Smart City Bus

Project Workbook

By

Venkata Adithya Boppana
Varsha Kodali
Sashank Malladi
Ravali Peddi

11/14/2016

Advisor: Frank Lin

ABSTRACT

Public Transportation is a necessity for many people who cannot drive or who cannot afford private vehicles. According to APTA (American Public Transport Association), the number of passengers relying on public transit has increased by 39 percent since 1995. Also, in other countries like India and China, public transport serves millions of people. Currently, there are more than 6400 providers of public transport in the USA alone. These transit agencies maintain dedicated scheduled services to help passengers in planning their trips.

Currently, transit agencies provide schedule information of their services in various formats. But, the information provided is not accurate as services may not adhere to these static schedules. Schedules are impacted by various factors like traffic congestion, average speed, change in service route etc. Accordingly, there is a need for a solution which provides more reliable information to passengers.

In this project, we propose a solution for transit agencies to maintain more reliable schedule by dynamically tracking their vehicles. Latest Information Technologies involving web applications are used to alert passengers about the exact location of the vehicle and its estimated arrival time. Vehicles are tracked using GPS (Global Positioning System) and estimated arrival time for each stop in a route is calculated considering the impacting factors. As a result, public transit services can be made more reliable means of transport for many people.

Chapter 1. Literature Search, State of the Art

Literature Search

Public transit services are being increasingly used by people who cannot drive or who cannot afford private vehicles. Most of these transit services provide their schedule freely available on web in various formats. However, public transport is not always reliable due to lack of real time transit information about bus locations and operational delays [2]. Only few transit agencies provide advanced services such as real-time tracking and arrival time prediction. Real-time tracking is a key feature for improving transit user experience by minimizing the waiting time at stops [4]. It also helps in motivating people to prefer public transport over other means of transport which in turn reduces traffic congestion, its environmental impact and also helps in improving a nation's economy. Location-based services (LBS) can be developed to make it feasible for all transit operators to provide enhanced services like real-time tracking and arrival time prediction. Location-based service is an information service, accessible with mobile devices through the network and has the ability to make use of the geographical location of the device. These services provide real time transit navigation information by tracking public transport vehicles. Real-time transit navigation information improves usability of public transit by making it more reliable means of transport.

Transit agencies use Automatic vehicle location (AVL) systems to monitor their vehicles. Dedicated GPS tracking devices are installed on the vehicle to acquire the GPS coordinates of its location. Some systems also use traditional smart phones which are equipped with suitable sensors to obtain location data. Location-based service on a GPS enabled smart phone can be used as AVL system to provide real-time tracking. Following factors have to be considered while obtaining accurate location of a mobile device.

1. Multiple location sources – Location can be acquired from different providers (GPS, WI-FI, Network Provider) which vary in accuracy, speed and power consumption.
2. User movement – Based on user movement, location data has to be refreshed at regular intervals.

Arrival time of a bus depends on various factors like traffic conditions, number of stops in a route and other incidents. Considering all these factors, vehicle arrival times can be estimated by developing dynamic modules by collecting data from the AVL system [3]. Several prediction algorithms are used to estimate the arrival time under homogenous traffic conditions [4]. Historic and real-time approaches like machine learning techniques (artificial neural network, support vector machines), model based approaches (Kalman filtering) and statistical methods (regression analysis, time-series) are used for estimating bus arrival time [4]. Existing prediction algorithms do not consider heterogeneous traffic conditions. Historical methods estimate the arrival time by calculating the mean of travel time over different days for same period. In real-time approach, arrival time for next time interval is predicted to be same as that of the present interval [3]. Bus location data is obtained from the AVL system, based on the schedule information, the difference between scheduled and actual arrival times is calculated.

There are different predefined formats for datasets of schedule information for transit agencies. GTFS is a General Transit Feed Specification which presents a common format for public transportation schedules. Transit agencies submit their transit data in a format specified by GTFS. Developers write applications which consume this transit data provided by the agencies for navigational purposes. Most of the current transit tracking applications are compatible with the GTFS feeds. These GTFS data feeds are also updated by the transit operators to get the most recent updated schedules. GTFS data feeds are static, cannot be impacted by the operational delays of vehicles [5]. Following datasets are used by conventional bus tracking systems [2].

1. Static dataset: Static dataset is static schedules containing routes, stops, trips provided in a specified format like GTFS.
2. Time point dataset: Time point data set is historical data of arrival time of vehicles including stop ID, route ID, bus ID.
3. Real-time transit feeds: Real-time transit data is the data collected from an AVL system.

These datasets are used for estimating the arrival time of a bus using different prediction algorithms. Location-based application on smart phone runs as a background process to send location updates continuously at regular intervals. GPS coordinates are transmitted to a central server through GPRS.

After thorough research on existing arrival time prediction algorithms, real-time approach is chosen for estimating accurate arrival time. Using real-time approach, dynamic traffic information and vehicle position can also be considered for calculating arrival time accurately.

State of the Art

While different bus tracking systems already exist, there usage is restricted to due to lack of real-time transit information. In this our project, our main focus is to make it feasible for smaller transit agencies to provide advanced services like real-time tracking and arrival time prediction.

Using our application, transit agencies can enhance user experience by simply installing our tracking application on smart phone placed in a bus. Our project has considerable advantages over the current state of the art. Firstly, use of a GPS enabled smart phone avoids the cost of a dedicated GPS tracking device. Tracking application can be easily installed on Android platform. Secondly, our application integrates both homogenous and heterogeneous traffic conditions and also dynamic location of vehicles. Efficient prediction algorithms are used to estimate arrival time considering operational delays. Current state of the art does not use GPS to obtain current location of the vehicles. Static schedules provided by transit operators are considered for estimating arrival time to upcoming stops in a route. Existing applications do not provide a way for transit operators to update their schedules. Our application provides a web portal for transit agencies to update their schedules. Also, historical data of arrival time of vehicles at each stop in a route is provided to help agencies monitor their vehicles.

References

[1] J. Biagioni, T. Gerlich, T. Merrifield and J. Eriksson, "EasyTracker: Automatic Transit Tracking, Mapping, and Arrival Time Prediction Using Smartphones*", in 9th ACM Conference on Embedded Networked Sensor Systems,* 2011, pp.68-81.

[2] F. Sun, Y. Pan, J. White and A. Dubey, "Real-time and Predictive Analytics for Smart Public Transportation Decision Support System", *in IEEE International Conference Smart Computing* (SMARTCOMP), 2016, pp. 1-8.

[3] J. Gong, M. Liu, and S. Zhang, "Hybrid dynamic prediction model of bus arrival time based on weighted of historical and real-time GPS Data", *in 25th Chinese Control and Decision Conference* (CCDC), 2013, pp. 972-976.

[4] R.P.S. Padmanaban, L. Vanajakshi, and S.C. Subramanian, "Estimation of bus travel time incorporating dwell time for APTS applications", *IEEE* Conference *Intelligent Vehicles Symposium,* 2009, pp. 955-959.

[5] P. Zhou, Y. Zheng and M. Li, “How long to wait?: predicting bus arrival time with mobile phone based participatory sensing” *10th international conference on Mobile systems, applications, and services*,2012, pp.379-392.

