ParkSpot V2:
Architectural Design

ParkSpot

Full stack Solutions: Tari Mautsa

Contents

1 Introduction

1.1 Definitions, Acronyms and Abbreviations

- PS- Parking Space
- PS(s)- Parking Space(s).
- UML- Unified Modeling Language.
- MTTF-Mean Time To Failure. This is the difference of time between two consecutive failures
- MTTR -Mean Time To Repair. This is the measure of the maintainability of repairable items in software system.
- MTBF -Mean Time Between Failure. This is the predicted elapsed time between inherent failures of a software system, during normal system operation
- A- Availability. This is the steady state availability that represents the percentage the software is operational.
- Mobile App- Mobile Application
- Car Park/ Parking Lot/Parking Space- A designated area to park vehicles of any type.
- Mobile Device- a portable computing device such as a smartphone or tablet computer.
- Bluetooth- This is a wireless technology standard used for exchanging data between fixed and mobile devices over short distances using short-wavelength .

1.2 Purpose

Quite a number of motorists face the challenge of securing a parking space once they arrive at a parking lot. This results in the motorist wasting time, energy and fuel trying to locate a parking space which may make one late for a commitment, become frustrated or even leave a parking lot and search for another, in the hope to locate a parking space. Some even end up parking in awkward and unsafe places.

Park Spot is the solution we have for all motorists as it is a mobile application that searches for available parking spaces for the motorist when then arrive or enter into the vicinity of a parking lot, saving the motorist time, energy, fuel and ensuring they get to their destination in time and not in a flustered manner.

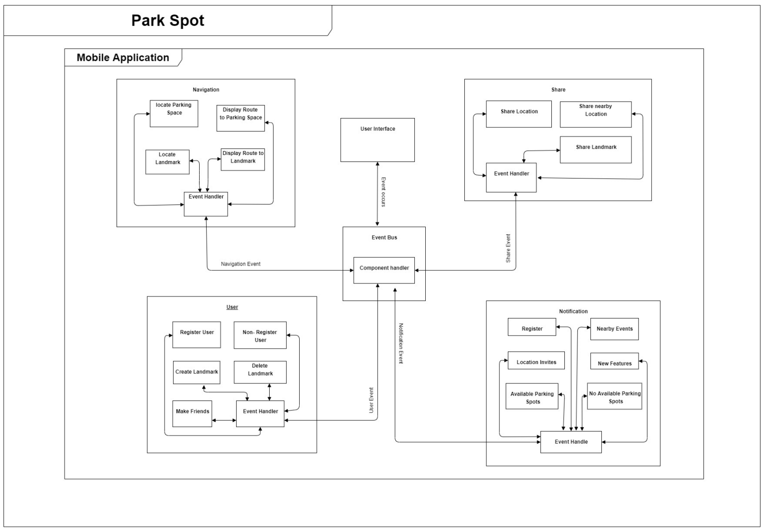
1.3 Project Scope

Park Spot has the following main functions that build up most of the project

- Upon entry of the vehicle into a parking lot, Park Spot scans the entire parking space for available parking .
- Park Spot notifies the motorist via audio or text for any available PS(s). It displays a route the motorist can follow to reach the PS. If there are no available PS(s), Park Spot informs the motorist to try the next parking lot if there is more than one.
- Park Spot helps the motorist locate their vehicle if they forget where they parked it.After parking, the mobile application creates a landmark of where the vehicle would be parked.
- Park Spot will notify user of near by drive-ins for food, petrol stations and car-washes areas.

