Send SSML


# ================ GET NUTRITION ==================
def handle_get_nutrition(event):
    print("Handler: get_nutrition")
    session_attrs = event.get('sessionState', {}).get('sessionAttributes', {}); intent_name = event['sessionState']['intent']['name']
    recipe_id = session_attrs.get('currentRecipeId'); title = session_attrs.get('currentRecipeTitle', 'the recipe')
    if not recipe_id: return build_lex_response(session_attrs, intent_name, {"ssmlMessage": "<speak>No recipe loaded.</speak>"}, "SSML")
    try:
        API_KEY = os.environ['SPOONACULAR_API_KEY']; url = f"{API_HOST}{NUTRITION_PATH.format(id=recipe_id)}?apiKey={API_KEY}"
        print(f"Fetching nutrition: {url}"); req = urllib.request.Request(url)
        with urllib.request.urlopen(req) as resp: nutrition = json.loads(resp.read().decode('utf-8'))
        cal, prot, fat, carb = nutrition.get('calories', 'N/A'), nutrition.get('protein', 'N/A'), nutrition.get('fat', 'N/A'),
nutrition.get('carbs', 'N/A')
        plain_msg = f"Nutrition for {title} (per serving): Calories are {cal}, Protein is {prot}, Fat is {fat}, and Carbohydrates are
{carb}."
        ssml_msg = f"<speak>{escape_ssml(plain_msg)}</speak>"
        return build_lex_response(session_attrs, intent_name, {"ssmlMessage": ssml_msg}, "SSML", "ElicitIntent")
    except HTTPError as e: print(f"API Error nutrition: {e.code}"); return build_lex_response(session_attrs, intent_name,
{"ssmlMessage": "<speak>Couldn't get nutrition info.</speak>"}, "SSML")
    except Exception as e: print(f"Parse Error nutrition: {str(e)}"); return build_lex_response(session_attrs, intent_name,
{"ssmlMessage": "<speak>Error getting nutrition info.</speak>"}, "SSML")

# ================ HELPERS ======================
def build_lex_response(session_attrs, intent_name, msg_content, msg_type_hint="PlainText", dialog_action_type="Close"):
    final_content, content_type = "", "PlainText"
    if isinstance(msg_content, dict) and "ssmlMessage" in msg_content: final_content, content_type = msg_content["ssmlMessage"], "SSML"
    elif isinstance(msg_content, dict) and "plainTextMessage" in msg_content: final_content = msg_content["plainTextMessage"];
content_type="PlainText" # Always use PlainText if ssmlMessage not present
    elif isinstance(msg_content, str): final_content, content_type = msg_content, "PlainText" # Assume string is PlainText
    else: final_content = "<speak>Processing complete.</speak>"; content_type="SSML" # Default SSML fallback

    # If the FINAL decision is SSML, ensure it's wrapped
    if content_type == "SSML" and not final_content.strip().startswith('<speak>'):
        final_content = f"<speak>{escape_ssml(final_content)}</speak>"

    intent_state = "InProgress" if dialog_action_type != "Close" else "Fulfilled"; current_intent = {"name": intent_name, "state":
intent_state, "slots": {}}
    dialog_action = {"type": dialog_action_type}
    # For Close action, Lex V2 requires intent details within dialogAction
    if dialog_action_type == "Close": dialog_action["intent"] = {"name": intent_name, "state": "Fulfilled", "slots": {}}
```

```
    safe_session_attrs = session_attrs or {}
    response = {"sessionState": {"sessionAttributes": safe_session_attrs, "dialogAction": dialog_action, "intent": current_intent},
"messages": [{"contentType": content_type, "content": final_content}]}
    if isinstance(msg_content, dict) and "recipeInfo" in msg_content:
        response['sessionState']['sessionAttributes']['appContext'] = json.dumps({"recipeInfo": msg_content["recipeInfo"]})
    print(f"Final sessionAttributes: {json.dumps(response['sessionState']['sessionAttributes'])}")
    print(f"Sending message type: {content_type}")
    return response

def get_user_profile(user_id):
    if not user_table: return {'allergies': []}
