YamuColombo

Colombo Bus Route Management System
Software Architecture Document

Version 1.0

Revision History

Date	Version	Description	Author
10/04/16	1.0	Software Architecture Document	Lakshan Gamage

Table of Contents

Software Architecture Document

1. Introduction

1.1 Purpose

This document provides a comprehensive architectural overview of the system, using a number of different architectural views to depict different aspects of the system. It is intended to capture and convey the significant architectural decisions which have been made on the system.

1.2 Scope

This is the Software Architecture Document for YamuColombo android application which provides the possible bus routes to reach a particular destination, once the starting point and the destination are given to the system.

1.3 Definitions, Acronyms, and Abbreviations

RUP – Rational Unified Process
API – Application Program Layer
UML – Unified Modeling Language
GPS – Global Positioning System
SAD – Software Architecture Document
SDK – Software Development Kit
IDE – Integrated Development Environment
ADT – Android Development Toolkit

1.4 References

[1] "Google Places API," Google, [Online]. Available: https://developers.google.com/places/android-api/. [Accessed 06 March 2016].

[2] "Google Maps Android API," Google, [Online]. Available: https://developers.google.com/maps/android/. [Accessed 06 March 2016].

[3] "Colombo Bus Route," Google Play, [Online]. Available: https://play.google.com/store/apps/details?id=cbr.arima&hl=en. [Accessed 06 March 2016].

[4] "Algorithms to find routes," StackOverflow, [Online]. Available: http://stackoverflow.com/questions/483488/strategy-to-find-your-best-route-via-public-transportation-only. [Accessed 03 03 2016].

[5] "Android Studio," Google, [Online]. Available: http://developer.android.com/sdk/index.html. [Accessed 03 03 2016].

[6] "Google APIs Terms of Service," Google, [Online]. Available: https://developers.google.com/terms/. [Accessed 16 03 2016].

1.5 Overview

The first section so far has described the purpose, scope, definitions and the references of this system to provide an introduction to the reader. The second section provides the overview of the system describing its functionalities in detail. The third section is dedicated for the system design description. It includes first level decomposition of the system into functional components with diagrams. From fourth section onwards, it describes each component, what's its purpose, how it can be decomposed and the dependencies of the system component.

2. Architectural Representation

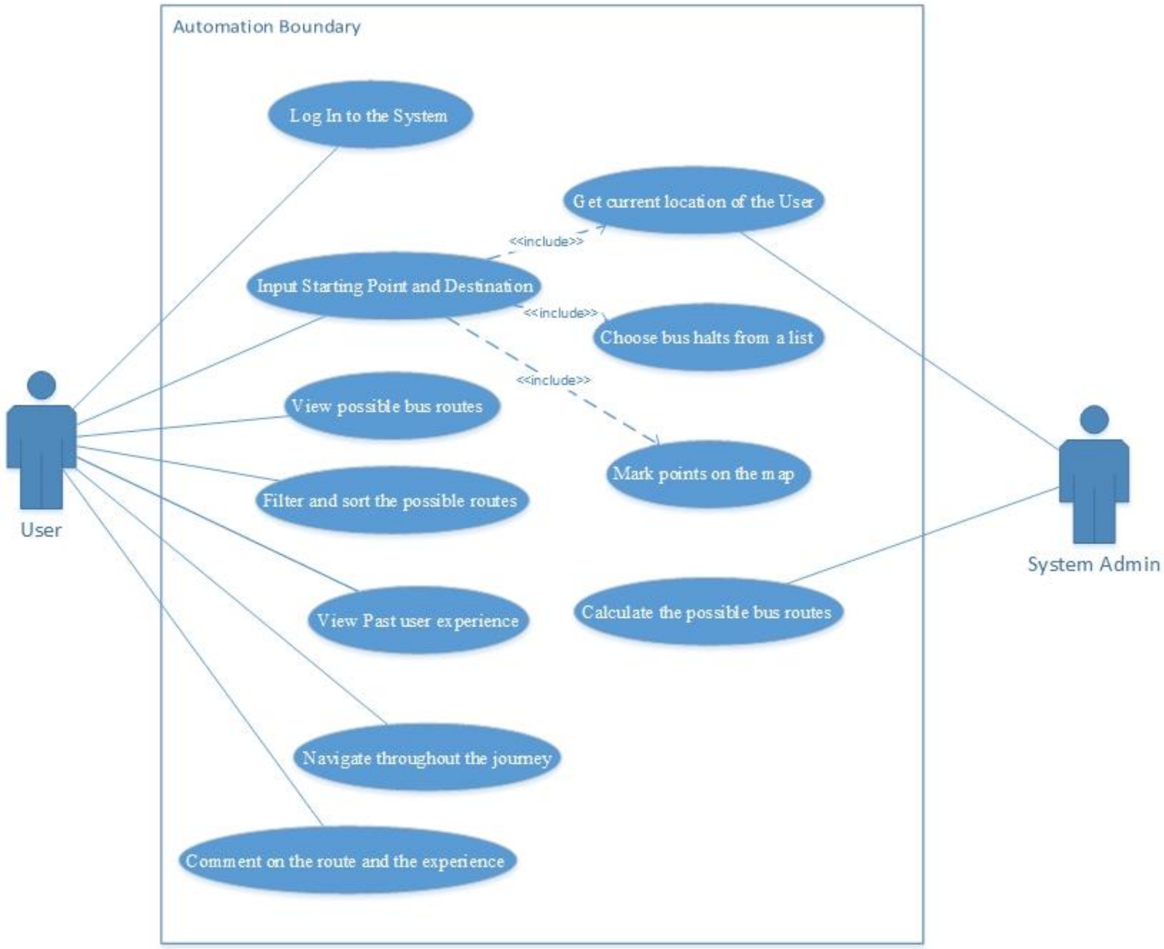
The architecture used in the current system is the client-server architecture and the Use-case, logical, Process, deployment and implementation views will be discussed further in each of the subsections in the document hereon.

3. Architectural Goals and Constraints

This system will be implemented based on the Client-Server architecture. The client will be the android phone and the back end server will be hosted which the client may connect through HTTP/HTTS. Therefore the server should have a considerable amount of performance to handle many users simultaneously.
All the considerations and the risks described in the system feasibility report should be considered when designing the system architecture.
As none of the private confidential details of the user will be taken for the application, there won't be any privacy issues for the user.
The selected architecture has to be implemented within the planned time limits in the project schedule.
Android SDK which is ADT will be used for the development and therefore its platform constraints have to be taken into consideration.

4. Use-Case View

4.1 Use-Case Realizations



Automation Boundary
Log In to the System
Get current location of the User
<<include>>
Input Starting Point and Destination
<<include>>
Choose bus halts from a list
<<include>>
View possible bus routes
Mark points on the map
User
Filter and sort the possible routes
System Admin
View Past user experience
Calculate the possible bus routes
Navigate throughout the journey
Comment on the route and the experience

Use case name	Log In
Actor	User
Description	In order to use the functionalities of the application the user has to log into the system providing the username and password given at the initial use.
preconditions	User should have an active internet connection in the phone.
Main flow	1. User open the application 2. Insert the username and password in the relevant space. 3. Click on log in button
Successful end/post condition	If log in successful, user will be able to use the functionalities of the system.
Fail end/post condition	If log in fails, user will be asked to enter the correct username or password.
Extensions	N/A

Use case name	Input Starting Point and Destination
Actor	User
Description	In order to find the possible bus routes for the journey, the user will have to enter the starting point of the journey and the destination.
preconditions	User should be logged into the system
Main flow	1. User choose an option to give those points. 2. Based on the option user defines the 2 locations. 3. Click on the find button.
Successful end/post condition	If the locations are successfully take the user will be given the list of all possible routes to go.
Fail end/post condition	If the locations are not recognized, the user will be asked to mark them properly again.
Extensions	Get current location of the user, choose bus halts from a list, mark points on the map

Use case name	View Possible Bus Routes
Actor	User
Description	User can view all the possible bus routes from his/her starting point to destination
preconditions	User should have entered the starting location and the destination correctly.
Main flow	1. User inputs the 2 locations 2. Click on the find routes button 3. Possible routes are calculated 4. Possible routes are displayed on the phone.
Successful end/post condition	If there are possible routes they will be shown to the user
Fail end/post condition	If there is no bus route to reach the destination, the user will be notified about that.
Extensions	

Use case name	Filter and sort possible routes
Actor	User
Description	When the user is given more than one possible route, he/she will be able to filter and sort the results based on the distance, cost and traffic.
preconditions	There should be more than one possible routes in the search results
Main flow	1. User receives the results 2. Choose sorting option 3. Choose the facts to be considered 4. View the results in sorted order
Successful end/post condition	User will be shown the results sorted in the order which he/she asked for.
Fail end/post condition	N/A
Extensions	

Use case name	View past user experience
Actor	User
Description	When a user is given all the possible routes to reach the destination, he/she will be able to view the comments given by the previous users who travelled along that route.
preconditions	There should be at least one previous user who has travelled in that route and shared his/her experience.
Main flow	1. User receives the results 2. Click on the comments button 3. Previous responses are displayed
Successful end/post condition	User will be shown the comments given by previous users about that route.
Fail end/post condition	N/A
Extensions	

Use case name	Comment on the route and the experience
Actor	User
Description	When a user reaches his/her destination, they will be asked to share their thoughts on the route and share the overall experience and whether they will suggest that route for someone else.
preconditions	User has finished the journey
Main flow	1. User reach the destination 2. Add comment pop up appears 3. User share his/her experience 4. Click submit button
Successful end/post condition	User will receive a Thanking message for taking part in the community service.
Fail end/post condition	User will be prompted that the response could not be saved at the moment. They will be allowed to do it later at any time they are free.
Extensions	

Use case name	Navigate throughout the journey
Actor	User
Description	When the user selects a particular bus, he/she will be able to navigate using google maps throughout the journey.
preconditions	User selects a route and the GPS function of the mobile phone should be turned on.
Main flow	1. User selects a route 2. User clicks on navigate button 3. User's route will be highlighted and displayed with the current location and the destination.
Successful end/post condition	User will be able to navigate successfully
Fail end/post condition	User will be notified with the possible reasons.
Extensions	

Use case name	Calculate the possible bus routes
Actor	Admin
Description	When the user gives the starting point and the destination, the system will calculate the possible bus routes between those 2 points and sort them according to the cost, distance and traffic at that moment.
preconditions	User should have given 2 points on the map correctly
Main flow	1. User inputs the 2 locations 2. Click on the find routes button 3. Possible routes are calculated
Successful end/post condition	All possible routes will be displayed on the user's device.
Fail end/post condition	Error condition will be notified to the user
Extensions	

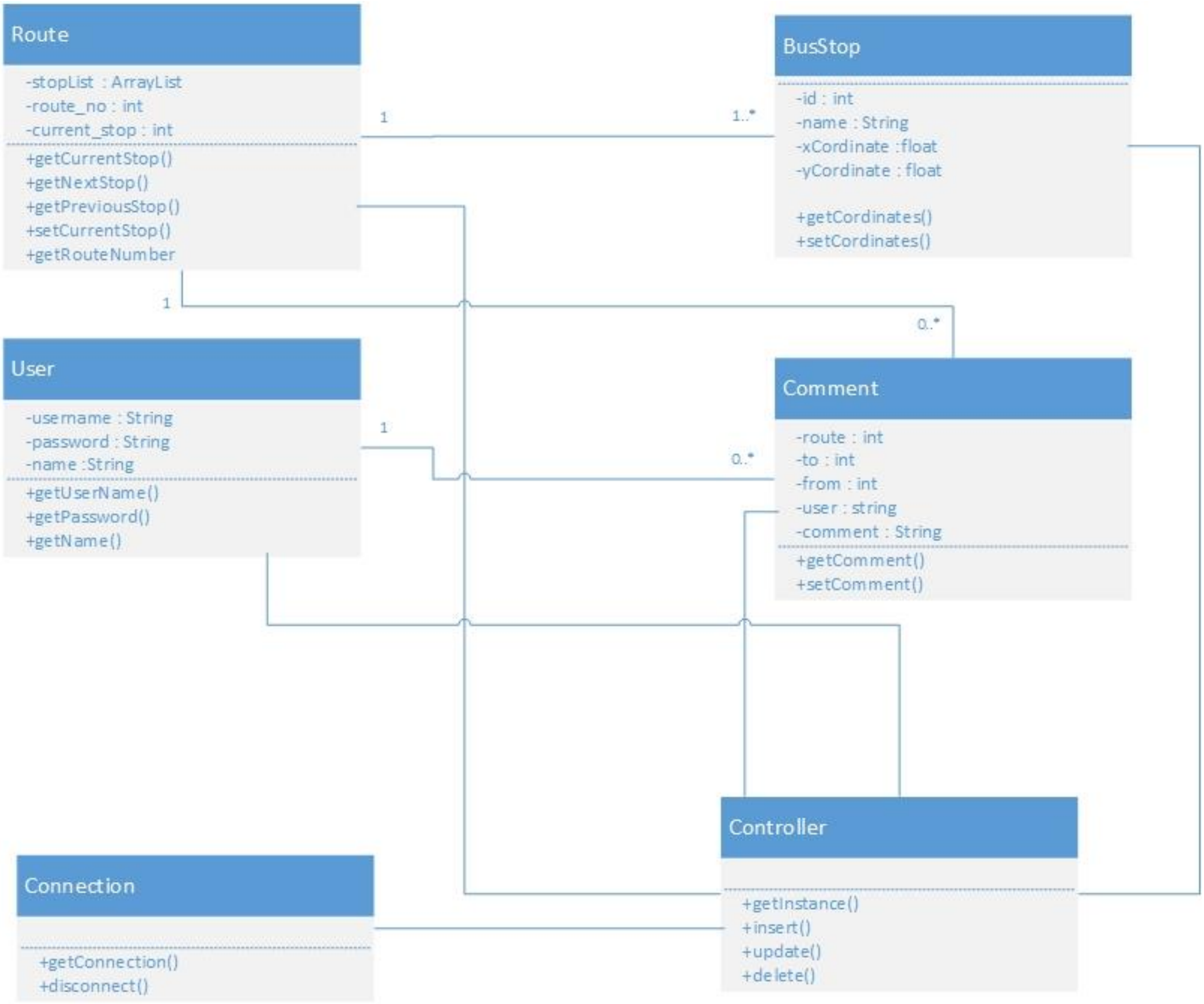
5. Logical View

5.1 Overview

This section will describe the class hierarchy and organization with their relationships.

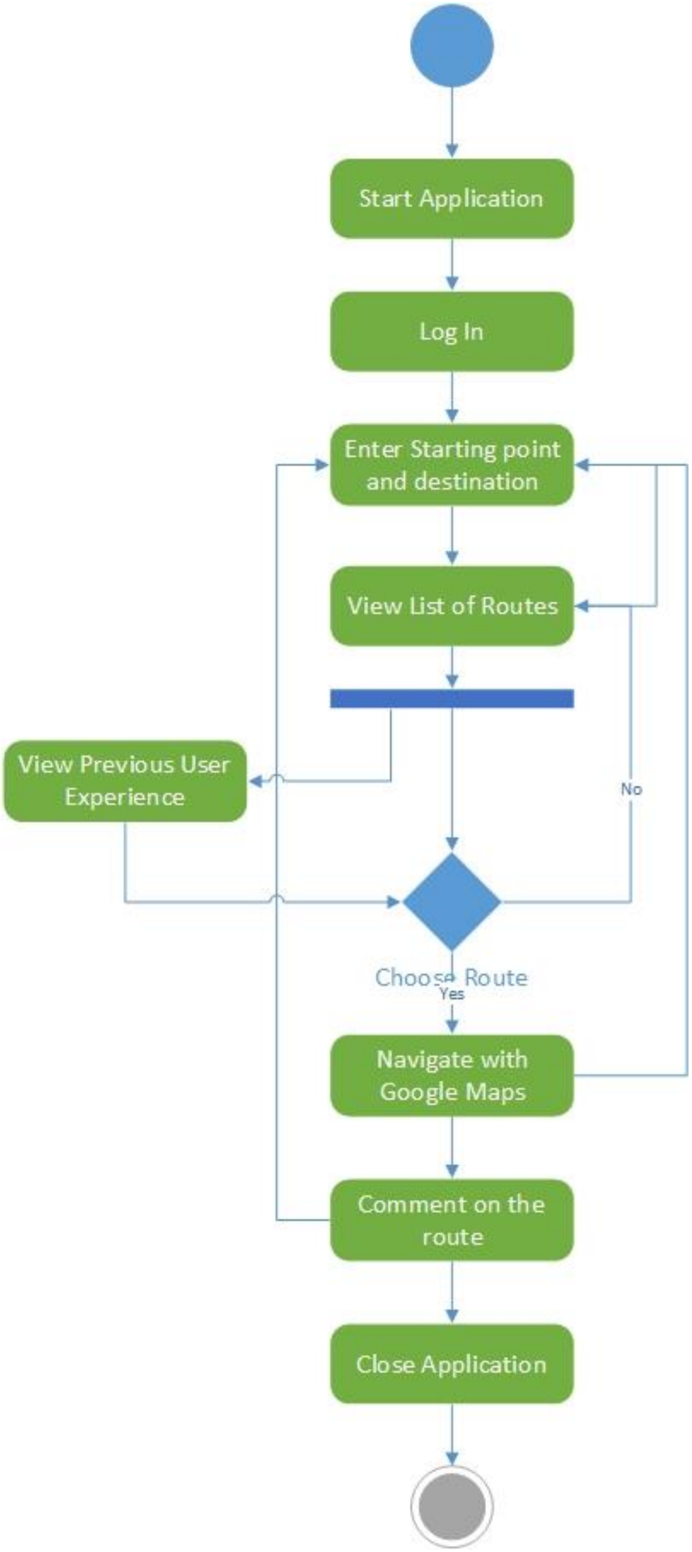
5.2 Architecturally Significant Design Packages

Controller class is used to do transactions with the database. It does all updates, inserts and deletions. The connection class is used to establish the connection and disconnect when needed. When a particular bus route is taken, it will be represented by the route class and that will contain a list of bus stops which has the sequence order of each bus stop. The user comments on a particular journey will be processed through the comment class and the details of a user is processed through the user class.



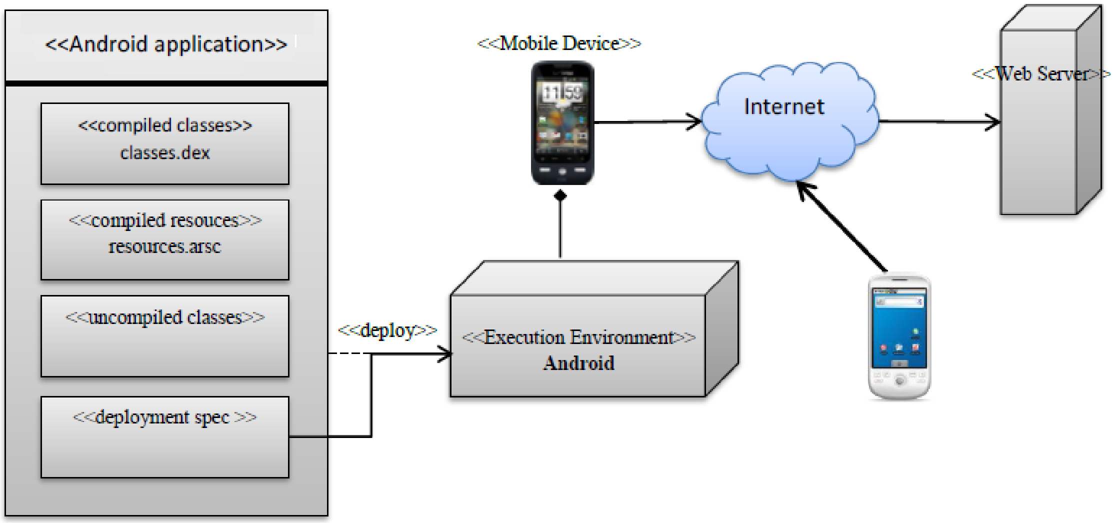
UML Class Diagram

6. Process View



UML Activity Diagram

7. Deployment View



Deployment Diagram

Android applications are composed of one or more application components (activities, services, content providers, and broadcast receivers). Each component performs a different role in the overall application behavior, and each one can be activated individually.

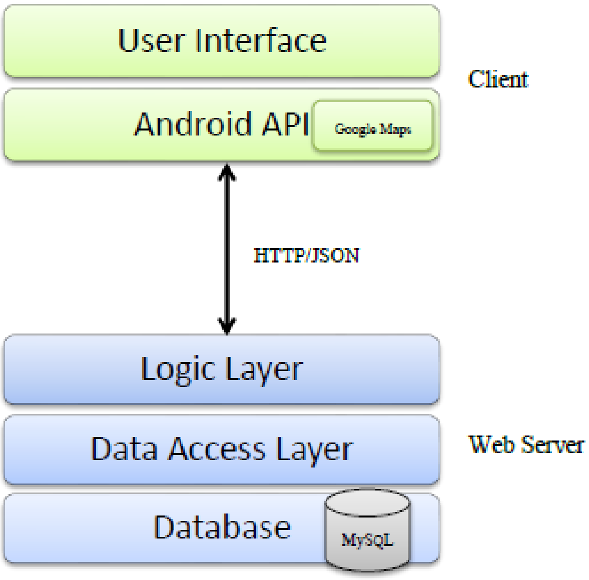
8. Implementation View

In this section, the structure of the implementation will be described and the layered architecture which will be used in this application will be discussed.

8.1 Overview

The application will be designed in client – server architecture. Message passing between the client and the server will be done in the form of HTTP/JSON objects. The logical layer of the server responses to the client requests and provide the relevant results. Logic layer, data access layer and database layer in the web server provides the functionality to handle user requests and respond with the best route from the database. If user wants to navigate with Google Maps, client uses the external Google Maps library.

8.2 Layers



Layered Architecture

Client Side

- User Interface

  This is the layer which contains the Graphical User Interface of the mobile application. It provides a means to the user to interact with the system.

- Android API

  Whole system is built on Android API. The application must employ common modules and libraries to achieve core functionality of an android application. In addition it uses Maps API and other libraries to include additional functionalities.

Server Side

- Logic Layer

  This layer controls the whole logic behind the route selection. The logic layer contains all the components related to selection of bus routes according to the algorithm. It gathers all the subsystems that meet the needs of the application domain.

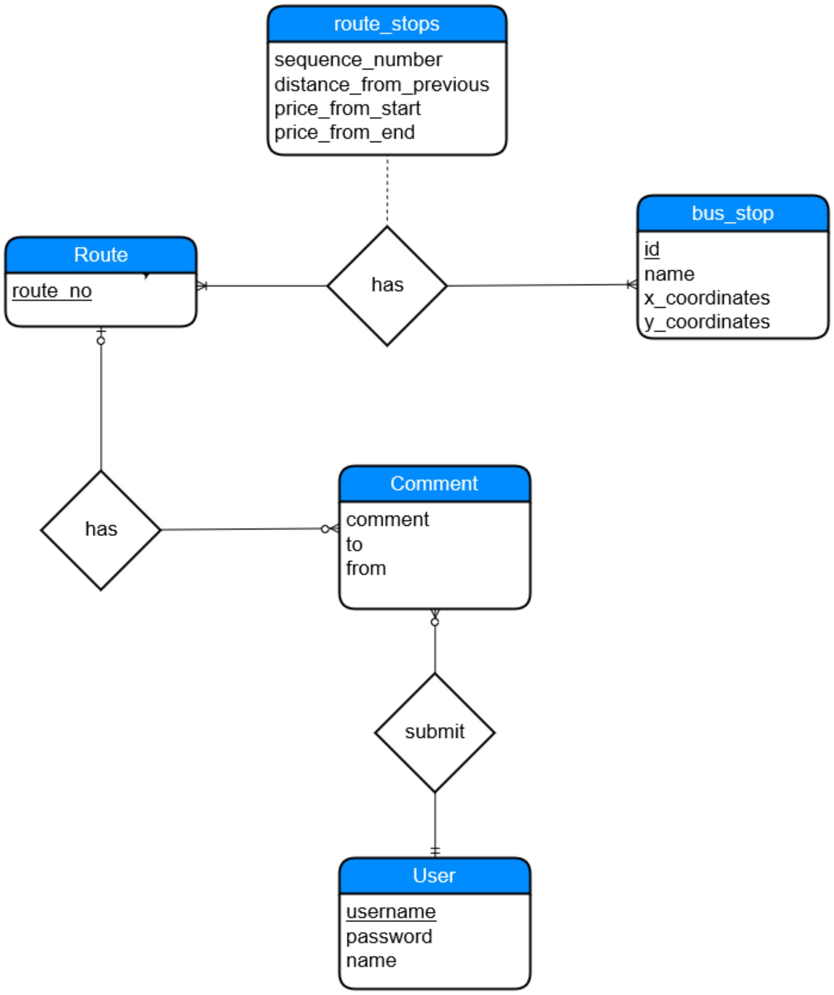
- Data Access Layer

  Data access layer provides simplified access to data stored in persistent storage such as an entity-relational database.

- Database

  Database will be implemented in MySQL and will reside in the web server to serve client requests.

9. Data View (optional)



ER Diagram

10. Size and Performance

10.1 Response Time

When the user inputs are taken, the calculated results should be available on the phone in less than 10 seconds. Users don’t want to wait a long time for the results to appear. However the available internet connection also affects this response time.

10.2 Transactions per second

The backend database can support up to 80 transactions per second. But it will depend on the use case as well.

10.3 Resource Utilization

The application won’t use much of the memory of the mobile device and if the GPS option is used, then that will cause the phone battery to drain much faster. Same if the data connection is weak, the battery drainage will be higher.

11. Quality

11.1 Usability

11.1.1 Simple GUIs

The user interfaces will be designed in a very simple way which won’t become complicated for the user to make use of the functionalities of the system. Complicated GUIs will make the user uncomfortable and will make them to take a long time to get familiar with.

11.1.2 Attractive GUIs

Even though the GUIs are simple, they will be designed in an attractive manner. Otherwise the app will be boring for most of the users.

11.1.3 Eye catching color schemes

Color schemes of the GUIs will be selected so as not to make the user uncomfortable.

11.1.4 Sufficiently large buttons and text

Text and buttons on the GUIs will be large enough for any user to see and touch easily.

11.1.5 User Guide

There will be a simple user guide in any case the user still finds it hard to understand to perform a certain functionality.

11.2 Reliability

11.2.1 Availability

As this application depends very much on the backend server, it has to be on and running all the time. Whenever a user needs to use the application, the server should be readily available to offer the service.

11.2.2 Accuracy

The results of the calculation should be accurate and should not have any misleading or unavailable routes. Otherwise the reliability of the application will reduce a lot and the application will have a negative impact from the users.

11.2.3 Mean Time to Recover

If the server break down for some reason, it should be back on at least within 24hours. More the time it takes, more will be the damage on the user base of the application.