Project Scope

Eduvos has commissioned the development of a Smart Parking Management System designed to optimize the utilization and management of their office parking garage. The system aims to alleviate key issues faced by management, including:

- Inefficient use of parking spaces
- Congestion caused by drivers searching for available spots

Key Features

Reservation System:
Clients can register on the Eduvos website to find and reserve parking spaces conveniently. During registration, clients must provide:

- Full name
- Valid email address
- Vehicle license plate number
- Reservation date and duration
- Credit card details

All client details are validated and stored securely. Clients may update their reservations up to 24 hours prior to the booking. If a vehicle's license plate differs from the client's profile (e.g., borrowed or rented vehicle), the system assigns a temporary license plate number valid for the reservation period.

Reservation Types:

- Daily Reservations
- Monthly Reservations

Garage Hardware Integration:

The parking garage will be equipped with:

- Two license plate readers (entrance and exit) using digital cameras
- Sensors on every parking spot to detect occupancy status
- Debit machines for processing additional payments

Payment Processing:

- Registered users pay the parking fee online at the time of reservation via credit or debit card.
- Unregistered users pay onsite at the debit machine, either with cash or card.
- Upon successful payment, receipt and reservation details are emailed to the client.
- Extra charges apply for overstaying beyond the reserved duration, payable at the debit machine before exit.

Contents

Actors

- Client
- Camera In
- Camera Out
- Debit Machine
- Sensor

Use Cases

- Register Client
- Make Reservation
- Make Payment
- Update Reservation
- Spot Occupied
- Open Gate

Elaborate Scenarios

Primary Scenario: The client goes to the website and attempts to register. They enter their details into the system. The system checks the details and confirms the booking for the parking spot with the client as well as the desired date and duration.

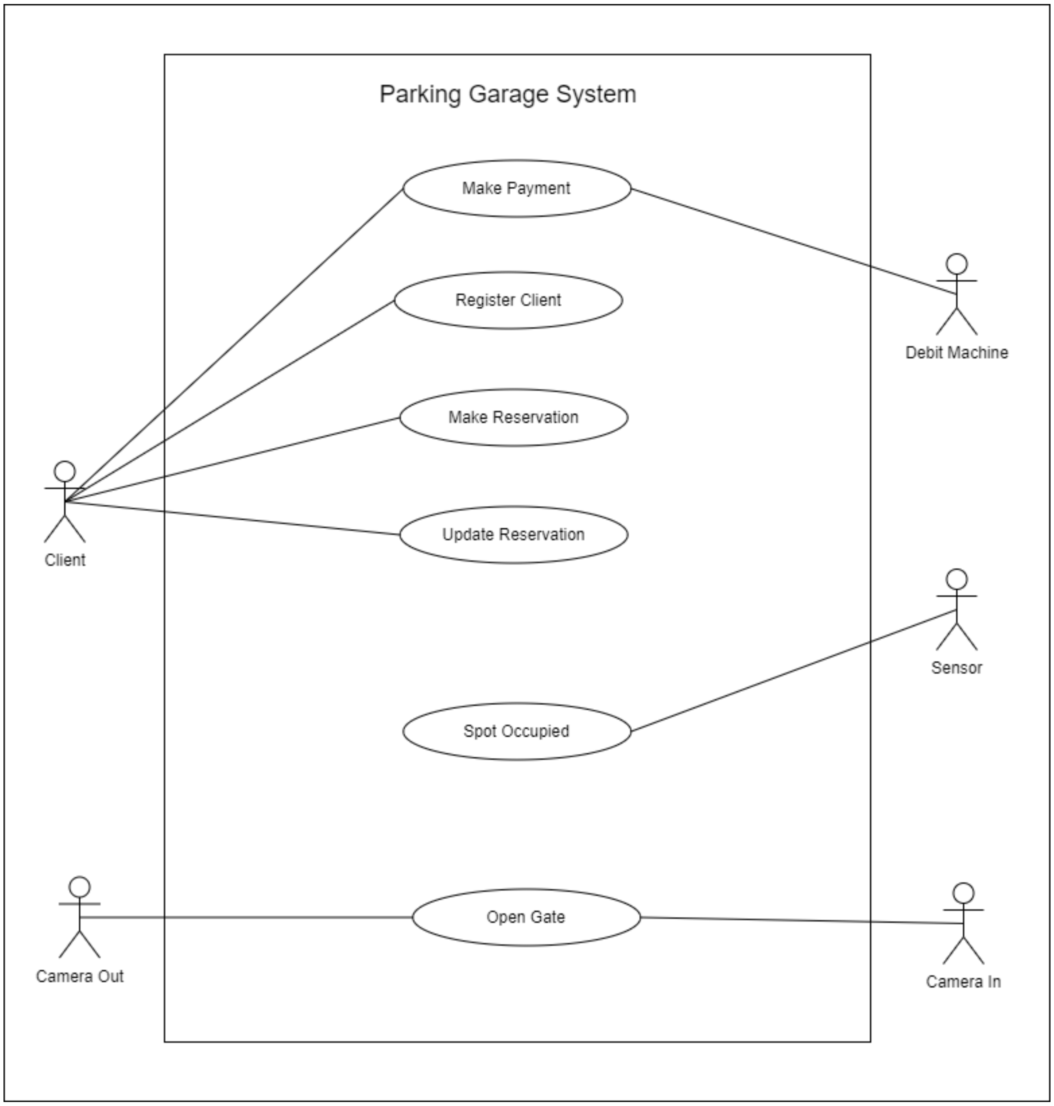
Alternative Scenario: The client enters incorrect details. The system will go back to the registration form and request the details again.

