

**REPORT ON**

**PARKING SPOT FINDER IN HOBOKEN**

**SSW – 564 Software Requirements Analysis & Engineering**

**TERM - SPRING 2016**

**GUIDED BY**

**Prof. STACEY GELMAN**

**SUBMITTED BY**

SIDDHESH PALAV

EKKASIT PINYOANUNTAPONG

**ACKNOWLEDGEMENT**

It is indeed a matter of great pleasure & privilege to be able to present this report on “**PARKING SPOT FINDER IN HOBOKEN”** under the valuable guidance of **Prof. Stacey Gelman** (PROFESSOR & SOFTWARE ENGINEERING PROGRAM DIRECTOR.)

I would like to express my deep sense of gratitude to my guide **Prof. Stacey Gelman** for her valuable guidance, advice, & constant aspiration to our work.

Lastly I would like to express my sincere gratefulness to all those people who have helped me directly or indirectly for the completion of this work.

**Note -** Pledge on my honor that I have not given or received any unauthorized assistance on this assignment/examination. I further pledge that I have not copied any material from a book, article, the Internet or any other source except where I have expressly cited the source*.*

Thanking,

SIDDHESH PALAV

EKKASIT PINYOANUNTAPONG

**APPROVAL SHEET**

The project repot entitled “**PARKING SPOT FINDER IN HOBOKEN” sub**mitted by SIDDHESH PALAV AND EKKASIT PINYOANUNTAPONG is approved for partial fulfillment of requirement for the course work SSW – 564 Software **Requirements Analysis & Engineering.**

Prof. STACY GELMAN

PROFESSOR & SOFTWARE ENGINEERING PROGRAM DIRECTOR

Stevens institute of technology

1 Castle Point Terrace, Hoboken,

NJ 07030

**ABSTRACT**

The parking spot finder for Hoboken is an application that shows an available parking spots in Hoboken NJ. This application is not just designed to show the parking spots but provides details about it, which will be useful to user to decide whether to select that particular spot or not.

We are presenting this document as a requirement document through which we are explaining working of application functionality in details.

**INDEX**

|  |  |  |
| --- | --- | --- |
| Main Index | Title | Page No. |
|  | Introduction   * 1. Background   2. Objective   3. Scope | 7  7  7  7 |
|  | Overall description   * 1. Description of Proposed System   2. Operating Environment   3. Design and Implementation Constraints   4. Constraints on system   5. Assumption and Dependencies   6. Modes of Operations   7. Support Environment | 8  8  10  12  13  14  15  16 |
|  | Stakeholders   * 1. User class   2. Additional Stakeholders | 17  17  17 |
|  | Functional and Nonfunctional Requirements   * 1. Core System Requirement      1. Data Collection   2. External Interface requirements      1. Interface      2. Software Interface      3. Hardware Interface      4. Quality Attributes      5. Constraints   4.3 Requirement Prioritization | 18  18  18  18  18  18  18  19  20  21 |
|  | Use case and sequence diagram   * 1. Definition of use cases and its importance   2. Basic assumptions for use cases and use case sequence diagram | 24  24  24 |
|  | Use cases   * 1. Case 1   2. Case 2   3. Case 3   4. Case 4   5. Case 5   6. Case 6   7. Case 7   8. Case 8   9. Case 9   6.10Case 10 | 25  25  27  28  29  30  31  32  34  36  38 |
|  | User Interface   * 1. What is user interface?   7.1.1 general definition   * 1. User interface representation      1. login page      2. Register page      3. Main page      4. List view      5. Filter page      6. Parking spot page      7. Get direction      8. Review page | 40  40  40  41  41  42  42  43  44  45  46  47 |
|  | Appendix – A   * 1. Requirement Elicitation techniques | 48  48 |
|  | Appendix – B | 52 |
|  | Appendix - C | 57 |
|  | Glossary | 58 |

**Diagram Index**

|  |  |  |
| --- | --- | --- |
| Index | Title of the Diagram | Page No. |
|  | Public parking system | 9 |
|  | Private Parking system | 9 |
|  | Application Architecture | 11 |
|  | User class | 17 |
|  | Diagram 5 – Account Registration | 26 |
|  | Diagram 6 – Application Login | 27 |
|  | Diagram 7 – Login as guest | 28 |
|  | Diagram 8 – Parking spots list | 28 |
|  | Diagram 9 – selecting parking spot | 30 |
|  | Diagram 10 – Details of Parking Spot | 31 |
|  | Diagram 11 – Navigation to parking spot | 33 |
|  | Diagram 12 – Customer review | 35 |
|  | Diagram 13 – Application Maintenance | 37 |
|  | Diagram 14 – Adding Parking Spot | 39 |
|  | Diagram 15 – Rough sketch of system | 53 |

**1 - Introduction**

**1.1 Background**

It’s a frustrating situation where driver aimlessly circling the blocks, hoping to stumble across a free parking space, but with no clue as to where such a space might be, which mainly happens because of lack of information, such as all parking spots in particular area, availability of the spot at any given time and finally rules related to the parking spot/place. There is not a particular application available in the market which can provide quantitative and qualitative information about parking spots in Hoboken, NJ. Considering the gravity of the problem, there is a need of an efficient application for finding parking spot.

Assumption - (rules are depends on whether the parking spot is using public road / private space).

**1.2 Objective**

The main objective for this application is that it can show all the parking spots in Hoboken with brief details about that spot, by using of which user can select the parking spot.

Thus this application will eventually saves user’s effort, time, fuel and money.

**1.3 Scope**

Considering the functionality of the application, scope is divided into two parts

**Present scope**

In this scope our focus is on development of an application which can delivering the prime requirement of the user i.e., providing application that can show all the parking spots brief details about that spot such as address, type of that spot ( public or private) and availability of the parking spots.

Assumption

Here user can see the available parking spot just for private parking spots and paid public parking spots. This functionality is not available for public parking spots that are free.

**Future scope**

Successful deployment of the application could have tremendous scope in future as it can be upgradeable in terms of functionality such as, “system should allow user to reserve a selected parking spot” where system can sync with its subsystem or another application which can allow user to select and reserve spot at the same time.

**2 Overall description**

* 1. **Description of the Proposed System**

All available parking spots are divided into two main category

**Public parking spot**

**Description of the public parking spot**

All the parking spots that comes under government control is considered as public parking spot. All public parking spots are divided into two categories mainly free and paid public parking spots. Each paid public parking spot vacancy is provided by parking meter. The free public parking spot vacancy will be shown.

