****

**EXECUTIVE SUMMARY**

UT Dallas recently implemented its Comet Cab shuttle system. Theoretically, these shuttles will allow students to traverse the campus quickly by giving them a ride from for example, the residence halls or remote apartments, to central locations on campus. Practically speaking, the shuttles are plagued by problems of ignorance and incompetence. There have been claims of a shuttle driver taking the cab home and only coming back at the end of his shift. Dr. Wenkstern herself has shown that routes and pickups are inconsistent at best, and getting a ride to make your movement across campus faster is a matter of sheer luck in being able to successfully flag down the shuttle you want.

The solution to this obvious problem is CometRide, a three-part application designed for shuttle riders, shuttle drivers and shuttle administrators. For the riders, CometRide is a mobile application that will keep them updated on the locations and statuses of shuttles as they traverse their routes, which partially solves the efficiency problem even if luck in terms of the shuttle's location compared to a prospective rider's location will always play a role. For the shuttle drivers, CometRide is a mechanism to broadcast their shuttle's current location and status. For the shuttle admins, CometRide is a way to keep track of statistics related to the shuttles, as well as potentially set special/temporary routes. For everybody, CometRide will promote use of the Comet Cabs by making them much more accessible to riders.

The technical of our solution rely on ubiquitous GPS and mobile data plans. Our idea is to equip all shuttles with basic Android-powered mobile phones that run our application to provide GPS capabilities for location tracking and a mobile data connection for status updating. GPS is a standard feature and mobile data plans can be had for $20/month if not cheaper. Likewise, we're assuming that anybody who uses CometRide to find a cab has a GPS and Mobile Data in their personal devices. However, our solution falls into place easily with just those common technologies.

Compared to the Transportation Department’s idea of equipping the shuttles with GPS trackers, our solution has the advantage of being much, much cheaper. For a fraction of the cost (Compared to the department’s proposed solution), we can provide a platform that is easier to use and easier to maintain/extend than what the department wants. Our solution makes the department’s proposed GPS tracker solution outdated and inferior, since it has all of the upside, the greatly reduced cost and no drawbacks compared to the GPS-based system.

Danny Matthew

Scott Crain

Trevor Holder

Vaibhav Prakash



****

**THE UNIVERSITY OF TEXAS AT DALLAS**

**PROJECT PLAN**

**1. Overview**

***1.1 Purpose, Scope and Objectives***

**Purpose**

To develop a software application (CometRide) that would enhance the efficiency & location tracking of the shuttle system at The University of Texas at Dallas.

**Scope**

This application will be used by the citizens of UTD and can be extended to anyone who uses the on campus shuttle system.

**Objectives**

1. Develop a mobile application which can be used by students, faculty and other member’s affiliated to The University of Texas at Dallas to request shuttle rides in campus.
2. CometRide will be used to give the driver of the shuttle real time updates of members who wish to request a ride in the campus.
3. Citizens of UTD will be able to hitch a ride at any location (pre-defined shuttle routes) and see the location of the shuttle in real time.
4. This application is intended to work with existing smartphone of students which are equipped with GPS, have access to Network Data(3G/4G LTE) and running Android(4.1+).
5. Each shuttle will have its own dedicated smart phone/tablet running CometRide.

***1.2 Assumptions and Constraints***

**Assumptions**

1. Majority of the students and faculty have smartphones.
2. Customers are familiar with using a smartphone application.
3. Network data and GPS availability is 100% throughout The University of Texas at Dallas.
4. The driver of the shuttle is familiar with interacting with a smartphone application.

**Constraints**

1. Customers using this application must have a smartphone equipped with GPS, have access to Network Data (3G/4G LTE) and running Android (4.1+).
2. Each shuttle will have its own dedicated smart phone/tablet running CometRide also equipped with GPS, have access to Network Data (3G/4G LTE) and running Android (4.1+).



***1.3 Project Deliverables***

1. CometRide running on client smartphone (citizens of UTD).
2. CometRide running on driver smartphone/tablet (driver of the shuttle).
3. Implement the objectives stated in “*Objectives – 1.1 Purpose, Scope and Objectives*”*, Page 1, Project Plan document.*
4. Deliver a working application of CometRide within April 25th, 2015.

***1.4 Schedule and Budget Summary***

**Schedule**

CometRide is being built as a part of the Capstone project (Advanced Software Engineering Project – spring 2015) at The University of Texas at Dallas. This project is guided by our ***Professor, Rym Z Wenkstern*** and actively supported by ***Ph.D. student Mohammad Al Zinati*** (Multi Agent & Visualization Lab). Since the spring semester is for a period of 4 months, we expect to deliver it by April 25th, 2015.

**Budget**

Currently we do not see any expenditure as all of us have smartphones. But, for project expenses including the necessary Bluetooth clicker ($30), we have a total budget of $200. The department's expenses of mobile phones, cases, mounts and data plans is not included in this since we don’t need them to develop the software.



**2. Project Organization**

**Roles and Responsibility**

**Program Manager:** Manages the workflow, co-ordinates portfolios, over sees communication channels, agrees on requirements through customer feedback and is responsible for the final product and defines deadlines (makes sure the project in on track).

**Software Design Engineer**: Responsible for designing the application in technical terms.

**Software Development Engineer**: Responsible for implementing the application from the design.

**Software Development Engineer in Test**: Responsible for testing the application and to approve it for commercial use.

**Site Reliability Engineer**: Making sure the application meets and adheres to the Service Level Agreement defined by the project. Is also responsible for de-bugging any live issues which may arise after the application is pushed into production.

*Note: Roles are not static. They may change based on the situation but are limited to the above roles only.*

**3. Managerial Process Plan**

***3.1 Startup plan***

1. Project roles are determined.
2. Vision document.
3. Come up with a clear and elaborate requirements document.
4. Discussing the risks with associated stakeholders.

***3.1.1 Estimation plan***

|  |  |
| --- | --- |
| ***Product*** | ***Cost*** |
| *Microsoft Project* | *NIL* |
| *IBM Rational Doors* | *NIL* |
| *IBM Rhapsody* | *NIL* |
| *Java IDE & Junit* | *NIL* |
| *Smatphones & Laptops* | *NIL* |

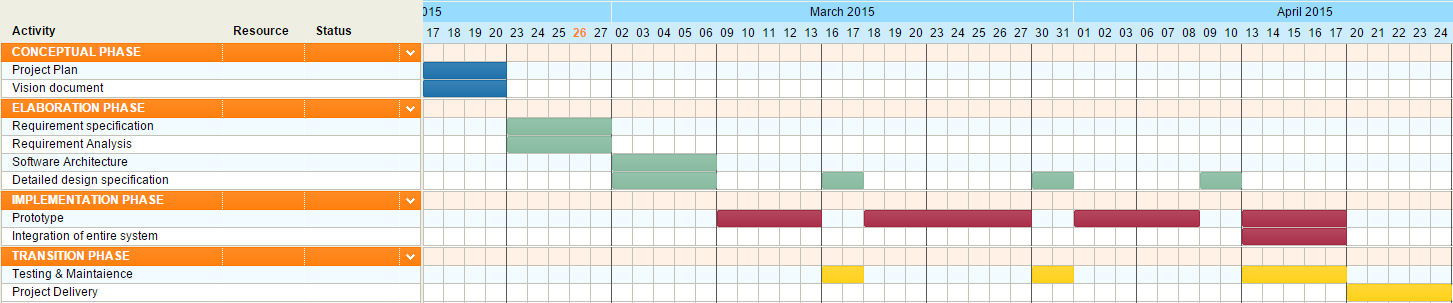
***Table 1 (Estimation plan)***

***3.1.2 Resource acquisition plan***

1. Most of the software needed for this project will be open – source. If they are proprietary, a trial version of the same or an open – source alternative will be adapted.
2. Since the team already has smartphones, we will use our own smartphones.
3. Since the team already has laptops, we will use the same for development.
4. Printing out documents will be done at the Open lab in the University.

***3.2 Work plan and schedule timeline***

1. **Conceptual Phase** – Vision document, Project Planning document
2. **Elaboration Phase** – Software requirement specification, software requirement analysis, software architecture and detailed design specification
3. **Implementation Phase** – Prototype with integration of all involved components
4. **Transition Phase** – Testing, Maintenance and Project delivery

******

***Figure 1 (Gantt Chart)***

**4. Technical Process Plan**

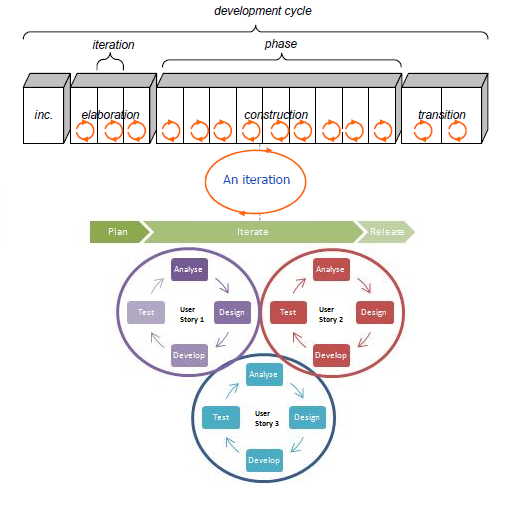
***4.1 Process model***

We have planned to use the Unified Process which is a standard software engineering process and the agile methodologies to develop the Comet Ride. The reason why we adopted UP is that it’s use case driven, risk driven, architecture centric with iterative and increment development. We have planned to incorporate the agile terminologies into the UP iterations. By which each use case would be further divided into user stories which would be cataloged in the product backlog.

Each user story would undergo a RADIT (Requirement, Analysis, Design, Implementation, and Testing) process with a different emphasis for each depending on where the iteration is in the lifecycle. Since we meet up with the product owner every week we believe that a constant feedback loop would exist which would greatly support the agile methodology.

We have modified the daily meeting propaganda of the agile to our need to an alternate day meeting during which we plan and compile our works together. Other than these minor changes UP would still follow all its development cycle phases namely inception, elaboration, construction and transition.

The following picture describes the project model that we would be using for our product. As it could be seen, it’s the incorporation of agile into UP.



***Figure 2***

***(Dev cycle)***

***4.2 Methods, Tools and Techniques***

**Methods**

Comet Ride will be developed using Unified Process development method. For the implementation cycle alone we would be incorporating agile methodologies.

**Development Environment**

The product will be developed in a cloud environment from the UTDallas campus. The cloud provider shall ensure a high availability of service.

**Target Environment**

The product will run on a mobile based environment supported by android. For future development we are also planning to launch the application in the iOS web platform as well.

**Techniques**

Comet Ride will be developed in an object oriented approach using Java. We plan to do the front end of the app using HTML5 and CSS3 so that it would be compatible across all android devices. For real time mapping of the shuttles we have planned to use the geo-location service provided by Google.

**Tools**

We have identified the following tools which we would be using over the course of this project.

* Amazon Elastic Compute Cloud Environment
* Github – Version Control
* Jira – Test Management Tool
* Microsoft Office 2010 / Microsoft Project
* IBM Thapsody/ IBM Rational Doors
* Eclipse / JUnit
* Notepad++
* Ripple –Emulator
* Smartphones/Tablets

**Document Standards**

The standards that we plan to follow are the once outlined by the IEEE standards association. 

***4.3 Infrastructure Plan***

**Desktop setting**

All PC’s would use Windows 7 as its operating system and would be provided with all the tools furnished in 4.2.5

**Network setting**

All the PC’s would be connected to the high speed internet. The project files would be maintained in a shared Github repository so that they could be easily updated by all the members. The project documents would be maintained in a shared Google drive.

**5. Supporting Process Plan**

***5.1 Configuration Management Plan***

### Configuration Management Tools

We would be using “Github” as the configuration management tool. GitHub is a web-based Git repository hosting service, which offers all of the distributed revision control and source code management (SCM) functionality of Git as well as adding its own features.

### Configuration Status Management

We would be using “Jira” as a task board to keep track of who is working on which part of the system. By linking the git repository to the jira account we can keep track of who is working on which version of the system as well.