    try:
        resp = user_table.get_item(Key={'UserId': user_id})
        if 'Item' in resp: profile = resp['Item']; profile.setdefault('allergies', []); return profile
        else: print(f"No profile for {user_id}. Creating."); new_profile = {'UserId': user_id, 'allergies': []};
user_table.put_item(Item=new_profile); return new_profile
    except Exception as e: print(f"DB Error get/create profile: {str(e)}"); return {'allergies': []}
```

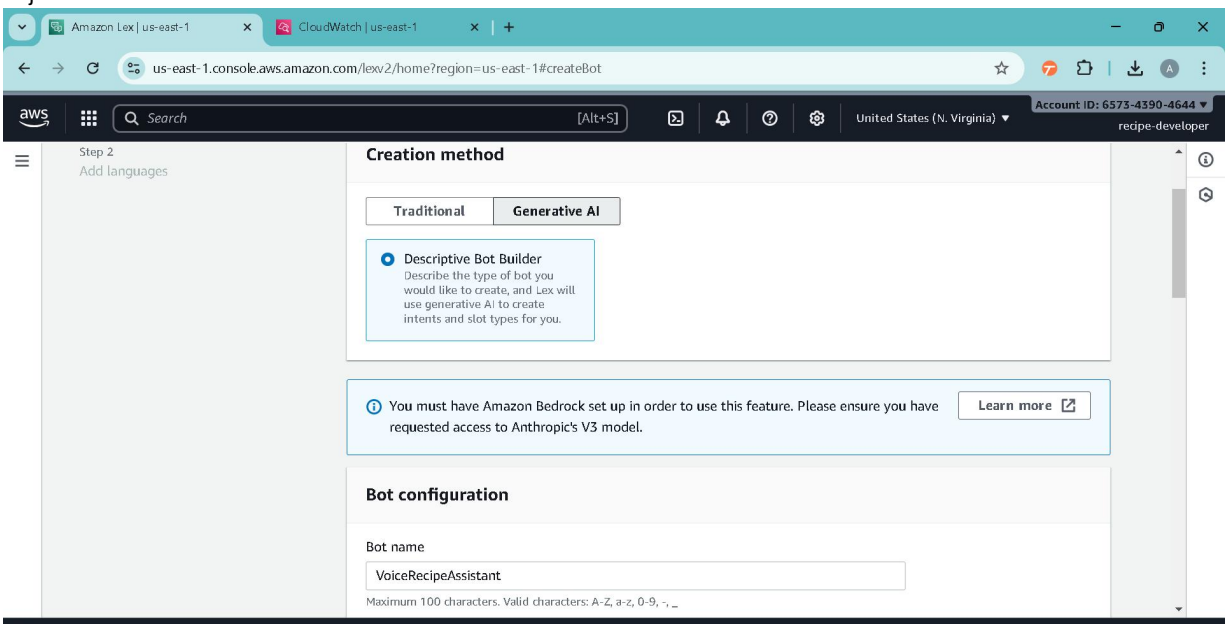


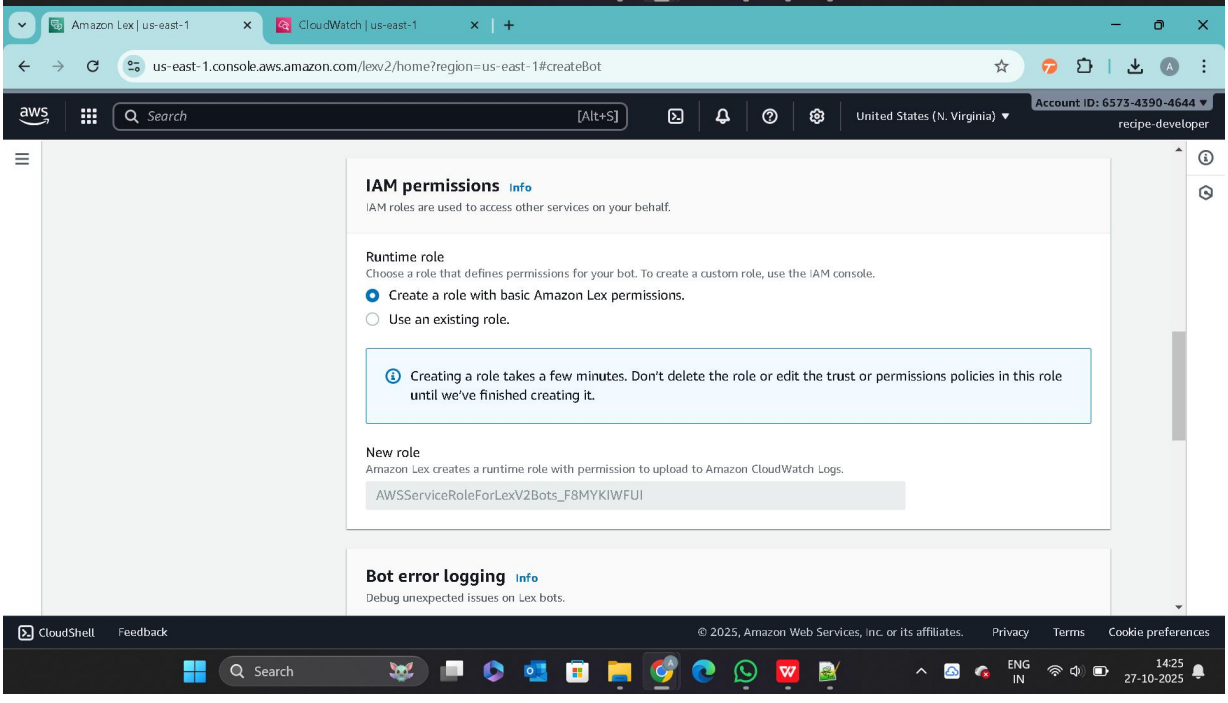


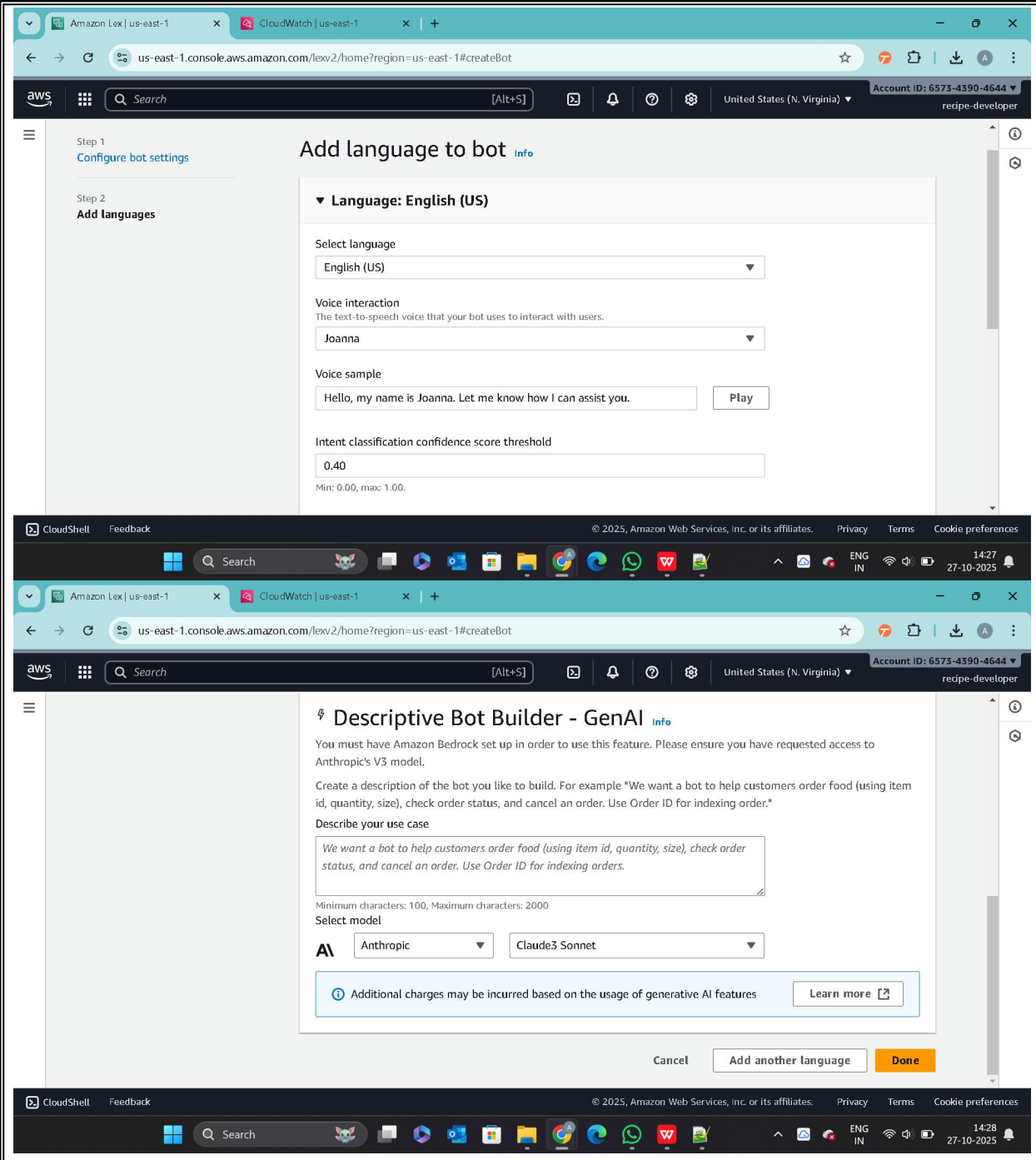
This were success details I got:


```
{
  "sessionState": {
    "dialogAction": {
      "type": "Close"
    },
    "intent": {
      "name": "SearchRecipes",
      "state": "Fulfilled"
    }
  },
  "messages": [
    {
```

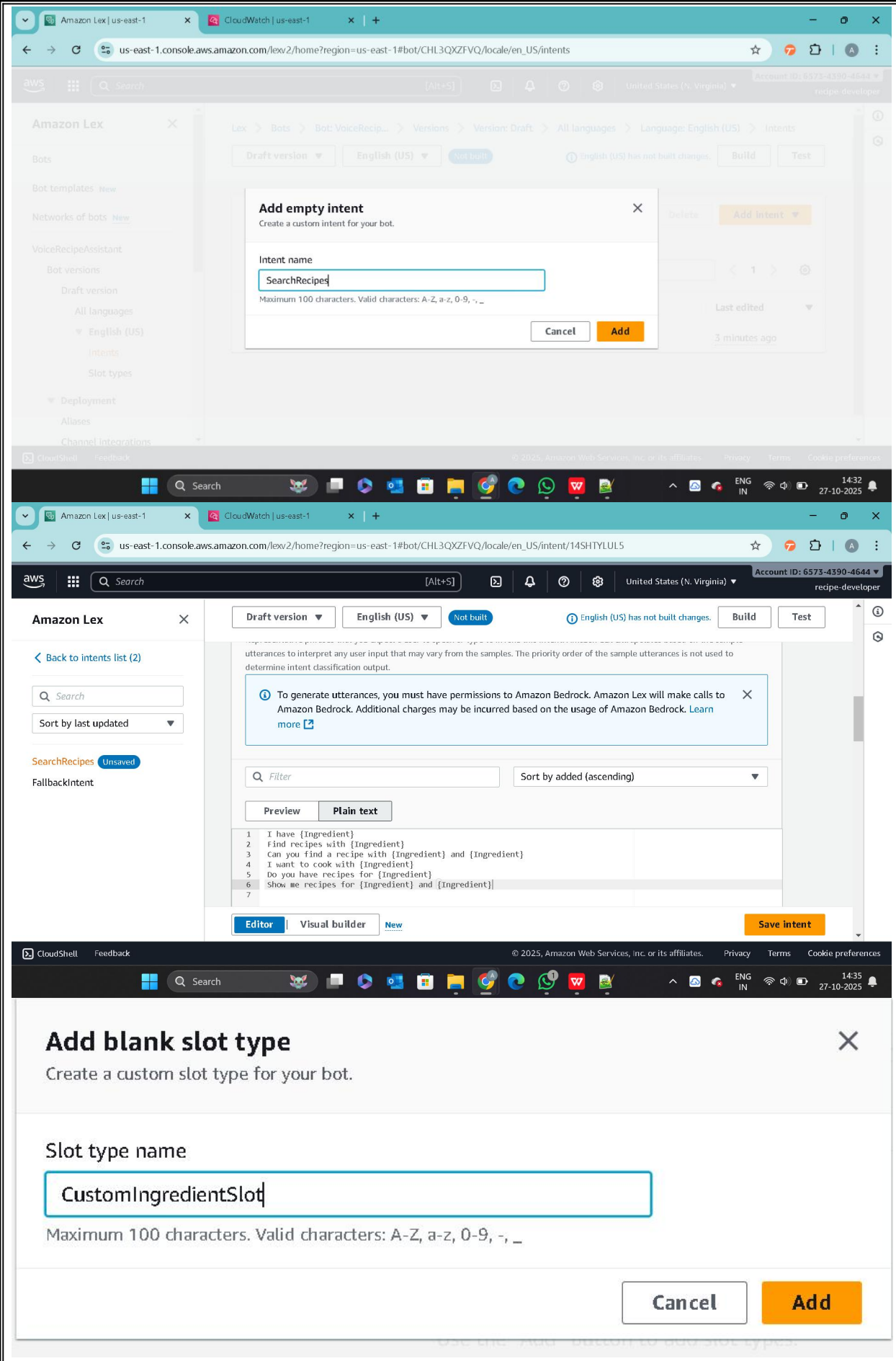
```
        "contentType": "PlainText",
        "content": "Success! I found 3 recipes using paneer,tomato. The top result is: Vegetable Paneer Skewers Or Paneer Tikka. You can see an image here: https://img.spoonacular.com/recipes/664575-312x231.jpg"
      }
    ]
  }
```

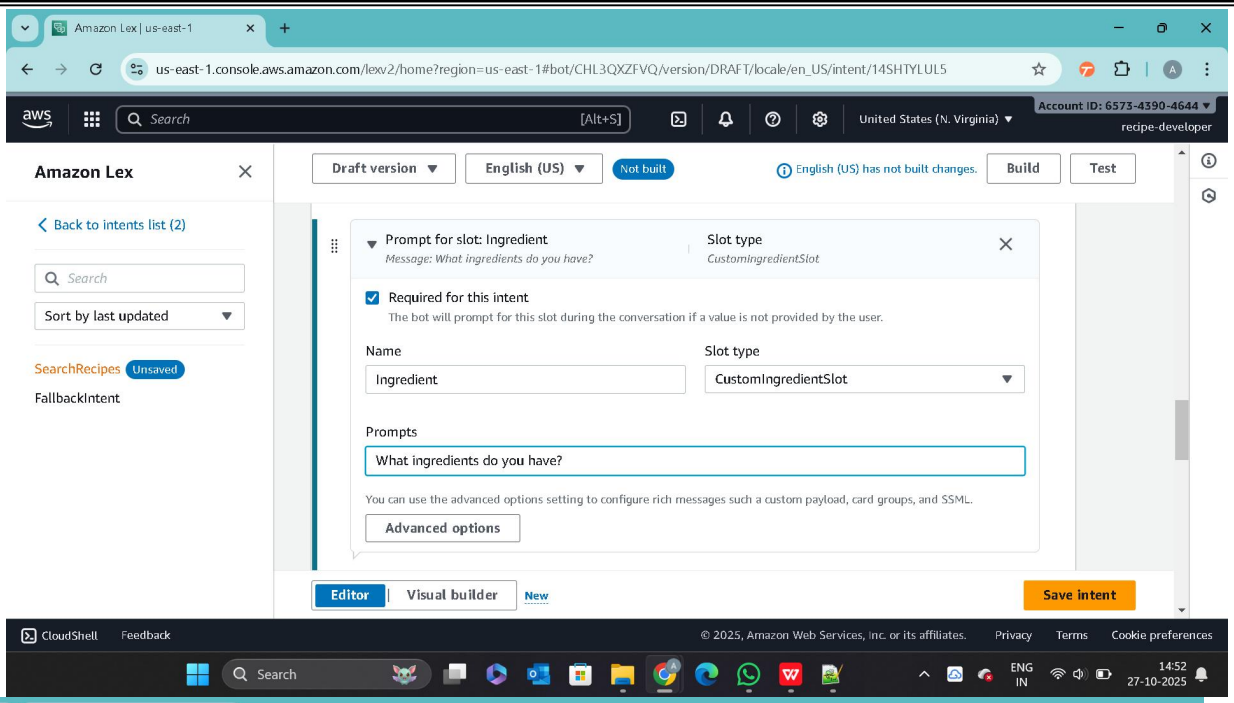



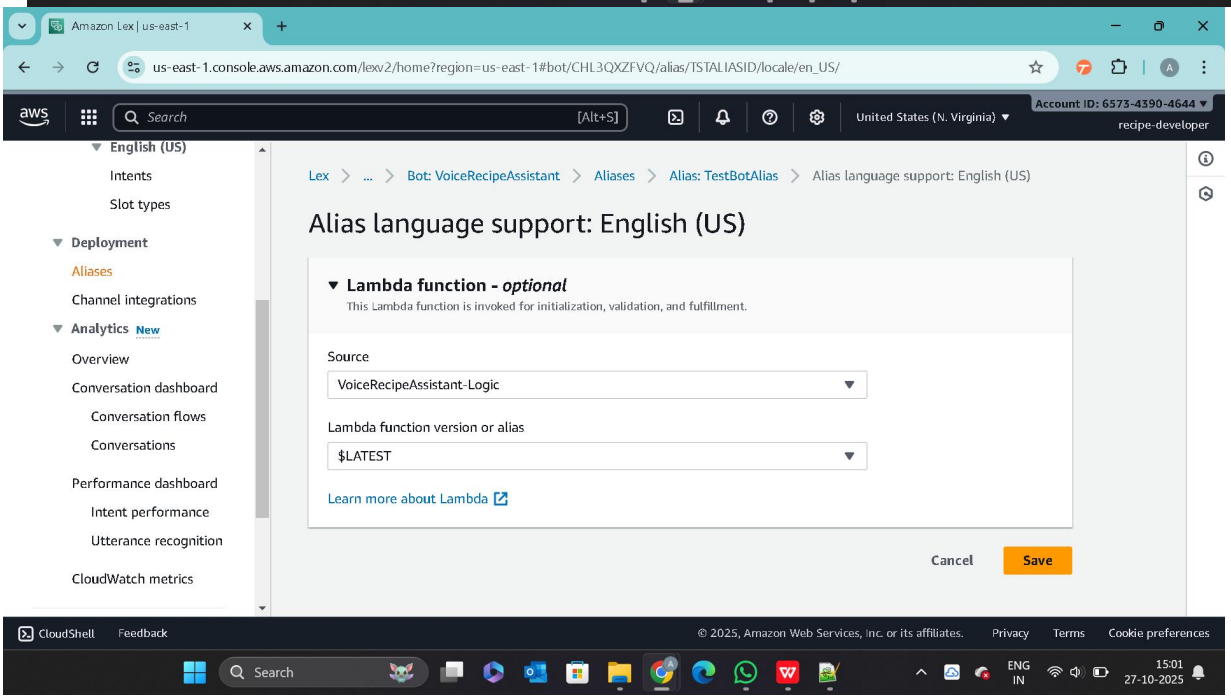


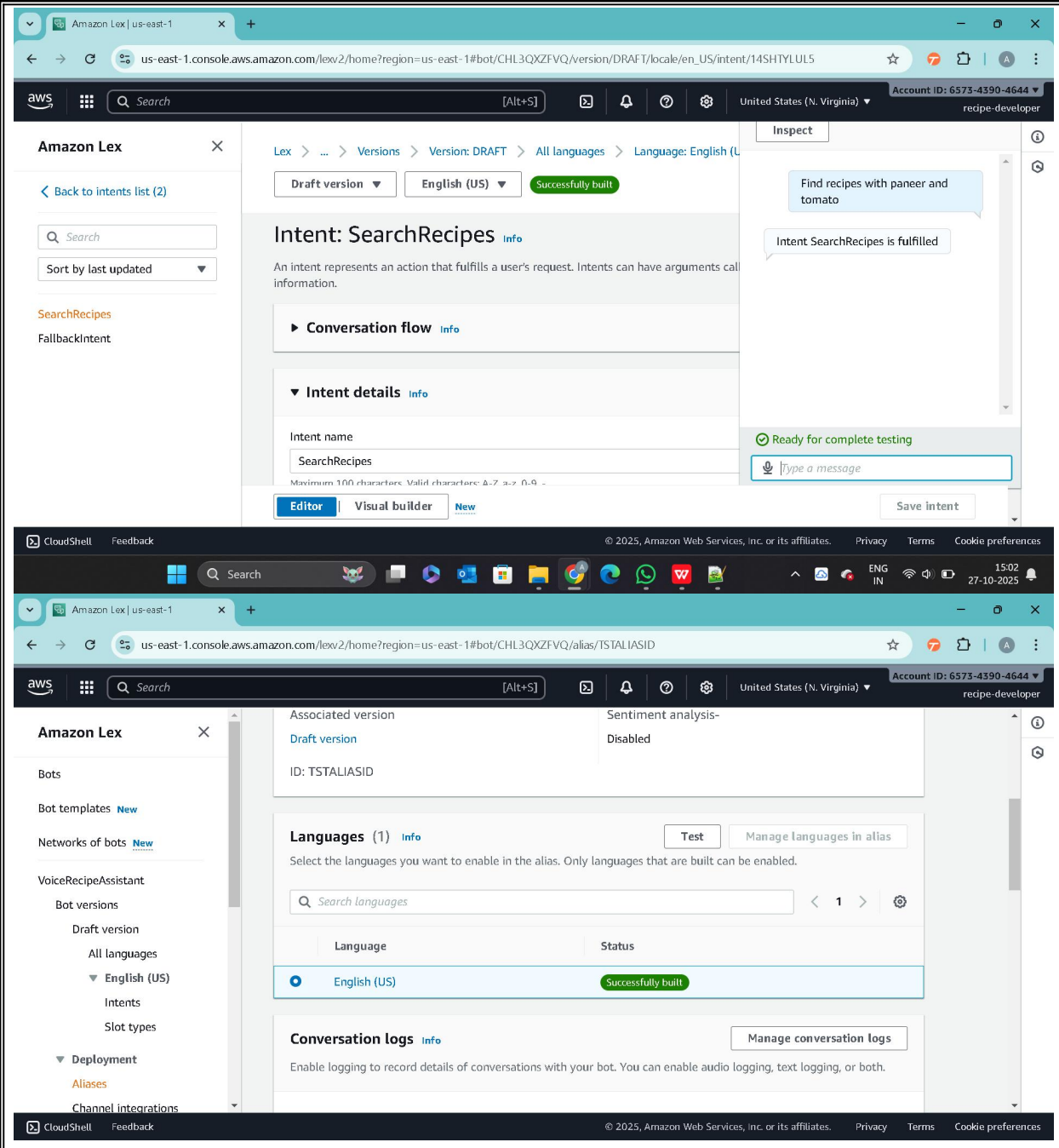

Amazon Lex | us-east-1
CloudWatch | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#createBot
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Step 1
Configure bot settings
Step 2
Add languages
Add language to bot Info
Language: English (US)
Select language
English (US)
Voice interaction
The text-to-speech voice that your bot uses to interact with users.
Joanna
Voice sample
Hello, my name is Joanna. Let me know how I can assist you.
Play
Intent classification confidence score threshold
0.40
Min: 0.00, max: 1.00.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:27
27-10-2025

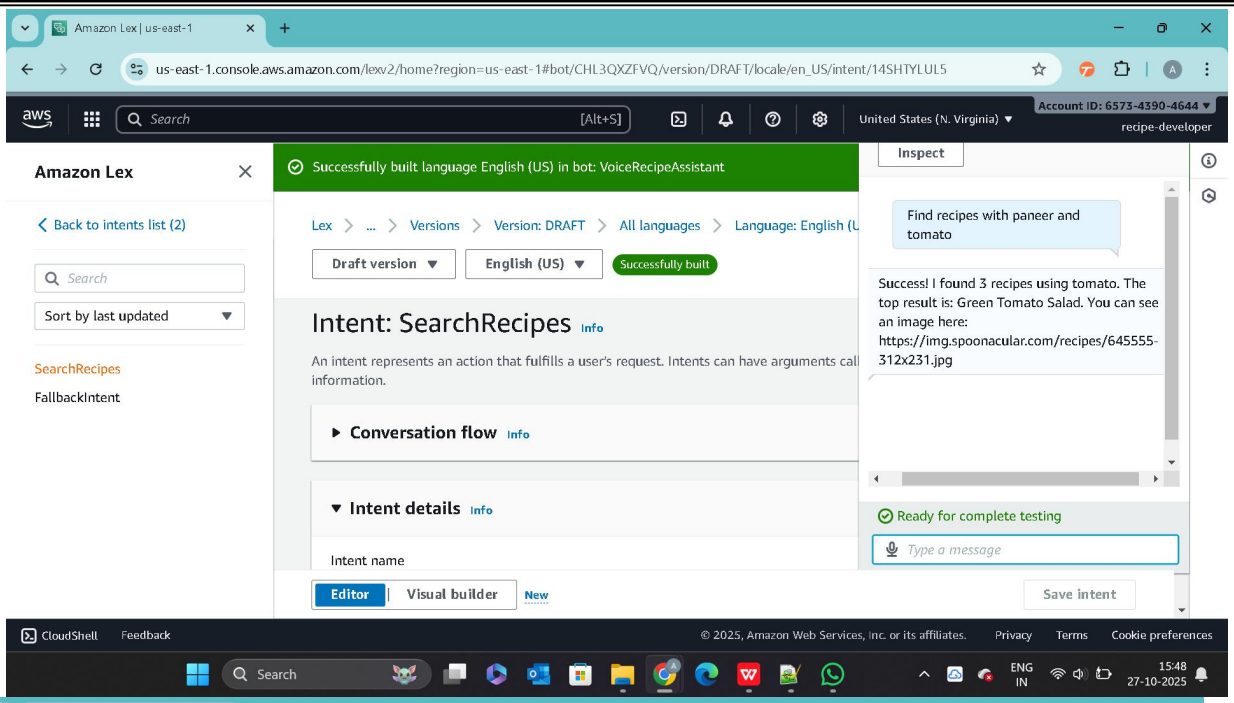
Amazon Lex | us-east-1
CloudWatch | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#createBot
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Descriptive Bot Builder - GenAI Info
You must have Amazon Bedrock set up in order to use this feature. Please ensure you have requested access to Anthropic's V3 model.
Create a description of the bot you like to build. For example "We want a bot to help customers order food (using item id, quantity, size), check order status, and cancel an order. Use Order ID for indexing order."
Describe your use case
We want a bot to help customers order food (using item id, quantity, size), check order status, and cancel an order. Use Order ID for indexing orders.
Minimum characters: 100, Maximum characters: 2000
Select model
Anthropic
Claude3 Sonnet
Additional charges may be incurred based on the usage of generative AI features
Learn more
Cancel
Add another language
Done
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:28
27-10-2025

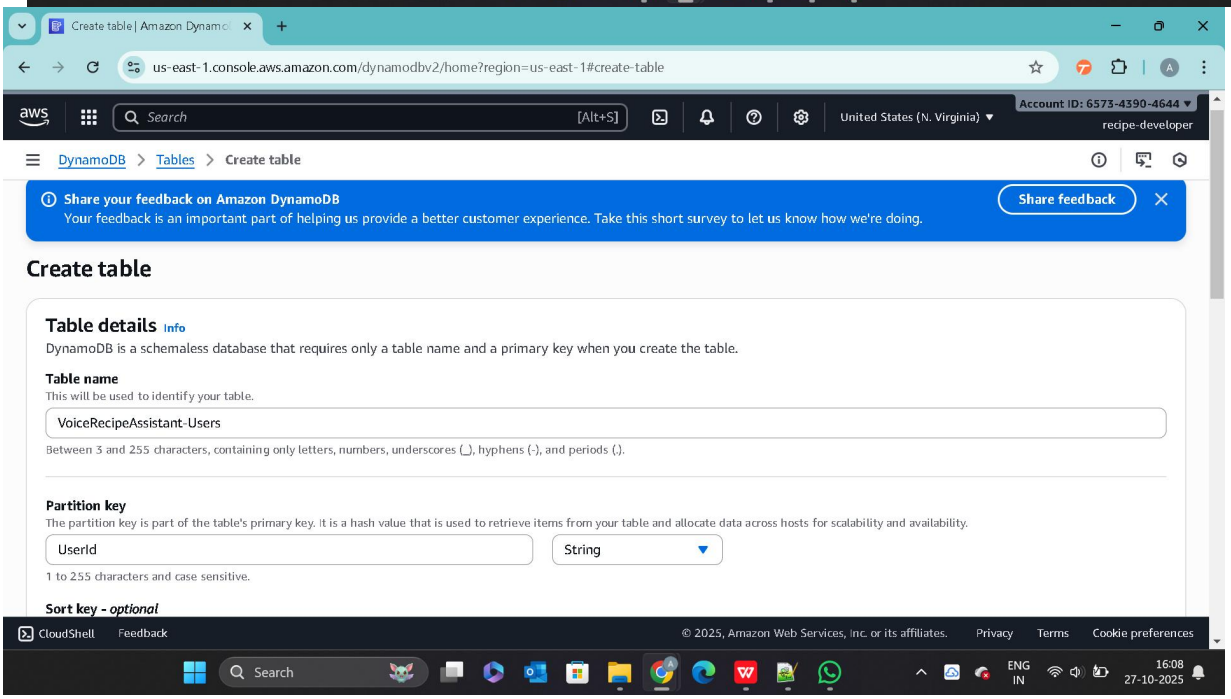

Amazon Lex | us-east-1
CloudWatch | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intents
Add empty intent
Create a custom intent for your bot.
Intent name
SearchRecipes
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/14SHTYLUL5
Amazon Lex
Back to intents list (2)
Sort by last updated
SearchRecipes
Unsaved
FallbackIntent
Draft version
English (US)
Not built
English (US) has not built changes.
Build
Test
To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more
Sort by added (ascending)
Preview
Plain text
I have {Ingredient}
Find recipes with {Ingredient}
Can you find a recipe with {Ingredient} and {Ingredient}
I want to cook with {Ingredient}
Do you have recipes for {Ingredient}
Show me recipes for {Ingredient} and {Ingredient}
Editor
Visual builder
New
Save intent
CloudShell
Feedback
Add blank slot type
Create a custom slot type for your bot.
