Comparative Evaluation of NLU Frameworks

Name: Harshith Manikhanta Sunkra

Frameworks: Google Dialogflow, Rasa NLU, Microsoft LUIS, OpenAI GPT

Objective: Build a simple FAQ chatbot using each NLU framework and evaluate them based on:
- Customization
- Ease of Use
- Deployment

1. Google Dialogflow

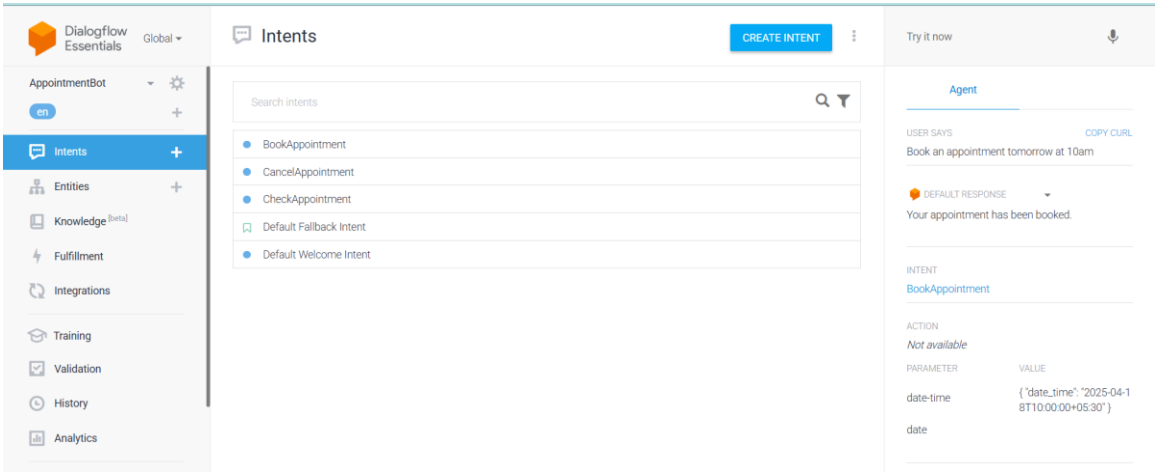
Overview:
Google Dialogflow is a cloud-based NLU platform that helps build conversational interfaces easily using pre-built components and an intuitive interface.

Customization:
Dialogflow allows defining intents and entities using a graphical interface. You can create custom training phrases, responses, and use webhooks for dynamic behavior. However, while it supports domain-specific customization, it still feels somewhat limited if you need full control over the backend or logic.

Ease of Use:
Very beginner-friendly. It was super easy to get started — just created a project, defined intents, and tested in the built-in simulator. No coding needed for basic bots.

Deployment:
Deployment is simple. Since it's cloud-hosted, once the bot is ready, you can easily integrate it with platforms like Telegram, Slack, and websites. SDKs are available.

Output Screenshot:



2. Rasa NLU

Overview:

Rasa is an open-source framework for building NLU and dialogue management models. It runs locally and gives complete control. Used Anaconda for working process

Customization:

This was the most customizable of all. You define intents and entities manually in YAML files, train your models, and can create complex workflows with Python actions.

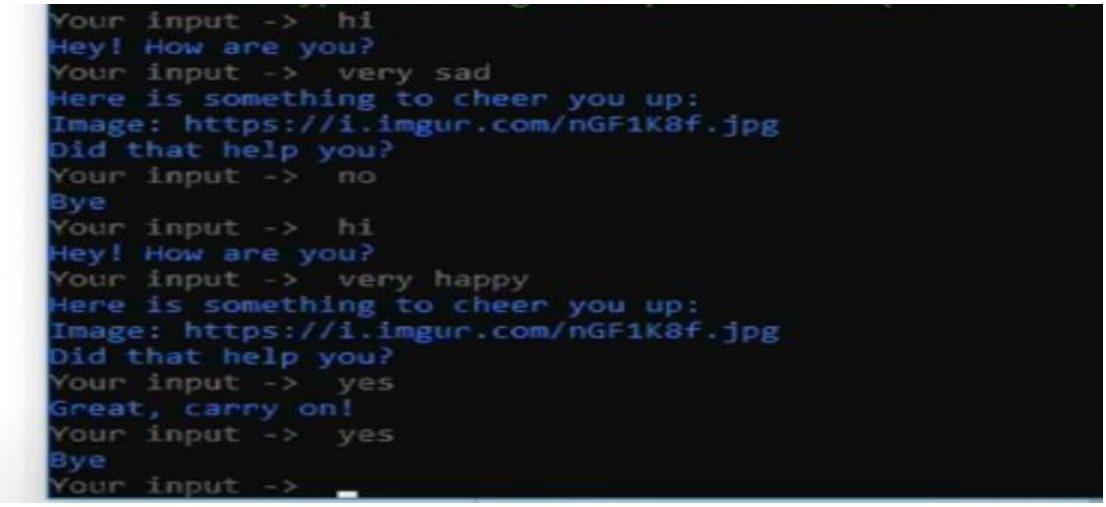
Ease of Use:

Not very easy at first. Required installation of Rasa CLI, understanding of the directory structure, writing training data manually, and running training commands.

Deployment:

Supports both local and cloud deployment. You can host it on your server or even use Docker and Kubernetes.

Output Screenshot:



3. Microsoft LUIS / Azure Language Studio

Overview:

Language Understanding (LUIS) is part of Microsoft's Azure Cognitive Services. It's designed to extract intents and entities from natural language input and power conversational AI. However, as of now, LUIS is being deprecated in favor of Conversational Language Understanding (CLU) within Azure Language Studio.