**Private parking spot**

**Description of private parking spot**

All the parking spots owned by a person or company is considered as private parking spot. Private parking spot is equipped with sensors which can keep track of all the incoming vehicle and outgoing vehicle, these sensor sends the data to machine which keeps the count of the number of vehicle in that particular space at any given time.

**General Application working**

* Denuding upon the parking spot status user decided which spot to select.
* Upon selection, application will show the details of that parking spot (parking rules, street cleaning dates and timing, review etc.) with an option to get the direction to the selected spot and option to write a review about it.
* Once user reached to the spot each parking spot (public or private) will notify user that he reached to its destination
* User can access information by Android, iOS application, and website.

C:\Users\shree\Downloads\public.png

**Diagram 1 – public parking system**

***C:\Users\shree\Downloads\private.png***

**Diagram 2 – private parking system**

**2.2 Operating environment**

**Working of the public parking spot**

**Paid public parking spot**

Both public and private parking machine will send the data to our server. The application will show the parking spot vacancy of public parking in real-time. Also the application will show the static information (not real time information) such as available time, fee, and direction to the spot.

(Assumption - We assume that we can deal with government to connect data to public parking machine)

**Free public parking spot**

Public parking spots are not provided with parking meters. Application will show the .Free spots as a static information to user. Application will not show whether free parking spot is available or not.

(Assumption - Number of public parking spots will remain constant)

**Working of the private parking spot**

We can assume that the each private parking place contains multiple parking spots. They are equipped with sensors which record incoming and outgoing vehicles. This sensors connected to the machine which keeps the count of vehicles. Depending on the count, Machine will shows available and occupied spots to user via application interface. When sensor detects the incoming vehicle, it generates the signal and send it to the machine which intern note it and reduce the available parking spot by one and increment the occupied spot by one. Machine process the same functionality in reverse order when vehicle leaves the parking spot. Machine is connect to the application server which get this data and updates application in real time. User can see the relevant changes via application interface.

**C:\Users\shree\Downloads\opertaion.png**

**Diagram 3 – Application Architecture**

**Private parking spot –** All the data is first collected by the machine. Machine then performs counting simple operation on it and sends the data to application server.

**Public parking spot –** Data collected form the parking meter is sent directly to the application server.

**Application server –** Server takes the data from each parking and public spots and deliver it to android, IOS or website. This devices then shows the updated data in real time through application interface. Server will send all the data to the database with its respective date and time.

**Application database**

Application database contains all the data entries sent from the application server. Public parking spots are divided into free and paid parking spots. Since free parking spots has no controlling device like parking meter as in case of paid public parking, database contains static data about it. Server sends this static data all the time to application. Application interface shows all the free public spots all the time without any change.

**Assumptions-** Application developed in such a way that it can work on iOS, android and on website. All the data is encrypted before storing to the database.

**2.3 Design & implementation constraints**

* New system should enumerate all the parking spots in Hoboken, NJ.
* Application should provide brief information (address, type of the spot and no of available spots) about the parking spot just by hovering it.
* System should provide detailed information (address, parking spot type, parking charges (if any), rules& regulations and review) about the parking spot with options to ‘get direction’ to that spot and to write a review.
* If user selects ‘get direction’ then application will pull up the google map and shows the direction to the spot and if user selects write an review then use can write his review about the application service in the comment area and submit it.
* User can see all the spots near him by clicking or touching on my location icon on the application interface.
* System interface must be graphical so that user of any age can use it without any external support System should be available on Android, IOS application, and website.
* If a private parking space owner wants to add his/her parking spot to the list then owner should send a request email with all the relevant information and document about the parking spot to the application administrator. (Process of adding parking spots – After receiving the request email administrator verifies all the information about the parking spot and its owner. After complementing the verification process administrator adds the parking spot to the list, which then displayed on the application interface.)
* System should send a notification about important event such as fire hazard, snow storm availability of the spot or other public emergency. (Notification is default function of this application and user can turn it off by using operating system’s settings)

**2.4 Constraint on the system**

* System should have its own information processing server with the maximum capacity to handle 10000 concurrent user at any time (maximum capacity is set to 10000 considering the future demand and development and to handle sudden increase in load on the server). The server environment can support communication from end user concurrently from all devices, (mobile application and website) at a time.
* System should record all the data of parking spot activity ( occupied or available ) and updates its database so that user can see the real-time data about parking spots,
* User can select at the most 3 spots at time. For an example – Since current version of application is not supporting the parking spot reservation hence it might be possible that user selects the spot and drive towards it but in the meantime someone else occupied the spot. In this case application notifies user and redirect him to next preferred spot. If all selected spots are occupied then application ask user to select the new spot or suggest nearest available spot.

(This functionality is only available for paid public and private parking)

* System should not allow user to use same information to create multiple profile.
* System must be accessible for 24hrs.
* System must be usable for Hoboken area only.
* System should respond to the user action under 2sec.

**2.5 Assumptions and dependencies**

**Assumption**

* User knows how to operate mobile phone and computer.
* User will use application on android, iOS or via website.
* Total shutdown of the system is not possible
* Sensors of private parking spots designed to detect presence of cars only and not any other object. All sensors are well calibrated and reliable i.e. detecting only cars not any other object)
* Parking meter will always available and functioning properly..
* Application will use google map as a third party application to display the route and locations of the parking spots.
* System will always available

**Dependencies**

* Application is depend on the data from private and public parking spot for performing its desired tasks.
* Application database is provided by third party service provider and they take all the responsibility of maintenance and support.
* Accurate working of the application is depends on the accurate user input.

**2.6 Modes of operation**

* **Normal mode** – Application/System functioning normally. User can see all the parking spots and select the parking spot depending upon the availability of the parking spot and get the direction to the parking spot by application function ‘get the direction’
* **No Availability mode –** All the listed parking spots are “occupied”. User will be notified once the parking spot became available (here parking spots include both private and public).
* **Maintenance Mode** – In a maintenance mode a damaged component or components may be replaced or upgradation of an old component take place. Maintenance activity is also applied to for software code, in which certain function are replaced with new functions which can improve performance. Also, some part of code is modified to incorporate new features available to particular platform.

According to the nature of the system maintenance plan, the subsystem‘s ability to provide services and own management may be impacted. (Assumption - maintenance mode will be carried out when there is low load on application (such as night time) so that system will not stay under maintenance mode more than 30 min. During maintenance mode user can see previous search results but cannot perform new search and selection operation Maintenance of hardware of private and public parking spots is totally depends upon the owner.)

