Mini Bank Chatbot

Table of Contents

Introduction

This paper details the creation of the **Mini Bank Chatbot**, an AI-incorporated Telegram bot that addresses clients' questions about home loans and applications to the bank. However, in today's world, thanks to the digital world, customer service has become highly competitive, and hence, it is important to provide fast as well as efficient customer service. The Mini Bank Chatbot meets this need by replying to such questions mechanically with appropriate, up-to-date information.

The chatbot works using **Natural Language Processing (NLP)** programming to allow it to understand the user inputs and give conversational outputs as a simpler and faster way to attend to basic banking inquiries without having to involve human personnel. This not only enhances ways to solve customers' issues but also brings less pressure to the agents of customer service as they can allocate more of their time to complicated questions. It is intended to be compatible with the popular Telegram application, allowing customers to easily engage with the Bank's services at their convenience.

Project Overview

The **Mini Bank Chatbot** was designed to increase efficiency of customer exchanges through programmed answers to basic banking questions related to home loans and bank applications. The bot covers usual questions, helps to define eligibility for the loan, or simply offers fundamental account information about opening or closing an account.

The project uses the **System Engineering Product Life Cycle (SEPLC)** as the model to guide the development process and keep everything in order. This framework was used to delicately navigate the project development from the conception stage through the design, development, testing, and implementation stages. Every phase of the SEPLC framework helped make the chatbot not only effective but also as easy to use as possible and tailored to the banking sector.

Key stages in the SEPLC framework for this project include:

- **Conceptualization**: The chatbot's goals are to discover user needs, define those needs, and highlight the required functions (inquiries about home loans and assisting in the application for a bank).
- **Design**: Designing the general technical solution, choosing the right technologies to support the project (Telegram API, Python, NLP libraries) and the pattern of user engagement.
- **Development**: Designing the coding of the chatbot, installing the APIs required for it and training the chatbot for natural language processing for interacting with the users.
- **Testing**: Performing rigorous tests to ascertain the chatbot deals with questions appropriately, operates effectively in different situations, and meets expectations of the users.

- **Deployment**: Introducing the chatbot to users through the Telegram platform and creating the availability of the chatbot for users with different needs to access and providing constant support in order to release updates and improvements.

The application of SEPLC framework ensures that when developing the chatbot, there is a clear understanding of what the chatbot is suppose to achieve in the first instance, it also makes sure that the chatbot developed fulfils the functional requirements we are likely to add in the future for instance adding machine learning models.

Setup Instructions

Step 1: Create the Bot

1. **Create a Telegram Bot**:
   - Open Telegram and search for the **BotFather**.
   - Start a conversation with BotFather and use the command /newbot to create your bot.
   - Follow the prompts to name your bot and receive your **bot token**. This token is essential for connecting your code to the Telegram Bot API.

Step 2: Set Up the Virtual Environment

1. **Navigate to Your Project Directory**
   - cd path/to/MiniBankChatbot_Telegram
   - Replace path/to/MiniBankChatbot_Telegram with the actual path to your project folder.
2. **Create a Virtual Environment**:
   - python -m venv venv
   - This command creates a folder named venv that isolates your project dependencies from the global Python environment.
3. **Activate the Virtual Environment**:

- **Windows**
   - venv\Scripts\activate
- **macOS/Linux**:
   - source venv/bin/activate

Once activated, your terminal prompt will display `(venv)`, indicating that the virtual environment is active.

