National College of Ireland

BSc in Computing

2015/2016

OLUROTIMI OLUOKUN

X12509997

olurotimi.oluokun@student.ncirl.ie

PAYD

Technical Report

National College of Ireland

Declaration Cover Sheet for Project Submission

SECTION 1

Name: OLUROTIMI OLUOKUN
Student ID: X12509997
Supervisor: PAUL STYNES

SECTION 2 Confirmation of Authorship

The acceptance of your work is subject to your signature on the following declaration:

I confirm that I have read the College statement on plagiarism (summarised overleaf and printed in full in the Student Handbook) and that the work I have submitted for assessment is entirely my own work.

Signature: ____ Date:___________

Table of Contents

Executive Summary

This report was written to show the progress of the Payd project, to highlight eCommerce and mCommerce problems and to describe the technical solutions within this project. The purpose of this project is to make it easy for organisations to provide their customers with a means of making secure in-store and online payments. This report explores the background of mobile payments and the different methods in which mobile payments are currently made. The research draws attention to the security issues faced by merchants when it comes to payments for online transactions. The benefits of the system are discussed and the project's approach to solving the ecommerce security concerns is outlined. The report lists the different objectives that will be accomplished in the development of this system. The different technologies that are used within this project are described along with their individual functions and roles within the project. The results of an on-going survey for the project is discussed and used to develop the requirements for the project. The report displays different diagrams and screen-shots that show all the different functionalities and APIs that have been implemented into the project. Finally, the report uses the results obtained from surveys to identify potential problems and hindrances for the system's users and how they can be overcome.

1 Introduction

For the purpose of my final year project I will build an android application that allows users to make or accept eCommerce and mCommerce transaction payments. The project has been titled 'Payd'. The Payd project is intended to make it easy for organisations to provide their customers with a means of making secure mobile and online payments. The system will reduce in-store cash handling, allow users to conveniently pay with their smartphones, automatically keep merchants' and customers' transaction records, provide merchants with periodical statistics about their customers' transactions and provide merchants with a secure way of receiving e-commerce payments.

1.1 Background

The idea of a mobile payment system has passed through everyone's mind at one point or another. From entertainment and shopping to security and emergencies our mobile phones assist us in almost every area of our lives. Our mobile phones are probably the most important gadget we possess. With the rise of smartphones our phones have replaced newspapers, GPS devices and even TVs; it is expected that over they completely replace our wallets. There's a lot of evidence that suggests cash payments will soon be a thing of the past and this project outlines one of the many methods in which our smartphones may replace our wallets. The project increments the recent success of eCommerce by providing added security.

"*Mobile payments are considered one of the next big growth areas for mobile software… US Mobile Payments will reach $142 billion by 2019*" (Jason A, 2015).

There are four primary ways to make mobile payments.

1) Premium SMS based transactional payments

SMS payment is one of the most secure methods of mobile payments. In this payment method the customer doesn't require a pin, password or even a bank account. The customer simply sends a text message to the number being advertised, the network providers then charge their mobile account and the payment is made to the business.

2) Direct Mobile Billing

In direct mobile billing customers make purchases on an e-commerce website e.g. mobile game. After a two-step authentication involving pin and password the customer's mobile account is charged for the purchase.

3) Mobile Web Payments (WAP)

With Mobile web Payments consumer make payments using displayed webpages. They may enter their bank information or use additional mobile apps or software to complete transactions, e.g. PayPal app.

4) Contactless NFC (Near Field Communication)
NFC mobile payments have gained a lot of popularity since the NFC technology was integrated to smartphones. Android and Apple have created their own mobile payment systems using this technology. In the NFC payment method the user's payment information is stored on the device, during transactions the mobile device is placed near a NFC reader which reads the payment information from the customer's device.

1.2 Research

Research was carried out to help better understand the factors surrounding eCommerce and mCommerce in Ireland to evaluate how the project can make an impact.

ECommerce

There was plenty of evidence that shows that eCommerce is growing very fast in Ireland.

"*17% of the 2010 Financial Year total turnover was derived through online sales*", this figure has grown a lot since then. (Irish ecommerce survey, 2011)

As of 2014 reports stated that:

"*Revenue has increased by more than half (51pc), website traffic is up by 30pc, and average revenue per visit has grown by 17pc compared to the same period last year*". (Burke, 2014)

Security has become a huge concern with the rise of eCommerce in Ireland and around the world. The fact the payments are made over the internet makes it difficult for merchant to identify customers. Passwords or PIN code can be stolen or guessed.

"*Fraud is inevitable. It raises costs for the industry and scares away consumers*"
(Biometrics meets e-commerce, 2003)

The project being proposed will solve this problem by providing biometric authentication for eCommerce payments. This is intended to give both organisations and customers more confidence in eCommerce and promote even more growth.

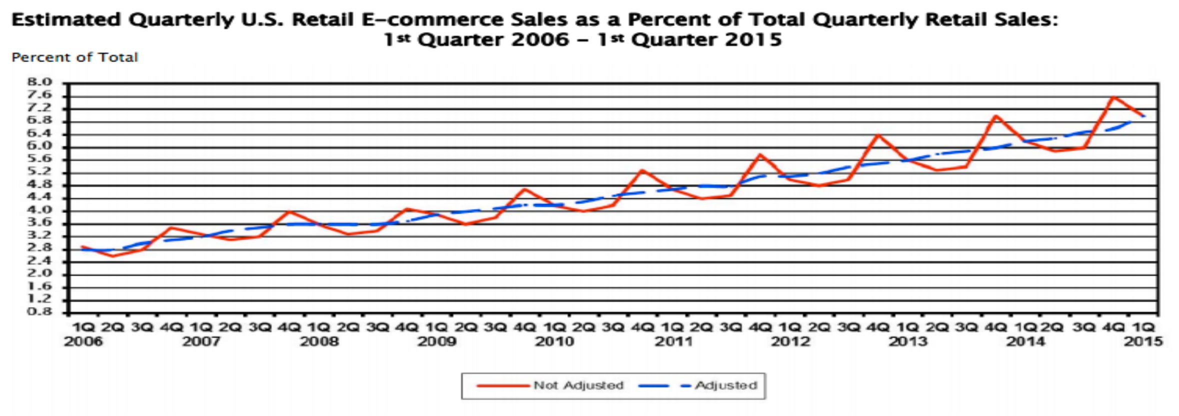


Figure 1 - US E-commerce sales currently average out at 6.8 percent of total sales

(Ecommerce growth statistics - UK and worldwide, 2015)

MCommerce

MCommerce hasn't grown as much as eCommerce has. Surveys show that consumers are positive about the potentials of mCommerce but organisations are lagging behind in terms of providing the means for them to do so. (*Retailers lagging in m-commerce stakes*, 2011) This project aims to make it easy for small and large organisations to provide their customers with a means of making mobile payments and in turn provides organisations with additional competitive advantages.

The system proposed will solve this problem by providing biometric authentication for eCommerce payments. This will hopefully give both organisations and customers more confidence in eCommerce and promote even more growth.

1.3 Aims

The following are the main objectives to be accomplished in the development of this project.

- **Create an android application that generates QR codes**

  As a foundation for this project an android application will be built to can scan and decode QR codes. The application will be able to scan QR codes that are on either paper or screens. The application will also include functionality to generate QR codes from plain text. The aim is to implement this functionality in a way that ensures minimal response time.

- **Create and connect android application to MySQL database**

  A database that will store user details and payment information will be created. A rational database will be used for the purpose of this project. Functionalities will be built to allow the app to communicate with the database. The functionalities will use PHP and Java POST and GET methods.

- **Integrate android personalised account with app**

  The android app will have functionality to allow users to sign-in, when the users sign-in a Payd account will be created on the user's device, this will allow the users' information to be stored on their device.

- **Create online resources for users**

  An online service that will allow merchants to register their store information and submit their PayPal information will be created. The website will allow Merchants to view their transaction history and view different statistics about their store and customers.

- **Implement the Onyx Fingerprint API**

  The ONYX fingerprint API will be implemented into the android application. This will be used to verify customers making e-commerce payments.

- **Setup and Integrate PayPal Android API**

  The android application will have the PayPal android API integrated into it, this will allow customers to sign into their PayPal account to make payments to merchants.

- **Setup and Integrate Clover API**

  The project will have the Clover API integrated into it, this will allow merchant to easily install and use the merchant app on their Clover devices.

- **Use datamining tools to process users' information**

  The apriori algorithm will be implemented using the R language.

1.4 Technologies

1.4.1 HTML, CSS & Bootstrap

A combination of PHP, HTML and CSS will be used to build the website. It will also be using bootstrap templates and tools to make sure the web-pages are responsive.

1.4.2 PHP

PHP will be used to print and modify database records.

1.4.3 MYSQL

The payment system will be use a MYSQL database to store user and transaction information.

1.4.4 Android

As the payment system is going to be an android application, the project will require a lot of android assets, tools and libraries.

1.4.5 Java

Android applications are developed using the Java programming language. Java will be used to manipulate and send data within the android app. The QR reader and scanner will also be implemented in java using the Zxing library.

1.4.6 Onyx

Onyx PassPrint SDK will be used to complete customer's registration
and validate customers using their fingerprint.

1.4.7 Zxing library

The Zxing library will be used to encode and decode QR codes. Implementing the Zxing library is straight forward and very reliable.

1.4.8 PayPal Android API

The android application will have the PayPal android API integrated into it to allow customers to sign into their PayPal account to make payments to merchants.

1.4.9 RStudio

RStudio is used to process the users' data which is collected at every transactions and is used to provide merchant with information on their store's performance.

1.4.10 Clover API

Implementing the Clover API allows the app to integrate with Clover devices. The app will be able to retrieve products names, prices and other transaction information from a Clover device, making the app much easier to use in a store environment.

2 System

2.1 Requirements

The requirements for this project were gathered using information retrieved from an online survey.

2.1.1 Functional requirements

- System should allow users to view their information and transaction history.
- The system should allow users to make or receive online payments.
- The application must be able to instantly scan QR codes.
- The system should be able to generate unique QR codes to process payments.
- The application must allow users to sign-in or register and store the details on the device.
- The application should be able to store and validate customers fingerprint.
- The system should allow merchants to view periodical reports on their customers and transactions.
- The system should use PayPal services to process payments.
- The Payd application should be usable with android devices.

2.1.2 User requirements

- Customers should be able to make in-store payments using their Android devices.
- Customers should be able to register on the system using their PayPal account.
- Merchant should be able to register their store details on the system.
- Merchant and customers will be able to view their transaction history on the system
- Merchants will be able to accept in-store payments using their Android devices.
- Merchants will be able to view weekly/monthly transaction summary diagrams.
- Customers should be able to use the application to scan QR codes

- Customers should be able to verify their identity with their fingerprint before completing an ecommerce transaction.

