**Medication Reminder Application Documentation**

**Table of Contents**

1. Introduction
2. Setup and Usage
   * Installation
   * Running the Application
3. Assumptions, Limitations, and Design Choices
4. API Integration Process
   * Handling API Errors

**1. Introduction**

The Medication Reminder application is a web-based tool that helps users manage their medication intake by providing reminders for taking medications at specified times. This documentation provides instructions on how to set up and use the application effectively.

**2. Setup and Usage**

**Installation**

To install and set up the Medication Reminder application, follow these steps:

1. Clone the repository from GitHub: [link to the repository]
2. Install the required dependencies by running the following command:

**pip install -r requirements.txt**

1. Configure the database connection in the **app.py** file. Modify the **SQLALCHEMY\_DATABASE\_URI** variable to match your database configuration.
2. Run the database migrations by executing the following commands:

**flask db init**

**flask db migrate**

**flask db upgrade**

**flask db init: This command created a new migrations directory along with some configuration files necessary for Alembic (the database migration tool used by Flask-Migrate) to run. It's a one-time setup command.**

**flask db migrate -m "Description of your changes": This command generated a new migration script. However, it reported "No changes in schema detected." This typically means that your database schema is already in sync with your models, or no changes were found since the last migration.**

**flask db upgrade: This command applies any pending migrations to the database. In your case, since there were no new migrations (as per the previous step), it didn't make any changes to the database.**

**Now, whenever you make changes to your models, you can run the flask db migrate command to generate a new migration script, and flask db upgrade to apply the changes to your database.**

**Remember to always review the generated migration scripts to ensure they reflect the changes you want to make, especially when making complex or potentially destructive changes to your database schema.**

**Running the Application**

To run the Medication Reminder application, use the following command:

**flask run**

or simply run app.py.

The application will be accessible at **http://localhost:5000** in your web browser.

**3. Assumptions, Limitations, and Design Choices**

**Assumptions**

* The application assumes that users will have a valid email address to receive reminders.
* Users are responsible for entering accurate medication information and setting appropriate reminders.

**Limitations**

* The application does not provide real-time medication data or drug interactions. Users should consult healthcare professionals for comprehensive medication advice.
* Email reminders are sent using a Gmail SMTP server. Ensure that your email configuration allows SMTP access and modify the email settings in the **app.py** file if necessary.

**Design Choices**

* The application is built using Flask, a lightweight web framework in Python, for simplicity and ease of development.
* The application uses a PostgreSQL database to store user information, medication entries, and reminders.
* Flask-SQLAlchemy is used as the ORM (Object-Relational Mapping) tool for database management.
* Flask-Login is used for user authentication and session management.
* BackgroundScheduler from the APScheduler library is used to schedule and send reminders at specified times.

**4. API Integration Process**

**Handling API Errors**

The Medication Reminder application integrates with an external drug event API to provide medication information. Here is the process for handling API errors:

1. When the user searches for medication information, the application sends a request to the API endpoint with the search term.
2. If the API request is successful (status code 200), the application processes and displays the retrieved medication information.
3. If the API request fails (status code other than 200), the application displays an error message indicating the failure to retrieve medication information from the API.

It is important to note that API availability and response quality are dependent on the external API provider. The Medication Reminder application relies on the API's availability and accuracy for retrieving medication information.

This documentation provides a guide to set up and use the Medication Reminder application effectively. It also outlines the assumptions, limitations, and design choices made during the development process. Additionally, it explains how API integration is handled and how API errors are managed. Feel free to update and expand this documentation as needed to meet your specific requirements.

