Application Setup and Usage Guide

This document provides detailed instructions on setting up and running the appointment scheduling application developed in a Jupyter Notebook. It outlines how the application processes user inputs to schedule appointments, list doctors, and manage patient information. It also lists the external libraries and tools required for its operation.

Setting Up the Application

<u>Prerequisites</u>

Before running the application, ensure you have the following installed:

- Python 3.6 or later

- Jupyter Notebook or JupyterLab

- SQLite3

<u>Required Python Libraries</u>

- `ipywidgets` for creating interactive UI elements in the Jupyter Notebook.

- `sqlite3` module for handling SQLite database operations.

- `IPython.display` for output display and management within the notebook.

To install the necessary Python packages, run the following command:

```bash
pip install ipywidgets sqlite3
```

<u>Database Preparation</u>

1. Ensure you have a SQLite database file named `hospital_management_system.db`. If not, create one using SQLite command-line tools or any SQLite GUI application.

2. Populate the database with the necessary tables (`Doctors`, `Patients`, `Appointments`, etc.) using the provided SQL schema.

Running the Application

1. Open the Jupyter Notebook (`main_code.ipynb`) in Jupyter Notebook or JupyterLab.

2. Run each cell sequentially to initialize the application components.

3. Use the interactive widgets displayed at the bottom of the notebook to interact with the application.

Application Functionality

Interpreting User Input

The application uses a chatbot-like interface where users can type commands to perform various actions:

- Add Doctor: Adds a new doctor to the system.

- Schedule Appointmen: Initiates the appointment scheduling process, asking for patient details, doctor selection, and appointment timing.

-List Doctors: Displays a list of all doctors.

- List Patients: Shows all registered patients.

- Give Feedback: Allows entering feedback for an appointment.

User inputs are processed by the `process_input` function, which interprets the command and triggers the corresponding action.

Data Retrieval

The application interacts with a SQLite database to retrieve and store information:

- Doctor and Patient Information: Retrieved from the `Doctors` and `Patients` tables, respectively.

- Appointment Scheduling: Availability is checked by querying the `Appointments` table to avoid scheduling conflicts.

External Libraries and Tools

- ipywidgets: Used for creating interactive UI elements in the notebook.

-SQLite3: Provides database management capabilities for storing and retrieving application data.

- Jupyter Notebook/Lab: Serves as the platform for running the interactive Python application.

Conclusion

This document outlines the steps to set up and run the appointment scheduling application, its functionality, and the external libraries it utilizes. Follow the instructions provided to ensure a smooth application experience.