- Merchants' application should be able to generate QR codes

- Merchant should be able to receive payments from online transactions.

- Customers should be able to authenticate using their fingerprint.

2.1.3 Environmental requirements

- The android application must be able to run on Android 3.0, Android 4.0, Android 4.1 and Android 4.4 for maximum usability.

2.1.4 Usability requirements

- The app's menus and functions should be easily visible and easy to use and understand.

- The interfaces should be consistent between merchant and user application.

- Permanent actions that change the user's information should require confirmation from the user.

2.2 Design and Architecture

2.2.1 Use Case Diagram

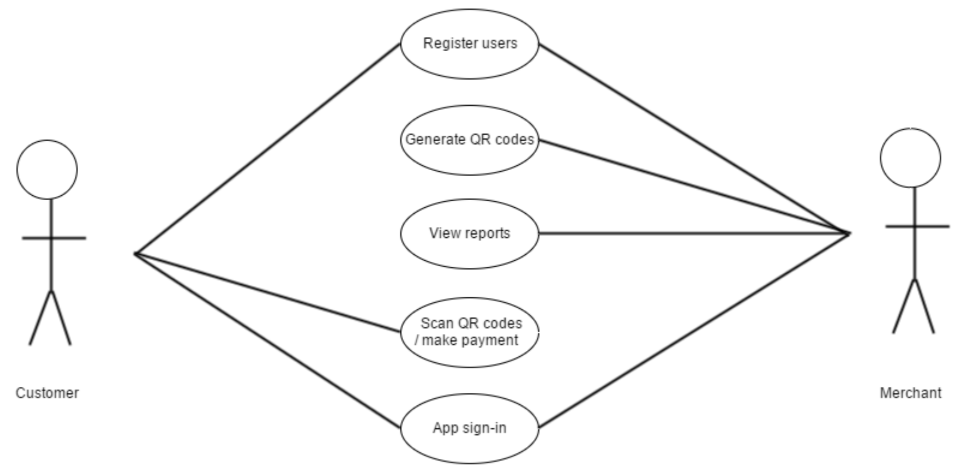


Figure 2 - Main Use Case

2.2.2 Sequence Diagrams

1. Customer Register Use Case Sequence Diagram

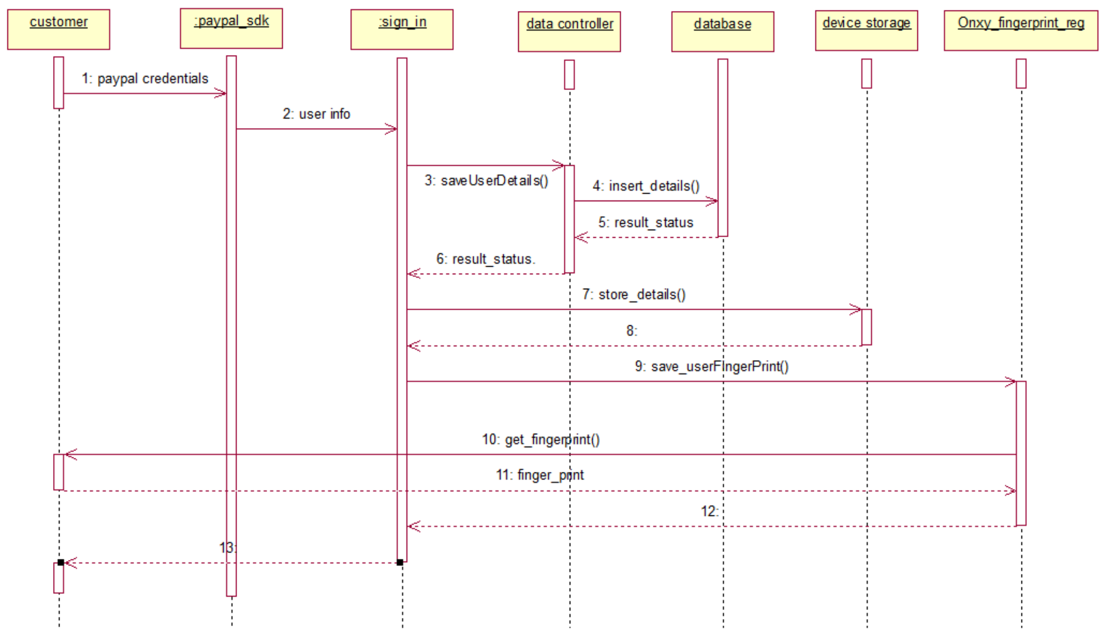


Figure 3 - Customer Register Use Case Sequence Diagram

2. Merchant Register Use Case Sequence Diagram

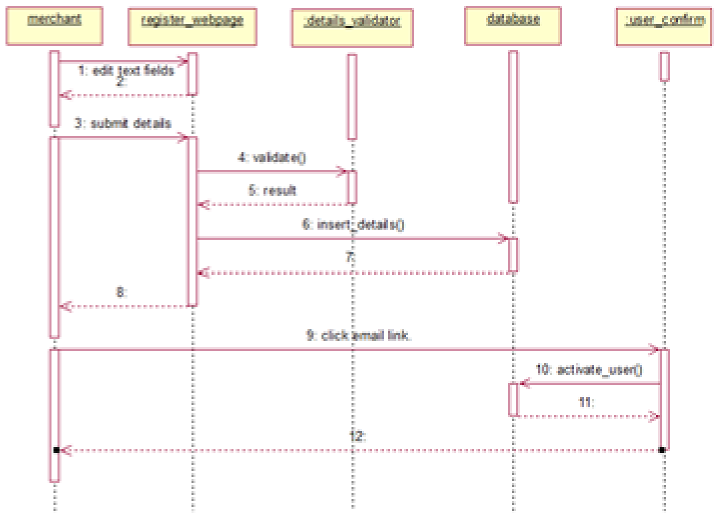


Figure 4 - Register Use Case Sequence Diagram

3. Merchant Generate QR Use Case Sequence Diagram

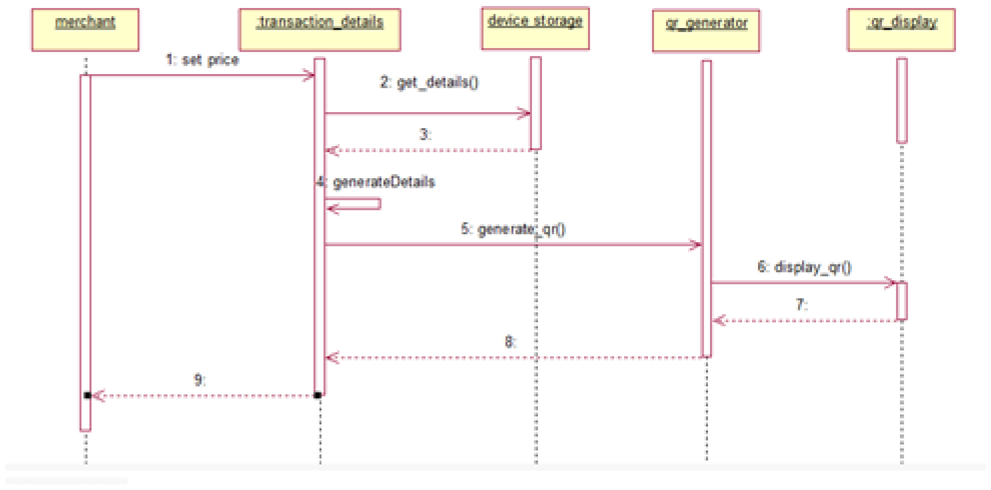


Figure 5 - Generate QR Use Case Sequence Diagram

4. Customer Scan QR Use Case Sequence Diagram

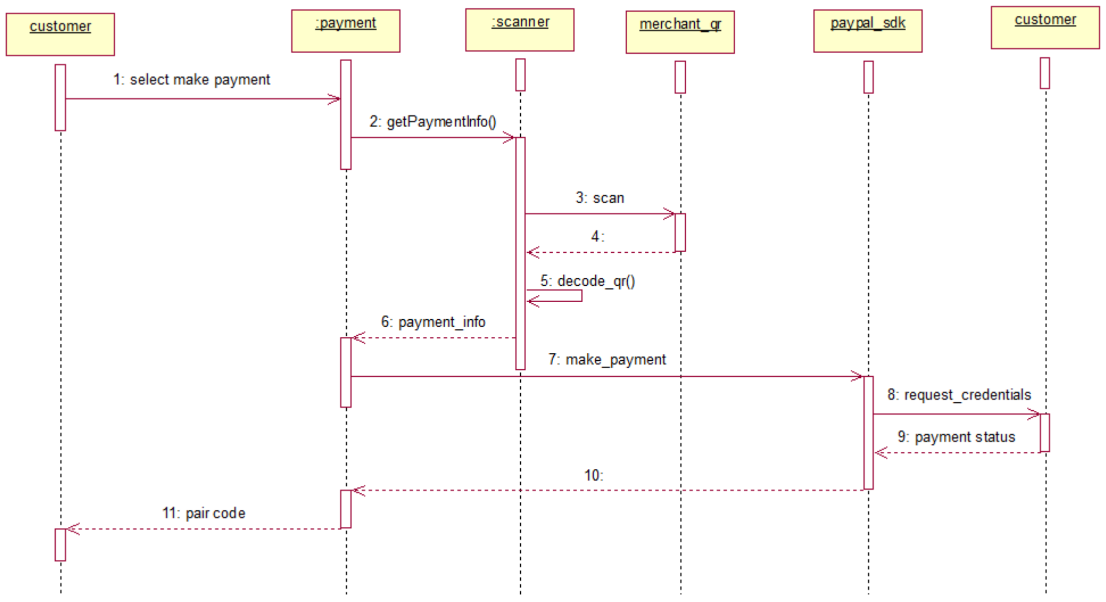


Figure 6 - Scan QR Use Case Sequence Diagram

5. Collaboration diagram

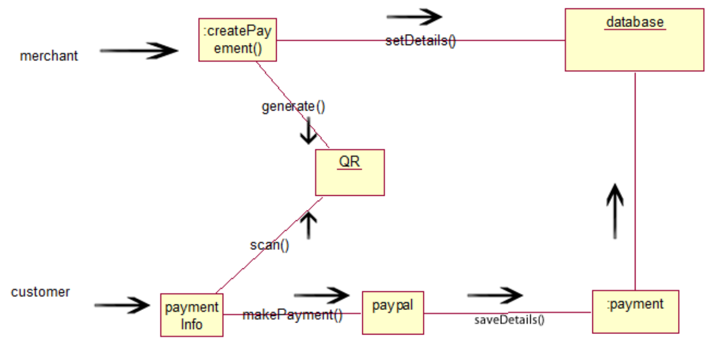


Figure 7 - System Collaboration diagram

2.2.3 Class Diagram

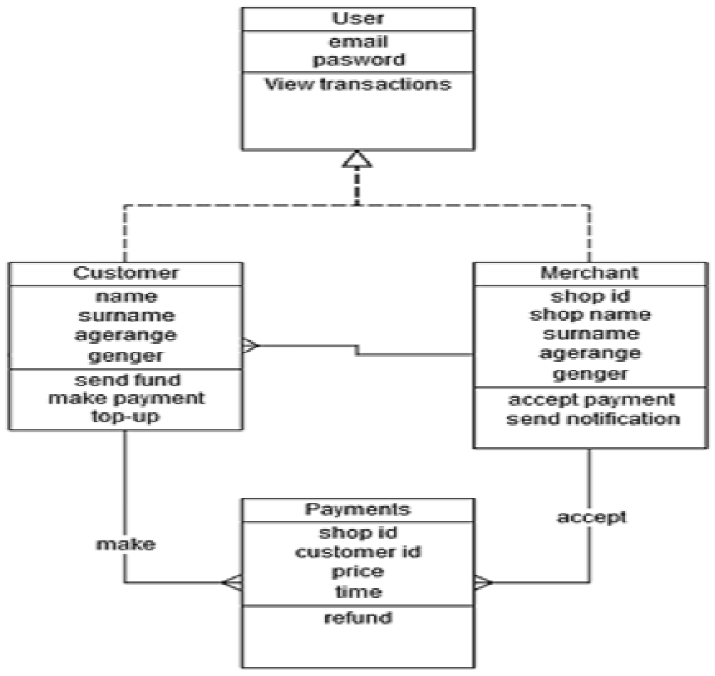


Figure 8 - System Class Diagram

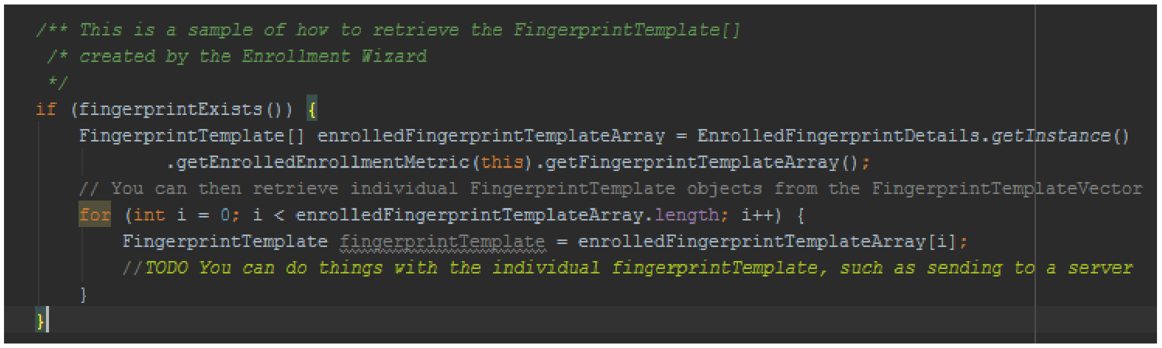
2.3 Implementation

Main classes/functions used in the code.

1. MainActivity.java

   The MainActivity class displays the activity (contents) that allows users to select the function they want to use. Before the class sets the activity it checks the file containing the user's information is present on the device, if the file is not found a new activity, PayPalDetActivity, is called.

2. OnyxEnroll.java
   The OnyxEnroll class is used to make calls to the PassPrint service. As shown below the OnyxEnroll class allows to check if there is a fingerprint already saved on the device. The class allows to either enrol a fingerprint or to validate one use the Onyx (PassPrint) service.



```
/** This is a sample of how to retrieve the FingerprintTemplate[]
 /* created by the Enrollment Wizard
 */
if (fingerprintExists()) {
    FingerprintTemplate[] enrolledFingerprintTemplateArray = EnrolledFingerprintDetails.getInstance()
            .getEnrolledEnrollmentMetric(this).getFingerprintTemplateArray();
    // You can then retrieve individual FingerprintTemplate objects from the FingerprintTemplateVector
    for (int i = 0; i < enrolledFingerprintTemplateArray.length; i++) {
        FingerprintTemplate fingerprintTemplate = enrolledFingerprintTemplateArray[i];
