

SWE 2 Project



Car management

**Car management system using
spring boot -microservices.**

CONTENTS

Introduction	3
Project Scope	3
Project purpose	3
System Requirements	7
Functional	7
Non-functional	7
Technical Architecture	8
System design and models	9
Use case diagram	9
Class	10
Activity	11
sequence	16
ERD	18
OCL	19
Development Platform	20
Programming Language & framework	10
Libraries	11
Team Member	20
Code Repository	20

Project overview

The Car Rental Management System is designed to streamline and optimize the processes involved in managing a car rental business. By leveraging microservices architecture, the system aims to provide scalability, flexibility, and robustness, catering to the dynamic needs of the rental industry.

Project purpose

The primary purpose of this project is to develop a comprehensive solution that addresses the challenges faced by car rental companies in managing their operations efficiently. By adopting microservices architecture, the system aims to achieve the following objectives:

- Enhanced Scalability:** The modular nature of microservices allows for independent scaling of different components based on demand, ensuring optimal resource utilization, and accommodating fluctuations in rental activity without compromising performance.
- Flexibility and Customization:** By breaking down the application into smaller, loosely coupled services, the system offers greater flexibility in terms of feature enhancements, updates, and customization according to specific business requirements and evolving market trends.
- Fault Isolation and Resilience:** Microservices promote fault isolation, meaning that a failure in one service does not cascade to others, thereby enhancing the overall resilience of the system. This ensures uninterrupted service availability and minimizes the impact of potential failures on the business operations.
- Improved Performance:** With each service dedicated to a specific function, the system can be optimized for performance, allowing for faster response times and efficient resource utilization, ultimately leading to a seamless user experience for both rental administrators and customers.
- Scalable Architecture:** The use of microservices architecture lays the foundation for a scalable and future-proof solution that can easily accommodate growth in the volume of rental transactions, expansion into new markets, and integration with emerging technologies or third-party services.

Project Scope

1. Core Functionality:

- **User Management:**

- Authentication and authorization mechanisms for administrators, employees, and customers.
- User profile management including registration, login, and password management.

- **Vehicle Management:**

- Adding, updating, and removing vehicle information including make, model, year, mileage, and condition.
- Tracking availability, status, and location of vehicles within the fleet.

- **Reservation and Booking:**

- Facilitating reservation requests, confirmation, modification, and cancellation processes.
- Real-time availability updates to prevent double bookings and optimize fleet utilization.

- **Billing and Payment:**

- Integration with payment gateways for secure and automated payment processing.
- Generating invoices, receipts, and financial reports for transactions.

- **Reporting and Analytics:**

- Generating reports and analytics dashboards to track key performance indicators (KPIs) such as revenue, utilization rates, and customer satisfaction metrics.
- Customizable reporting features to cater to specific business requirements.

2. Additional Features (Optional):

- **Customer Relationship Management (CRM):**

- Managing customer interactions, feedback, and support tickets.
- Implementing loyalty programs and promotional offers.

- **Fleet Maintenance:**

- Tracking vehicle maintenance schedules, service history, and alerts.
- Automating maintenance reminders and scheduling service appointments.

- **Integration with Third-Party Services:**

- Integrating with external systems such as GPS tracking, insurance providers, and roadside assistance services.
- Ensuring compatibility with industry standards and protocols for seamless data exchange.

- **Mobile Application:**

- Developing a mobile app for customers to browse inventory, make reservations, and manage bookings on-the-go.
- Implementing push notifications and location-based services for enhanced user experience.

3. Exclusions:

- **Vehicle Procurement:** The system will not handle the procurement or acquisition of new vehicles into the fleet.
- **Regulatory Compliance:** Compliance with specific regional or industry regulations will be considered but not implemented as part of this scope.
- **Hardware Infrastructure:** The project scope does not include provisioning or configuring hardware infrastructure such as servers or networking equipment.

4. Constraints:

- **Technology Stack:** The system will be developed using specific technologies and frameworks as outlined in the project requirements.
- **Timeline:** The project will adhere to the agreed-upon timeline and milestones as defined in the project plan.
- **Budget:** Development costs and resource allocations will be within the approved project budget.

5. Assumptions:

- **Data Security:** It is assumed that appropriate security measures will be implemented to protect sensitive data and prevent unauthorized access.

- **Stakeholder Involvement:** Stakeholders will be actively involved in requirements gathering, feedback sessions, and user acceptance testing (UAT) to ensure alignment with business objectives.

6. Risks:

- **Technical Challenges:** Potential challenges related to integration, scalability, and performance optimization will be addressed through proactive risk management strategies.
- **Change Management:** Any changes to project scope, requirements, or timelines will be assessed and managed through proper change control procedures.
- **Resource Constraints:** Adequate resources including personnel, budget, and technology infrastructure will be allocated to mitigate risks associated with resource constraints.

7. Deliverables:

- **Functional Requirements Document:** Detailed documentation outlining the functional specifications and features of the Car Rental Management System.
- **Software Prototype/Minimum Viable Product (MVP):** A working prototype or MVP demonstrating key functionalities of the system for stakeholder review and validation.
- **User Manuals and Training Materials:** Comprehensive user manuals and training materials for administrators, employees, and customers to facilitate system adoption and usage.

8. Acceptance Criteria:

- **User Acceptance Testing (UAT):** The system will undergo UAT to validate that it meets the specified requirements and achieves the desired outcomes.
- **Stakeholder Approval:** Final acceptance and approval of the system will be contingent upon stakeholder satisfaction with the delivered product.