Step 3: Install Dependencies

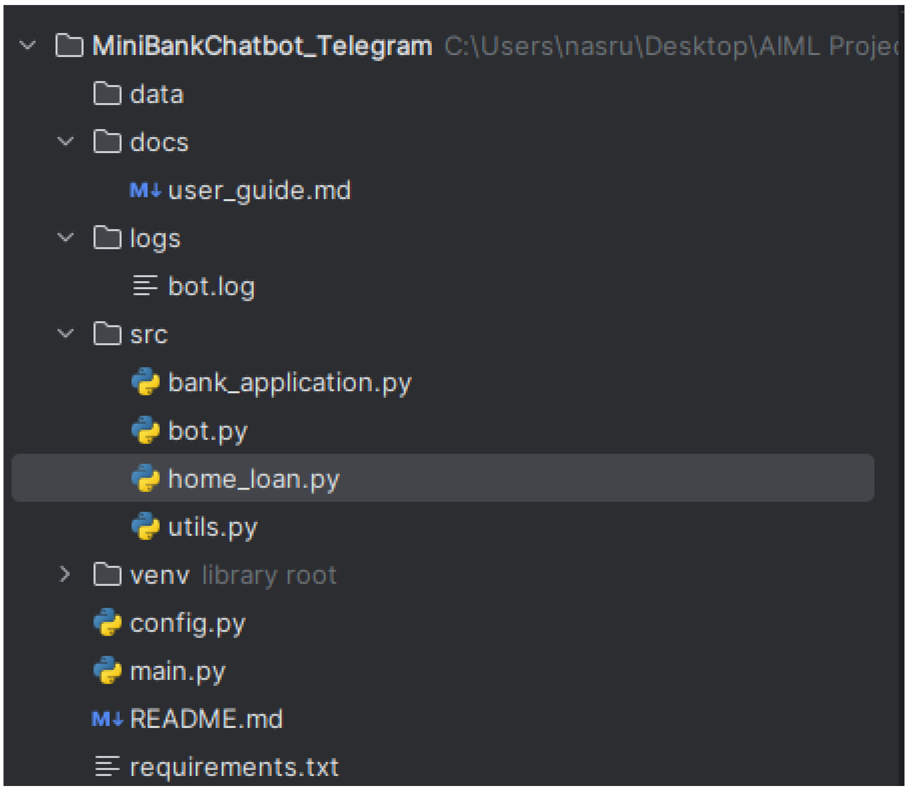
1. **Install Dependencies**: With the virtual environment activated, run:
   - pip install -r requirements.txt
   - This command installs all the libraries listed in `requirements.txt` within your virtual environment.

Step 4: Run the Chatbot

1. **Ensure All Files Are in Place**: Verify that your project structure includes all necessary files (`main.py`, `config.py`, `src/`, etc.).
2. **Run the Chatbot**: Execute the following command to start the chatbot:
   - python main.py
   - The bot should now be running and accessible via Telegram.

Project Structure

Ensure your project directory follows the structure below for optimal organization and scalability:



```
MiniBankChatbot_Telegram C:\Users\nasru\Desktop\AIML Projec
    data
    docs
        user_guide.md
    logs
        bot.log
    src
        bank_application.py
        bot.py
        home_loan.py
        utils.py
    venv library root
    config.py
    main.py
    README.md
    requirements.txt
```

README.md: Overview and setup instructions.

requirements.txt: Python dependencies.

main.py: Entry point for running the bot.

config.py: Stores sensitive information like the bot token.

data/: Contains sample data files.

src/: Source code for core functionalities.

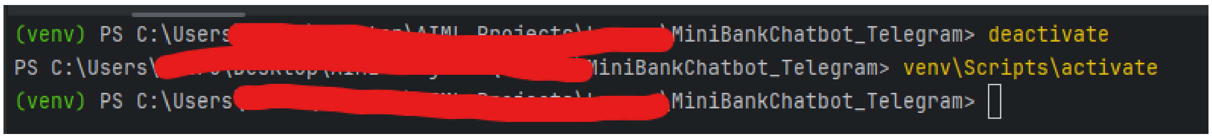
logs/: Log files for debugging.

docs/: Documentation, including user guides.