        //TODO You can do things with the individual fingerprintTemplate, such as sending to a server
    }
}
```

Figure 9 - OnyxEnrol Screenshot

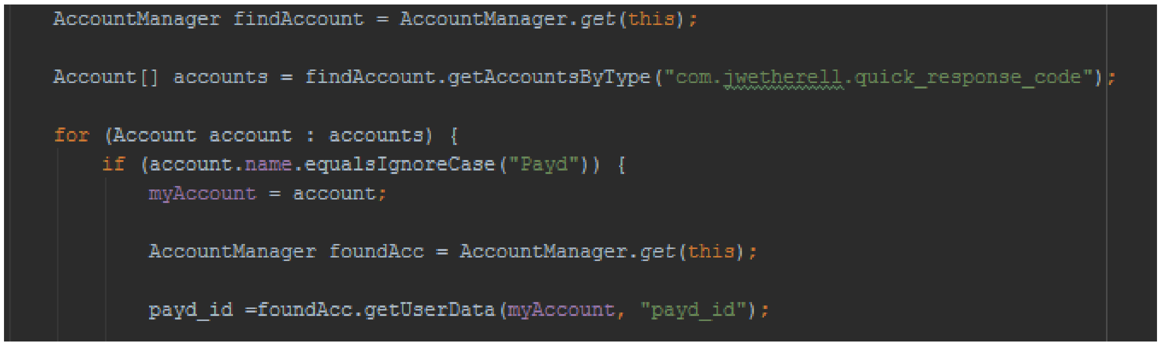
3. PayPalDetActivity.java
   The PayPalDetActivity is called to allow users to log into the app through their PayPal account. The class uses the PayPal API to allow users to log into their PayPal account, it then retrieves their information and saves it on the device.

4. PaymentActivity.java
   The PaymentActivity class also uses the PayPal API and is very similar to the PayPalDetActivity class. After a QR code has been scanned the price and merchant details retrieved is sent as parameter to this class. The PayPal service is then used to execute the payment, the class handles whatever response is received appropriately.

5. CaptureActivity.java & DecoderActivity
   The DecoderActivity class is used to recognize and scan QR codes, the class inherits the CaptureActivity class (used to design the camera layout) and implements the CameraManager class (an inbuilt android class that is imported).

6. Encoder.java
   The Encoder class is responsible for using the Zxing library to generate QR codes. The class receives a string as parameters and creates a QR code representing the string, in this case the details of the transaction are sent to the class.

7. AccountManager
   The android in-built AccountManager class was implemented in many of the classes developed for this application. The AccountManager class allows application accounts to be stored and accessed on a device, allowing the application to store the user ID and password securely on the device.



```
AccountManager findAccount = AccountManager.get(this);

Account[] accounts = findAccount.getAccountsByType("com.jwetherell.quick_response_code");

for (Account account : accounts) {
    if (account.name.equalsIgnoreCase("Payd")) {
        myAccount = account;

        AccountManager foundAcc = AccountManager.get(this);

        payd_id =foundAcc.getUserData(myAccount, "payd_id");
```

Figure 10 - AccountManager Screenshot

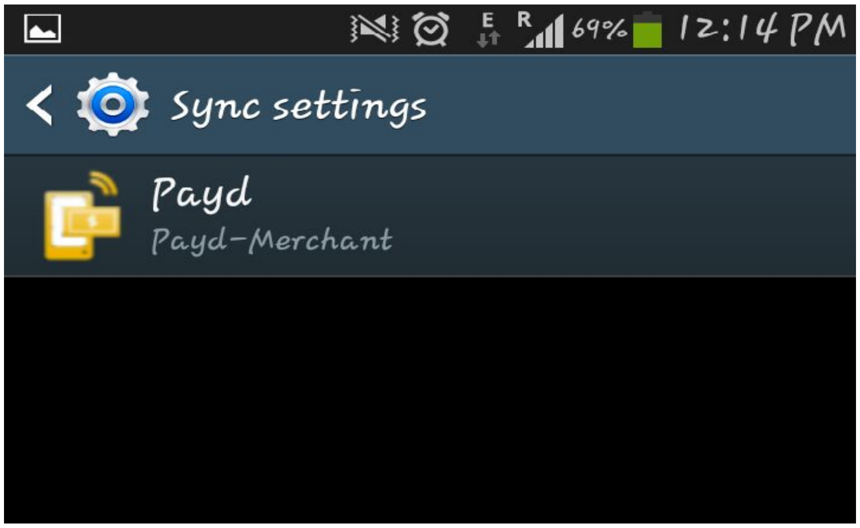
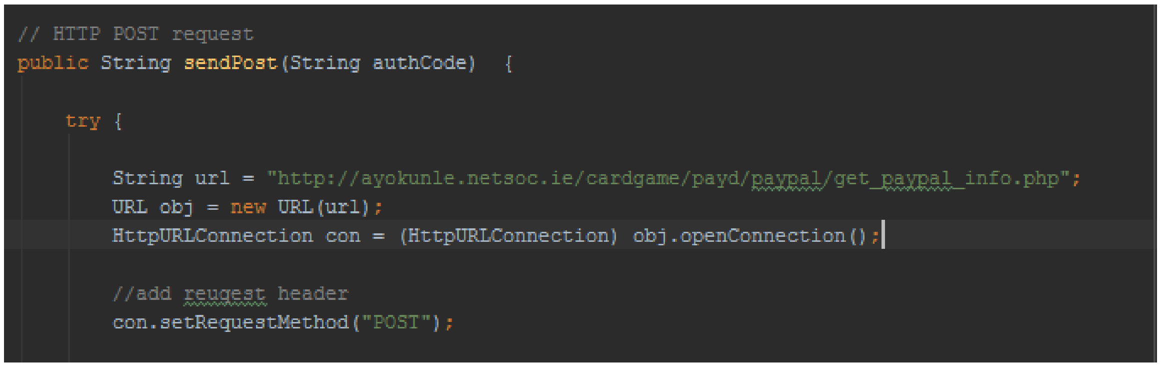


Figure 11 - Android Accounts Demonstration

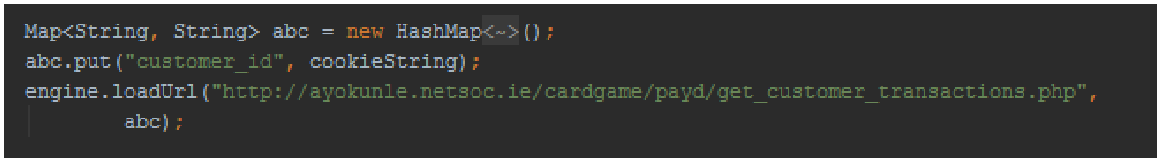
8. ServerComm.java
The ServerComm class uses the java.net.HttpURLConnection and java.net.URL classes to make http requests to the Payd server. This is the class responsible for sending and receiving information from the server. Requests made by the methods in the class are handled by a number of PHP files that are present on the server.



```
// HTTP POST request
public String sendPost(String authCode)  {

    try {

        String url = "http://ayokunle.netsoc.ie/cardgame/payd/paypal/get_paypal_info.php";
        URL obj = new URL(url);
        HttpURLConnection con = (HttpURLConnection) obj.openConnection();

        //add reuqest header
        con.setRequestMethod("POST");
```

