Project Design Phase-II

|  |  |
| --- | --- |
| Date | 11october2023 |
| Team ID | NM2023TMI5559 |
| Name | A.yuvashree |
| Project Name | Chatbot deployment with IBM cloud Watson assistant |

**Chatbot deployment with IBM cloud Watson assistant**

**PHASE 2- INNOVATION**

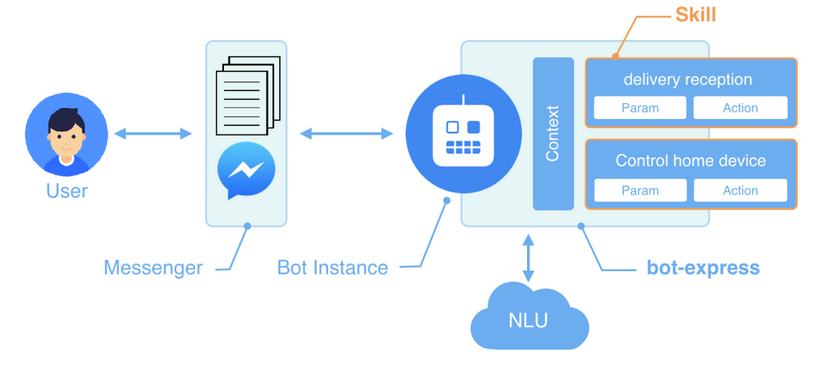
In this section you need to put your design into innovation to solve the problem. Create a document around it and share the same for assessment as per the instructions mentioned.

Consider implementing advanced features such as natural language understanding (NLU) for more accurate user intent recognition.

**Chatbot deployment**:

IBM Cloud Watson Assistant is a powerful chatbot platform that enables businesses to create and deploy virtual assistants for various applications. It leverages natural language processing to understand and respond to user queries, making it a versatile tool for customer support, information retrieval, and more. With Watson Assistant, you can design conversational interfaces, integrate with different channels, and analyze interactions to improve user experience.

**Architecture of chatbot deployment(NLU) diagram:**



**Example Dataset:**

You can create an example dataset in JSON format.

For instance:

json

{

"intents": [

{"intent": "greetings", "examples": ["Hello", "Hi", "Hey"]},

{"intent": "goodbye", "examples": ["Goodbye", "See you", "Bye"]},

{"intent": "help", "examples": ["Help me", "I need assistance", "Can you help?"]}

]

}

**Step 1:** Create a Watson Assistant Service on IBM Cloud

1. Log in to your IBM Cloud account and go to the catalog.

2. Search for "Watson Assistant" and select it.

3. Follow the prompts to create a new Watson Assistant instance.

**Step 2:** Create a Skill

1. Once your Watson Assistant service is created, open it.

2. Create a new skill or import an existing one. This is where you'll define the dialog flow and responses for your chatbot.

**Step 3:** Obtain Credentials

1. In your Watson Assistant instance, navigate to the "Manage" tab.

2. Under the "API Details" section, note down your API Key, Service URL, and Assistant ID. You'll need these to interact with the Watson Assistant service through code.

**Step 4:** Set up your Development Environment

You can use any code editor or IDE for this. Let's assume you're using Visual Studio Code (VS Code) for this example.

**Step 5:** Create a New Project

1. Create a new folder for your project. In this folder, you'll organize your code, configurations, and any other files related to your chatbot.

2. Inside the project folder, create subfolders for different components.

**For example:**

**/my\_chatbot\_project**

**/src # Source code**

**/intents # Intent recognition logic**

**/actions # Actions taken based on user input**

**/config # Configuration files (credentials, etc.)**

**/static # Static files (images, etc.)**

**/tests # Test cases (if applicable)**

**Step 6:** Set Up Code to Interact with Watson Assistant

1. In the `/src` folder, create a file for your chatbot logic (e.g., `chatbot.py` if you're using Python).

2. Use the Watson Assistant SDK or API to connect to your Watson Assistant service using the credentials you obtained earlier.

**Step 7:** Define Intent Recognition and Actions

1. In the `/src/intents` folder, create files for intent recognition logic. This can include code for natural language processing (NLP) or any other methods you're using to recognize user intents.

2. In the `/src/actions` folder, create files for handling actions based on user input. This could be sending responses, performing tasks, etc.

**Step 8:** Manage Configuration

1. In the `/src/config` folder, create files to store configuration settings. This could include your Watson Assistant credentials, environment variables, etc.

**Step 9:** Add Static Files (Optional)

If your chatbot uses images, CSS, or other static assets, place them in the `/src/static` folder.

**Step 10:** Testing (Optional)

If applicable, create a `/tests` folder and add unit tests for your chatbot's components.

**Step 11:** Deployment

To deploy your chatbot, you'll need to package the code, dependencies, and configurations. This could involve creating a Docker container, setting up a web server, or deploying it on a cloud platform.

**NLU TECHNIQUE FOR IMPROVE THE CHATBOT DEPLOYMENT:**

The steps to create a Chatbot deployment with IBM Cloud Watson Assistant using Natural Language Understanding (NLU) and provide you with a basic outline of the code. Please note that I'll be providing a simplified example, and you might need to customize it further based on your specific requirements.