## *Documentation Plan*

The documents that would be developed during the lifetime of the project would adhere to the IEEE standards. The lists of documents that we are planning to maintain over the course of the project are

* Project Management Plan
* Project Vision
* Requirement Specification
* Design Specification
* System Architecture Document
* System Test Plan

We intend to update the above listed documents over different iteration as the requirement raises.

## *Quality Assurance Plan*

We plan to achieve the highest quality possible for each artifact. To ensure high quality several activities like code review would be conducted in a periodic basis to ensure this. Quality reviews done on documentation artifacts would ensure that they adhere to the standards on which they are based, and that non-documentation artifacts would adhere to the plans/designs laid out by their requirement. .Some of the quality attributes that are important to our project are

**Availability**: The cloud service provided by the service provider has a high availability rate which would give the app real time data.

**Usability**: The app will provide the user with information like wait time and request shuttle with the click of a single button

**Performance**: The app shall provide the user with data within 3 sec it has been requested for.

****

**FEASEABILITY ANALYSIS**

**THE UNIVERSITY OF TEXAS AT DALLAS**

**1. Economic Feasibility Analysis**

A single Spy Tec GPS real time tracker with a weather proof magnetic case costs about $154 in Amazon. Additionally, the tracking plan requires no contract or activation fee and can be stopped or started at any time. The monthly subscription (starting at $25 a month) allows for real time tracking beginning with minute-by-minute updates, upgradable to as often as every five seconds.

**2. Technical Feasibility Analysis**

The Spy Tec STI\_GL300 Mini Portable Real Time GPS Tracker, as the name suggests uses GPS satellite technology to ensure a tight fix on the tracker’s location for an unsurpassed accuracy. In addition, the 3-axis accelerometer improves accuracy with motion detection. The real time location is updated in the Spy Tec GPS platform.

**3. Operational Feasibility Analysis**

The Spy Tec STI\_GL300 Mini Portable Real Time GPS Tracker is a pocket-friendly device, it is only a little larger than a matchbox and can be discreetly placed nearly anywhere. So, the drivers can just carry it in their pockets. Spy Tec also provides a waterproof magnetic case which allows for the tracker to be placed underneath the vehicles. This prevents the problem of the device getting misplaced or lost.

The device comes with a long-lasting battery allowing for two weeks of continuous use, or even longer when used periodically. Spy Tec also provides a battery pack and case accessory which can extend the battery life to a whopping six months.

**4. Schedule Feasibility Analysis**

Find attached Gantt chart in the Project Plan document

**PS: We are using the GPS in the smartphone itself for the prototype and not using the above Spy Tec GPS**

****

**THE UNIVERSITY OF TEXAS AT DALLAS**

**VISION DOCUMENT**

**1. Introduction**

The CometRide system will provide a solution to the low ridership and ignorance surrounding UT Dallas’s Comet Cab (“shuttle”) system by providing a mobile application for prospective riders to keep tabs on the shuttles.

**2. Positioning**

**2.1 Business Opportunity**

The market for our project is completely open for the taking and one that the Transportation Department has been looking to expand into. There’s no current shuttle management system in place even though it’s evident that one is desperately needed. By providing a means for the shuttles and riders to know about each other, we can turn the shuttle system into something that’s actually useful for people on campus.

**2.2 Problem Statement**

The only means the students have of knowing where the shuttles are and what their routes are is a static map on the UT Dallas website, meaning that trying to catch a shuttle to speed up movement around campus is at best a guessing game, and at worst impossible. Our system can easily fix this issue by making shuttle locations available to students.

**2.3 Product Position Statement**

For the UTD students who wish to make better use of the shuttles, CometRide is a mobile application that allows them to effectively keep track of the shuttles and their routes to better catch rides. Unlike the current static map, CometRide is dynamic and allows the user to judge whether the shuttle is the right call.

For the UTD Transportation Department, CometRide is a system that will let them keep track of their shuttles, as well as create and modify routes for the shuttles.

**2.4 Alternatives and Competition**

At present, the only alternative is a static map displaying the different routes viewable online. However, it provides no dynamic information, such as where the cabs are. Purchasable alternatives such as fleet management systems used by trucks are both expensive and lacking when it comes to the specifics needed by UTD. Our system could be built using GPS Units in place of cell phones, and issues such as phone battery life and quality of service initially lead us to believe the GPS units were the superior option, but the incredibly lower cost of the cell phones with extra hardware to make up for the shortcomings overshadowed that.

**3. User Description**

**3.1 User/Market Demographics**

There are two primary demographics for CometRide. First is students and university staff, who wish to use the shuttle service - theoretically speaking, every single one of the thousands of students and faculty at UTD are potential users. Guests to the university may also be short-term users, if they know of the application. The second is the UT Dallas Transportation Department, which is responsible for overseeing and managing the shuttle system.

**3.2 User Profiles**

As stated above, our system will have two types of users.

* The riders will be UT Dallas students (or, on occasion, university staff) who wish to use the shuttle service and will rely on the application to help them find their shuttle.
* The Transportation Department may want to generate temporary routes for special events or keep track of their shuttles.

**3.3 Key User Needs**

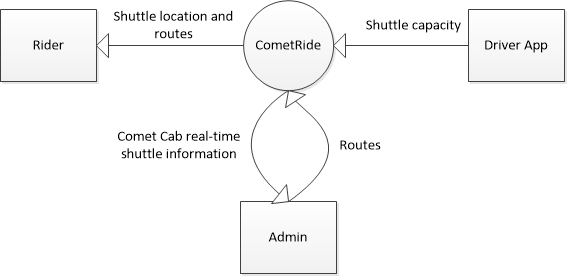
The key user needs for this application, aside from the functionality that have been discussed at length, are ease of use and responsiveness. After all, no users will use a difficult-to-use application, and if the application is slow to update the shuttle positions, the application is worse than useless because it will potentially mislead users.

**3.4 User Environment**

All users of this app will be located on the UT Dallas campus, regardless of what type of user they are. This application will also be exclusively accessed via Android smartphone. Data is generated by the shuttles (Their location) and consumed by the users (Who can view the shuttle locations and routes), making the application a relatively closed system. Given the time sensitivity of using the shuttles compared to walking, a fast response time is of the utmost importance to our system. If response times are slow, a potential rider can entirely miss a shuttle, causing them to uselessly stand around waiting for a shuttle that will not come any time soon. On the other hand, while service interruptions are far from ideal they’re actually more tolerable as long as they’re known about, since they will merely mark a return to today’s status quo. Security issues are of lesser concern with this system, however - The only information that will pass through the system is shuttle locations. Nothing that requires protecting will be used by our system. The primary security concern is protection of the physical GPS units, which will be placed under the shuttles.

**4. Product Overview**

**4.1 Product Perspective**



**4.2 Summary of Capabilities and benefits**

|  |  |
| --- | --- |
| **Feature** | **User Benefit** |
| Riders and Admins can track location of shuttles on campus | Save time - Riders will be able to know if it is worth waiting for the shuttle to come by, while Admins can ensure their drivers are actually doing their jobs |
| Riders can view shuttle routes | Riders will gain information vital to using the shuttles |
| Admins can create and modify routes | Riders will now know what the temporary routes are so they can take advantage of them |

**4.3 Assumptions and Dependencies**

It is assumed that riders of Comet Cab already own smart phones, and if they do not or wish to not participate that this system will not interfere in the use of the shuttles as they are used today. Also that there is enough internet on the UTD campus to support our needs. For success the UTD transportation department must be willing to install our final product on their shuttles. We are also dependent on the Android platform (Perhaps eventually IOS as well), our device’s GPS, Google Cloud Messaging, HTML, and sufficient internet access.

**4.4 Cost and Pricing**

The mobile application will be free to download, but smartphone will not be provided and must be bought separately, which would cost about $30. The monthly subscription for the GPS data connection would vary between $30 (3GB at 4G speeds) to $50 (5GB at 4G speeds). Cloud services can be handled via Google App Engine for free or a nominal monthly fee if the app proves very popular. The costs of things such as casings and mountings is presently unknown because a decision on specific models was apparently never made.

**4.5 Licensing and Installation**

A mobile holder has to be purchased for each shuttle. At this point licensing does not seem to be needed, but we will discuss this later if something arises.

****

**TRADE OFF ANALYSIS**

**THE UNIVERSITY OF TEXAS AT DALLAS**

**1. Introduction**

This trade off analysis was compiled based on the discussion we had with the P&T manager. Considering the various new constrains placed on the problem statement we came up with the following alternatives to track the shuttles location.

* GPS based tracking
* Mobile based tracking
* Radio based tracking

As far as tracking the capacity of the shuttle we have decided to track it based on the rider’s location. The implementation behind this idea is that we enforce the rider to hit the “Request Ride” button in the app if he/she wants to track the shuttle to hitch a ride. Then the rider has to select a drop location as well. So, once a shuttle crosses the rider’s location (the one who has requested for the ride) we take it that the rider has boarded that shuttle and increment its capacity by one provided it’s not already full. Similarly we decrement the count by one once the rider’s destination is met. By this we completely avoid the driver from interacting with the system. But of course we can’t track the riders who aren’t using the app. Thus the accuracy rate of this technique is debatable however; it satisfies the clients constraint by which the driver shouldn’t do any other activity while driving and this method doesn’t modifies the current shuttle by any means.

**2. Trade-Off Table**

The following trade-off table analyses the different alternatives we have come up with to track a shuttles location in real-time.

|  |  |  |  |
| --- | --- | --- | --- |
| **Project factor** | **Alternative 1** | **Alternative 2** | **Alternative 3** |
| **Description** | We use a real-time GPS tracker | We use a mobile phone which has an inbuilt GPS feature | We use a 2-way radio with GPS tracking |
| **Device** | Spy Tec STI\_GL300 Mini Portable Real Time GPS Tracker | ZTE Zinger | Garmin Rino 650 |
| **Critical features/ specs** | Battery life = 2 weeks  HxWxD = 5.4 x 1.9 x 2.2 inches; Weight 5.6 ounces. | Battery life = 6 hours  HxWxD = 4.6 X 2.4 X 0.48 inches; Weight 3.8 ounces.  CPU: Dual-core 1.2GHz MSM8210  OS: Android 4.4 (KitKat) | Battery life: 16 Hours of Battery  Channels: 22 channels available (15-22 License required)  Coverage: 3 km  HxWxD = 7.5 x 2.4 x 1.8 onches; weight 11.3 ounces. . |
| **Working** | The information from the tracker can be followed on any device from computers to tablets or mobile phones via the Spy Tec GPS platform. | We build an app which keeps updating the cloud server of the shuttles current location every 30seconds. | We need to load the BaseCamp software provided by Garmin to track the multiple Rino units. The Rino's serve as a router/antenna. This would replace the existing walkie-talkies used by the driver and admin. |
| **Monthly Expense** | Monthly subscription starts at $25 (Updates location every 1 min) - $45 (Updates location every 5 sec). | Monthly subscription starts at $30 (3GB @ 4G speeds) - $50 (5GB @ 4G speeds) | No monthly subscription. But a one time investment of $60 is required to purchase the map. |
| **Technical innovations/ risks** | Have to find a way to port the location from the Spy Tec GPS platform to the CometRide app. | The battery life is not that great. We need to allocate a mobile for each driver rather than the cab. | Communication range is low and polling is not that great. Additionally, have to find a way to port the location from the BaseCamp software to the CometRide app. |
| **Schedule** | 4 weeks | 3 weeks | 6 weeks |
| **Product Cost** | Price/Unit with Weatherproof Magnetic Case: $154 | Price/Unit: $30 | Price/Unit: $500 |
| **Notes and Recommendations** | Based on all analysis we have done. We have decided to go with this alternative | The battery life is a big let down and the clients weren’t very much interested with this alternative | The device is too pricy |

****

**DESIGN DOCUMENT**

**Transition from System to Software Engineering**

**THE UNIVERSITY OF TEXAS AT DALLAS**

**1. Introduction**

***1.1 Purpose, Scope and Overview***

**1.1 Purpose**

To develop a software application(CometRide) to enhance the efficiency & location tracking of the shuttle system at The Univesity of Texas at Dallas.