Figure 12 - ServerComm screenshot

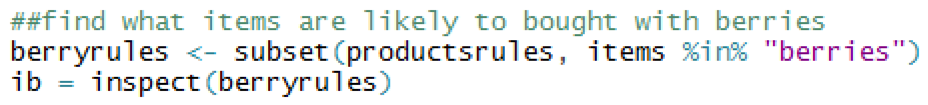
9. TransactionView.java
This TransactionView class uses a 'WebView' activity to display content from a specified webpage. A webpage that displays users' transaction information was developed and then loaded to the TransactionView class. The TransactionView class retrieves the user's unique ID from the device and sets it as a cookie so that the webpage displays information on the correct users.



```
Map<String, String> abc = new HashMap<~>();
abc.put("customer_id", cookieString);
engine.loadUrl("http://ayokunle.netsoc.ie/cardgame/payd/get_customer_transactions.php",
        abc);
```

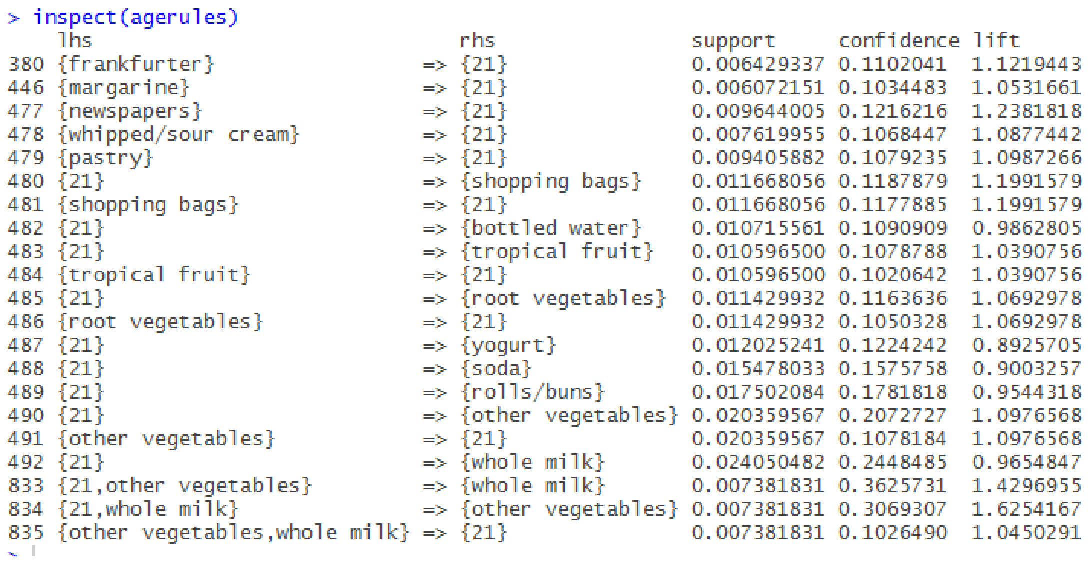
Figure 13 - TransactionView Screenshot

10. Datamining
The R language was used to mine the data being collected from transactions. A file containing sample transaction information was imported and assigned to variables in the environment, using the 'arules' package the sample data was processed to identify associations. The image below shows an example of how information was extracted after applying the apriori algorithm.



```
##find what items are likely to bought with berries
berryrules <- subset(productsrules, items %in% "berries")
ib = inspect(berryrules)
```

Figure 14 - Arules Demonstration



```
> inspect(agerules)
    lhs                               rhs                  support     confidence lift
380 {frankfurter}                  => {21}                 0.006429337 0.1102041  1.1219443
446 {margarine}                    => {21}                 0.006072151 0.1034483  1.0531661
477 {newspapers}                   => {21}                 0.009644005 0.1216216  1.2381818
478 {whipped/sour cream}           => {21}                 0.007619955 0.1068447  1.0877442
479 {pastry}                       => {21}                 0.009405882 0.1079235  1.0987266
480 {21}                           => {shopping bags}      0.011668056 0.1187879  1.1991579
481 {shopping bags}                => {21}                 0.011668056 0.1177885  1.1991579
482 {21}                           => {bottled water}      0.010715561 0.1090909  0.9862805
483 {21}                           => {tropical fruit}     0.010596500 0.1078788  1.0390756
484 {tropical fruit}               => {21}                 0.010596500 0.1020642  1.0390756
485 {21}                           => {root vegetables}    0.011429932 0.1163636  1.0692978
486 {root vegetables}              => {21}                 0.011429932 0.1050328  1.0692978
487 {21}                           => {yogurt}             0.012025241 0.1224242  0.8925705
488 {21}                           => {soda}               0.015478033 0.1575758  0.9003257
489 {21}                           => {rolls/buns}         0.017502084 0.1781818  0.9544318
490 {21}                           => {other vegetables}   0.020359567 0.2072727  1.0976568
491 {other vegetables}             => {21}                 0.020359567 0.1078184  1.0976568
492 {21}                           => {whole milk}         0.024050482 0.2448485  0.9654847
833 {21,other vegetables}          => {whole milk}         0.007381831 0.3625731  1.4296955
834 {21,whole milk}                => {other vegetables}   0.007381831 0.3069307  1.6254167
835 {other vegetables,whole milk}  => {21}                 0.007381831 0.1026490  1.0450291
```

Figure 15 - Associations with 21 year olds

The above screenshot shows the different items that are most frequently bought by 21 year olds. The screenshot also shows the support, confidence and lift of each association. The confidence of a rule indicates the probability of both the antecedent and the consequent appearing in the same transaction. Confidence is the conditional probability of the consequent given the antecedent. The support of a rule indicates how frequently the items in the rule occur together. Both support and confidence must be used to determine if a rule is valid. The Lift indicates the strength of a rule over the random co-occurrence of the antecedent and the consequent, given their individual support. It provides information about the improvement, the increase in probability of the consequent given the antecedent.
(Oracle, 2010)

2.4 Testing

2.4.1 Compatibility Testing

1. **Test Risks / Issues**
   - Application installation or operation failure due to android version.
   - Incompatible hardware on different android devices.
   - Permissions requests process may vary between devices.

2. Items to be Tested / Not Tested

Item to Test	Test Description
App installation	Using the android Compatibility Test Suite, tests will be carried out on Android 3.0, Android 4.0, Android 4.1 and Android 4.4.
QR Scan	Tests will be carried out on different tablets and android devices to ensure the QR scanner operates correctly on Android 3.0, Android 4.0, Android 4.1 and Android 4.4.
Android Permissions	Tests will be carried out on different tablets and android devices to ensure the app receives all the requested permissions required to perform different functions.
Payment	Functions that require the PayPal API will be tested on multiple devices and android versions. These test will also be carried out using the android Compatibility Test Suite.
Fingerprint Scan	Functions that require the PassPrint API will be test on multiple devices and android versions.

3. Test Approach(s)

Tests will be carried out on Android 3.0, Android 4.0, Android 4.1 and Android 4.4. Most of the test will be perform using the android Compatibility Test Suite (CTS). The CTS runs on a desktop machine and executes test cases directly on attached devices or an emulator. The CTS is a set of unit tests designed to be integrated into the daily workflow (such as via a continuous build system) of the engineers building a device. Its intent is to reveal incompatibilities early on, and ensure that the software remains compatible throughout the development process.

(Compatibility test suite, no date)

4. Test Regulatory / Mandate Criteria

The android application should function and respond identically across all the android versions it is tested against. The android CTS will be used to measure the application's performance (e.g. audio quality, accelerometer, etc.).

5. **Test Pass / Fail Criteria**

The application can only pass the compatibility test when all the functions listed above respond correctly on Android 3.0, Android 4.0, Android 4.1 and Android 4.4.

6. **Test Deliverables**

The test should identify which is the optimum android version for the application and which android versions require the app to have additional support to function correctly. The test should also provide measurements on the application's performance on each of the android versions tested (e.g. audio quality, accelerometer, etc.).

7. **Test Suspension / Resumption Criteria**

The test is suspended if any of the functions listed above fails to respond correctly and is resumed after the issues have been resolved.

8. **Test Environmental Needs**

The android Compatibility Test Suite software is required to perform the compatibility tests. The CTS runs on a desktop machine and executes test cases directly on attached devices or an emulator.

9. **Test Summary**

   I. Metrics

Test Case Status

Test cases planned	Test cases executed	Test cases Pass	Test cases Failed
42	42	42	0

II. Results

Items Tested	Test Results
App installation	Using the android Compatibility Test Suite the app installed successfully on Android 3.0, Android 4.0, Android 4.1 and Android 4.4. A total of ten test cases were carried out.
QR Scan	Six different android version were used to test the QR scan function. All six of the tests were successful.
Android Permissions	Using the android Compatibility Test Suite a total of ten test cases were carried out. The app successfully received the requested permissions on all android versions between 14 and 23
Payment	Functions that require the PayPal API were tested on multiple devices and android versions. All ten of the test cases performed were successful.
Fingerprint Scan	Functions that require the PassPrint API were test on multiple devices and android versions. All six of the test cases performed were successful.

2.4.2 Functional Testing

1. Test Risks / Issues

- Programming errors causing app to misbehave or crash.
- The APIs that the app uses may cause errors or misbehave.

2. Items to be Tested / Not Tested

Item to Test	Test Description
Login / Logout	Tests will be carried out using correct and incorrect account credentials and information to ensure the app responds correctly. The tests will also ensure the correct user information is stored on the device and is removed when user logs out.
QR Generator	Tests will be on different android devices to ensure that QR codes are generated and positioned correctly on different devices and screen sizes. The test will also ensure that the QR code generated are valid.
QR Scan	Tests will be carried out using different tablets and android devices to ensure the app can detect and scan both valid and invalid QR codes. These test will also ensure that the app can handle erroneous QR and handle exceptions. These tests will also ensure QR codes can be scanned using different camera specifications.
Clover API	Tests will be carried out to make sure the Clover API is integrated correctly with the application and returns all the required information without errors.
Payment	Payment and PayPal login functions that require the PayPal API will be tested on multiple devices to ensure the API integrates with the app correctly. Different use cases will be tested to ensure there are no errors within the API.
Fingerprint Scan	Functions that require the PassPrint API will be test on multiple devices to make sure the API functions correctly using different camera specifications. Different use cases will be tested to ensure there are no errors or uncaught exceptions.
Server Communication	Test for server communication will ensure that the application receives all the permissions required to access the internet and that it is able to respond appropriately when there is no internet connection available.

