

TECHNICAL HUB CHATBOT

TH – BOT



Problem Statement:

The TH-Bot website project aims to develop an interactive, web-based chatbot interface that integrates with Amazon Lex to provide intelligent, real-time conversations with users. The chatbot is accessible through a user-friendly frontend built using HTML, CSS, and JavaScript, and communicates with the Lex bot via Amazon API Gateway, optionally using AWS Lambda for backend logic. The system ensures secure access through IAM roles and policies, and supports basic user queries such as greetings, FAQs, and informational responses. The goal is to deliver a responsive, cloud-powered chatbot experience that enhances user engagement and support through natural language interaction.

Team Members:

23MH1A05K1 – P. Nithya Santhoshi

23MH1A05N8 – N. Yuktha Sri

23MH1A05O0 – P. Harshini

23MH1A0524 – K. Bhavani Surya Prabha

23P31A0543 – N. Suvarna Ratnam

23P31A0565 – Y. Sri Veni

Table of Contents:

Content	Page No:
1. Abstract	3
2. Purpose	3
3. Scope	4
4. Tools and Services Used	4-5
5. Work Flow	6
6. Architecture	7-8
7. Approach	9-18
8. Benefits	18-19
9. Future Enhancements	19-20
10. Team Photos	21-23

◆ Abstract

- The Technical Hub Chatbot is a smart, user-friendly virtual assistant developed specifically for the Technical Hub platform. It acts as a guide for students, visitors, and other users by providing instant answers to common questions, mentor details, available technologies, certification info, event updates, and course registration assistance.
- This chatbot is designed to improve the overall user experience by making information easily accessible without the need for human intervention. Instead of manually searching through different pages or waiting for someone to respond, users can simply ask the chatbot and get real-time, accurate answers within seconds.
- Whether someone wants to know about the certifications available in the second year, the name of a mentor, upcoming tech events, or how to register for a course—this bot has it covered.

◆ Purpose

- The main purpose of the TH-Bot project is to create a smart chatbot that can chat with users through a website and provide quick, helpful responses.
- Many users visit websites with questions or need support, and waiting for a human to reply can be slow.
- This chatbot solves that problem by instantly responding to users in a friendly and intelligent way. It's built using Amazon Lex, which understands natural language, so users can talk to the bot just like they would talk to a person.
- This reduces the workload on human support teams and gives users a better, faster experience.

◆ Scope

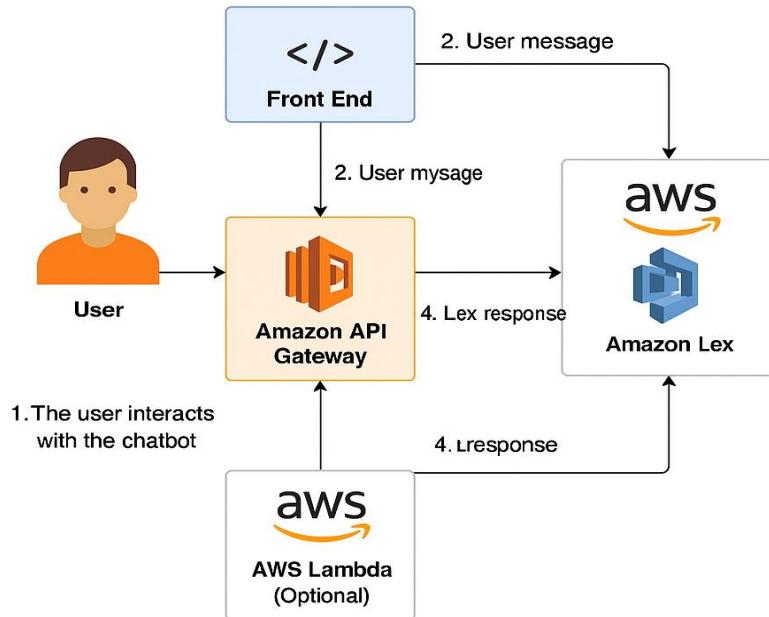
- The project involves creating a **simple and user-friendly chatbot interface** using **HTML, CSS, and JavaScript**.
- The chatbot is connected to **Amazon Lex** for processing user input and generating responses.
- **Amazon API Gateway** acts as a **bridge** between the frontend website and the backend chatbot logic.
- **AWS Lambda** can be optionally used to perform **custom backend operations** (e.g., fetching dynamic data).
- **AWS IAM** is used to **manage secure access** between all integrated AWS services.
- The chatbot is currently designed to handle **basic user queries** such as:
 - Greetings (e.g., “Hi”, “Hello”)
 - Help messages
 - Frequently Asked Questions (FAQs)
- The current version supports **text-based interaction only**.

