MILESTONES AND ACTIVITY LISTS

TEAM ID : PNT2022TMID20673

AI-based discourse for Banking Industry

✓ The solution to the problem is Artificial intelligence in the banking sector makes banks efficient, trustworthy, helpful,and more understanding. It is strengthening the competitive edge of modern banks in this digital era. The growing impact of AI in banking sector minimizes operational costs improves customer support and process automation.

✓ Nearly 40% to 50% of financial and banking service providers are using AI in their processes to harness the power of next-generation AI capabilities. The companies believe that AI is the future of banking sector which can perform a range of banking operations in faster, easier, and more secure ways.

✓ AI banking Chatbots help customers in many ways. AI-based chatbot service for financial industry is one of the significant use cases of AI in banking sector. AI chatbots in banking are modernizing the way how businesses provide services to their customers.

✓ AI chatbots in the banking industry can assist customers 24*7 and give accurate responses to their queries. These chatbots provide a personalized experience to users.

✓ AI chatbots in banking is providing a better customer experience.

✓ Hence, AI chatbots for banking and finance operations let banks attract customer attention, optimize service quality,and expand the brand mark in the market.

1. Prerequisites

- IBM Cloud Services
- Software

2.Project Objectives

ABSTRACT

- Work with Watson Assistant
- Create Skills in Watson Assistant
- Use Entities, Intents, Dialogues
- Deploy skill to generate a preview

link BRAINSTORMING

3.Create And Configure IBM Cloud Services

- IBM Watson Assistant

4.Creating Skills & Assistant For Chatbot

Skills are nothing but actions and steps. Steps are the subset of actions where conversations are built and Assistant is used to integrating skills.

- Chatbot Skills Creation
- Creating Saving Account Action
- Creating Current Account Action
- Creating Loan Account Action
- Creating General Query Action
- Creating NetBanking Action

5. Creating Assistant & Integrate with Flask Web

Page 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 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 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 Watson Assistant.

A simple HTML page is created. Auto-generated source code from IBM Watson Assistants is copied and pasted inside the body tag.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Output</title>
    <link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">
    <style>
        body
        {
        background-image: url("https://www.apptunix.com/blog/wp-content/uploads/sites/3/2021/04/show-chatbots-for-banking.jpg");
        background-size: cover;
        }
    </style>
</head>
<body>
<script>
  window.watsonAssistantChatOptions = {
      integrationID: "b5dfbad0-cbc6-4961-a184-4966dd8ba21f", // The ID of this integration.
      region: "us-south", // The region your integration is hosted in.
      serviceInstanceID: "2b1fb818-410e-4536-b69e-dbf938d6211c", // The ID of your service instance.
      onLoad: function(instance) { instance.render(); }
    };
  setTimeout(function(){
    const t=document.createElement('script');
    t.src="https://web-chat.global.assistant.watson.appdomain.cloud/versions/" + (window.watsonAssistantChatOptions.clientVersion || 'latest') +
    document.head.appendChild(t);
  });
</script>
</body>
</html>
```

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 does navigate me to where you can view your web page.

Then it will run on localhost:5000

```
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: off
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

FINAL OUTPUT:

6.Ideation Phase

-》Literature Survey On The Selected Project & Information Gathering

-》Prepare Empathy Map

-》Ideation

7.Project Design Phase -1

-》Proposed Solution

-》Prepare Solution Fit

-》Solution Architecture

8. Project Design Phase -2

-》Customer journey

-》Functional Requirement

-》Data Flow Diagram

-》Technology Architecture

9. Project Planning Phase

-》Prepare Milestones & Activity List

-》Sprint Delivery Plan

10. Project Development Phase

-》Project Development-Delivery Of Sprint-1

-》Project Development-Delivery Of Sprint-2

-》Project Development-Delivery Of Sprint-3

-》Project Development-Delivery Of Sprint-4