Customization:

LUIS originally allowed users to create intents and entities via a clean web interface, and also supported JSON import/export. However, since new users can no longer access luis.ai, I used Azure Language Studio (CLU) instead.

CLU offers very similar functionality — you can define intents, utterances, and entities, and train a model. It also allows you to test your bot and view how well it understands different inputs.

Ease of Use: Setting up CLU required a few extra steps. I had to:

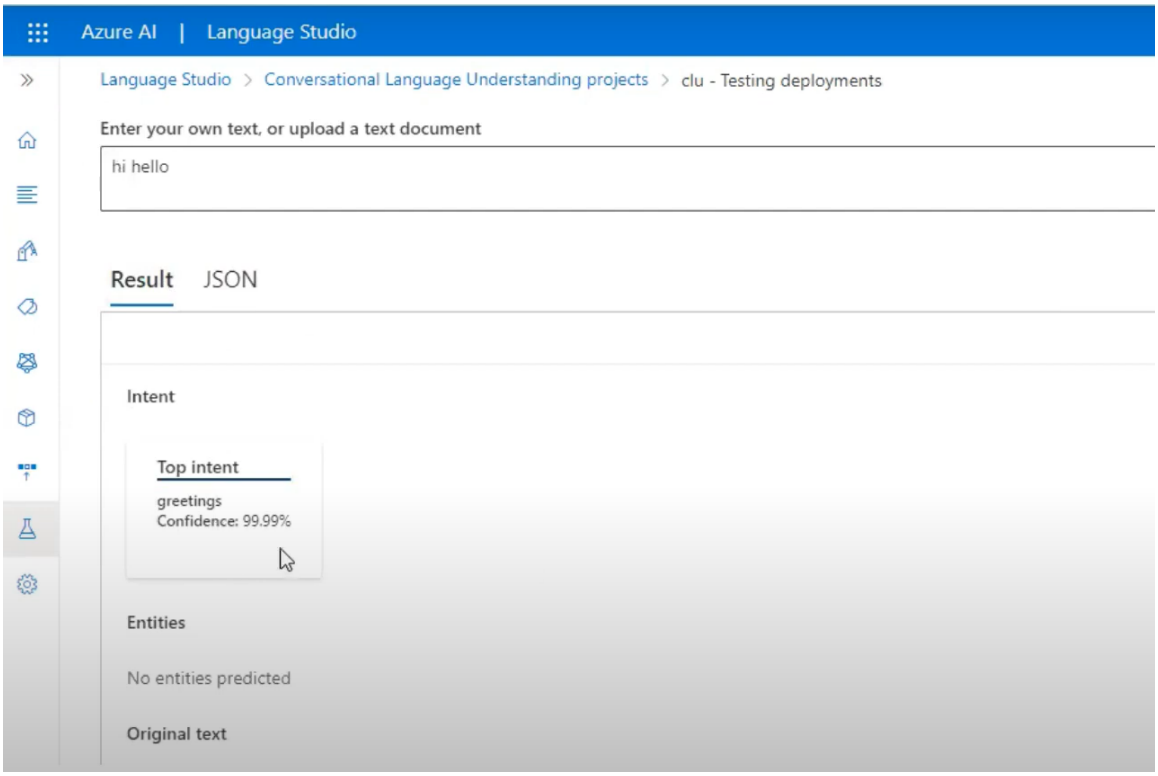
- Create an Azure account.
- Set up a Cognitive Services resource.
- Navigate to language.azure.com and build the CLU project.

Once inside the studio, building the bot was smooth — the UI is beginner-friendly and lets you easily test the model. However, the setup (especially configuring Azure) might be a bit much for someone new.

Deployment: Deployment in CLU is tied to Azure Cloud. Once trained, the model can be accessed using REST APIs, and integrated into apps, websites, or bots via Azure Bot Services or SDKs like Bot Framework. This gives flexibility, but again, everything runs through Azure.

Why I Used Azure Language Studio Instead of LUIS: When I tried to sign up at luis.ai, it redirected me to the homepage repeatedly. After researching, I found that LUIS is no longer supported for new users and is being phased out. So, I used the Azure Language Studio instead, which is Microsoft's new platform for natural language understanding.

Output Screenshot:



4. OpenAI GPT (ChatGPT API)

Overview:
GPT uses large-scale language models to understand and respond using context. I used the ChatGPT API to simulate a chatbot.

Customization:
No need to define intents or entities. Used prompt engineering to guide the bot's behavior. Fine-tuning possible but not beginner-friendly.

Ease of Use:
Very easy. Used Python with Flask to create a simple interface. The GPT API handled all logic.

Deployment:
Can be deployed anywhere since it's API-based. Requires an internet connection and an API key.

Output Screenshot:

Final Comparison Table

Feature	Dialogflow	Rasa NLU	Microsoft LUIS	OpenAI GPT	Best For
Customization	Medium	High	Low	Flexible	Advanced Devs (Rasa)
Ease of Use	Very Easy	Moderate	Easy	Very Easy	Beginners (Dialogflow/GPT)
Deployment	Cloud (Google)	Cloud/Local	Azure Only	Anywhere (API)	Cloud Agnostic (GPT)
Learning Curve	Low	High	Medium	Low	Low (Dialogflow/GPT)

Conclusion

All four platforms have different strengths:
- Dialogflow: Best for quick and simple bots.
- Rasa: Most flexible and powerful but needs more effort.
- LUIS: Good for Azure-based projects.
- GPT: Ideal for natural conversation and creativity.

For my FAQ bot, Dialogflow and GPT gave the best experience in terms of setup and natural responses, while Rasa gave the most control.