Slot type name
CustomIngredientSlot
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add

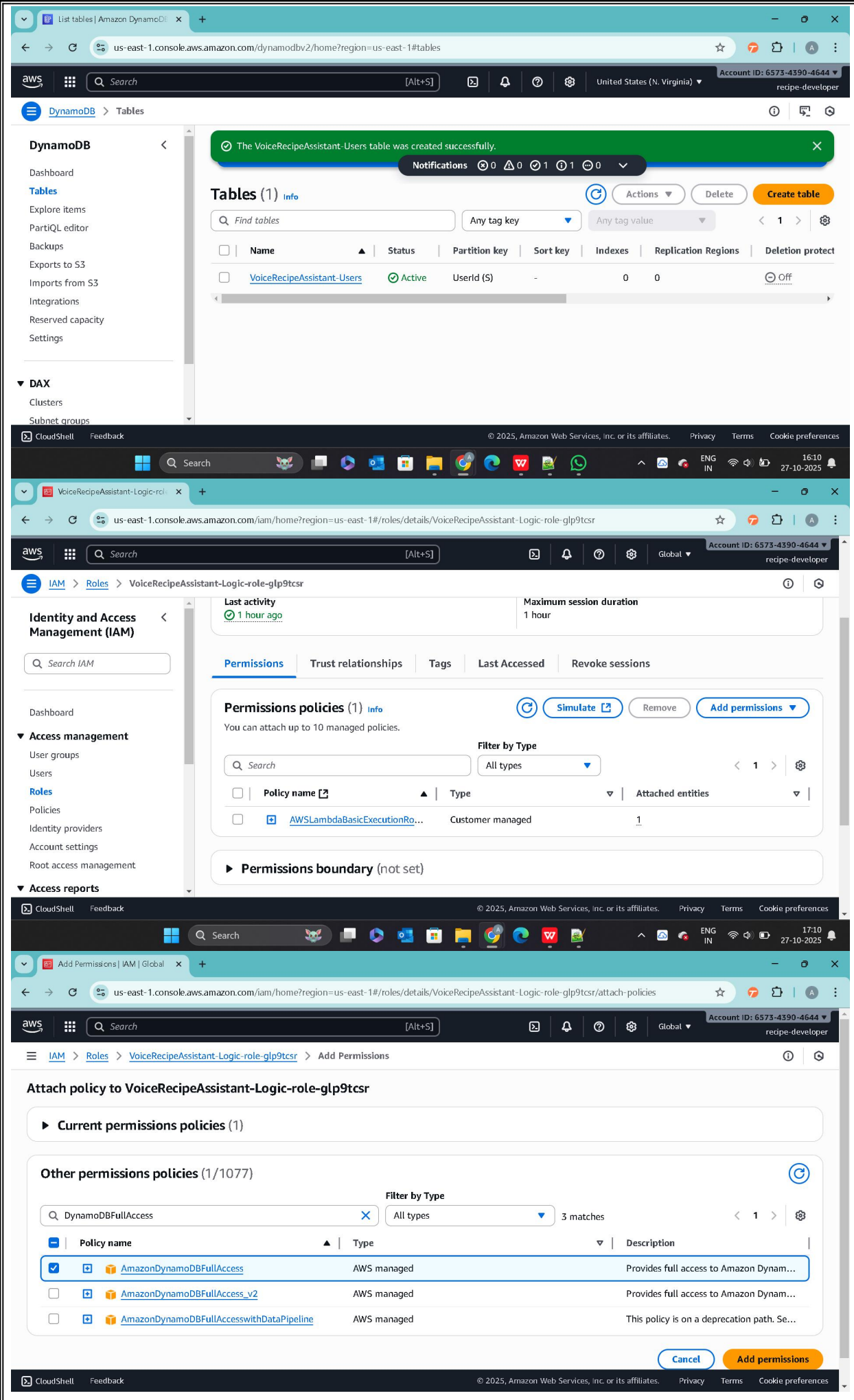

Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/version/DRAFT/locale/en_US/intent/14SHTYLUL5
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (2)
Search
Sort by last updated
SearchRecipes Unsaved
FallbackIntent
Draft version
English (US)
Not built
English (US) has not built changes.
Build
Test
Prompt for slot: Ingredient
Message: What ingredients do you have?
Slot type
CustomIngredientSlot
Required for this intent
The bot will prompt for this slot during the conversation if a value is not provided by the user.
Name
Ingredient
Slot type
CustomIngredientSlot
Prompts
What ingredients do you have?
You can use the advanced options setting to configure rich messages such a custom payload, card groups, and SSML.
Advanced options
Editor
Visual builder
New
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:52
27-10-2025

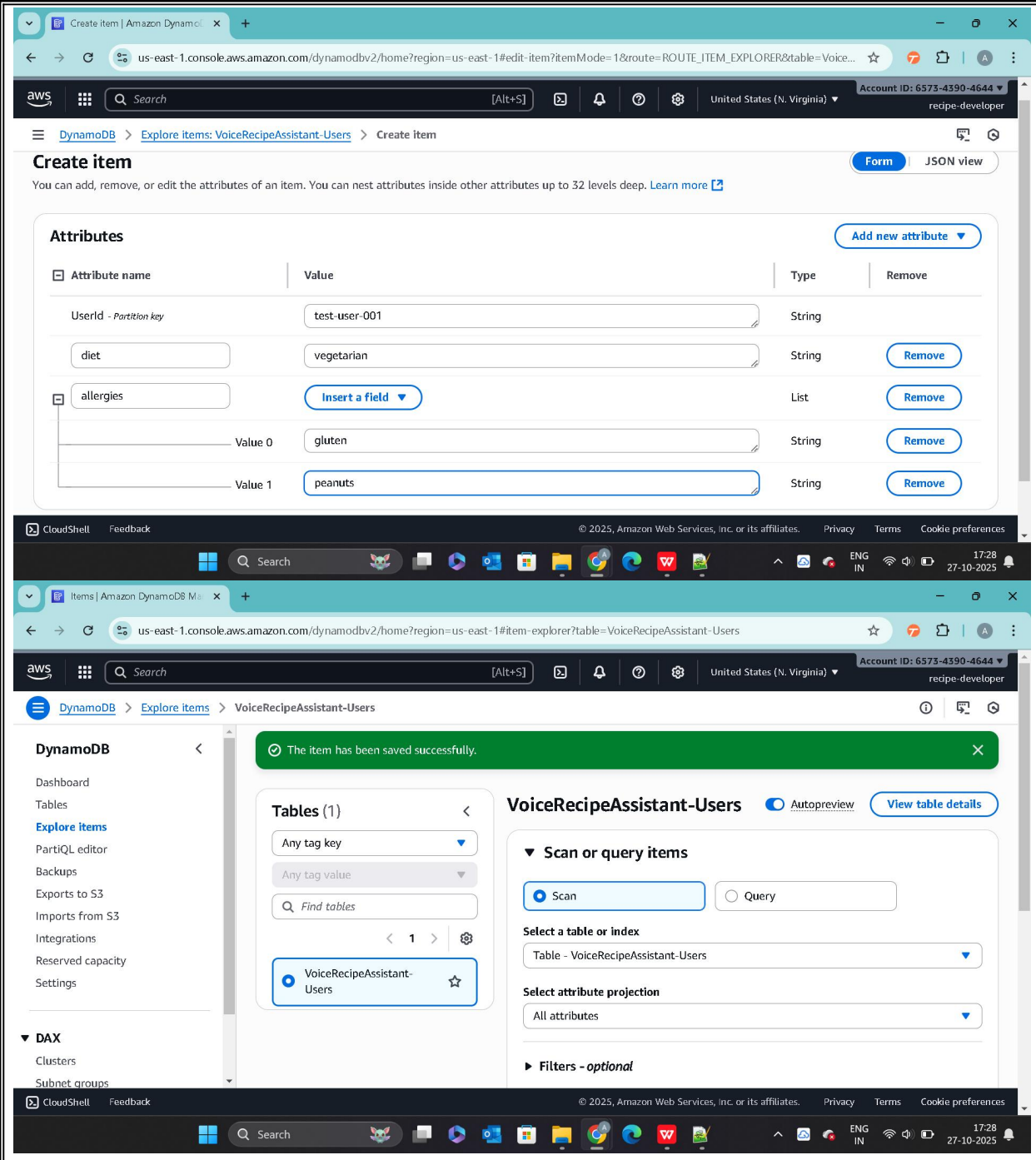

Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/alias/TSTALIASID/locale/en_US/
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
English (US)
Intents
Slot types
Deployment
Aliases
Channel integrations
Analytics New
Overview
Conversation dashboard
Conversation flows
Conversations
Performance dashboard
Intent performance
Utterance recognition
CloudWatch metrics
Lex > ... > Bot: VoiceRecipeAssistant > Aliases > Alias: TestBotAlias > Alias language support: English (US)
Alias language support: English (US)
Lambda function - optional
This Lambda function is invoked for initialization, validation, and fulfillment.
Source
VoiceRecipeAssistant-Logic
Lambda function version or alias
$LATEST
Learn more about Lambda
Cancel
Save
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
15:01
27-10-2025

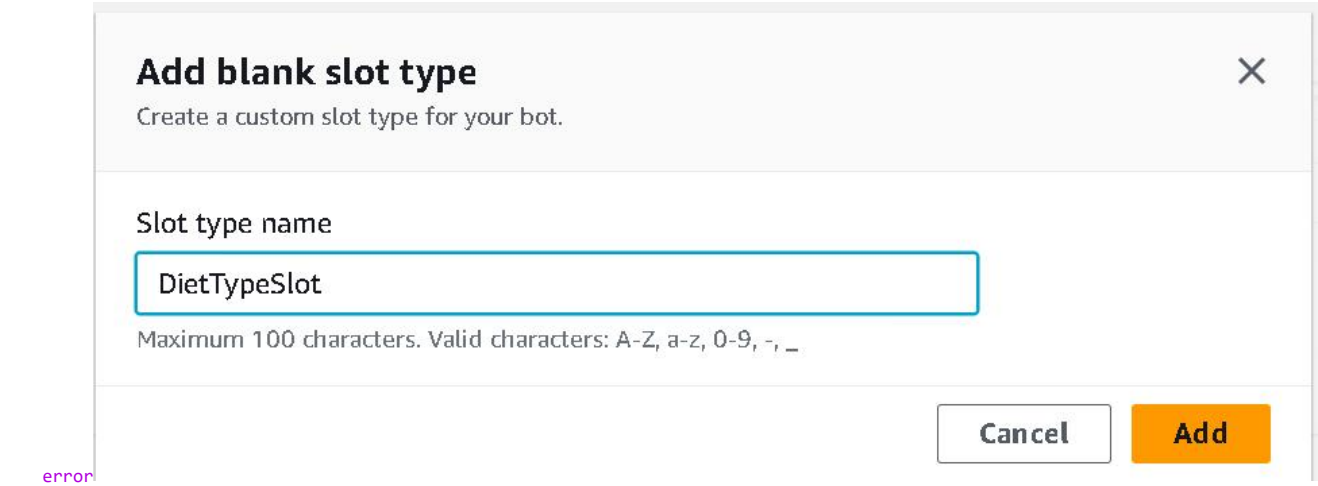

Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/version/DRAFT/locale/en_US/intent/14SHTYLUL5
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (2)
Search
Sort by last updated
SearchRecipes
FallbackIntent
Lex > ... > Versions > Version: DRAFT > All languages > Language: English (U
Draft version
English (US)
Successfully built
Intent: SearchRecipes Info
An intent represents an action that fulfills a user's request. Intents can have arguments call
information.
Conversation flow Info
Intent details Info
Intent name
SearchRecipes
Editor | Visual builder New
Save intent
Inspect
Find recipes with paneer and tomato
Intent SearchRecipes is fulfilled
Ready for complete testing
Type a message
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences
Search
ENG IN
15:02 27-10-2025
Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/alias/TSTALIASID
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Bots
Bot templates New
Networks of bots New
VoiceRecipeAssistant
Bot versions
Draft version
All languages
English (US)
Intents
Slot types
Deployment
Aliases
Channel integrations
Associated version
Draft version
ID: TSTALIASID
Sentiment analysis-
Disabled
Languages (1) Info
Test
Manage languages in alias
Select the languages you want to enable in the alias. Only languages that are built can be enabled.
Search languages
1
Language
Status
English (US)
Successfully built
Conversation logs Info
Manage conversation logs
Enable logging to record details of conversations with your bot. You can enable audio logging, text logging, or both.
CloudShell Feedback
© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences


Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/version/DRAFT/locale/en_US/intent/14SHTYLUL5
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (2)
Search
Sort by last updated
SearchRecipes
FallbackIntent
Successfully built language English (US) in bot: VoiceRecipeAssistant
Lex > ... > Versions > Version: DRAFT > All languages > Language: English (U
Draft version
English (US)
Successfully built
Intent: SearchRecipes Info
An intent represents an action that fulfills a user's request. Intents can have arguments call
information.
Conversation flow Info
Intent details Info
Intent name
Editor
Visual builder
New
Inspect
Find recipes with paneer and tomato
Success! I found 3 recipes using tomato. The top result is: Green Tomato Salad. You can see an image here: https://img.spoonacular.com/recipes/645555-312x231.jpg
Ready for complete testing
Type a message
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
15:48
27-10-2025

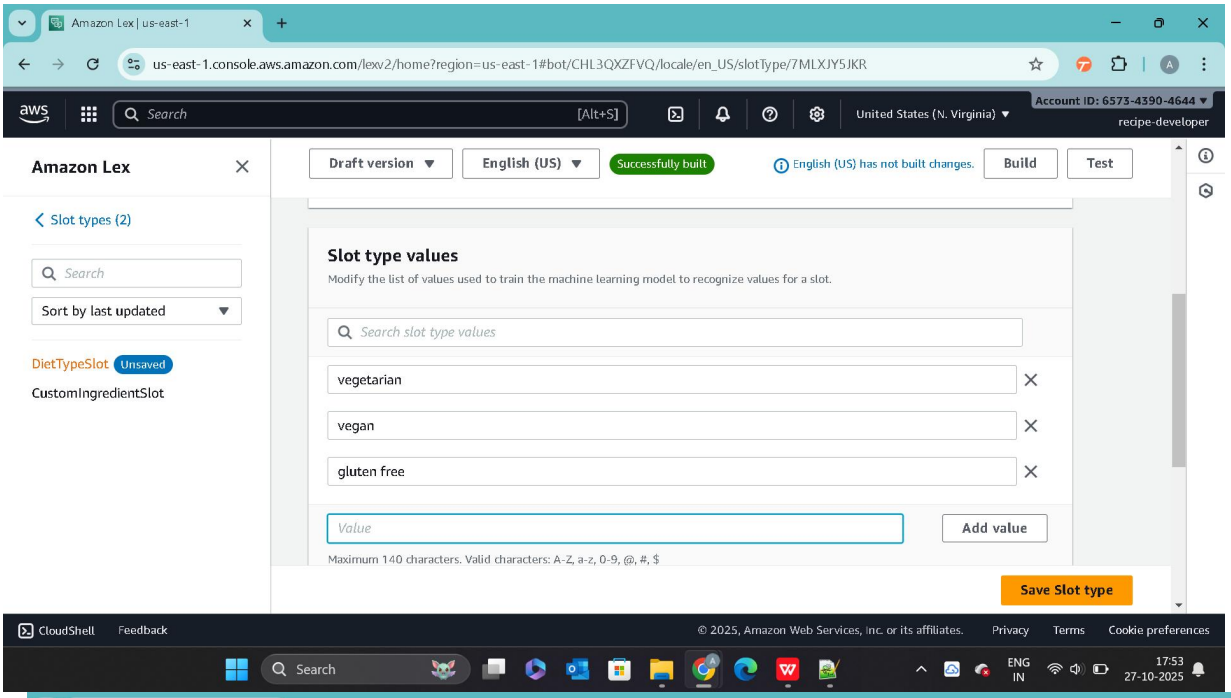

Create table | Amazon Dynamo
us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#create-table
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
DynamoDB > Tables > Create table
Share your feedback on Amazon DynamoDB
Your feedback is an important part of helping us provide a better customer experience. Take this short survey to let us know how we're doing.
Share feedback
Create table
Table details Info
DynamoDB is a schemaless database that requires only a table name and a primary key when you create the table.
Table name
This will be used to identify your table.
VoiceRecipeAssistant-Users
Between 3 and 255 characters, containing only letters, numbers, underscores (_), hyphens (-), and periods (.).
Partition key
The partition key is part of the table's primary key. It is a hash value that is used to retrieve items from your table and allocate data across hosts for scalability and availability.
UserId
String
1 to 255 characters and case sensitive.
Sort key - optional
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
16:08
27-10-2025

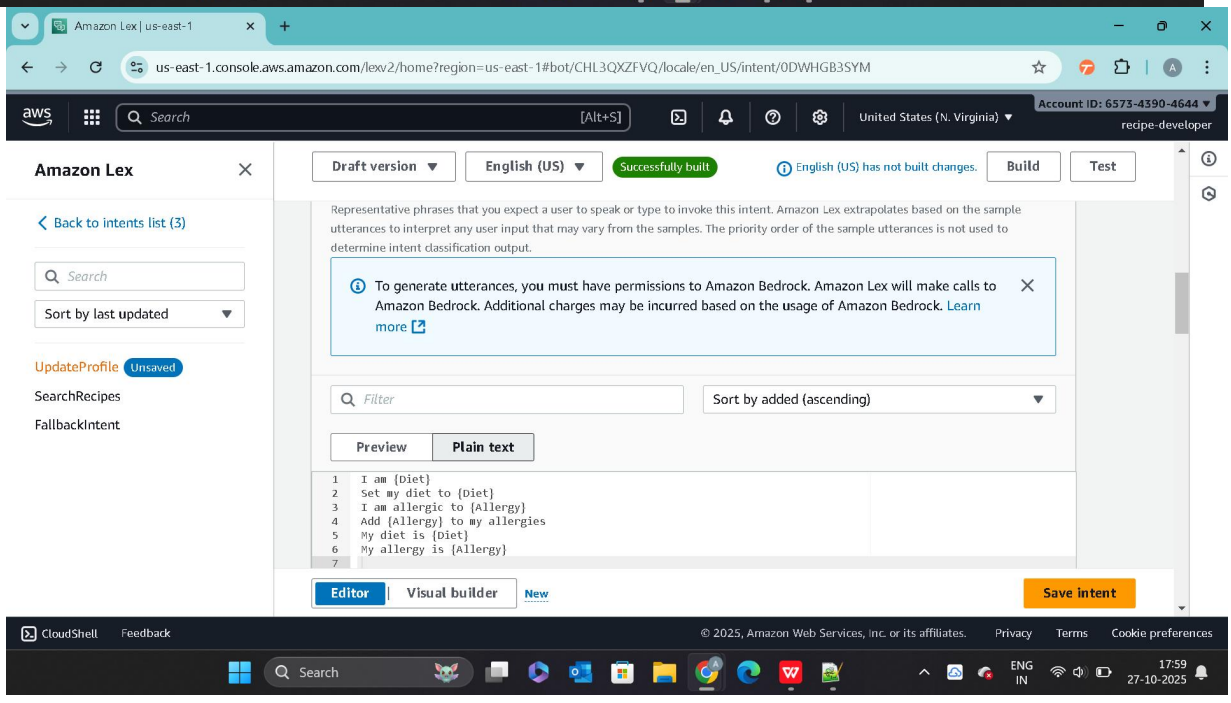

List tables | Amazon DynamoDB
us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#tables
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
DynamoDB > Tables
DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
The VoiceRecipeAssistant-Users table was created successfully.
Notifications
Tables (1) Info
Actions
Delete
Create table
Find tables
Any tag key
Any tag value
Name
Status
Partition key
Sort key
Indexes
Replication Regions
Deletion protect
VoiceRecipeAssistant-Users
Active
UserId (S)
0
0
Off
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
16:10
27-10-2025
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/VoiceRecipeAssistant-Logic-role-glp9tcsr
Global
IAM > Roles > VoiceRecipeAssistant-Logic-role-glp9tcsr
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Access reports
Last activity
1 hour ago
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (1) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
Search
All types
Policy name
Type
Attached entities
AWSLambdaBasicExecutionRo...
