

## Project Development Phase

### Delivery of Sprint - 4

Date	9 November 2022
Team ID	PNT2022TMID05100
Project Name	AI based discourse for Banking Industry

### Creating Assistant & Integrate With Flask Web Page

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

In the flask application, users will interact with the chat bot, and based on the user queries they will get the Chatbot responses.

### Building 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 Flask 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 local host.

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

## Building HTML Code

We have used HTML to create the front-end part of the web page. Here, we have created “index.html” displays the home page which gets integrated with Watson Assistant.

Auto-generated source code which contains the Integration ID of IBM Watson Assistants is copied and pasted inside the body tag.

## Run the application

Open Jupyter notebook (anaconda3). Navigate to the folder where app.ipynb resides. Run the python code

Open a browser and type this URL <http://127.0.0.1:5000/>

It launches the application integrated with IBM Watson Assistant.



