Specifications and Design

# Requirement Specifications

## Functional requirements

1. Real Time Bus Tracking
   1. The ability to request the current location of a bus and report its current location. This is a high priority feature as it is at the core of the app’s purpose.
   2. The bus location will be requested when the user selects a route. This will make a request to the Firebase database and retrieve the last known location of the bus.
   3. Req-1: Bus must be logged in as a driver to the app

Req-2: App’s Android device must have GPS/Location services enabled.

Req-3: App must push location to Firebase server at a TBD fixed interval.

Req-4: App must show notification if the last known location is older than a TBD time.

1. Driver Registration and Login
   1. The ability to register a user as a driver and login to the app to begin location tracking. This is a high priority feature and mandatory for the app’s core functionality.
   2. A simple web form will be used to register a user as a driver, being finalized after the driver and information are verified on the server’s side. The app will contain a driver’s login button on the main screen.
   3. Req-1: Driver must be on a pre-validated list of recognized drivers for an organization

Req-2: Driver must register their current bus/vehicle, and specify a change of current bus/vehicle on login.

Req-3: Driver must be able sign off/ log out of the app so that server or Firebase is informed about the status of the bus, whether it is being operated or not.

1. Display Bus Routes and Stops on Google Maps
   1. The ability to display a route and stops using the Google Maps API to draw a line between stops along the bus route and an image marking a specific stop. This is a high priority feature.
   2. The user will be provided a list of routes to choose from in a basic ListView. When selecting a route, the user will be sent to a Google Maps screen with the route drawn and the stops marked.
   3. Req-1: Bus stops geolocations must be recorded. Route locations must be recorded as well.

Req-2: Show bus locations on map, if any buses are currently running.

1. A search box will be provided to the user.
   1. Where the destination point is given, based on the available routes that go to the given destination point, list of all possible routes are shown.
   2. If no route is found for the given destination, user is prompted to enter more specific location that are operated by routes.
   3. If multiple destinations are found, user is asked again to select one from them.
2. A highlighted route and navigation view to the user from the current location to the nearest bus stop.
   1. As he selects the destination, he will be informed about the route and bus that goes to that destination, in addition to that, we highlight route and navigate him to the nearest available bus stop so that he doesn’t have to look for bus stop on his own.
3. A static map view of all routes are displayed.
   1. On selecting route view of a specific bus, a map with highlighted route and marked bus stops are shown on the map that doesn’t change, this map is same and constant as long as the bus route doesn’t change.
4. A kind of notification is provided, if the bus in specific route is not operating because of some reason, like, bus has got some technical issues, route is closed, special day so its being re-routed, all such things are notified to the user in the form of notifications.

## Non-functional requirements

1. Validated driver’s list
   1. High priority non-functional requirement. To ensure that only valid drivers are logging in to the app, their initial registration must be verified against a list of employed drivers. This list should be sent by the organization employing the drivers. They will be verified by submitting their name, birthdate, and driver’s license state and number during the registration process.
2. Encrypted user data
   1. Driver login information should be encrypted in the database. Specifically, passwords, but preferably other information such as driver’s license numbers, birthdates, and other personal information
3. Communication with Firebase app
   1. Each app on each device must be able to communicate with the Firebase database by sending JSON requests and responses to and from their Android device. In case of a failure to communicate with the Firebase server, a message should be displayed to the user that they are offline and that the data may be out of date.
4. Firebase, user’s app, and driver’s app all must be synced with in no time, only then real time GPS data is shared and shown on the map without any hassle or delay.
5. JSON structure on the firebase has to be declared such a way that even xml reading is possible; this is for platforms where JSON format is not supported.
6. GPS Location
   1. GPS location given by firebase is in the form of a string, there are some Google Map API versions where the string format is not supported, so there must be mechanism where string based location can be inter converted to number and vice versa.
7. Drivers and users App mode of working
   1. Both of the apps must also be working as an background application, because users may open it and use another apps of their specific choice, so map and navigation activities must be on hold or kept as background apps.

## Interfaces

1. There are 2 interfaces, between 3 software components, involved in our project.

a. Interface between user’s app and firebase.

b. Interface between Firebase and driver’s app.