**1.2 Scope**

This application will be used by the citizens of UTD and can be extended to anyone who uses the on campus shuttle system. It will be used for locating operating shuttles in real time.

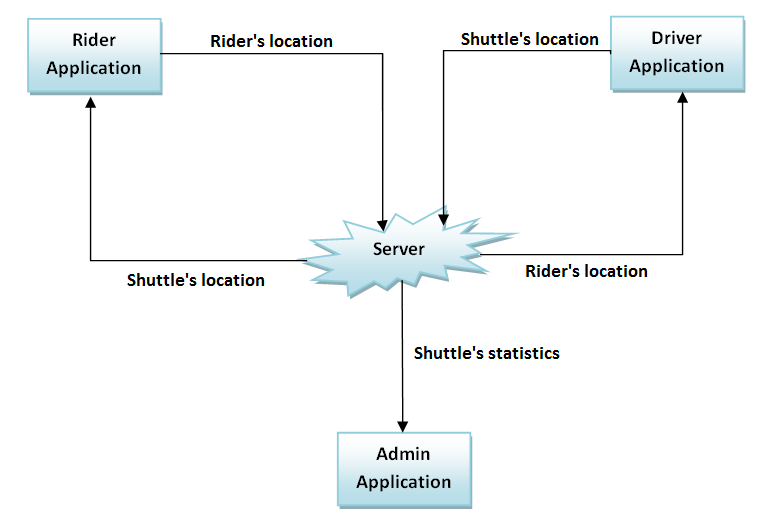
**1.3 Overview**

The rest of the document gives us an overall view and description about the CometRide system. In section 2, the overall description of the product in hand is analyzed. In section 3, the hardware specification of the product is discussed. In section 4, the external interface requirement is dealt upon. In section 5, the different features provided by the product are discussed extensively. In section 6, concentrates on the NFR’s and constraints imposed on the product.

**2. Overall Description**

CometRide is an android app which is designed to support three types of user roles namely the rider, driver and admin. The app pushes real-time geo location to the cloud server for a driver role. The customer can see the real time location of the shuttles in a map and can request for a ride based on his/her current location.

The following diagram explains the capability of different roles in the system.

***2.1 Product Perspective***

**Figure 1(Product Perspective)**

***2.2 Product Functions***

1. The rider can view the real-time location of the shuttle in a map.
2. The rider can see the number of free seats available in a respective shuttle.
3. The admin can alter shuttle routes.
4. The admin can view statistics of each shuttle.
5. The driver can update the shuttles status and capacity.

***2.3 User Characteristics***

The users don’t need any special training to use the app. For usability reason a onetime tutorial is integrated with app which explains how to use the app to the users for the first time.

***2.4 Constraints***

1. Customers using this application must have a smartphone equipped with GPS, have access to Network Data(3G/4G LTE) and running Android(4.1+).
2. Each shuttle will have its own dedicated smart phone/tablet running CometRide also equipped with GPS, have access to Network Data(3G/4G LTE) and running Android(4.1+).

***2.5 Assumptions & Dependencies***

1. Majority of the students and faculty have smartphones.
2. Customers are familiar with using a smartphone application.
3. Network data and GPS availability is 100% throughout The University of Texas at Dallas.
4. The driver of the shuttle is familiar with interacting with a smartphone application.

***C:\Users\Vaibhav\Desktop\2000px-Bluetooth.svg.png*3. Hardware Specification**

***3.1 Hardware Component 1 – Android powered tablet coupled with clicker***

A mobile that runs the Android operating system. Must have GPS and mobile data capabilities, but beyond that any model will do. Should have Bluetooth protocol to connect clicker.

**3.1.1 Functionality**

* The tablet shall possess a standard GPS location sensor and active Bluetooth.
* The tablet shall also be able to use mobile data via 3G or 4G connections.

**3.1.2 Operational Requirements**

* The tablet shall be placed within the shuttle it works with.
* The tablet and clicker shall be operated by the shuttle driver.

**3.1.3 QoS Requirements**

N/A

**3.1.4 Parametric Requirements**

N/A

***3.2 Hardware Component 1 – Rider’s smartphone***

A mobile that runs the Android operating system. Must have GPS and mobile data capabilities, but beyond that any model will do.

**3.1.1 Functionality**

* The smartphone shall possess a standard GPS location sensor and active data connection.
* The smartphone shall run Android 4.2 +

**3.1.2 Operational Requirements**

* The smartphone belongs to the Rider.
* The smartphone is personal property of the Rider wishing to use the system.

**3.1.3 QoS Requirements**

N/A

**3.1.4 Parametric Requirements**

N/A

**4. External Interface Requirements**

***4.1 User Interfaces***

### 4.1.1 General

* The user interface shall be user friendly by having a uniform look and style across all screens.
* The user interface shall have understandable icons and operational instructions.
* The user interface shall provide a pictorial map view similar to the standard Google Maps application and markers to show both shuttle locations and rider locations.

### 4.1.2 Riders

* The user interface shall allow a rider to track the real-time location of all the cabs with their capacity once they open the app.
* The user interface for riders shall be consistent under popular smartphone resolutions.

### 4.1.3 Drivers

* The drivers shall be able to update their status (On/off duty) with a single button from the same screen as the map.
* The drivers shall be able to increment or decrement their rider count with buttons on the same screen as the map.
* The driver screen shall be disabled when the shuttle is in motion.

### 4.1.4 Admins

* The admins shall be able to create new routes using a pictorial map interface.
* The admins shall be able to view shuttle locations in real time using a pictorial map.
* The admins shall be able to select a shuttle to view detailed statistics on with no more than two "clicks" from the shuttle locations screen.

***4.2 Hardware Interfaces***

N/A

***4.3 Software Interfaces***

* The Android mobiles on the shuttles and rider smartphones shall connect to a cloud server via mobile data connections.
* The Cloud Server shall receive data on users who desire a ride as well as shuttle status and location and maintain this data in a database.
* The Driver, Rider and Admin systems shall all be able to read the data they need from the Cloud Server.

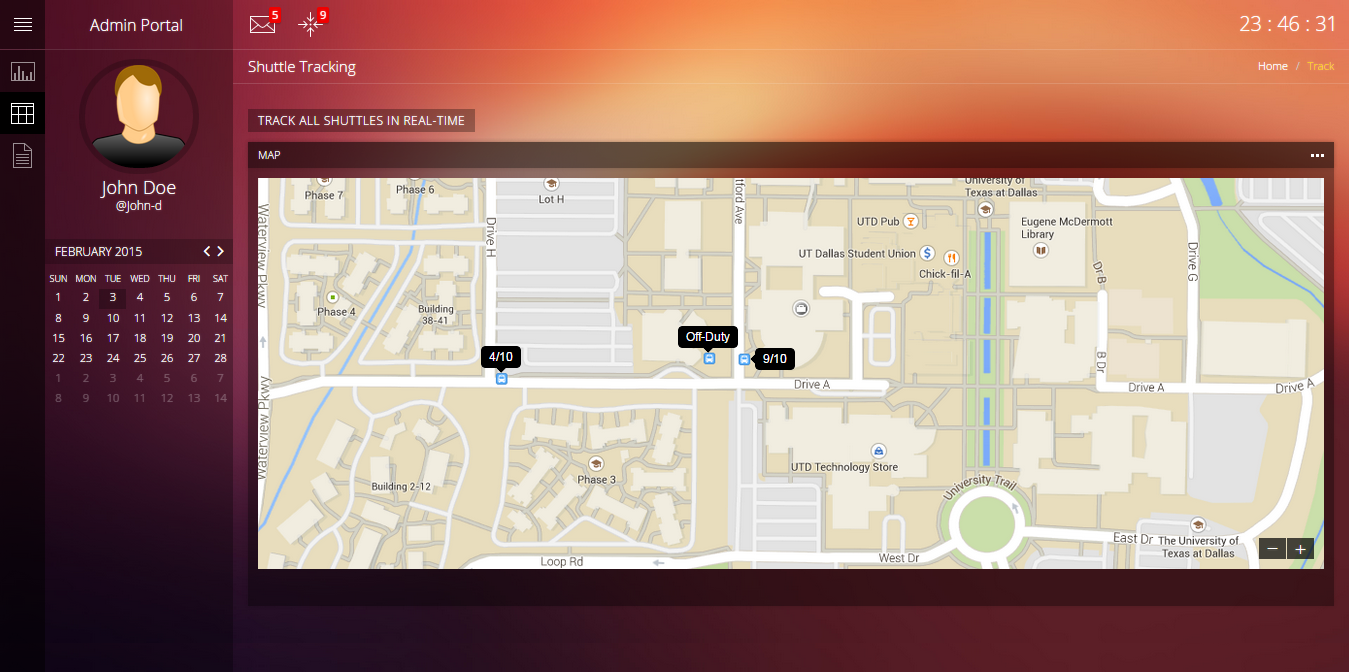


Figure 2(Admin portal tracking) – Mockup only. NOT final UI.



Figure 3(Admin portal home) – Mockup only. NOT final UI.

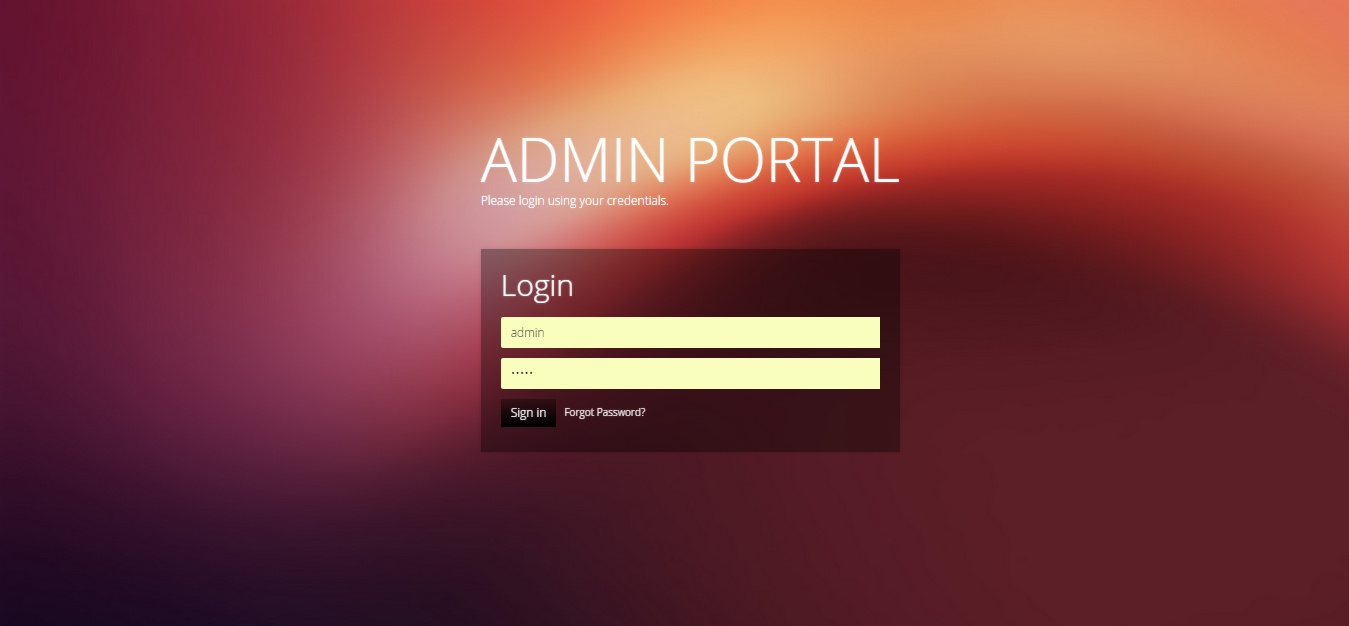


Figure 4(Admin portal login) – Mockup only. NOT final UI.



Figure 5(Admin portal) – Mockup only. NOT final UI.