3. **Test Approach(s)**

To test the app's functions it will be installed on an emulator and then tested using AndroidJUnitRunner. The AndroidJUnitRunner class is a JUnit test runner that lets developers run JUnit 3 or JUnit 4-style test classes on Android devices. (Testing support library, no date)

4. **Test Pass / Fail Criteria**

The application will only pass the functionality test when all the functions and APIs listed above respond correctly on multiple devices without errors.

5. **Test Deliverables**

The test should identify which is the optimum android version for the application and which android versions require the app to have additional support to function correctly. The test should also provide measurements on the application's performance on each of the android versions tested (e.g. audio quality, accelerometer, etc.).

6. **Test Suspension / Resumption Criteria**

The test is suspended if any of the functions listed above fails to respond correctly and is resumed after the issues have been resolved.

7. **Test Environmental**

The android Testing Support Library will be required along with Android Studio to perform these tests.

8. Test Summary

I. Metrics

Test Case Status

Test cases planned	Test cases executed	Test cases Pass	Test cases Failed
30	30	28	2

II. Results

Items Tested	Test Results
Login / Logout	Tests were carried out using the AndroidJUnitRunner. Correct and incorrect account credentials and information were entered into the login forms. The application responded as expected when incorrect information was entered and logged users in when correct information was entered.
QR Generator	Tests performed on this function were successful. Different QR codes were generated and positioned correctly on different devices and screen sizes. All the QR codes generated were valid.
QR Scan	All the test carried out for this function were successful on the different types of devices used. The QR scanner was able to recognize valid and invalid QR codes and responded as expected.
Clover API	Tests on this function were successful, the API returned all the correct values (e.g. Prices, product name, etc.)
Payment	Payment and PayPal login functions that require the PayPal API were tested. Test cases for all possible scenarios were performed on the payment function and were all successful.
Fingerprint Scan	Functions that require the PassPrint API were tested on multiple devices to make sure the API functions correctly using different camera specifications. Four of the six devices used to test the function performed successfully. The two devices that failed this test had cameras that were below 4 mega-pixels

Server Communication	Tests for server communication showed that the app received all the permissions it requested for. The applications also behaved appropriately when there was no internet connection available on the devices. .

2.4.3 Load Testing

1. **Test Risks / Issues**

- The app's performance may vary on different android versions and type of devices.
- The app's performance may also vary depending on its load and device specifications.

2. **Items to be Tested / Not Tested**

Item to Test	Test Description
QR Generator	Tests will be carried out on different android devices to ensure that QR codes are generated and positioned correctly with increasing amount of data. This will be done by modifying the application so it generates and displays mock data that will be increased steadily.
QR Scan	Tests will be carried out using different android devices to ensure the app can detect and scan QR codes that contain an increasing volume of data. This test will also determine how different camera specifications respond to large amounts of QR patterns. The test will involve mock data being generated and scanned, the volume of data in the QR codes being scanned will be increased steadily.
Server Communication	Test for server communication will ensure that the application is able to correctly send and receive increasing amounts of data. This will be achieved by modifying the application so it sends and requests mock data that will be increased steadily.

3. Test Approach(s)

To perform the load tests on the app’s functions it will be installed on an emulator and then tested using AndroidJUnitRunner. Some of the tests will be carried out using real android devices so that the app’s performance can be analyzed using different camera specifications.

4. Test Pass / Fail Criteria

The QR Generator function will only pass the load test if it is able to generate and display QR codes that contain a minimum of 300 bytes of data on devices with different specifications.

The QR scan function will only pass the load test if it is able to recognize and scan QR codes that contain a minimum of 300 bytes of data on devices with different specifications.

The Server Communication function will only pass the load test if it is able to send and receive a minimum of 5.00 kilobytes on devices with different specifications.

5. Test Deliverables

The test should identify any errors or factors that may cause the application to crash or misbehave. The tests should also provide measurements on the application’s performance as the load on the listed functions is increased.

6. Test Suspension / Resumption Criteria

The test is suspended if any of the functions listed above fails to respond correctly and is resumed after the issues have been resolved.

7. Test Environmental

The android Testing Support Library will be required along with Android Studio to perform these tests. Addition android devices with different camera specifications will be required to test the QR scan functionality.

8. **Test Summary**

I. Metrics

Test cases planned	Test cases executed	Test cases Pass	Test cases Failed
25	25	25	0

II. Results

Item Tested	Test Results
QR Generator	Tests carried out on different android devices show that the QR generator functions correctly independent of the screen sizes. The generator was able to generate QR codes that contained between 200 and 400 bytes of data on all devices.
QR Scan	The QR scan functionality was tested on six different android devices with different camera specifications. The scanner was able to recognize QR codes that contained between 200 and 400 bytes of data on all devices.
Server Communication	Test on this function showed that the server communication function is able to handle more than 2mb of data being received and sent.

2.4.4 Stress Testing

1. **Test Risks / Issues**

- Loss of data may occur in cases where the application is put under stress
- The app may crash if a device has significantly low computational resources. (e.g., RAM, memory, etc.)
- Deadlocks or unavailable resources and permissions issues may cause the app to crash or misbehave.

2. Items to be Tested / Not Tested

Item to Test	Test Description
QR Scan	The stress test for the QR scan function will be done on android emulators with low quality specifications and limited computational resources This test will help analyze how the QR scanner responds under stress and with limited resources, it will also help in making sure all exceptions and errors are handled correctly. The QR scanner will also be tested against erroneous and invalid QR codes that exceed the anticipated volume of data (300 bytes).
QR Generator	Tests for the QR Generator function will also be done on android emulators with low quality specifications and limited computational resources. Again the test will help analyze how the QR generator responds under stress and with limited resources. The app will be modified to generate large amounts that exceed the anticipated volume of data (300 bytes).
Server Communication	To stress test the server communication functions the app will be modified to send and receive multiple requests simultaneously using android emulators with low quality specifications and limited computational resources. (Limited internet connection).
File Handling	The file handling functionalities within the app will be tested by deleting, altering and denying the app access to files that are essential to the app.

3. Test Approach(s)

To perform stress tests on the app’s functions it will be installed on an emulator and tested using AndroidJUnitRunner. Again, some of the tests will be carried out using real android devices so that the app’s performance can be analyzed using different camera specifications.

4. Test Pass / Fail Criteria

The QR Generator function will only pass the stress test if it is able alert the user that the process has been cancelled when the QR code's data exceeds 400 bytes of data.

The QR scan function will only pass the stress test if it is able to recognize and scan QR codes that exceed the maximum 400 bytes of data and alert users that the QR scanned is invalid.

The Server Communication function will only pass the stress test if it is able to alert users of connection failures and locally store data that has failed to send.

The File-handling functions will only pass the stress test if the app is able to alert the user of the problem (e.g. memory full) and cancel the payment processes without crashing.

5. Test Deliverables

The test should identify any errors or loss of data that hasn't been handled when the application fails under stress. The tests should also provide measurements on the application's performance before it fails under stress.

6. Test Suspension / Resumption Criteria

The test is suspended if any of the functions listed above fails to respond correctly and is resumed after the issues have been resolved.

7. Test Environmental

The android Testing Support Library will be required along with Android Studio to perform these tests. Addition android devices with different camera specifications will be required to test the QR scan functionality under stress.

8. Test Summary

I. Metrics

Test cases planned	Test cases executed	Test cases Pass	Test cases Failed
35	35	35	0

II. Results

Items Tested	Test Results
QR Scan	The stress test on this functionality showed that it was able to function under stress and with limited resources (e.g. RAM, etc.). The QR scanner was able to scan QR codes that contained a maximum of 466 bytes. After this maximum is reached the app simple alerts the user that the QR code is invalid. The specifications of the devices did not prevent the function from performing correctly and the app did not crash.
QR Generator	The app was able to generate large amounts that exceeded 466bytes which is the maximum that can be scanned. The specifications of the devices did not prevent the function from performing correctly and the app did not crash while under stress.
Server Communication	The app was able to handle up to fifteen heavy simultaneous request before its performance began to deteriorate. When the app eventually crashed from the load it was put under vital information were stored in a file.
File Handling	The test showed that the app alerts users when there was a problem with accessing or handling files without crashing.

2.4.5 Integration testing

1. **Test Risks / Issues**

- The application may lose data or crash if one of the APIs being used returns an unprecedented response.

2. **Items to be Tested / Not Tested**

Item to Test	Test Description
Clover API	To ensure that the Clover API fully integrates with the application tests will be carried out to exploit all the possible usages and outcomes of the Clover system. The app will be monitored using the android Testing Support Library and emulator.
Payment	To make sure the PayPal API fully integrates with the application the API will also undergo tests to exploit all the possible usages and outcomes of the PayPal API. The app will also be monitored using the android Testing Support Library and emulator. Responses such as 'Cancelled Payments' and 'Incorrect Credentials' must be handled correctly by the app
Fingerprint Scan	The integration test for the PassPrint API will be conducted similar to the Clover and PayPal API. Different use cases will be tested to ensure there are no errors or uncaught exceptions.
Zxing Library	Again, all the possible scenarios and responses by the Zxing library will be tested to ensure that there are no errors or uncaught exceptions.

3. **Test Approach(s)**

As with the other tests, the app will be installed on an emulator and tested using AndroidJUnitRunner. Some of the tests will be carried out using real android devices so that the app's performance can be analyzed using a real device camera.

4. Test Pass / Fail Criteria

The application will only pass the integration test if all the API's responses have been tested and handled correctly by the app.

5. Test Deliverables

The test should identify any of the API's exceptions or responses that hasn't been handled by the application.

6. Test Suspension / Resumption Criteria

The test is suspended if any of the API's responses are not handled correctly and is resumed after the issues have been resolved.

7. Test Environmental

The android Testing Support Library will be required along with Android Studio to perform these tests. Addition android devices with different camera specifications will be required to test the QR scan functionality under stress.

8. Test Summary

I. Metrics

Test cases planned	Test cases executed	Test cases Pass	Test cases Failed
25	25	25	0

II. Results

Items Tested	Test Results
Clover API	Test cases for all the possible outcomes of the Clover API were performed. The tests were successful, the app handled all the responses from the Clover API correctly.
Payment	Test cases for the payment function showed that the PayPal API fully integrates with the application. All the possible responses from the PayPal API were handled appropriately and the user are informed of the outcomes of the payment.
Fingerprint Scan	Different use cases were tested and showed that there are no errors or uncaught exceptions in the communication between the API and the app.
Zxing Library	All the possible outcomes of scanning and generating a QR code (e.g. Invalid QR, etc.) were tested and were successful.

