ADVANCED PYTHON PROGRAMMING

Project Assignment

Due on: December 31, 2024

Real-Time Vehicle Rental Management System

Objective:

Develop a Real-Time Vehicle Rental Management System using Python. The system should incorporate object-oriented domain modeling, MongoDB (via PyMongo), a RESTful API with FastAPI, Apache Kafka for event streaming, and WebSocket for real-time updates.

Problem Statement:

You will create a system for a vehicle rental company to manage their fleet and rental operations. Employees manage vehicle availability and rental status, while customers can browse and book vehicles. The system must support real-time updates for both customers and employees regarding vehicle status and rental transactions.

Domain Definition:

Entities and Relationships

1. **`User`**
   - Represents a system user.
   - Types:
     - **`Customer`**: Can browse and rent vehicles.
     - **`Employee`**: Can manage vehicle inventory and handle rental operations.
   - Attributes:
     - **`user_id`** (unique identifier)
     - **`name`** (string)
     - **`email`** (string)
     - **`password`** (hashed string)
     - **`role`** (enum: CUSTOMER, EMPLOYEE)
2. **`Vehicle`**
   - Represents a vehicle available for rent.
   - Attributes:
     - **`vehicle_id`** (unique identifier)
     - **`make`** (string, e.g., Toyota)
     - **`model`** (string, e.g., Corolla)
     - **`type`** (enum: CAR, TRUCK, SUV, VAN, MOTORCYCLE)
     - **`rental_price_per_day`** (float)
     - **`availability_status`** (enum: AVAILABLE, RENTED, MAINTENANCE)
     - **`location`** (string, branch name or location identifier)
3. **`Rental`**
   - Represents a rental transaction.
   - Attributes:
     - **`rental_id`** (unique identifier)

    - **`vehicle_id`** (foreign key to Vehicle)
    - **`customer_id`** (foreign key to User)
    - **`rental_start_date`** (datetime)
    - **`rental_end_date`** (datetime)
    - **`total_cost`** (float, calculated)
4. **`Branch`**
    - Represents a rental company branch.
    - Attributes:
        - **`branch_id`** (unique identifier)
        - **`name`** (string)
        - **`location`** (string)
        - **`contact_number`** (string)

System Features

Core Functionalities

1. **Authentication and Authorization**:
    - Customers and employees can log in and register.
    - Role-based access control (RBAC) ensures appropriate permissions.
2. **Vehicle Management (for Employees):**
    - Add, update, and remove vehicles.
    - Update vehicle availability status.
    - View rental history for a specific vehicle.
3. **Rental Operations:**
    - Customers can browse and filter available vehicles by type, price, or location.
    - Customers can book a vehicle for a specified period.
    - Employees can approve or reject rental requests.
    - Calculate the total rental cost based on the rental period.
4. Real-Time Updates:
    - Customers are notified in real-time when a vehicle's status changes (e.g., becomes available or is rented).
    - Employees are notified of new rental requests and approvals.
5. **Event Logging:**
    - Use Apache Kafka to log events such as vehicle additions, rentals, and returns for analytics and auditing.