Customer managed
1
Permissions boundary (not set)
17:10
27-10-2025
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/VoiceRecipeAssistant-Logic-role-glp9tcsr/attach-policies
IAM > Roles > VoiceRecipeAssistant-Logic-role-glp9tcsr > Add Permissions
Attach policy to VoiceRecipeAssistant-Logic-role-glp9tcsr
Current permissions policies (1)
Other permissions policies (1/1077)
DynamoDBFullAccess
3 matches
Policy name
Type
Description
AmazonDynamoDBFullAccess
AWS managed
Provides full access to Amazon Dynam...
AmazonDynamoDBFullAccess_v2
AWS managed
Provides full access to Amazon Dynam...
AmazonDynamoDBFullAccesswithDataPipeline
AWS managed
This policy is on a deprecation path. Se...
Cancel
Add permissions


Create item | Amazon Dynamo
us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#edit-item?itemMode=1&route=ROUTE_ITEM_EXPLORER&table=Voice...
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
DynamoDB > Explore items: VoiceRecipeAssistant-Users > Create item
Create item
Form
JSON view
You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. Learn more
Attributes
Add new attribute
Attribute name
Value
Type
Remove
UserId - Partition key
test-user-001
String
diet
vegetarian
String
Remove
allergies
Insert a field
List
Remove
Value 0
gluten
String
Remove
Value 1
peanuts
String
Remove
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
17:28
27-10-2025
Items | Amazon DynamoDB Ma
us-east-1.console.aws.amazon.com/dynamodbv2/home?region=us-east-1#item-explorer?table=VoiceRecipeAssistant-Users
DynamoDB > Explore items > VoiceRecipeAssistant-Users
DynamoDB
Dashboard
Tables
Explore items
PartiQL editor
Backups
Exports to S3
Imports from S3
Integrations
Reserved capacity
Settings
DAX
Clusters
Subnet groups
The item has been saved successfully.
Tables (1)
Any tag key
Any tag value
Find tables
1
VoiceRecipeAssistant-Users
VoiceRecipeAssistant-Users
Autopreview
View table details
Scan or query items
Scan
Query
Select a table or index
Table - VoiceRecipeAssistant-Users
Select attribute projection
All attributes
Filters - optional


Add blank slot type
Create a custom slot type for your bot.
Slot type name
DietTypeSlot
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add

error


Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/slotType/7MLXJY5JKR
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Slot types (2)
Search
Sort by last updated
DietTypeSlot Unsaved
CustomIngredientSlot
Draft version
English (US)
Successfully built
English (US) has not built changes.
Build
Test
Slot type values
Modify the list of values used to train the machine learning model to recognize values for a slot.
Search slot type values
vegetarian
vegan
gluten free
Value
Add value
Maximum 140 characters. Valid characters: A-Z, a-z, 0-9, @, #, $
Save Slot type
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:53
27-10-2025


Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/0DWHGB3SYM
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (3)
Search
Sort by last updated
UpdateProfile Unsaved
SearchRecipes
FallbackIntent
Draft version
English (US)
Successfully built
English (US) has not built changes.
Build
Test
Representative phrases that you expect a user to speak or type to invoke this intent. Amazon Lex extrapolates based on the sample utterances to interpret any user input that may vary from the samples. The priority order of the sample utterances is not used to determine intent classification output.
To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more
Filter
Sort by added (ascending)
Preview
Plain text
1 I am {Diet}
2 Set my diet to {Diet}
3 I am allergic to {Allergy}
4 Add {Allergy} to my allergies
5 My diet is {Diet}
6 My allergy is {Allergy}
7
Editor
Visual builder
New
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
17:59
27-10-2025

Add empty intent

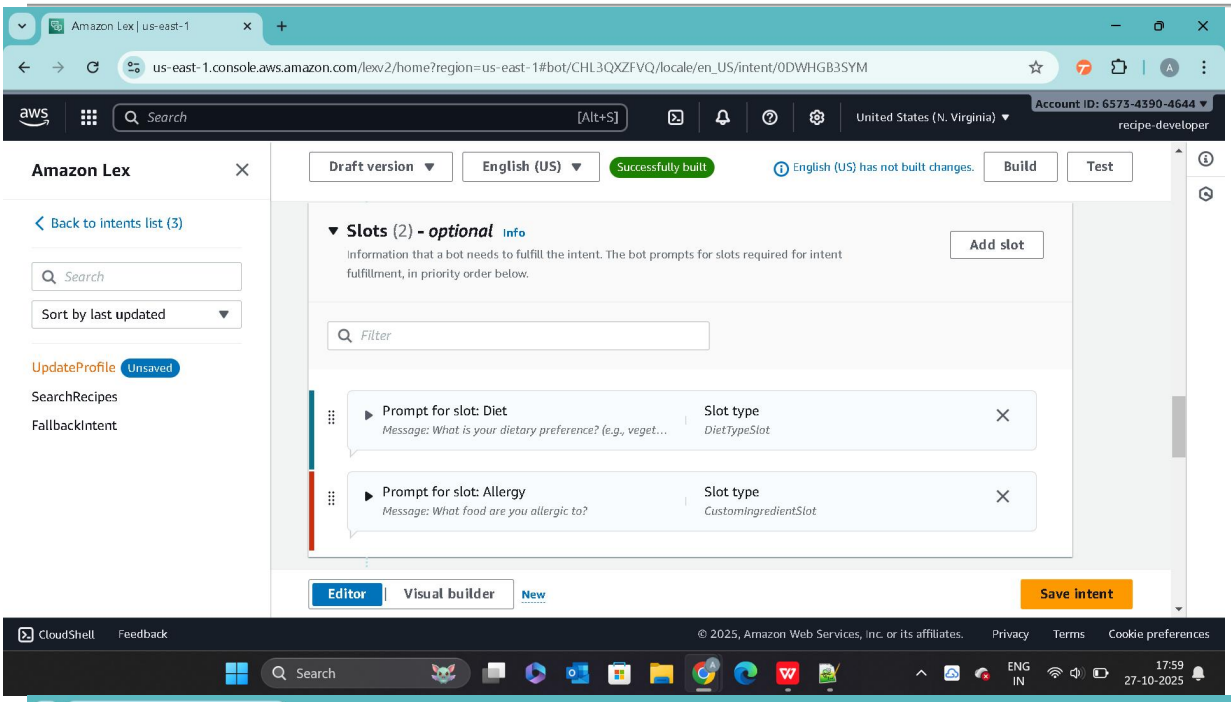
Create a custom intent for your bot.

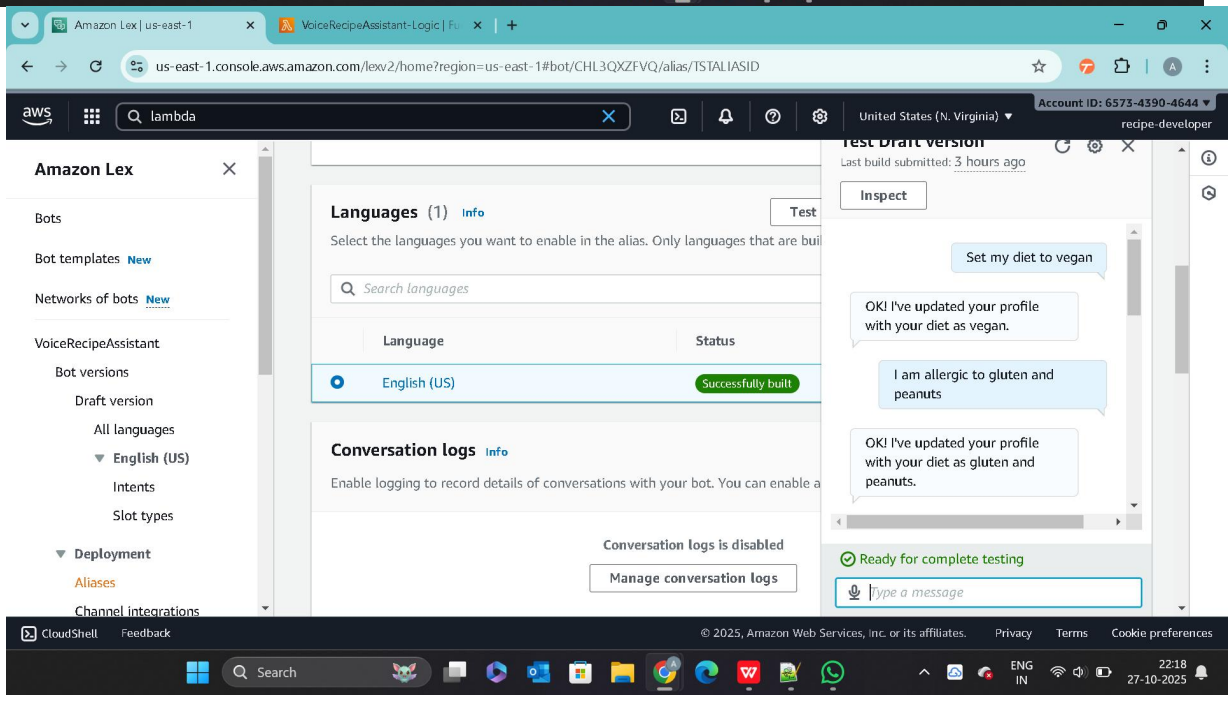
Intent name

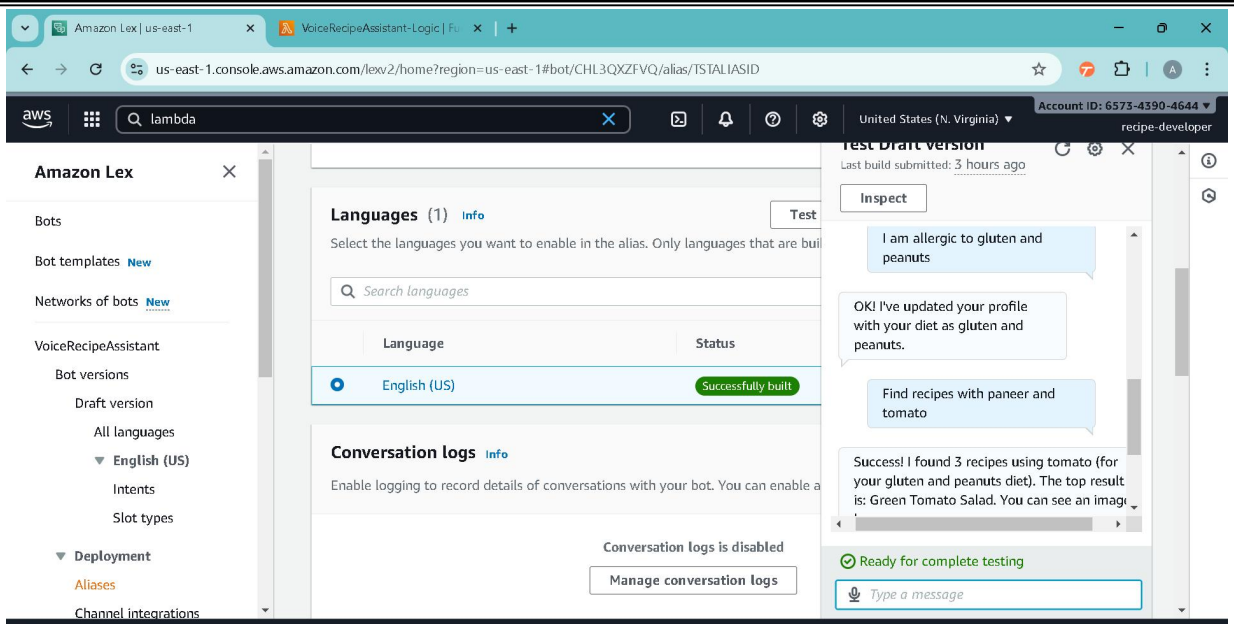
UpdateProfile

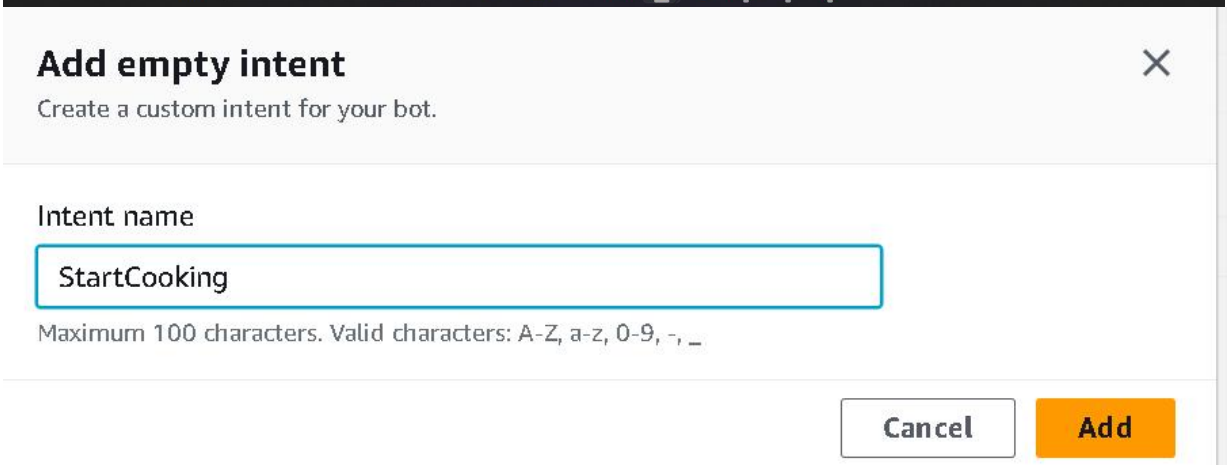