2.4.6 User Acceptance Testing

1. Test Risks / Issues

- Users' views, perspectives and expectations of the application may differ from developers'.
- Application design or usage may too complex or too simple for intended users.
- Applications functionality may not suit the intended users.

2. Items to be Tested / Not Tested

Item to Test	Test Description
Login / Logout	To test the usability of this function test users will be presented with a questionnaire after they have used the login and logout functions.
QR Generator	To test the usability of this function test users will be presented with a questionnaire after they have used the application to scan a QR.
QR Scan	To test the usability of this function test users will be presented with a questionnaire after they have used the app to scan a QR.
Clover API	To test the usability of this function test users will be presented with a questionnaire after they have used this function to initiate a transaction.
Payment	To test the usability of this function test users will be presented with a questionnaire after they have completed a PayPal transaction with the functionality.
Fingerprint Scan	To test the usability of this function test users will be presented with a questionnaire after they have used the PassPrint functions.

3. Test Approach(s)

A survey was conducted using knowledge gained from chapter 6 of Jakob Nielsen's book on usability testing. (Nielsen, 1993). Users will first be given a short demonstration of the application before testing it themselves. After the users test the application they will be given the survey to complete. The questions on the survey are mostly based on Nielson's usability questionnaire. (Nielsen, 1997)

4. Test Deliverables

This test should help understand users' views and expectations of the application.

The test should also help determine if the system solves problems for the intended users.

The test should identify any issues with user interaction or complexity.

5. **Usability Test Questionnaire**

How easy is it to find specific information on this app?

How satisfied are you with the app's quality of language?

How frustrated did you feel while working in this app?

Compared to what you expected, how quickly did the tasks go?

How easy is it to work with the contents app?

How complete is the app's solution to mobile payments?

How confused did you feel while working on this app?

6. **Test Summary**

I. Metrics

No. test users	Tests performed	Positive Feedback	Negative Feedback
10	10	58	12

II. Results

The general user feedback were positive, 85 percent of the questions asked got a positive response. Some users had issues with the amount of information on that was on the generated QR while others had issues with completing a payment as they did not have a PayPal account. The test users also felt that the fingerprint scanning while logging in to the application took too long. The result of the survey showed that the app was simple to use and responded quickly.

2.4.7 Evaluation

All the test cases that were planned and performed were successful except for the PassPrint API functional test. Whenever any of the functions didn’t respond as expected the issue was documented and solved, the test was then repeated. The user acceptance test was also successful, the general feedback showed that the system was user-friendly and was a suitable solution to mobile payments.

2.5 Graphical User Interface (GUI) Layout

- Merchant Registration GUI

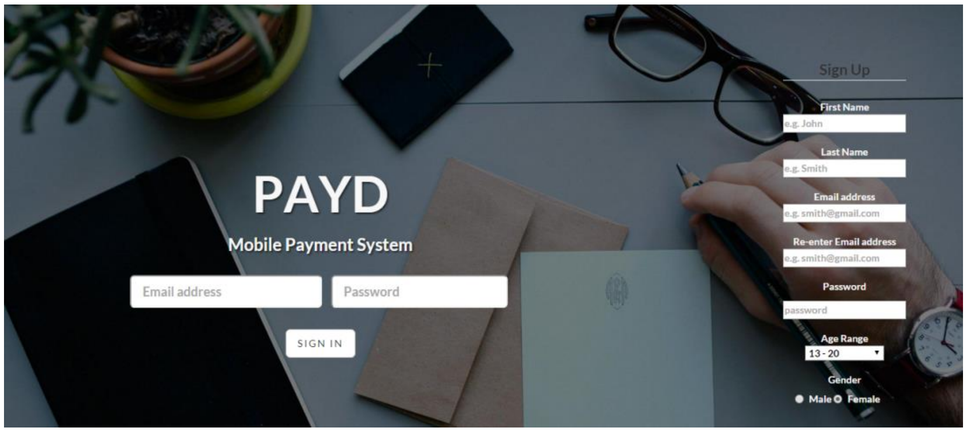


Figure 16 - Registration GUI

Merchants may register their personal and store details on the above website.

- Merchant Sign-in GUI

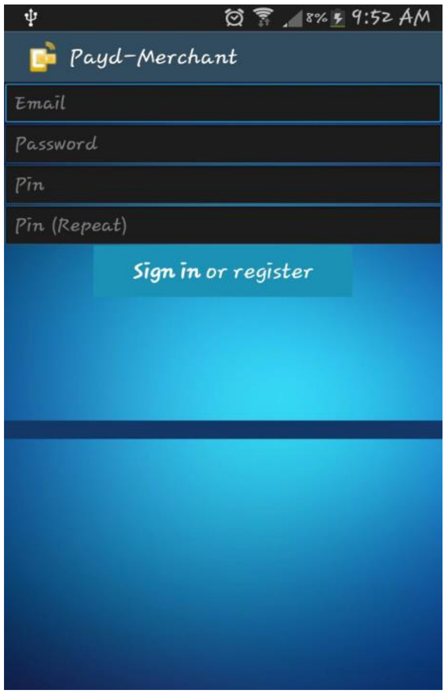


Figure 17 - Merchant Login

The above GUI allows merchants to log into to android devices.

- Merchant Performance

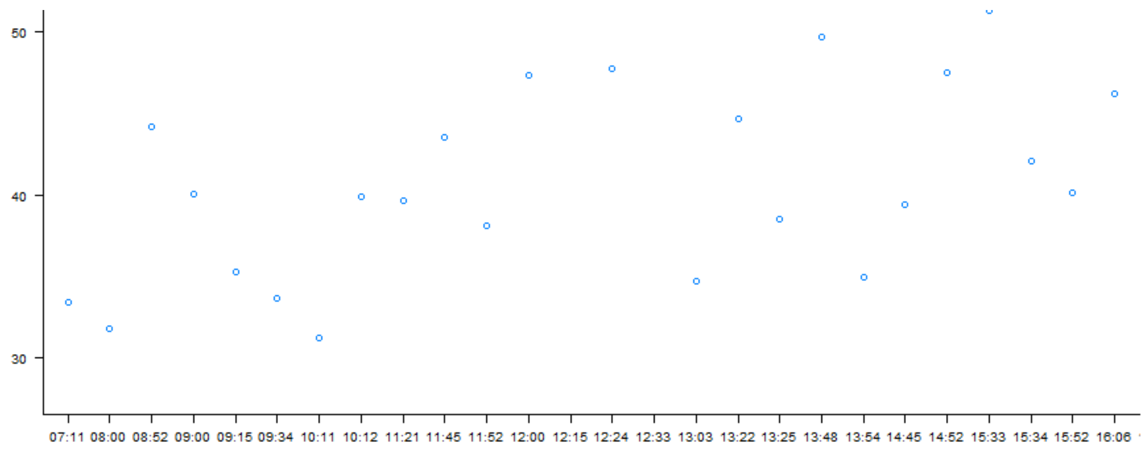


Figure 18 - Merchant Time Performance

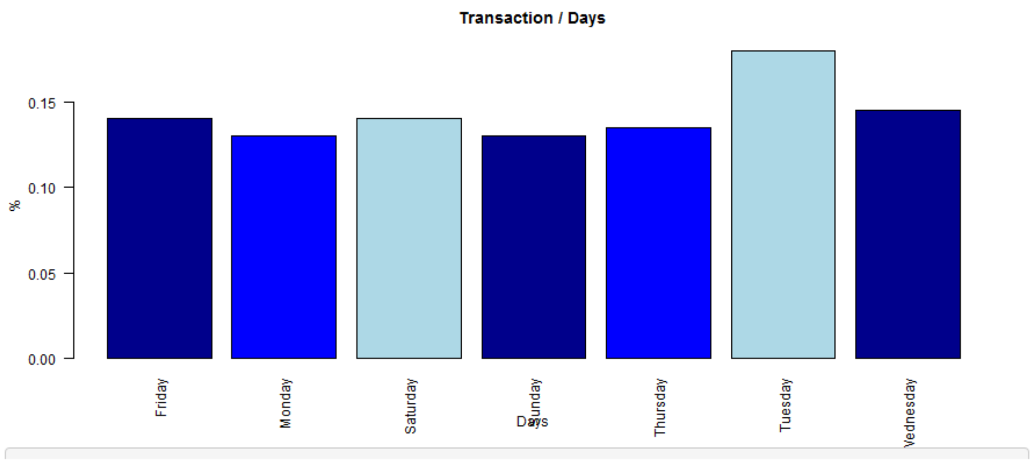


Figure 19 - Merchant Weekday Bar chart

The above graphs are examples of some of the information provided to merchants on their store's performance. The first graph show how many transactions are completed approximately every hour, while the second graph shows how many transactions are completed on each day of the week.

- Customer PayPal Login

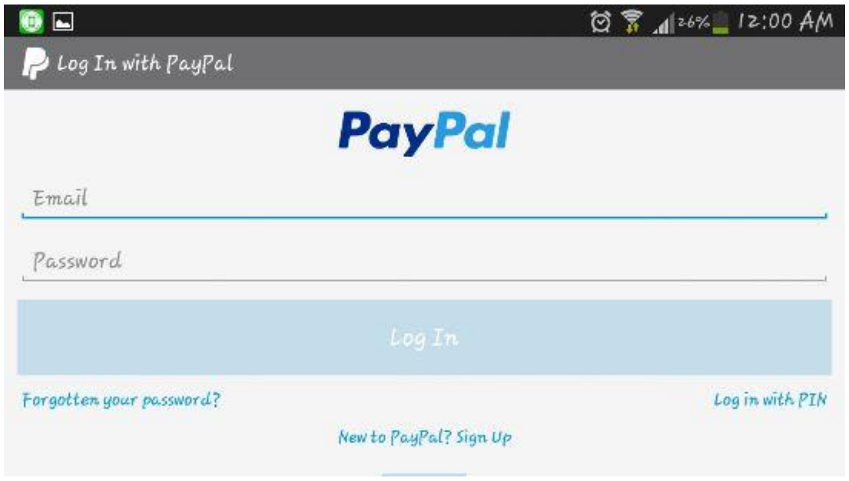


Figure 20 - PayPal Login

- Customer Payment GUI

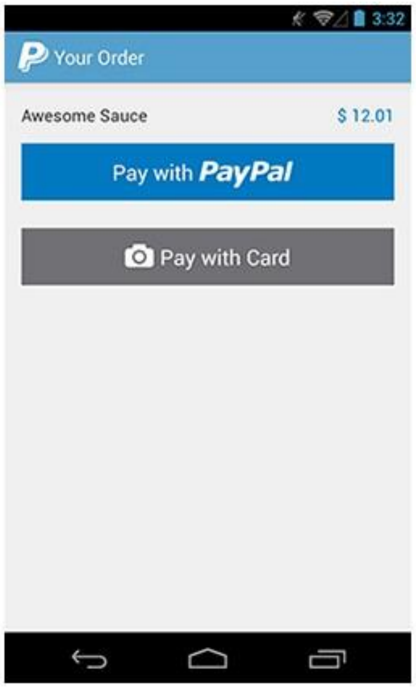


Figure 21 - PayPal Payment

The above screen shots show the GUI displayed to the customers when they are required to sign into PayPal.