◆ Tools and Services Used

- **Amazon Lex**
 - Powers the chatbot by understanding user questions and matching them to the right responses.
- **AWS Lambda**
 - Runs backend code (like fetching data or processing responses) without needing a server.
- **Amazon API Gateway**
 - Connects the frontend chatbot with backend services securely through APIs.

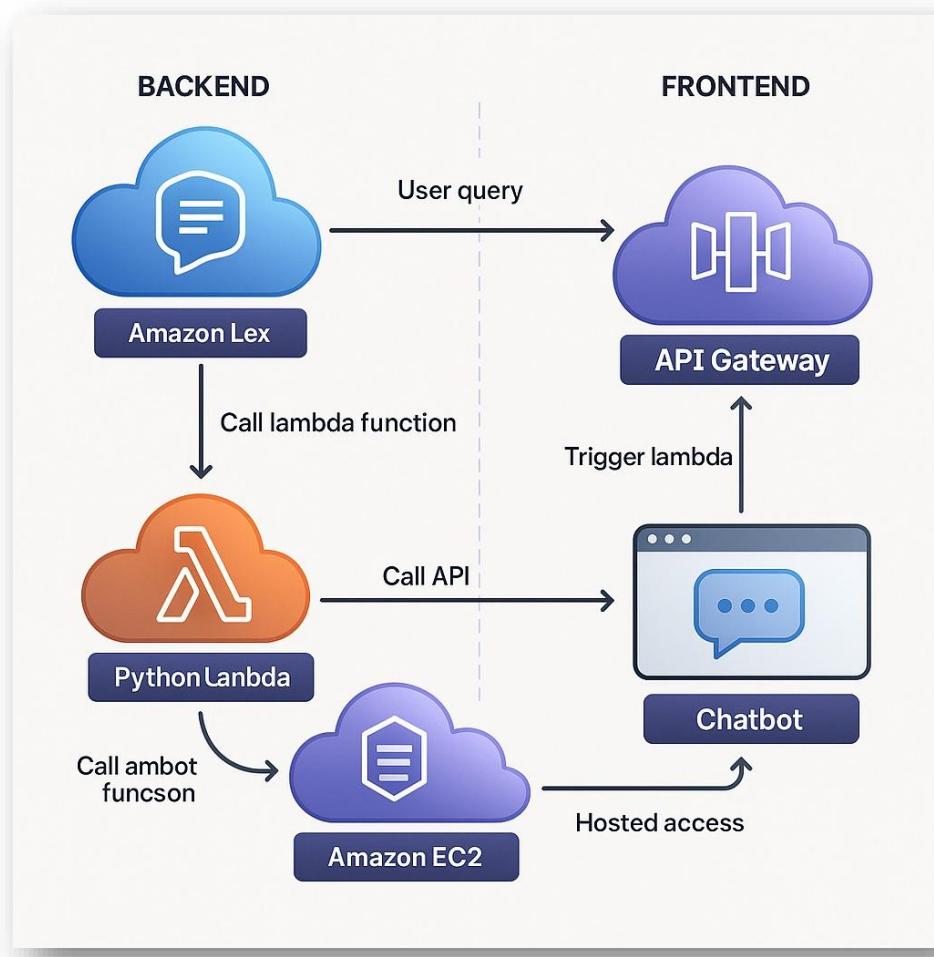
- **AWS IAM**
 - Manages roles and permissions to make sure only authorized components access the system.
- **Frontend (HTML/CSS/JavaScript or React)**
 - Creates the chatbot interface that users interact with on the website.
- **Amazon CloudWatch**
 - Monitors the system and logs user interactions to help fix issues and track performance.
- **AWS S3 (Optional)**
 - Hosts the chatbot frontend files (like HTML and JS) for easy access and deployment.

◆ Work Flow



- User opens the chatbot on the website.
- User types a question in the chatbox.
- The message is sent to API Gateway.
- API Gateway triggers a Lambda function.
- Lambda sends the message to Amazon Lex.
- Lex identifies the intent and prepares a reply.
- The reply goes back to Lambda.
- Lambda sends the final response to API Gateway.
- API Gateway returns the reply to the frontend.
- Chatbot displays the answer to the user

◆ Architecture



● Frontend Side (User Interaction)

1. Chatbot Interface

- This is the web-based chatbot (built with HTML/JavaScript) where users type their queries.
- It interacts with the backend through a REST API.

2. Amazon API Gateway

- Acts as a secure doorway between the chatbot frontend and backend AWS services.
- When a user submits a question, the API Gateway captures the request and sends it to the backend services securely.