**a. Interface between user’s ap and firebase**

Interface Data: It is a 2-way communication between systems. One way is app requests for updated location of the bus on the specific route, other is the Firebase app pushes the latest GPS location data to the user’s app.

It also get the static route and bus stop location from firebase app.

Data Format is in the form of below:

**+<Routes>:**

**---------- +<Route>:**

**---------- +<ID>:**

**---------- -<Lat: Long>:**

**---------- +<Route>:**

**---------- +<ID>:**

**---------- +<Lat: Long>:**

Interface Methods: Authenticate is one method that does authentication, **getUpdatedLocation** or **pushUpdatedLocation** are the methods that are used as interface methods for dynamic location, **getRoutes** and **getLocations** are the methods used for retrieving static data.

**a. Interface between Driver’s app and firebase**

Interface Data: It is a 1-way communication from driver’s app to Firebase, Driver’s app always pushes data to Firebase.

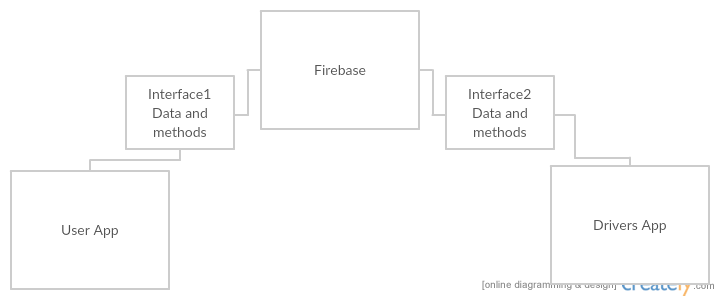
Data format is the form of below:

**+<RouteID>:**

**---------- +<BusID>:**

**---------- +<Status>:**

**---------- +<Lat>:  
 ---------- +<Long>:**

****

# Architecture of the system

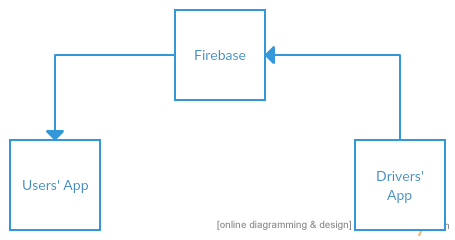
We use Client-Server architecture model for our project, where

Firebase: acts as server.

Driver’s and user’s app: acts as clients.