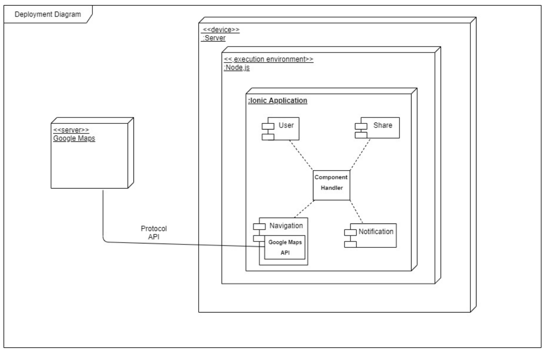
2 Architectural Design

2.1 Event Driven Microservice Architecture



The Event Driven Microservice Architecture is suitable for Park spot as it an architecture that encourages loosely coupled components, which is a benefit in software engineering as no components will be dependant on another. So if one component is to undergo repair, it does not affect the other components. This architecture allows for asynchronous behavior which means data will not only be provided when requested, as long as its been received into the event bus, the components in need can consume the data before its requested. This increases overall system performance as performance is one the quality requirements we are addressing with the architecture. This architecture also allows for versatility in the system, meaning there is greater responsiveness of the system to a user's interaction and it is adaptable to different environments. We also address reliability with this architecture as it is optimized for real time analytics, so we are able to find, analyze and respond to patterns in time before any critical events occur.

2.2 Deployment Diagram



With the

3 Non-Functional Requirements

3.1 Quality Requirements

- Reliability
- Performance
- Usability
- Flexibility
- Extensibility
- Accessibility
- Availability
- Interoperability

Park Spot will focus on reliability and performance as its core quality requirements. .

- Reliability

  The architecture styles for reliability are; Use of Geo-location to display a user's/vehicle's location/position in respect to nearby parking spaces or events as well as use of GPS-Navigation to navigate a user/vehicle to a location of an available parking space or location invite. Reliability is quantified in terms of failure rate (λ)

$\lambda = \dfrac{1}{MTTF}$

$R(t) = e^{-\lambda \cdot t}$

$A = \dfrac{MTTF}{MTTF + MTTR} = \dfrac{MTTF}{MTBF}$

Practical Example

lf MTTF = 1000 hours for Park Spot, then this means Park Spot should work for 1000 hours of continuous operations.

Say MTTR = 2 hours for Park Spot, then the $MTBF = 1000 + 2 = 1002$

Accordingly, $A = 1000/1002 \approx 0.998$

- Performance

  The architecture styles for performance are; making use of multiple servers to distribute the workload of the mobile application and making use of the local cache of a user's device to reduce network activities.

  - Load Test The load test will be to understand the behaviour of Park Spot under a specific expected load. This can be the expected concurrent users on the application performing a specific number of transactions within the set duration. This test will give out the response times of all the important business critical transactions.
  - Stress Test This kind of test determines the Park Spot's robustness in terms of extreme load and helps determine if the system will perform sufficiently if the current load goes well above the expected maximum.
  - BreakPoint Test This is similar to stress testing. An incremental load is applied over time while the system is monitored for predetermined failure conditions. Breakpoint testing is sometimes referred to as Capacity Testing because it can be said to determine the maximum capacity below which the system will perform to its required specifications or Service Level Agreements.
  - Internet Test

3.2 Architectural Constraints

- User can only locate a parking space when they are inside a parking lot.

- Park Spot will not pre-book/reserve a parking space before hand for a user
- If a user isn't in the vicinity of a parking lot, Park Spot will not be able to locate a parking space.
- If there are no available PS(s) in a parking lot, Park Spot will not inform/notify the user in terms of time when the next available PS will be.
- For Park Spot to be installed on a mobile device, a user's mobile device should be a smartphone that has internet and Bluetooth connectivity.

3.3 Technology Requirements

- Ionic Framework(Ionic 4)

  This is a cross platform for developing mobile applications. This technology is suitable for Park Spot as it will allow us to have one created code base that can be deployed on different operating systems such as, IOS, Android and Windows devices. This will ensure fast development as time is a constraint. Ionic 4 will work well with the Geo-location technology and the GPS Navigation as they are simply plugins that are supported and can be added to Ionic projects.