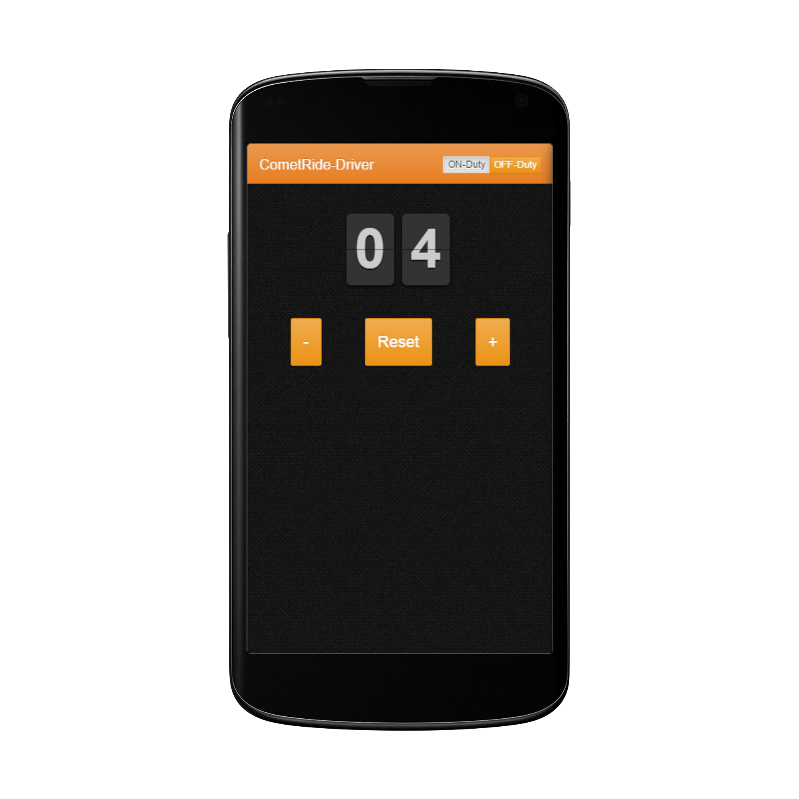


Figure 6(Driver counter) – Mockup only. NOT final UI.



Figure 7(Shuttle status) – Mockup only. NOT final UI.

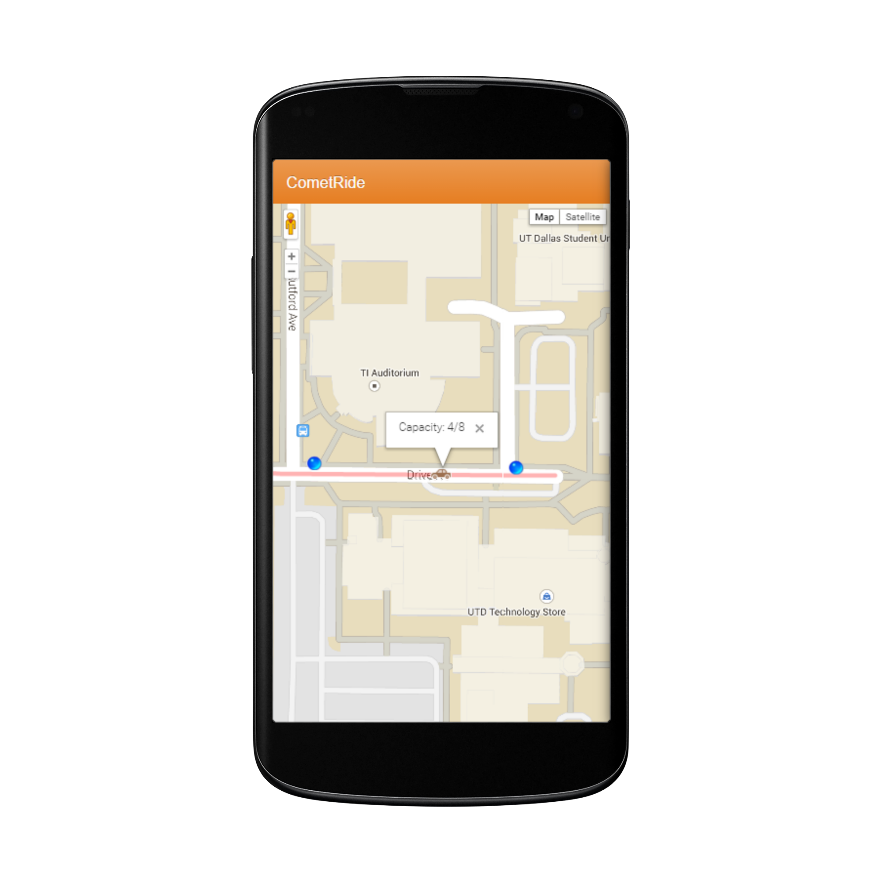


Figure 8(Rider screen) – Mockup only. NOT final UI.

***4.4 Communication Protocol and Interfaces***

The Android Tablets and rider smartphones shall use standard 3G or 4G (As applicable) mobile data connections as provided by a cellular service provider.

**5. System Features**

* 1. ***System Feature(Rider - A): View and track shuttles***

**5.1.1 Description**

The potential rider is given the option to view the shuttle’s location in real-time on their mobile device. It will show a map with the shuttle’s location displayed as an icon that will be updated with the shuttle’s location. The system will start by receiving data from the cloud (GPS location).

**5.1.2 Action/Result**

Action: The rider selects the “shuttle location” option in the application.

Result: A map with the shuttle’s location displayed as an icon will be displayed.

**5.1.3 Functional Requirements**

1. The rider’s mobile application shall display a map with the shuttle’s location designated.
2. The driver’s application shall send its location to the server.
3. The rider’s application shall receive the location from the server.
4. The location of the shuttle shall be displayed within 5 seconds once the user opens his/her app.
5. The rider’s app shall update whenever there is an update in the shuttles location or capacity in the server.
   * 1. **NFR**

* The shuttle’s location shall be updated every second.
* The map and location shall load in under 5 seconds when sufficient data connection exists.

***5.2System Feature (Rider -B): View the shuttle capacity in real time***

**5.3.1 Description**

The potential rider is given the option to view the shuttle’s capacity in real-time on their mobile device. It will show the total number of seats available in the shuttle and the number of them that are occupied.

**5.3.2 Action/Result**

Action: The rider selects the shuttle icon that appears on the map.

Result: A pop up appears above the shuttle icon which displays the number of seats available.

* + 1. **Functional Requirements**

1. The rider’s application shall receive the capacity from the server.
   * 1. **NFR**

* The popup shall appear in less than 3 seconds when sufficient data connection exists.
* The capacity of the shuttle shall be updated every 30 seconds in the server from the driver’s app.

***5.3System Feature (Rider -C): View the shuttle routes in real-time***

**5.3.1 Description**

The potential rider is given the option to view the shuttle’s routes in real-time on their mobile device. It shall show the currently active routes the shuttles are using for that particular day. It will also show the safe spots to board and get-off the shuttle.

**5.3.2 Action/Result**

Action: The rider opens the CometRide app in his/her phone.

Result: All routes would be shown in different colors with the cabs running on them.

* + 1. **Functional Requirements**
  1. The current route shall be synced from the server every 15 minutes.
  2. The routes shall be displayed with a highlighted color.
     1. **NFR**
* Safe zones to board and de-board should be clearly marked.

***5.4 System Feature (Driver - A): Update the shuttle capacity***

**5.5.1 Description**

The diver will have the capability to keep count of the number of riders who are currently in the shuttle at any given time. He will be able to do the same by pressing the plus/minus option which appears above the map in the application.

**5.5.2 Action/Result**

Action: The driver can select “plus/minus” option in the application as shown in figure 7 to update the counter based on the number of riders in the shuttle at a given time.

Result: The seat count gets updated in apps display.

* + 1. **Functional Requirements**

1. The driver’s application shall update the server of the number of seats available in the shuttle.
2. The driver application shall disable the counter when the shuttle is in motion.
   * 1. **NFR**

* The app shall update the server every 30 seconds.

***5.5 System Feature (Driver - B): Update shuttle status***

**5.6.1 Description**

At any given time, a shuttle can be "on duty", meaning that it is actively carrying passengers along its route or "off duty", meaning that it is not accepting passengers at that time. A shuttle can go off duty for a variety of reasons (Driver change, out of gas, and so forth), which the driver needs to be able to indicate so people don't wait on a ride that will not come.

**5.6.2 Action/Result**

Action A: The driver switches his status to "Off Duty"

Result A: The status is pushed to the rider application, notifying them that the shuttle is off duty and not accepting riders.

Action B: The driver switches his status to "On Duty"

Result B: The status is pushed to the rider application, notifying them that the shuttle is on duty and accepting riders

**5.6.3 Functional Requirements**

1. When the driver indicates a status change, the new status shall be pushed to the cloud server, where it will be read by the riders.
2. The shuttle icon shall disappear from the rider’s app once the status is changed to off duty.
   * 1. **NFR**

* The driver’s application shall update the server in 3 seconds once its status is changed.

***5.6System Feature (Admin - A): Create and modify routes***

**5.8.1 Description**

Create and modify routes of shuttle in real time. This feature will help admin to modify the existing routes or create new once for special occasions.

**5.8.2 Action/Result**

Action: Click on modify shuttle routes.

Result: Show existing shuttle routes (different shuttles) as shown in figure 5. The admin would be able to modify the existing or create a new route by dragging or drawing new lines in the map using the line tool available in the map.

**5.8.3 Functional Requirements**

1. The admin page shall have an option to modify routes and update it to all shuttles in real time.
2. The updates made by the admin shall get reflected in the rider’s application.
3. The admin shall be able to allocate each route to a driver from a list of drivers which is made available.

**5.8.4 NFR**

* Modifying a route shall not disturb the current route (or) confuse the driver

***5.7System Feature (Admin -B): View statistics***

**5.7.1 Description**

The admin shall be able to view the statistics information which comprises of the log for each day. The log for each day comprises of the number of people who traveled on that shuttle, what was the time period for peak traffic and the busiest locations on that route.

**5.7.2 Action/Result**

Action: Click on view statistics button and select one of the shuttles.

Result: Shall display the different shuttles operational on that day including their travel log, peak student traffic time and view busiest student pick up locations as shown in figure 3.

* + 1. **Functional Requirements**

1. The capacity count of each shuttle shall be used to estimate the peak student traffic at a particular stop.
2. The geolocation tracked from each shuttle shall be used to compile the travel log for individual shuttles.
   * 1. **NFR**

* Logs older than 6 months shall be deleted.
* The statistical information is expected to grow in an exponential manner so we shall store these information in a MongoDB which would increase the performance of the entire system to a great extent when compared to the traditional RDBMS.

***5.8 System Feature (Admin - C): Create and Modify accounts***

**5.9.1 Description**

The admin is able to create, authenticate and modify all the accounts associated with CometRide.

**5.9.2 Action/Result**

Action: The admin logs into the admin website, selects create/modify/delete preferences.

Result: Based on the selection, the respective action is implemented and committed.

**5.9.3 Functional Requirements**

1. The website shall display the current accounts holders, dependencies before deleting an account and other related statistical information based on the selection made by the Admin

**5.9.4 NFR**

* Admin will be authenticated again before providing access to this module.
* For account deletion, a second approval can be implemented.

**6. Non-Functional Requirements**

***6.1 Produt NFR***

**Usability**

* The product shall provide information about the location of shuttles when the user accesses the mobile app.
* The product shall provide a uniform look and feel between app screens.
* The product shall provide a user instruction with specific operational steps if the user finds it hard to navigate the app.

**Avaialbility**

* The product is required to be available 100% from 7am to 11pm everyday.
* The cloud server (Google app engine) to be used garantees a 99.9% uptime.
* Data connectivity on the mobile shall be reliable.

**Performance**

* The maximum time limit for the app to respond for any service will be 5 seconds.
* The MongoDB has a higher performance than the traditional RDBMS when exponential amount of data is considered.

**Scalability**

* The product shall be scalable to include new shuttle routes & new student users.

**Maintainability**

* All modules are required to have low coupling and high cohesion for inter-platform portability and debugging. The layer architecture that we have planned to use would reduce the dependecy of the modules as shown in the component diagram.

**Portability**

* Currently CometRide app will only be available on Android devices.