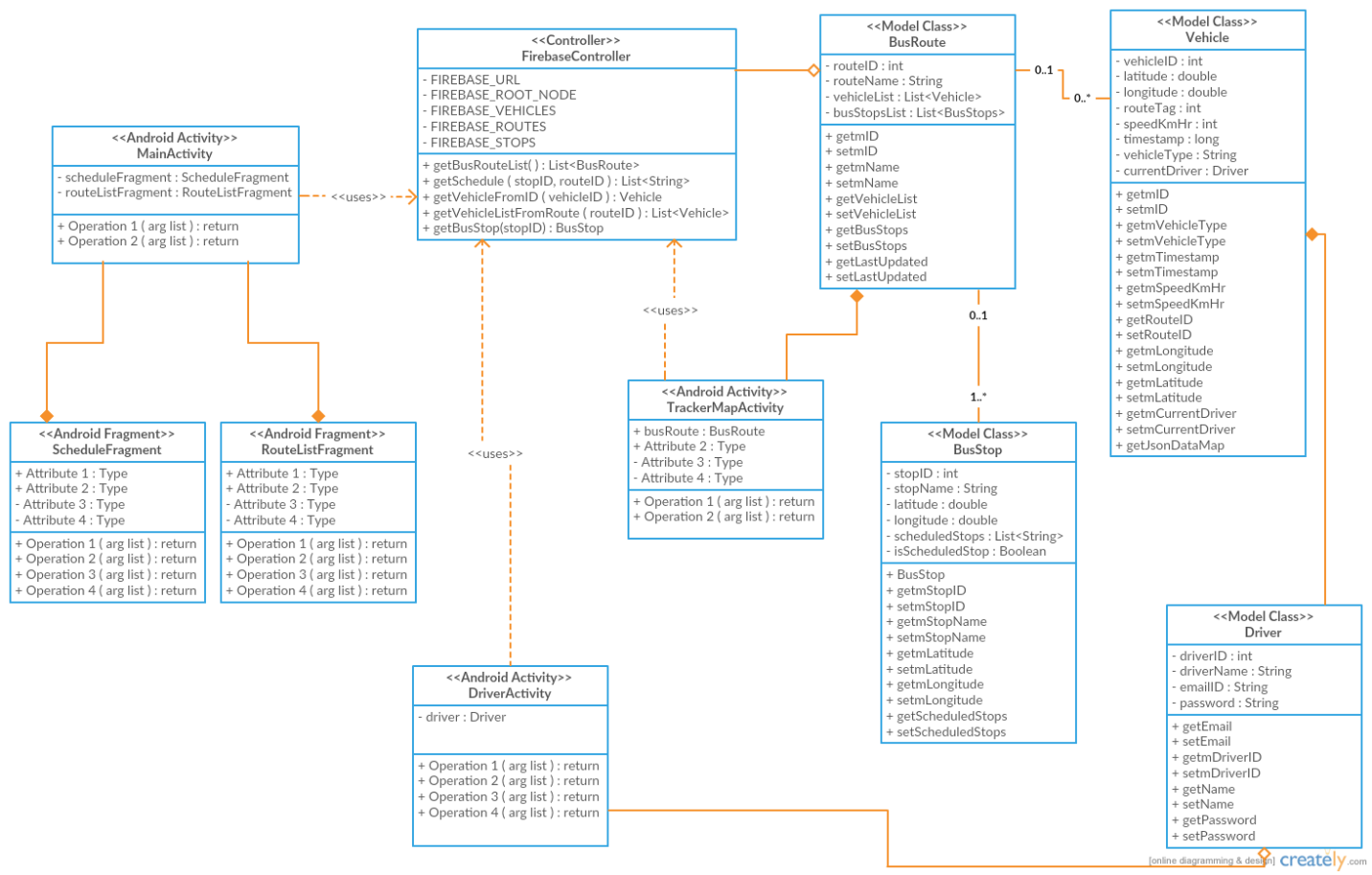
User’s app which is a client, receives data from firebase as response to firebase’s data push from driver’s application.

Driver’s app which is a client, pushes data to firebase as a result, firebase shares that to user’s application.

