Fall 2013

CPCS 462

Group 5

Metro Live™ Tracking System

Team Members:

Sultan Aabdullatif
Seyed Ahmadpanah
Jeff Bohlin
Pruthivin Reddy Madduri
Amir Shokrollahshirazi

Instructor:

Sara Hariri

Abstract

Smart-phone uses are expanding rapidly through a number of applications that enhance the day-to-day lives of people worldwide. Public transportation is one such area that we can add technology to achieve this. Geographic Information Systems (GIS) can be embedded into any modern application to greatly expand functionality and create possibilities that are limited only by our imaginations. In this project, we will develop an application that combines GPS, GIS, public transportation, and various sources of real-time data in order to deploy the "Metro Live Tracking System."

Keywords List

Live: referring to real-time GPS data.

Tracking System: the observing of persons or objects on the move and supplying a timely ordered sequence of respective location data to a model e.g. capable to serve for depicting the motion on a display capability.

GIS: a Geographic Information System (GIS) which integrates hardware, software, and data for capturing, managing, analyzing, and displaying all forms of geographically referenced information.

GPS: the Global Positioning System (GPS) is a satellite-based navigation system made up of a network of 24 satellites placed into orbit by the U.S. Department of Defense.

LAMTA: Los Angeles Metropolitan Transportation Authority

Table of Contents

List of Figures

List of Tables

1. Introduction

1.1 Description of the Problem

The city of Los Angeles's Metropolitan Transportation Authority (LAMTA) has no system in place that allows its population to view and track buses in real-time. Currently, patrons must refer to a paper based bus schedule that is posted at each individual bus stop, or go online to view a static web page.

Time management is an important aspect for most people and the need for access to real-time data is high. One core problem that Angelenos face is how to actually obtain this real-time data when needed, especially in the world of public transportation. The goal of this project is to create a mobile application that will allow the people of Los Angeles to track buses in real-time on a Google-like map display.

1.2 Goal and Objectives

- Users are able to view and track buses.
- GPS tracking data must be received quickly and accurately.
- Make the system available 24/7.
- Build a system which can handle large amounts of data from various sources.
- Create a user-friendly interface for users.

1.3 Purpose

The purpose of the Metro Live Tracking System is to provide a solution to the problem outlined in *1.1 Description of the Problem*. This project aims to create an application that allows the user to track and view bus routes in real-time. The project's main objective is to ensure that the overall experience is smooth, efficient, and provide the user with accurate data quickly, so that the task of locating a bus is as painless as possible.

1.4 Vision and Scope

The Metro Live Tracking System is a system that will allow the user to track buses in the city of Los Angeles. Each user will be able to track, locate, and receive time estimations for bus routes within the city limits. The user will also able to search a directory for a specific bus number as well as view all bus stop locations within the city. The application will provide real-time bus status updates and display the bus's current location on a Google-like map. The application will provide an estimated distance

between the user's current location and the bus they choose. Finally, all GPS data will be provided and updated by the LAMTA.

1.5 Definitions, Acronyms, and Abbreviations

Android: an open source based operating system designed primarily for touch-screen mobile devices such as smart phones and tablet computers.

Apache: the HTTP server, commonly referred to as Apache, is a web server application notable for playing a key role in the initial growth of the World Wide Web.

API: Application Programming Interface.

iOS: iPhone Operating System.

JAVA: a programming language and computing platform.

JavaScript: a programming language used to make web pages interactive. It runs on your visitor's computer and doesn't require constant downloads from your website.

Notepad++: a free source code editor and Notepad replacement that supports several programming languages.

SMS: Short message service.

OS: Operating System.

Visio: a diagramming and vector graphics application that is part of the Microsoft Office suite.

1.6 Development Environment

1.6.1 Software

- Programming Languages JAVA, JAVAScript
- Code Editor Notepad++, Eclipse
- Web Server Apache
- Support Tools Visio

1.6.2 Hardware

Hardware Type 1 - PC	Sony Vaio	
	CPU	Intel i7 3.5GHz
	RAM	8GB
	Hard Drive Space	2TB
Operating System	Windows 7	
Database	MySQL 5.6	
Server Software	Apache Accumulo	

Hardware Type2 - Cellular Phone	HTC	
	CPU	Qualcomm® Snapdragon 600 quad-core 1.7ghz
	RAM	2GB
	Hard Drive Space	32GB
Operating System	Android	

1.7 Operational Environment

1.7.1 Software

- Hosting Private Website, Android Mobile
- OS Android

1.7.2 Hardware

- Mobile device Cellular phone capable of running Android OS

2. Requirements Description

2.1 Functional Requirements

2.1.1 Functional Requirements for Users

Functional Requirements	
Track	T1: The user shall be able to track all buses within the system.
	T2: The user shall be able select a specific bus to track.
	T3: The user shall be able to locate the bus on map.
	T4: The user shall be able to view the live bus location on the map.

Functional Requirements	
Search	S1: The system shall display the number of active routes and buses.
	S2: The system shall be able to find buses by route.
	S3: The system shall provide the next bus stop location.
	S4: The system shall provide detailed information about bus arrival.
	S5: The user shall be able to view bus route and destination information directly through the application interface.
	S6: The user shall be able to retrieve route and destination information through text message.
	S7: The user shall be able to retrieve route and destination information from LAMTA in the form of a .pdf file.

Functional Requirements	
Favorite	F1: The user shall be able add a bus route to their favorites list.
	F2: The user shall be able to delete a bus route from their favorites list.
	F3: The user shall be able to display a bus routes favorite list.

Functional Requirements	
Contact	C1: The user shall be able to call the LATMA help center.
	C2: The user shall be able to view the Metro Live Facebook page.
	C3: The user shall be able to view the Metro Live Twitter page.
	C4: The user shall be able to e-mail the Metro Live team with suggestions, comments, or concerns.

2.1.2 Functional Requirements for Administrators

N/A

2.2 Non-Functional Requirements

2.2.1 Security Requirements

Non-Functional Requirements	
Security	S1: The minimum length of a password is 6 digits.
	S2: The system shall provide a password recovery function.
	S3: The system shall provide cell phone number verification capability.
	S4: The system shall access the API by a key assigned by the Transportation Authority.
	S5: The system shall be able to prevent SQL injections.
	S6: The system shall prevent invalid e-mail access.
	S7: The system shall clear the user's session upon logout.

2.2.2 Performance Requirements

Non-Functional Requirements	
Performance	P1: The system shall able to handle more than 500 users at the same time.
	P2: The system shall retrieve live data.

2.2.3 Reliability Requirements

Non-Functional Requirements	
Reliability	R1: The system shall execute the expected functionality correctly.
	R2: The system shall provide accurate data.
	R3: The system shall provide appropriate error handling and recovery methods.
	R4: The system shall be maintained regularly through updates and bug fixes.

2.2.4 Availability Requirements

Non-Functional Requirements	
Availability	A1: The system shall be available 24 hours, 7 days a week.
	A2: The LAMTA help center shall be available from 9 AM to 5 PM PST, Monday through Friday.

2.2.5 Efficiency Requirements

Non-Functional Requirements	
Efficiency	E1: The system shall retrieve real-time data quickly and accurately, while requiring minimal system resources.

2.2.6 Usability Requirements

Non-Functional Requirements	
Usability	U1: The system's interface shall be clear and easily navigable.
	U2: The system shall provide operational instructions to new users.
	U3: The system shall display error messages when the user enters wrong or invalid information.

2.2.7 Maintainability Requirements

Non-Functional Requirements	
Maintainability	M1: The system shall provide the following maintainability tools: • Data backup tools • Reporting Tools • Policy adjustment tools

2.2.8 Portability Requirements

N/A

2.2.9 Testability Requirements

Non-Functional Requirements	
Testability	T1: The system shall provide testability tools for the administrators.
	T2: Each function shall have its own specific method for testing.

3. Management Process

Team Background

Team Member	Title/Role	Strengths
Jeff Bohlin	Project Manager -UI Design	General Programming, C++/C, Assembly (Intel), HTML/CSS, GUI
Sultan Aabdullatif	System Engineer / Implementer -Lead Programmer	C, C++, HTML, Java, PHP, VB, MySQL , SQL2008 Software Engineering, Database & Web Programming
Amir Shokrollahshirazi	Integrator -Graphic Design	ASP.NET, SQL Server ,C#, JavaScript , PHP, MySQL , CSS, JQuery Mobile, HTML, UML
Seyed Ahmadpanah	Tester, Analyst	ASP.Net MVC, ASP.Net, C#, C++, SQL Server, My SQL, Visual Basic, HTML, PHP, CSS
Pruthivin R. Madduri	Tester, Analyst	C, C++, SQL, HTML, Microsoft OS, Linux, UNIX, Oracle, SQL/PL-SQL, MS-Access, Publisher

Table 1: Team Background

Role Description

Role	Responsibility
Project Manager	Responsible for the planning, sequencing, scheduling, and the allocation of resources throughout the software development process.
Implementer	Responsible for the development of specific elements according to designs, requirements, and the architecture.
Integrator	Responsible for taking individual components and integrating them, according to the architecture and system designs.
System Engineer	Responsible for design and development of systems or system components in which software plays a role in.
Tester	Responsible for the (independent) test and verification of the system or its elements against the formal requirements and the architecture.
Analyst	Responsible for the documentation of all facets of the project as well as analytical and statistical tools to measure the progress of the project as a whole.

Table 2: Role Description

3.1 Project Schedule

3.1.1 Working Days Overview

[Breakdown by weekly tasks]

Note that while all team members contributed at opportune times according to strength, listed team members contributed heavily during noted weeks.

Week	Team Member	Tasks
1: Aug 24 - 30	ALL	Formed group 5. Came up with project idea (bus tracking software).
2: Aug 31 -Sep 6	ALL	Refined project idea. Developed first two core use cases.
3: Sep 7 -13	ALL	Worked with customer for more detail. Determined preliminary requirements.
4: Sep 14 -20	ALL	Worked with customer for more detail. Determined the rest of the use cases.
5: Sep 21 – 27	Pruthivin	Worked with customer for more detail. Designed look of software (prototype).
6: Sep 28 –Oct 4	Sultan, Pruthivin, Amir	Built the application interface based on sketches (writing CSS code and template construction).
7: Oct 5 -11	Sultan	Built the database and application structures (MySQL tables, data flow, etc.).
8: Oct 12 -18	Sultan	Built the application connectivity to API systems and database (test and connect to external servers).
9: Oct 19 -25	Sultan	Started the core coding to develop client requirements (Track, Search and Locate functions).
10: Oct 26 –Nov 1	Sultan	Continued coding the core functionality (Track, Search and Locate functions).
11: Nov 2 - 8	Sultan	Completed core functionality. Began coding secondary functionality (schedule via .pdf, via text message, phone calling, Facebook page, Twitter page).
12: Nov 9 - 15	Pruthivin, Mike	Continued coding secondary functionality. Began intensive testing of core functionality and basic testing of secondary functionality.
13: Nov 16 - 22	Pruthivin, Mike	Completed Metro Live Beta version. Intensive testing continued on both core and secondary functionality.
14: Nov 23 - 29	Mike, Amir, Jeff	Continued intensive testing. Created all user documentation (Power Point, user and developer instruction manuals, and all system interaction diagrams).
15: Nov 30 - Dec 6	All	(Tentative) Final testing and bug fixes.
16: Dec 7 - 13	All	(Tentative) Metro Live 1.0 release.
Beyond	Sultan	(Tentative) General maintenance and continual bug fixes.

Table 3: Working Days Overview

3.1.2 Top Level Tasks

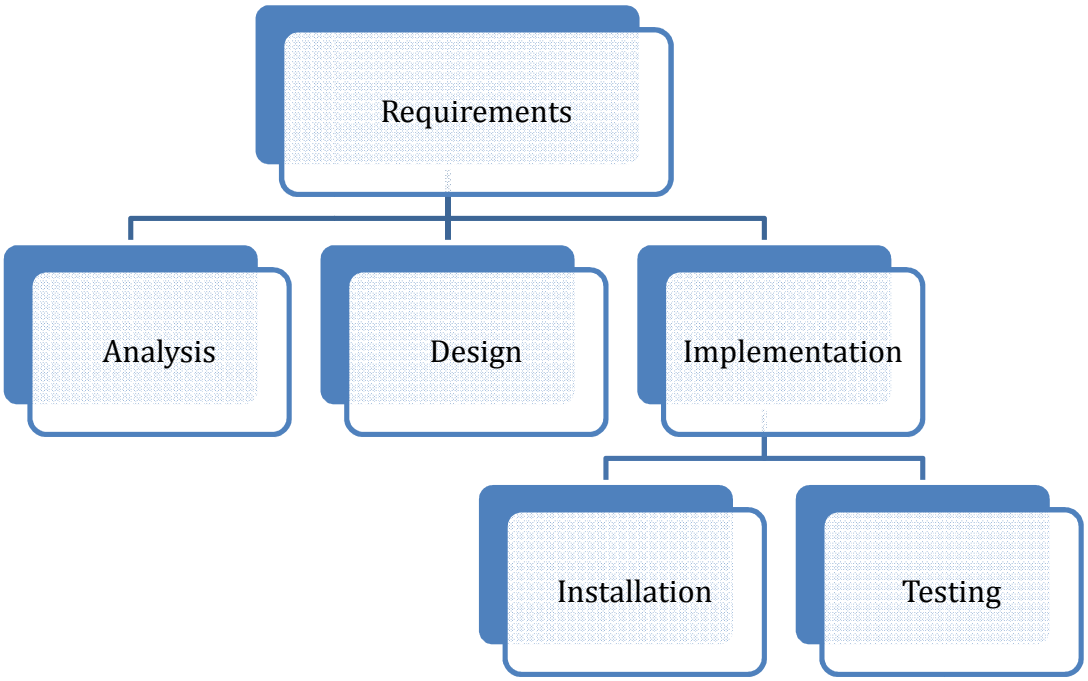


Figure 1: Top Level Tasks

3.2 Iteration Plan

Note: There are exactly 8 iterations, as outlined by the Rational Unified Process.

Phase	Iteration	Tasks Completed
Inception	Inception: Iteration 1	- Formed Group 5. - Decided on a topic for our project (bus tracking software). - Met with our client and began discussing requirements. - Discussed the two most important use cases. - Developed our contract and scope.
Elaboration	Elaboration: Iteration 1	- Identified the rest of the requirements. - Identified the additional use cases. - Continued developing our overall project plan. - Began building the application interface based on prototype. - Wrote the CSS code that controls the UI. - Built the preliminary UI template. - Began building the database to house user accounts. - Built the application data structures to hold the GPS data.
Elaboration	Elaboration: Iteration 2	- Finalized the Functional and Non-functional requirements. - Finalized the core use cases. - Finalized our development plan. - Began coding the client functional requirements. - Built the core application functions: Track, Search and Locate. - Finished building the database. - Modified the UI according to client specifications. - Built the applications functionality to allow for connectivity to the Transport Authority's API. - Tested connectivity between the application and external servers (GPS). - Began preparations for entering the Construction phase.
Construction	Construction: Iteration 1	- Completed core functionality. - Began coding secondary functionality (schedule via .pdf, via text message, phone calling, Facebook page, Twitter page). - Continued coding secondary functionality. Began intensive testing of core functionality and basic testing of secondary functionality.
Construction	Construction: Iteration 2	- Completed Metro Live Beta version. - Continued intensive testing on both core and secondary functionality.
Construction	Construction: Iteration 3	- Continued ongoing testing. - Created all user documentation (Power Point, user and developer instruction manuals, and all system interaction diagrams). - Began preparations for entering the Transition phase.
Transition	Transition: Iteration 1	- (Tentative) Final testing and bug fixes. - (Tentative) Metro Live 1.0 release.
Transition	Transition: Iteration 2	- (Tentative) General maintenance and continual bug fixes.

Table 4: Iteration Plan

3.3 Project Plan

3.3.1 Phase Plan

[Breakdown by Unified Process Phases]

Phase	Week	Tasks and Disciplines
Inception	1: Aug 24 - 30	**Business Modeling:** Came up with project idea. Formed group 5.
Inception	2: Aug 31 -Sep 6	**Requirements:** Determined preliminary requirements. Refined project idea.
Elaboration	3: Sep 7 -13	**Analysis and Design:** Developed first two core use cases. Worked with customer for more detail.
Elaboration	4: Sep 14 -20	**Analysis and Design:** Determined rest of use cases. Worked with customer for more detail.
Elaboration	5: Sep 21 – 27	**Analysis and Design:** Designed look of software (prototype) Worked with customer for more detail.
Elaboration	6: Sep 28 –Oct 4	**Analysis and Design:** Built the application interface based on sketches (writing CSS code and template construction).
Elaboration	7: Oct 5 -11	**Analysis and Design:** Built the database and application structures (MySQL tables, data flow, etc.).
Elaboration	8: Oct 12 -18	**Analysis and Design:** Built the application connectivity to API systems and database (test and connect to external servers).
Elaboration	9: Oct 19 -25	**Analysis and Design:** Started the core coding to develop client requirements (Track, Search and Locate functions).
Elaboration	10: Oct 26 –Nov 1	**Analysis and Design:** Continued coding the core functionality (Track, Search and Locate functions).
Construction	11: Nov 2 - 8	**Implementation:** Completed core functionality. Began coding secondary functionality (schedule via .pdf, via text message, phone calling, Facebook page, Twitter page).
Construction	12: Nov 9 - 15	**Implementation | Test:** Continued coding secondary functionality. Began intensive testing of core functionality and basic testing of secondary functionality.
Construction	13: Nov 16 - 22	**Implementation | Test:** Completed Metro Live Beta version. Intensive testing continued on both core and secondary functionality.
Construction	14: Nov 23 - 29	**Implementation | Test:** Continued intensive testing. Created all user documentation (Power Point, user and developer instruction manuals, and all system interaction diagrams).
Transition	15: Nov 30 - Dec 6	**Test | Configuration and Change Management:** (Tentative) Final testing and bug fixes.
Transition	16: Dec 7 - 13	**Deployment:** (Tentative) Metro Live 1.0 release.
Transition	Beyond	**Configuration and Change Management:** (Tentative) General maintenance and continual bug fixes.

Table 5: Phase Plan

3.3.2 Iteration Objectives

Referring to **Table 4** under *3.2 Iteration Plan*, the iteration objectives are:

Inception 1 - Iteration 1:

1) Find like-minded team members to form a cohesive and efficient group that will create a successful project.
2) Choose the topic of our project, agree on team member roles, and agree on the overall scope / contract of the project.
3) Develop the first 2 most important use cases.

Elaboration 1 - Iteration 2:

1) Develop the functional and non-functional requirements of the project.
2) Develop the preliminary use cases that describe the user / system interactivity within the application.
3) Create our initial plan for developing the technical aspects of the application (programming language, operating system, etc.).
4) Build the preliminary UI template.
5) Build the application interface based on the prototype sketches.
6) Write the CSS code that controls the appearance of the user interface.
7) Build the database which will house the user accounts.
8) Build the application data structures to hold the GPS data.

Elaboration 2 - Iteration 3:

1) Finalize the functional and non-functional requirements.
2) Finalize the tentative use cases.
3) Finalize plan for developing the technical aspects of the application (programming language, operating system, etc.).
4) Begin coding the client's functional requirements.
5) Build the core application functions: Track, Search, and Locate.
6) Finish building the database.
7) Build the functionality to allow for connecting the Transport Authority's API.
8) Test connectivity between the application and the external GPS servers.
9) Modify the UI according the new client specifications.

Construction 1 – Iteration 4:

10) Complete the core functionality.
11) Begin coding secondary functionality: Schedule via .pdf and text message, phone calling, Facebook page, Twitter page.
12) Begin intensive testing of core functionality and basic testing of secondary functionality.

Construction 2 – Iteration 5:

13) Complete Metro Live Beta version.
14) Continue intensive testing on both core and secondary functionality.

Construction 3 – Iteration 6:

15) Continue intensive testing.
16) Create all user documentation: Power Point, user and developer instruction manuals, and all system interaction diagrams.

Transition 1 – Iteration 7:

17) (Tentative) Complete final testing and bug fixes.
18) (Tentative) Launch Metro Live version 1.0.

Transition 2 – Iteration 8:

19) (Tentative) General maintenance and continual bug fixes.

3.3.3 Releases

(Tentative) Metro Live version 1.0 – December 13th, 2013.

3.3.4 Project Schedule and Timeline Summary

The projected phases of our software process are in the following diagram. The sizes of each individual chart are relative and an estimate.

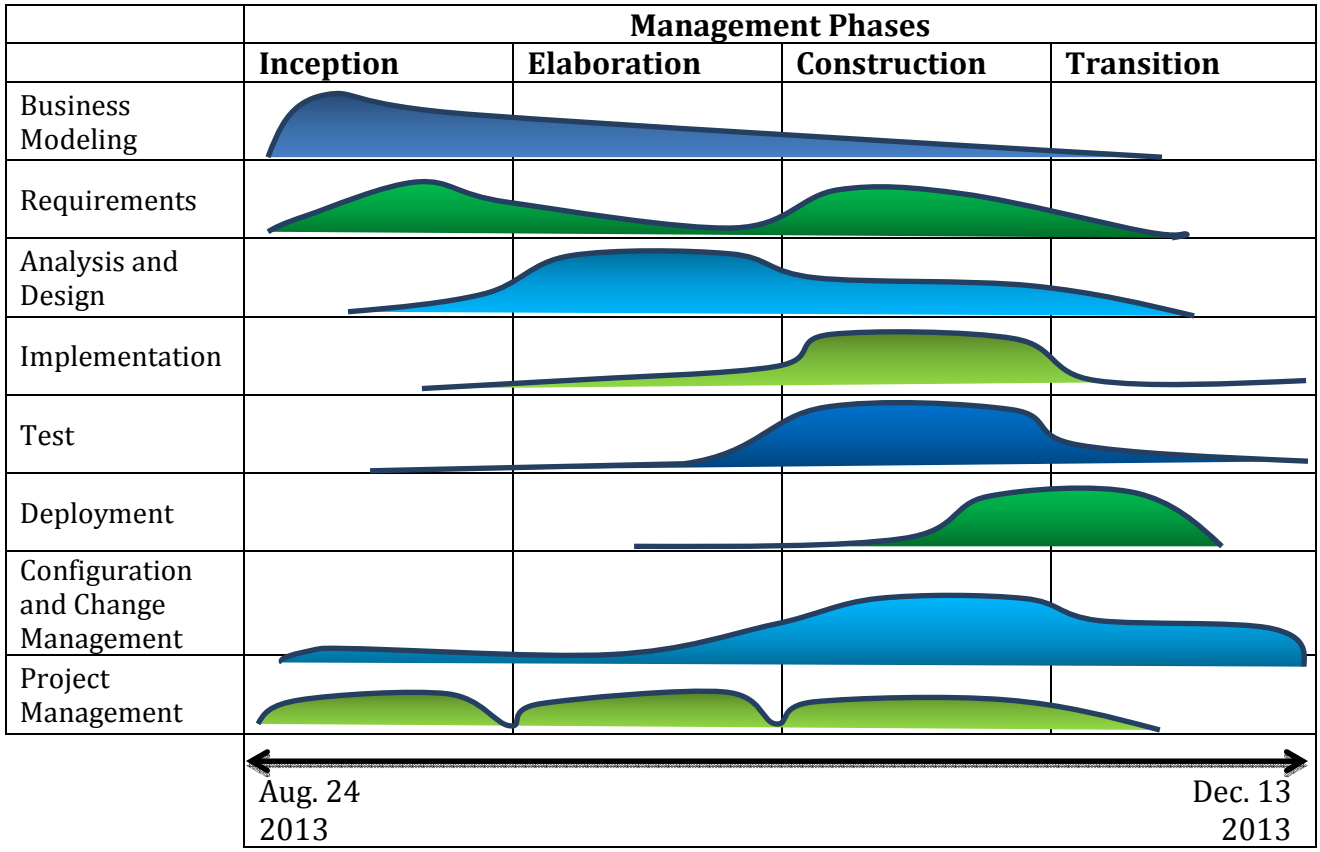


Table 6 – Project Schedule and Timeline

3.3.5 Project Resourcing

Each team member has specific roles and skills that will be used as resources for this project:

Jeff has general programming experience and experience leading small groups. His role as Project Manager will be to delegate tasks, assume overall responsibility to the client for the success (or failure!) of the application, resolve any issues within the group, and lead the group toward the goal of creating a successful application.

Sultan has extensive programming experience in both desktop and web based applications. His role as Lead Programmer and Implementer will be the development of specific elements according to designs, requirements, and the

architecture of the core project plan. He will lead the coding effort and delegate any additional programming tasks to Amir as needed.

Amir has experience in web and mobile programming, as well as graphic design. His role as Integrator will be to integrate the individual components of the project (source code, graphic design, UI) together into the final application. He will also be responsible for the graphics (splash screen, icons, presentations, etc.) of the application and assist Sultan with subordinate programming tasks.

Seyed has extensive experience in database and .NET programming, as well as in web design. He has a dual purpose role as both Tester and Analyst. He will be responsible for testing, verifying, and performing quality control on the software. He will also track statistical data to monitor the progress of the project as a whole.

Pruthivin has experience in database and multi-platform programming, as well as in web design. He also has a dual purpose role as both Tester and Analyst. Along with Seyed, he will be responsible for testing, verifying, and performing quality control on the software. He will also create all user documentation that will accompany the software (user manuals, performance charts, etc.).

3.3.6 Budget (If Applicable)

N/A

3.4 Project Monitoring and Control

3.4.1 Budget Control Plan (If Applicable)

N/A

3.4.2 Quality Control Plan

As the Tester/Analysts of the group, Seyed and Pruthivin will lead the Quality Control effort. They will collaborate together and perform multiple tasks such as testing, verifying, and validating the software throughout the development process. They will report any findings to the project manager and lead programmer so that any issues can be addressed as they arise. They will hold the group's goal of producing quality software as the highest priority.

3.4.3 Reporting Plan (If Applicable)

N/A

3.4.4 Measurement Plan

Seyed and Pruthivin will also lead the Measurement Plan effort. They will work in collaboration and monitor the progress of the group. As analysts, they will not only maintain documentation pertaining to the software itself, but also maintain a record of work flow for each iteration. They will then compare records of different iterations to determine whether the project is progressing efficiently and meeting schedule checkpoints. Any deviation from the project schedule shall be reported immediately to the project manager so that he may resolve the issue.

3.5 Risk Management Plan

As Project Manager, Jeff will be responsible for assessing and handling any risks that the project team may encounter. The following risks are common to any group environment:

Managerial Risks

1) The risk that some group members may not contribute evenly to the group.

Solution: Jeff will be diligent in the delegation of work tasks. Tasks will be assigned to group members based on their skill set and current workload. Ideally, each group member will have an equal share of tasks to be completed in a timely manner. Team members are encouraged to work closely together and help each other for the greater good of the group.

2) The risk that a member may fall ill, or leave the group due to extenuating circumstances and be unable to fulfill their duties within the group.

Solution: Redundancy. Each member of our group has the skills to complete work in other disciplines. For example, if Sultan were to leave the group unexpectedly, Amir would be able to fulfill the role of Lead Programmer due to his experience with JavaScript. Seyed would then be able to help with the graphic design and Jeff would be able to then help with the documentation. Pruthivin would be able to support all areas as well because of his extensive skill set.

3) The risk that our group may miss the project deadline.

Solution: Our team has excellent Quality Control and Measurement plans in place to assure that this does not become an issue. Seyed and Pruthivin's main role as Analysts

is to report any deviations from the project schedule immediately to the project manager.

Technical Risks

1) The risk that the project scope is to vast.

Solution: Our group purposely erred on the side of caution when establishing the scope of our project. We communicated to our client that it would be best if the project concentrated solely on the core functionality of a typical bus tracking application. Additional features may be added if time and budget allows, however, they are not currently within the scope, and would have to be negotiated with the client at another time.

2) The risk that the software may not work cross platform.

Solution: Ideally, our application would be able to run on multiple mobile operating systems. However, we decided that due to the limited time frame allotted for this project, it would be best to restrict the software to only the Android operating system. The ability to run on iOS in the future may be an additional feature that the client would have to pay for.

3) The risk that the software does not meet the client's expectation or does not function properly.

Solution: We have chosen an Agile software development methodology, specifically the Unified Process, for our project's development cycle. This method stresses the importance of client involvement and mini-checkpoints called iterations. These iterations allow the group and the client to gauge the progress of the project, as well as correct any small issues before they become large ones. This process, in conjunction with a solid working relationship between the client and the development team, will result in a fully functional, quality product delivered on time.

Unforeseen Risks

With any project comes the potential for unforeseen risks. When such risks should arise, they will be evaluated on a case-by-case basis by the project manager and the client. The risk will be prioritized (as acceptable or unacceptable), evaluated with the input from the development team, and then decided upon. One major benefit of using an Agile development process is that the use of iterations mitigate any potential detriment that an unforeseen risk may have to the project.

4. Design Description

4.1 Product Perspective

The development team would like to make Metro Live™ available on the following mobile operating systems:

- **iOS** – Apple's iPhone provides a rich environment for interactive applications. In the future, the development team would like to port the Android version of Metro Live over to iOS.

4.2 Product Features

- **Tracking a Bus** - The main feature of the Metro Live™ application is the ability to track buses, in real time, within the city limits of Los Angeles. The user will view a Google-like map of Los Angeles and be able to see all of the currently operating buses. After selecting a specific bus to track, the application will then continually update the bus's current location on the map. It will also display useful information including the bus's distance to the user, next stop, and estimated arrival time.

- **Finding a Bus** - The user will be able to find a bus according to a specific route number and destination. The application displays a list of all of the currently active buses which match the user's search criteria. Detailed information about each bus (ETA, location, on-time status, etc.) is also displayed. After the user selects a bus from the search results, the map is displayed showing its current location.

- **Favorites List** – The application provides the user with ability to add a bus or a specific route to their favorites list, allowing convenient access to this information in the future. The user must create a login account to access this feature.

The user also has 2 alternative ways to receive bus tracking information:

- **By Phone** - The user will be able to phone a Los Angeles Metropolitan Transportation Authority representative, via the application, to get specific bus information in the event that mobile internet connectivity is unavailable (e.g. roaming coverage).

- **By Text** - The user will be able to send a text message with the bus route number, via the application, to the Los Angeles Metropolitan Transportation Authority SMS system. The LAMTA's automated SMS system will respond with all the appropriate information regarding that route number.

4.3 Use Cases

4.3.1 UC1 – Create an Account

UC [001]: Create an Account
Goal in Context: The user is able to create an account.
Scope: The System.
Level: Primary Task.
Precondition: The user has an e-mail account, a cellular phone with access to the Internet, and text message capabilities.
Success End Condition: the user has created an account.
Failed End Condition: the user is unable to create an account.
Primary Actors: the user.
Trigger: The user wants to create an account.

Main Success Scenario:

1. The user selects “Login” from the main menu.
2. The system displays a login box with the option to “Register.”
3. The user selects “Register.”
4. The system displays a form prompting the user for information.
5. The user enters his personal information (e-mail, username, password, phone number).
6. The user selects “Register.”
7. The system adds the new account.
8. The system displays the message “Account created successfully.”

Extensions: NONE.
6a: The user has entered an invalid or blank username.
    6a.1: The system displays the message "Please enter a valid Username.”
        6a.1.1: The user enters a valid username.
6b: The user has entered a username that already exists.
    6b.1: The system displays the message “Username already exists!”
        6b.1.1: The user chooses a different username.
6c: The user has entered an invalid or blank password.
    6c.1: The system displays the message "Please enter a valid Password.”
        6c.1.1: The user enters a valid Password.
6d: The user has entered an invalid or blank e-mail.
    6d.1: The system displays the message "Please enter a valid E-mail address.”
        6d.1.1: The user enters a valid e-mail address.
6e: The user has entered an e-mail address that already exists.
    6e.1: The system displays the message “E-mail address already exists!”
        6e.1.1: The user enters a different e-mail address.
6f: The user has entered an invalid or blank phone number.
    6f.1: The system displays the message "Please enter a valid phone number.”
        6f.1.1: The user enters a valid phone number.

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.2 UC2 - Login

UC [002]: Login
Goal in Context: The user logs in successfully.
Scope: The System.
Level: Primary Task.
Precondition: UC001. The user has a cellular phone with access to the Internet.
Success End Condition: The user is logged in.
Failed End Condition: The user is unable to login to the system.
Primary Actors: The user.
Trigger: The user wants to login to the system.

Main Success Scenario:
1. The user selects "Login" from the main menu.
2. The system displays a login box.
3. The user enters a username and password.
4. The user selects "Login."
5. The system displays "Welcome, [username]."

Extensions:
4a: The username or password is invalid.
    4a.1: The system displays the message "Invalid Username or Password."
        4a.1.1: The user enters a valid username and password.
4b: The account exists, but has not been activated.
    4b.1: The system asks the user for their activation number.
        4b.1.1: The user enters a valid account activation number.
            4b.1.1.1: The user selects "Activate."
            4b.1.1.2: The system displays "Welcome, [username]."
            4b.1.1.3: The system returns to main menu.
        4b.1.2: The user enters an invalid account activation number.
            4b.1.2.1: The system displays "Wrong Number."
            4b.1.2.2: The user enters a valid account activation number.

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.3 UC3 – Check Active Routes/Buses

UC [003]: Check Active Routes/Buses
Goal in Context: The user will be able to see the number of active routes and buses.
Scope: The System.
Level: Secondary Task.
Precondition: The user has a cellular phone with access to the Internet.
Success End Condition: The user will see the number of active routes and buses.
Failed End Condition: The user will be unable to see the number of active routes and buses.
Primary Actors: The user.
Trigger: The user wants to see the number of active routes and buses.

Main Success Scenario:
1. The user selects "HOME" from the main menu.
2. The system displays the number of currently active routes and buses.

Extensions: N/A

Related Information: NONE.
Priority: Medium
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.4 UC4 – Track a Bus

UC [004]: Track a Bus
Goal in Context: The user can track a selected bus's current location.
Scope: The System.
Level: Primary Task.
Precondition: The user has a cellular phone with access to the Internet. The user knows the bus number he wishes to track.
Success End Condition: The user is able to see the current location of the bus on the mini map.
Failed End Condition: The user is unable to see the current location of the bus on the mini map.
Primary Actors: The user.
Trigger: The user wants to track a bus.

Main Success Scenario:
1. The user selects "Map" from the main menu.
2. The system displays the transit map.
3. The system displays a bus number input box.
4. The user inputs the desired bus number.

5. The user selects “Track.”
6. The system updates the map with the current location of the bus.

Extensions:
5a: The bus number is wrong.
5a.1: The system displays the message “The bus number does not exist.”
5a.1.1: The user enters a valid bus number.
5b: The bus number is correct but not in use.
5b.1: The system displays the message “The bus number is correct but not in use.”
5b.1.1: The user enters a different bus number.
5c: The user is not logged in.
5c.1: The system displays the message “The bus number does not exist.”
5c.1.1: The user enters a valid bus number.

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.5 UC5 – Find a Bus

UC [005]: Find a Bus
Goal in Context: The user searches for a bus from a directory.
Scope: The System.
Level: Primary Task.
Precondition: The user has a cellular phone with access to the Internet.
Success End Condition: the user has a list of bus destinations that match his search criteria.
Failed End Condition: the user is unable to find a bus that matches his search criteria.
Primary Actors: the user.
Trigger: The user wants to search for a bus.

Main Success Scenario:
1. The user selects the “Find Bus” from the main menu.
2. The system displays “Route Number” list.
3. The user selects a “Route Number” from the list.
4. The system displays “Destination” list.
5. The user selects a “Destination” from the list.
6. The system shows a list of buses that match the search criteria.

Extensions:
3a: There are no buses currently available for the selected route.
3a.1: The system displays the message "No buses available for this route.”
3a.1.1: The user selects a different route.

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.6 UC6 – Add a Favorite

UC [006]: Add a Favorite
Goal in Context: The user is able to add a bus to his favorites list.
Scope: The System.
Level: Primary Task.
Precondition: UC001, UC005.
Success End Condition: the user adds a bus to his favorites list.
Failed End Condition: the user is unable to add a bus to his favorites list.
Primary Actors: the user.
Trigger: The user wants to add a bus to his favorite list.

Main Success Scenario:

1. The user selects “Add to Favorites” icon.
2. The system adds the bus to the user’s favorites list.
3. The system displays “The bus has been added!”

Extensions:
1a: The bus is already in the list.
    1a.1: The system displays the message "The bus is already in your favorites list.”

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.7 UC7 – Delete a Favorite

UC [007]: Delete Favorite
Goal in Context: The user is able to delete a favorite from his favorites list.
Scope: The System.
Level: Primary Task.
Precondition: UC001, UC006.
Success End Condition: the user deletes a favorite bus.
Failed End Condition: the user is unable to delete a favorite bus.

Primary Actors: the user.
Trigger: The user wants to delete a favorite bus.

Main Success Scenario:

1. The user selects “Favorites” icon from the main menu.
2. The system displays the favorites list.
3. The user selects the “Delete Favorite” icon of a particular favorite.
4. The system displays a confirmation box that says “Are you sure you want to delete the bus?”
5. The user selects “OK.”
6. The system displays “Bus Deleted!”
7. The system updates the favorites list showing that the bus has been deleted.

Extensions:
5a: The user selects “Cancel.”
5a.1: The system returns to favorites list.

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.8 UC8 – Display Favorites

UC [008]: Display Favorites
Goal in Context: The user is able to display his favorites list.
Scope: The System.
Level: Primary Task.
Precondition: UC001.
Success End Condition: the user displays his favorites list.
Failed End Condition: the user is unable to display his favorites list.
Primary Actors: the user.
Trigger: The user wants to display his favorite list.

Main Success Scenario:

1. The user selects “Favorites” icon from the main menu.
2. The system displays the favorites list.

Extensions:
2a: There are no favorites in the favorites list.
2a.1: The system displays the message "Sorry, you have no favorite buses. To add, go to Find Bus.”

Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.9 UC9 – Schedule via .pdf

UC [009]: Schedule via .pdf
Goal in Context: The user receives the schedule of the specific route in .pdf format.
Scope: The System.
Level: Secondary Task.
Precondition: The user has a cellular phone with access to the Internet.
Success End Condition: The user gets the schedule information.
Failed End Condition: The user is unable to get the schedule information.
Primary Actors: The user.
Trigger: The user wants to get the schedule of the desired route.

Main Success Scenario:
1. The user selects "Schedule."
2. The user selects the route number from the drop down list menu.
3. The user selects the .pdf option below the drop down menu to receive the PDF file of the route's schedule.
4. The system displays the schedule in the PDF format.

Extensions: NONE.

Related Information: NONE.
Priority: Medium.
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.10 UC10 – Schedule via Text Message

UC [0010]: Schedule via Text Message
Goal in Context: The user receives text messages displaying route information via SMS.
Scope: The System.
Level: Secondary Task.
Precondition: None.

Success End Condition: The user is able to view the bus schedule through text messages.
Failed End Condition: The user is unable to view the bus schedule through text messages.
Primary Actors: The user.
Trigger: The user wants view the bus schedule through text messages.

Main Success Scenario:
1. The user selects "TEXT" from the main menu.
2. The system displays a message: "Type 'Metro' when sending SMS."
3. The user selects "OK".
4. The system navigates the user to their default SMS interface.
5. The user types "Metro" in the SMS Textbox area.
6. The user selects on the send button.
7. The system responds with a message that includes the number of available routes to choose from and 2 key letters to type in order to: either get help by typing "H," or get the route information info by typing "L".
8. The user types "H" to get help from system.
9. The user selects the send button.
10. The system shows the help options via text message.
11. The user types "L" to get the available routes.
12. The user selects the send button.
13. The system shows the number of available routes.

Extensions:
6a: The user has typed something other than acceptable keywords.
6a.1: The system displays an error message.
6a.1.1: The user types "HELP".
6a.1.1.1: The system displays help options.
6a.1.1.2: The user types the correct keyword.
9a: The user has typed something other than acceptable keywords.
9a.1: The system displays an error message.
9a.1.1: The user types "HELP".
9a.1.1.1: The system displays help options.
9a.1.1.2: The user types the correct keyword.
12a: The user has typed something other than acceptable keywords.
12a.1: The system displays an error message.
12a.1.1: The user types "HELP".
12a.1.1.1: The system displays help options.
12a.1.1.2: The user types the correct keyword.

Related Information: "L": number of routes, "H" and "HELP": Help options. "Metro": retrieves routes information from system via text messages.
Priority: Medium.
Super Ordinate USE CASE: NONE.

Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.11 UC11 – E-mail Feedback

UC [011]: E-mail Feedback
Goal in Context: The user e-mails any feedback, comments, or suggestions successfully.
Scope: The System.
Level: Secondary Task.
Precondition: The user has an e-mail account and a cellular phone with access to the internet.
Success End Condition: The user is able to e-mail the feedback.
Failed End Condition: The user is unable to e-mail the feedback.
Primary Actors: The user.
Trigger: The user wants to e-mail any feedback, comments, or suggestions.

Main Success Scenario:
1. The user selects "Mail" from the main menu.
2. The system displays "Please send us your feedback and suggestions."
3. The user selects "OK".
4. The system opens the preconfigured e-mail application.
5. The user fills out the subject line and composes an e-mail.
6. The user selects the "Send e-mail" option.
7. The system displays that the message has been sent successfully.

Extensions:
4.a: The system cannot open the preconfigured application for e-mail.
    4.a.1: The system displays the message, "Please configure your e-mail application."
        4.a.1.1.: The user configures the e-mail application.

Related Information: NONE.
Priority: Medium.
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.12 UC12 – Call Help Center

UC [012]: Call Help Center
Goal in Context: The user will be able to receive schedule help from a LAMTA representative.
Scope: The System.
Level: Secondary Task.

Precondition: The user has an active cellular phone.
Success End Condition: The user is able to call the help center.
Failed End Condition: The user is unable to call the help center.
Primary Actors: The user.
Trigger: The user wants to call the help center.

Main Success Scenario:
3. The user selects "CALL" from the main menu.
4. The system brings up the default call interface with the number "511" already entered.
5. The user selects the call option on the phone.
6. The user connects to LAMTA help center.

Extensions: N/A

Related Information: NONE.
Priority: Medium
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.13 UC13 – View Facebook Page

UC [013]: View Facebook Page
Goal in Context: The user is able to view the Metro Live Facebook page.
Scope: The System.
Level: Secondary Task.
Precondition: The user has a cellular phone with access to the internet.
Success End Condition: The user views the Metro Live Facebook page.
Failed End Condition: The user is unable to view the Metro Live Facebook page.
Primary Actors: The user.
Trigger: The user wants to view the Metro Live Facebook page.

Main Success Scenario:
1. The user selects "Facebook" from the main menu.
2. The system displays the Metro Live Facebook page.

Extensions:
N/A

Related Information: NONE.
Priority: Low.
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.14 UC14 – View Twitter Page

UC [014]: View Twitter Page
Goal in Context: The user is able to view the Metro Live Twitter page.
Scope: The System.
Level: Secondary Task.
Precondition: The user has a cellular phone with access to the internet.
Success End Condition: The user views the Metro Live Twitter page.
Failed End Condition: The user is unable to view the Metro Live Twitter page.
Primary Actors: The user.
Trigger: The user wants to view the Metro Live Twitter page.

Main Success Scenario:
3. The user selects "Twitter" from the main menu.
4. The system displays the Metro Live Twitter page.

Extensions:
N/A

Related Information: NONE.
Priority: Low.
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.3.15 UC15 - Logout

UC [015]: Logout
Goal in Context: The user logs out successfully.
Scope: The System.
Level: Primary Task.
Precondition: UC002.
Success End Condition: The user is logged out of the system.
Failed End Condition: The user is unable to logout of the system.
Primary Actors: The user.
Trigger: The user wants to logout of the system.

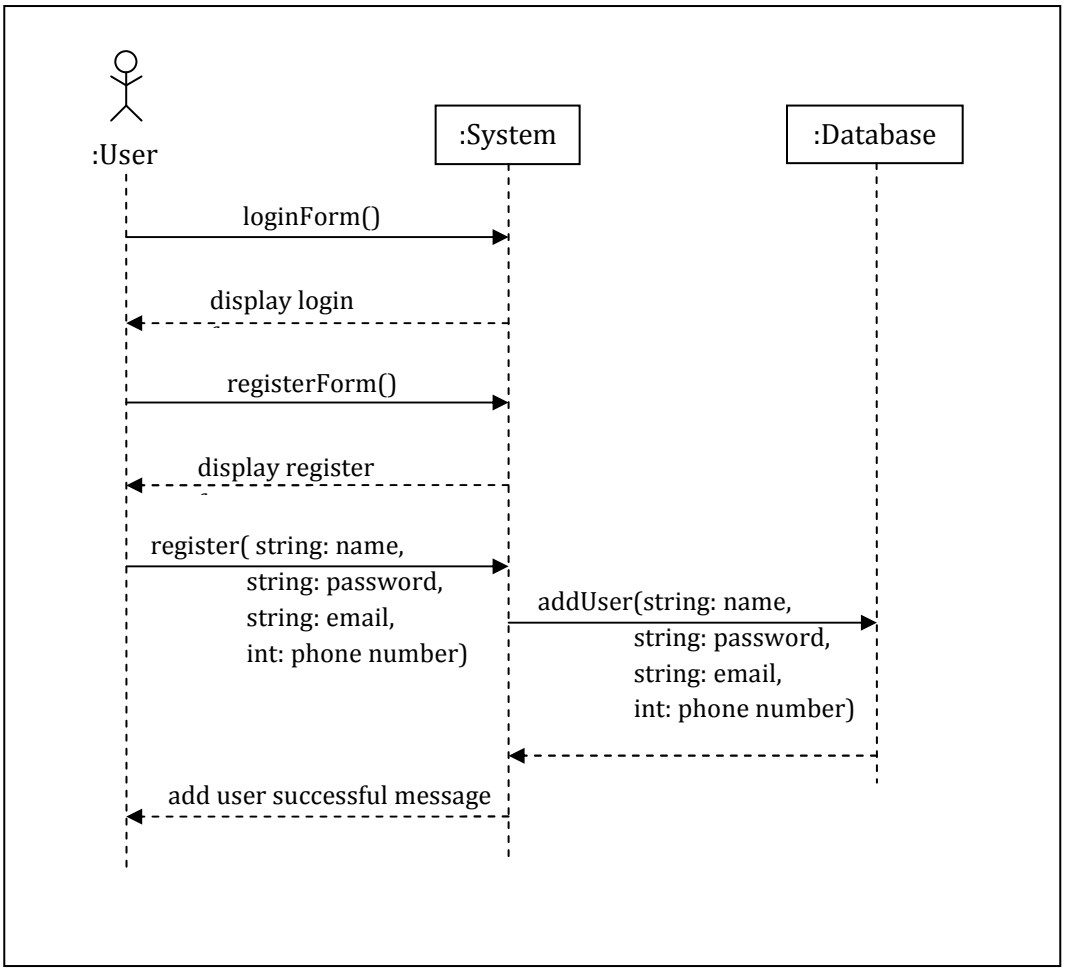
Main Success Scenario:
5. The user selects "Logout" from the main menu.
6. The system displays "Logout successful."

Extensions:
N/A

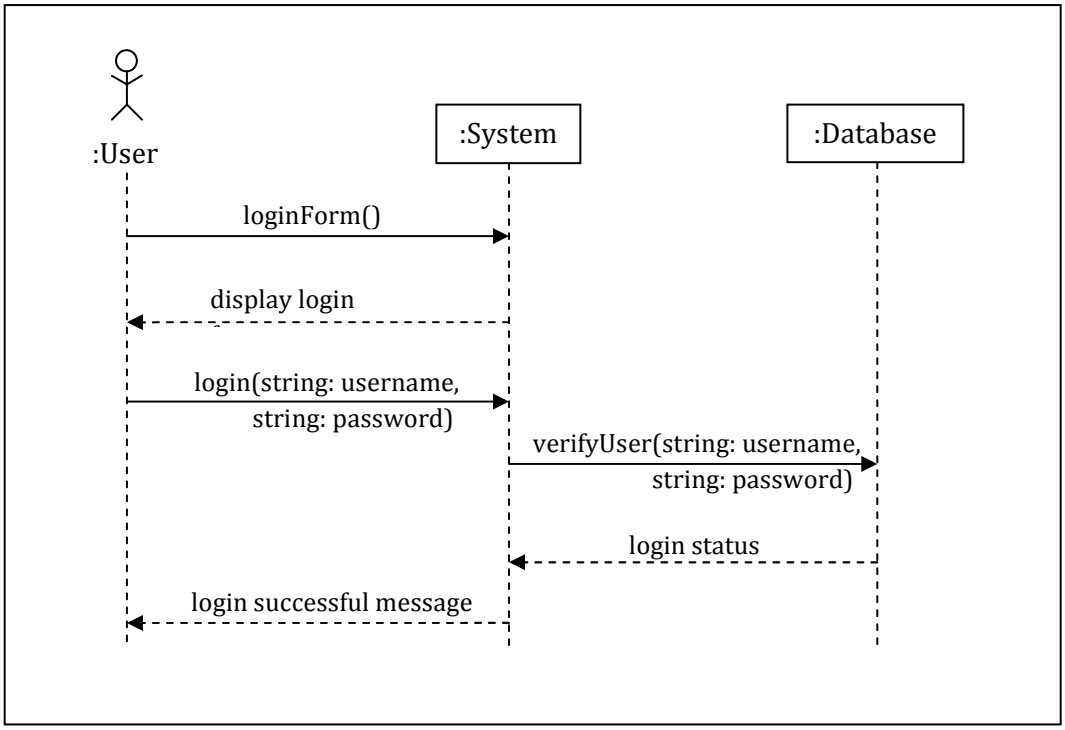
Related Information: NONE.
Priority: High
Super Ordinate USE CASE: NONE.
Secondary Actor: Technical Support.
Open Issue: NONE.

4.4 System Sequence Diagrams

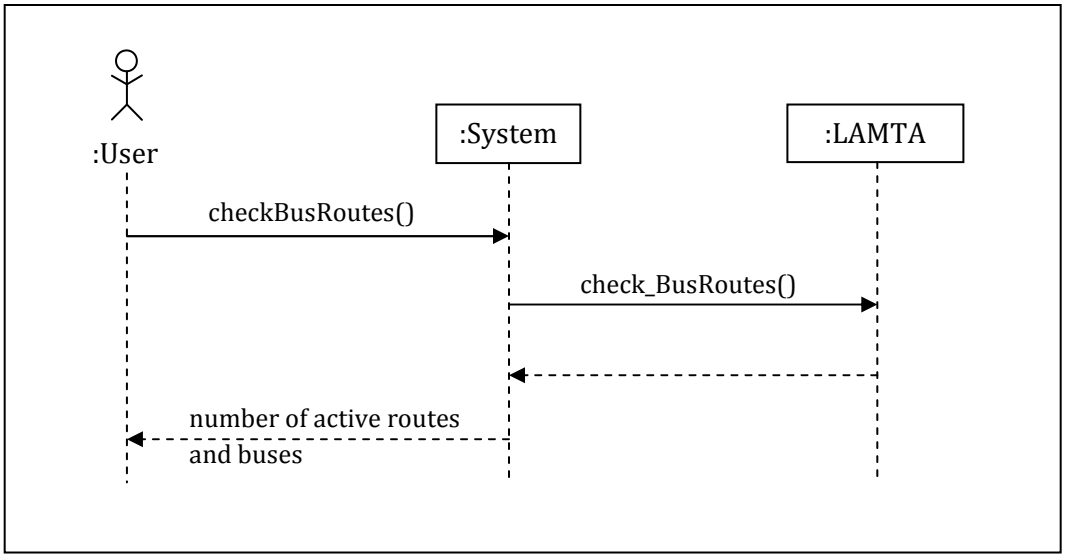
UC001 – Create Account



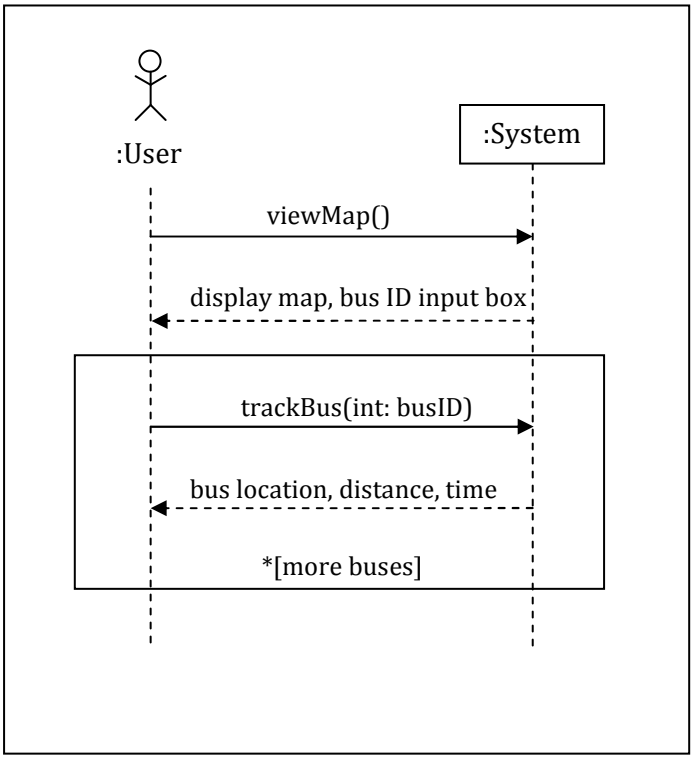
UC002 – Login



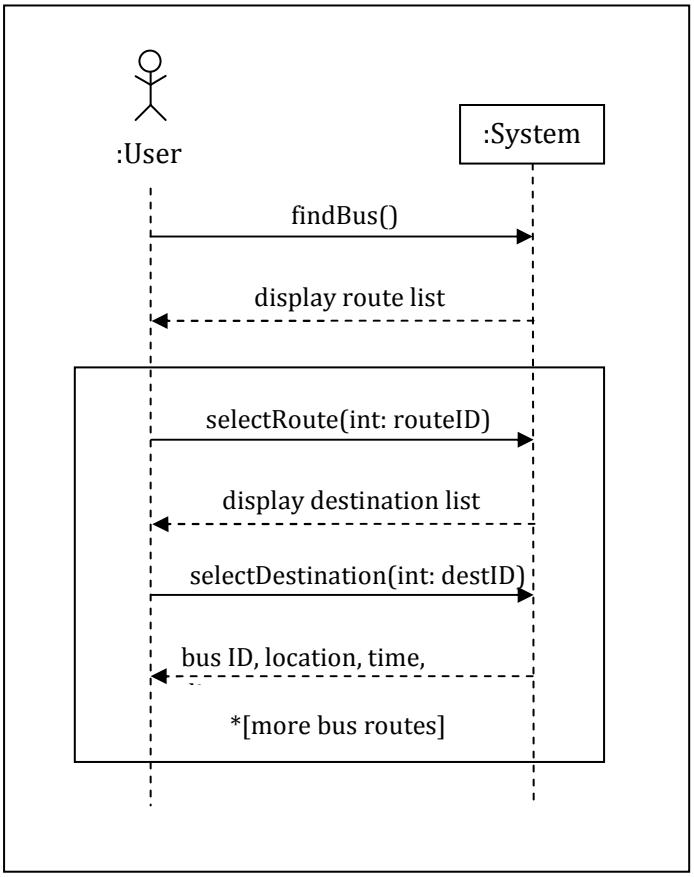
UC003 – Check Active Routes/Buses



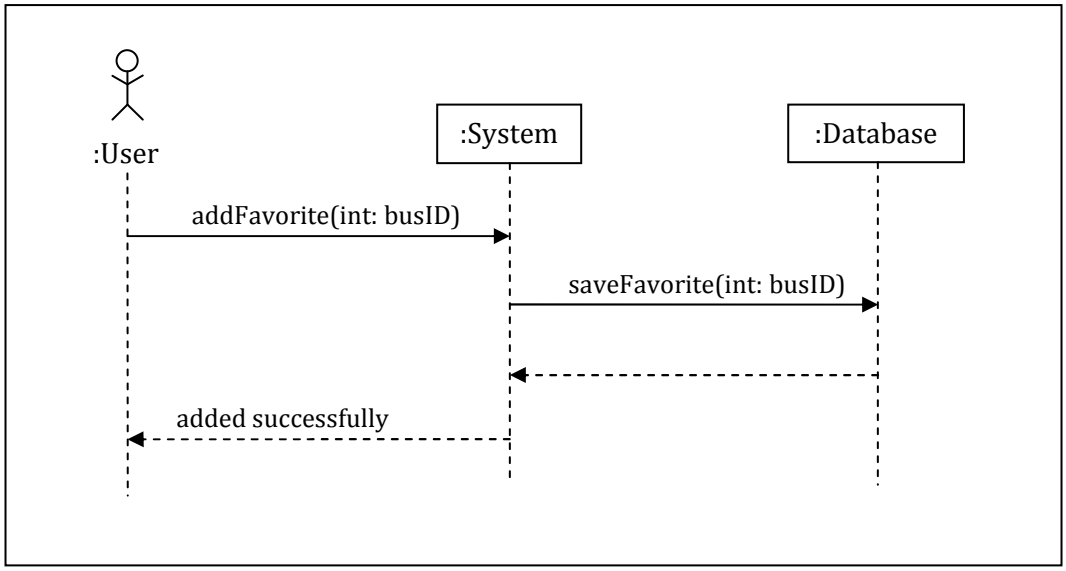
UC004 – Track a Bus



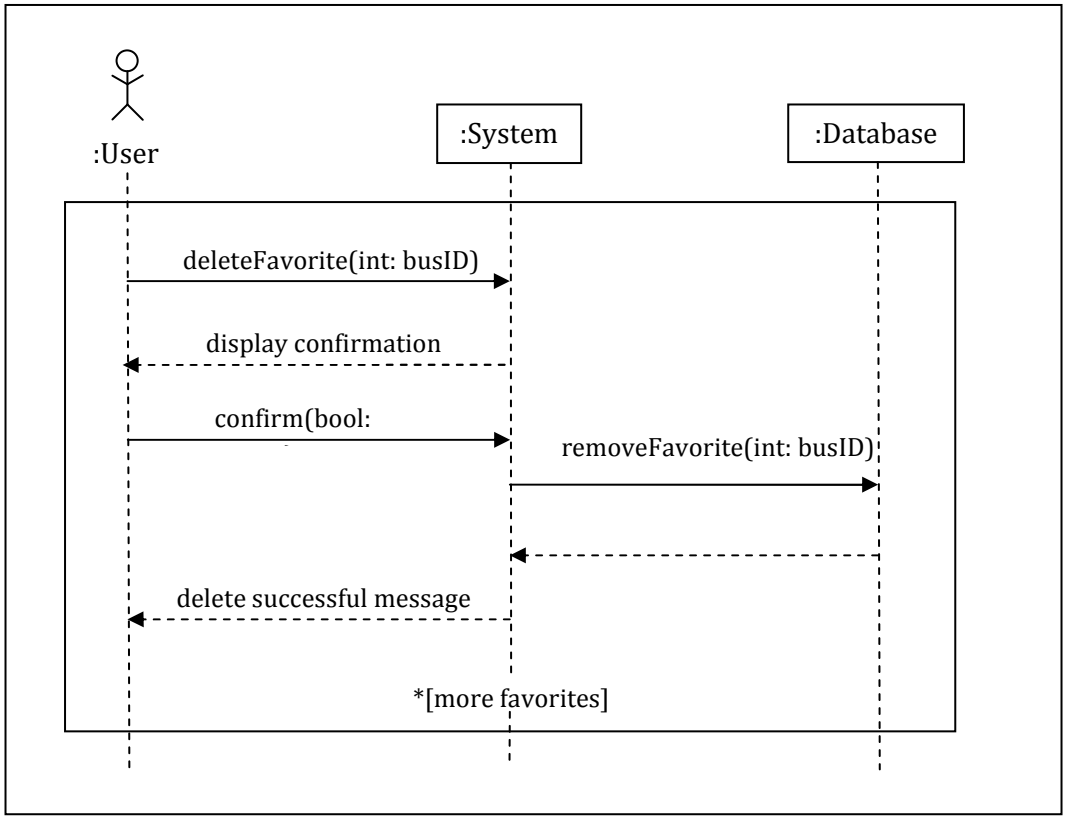
UC005 – Find a Bus



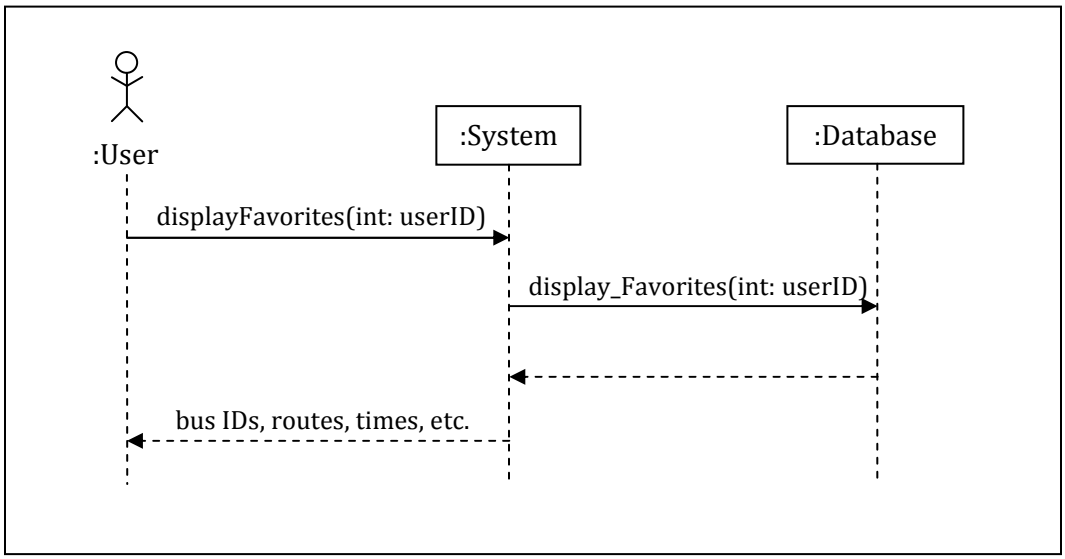
UC006 – Add a Favorite



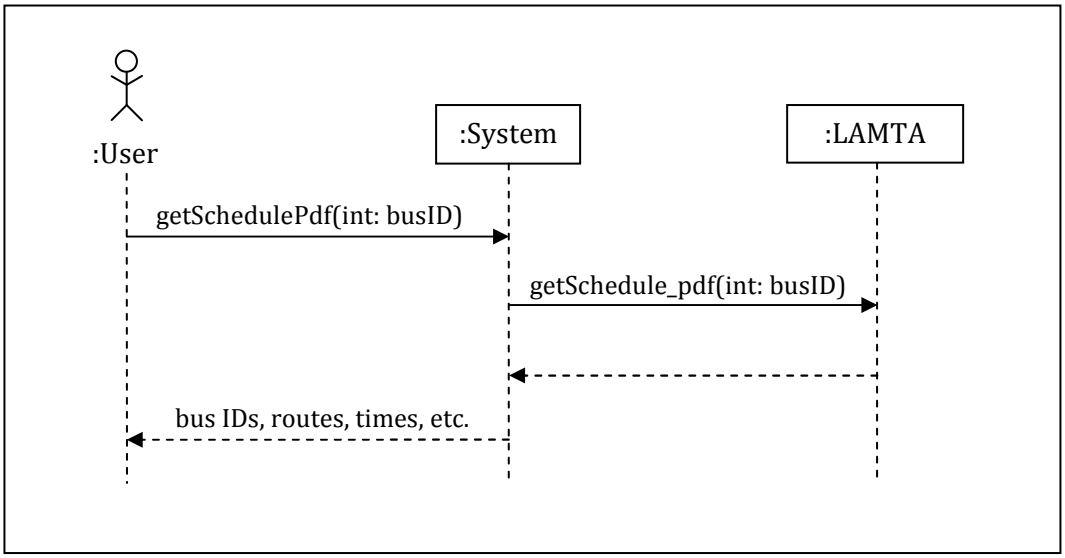
UC007 – Delete a Favorite



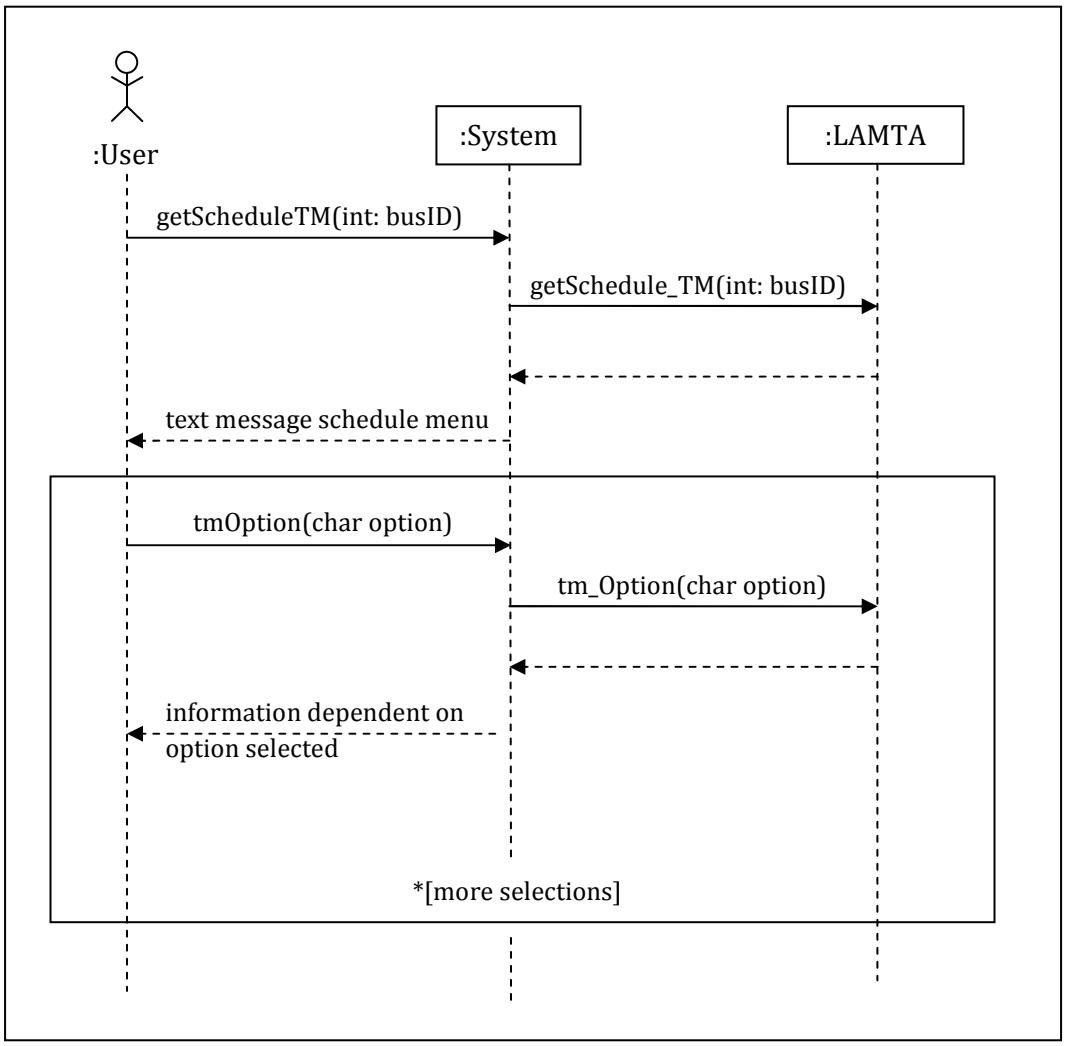
UC008 – Display Favorites



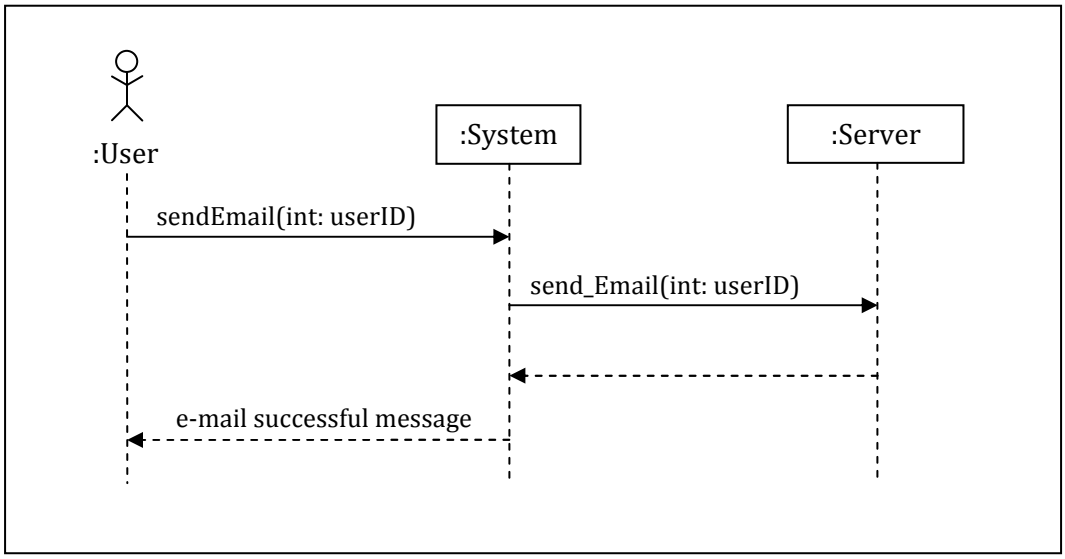
UC009 – Schedule via .pdf



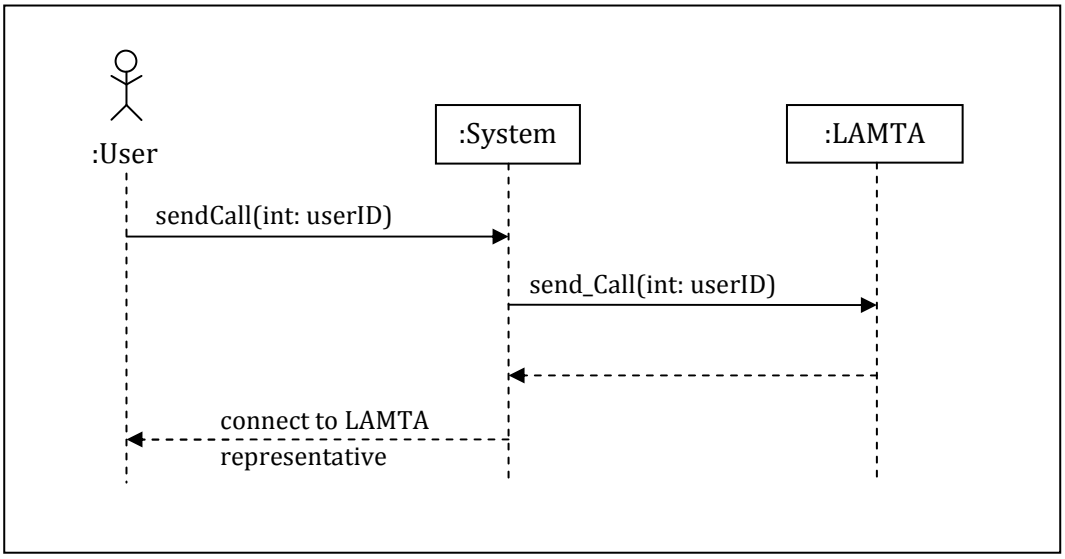
UC010 – Schedule via Text Message



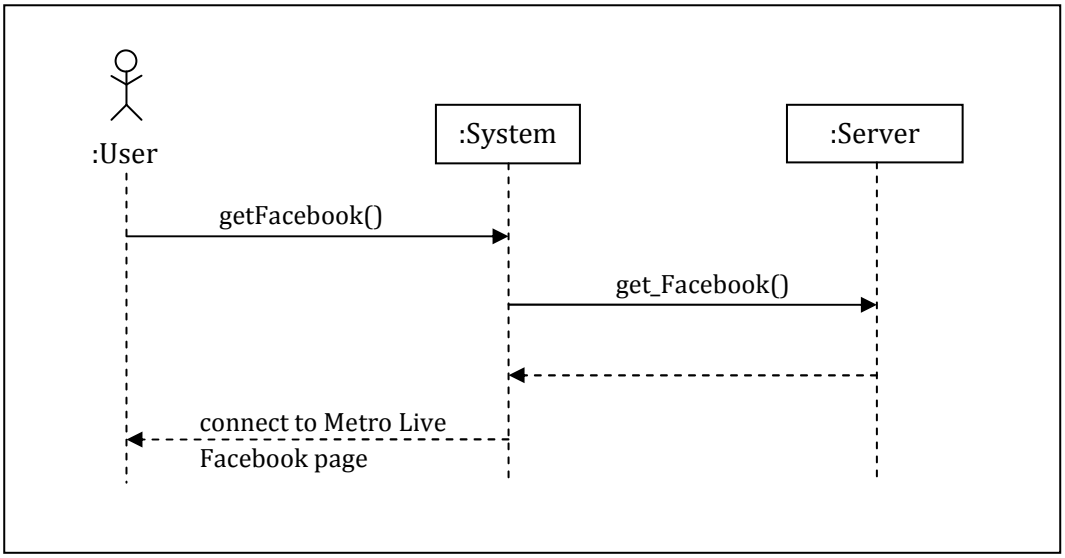
UC011 – E-mail Feedback



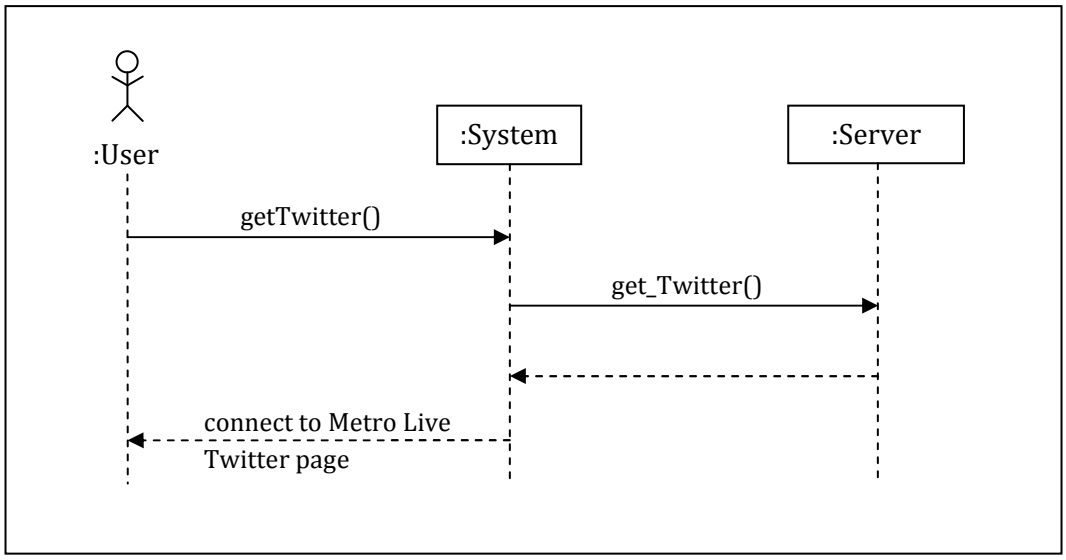
UC012 – Call Help Center



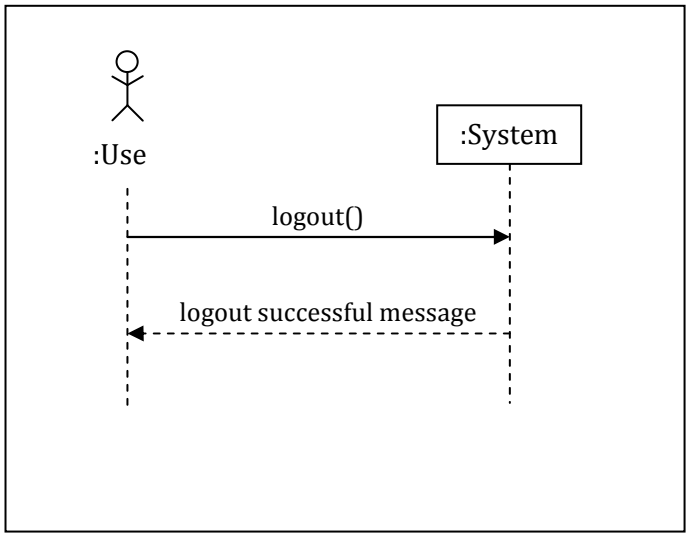
UC013 – View Facebook Page



UC014 – View Twitter Page



UC015 – Logout



4.5 Domain Model Diagram

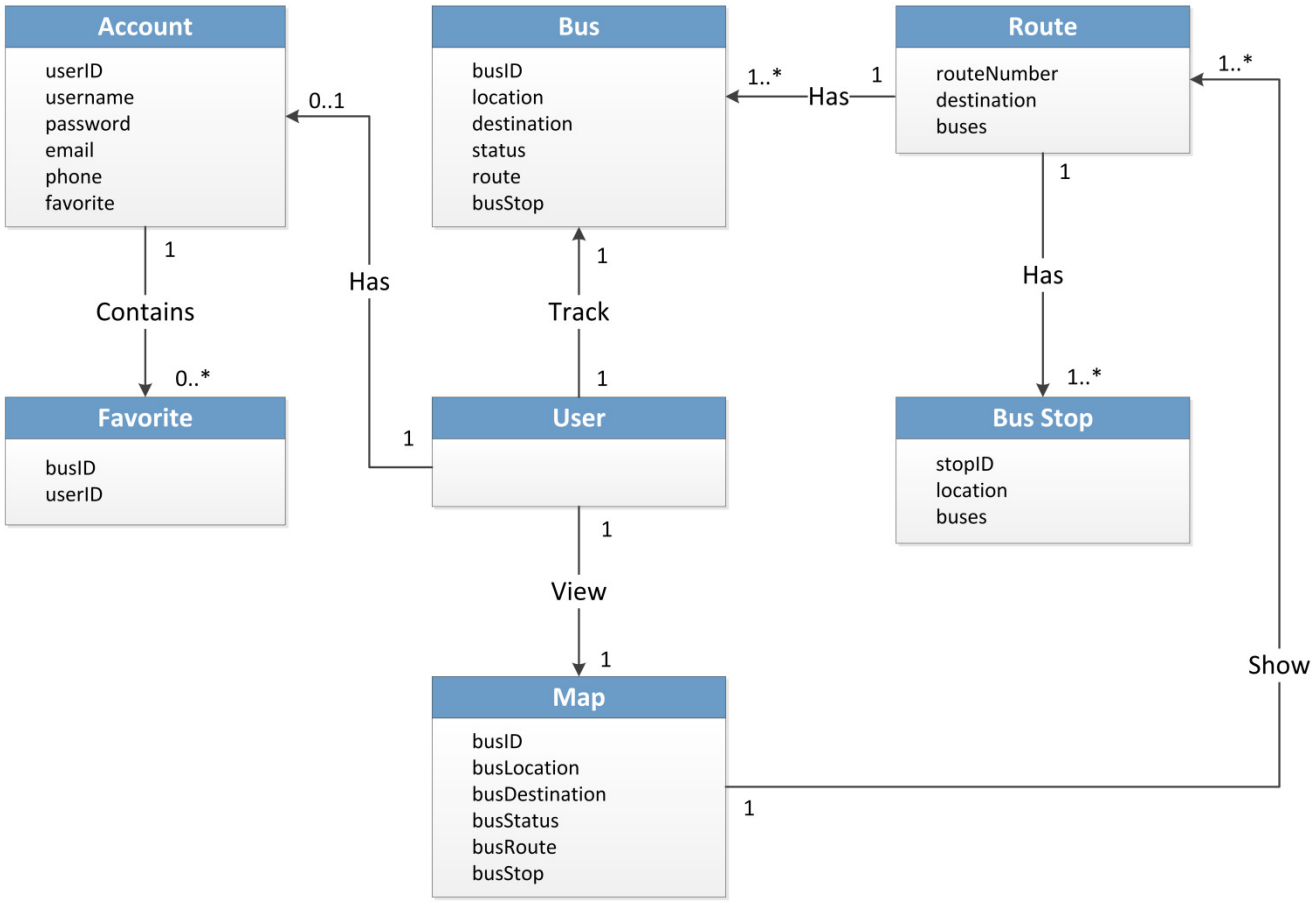


Figure 2: Domain Model

4.6 System Class Diagram

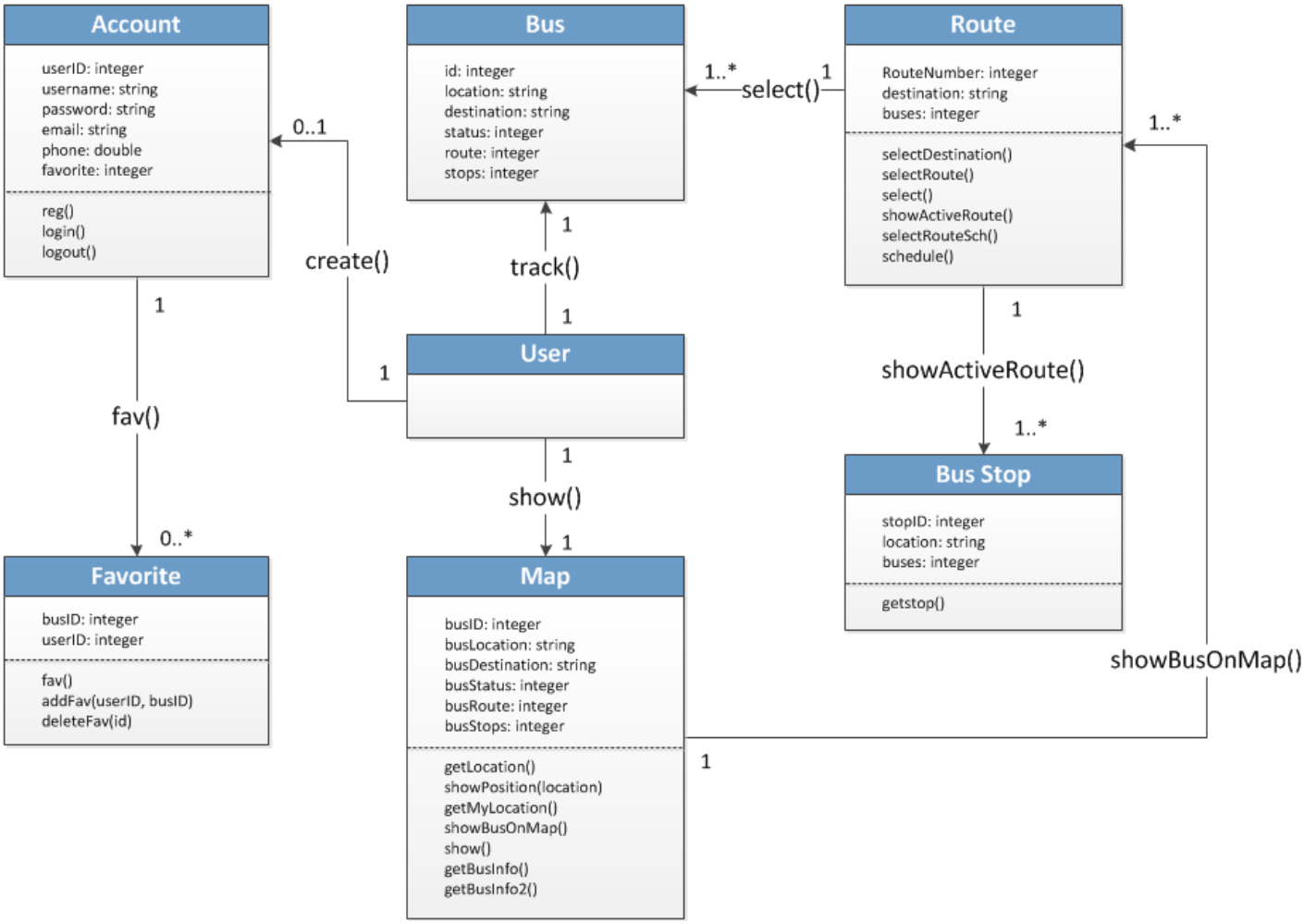


Figure 3: System Class Diagram

4.7 Database Information

4.7.1 Database Tables

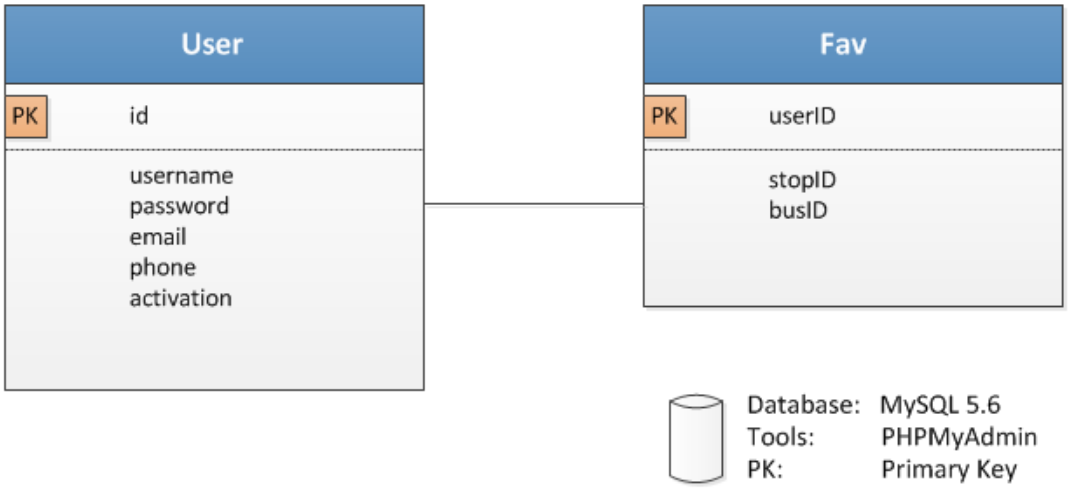


Figure 4: Database

5. Test and Integration (Plan & Results)

5.1 Test Cases

Test 1: Create an Account

Test Case ID: 001
Unit to Test: Creating an account.

How the system works:

The user is required to create an account before using the System for the first time. The user fills in the required information according to the following:

- The entered username shall not be taken.
- The password shall be at least 6 characters long.
- The phone number shall be written in (###-###-####) format.
- The e-mail address shall be written in (address@mail.com) format.

Assumption: N/A

Test Scenario 1:
The user entered the phone number 9161111111.

Expected Results:
The system detects the error and displays an error indication along with a message of the correct way to write the phone number.

Actual Results:
All the expected results were met.

Test Scenario 2:
The user entered a password with only 4 characters.

Expected Results:
The system detects the error and displays an error indication along a message “The password shall be 6 characters long” error message.

Actual Results:
All of the expected results were met.

Test 2: Track a Bus

Test Case ID: 002
Unit To Test: Track a bus.

How the system works:
The user already has an account and is logged into the account. The user has the privilege of tracking a bus according to the route number and the destination.

Assumptions:
The user is already logged in.

Test Scenario:
The user wants to track a bus, the user selects the desired bus route number and destination from the drop down menu.

Expected Results:
The system displays all the possible buses in different timings in the selected route and destination. Once the user selects a bus to track, the system updates the map with the bus's current location.

Actual Results:
All of the expected results were met.

Test 3: Add Favorite

Test Case ID: 003
Unit To Test: Adding a bus to the user's favorites list.

How the system works:
The user already has an account and is logged into the account. After the user has selected a bus to track, they then have the option to add the route to their favorites list.

Assumptions:
The user is already logged in and they have tracked a bus.

Test Scenario:
The user clicks the "Add Favorite" star icon next to the desired route. The route is then added to the user's favorites list.

Expected Results:
The system displays the message "Added to Favorites!" and updates the user's favorites list with the newly added route.

Actual Results:
All of the expected results were met.

5.2 Test Results

Case ID	Case Type	Case Name	Action	Expected	Actual	Status
TC-001-A	Integration User Interface	Create an Account	The user entered the phone number 9161111111.	The system detects the error and displays an error a message of the correct way to write the phone number.	All of the expected results were met.	Pass
TC-001-B	Integration User Interface	Create an Account	The user entered a password with only 4 characters.	The system detects the error and displays an error message “The password shall be 6 characters long.”	All of the expected results were met.	Pass
TC-002	Integration User Interface	Track a Bus	The user wants to track a bus, the user selects the desired bus route number and destination from the drop down menu.	The system displays all of the current buses with the selected route and destination. Once the user selects a bus to track, the system updates the map with that bus’s location.	All of the expected results were met.	Pass
TC-003	Integration User Interface	Add Favorite	The user clicks the “Add Favorite” star icon next to the desired route. The route is then added to the user’s favorites list.	The system displays the message “Added to Favorites!” and updates the user’s favorites list with the newly added route.	All of the expected results were met.	Pass

Table 7: Test Results

6. Installation Instructions and User Documentation

6.1 Installation - Developer

6.1.1 Prerequisites - Developer

In order to install Metro Live as a developer you must have the following:

- Windows server (with Apache or IIS), UNIX server, or Linux server.
- Windows 7 or higher.
- SDK for JAVA.
- Android developer IDE (e.g. Eclipse)
- Database support (MySQL, Sql2008, etc.)
- Database management tool (e.g. PhpMyAdmin, SQL Management Studio, etc.)

You will need to configure and adjust your settings according to the specifications of the operating system / IDE / database management tools that you choose. Details on how to do this can be found online at each company's respective websites or in various tutorials.

6.1.2 Database Installation - Developer

1) Download and install PhpMyAdmin from the link below:

http://www.phpmyadmin.net/home_page/downloads.php

Figure 1: Database Installation - Developer (1 of 2)

2) Execute these instructions in PhpMyAdmin:

```
-- phpMyAdmin SQL Dump
-- version 3.5.5
-- http://www.phpmyadmin.net
--
-- Host: localhost
-- Generation Time: Nov 18, 2013 at 04:44 PM
-- Server version: 5.5.33-31.1
-- PHP Version: 5.3.17

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";


/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;


--
-- Table structure for table `fav`
--

CREATE TABLE IF NOT EXISTS `fav` (
  `busid` int(5) DEFAULT NULL,
  `userid` int(10) NOT NULL,
  `stopid` int(6) DEFAULT NULL,
  `id` int(11) NOT NULL AUTO_INCREMENT,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=10 ;

--

--
-- Table structure for table `user`
--

CREATE TABLE IF NOT EXISTS `user` (
  `username` varchar(30) COLLATE utf8_unicode_ci NOT NULL,
  `password` varchar(30) COLLATE utf8_unicode_ci NOT NULL,
  `e-mail` varchar(30) COLLATE utf8_unicode_ci NOT NULL,
  `phone` text COLLATE utf8_unicode_ci NOT NULL,
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `activation` int(4) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=MyISAM  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
AUTO_INCREMENT=38 ;

--
```

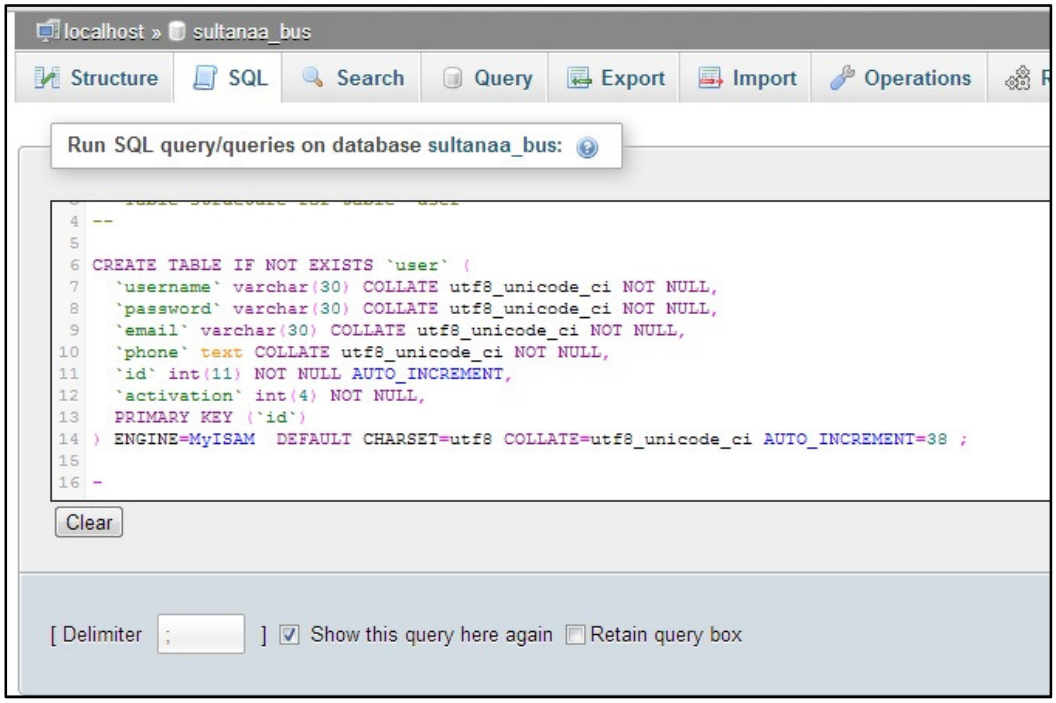


Figure 6: Database Installation - Developer (2 of 2)

6.1.3 System Administration User - Developer

N/A

6.1.4 Operational Manual and Instruction - Developer

Important: Be sure to complete the steps in **6.1.1** (Prerequisites – Developer) and **6.1.2** (Database Installation – Developer) before continuing with this section!

1) Upload the .php files that came bundled with the Metro Live source code to your configured server.

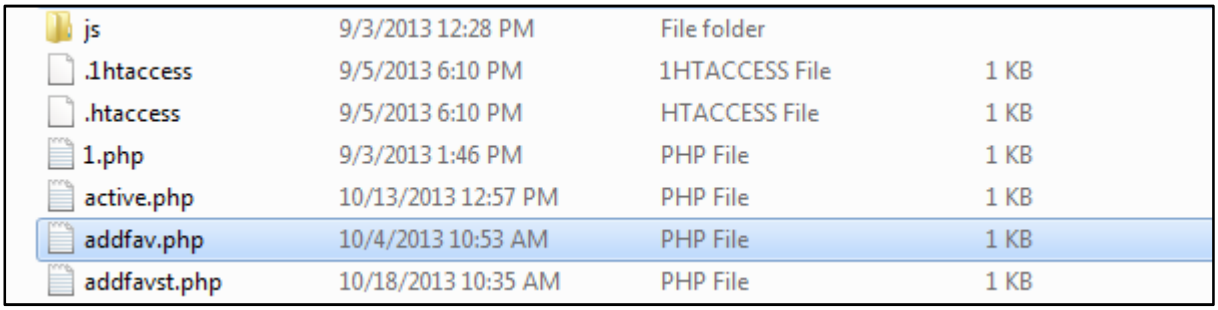


Figure 7: Operational Manual and Instruction - Developer (1 of 6)

2) Download and install Android SDK from the link:

http://developer.android.com/sdk/index.html

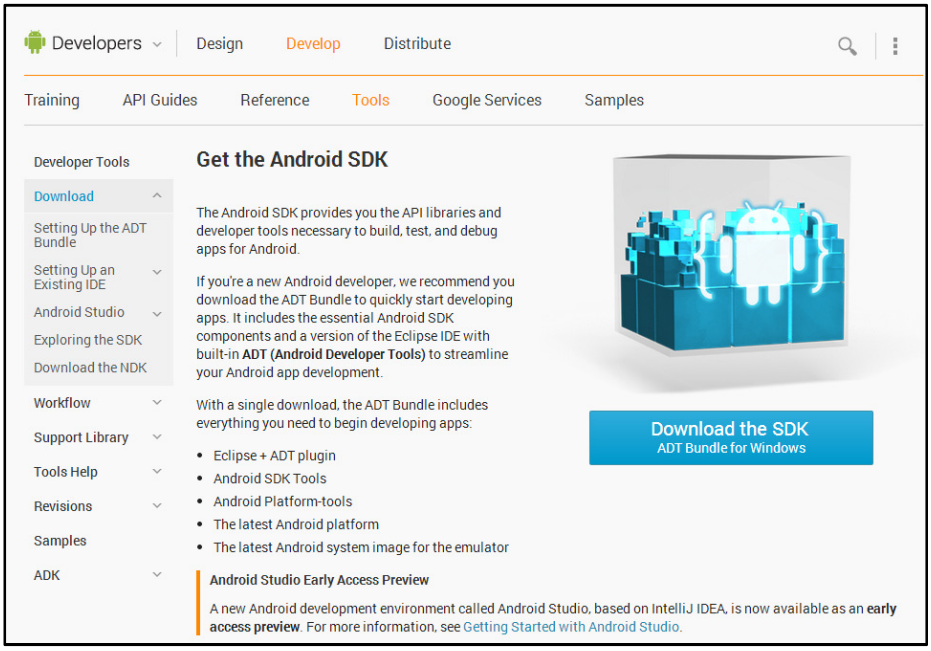


Figure 8: Operational Manual and Instruction - Developer (2 of 6)

3) Open the APK file using the Android SDK.
Select *File->Open->existing Android project-> Finish the wizard steps*

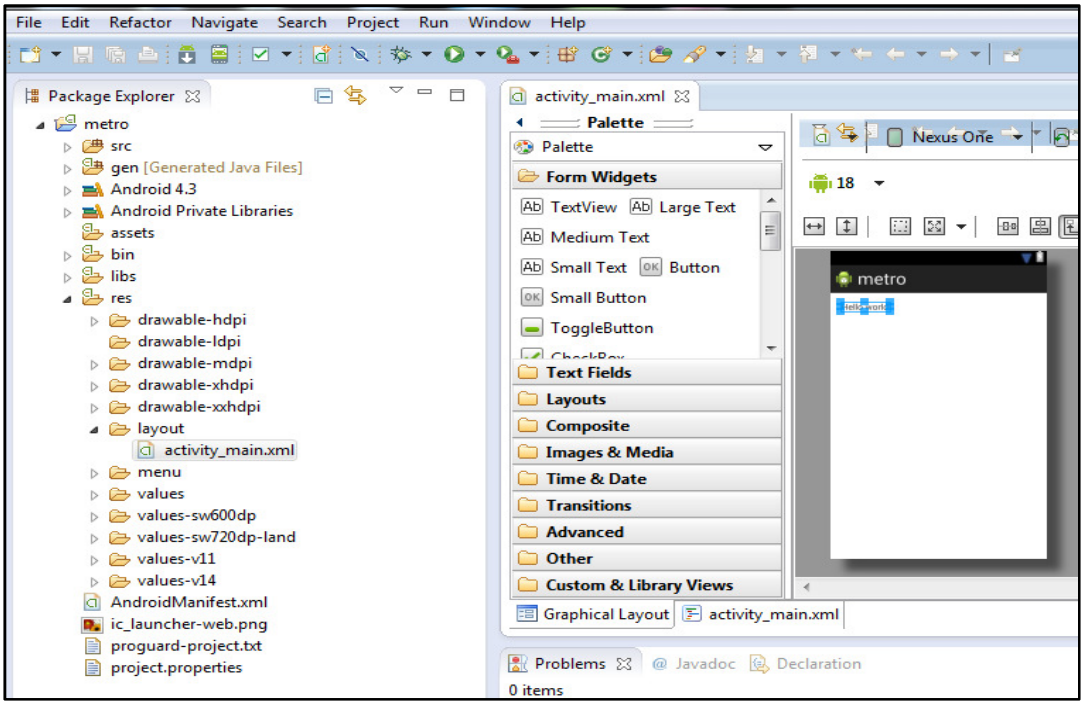


Figure 9: Operational Manual and Instruction - Developer (3 of 6)

4) Open the Android Developer Tool Virtual Simulator by selecting Android Virtual Device Manager.

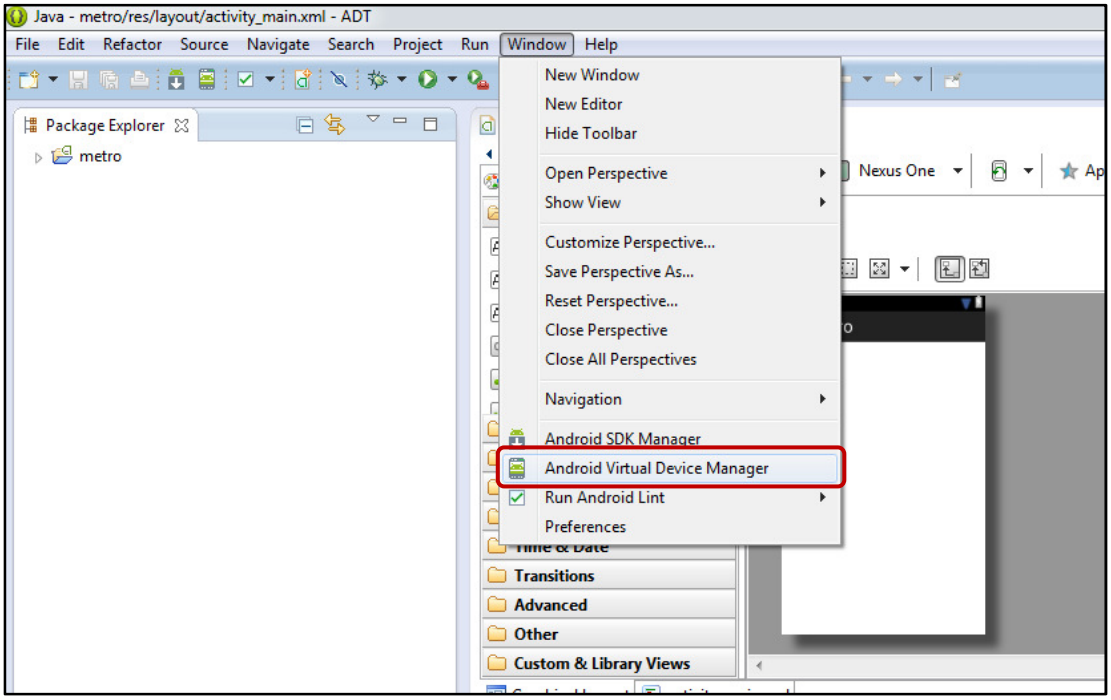


Figure 10: Operational Manual and Instruction - Developer (4 of 6)

5) Click the green "Play" button to run the Virtual Simulator.

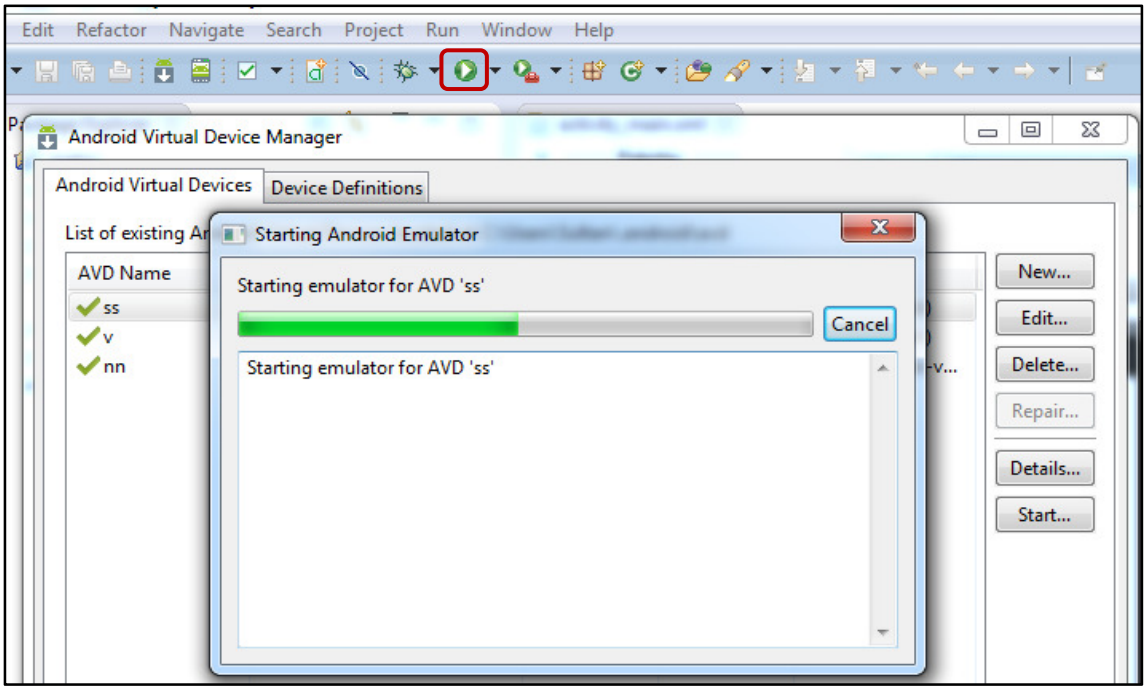


Figure 21: Operational Manual and Instruction - Developer (5 of 6)

6) Finally, within the Virtual Simulator, click on the application icon (not shown in this screen shot – you may need to scroll the emulator window to show the next set of icons) to run it.

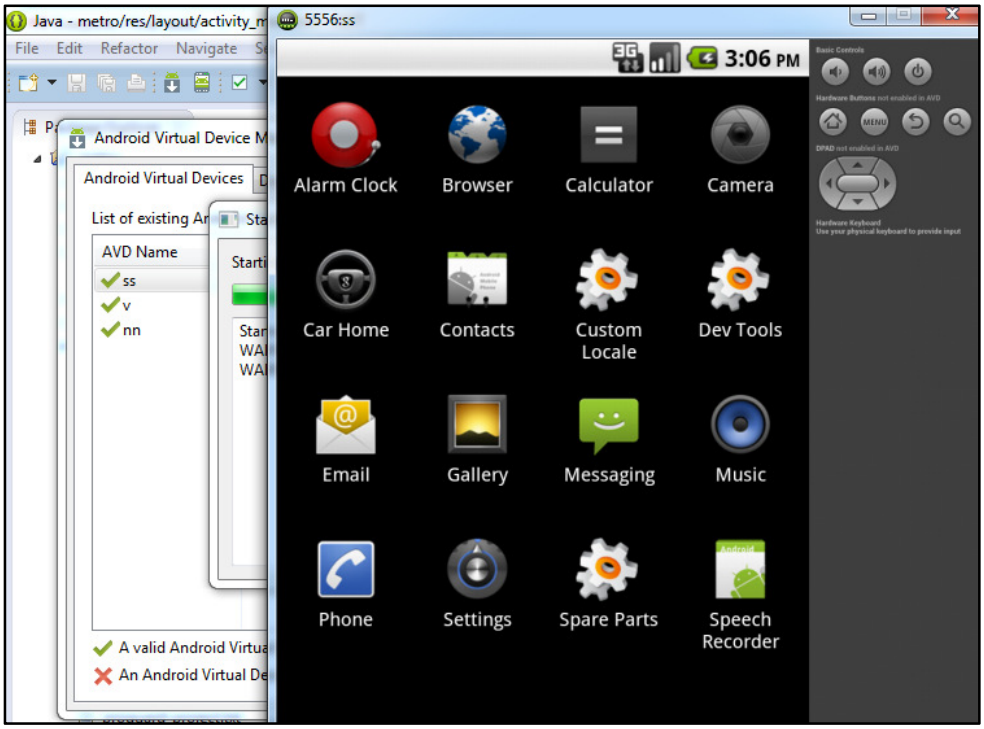


Figure 32: Operational Manual and Instruction - Developer (6 of 6)

6.2 Installation - User

6.2.1 Prerequisites – User

In order to install Metro Live as a user you must have the following:

- Cellular phone capable of running Android OS 4.xx.
- Active account with Google Play™

6.2.2 Database Installation - User

N/A

6.2.3 System Administration User - User

N/A

6.2.4 Operational Manual and Instruction – User

Metro Live User Manual

This manual will guide you through the following steps:

- Installing the Application
- Registering an Account
- Logging In
- Checking the Number of Active Routes and Buses
- Checking a Specific Bus Schedule
- Finding a Specific Route
- Finding a Specific Bus
- Managing Favorites
- Calling the Help Center
- Sending a Text Message to the System
- Sending an E-mail to the System
- Connecting to the Application's Facebook Page
- Following the Application on Twitter
- Logging Out

Installing the Application

1) Click on the **Play Store** icon on your Android smart phone. This will launch the Google Play™ store.
2) Type ”metrolivela” in the text box and click **Search**.

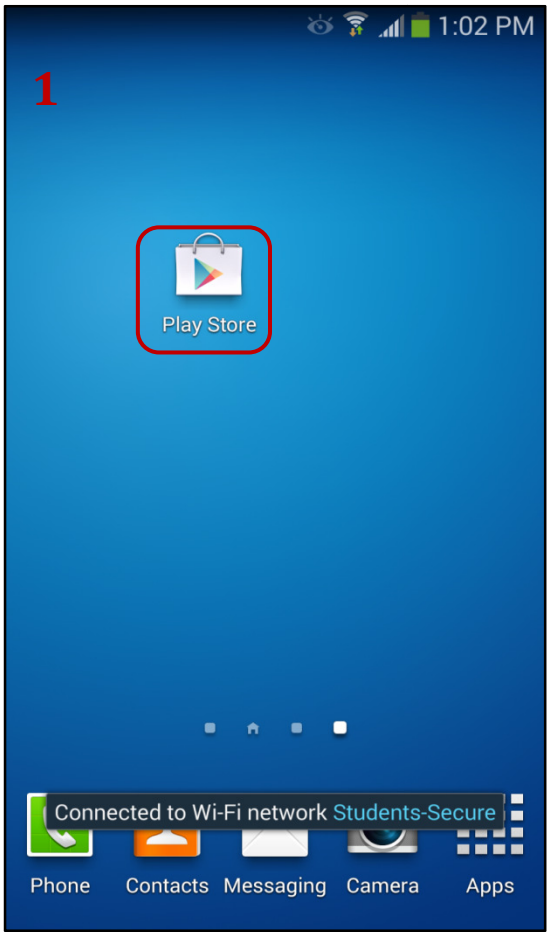


Figure 13: Installing the Application (1 of 4)

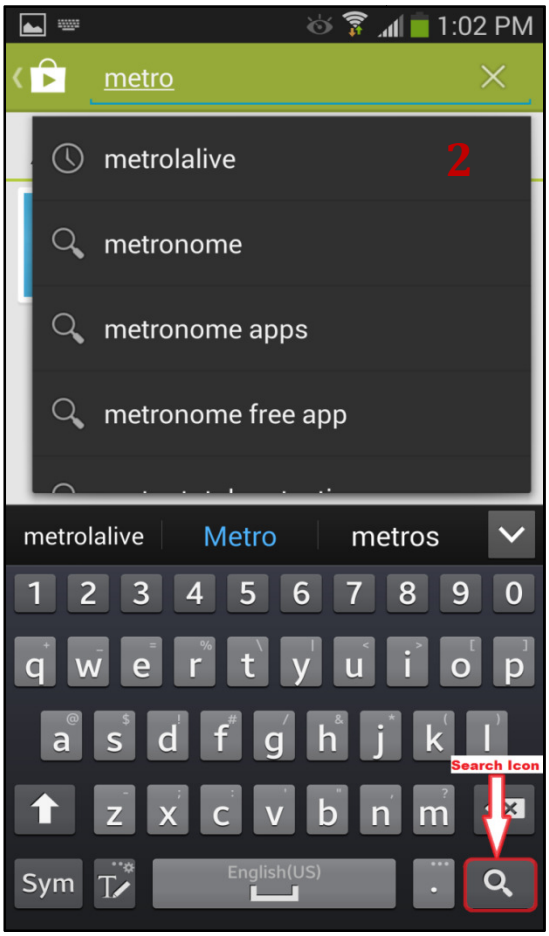


Figure 14: Installing the Application (2 of 4)

[Installing the Application continued on next page]

3) Click the on **Metro LA Live** icon to download the application.
4) Click on the **Install** button to install the application.

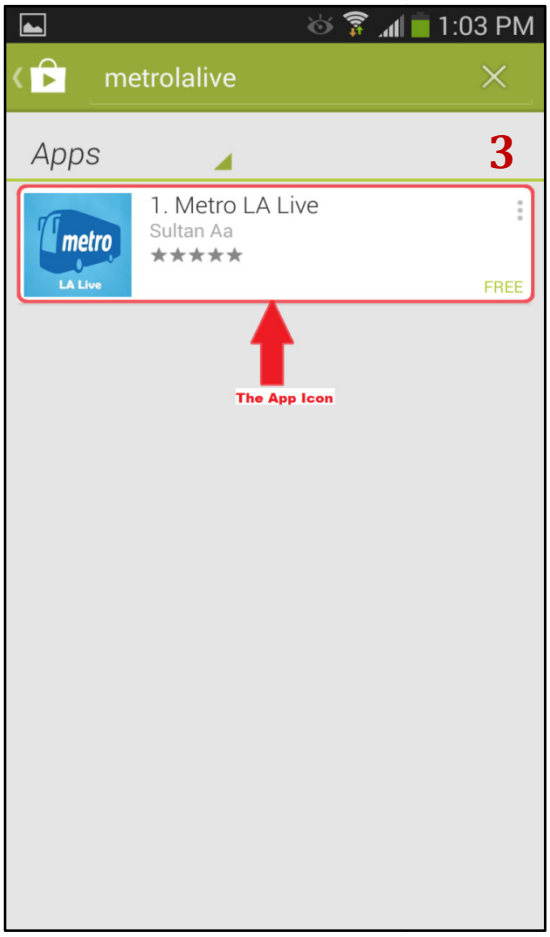


Figure 14: Installing the Application (3 of 6)

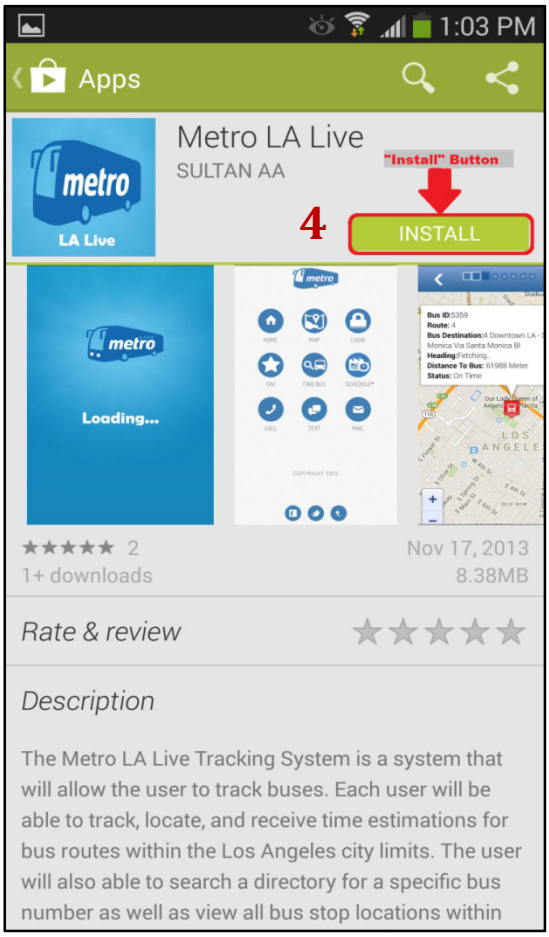


Figure 16: Installing the Application (4 of 6)

[Installing the Application continued on next page]

5) Click on the **Accept** button to accept the terms and conditions of the application.
6) After the download has finished, you can either open the application or uninstall it.

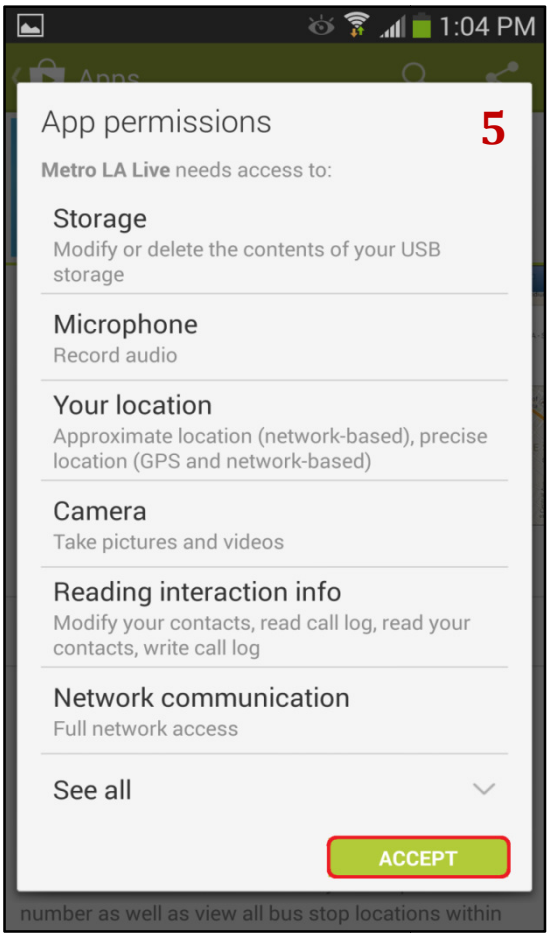


Figure 17: Installing the Application (5 of 6)

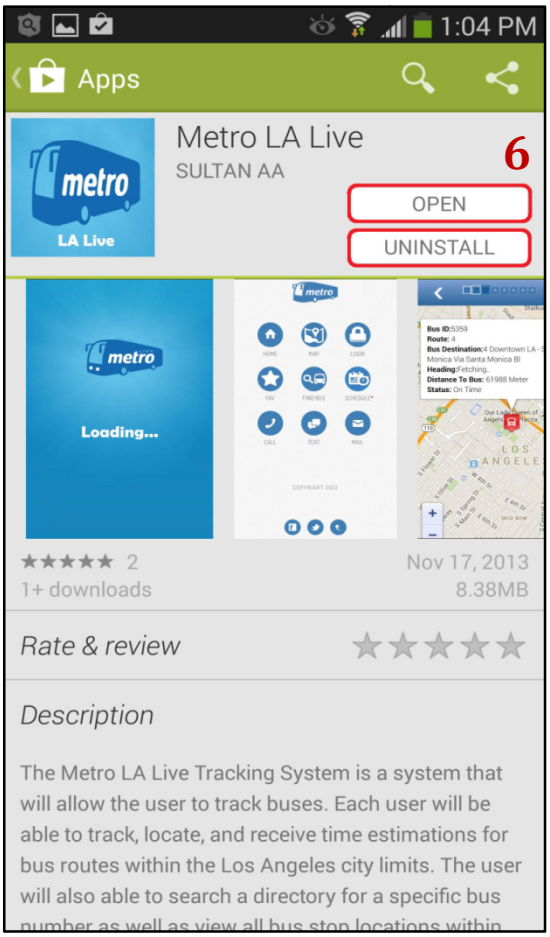


Figure 18: Installing the Application (6 of 6)

Registering an Account

7) To register, click on the **Login** icon.
8) On the login page, select the **Register** tab and fill out the form with your desired **Username**, **Password**, **E-mail** and **Phone**. Then click the **Register** button.

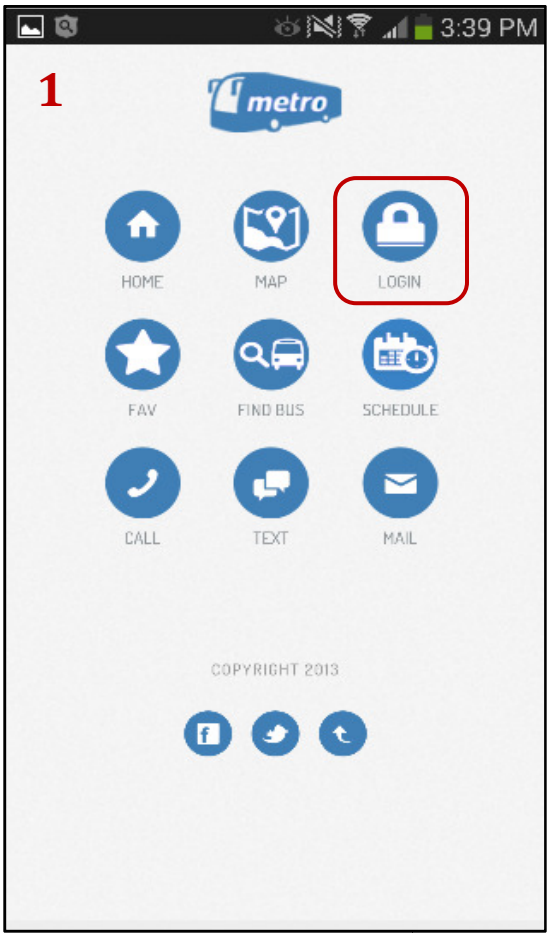


Figure 19: Registering an Account (1 of 2)

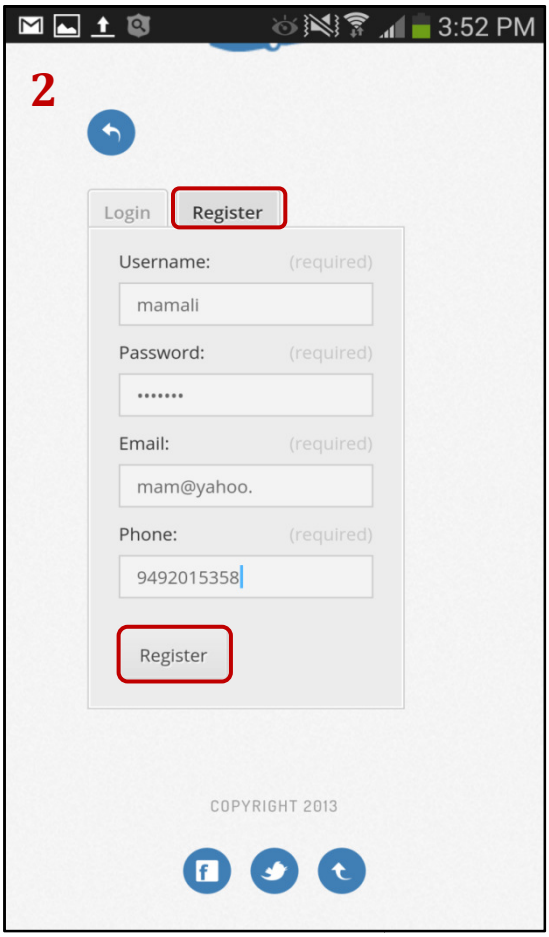


Figure 20: Registering an Account (2 of 2)

Logging In

1) Click on the **Login** icon on the home page of the application.
2) Enter your **Username** and **Password** into the appropriate text boxes. Then select **Login**.

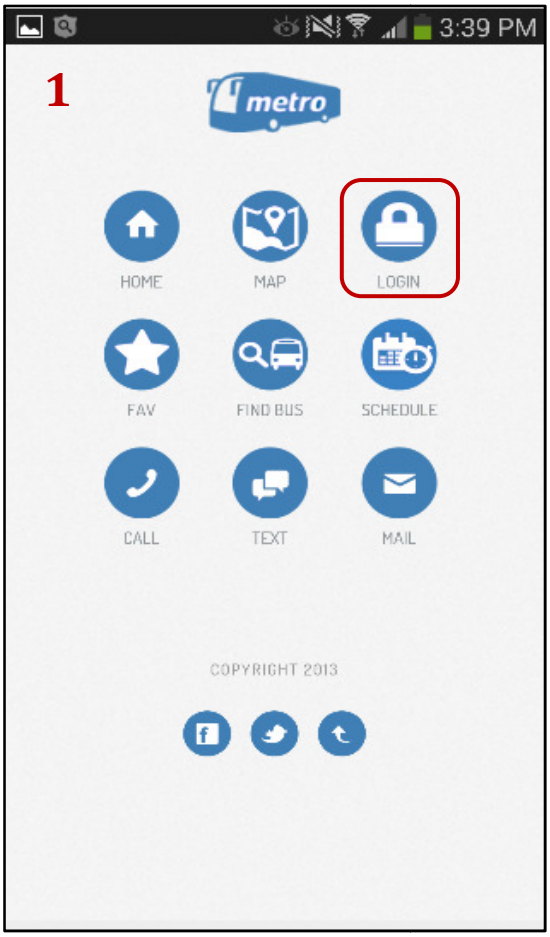


Figure 25: Logging In (1 of 4)

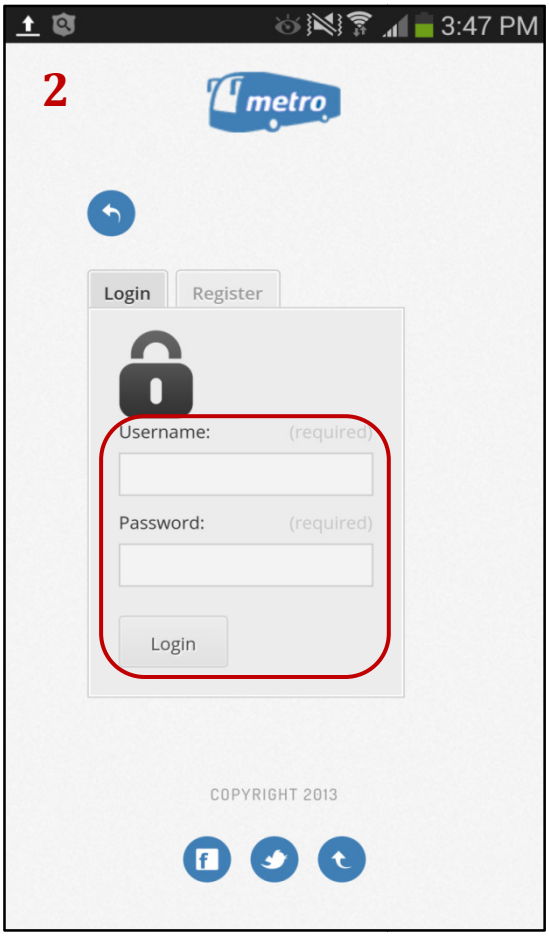


Figure 62: Logging In (2 of 4)

[Logging In continued on next page]

3) If this is the first time that you have logged into the system, a text box will appear that requires your **Activation Code**. The activation code will be sent to you via SMS automatically after the registration process finished successfully.
4) After username and password entered, if the username and password are correct a **Welcome Message** should be displayed on the top of the home page.

Figure 23: Logging In (3 of 4)

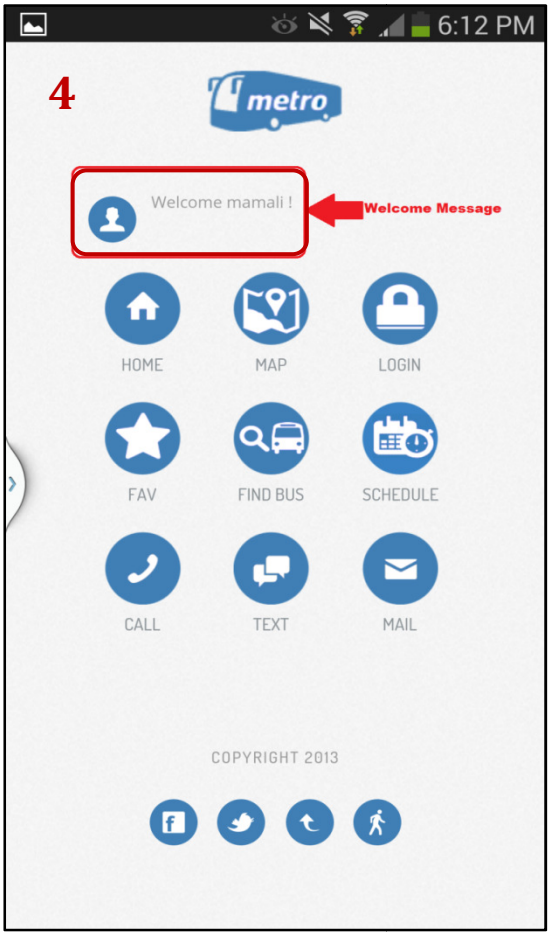


Figure 24: Logging In (4 of 4)

Checking the Active Routes and Buses

1) Click on the **Home** icon.
2) The application displays the number of currently active routes and buses, as shown in the image below.

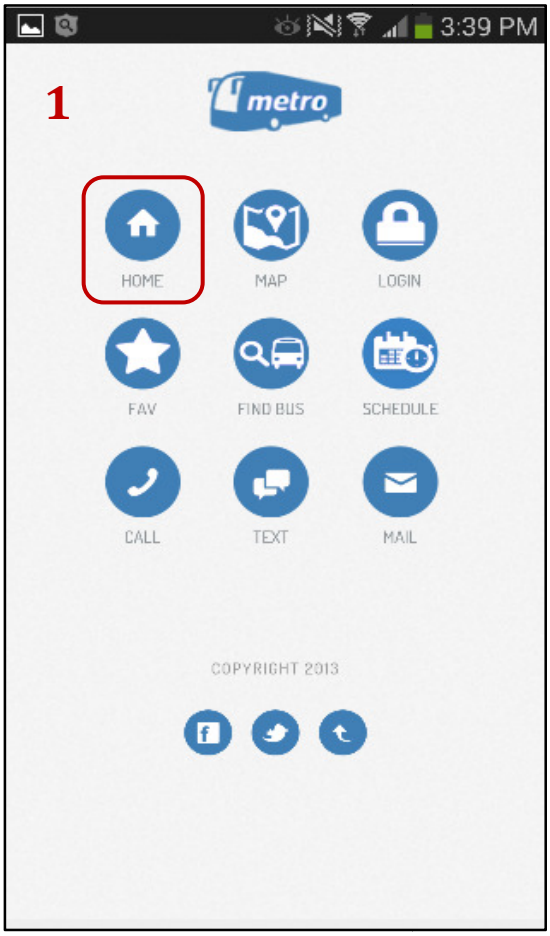


Figure 25: Checking Active Routes (1 of 2)

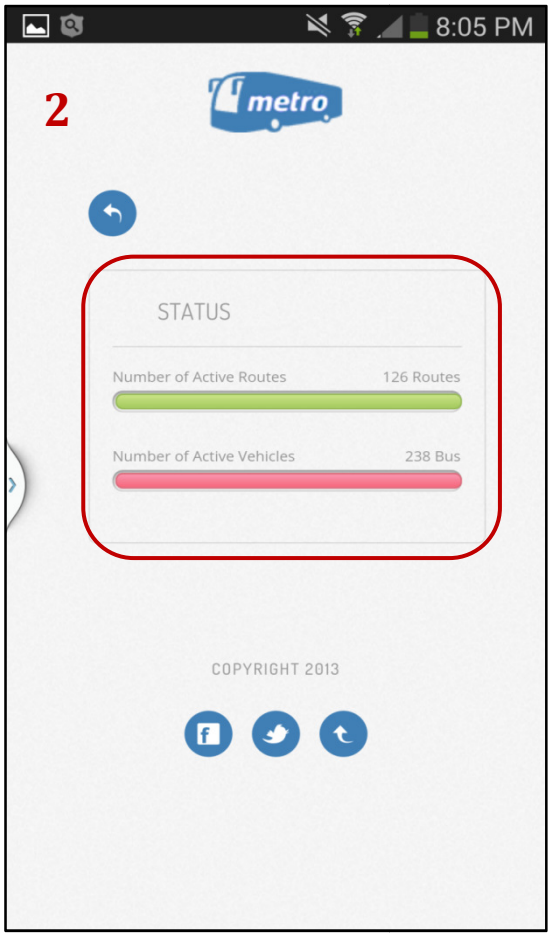


Figure 26: Checking Active Routes (2 of 2)

Checking a Specific Bus Schedule

1) Click on the **Schedule** icon.
2) Select the route you wish to view from the "CHOOSE A ROUTE NUMBER" drop down menu and then click the **.pdf** icon.

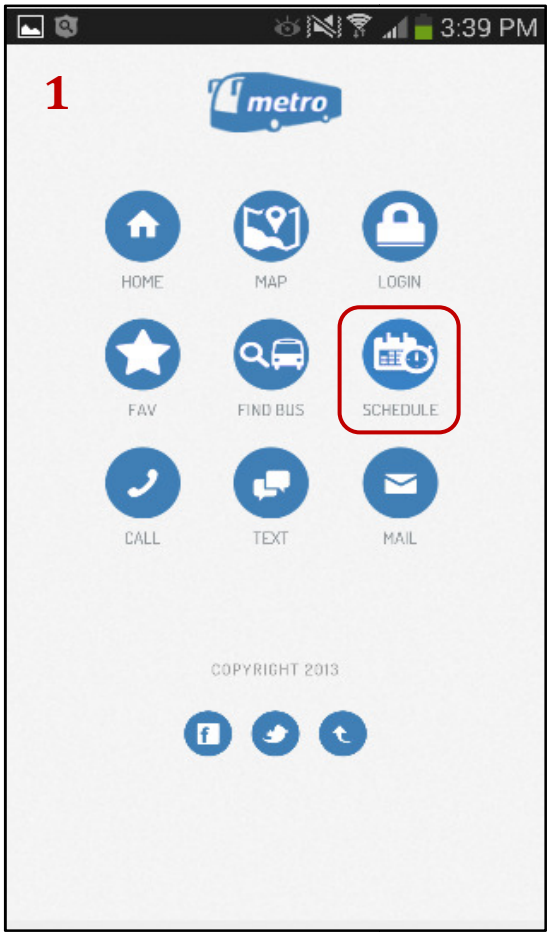
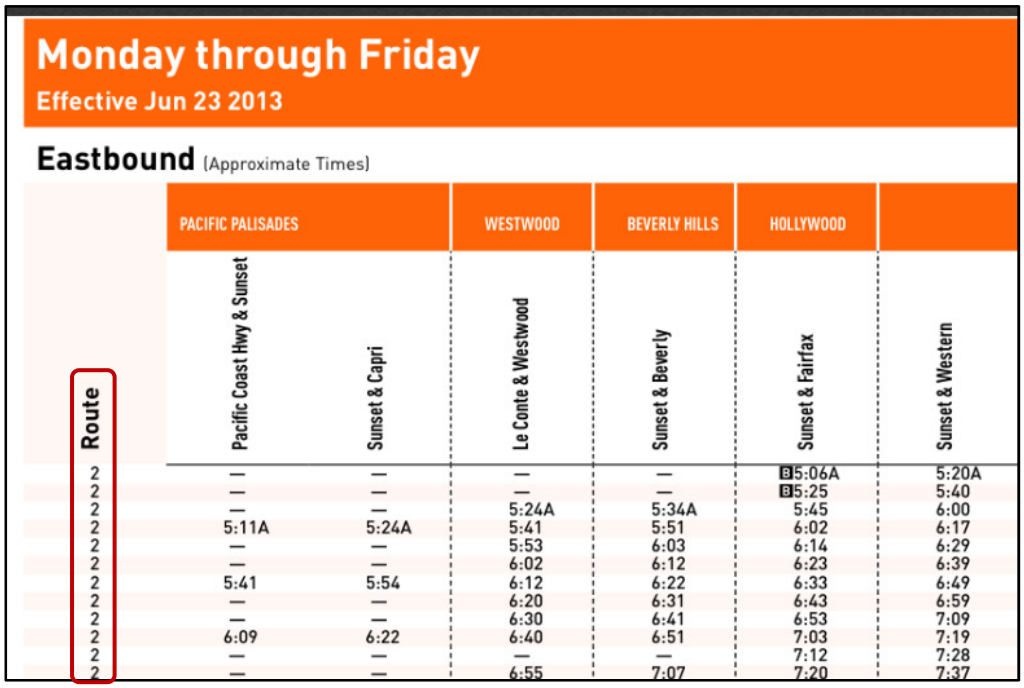


Figure 27: Checking Bus Schedule (1 of 4)

Figure 28: Checking Bus Schedule (2 of 4)

[Checking a Specific Bus Schedule continued on next page]

3) The application will display a .pdf of the selected bus schedule and route map.



Monday through Friday

Effective Jun 23 2013

Eastbound (Approximate Times)

Route	PACIFIC PALISADES		WESTWOOD	BEVERLY HILLS	HOLLYWOOD	
	Pacific Coast Hwy & Sunset	Sunset & Capri	Le Conte & Westwood	Sunset & Beverly	Sunset & Fairfax	Sunset & Western
2	—	—	—	—	B5:06A	5:20A
2	—	—	—	—	B5:25	5:40
2	—	—	5:24A	5:34A	5:45	6:00
2	5:11A	5:24A	5:41	5:51	6:02	6:17
2	—	—	5:53	6:03	6:14	6:29
2	—	—	6:02	6:12	6:23	6:39
2	5:41	5:54	6:12	6:22	6:33	6:49
2	—	—	6:20	6:31	6:43	6:59
2	—	—	6:30	6:41	6:53	7:09
2	6:09	6:22	6:40	6:51	7:03	7:19
2	—	—	—	—	7:12	7:28
2	—	—	6:55	7:07	7:20	7:37

Figure 29: Checking Bus Schedule (3 of 4)

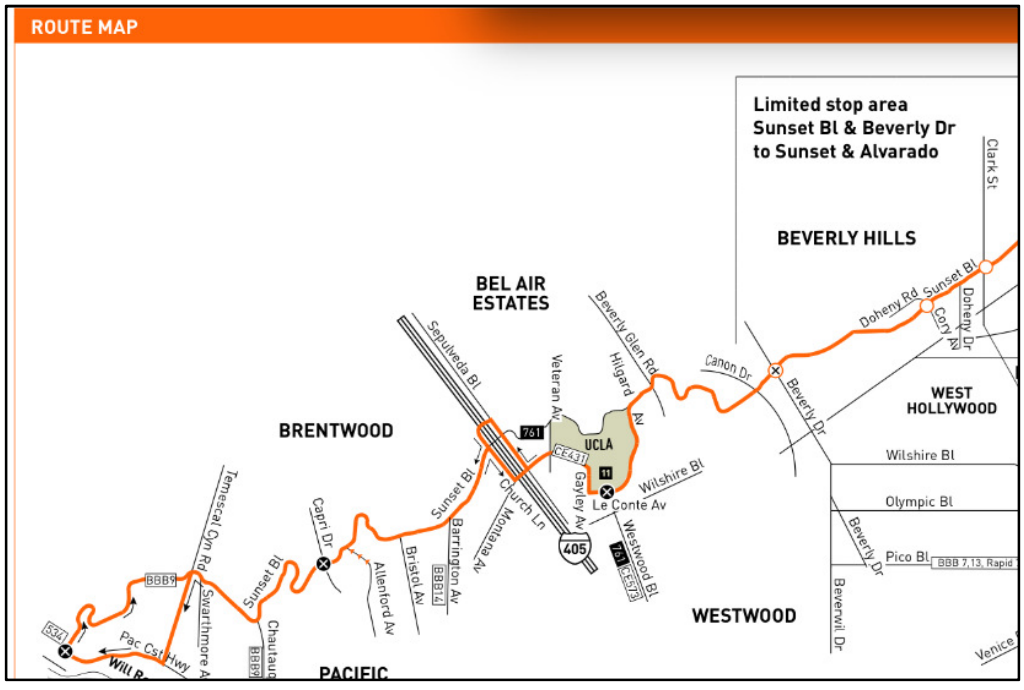


Figure 30: Checking Bus Schedule (4 of 4)

Finding a Specific Route

1) Click on the **Find Bus** icon.
2) Select the route from the "FIND BY ROUTE NUMBER" drop down list.

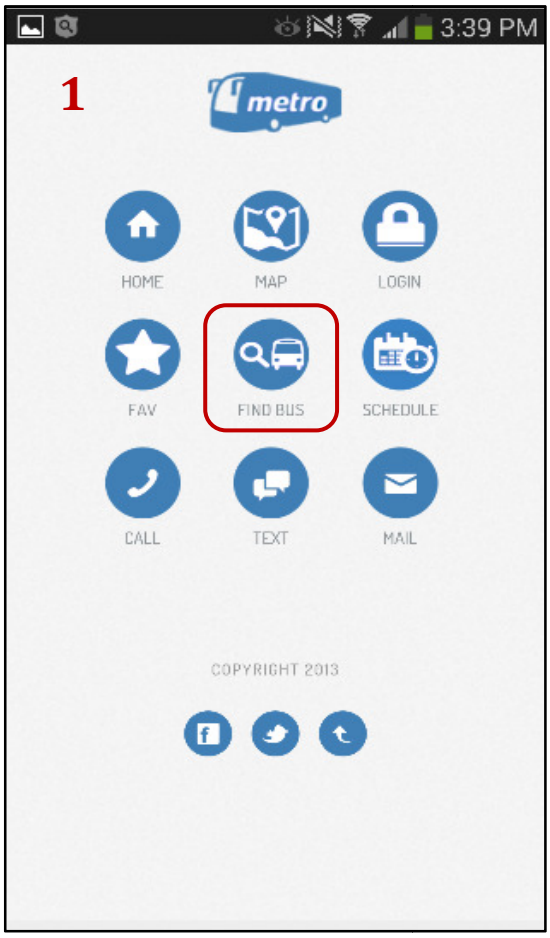


Figure 31: Finding a Specific Route (1 of 4)

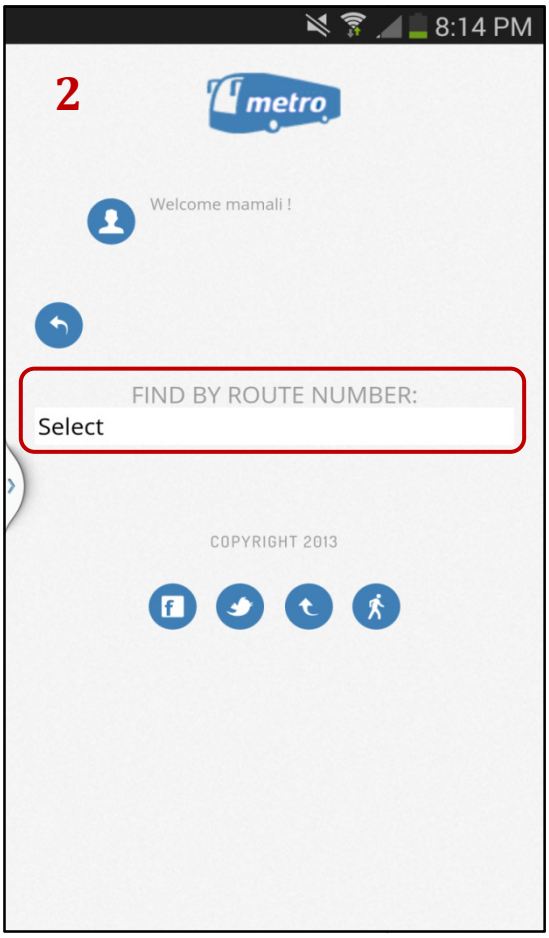


Figure 32: Finding a Specific Route (2 of 4)

[Finding a Specific Route continued on next page]

3) Select the desired destination from “Select Destination” drop down list.
4) A list of routes including route number, bus number, and arrival status will be displayed on the screen. In addition, there are two more icons available on the bottom-right corner of each route to either add the route to the favorites list or to view the route live on the map.

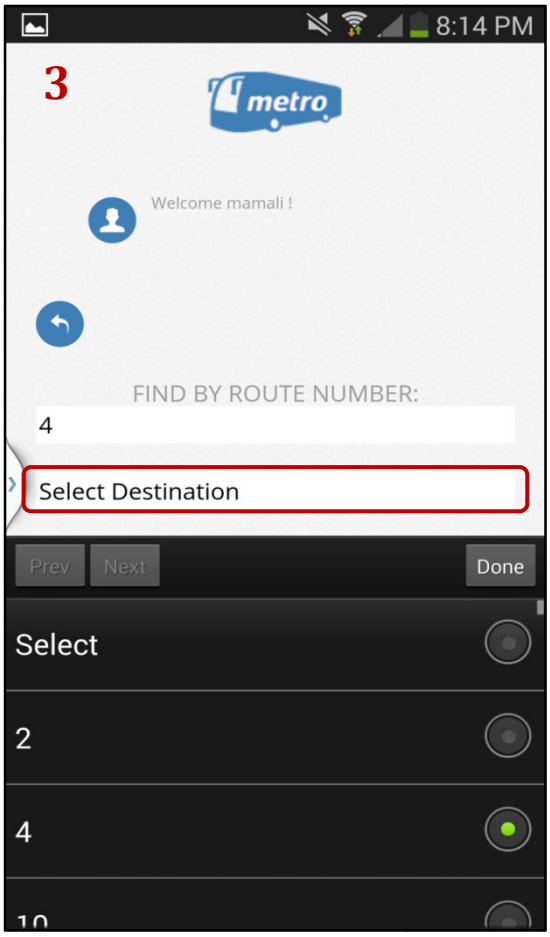


Figure 33: Find a Specific Route (3 of 4)

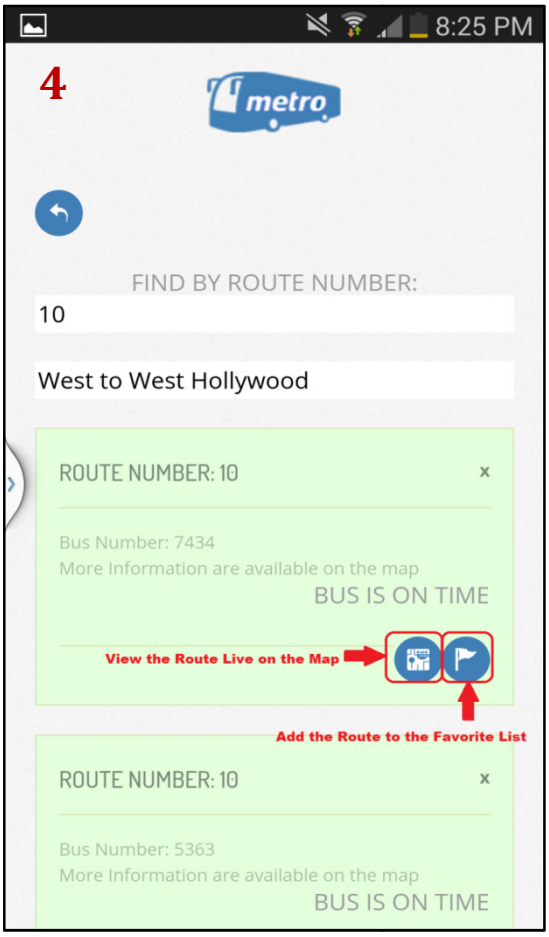


Figure 34: Find a Specific Route (4 of 4)

Finding a Specific Bus

1) Click on the **Map** icon.
2) Enter the desired bus number in the black text box located on the top-center of the screen.

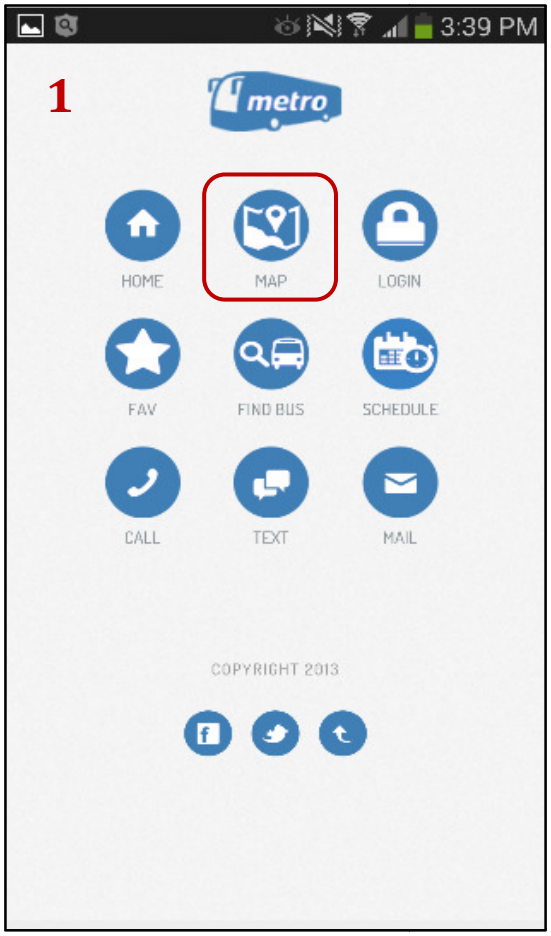


Figure 35: Finding a Specific Bus (1 of 4)

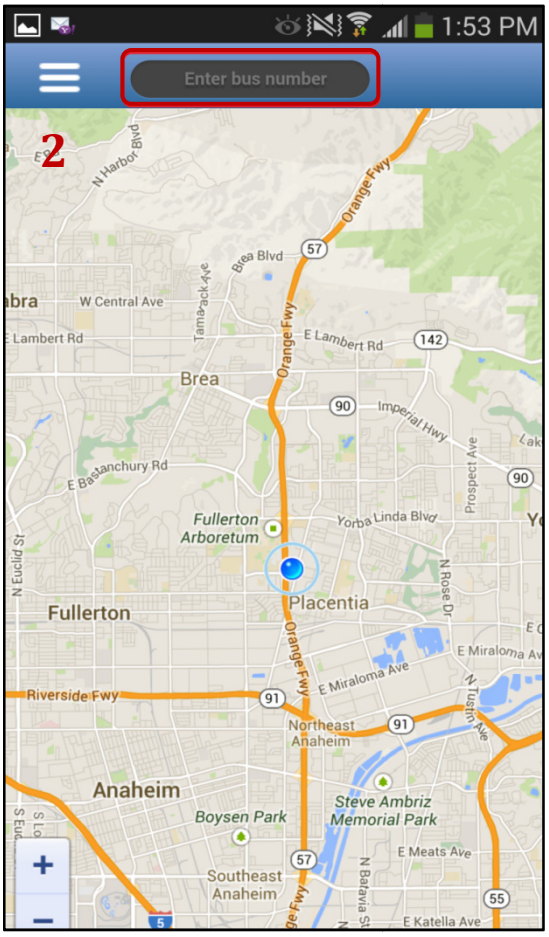


Figure 36: Finding a Specific Bus (2 of 4)

[Finding a Specific Bus continued on next page]

3) The bus location should appear on the map like the page below.
4) Then, tap on the red bus icon on the map to see more detailed information including: the bus ID, route number, bus destination, heading, distance to bus and status.

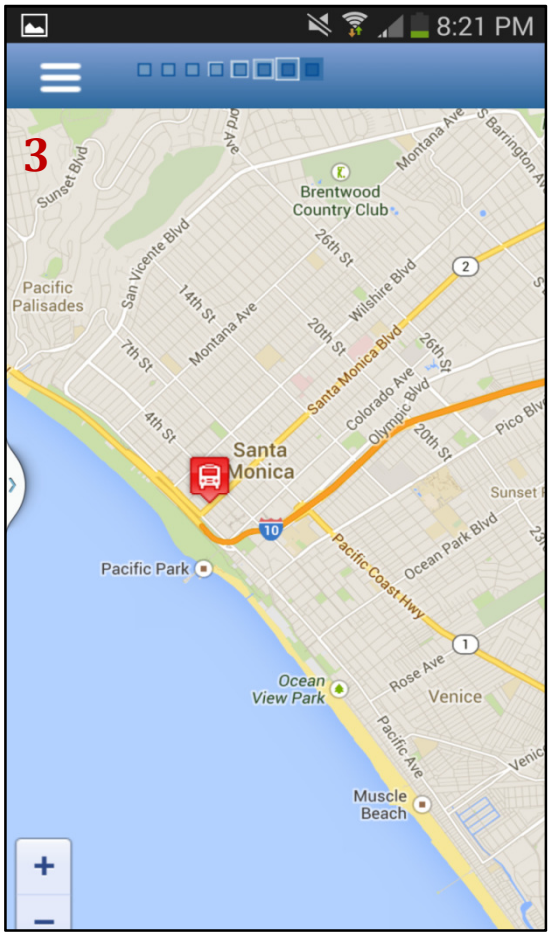


Figure 37: Finding a Specific Bus (3 of 4)

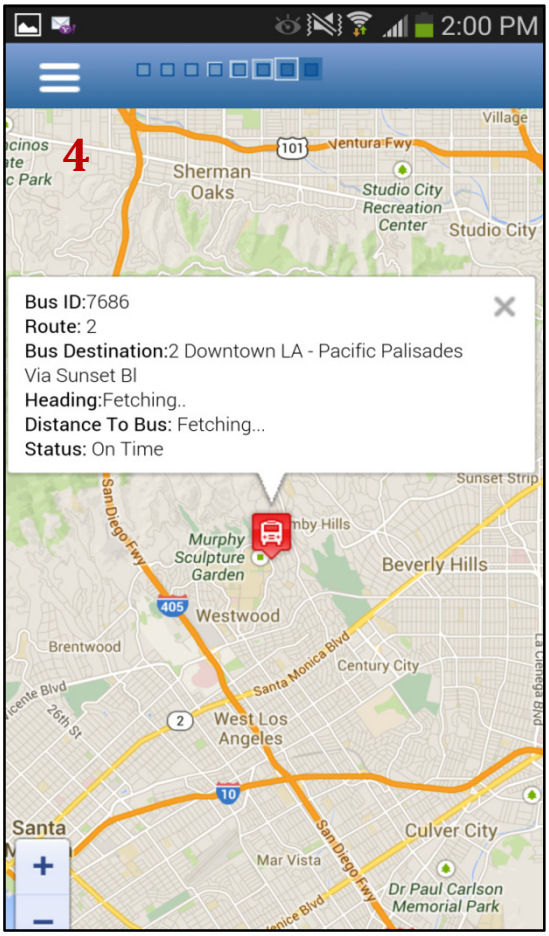


Figure 38: Finding a Specific Bus (4 of 4)

Managing Favorites

1) Click on the **Favorites** icon. Note: You must be logged in to use this feature.
2) If you have added any bus routes to your favorites list (see Find a Specific Route: Step 4), they will be displayed on this screen. You can either delete the route from your favorite list by clicking the trashcan icon or you can view the bus location on the map by clicking on the mini map icon.

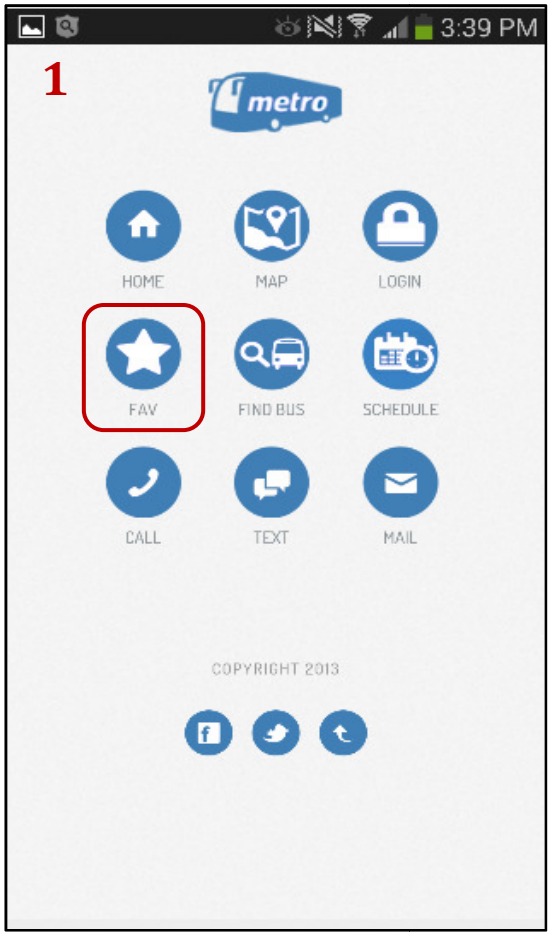


Figure 39: Managing Favorites (1 of 2)

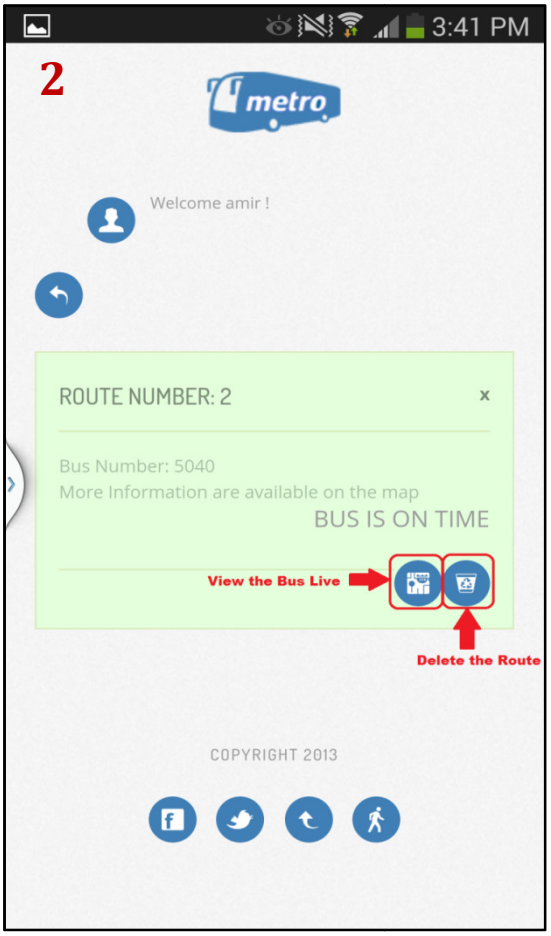


Figure 40: Managing Favorites (2 of 2)

Calling the Help Center

1) Click on the **Call** icon.
2) The application will automatically navigate you to the call interface with the default contact number already entered. Simply press the "Call" button to call the help center.

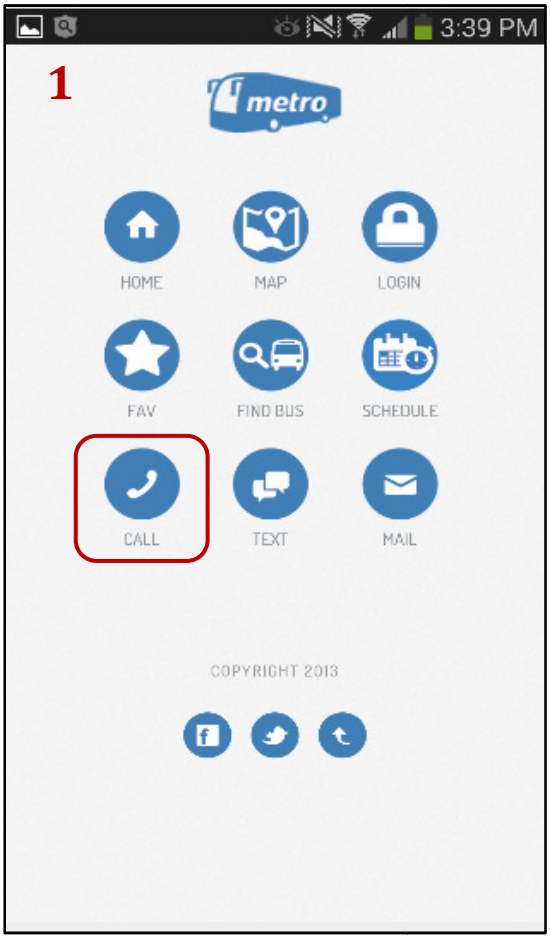


Figure 41: Calling the Help Center (1 of 2)

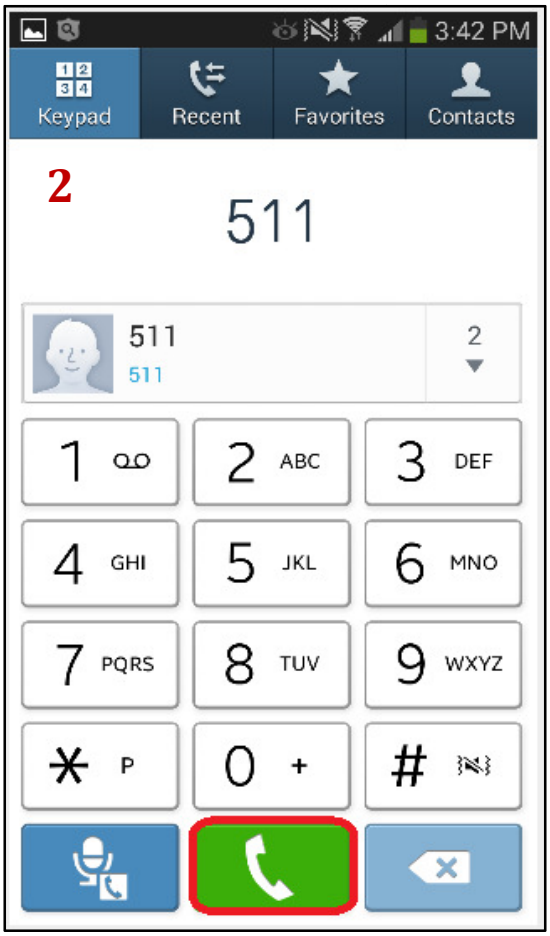


Figure 42: Calling the Help Center (2 of 2)

Sending a Text Message to the System

1) Click on the **Text** icon.
2) The system displays a dialog box which will guide you toward receiving messages via SMS. Simply type "Metro" in the reply text box and click send in order to receive messages via SMS.

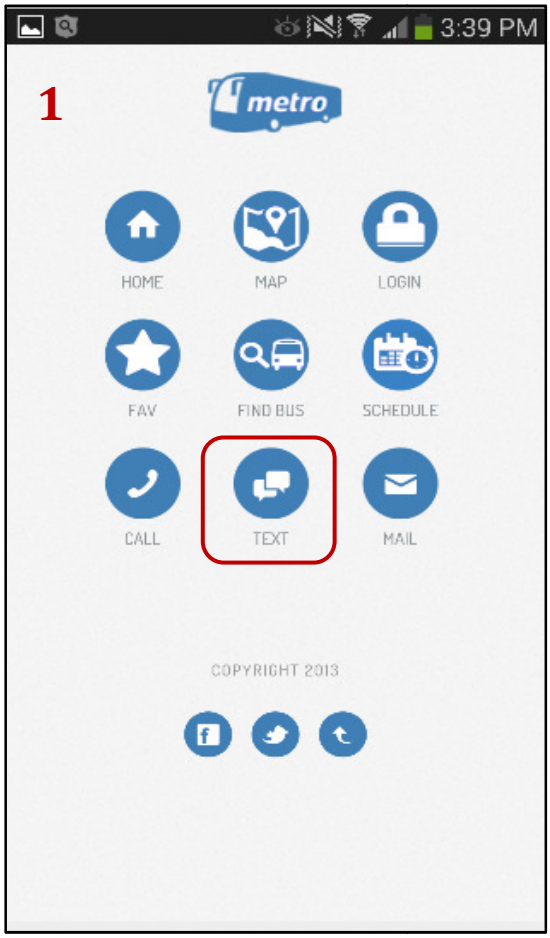


Figure 43: Sending a Text Message (1 of 5)

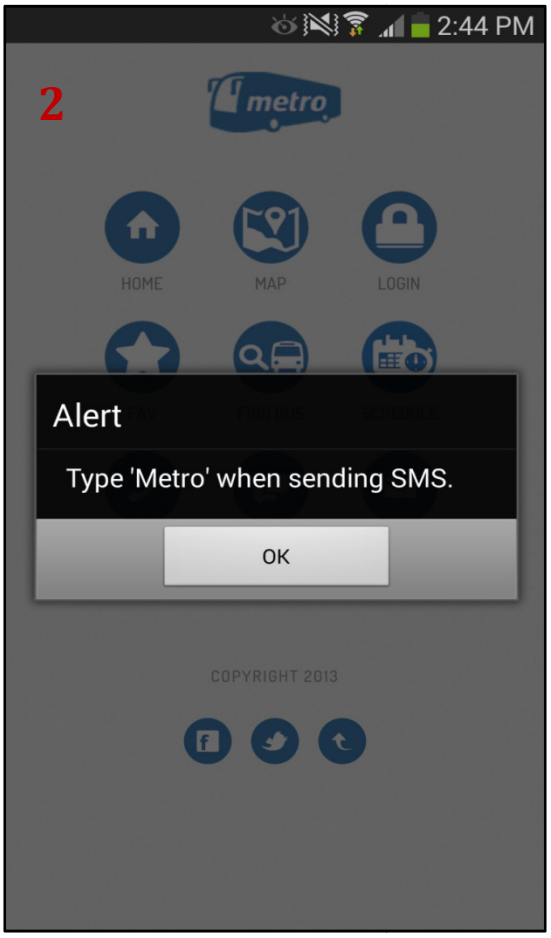


Figure 44: Sending a Text Message (2 of 5)

[Sending a Text Message continued on next page]

3) After the message has been sent, you will receive a reply message with further options on how you can navigate the SMS menu.
4) If you choose "L", a list of available routes will be displayed on the screen as shown in the picture.

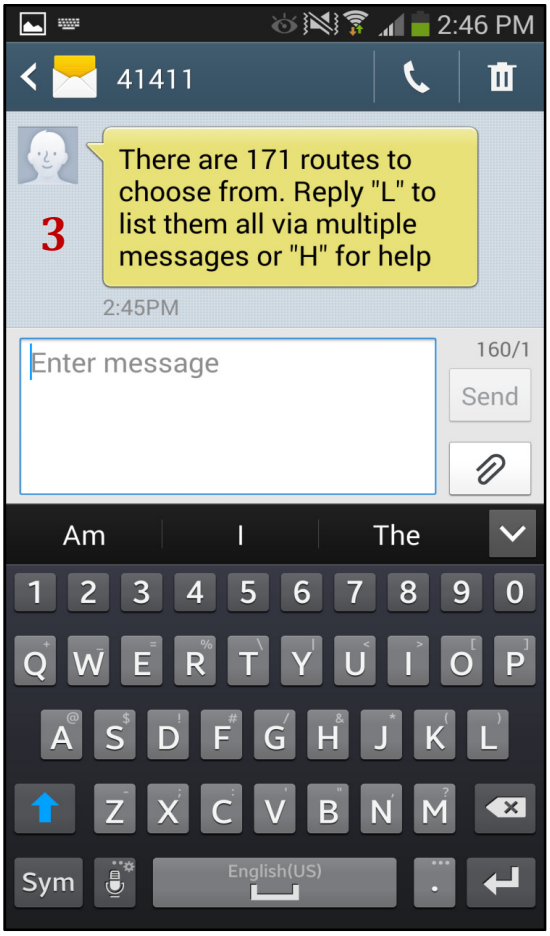


Figure 45: Sending a Text Message (3 of 5)

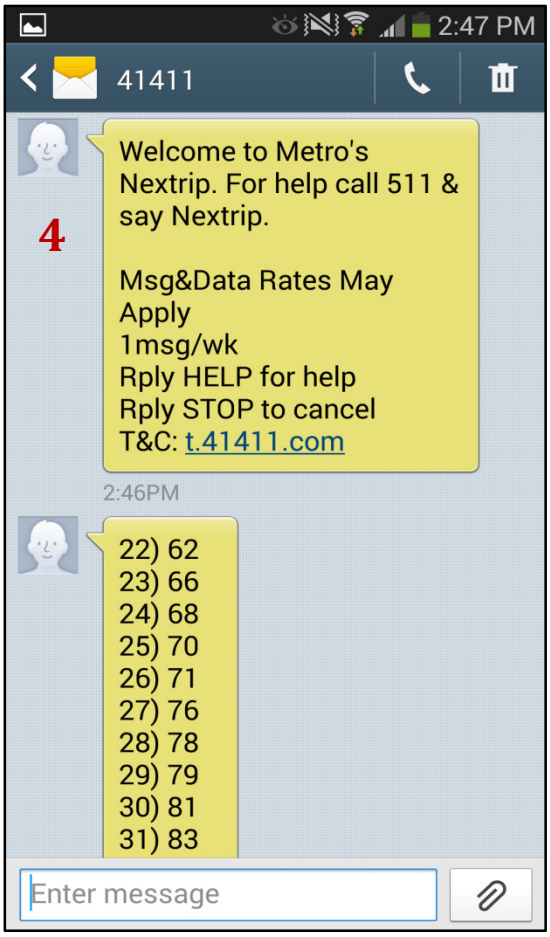


Figure 46: Sending a Text Message (4 of 5)

[Sending a Text Message continued on next page]

5) If user types the letter “H”, the user receives instructions on how to get help from the system via SMS as shown on the picture.

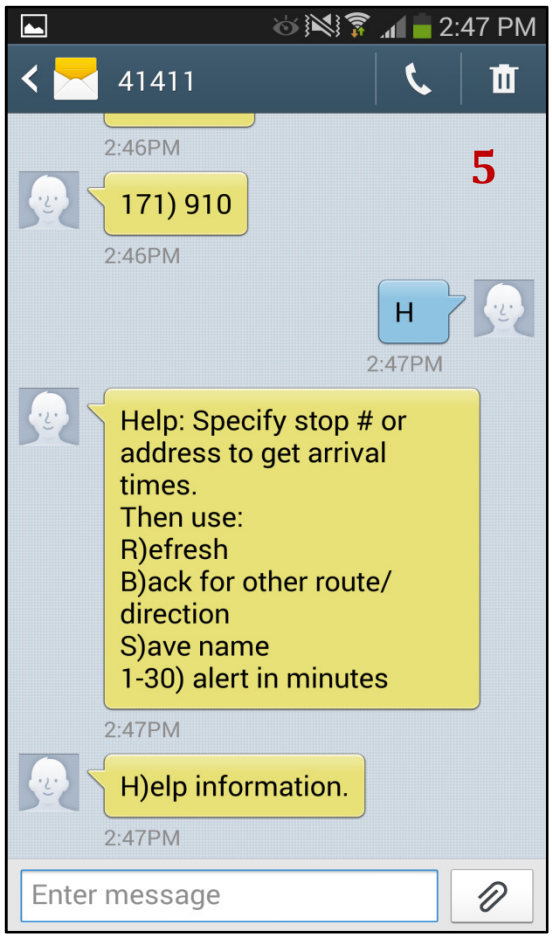


Figure 47: Sending a Text Message (5 of 5)

Sending an E-mail to the system

1) Click on the **Mail** icon.
2) Depending on your phone's e-mail settings, you will be given an option for which e-mail client to use. Choose one.

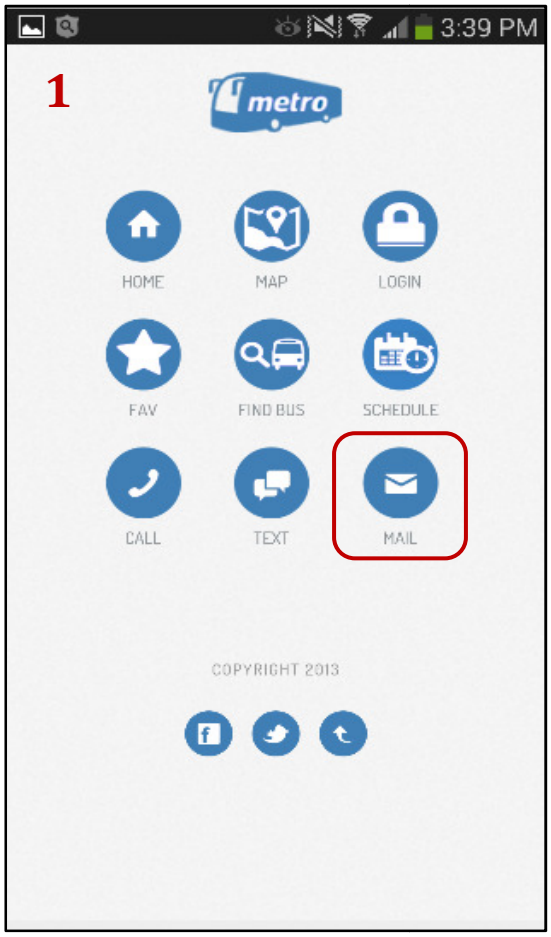


Figure 48: Sending an E-mail (1 of 3)

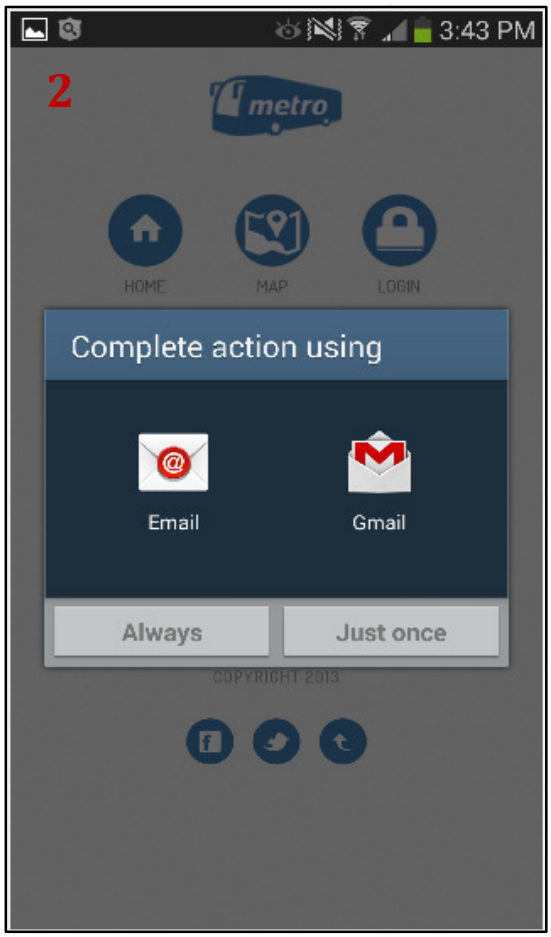


Figure 49: Sending an E-mail (2 of 3)

[Sending an E-mail continued on next page]

3) Finally, after choosing one of the two options available to send an e-mail, the page below will appear. Here you are able to send any feedback, comments, or concerns to the Metro Live team.

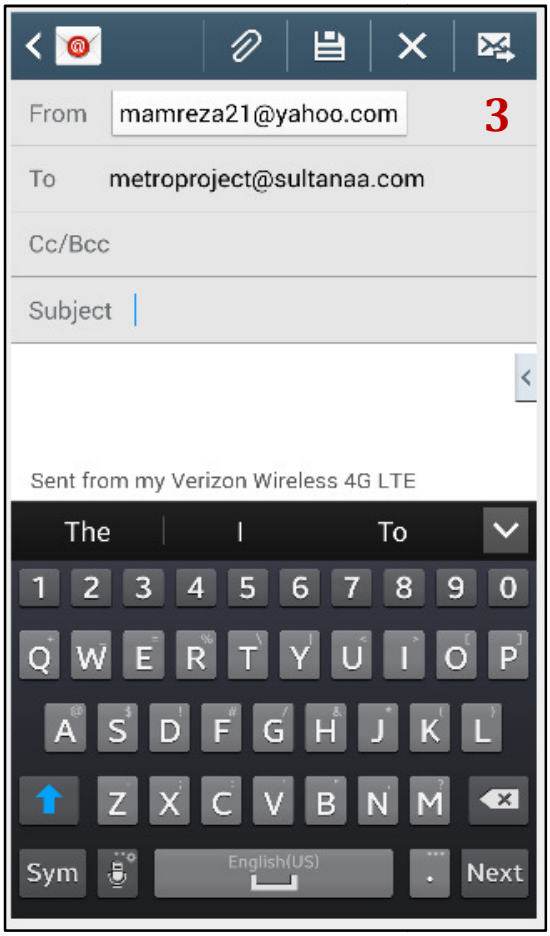


Figure 50: Sending an E-mail (3 of 3)

Connecting to the Application's Facebook Page

1) Click on the **Facebook** icon.
2) The application will navigate you to the Metro Live Facebook page. You can follow any updates for the application by selecting "Like."

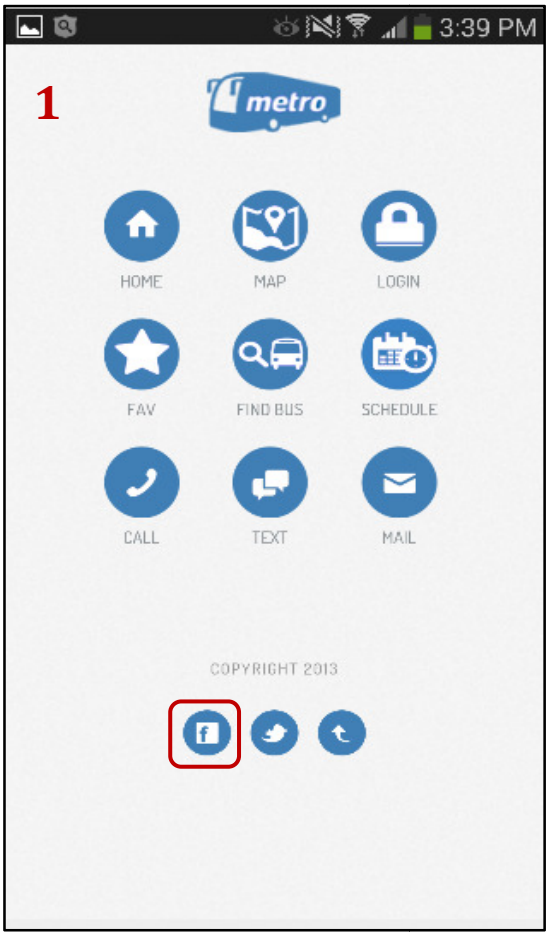


Figure 51: Connecting to Facebook (1 of 2)

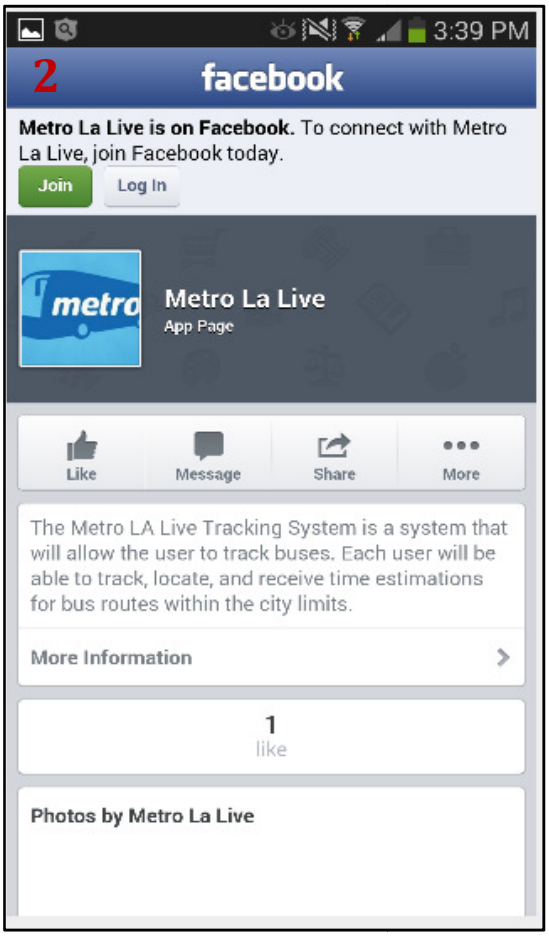


Figure 52: Connecting to Facebook (2 of 2)

Following the Application on Twitter

1) Click on the **Twitter** icon.
2) The application will navigate you to the application's Twitter page. Click on the "Follow Metro LA Live" button to be alerted of any new updates to the application.

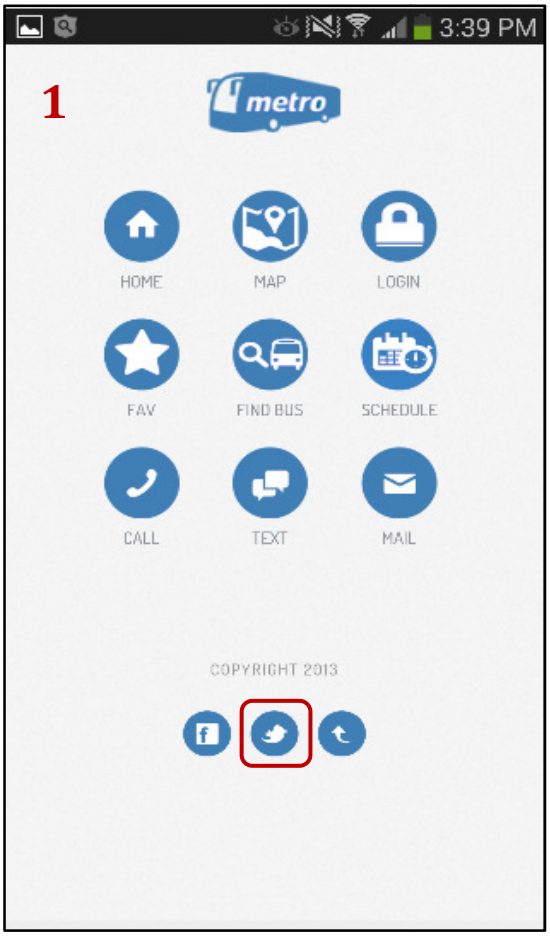


Figure 53: Connecting to Twitter (1 of 2)

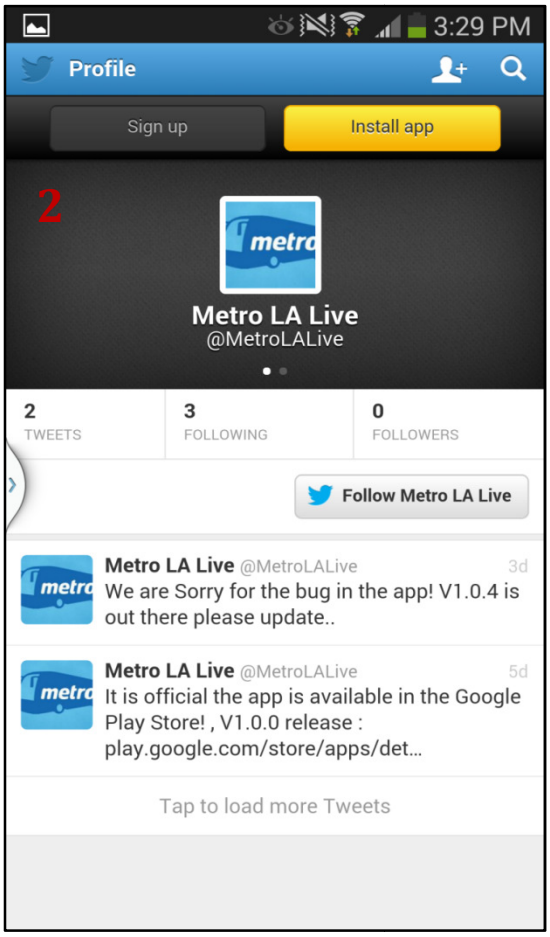


Figure 54: Connecting to Twitter (2 of 2)

Logging Out

1) Click on the **Logout** icon to successfully log out of the application.

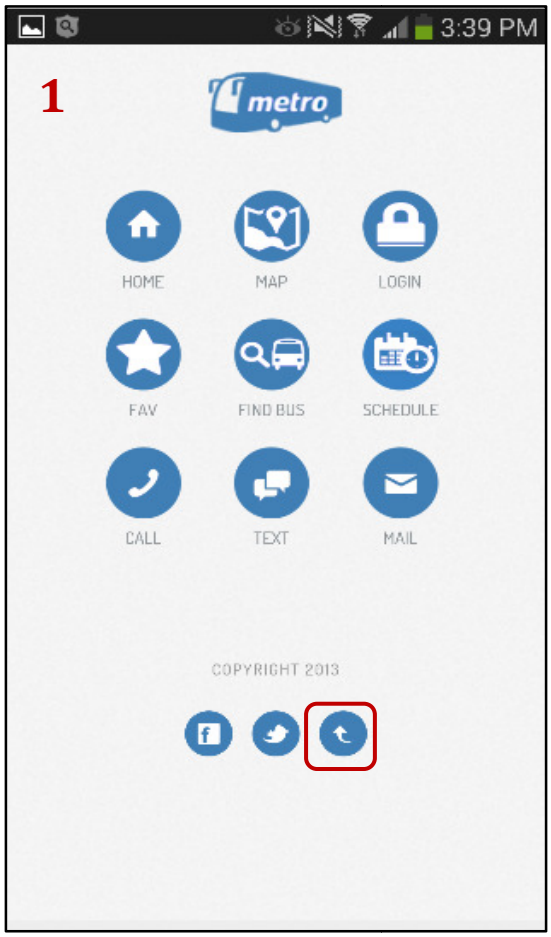


Figure 55: Logging Out (1 of 1)

7. Recommendations for Enhancement

This project met all of (and exceeded!) the client's requirements and expectations. However, there is always room for improvement. The biggest drawback of this application was beyond the control of the development team. Unfortunately, the LAMTA GPS transmitters used by its buses only update their location once every 2 minutes. Thus, there is a significant delay on the map animation when tracking a bus.

Our solution:

- **Interpolation** – In the future, Metro Live will smoothen the animation of a tracked bus by analyzing the bus's heading and speed as indicated by its previous GPS locations. The application will then interpolate (make an educated guess) as to when the bus should reach its next stop. Using this information, the bus animation can be generated in a way that is more aesthetically pleasing to the user.

8. References and Bibliography

Ganesan, Suriya. "How to Write Test Cases for Login." *How to Write Test Cases for Login*. 9 Lessons, 21 Feb. 2011. Web. 27 Apr. 2013.

Software Engineering: A Practitioner's Approach
Pressman, Roger S., McGraw-Hill Higher Education, 7th Edition, 2010

Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development, Third Edition
Larman, Craig., Addison Wesley Professional, October 20, 2004.