CAPSTONE FINAL REPORT

RETAIL STORE NAVIGATION:
AN APP FOR MODERN SHOPPERS

GROUP NUMBER:
10

NAMES AND STUDENT NUMBERS:

LINGFEI LI	998211046
KAMMY NG	998125461
MEGHNA PATEL	998404873
ZHIXIN ALICE YE	998039689

DATE:
APRIL, 2015

ABSTRACT

Consumers need to regularly shop for groceries. However, grocery stores are often large and complex, resulting in: 1) customers being unable to locate their desired items, and 2) significant amounts of time wasted searching for items. The objective of the project is to create a working prototype of an Android application that will reduce the amount of time spent shopping for groceries in a large retail store. It will guide customers through an optimal path in order to obtain items from a shopping list that they input. The app will have a user interface that contains a database with search suggestions that allow users to input items into a shopping list. A path optimization algorithm will be implemented to determine the optimal path. A map, with an optimal path and suggested time required to traverse the path, will be displayed to guide the customer to retrieve items on their list. Upon implementation of the project, the efficiency of the application will be tested by measuring traversal time through the store using the application.

TABLE OF CONTENTS

1.0 INTRODUCTION

1.1 MOTIVATION

Shopping is a regular task in everyone's daily living activities, whether it is out of necessity or for leisure. Despite the recent trend in online shopping, there will always be a need for physical retail stores due to the need of instant gratification among shoppers. In particular, groceries are items that generally require prompt and constant purchase. Grocery stores are often large and complicated; resulting in a large amount of time wasted looking for and obtaining items. Today's society desires efficient and easy methods to complete their tasks. Thus, there is a need for solutions to alleviate consumer stress by reducing time spent on menial tasks such as grocery shopping.

1.2 BACKGROUND RESEARCH

The following collection of existing research and technology focuses on improving a customer's satisfaction by reducing shopping time and simplifying their search process.

Large retail stores such as Metro and Wal-Mart have created apps which provide functions and services such as: displaying flyer offers and coupons, product availability and prices in stores, creating a shopping list for users through typographical or scanner input, and store finders [1] [2]. These apps are free and helpful in determining product information and allow the user to keep track of a shopping list. However, it does not assist the customer in finding items at a specific location within the store or in reducing shopping time.

Grocery stores are researching "Smart Shopping Carts" to aid customers as they shop [3]. Whole Foods is testing a motorized prototype of a grocery cart [4]. The cart uses Kinect sensor technology to follow customers as they browse the store, track their shopping list, scan items in the basket, and store credit card information for checkout [4] [5]. This is advantageous as it facilitates faster checkout and item tracking, thus reducing shopping time. However, prototypes do not have a map or tracking system to guide customers to find items and will require a large implementation cost for stores.

Philips is researching a trial GPS app to help shoppers navigate supermarkets [6]. The app is linked to a system of "intelligent LED in-store lights" that creates a GPS grid. It allows shoppers to enter items from recipes, lists or suggestions and outputs the best route to retrieve the products. The app is beneficial because it guides the user through the store to retrieve items using an indoor GPS system. However, this requires the smart light system to be installed at the store, which demands a larger cost and effort to implement.

Aisle411 is researching an indoor positioning system which is integrated into an app to map products by aisle location [7] [8] [9]. The app aims to use its indoor retail mapping and location service, and mobile platform allows shoppers to navigate through the store [10]. Its current implementation provides a floor plan that displays the position of products from a shopping list inputted by the user [11]. This is useful; by providing a free user interface that guides customers through the display of product locations, it will save time spent in searching for items. However, the current version does not display an optimal route to reduce time spent to retrieve the items [11].

1.3 Problem Definition & Project Description

Consumers need to regularly shop for groceries. However, grocery stores are often huge and complex, resulting in: 1) customers being unable to find their items, and 2) large amounts of time wasted searching for items. Despite vast quantities of existing solutions, they either: 1) require high implementation cost, or 2) do not provide capabilities to reduce shopping time. The project aims to create a solution in the form of a low-cost android app that does not require additional hardware apart from a smartphone to aid customers by guiding them to retrieve items in an optimal time. The app will have a user interface that contains a database with search suggestions that allow users to input items into a shopping list. A map, with an optimal path and suggested time required to traverse path, will be displayed to guide the customer to retrieve items on their list.

2.0 PROJECT GOAL:

The objective of the project is to create an app that will reduce the amount of time spent shopping for groceries in a large retail store by guiding a customer through an optimal path to travel in order to obtain items from a list they input.

3.0 SPECIFICATIONS:

3.1 REQUIREMENTS

Requirement	Details
#1: Platform	• The app must be built and tested on an Android mobile platform
#2 :User Interface	• The app must provide a touchscreen user interface that allows them to input items to a shopping list
#3: Storage	• The app must construct, store and display a shopping list of at least 10 items based on user input successfully
#4: Display Floor Plan	• The app must display a store floor plan with locations of the inputted grocery items indicated on it
#5: Optimal Path	• The app must provide an optimal path through the store in order to obtain items on the grocery list
#6 & #7: Time Prediction	• The app should provide an approximate time required to traverse the optimal path in order to obtain the items • The predicted time will be compared to the actual time along the optimal path (under normal conditions). The error is to be less than 5%
#8: Database	• The app must provide a database of categorized items from the store

Requirement	Details
#9: Search Suggestions	• The app must provide a search suggestion drop-down list based on user input with at least 1 suggestion
#10: Privacy Items	• The app must not request for the user's private information

3.2 Constraints

Constraints	Details
#1: Mobile Device	• The user must have an Android smartphone and the app installed
#2: Memory Usage	• The total size of the app must not exceed 4GB [12]
#3: Item Availability	• The items on the grocery list must be sold in the store
#4: Registered Map	• The store must have their floor registered in the app
#5: Wi-Fi	• The store must have Wi-Fi available

3.3 Objectives

Objectives	Details
#1: Price Optimization	• The app will compare prices of each item on the list at nearby grocery stores and output the location that will minimize cost
#2: Track User Location	• The app should track the user's location inside the store
#3: Ease of Shopping	• The app should reduce amount of time a user spends to shop

4.0 System Block Diagram & Description

The system consists of three components; the user interfaces, the algorithm, and database. The user interface component contains a shopping list module and a map display module, which allows the user to input and search for items and view the map respectively. These access the keyword search and path optimization algorithms respectively to function. The algorithms aid in item searching and optimal path generation. The database component contains the shopping list, store item and store floor plan information modules. The shopping list and store item modules interact with the keyword search by providing and storing data on the store items. The store item and store floor plan modules provide information to the path optimization algorithm to generate maps and locations of items. FIGURE 4.1 displays the System Block Diagram.

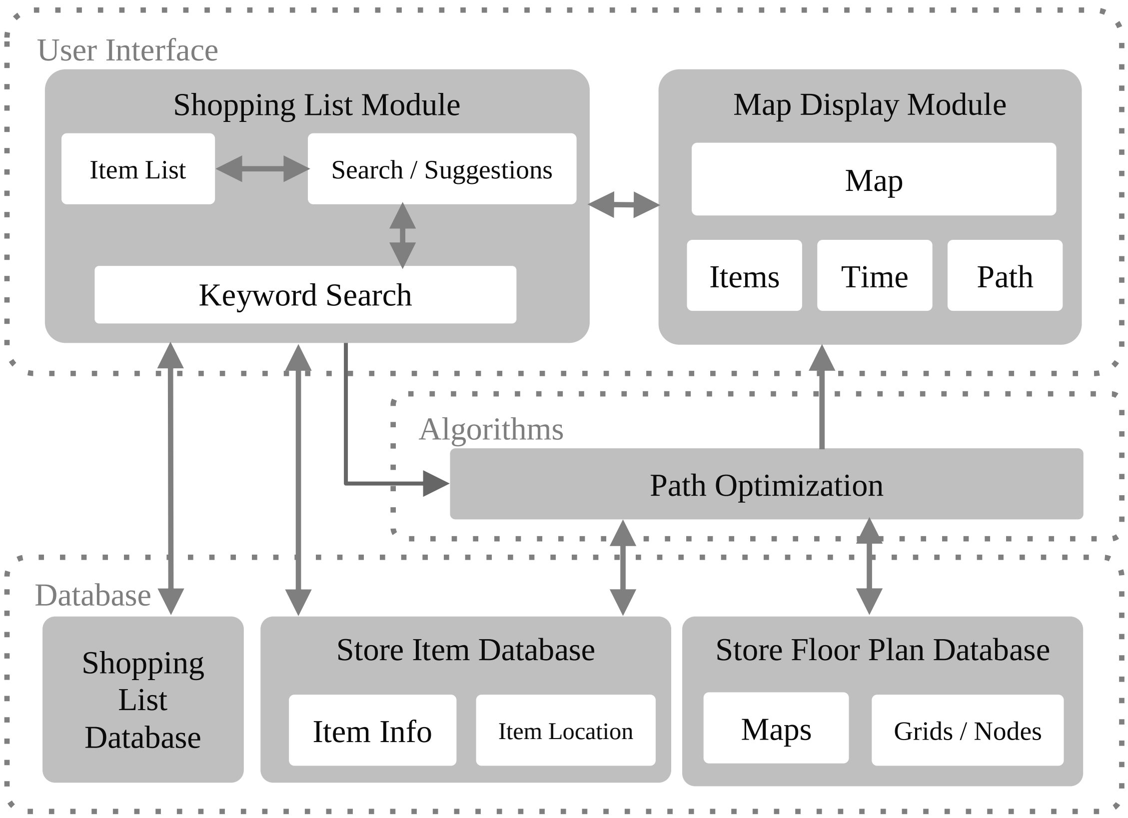

FIGURE 4.1: SYSTEM BLOCK DIAGRAM

5.0 TECHNICAL IMPLEMENTATION:

5.1 USER INTERFACE

The application was designed for an android platform, and implemented in a Java and XML framework in Android Studio. The application user interface (UI) is defined as a relative layout in XML. Each XML file is programmatically linked to a Java activity, where user events, such as buttons clicks and input events, are handled appropriately. The UI consists of three modules; home screen, shopping list module, and map display module as illustrated in FIGURE 5.1.

FIGURE 5.1: USER INTERFACE MODULES; HOME SCREEN, SHOPPING LIST MODULE AND MAP DISPLAY MODULE (LEFT TO RIGHT)

In addition to displaying the product logo, the home screen consists of two buttons; NAVIGATE MAP and CREATE GROCERY LIST. The map display module presents the store map, which the user can navigate to via NAVIGATE MAP button. The user can navigate to the shopping list module in order to create a grocery list via CREATE GROCERY LIST button. The shopping list module is designed to receive user input and display this input in a dynamically growing list on the screen. In the event that a user enters an incorrect grocery item, they can hold the item to display the delete option. A check box has been included adjacent to each item to aid the user. Once the user is satisfied with their grocery list they can click on the MAP button, leading them to the map display module. Here the user is presented with the optimal path to travel through the store in order to obtain items on their grocery list. The map module is constructed using the Android Canvas class and drawables. The map module is divided into two components; the first component constructs a canvas with the store floor plan to draw onto and the second layer uses

Android drawables to trace the optimal path onto the floor plan. In addition to displaying the optimal path, the shopping module computes and displays the estimated shopping time. The UI integrates with data base and path optimization algorithms in two main ways; (1) to obtain the user input and supply it to the database and (2) obtain the optimal path and display it on the UI.

5.2 Data Storage of Store Item Information

Real data from nearby stores were obtained. This information included the store floorplans, as well as the location of a number of items within the store. To keep track of item locations within the store, a database in MySQL was developed that mapped store items from the floorplan to an uploaded map on the store. This MySQL database then took advantage of a wrapper/helper function to update, add, or delete items from the store. The database contains a Java Class, 'StoreItem' that contains the following fields:

Table 5.1: Shopping item database fields

Field Name	Purpose
Id	Unique ID code for each StoreItem instantiation
StoreId	ID Code corresponding to each store
ItemName	Name of the item, eg. "Apple"
ItemX	Location of the corresponding row on the map
ItemY	Location of the corresponding column on the map
PathDirection	Direction of the nearest aisle from the item (North, East, South or West, which corresponds to 0, 1, 2, or 3)
PictureLocation	Path of a picture that can be used to represent the item on a map (optional)
Aisle No.	Which aisle of the store the item is located (optional)

The StoreItem database is used to interface with a number of other parts of the system: it takes a list of item names entered by the user from the user interface to provide a list of locations to the optimization algorithm in order to process the fastest possible path. In addition, an Android feature is used to map the item names from the database into search suggestions for developing the grocery item list.

5.3 Path Optimization Algorithm

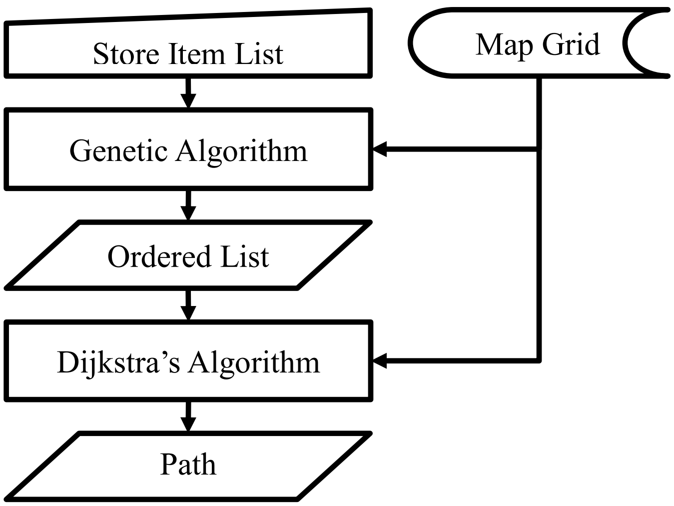

FIGURE 5.2: POA System Flow Chart

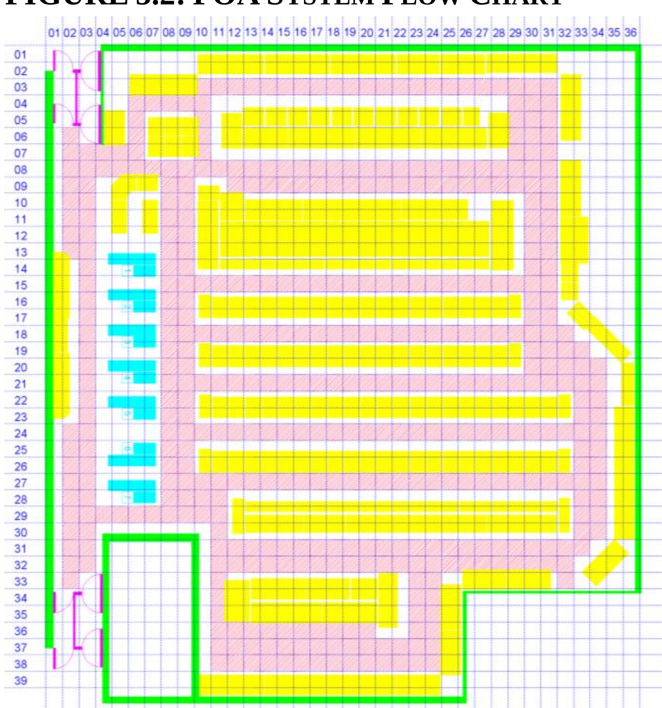

FIGURE 5.3: Map Grid Diagram

The purpose of the path optimization algorithm (POA) is to compute the shortest path the user has to travel in order to obtain all the grocery items in their list. There are two main inputs to the POA. The first input is a list of grocery items received from the grocery list module. It is an array of store item objects as defined by the database. Each store item object contains information specific to that item, and the POA uses the coordinate information to locate each item on the store map. The second input to the POA is the map grid (see Figure 5.3). The map grid divides the entire store map into grid squares, where each square has a unique pair of x and y coordinate. The square size is selected such that each square contains at most one shelving unit (in the case of the sample map, the square size is 3’ x 3’). This is to enable the user to quickly locate the target grocery item. The map grid also contains information on the layout of the store. It indicates the location of obstacles (shelves, walls, etc.) as well as walking space. The map grid is different for each store, and the collection of these map grids are stored in the store floor plan database. Together, the map grid and the store item list are passed into the first layer of the POA for path computation.

The first layer of the POA uses a variant of the genetic algorithm (GA) to compute the purchase order for the grocery items in the input list. The objective of the GA is to solve the travelling salesman problem (TSP), where the starting and ending city is the main entrance and the cities to be visited are the grocery items in the list. The GA starts from a population of randomly generate solutions. It selects the two parents from subsets of the population using a fitness function computing the shortest path. A child in the next generation is created from the two parents using an ordered crossover function, where a subset of the cities in the first parent is added to the child followed by the rest in the second parent’s order. The new population is then mutated using swap mutation, where cities inside a child is swapped to meet the constraint of the TSP of one visit per city. After several generations, the fittest candidate is selected out of the evolved population as the shortest path. Even though the GA uses a heuristics approach to solve the TSP, it was selected over an exact algorithm due to runtime and memory usage limits for a mobile application. The output of the GA is a list of grocery items in the order to be visited at, and it is passed into the next layer of algorithm for detailed path coordinate computations.

The second layer of the POA creates the shortest path from item to item using Dijkstra’s algorithm. The algorithm utilizes a graph or grid search method that divides the map into separate grid points or nodes. An adjacency matrix is built from the map grid based on connections between each node, while ignoring nodes that are considered obstacles (i.e. shelves). Each node in the grid contains a *cost value function*, which is reset to zero at the beginning. An initial node (i.e. start point) is set as the *current* node, while all the other nodes are considered *unvisited*. The *current* node considers and expands to all its adjacent nodes, while calculating and keeping track of the cost value function of each adjacent node. Once each neighbor is considered and is marked as *alive*, the current node is discarded and marked as *dead*. The algorithm then chooses the node with the lowest value function to continue expanding. The process is repeated until the target node has been marked as *alive*. The path can then be found by tracing backwards through the nodes using the cost value function. The algorithm was utilized by taking the ordered list of items outputted from the genetic algorithm and computes the shortest path between each pair of items. Once completed, the ordered set and path containing the points to be traversed are passed to the user interface as a path structure.

FIGURE 5.4: TIMING PERFORMANCE OF POA

To assess the performance of the POA within the requirement boundary of ten store items. An experiment of 33 trials runs has been conducted to verify that the output path results are correct and are computed within a reasonable amount of time. Each trial uses a randomly generated grocery item list. The result of the experiment is shown is Figure 5.4. All trials produced correct path results and the worst runtime for ten items is 1.3 seconds (reasonable).

6.0 RESULTS

The following summarizes the design functions and results. It provides a description of how the requirements were fulfilled.

Requirement	Project Analysis and Outcome
#1: Platform	The app was built and tested on an Android mobile platform
#2 :User Interface	The app provided a touchscreen user interface that allows the user to input items to a shopping list (Refer to Figure 5.1)
#3: Storage	The app stored and displayed a shopping list of at least 10 items based on user input successfully (Refer to Figure 5.1)
#4: Display Floor Plan	The app displayed a store floor plan with locations of the inputted grocery items indicated on it (Refer to Figure 5.1)
#5: Optimal Path	The app provided an optimal path through the store in order to obtain items on the grocery list (Refer to Figure 5.1)
#6 & #7: Time Prediction	The app provided an approximate time required to traverse the optimal path in order to obtain the items (Refer to Figure 5.1). The prediction for a 3 Item path was 1 minute 30 seconds; an experiment was performed to determine the time required to traverse through the path in the actual store. The resulting time was 1 minute 25 seconds. This value is within and under the required time at a 5% error margin.
#8: Database	The app provided a database of categorized items from the store (Refer to Section 5.2)
#9: Search Suggestions	The app provided a search suggestion drop-down list based on user input with at least 1 suggestion (Refer to Figure 5.1)
#10: Privacy Items	The app does not request for the user's private information

7.0 BUDGET:

Item	Cost	Justification
Android Phone –Nexus 5 16GB	$375 [14]	The hardware used for testing the mobile application was the Nexus 5 Android smartphone with a market cost at $375.
mySQL Database Server	Free	The database server containing the list associated store item information was completed using the free version of the MySQL database software. It is to be assessed offline.
Android Studio	Free	The design of the mobile application was completed using the free open source software Android Studio.
Eclipse Programming IDE	Free	The Java code for the optimal path algorithm was developed and tested in the free open source software Eclipse IDE.
Store Floor Plans	Free	Information regarding the store, such as floor plans and store items, were obtained from the retailer without any fees.

8.0 TASK DIVISION:

Tasks were assigned such that at least two people worked on a major system component. This improved effectiveness during integration and troubleshooting phase of the project. Kammy and Lingfei were responsible for designing and implementing the algorithm for optimal path planning. This component creates a grid system of the store to locate items and generates the shortest path to retrieve all the items on the shopping list. Alice and Meghna were responsible for building the user application interface and supporting databases. The two main user interfaces include the shopping list and the map display. The shopping list accesses the database, which contains the keyword search algorithm and the list of information for the store items. All members were responsible for module integrations and system troubleshooting.

9.0 CONCLUSION:

8.1: FUTURE WORK

The app contains the fundamental features to allow a user to 1) input a list of shopping items, and 2) follow a path displayed on a map to retrieve the inputted items. However, various features can be added to improve the app through areas of: 1) store app feature integration, 2) indoor positioning and 3) process and management improvement.

Features that can be integrated from store apps include: price matching, suggested shopping locations, flyers, advertisements, discounts, coupons, and a store points system. An indoor positioning feature can also guide users to keep track of where they are within the store. These features enhance a shopper's customer experience by providing useful services.

The app can also aid in improving a store's productivity and app maintenance by allowing for automatic updating of store layouts allow for stores and app makers to better integrate and update store information. Additional attributes to allow for easier management of merchandise amounts and locations include incorporating a computer vision system to visually keep track of item locations and store layouts.

In addition, the app can be further expanded into various sectors to increase efficiency in indoor navigation. Primarily, the idea can be applied to shopping malls, schools, libraries, airports, and hospitals. The *grocery items*, as referred to in the app, can be replaced with *stores, classrooms, books, terminals or gates, and hospital rooms* respectively.

8.2: SUMMARY & SIGNIFICANCE

Due to the busyness of today's society, methods of increasing productivity in mundane tasks, such as grocery shopping, are well sought after. An app is created to allow users to navigate through big box retail stores quickly to optimally retrieve their grocery items. This idea can be further expanded into various sectors to improve society's productivity.

REFERENCES:

[1] " Ways the Walmart iPhone app saves you time and money." Internet: http://www.walmart.com/cp/Walmart-Mobile-App/1087865, [Jan. 22, 2015].

[2] "Mobile App: My Metro." internet: http://www.metro.ca/mobile-application.en.html#, [Jan. 21, 2015].

[3] A. Welch. "The Age of Grocery "Smart Carts" Creeps Closer." Internet: http://www.retailsolutionsonline.com/doc/the-age-of-grocery-smart-carts-creeps-closer-0001, Mar. 5, 2014 [Jan. 26, 2015].

[4] M. Woodruff. "Watch a Demo of Whole Food's Eerily Smart Motorized Grocery Cart." Internet: http://www.businessinsider.com/watch-a-demo-of-whole-foods-eerily-smart-motorized-grocery-cart-2012-3, Mar. 2, 2012 [Jan. 23, 2015].

[5] M. Lindstrom. "Shopping Carts Will Track Consumers' Every Move." Internet: https://hbr.org/2011/12/shopping-carts-will-track-cons, Dec. 9, 2011 [Jan. 23, 2015].

[6] V. Woollaston. "Never get lost in frozen food again: Philips trials GPS app to help shoppers navigate supermarkerts." Internet: http://www.dailymail.co.uk/sciencetech/article-2561226/Now-GPS-SUPERMARKET-Philips-trials-app-help-shoppers-navigate-stores-maze-aisles.html, Feb. 17, 2014 [Jan. 22, 2015].

[7] "Aisle 411." Internet: http://aisle411.com/, [Jan. 20, 2014].

[8] J. Crowl. “Google Hooks Up with Aisle411 to Map Stores” Internet: http://streetfightmag.com/2014/06/30/aisle411-google-partner-to-launch-3d-mapped-shopping-solution/, Jun. 30, 2014 [Jan. 23, 2015].

[9] “Indoor Retail Mapping Leader aisle411 Delivers In-Store 3D Mapping on Google’s Project Tango.” Internet: http://www.businesswire.com/news/home/20140625005097/en/Indoor-Retail-Mapping-Leader-aisle411-Delivers-In-Store#.VMqKLmjF9-J, Jun. 25, 2014 [Jan. 23, 2014].

[10] “Company Overview of Aisle411 Inc.” Internet: http://www.bloomberg.com/research/stocks/private/snapshot.asp?privcapid=60042856, [Jan. 29, 2015].

[11] “Aisle411 – shopping companion.” Internet: https://play.google.com/store/apps/details?id=com.aisle411&hl=en, [Jan. 20, 2014].

[12] “Android Apps Break the 50MB Barrier.” Internet: http://android-developers.blogspot.ca/2012/03/android-apps-break-50mb-barrier.html, Mar. 05, 2012 [Jan. 20, 2015].

[13] “Android Studio.” Internet: http://developer.android.com/sdk/index.html, [Jan. 19, 2015].

[14] ” Rogers.” Internet: http://www.rogers.com/web/link/wirelessBuyFlow?gclid=CJuusYror8QCFYQCaQodDkgANg&forwardTo=PhoneThenPlan&productType=normal&productId_Detailed=NEXUS5BLK&N=&s_kwcid=AL!4194!3!47704220018!b!!g!!%2Bnexus%20%2B5&ef_id=VQhUXwAABJ1ykiu7:20150317163156:s#hiddenContent, [March 17, 2015].