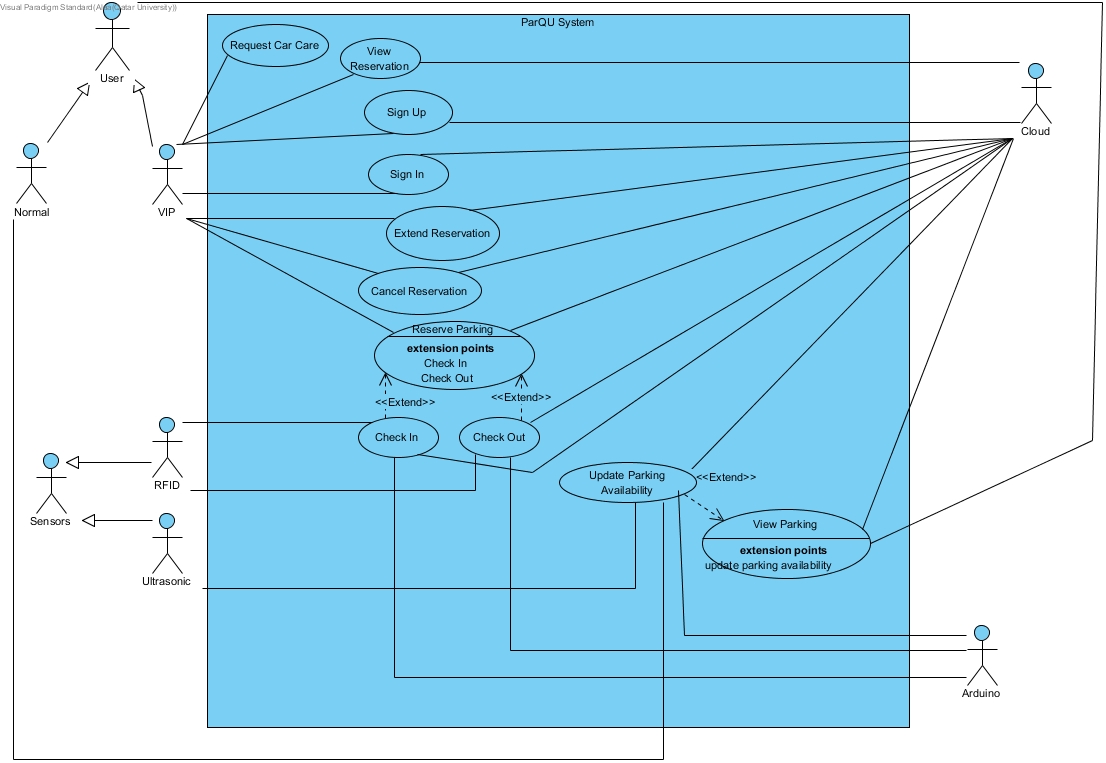
Use case diagram



# Appendix A – Use cases specification

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**01** | Register/ Sign-up | |
| **Brief Description:** | VIP user registers in the system. | |
| **Primary Actors:** | VIP user, Cloud | |
| **Trigger:** | VIP user asks to register in the system. | |
| **Preconditions:**   * VIP user must have a car plate number. | | |
| **Post-conditions:**   * A new record is created for the VIP user in the database. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user enters his/her data (ID, name, car plate number, etc.). | | 2. check for the validity of the information. (see 2.a) |
|  | | 3. Create the account and save it in the cloud. |
| **Alternative Flows:**  2.a. Ask VIP user to renter any missing information. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**02** | Login / Sign-in | |
| **Brief Description:** | VIP user logs into the system. | |
| **Primary Actors:** | VIP User, Cloud | |
| **Trigger:** | VIP user asks to login to the system. | |
| **Preconditions:**   * The VIP user has a valid account. | | |
| **Post-conditions:**   * The VIP user is logged into the system. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user enters username and password. | | 2. Validate the entered username and password. with stored username and password in the cloud. (See 2.a.) |
|  | | 3. log VIP user into the system. |
| **Alternative Flows:**  2.a. If VIP user enters an invalid username and/or password, the system displays an error message. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**03** | Reserve Parking | |
| **Brief Description:** | VIP user reserves a parking spot for several hours. | |
| **Primary Actors:** | VIP user, Cloud. | |
| **Trigger:** | VIP user asks to reserve a parking spot. | |
| **Preconditions:**   * VIP user must be registered to the system. | | |
| **Post-conditions:**   * Reservation is created. * Total number of available parking at the time of reservation is updated. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user selects parking area. | |  |
| 2. VIP user selects zone. | |  |
| 3. VIP user selects a date. | | 4. Request parking data from cloud. |
| 5. Cloud sends data. | | 6. Display if there are available parking spots at selected zone for each hour. |
| 7. VIP user selects start time and duration. | | 8. Check if the number of selected hours with the total reservation hours for this date is less than or equal to the number of allowable reservation hours per day. (see 8.a) |
|  | | 9. Check if there is an available parking spot at all selected hours. (see 9.a) |
|  | | 10. Add selected reservation hours to the total reservation hours in this date. |
|  | | 11. Create a reservation record in the cloud. |
|  | | 12. Calculate the total price and add it to the VIP user account. |
|  | | 13. Update the total number of available parking in the cloud. |
|  | | 14. Send confirmation of reservation to the VIP user. |
| **Alternative Flows:**  8.a. If the number of selected hours with the total reservation hours for this date is more than the number of allowable reservation hours per day, display error message.  9.a. If there is no available parking spot at one of the selected hours, display error message. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**04** | View Reservation | |
| **Brief Description:** | The VIP user views a reservation. | |
| **Primary Actors:** | VIP user, Cloud | |
| **Trigger:** | VIP user asks to view reservation. | |
| **Preconditions:**   * The VIP user must have a reservation. | | |
| **Post-conditions:**   * Reservation details was displayed to the VIP user. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user asks to view a reservation. | | 2. Retrieve reservations related to the VIP user and display it. (see 2.a) |