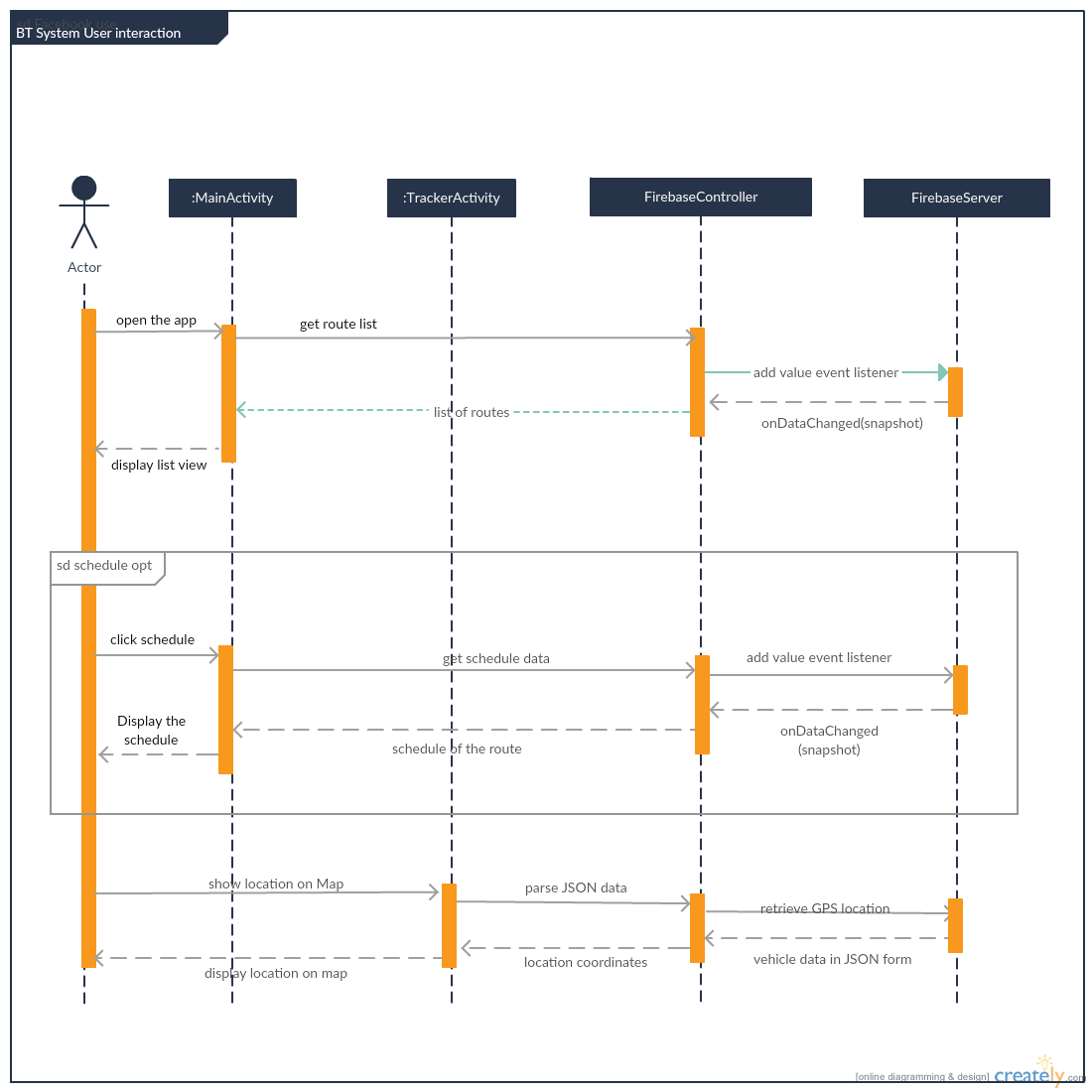


# UML Design Documentation

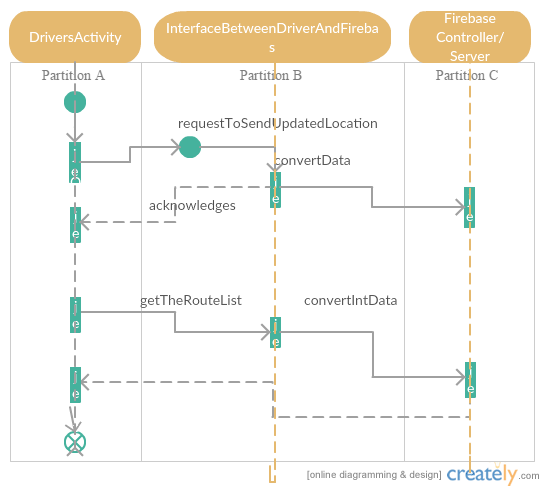
## Class diagram



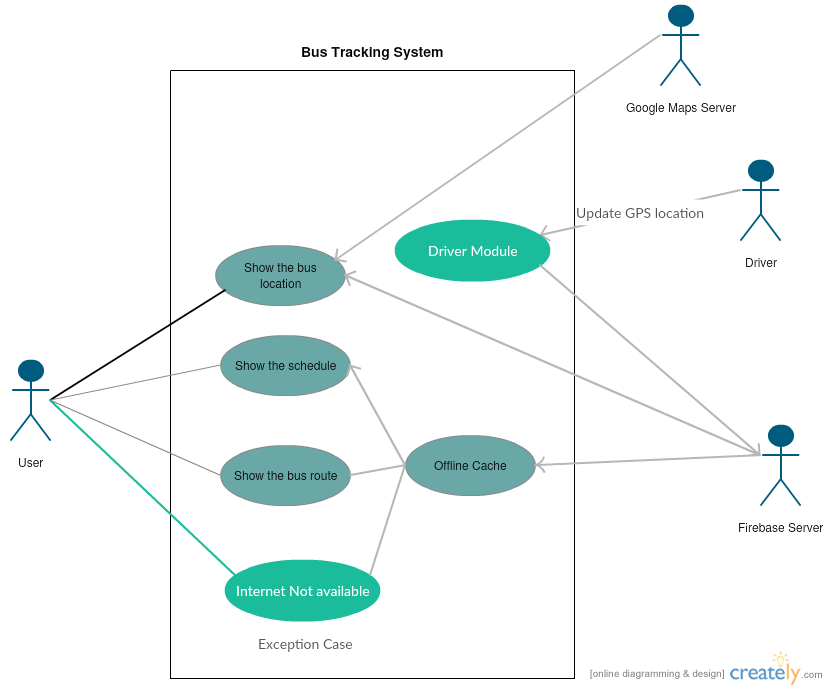
## Sequence diagram



## Sequence diagram of Drivers App



## Use case diagram



# Test Plan

# Updated risk management

The risks addressed in the early stages of the project can be summarized as follows:

1. Inexperience with Android development
2. Inexperience with Firebase database, as well as fewer easily found resources for issues due to Firebase being a relatively new technology.
3. Requires commitment from UNT and their transport department.
4. Potential for limited user base at UNT
5. Time constraints to fit in all planned functionality

The above risks were discussed and monitored through each meeting. For the risk of selecting a newer technology as Firebase, the risk has been mostly avoided to date. A proof of concept for the Firebase and Android integration was created and showed the simplicity of working with Firebase. Additional discussions amongst the entire group have led to the implementation of a database structure that Firebase can easily follow. For the remainder of the project, this risk will still be monitored as further integration could lead to issues, but it will be downgraded. While initially 2 group members would focus on Firebase, the work will instead be spread out amongst all the group members as needed. In case a problem will arise, the Firebase risk will receive more focused attention from multiple group members.

The issues of inexperience with Android have also been mitigated so far. The group split up the work to have the more experienced Android programmers work on more UI elements and integration with Android specific libraries. The other group members received work that was more independent