Non-functional requirements

1. Performance

- The system should respond to user input within 2 seconds
- The system should handle a minimum of 50 concurrent users

2. Security

- The system should enforce password complexity and expiration policies
- The system should encrypt sensitive data (credit card information, etc.)

3. Usability

- The system should provide an intuitive user interface for easy navigation
- The system should provide error messages and feedback for invalid user input

4. Scalability

- The system should be able to handle an increase in rental agreements by 20% within 3 months
- The system should be able to integrate with third-party services (e.g. payment gateways)

Functional requirements

Admin:

- Manage listings: Add, edit, and remove car listings.
- Access and modify system settings and configurations.
- View analytics and reports.
- Access all features and functionalities of the system.

Customer:

- Browse and search car listings.

- View detailed information about listed cars, including specifications, pricing, and photos.
- Book cars for rental or test drives.
- Leave reviews and ratings for cars they have interacted with.
- Save favorite cars and create wish-lists.
- View their booking history and manage bookings.
- Contact car owners for inquiries and negotiations.
- Make payments for bookings or purchases.
- Receive notifications about booking confirmations, updates, and promotions

Guest/User without Account:

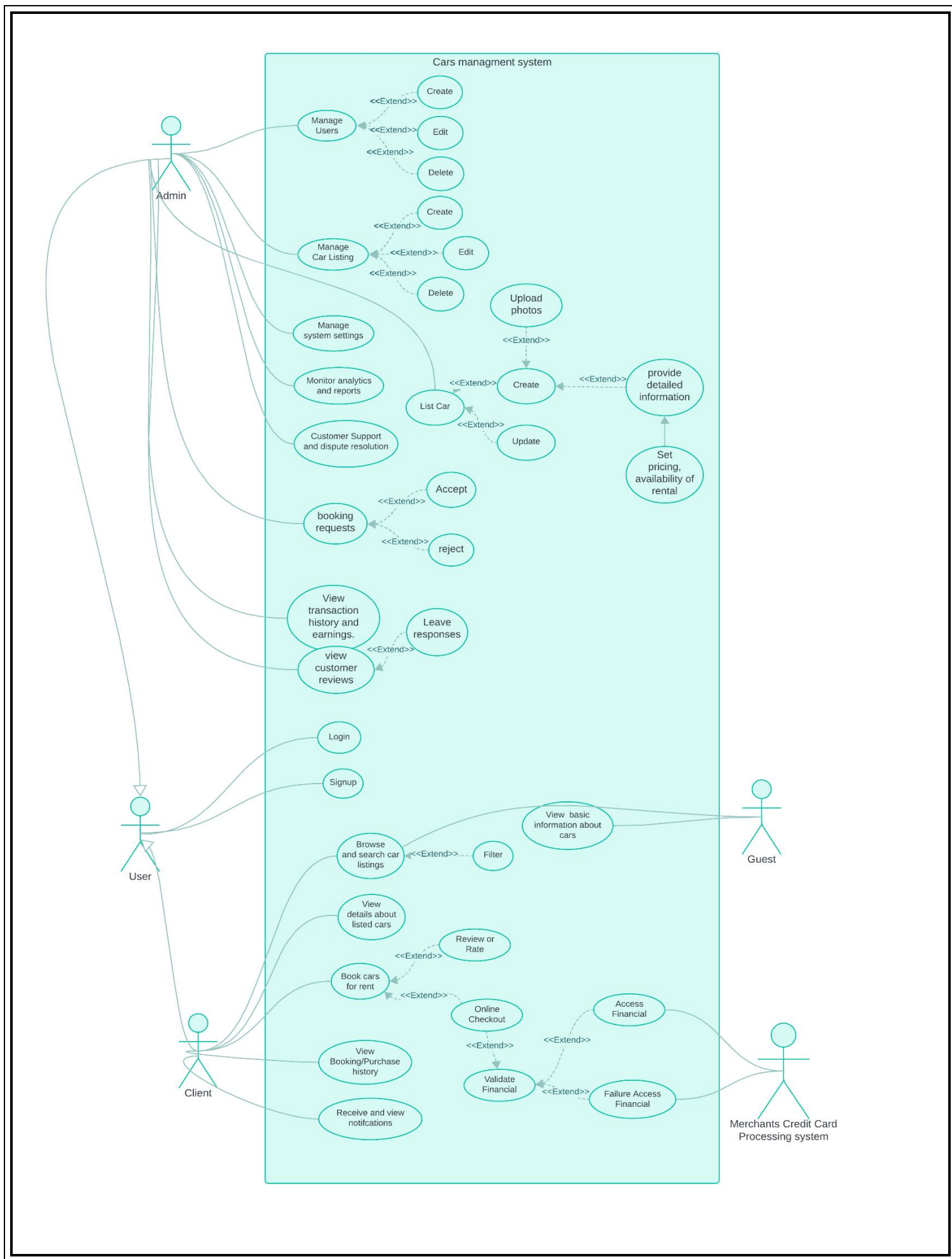
- Browse and search car listings.
- View basic information about listed cars.