Screenshots

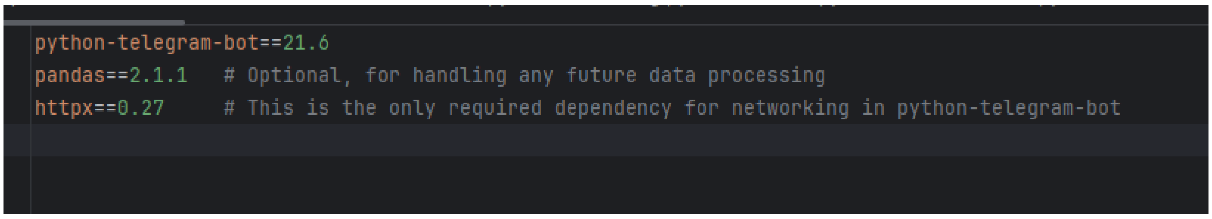
Include screenshots at relevant sections to illustrate each step. Ensure that screenshots are clear and appropriately named.

1. **Virtual Environment Activation**:



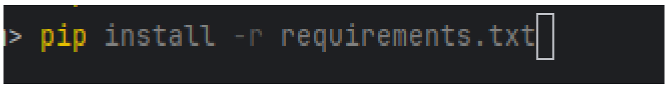
```
(venv) PS C:\Users[redacted]\MiniBankChatbot_Telegram> deactivate
PS C:\Users\[redacted]MiniBankChatbot_Telegram> venv\Scripts\activate
(venv) PS C:\Users[redacted]\MiniBankChatbot_Telegram>
```

2. **requirements.txt**:



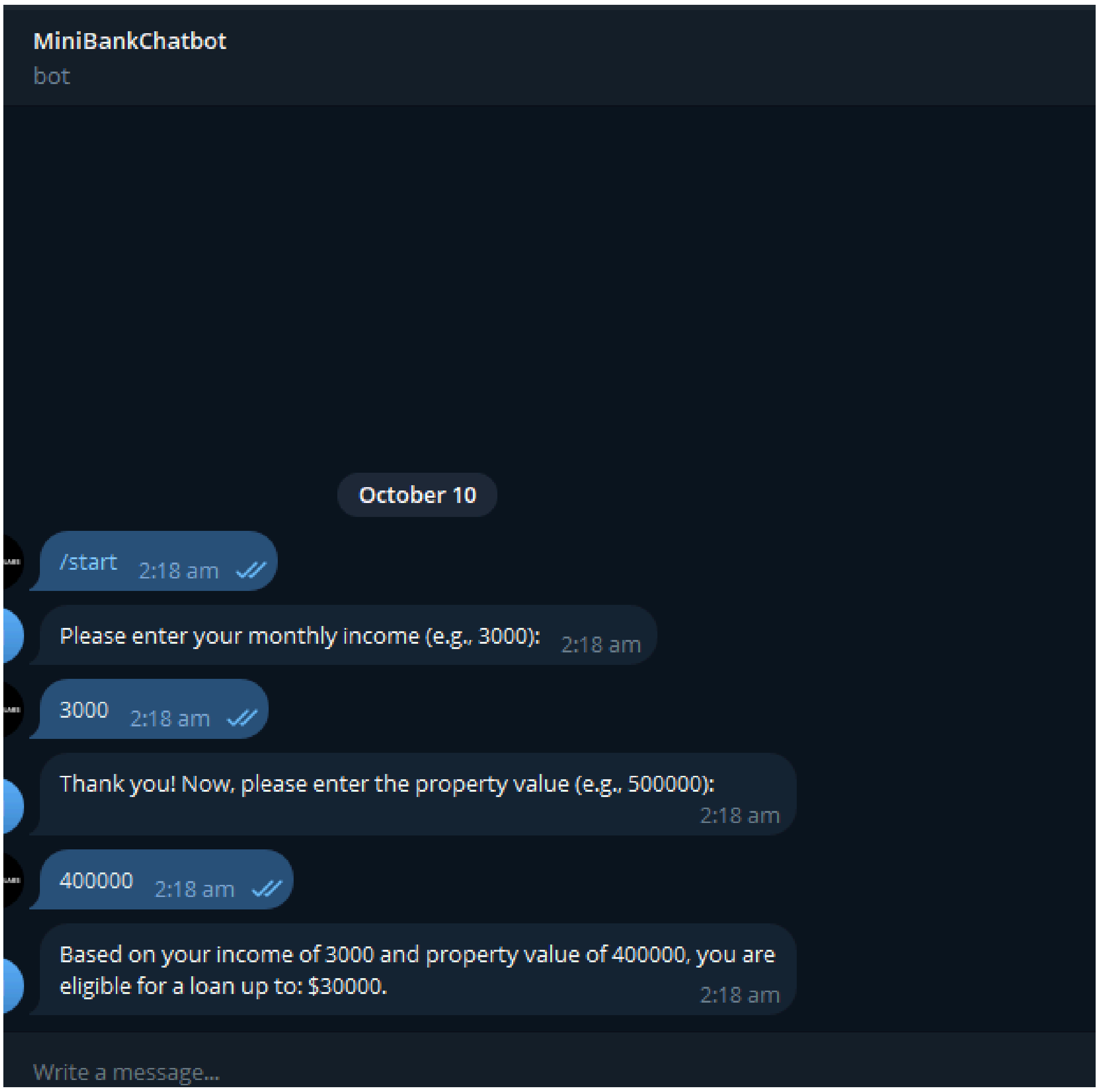
```
python-telegram-bot==21.6
pandas==2.1.1    # Optional, for handling any future data processing
httpx==0.27      # This is the only required dependency for networking in python-telegram-bot
```