- Main Menu GUI

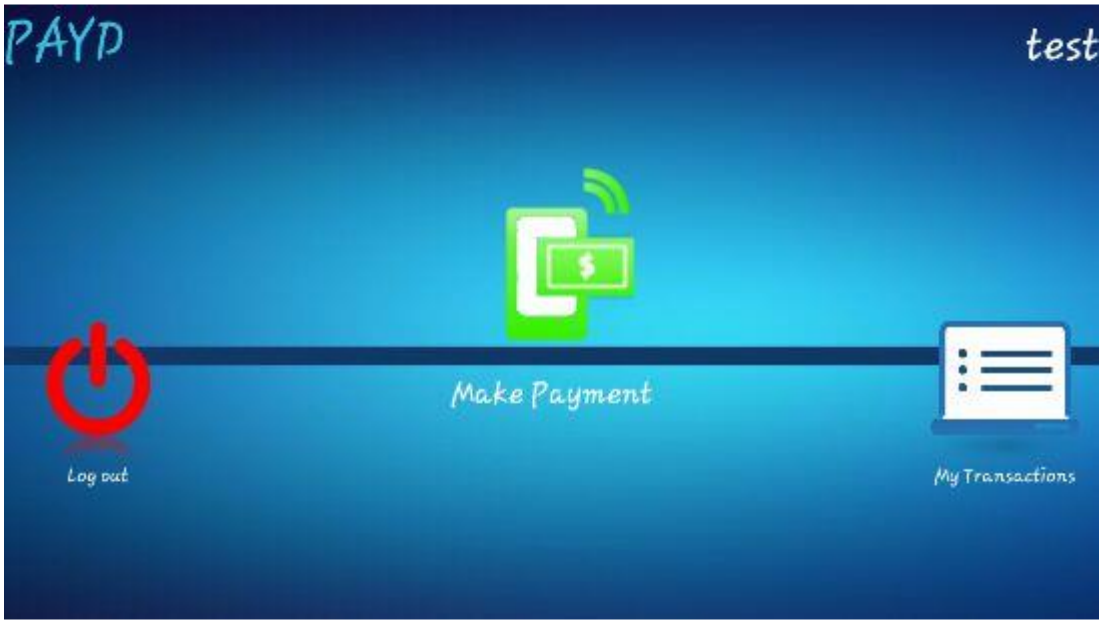


Figure 22 - Main Menu

- Fingerprint validation GUI

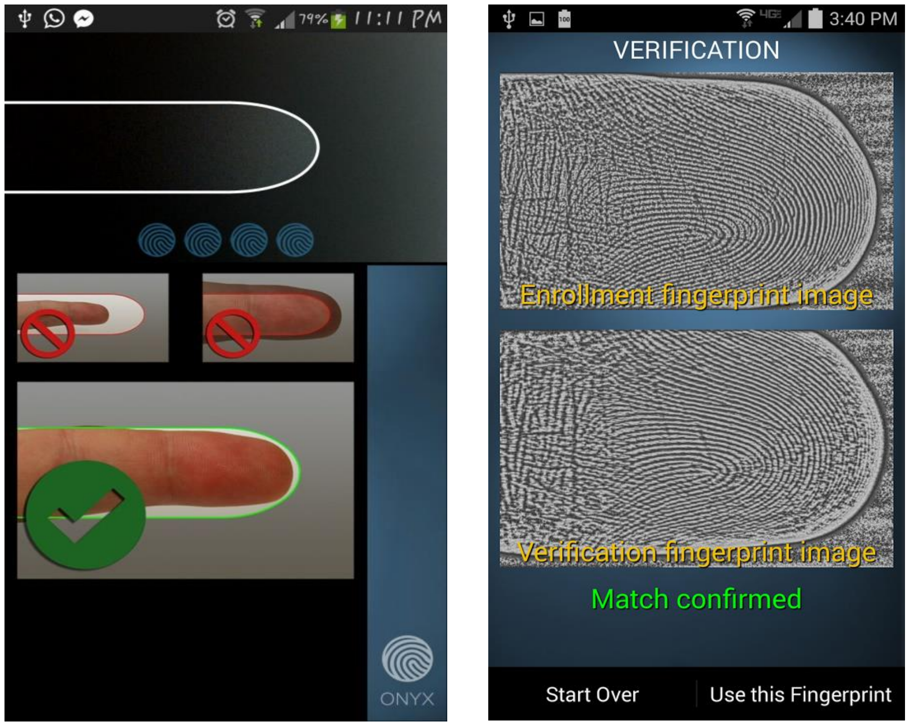
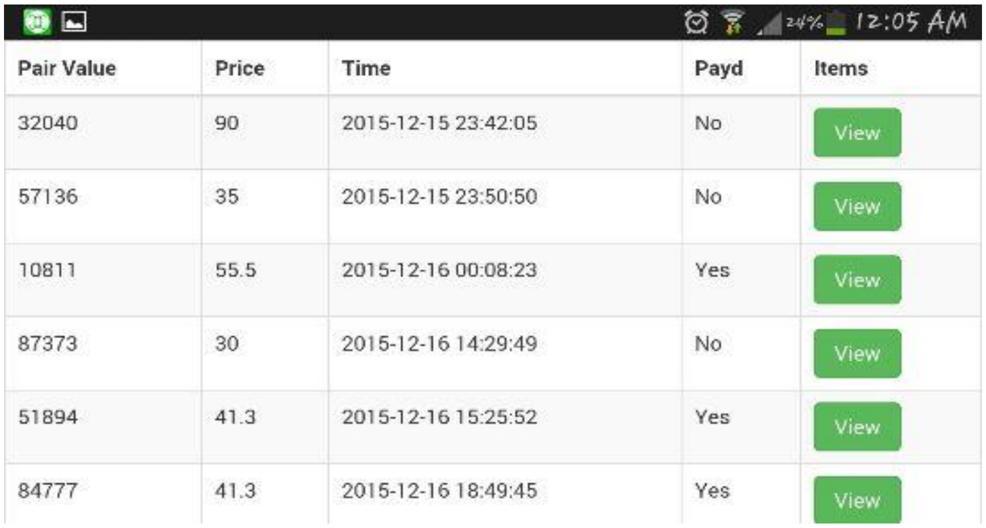


Figure 23 - Fingerprint Scanner

This is the GUI displayed by the Onyx service, it is used to collect and validate users' fingerprints.

- Transaction History



Pair Value	Price	Time	Payd	Items
32040	90	2015-12-15 23:42:05	No	View
57136	35	2015-12-15 23:50:50	No	View
10811	55.5	2015-12-16 00:08:23	Yes	View
87373	30	2015-12-16 14:29:49	No	View
51894	41.3	2015-12-16 15:25:52	Yes	View
84777	41.3	2015-12-16 18:49:45	Yes	View

Figure 24 - Transaction View

The above layout displays the user's transactions details, the contents are displayed using a webview.

- Logout

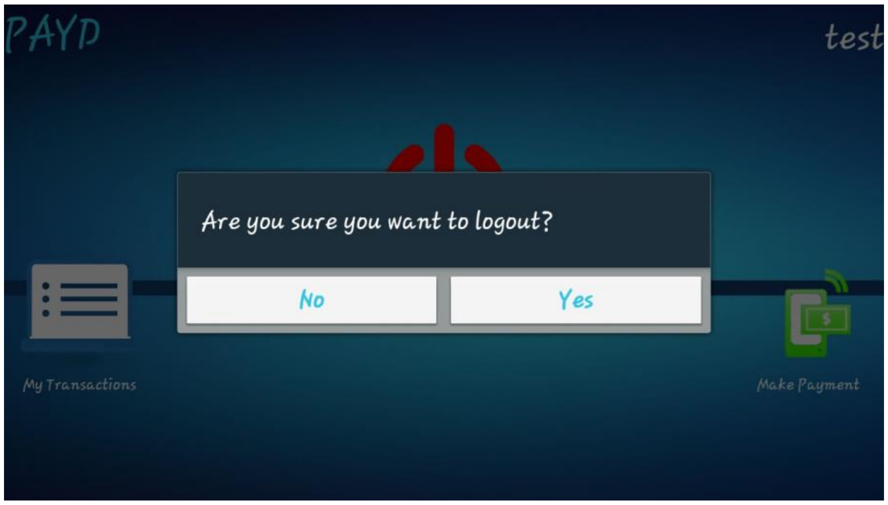


Figure 25 - Logout Confirmation

The above screen shot show an example of the application requiring confirmation from the user the user before performing permanent actions (Usability Requirement).

3 Conclusions

Implementing the different features into this project was both challenging and educational. A minor setback I encountered during the development of this project was deploying the R project I developed to an online server. The reason behind this problem was that I could not find a free hosting service. The results from the survey show that users may have to adapt to the QR technology. The results showed that people do not use QR codes frequently, this may potentially hinder customers from using the Payd payment system. Another potential limitation that was identified during the requirements gathering stage is the fact that people do not use biometric technology very often, this may deter users from using the application. However, the project's innovative features and benefits should be able to attract customers and merchants to the systems. The fact that the project implements biometric technology means it stands out from many other payment systems online payment systems. The application also acts as an interactive alternative to loyalty/club cards allowing customers and merchants to build relationships. With the new release of Fingerprint Authentication on Android 6.0, the Payd project could be a forerunner of biometrics in mobile applications.

4 Further development or research

Apriori algorithm

The Payd system collects and stores all the details of transactions made using the system. The Apriori algorithm was used to process the information stored by the system. The Apriori Algorithm is an influential algorithm for mining frequent item-sets for Boolean association rules. The Eclat algorithm is another algorithm that can be used to identify associations within a dataset. I chose the Apriori algorithm over the Eclat algorithm because although the Apriori is slower, it works well with large datasets while Eclat works well with small and medium sized datasets.

The Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

5 Bibliography

JASON ABBRUZZESE. (2015). *Android Pay has finally arrived.* Available: http://mashable.com/2015/05/28/android-pay-google-io/#TokKa0vFM5q5. Last accessed 28th Sept 2015.

Burke, E. (2014) *Irish e-commerce revenue increases by 51pc - companies | siliconrepublic.Com - Ireland's technology news service*. Available at: https://www.siliconrepublic.com/companies/2014/08/22/irish-e-commerce-revenue-increases-by-51pc (Accessed: 27 January 2016).