Technical Architecture

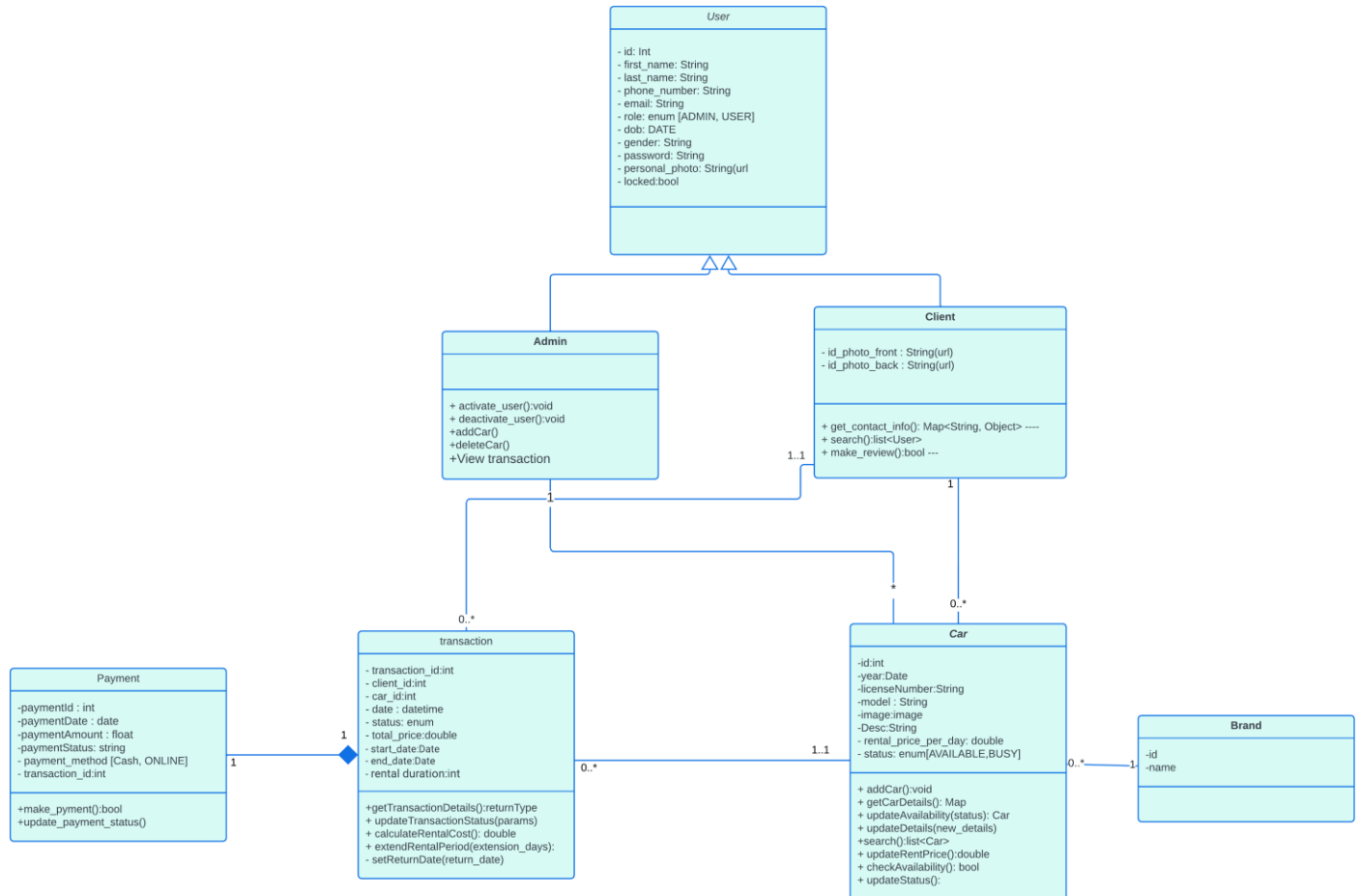
- Spring Boot Framework (JAVA)
- Micro services
- Database pgSQL/MySQL
- Ajax