● Backend Side (Processing the Query)

3. Amazon Lex

- This is the heart of your chatbot's brain.
- It understands what the user is saying (using Natural Language Processing) and figures out the intent.
- Based on the intent, it may respond directly or call a backend service for help.

4. AWS Lambda (Python Function)

- Amazon Lex invokes a Lambda function when it needs to do some logic or fetch data.
- This Lambda function is written in Python and acts as a middle layer.
- It processes the intent and performs actions like database queries, calculations, or sending data to other AWS services.

5. Amazon EC2 (Elastic Compute Cloud)

- If heavy lifting or more custom functionality is needed, Lambda calls an API hosted on EC2.
- EC2 runs your custom application and returns the result.
- This service gives your bot more power and flexibility to handle complex queries.

◆ Approach

To build the TH-Bot (AI chatbot for Technical Hub), a systematic approach was followed. Each step involved the use of specific AWS services and development techniques to ensure smooth interaction between the user and backend services.

1. Defining Intents and Utterances in Amazon Lex

We started by setting up the core of the chatbot:

- Created a new Amazon Lex bot named **TH - Bot**
- Defined **multiple intents** like greetings, course info, trainer details, certifications, etc.
- Added **sample utterances** for each intent to help Lex understand user queries.
- For some intents, slot types were also defined.

The screenshot shows the Amazon Lex Bots page. On the left, there's a sidebar with options like 'Bots', 'Bot templates', 'Networks of bots', 'Test workbench', 'Related resources', and a link to 'Return to the V1 console'. The main area has a heading 'Lex > Bots'. A prominent message box says 'Announcing Amazon Lex Generative AI features powered by Amazon Bedrock.' It lists several features: Descriptive Bot Builder, Assisted Slot Resolution, Sample Utterance Generation, AMAZON.QnAIntent, and Generative AI Bots. Below this, a table titled 'Bots (1) Info' shows one entry: 'TH-Bot' with status 'Available' and last updated '4 hours ago'. At the bottom, there's an 'Import/export history (0)' section with a table header 'Type', 'Bot', 'Status', 'Errors', 'Last updated', 'File', and 'Version'.

Intents (5) [Info](#)

An intent represents an action that the user wants to perform.

Name	Description	Last edited
Certifications	-	Now
TrainerDetails	-	Now
Courseinfo	-	1 minute ago
Greetings	-	1 minute ago
FallbackIntent	Default intent when no other intent matches	3 minutes ago

Successfully built language English (US) in bot: TH-Bot

Draft version English (US) Successfully built

Initial response [Info](#)

You can provide messages to acknowledge the user's initial request. You can also configure the next step in the conversation and branch.

[Editor](#) [Visual builder](#) [New](#)

2. Creating Lambda Function for Logic Handling

A backend Lambda function was created to:

- Handle complex logic and fetch dynamic responses.
- Process different inputs like trainer names, courses, or placement-related queries.
- Return structured responses to Lex in Python format.

Sri Veni ▾

Last fetched 0 seconds ago Actions ▾ Create function

Filter by attributes or search by keyword:

Function name	Description	Package type	Runtime	Last modified
TH-function	-	Zip	Python 3.13	1 hour ago

Successfully updated the function TH-function.

```

104
105     def lambda_handler(event, context):
106         try:
107             # Get user input from event
108             if 'body' in event:
109                 body = json.loads(event['body'])
110                 user_input = body.get('message', '')
111             elif 'inputtranscript' in event:
112                 user_input = event['inputtranscript']
113             else:
114                 return {"statusCode": 400, "body": json.dumps({"error": "No input provided"})}
115
116             # Use custom response logic
117             custom_response = generate_response(user_input)
118             if custom_response:
119                 return {
120                     "statusCode": 200,
121                     "headers": {"Access-Control-Allow-Origin": "*"},
122                     "body": json.dumps({"response": custom_response})
123                 }