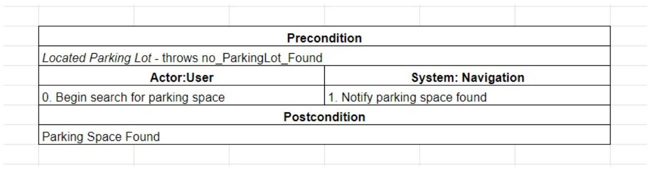
- GPS Navigation

  We will incorporate this technology as it will help us with navigating to correct location once a Parking space has been located or an invitation has been sent by a friend who is a registered user on the application, It is a plugin that can be added to Ionic Projects.

- Geo-location

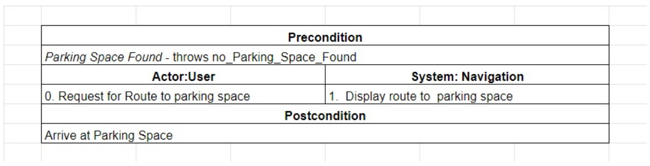
  We will incorporate this technology so that we are able to identify the geographical location of the user's car through the mobile device and from there when the search for the parking space occurs, it can be made relative to where the user/vehicle will be. It is a plugin that can be added to Ionic Projects.

3.4 Actor-System Interaction Model



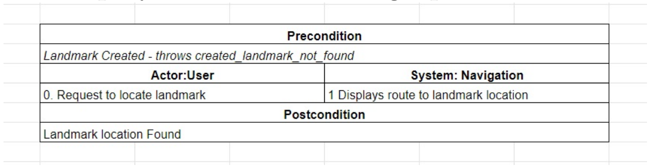
Precondition	
Located Parking Lot - throws no_ParkingLot_Found	
Actor:User	**System: Navigation**
0. Begin search for parking space	1. Notify parking space found
Postcondition	
Parking Space Found	

Locate Parking Space Service



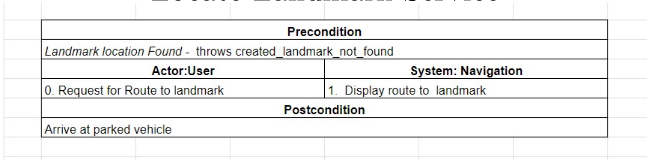
Precondition	
Parking Space Found - throws no_Parking_Space_Found	
Actor:User	**System: Navigation**
0. Request for Route to parking space	1. Display route to parking space
Postcondition	
Arrive at Parking Space	

Display Route to Parking Space Service



Precondition	
Landmark Created - throws created_landmark_not_found	
Actor:User	**System: Navigation**
0. Request to locate landmark	1 Displays route to landmark location
Postcondition	
Landmark location Found	

Locate Landmark Service



Precondition	
Landmark location Found - throws created_landmark_not_found	
Actor:User	**System: Navigation**
0. Request for Route to landmark	1. Display route to landmark
Postcondition	
Arrive at parked vehicle	

Display Route to Landmark Service

4 Functional Requirements

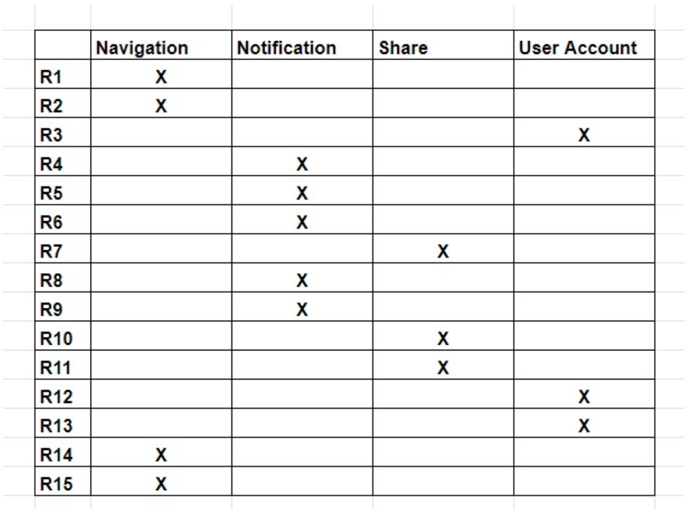
4.1 System Components

- Navigation- This functionality is for locating an available parking space as well as displaying the route to the parking space.
- Notification-This functionality is for notifying the user of available parking spaces, nearby events.
- Share- This functionality will be to share information with other registered users of the mobile application
- User Account- A user has the option to use the mobile application with a registered user account or without. A registered user account has additional features which a non registered User account does not have.

4.2 Requirements

- R1 Locate available parking space.
- R2 Display route to navigate to parking space
- R3 Create a landmark at parking space after car has parked.
- R4 Notify User of available parking spaces
- R5 Notify user of nearby events such as petrol stations, car-washes, drive-ins.
- R6 Notify user when there are no parking spaces.
- R7 Share parking lot location with other registered users
- R8 Notify registered users of new features on the app
- R9 Notify registered user of location invites from other registered users
- R10 Share near by events' location with other registered users
- R11 Share created landmark with other users.
- R12 User can use the app with a registered user account or without a registered user account.

- R13 User can make friends on the application if they are a registered user.
- R14 Locate created Landmark to locate where the car has been parked.
- R15 Display Route to located Landmark.



	Navigation	Notification	Share	User Account
R1	X			
R2	X			
R3				X
R4		X		
R5		X		
R6		X		
R7			X	
R8		X		
R9		X		
R10			X	
R11			X	
R12				X
R13				X
R14	X			
R15	X			

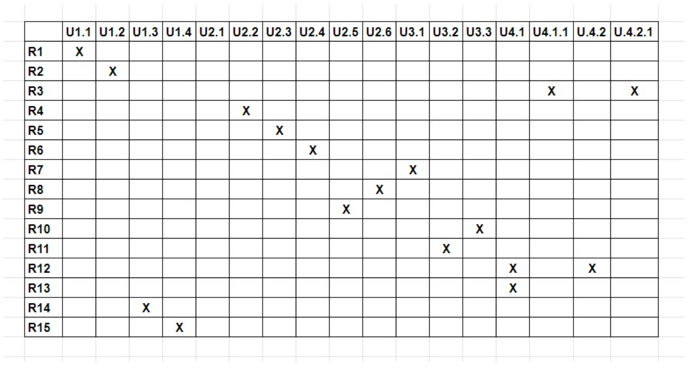
Traceabillity Matrix of Requirements and Use case

5 User Characteristics

5.1 Users

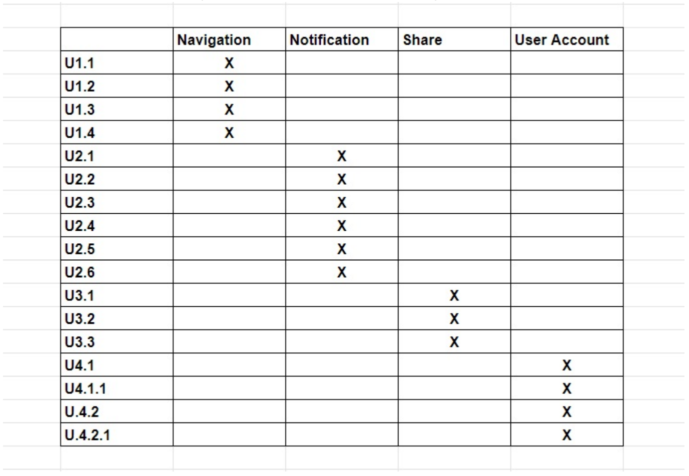
- Motorist
  This type of user should be able to operate and interact with mobile applications on a mobile device. If they are able to interact with applications such as Google maps, Waze, they will be able to interact with Park Spot. The user should also be knowledgeable with Wireless Technologies such as Bluetooth and Wifi and be able to create a user profile. The user will use Park Spot for the purpose of locating a PS in a Parking lot as well as locating their vehicle.

6 Traceability Matrix



	U1.1	U1.2	U1.3	U1.4	U2.1	U2.2	U2.3	U2.4	U2.5	U2.6	U3.1	U3.2	U3.3	U4.1	U4.1.1	U.4.2	U.4.2.1
R1	X																
R2		X															
R3															X		X
R4						X											
R5							X										
R6								X									
R7											X						
R8										X							
R9									X								
R10													X				
R11												X					
R12														X		X	
R13														X			
R14			X														
R15				X													

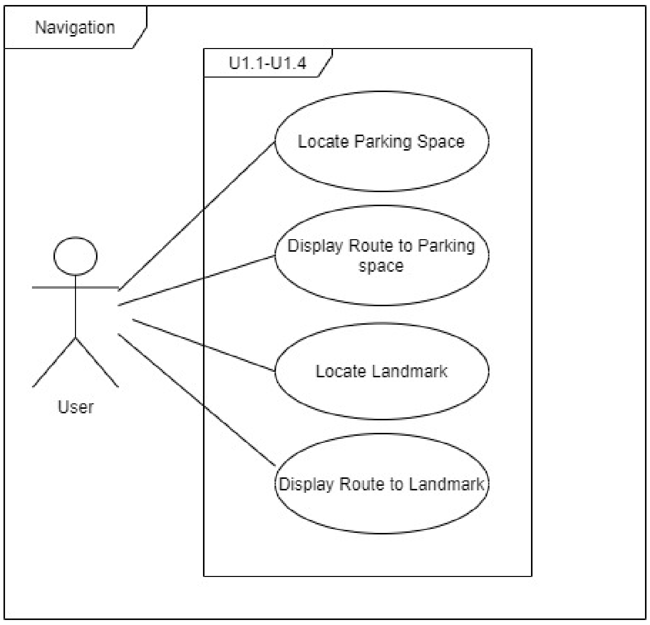
Subsytem Traceabillity Matrix



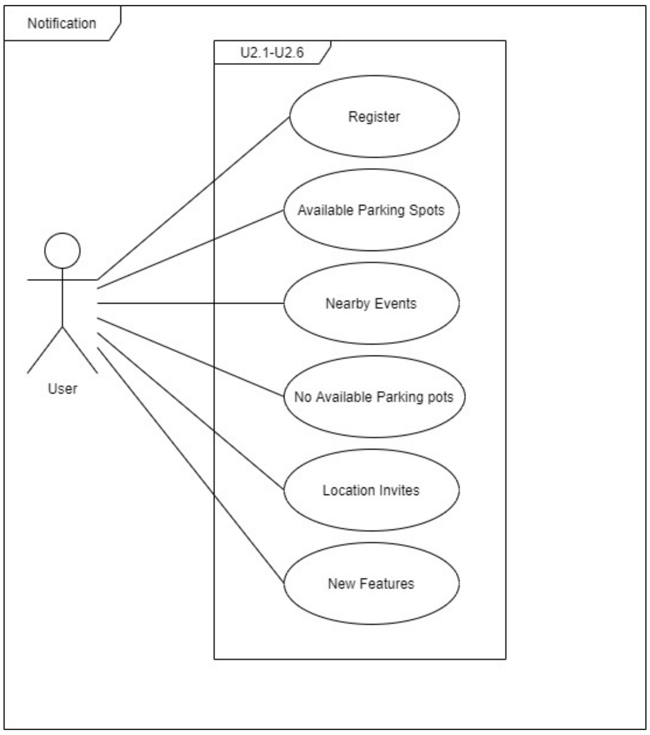
	Navigation	Notification	Share	User Account
U1.1	X			
U1.2	X			
U1.3	X			
U1.4	X			
U2.1		X		
U2.2		X		
U2.3		X		
U2.4		X		
U2.5		X		
U2.6		X		
U3.1			X	
U3.2			X	
U3.3			X	
U4.1				X
U4.1.1				X
U.4.2				X
U.4.2.1				X