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _

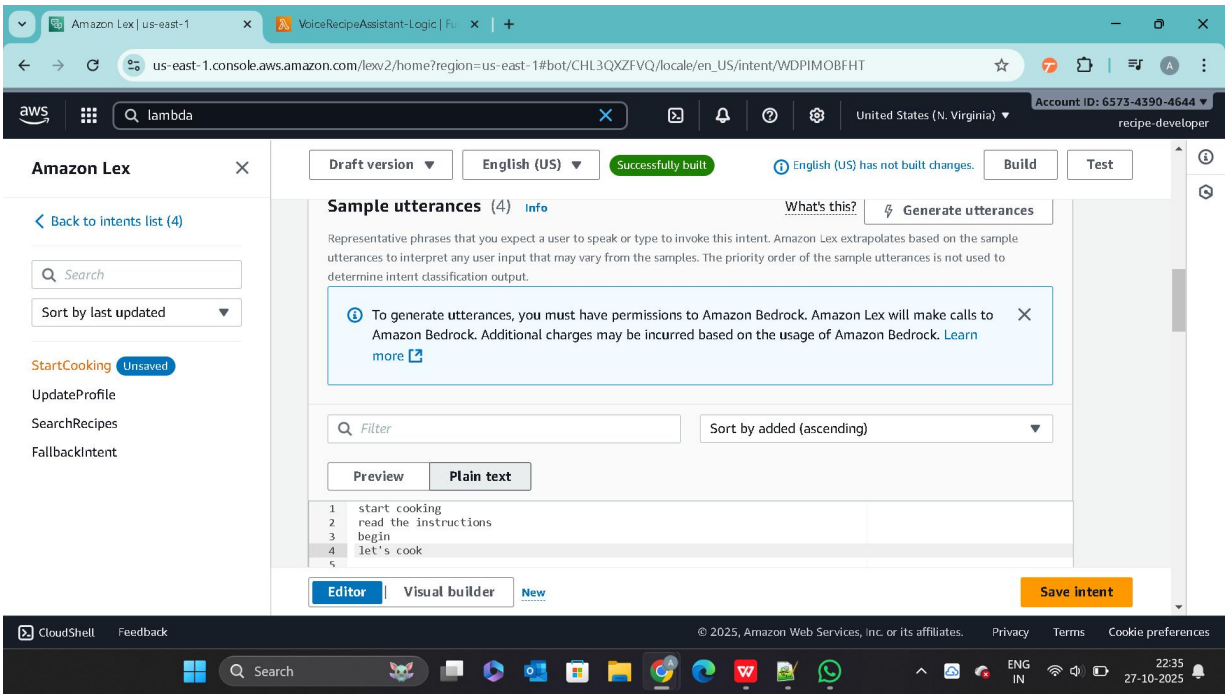
Cancel | Add

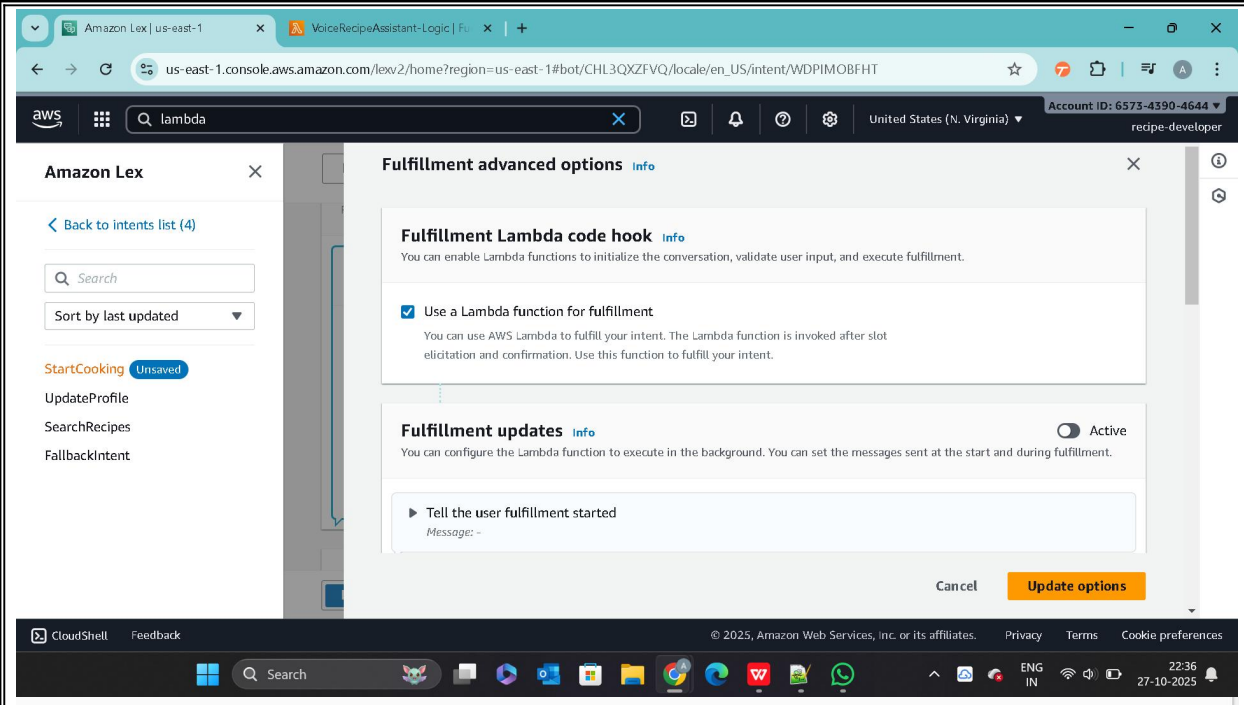


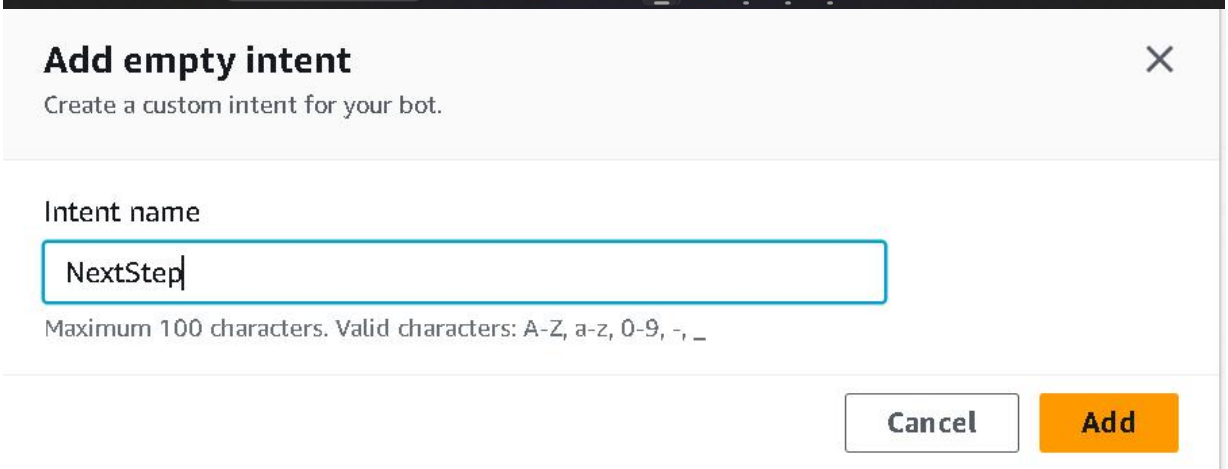


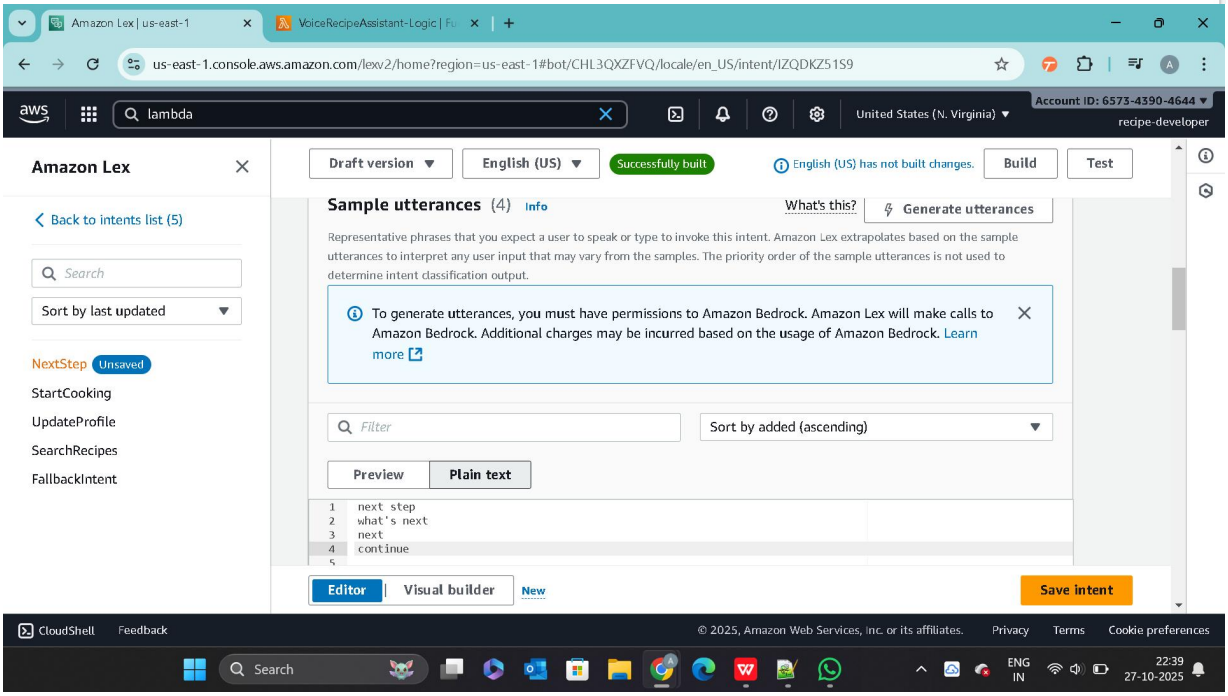

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic | Fu
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/alias/TSTALIASID
aws
lambda
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Bots
Bot templates New
Networks of bots New
VoiceRecipeAssistant
Bot versions
Draft version
All languages
English (US)
Intents
Slot types
Deployment
Aliases
Channel integrations
Languages (1) Info
Test
Select the languages you want to enable in the alias. Only languages that are bui
Search languages
Language
Status
English (US)
Successfully built
Conversation logs Info
Enable logging to record details of conversations with your bot. You can enable a
Conversation logs is disabled
Manage conversation logs
Test Draft version
Last build submitted: 3 hours ago
Inspect
I am allergic to gluten and peanuts
OK! I've updated your profile with your diet as gluten and peanuts.
Find recipes with paneer and tomato
Success! I found 3 recipes using tomato (for your gluten and peanuts diet). The top result is: Green Tomato Salad. You can see an image
Ready for complete testing
Type a message
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
22:19
27-10-2025

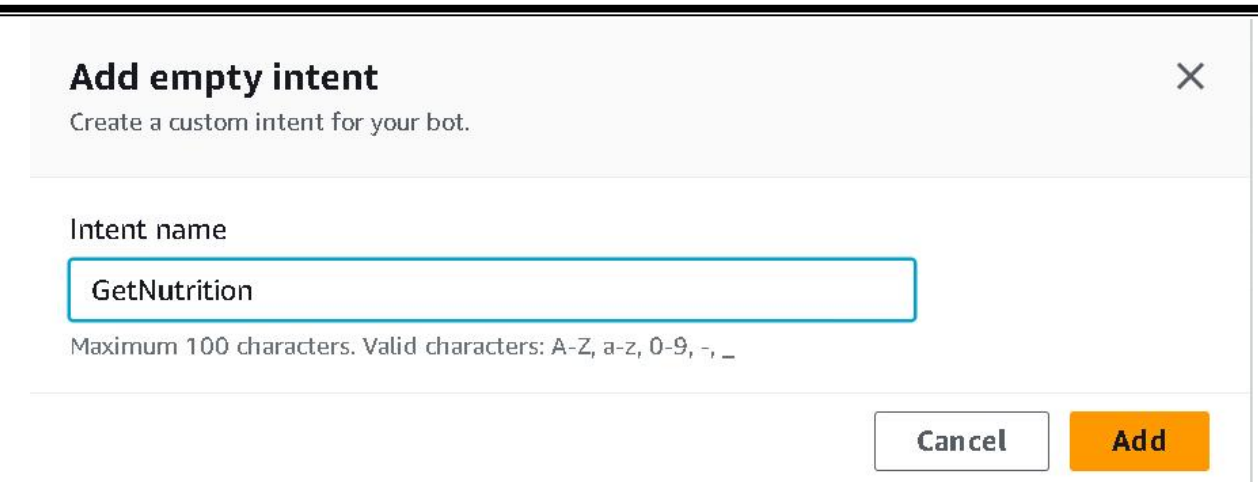

Add empty intent
Create a custom intent for your bot.
Intent name
StartCooking
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add

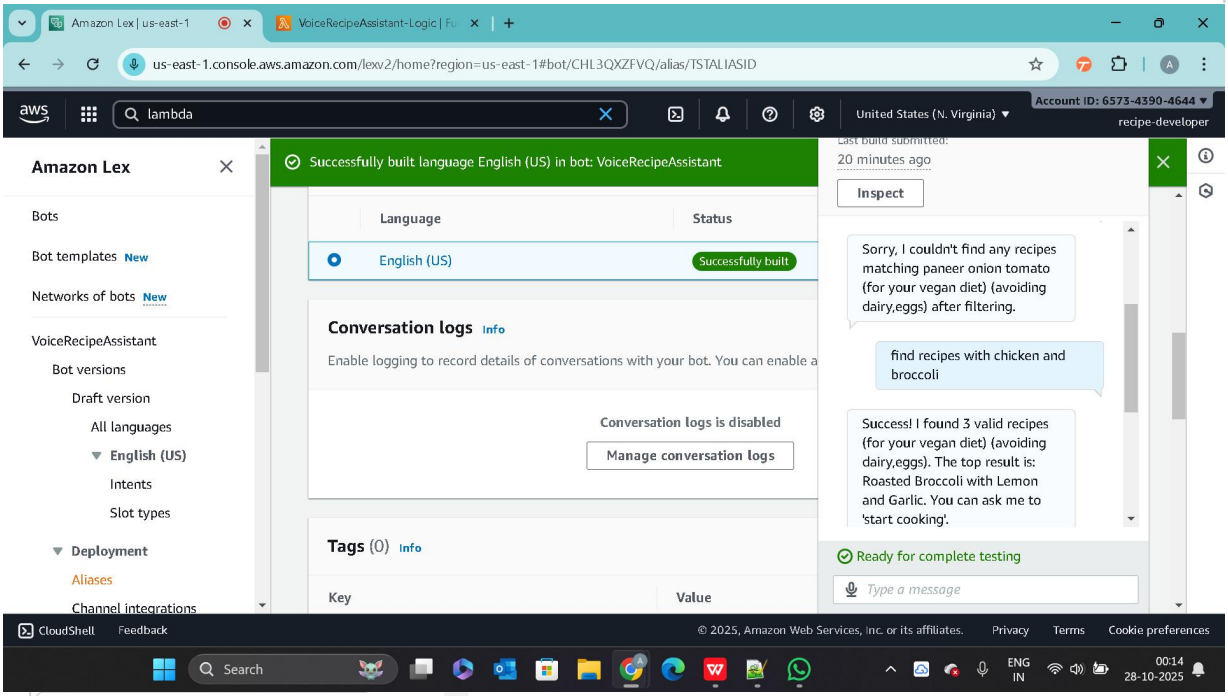

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic | Fu
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/WDPIMOBFHT
aws
lambda
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (4)
Search
Sort by last updated
StartCooking Unsaved
UpdateProfile
SearchRecipes
FallbackIntent
Draft version
English (US)
Successfully built
English (US) has not built changes.
Build
Test
Sample utterances (4) Info
What's this?
Generate utterances
Representative phrases that you expect a user to speak or type to invoke this intent. Amazon Lex extrapolates based on the sample utterances to interpret any user input that may vary from the samples. The priority order of the sample utterances is not used to determine intent classification output.
To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more
Filter
Sort by added (ascending)
Preview
Plain text
1 start cooking
2 read the instructions
3 begin
4 let's cook
5
Editor
Visual builder
New
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
22:35
27-10-2025

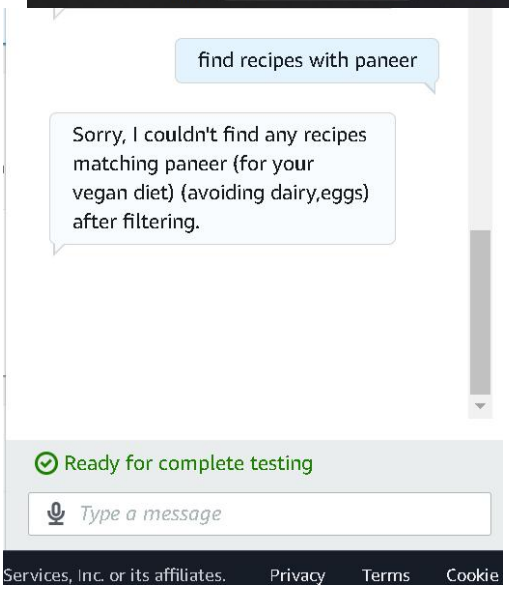

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/WDPIMOBFHT
lambda
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (4)
Search
Sort by last updated
StartCooking
Unsaved
UpdateProfile
SearchRecipes
FallbackIntent
Fulfillment advanced options
Info
Fulfillment Lambda code hook
Info
You can enable Lambda functions to initialize the conversation, validate user input, and execute fulfillment.
Use a Lambda function for fulfillment
You can use AWS Lambda to fulfill your intent. The Lambda function is invoked after slot elicitation and confirmation. Use this function to fulfill your intent.
Fulfillment updates
Info
Active
You can configure the Lambda function to execute in the background. You can set the messages sent at the start and during fulfillment.
Tell the user fulfillment started
Message: -
Cancel
Update options
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
22:36
27-10-2025

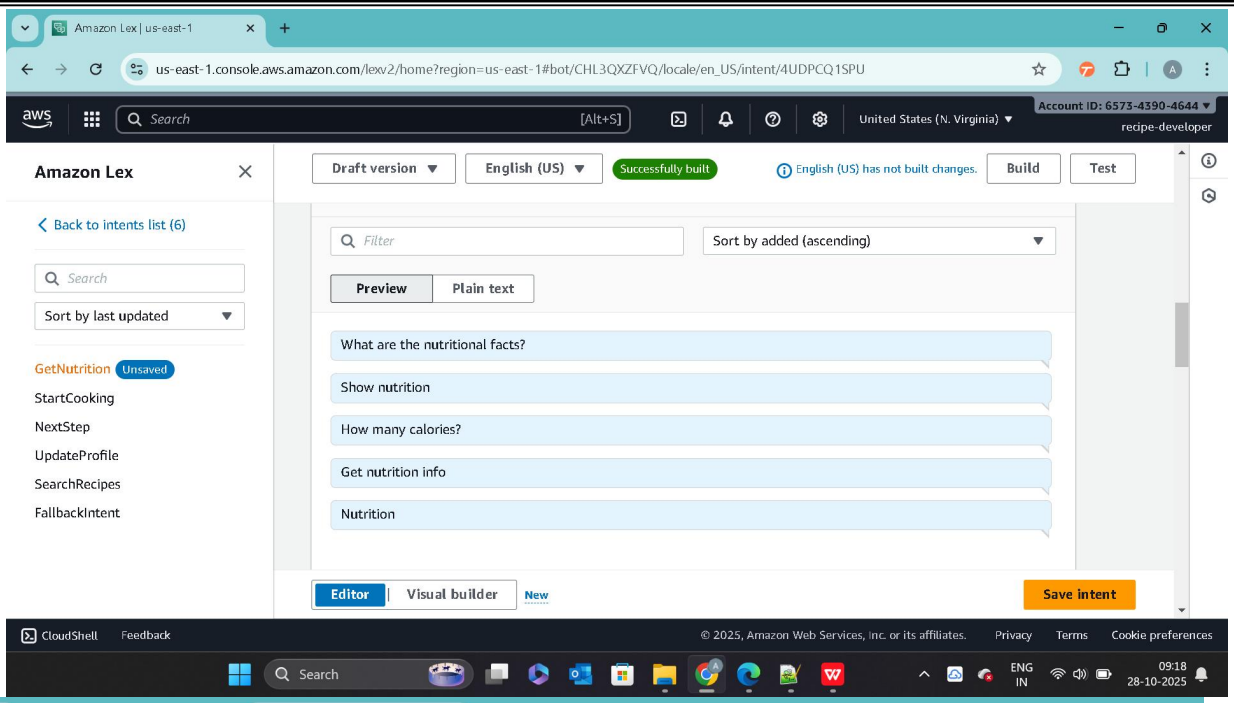

Add empty intent
Create a custom intent for your bot.
Intent name
NextStep
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add

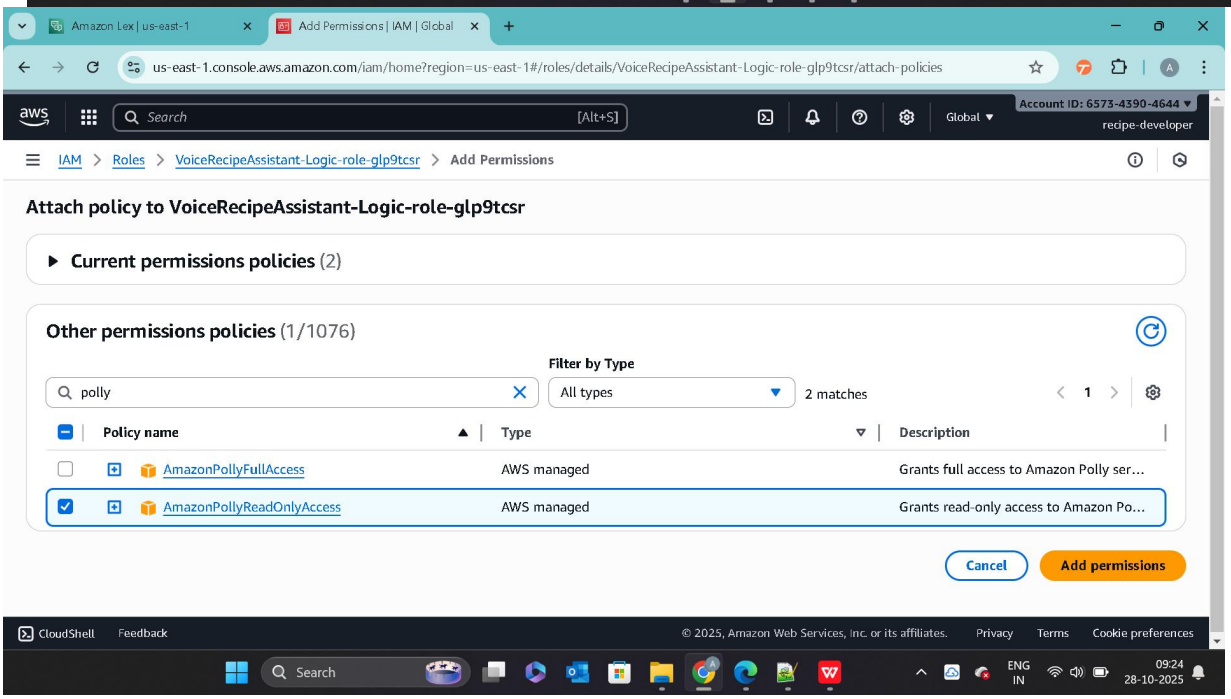

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/IZQDKZ51S9
lambda
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (5)
Search
Sort by last updated
NextStep
Unsaved
StartCooking
UpdateProfile
SearchRecipes
FallbackIntent
Draft version
English (US)
Successfully built
English (US) has not built changes.