Primary Scenario: The client pays via the online payment gate. The payment is successful, and the account is debited. The parking spot is successfully reserved for the client. The client arrives on the day of the reservation and leaves the parking garage at the allotted time. The system then opens the gate for them.

Alternative Scenario: The client stays in the parking garage over the allotted time and now needs to pay at the debit machine for any overtime hour accrued before the system will open the gate for them.

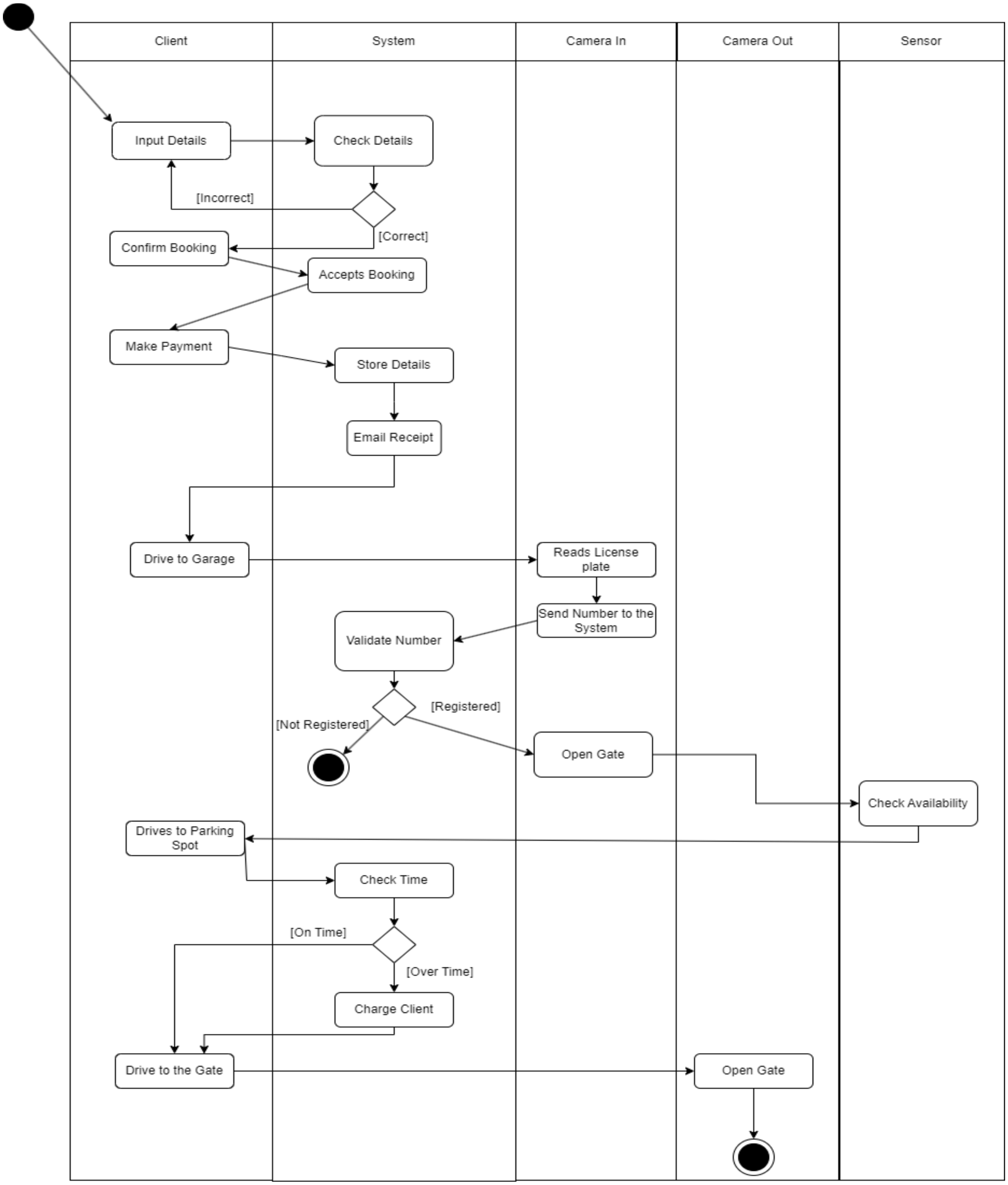
Use Case Diagram

Use Case Diagram Documentation:



The use case diagram shows the relationships between the actors and the use cases. The client is responsible for making the payment, however if the allotted time is exceeded the debit machine will also be involved in the payment. The client is also responsible for registering themselves, making their reservation and, if applicable, updating their reservation. The sensors are responsible for checking if the parking spots are occupied. The cameras both in and out are responsible for opening the gate.

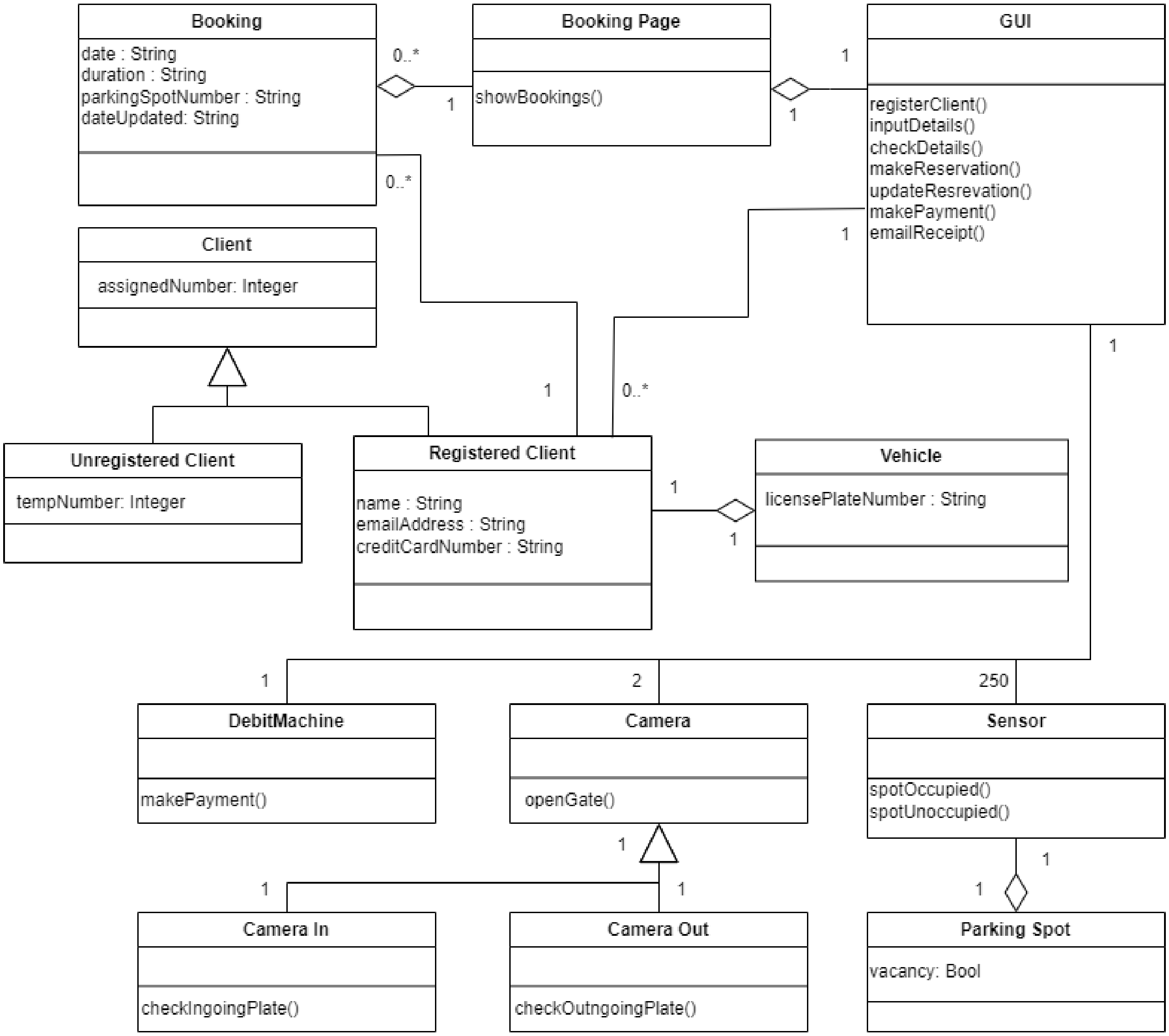
Activity Diagram



Activity Diagram Documentation:

The activity diagram shows the flow of the process, and the various activities performed by the client, the system the cameras and the sensor The client inputs their details then the system will check the details, if they are correct the system will allow the client to confirm their booking. The system will then accept the booking and ask the client to make the payment. Once the payment is successfully made the system will store the details and then email the receipt to the client. The client will then drive to the garage and the ingoing camera will read the license plate and send the number to the system. The system will then validate the license plate against the bookings. If it is registered the camera will open the gate and the sensors will check the availability for parking, the client will then drive to the parking spot and the system will stamp the time of entry and time of stay. If the client stays overtime the system will charge the client an extra amount. The client will then drive to the gate and the outgoing camera will open the gate.

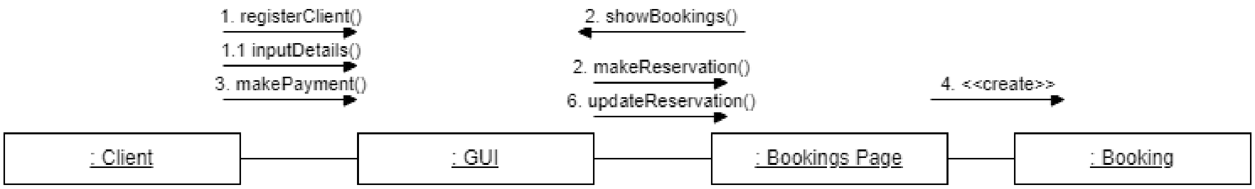
Class Diagram



Class Diagram Documentation:

The class diagram shows the different classes of objects and how they relate to each other. The GUI can register a client, input their details, check the details, make a reservation, update a reservation, make a payment and email a receipt. It has a relationship to the clients who have assigned numbers and the booking page which can show the bookings. The booking page and the client can have multiple bookings which store the details of the reservation. There are two types of clients a registered client who has comprehensive details and an unregistered client who is allocated a temporary number. The GUI also has a debit machine which can make payments, cameras and sensors in the parking garage. Each parking spot has a sensor. There are two types of cameras, the ingoing and outgoing cameras which can open the gate.

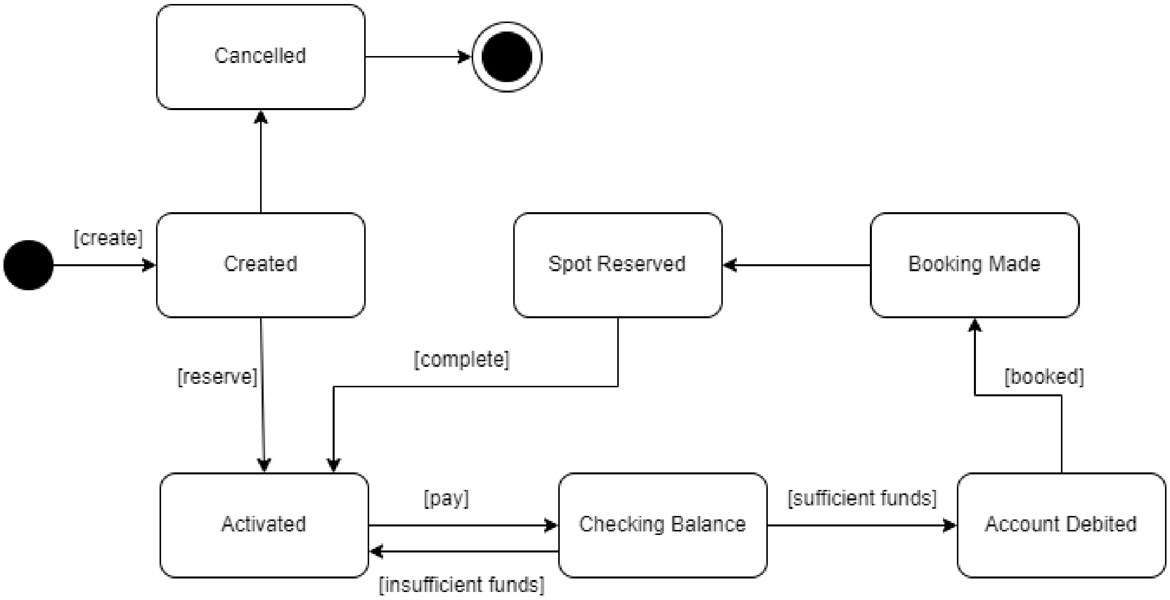
Communication Diagram



Documentation:

This communication diagram shows the how the different classes in the booking system communicate to one another in a sequential way. The client can register on the GUI and add their details, they can also make a payment on the GUI. The GUI makes reservations on the bookings page and can also update those reservations. The bookings page can show the bookings to the GUI and creates the bookings themselves.

State Machine Diagram



Documentation :

This state machine diagram shows the the states the account class goes through during the registration, payment and reservation processes. The account is created, it goes into the created state, once the spot is reserved the account goes into the activated state, however it can also be cancelled. The client then pays and the account goes into the checking balance state. If the funds are insufficient it returns to the activated state. If there funds are sufficient it goes into the account debited state and then once the spot is booked it goes into the booking made and then the spot reserved state. Once the customer exits and the stay is completed it returns to the activated state.

Software Methodology Research

Waterfall Model:

Description: The Waterfall Model is a linear and sequential approach where each phase must be completed before moving on to the next. It makes use of stages, mainly: requirements gathering, design, implementation, testing, deployment, and maintenance.

Best for: Projects where the requirements are well-understood and are unlikely to change.

Agile Methodology:

Description: An iterative approach that deals with collaboration, customer feedback, and small, rapid releases. Agile methodologies include frameworks like Scrum, Kanban, and Extreme Programming (XP).

Best for: Projects where the requirements are expected to evolve, and the project requires frequent feedback.

Scrum:

Description: Scrum is a type of Agile methodology; it makes use of short development cycles called "sprints". The teams work through prioritized tasks (backlogs) and then review the progress at the end of each sprint.

Best for: Teams who are looking for a structured and flexible framework to manage complex projects.

DevOps:

Description: DevOps uses software development and IT operations together to shorten the development lifecycle and deliver good-quality software continuously. It uses automation, continuous integration, and continuous delivery.

Best for: Projects that require the development cycles to be faster. It places an emphasis on the collaboration of the development and operations teams.

V-Model (Verification and Validation Model):

Description: The V-Model is an extension of the Waterfall model where each development stage is associated with a testing phase. The V-shape represents the development process as well as its parallel testing activities.

Best for: Projects where the requirements are well-defined, and testing is critical at each stage.

The Model I have chosen for delivering the system to Eduvos is the Scrum Framework of the Agile Methodology.

The Scrum Framework, a core component of Agile Methodology, is a highly iterative and incremental approach designed to manage complex projects with flexibility and adaptability. It is structured around small, cross-functional teams that work in time-boxed iterations called "sprints," typically lasting two to four weeks. During each sprint, the team focuses on delivering a potentially shippable product increment, allowing for continuous feedback and improvement. The framework includes key roles such as the Product Owner, who defines the project vision and prioritizes work; the Scrum Master, who facilitates the process and removes obstacles; and the Development Team, who are responsible for delivering the work. Scrum emphasizes transparency, inspection, and adaptation, with regular ceremonies like daily stand-ups, sprint planning, sprint reviews, and retrospectives to ensure alignment, address challenges, and refine the process. This approach is particularly effective in environments where requirements evolve rapidly, making it ideal for projects like the one at Eduvos, where collaboration, flexibility, and continuous improvement are crucial for success.

Motivation for Model Choice

- **Flexibility:** An Agile methodology will allow the team to adapt to the changes in the project requirements or client feedback faster, ensuring the final product meets the client's needs as they evolve.
- **Client Collaboration:** The Scrum framework makes use of close collaboration with the client, allowing Eduvos to be more involved in the development process and provides feedback on a regular basis.
- **Frequent Deliverables:** With Scrum's iterative style, the team can deliver fully functioning increments of the system to Eduvos at the end of each sprint. This will enable the client to see the progress as it happens and suggest adjustments earlier in the development process.
- **Risk Management:** By developing the system in smaller and more manageable increments, any potential issues or risks can be identified and addressed earlier in the process, reducing the likelihood of any major problems later on in the project.
- **Efficiency:** Scrum's structure, with time-boxed sprints and defined roles, ensures that the team remains focused and productive throughout the process and helps to deliver a high-quality system within the timeframe that was agreed to.

These factors make Agile Scrum the most effective methodology for delivering a working system like this to Eduvos. It ensures that the project is completed timeously, successfully and meets all the client's expectations.