Project Documentation: Flask-based Ride Booking Application

Overview

This project is a Flask-based web application mimicking a ride-booking service. It utilizes various Flask routes and templates to manage user interactions, from account registration and login to booking rides and processing payments.

Structure

The application consists of two primary Python files: route.py and app.py, serving distinct roles:

- app.py: This is the core of the Flask application. It initializes the Flask app and configures routes, templates, and the MongoDB connection for handling different functionalities like user registration, login, booking, and payment.
- route.py: This script is responsible for running the application. It imports the Flask app from app.py and starts the server, mainly used for development purposes with debug mode enabled.

Key Components

1. **Flask and MongoDB Integration**:
- Flask is used as the web framework.
- MongoDB, integrated via Flask-PyMongo, manages driver data.

2. **Routes and Functionalities**:
- '/' (Home): Displays the landing page of the application.
- '/login': Manages user login. Renders a login form and processes submissions.
- '/register': Handles user registration. Displays a registration form and processes submissions.
- '/booking': Manages ride bookings. Shows a booking form and handles submissions.
- '/price_ride': Calculates and displays ride prices.
- '/payment': Processes payment transactions.
- '/ride_confirmed': Confirms a successful ride booking.
- '/logout': Logs out the user and redirects to the login page.

3. **Forms Management**:
- Utilizes Flask-WTF for form handling, including LoginForm and RegistrationForm.

4. **Map and Geolocation**:
- Implements Folium for map rendering, marking locations like Paris and Lyon.
- Geopy for geolocation services.

5. **Security Features**:
- Password hashing for secure storage and verification.

6. **Running the Application**:
- Use route.py for starting the application in a development environment with debug mode.

Important Notes

- route.py and app.py are separate files serving different purposes. route.py is for running the server, whereas app.py contains the app's main configuration and route definitions.
- For production deployment, consider using a more robust server and disabling debug mode for security.

Further Development

- Implement user session management for enhanced security.
- Integrate advanced payment processing methods.
- Enhance UI/UX for a better user experience.
- Implement additional features like ride sharing or driver rating systems.