* **Software Update Mode -** If new feature launch at the first time, it will be deliver to the user representative first to perform Beta – testing and to get their reviews about an update. Based on the review further activities decided i.e. if there are no bugs or errors found, then this new update will send to all users as a software update else, further work will carried out to remove all the bugs and errors from it.
* **Emergency mode** – Emergency mode will be deployed under special cause (such as natural calamity e.g.:-snow storm, flooding etc.) user cannot access the parking spot. System will send notification to user to remove vehicle from the spot for safety.
  1. **Support Environment**

The support activities for the information and communication systems are performed by supporting staff, utilizing existing equipment. Maintenance staff performs regularly scheduled maintenance. Replacement activities performed on needed basis or according to schedule. Call support is available for those users who couldn’t operate the application. This call support will extended further in future release where user can reserve the spot just by calling. Technical support is provided to solve any technical issues related to the system. All supporting activities are available 24hrs throughout the year to make sure that application will deliver its functionality 24 hrs.

**3 Stakeholders**

**3.1 User Class (Users of the System)**



**Diagram 4 – User class**

* **Customer** – owner of the vehicle looking for parking.
* **Government body** – All the public spots are under control of government. It will take care of all of its parking spot and parking meters.( assumption – No. of all the public parking spots are constant)
* **Private Parking place owner**- Owner of the private parking place will take care of all hardware components and parking spots.

**3.2 Additional Stakeholders (Involved with the system)**

* **Developers** – construct and deliver the system according to the user requirements.
* **Tester** - Test the system to ensure that it is suitable for use.
* **Maintainers** – manage the evolution of the system once it’s deployed.
* **Suppliers** – build / supply the required infrastructure for the system deployment.
* **Support staff** - Provide support to users for the product or system when it is running.
* **System administrator** - Run the system once it has been deployed.

1. **Functional and Nonfunctional requirements**

**4.1 Core System Requirements**

4.1.1 **Data Collection**

Application shall collect the data from parking meter forpublic parking spot andfrom machine from private parking spot (functional)

* 1. **External Interface Requirements**
     1. **Interfaces**
        1. User shall search all the parking spots in the Hoboken area by using filter (Functional)
        2. The application shall provide all the detail information of parking spot for each parking spot such as, available time, fee, and capacity of parking, (Functional)
     2. **Software Interfaces**
        1. Application Web browser should supported minimum browser version which are IE 8, Firefox 4, Safari 9, Opera 35, and iOS Safari 8.4, Android Browser 4.3. (non-functional)
        2. The minimum supported version of Android Operation System should be Android 4.0 (ICE\_CREAM\_SANDWICH)   (non-functional)
        3. The minimum supported version of iOS Operation System should be iOS 6.  (non-functional)
        4. Website shall developed using HTML 5, CSS 5 and JavaScript (functional)
     3. **Hardware Interfaces**
        1. Minimum Requirement
           1. Client Side (non-functional)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Device | Processor | RAM | Disk Space | Screen (pixels) |
| Computer | Intel Pentium III or AMD - 800 MHz | 512 MB | 100 MB | 800 x 600 |
| Mobile Device | 500 MHz | 256 MB |  | 1. 320 |

**4.2.3.1.2 Server Side (non-functional)**

|  |  |  |
| --- | --- | --- |
| Processor | RAM | Disk Space |
| Intel Pentium III or AMD - 800 MHz | 1 GB | * 1. GB |

* + - 1. **Recommended Requirements**
         1. Client Side (non-functional)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Device | Processor | RAM | Disk Space | Screen (pixels) |
| Computer | All Intel or AMD Dual Core- 1 GHZ | 1 GB | 500 MB | 800x600 |
| Mobile Device | 1 GHz | 500 MB | 100 MB | * 1. x 480 |

4.2.3.2.2 Server Side (non-functional)

|  |  |  |
| --- | --- | --- |
| Processor | RAM | Disk Space |
| All Intel or AMD Quad Core- 2 GHZ | 4 GB | * 1. GB |

* + - 1. **Communications Interfaces**
         1. Client (website and mobile application) and server should communicate using HTTP/HTTPS protocol. (non-functional)
    1. **Quality Attributes**
       1. **Usability**
          1. System should be operable 24 hours a day and accessible in real-time. (non-functional)
          2. In case of system down, the system should not be down more than 3 hours.(Non-functional)
          3. Application crash frequency should be less than once a month for user who uses the application every day. (Non-functional)
          4. User shall set the preference up to 3 parking spots. (Functional)
          5. User shall access the website and application as a guest.(functional)
       2. **Performance**
          1. 95% of each Web page generated by the system should be fully downloadable in less than 5 seconds over a 1 MBPS connection. (non-functional)
          2. 95% of finding parking spot should be found in less than 3 seconds after server receives user request. (Non-functional)
          3. The system should support 1000 users concurrent include all access from user device, admin backend, and all private parking machine, all the time. (Non-functional)
       3. **Security**
          1. Username and password is required for every login of both website and mobile application. (functional)
          2. More than three attempts at login and failure should produce a red flag to application administrator, by following of which admin shall locks the application. (Non-functional)
          3. Admin can monitor and edit only private parking spot. (Functional)
          4. Data transfer from mobile application must be encrypted by using MD5 or SHA1 algorithms and the data transferring from website must be operated on Secure Browsing (https). (functional )
       4. **Safety**
          1. Database should be backed up at the end of each month. (non-functional)(Assumption – data available in database is useful for future development of application)
          2. Power supply should have a backup and a disaster recovery plan to keep the Application and its supporting hardware functional. (Non-functional)
    2. **Constraints**
       1. GUI must be in English. (functional)
       2. All HTML code should be conform to the HTML 5.0 and CSS5 standard. (Non-functional)
       3. System should be developed within 6 months after the sign date of the TOR to the launch of beta version (non-function)
  1. **Requirement Prioritization**
* Assumption - All the requirements are classified into three categories high, medium and low priorities. We planned to developed application by applying high level priorities in conjunction with medium level requirements as they are correlated with each other.
* We are planning to implement low level priorities in next iteration of development cycle as those functionalities are out of scope for this iteration.