System Traceabillity Matrix

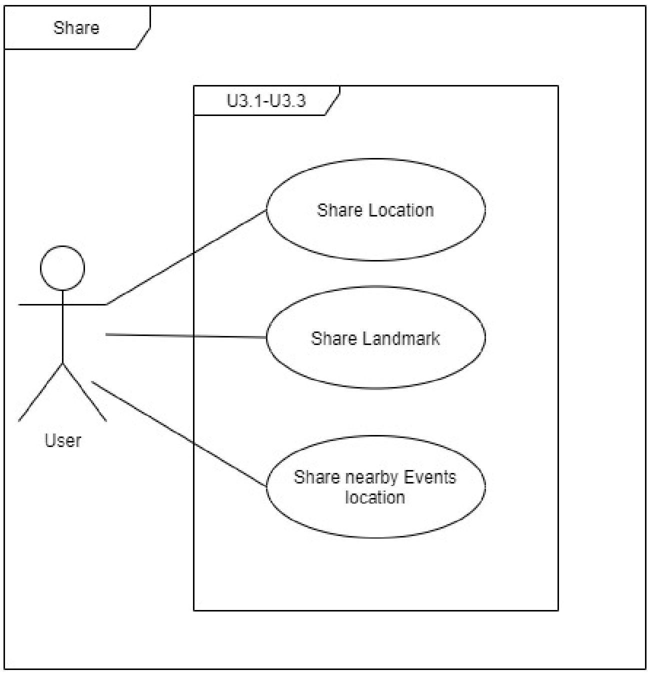
7 Use Case Diagrams



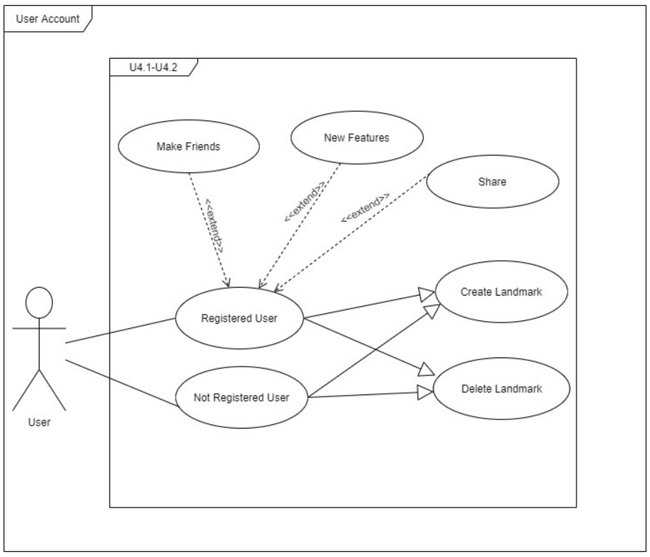
Navigation Use Case Diagram



Notification Use Case Diagram

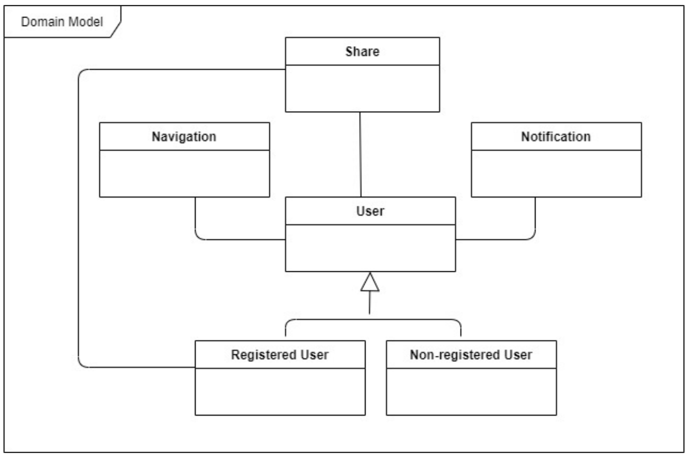


Share Use Case Diagram



User Account Use Case Diagram

8 Domain Model



Domain Model for System