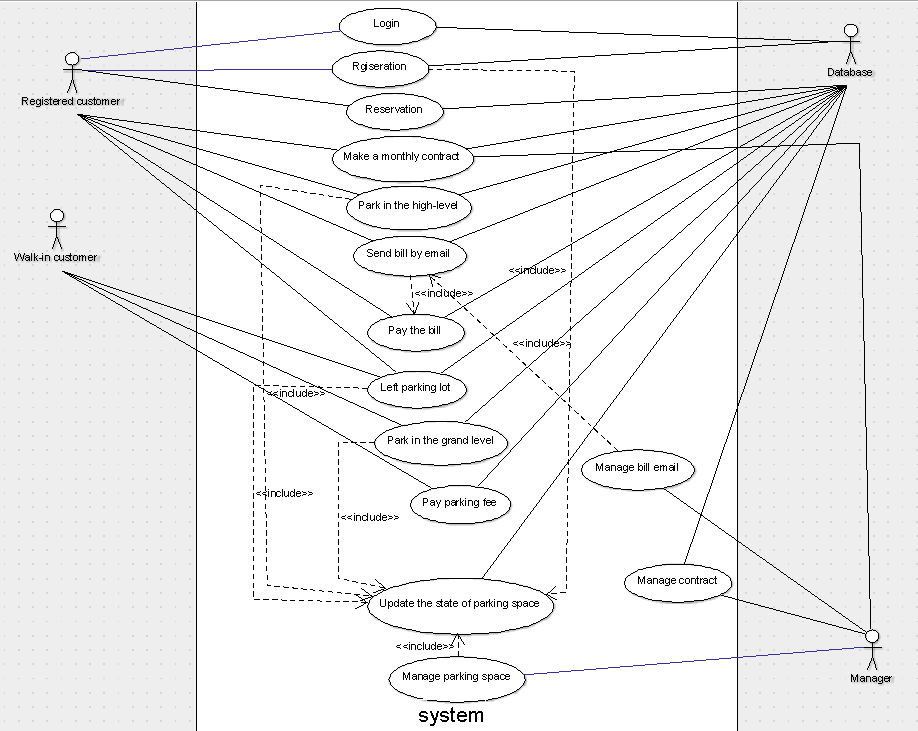
**Software Quality**

**Deliverable 1.2**

**Requirement**

1. **Use case diagram**
2. **Use case elaboration**

|  |  |
| --- | --- |
| Name | Login |
| ID | 1 |
| Description | Register customer login the reservation system |
| Primary Actor | Register customer |
| Secondary Actor(s) |  |
| Pre-condition | The database contains the user registration information |
| Post-condition |  |
| Trigger | User wants to make a reservation |

|  |  |
| --- | --- |
| Name | Registeration |
| ID | 2 |
| Description | Customer register as user in the system |
| Primary Actor | Register customer |
| Secondary Actor(s) |  |
| Pre-condition | User does not exist |
| Post-condition | User’s information saves in the database |
| Trigger | Customer wants to access to the reservation system |

|  |  |
| --- | --- |
| Name | Reservation |
| ID | 3 |
| Description | Register customer can make parking reservation in the system. |
| Primary Actor | Register customer |
| Secondary Actor(s) |  |
| Pre-condition | 1. 1. There is empty parking space in the parking lot 2. Register customer need provide parking plan and plate number |
| Post-condition | 1. Parking plan and plate number save in the database 2. The state of parking space has been updated to “reserved” |
| Trigger | Register customer have a plan to park in the parking lot |

|  |  |
| --- | --- |
| Name | Park in the high-level |
| ID | 4 |
| Description | Register customer can park in the high-level of the parking lot. |
| Primary Actor | Register customer |
| Secondary Actor(s) |  |
| Pre-condition | 1. Customer must be register customer 2. Register customer has made a reservation 3. Register customer provide parking plan and plate number at the entrance of the parking lot. 4. There is empty parking space in the high-level of parking lot. |
| Post-condition | 1. Parking plan and plate number save in the database 2. The state of parking space has been updated to “used” |
| Trigger | The sensor at the entrance detects the register customer |

|  |  |
| --- | --- |
| Name | Send bill by email |
| ID | 5 |
| Description | The system sends bill to register customer every month. |
| Primary Actor | Database |
| Secondary Actor(s) |  |
| Pre-condition | 1. Mail recipient must be register customer 2. Monthly fixed date |
| Post-condition | Register customer accepts the bill email |
| Trigger | The system sends email automatically by the fixed date of each month |

|  |  |
| --- | --- |
| Name | Pay the bill |
| ID | 6 |
| Description | Register customer can read and pay the bill by clicking the link in the email. |
| Primary Actor | Register customer |
| Secondary Actor(s) |  |
| Pre-condition | Register customer accepts the bill email. |
| Post-condition |  |
| Trigger | Register customer clicks the link in the email |