3. **Installing Dependencies**:



```
> pip install -r requirements.txt
```

4. **Running the Chatbot**:


MiniBankChatbot
bot

October 10

/start 2:18 am

Please enter your monthly income (e.g., 3000): 2:18 am

3000 2:18 am

Thank you! Now, please enter the property value (e.g., 500000): 2:18 am

400000 2:18 am

Based on your income of 3000 and property value of 400000, you are eligible for a loan up to: $30000. 2:18 am

Place all screenshots in the `docs/screenshots/` directory and reference them accordingly in the report.

Future Enhancements

1. **Predictive Loan Eligibility with Machine Learning**:
    - **Current Status**: The chatbot currently uses static parameters to estimate loan eligibility based on user-provided income and property value.
    - **Future Enhancement**: Implement a machine learning model, such as **linear regression** or **decision trees**, trained on historical loan approval data. This model will predict a user's loan eligibility with higher accuracy, providing personalized loan options based on their financial situation.
    - **Steps**:

        1. Collect relevant data, including income, credit scores, loan terms, and repayment histories.
        2. Train a machine learning model to predict loan amounts or approval likelihood.
        3. Integrate the model into the chatbot’s backend for real-time predictions.
2. **User Behavior Analysis and Feedback Learning**:
    - **Current Status**: The chatbot currently responds to user inputs without learning from past interactions.
    - **Future Enhancement**: Implement a simple feedback system where users can rate their experience after each interaction. Over time, the chatbot can use this data to improve responses, making it more effective and user-friendly.
    - **Steps**:
        1. Add a feedback prompt at the end of each interaction (e.g., “Was this helpful? Yes/No”).
        2. Collect feedback and analyze it to adjust the chatbot’s future responses.
        3. Use this feedback data to train a simple model that helps the bot improve its responses over time.

Conclusion

Mini Bank Chatbot project was able to design an AI system in a Telegram platform to respond to home loan related questions and bank applications. It is easy to manage since it follows the structured setup process, creates a virtual workspace, and installs all the dependencies needed by the system to make it fully functional for users. Additional features consist in employing more sophisticated machine learning algorithms for increasing the quality of answers, and in adding new features according to the user’s needs.

Appendix

- **Source Code**: All relevant Python files (`main.py`, `bot.py`, etc.) are included in the project repository.
- **User Guide**: Detailed instructions on running and interacting with the chatbot are provided in `docs/user_guide.md`.
- **Sample Data**: Located in `data/sample_data.csv`.
- **Log Files**: Accessible in the `logs/` directory for debugging purposes.