VELOCITY RAPTORS

Velocity Raptors Project Outline

William (Aidan) Maher Nic Durish Jackson Keenan
Anthony Mazzawi

University of Guelph
CIS*3760 Software Design
Dennis Nikitenko

April 1, 2015

Contents

1 Functionality

1.1 App Vision

Currently, the community of Guelph lacks a consistent, accurate and functional Public Transit phone application. The applications which are currently on the market either lack Global Position Satellite accuracy (RideGuelph, GryphPhone) or necessary community attention (Nextbus). Nextbus is currently the go-to in terms of public transit feedback, as they guarantee times accurate to 1 minute. But in Guelph, Ontario we often see buses off route by over an hour. This is due to the lack of GPS's on some of the buses and to unscheduled route changes by Guelph Transit. The Cannon outlined some of the systems flaws in this article - `http://www.thecannon.ca/page.php?id=24&n=13970`

We intend on building an Android Application that provides Nextbus' GPS information in a functional interface (which includes an actual map). However, unlike Nextbus we intend on tracing each bus route in Guelph, so we can easily remove buses that are too far off-route or off-schedule. We may then either inform Guelph Transit administration of the error or fall-back to the scheduled times and notify the user of the changes.

1.2 Users

Transit Riders will use the application to get feedback on the current position of the bus. They can view bus locations, view their location, check the current state of the bus (GPS or schedule), check the bus schedule, favourite bus routes, change location settings, will be able to leave and view comments for each bus.

Transit Drivers will be able to use their phones GPS for routes if the bus is unable to be reached.

Transit Administration will be able to change a buses state from GPS to scheduled, and update the current location of each bus.

1.3 Requirements Table

#	Type	Requirement	MuSCoW
1	System	Build and organize MySQL tables for transit schedule	Must
2	System	Create excel parser, to pull transit times	Must
3	System	Fill MySQL tables with transit times	Must
4	System	The system provides a Home-Button, to navigate back to the home menu	Must
5	User	Users are able to view Schedules in a Graphic User Interface	Must
6	System	The system pulls GPS locations or Nextbus times	Should
7	System	The system calculates the buses current location and estimated times to next stops	Should
8	User	The user is able to select stops and buses to view estimated times (No map is available)	Must
9	System	The system has integrated the Google Maps API	Should
10	System	A map interface is built using the location of stops and buses	Should
11	System	The pins on the map are linked to their respective bus and stop pages if map is implemented	Should
12	User	The user is able to view a map of their community	Should
13	User	The user is able to view their current location	Should

14	User	The user is able to select stops and buses on the map	Should
15	System	The system shows NextBus times and scheduled times	Should
16	User	The user recognizes GPS times vs Scheduled times	Should
17	System	A 'Settings' page is available from the menu	Must
18	System	An 'About' page is available from the menu	Must
19	User	The user is able to change settings and learn about the app using the menu	Must
20	System	The system is able to locate the user and find the nearest stops	Should
21	System	The user is able to toggle 'Location'. The viewpoint zooms into the users current location and the nearest stops are shown.	Should
22	System	The system is able to update instances of the database if requested	Should
23	System	The system is able to auto-update the database if requested	Should
24	User	The user is able to update their bus schedule from the 'Settings' menu	Should
25	User	The user is able to toggle 'Auto-Updates' from the 'Settings' menu	Should
26	System	The system is able to save a users favourite between sessions. The system orders these icons.	Should
27	System	The system is able to add and clear favourites from a users list	Should

28	User	The user is able to save favourites by holding down an icon	Should
29	User	The user is able to to 'Favourites' to view all of their selected favourite buses and stops	Should
30	User	The user is able to clear favourites through the 'Settings' menu	Should
31	System	The system creates a 'Help' page in the menu, to help the user navigate the application	Should
32	User	The user is able to navigate to the 'Help' page using the main menu.	Should
33	System	The system builds a database for Administrator accounts and Transit-Driver tickets	Should
34	System	The system allows users to sign in as a Driver or Administrator	Should
35	System	System updates bus routes based on location of activated tickets (Transit-Drivers phone GPS)	Should
36	System	System is able to deactivate invalid tickets, or tickets too far away from bus route.	Should
37	System	The system is able to create tickets, designated to certain bus routes	Should
38	System	The system deletes tickets after a designated amount of time	Should

39	User	Administrators are able to sign in to create and delete tickets. Tickets decay over time	Should
40	User	Transit-Drivers are able to log-in using ticket numbers and accept using their phone as the routes current location	Should
41	System	The system is able to store user comments and user's average delay in a database	Could
42	User	The user is able to submit what the average delay of a bus is at a certain stop	Could
43	User	The user is able to comment on specific buses and stops and view others comments	Could
44	System	The system provides a search functionality to search for, buses, stops and addresses	Could
45	System	The user is able to search for buses, stops and addresses using the search function	Could
46	System	The map should have colours on the pins indicating where the bus is located based on the current time (ex. Red for the bus has gone by, Green for next coming stop, Yellow for coming soon etc)	Should

