International Journal of Computing Sciences Research (ISSN print: 2546-0552; ISSN online: 2546-115X)
Vol. 6, pp. 1032-1045
doi: 10.25147/ijcsr.2017.001.1.86
https://stepacademic.net

Short Paper*

Development of an Android Grocery Checklist Application

Ana Antoniette C. Illahi
Department of Electronics and Computer Engineering, De La Salle University, Philippines
ana.illahi@dlsu.edu.ph
(corresponding author)

Gershom Rob A. Narag
Department of Electronics and Computer Engineering, De La Salle University, Philippines
gershom_narag@dlsu.edu.ph

Manuel Lorenzo L. Parro
Department of Electronics and Computer Engineering, De La Salle University, Philippines
manuel_parro@dlsu.edu.ph

Luis Paolo D. Wenceslao
Department of Electronics and Computer Engineering, De La Salle University, Philippines
luis_wenceslao@dlsu.edu.ph

Date received: November 15, 2021
Date received in revised form: January 20, 2022
Date accepted: January 20, 2022

Recommended citation:

> Illahi, A. A. C., Narag, G. R. A., Parro, M. L. L., & Wenceslao, L. P. D. (2022). Development of an Android grocery checklist application. *International Journal of Computing Sciences Research, 6*, 1032-1045. doi: 10.25147/ijcsr.2017.001.1.86.

**Special Issue on National Research Conference in Computer Engineering and Technology 2021. Guest Editor: Dr. Luisito Lolong Lacatan, PCpE (Laguna University, Philippines). Associate Editors: Dr. Nelson C. Rodelas, PCpE (University of the East-Caloocan City, Philippines), and Engr. Ana Antoniette C. Illahi, PCpE (De La Salle University, Manila, Philippines).*

Abstract

Purpose – Grocery shopping is a regular affair in most people's lives. With a variety of necessities such as food, toiletries, and the like being purchased whenever people go grocery shopping, shopping lists are usually made to help buyers remember and organize

what they need to purchase. This research introduces a mobile application that allows users to create and manage shopping lists for grocery shopping, as well as allowing them to mark what they've purchased by scanning the barcode of the purchased product. It also allows users to track their stocks of purchased items as well as view their expenditures and what kinds of products take up most of their expenses. 4

Method – The barcode data used is a selection of 100 unique products taken by the researchers, and the application uses SQLite to store both checklist data and the barcodes.

Result – Based on the results, all the test cases were able to pass the expected result which means that the main features of the application were able to run smoothly. There are times that the barcode scanner was not able to read the barcode properly. This scenario is seen especially when the camera is at the right angle when scanning the product's barcode. The front angle the scanner's success rate was 99%, but from the right, left, top, and bottom, the success rate was 84%, 93%, 97%, and 94%, respectively. Although the success rate is still high, it is still preferable to scan barcodes from the front. Overall, it has an average success rate of 93.8% on all the angles. The stocks screen was able to change the number of stocks it recorded when adding and removing items from each category of item. The system was also able to display if an item was running out of stock and if it already is out of stock. It was found that the stock system's functions were working properly. The maps function was tested via changing the current area and testing if it can pin locations based on the set condition. All in all, 5 locations were tested, with each location successfully having pins on multiple locations. The charts were able to display different expenditures for different weeks and months.

Conclusion – The research has resulted in the creation of an app that can help users in managing their grocery shopping. It can create and edit checklists, access, and manage checklists from the local storage, as well as add and remove items from the created checklists. The system can scan barcodes of items that are in the internal database and automatically record them in both item stocks and expenditure statistics. These can help the user in managing their shopping, as well as showing them what they need to buy and how much they are spending on certain kinds of items. The app was built at a minimum Android SDK of 16. This left the researchers with difficulties in implementing some functions due to much simpler ways of implementation added in later SDKs being unavailable, but this also meant that the app was compatible with a lot more devices.

Recommendation – Organize the list of products into categories to make searching for items faster. Add an option for statistics to break down expenditures even further to individual items instead of just categories. It is also recommended to integrate the product database with different grocery stores to be more accurate in terms of getting the total price of a checklist.

Practical Implication – Developing of an Android grocery checklist application for the beneficiaries is useful because it can be use and can able to track the grocery or stock the user has. Also, the application as the capability to pin the location of the available grocery store with in the area.

