UM-AI Task

The goal of this task is to develop an application that demonstrates the "AI Responding to Connection Requests" feature. The application should simulate a "Digital Twin" that can mimic a user's conversational style and provide responses to incoming messages.

Requirements

1. User Onboarding

- Create a user onboarding flow where users can input the following details:
    - **Name**
    - **Age**
    - **Sex**
    - **Location**
    - **Education**
    - **Professional Details** (e.g., Job Title, Company Name)
- Store these details in a database or a simple file for later retrieval.

2. Learning User's Talking Patterns

- When onboarding a user, display a **questionnaire of 15 questions** to capture their communication style and preferences.
- Example questions:
    1. How do you introduce yourself to new people?
    2. What kind of people do you like to talk to?
    3. How would you politely decline an offer you’re not interested in?
    4. What's your favorite way to start a conversation?
    5. How do you respond to compliments?
- Store the responses for building a basic user personality model.

3. Message Handling

- Provide a feature where the app can simulate receiving a message from another user.
- The system should:
    - Accept a message as input.
    - Generate a response that mimics the user's conversational style based on their questionnaire answers.
- Use OpenAI's GPT (or another NLP model) to create personalized responses.

Technical Details

- **Backend**: Python (Django is preferred)
- **Frontend**: Flutter is preferred
- **Database**: PostgreSQL, or any preferred database to store user data and questionnaire responses.
- **AI Model**: OpenAI GPT-4 (preferred) or any pre-trained NLP model.

Deliverables

1. **Code**: Fully functional code with clear instructions to set up and run the application.
2. **Documentation**:
    - Setup and installation instructions.
    - Explanation of how the system learns the user's communication style.
3. **Demo**:
    - A working demo where:
        - A user is onboarded with basic details and answers the questionnaire.
        - A sample message is sent to the system, and it responds appropriately.

Evaluation Criteria

1. **Functionality**:
    - Does the app cover all the requirements (onboarding, questionnaire, and message response)?
2. **Response Quality**:
    - Are the responses coherent and reflective of the user's style based on the questionnaire?
3. **Code Quality**:
    - Is the code modular, clean, and well-documented?
4. **Scalability**:
    - Is the system designed to handle multiple users in the future?
5. **Creativity**:
    - Has the candidate added any innovative features or enhancements?

Bonus Points

- Implementing a mobile interface for managing users.
- Adding an admin dashboard to view user details and their generated responses.
- Providing a mock API for testing.

How to Submit

GitHub Repository:

1. Create a public GitHub repository for your solution with all the necessary code and files.
2. Include a `README.md` file with:
    - Project description.
    - Installation and setup instructions.
    - Instructions for running the application and testing the features.
    - Any assumptions or decisions made during development.