```

EXPLORER

- λ TH-FUNCTION
- λ lambda_function.py

```

    4     def generate_response(user_input):
    5         # Technical Hub Information
    6         elif "technical hub" in user_input and (
    7             "what is" in user_input or "tell me" in user_input or "explain" in user_input or "about" in user_input):
    8             return "Technical Hub is an industry-focused training and certification platform providing global tech education and placement support for students and professionals. We offer courses in various fields like Cloud Computing, Data Science, Machine Learning, DevOps, Network Security, and more."
    9             return "Technical Hub offers training in Cloud (AWS, Azure), Networking (Cisco), Red Hat, DevOps, Data Science, and more."
   10            return "You can enroll by contacting our team, your college coordinator, or visiting our official website."
   11            return "Yes, we support placements with mock interviews, aptitude training, and company-specific preparation."
   12            return "ProjectSpace is a platform where students work on real-time industry projects under expert mentorship."
   13            elif "maya" in user_input and (
   14                "what is" in user_input or "explain" in user_input or "about" in user_input or "tell me" in user_input):
   15                return "MAYA is an AI-enabled learning platform for coding (Owl Code), aptitude, and technical interview prep."
   16            elif "location" in user_input or "where" in user_input:
   17                return "Technical Hub is based in Surampalem, Andhra Pradesh, serving learners across the country."
   18            elif "different" in user_input or "what makes" in user_input:
   19                return "We provide live project experience, global certification training, and tailored learning paths."
   20            elif "learning paths" in user_input or "path" in user_input:
   21                return "Our key learning paths include Owl Coder, T-Shaped Engineer, Drive Ready, and DevOps Cloud."
   22            elif "Owl coder" in user_input:
   23                return "Owl Coder builds mastery in programming, DSA, and system design with hands-on coding."

```

DIRECTORY

- λ Deploy (Ctrl+Shift+U)
- λ Test (Ctrl+Shift+I)

TEST EVENTS [SELECTED: THUBTEST]

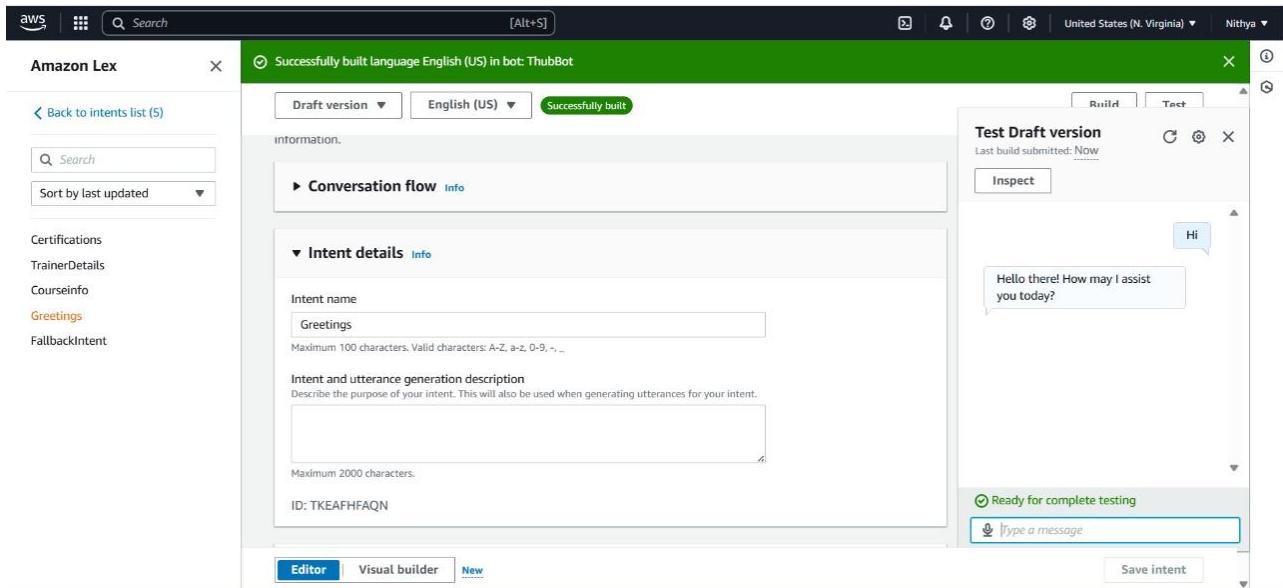
- + Create new test event
- ▽ Private saved events

CloudShell Feedback

3. Connecting Lex to Lambda

The Lambda function was integrated with the Lex bot:

- Each intent was linked to the Lambda function using fulfillment.
- This allowed Lex to call the function after recognizing the intent.



4. Creating API Gateway for Frontend Integration

To connect the frontend to backend:

- An **Amazon API Gateway** was created.
- Configured to trigger the Lambda function when users send messages.
- CORS was enabled to allow requests from browser-based frontend.

Screenshot of the AWS API Gateway 'APIs' page showing a successful API creation.

APIs (1/1)

Name	Description	ID	Protocol	API endpoint type	Created
TH-API		hnprj0wvp2	REST	Regional	2025-08-04

Actions: Delete, Create API

Screenshot of the AWS API Gateway 'Resources' page for the 'Thubapi' API.