[6] C.-H. Wu, J.-M. Ho, and D.-T. Lee, “Travel-time prediction with support vector regression,” *Intelligent Transportation Systems, IEEE Transactions on*, vol. 5, 2004, pp. 276–281.

[7] D. Ingle, N. Navi, Mumbai, “Experimental Estimates of Low-Cost Bus Tracking System Using Area-Trace Algorithm”, *Fifth International Conference on Communication Systems and Network Technologies,* 2015, pp.525-529.

[8] C.I.J. Steven, Y. Ding and C.Wei, “ Dynamic bus arrival time prediction with artificial, Natural Networks”, *Journal of Transportation Engineering,* 2002, 128(5):429-438.

Chapter 2. Project Justification

In our project, Smart City Bus, our effort is to develop an application that provides more convenience to bus users with most accurate transit bus schedules as they accomplish their trips.

Many transit systems operating nowadays have online and printed schedules for their services. Generally, the schedules are not updated based on traffic conditions daily. In such cases, the passenger may have no idea about current location of the bus and if he/she has already missed the bus if it arrived five minutes early than the posted time on the schedule.

The proposed application will be useful for the passengers to know the exact arrival or departure times of desired bus at a bus stop. Thereby this eliminates an effort for the passengers to visit the service website for the scheduled times posted online by the transit agency. In addition, as the advertised schedules may not be accurate, it may result in long waiting time for the passengers at the bus stops. The passengers may feel unpleasant.

Prevalent use of smartphones and Global Positioning System (GPS) technology has made the proposed system possible. The bus can be tracked using GPS and the schedules are updated frequently in the database on a dedicated server based on the traffic conditions. Hence, with the availability of mobile network, it is possible for the passenger to get responses like most recent and accurate update about desired bus at desired bus stop from the server.

Chapter 3. Project Requirements

1. Introduction

The completion of project requires design and development of three different modules separately. Integrating all these three modules constitutes the entire project. The three modules are namely, bus-side application, server-side database, client-side application. Network communication enables the interaction between these modules.

Product Perspective

Smart City Bus is a bus tracking innovative application for passengers. The application makes use of a smartphone with GPS receiver capability mounted on the bus. Space-based satellite communicates with the GPS. A central dedicated server is designed with a database built into it and an algorithm is implemented to calculate the distance and Estimated Arrival Time (ETA). As soon as the GPS sends the latest information to the server, the algorithm is applied on the data, distance and ETA are logged accordingly. The user-side application allows the passengers to access the bus information like arrivals, departures, delays from the server and plan their trips.

User roles and Responsibilities

The major end users of the proposed system are people who use public transit bus services to commute from place to place within a city or town. Currently, a fixed route will be considered for developing the modules. The project can later be implemented with multiple routes.

Future enhancements to the project can add analyzing the data stored on the server-side and user queried data. This gives an opportunity for the management to improve the user experience by knowing which route bus is being mostly used by the passengers and make some critical decisions such as whether or not to start an additional service in that route etc. An additional feature to sign-up for receiving notifications about ETA can also be implemented. Other features like fare updates, seat availability in the bus. Hence, the application will be used by more number of end users significantly.

Constraints and Assumptions

1. Studies show that the GPS signal are inaccurate in the presence of high buildings and trees. So, the buses being operating in downtown may have less precision than expected.
2. Data stored on the server must be accessible simultaneously at all the times.
3. The GPS module should have internet connection all the time

Functional Requirements

The proposed application has three modules. The first is a Bus positioning module (Global Positioning System-GPS) for which we plan to use an Android running smartphone with a GPS module installed in it. Second is the server-side database and a processing module. The third one is a user application which is an android application.

For testing the application, after ensuring that individual applications are working properly, network communication is tested between the bus-side, server-side and client-side applications. First, hard coded location as latitude and longitude values are input in the bus side application and periodically send to the server to ensure that the data is received correctly, stored in the database and processed. The client-side application is tested by requesting the same information and ensuring that map is displayed.

Project Specifications

1. Place an android phone with GPS receiver capability on the bus during its operating hours
2. The GPS module receives the position as latitude and longitude of the object from space-based satellite navigation system.
3. The module will send the position information to a central server once in every twenty seconds.
4. The server-side database is updated with the latest location information
5. Distance and duration for each stop along the route from current location are computed and stored in the server database.
6. The client application has an interactive display to request the ETA, delay status etc.
7. The requested information is retrieved from the central server and displayed on the user application.

Non-functional Requirements

The user must have the application installed in their android phone. The device should be actively connected to the network.

Hardware Requirements

1. Android Phone with GPS receiver capability
2. A computing device with active internet to run the server application.
3. End user must have an android phone with active internet connection to access the application

Scope of the Project

<u>In Scope:</u>

1. Developing a server application that stores position details of the bus and calculates the distance, ETA and other parameters
2. The server application must demonstrate efficiency in speed and latency while handling multiple requests.
3. Multiple routes with multiple bus stops along the route will be used.
4. Native Android application for the end user.
5. Purpose of project is currently for coursework demonstration only.

<u>Out of Scope:</u>

1. Offline access to the application
2. End user application on other wearable devices with internet connection.
3. Automatic suggestions based on the search history of the user.
4. Automated notification service.
5. Different forms of viewing data (either map or line chart) on the user application
6. Voice interaction with the application.

Chapter 4. Dependencies and Deliverables

Dependencies

Data of routes and stops:

For delivering precise arrival time of the bus, we have to take into account not only its current location but also the route it takes and no of stops between the location of bus and user's location. Therefore, the exact routes and all the stops in a particular route is a necessary data for the project. This kind of information is only available from the operators of public transportation services. Luckily, VTA provides use with the required data through its open data portal. Also datasets can be obtained in general formats specified by GTFS (General Transit Feed Specification). GTFS provides a common format for transit data. It includes details about routes, stops, trips and other agency details.

(data of routes:https://data.vta.org/Transit-Operations/Routes-January-2015/5zq9-avwa)

(data of stops: https://data.vta.org/Transit-Operations/Stops-January-2015/iy3q-7kq5)

GPS enabled android phone:

A GPS enabled android phone which can accurately send out the GPS coordinates regularly. As the purpose of the project is to provide real-time information about the status of buses, it is imperative that we have real-time GPS data of the buses. We chose GPS enabled android phone over a small GPS device because of mainly two reasons. Firstly, they can connect to the internet and send the GPS coordinates to a server through an application. Secondly, ease of use and no need to have a fixed installation.

Fig.1 GPS coordinates are collected from android phone

Efficient and scalable algorithm for time estimation:

The essential component of the project is an algorithm which can estimate of arrival time of the bus. To achieve a reasonably accurate estimate of time we have to take into account the traffic conditions at a given time, route the bus is taking, no of stops in the route, amount of time bus stops at each stops, latency delays in computing. Also, the algorithm should be able to scale up or down with the changes in routes and stops.

Deliverables

The final deliverables are two android applications 1) android application for user 2) android application for bus GPS coordinates

1) Android application for user: This is the application which user will use to see the time of arrival of selected bus to user’s location. It will show the arrival time and the current location of the bus on a map.

2) Android application for bus: This application will be running on the android phone which will be located on the bus. It collects the GPS coordinates of the bus regularly. The collected data is sent to the server which will be hosted on the cloud.

