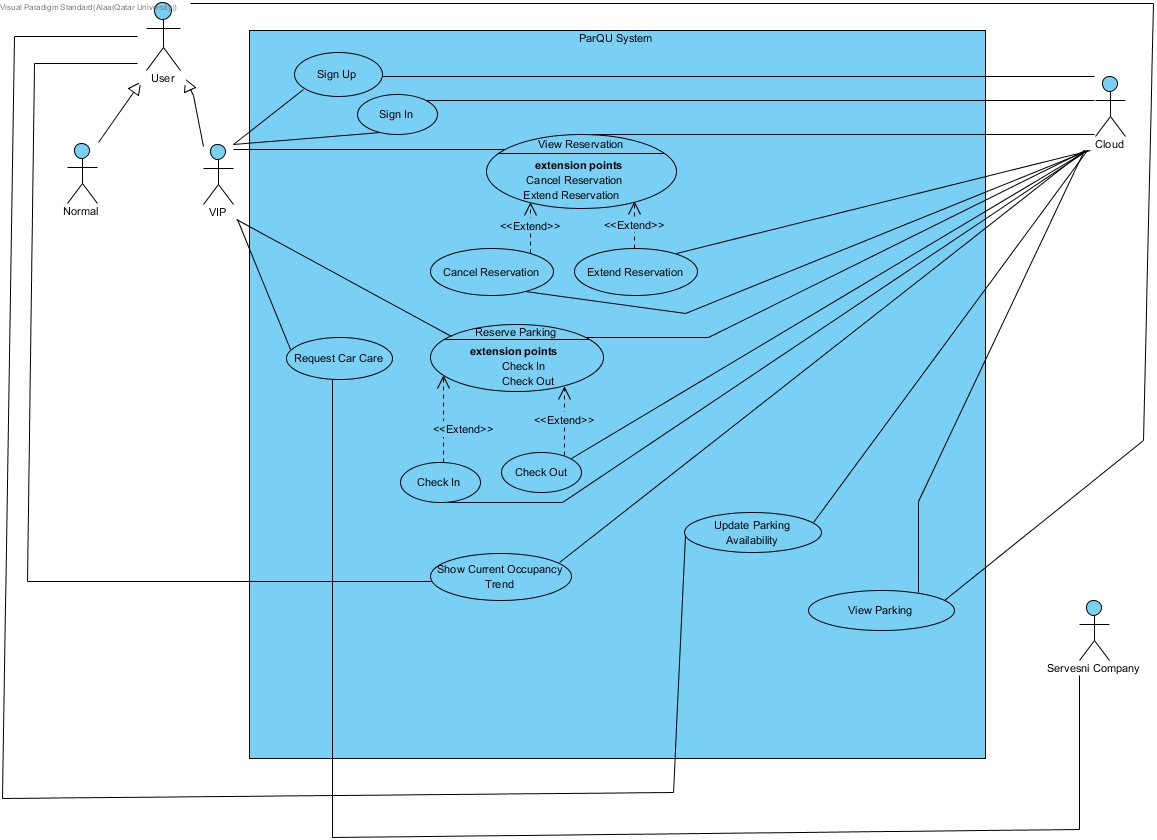
# Use case diagram



# 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 | |
| **Secondary Actors:** | 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 | |
| **Secondary Actors:** | 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 | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | VIP user asks to reserve a parking spot. | |
| **Preconditions:**   * VIP user must be registered to the system. | | |
| **Post-conditions:**   * Reservation is created. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user selects zone. | |  |
| 2. VIP user selects a date. | | 3. Request parking data from cloud. |
| 4. Cloud sends data. | | 5. Display if there are available parking spots at selected zone for each hour. |
| 6. VIP user selects start time and duration. | | 7. Check if the user does not have a reservation at the selected hours. (see 7.a) |
|  | | 8. Check if there is an available parking spot at all selected hours. (see 8.a) |
|  | | 9. Check if selected date is equal to today or tomorrow (selected date == current date || selected date == current date +1). (see 9.a) |
|  | | 10. If selected date is today, check if the reservation start time has passed. (selected start hour > current hour). (see 10.a) |
|  | | 11. 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 (6 hours). (see 11.a) |
|  | | 12. Create a reservation record in the cloud. |
|  | | 13. Calculate the total price and assign it to the reservation. |
|  | | 14. Assign “created” to the reservation status. |
|  | | 15. Start background notification service that notifies the VIP user 30 minutes before expiring time. |
| **Alternative Flows:**  7.a. If the user has a reservation at the selected time, display error message.  8.a. If there is no available parking spot at one of the selected hours, display error message.  9.a. If selected date is not equal to the current date or greater than the current date by more than 1 day, display error message to inform user that he can only reserve at the same day or one day before the reservation date.  10.a. If the selected start hour is less than or equal to current hour, display error message to inform user that selected start time has elapsed.  11.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. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**04** | View Reservation | |
| **Brief Description:** | The VIP user views a reservation. | |
| **Primary Actors:** | VIP user | |
| **Secondary Actors:** | 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. Request current and upcoming reservations related to the VIP user. |
| 3. Cloud sends data. | | 4. Display requested data. (see 4.a) |
|  | | 5. If user select extend option, <extend: Extend Reservation use case>. |
|  | | 6. If user select cancel option, <extend: Cancel Reservation use case>. |
| **Alternative Flows:**  4.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 | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | VIP user asks to extend reservation. | |
| **Preconditions:**   * The VIP user must have reservation. | | |
| **Post-conditions:**   * Reservation status was extended. * Total price was updated. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user selects extend option. | | 2. Check if the selected reservation is at its last hour (current hour == last hour of reservation). (see 2.a) |
|  | | 3. Check if there is an available parking spot after the reservation time. (see 3.a) |
|  | | 4. Add one hour to the end of the current reservation time. |
|  | | 5. Change the status of the reservation to “extended” in the cloud. |
|  | | 6. Add the extension price to the reservation. |
| **Alternative Flows:**  2.a. if the current hour is not equal to the last hour of the selected reservation time., display an error message to inform VIP user that he can only extend his reservation in the last hour of his reservation.  3.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 | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | VIP user requests to cancel reservation. | |
| **Preconditions:**   * The VIP user must already have a reservation. | | |
| **Post-conditions:**   * Reservation status is cancelled. * The price of the reservation is deducted. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1.VIP user selects cancel option. | | 2. Check if the reservation has not started in order to cancel the whole reservation (current time < start time). (see 2.a) |
|  | | 3. Change the status of the reservation to “cancelled” or “subcancelled” in the cloud. |
|  | | 4. Calculate the deducted amount (depending on the number of hours cancelled) and deduct it from the total price of the reservation. |
| **Alternative Flows:**  2.a. If the reservation has started, cancel remaining reservation hours by going to step 3 with status “subcancelled”. | | |

|  |  |  |
| --- | --- | --- |
| **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 to enter the parking lot. | |
| **Primary Actors:** | VIP user | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | VIP user arrives 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. * Reservation status is updated to “arrived” | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user arrives at the gates of the parking lot. | | 2. Read the car tag (UID) at the parking lot gates by RFID sensor reader. |
|  | | 3. Receive UID from reader by Arduino and send it with the zone name to NodeMCU. |
|  | | 4. Request all the reservations with a selected UID and zone from the cloud by NodeMCU. |
| 5. Cloud sends data. | | 6. Search for all the reservations where date is today by NodeMCU. (see 6.a) |
|  | | 7. Search for all the reservations where status is “created” by NodeMCU. (see 7.a) |
|  | | 8. Find a reservation where the current hour is at the reservation time by NodeMCU (start time <= current hour <= last hour). (see 8.a) |
|  | | 9. Change reservation status to “arrived” by NodeMCU |
|  | | 10. Send response (“open” or “close”) to Arduino by NodeMCU. |
|  | | 11. Check response in order to open the gate or not by Arduino. |
| **Alternative Flows:**  6.a. If no reservation exists, go to step 10 with “close” response.  7.a. If no reservation exists, go to step 10 with “close” response.  8.a. If no reservation exists, go to step 10 with “close” response. | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**08** | VIP Check-Out | |
| **Brief Description:** | The system reads the RFID tag on the car, checks its validity, then allow VIP users to exit the parking lot | |
| **Primary Actors:** | VIP user | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | VIP user exits 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:**   * The parking area gate is opened for the VIP user. * Reservation status is updated to “ended” | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user arrives at the gates of the parking lot. | | 2. Read the car tag (UID) at the parking lot gates by RFID sensor reader. |
|  | | 3. Receive UID from reader by Arduino and send it with the zone name to NodeMCU. |
|  | | 4. Request all the reservations with a selected UID and zone from the cloud by NodeMCU. |
| 5. Cloud sends data. | | 6. Search for all the reservations where date is today by NodeMCU. (see 6.a) |
|  | | 7. Search for all the reservations where status is “arrived” or “extended” or “subcancelled” by NodeMCU. (see 7.a) |
|  | | 8. Check if the reservation has more than or equal to 1 hour left by NodeMCU (current hour < end time) (see 8.a) |
|  | | 9. Apply automatic cancellation deduction by NodeMCU, then go to step 13 |
|  | | 10. Find a reservation where its time has passed by NodeMCU (current hour >= end time) (see 10.a) |
|  | | 11. Apply penalty for exceeding reservation time by NodeMCU. |
|  | | 12. Change reservation status to “ended” by NodeMCU. |
|  | | 13. Send response (“open” or “close”) to Arduino by NodeMCU. |
|  | | 14. Check response in order to open the gate or not by Arduino. |
| **Alternative Flows:**  6.a. If no reservation exists, go to step 13 with “close” response.  7.a. If no reservation exists, go to step 13 with “close” response.  8.a. If the reservation has less than 1 hour left, go to step 12  10.a. if no reservation was found, go to step 12 | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**09** | View Parking | |
| **Brief Description:** | The user views a map with the current status of parking spots and get directions for a specific spot. | |
| **Primary Actors:** | User | |
| **Secondary Actors:** | 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. Request zone data from cloud. |
| 3. Cloud sends data. | | 4. Display a map with the current status of parking spots. |
| 5. User select a parking spot to get direction. | | 6. Redirect to the google map |
| **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:** | User | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | Sensor senses a user’s car entering or leaving the parking spot. | |
| **Preconditions:** | | |
| **Post-conditions:**   * Status of a specific parking spots is updated. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. User enters or leaves a parking spot in a specific zone. | | 2. Sense a car entering or leaving the parking spot in a specific zone by Ultrasonic sensor. |
|  | | 3. Receive data (status and zone name) from sensors by Arduino and send it to the NodeMCU. |
|  | | 4. Update the status of the parking spot in a specific zone to “available” or “unavailable” in the cloud by NodeMCU. |
| **Alternative Flows:** | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**11** | Request Car Care | |
| **Brief Description:** | The VIP user request car care services. | |
| **Primary Actors:** | VIP user | |
| **Secondary Actors:** | Servesni company | |
| **Trigger:** | VIP user asks to request car care service. | |
| **Preconditions:**   * VIP user must be registered to the system. | | |
| **Post-conditions:**   * VIP user is redirected to the Servesni application or website. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
| 1. VIP user asks to request car care service. | | 2. Display services that Servesni company provides. |
| 3. VIP user selects one of the services. | | 4.Redirect user to the Servesni application or website. |
| **Alternative Flows:** | | |

|  |  |  |
| --- | --- | --- |
| **Use case Id:** UC**12** | View Current Occupancy Trend | |
| **Brief Description:** | The user views statistical data that represent occupied percentage in each hour for selected zone for last four weeks. | |
| **Primary Actors:** | User | |
| **Secondary Actors:** | Cloud | |
| **Trigger:** | User select to view current occupancy trend option (Histogram). | |
| **Preconditions:** | | |
| **Post-conditions:**   * Statistical data (Histogram) is displayed to the user. | | |
| ***Normal Scenario*** | | |
| **Actor Action** | | **System Response** |
|  | | 1. Request zone data from cloud. |
| 2. Cloud send data. | | 3. Display the zone list along with showing current occupancy trend option (Histogram). |
| 4. User select show current occupancy trend option. | |  |
| 5. User select a day in the week (Sun, Mon, Tue, …. etc.). | | 6. Find statistical data for the selected day. |
|  | | 7. Calculate for each hour, % Occupancy = total sum of occupancy in selected day for past four weeks / (number of weeks \* total number of spots). |
|  | | 8. Display % Occupancy in each hour in the form of a histogram. |
| **Alternative Flows:** | | |