|  |  |  |
| --- | --- | --- |
| **Index** | **Priority Level** | **Requirements corresponding to the Propriety level** |
|  | High | **Interfaces**   * User shall search all the parking spots with brief details in the Hoboken area by using filter (Functional)   **Usability**   * System should be operable 24 hours a day and accessible in real-time. (non-functional) * User shall set the preference up to 3 parking spots. (Functional) * User shall access the website and application as a guest.(functional)   **Performance**   * 95% of each Web page generated by the system should be fully downloadable in less than 5 seconds over a 1 MBPS connection. (non-functional) * 95% of finding parking spot should be found in less than 3 seconds after server receives user request. (Non-functional)   **Constraints**   * GUI must be in English. (functional) * System should be developed within 6 months after the sign date of the TOR to the launch of beta version (non-function)   **Security**   * Username and password is required for every login of both website and mobile application. (functional) * More than three attempts at login and failure should produce a red flag to application administrator, by following of which admin shall locks the application. (Non-functional) |
|  | Medium | **Interfaces**   * The application shall provide all the detail information of parking spot for each parking spot such as, available time, fee, and capacity of parking, (Functional)   **Software Interfaces**   * Application Web browser should supported minimum browser version which are IE 8, Firefox 4, Safari 9, Opera 35, and iOS Safari 8.4, Android Browser 4.3. (non-functional) * The minimum supported version of Android Operation System should be Android4.0 (ICE\_CREAM\_SANDWICH)   (non-functional) * The minimum supported version of iOS Operation System should be iOS 6.  (non-functional)   **Hardware Interfaces**   * Minimum Requirement   Client Side (non-functional)   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Device | Processor | RAM | Disk Space | Screen (pixels) | | Computer | Intel Pentium III or AMD - 800 MHz | 512 MB | 100 MB | 800 x 600 | | Mobile Device | 500 MHz | 256 MB |  | 1. 320 |  * **Server Side (non-functional)**  |  |  |  | | --- | --- | --- | | Processor | RAM | Disk Space | | Intel Pentium III or AMD - 800 MHz | 1 GB | * 1. GB |  * **Recommended Requirements**   + - * + Client Side (non-functional)  |  |  |  |  |  | | --- | --- | --- | --- | --- | | Device | Processor | RAM | Disk Space | Screen (pixels) | | Computer | All Intel or AMD Dual Core- 1 GHZ | 1 GB | 500 MB | 800x600 | | Mobile Device | 1 GHz | 500 MB | 100 MB | * 1. x 480 |   4.2.3.3.2 Server Side (non-functional)   |  |  |  | | --- | --- | --- | | Processor | RAM | Disk Space | | All Intel or AMD Quad Core- 2 GHZ | 4 GB | * 1. GB |   **Quality Attributes**   * In case of system down, the system should not be down more than 3 hours.(Non-functional) * Application crash frequency should be less than once a month for user who uses the application every day. (Non-functional)   **Performance**   * The system should support 1000 users concurrent include all access from user device, admin backend, and all private parking machine, all the time. (Non-functional) * Power supply should have a backup and a disaster recovery plan to keep the Application and its supporting hardware functional. (Non-functional)   **Communications Interfaces**   * Client (website and mobile application) and server should communicate using HTTP/HTTPS protocol. (non-functional)   **Safety**   * Database should be backed up at the end of each month. (non-functional) (Assumption – data available in database is useful for future development of application)   **Security**   * Admin can monitor and edit only private parking spot. (Functional) * Data transfer from mobile application must be encrypted by using MD5 or SHA1 algorithms and the data transferring from website must be operated on Secure Browsing (https). (functional ) |
|  | Low | * User can reserve the parking spot by using the parking spot application * User can pay for the reserved parking spot using application. (Assumption – parking reservation and payment operation is only applicable for only paid private and public parking ) |

1. **Use cases and sequence diagram** 
   1. **Definition of use cases and its Importance**

In software and systems engineering, a **use case** is a list of actions or event steps, typically defining the interactions between a role (known in the Unified Modeling Language as an *actor*) and a system, to achieve a goal. The actor can be a human, an external system, or time.

In systems engineering, use cases are used at a higher level than within software engineering, often representing missions or stakeholder goals. The detailed requirements may then be captured in the Systems Modeling Language or as contractual statements.

The requirements give us an outline of what we are trying to build. Use Cases are the next step in the design process. Use cases integrate the requirements into a comprehensive package that describes the interaction of the user with the system.

**5.2 Basic assumptions for use cases and use case sequence diagram**

* System online is common pre-condition and special requirement and mentioned in the most of the cases.
* System off-line is common scenario and assumed in most of the alternate flow.
* System login is basic step for user to perform any activity hence it’s a common step for all the basic flow.
* We uses the concept of pressing button to indicate activity of user with system interface.
* System has full access to database and can access or modify the data upon user request. System access to the database is underlying assumption in most of the cases.
* User can access the application (without login) as a guest and it can be done just by clicking skip login on the user interface of the application.

**Assumption for use cases sequence diagram**

* In this document all the diagrams are created to represent basic flow of the use cases hence all the diagrams does not include alternate flow
* All the diagrams has arrow at the beginning of the diagram which indicates the title/activity of the use case.
* Application interface 1 and 2 are created to show the happening of the events followed by the button pressing event. In both cases button pressing event is shown by the arrow before of that column.
* System login is most common event for application interface 1 where as there are multiple event for application interface 2.
* In all the diagram output is indicated by dotted line as it was the only line available in the tool that we used Draw.io.

1. **Use Cases** 
   1. **Case 1**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Account Registration |
|  | Brief Description | User creates a new account to access the system |
|  | Actor | User |
|  | Basic flow | * User wants to use the application * User press the register button on system interface. * User creates his profile. * User sets username and password. * System accepts the user information and request for new account. * System creates new user account. * User gets confirmation notification. |
|  | AlterNet flow | * System is offline. * User information doesn’t matches the system standards (i.e. standards regarding the user details) and system ask user to fill the information again. * User already has account and system denies user to create another account for under same user name. * User finds another application/system better than ours and decides not to continue the registration process |
|  | Pre-conditions | * System must be online. * User wants to use the system and ready to register for new account. |
|  | Post-conditions | User enters the information and register himself. |
|  | Special Requirements | * User must enter authentic and new info to register account. * User must not use the same information to create another account if he already created account. |

C:\Users\shree\Downloads\usercase1.png

**Diagram 5 – Account Registration**

**6.2 Case 2**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | System login |
|  | Brief Description | User enters the username and password to login to the system |
|  | Actor | User |
|  | Basic flow | * User opens the application interface. * User press the login button from system interface. * User enters the username and pin. * System checks the entered information via its subsystem called ‘Authorization’ * System authorized the user to enter into the system. |
|  | AlterNet flow | * System is offline. * User entered wrong information and access is denied by the system. * User terminate the process abruptly. * User has no account and system ask him to register before login |
|  | Pre-conditions | * System must be operational / online. * User must have registered account. |
|  | Post-conditions | User enters into the system. |
|  | Special Requirements | User must have registered account. |