Keywords – barcode scanning, grocery list, mobile application, SQLite

INTRODUCTION

In this modern-day age, different stores like grocery stores and convenience stores have emerged almost everywhere inside and outside of Metro Manila. The industry of supermarkets and groceries in the Philippines is expected to be the 5th largest in the Asia Pacific region (BusinessWorld, 2017). With the population of Metro Manila alone standing at 12.88 million (Bersales, 2016), the goods required for production are expected to rise to match the demand. Supermarket retailers like Puregold Price Club and SM Retail are expected to further expand to accommodate the rising demand. 7-Eleven alone opened a total of 2,031 stores nationwide, of which 1,649 are in Luzon, 261 in the Visayas, and 121 in Mindanao. Of the 1,649 located in Luzon, only 811 are in Metro Manila (BusinessWorld, 2017). According to Ms. Zhu, Filipinos are driven by more disposable income and increasingly urbanized lifestyles that they demand more convenience in grocery shopping (BusinessWorld, 2017).

The use of devices like computers or smartphones has become part of daily human lives. Nowadays, people use smartphones for communication, business, leisure, and all sorts of activities much like how they would use their personal computers. In a supermarket setting, previously people used a notepad and a pen as the conventional way to monitor their shopping list as well as their total spending. With the increasing ubiquity of smartphones, people have switched to using their mobile devices into tracking their spending in grocery stores or convenience stores.

In this research, an application is to be developed that would utilize notable features of a smartphone such as its camera for scanning barcodes. The application tracks the shopping list of the shopper and gives out the prices of the products marked by the list. The data gathered by the application are stored in a database. When the total price of the products is calculated, the app takes note of it to track the expenditures of the user on a timely basis. Another feature that the application will have is the capability of detecting if the user's stocked goods are running out based on the deductions the user inputs to the app every time, he or she uses an item in storage. The app notifies the user of the matter when the stocks run low based also on the database. The application also has a feature which can give out information on where the location of the nearest stores, or supermarkets.

METHODOLOGY

The software development model implemented for the application's development is the incremental development model (Taktak, Long, Ganney, & Axell, 2020). The incremental model starts with just a few key features for the product, but as features are completed more modifications and functions are added and the system evolves until the final product is reached.

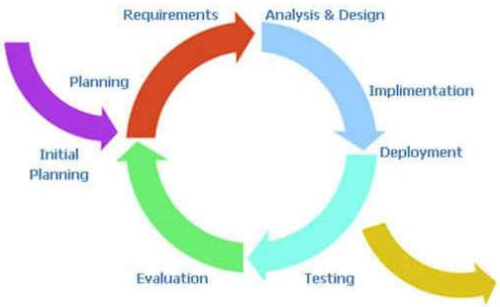

Figure 1. Incremental Modelling

Figure 1 shows the flow of the development of the application Taktak et al. (2020). The flow starts with the planning phase. In this phase, the user identifies the different requirements required for app development. To achieve these objectives, planning on how to implement the requirements and what tools to use are to be set. The next phase is the analysis and implementation of the design. With the necessary methods and tools on app development in place, the next step is to program the application to meet the objectives. The third is the testing phase where the app is tested multiple times to see how well the app performs and meets the objective. Lastly is the evaluation of the application. The app is evaluated for errors and its efficiency. The app is then deployed if everything is working excellently. However, if the app is still full of bugs or does not function properly to meet the objectives, it is sent back to the planning stage for further development.

Shown in Figure 2 below is the data flow diagram of the grocery application. Four inputs are coming from the user which are the list name, product name, product barcode, and user location. Coming from those 4 inputs, the application can provide 6 outputs to the user which are the checklists, grocery locations, user location, updated cart price, stocks, and expenditures in a daily, weekly, monthly format.

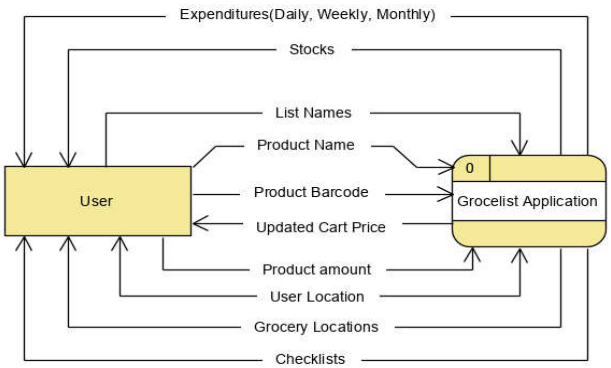

Figure 2. Dataflow Diagram of the Application

Figure 3 below is the Entity-Relationship Diagram (ERD) of the application. ERD is a graphical representation of how databases relate to another database. In this research, crow's foot notation was used. Based on the notation, it is seen that an instance of Product Database is linked to zero or many checklist databases which means that only one Product Database can be used for zero or multiple checklists. For the Statistics and Stock Database, they are linked to one or many Checklist Databases which means that there are only one statistics and stock database for multiple checklists. Table 1 below shows the libraries used in developing the application.

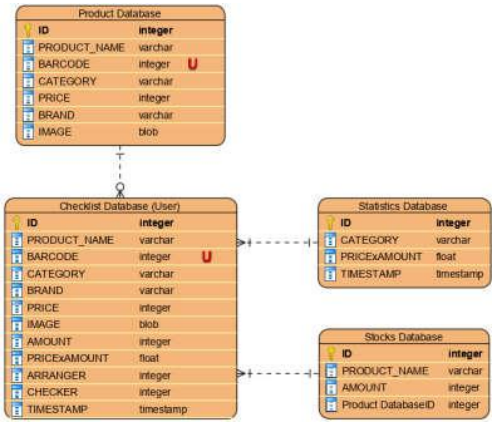

Figure 3. ERD of the Application

Table 1. Libraries Used in the Application

Libraries
com.android.support:recyclerview-v7:28.0.0
com.android.support:cardview-v7:28.0.0
com.github.lecho:hellocharts-library:1.5.8@aar
com.readystatesoftware.sqliteasset:sqliteassethelper:2.0.1
com.journeyapps:zxing-android-embedded:3.4.0
com.google.android.gms:play-services-maps:17.0.0
com.google.android.gms:play-services-places:17.0.0
com.google.android.libraries.places:places:2.2.0
com.google.android.gms:play-services-location:17.0.0

Application Setup

The researchers installed Android Studio and all the other necessary software add-ons required in doing the app development (Karch, 2021). The version of Android Studio at the time of installation is version 3.5 (Chen, 2021). For the SDKs, SDK Platform 16, 28, and 29 which correlates to Android 4.1, 9.0, and 10.0 respectively, were installed (Mullis A., 2017). SDK 29 was installed because of the move made by the developers of Android Studio encouraging users to update most of the libraries, past SDK 28. For the DB Browser, the main purpose of the installation is to easily access and view the SQLite database generated by the Android app (Coronel & Morris, 2017). To set up the libraries to be used in developing the application. The researchers added the libraries to be used inside dependencies in build.gradle under Gradle Scripts. When the library will be used in a class, it should be imported first in that class.

Data Gathering Process

The researchers had gathered 100 unique products from SM Savemore Supermarket and divided them into 10 unique categories. These categories are canned goods, cereal, cleaning agents, coffee, condiments, dairy, instant noodles, snacks, spreads, and toiletries. After creating the database, the researchers manually inputted the product information to their corresponding section as shown in Figure 4 below. The researchers have used SQLite DB Browser software for creating and editing the product database Chan (2018).

	ID	PRODUCT_NAME	BARCODE	CATEGORY	BRAND	PRICE	IMAGE
		Filter	Filter	Filter	Filter	Filter	Filter
1	1	Nescafe Classic Refill 200g	4800361393669	Coffee	Nescafe	147.5	BLOB
2	2	Nescafe Classic Strong 80g	8850125084742	Coffee	Nescafe	75.95	BLOB
3	3	Angel Coffee Creamer 200g	748485400846	Coffee	Angel	33.5	BLOB
4	4	Cream All Coffee Creamer 300g	4800016011788	Coffee	Cream All	52.5	BLOB
5	5	Nescafe Classic Decaf 80g	4800361393744	Coffee	Nescafe	88.5	BLOB
6	6	Nestle Koko Crunch 500g	4800361090239	Cereal	Nestle	189.5	BLOB
7	7	Nestle Milo Balls Cereal 300g	4800361355872	Cereal	Nestle	212.5	BLOB
8	8	Nestle Honey Stars 300g	4800361028165	Cereal	Nestle	154.5	BLOB
9	9	Nestle Corn Flakes 500g	4800361310031	Cereal	Nestle	169.5	BLOB
10	10	Nestle Fitnesse Original 375g	4800361355704	Cereal	Nestle	174.5	BLOB
11	11	Century Tuna Hot and Spicy 180g	748485100098	Canned Goods	Century	37.75	BLOB
12	12	Gold Seas Yellowfin Tuna Chunks in Olive Oil 180g	4806530070034	Canned Goods	Gold Seas	62.5	BLOB
13	13	Purefoods Chunkee Corned Beef 350g	4808887020106	Canned Goods	Purefoods Hormel	92.25	BLOB
14	14	SPAM 30% LESS SODIUM 340g	37600240345	Canned Goods	Hormel Foods	179.5	BLOB
15	15	SPAM Original 340g	37600130783	Canned Goods	Hormel Foods	179.5	BLOB
16	16	Pancit Canton Kalamansi Flavor	4807770273674	Noodles	Lucky Me	11.8	BLOB
17	17	Pancit Canton Original Flavor	4807770273704	Noodles	Lucky Me	11.8	BLOB
18	18	Quickchow Instant Mami Hot & Spicy Beef Flavor 55g	4804888889056	Noodles	Quickchow	6.75	BLOB
19	19	Nissin Ramen Instant Noodles Beef Flavor	4800016551581	Noodles	Nissin	8.5	BLOB

Figure 4. Product Database Window on SQLite DB Browser

Development Process

After setting up the product database, the researchers uploaded the product database DB file to the assets folder. This will enable the application to access the product database locally. After uploading, the researchers created java classes that utilize the DB file uploaded. This will also be used in conjunction with checklist creation, stock

monitoring, and expenditure monitoring. To help with the initial creation of the product database, the researchers used the "SQLiteAssetHelper" dependency which automatically manages the initial creation of the product database (SQLite, 2021).

For the creation of the checklist screen, RecyclerView library was used in listing the products selected from the product database. It is also used in the stock screen and product database screen. For the barcode development, the researchers integrated an open-source library called ZXing. This library automatically decodes 1-dimensional and 2-dimensional barcodes. Combined with the RecyclerView, the researchers added a function in the adapter class of the RecyclerView which crosses out the product in the checklist screen. The researchers use setPaintFlags() function to cross-out the text.

For the development of the maps, the researchers first obtained an API key from Google which would enable the use of Google APIs. Google Maps and Google Places were activated for use since Google Maps will be displaying the maps while Google Places will gather information regarding locations from Google servers. For the location of nearby supermarkets, Google Places API uses a text search in URL format that has the following required parameters: query, and API key. In addition to that, location, and radius were also used. The text search then returns 20 results per execution. In getting all the results, an Asynctask was used, inputting the results into a HashMap list and performing a for loop to access the list and pin the locations.

Lastly, the app's statistics feature uses a table from the database separate from the tables used by the checklists, so that if checklists get deleted or otherwise modified the already-bought items will not be removed from the sum of expenses. This section will display the GUI for all the major classes of the application. The figure will start with the main menu. Next will be the main interfaces: checklist, stocks, nearby supermarkets, and statistics.

Figure 5. Main Menu of the Application

Figure 5 shows the main menu after launching the application. The main menu displays the application's logo, and buttons that open the app's checklist, stocks, nearby supermarkets, and statistics features.

Figure 6. Product Database Screen

Figure 6 displays the product database and its contents. The product database is linked to the application's local database. New products can be added, edited, and removed from this database. The stock screen shows the current stock of the user based on the products scanned. The user can update each product by pressing the number beside the product's name. It can also alert the user if they are running out of a product. The nearby supermarkets' screen will show the device's current location. The "Find Supermarkets" button will mark supermarkets within a 1km radius of the device. The statistics screen shows the expenses of the user based on the checklists made in a daily, weekly, or monthly period. It will also show what category of items make up the percentage of expenses.

TESTING AND EVALUATION

To test the different functions of the application, the researchers created test cases for each function and on how to properly execute the task. For the test cases, the researchers used created a template in an excel file where every test case is listed by row. The test cases to be executed by the researchers are enumerated in Table 2 below.

The researchers scanned each of the products a total of 10 times on every angle as shown in Table 3. The researchers created 5 different angles in testing the barcode. Front angle, left angle, right angle, bottom angle, and top angle. Each angle is approximately 45 degrees and 20cm from the camera to the product's barcode. Figure 7 below shows the front angle setup on testing the barcode.

Table 2. Test Cases to Be Executed

Test Case ID #	Description
TC001	Create multiple checklists
TC002	View created checklists
TC003	Add item to the checklist
TC004	Delete an item from a checklist
TC005	Scan barcode from item in a checklist
TC006	Rename checklist
TC007	Check stocks database in the app
TC008	Manage the number of stocks in the app
TC009	Display map and nearby supermarket locations
TC010	Display statistics charts based on user expenditures
TC011	Add a new item to the product database
TC012	Edit a product from the product database
TC013	TC013 Delete a product from the product database
TC014	Update image of a product

Table 5 below shows the average success rate of scanning the barcode on all the angles. Each row represents 50 iterations of barcode scanning per product. As observed on the data gathered, there are times that the barcode scanner was not able to read the barcode properly. This scenario is seen especially when the camera is at the right angle when scanning the product's barcode.

Table 3. Products used for testing

Product #	Product Name
1	Nescafe Classic Refill 200g
2	Nestle Koko Krunch 330g
3	Nescafe Classic Decaf 80g
4	Century Tuna Hot and Spicy 180g
5	Pancit Canton Kalamansi Flavor
6	Safeguard Pure White Jumbo 175g
7	Jack &Jill Piattos Cheese 85g
8	Nestle Fresh Milk 1L
9	Nissin Yakisoba Spicy Chicken Flavor 59g
10	Cheez Whiz Spread Plain 220g

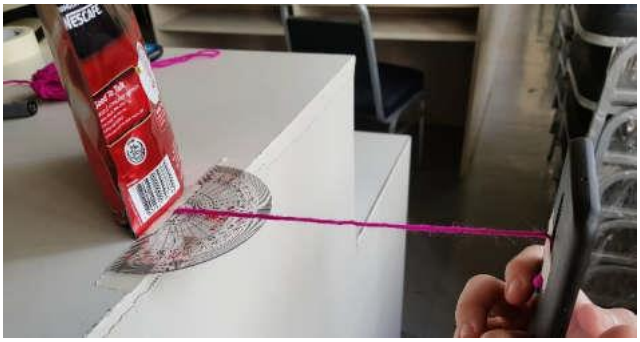

Figure 7. Front Angle Barcode Testing

To compute for the accuracy of the barcode scanning, the researchers created a template in an excel file and record each successful and failed scan for each product for all the angles. Each of the products for every angle is scanned a total of ten times. Successful scans are marked as 1 on the template while failed scans are marked as 0. For it to be called a successful scan, an alert dialog must pop up showing the details of the product. Overall, the researchers were able to gather 500 scans of the products.

For the testing of the Google Maps feature in the project, the researchers utilized the location feature built in the Android Studio emulator. The map was tested in multiple hotspots around Metro Manila. These hotspots were De La Salle University, Cubao in Quezon City, 10th Avenue Grace Park in Caloocan City, Makati Central Business District (CBD), and Bonifacio Global City (BGC). The query used in forming the URL for Google Places text search indicates the following: supermarket, market, hypermarket, puregold, rustan, savemore, and shopwise. These are the chosen keywords for the query to retrieve results that are relevant to supermarkets. The Google Places text search returns 20 results all relevant to the search query.

$$d = 2rsin^{-1}\left(\sqrt{sin^2\left(\frac{\Phi_2-\Phi_1}{2}\right) + cos(\Phi_1)cos(\Phi_2)sin^2\left(\frac{\lambda_2-\lambda_1}{2}\right)}\right) \qquad \textit{Equation 1}$$

To test whether the program is displaying places within the specified 1000m radius, the Haversine formula shown in Equation 1 is used to calculate the distance from one coordinate to another coordinate. Phi in the formula are the latitudes and Lambda are the longitudes. R is the Earth's radius which is 6,371 km. Plugging in the results of the value to the distance between the coordinates in kilometers. Results are then used as a basis if the markers pinned on the map are within 1000m. For verification, Excel will be used in getting the distances of the 20 results returned by Google. For user testing, the researchers surveyed 31 people to use the app. The test environment is held in a vacant room wherein the products are displayed on a table. To have a systematic way of testing, the researchers created a test procedure for the users to follow.

RESULTS AND DISCUSSION

Table 4 below shows the results of the test cases. Based on the results, all the test cases were able to pass the expected result which means that the main features of the application were able to run smoothly.

Table 4. Results of Test Cases

Test Case ID #	Description	Result
TC001	Create multiple checklists	Passed
TC002	View created checklists	Passed
TC003	Add item to the checklist	Passed
TC004	Delete an item from a checklist	Passed
TC005	Scan barcode from item in a checklist	Passed
TC006	Rename checklist	Passed
TC007	Check stocks database in the app	Passed
TC008	Manage the number of stocks in the app	Passed
TC009	Display map and nearby supermarket locations	Passed
TC010	Display statistics charts based on user expenditures	Passed
TC011	Add a new item to the product database	Passed
TC012	Edit a product from the product database	Passed
TC013	TC013 Delete a product from the product database	Passed
TC014	Update image of a product	Passed

Table 5 below shows the average success rate of scanning the barcode on all the angles. Each row represents 50 iterations of barcode scanning per product. As observed on the data gathered, there are times that the barcode scanner was not able to read the barcode properly. This scenario is seen especially when the camera is at the right angle when scanning the product's barcode.

Table 5. Average Success Rate of Barcode Scanning

Product #	Success Rate
1	88%
2	100%
3	82%
4	96%
5	92%
6	98%
7	98%
8	100%
9	88%
10	96%
Average	93.8%

The researchers found that from the front angle the scanner's success rate was 99%, but from the right, left, top, and bottom, the success rate was 84%, 93%, 97%, and 94%, respectively. Although the success rate is still high, it is still preferable to scan barcodes from the front. Overall, the researchers were able to have an average success rate of 93.8% on all the angles.

The stocks screen was tested by adding and removing items from the checklist. Upon testing, the stocks screen was able to change the number of stocks it recorded when adding and removing items from each category of item. The system was also able to display if an item was running out of stock and if it already is out of stock. It was found that the stock system's functions were working properly.

The maps function was tested via changing the current area and testing if it can pin locations based on the set condition. All in all, 5 locations were tested, with each location successfully having pins on multiple locations.

Figure 8. Nearby Groceries near De La Salle University

For the data gathered about supermarkets nearby De La Salle University shown in Figure 8, the URL formed by the application was:

https://maps.googleapis.com/maps/api/place/textsearch/json?query=supermarket%7Cmarket%7Chypermarket%7Cpuregold%7Crustan%7Csavemore%7Cshopwise&location=14.56474,120.9931787&radius=1000&key=AIzaSyAPJetGZnzkERkYoHuTmnCzx8E4uiWEHmM

This URL contains 20 results relevant to the search query and is based on prominence. The results returned by Google are shown in Figure 9 already encoded in Excel for verification.

NAME	LATITUDE	LATITUDE ϕ_2 (Radians)	LONGITUDE	LONGITUDE λ_2 (Radians)	DISTANCE FROM USER (m)	RANGE
Puregold Extra	14.572371000	0.254335854	120.999269000	2.111835637	1073.137	Not in Range
Shopwise Manila	14.562742500	0.254167805	120.989593200	2.111666762	444.711	Within Range
Puregold San Andres	14.570155700	0.254297189	120.988646000	2.111650230	776.084	Within Range
Savemore Market San Andres, Manila	14.569730300	0.254289765	120.988996700	2.111656351	715.572	Within Range
SM Hypermarket	14.563804500	0.254186340	120.990028400	2.111674358	354.395	Within Range
Savemore	14.567196300	0.254245538	120.992319600	2.111714696	288.949	Within Range
Puregold Makati	14.574159600	0.254367071	121.012309400	2.112063234	2310.414	Not in Range
Puregold Paco	14.576808400	0.254413301	120.993414600	2.111733458	1343.483	Not in Range
Puregold Jr.	14.547052000	0.253893954	120.998187700	2.111816764	2038.069	Not in Range
Puregold Jr.	14.585697300	0.254568442	121.001537100	2.111875222	2499.085	Not in Range
Super Shopping Market Incorporated	14.563045000	0.254173084	120.990132000	2.111676166	377.676	Within Range
SM Hypermarket Adriatico	14.569330000	0.254282778	120.984160000	2.111571935	1097.338	Not in Range
Puregold Makati	14.574507800	0.254373148	121.012026200	2.112058292	2301.337	Not in Range
Prime Mini Grocery	14.571235000	0.254316027	120.984181000	2.111572301	1208.887	Not in Range
Shopwise Makati	14.566933700	0.254240955	121.013533000	2.112084590	2204.096	Not in Range
Fortune Mart	14.562902400	0.254170596	120.996101100	2.111780346	374.237	Within Range
Puregold Jr.	14.579916000	0.254467539	120.981231000	2.111520814	2122.648	Not in Range
Leverza Super Market	14.569617000	0.254287787	120.988915300	2.111654930	711.443	Within Range
Puregold Libertad	14.547199400	0.253896526	120.999082500	2.111832382	2050.035	Not in Range
Savemore Market Green Mall	14.567417000	0.254249390	120.992301000	2.111714022	313.579	Within Range

Figure 9. Results from Google Places Calculated in Excel Using the Haversine Formula

An if-else statement was then used to limit the markers from being pinned to the map which is set to within 1000m or 1km radius from the current location. The statistics feature was tested by changing the system clock to different dates to simulate expenses over time. The statistics are based on expenditures at different times. The charts were able to display different expenditures for different weeks and months. For the survey results, the researchers found out that most of the respondents were mostly students, with most respondents going grocery shopping at least once a month. Regarding the user experience with the app, the users were generally positive regarding the overall built of the app.

CONCLUSION AND RECCOMENDATIONS

The research has resulted in the creation of an app that can help users in managing their grocery shopping. It can create and edit checklists, access, and manage checklists from the local storage, as well as add and remove items from the created checklists. The system can scan barcodes of items that are in the internal database and automatically record them in both item stocks and expenditure statistics. These can help the user in managing their shopping, as well as showing them what they need to buy and how much they are spending on certain kinds of items. The app was built at a minimum Android SDK of 16. This left the researchers with difficulties in implementing some functions due to much simpler ways of implementation added in later SDKs being unavailable, but this also meant that the app was compatible with a lot more devices.

For recommendations, organize the list of products into categories to make searching for items faster. Add an option for statistics to break down expenditures even further to individual items instead of just categories. It is also recommended to integrate the product database with different grocery stores to be more accurate in terms of getting the total price of a checklist.

REFERENCES

BusinessWorld (2017). *Supermarket, grocery industry could be region's 5th largest, 2021 BusinessWorld Publishing*. Retrieved from https://www.bworldonline.com/supermarket-grocery-industry-regions-5th-largest/.

Bersales, L. G. (2016). *Highlights of the Philippine Population 2015 Census of Population. Philippine Statistics Authority. Reference Number: 2016-058*. Retrieved from https://psa.gov.ph/content/highlights-philippine-population-2015-census-population

Chan J. (2018). *Learn SQL (using MySQL) in One Day and Learn It Well: SQL for Beginners with Hands-on Project*. LCF Publishing.

Chen J. (2021). *Android Operating System. Business Essentials*. Retrieved from https://www.investopedia.com/terms/a/android-operating-system.asp

Coronel C., & Morris S. (2017). *Database Systems: Design, Implementation, & Management (13th Edition)*. Cengage.

Karch M. (2021). *What Is Android? Lifewire Tech for Humans*. Retrieved from https://www.lifewire.com/what-is-google-android-1616887.

Mullis A. (2017). *Android SDK tutorial for beginners. Android Authority*. Retrieved from https://www.androidauthority.com/android-sdk-tutorial-beginners-634376/.

MySQL (2021). *What is MySQL?*. Retrieved from https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html

SQLite (2021). *About SQLite*. Retrieved from https://www.sqlite.org/about.html

Taktak A., Long D., Ganney P., & Axell R. (2020). *Clinical engineering*. Academic Press. doi: https://doi.org/10.1016/C2017-0-01525-2.