3) Agency web portal: A web portal for transit agencies to update their schedules. It also provides historical data of bus arrival time at each stop along a route. This data helps agencies in monitoring their services.

Chapter 5. Project Architecture

Smart City Bus is an android application which provides passengers with most reliable bus schedules. Daily commuters can gain maximum benefit in planning their trips. As part of proposed solution, a user interface will show arrival time of busses at nearby bus stops.

Fig 2 depicts architecture of the project.

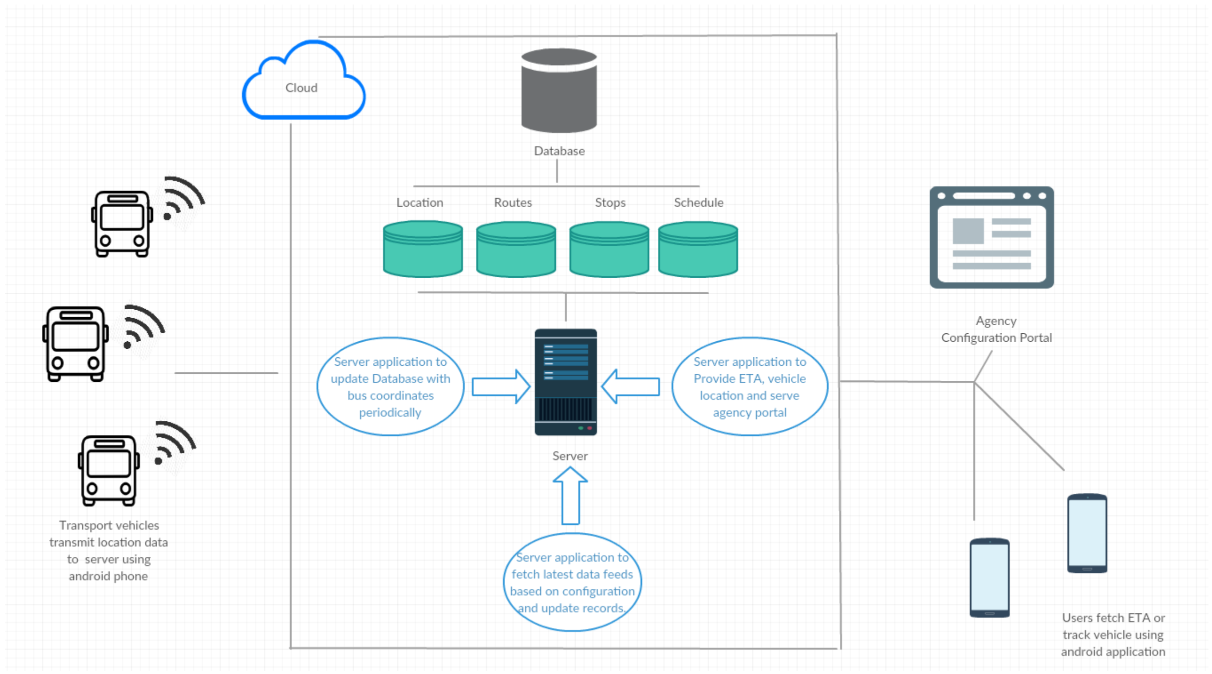

Cloud
Database
Location
Routes
Stops
Schedule
Agency Configuration Portal
Server application to update Database with bus coordinates periodically
Server application to Provide ETA, vehicle location and serve agency portal
Server
Server application to fetch latest data feeds based on configuration and update records.
Transport vehicles transmit location data to server using android phone
Users fetch ETA or track vehicle using android application

Fig.2 Architecture of Smart City Bus Application

Above architecture consists four important component implementations.

- Client Application to transmit live vehicle coordinates.
- Server applications to process data, calculate estimated arrival time, handle client requests.
- Client Application to end user providing most reliable transit schedule and features.
- Web portal access to Transit agencies to manage their schedule data.

Vehicle Location

Each vehicle of an agency will be installed with an Android phone and an application running on it. The app will be running in the background periodically sending its location to the server hosted on the cloud.

Server

Three server applications will be serving the users, vehicles and feed jobs. Node JS technology is used to build these servers. Dedicated server application are assigned to serve request from vehicles and users. Node JS being event driven, it can handle many concurrent request with low CPU resources.

Database

Each vehicle is required to be accessed in the form of object data for processing or responding to client requests at faster rates. This application being more data oriented because of its estimation arrival time calculation and relating real-time time data to static schedule, data should be preferably organized in objects for faster access and processing. Mongo dB being open source and more compatible with node technology, Data for this project is planned to be maintained in Mongo database.

End User Android Application

A Hybrid application built using ionic framework JS technology will serve users by providing live vehicle location and reliable estimated time of arrival. Considering Scalability and maintenance, this project is built as hybrid application instead of a native app.

Each client side application incorporates a MVC architecture with node as backend server and Mongo dB as database.

Chapter 6. Project Design

The following block figure describes the overall design of our project.

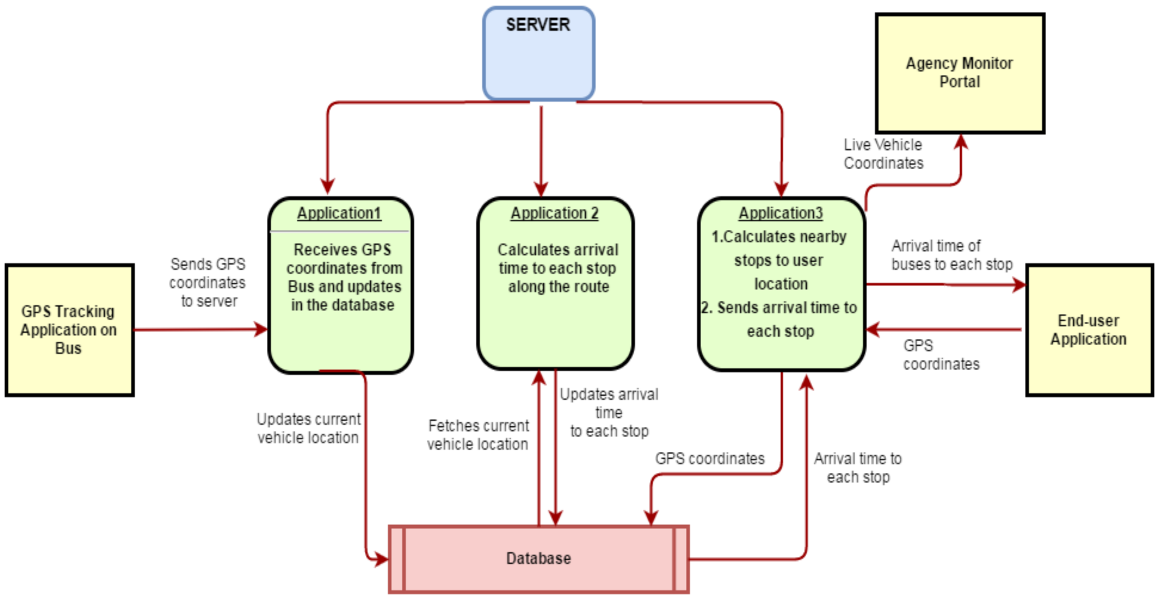

SERVER
Agency Monitor Portal
Live Vehicle Coordinates
Application1
Receives GPS coordinates from Bus and updates in the database
Application 2
Calculates arrival time to each stop along the route
Application3
1.Calculates nearby stops to user location
2. Sends arrival time to each stop
GPS Tracking Application on Bus
Sends GPS coordinates to server
Arrival time of buses to each stop
GPS coordinates
End-user Application
Updates current vehicle location
Fetches current vehicle location
Updates arrival time to each stop
GPS coordinates
Arrival time to each stop
Database

Fig.3 Overall System Block diagram

GPS services on a smartphone will be used to track transit vehicles. An application installed on the smartphone sends GPS coordinates to the server at regular intervals. Hybrid application will be developed using ionic framework for android platform and runs in the background.

The Server Application1 is designed to receive location updates from the bus-side application and update the location information of all the buses to all the stops in their respective routes.

The Server Application2 is dedicated to calculate the Estimated Arrival Time. It queries the database to get the latest GPS location of the buses. Then calculates arrival time of the bus to all the bus stops in its route. The new arrival times are updated to database which are used by Server Application 3.

The Server Application3 handles the user's requests. It receives the current location of the user. Based on the location, the Server Application2 computes the distance from user location to all the stops on the go and fetches the nearest stops. Then a sub-query is run based on the stop_id to fetch the list of buses and send the data to the user along with ETA of each bus to its stop.

End-user application provides real time transit information to the end user. As soon as a user opens the application, GPS coordinates will be sent to the server to retrieve transit information to nearby stops.

Transit agencies can monitor their vehicles on the web portal. Also, agencies can update their latest schedules using the web portal.

UML Diagram:

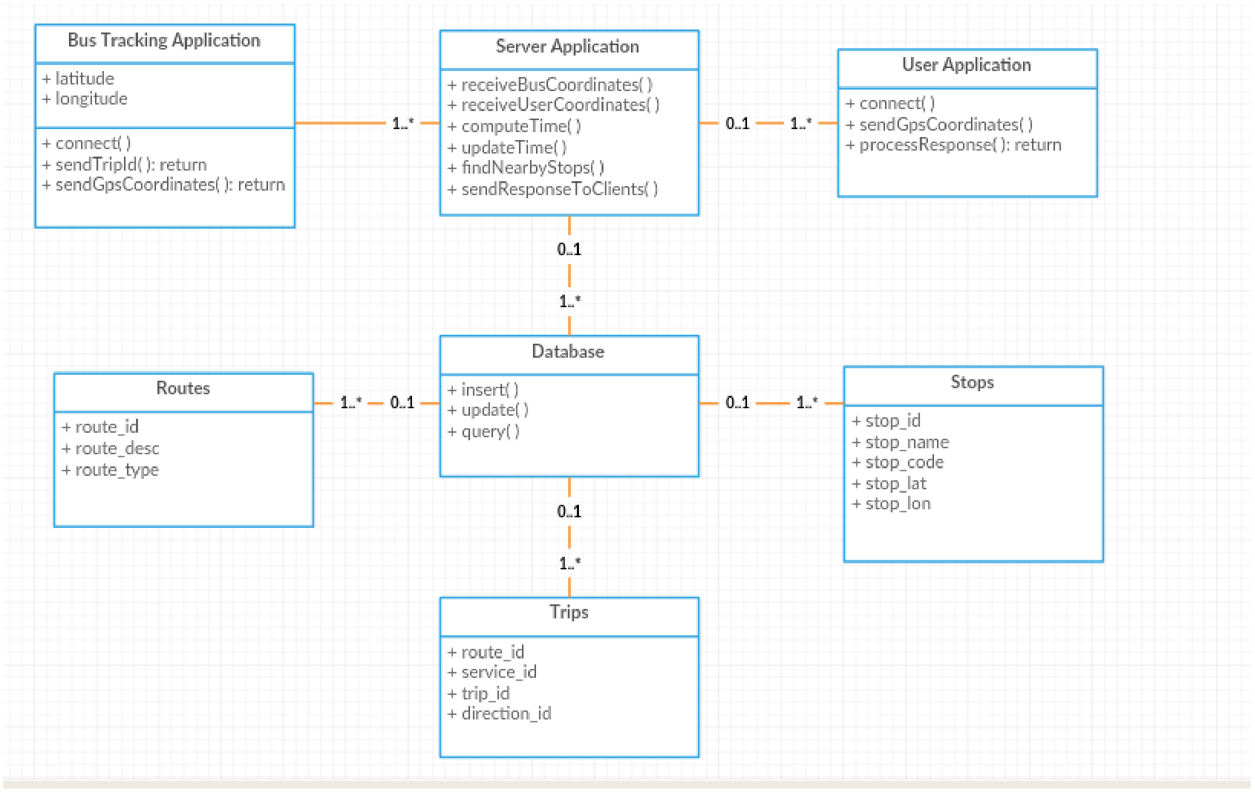
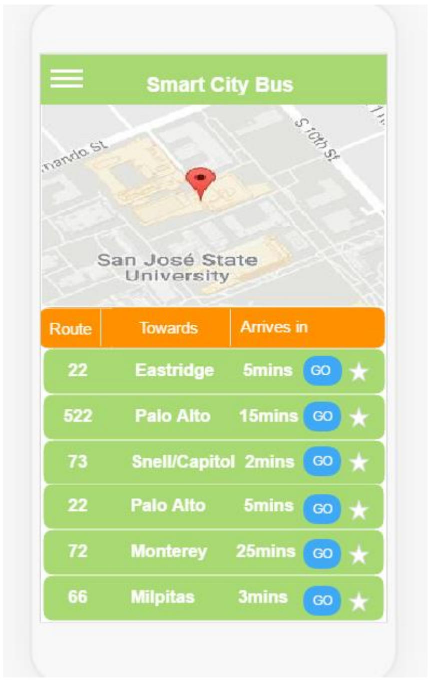


Fig.4 UML Diagram

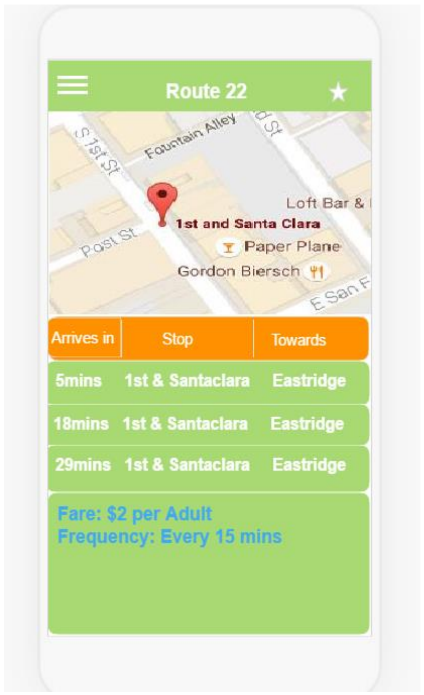
UI Mockups:

UI mockups are developed for both application GPS tracking application on the vehicle and the end-user application. Figures 1 and 2 are the mockup screens for the end-user application.



Route	Towards	Arrives in		
22	Eastridge	5mins	GO	★
522	Palo Alto	15mins	GO	★
73	Snell/Capitol	2mins	GO	★
22	Palo Alto	5mins	GO	★
72	Monterey	25mins	GO	★
66	Milpitas	3mins	GO	★

Fig.5 Screen1 when user opens the application

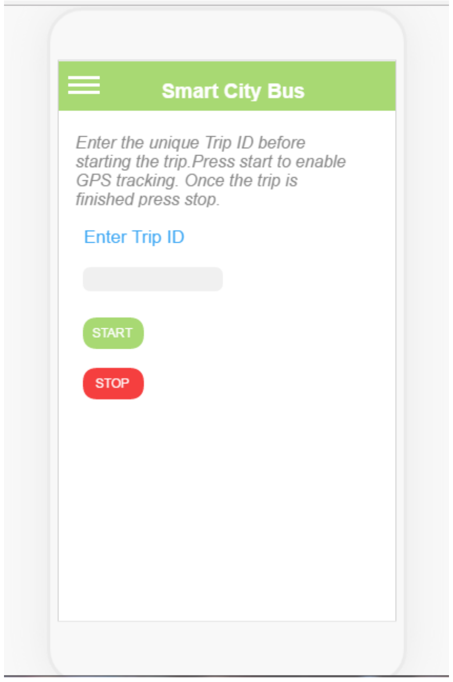


Arrives in	Stop	Towards
5mins	1st & Santaclara	Eastridge
18mins	1st & Santaclara	Eastridge
29mins	1st & Santaclara	Eastridge

Fare: $2 per Adult
Frequency: Every 15 mins

Fig.6 Screen2 when user clicks GO button

The following figure is the mockup developed for the GPS tracking application on the bus.



Enter the unique Trip ID before starting the trip.Press start to enable GPS tracking. Once the trip is finished press stop.

Enter Trip ID

Fig.7 Screen1 for GPS tracking application

The following figure is the UI mockup developed for the agency monitor portal where agencies can view the live locations of their vehicles.

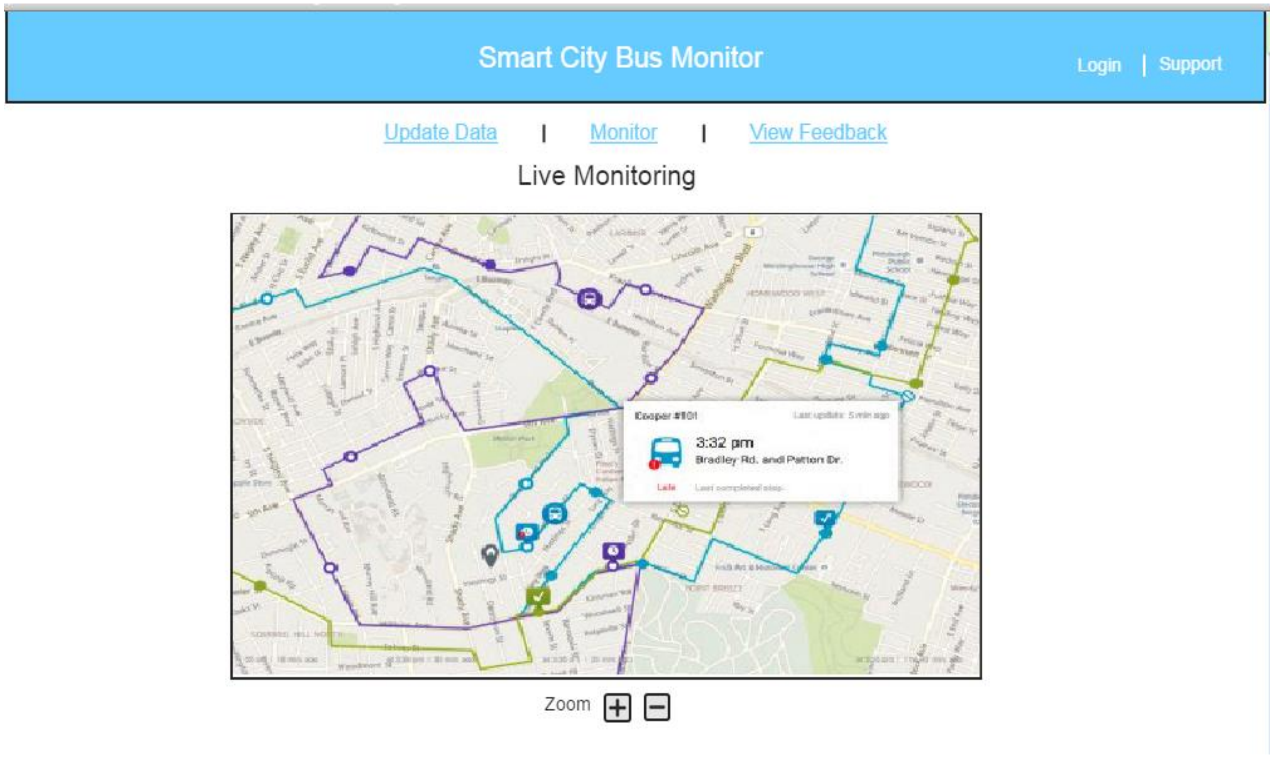


Fig. 8 Agency portal for monitoring transit vehicles.

Database Design:

As part of database design, we should be able to handle concurrent requests from users and feed files from the agency. Concurrent requests from users doesn't involve any high cpu computation apart from handling high traffic with smaller tasks. In case of feed file processing, standardization of data is expected for easy and reliable processing.

For the purpose of standardization, google have designed a schema known as GTFS (General Transit Feed Specification). Many of the existing transport agencies across the globe agreed and published their transit schedules according to proposed schema. As part of GTFS data transit agencies provide few csv files with inner dependencies. Depending on the standard relations maintained between them it is feasible to use relational database such as mysql to maintain such data. This provides easy maintenance and high flexibility to perform complex queries. This following figure depicts ERD (Entity Relationship Diagram) for maintaining feed data from agencies.

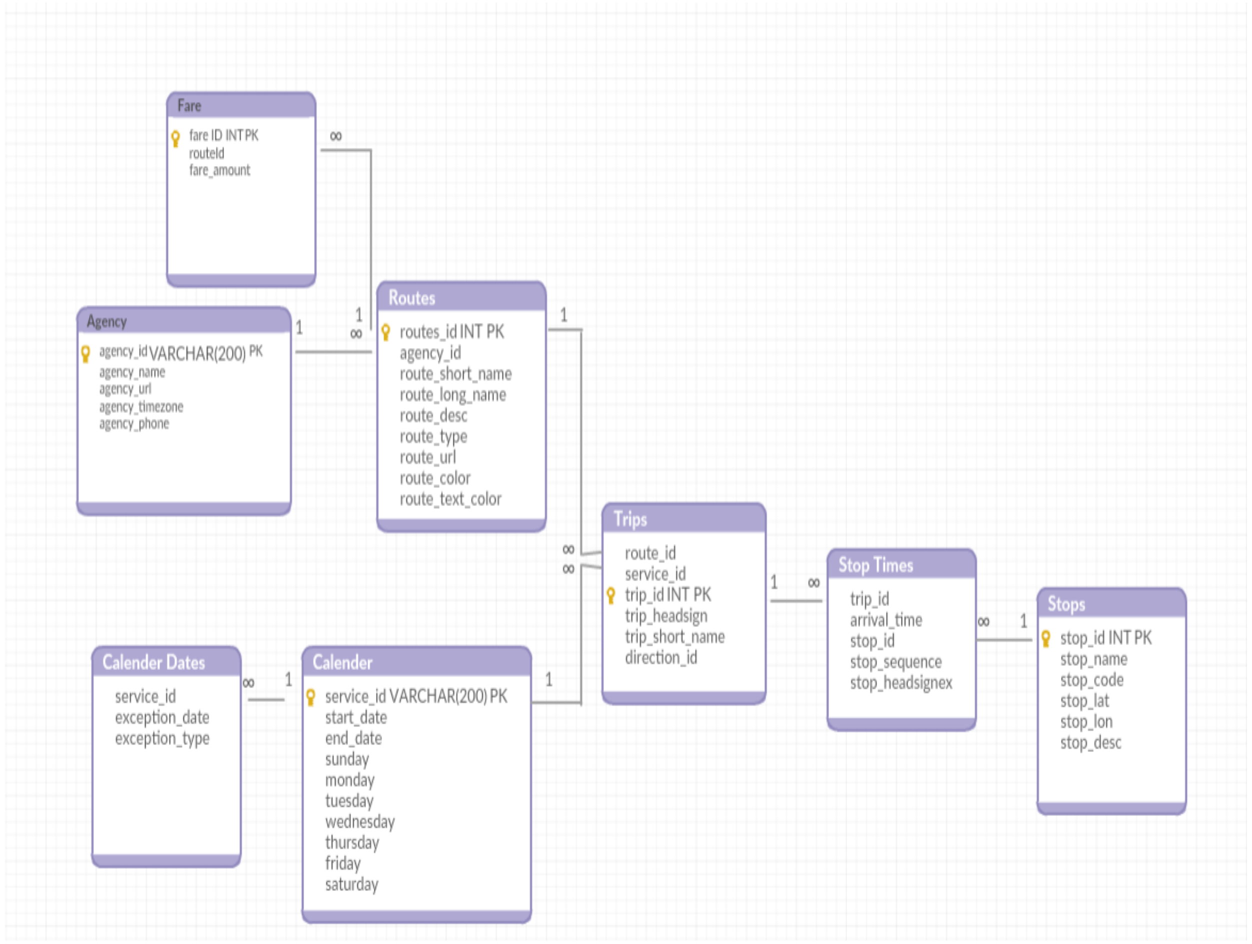
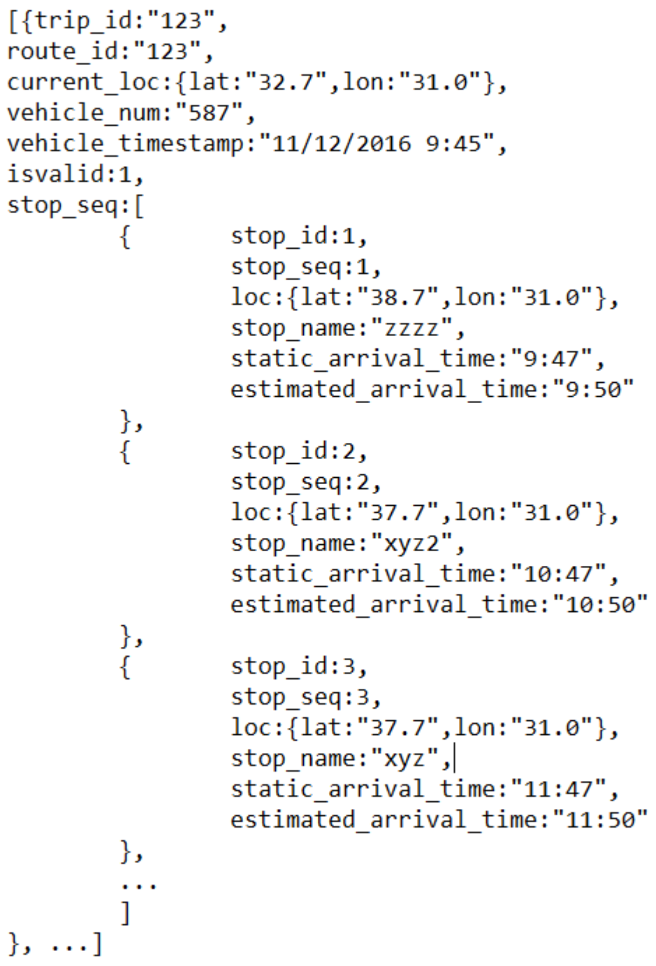


Fig. 9 Entity Relationship Diagram of feed database.

In case of handling the user requests and maintain current position of the vehicle, NoSQL database is utilized. As these requests are smaller tasks with minimum cpu computation, the node server uses mongo dB to query desired data in the form of objects and provide high responsiveness. When a new trip is initiated, suitable Entity to be inserted into mongo dB is queried from feed database. The mongo dB provides sufficient data to handle all trip related requests including estimated arrival time. Mongo dB being schema less, entities are flexible and independent from feed database. The data in mongo dB can be archived for any future utilization. The following picture shows a sample entity to be stored onto the NoSQL database.



```
[{trip_id:"123",
route_id:"123",
current_loc:{lat:"32.7",lon:"31.0"},
vehicle_num:"587",
vehicle_timestamp:"11/12/2016 9:45",
isvalid:1,
stop_seq:[
        {       stop_id:1,
                stop_seq:1,
                loc:{lat:"38.7",lon:"31.0"},
                stop_name:"zzzz",
                static_arrival_time:"9:47",
                estimated_arrival_time:"9:50"
        },
        {       stop_id:2,
                stop_seq:2,
                loc:{lat:"37.7",lon:"31.0"},
                stop_name:"xyz2",
                static_arrival_time:"10:47",
                estimated_arrival_time:"10:50"
        },
        {       stop_id:3,
                stop_seq:3,
                loc:{lat:"37.7",lon:"31.0"},
                stop_name:"xyz",
                static_arrival_time:"11:47",
                estimated_arrival_time:"11:50"
        },
        ...
        ]
}, ...]
```

Fig. 10 Mongo dB entity sample.

Chapter 7. QA, Performance, Deployment Plan

Test plan:

First, we are testing individual modules for their correct behavior. Once these modules pass individual behavior tests they will be combined with other modules and will be tested again for compatibility problems. Most of the tests are performed manually and tests are conducted in Linux environment. The individual modules which will be tested and their expected results are mentioned below.

S.NO	Module name/Event Occurrence	Expected Result
1	Opening user android application button.	Should open up a screen with nearby buses and their arrival time to their bus stops.
2	Clicking on ‘GO’ button in user android application	Should fetch up more details for the particular bus (e.g.: fare, frequency.)
3	Opening bus driver’s android application	A prompt asking to enter the unique trip id with buttons ‘start’, ‘stop’
4	Clicking ‘start’ button	Request connection to server and send GPS coordinates every 20 seconds.
5	Clicking ‘stop’ button	Request server to end the connection.
6	Network connectivity	The server applications should be tested for successful connection with both bus-side and user android applications
7	Operations on database	Check for server interaction with database by doing some manual insert, update, delete operations from the server.
8	Start the ETA calculation server	Will create a separate process for each route and a master thread which waits for change in mongo dB database.
9	Insertion of transit id object in database	Master thread should create a slave thread which will calculate the ETA for every bus stop for this transit id bus

10	Deletion of transit id object in database	Master thread should kill the slave thread calculating ETA.

Performance:
After completing the functionality testing it is necessary to check the performance of the application under different conditions to ensure it's working with no glitches at higher load. Performance testing can be done in this project by increasing and decreasing the variables and checking if the performance is stable. The following parameters will be varied during performance testing.

- Number of the Routes
- Number of the buses (transit ids) on each Route
- Number of stops in a particular Routes
- Number of the users requesting ETA

Deployment plan:
Before deploying the project some environment variables and dependencies should be set. Following are the requirements for the deployment:

- Amazon EC2 instances
- Linux operating system
- Node.js
- Python
- MongoDB Database
- MySQL Database
- Google Traffic API
- Google Play Store

Follow flowing steps for deployment

- Login to the Ec2 instance.
- Check if mentioned software were installed. If not install pending software and create stable environment.
- Dump GTFS data onto MYSQL database if it is empty.
- Start the servers.

Chapter 8. Implementation Plan and Progress

Our main deliverable for this projects is Android application to users. Users will be showed arrival time of buses for nearby bus stops. Users might select a bus to track it on the map. An efficient algorithm should be implemented to make schedules more reliable to clients.

The flowing are the implementation steps to be followed in the process of building Smart City Bus application.

The following steps define the implementation process.

1. Sample data is to be collected from different transit agencies operating at a location.
2. Using the GPS co-ordinates from the above collected data, Identify routes and corresponding stops in each route.
3. Database schema should be designed and implemented for easy and faster access of available data.
4. Build android app to receive live GPS coordinates from the vehicles travelling in identified routes.
5. Implement very efficient algorithm to predict estimation arrival time of next bus to the stop in all identified routes.
6. Build a server application to maintain latest schedules from the agencies enrolled for this application.
7. Host database and server applications on to cloud instance.
8. Build a user side hybrid android application to request estimated arrival time of bus at nearby stops in a filtered route.
9. A user may choose from filtered search to start tracking the bus. The node server should be able to serve client by sending current location vehicle periodically.
10. Project part A focus more on the design and specifications of our application. We will focus on researching different models to predict estimated arrival time very efficiently and structuring database. Apart from the algorithm we will try to build a small prototype for receiving live co-ordinates from android phone and update database.

11. Project part B will focus more on the development of end user android application and implementation of Estimation algorithm. This includes testing and verification especially after hosting on to the cloud platform.

Progress

We are currently in the process of going through literature to find efficient algorithm for reliable ETA prediction based on real time data. **We have done research on the technologies and the development tools required for successfully accomplishing our tasks. After a lot of research on ionic framework, we have decided to build our hybrid applications using ionic framework due to its extensible documentation and flexibility over other frameworks. Also, we have decided to build our server applications in NodeJs because of its compatibility in receiving and sending JSON data. As, JSON data uses less bandwidth when compared to XML and other file formats, we have decided to build NoSQL database using Mongo db. As these technologies are new to us, we are currently working on getting ourselves acquainted to the technologies and the tools.**

Following points describes progress on the application.

- **We have developed UI mockups for our android application and also for the agency monitor portal.**
- **Designed data model for the application which includes MySQL database entities and Mongo dB object entities.**
- **Built an android application to generate GPS co-ordinates and track it on google maps on ionic framework.**
- **Built a prototype for the bus tracker application which involves full stack development with regards this applications architecture mentioned. Following figure depicts above implementation with a help of blue line passing through block diagram.**

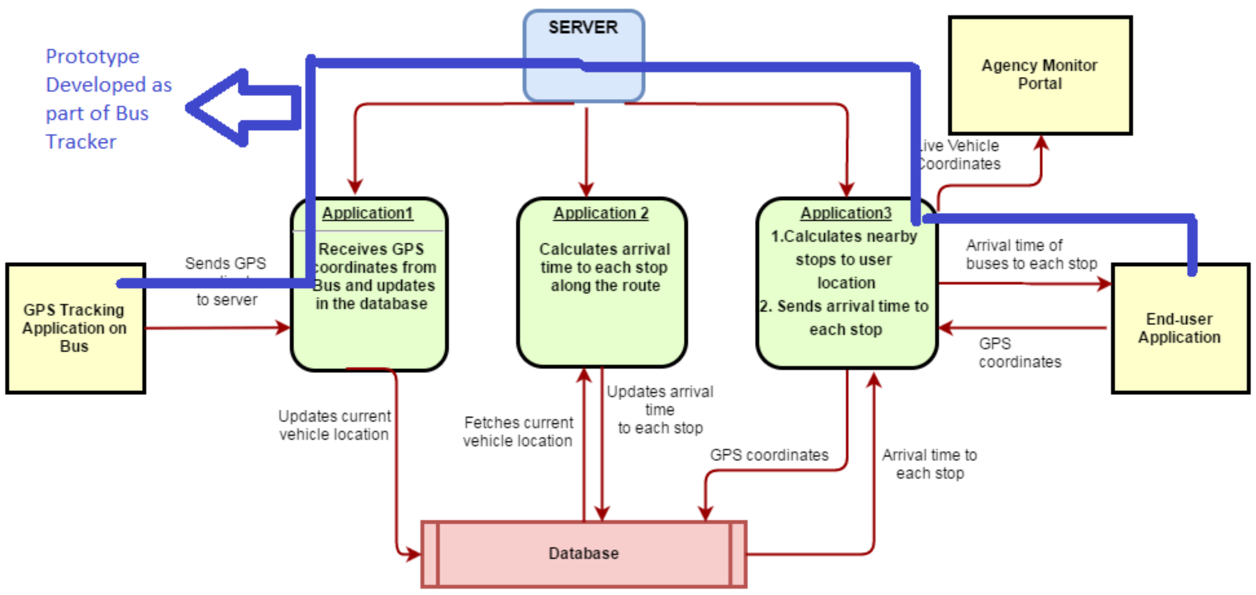

Prototype Developed as part of Bus Tracker
SERVER
Agency Monitor Portal
Live Vehicle Coordinates
Application1
Receives GPS coordinates from Bus and updates in the database
Application 2
Calculates arrival time to each stop along the route
Application3
1.Calculates nearby stops to user location
2. Sends arrival time to each stop
Sends GPS coordinates to server
GPS Tracking Application on Bus
Arrival time of buses to each stop
GPS coordinates
End-user Application
Updates current vehicle location
Fetches current vehicle location
Updates arrival time to each stop
GPS coordinates
Arrival time to each stop
Database

Fig. 11 Prototype implementation.

The prototype involves following applications

- **Two android applications**
- **One server application**
- **Mongo DB database**

As part of two android applications, the first application collects trip id and periodical GPS coordinates from the phone (even works in the background). The second application fetches the current location and displays on the google map.

The server application fetches the GPS co-ordinates from the tracker application and inserts or updates periodically to the database. It also involves fetching the current location if requested by the user.

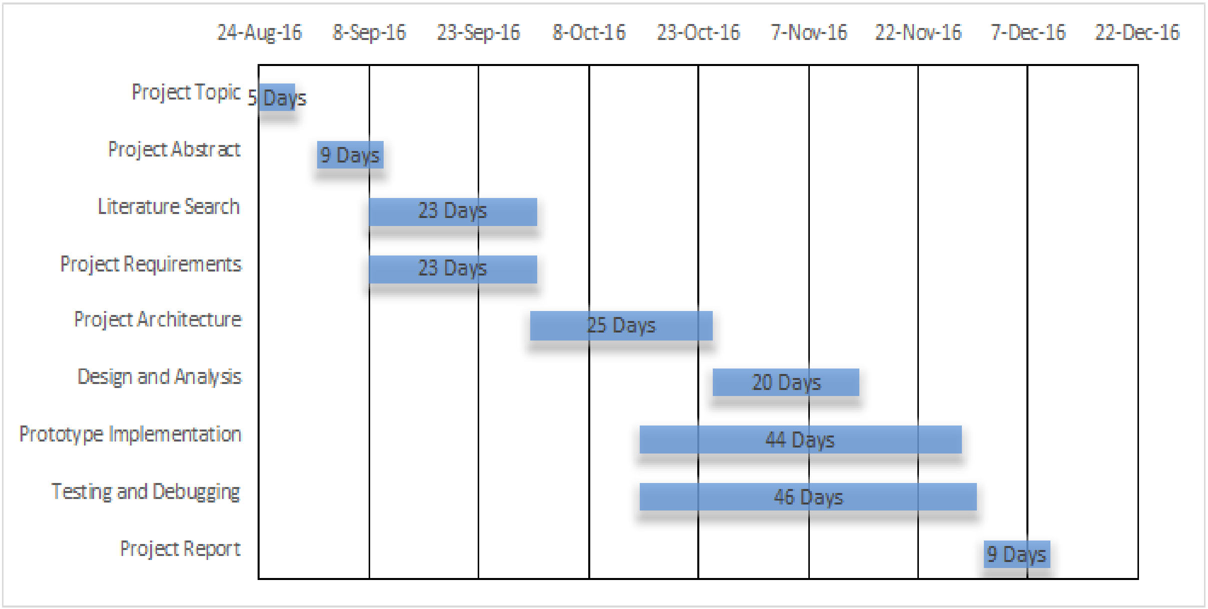
The mongo DB database holds the current position of the bus based on the trip Id.

Chapter 9. Project Schedule

Gantt chart – 295A

S. NO	Task	Owner	Start Date	End Date	Duration	Days Left	Status
1	Project Topic	Team	24-Aug-16	30-Aug-16	5	0	Completed
2	Project Abstract	Team	1-Sep-16	8-Sep-16	9	0	Completed
3	Literature Search	Varsha	8-Sep-16	30-Sep-16	23	0	Completed
4	Project Requirements	Ravali	8-Sep-16	30-Sep-16	23	0	Completed
5	Project Architecture	Sashank	30-Sep-16	25-Oct-16	25	0	Completed
6	Project Deliverables	Aditya	25-Oct-16	14-Nov-16	20	0	Completed
7	Implementation Plan	Sashank	15-Oct-16	28-Nov-16	44	14	Completed
8	Testing and Debugging	Team	15-Oct-16	30-Nov-16	46	16	In Progress
9	Project Report	Team	1-Dec-16	8-Dec-16	9	9	In Progress

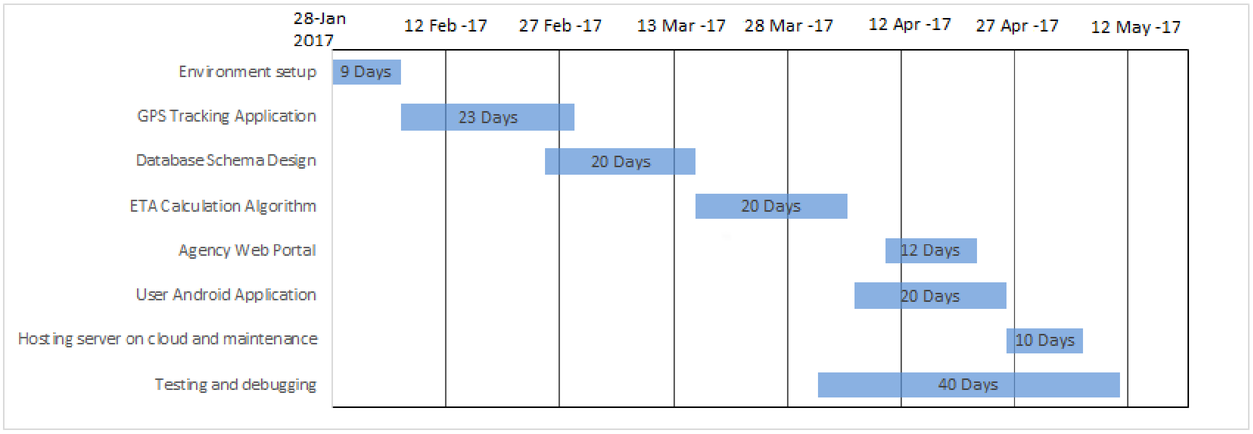
Gantt chart – 295A



Gantt chart – 295B

S. NO	Task	Owner	Start Date	End Date	Duration	Days Left
1	Environment setup	Team	28-Jan-17	5-Feb-17	9	9
2	GPS Tracking Application	Sashank	6-Feb-17	28-Feb-17	23	23
3	Database Schema Design	Sashank	25-Feb-17	15-Mar-17	20	20
4	Server Applications	Aditya, Ravali	16-mar-17	05-Apr-17	20	20
5	Agency Web Portal	Varsha	10-Apr-17	22-Apr-17	12	12
5	User Android Application	Varsha	06-Apr-17	25-Apr-17	20	20
6	Hosting server on cloud and maintenance	Aditya	26-Apr-17	06-May-17	10	10
7	Testing and debugging	Team	01-Apr-17	10-May-17	40	40

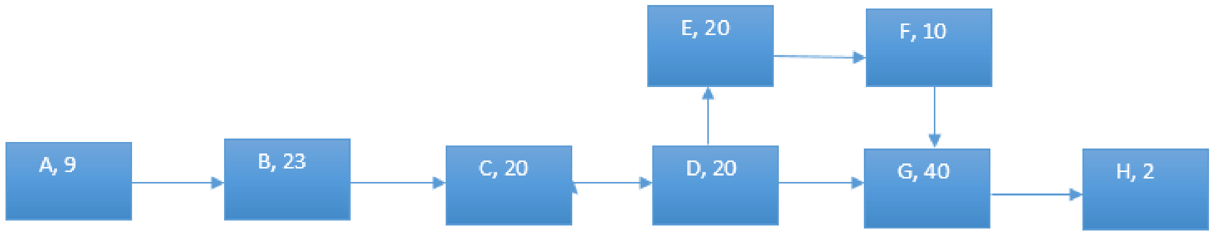
Gantt chart – 295B



Pert Chart -295B

S.NO	Task	Duration
A	Environment Setup	9
B	GPS Tracking Application	23
C	Database Schema Design	20
D	ETA Calculation Algorithm	20
E	User Android Application	20
F	Hosting Server on Cloud	10
G	Testing & Debugging	40
H	Analyzing results	2

Pert Chart- 295B


A, 9
B, 23
C, 20
D, 20
E, 20
F, 10
G, 40
H, 2