of Android, focusing only on general Java libraries and code. If they run into Android-specific problems, that part of the program can be reassigned to other members. The current plans for the UI also allow for a simple design, so issues with Android UI elements can be solved by simplifying the design. The “flashier” UI elements can be pushed back to a future release.

For risks 3 and 4, the group discussed a change to the initial design plans. The Firebase database along with the main Java classes were made more generic to apply to a large range of vehicle transports. The app’s success would not just rely on one user base in one limited area. Instead, the app is designed for use by any transportation department. With some additional future functionality, it would also be able to be generalized to all vehicles rather than just buses. As a result, the risk shifts away from a limited group of willing drivers and users more towards being able to manage a larger, potentially nationwide population of users.

Risk 5 remains as the most significant risk to the group. The project requires a lot of initial planning and as a result, the actual code production has somewhat stalled. Proof of concepts and prototype code have been created that should allow for faster development, and that is the current plan the group is relying on. In case of a bigger time crunch, the group has decided that it may remove some desired features or limit the data gathered to complete the project on schedule.

# Updated project plan

# Meeting minutes

'Android Developers' meeting minutes

|  |  |
| --- | --- |
| Location: | Discovery Park NTDC B157 |
| Date: | 09/22/2016 |
| Time: | 10:00 a.m. |
| Author: | Gil Wasserman |
| Attendees: | Dr. Hyunsook Do, Anurag Chitnis, Nitesh Kumar Sharma, Satyanarayana Chivukula, Gil Wasserman |