Resource details:

Path	Resource ID
/	a878eyyqoc

Methods (2):

Method type	Integration type	Authorization	API key
OPTIONS	Mock	None	Not required
POST	Lambda	None	Not required

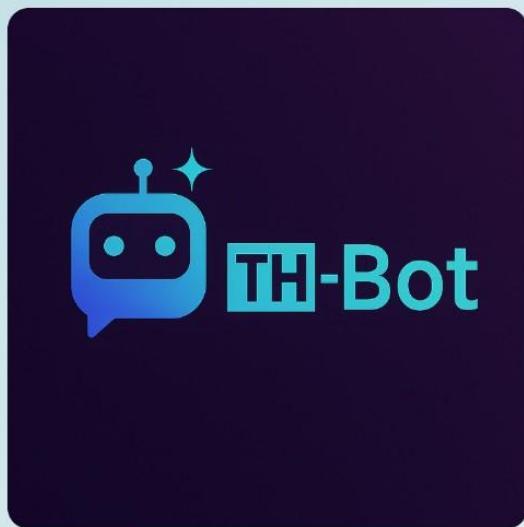
The screenshot shows the AWS Lambda function configuration page for 'TH-function'. The function is triggered by an API Gateway. The code source is a file named 'lambda_function.py'. The function ARN is arn:aws:lambda:us-east-1:767224848486:function:TH-function.

The screenshot shows the AWS API Gateway Stages page for 'Thubapi'. A deployment for stage 'stage' has been successfully created. Stage details include a Rate limit of 10000 and a Burst limit of 5000. The Invoke URL is https://xbwinpkw03.execute-api.us-east-1.amazonaws.com/stage.

5. Developing the Frontend (Chatbot UI)

We created a user-friendly chatbot interface using:

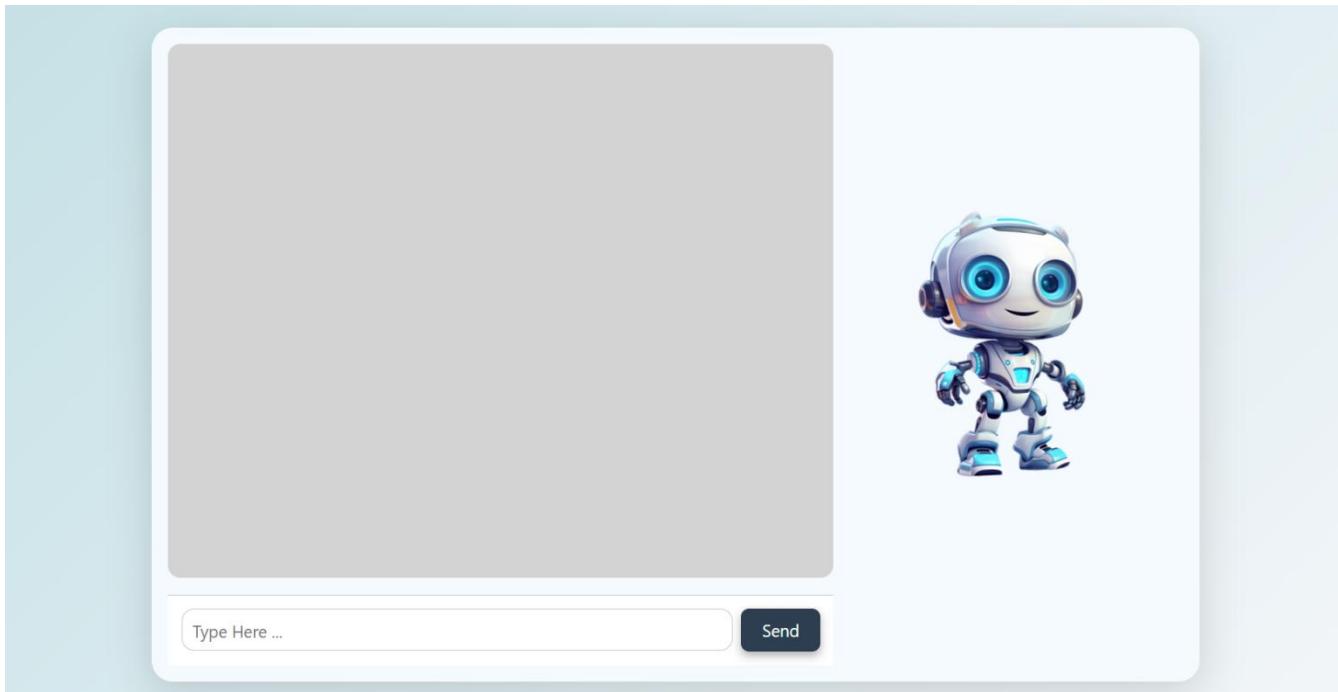
- HTML, CSS, and JavaScript.
- Input box to capture user queries.
- Display area to show chatbot responses dynamically.