**C:\Users\shree\Downloads\usercase2 .png**

**Diagram 6 – Application Login**

**6.3 Case 3**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Guest login |
|  | Brief Description | User access the application without creating account or login into application. |
|  | Actor | User |
|  | Basic flow | * User opens the application interface. * User press the button ‘ use application as a guest’ * System authorized the user to enter into the application. * User access the application functionality |
|  | AlterNet flow | * System is offline. * User terminate the process abruptly. * User press the different button on application interface. |
|  | Pre-conditions | * System must be operational / online. * User must have registered account. |
|  | Post-conditions | User enters into the system. |
|  | Special Requirements | User must have application installed on his device |

**C:\Users\shree\Downloads\usercase2  (1).png**

**Diagram 7 – Login as guest**

**6.4 Case – 4**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Viewing Parking spots. |
|  | Brief Description | User uses the application to take a quick look at all the parking spot. |
|  | Actor | User |
|  | Basic flow | * User enters into the application by logging in or user enters the application as a guest. * Once entered application shows the front page of the application * User can see all the parking spot in that particular area i.e. Hoboken * User can see the brief details of the parking spot just by hovering over it.   Such as address of the spot, type of the parking and no of available spot. |
|  | AlterNet flow | * User couldn’t login to system. * User cancels the process abruptly. * User couldn’t see all the parking spot because of technical issues. * Data connectivity is not available. |
|  | Pre-conditions | * Application must be online. * User wants see all the parking spots in Hoboken area. |
|  | Post-conditions | User can see all the parking spots in Hoboken. |
|  | Special Requirements | User can see all the parking spots in Hoboken. |

C:\Users\shree\Downloads\usercase 3 .png

**Diagram 8 – Parking spots list**

**6.5 Case 5**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Parking spot selection |
|  | Brief Description | 1. User selects the particular parking spot according to the availability of the parking. |
|  | Actor | User |
|  | Basic flow | * User login to the applications or User access the application as a guest. * User can see the brief details of the parking spot just by hovering over it. Such as address of the spot, type of the parking and no of available spot. * User decides which parking spot he/she wants and selects that particular spot by double tapping on it |
|  | AlterNet flow | * Parking spot is not visible due to technical fault. * Application is offline. * User terminate the process abruptly. * User can see the parking spot but couldn’t select it due to technical malfunction |
|  | Pre-conditions | * User wants to select parking spot. * All the parking spots in the Hoboken are listed properly in the application database and they are accessible upon user request. |
|  | Post-conditions | User selects the spot in particular area according to his/her requirement from the parking spot shown in application interface. |
|  | Special Requirements | * Parking spots must be available in the area of user interest. * Park selection functionality should be available and accessible all time. |

C:\Users\shree\Downloads\usercase4.png

**Diagram 9 – selecting parking spot**

**6.6 Case 6**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Details of parking spot. |
|  | Brief Description | User want to check the information /details about parking spot |
|  | Actor | User |
|  | Basic flow | * User login to the system or access the applications as a guest * User confirms which spot to select and selects that parking spot. * System process the user request and shows the details about the selected parking spot such as address of the spot, price, type of the spot, details of the spot ( rules and regulation) and reviews. |
|  | AlterNet flow | * System is offline. * System unable to process the user request. * Details about the parking spot is not available. * User interrupts the process flow by sudden cancelation of request. |
|  | Pre-conditions | * Details about listed parking spots must be stored in system database. |
|  | Post-conditions | System process user request successfully and user sees the details about selected parking spot. |
|  | Special Requirements | System database must have all the details about the listed parking spots |

C:\Users\shree\Downloads\usercase6.png

**Diagram 10 – Details of Parking Spot**

**6.7 Case 7**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Navigation |
|  | Brief Description | User wants direction to go to the spot from his current location |
|  | Actor | User |
|  | Basic flow | * User login to the system or access the application as a guest. * Selects the desired parking spot. * Request application to show the way to the spot from his current location. * Application shows the direction using google maps. |
|  | AlterNet flow | * Data connection is not available * Google maps are not accessible through parking spot finder application. * User cancel the process. |
|  | Pre-conditions | * Application uses goggle maps as its functionality to show the direction. * Navigation system must be accurate. |
|  | Post-conditions | * User selects the parking spot * Application shows the best way to go to the desired spot using navigation system /application. * User reached to its destination. |
|  | Special Requirements | Application must be capable to use the third party navigation application like google map as its functionality |

C:\Users\shree\Downloads\usercase 7.png

**Diagram 11 – Navigation to parking spot**

**6.8 Case 8**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Customer Review |
|  | Brief Description | User post reviews about application after using it. |
|  | Actor | User |
|  | Basic flow | * User uses the application to find parking spot. * User evaluate his/her experience after using the system. * User decides to write a review and writes the review. * User posts the review. * Application database updated and shows the user post into the application. |
|  | AlterNet flow | * User couldn’t use the system as system is not accessible * User use the system but decided not write review about it. * System unable to accept the user review. * System fail to update the system database after accepting the review from user. |
|  | Pre-conditions | * Application should be able to take user review and update its database. * User is ready to write review after using the application. |
|  | Post-conditions | * User write the review. * Application accepts the user review and updates its database successfully. |
|  | Special Requirements | * Users wants to write a review. * Applications acceptance of the user review and updating the database are essential activities. |

**C:\Users\shree\Downloads\usercase9.png**

**Diagram 12 – Customer review**

**6.9 Case 9**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Maintenance Mode |
|  | Brief Description | Maintenance personal performs the maintenance activity. |
|  | Actor | Maintenance personal |
|  | Basic flow | * User uses the application or uses the application as guest. * User uses the application and writes the review. * Maintenances personal converts reviews into requirements. * Converts the requirements into functionality. * Add this functionality into application and Test it. * Set the schedule for maintenance to upgrade the application functionality. * Notify users for upcoming maintenance activity. * Change the mode of the system to maintenance. * Updates the system. * Change the mode of system from maintenance to normal and notifies the user. |
|  | AlterNet flow | * No need of maintenance activity as system functioning flawlessly. * Personal ignores the user complaints and reviews. * Personal perform maintenance activity in real time without putting application on maintenance mode. |
|  | Pre-conditions | Customers must have suggestion, complaints or reviews worth paying attention to. |
|  | Post-conditions | * User request, complaints are review converted into requirements and implemented it to the application as a software update. * Application updated successfully |
|  | Special Requirements | * Customer reviews, complaints and request must have potential to perform the maintenance activity for system upgrades. * Scheduled maintenance activity needed to keep systems smooth working. |