Diagrams

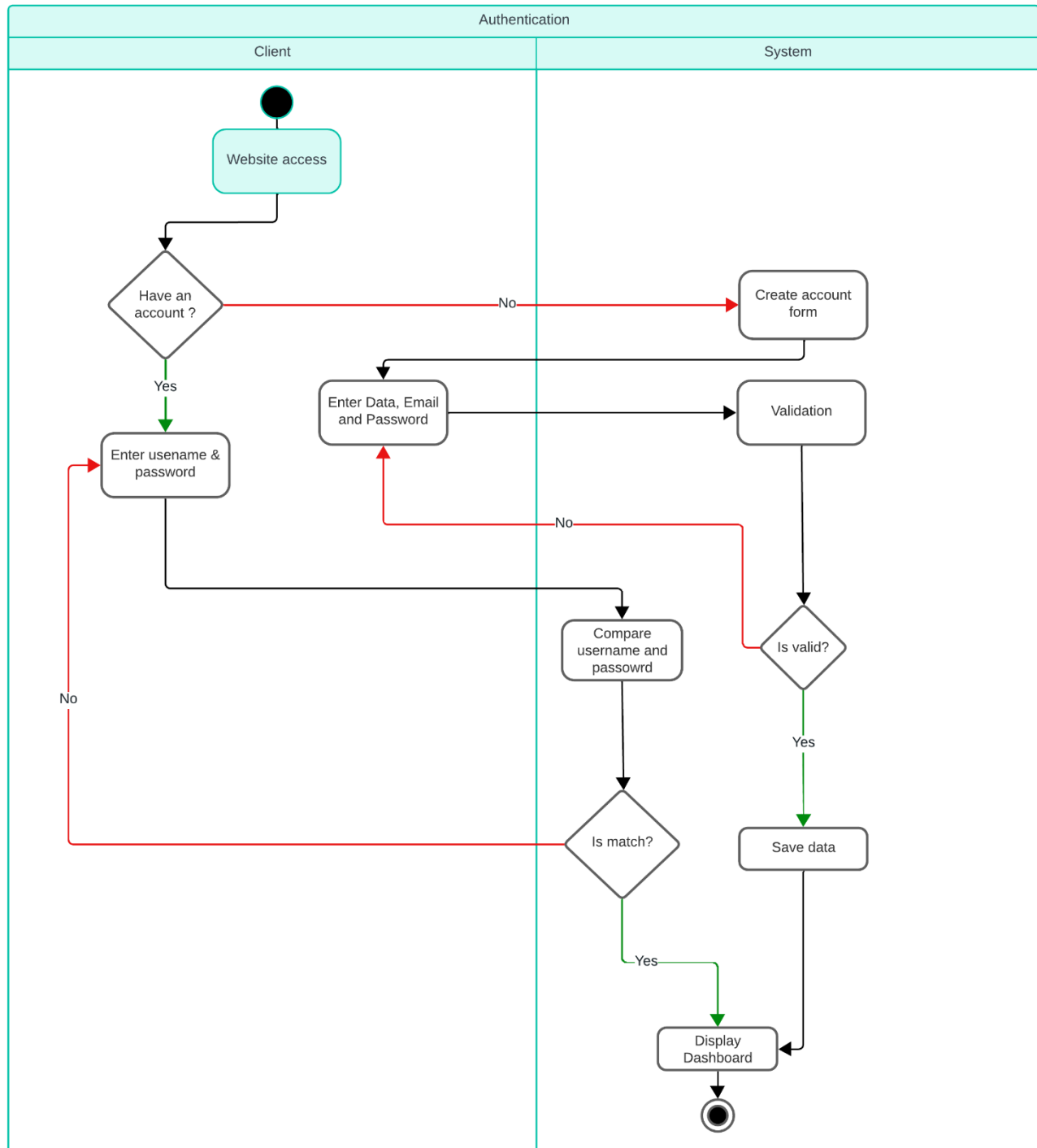
1-use case

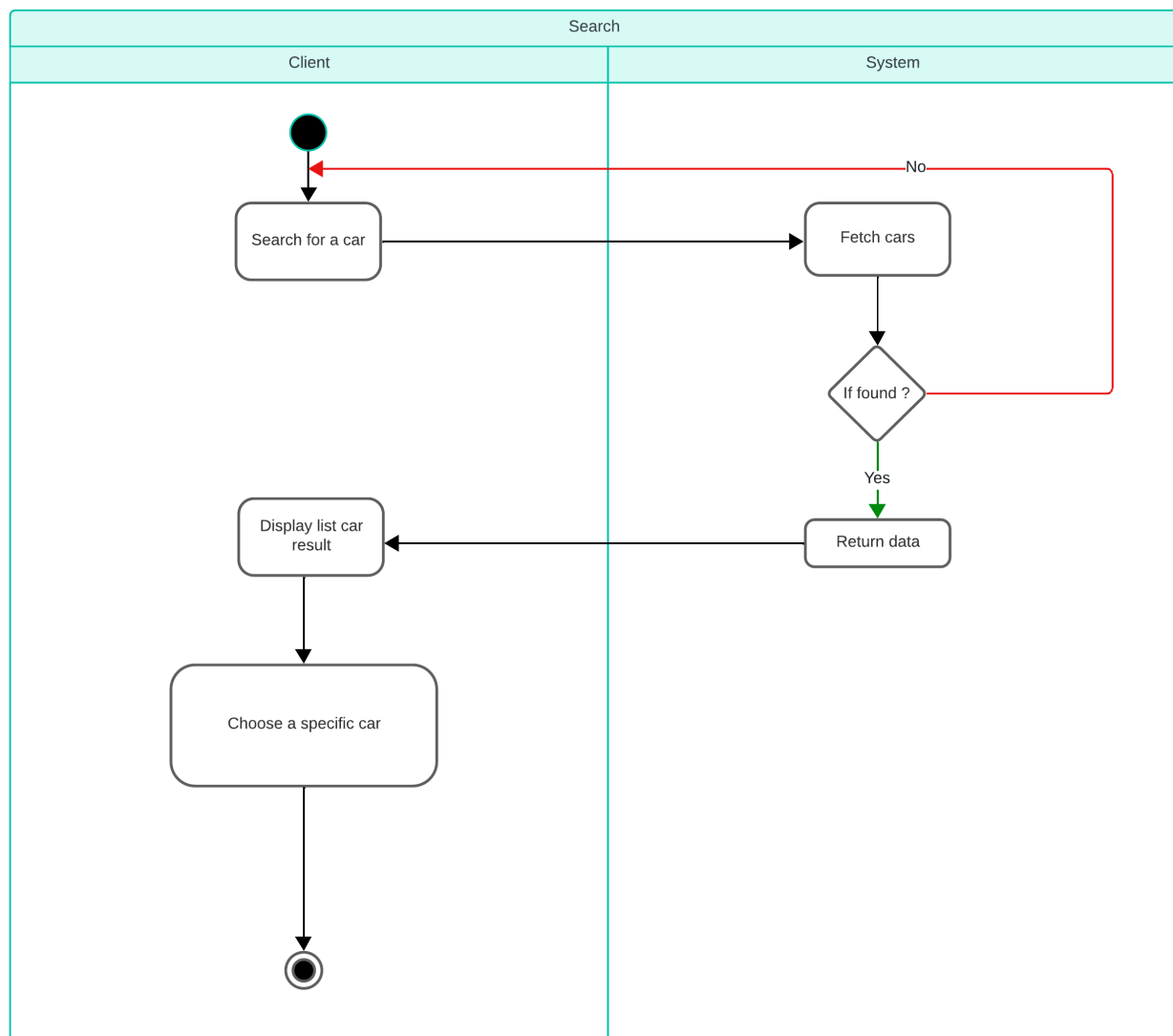


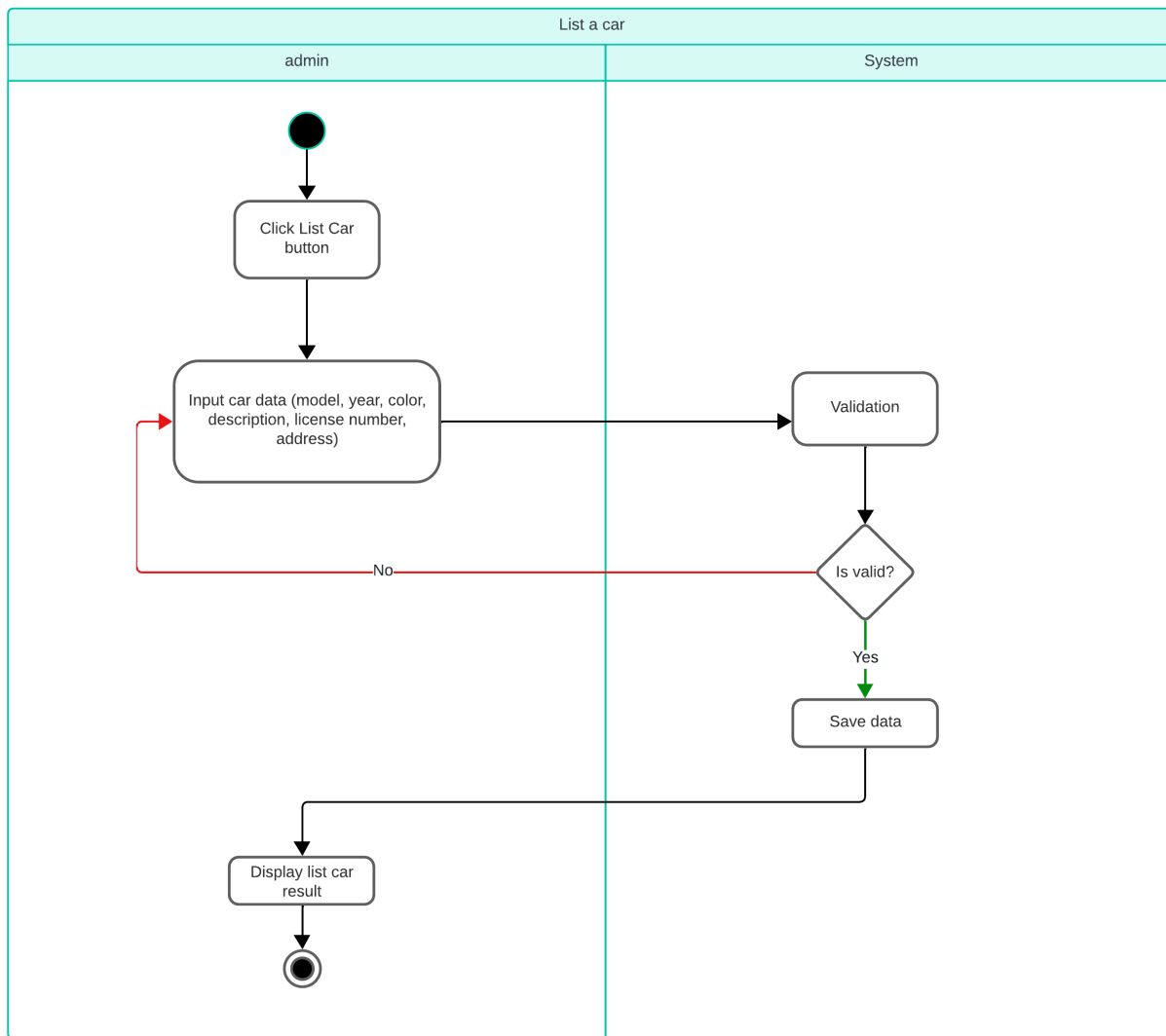