***6.2 Process NFR***

* Agile model with the Rational Unified Process is being used in the creation of the product
* Configuration management tool Github shall be used
* Cloud documentation repository Google Drive is being used to track all documents
* Every week the team is required to give a status report to Prof. Wenkstern which contains the next upcoming deliverable, portfolio breakdown and what is being delivered in the current iteration
* We shall work towards delivering the final product on April 25th, 2015.

***6.3 External NFR***

* Before a student (or) driver can use the application, a disclaimer should be made avaialble on the starting screen claiming CometRide shall not be responsbile for any damage to proeprty, injury to citizens of UTD (or) loss of life as a result of using the application.
* A solid terms and conditions shall be avaialble before the final delivery date by consulting with a law student (or) a UTD attorney.
* Our team shall not be responsible for the application after the delivery date as we will be graduating. All rights will be with the Multi Agent and Visualization Lab at The University of Texas at Dallas.
* The cloud backend we use shall have a response time of 3 to 5 seconds on 4G LTE data.

**7. Use Case Diagram**

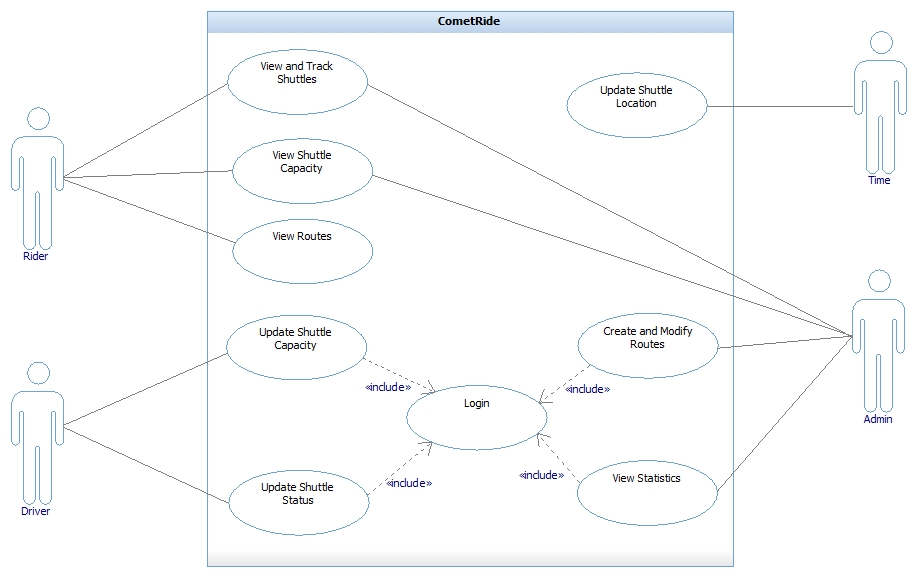
******

Figure 9(Use Case Diagram)

***1.1Description***

* **View and Track Shuttles:** The riders and the admin would be able to view the location of different shuttles in real time.
* **View Shuttle Capacity:** The riders and the admin would be able to view the number of available seats in different shuttles in real time.
* **View Routes:** The rider can see the entire servicing route of the shuttle(with safe spots to get on & get off the shuttle) for that entire day
* **Update Shuttle Capacity:** The driver has the ability to update the shuttle’s capacity.
* **Update Shuttle Status:** The driver has the ability to update whether the shuttle is on-duty or not.
* **View Statistics:** The admin can view different statistics of a shuttle and its driver.
* **Create and Modify Routes:** The admin has the ability to modify an existing route or create a new one.
* **Update Shuttle Location:** The driver’s application would update the shuttle’s location every 5 seconds.
* **Create and Modify Accounts:** The admin has the ability to create, modify, and delete accounts for drivers and other admins.

**8. Prority List**

The list of use cases ordered based on their priority is listed below.

1. View and Track Shuttles
2. Update Shuttle Location
3. View Shuttle Capacity
4. View Routes
5. Update Shuttle Capacity
6. Update Shuttle Status
7. Create and Modify Routes
8. View Statistics
9. Create and Modify Accounts

|  |  |  |
| --- | --- | --- |
| **Fully Dressed** | **Casual** | **Brief** |
| 1.    View and Track Shuttles | 2.    Update Shuttle Location | 3.    View Shuttle Capacity |
|  |  | 4. View Routes |
|  |  | 5.    Update Shuttle Capacity |
|  |  | 6.    Update Shuttle Status |
|  |  | 7.    Create and Modify Routes |
|  |  | 8.    View Statistics |
|  |  | 9. Create and Modify Accounts |

Table 1 (Priority List)

**9. Use Case Description**

|  |  |
| --- | --- |
| **Use Case Name** | **Login** |
| ID | 0 |
| Brief description | The drivers and the admin will be able to log into the CometRide system. |
| Primary actors | Driver, Admin |
| Second actors | None |
| Pre-Conditions | The drivers and admin must have an account in the system. |
| Main flow | 1. The user shall access the login screen of the mobile/web application 2. The user enters the username and password via the keyboard/touch screen in the respective fields provided. 3. The system verifies the user. |
| Post conditions | 1. The user is logged in to the system. 2. The user can access the assigned home page. |
| Non-Functional Requirements | 1. The login should happen within 5 seconds. 2. Multiple users can be logged in at any given time. |
| Technology and Data Variation List | The driver can use the portable device provided to him by the admin or his own. |
| Open Issues | Will the account freeze if the wrong username and password is typed more than three times? |

|  |  |
| --- | --- |
| **Use Case Name** | **View and Track Shuttles** |
| ID | 1 |
| Brief description | The user can view and keep track of shuttle locations, which will be updated for them in real time. |
| Primary actors | Rider or Admin |
| Second actors | None |
| Pre-Conditions | At least one shuttle is active and broadcasting its location |
| Main flow | 1. The user opens their application 2. The application receives information on the current shuttle routes and location 3. The user views the shuttle route and information location overlaid on a map. 4. Periodically, the application receives updated information, which the user can then view (Repeat steps 2 and 3 as new information comes in) |
| Post conditions | None |
| Alternative Flows | 1. If the user is an Admin instead of a Rider, they must log in between steps 1 and 2. 2. If no shuttles are active and broadcasting location, the user will view a map with no shuttle information until a shuttle begins broadcasting, at which point go to step 2. |
| Non-Functional Requirements | 1. Performance: The application shall update the data every second 2. Performance: The map and location shall load in under 10 seconds when a mobile sufficientdata connection exists |
| Technology and Data Variation List | None |
| Open Issues | None |

|  |  |
| --- | --- |
| **Use Case Name** | **Update Shuttle Location** |
| ID | 2 |
| Brief description | The driver's application will update the shuttles location after 1 second |
| Primary actors | Time |
| Second actors | Driver |
| Pre-Conditions | 1. Driver is logged into the driver application 2. Shuttle's status is indicated as “On Duty” |
| Main flow | 1. Obtain device's location 2. Send location to server 3. Server sends location to riders who are online |
| Post Conditions | None |

|  |  |
| --- | --- |
| **Use Case Name** | **View Shuttle Capacity** |
| ID | 3 |
| Brief description | The rider has the ability to view each shuttle’s capacity |
| Primary actors | Rider |
| Second actors | None |
| Use Case Name | View Routes |
| ID | 4 |
| Brief description | The rider has the ability to view each shuttle’s route |
| Primary actors | Rider |
| Second actors | None |

|  |  |
| --- | --- |
| **Use Case Name** | **Update Shuttle Capacity** |
| ID | 5 |
| Brief description | The driver has the ability to update the shuttle capacity |
| Primary actors | Driver |
| Second actors | None |

|  |  |
| --- | --- |
| **Use Case Name** | **Update Shuttle Status** |
| ID | 6 |
| Brief description | The driver has the ability to update the shuttle’s status. Whether it is on duty or off duty. |
| Primary actors | Driver |
| Second actors | None |
| **Use Case Name** | **Create and Modify Routes** |
| ID | 7 |
| Brief description | The admin has the ability to modify an existing route or create a new one. |
| Primary actors | Admin |
| Second actors | None |
| **Use Case Name** | **View Statistics** |
| ID | 8 |
| Brief description | The admin can view different statistics of a shuttle and its driver. |
| Primary actors | Admin |
| Second actors | None |
| **Use Case Name** | **Create and Modify Accounts** |
| ID | 9 |
| Brief description | The admin has the ability to create, modify, and delete accounts for drivers and other admins |
| Primary actors | Admin |
| Second actors | None |

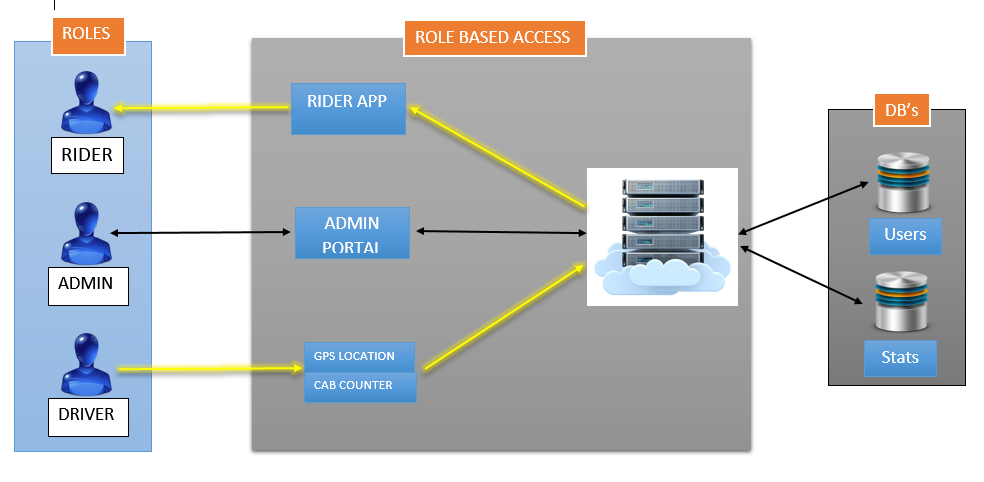
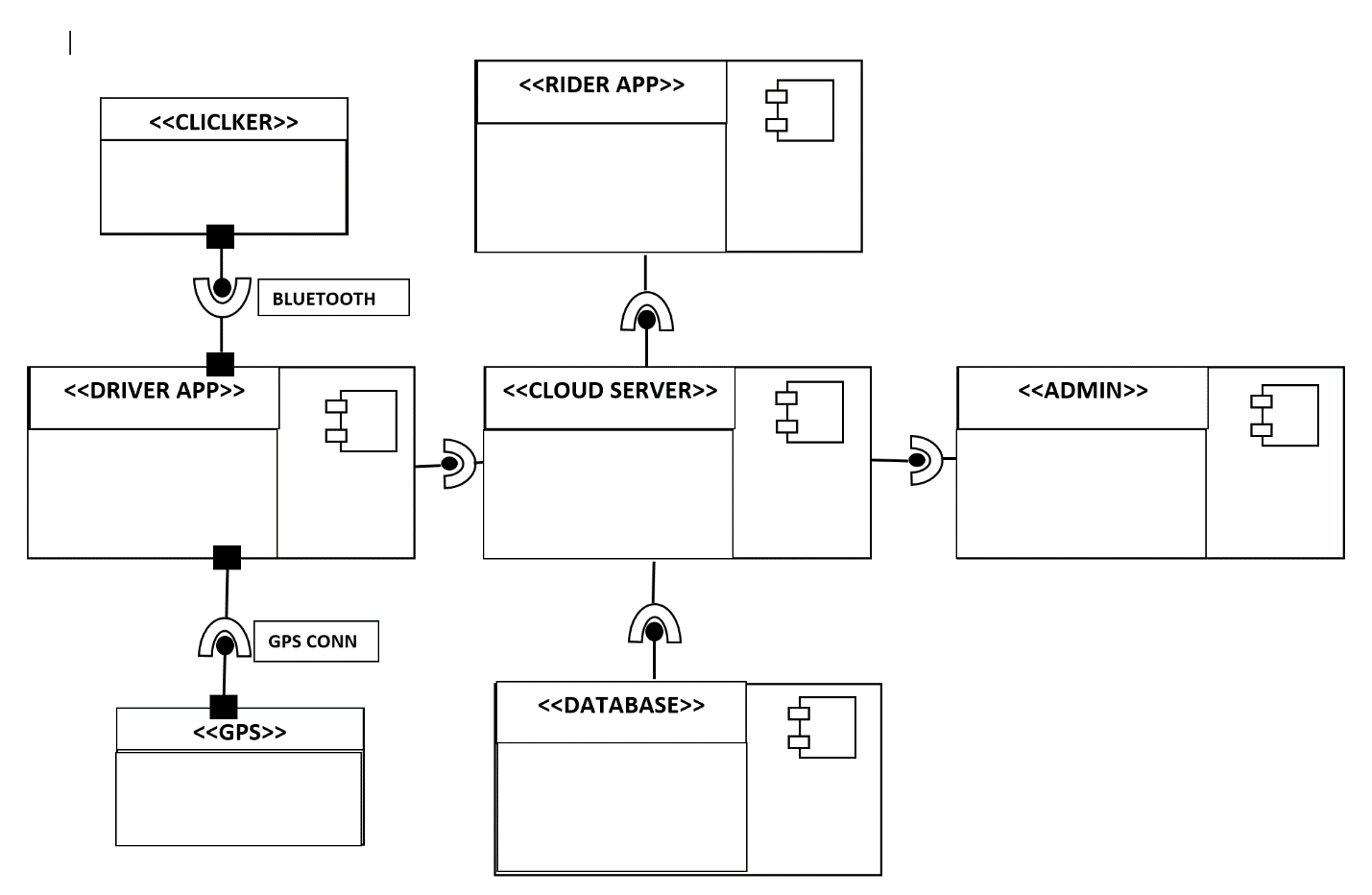
1. **High – Level Architecture Diagram**