**Step 1:** Set Up IBM Cloud Account

1. Sign up for an IBM Cloud account if you haven't already.

2. Navigate to the Watson Assistant service and create a new instance.

**Step 2:** Create Watson Assistant Workspace

1. Go to your Watson Assistant instance in IBM Cloud.

2. Create a new workspace.

3. Define intents, entities, and add sample user inputs for training.

**Step 3:** Set Up NLU Service

1. Go to the IBM Cloud catalog and create a new Natural Language Understanding (NLU) service.

2. Obtain the credentials for this service.

**Step 4:** Code the Chatbot

Here's a simplified Python example using the Flask framework to create a basic web application:

python

from flask import Flask, request, jsonify

from ibm\_watson import AssistantV1, NaturalLanguageUnderstandingV1

from ibm\_cloud\_sdk\_core.authenticators import IAMAuthenticator

app = Flask(\_\_name\_\_)

# Set up Watson Assistant

authenticator = IAMAuthenticator('<ASSISTANT\_API\_KEY>')

assistant = AssistantV1(

version='2018-09-20',

authenticator=authenticator

)

assistant.set\_service\_url('<ASSISTANT\_URL>')

# Set up NLU

nlu\_authenticator = IAMAuthenticator('<NLU\_API\_KEY>')

nlu = NaturalLanguageUnderstandingV1(

version='2018-09-21',

authenticator=nlu\_authenticator

)

nlu.set\_service\_url('<NLU\_URL>')

@app.route('/chat', methods=['POST'])

def chat():

input\_text = request.json['text']

# Use NLU to analyze user input

response = nlu.analyze(text=input\_text, features={'entities': {'model': 'YOUR\_CUSTOM\_MODEL\_ID'}}).get\_result()

entities = response['entities']

# Pass user input and entities to Watson Assistant

response = assistant.message(

workspace\_id='<WORKSPACE\_ID>',

input={

'text': input\_text,

'entities': entities

}

).get\_result()

return jsonify(response)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Step 5:** Deploy on IBM Cloud

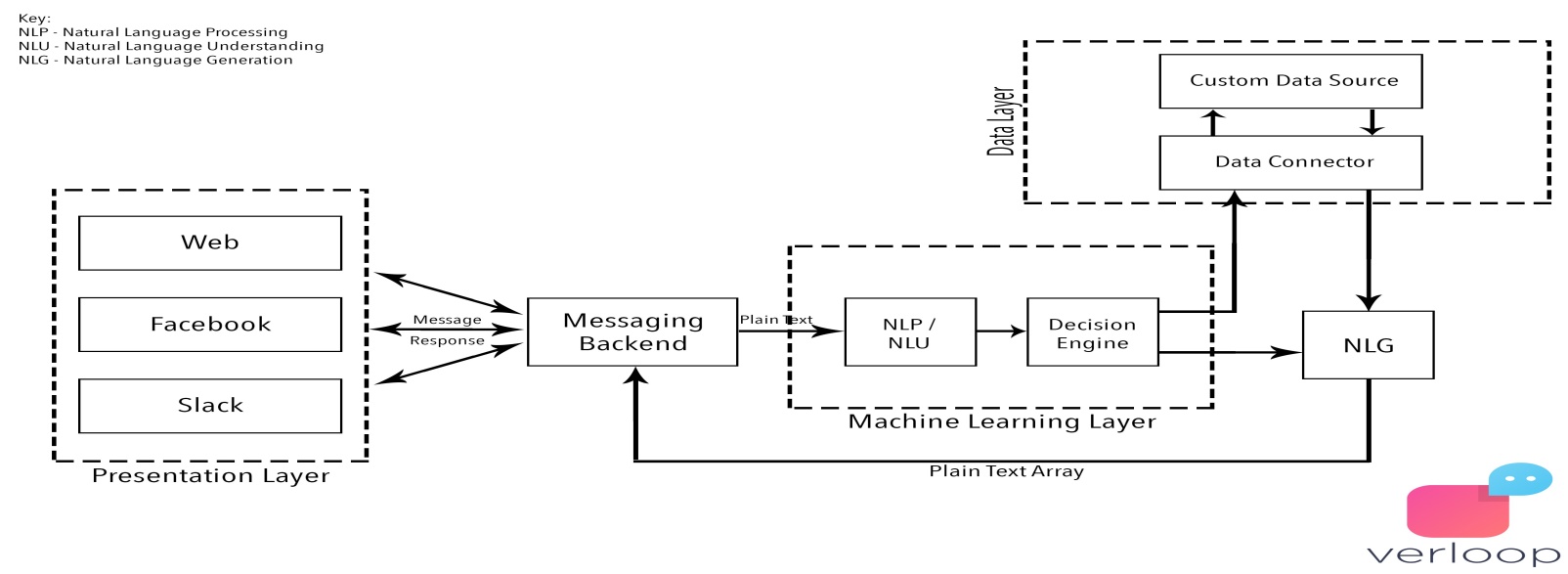
1. Create a `manifest.yml` file with necessary configurations.

2. Push your application to IBM Cloud using the `cf push` command.

**Step 6:** Test Your Chatbot

Access your deployed application's URL and interact with the chatbot.

**Simple diagram for NLU technique in chatbot deployment**

****

**Conclusion:**

Integrating Watson Assistant with Natural Language Understanding can greatly enhance the capabilities of your chatbot by providing deeper insights into user intents and sentiments. This combination allows for more contextually aware responses. Remember to replace placeholders (e.g., YOUR\_ASSISTANT\_API\_KEY, YOUR\_NLU\_API\_KEY, etc.) with your actual credentials.