| 3. VIP user selects one of the reservations. | | 4. Retrieve and display all the information related to the reservation such as date, start time, duration, zone name and price. |
| **Alternative Flows:**  2.a. If there are no reservations associated with the user, display a message. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**05** | Extend Reservation | |
| **Brief Description:** | The VIP user extends a reservation and the extension time is by default one hour. | |
| **Primary Actors:** | VIP user, Cloud | |
| **Trigger:** | VIP user asks to extend reservation. | |
| **Preconditions:**   * The VIP user must have reservation. | | |
| **Post-conditions:**   * Reservation status was extended. * Total number of available parking was updated. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user selects extend option. | | 2. Retrieve current reservation related to the VIP user and display it. |
| 3. VIP user confirms reservation extension. | | 4. Check if there is an available parking at the end of the current reservation time. (see 4.a) |
|  | | 5. Add one hour to the end of the current reservation time. |
|  | | 6. Change the status of the reservation to extended in the cloud. |
|  | | 7. Add the extension price to the VIP user account. |
|  | | 8. Update the total number of available parking in the cloud. |
| **Alternative Flows:**  4.a. If there is no available parking, displays a message to the VIP user. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**06** | Cancel Reservation | |
| **Brief Description:** | The VIP user cancels a reservation. | |
| **Primary Actors:** | VIP user, Cloud | |
| **Trigger:** | VIP user requests to cancel reservation. | |
| **Preconditions:**   * The VIP user must already have a reservation. | | |
| **Post-conditions:**   * Reservation status is cancelled. * Total number of available parking is updated. * The price of the reservation is deducted from the VIP user account. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1.VIP user asks to cancel reservation. | | 2. Retrieve his reservations that have a start time is greater than current time by 1 hour and display them. |
| 3.VIP user selects one of the reservation number. | | 4. Change the status of the reservation to cancelled in the cloud. |
|  | | 5. Update the total number of available parking in the cloud. |
|  | | 6. Calculate the deducted amount and deduct it from the VIP user account. |
|  | | 7. Send notification of cancellation to the VIP user. |
| **Alternative Flows:** | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**07** | VIP Check-In | |
| **Brief Description:** | The system reads the RFID tag on the car, checks its validity, then allows reserved VIP users only to enter the parking lot. | |
| **Primary Actors:** | RFID Sensor Reader, Arduino Board | |
| **Trigger:** | RFID reader reads a car tag at the gates of the parking lot. | |
| **Preconditions:**   * The parking lot must have at least one reserved available parking spot. * The VIP user must be registered to the system. * The VIP user must have reserved a parking spot. | | |
| **Post-conditions:**   * The parking area gate is opened for the VIP user. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. RFID Sensor Reader senses a car tag at the parking lot gates. | |  |
| 2. Arduino receives ID data from reader and sends it to the cloud. | | 3. Check if ID exist in the cloud database. (see 3.a) |
|  | | 4. Retrieve all associated reservation with ID for that date from cloud. |
|  | | 5. Check if there is a reservation at this time. (see 5.a) |
|  | | 6. Inform the Arduino to open the gate. |