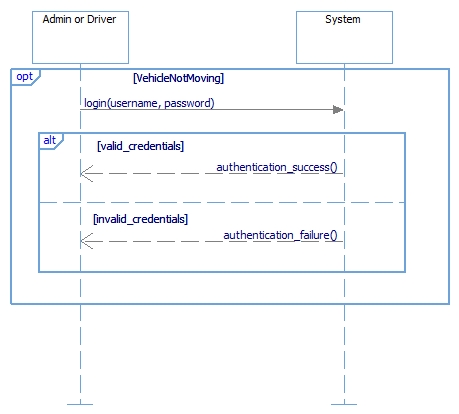
Figure 10(High Level Architecture Diagram)

**11. Component Diagram**

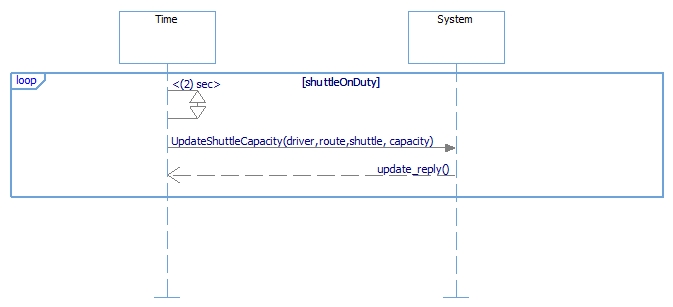
Figure 11(Component Diagram)



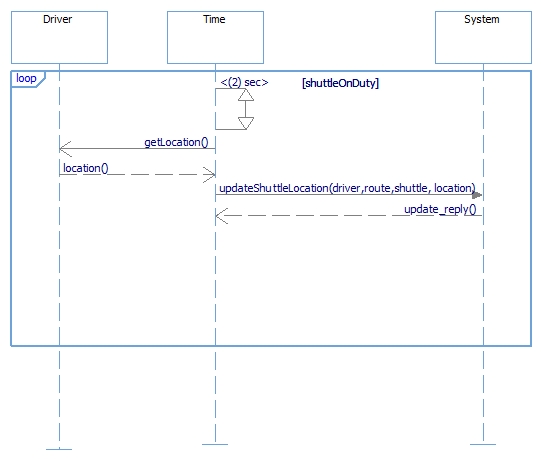
**12. Black Box Sequence Diagram**



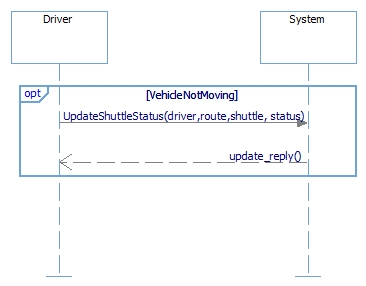
**Figure 12. LOGIN**



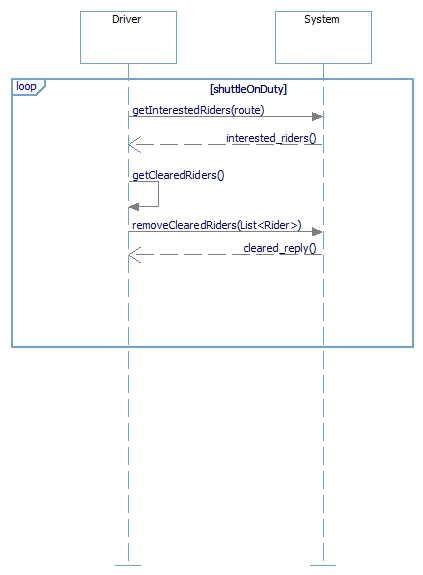
**Figure 13. DRIVER: UPDATE SHUTTLE CAPACITY**



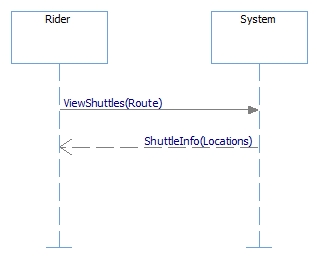
**Figure 14. GPS/DRIVER: UPDATE SHUTTLE LOCATION**



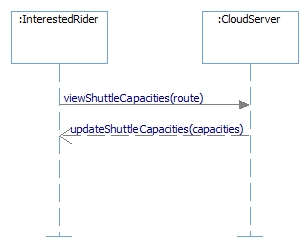
**Figure 15. DRIVER: UPDATE SHUTTLE STATUS**

****

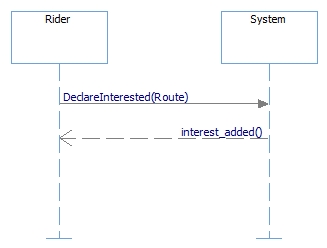
**Figure 16. DRIVER: HANDLE INTEREST**

****

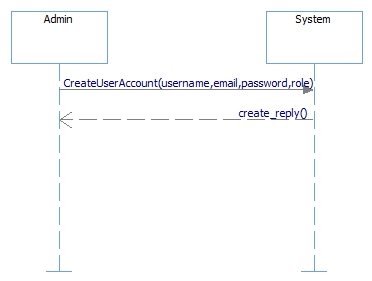
**Figure 17. RIDER: VIEW SHUTTLE ROUTE**



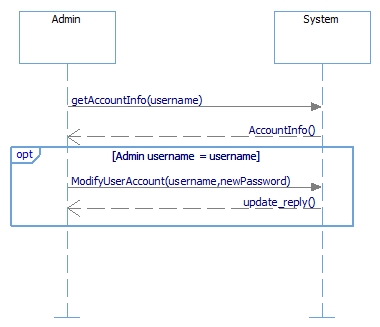
**Figure 18. RIDER: VIEW SHUTTLE CAPACITY**

****

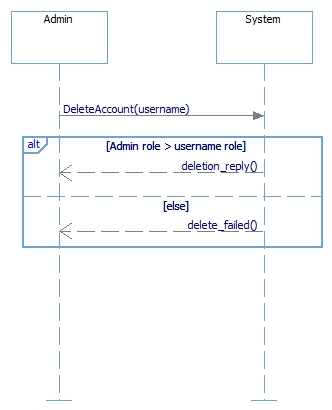
**Figure 19. RIDER: DECLARE INTEREST**



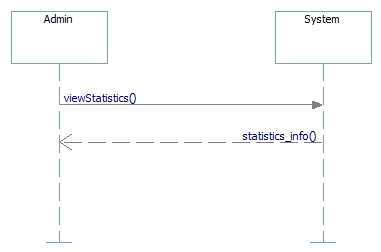
**Figure 20. ADMIN: CREATE ACCOUNTS**

****

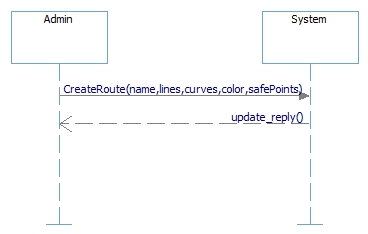
**Figure 21. ADMIN: MODIFY ACCOUNTS**

****

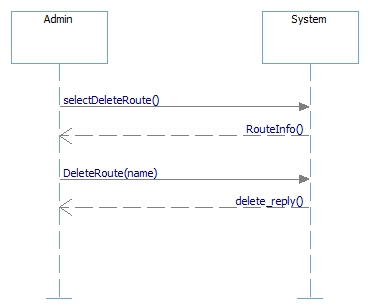
**Figure 22. ADMIN: DELETE ACCOUNTS**

****

**Figure 23. ADMIN: VIEW STATISTICS**

****

**Figure 24. ADMIN: CREATE ROUTES**

****

**Figure 25. ADMIN: DELETE ROUTES**

**13. Operation Contracts**

Contract CO1: viewCapacity()

Cross Reference: View Shuttle Capacity

Pre Conditions: A shuttle instance s that the user wants to view has already been created

Post Conditions: s.capacity is sent to the user

Contract CO2: updateShuttleStatus(shuttle, status)

Cross Reference: Update Shuttle Status

Pre Conditions:

- Driver is logged in to the system

- A shuttle instance s that the user wants to update has already been created

Post Conditions:

- If a matching shuttle instance s does not already exist, it is created

- s.status is set to status based on a shuttleID match

Contract CO3: selectModifyRoute()

Cross Reference: Create and Modify Routes

Pre Conditions:

- Admin is logged in to the system

- At least one route r has already been created. If no route exists, one is created.

Post Conditions: A route r is selected for modification

Contract CO4: modifyRoute(route, route\_info)

Cross Reference: Create and Modify Routes

Pre Conditions: A route r has been selected for modification

Post Conditions: [r.info](http://r.info/) becomes route\_info

Contract C05: viewStatistics

Operation: viewStatistics()

Cross References: View Statistics

Preconditions: Admin is logged in to the admin website

Postconditions:

* a Statistics instance s was created
* attributes of s were initialized
* DriverStatistics instances ds are formed for each Driver
* each ds is associated with s
* attributes for each ds are set from database information
* each ds is associated with routes

Contract C06: updateShuttleLocation

Operation: updateShuttleLocation(location, shuttle)

Cross References: Update Shuttle Location

Preconditions: riders who are connected are known through protocol

Postconditions:

* shuttle's location and info is sent to connected riders

Contract C07: selectModifyDriverAccount

Operation: selectModifyDriverAccount()

Cross References: Create and Modify Account

Preconditions: Admin is logged in to the admin website

Postconditions:

* Driver instances d are created
* attributes of each d were initialized from database
* each d is associated with a DriverList

Contract C08: modifyDriverAccount

Operation: modifyDriverAccount(account, account\_info)

Cross References: Create and Modify Account

Preconditions: Admin is logged in to the admin website, Driver instance has been created and modified

Postconditions:

* attributes of Driver instance d are updated in database

Contract CO9: login(username, password)

Cross Reference: Login

Pre Conditions: A driver/admin instance e already exists in the system

Post Conditions: - e.employeeUserId becomes username and e.employeePassword becomes password.