Login to TH-Bot

[Forgot Password?](#)

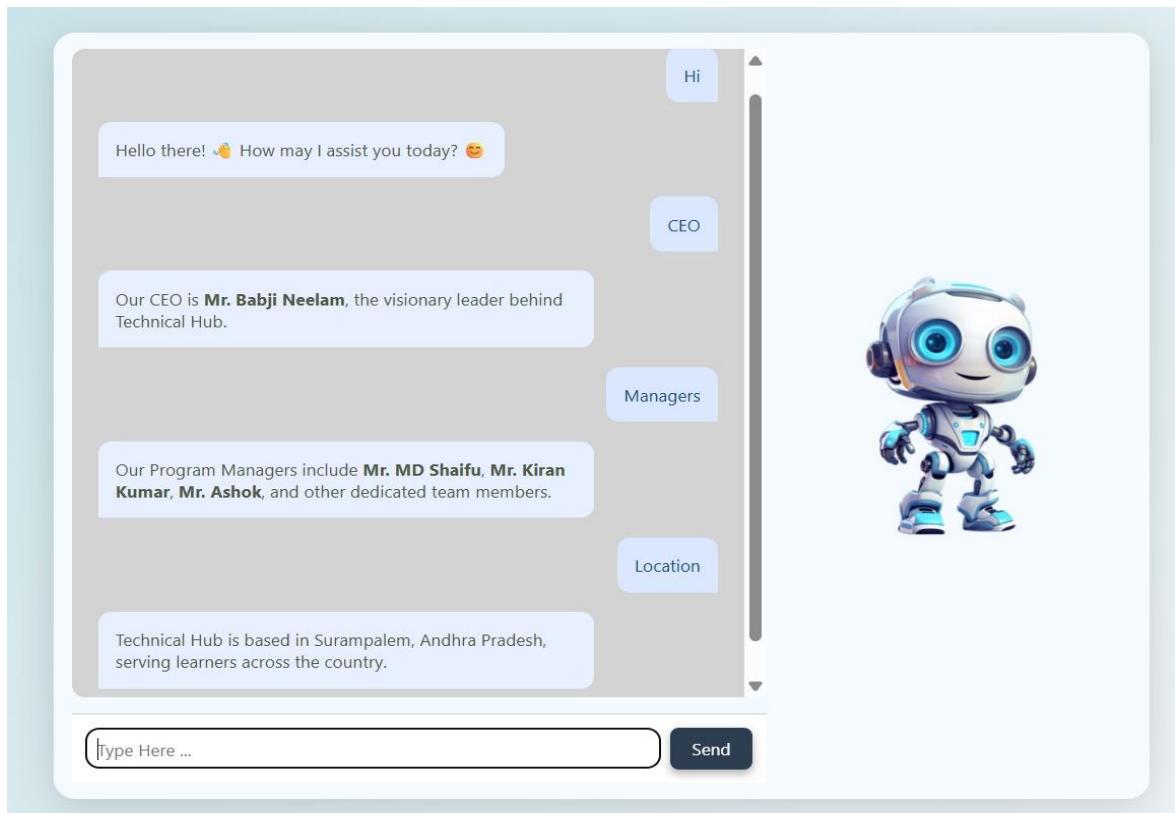
Login



6. Hosting Frontend (Optional)

The frontend was optionally hosted:

- Either locally or on **Amazon S3** for static hosting.
- API keys and endpoints were securely linked to the hosted site.

