

Project Development Phase

Delivery of Sprint - 4

| | |
|--------------|---|
| Date | 09 November 2022 |
| Team ID | PNT2022TMID19961 |
| Project Name | AI-based discourse for Banking Industry |

Creating Assistant & Integrate With Flask Web Page

You will be creating a banking bot in this activity that has the following capabilities

1. The Bot should be able to guide a customer to create a bank account.
2. The Bot should be able to answer loan queries.
3. The Bot should be able to answer general banking queries.
4. The Bot should be able to answer queries regarding net banking.
5. With the help of this bot, you can get all the required details related to banking.

Let us build our flask application which will be running in our local browser with a user interface.

In the flask application, users will interact with the chatbot, and based on the user queries they will get the outcomes.

Build Python Code

1: Importing Libraries

The first step is usually importing the libraries that will be needed in the program.

```
from flask import Flask, render_template
```

Importing the flask module into the project is mandatory. An object of the class is our WSGI application. Flask constructor takes the name of the current module (`__name__`).

2: Creating our flask application and loading

```
app = Flask(__name__)
```

3: Routing to the Html Page

Here, the declared constructor is used to route to the HTML page created earlier.

The '/' route is bound with the bot function. Hence, when the home page of a web server is opened in the browser, the HTML page will be rendered.

```
@app.route('/')
def bot():
    return render_template('chatbot.html')
```

Main Function

This is used to run the application in localhost.

```
if __name__ == '__main__':
    app.run()
```

Build HTML Code

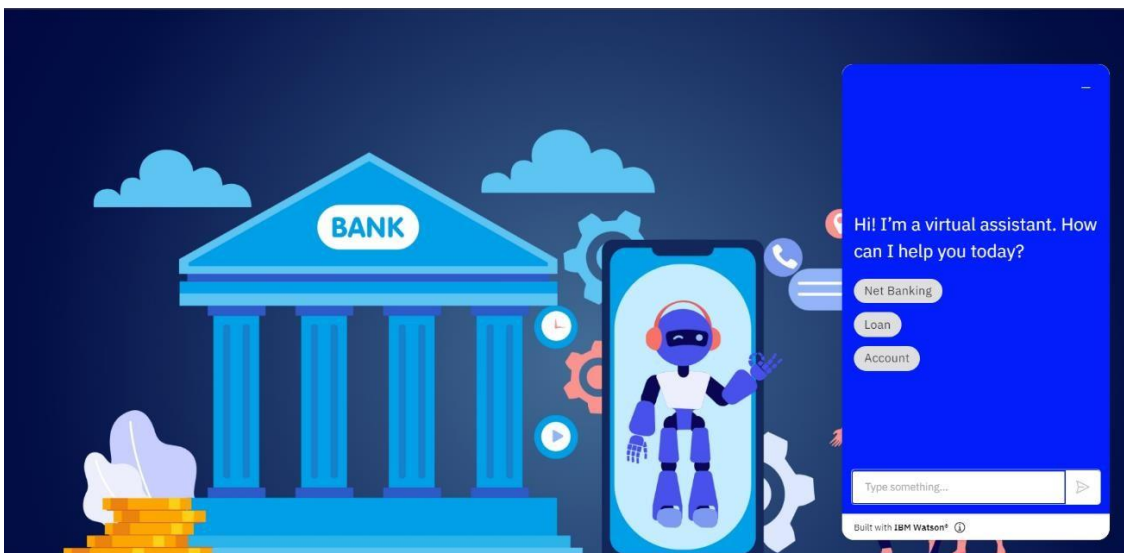
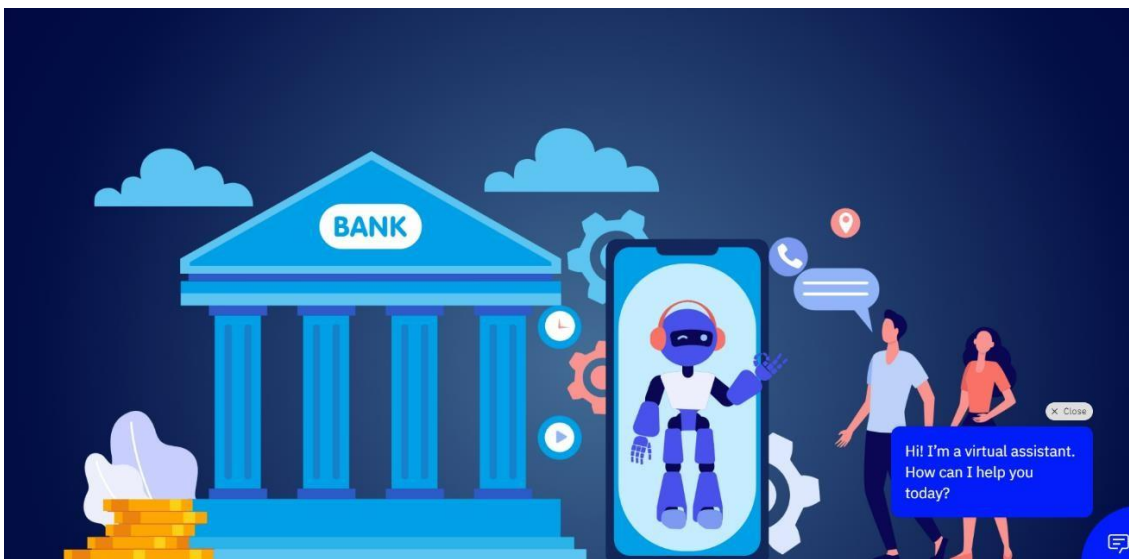
- 」 We use HTML to create the front-end part of the web page.
- 」 Here, we have created 1 HTML page-Chatbot.html
- 」 Chatbot.html displays the home page which integrates with WatsonAssistant.
- 」 A simple HTML page is created. Auto-generated source code from IBM Watson Assistants is copied and pasted inside the body tag

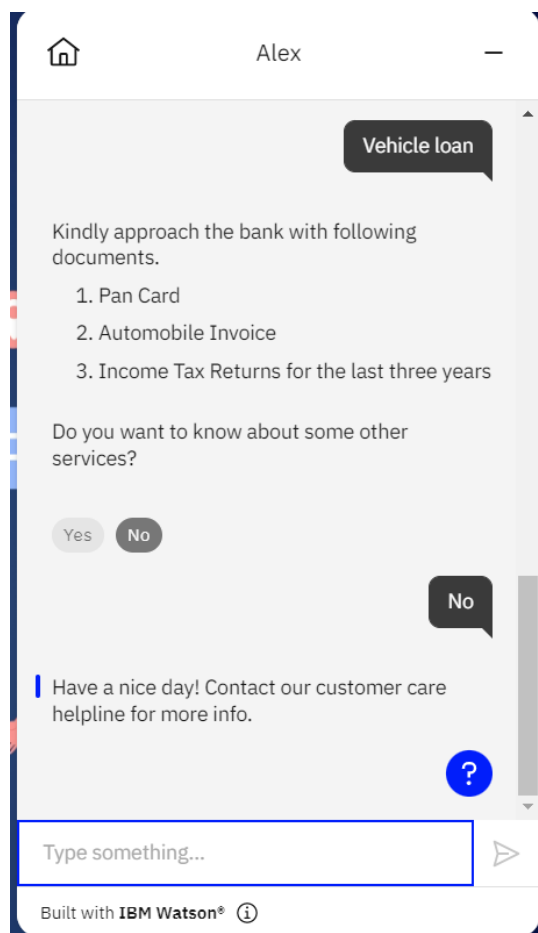
Run The Application

Run the application

- ☐ Open the anaconda prompt from the start menu.
- ☐ Navigate to the folder where your app.py resides.
- ☐ Now type the “python app.py” command.
- ☐ It will show the local host where your app is running on <http://127.0.0.1:5000/>
- ☐ Copy that localhost URL and open that URL in the browser. It doesnavigate me to where you can view your web page.

Then it will run on localhost:5000





PREVIEW OF CHATBOT:

<https://web-chat.global.assistant.watson.appdomain.cloud/preview.html?backgroundImageURL=https%3A%2F%2Fus-south.assistant.watson.cloud.ibm.com%2Fpublic%2Fimages%2Fupx-d06762b6-396b-4236-8512-e118333e6832%3A%3A495bdb54-8670-415a-a8aa-5480cc1528ef&integrationID=2d33609b-c061-4015-9c69-6993e984734c®ion=us-south&serviceInstanceID=d06762b6-396b-4236-8512-e118333e6832>

Source code is attached in Final Deliverables.

Note: No code for this project. So, I attached the screenshot and step to build it.