| 7. Arduino opens the gate. | | 8. Update that the VIP user has arrived for his reservation. |
| **Alternative Flows:**  3.a. If ID does not exist, the system informs the Arduino to not open the gates.  5.a. If the VIP user has not reserved a parking spot, the system informs the Arduino to not open the gates. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**08** | VIP Check-Out | |
| **Brief Description:** | The system reads the RFID tag on the car, checks its validity, then update available parking spots. | |
| **Primary Actors:** | RFID Sensor Reader, Arduino Board | |
| **Trigger:** | RFID reader reads a car tag at the gates of the parking lot. | |
| **Preconditions:**   * The VIP user must be registered to the system. * The VIP user must have reserved a parking spot. * The VIP user must have arrived to his reservation. | | |
| **Post-conditions:**   * Update available parking spots. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. RFID Sensor Reader senses a car tag at the parking lot gates. | |  |
| 2. Arduino receives ID data from reader and sends it to the cloud. | | 3. Retrieve the associated reservation with ID. |
|  | | 4. Check if the VIP user did not exceed the limit time that he reserved. (see 4.a) |
|  | | 5. Inform the Arduino to open the gate. |
| 6. Arduino opens the gate. | | 7. Update available parking spots. |
| **Alternative Flows:**  4.a. If the VIP user exceed the limit time, the system calculates and add penalty to the VIP user account. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**09** | View Parking | |
| **Brief Description:** | The user views a map with the current status of parking spots. | |
| **Primary Actors:** | User, Cloud | |
| **Trigger:** | User asks to view current available parking spots. | |
| **Preconditions:** | | |
| **Post-conditions:**   * A map with the current status of parking spots is displayed to the user. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. User select parking zone. | | 2. Retrieve zone data from cloud. |
| 3. Cloud sends data. | | 4. Display a map with the current status of parking spots. |
| **Alternative Flows:** | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**10** | Update Parking Availability | |
| **Brief Description:** | The system receives updated data from sensors through the Arduino board. | |
| **Primary Actors:** | Sensor, Arduino Board, Normal user | |
| **Trigger:** | Sensor senses a car entering or leaving the parking spot. | |
| **Preconditions:** | | |
| **Post-conditions:**   * Number of available parking spots is updated. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. Normal user enters or leaves a parking spot. | |  |
| 2. Sensor senses a car entering or leaving the parking spot. | |  |
| 3. Arduino receives data from sensors and sends it to the cloud. | | 4. Receive data from cloud. |
|  | | 5. Change status of the parking spot to unavailable or available. |
|  | | 6. Decrement or increment the available parking spots. |
| **Alternative Flows:** | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**11** | Request Car Care | |
| **Brief Description:** | The VIP user request car care services. | |
| **Primary Actors:** | VIP user | |
| **Trigger:** | VIP user asks to request car care service. | |
| **Preconditions:**   * VIP user must be registered to the system. | | |
| **Post-conditions:**   * VIP user is presented with car care contact info. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user asks to request car care service. | | 2. Display companies that provide this service. |
| 3. VIP user selects one of the companies. | | 4. Display phone numbers of company. |
| 5. VIP user calls company. | |  |
| **Alternative Flows:** | | |