C:\Users\shree\Downloads\usercase8.png

**Diagram 13 – Application Maintenance**

**6.10 Case 10**

|  |  |  |
| --- | --- | --- |
| **Index** | **Title** | **Description** |
|  | Name | Adding parking spot ( Feature only available for private parking spot) |
|  | Brief Description | Owner of the private parking spot adds new parking spot. |
|  | Actor | User |
|  | Basic flow | * User decided to add his/her parking space to the application. * User sends and email request to add a parking spot with its relevant details. * Application admin accept the information of the new spot * Admin accepts the request and adds the information to the application database. * Application updates its database and adds the new parking spot. |
|  | AlterNet flow | * User don’t follow the rules of adding the new parking spot and hence admin refuses to add the parking spot to the existing list. * Admin accepts the information from user but fail to updates its database and list of parking spots. * User change his mind and decides not to add his spot to the existing parking spot list. |
|  | Pre-conditions | * New parking spot is available and its owner wants to add it into system list. * User have valid account ( guest can’t add the parking spot) * System is ready to accept and update the new information. |
|  | Post-conditions | * User successfully submit the information to system. * System successfully adds it to its database. |
|  | Special Requirements | Availability of the new parking spot. |

C:\Users\shree\Downloads\usercase9 (1).png

**Diagram 14 – Adding Parking Spot**

**7 User interface**

**Assumptions**

* We are demonstrating user interface for the mobile application with the control flow using separate diagram for better understanding of the actual process.
* We are using tool https://www.draw.io/ and adobe flash instead of power point as it has all the necessary shapes to draw required renderings fast and easy.
* Outer rectangle box is representing the display screen of mobile device and each line on the interface starting from diagram no 2 indicates the touch button which upon receiving touch from user triggers a corresponding event to perform specific predefined task.
* Each interface rendering has flow control diagram below it.
* Solid line connecting two blocks of figure indicating user interaction mainly touch and dotted line indicates internal system functionality which do not require user interaction.
* Though we can develop similar interface for web application but for time being we can focus on mobile application
* Buttons on the interface indicates the functionality which user can select.

**7.1 What is User Interface?**

**7.1.1 General definition**

A mobile user interface (mobile UI) is the graphical and usually touch sensitive display

on .Mobile device, such as a smartphone or tablet that allows the user to interact with the

devices. Apps, features, content and functions’ thus, considering above definition we can

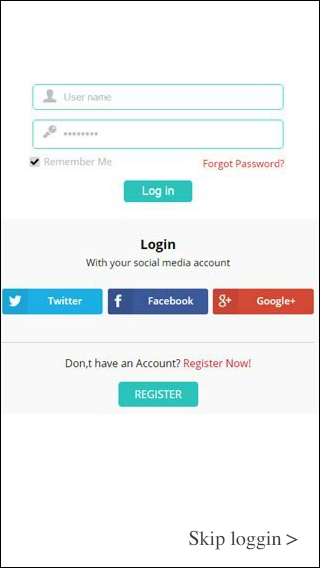
say that. Application interface is the display of the functionality of application presented

in graphical form. Which user can access by physical touch to perform specific task.

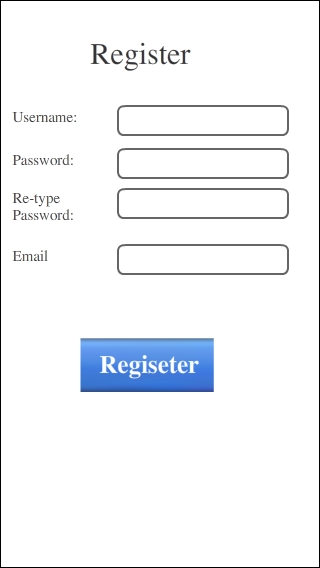
* 1. **User Interface Representation**

**7.2.1 Login page**

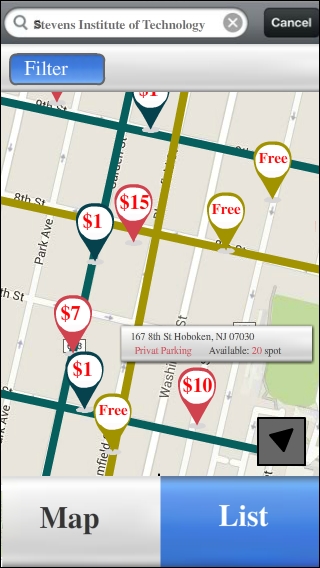
* In the first time user download application or access to the website in the first time or did not log in before the login page will show.
* Use can log in by user name and password which is signed up before or login by social login.
* In case that the user does not have an account. User can click register to sign up.
* If user does not want to register. User can click skip log in to login without register.



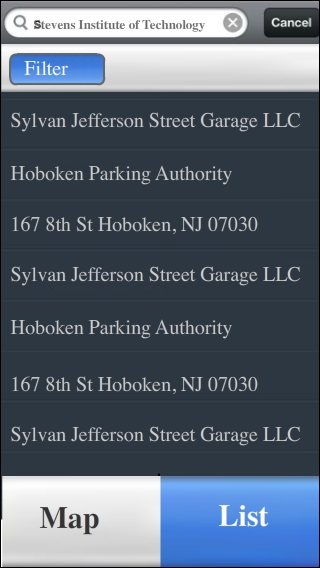
* + 1. **Register Page**
* User can register in this page by create user name, password, and email. And re-type the password again to confirm the password.



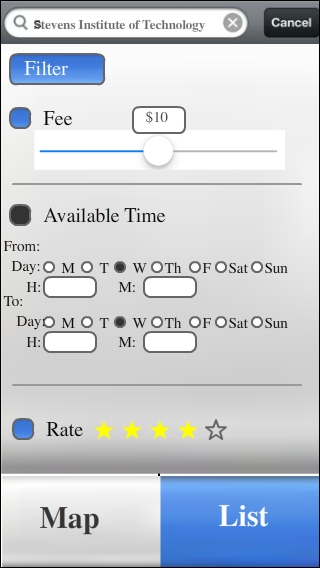
* 1. **3 Main page**
  + In main page, the application use Split View to access to parking spot by two type page which are
    1. **Map View** - the map view shows the parking spots by three type
       - * **Free public street parking (Yellow Pin)** in this type the pin will show the streets that are free parking. The yellow line will draw on the streets which are free parking.
         * **Paid Public Street parking (Green Pin)** green pins and green lines will draw on the street parking which are paid public street parking. And show the fee for parking on the green pin.
         * **Paid private parking (Red Pin)** red pin will locate on the private parking spot and show the price on the pin.