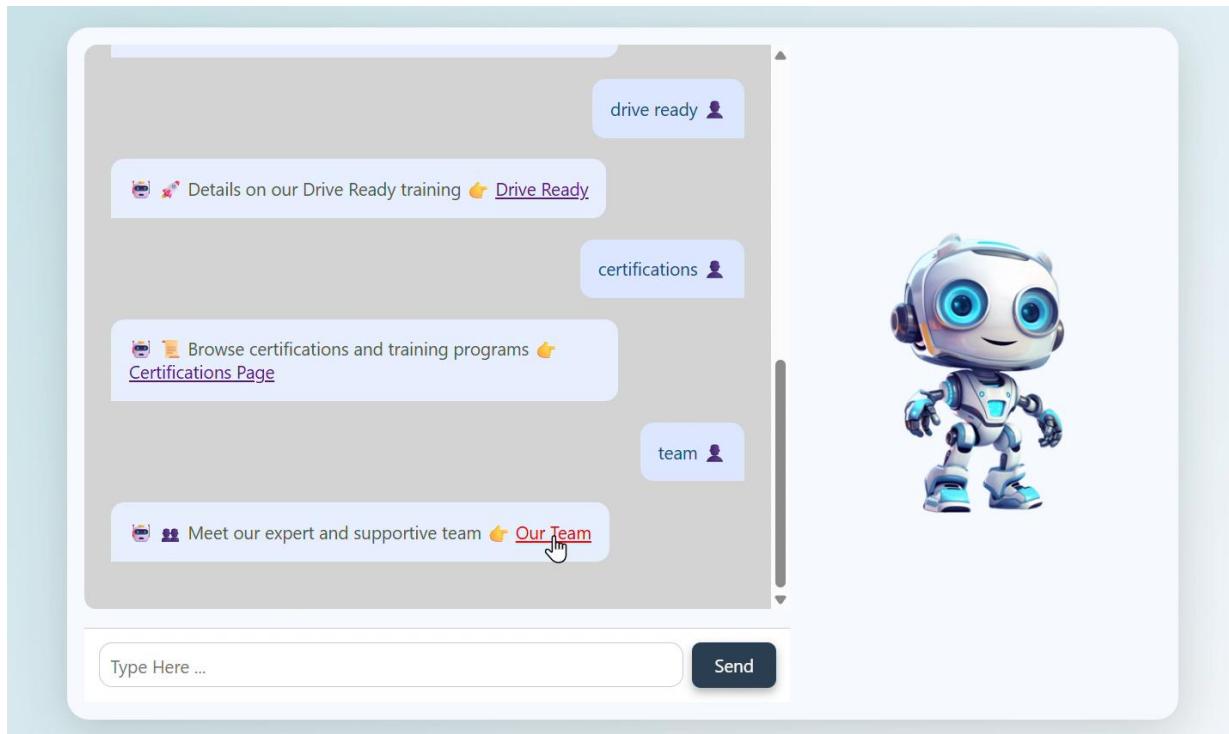
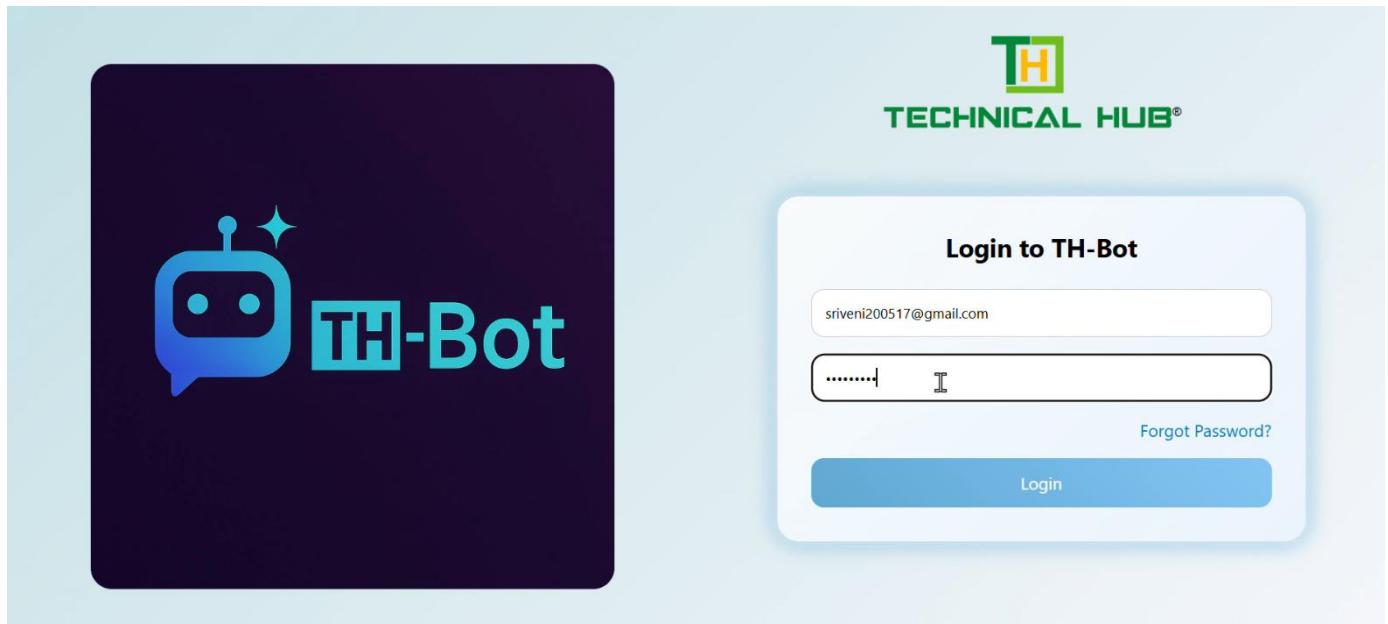