Build
Test
Sample utterances (4)
Info
What's this?
Generate utterances
Representative phrases that you expect a user to speak or type to invoke this intent. Amazon Lex extrapolates based on the sample utterances to interpret any user input that may vary from the samples. The priority order of the sample utterances is not used to determine intent classification output.
To generate utterances, you must have permissions to Amazon Bedrock. Amazon Lex will make calls to Amazon Bedrock. Additional charges may be incurred based on the usage of Amazon Bedrock. Learn more
Filter
Sort by added (ascending)
Preview
Plain text
1 next step
2 what's next
3 next
4 continue
Editor
Visual builder
New
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
22:39
27-10-2025

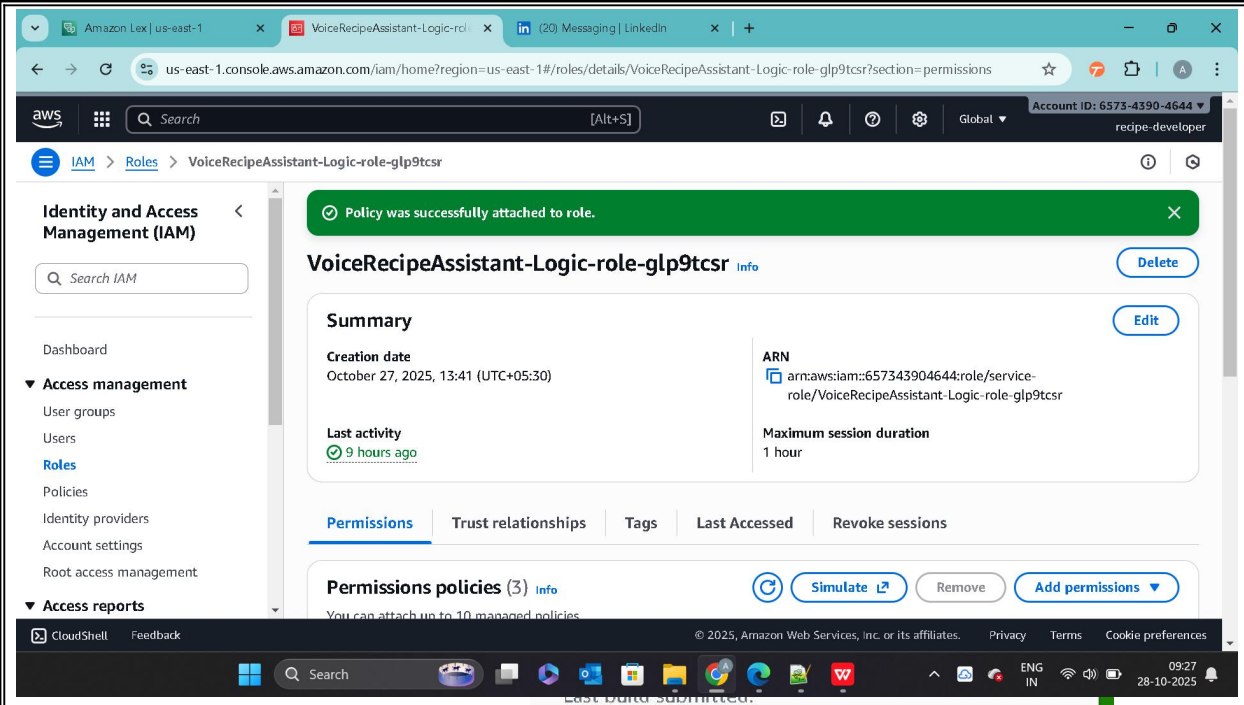

Add empty intent
Create a custom intent for your bot.
Intent name
GetNutrition
Maximum 100 characters. Valid characters: A-Z, a-z, 0-9, -, _
Cancel
Add

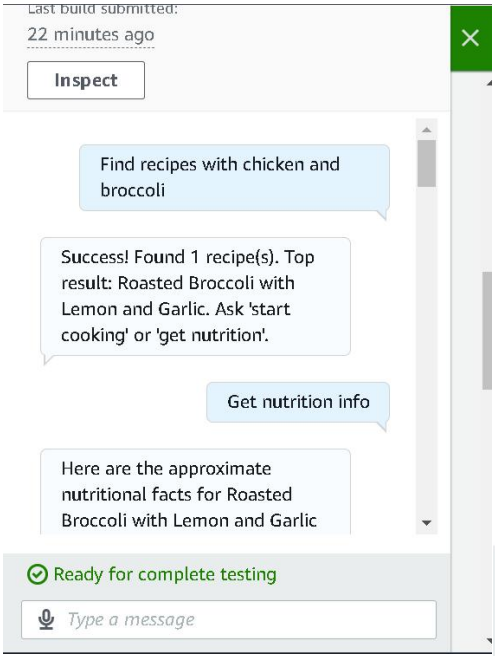

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/alias/TSTALIASID
aws
lambda
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Bots
Bot templates New
Networks of bots New
VoiceRecipeAssistant
Bot versions
Draft version
All languages
English (US)
Intents
Slot types
Deployment
Aliases
Channel integrations
Successfully built language English (US) in bot: VoiceRecipeAssistant
Language
Status
English (US)
Successfully built
Conversation logs Info
Enable logging to record details of conversations with your bot. You can enable a
Conversation logs is disabled
Manage conversation logs
Tags (0) Info
Key
Value
Last build submitted:
20 minutes ago
Inspect
Sorry, I couldn't find any recipes matching paneer onion tomato (for your vegan diet) (avoiding dairy,eggs) after filtering.
find recipes with chicken and broccoli
Success! I found 3 valid recipes (for your vegan diet) (avoiding dairy,eggs). The top result is: Roasted Broccoli with Lemon and Garlic. You can ask me to 'start cooking'.
Ready for complete testing
Type a message
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
00:14
28-10-2025

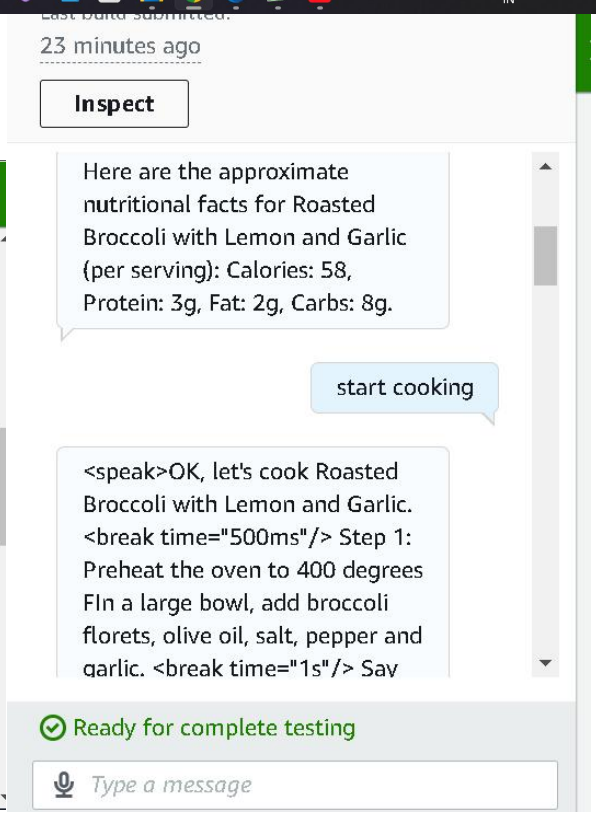

find recipes with paneer
Sorry, I couldn't find any recipes matching paneer (for your vegan diet) (avoiding dairy,eggs) after filtering.
Ready for complete testing
Type a message
Services, Inc. or its affiliates.
Privacy
Terms
Cookie

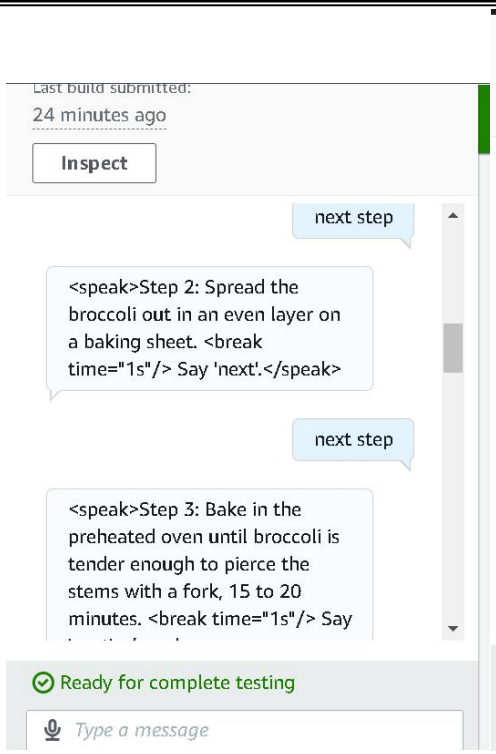

Amazon Lex | us-east-1
us-east-1.console.aws.amazon.com/lexv2/home?region=us-east-1#bot/CHL3QXZFVQ/locale/en_US/intent/4UDPCQ1SPU
Search [Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Lex
Back to intents list (6)
Search
Sort by last updated
GetNutrition Unsaved
StartCooking
NextStep
UpdateProfile
SearchRecipes
FallbackIntent
Draft version
English (US)
Successfully built
English (US) has not built changes.
Build
Test
Filter
Sort by added (ascending)
Preview
Plain text
What are the nutritional facts?
Show nutrition
How many calories?
Get nutrition info
Nutrition
Editor
Visual builder
New
Save intent
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
09:18
28-10-2025

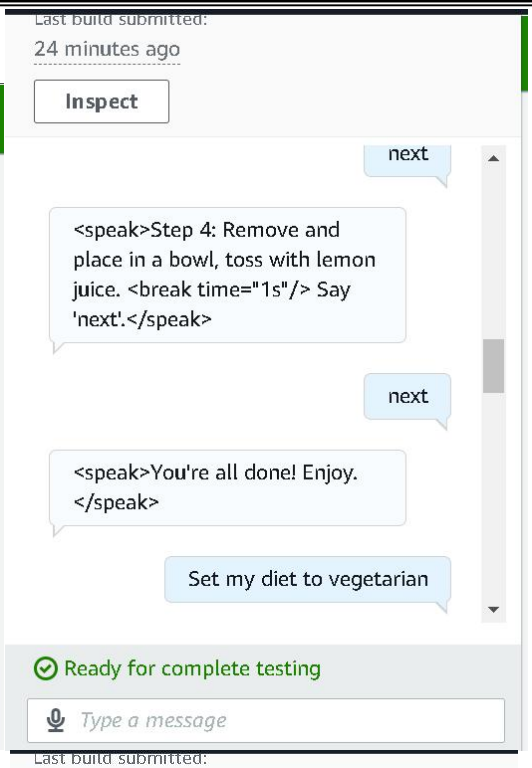

Amazon Lex | us-east-1
Add Permissions | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/VoiceRecipeAssistant-Logic-role-glp9tcsr/attach-policies
Search [Alt+S]
Global
Account ID: 6573-4390-4644
recipe-developer
IAM > Roles > VoiceRecipeAssistant-Logic-role-glp9tcsr > Add Permissions
Attach policy to VoiceRecipeAssistant-Logic-role-glp9tcsr
Current permissions policies (2)
Other permissions policies (1/1076)
Filter by Type
polly
All types
2 matches
1
Policy name
Type
Description
AmazonPollyFullAccess
AWS managed
Grants full access to Amazon Polly ser...
AmazonPollyReadOnlyAccess
AWS managed
Grants read-only access to Amazon Po...
Cancel
Add permissions
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
09:24
28-10-2025

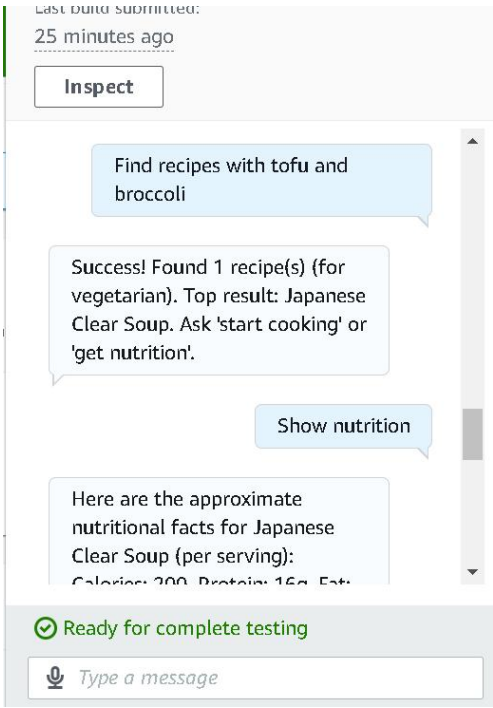

Amazon Lex | us-east-1
VoiceRecipeAssistant-Logic-rol
(20) Messaging | LinkedIn
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/VoiceRecipeAssistant-Logic-role-glp9tcsr?section=permissions
Search
[Alt+S]
Global
Account ID: 6573-4390-4644
recipe-developer
IAM > Roles > VoiceRecipeAssistant-Logic-role-glp9tcsr
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Root access management
Access reports
Policy was successfully attached to role.
VoiceRecipeAssistant-Logic-role-glp9tcsr Info
Delete
Summary
Edit
Creation date
October 27, 2025, 13:41 (UTC+05:30)
ARN
arn:aws:iam::657343904644:role/service-role/VoiceRecipeAssistant-Logic-role-glp9tcsr
Last activity
9 hours ago
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Last Accessed
Revoke sessions
Permissions policies (3) Info
Simulate
Remove
Add permissions
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG IN
09:27
28-10-2025

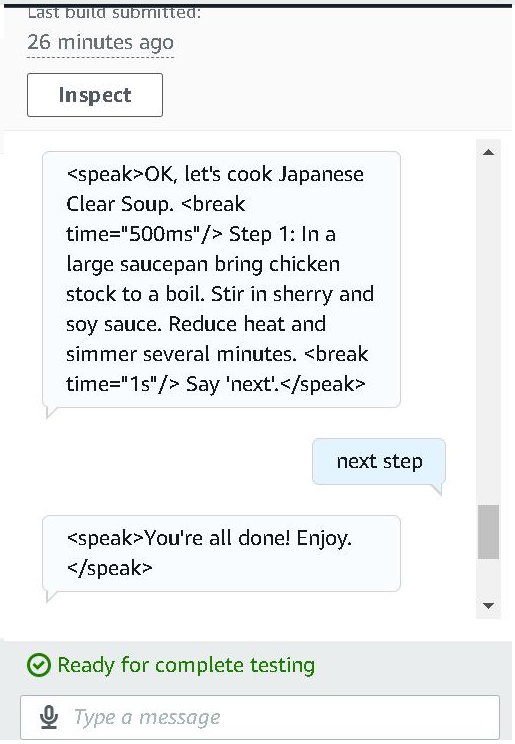

Last build submitted:
22 minutes ago
Inspect
Find recipes with chicken and broccoli
Success! Found 1 recipe(s). Top result: Roasted Broccoli with Lemon and Garlic. Ask 'start cooking' or 'get nutrition'.