2-Class

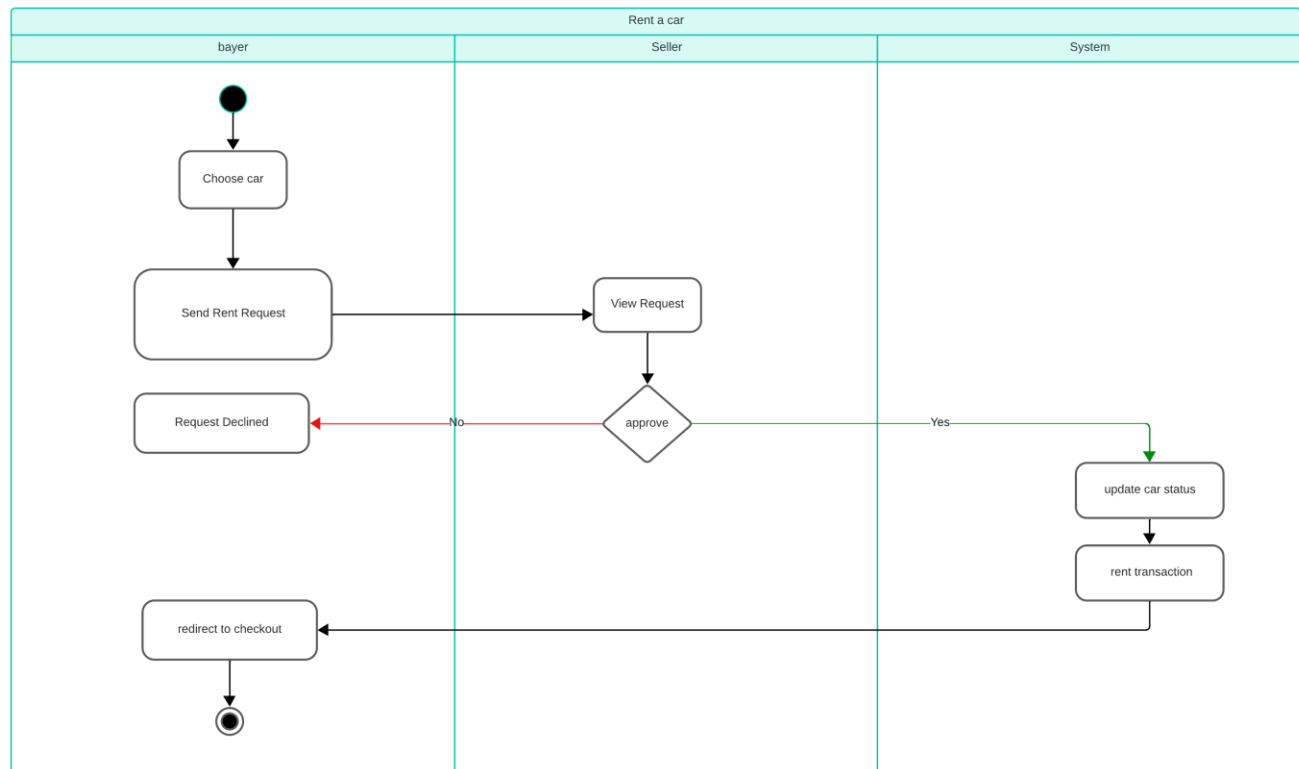


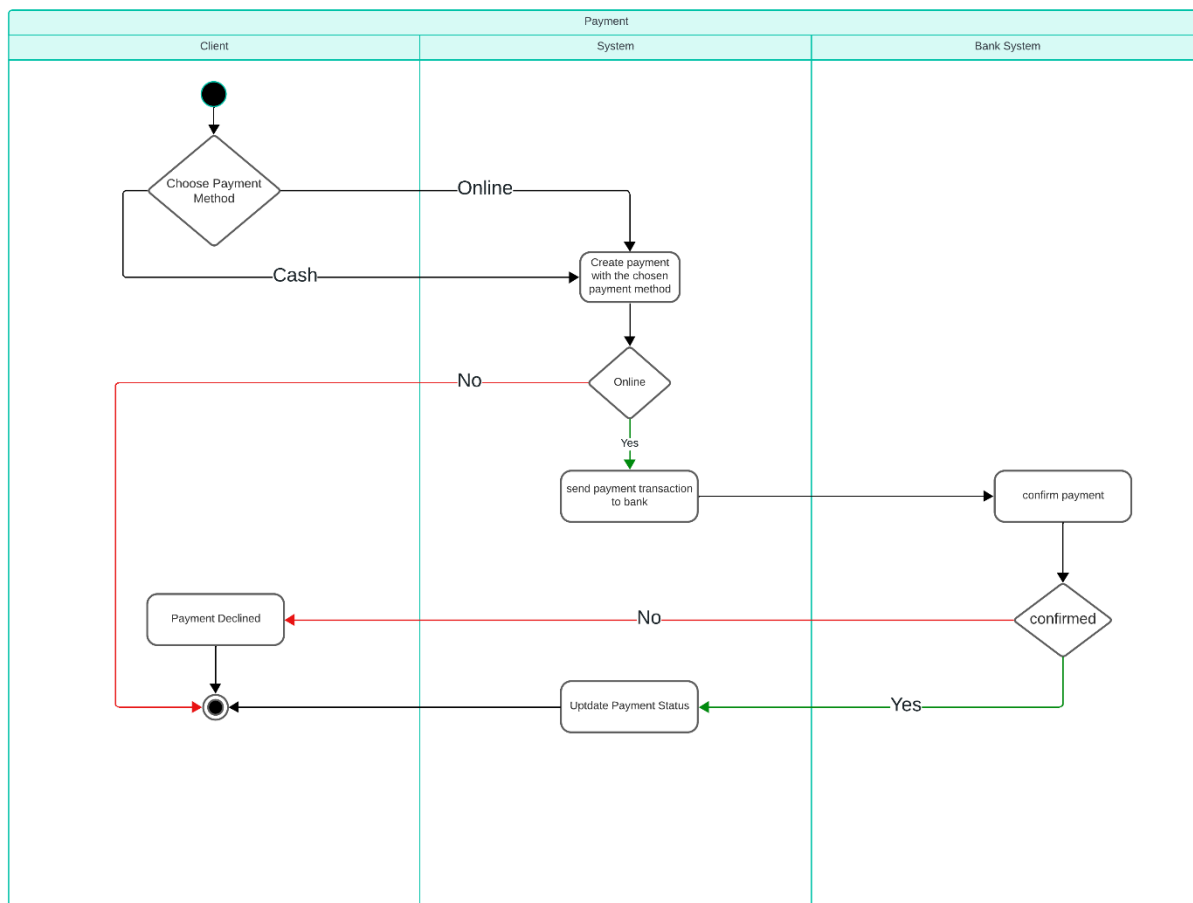
3-Activity