|  |  |
| --- | --- |
| Name | Left parking lot |
| ID | 7 |
| Description | Customer left the parking lot via exit |
| Primary Actor | 1. Register customer 2. Walk-in customer |
| Secondary Actor(s) |  |
| Pre-condition | 1. Customer is register customer 2. Walk-in customer has paid the parking fee |
| Post-condition | 1. The state of parking space has been updated to “empty” 2. The parking fee charged to bill |
| Trigger | The sensor at the exit detects the leaving customer |

|  |  |
| --- | --- |
| Name | Park in the grand level |
| ID | 8 |
| Description | Walk-in customer can only park in the grand level of the parking lot |
| Primary Actor | Walk-in customer |
| Secondary Actor(s) |  |
| Pre-condition | There is empty state parking space in the grand level of parking lot |
| Post-condition | The state of parking space has been updated to “used” |
| Trigger | The sensor at the entrance detects arrived customer |

|  |  |
| --- | --- |
| Name | Pay parking fee |
| ID | 9 |
| Description | Walk-in customer should pay parking fee before they leaving the parking lot |
| Primary Actor | Walk-in customer |
| Secondary Actor(s) |  |
| Pre-condition | Walk-in customer stays at the exit of parking lot |
| Post-condition |  |
| Trigger | The sensor at the exit detects the leaving customer |

|  |  |
| --- | --- |
| Name | Update the state of parking space |
| ID | 10 |
| Description | 1. The state of parking space will be change if the customer left the parking space 2. Manager can update the state of parking space in real time |
| Primary Actor | 1. Register customer 2. Walk-in customer |
| Secondary Actor(s) | Manager |
| Pre-condition | 1. Customer make reservation, park or leave 2. Manager update the state manually |
| Post-condition | The state of parking space has been updated |
| Trigger | Reservation, parking or leaving |

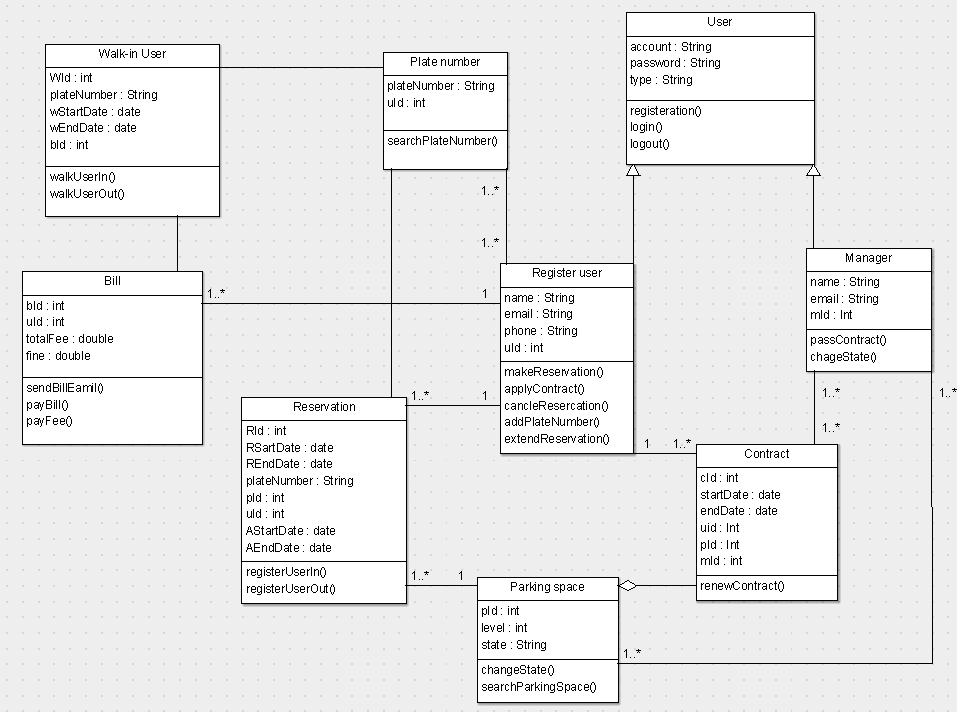
|  |  |
| --- | --- |
| Name | Manage parking space |
| ID | 11 |
| Description | Manager can read and search the state of parking space |
| Primary Actor | Manager |
| Secondary Actor(s) |  |
| Pre-condition |  |
| Post-condition |  |
| Trigger | Manager stays in the page of managing parking space |