### Agenda items

1. Requirement Elicitation
2. Project status and feasibility

### Discussion:

* Discussed with customers from other groups about the functionality of the project. Some parts of the project’s desired functionality were clarified for the customers i.e. how the user interface will be designed, what it is intended to do, etc. Received input from customers about desired functionality like showing multiple routes that go to one stop, having the ability to determine which route gets the user to their location fastest, and having static maps of each route easily accessible for the user.
* Discussed these desired requirements with group members, determined that static maps and showing multiple routes should be feasible. Designing an algorithm to determine which bus out of multiple routes will get to a location fastest was a feature that was agreed on would be a great future feature to add, but not realistic in the current timeline of the project.
* Started discussion on deliverable 2 requirements

| Sr. No. | Action items | Owner(s) | Deadline | Status |
| --- | --- | --- | --- | --- |
| 1 | Push the POC of the Firebase integration to the repository | Anurag | 09/22/2016 | Completed |
| 2 | Push the POC of the Google Maps integration to the repository | Gil | 09/22/2016 | Completed |
| 3 | Create UML design documents, requirement specification report, updated risk management and project plan, and test plan for deliverable document | Satyanarayana | 10/11/2016 | In Progress |
| 4 | Create UML design documents, requirement specification report, updated risk management and project plan, and test plan for deliverable document | Nitesh | 10/11/2016 | In Progress |
| 5 | Create UML design documents, requirement specification report, updated risk management and project plan, and test plan for deliverable document | Gil | 10/11/2016 | In Progress |
| 6 | Create UML design documents, requirement specification report, updated risk management and project plan, and test plan for deliverable document | Anurag | 10/11/2016 | In Progress |

'Android Developers' meeting minutes

|  |  |
| --- | --- |
| Location: | Discovery Park NTDC B157 |
| Date: | 10/06/2016 |
| Time: | 10:40 a.m. |
| Author: | Gil Wasserman |
| Attendees: | Dr. Hyunsook Do, Anurag Chitnis, Nitesh Kumar Sharma, Satyanarayana Chivukula, Gil Wasserman |

### Agenda items

1. Class Diagram
2. Assign code responsibilities
3. Deliverable 2 Progress

### Discussion:

* Went over the class diagram created by Anurag. Discussed the different major parts of the project in the diagram and the functionality each class would provide to the overall project.
* Discussed the Firebase database structure and how it relates to the class diagram, specifically the main root nodes needed in the database and how some data would be duplicated for easier access in the app
* Split up the work on the project, assigning Gil the ScheduleFragment and RouteListFragment, Anurag the TrackerMapActivity, Satyanarayana the DriverActivity, and Nitesh the FireBaseController.
* Continued discussion on deliverable 2 and finalizing responsibilities for the submission. Gil will do the updated risk management and will also work on the requirements specification report.

| Sr. No. | Action items | Owner(s) | Deadline | Status |
| --- | --- | --- | --- | --- |
| 1 | Create UML design documents, requirement specification report for deliverable document | Satyanarayana | 10/11/2016 | In Progress |
| 2 | Create requirement specification report, and test plan for deliverable document | Nitesh | 10/11/2016 | In Progress |
| 3 | Create requirement specification report, updated risk management for deliverable document | Gil | 10/11/2016 | In Progress |
| 4 | Create UML design documents and project plan for deliverable document | Anurag | 10/11/2016 | In Progress |
| 5 | Push draft of ScheduleFragment and RouteListFragment in preparation of Peer Review | Gil | 10/13/2016 | In Progress |
| 6 | Push draft of TrackerMapActivity in preparation of Peer Review | Anurag | 10/13/2016 | In Progress |
| 7 | Push draft of DriverActivity in preparation of Peer Review | Satya | 10/13/2016 | In Progress |
| 8 | Push draft of FireBaseController in preparation of Peer Review | Nitesh | 10/13/2016 | In Progress |



# Progress report

We have finished a basic proof of concept in which a user could checkout a particular location on google map and save it to the Firebase real-time database. If the same application is installed on more than one phone, multiple people will be able to fetch the checked out location at the same time. In this activity we performed the feasibility test of the overall project.

Second proof of concept we performed is to show the markers on the particular GPS co-ordinates. We learned how to show the markers on google map, which will further help in plotting the bus stops.

# Member contribution

|  |  |  |  |
| --- | --- | --- | --- |
| Member name | Contribution description | Overall Contribution (%) | Note  (if applicable) |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |