Crime Busters Android Application

Final Documentation

Team CBS

May 4, 2014

Development Team

The following stakeholders are responsible for approving or reviewing this document:

Name	Role	Date Approved
Chris Ababan	Software Engineer	5/4/2014
Matt Hubbs	Software Engineer	5/4/2014
Boris Sadkhin	Software Engineer	5/4/2014
Joshua Situka	Software Engineer	5/4/2014
Khushbu Vaishnav	Software Engineer	5/4/2014

Table of Contents

1.0 PROJECT DESCRIPTION

After a crime happens in the Urbana Champaign area, University of Illinois students receive an email from the chief of police with the email containing the details of the incident, along with descriptions of the offender, such as the color of the clothes the offender was wearing, their race, or other physical distinguishing features, or a video link.

Unfortunately, a large number of people could match the same descriptions, making it difficult for them to catch the perpetrators – our project is a security system that can help aid crime solving, as well as potentially prevent crimes in progress, by offering a location tracking based SOS Widget on the student's mobile phone. In addition, other crimes that could normally go unreported, such as vandalism and bike theft, can also be recorded.

The University of Illinois Police Department will be able to track their criminal investigation faster by having more data, such as the GPS coordinates of the phone, photographs, audio and video links on a map.

1.1 Terminology and Acronym Definitions

The following terms/acronyms are employed within this document:

Term/Acronym	Definition
XP	Extreme Programming
ADT	Android Developer Tools
SDK	Software Developer Kit

2.0 DEVELOPMENT PROCESS - XP

For the development of the Crime Busters Application our project team utilized the Extreme Programming (XP) development process. Given the limited time we had to develop the project and the limited number of developers we felt that this agile approach was appropriate for our needs. XP also gave us the flexibility to try new technologies and change business decisions in a quick and efficient manner. Some of the primary concepts of XP that we utilized include the planning game, refactoring, and continuous integration.

The planning game helped us frequently reevaluate our project and stay on pace with a reasonable timeline while aiding in our task prioritization. Our team had a standing meeting on Saturday mornings to discuss our planning game tasks and what needed to be complete in the following week. We also discussed smaller group meetings (groups of 2 or 3) that would be held in the following week. The user stories, tasks, and collaboration used with this planning process helped us appropriately prepare for our iteration meetings.

Code refactoring was a fundamental process for our team. Mobile technology was something we were not very familiar with when we began this process. It was an excellent learning opportunity for everyone on our team. The learning curve needed for the development of the crime busters app frequently resulted in disparate or unorganized code. Continual refactoring allowed us to create a much more readable and simplified body of code.

Continuous integration was necessary for our team to be productive. We utilized Github for our versioning and committed changes as frequently as possible. This allowed us to build upon each others work and prevented

double work. Testing was also integral to our success. Test Driven Development (TDD) was not a practice that we consistently used, but we did frequently test our code and develop unit tests. Unit testing allowed us to quickly identify issues and remove bugs. The developers in our group had used TDD in the past (notably in CS427), but we felt more comfortable writing the code then testing for this project, therefore TDD was not a common practice within our group due to our preferences.

Pair Programming is a process that we attempted to use as often as possible. Given that we were a team of online students that live in multiple time zones we often found it difficult to strictly adhere to this practice. Knowing that going in we recognized as a group that pair programming would not always be a possibility. We compensated for it by regularly reviewing changes/updates and staying in constant communication via email.

We found XP to be a beneficial methodology for the development of our application. It kept us in constant communication and allowed us to have open conversations about the application and its requirements. We did not strictly adhere to all of its practices, but we felt we extrapolated all of the of the features that would work with our team and the project we were developing.

3.0 REQUIREMENTS & SPECIFICATIONS

The requirements and specifications for the project outline the details of the project deliverables. The Use Cases are the high level items of the main implementation of the project. The user stories are the implementation steps the team took in order to accomplish the use cases.

Actor	Goal	Brief
Android User	Send Crime Report to Authorities	Fill out a Report, which is sent to the web application for later vie police/authorities
Android User	Create an Account	Fill out profile, that is used to provide authentication to login in ar configuration settings
Android User	Provide Media for the police about the crime	Take pictures and video which are submitted with the report and saved for later viewed
Police	View crime reports on map by priority	Login as a police officer, and view by icon indicator the crime importance
Police	View dashboard of reported crimes	Click on the report dashboard, which displays a table of all the reports on the map filtered by date reported

Table 1. Use Cases

3.1 User Stories & Iterations

The user stories and iterations below were used as part of the Agile process to keep track of progress. Iteration 1 had no software product deliverables, therefore it was omitted.

actual	estimated	story description
8 units	8 units	As a police officer, I would like to have the ability to see a big map of the University of Illinois campus and an icon.
7 units	8 units	As a user, I would like to have a user friendly interface for android app.

Table 2. Iteration 2 (Feb 8 - Feb 22)

actual	estimated	story description
18 units	10 units	As a police officer, I would like to have the ability to see icons (student or faculty locations) on the University of Illinois Map.
18 units	6 units	As an Android user, I would like to have a user friendly application that can send data to the server.

Table 3. Iteration 3 (Feb 22 - Mar 8)

actual	estimated	story description
8 units	10 units	As a police officer, I would like to see the image uploaded by the user (if there is any) when I clicked on a user on the map so that I can initially see what is going on.
7 units	8 units	As a police officer, I would like to have the ability to see icons (student or faculty locations) on the University of Illinois Map.
10 units	10 units	As an Android user, I would like to have a user friendly application that can send reports to the server.
9 units	8 units	As an Android user, I would like to notify the campus security during a crime incident by sending low or high priority reports to the police department.
3 units	3 units	As a police officer, I would like to have a log in page so the map can be secured by authentication.
3 units	2 units	As a police officer, I would like to have a remember check box from a trusted computer so that I do not have to log in every time I access it.
7 units	8 units	As an android user, I would like to have the ability to create an account from the app, so that I may use the app.

Table 4. Iteration 4 (Mar 8 - Mar 29)

actual	estimated	story description
8 units	8 units	As an Android user, I would like the app to take photographs and automatically upload them, together with the GPS location, to the police web platform so that the police can have a quick look of the incident on the map.
8 units	10 units	As an Android user, I would like to have the ability to update my profile via the application.
12 units	12 units	As a Police Officer, I would like to have the ability to display filter high level reports and low level reports on the map.
2 units	2 units	As a Police Officer, I would like to have the ability to view my profile name on the website.
2 units	2 units	As a Police Officer, I would like to have the ability to log out of the website.
2 units	2 units	As a Police Officer, I would like to see a different icon for different report types to easily differentiate reports on the map.

Table 5. Iteration 5 (Mar 29 - Apr 12)

actual	estimated	story description
10 units	8 units	As an Android user, I would like the app to take photographs and automatically upload them, together with the GPS location, to the police web platform so that the police can have a quick look of the incident on the map.
11 units	10 units	As an Android user, I would like to have the ability to upload a video to my report.
6 units	8 units	As an Android user, I would like to have the ability to send an audio recording with my report.
4 units	5 units	As an Android user, I would like the app to remember me after successful authentication so that I would not need to log in every time I open the app.
2 units	2 units	As an Android user, I would like have a log out button so that I can still access the log in page after a successful authentication.
8 units	8 units	As a Police Officer, I would like to have the ability to view all or my reports on a consolidated dashboard for easy access.

Table 6. Iteration 6 (Apr 12 - Apr 26)

4.0 ARCHITECTURE & DESIGN

Android Architecture –

Our Android application used the standard android architecture model. XML was used for the user interface (UI) design and java was utilized for the system. Android refers to its OS as a software stack. The base of the stack is the kernel (Google used Linux for this layer). One of the primary purposes of this layer is Drivers that control hardware devices (including our camera functionality). The next level contains the Android libraries and Java libraries. The final layer we will reference is the application framework. This layer helps switch between processes.

Web Architecture –

Our web application follows a 3-tier architecture that cleanly separates the presentation, business logic and data access layer. In addition, we also follow a service oriented architecture to allow communication with our Android application. ASP.NET influenced our design because the technology provide much needed backend functionality. Communication to the database ,login/logoff, authentication, and cookie management was already provided for us to implement with our application.

4.1 UML Diagrams

CLASS DIAGRAMS

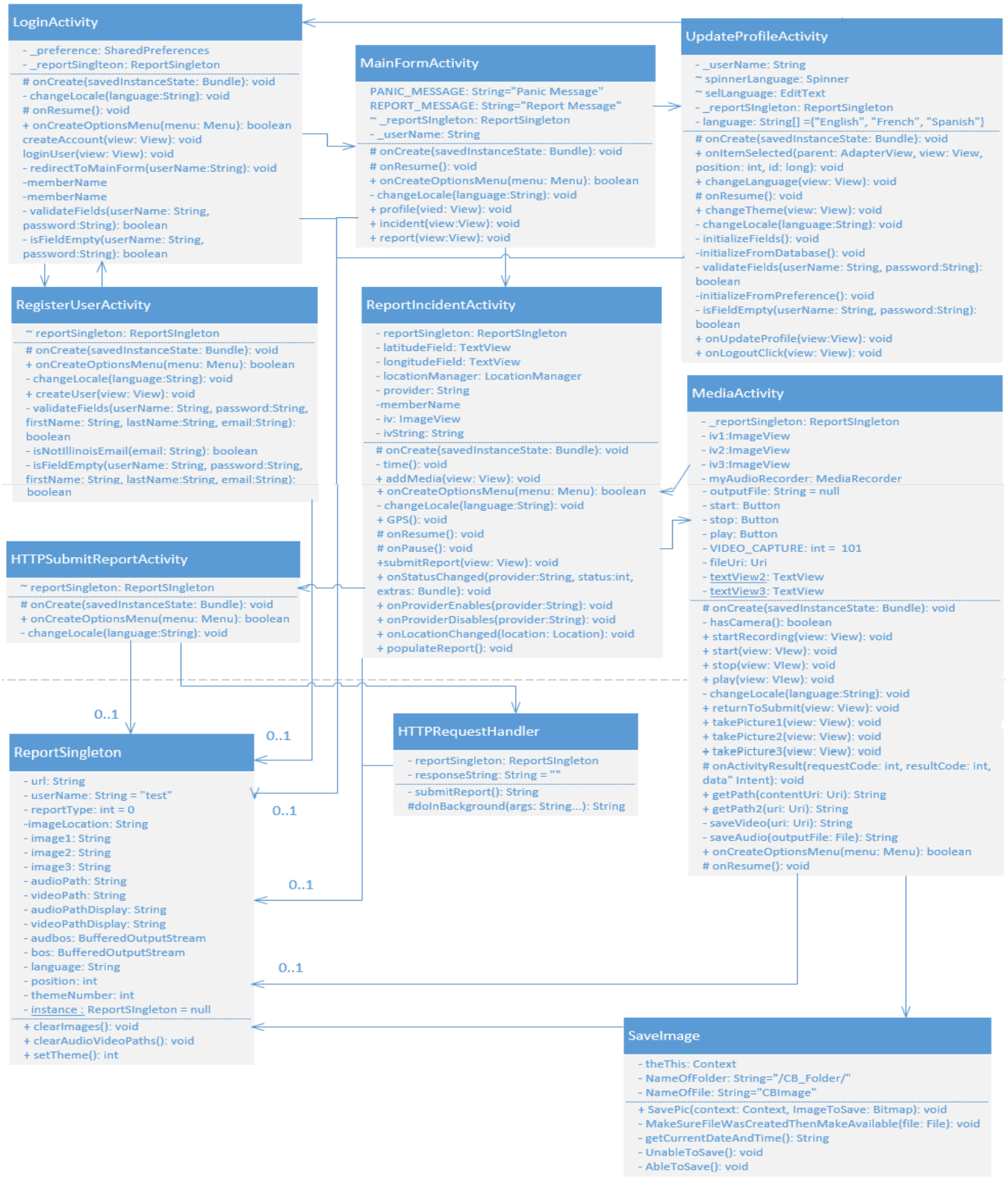


Figure 1 - Android Class Diagram

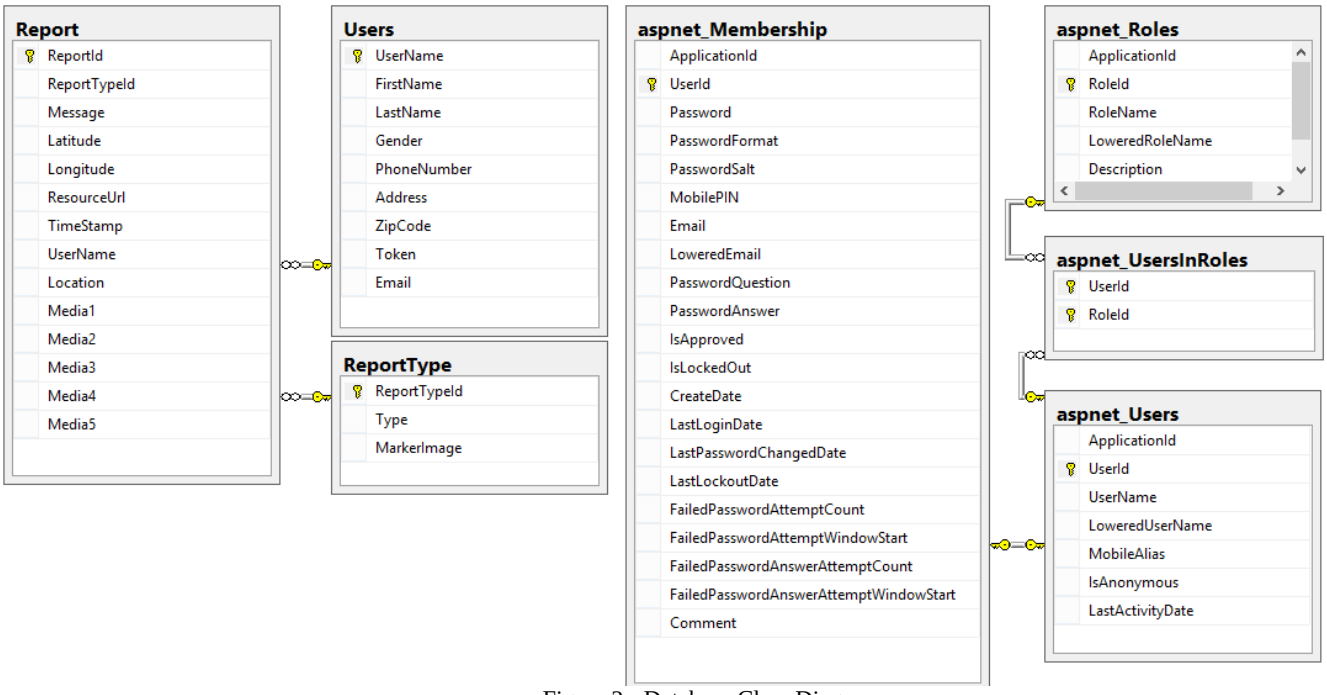


Figure 2 - Database Class Diagram

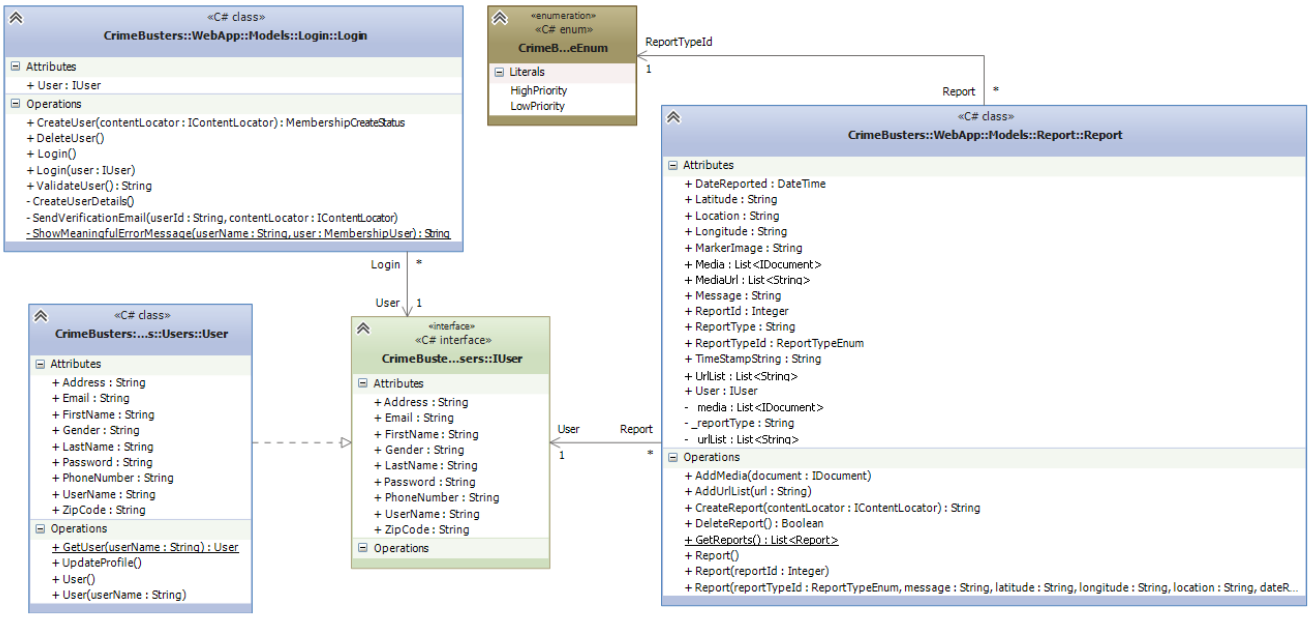


Figure 3 - Login, User, and Report Class Diagram

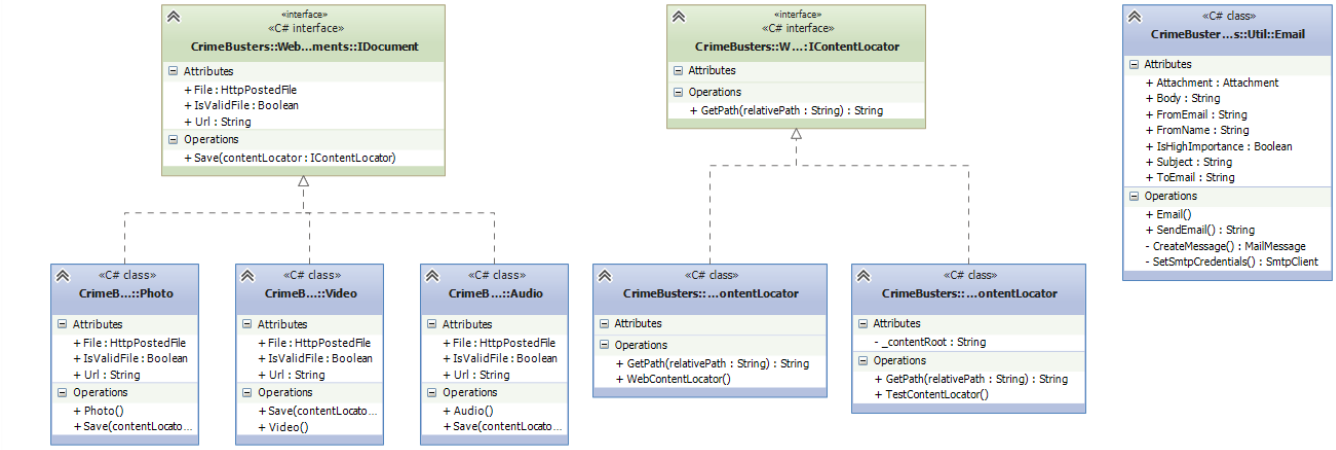


Figure 4 - Multimedia Class Diagram

SEQUENCE DIAGRAMS

Android

Capturing and sending media –

This sequence diagram explains the process of sending pictures, audio and video files to the web service for low priority reports. All the fields are optional, it is as per the user's convenience what needs to be sent. Once the user is logged into the application and he clicks on the low priority report, the app populates GPS coordinates, if the location service is on. He will see a button called "Add Media" which will take him to the page where he can upload 3 images/an audio and a video file. All of the media will be saved in the Gallery, when the user captures it and once he is done uploading whichever media he wants, a report will be created and all of the media files will be sent as attachments to the web service (police). Error handling is implemented so that all the exceptions are captured.

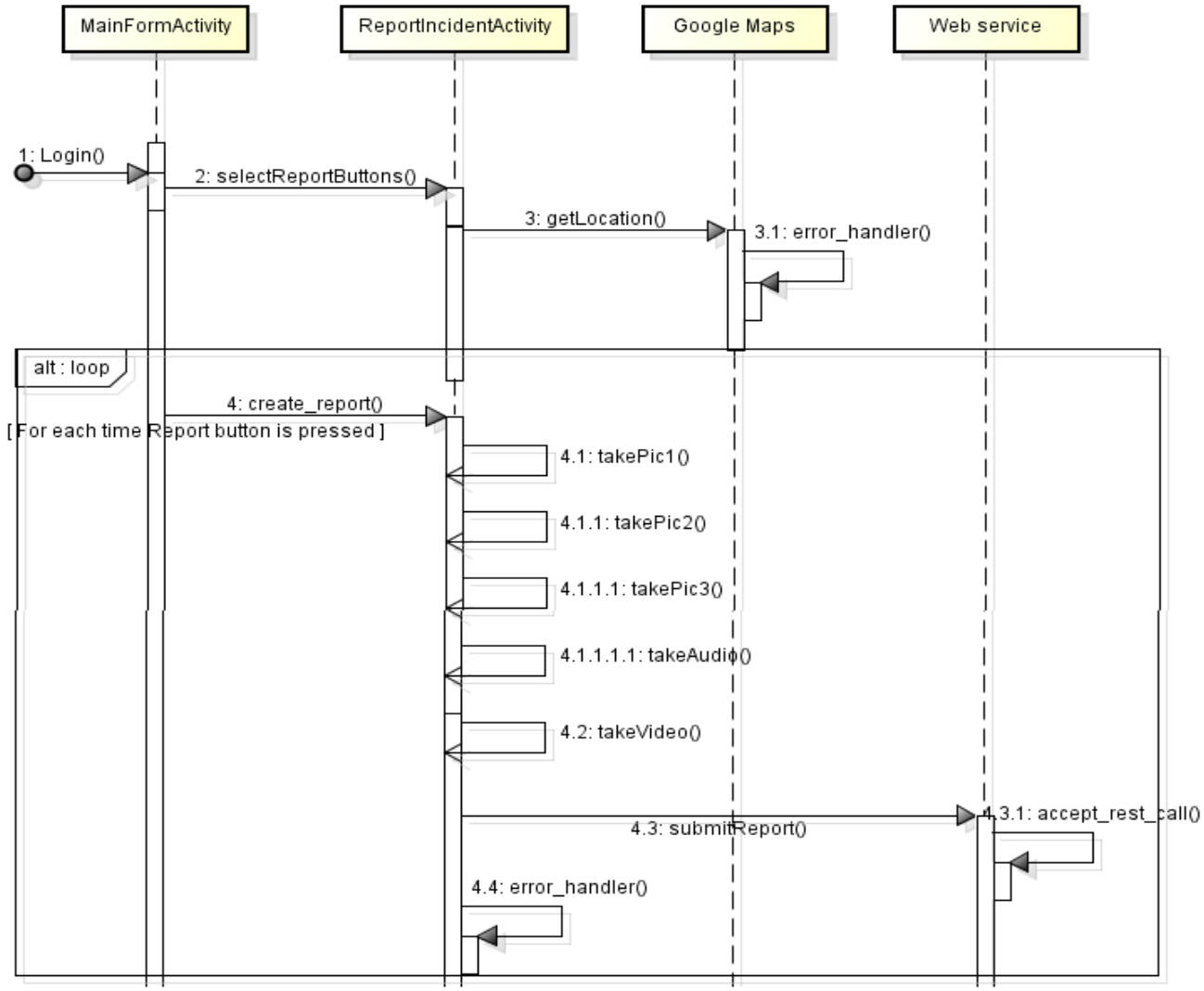


Figure 5 - Android Report Sequence Diagram

Setting User preferences and logging in –
Initially, the user has to register himself in the system and perform email authentication. Once he does this, he can log in to the system and after his credentials are validated by the system, they are remembered by the application for future use. Hence, the next time an existing user opens the application, he will be auto-logged in and will be directed to the main form, where he can create low/high priority reports or update his profile. We are giving the user an option to set the look and feel of the application as per his convenience, by offering dynamic themes and language preferences. If he clicks on "Update Profile" button, he is taken to preferences screen, from which he can set his favourite app background from 3 available choices. We are also allowing him the option to select language of his choice by selecting between English, French and Spanish. His choices are reflected in the application persistently once he makes his selections. This page also allows him to log out, which re-directs him to the application login page.

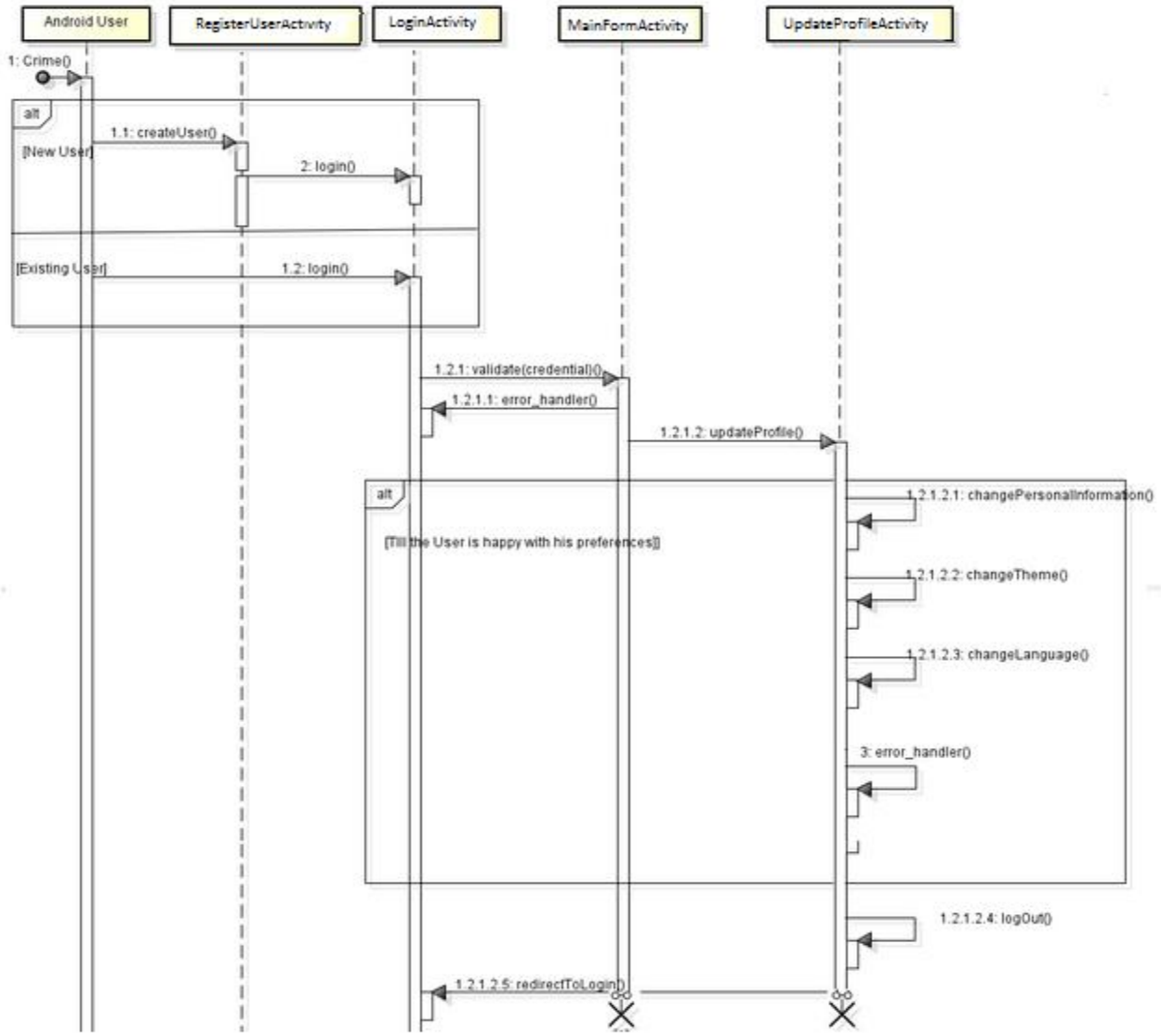


Figure 6 - Android Registration Sequence Diagram

Web Services

In order to login into the web application provided for police personnel the user must first enter their credentials. Login.js provides the functionality to login and call on the web services to do authentication for the user. ASP.NET provides API in the Membership Class and FormsAuthentication to check if the user is in the database and if they are in fact in the “police” role, which allows the ability to enter the web application and view reports.

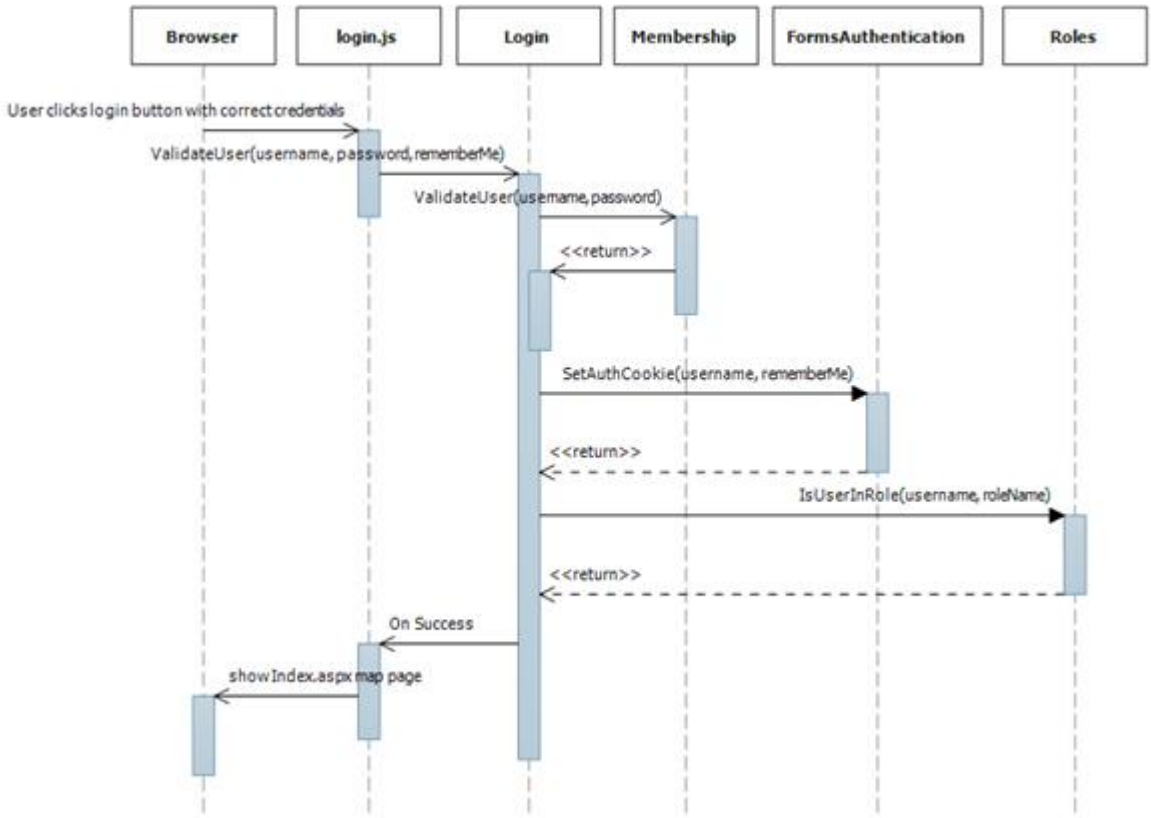


Figure 7 - Login to Web App Sequence Diagram

In order to view the reports on the Web Application the user has to be directed to Index.aspx. Either this is by login or revisiting the page. On Index.aspx the index.js handles all the map functionality. It uses Google’s API to get a map for the area. Then it plots users on the map. It uses the Web services in Index to grab the Reports which ultimately goes to the ReportDAO class to execute the stored procedures on the database. The resulting information is parsed by the Report Class and sent back up to the client with index.js. Index.js handles the rest of the functionality of taking the reports, and labeling which ones are hi and lo report types and displaying on the map appropriately.

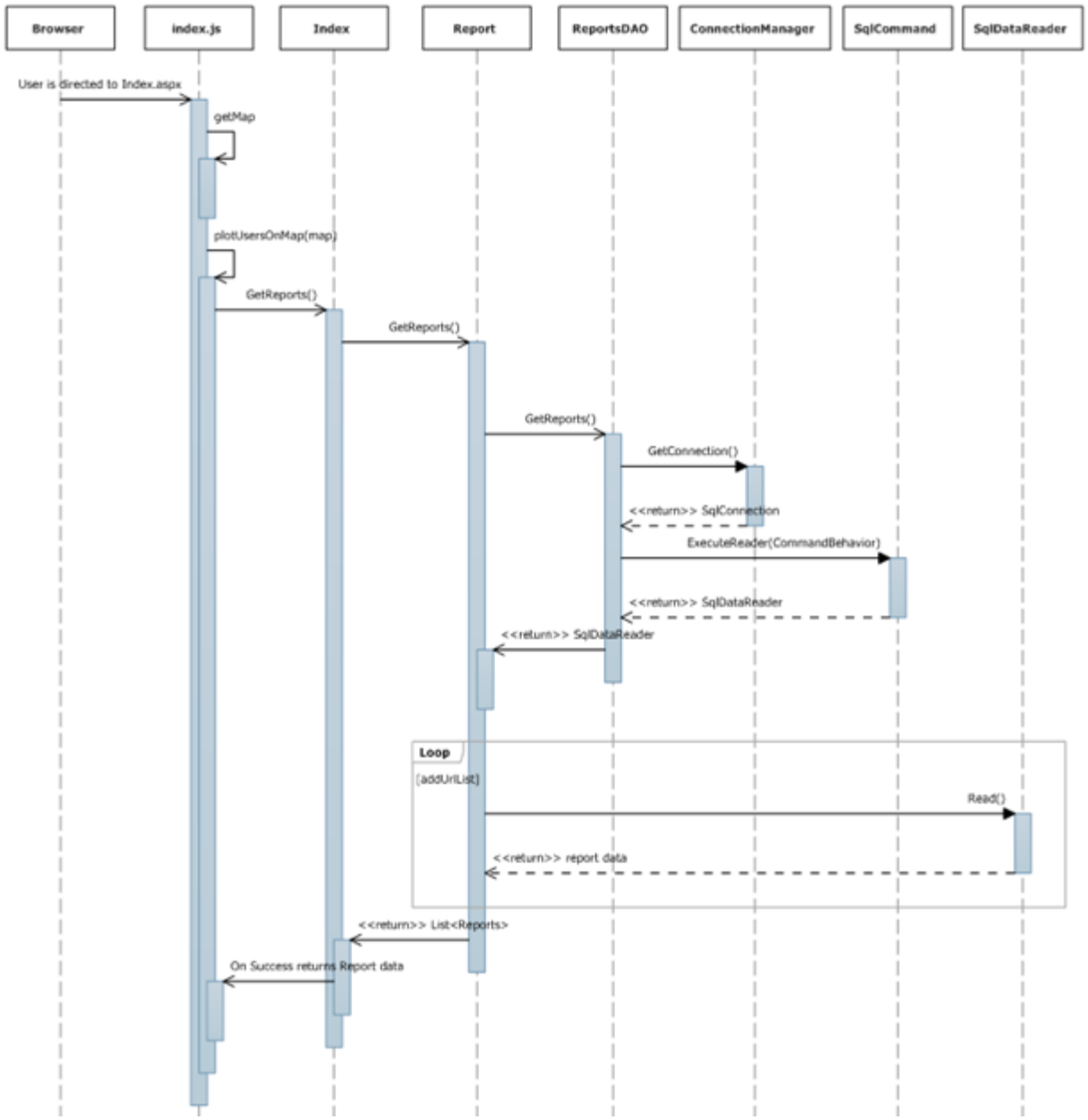


Figure 8 - View Reports on Web App Sequence Diagram

5.0 FUTURE PLANS

Our team feels that the Crime Buster App is a useful tool that fills a need within the campus environment. Given the limited amount of time of the spring semester we believe that we accomplished a great deal, however, we feel that more can be done. We would like to test our application and get feedback from the local police department. We would also like to make the application available for iOS and windows smartphones instead of just android.

Everyone on the team did a fantastic job contributing their knowledge and skills to this project. It was the first time any of us had developed a mobile application and we went through growing pains together. The XP process helped us constantly collaborate and discuss not only our findings but our concerns as we progressed. We believe in our product and plan to continue enhancements on it after the conclusion of this semester.

Individual Comments/Personal Reflections from Team Members -

"*Pair Programming is a powerful tool. Use it.*" --Boris

"*I learned that the XP process really works, and gives you the flexibility to change as the project evolves.*" --Josh

"Extreme Programmings emphasis on collaboration and having the ability to bounce ideas off of team mates proved to be invaluable. Due to the process and the vibrancy of this team, at no point did I feel isolated or lack direction." --Matt

"Using Github and maintaining an iterative process helped us stay on track. It removed large amounts of redundancy." --Chris

"Mobile software development was new to everyone on the team. Frequent refactoring allowed us to experiment and explore capabilities, while adding additional skills to our resumes." --Khushbu

Appendices

Appendix A – Software Requirements

Operating System: Android OS
Database: SQL Server 2012
Android IDE: Eclipse for Android
Web Services: Visual Studios 2013

Code Repository: https://github.com/CrimeBuster
(Location Services should be on for this application, for accurate location information)

WebService -
Browsers: IE/Chrome/Firefox

Appendix B – Installation Instructions

Android phone –

For development purposes the software engineer will need to download the Android SDK (http://developer.android.com/sdk/index.html?hl=sk). This download will include the Android Developer Tools (ADT). Which includes Eclipse for Android.

After downloading and opening eclipse you can click on file -> import. Click on the Git folder then click on the Projects from Git -> Next -> Clone URI. In the Source Git Repository screen that pops up, you need to enter the following information...

URI: https://github.com/CrimeBuster/CrimeBustersAndroid.git
Host: github.com
Repository path: /CrimeBuster/CrimeBustersAndroid.git
User: Your github user name
Password: Your github password

In the Branch Selection screen that pops up you can leave all of the branches selected and click next. The following screen is setting the local destination. Create a directory on your local drive to save the information and in the initial branch drop down choose the master branch. Click next.
The following screen allows you to import your project into Eclipse. Select Import existing projects and click next. Select the crimebusters project and click finish on the next screen. This will import the project into Eclipse for development purposes. At this point you can use the Eclipse IDE to run the code and do any updates/refactoring that you wish.

To run the android application you can either connect your android phone to your computer or you can create an emulator and run it on your PC. For instructions on how to create/run an Android emulator, you can visit http://developer.android.com/tools/building/building-eclipse.html#RunningOnEmulatorEclipse.

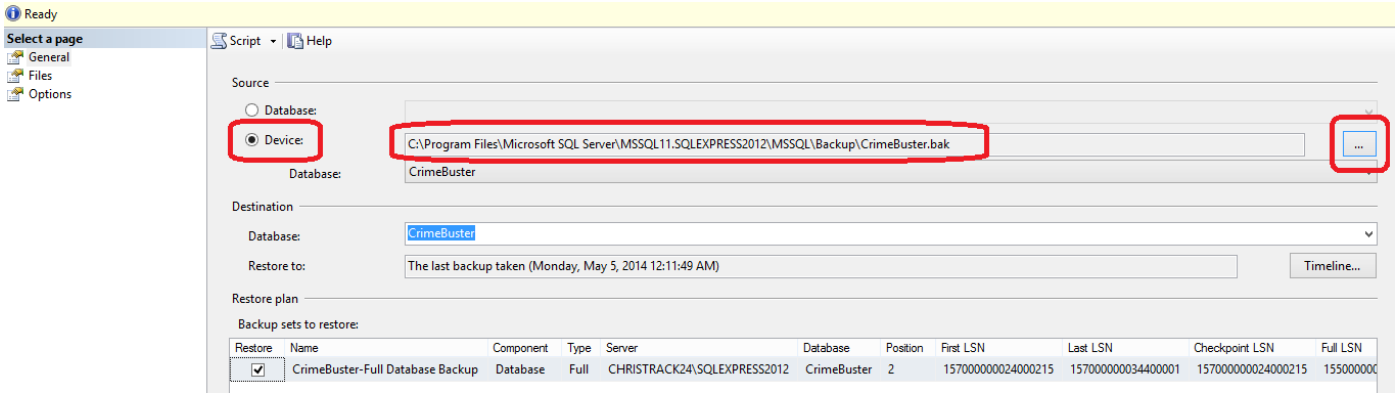
Database -

To setup the database for the web project, you need the following:
1. Microsoft SQL Server 2012 (database server)
2. Microsoft SQL Server 2012 Management Studio (tool to access the SQL Server database)

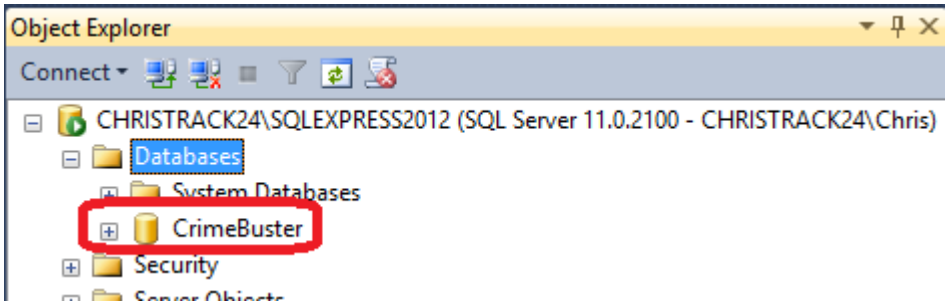
You can find step-by-step instructions on how to install the necessary tools here (http://blog.arvixe.com/manage-your-database-with-ssms-2012/).

After successful installation of the required software tools, open the SQL Server Management Studio, then connect to the server. On the "Databases" folder, right click to open the context menu then click on "Restore Database".

Choose Device, then locate the CrimeBuster.bak file. After which, everything will be automatically populated as shown below:



Click OK to see the database added to your local database server.



WebServices -

I. To set up the web project for a new developer joining the team, you need the following:

1. Visual Studio 2013 Ultimate (.NET 4.5 Framework) https://webstore.illinois.edu/Shop/product.aspx?zpid=2039
2. Git Client - Recommend Source Tree. http://www.sourcetreeapp.com/
3. GitHub Account

Once you have Visual Studio installed, you can clone the Web Application code from https://github.com/CrimeBuster/CrimeBusters

It consists of three projects: CrimeBusters.UML, CrimeBusters.WebApp and CrimeBusters.WebApp.Tests. From the project name itself, the first one consists of the UML diagrams, the next contains the core web application and the last contains the unit and UI tests for the web application.

After downloading the codes, open Visual Studio as an administrator. Go to the File Menu > Open > Project/Solution then look for CrimeBusters solution file (SLN file). To run the application, simply go to the Debug menu > Start Debugging. The login screen will show up. You can use the following credentials as a test user:

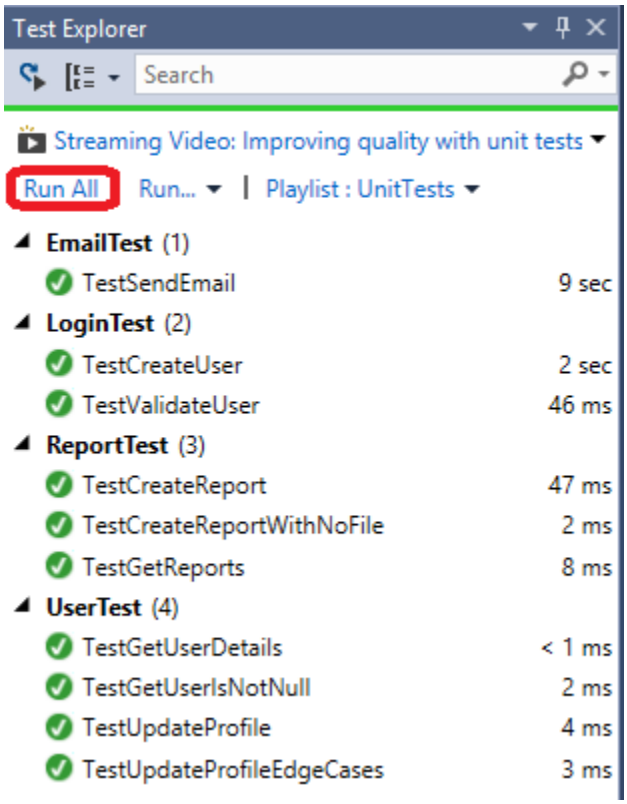
username: **police**
password: **test123**

II. To deploy the application to the live site, simply right click on the CrimeBusters.WebApp project, then on the context menu, click on "Publish". You will need to enter server credentials which we will not provide here due to privacy reasons. You can find more information on how to quickly publish a web project using one-click publishing here (http://msdn.microsoft.com/en-us/library/dd465337(v=vs.110).aspx).

A packaged version, similar to APK files in Android, of our web application that can be manually installed on any IIS server can be downloaded here (https://www.dropbox.com/sh/hx3o0luls9bnvyj/Hz84RpXmif).

Appendix C – Testing Instructions

For the web project, the test is integrated in Visual Studio and can be easily run by going to the Test Menu > Windows > Test Explorer to open the Test Explorer window. Simply hit the “Run All” if you want to run all the test cases.



For the Android application, you need to clone the test project at https://github.com/CrimeBuster/CrimeBustersAndroidTest. Please note that this project depends on the main Android project, so you also need to download it at https://github.com/CrimeBuster/CrimeBustersAndroid.

Before you can run the automated tests, you need to have a running emulator. For instructions on how to create/run an Android emulator, you can visit http://developer.android.com/tools/building/building-eclipse.html#RunningOnEmulatorEclipse for documentation from Google. You also need to set up your Eclipse IDE to use the Espresso Framework. You can find step-by-step instructions at https://code.google.com/p/android-test-kit/wiki/Espresso#Getting_Started.

Once everything is set up, you can simply right click on the CrimeBustersAndroidTest project, then on the context menu, go to Run As > Android JUnit Test. The debug window will open up with passing and failing tests. You can also see on the emulator that the fields are automatically populated.