|  |  |
| --- | --- |
| Name | Manage bill email |
| ID | 12 |
| Description | Manager sets the temple of the bill email and which date of every month to send the bill email |
| Primary Actor | Manager |
| Secondary Actor(s) |  |
| Pre-condition |  |
| Post-condition | The temple of email and date have been saved in the database |
| Trigger | Manager stays in the page of managing bill email. |

|  |  |
| --- | --- |
| Name | Make a monthly contract |
| ID | 13 |
| Description | Register customer can apply for a monthly contract with manager. Manager need to review the contract and pass the application. If register customer makes a contract, the fixed parking space will be reserved for him permanently |
| Primary Actor | Register customer |
| Secondary Actor(s) | Manager |
| Pre-condition | 1. Customer must be register customer 2. Manager passes the application |
| Post-condition | 1. The information of contract has been saved in the database 2. The parking space relates to the register customer |
| Trigger | Register customer apply for a contract |

|  |  |
| --- | --- |
| Name | Manage contract |
| ID | 14 |
| Description | Manager can read and search contract. |
| Primary Actor | Manager |
| Secondary Actor(s) |  |
| Pre-condition |  |
| Post-condition |  |
| Trigger | Manager stays in the page of managing contract |

1. **UML Class Diagram**

**Quality Plan**

1. **Quality goals and metrics [1]:**

|  |  |  |  |
| --- | --- | --- | --- |
| 1. Product Quality | Quality Goals | Quality Metrics | Strategy |
| Product Operation |  |  |  |
| Correctness | The system shall meet the specifications | The functionality issue reports should less than 5 in the whole lifecycle | Provide beta version to allow customers to experience and provide feedback |
| Reliability | The system shall work accuratly all of the time | The crashing frequency should be less than 3 per year | Stress testing |
| Usability | The system should be easy to learn and operate | A new user should be able to start using it with less than a day’s training | Provide beta version to allow customers to experience and provide feedback |
| Integrity | They system should not be tampered | The data leakage should be less than 1 in 3 years | Unit test on normal cases and boundary cases |
| Efficiency | The system should quickly solve the intended problems | The system should finish all the data query and manipulation in less than one second | Unit test on normal cases and boundary cases |
| Product Revision |  |  |  |
| Maintainability | The bugs and errors can be easily fixed | The source code should be readable, easily traced back to documents. | Code Review |
| Flexibility | The system can be easily changed | The changing in some parts should not affact the whole running. | Objected Oriented Design |
| Testability | The system should be testable | Every class must available for white and black box testing. | Mutation and Unit testing tools |
| Product Transition |  |  |  |
| Reusability | Parts of the system should be able to be reused in another system | Modules in the system can be used in another system in less than 1 week configuring | Standard interface design and Objected Oriented Design |
| Portability | The system should be easily moved to a new platform | The system should work correctly on different models of devices | Standard interface design |
| Interoperability | The system should be able to interaction with other systems | The system should be able to work with other data management systems | Standard interface design |

**Priority of quality goals: (high->low)**

1. Correctness

2. Maintainability, Testability

3. Usability, Efficiency, Reliability

4. Flexibility, Portability

5. Integrity

6. Reusability, Interoperability

**Additional notes:** To finish the project in a short time, our team weighed the priority of quality goals as such a way: Our main goal of this project is to finish a functional and usable product that satisfies the project specification. Thus **Correctness** is the most important metric in our project. If we cannot provide a usable software, the project is meaningless. Then, we have to make sure we can finish the project efficiently. Thus **Maintainability** and **Testability** is of secondary importance. Then, we have to make sure the user experience not too bad. Thus **Usability**, **Efficiency**, **Reliability** must be concerned. **Flexibility** and **Portability** also affect the efficiency and quality of our project. **Integrity** is not a big concern in our project. As we do not have the requirements for reusing and interoperating with other systems, Reusability and Interoperability take a back seat for this project.

1. **Costs of quality**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Task Name | Estimated Effort(hrs) | Implementation | Evaluation | Prevention |
| Project Planning | 30 | 25 | 0 | 0 |
| Simulation Environment Construction | 10 | 8 | 2 | 0 |
| User Interface Design | 30 | 20 | 5 | 5 |
| Database Settings | 20 | 15 | 2 | 3 |
| Garage Access Control | 30 | 20 | 4 | 6 |
| Monitoring of Occupancy and Space Reassignment | 40 | 30 | 5 | 5 |
| Simulation of Arrivals and Departures | 40 | 30 | 5 | 5 |
| Data Collection | 20 | 15 | 5 | 0 |
| System Administration | 25 | 20 | 2 | 3 |
| System Testing | 29 | 15 | 4 | 10 |
| Software Bug Correction | 28 | 5 | 3 | 20 |
| Total | 302 | 203 | 37 | 57 |