Contract CO10: viewShuttle(shuttle)

Cross Reference: View Shuttle and View Shuttle Capacity

Pre Conditions: A shuttle instance s that the user wants to view has already been created

Post Conditions: s.location and s.capacity is sent to the user and displayed

Contract CO11: updateShuttleCapacity(shuttle, capacity)

Cross Reference: Update Capacity

Pre Conditions:

- Driver is logged in to the system

- A shuttle instance s that the user wants to update has already been created

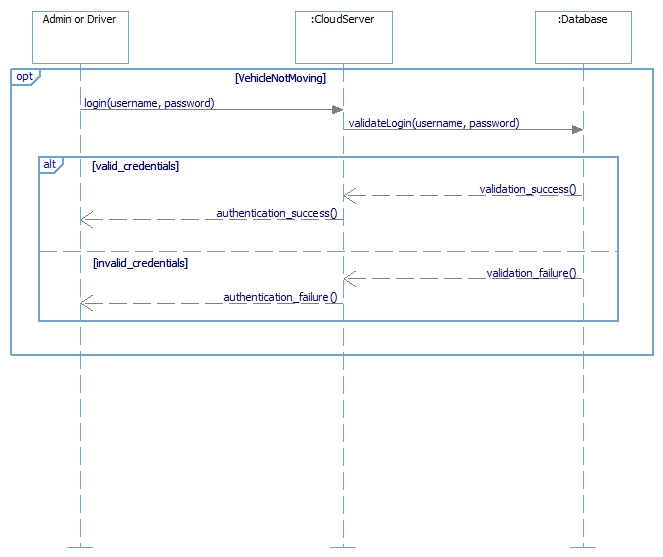
Post Conditions:

- If a matching shuttle instance s does not already exist, it is created

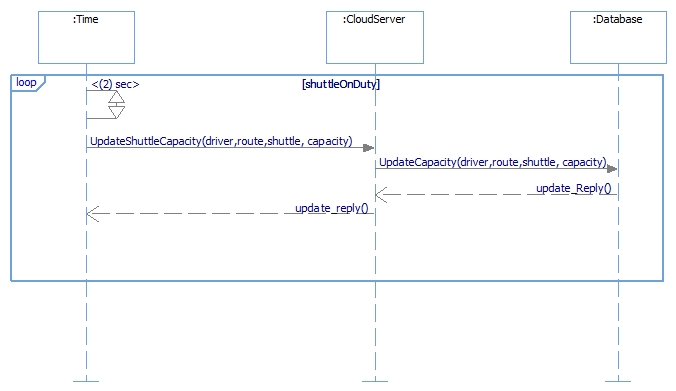
- s.capacity is set to capacity based on a shuttleID match

https://ssl.gstatic.com/ui/v1/icons/mail/images/cleardot.gif

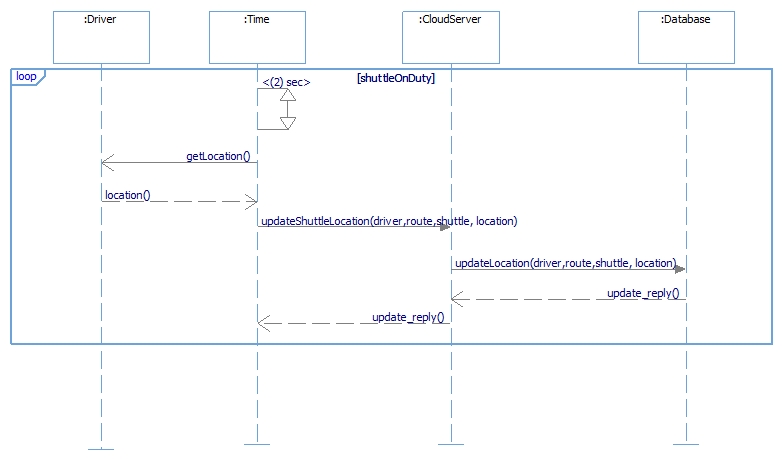
**14. Whitebox Sequence Diagrams**



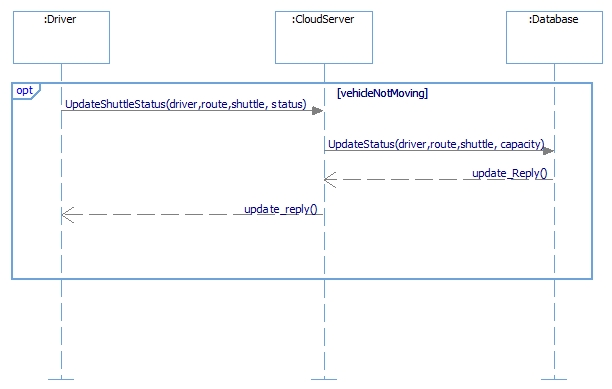
**Figure 26. LOGIN**



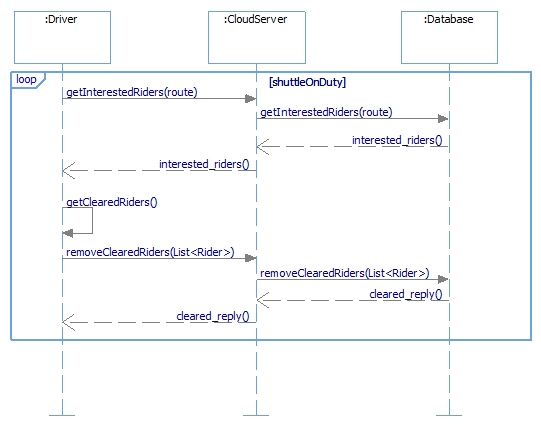
**Figure 27. DRIVER: UPDATE SHUTTLE CAPACITY**



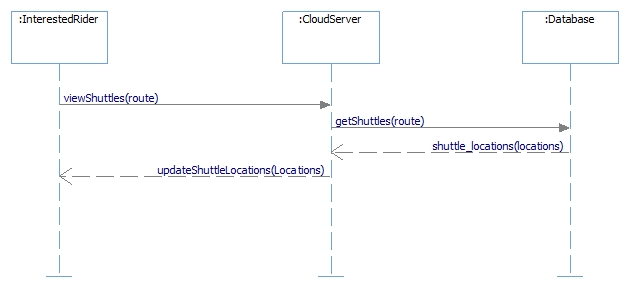
**Figure 28. GPS/DRIVER: UPDATE SHUTTLE LOCATION**



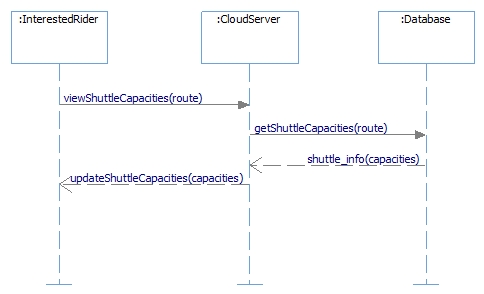
**Figure 29. DRIVER: UPDATE SHUTTLE STATUS**