Get nutrition info
Here are the approximate nutritional facts for Roasted Broccoli with Lemon and Garlic
Ready for complete testing
Type a message

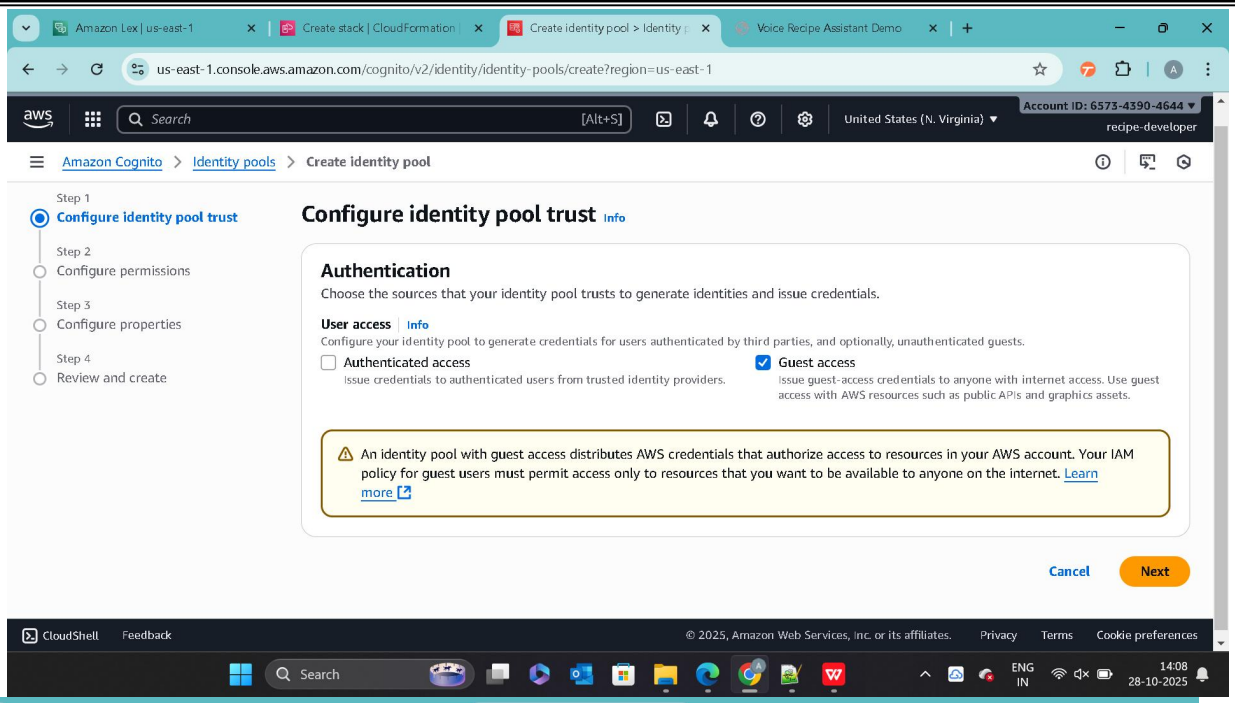

23 minutes ago
Inspect
Here are the approximate nutritional facts for Roasted Broccoli with Lemon and Garlic (per serving): Calories: 58, Protein: 3g, Fat: 2g, Carbs: 8g.
start cooking
<speak>OK, let's cook Roasted Broccoli with Lemon and Garlic. <break time="500ms"/> Step 1: Preheat the oven to 400 degrees FIn a large bowl, add broccoli florets, olive oil, salt, pepper and garlic. <break time="1s"/> Say
Ready for complete testing
Type a message

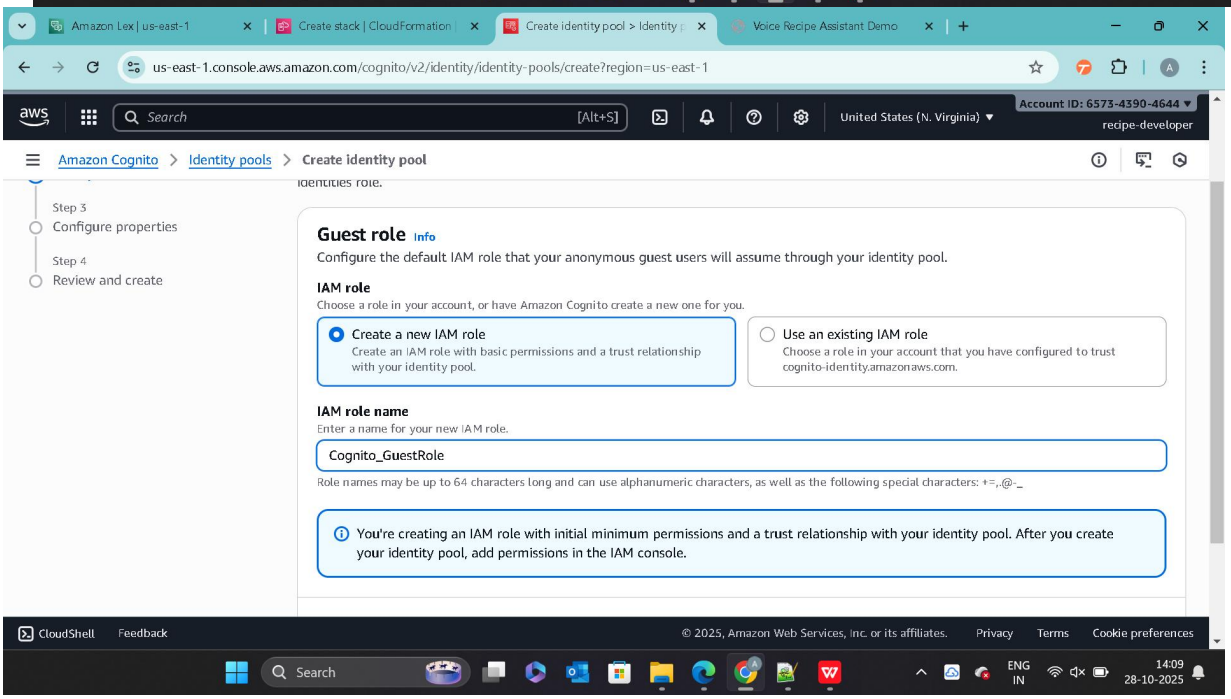

Last build submitted:
24 minutes ago
Inspect
next step
<speak>Step 2: Spread the broccoli out in an even layer on a baking sheet. <break time="1s"/> Say 'next'.</speak>
next step
<speak>Step 3: Bake in the preheated oven until broccoli is tender enough to pierce the stems with a fork, 15 to 20 minutes. <break time="1s"/> Say
Ready for complete testing
Type a message

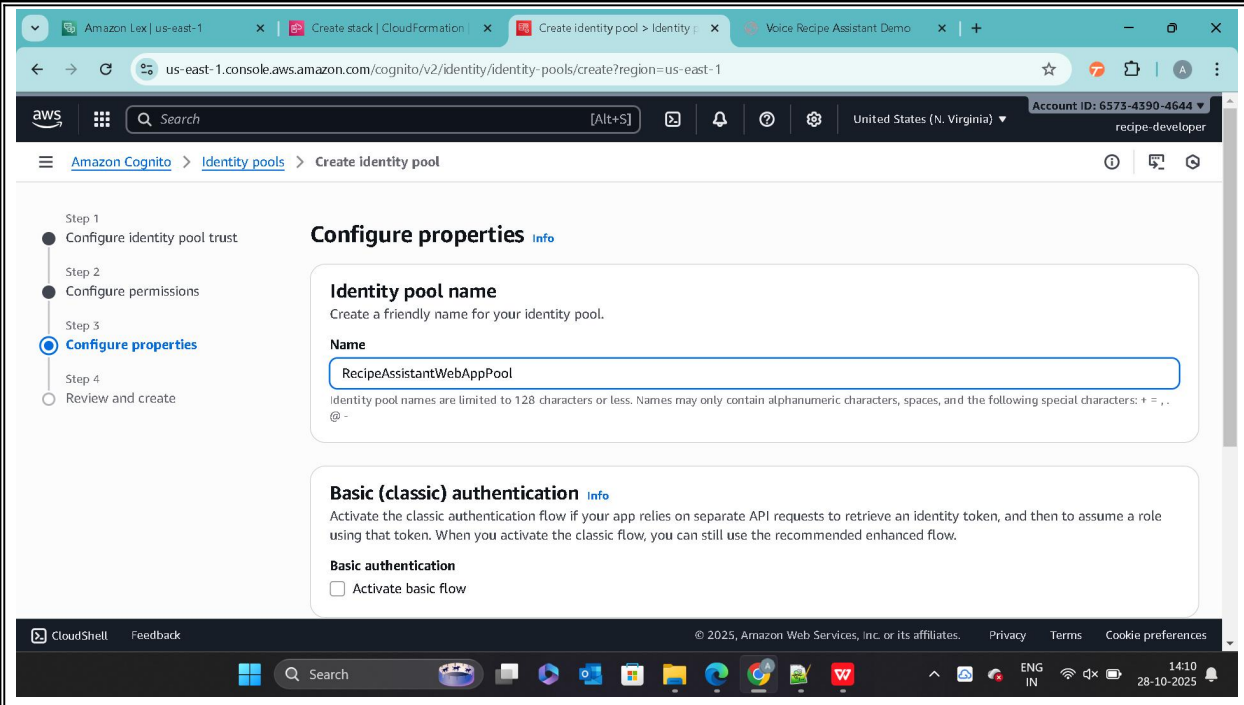

Last build submitted:
24 minutes ago
Inspect
next
<speak>Step 4: Remove and place in a bowl, toss with lemon juice. <break time="1s"/> Say 'next'.</speak>
next
<speak>You're all done! Enjoy. </speak>
Set my diet to vegetarian
Ready for complete testing
Type a message

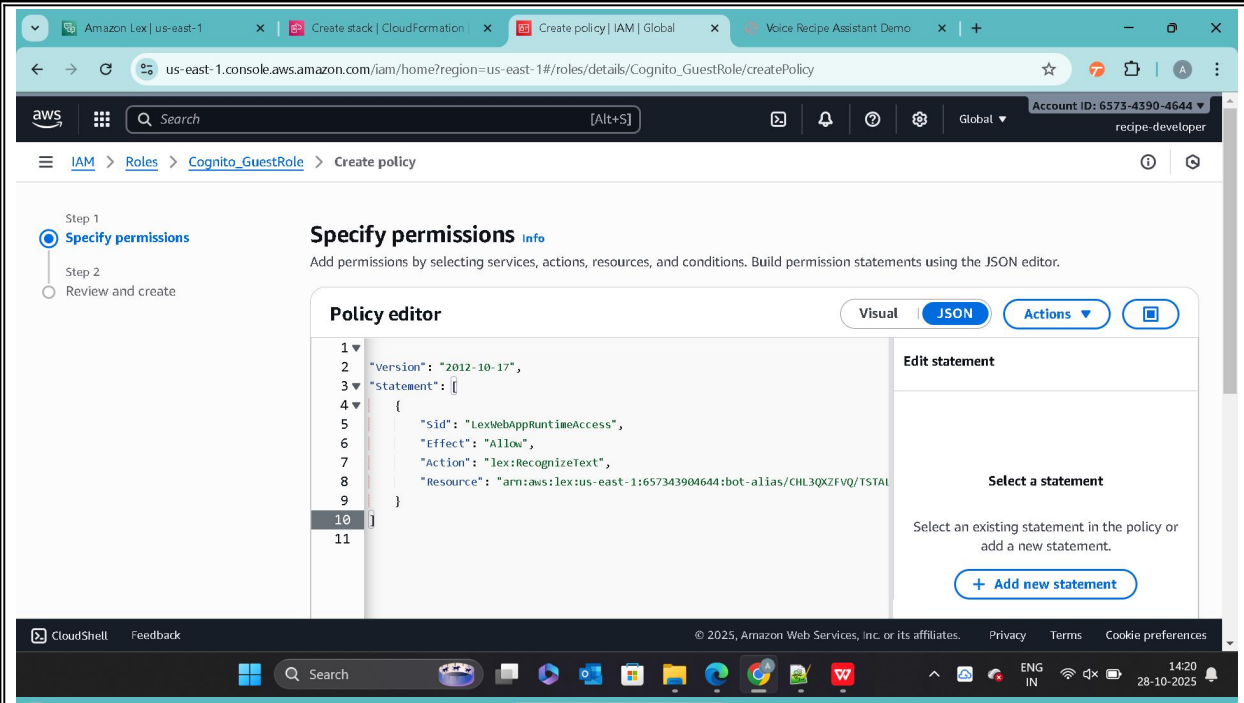

Last build submitted:
25 minutes ago
Inspect
Find recipes with tofu and broccoli
Success! Found 1 recipe(s) (for vegetarian). Top result: Japanese Clear Soup. Ask 'start cooking' or 'get nutrition'.
Show nutrition
Here are the approximate nutritional facts for Japanese Clear Soup (per serving):
Ready for complete testing
Type a message

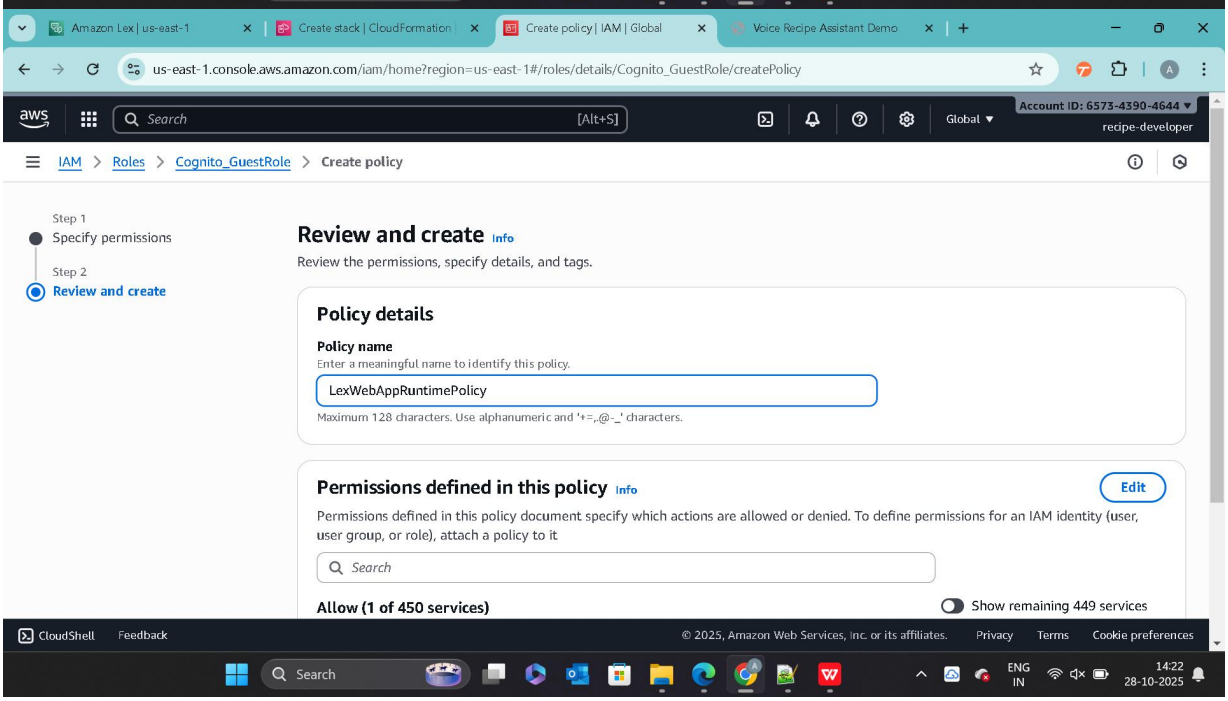

Last build submitted:
26 minutes ago
Inspect
<speak>OK, let's cook Japanese Clear Soup. <break time="500ms"/> Step 1: In a large saucepan bring chicken stock to a boil. Stir in sherry and soy sauce. Reduce heat and simmer several minutes. <break time="1s"/> Say 'next'.</speak>
next step
<speak>You're all done! Enjoy. </speak>
Ready for complete testing
Type a message

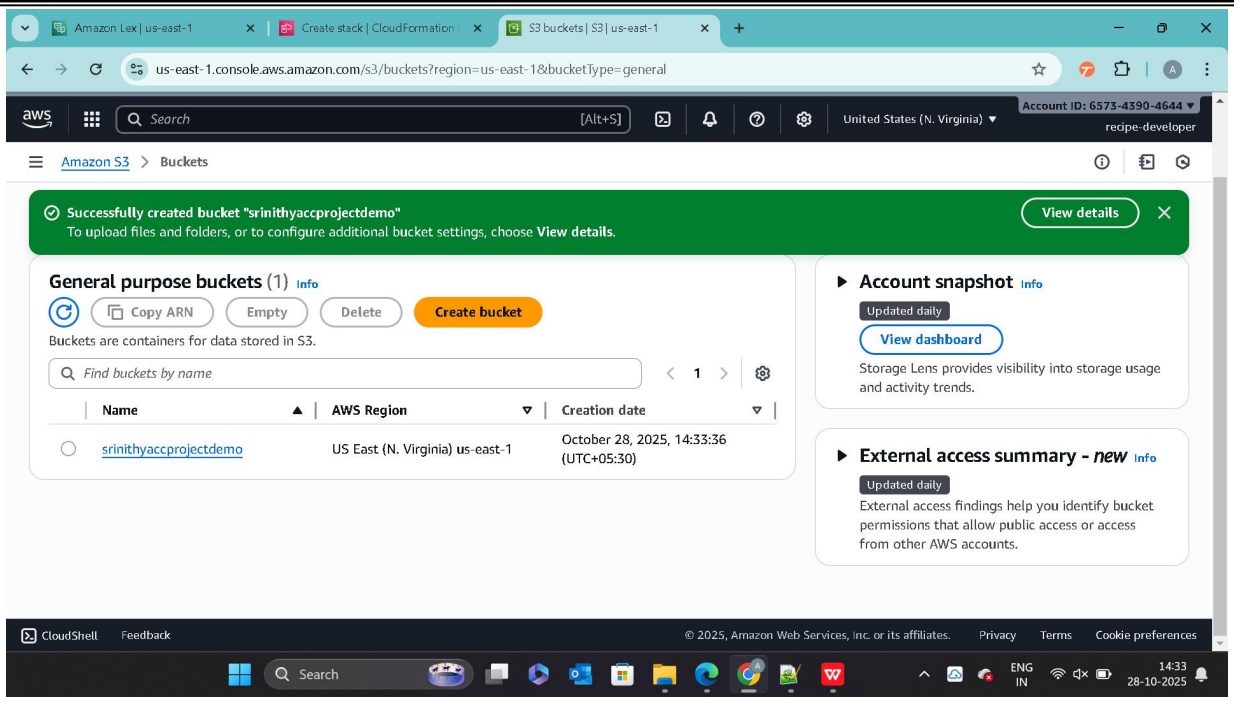

Amazon Lex | us-east-1
Create stack | CloudFormation
Create identity pool > Identity
Voice Recipe Assistant Demo
us-east-1.console.aws.amazon.com/cognito/v2/identity/identity-pools/create?region=us-east-1
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Cognito > Identity pools > Create identity pool
Step 1
Configure identity pool trust
Step 2
Configure permissions
Step 3
Configure properties
Step 4
Review and create
Configure identity pool trust Info
Authentication
Choose the sources that your identity pool trusts to generate identities and issue credentials.
User access Info
Configure your identity pool to generate credentials for users authenticated by third parties, and optionally, unauthenticated guests.
Authenticated access
Issue credentials to authenticated users from trusted identity providers.
Guest access
Issue guest-access credentials to anyone with internet access. Use guest access with AWS resources such as public APIs and graphics assets.