1.4 Features

Numbers in brackets are the rated score of the feature by each team member (10-50 increments of 10)

Must haves

Complete list (database) of all city buses and schedules. (200)

Ability to track bus location via GPS to give users a more accurate GUI allowing users to interact with the database to attain the transit information they need (200)

A map interface showing the physical location of stops / buses. (180)

Should Haves

Location based Services, to quickly find nearby buses / stops / you. (140)

Ability for users to select frequently used or favourite buses for easy access. (130)

Ability for app to update the buses/schedules. (120)

Ability for transit drivers to use their phones as a GPS locator for the bus if the bus doesn't have a GPS on-board. (110)

Would Be Nice

Ability for transit admins to confirm whether or not buses have a GPS on-board. (90)

Ability to crowd-source bus arrival information from users (This is done by users submitting reports on the time of bus arrivals, independent of posted times) (70)

Ability to comment leave / review comments for buses / routes (60)

2 Stuff

2.1 Platform

The target platform this project is a mobile app which can be used whenever someone needs to catch a bus. More specifically, it will be developed on Android.

2.2 Development Platform

The development platform is Android Studio and the app will be done in Java and XML.

In order to store data Python will be used to parse the Guelph transit Excel files and put it in our SQL Database.

2.3 Source code Storage

The team will be using git to cooperatively develop the project. The source code as well as the documentation for this project will be hosted on Bit Bucket. The repository is located at :

https://yourbitbucketusername@bitbucket.org/JacksonKeenan/3760transit.git

SSH : git@bitbucket.org:JacksonKeenan/3760transit.git

2.4 Third Party API's and SDK's

For APIs, the plan is to use the Google Maps API and possibly NextBus'. The original plan was to contact Guelph Transit to use the GPS on their buses. Unfortunately, they have not yet responded which means the next step in the plan is the NextBus API.

In order to parse the Excel files Python will be using a third party library called xlrd (Excel Read).

2.5 Artwork

The group will be paying a graphic artist (Charlotte Gao) from Toronto to make a logo for the company. It will cost $10 which the team will cover. She works for ExperiencePoint, a company selling leadership training and development process training, "Working with most Fortune 100's top companies and the world's leading business schools." - http://www.experiencepoint.com/

The artwork will be submitted with the submission of the design document on Feb 1.

2.6 Data Storage for App

A database will be used to store the Guelph Transt bus schedule. Guelph Transit has posted their schedule online in a Google Doc, but it would be wise to save that information in our own database just in case they decide to remove that information. The database will be set up using Python, mySQL and PHP will be used to connect it to Java (the app).

Python will parse the Guelph Transit Excel files using the xlrd Python package then store it in the database. The app will then pull the SQL Database and use Android's internal SQL-Lite to store the whole database. If any changes are made to the database, the database will automatically update and re-populate the Route and Stop data structures to be up to date.

2.7 Starting Point

Android/Google has recently released in the past two years more and more documentation, Devkits, and Documentation how to make Android apps. There are thousands of coding tutorials on Android made by third party users and people who make tutorials. We have begun studying and watching videos on how Android folders are set up in the app, we have found tutorials on connecting mySQL to PHP to Java for database access on Android applications.

3 Plan

*Refer to feature-list for milestones requirements.

3.1 Milestone 1

At this point we should have a program to fill the database (SQL) for the application. This database will be populated with the city's various bus schedules and routes. Using the application, users are able to navigate to each bus and stop to view its respective schedule.

3.1.1 Completion Date:

Wednesday, February 11th, 2015

3.1.2 Responsibilities:

Aidan & Anthony: Program for parsing data and filling the database. Jackson & Nic: UI for viewing route schedules.

3.1.3 Requirements:

(1, 2, 3, 5)

3.2 Milestone 2

A home button is introduced, to allow users to navigate back to the Home screen. The schedules should be available in the user interface, however currently a map is unavailable.

3.2.1 Completion Date:

Wednesday, February 25th, 2015

3.2.2 Responsibilities:

Aidan & Anthony : Writing Route and Stop Java classes. Pulling parsed data from the database and populating Java classes inside the Android app for Nic and Jackson to use to view on the UI. Jackson & Nic: UI for navigating through buses / stops.

3.2.3 Requirements:

(4, 8)

3.3 Milestone 3

The team will attempt to implement the Google Maps API and connect our information with the map, including the pins on the map. If the team cannot implement the map, the application will be a very basic UI showing the schedules from Guelph transit.

Supposing the map is implemented, information is now graphically represented on the map using the Google Maps API (Pins[buses, stops, the user], Routes). The user can interact with all of these objects (click on pins to get additional information, etc). Scheduled information is now available in the Schedule page on the main menu.

3.3.1 Completion Date:

Sunday, March 15th, 2015

3.3.2 Responsibilities:

Aidan & Anthony: Smooth out parser bugs / Obtain stop coordinates manually Jackson & Nic: Introduce Google Maps, integrate location pins, and polish UI

3.3.3 Requirements:

(6,7,9, 10, 11, 12, 13, 14, 16)

3.4 Milestone 4

A Settings and About page are introduced. The Current Location feature is introduced, allowing users to find their current location as well as easily view nearby stops. The auto-updater and schedule updater are now introduced to the Settings page.The team will have a script that should be able to scrape the estimated time of arrival of each bus for each stop.

3.4.1 Completion Date:

Sunday, March 22nd, 2015

3.4.2 Responsibilities:

Aidan & Anthony: Put scraper on server, set up JSON POST request / Update schedule on application and server-side if requested by user.
Jackson & Nic: UI for; settings, and about page, as well as the UI for the current location feature. This will include the bottom panel and the map overlay.

3.4.3 Requirements:

(17, 19, 20, 21, 22, 23, 24, 25)

3.5 Milestone 5

Users are now able to favourite stops and buses by holding down the icon. The 'Favourites' feature is introduced, allowing users to save their favourites between sessions. Users can clear their current favourites from the 'Settings' page.

3.5.1 Completion Date:

Sunday, March 29th, 2015

3.5.2 Responsibilities:

Aidan & Anthony: Implement server web-scraper to application requests / Implement schedule update to automatically update Jackson & Nic:Favourites UI (Clear option in settings), Help Page

3.5.3 Requirements:

(15, 26, 27, 28, 29, 30)

3.6 Milestone 6

Main features should now be implemented into the application. The team should be debugging, polishing and fixing up any problems the application has. Such as resizing the map properly, A 'Help' and 'About' page is also introduced to the main menu, giving users a preview of how to navigate the application and a little bit about the developers. Possibly a colouring system for markers to indicate where the bus is located will be introduced.

3.6.1 Completion Date:

Sunday, April 5th, 2015

3.6.2 Responsibilities:

Aidan & Anthony: Implement map colouring schema / Fix any remaining data bugs
Jackson & Nic: Polish UI / implement Help and About page

3.6.3 Requirements:

(18, 31, 32)

3.7 After 3760

The following is would-be nice features which got pushed back to beyond milestone 6 since these features are heavily inter-related and are not necessary for main functionality of a transit app.
A setting is introduced to allow users to sign in as a Transit Driver or Administrator. Administrators have a permanent username and password which they can use to sign on and create or delete Transit-Tickets. These tickets will be associated with a unique bus route and a decay time. Drivers may use these tickets to sign in to the application and use their phones GPS to update the buses current location. Tickets are deleted after their delay time.

3.7.1 Completion Date:

TBD

3.7.2 Requirements:

(33-45)

3.8 Alpha Submission

By Week 7 we hope to release the Alpha version of our application. This version will include all features in Milestones 1 & 2. If time permits we hope to include some of the features included in Milestone 3. These include the requirements; 1-8 and possibly 9-16.

3.8.1 Use Cases

The Transit Rider will be able to select buses/stops in the left/right scrollable bottom bar and in the Options (top right 3 dotted button) Schedule page (which is up/down scrollable).

3.9 Release Candidate

By week 11 we hope to have a bug-free application that has been thoroughly tested. Depending on if we can get the Google maps API implemented properly the application will have a map showing all stops with pins and the user is able to favourite stops. The phone application will also be able to detect whether or not the database is different than from what's on the phone and update. This could also be implemented by having the application check for a "version" of the database and whether or not it matches the "version" on the server every time the application is started, or after a certain amount of time has passed. Requirements for this release include the previous from the Alpha, and requirements 17-30 (Except Req 18).

3.9.1 Use Cases

Including the use cases from the Alpha, the user will be able to (*depending on if maps can be implemented):

The Transit Rider will be able to : View/Select Bus Locations and View/Select the scheduled time or estimated times for Stops. Select a route or stop and favourite it. View/Select previous favourites. Show the users location (based on if we get maps implemented) and show nearby stops. View bus locations and select pins of stops to see estimated or scheduled times. Change application settings such as update schedule,

clear favourites and toggle auto-updater. Show the Transit Rider's location by somehow putting some form of indicator on the map (possibly blue circle).

3.10 Gold Master

The team will smooth out UI design and implement the Help and About page in the application. These need to be done last since we will not be able to entirely write these pages without actually having a final product. Pin colouring will be introduced so the user can easily see where the bus is about to arrive based on the schedule times.

3.10.1 Use Cases

View application info in Help/About. View approximate location of bus based on schedule times.

4 Architecture

4.1 Use Case Diagram

Please See PDF included with this document

4.2 Class Diagram

Please see PDF included with this document

4.3 Entity Relationship Diagram

Please see PDF included with this document

4.4 Rough Paper Prototypes

Please see Paper Prototype folder included with this document