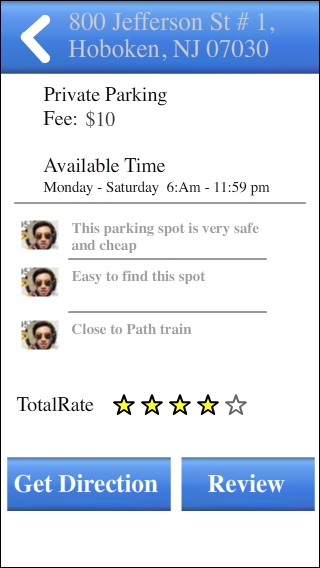
* + 1. **List view -**  the parking spot view will be shown in list type



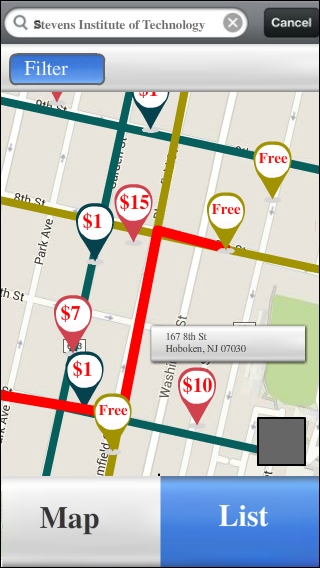
* Use can switch map view and list view by click the button on the bottom. Both view can search specific spot by search function and filter by click filter button.
  + 1. **Filter page** from both map view and filter view user can click filter to filter specific spot which the user wants. User can choose the filter type by fee, available time, and rate.



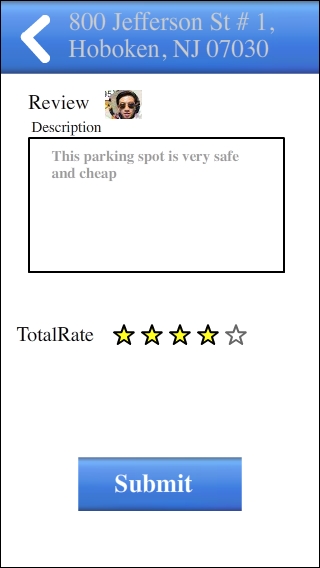
* + 1. **Parking spot page -**  from the map view or list view page, when user clicks the parking spot the application view navigate to this page which can show type of parking spot, fee, description (such as available time, 4 hours parking, resident parking ,etc..), comment from other users, and rate. From this page user can click “get direction” to display route to this parking spot or click review to write the comment to this spot.



* + 1. **Get direction page** - from parking spot page user can click “get direction” to navigate to the spot, the application view shows this the route in this page. Other parking spots still show, so in case of the current parking spot is full, the user can find other parking spot quickly.



* + 1. **Review page -** from parking spot page user can click “Review” to write the comment in the parking spot. The application will show navigate to this page. User can write comment and rate the spot. After finish, click submit to send the comment and rating to server.



**8 Appendix – A**

**8.1 Requirement Elicitation techniques**

Numerous elicitation techniques can be employed on software projects. In fact, no project team should expect to use only one elicitation technique. Before we jump into the actual process we must have to take a look at the differences between requirement gathering and elicitation

Requirement gathering

* More like collecting what you get from user class
* More reactive and less proactive

Requirement Elicitation

* Totally planned collection of the requirements from user class
* Less reactive and more proactive

We can perform this activity by using one or multiple techniques according to the situation at that time.

Elicitation techniques used are as follows

1. **Brainstorming**

It is the procedure of producing numerous ideas regarding particular multiple agendas and at the end of it important ideas are collected and documented by using standard criterion (according to organization policies).Brainstorming activity is required to find out the best way or method to get requirement i.e. mostly hidden requirement. Brainstorming works best when RE (Requirement Engineer) and domain experts (DE) meet face to face. Even though both are geographically distant from one another, using modern techniques allow both to communicate using pc/laptop using webcam. For an ex – skype conference call, TeamViewer.

1. **Domain Analysis**

A domain is a representative group of users who convene in a facilitated elicitation activity to generate input and ideas on a product’s functional requirements. Domain analysis sessions must be interactive, allowing all users a chance to voice their thoughts. Domain analysis is useful for exploring users’ attitudes, impressions, preferences, and needs. Domain must include RE and DE and must keep them on topic, but without influencing the opinions being expressed. Considering the importance and details of the activity we can say that, the technique cannot be used at all without being geographically co—located.

1. **Ethnographic Observation**

This technique is related to Study a stakeholder's activity in their work environment. It is very important for both RE and DE should present at the work environment to conduct deep analysis of work environment, which will eventually help them to get quality requirements. If it is not possible because of geographically distant from one another, then they can share a video surveillance footage of the stakeholders working at their work environment and share their details with one another via internet.

1. **Interviews**

Successful interview can provide most important details about user requirement. RE needs DE to conduct this process flawlessly but if they are geographically distant from one another then this technique cannot work. RE and DE can use internet tools to conduct interview session with user but it might not be possible for every user/user class, which may result into confusion, time and money wastage and moreover it would be difficult for both RE and DE to keep tract of the users and elicited requirements. Because interview activity works best in physical way (face to face) hence, the technique cannot be used at all without being geographically co—located.

1. **Prototyping**

It’s a technique of providing working model/visualization of system to user to evaluate whether given prototype is compatible with their requirement or not. It’s a good way to test the user requirements also user can notify if any requirement/function is missing. Prototype deployment and its related queries could be handled either by RE or DE or by both and geographic differences should not be a problem.

1. **Questionnaires**

Questionnaires are a way to survey large groups of users to understand their needs. They are inexpensive, making them a logical choice for eliciting information from large user populations, and they can be administered easily across geographical boundaries hence it could be performed when the requirements engineer and the domain experts are geographically distant from one another. The analyzed results of questionnaires can be used as an input to other elicitation techniques.

1. **Scenarios**

Scenario-oriented solutions are among the most successful attempts to produce quality requirements. In such approaches, the initial development of a system is based on the knowledge about certain interactions among the user classes involved in it. From those known interactions, the RE and DE can create a set of scenarios that constitute the starting point for eliciting the requirements of the system. Since physical presence of the RE and DE is necessary, in this case, the technique cannot be used at all without being geographically co-located.

1. **Task Analysis**

Task analysis is basically observing end users in their operational environment. Which is similar to the ethnographic observation discuss above. Hence, we can say that, it could be performed when the requirements engineer and the domain experts are geographically distant from one another.

1. **Viewpoints**

A specific technique-viewpoint resolution-is proposed as a means of providing early validation of the requirements for a complex system. The technique is based on the fact that software requirements can and should be elicited from different viewpoints, and that examination of the differences resulting from them can be used as a way of assisting in the early validation of requirements. Since presence of the RE and DE is required for this technique to work out hence the technique cannot be used at all without being geographically co--located .This analysis of views is capable of differentiating between missing information and conflicting information, thus providing support for viewpoint resolution.