An identity pool with guest access distributes AWS credentials that authorize access to resources in your AWS account. Your IAM policy for guest users must permit access only to resources that you want to be available to anyone on the internet. Learn more
Cancel
Next
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:08
28-10-2025


Amazon Lex | us-east-1
Create stack | CloudFormation
Create identity pool > Identity
Voice Recipe Assistant Demo
us-east-1.console.aws.amazon.com/cognito/v2/identity/identity-pools/create?region=us-east-1
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Cognito > Identity pools > Create identity pool
Step 3
Configure properties
Step 4
Review and create
Guest role Info
Configure the default IAM role that your anonymous guest users will assume through your identity pool.
IAM role
Choose a role in your account, or have Amazon Cognito create a new one for you.
Create a new IAM role
Create an IAM role with basic permissions and a trust relationship with your identity pool.
Use an existing IAM role
Choose a role in your account that you have configured to trust cognito-identity.amazonaws.com.
IAM role name
Enter a name for your new IAM role.
Cognito_GuestRole
Role names may be up to 64 characters long and can use alphanumeric characters, as well as the following special characters: +=,.@-_
You're creating an IAM role with initial minimum permissions and a trust relationship with your identity pool. After you create your identity pool, add permissions in the IAM console.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:09
28-10-2025

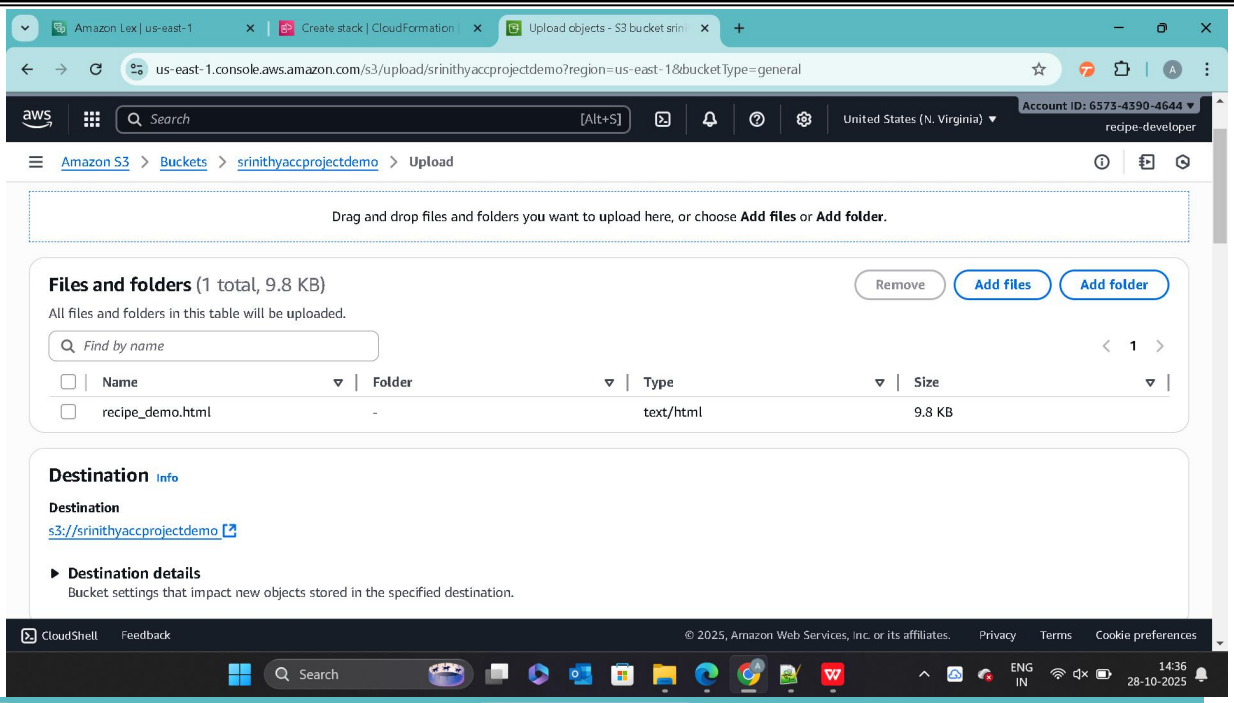

Amazon Lex | us-east-1
Create stack | CloudFormation
Create identity pool > Identity
Voice Recipe Assistant Demo
us-east-1.console.aws.amazon.com/cognito/v2/identity/identity-pools/create?region=us-east-1
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon Cognito > Identity pools > Create identity pool
Step 1
Configure identity pool trust
Step 2
Configure permissions
Step 3
Configure properties
Step 4
Review and create
Configure properties Info
Identity pool name
Create a friendly name for your identity pool.
Name
RecipeAssistantWebAppPool
Identity pool names are limited to 128 characters or less. Names may only contain alphanumeric characters, spaces, and the following special characters: + = , . @ -
Basic (classic) authentication Info
Activate the classic authentication flow if your app relies on separate API requests to retrieve an identity token, and then to assume a role using that token. When you activate the classic flow, you can still use the recommended enhanced flow.
Basic authentication
Activate basic flow
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:10
28-10-2025

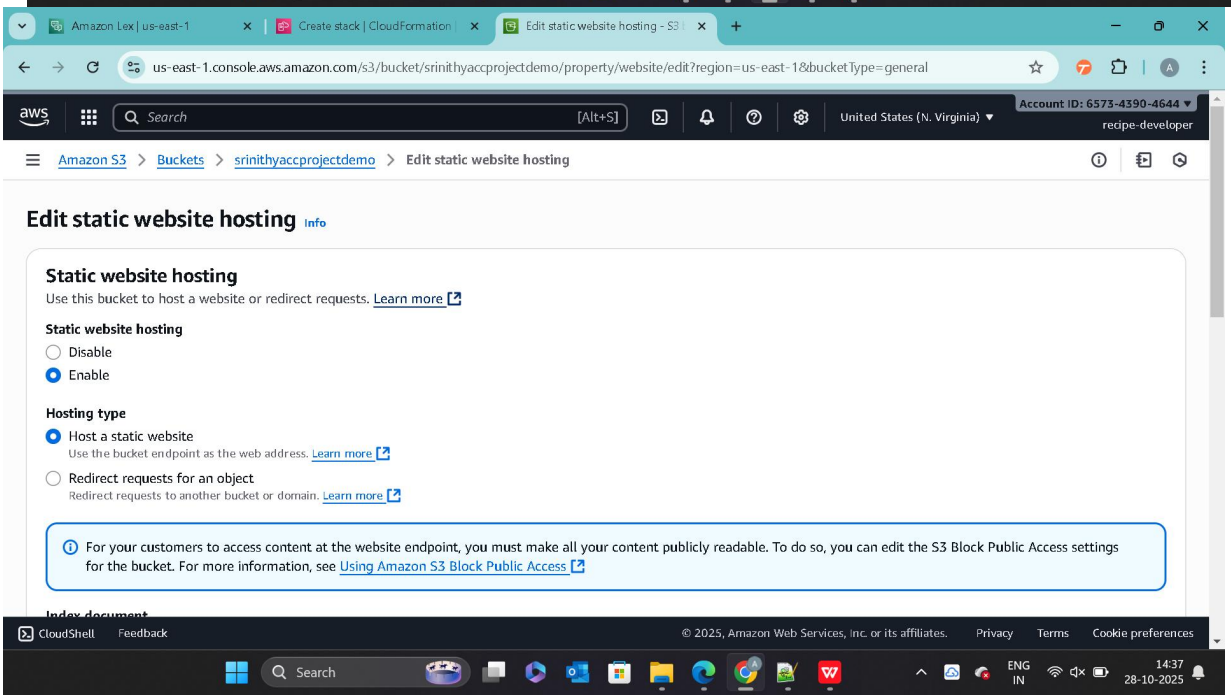

Amazon Lex | us-east-1
Create stack | CloudFormation
Create policy | IAM | Global
Voice Recipe Assistant Demo
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/Cognito_GuestRole/createPolicy
Search
[Alt+S]
Global
Account ID: 6573-4390-4644
recipe-developer
IAM > Roles > Cognito_GuestRole > Create policy
Step 1
Specify permissions
Step 2
Review and create
Specify permissions Info
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
1 {
2 "Version": "2012-10-17",
3 "Statement": [
4 {
5 "Sid": "LexWebAppRuntimeAccess",
6 "Effect": "Allow",
7 "Action": "lex:RecognizeText",
8 "Resource": "arn:aws:lex:us-east-1:657343904644:bot-alias/CHL3QXZFVQ/TSTAL
9 }
10]
11
Edit statement
Select a statement
Select an existing statement in the policy or add a new statement.
+ Add new statement
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:20
28-10-2025

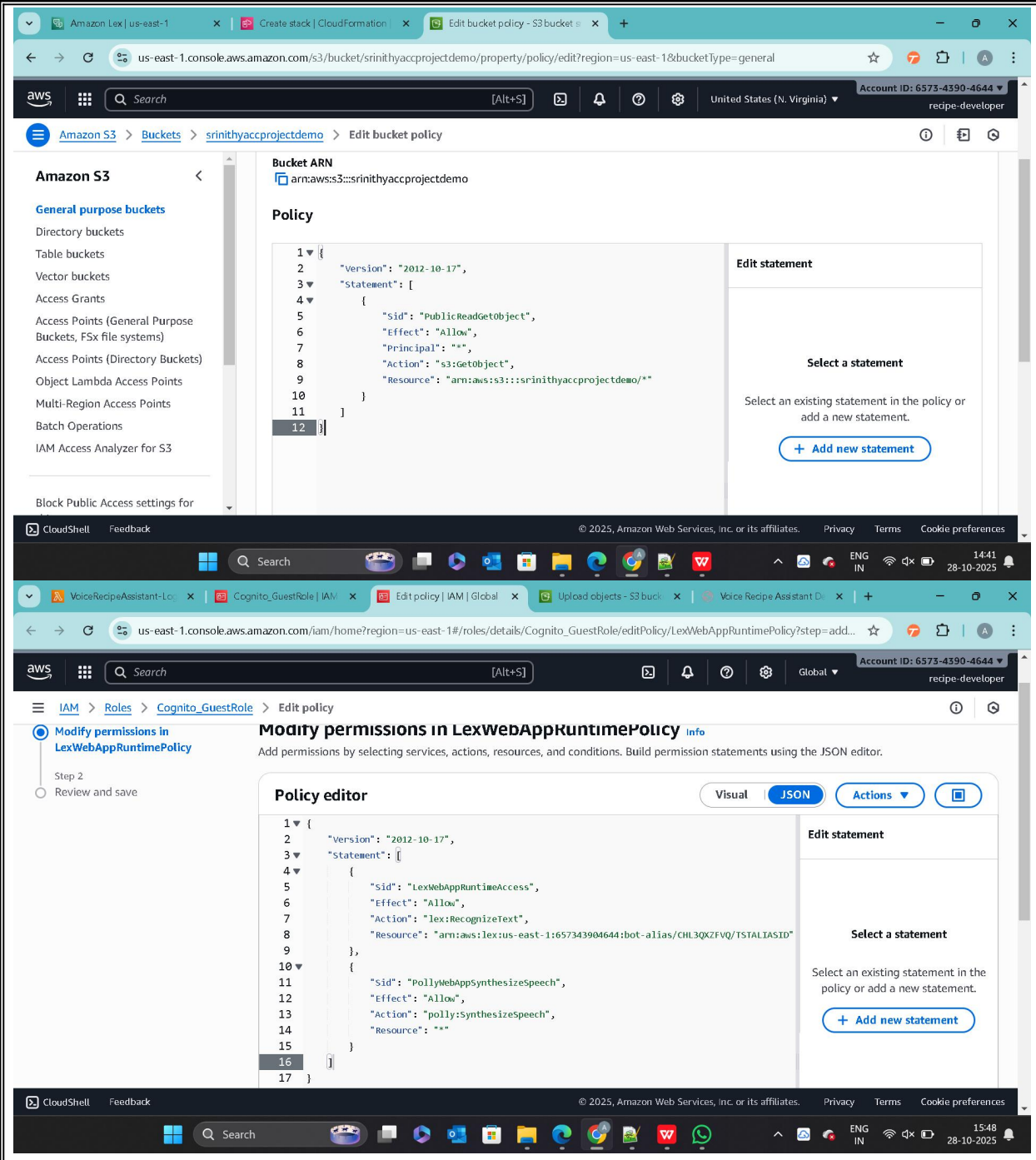

Amazon Lex | us-east-1
Create stack | CloudFormation
Create policy | IAM | Global
Voice Recipe Assistant Demo
us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/details/Cognito_GuestRole/createPolicy
Search
[Alt+S]
Global
Account ID: 6573-4390-4644
recipe-developer
IAM > Roles > Cognito_GuestRole > Create policy
Step 1
Specify permissions
Step 2
Review and create
Review and create Info
Review the permissions, specify details, and tags.
Policy details
Policy name
Enter a meaningful name to identify this policy.
LexWebAppRuntimePolicy
Maximum 128 characters. Use alphanumeric and '+=,.@-_' characters.
Permissions defined in this policy Info
Edit
Permissions defined in this policy document specify which actions are allowed or denied. To define permissions for an IAM identity (user, user group, or role), attach a policy to it
Search
Allow (1 of 450 services)
Show remaining 449 services
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:22
28-10-2025



Upload html according to your style


Amazon Lex | us-east-1
Create stack | CloudFormation
Upload objects - S3 bucket srin
us-east-1.console.aws.amazon.com/s3/upload/srinithyaccprojectdemo?region=us-east-1&bucketType=general
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon S3 > Buckets > srinithyaccprojectdemo > Upload
Drag and drop files and folders you want to upload here, or choose Add files or Add folder.
Files and folders (1 total, 9.8 KB)
Remove
Add files
Add folder
All files and folders in this table will be uploaded.
Find by name
Name
Folder
Type
Size
recipe_demo.html
-
text/html
9.8 KB
Destination Info
Destination
s3://srinithyaccprojectdemo
Destination details
Bucket settings that impact new objects stored in the specified destination.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:36
28-10-2025


Amazon Lex | us-east-1
Create stack | CloudFormation
Edit static website hosting - S3
us-east-1.console.aws.amazon.com/s3/bucket/srinithyaccprojectdemo/property/website/edit?region=us-east-1&bucketType=general
Search
[Alt+S]
United States (N. Virginia)
Account ID: 6573-4390-4644
recipe-developer
Amazon S3 > Buckets > srinithyaccprojectdemo > Edit static website hosting
Edit static website hosting Info
Static website hosting
Use this bucket to host a website or redirect requests. Learn more
Static website hosting
Disable
Enable
Hosting type
Host a static website
Use the bucket endpoint as the web address. Learn more
Redirect requests for an object
Redirect requests to another bucket or domain. Learn more
For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access
Index document
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
ENG
IN
14:37
28-10-2025



6. Working Applications & Features

Core Project Features

Natural Language Conversation: Users can type requests in plain English (e.g., "what can I make with chicken" or "get me a pasta recipe") instead of clicking buttons.

Real-Time Voice Feedback: Every bot response is automatically converted to speech, making it a true "voice" assistant and ideal for hands-free kitchen use.

Dynamic Recipe Search: Connects to the Spoonacular API to search a database of over 300,000+ recipes. The app is not limited to a static or hard-coded list.

Rich Content Display: The interface displays not just text but also relevant recipe images, making the experience more engaging.

Fully Serverless & Scalable: The entire architecture runs without any dedicated servers. It can handle one user or one million users without any code changes and only incurs costs based on actual usage.

Working Applications (Use Cases)

Hands-Free Kitchen Companion: The primary use case. A user can be cooking and ask for the next step, a measurement conversion, or an ingredient substitution without having to stop and wash their hands to touch a screen.

Recipe Discovery & Meal Planning: Helps users find new recipes based on ingredients they already have ("What can I make with beef and broccoli?") or based on dietary needs.

Accessibility Tool: The strong voice-first design makes this an excellent tool for users with visual impairments, allowing them to browse and follow recipes audibly.

Dietary & Allergy Management: The Lambda function can be easily modified to include Spoonacular's powerful filters for dietary restrictions (e.g., diet=vegan, intolerances=gluten).

Educational Cooking Tool: The assistant can be used to teach cooking basics, explain complex culinary terms, or provide step-by-step guidance.

6. Conclusion & Future Scope

Conclusion

This project successfully demonstrates the power of "stitching together" multiple managed services to create a complex, intelligent, and feature-rich application. By integrating Amazon Lex for conversational AI, Amazon Polly for voice, AWS Lambda for logic, and the Spoonacular API for data, we built a "Voice Recipe Assistant" that is both highly functional and user-friendly.

The serverless architecture proves to be the ideal model for this type of application, offering immense scalability, low operational overhead, and a cost-effective pay-per-use model. This project highlights that the future of application development lies not just in writing code, but in intelligently connecting powerful, pre-built services.

7.Future Scope

The platform is built on a foundation that is perfect for expansion. Future improvements could include:

Full Voice-to-Voice: Integrating Amazon Transcribe to convert the user's spoken words into text. This would complete the loop (Speech-to-Text -> Lex -> Lambda -> Polly -> Text-to-Speech) for a truly hands-free experience.

User Profiles & Favorites: Using Amazon DynamoDB (a NoSQL database) to store user profiles, allowing them to save their favorite recipes or set permanent dietary preferences.

Step-by-Step Cooking Mode: Enhancing the Lex bot and Lambda function to guide a user through a recipe one step at a time, waiting for a "next step" command from the user.

Advanced Spoonacular Integration: Expanding the Lambda function to use more of the Spoonacular API's features, such as finding recipes by nutritional content (e.g., "find a high-protein breakfast"), substituting ingredients, or pairing wine with a dish.

8. References

Spoonacular API Documentation: https://spoonacular.com/food-api/docs

Amazon Lex V2 Developer Guide: https://docs.aws.amazon.com/lexv2/latest/dg/what-is.html

Amazon Polly Developer Guide: https://docs.aws.amazon.com/polly/latest/dg/what-is.html

AWS Lambda Developer Guide: https://docs.aws.amazon.com/lambda/latest/dg/welcome.html

Amazon Cognito Developer Guide: https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html

Amazon S3 User Guide (Static Hosting): https://docs.aws.amazon.com/s3/latest/userguide/WebsiteHosting.html

AWS SDK for JavaScript Developer Guide: https://docs.aws.amazon.com/sdk-for-javascript/v2/developer-guide/welcome.html

Output website: