Car Park SA

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| <27/August/23> | <1.0> | First revision of the Record Entry use case | <Wyatt Coff> |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

1. Record Entry 3

1.1 Brief Description 3

2. Flow of Events 3

2.1 Basic Flow 3

2.2 Alternative Flows 3

2.2.1 If The License Plate Isn’t Recognized 3

2.2.2 If The Car Park Is Full 3

3. Special Requirements 3

3.1 Responsiveness, Reliability, Performance 3

4. Preconditions 3

4.1 System Availability, Entry Point Sensor Activation, Database Accessibility, Space Availability 3

5. Post Conditions 3

5.1 Car Entry Recorded, Availability Status Updated, Message Displayed, Database Consistency, System State, Print Queue 3

6. Extension Points 3

6.1 3

# Record Entry

## Brief Description

This use case describes the process of recording the entry of a car into the CarParkSA parking facility. The car's license plate is recognized, the type of customer (Monthly, Casual Registered, or Unregistered) is determined, and the car is either admitted into the parking area or advised to wait if the facility is full.

# Flow of Events

## Basic Flow

This use case starts when the CarInCarPark actor arrives at the entry point of the car park. The car park's license plate recognition system is activated to identify the car's license plate number.

CarInCarPark: Arrives at the entry point of the car park.

System: Activates the license plate recognition system

System: recognize the license plate of the approaching car

CarInCarPark: Waits for system’s confirmation.

System: Checks the database to see if the license plate number is registered.

CarInCarPark: Continues to wait for system's action.

System: Determines the type of car based on registration—either "RC" (Registered Casual), "RM" (Registered Monthly), or "C" (Casual).

CarInCarPark: Awaits instructions on whether to enter or not.

System: Checks the availability of parking spaces.

CarInCarPark: Waits for the availability status.

System: If spaces are available, displays the message "Availability, please drive in.".

CarInCarPark: Drives into the car park.

System: Records the date and time of the entry.

CarInCarPark: Is now inside the car park.

System: Adds 1 to the total number of cars in the car park.

CarInCarPark: Occupies a parking space.

System: Saves the CarInCarPark record to the database for billing purposes.

CarInCarPark: Parked and inactive until exit.

If the system determines the car type as either "RC" or "RM", the registered details are also retrieved from the database.

## Alternative Flows

### If The License Plate Isn’t Recognized

In the event that the license plate is not recognized or found in the database, the system will default to identifying the car as a Casual type ("C"). This triggers an alternative behavior in the system.

System: Fails to recognize the license plate or does not find the license plate in the database.

CarInCarPark: Awaits further instructions from the system.

System: Automatically sets the car type to "C" (Casual).

CarInCarPark: Continues to wait for the system's action.

System and CarInCarPark: Proceed to Step 5 in the Basic Flow, where the system checks for parking space availability.

After the completion of this alternative flow, the system returns to the main flow starting at Step 5.

### If The Car Park Is Full

If the car park is full, an alternative flow is triggered to handle this exceptional situation.

System: Detects that the car park is full.

CarInCarPark: Awaits further instructions from the system.

System: Displays the message "Full, please wait" on the display board.

CarInCarPark: Waits outside the car park for an available space.

System: Continuously checks for available spaces.

CarInCarPark: Continues to wait.

System: Once a parking space becomes available, displays the message "Availability, please drive in."

CarInCarPark: Proceeds to drive into the car park.

Upon completion of this alternative flow, the system returns to the main flow starting at Step 7, where it records the date and time of entry.

This alternative flow provides concise and clear documentation of how the CarParkSA system behaves under certain exceptional conditions, ensuring that even in such situations, the system can handle them efficiently.

# Special Requirements

In the CarParkSA system, special attention must be given to specific non-functional requirements to ensure the reliability, performance, and usability of the system. These requirements are crucial for maintaining a high level of service and for fulfilling the needs of the users and administrators.

Non-functional Requirements:

1. **Responsiveness:** The system needs to **respond within 3 seconds** to any input data. This is essential to provide real-time information to the drivers and ensure the smooth flow of vehicles in and out of the parking area.
2. **Reliability:** The **Mean Time Between Failures (MTBF) should be no more than once per half-year**. A reliable system is crucial in a car parking environment where any system downtime can cause significant disruptions resulting in revenue loss.

3. **Performance:** The system should be able to **print reports within 3 minutes**. Speedy report generation is necessary for administrative purposes and for keeping track of the parking space availability, usage statistics, and revenue.

These non-functional requirements are vital for the CarParkSA system to meet the expectations for performance, reliability, and responsiveness, and thereby ensuring customer satisfaction.

# Preconditions

1. **System Availability:** The CarParkSA system must be operational and responsive.
2. **Entry Point Sensor Activation:** The sensor at the car park's entry point must be fully operational to detect the arriving cars.
3. **Database Accessibility:** The system must have access to the car registration database to verify and record entry data.
4. **Space Availability:** The total number of cars currently in the car park should be known to the system.

These preconditions ensure that the "Record Entry" use case can proceed smoothly without interruption, thereby aiding the flow of cars into the parking area efficiently and securely.

# Post Conditions

1. **Car Entry Recorded:** The entry details of the car are recorded in the CarInCarPark database table.
2. **Availability Status Updated:** If the car is successfully parked, the total number of cars in the car park is increased by 1.
3. **Message Displayed:** A message is displayed to the car driver indicating either successful entry or, in the case the car park is full, a message to wait.
4. **Database Consistency:** The system database remains in a consistent state, with the newly entered data properly stored.
5. **System State:** The system remains operational and ready for the next entry or another car park operation.
6. **Print Queue:** If any reports are triggered to print (e.g., parking receipts, logging information), they are placed in the print queue.

# Extension Points