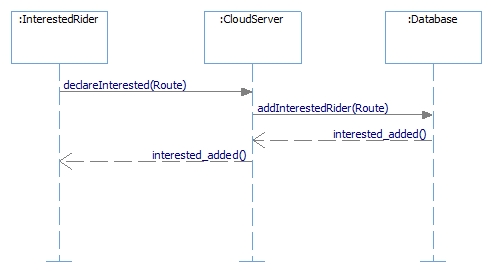
**Figure 30. DRIVER: HANDLE INTEREST**

****

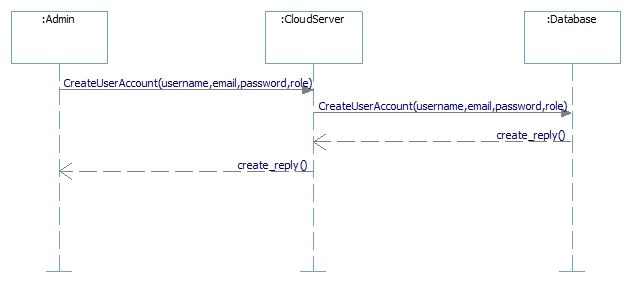
**Figure 31. RIDER: VIEW SHUTTLES**



**Figure 32. RIDER: VIEW SHUTTLE CAPACITY**



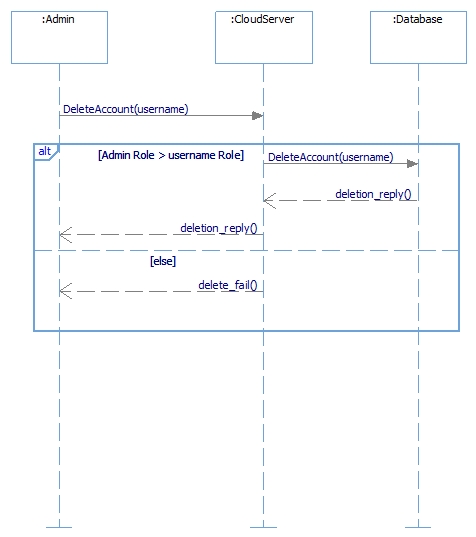
**Figure 33. RIDER: NOTIFY INTERESTS**

****

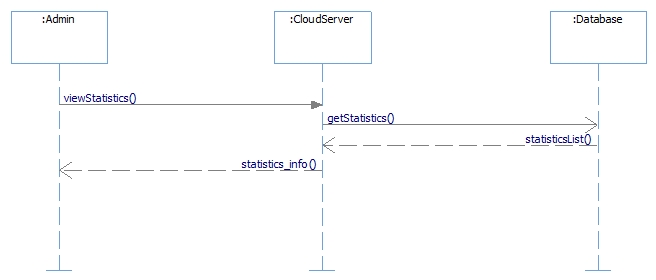
**Figure 34. ADMIN: CREATE ACCOUNTS**

****

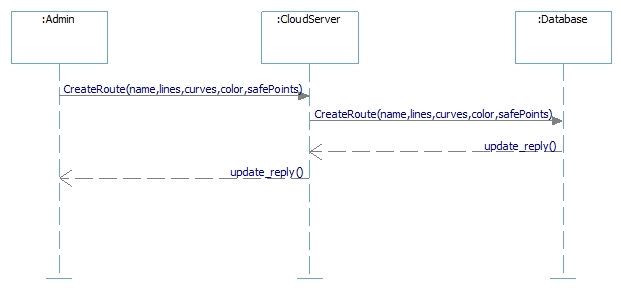
**Figure 35. ADMIN: MODIFY ACCOUNTS**

****

**Figure 36. ADMIN: DELETE ACCOUNTS**

****

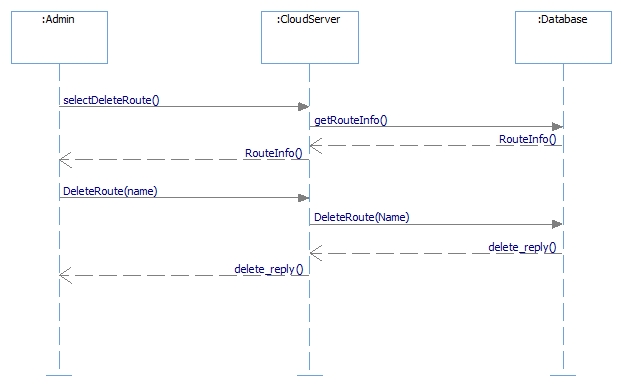
**Figure 37. ADMIN: VIEW STATISTICS**

****

**Figure 38. ADMIN: CREATE ROUTES**

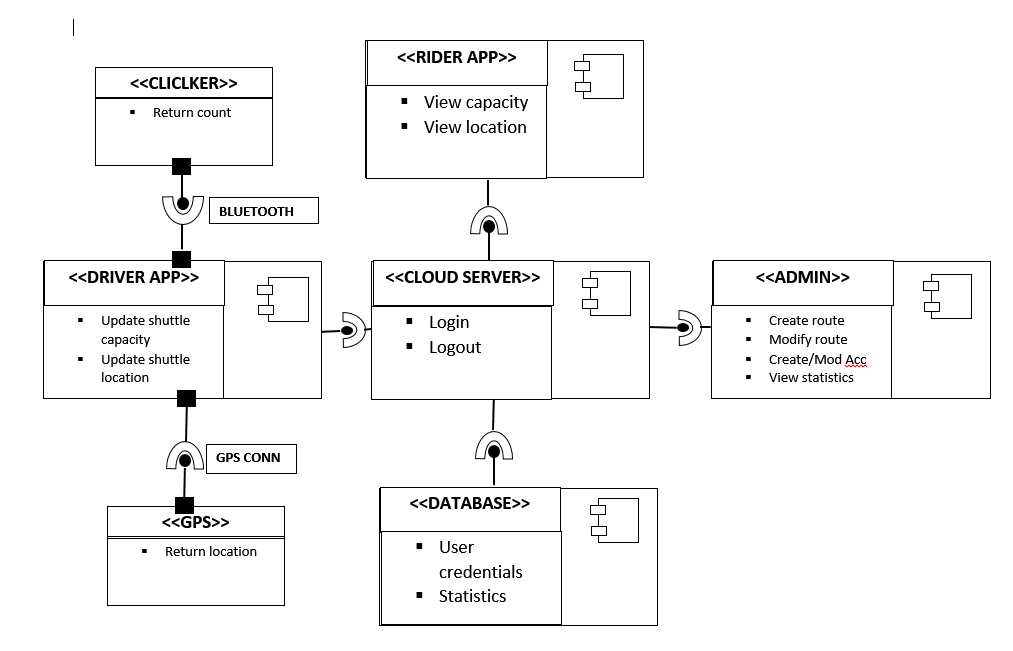
****

**Figure 39. ADMIN: MODIFY ROUTES**

****

**Figure 40. ADMIN: DELETE ROUTES**

**17. Architecture with Operations(from White Box)**



**Figure 41. Operations**

****

**Figure 42. Class Diagram**

****

**KNOWLEDGE TRANSFER DOCUMENT**

**Implementation**

**THE UNIVERSITY OF TEXAS DALLAS**

**1.Implementation Details**

**ADMIN PAGE**

Front end: JavaScript, HTML5 and CSS

Back end: PHP coupled with MySQL hosted on Google Cloud server

Statistics: Info from driver app and JavaScript calls to draw them

Login: Info hashed using MD5

Create routes: Google Map API plug in

**RIDER APPLICATION**

Front end: JavaScript, HTML 5 and CSS

Back end: The same as admin but all calls are made from an Android and IOS web view

**DRIVER APPLICATION**

Front end: Java and JavaScript. HTML5 and CSS for UI styling

Back end: Event handing in Java

The above two modules will call the JDBC server which in turns stores data in MySQL. All this is done through sockets.

Login module: Java and HTML5 and CSS as styling

****

**THE UNIVERSITY OF TEXAS DALLAS**

**CLOSE OUT REPORT**

**1.General Information**

Project Title & Working Title: CometRide – Smart Campus Initiative

Department: Software Engineering, The University of Texas at Dallas

Prepared by: Graduate Students, Software Engineering, The University of Texas at Dallas

Teams members: Danny Sundaresan, Scott Crain, Trevor Holder, Vaibhav Prakash

Date: April 15th, 2015

**2.Project Deliverables**

|  |  |  |
| --- | --- | --- |
| ***Deliverable*** | ***Date Accepted*** | ***Contingencies or Conditions*** |
| EXECUTIVE SUMMARY | January 23rd, 2015 | None |
| PROJECT PLAN | January 30th, 2015 | None |
| FEASABILITY ANALYSIS | February 13th, 2015 | None |
| VISION DOCUMENT | February 20th, 2015 | None |
| TRADE OFF | February 27th, 2015 | None |
| REQUIREMENTS & DESIGN DOCUMENT | March 13th, 2015 | None |

**3.Performance Goals**

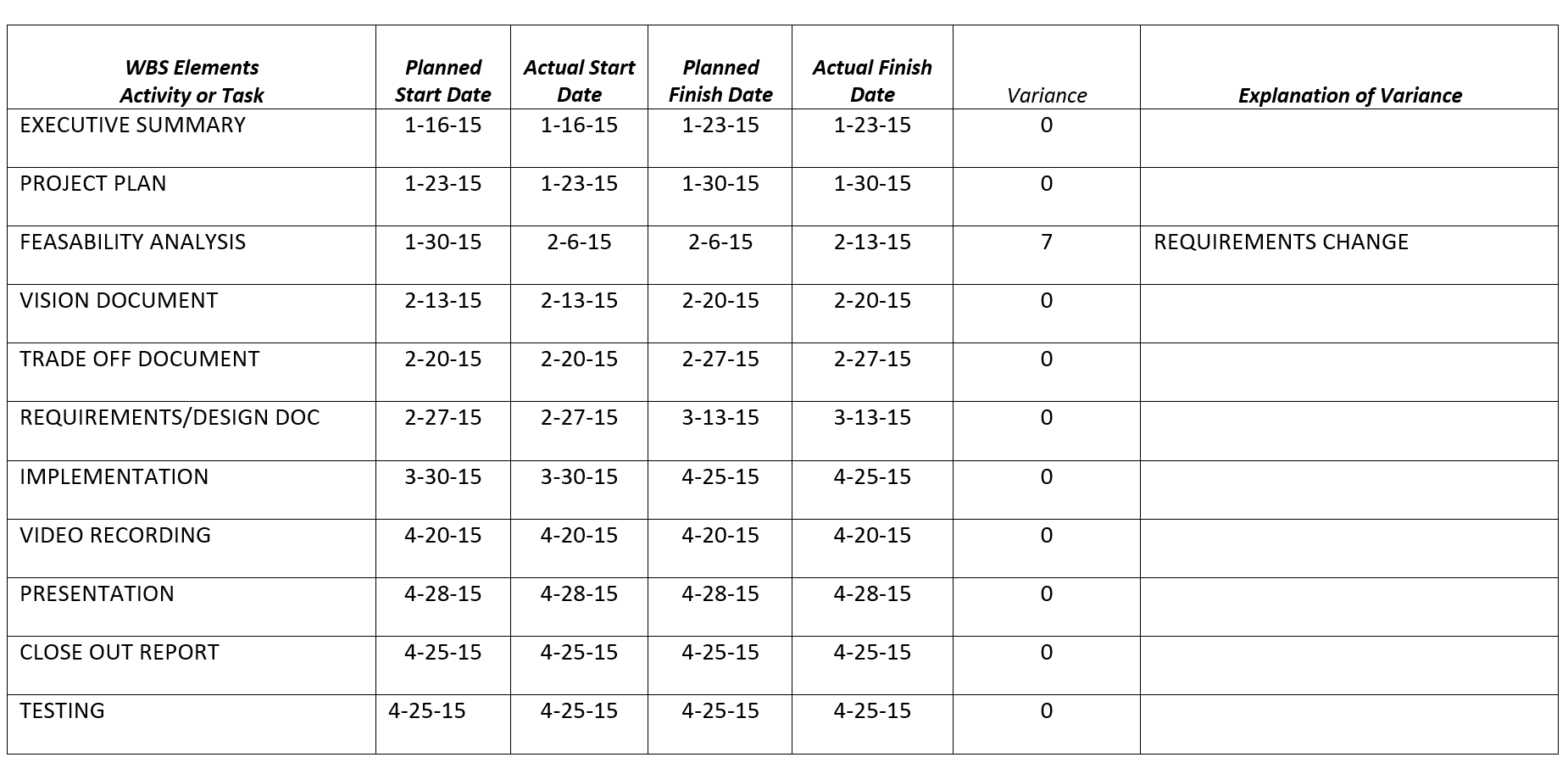
|  |  |  |
| --- | --- | --- |
| Project Business Objective | Project Goal | Status |
| Objective 1, Project Plan | Portability | Completed |
| Objective 2, Project Plan | Ease of access | Completed |
| Objective 3, Project Plan | User independency | Completed |
| Objective 4, Project Plan | Backward compatibility | Completed |
| Objective 5, Project Plan | Reliability | Completed |

**5.Cost Baseline**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Expenditures** ($000) | | | | |
|  | ***Planned*** | ***Actual*** | ***Variance*** | ***Explanation*** |
| **Internal Staff Labor** | 0 | 0 | 0 | 0 |
| *Services* | 0 | 0 | 0 | 0 |
| *Software Tools* | 0 | 0 | 0 | 0 |
| *Hardware* | 30 | 30 | 0 | 0 |
| *Materials and Supplies* | 0 | 0 | 0 | 0 |
| *Facilities* | 0 | 0 | 0 | 0 |
| *Telecommunications* | 0 | 0 | 0 | 0 |
| *Training* | 0 | 0 | 0 | 0 |
| *Contingency (Risk)* | 0 | 0 | 0 | 0 |
| *Total* | 30 | 30 | 0 | 0 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Funding Source** ($000) | | | | |
|  | ***Planned*** | ***Actual*** | ***Variance*** | ***Explanation*** |
| General Fund | 30 | 30 | 0 | 0 |
| *Non-General Fund* | 0 | 0 | 0 | 0 |
| *Federal* | 0 | 0 | 0 | 0 |
| **Other** | 0 | 0 | 0 | 0 |
| *Total* | 30 | 30 | 0 | 0 |

**6.Schedule Baseline**



**7.Scope**

|  |  |
| --- | --- |
| ***Scope Change*** | ***Impact of Scope Change*** |
| During the first week of February, there was a requirements change from the customer. The customer did not want the drivers of the shuttle to interact with any electronic device while the cab is in motion. This resulted in a delay of 2 weeks. | We had to re-design our use case which eventually resulted in res-design the entire design from the Driver’s perspective. Impact of change was 20%. |

**9.Project Documentation**

|  |  |  |  |
| --- | --- | --- | --- |
| *Report(s) and Document(s)* | *Media Used* | *Storage Location* | *Disposition* |
| EXECUTIVE SUMMARY | Microsoft Word | Google drive | NIL |
| PROJECT PLAN | Microsoft Word | Google drive | NIL |
| FEASABILITY ANALYSIS | Microsoft Word | Google drive | NIL |
| VISION DOCUMENT | Microsoft Word | Google drive | NIL |
| TRADE OFF DOCUMENT | Microsoft Word | Google drive | NIL |
| REQUIREMENTS/DESIGN DOC | Microsoft Word | Google drive | NIL |
| PRESENTATION | Microsoft Power point | Google drive | NIL |

**10.Lessons Learnt**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Statement of Problem*** | ***Discussion*** | ***References*** | ***Corrective Actions*** |
| Handling requirements change | The transportation department did not want the driver to interact with any device while operating the cart | Prof. Wenkstern’s guidance and UTD department of transportation | Re-defining requirements document |
| Importance of traceability in Software Engineering | To make sure the implemented software adheres to the requirements defined in the requirements phase | Prof. Wenkstern’s template | Improving each document in each phase based on the feedback from Mohammad Al Zinati |
| Transition and intersection of System and Software Engineering | Crucial in developing and deploying software systems which have to co-ordinate with respective hardware components | Prof. Wenkstern’s template | Improving the design based on Prof. Wenkstern’s advice |
| End to End implementation of a simulated industry software project | Essential in knowing the intricacies of deploying a software system in the real world | Prof. Wenkstern’s templates and guidance | Constant feedback from the Professor and Mohammad helped us to learn the true principles of Software Engineering |

**11.Post Implementation Review**

*Identify the date for completing the post implementation report and the person responsible for this action.*

|  |  |  |
| --- | --- | --- |
| ***Action*** | ***Date*** | ***Responsible Person*** |
| Post - Implementation Review | April 28th, 2015 | Entire team |
| *Post - Implementation Report* | April 28th, 2015 | Entire team |

**12.Project Approval**

|  |  |  |
| --- | --- | --- |
| ***Position/Title*** | ***Signature/Printed Name/Title*** | ***Date*** |
| Project Manager |  |  |
| Project Sponsor |  |  |
| Program/Agency Management |  |  |

****

**THE UNIVERSITY OF TEXAS**

**APPENDIX & REFERENCES**

**APPENDIX**

1. UTD – The University of Texas at Dallas
2. IBM – International Business Machines
3. GPS – Global Positioning System
4. LTE – Long Term Evolution

**REFERENCES**

[1]Wensktern, R. Z. , Advanced Software Engineering Project, eLearning, Spring 2015, The University of Texas at Dallas. All document templates have been adopted from course portal on e-Learning.