Domain Modeling

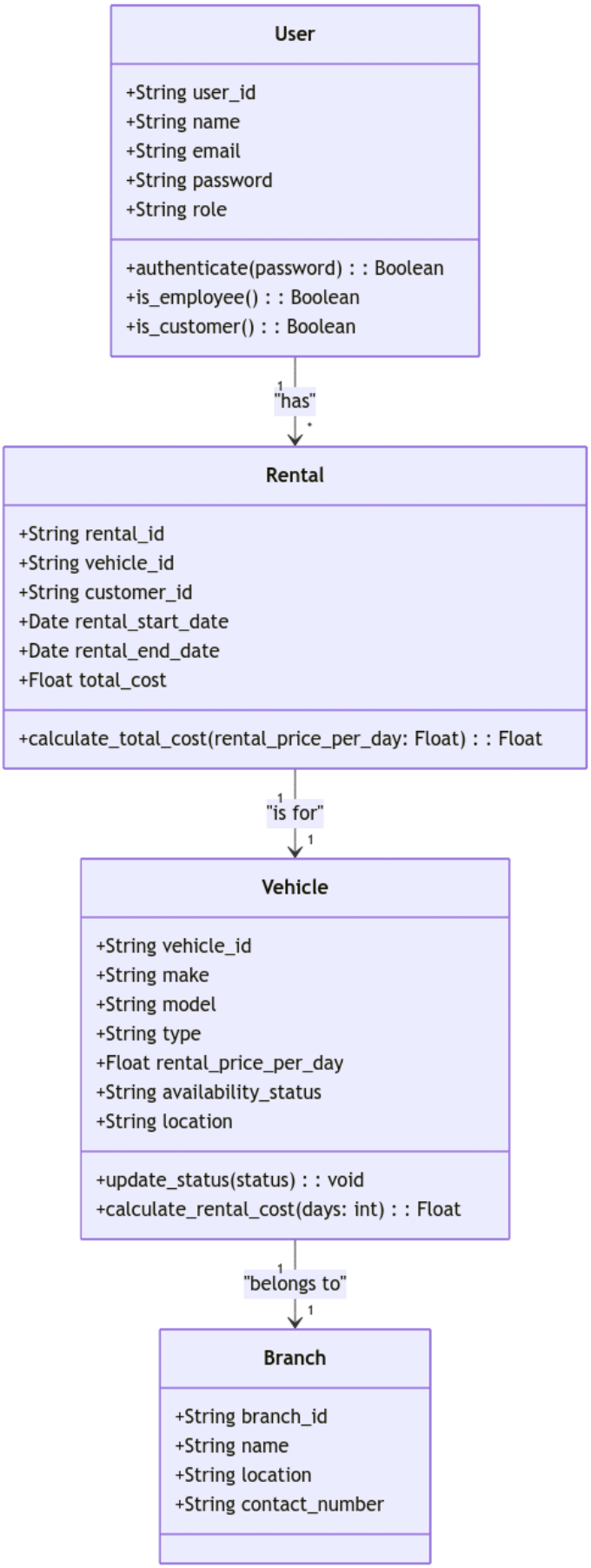
Classes and Their Methods

1. **`User`**
    - **`__init__(user_id, name, email, password, role)`**
    - **`authenticate(password)`**
    - **`is_employee()`**
    - **`is_customer()`**
2. **`Vehicle`**
    - **`__init__(vehicle_id, make, model, type, rental_price_per_day, availability_status, location)`**
    - **`update_status(status)`**

    - **`calculate_rental_cost(days)`**
3. **`Rental`**
    - **`__init__(rental_id, vehicle_id, customer_id, rental_start_date, rental_end_date, total_cost)`**
    - **`calculate_total_cost(rental_price_per_day)`**
4. **`Branch`**
    - `__init__(branch_id, name, location, contact_number)`

Technical Requirements

1. Database Design:
    - Use MongoDB to store collections for users, vehicles, rentals, and branches.
    - Use PyMongo for CRUD operations.
2. API Development:
    - Expose RESTful endpoints for all core functionalities using FastAPI.
    - Include Swagger documentation.
3. Real-Time Communication:
    - Use WebSocket for live notifications of vehicle status and rental transactions.
4. Event Streaming:
    - Configure Apache Kafka to log system events such as:
        - New vehicle added.
        - Rental approved or rejected.
        - Vehicle returned.


User
+String user_id
+String name
+String email
+String password
+String role
+authenticate(password) : : Boolean
+is_employee() : : Boolean
+is_customer() : : Boolean
1
"has"
*
Rental
+String rental_id
+String vehicle_id
+String customer_id
+Date rental_start_date
+Date rental_end_date
+Float total_cost
+calculate_total_cost(rental_price_per_day: Float) : : Float
1
"is for"
1
Vehicle
+String vehicle_id
+String make
+String model
+String type
+Float rental_price_per_day
+String availability_status
+String location
+update_status(status) : : void
+calculate_rental_cost(days: int) : : Float
1
"belongs to"
1
Branch
+String branch_id
+String name
+String location
+String contact_number

Submission:

Submit **all your project files** in a file named *your-student-id*-**project.zip**.

Important Notes:

- Plagiarism and dishonest behavior are strictly prohibited and will result in disciplinary action.
- Assignments are due in class on the specified date.
- Late submissions are generally not accepted.
- Exceptions require prior approval and may incur penalties.