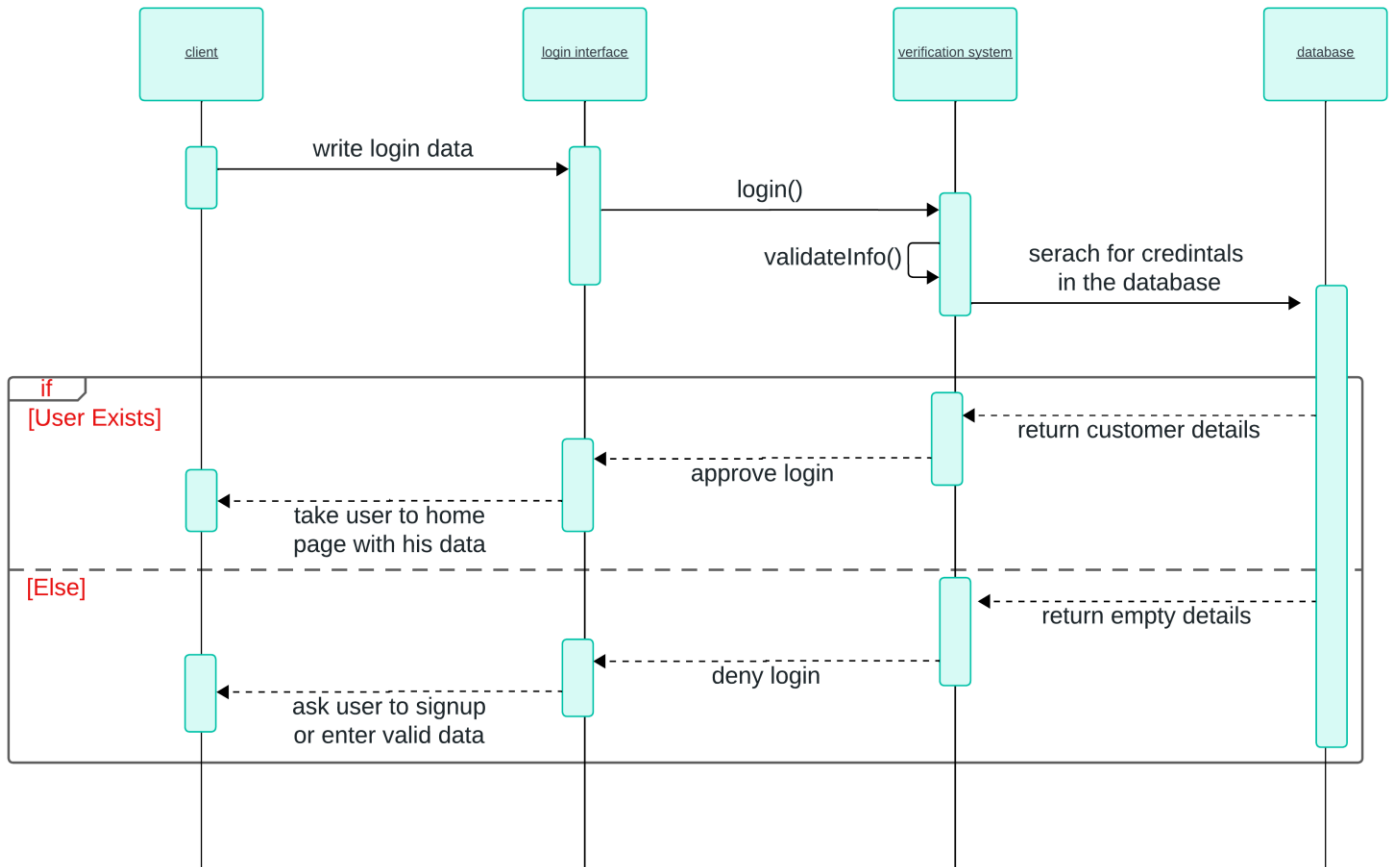


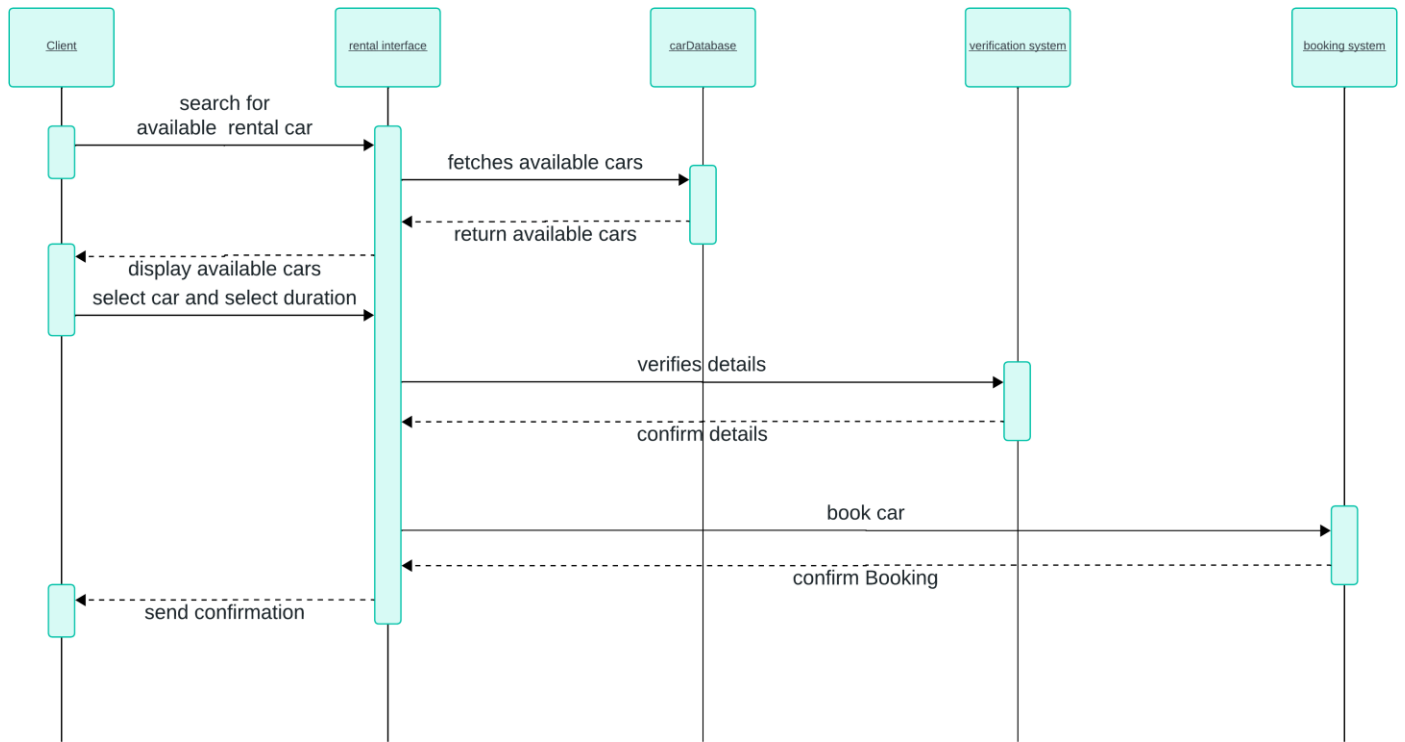






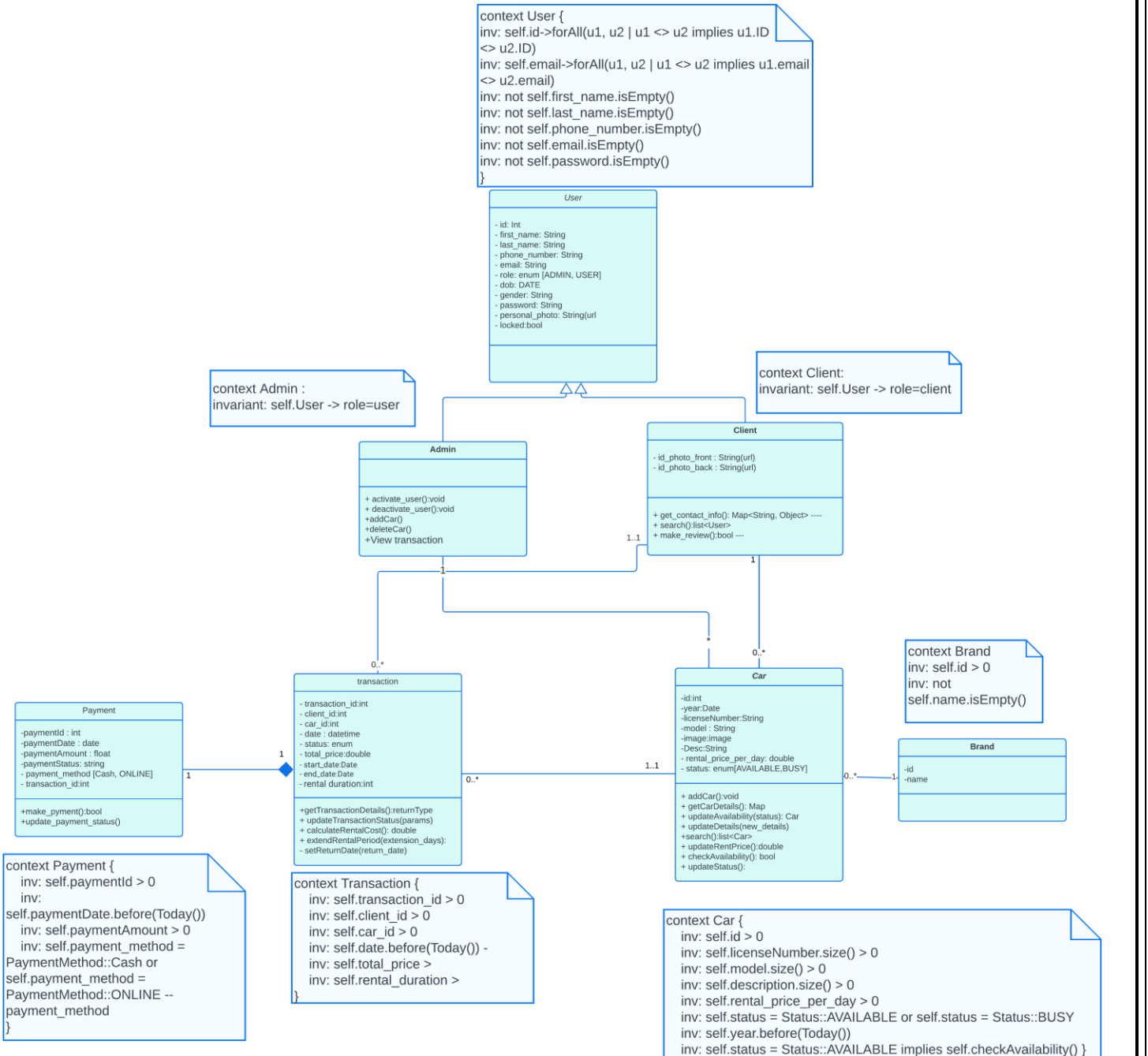
4- sequence





5-Erd

OCL



Team Members:

ID	NAME
20210715	مازن اسلام صبحي
20210694	كيرلس سامي عطا الله
20210695	كيرلس سليمان وهيب
20210712	مارينا عطاس
20210669	فام فايز
20210710	مارينا ايمن حنا
20210679	فيلوباتير عزت رزق الله

Code Repository:

https://github.com/kerolus77/car_module