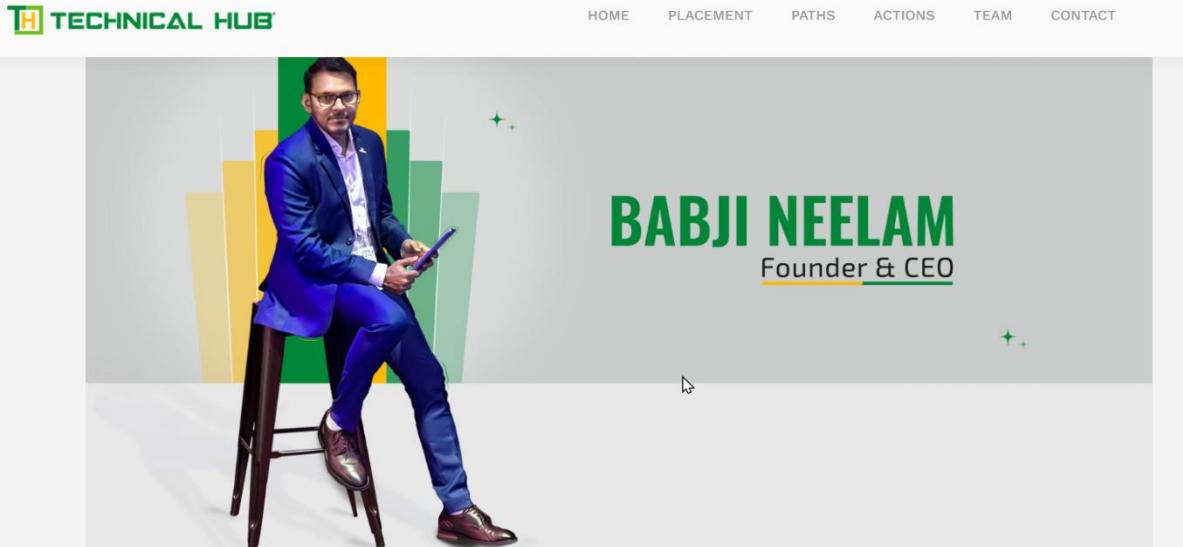
A screenshot of a website for "TECHNICAL HUB". The top navigation bar includes "HOME", "ABOUT", "SERVICES", "BLOG", "CONTACT", and a search bar. The main content area features a banner for "Drive Ready" with the tagline "#Ready to hire". The banner text states: "Drive Ready is a program designed to help attendees familiarize themselves with a technology and coding language of their choice. This hybrid mode of training ensures that every attendee is ready to ace every interview they attend. The trainings offered in this program are usually job specific and is designed to help the attendee acquire the adequate skills required to efficiently begin their career." Below the banner is a video player with the title "Technical Hub | Drive Ready" and a "Share" button. To the left, there is a sidebar with a message from the AI bot and a "Type Here ..." input field.

8. Final Deployment and Integration

After all configurations:

- The frontend, Lambda, Lex, and API Gateway were fully connected.
- The chatbot was now capable of handling real-time student queries regarding Technical Hub.





✓ 8. Benefits

This chatbot project offers several key benefits both technically and functionally:

- ◆ **Automated Query Support**
 - It reduces the dependency on human operators for basic queries.
 - Students and visitors can get instant responses without delays.
- ◆ **Reduces Manual Workload**
 - Faculty or support teams no longer need to answer repeated questions manually.
 - Routine queries like course details, certification info, or trainer names are handled by the bot automatically.
- ◆ **Easy Scalability with AWS Serverless**
 - With services like Lambda and API Gateway, the system is inherently scalable.
 - No need to manage physical servers or worry about load balancing.
- ◆ **Improves Student Engagement**

- Students feel more connected when they can interact with a smart assistant.
- Encourages curiosity and exploration of the Technical Hub's offerings.

9. Future Enhancements

To make the chatbot even more useful and intelligent, the following improvements can be considered in the future:

- ◆ **Add Voice Support**

- Integrate Amazon Polly or Alexa voice services for voice-based interaction.
- Make the chatbot more inclusive for users with reading or typing difficulties.

- ◆ **Integrate with Maya Platform**

- Expand the chatbot's knowledge to include the Maya student platform.
- Allow students to check attendance, internal marks, and schedules through the bot.

- ◆ **Add Admin Panel for Dynamic Updates**

- Create a simple web-based admin interface.
- Enable non-technical staff to update FAQs, responses, or announcements easily.

- ◆ **Multi-Language Support**

- Support regional languages like Telugu or Hindi to increase accessibility.
- Use Amazon Lex's language support to expand multilingual capabilities.

◆ Team Photos:

