VOL. 13, NO. 12, JUNE 2018 ISSN 1819-6608
ARPN Journal of Engineering and Applied Sciences

www.arpnjournals.com

ANDROID APPLICATION IN FOOD ORDERING SYSTEM

Ann Janeth Garcia[1], Danielle Jaye Agron[1] and Wansu Lim[2]
[1]School of Electronics Engineering Mapúa Institute of Technology Manila, Philippines
[2]Department of IT convergence Kumoh National Institute of Technology, South Korea
E-Mail: ajg.garcia@yahoo.com.ph

ABSTRACT

Due to the fast phase lifestyle nowadays, it is truly a setback to spend a lot of time in queuing to the traffic encountered in ordering food on school cafeteria. The researchers develop "cAPPeteria", a food ordering application where the users can directly order through, customize the food quantity, confirm order and generate a one-dimensional barcode on device A, controlled by the user, then the simulated code is scanned through device B, which will scan and decode the information encrypted to the image from device A. Improving the existing wireless food ordering system limits the possible complication faced on connecting to an internet based server thus increasing mobility and control to the user.

Keywords: android application, one-dimensional barcode.

1. INTRODUCTION

The amount of time consumed in ordering and waiting for food in school cafeteria; not to mention the minutes wasted just by standing in a queue for a chance to take an order on is truly a setback for students and professors who greatly value their time.

The traditional way of ordering meal [1] [2] consumes a lot of time and effort and to top all of that, during peak hours, the density of people is distinctly large that causes congestion to build up on the counter due to the disproportionate size of school cafeteria to the customers [3]. The researchers look forward to tear down the time consumed through integration of the mobile technology in automating the task of conventional food ordering system [4] where a wireless interface using an Android application, since the number of Android user is superior [5] compared to the other existing mobile operating system. The cAPPeteria is a food ordering system where it is divided into two complementary android applications.

It was developed to cut the congestion experienced in school cafeteria during peak hours. Another is to solve the challenges of meeting food service in a short time offering flexibility for the customer, and minimizing the time and energy they use. Also, to provide a user-friendly application to the users that escalates mobility antiquated to today's standards.

To achieve these objectives, the device A is designed to be interactive with the user because this displays the set of menu identically patterned to what the cafeteria is offering, hence the application considered organizing the menu by sub-categorizing the list of dishes that user can effortlessly select their choice of meal afterwards, the customer can adjust the quantity to purchase. Another feature of this application is to generate a barcode which on the packaging process [6] encrypted on it is the summary of the confirmed menu and the barcode generated that is presented on the counter embodied by device B. Device B utilize the phone's camera to scan the barcode for the application to decrypt the information by processing the image using the application developed the researchers. The scanned barcode would contain the summary of order made on device A, the breakdown of prices calculated reliant to the quantity confirmed displaying the total price of the placed order.

In planning for the system architecture, the researchers took consideration upon researches that are accomplished to improve the conventional ordering system [7] like of PDA-based wireless food ordering system for hospitality industry [8] that uses application software written in PHP and JavaScript that modulate the communication between PDA-user to the server located on centralized relational database (CDR) by means of wireless LAN, after the confirmation of order a simultaneous command to print to kitchen and print to cashier was executed and finally clearing the data. Similarly, the architecture of Self-Service Restaurant Ordering System (SROS) [9] includes a Graphical User Interface (GUI) develop under Microsoft Office applications stores the database that host information to share by both admin and kitchen. The data is accumulated by receiving customers order on LCD touch screen devices present on each table, increasing the user interface. The downside of which is it requires the customer to be on the facility of the restaurant to access the menu and to make an order. Relatively to that, in the case of Customizable Wireless Food Ordering System with Real Time Customer Feedback (CWOS-RTCF) [10] a mobile application accesses the menu by connecting into a WLAN to enter a web-based application, where a login information is required to allow an access on the server outside the vicinity of the restaurant, making order and food preparation prior to the customer's arrival. Lastly, Wireless Food Ordering System Based on Web Services [11] is much comparable to the CWOS-RTCF architecture; nonetheless, the application interface expands to the availability of the application on desktop computer.

The cAPPeteria can be classified as a wireless food ordering system; and based from the researches mentioned; they consider the application to be independent form internet connection that links to the probability of limited connection, worse, disconnection. Detachment on web servers means no database server was constructed; the

order information is compressed on a barcode to increase portability. Additionally, bearing in mind to eliminate the risk of security threats associated with mobile devices [12].

2. SYSTEM ARCHITECTURE

In Figure-1, it illustrates the system architecture of cAPPeteria that consist of two android phones with an operating system of 4.0 Ice Cream Sandwich at least and installed with cAPPeteria application.

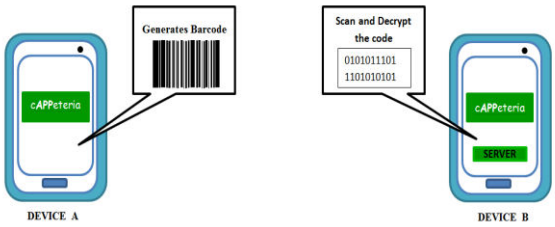

Figure-1. Device A (left) and device B (right).

In Figure-1, device A is where the cAPPeteria application is installed. This is the device the customer will be handling, it is where the menu is displayed and composition of order is created, after finalizing, the application will generate the barcode to be presented on the counter, represented by Device B, where it scans the barcode using image processing offered by the barcode scanner that decrypts the order information for it to be displayed on a language that humans can understand; process of demultiplexing.

A. Layout of device A

The layout and interfaces in this interactive application is simplified for the ease on the part of users.

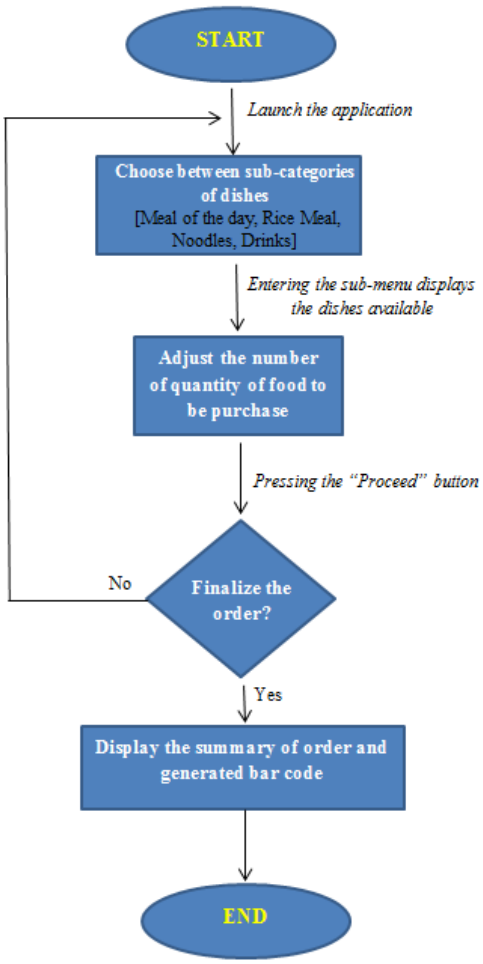

Figure-2. Flow of the interface at device A.

Figure-2 shows the flow of the interface at Device A, from launching of the application, the customer can choose their dishes from the Menu. It is classified into Rice Meal, Noodles, Meal of the Day, and Fast Food which are displayed as buttons as shown in Figure-3. After the user choose from one of the sub-categories, it will be directed to the page where the available dishes under that group is viewed in details then adjacent to the name of the dish, the user can adjust the number of quantity of food to be purchased as displayed in Figure-4. Pressing the "proceed" button will direct the user to the summary of what they've order under that specific category. On that point, the user can either continue to finalize the order or if not, the user can touch the back button for redirecting the display back to the application's homepage, the menu and repeating the selection from the sub-categories. Finalizing the order will display the summary of all the orders that the user had made and the prices multiplied by the quantity beside the name of the dish and on the bottom side is the total price of the order made.

The interface shown in Figure-3 (right) is showed after the "FASTFOOD" button is clicked in the Menu. This is where the customer will choose the food they want and vary the quantity by clicking the "+" button to increase the quantity of the food up to only 15 servings

and users must press "-" button to decrease the quantity. The customer must click the "PROCEED" button to review the summary of the order made that will be shown in Figure-4.

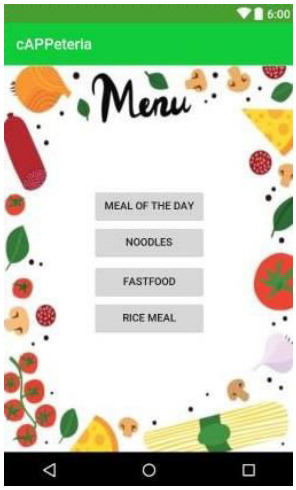

Figure-3. Menu tab.

Figure-4. The sub-category "Fast food".

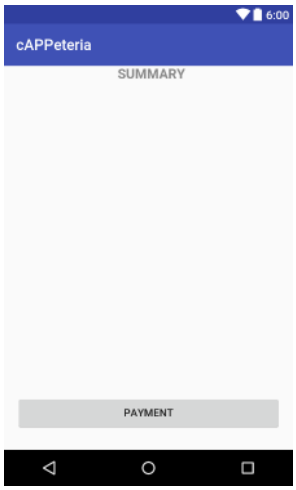

Figure-5. Summary of the order made.

A summary is displayed and reviewed by the customer, the "PAYMENT" button should be clicked for the order to be confirmed that is located below of the summary as presented in Figure-5. If the customer wants to make changes in their order, they can simply click the back button to return to the previous application interface which is selecting food from the menu. Clicking the "PAYMENT" button will proceed to an interface that will generate a barcode.

This barcode generated in device A is scanned on the counter represented by another phone, the device B. This device contains a barcode scanner application that decrypted the data in the barcode such as the amount to be paid by the customer.

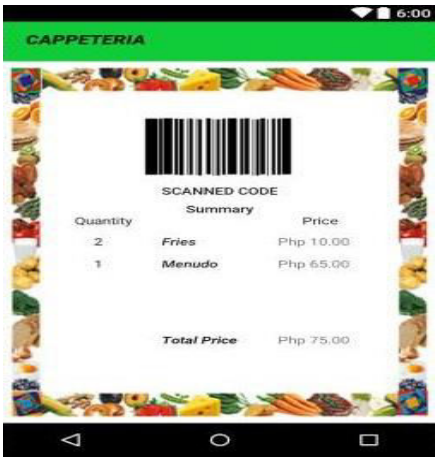

Figure-6. The scanned code and summary shown in device B.

Figure-6 is the interface in the device B after scanning the barcode in the device A. After a glance, it will show a brief invoice of the order including the quantity, price, and total amount to be paid by the customer.

B. Generating the barcode

Figure-7. Barcode generated by device A.

Figure-7 shows the barcode generated by the device A after reviewing the summary of the order and the "PAYMENT" button is pressed. This barcode is generated using ZXing "Zebra Crossing" library, the most known all-in-one codes library for Android. ZXing library is an open-source, multi-format 1D/2D barcode image processing library implemented in Java, which had ports to other languages. The barcode scanner and generator used in the applications of the devices A and B are based on

this library. The 1D code EAN-8 used is Code 128, which is an Aztec (beta) 2D product.

The pseudocode used in device A for generating a barcode is written below.

```
public final class AztecWriter implements Writer {
private static final Charset DEFAULT_CHARSET = Charset.forName("ISO-8859-1");
Public BitMatrix encode(String contents, BarcodeFormat format, int width, int height) {
        return encode (contents, format, width, height, null);}
private static BitMatrix renderResult(AztecCode code, int width, int height) {BitMatrix input = code.getMatrix();
if (input == null) {throw new IllegalStateException();}
   int inputWidth = input.getWidth();
   int inputHeight = input.getHeight();
   int outputWidth = Math.max(width, inputWidth);
   int outputHeight = Math.max(height, inputHeight);
BitMatrix output = new BitMatrix(outputWidth, outputHeight);
for (int inputY = 0, outputY = topPadding; inputY < inputHeight; inputY++, outputY += multiple) {contents
    for (int inputX = 0, outputX = leftPadding; inputX < inputWidth; inputX++, outputX += multiple) {
      if (input.get(inputX, inputY)) {
       output.setRegion(outputX, outputY, multiple, multiple);
      }
    }
   }
return output;
}
```

C. Scanning the barcode

Figure-8. Barcode scanned by device B.

Figure-8 displays the scenario of scanning the generated barcode from the device A using the camera of the device B. In here, the device B has an installed barcode scanner application. The researchers used a device B that that will represent the counter.

The pseudocode for the device B that is used to scan the barcode generated by the other device is:

```
public final class AztecReader implements Reader {public Result decode(BinaryBitmap image) throws NotFoundException, FormatException { return decode(image, null);}
public Result decode(BinaryBitmap image, Map<DecodeHintType,?> hints)
Detector detector = new Detector(image.getBlackMatrix());
ResultPoint[] points = null;
DecoderResult decoderResult = null;
Result result = new Result(decoderResult.getText(),
                   decoderResult.getRawBytes(),
                   decoderResult.getNumBits(),
                   points,
                   BarcodeFormat.AZTEC,
                   System.currentTimeMillis());
  List<byte[]> byteSegments = decoderResult.getByteSegments();
  if (byteSegments != null) {
result.putMetadata(ResultMetadataType.BYTE_SEGMENTS, byteSegments);
   }
  return result;
 }
}
```

Figure-9. Barcode encoded by device B.

The barcode scanned from device A by the device B will now be processed and encoded. Figure-9 illustrates the barcode encoded by the device B. As observed from the figure above, the device B encoded the data from the device A which includes the summary of order made by the customer and the amount that should be made.

The pseudocode in gathering the data from the scanned barcode is as follows:

```
public final class AztecDetector result extends DetectorResult{
 public AztecDetectorResult(BitMatrix bits,
                   ResultPoint[] points,
                   boolean compact,
                   int nbDatablocks,
                   int nbLayers) {
  super(bits, points);
  this.compact = compact;
  this.nbDatablocks = nbDatablocks;
  this.nbLayers = nbLayers;}
 public int getNbLayers() {
  return nbLayers;}
 public int getNbDatablocks() {
  return nbDatablocks;}
 public boolean isCompact() {
  return compact; } }
```

3. RESULTS AND DISCUSSIONS

Nowadays, almost every student owns a mobile device which has a feature of a computer but for handheld use. Android powers a lot of these devices, specifically smartphones.

Android applications are a software application running on the android platform.

The application created by the researchers will provide an assistance to students in ordering food in their school cafeteria.

The scenario starts with the student installing the "cAPPeteria" application. Opening the application will direct the student to the homepage that is set to be the "Menu". This contains buttons that are the classifications of food that can be ordered. The sub-categories are Rice Meal, Meal of the Day, Noodles, and Fast Food. Clicking the Fast Food button will display a list of food choices that are available. The customer can vary the amount of food by clicking the "+" and "-" buttons beside the food name and then clicking the "Proceed" button will direct the customer to an interface that contains the summary of the order that has been made together with the details such as the quantity, price and total amount that should be paid. Below that interface is a "Payment" button, clicking this button will provide the customer a barcode that will be presented to the counter to get the order after making their payment. This barcode holds a data that will be cracked at the counter which is represented by another device, named "device B". The device B has an installed barcode scanner that will decode the data in the generated barcode. Decoding this will display the brief invoice sales of the order made by the customer and the amount to be paid. The customer will then pay the amount and then claim the order that has been placed.

With this application, the customer can order food in the school cafeteria without lining up in the the school's dining hall, which saves the students a lot of their time and energy. The order can be made anywhere as long as the device used by the customer is connected to an internet connection. In this research, the "cAPPeteria" contains a given fixed set of food choices saved in the customer's device and has the ability to transfer the data through the barcode that is generated using a ZXing library. The barcode formed will be scanned by the device B in the counter in able to decode the data, and then payment will be made.

4. CONCLUSIONS

Relatively, with the "cAPPeteria" food ordering system, the researchers revolutionized the way students order their food in their school cafeterias, reduced the time spent in making orders and lessened the manpower that is a benefit for the cafeteria owner.

The application created by the researchers is user-friendly. It is easy to install, efficient, contains an easy-to-navigate GUI, easy to troubleshoot and adheres to standards.

For future study, there are many improvements that can be done.

The menu can be saved in the school server's database for it to be updated instantaneously with the application. The creators can give an update weekly with a new set of menus for the students.

The barcode scanning process can be done using a different device. It can be a real bar code scanner device that is found in stores for more accurate and fast speed scanning.

In the payment process, an additional mode of payment is possible. The customer can add their own credit or atm card for more convenient paying. No more cash to be counted at the counter, it can be just scanning the bar code then it will automatically show that the order has already been paid by the customer.

With these improvements, it can not only be applied on a simple school campus cafeteria. It can be expanded or applied into different fields that include queuing in line, making an order, paying and claiming of order.

The "cAPPeteria" can make the customers use their time efficiently and make their ordering of products using their smart phones conveniently.

ACKNOWLEDGEMENTS

Following are results of a study on the "Leaders in Industry-university Cooperation +" Project, supported by the Ministry of Education and National Research Foundation of Korea.

REFERENCES

[1] John T Conlan Thomas J Conlan. 1995. Manual Food Serving System. US Patent 5573082 Milwaukee.

[2] Green Daniel Curtis Jace. 2000. Restaurant Waiter Paging System. US Patent 6366196 Tennessee.

[3] 1958. University Bulletin: A Weekly Bulletin for the Staff of the University of California. 7: 83.

[4] T. Shimmura, T. Takenaka, M. Akamatsu. 2009. Real-time process management system in a restaurant by sharing food order information. 2009 International Conference of Soft Computing and Pattern Recognition.

[5] Arthur. 2014. Digital Wars: Apple, Google, Microsoft and the Battle for the Internet. p. 250.

[6] T. MA, J. Qian, Y. Tian, H. Zhang, X. Hu. 2015. The Design and Implementation of an Innovative Mobile Payment System Based on QR Barcode. 2015 International Conference on Network and Information Systems for Computers, System Frame.

[7] M.Z.H. Noor, A.A.A Rahman, *et al*. 2012. The Development of Self-service Restaurant Ordering System. University Teknologi MARA, Malaysia.

[8] Patel K.J.; Patel Umesh; Obersnel A. 2007.PDA-based wireless food ordering system for hospitality industry - A case study of Box Hill Institute. Wireless Telecommunications Symposium, 2007. WTS 2007

[9] M. Z. H. Noor , A. A. A. Rahman , M. F. Saaid, M. S. A. M.Ali, M. Zolkapli. 2012. The development of Self-Service Restaurant Ordering System (SROS).Control and System Graduate Research Colloquium (ICSGRC), IEEE

[10] A Samsudin, *et al.* 2011. A Customizable Wireless Food Ordering System with Real time Customer Feedback. IEEE Symposium on Wireless Technology and Application (ISWTA).

[11] H. Xu, B. Tang; W. Song. 2009. Wireless Food Ordering System Based on Web Services. Intelligent Computation Technology and Automation, 2009. ICICTA '09. Second International Conference on, 2009N.

[12] R. G. Duncan and M.M. 2000. Shabot "Secure remote access to a clinical data repository using a wireless personal digital assistant (PDA)".Proceedings of the American Medical Informatics Association Symposium. pp. 210-214.