1. **Workshops**

Workshops encourage stakeholder collaboration in defining requirements. Workshops are arranged in sessions with multiple stakeholders, from users to developers and testers. They are used to elicit requirements from multiple stakeholders. Working in a group is more effective for resolving disagreements than is talking to people individually. Also, workshops are helpful when quick elicitation turnaround is needed because of schedule constraints. It is an extremely challenging job for RE and DE, also workshop can be arranged using internet online sessions but its effectiveness may vary, hence the technique cannot be used at all without being geographically co-located.

1. **Document analysis**

In this technique existing documentations are studied to understand user relevant details. The technique is Helpful when subject matter experts are not available or no longer with the organization, hence RE and DE can work independently and to communicate with each other via internet tools such as email. Hence, this technique is achievable regardless of the geographical distance between RE and DE.

**9 Appendix – B**

In our project i.e. “Developing application for parking spot finder in Hoboken”, we used interview technique to elicit the requirements from user.

**8.1 Interview Questions and Answers**

**Both Functional and Nonfunctional**

* 1. **What is the background/need for the new system?**

Finding parking spot in the city like Hoboken could be hard and challenging. You just have to travel road to road to find the perfect spot, which not only waste time but also wastes money and fuel. To avoid this we want a new system.

* 1. **Would you like to pay for the application that will be developed?**

Ans: - I don’t want to pay anything for the system, but he is willing to pay the charges related to parking spot (if application expands its functionality and allow user to reserve and pay for the particular parking spot in near future.)

1. **Have you prioritized your requirements?**

Yes, I have prioritized my requirements in 3 categories.

**Must have**

System must show me all the available parking spot in Hoboken at particular time.

System should be free of cost

Must be easy to use.

**Should have -** System must show me the direction to parking spot from the current location.

**Could have –** Can allow me to reserve and pay for the parking spot from application.

1. **What is your measurer to define success of the system?**

I could call the application as a success if and only if it can complete my first and important requirements i.e. showing me available parking spot.

1. **Would you like to involve in development activities? If yes, then How? And If no, why?**

As far as the involvement concerns, yes I would like to participate in the development activities. I would like to participate in the activities where I can provide my feedback without wasting my time or I don’t have to allocate separate time for this activities. For an example – we can use online surveyed.

**Functional**

1. **You mentioned that system shall display all the available spots. Can you please specify the term ‘available’ and ‘all’ in terms of system functionality? (functional)**

‘Available’ – whether particular spot is occupied or not and it is for what type of

vehicle. ‘All’ – application must enumerate public and private spots.

1. **Is it possible for you to provide rough draft of schematic or sketch that can help us to understand what kind of system that you have in your mind?**

Yes

hw 5 stacey 

**Diagram 14 – Rough sketch of system**

Here parking spots should be listed with the specific code which will allow user/us to find the spot easily once we reached there (e.g. – finding parking spot in huge parking lot)

1. **You demanded that system shall notify user, can you elaborate term ‘notify’ and how would you like to receive them?(functional)**

Here notification means ‘alert’. System should notify me if the system is under maintenance or if no vacant spots are available for parking etc.

I would like to receive them on my phone in the form of text and email.

1. **Can you describe, what is the easy to use application means?**

Yes, easy means I don’t have to search or to look for another system to explain me how to use particular function of the application. I need application which I can operate easily (or under 5 min with the help of the tutorials, which can make me familiar to system and its functionality)

1. **Do you want this application to use other applications( for which you have account and access daily ) like, Yelp, Uber, Food panda, Google Map.**

I want this application to be simple, adding such functionality could make it more complex to use, hence I don’t want to see it on my application except application like google map which can provide data which can improve the functionality of the application.

**Non-functional**

1. **What is your objective regarding the new system?**

The main objective is that the system should find a parking spot within 10sec after sending request and it must show all the related information to that park.

1. **Are you considering that system should be upgradable for future**

**Requirements?**

Yes, system should allow me to reserve the spot for the required amount of time (where I can pay for spot through the application and can pay only for the used time)

1. **What kind of support system/activates are you expecting for new system?**

I would like a following support activates from system Maintained system to ensure well running of the system. Call support to provide assistance on the spot.

Efficient notification system to alert user under specific instance like notification under maintenance

1. **What kind of documentation you would like to see to check the progress of the system?**

A document with graphic and numerical data in standard presentation format would be necessary to evaluate the progress. Standard should be maintain throughout the document so that any user can read and understand what is going on.

1. **What is the measure of reliability according to you?**

System should perform all the task requested by the user under 10 sec.

Frequency of the system crash should be very less like once in a week etc.

System should pinpoint the spot location accurately i.e. within 1 meter radius from target.

System should handle specific user request per day without crashing

1. **What kind of safety and security are you expecting from system? ( functional /nonfunctional)**

System should encrypt data while communicating with server.

System should not share user data with third party.

System should not allow user to see other user profile without permission.

System should not update user’s personal information without user permission

1. **Are you looking for system to be used during specific time of the day? ( nonfunctional)**

I want system to remain functional 24 hrs. So that I can use it whenever I want.

1. **You mentioned that system interface should be available in different language, in that sense what kind of functionality you want from system in terms of language? ( nonfunctional)**

System should be available in language according to its locality and geographical feature. Since this system is limited to Hoboken only where majority (almost all people) speaks, read and write English language, hence system should be available in English language.

1. **In case of system failure, what kind of performance are you expecting from system? ( nonfunctional)**

I would like to get notification from system regarding its failure status so that use can search alternate way to complete his intended task.

1. **What other kind of functionality you are expecting from system other than finding parking spot (main goal)? ( nonfunctional)**

System can allow user to reserve parking spot for required amount of time i.e. in terms of minutes and hrs.

System can show nearby attraction and useful information near required spot such as medical facilities, restaurants etc.

1. **What should be the expected time period from development to deployment of the system?(nonfunctional )**

It be hard to say at this time but we would like to system up and running within 1 year from now.

1. **Is there any specification you want for system?**

It should be available on IOS, android and windows platform

Should be function with 1 gb or ram and single core processor.

Applications web interface should be accessible with chrome, Firefox, IE and safari.

**10 Appendix – C**

**References and Tools Used to create Document**

* Software requirements Karl wiegers
* Internet Resources – Wikipedia, Google Search
* 1st page image source – Google Images
* Drawing tool –Draw.io (it’s an online tool specially developed for UML diagrams)

**11Glossary**

* **OS** - Operating System
* **Objective** – description of what the end product look like
* **Scope** – Features delivered to user in the form of application
* **Concurrent User** – Number of user in each specific time
* **GUI** – Graphic User Interface
* **TOR** - Term of Reference
* **GPS** – Global Positioning System