Retailers lagging in m-commerce stakes (2011) Available at: https://www.siliconrepublic.com/companies/2011/09/23/retailers-lagging-in-m-commerce-stakes-survey (Accessed: 27 January 2016).

Biometrics meets e-commerce (2003) Available at: http://www.bloomberg.com/bw/stories/2003-06-19/biometrics-meets-e-commerce (Accessed: 27 January 2016).

(2011) Available at: http://www.irishecommercesurvey.com/wp-content/uploads/2011/04/Irish-E-Commerce-Survey-Report.pdf (Accessed: 27 January 2016).

Ecommerce growth statistics - UK and worldwide (2015) Available at: http://www.smartinsights.com/digital-marketing-strategy/online-retail-sales-growth/ (Accessed: 27 January 2016).

Compatibility test suite (no date) Available at: https://source.android.com/compatibility/cts/ (Accessed: 4 May 2016).

Testing support library (no date) Available at: http://developer.android.com/tools/testing-support-library/index.html#AndroidJUnitRunner (Accessed: 4 May 2016).

Jakob Nielsen (1997). Questionnaire. Available: https://www.nngroup.com/articles/questionnaire-part-1/. Last accessed 5th May 2016.

Oracle. (2010). *Apriori.* Available: https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/algo_apriori.htm. Last accessed 2nd Apr 2016.

Jakob Nielsen (1993). *Usability Engineering*. San Francisco: Morgan Kaufmann. Chapter 6.

6 Appendix

6.1 Survey

Payd requirements gathering survey:

The purpose of this survey is to help me better understand the factors surrounding eCommerce and mCommerce in Ireland to evaluate how the Payd project can make an impact.

https://www.surveymonkey.com/results/SM-CCKHXDMQ/

6.2 Requirements Specification

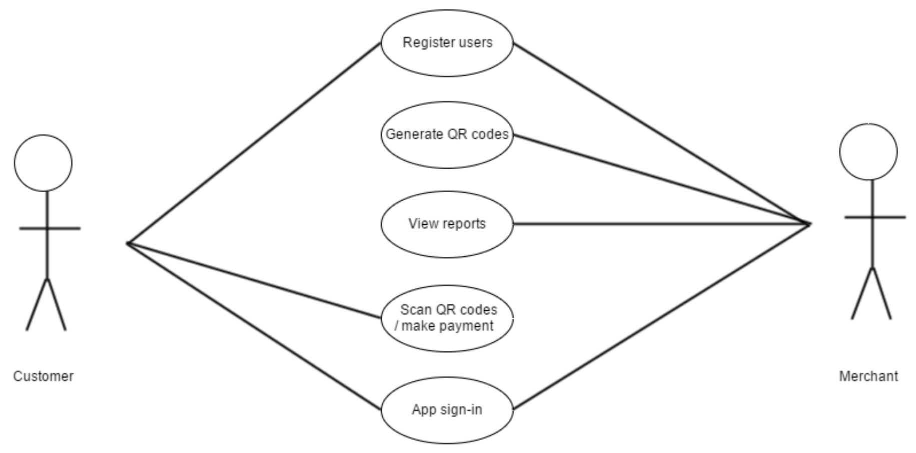
The requirements for this project were gathered using information retrieved from an online survey.

6.2.1 Functional requirements

- System should allow users to view their information and transaction history.
- The system should allow users to make or receive online payments.
- The application must be able to instantly scan QR codes.
- The system should be able to generate unique QR codes to process payments.
- The application must allow users to sign-in or register and store the details on the device.
- The application should be able to store and validate customers fingerprint.
- The system should allow merchants to view periodical reports on their customers and transactions.
- The system should use PayPal services to process payments.
- The Payd application should be usable with android devices.
- Every QR code that is generated by the application must be unique.

- QR codes must be invalidated once they have been scanned.
- Customers' fingerprint must be requested before an online payment is completed.
- Users should be able to understand and use the app properly after their first in-store or online transaction.
- Merchants and Customers should be able to view transaction details immediately the transaction is completed.
- The payment process during a transaction should not take more than 15 seconds.

6.2.1.1 *Use Case Diagram*



6.2.1.2 *Requirement 1: System should allow merchant registration*

6.2.1.2.1 Description & Priority

Merchant registration is required to allow merchant to setup a Payd account and save their store details. Merchants must have an account in order for them to use the system.

6.2.1.2.2 Use Case

Register merchant

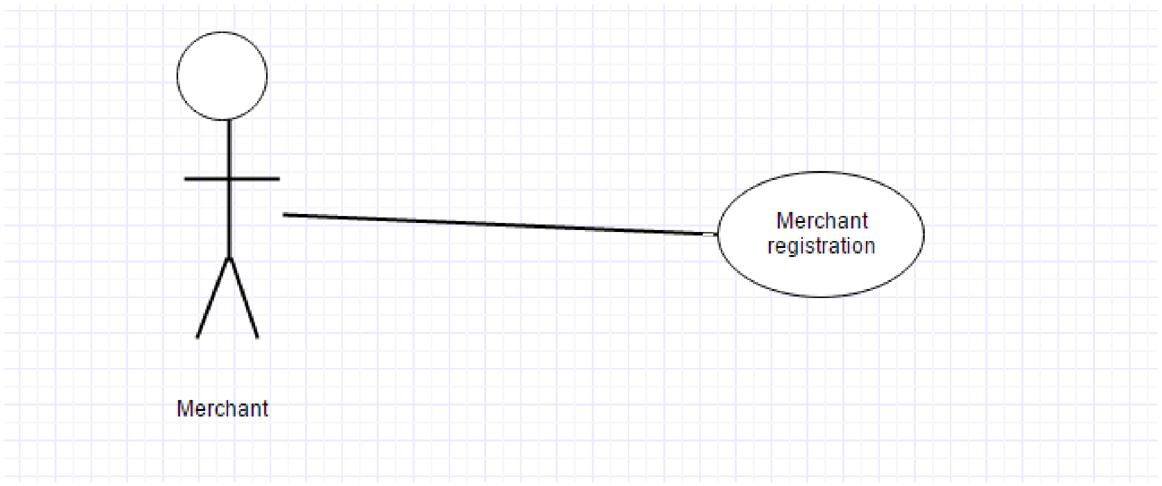
Scope

This use case demonstrates how merchants to register their store details on the Payd system.

Description

This use case describes the processes involved in registering a merchant on to the Payd database.

Use Case Diagram



Flow Description

Precondition

The merchant has not registered with the intended email address before.

Activation

This use case starts when the merchant fills in the provided registration form and clicks the submit button.

Main flow

1. The system validates the information in the form.
2. The system checks to verify that the email address does not already exist in the database.

3. The system saves all the information from the form into the Payd database.
4. The system sends a confirmation link to email address the merchant submitted.
5. When the merchant clicks the link, the system activates the account.

Alternate flow

A1: Invalid Information

1. The form warns the merchant that they have submitted wrong information.
2. The system prevents the merchant from submitting until all the required details are valid.

A2: Email Address Already Exists

1. The system notifies the merchant the email address they have submitted already exists in the database.
2. The merchant is advised to register using another email address

A3: Confirmation Link Not Clicked

1. The account remains de-activated and merchant cannot use the account.

Exceptional flow

E1: Server/ Database error

1. The system alerts the merchant of the error and advises user to try again at another time.

Termination

The system activates the merchant's account.

Post condition

The system logs the merchant into their account.

6.2.1.3 Requirement 2: System should allow customer registration via PayPal

6.2.1.3.1 Description & Priority

Customer registration is required to allow customers to setup a Payd account. Every customer must have an account in order to use the system.

6.2.1.3.2 Use Case

Register customer

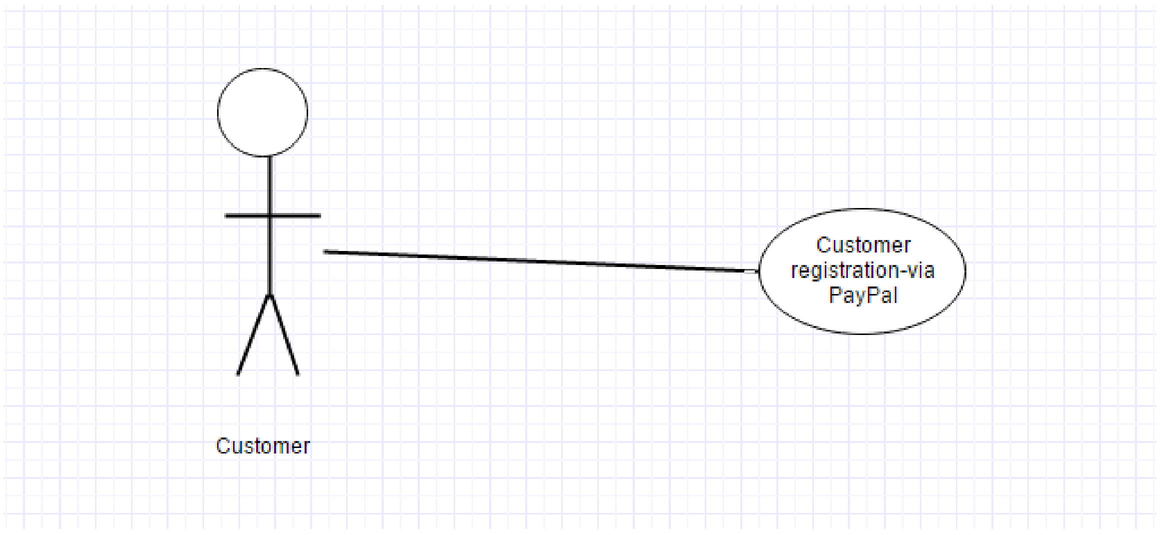
Scope

The use case shows how customers to register their details on the Payd system.

Description

This use case describes the processes involved in registering a customer on to the Payd database.

Use Case Diagram



Flow Description

Precondition

The customer must have a PayPal account.

Activation

This use case starts when the customer selects the option to sign in using PayPal

Main flow

1. The application uses the PayPal Android API to display a PayPal login interface.
2. The customer submits their credentials and are prompted to grant the Payd system access to their PayPal information.
3. The customer details are returned to the Payd application and is saved on the database and on the device.
4. The application then uses the ONYX fingerprint API to collect 8 instances of one of the customer's fingerprint.

Alternate flow

A1: Invalid Information returned

1. The application alerts the customer that there is a problem with the retrieved information.

A2: PayPal permission not granted

1. The application alerts the customer that they need to grant the Payd application permission to access their information.

Exceptional flow

E1: Server/ Database error

1. The system alerts the customer of the error and advises user to try again at another time.

Termination

The system activates the customer account.

Post condition

The system logs the customer into their account.

6.2.1.4 Requirement 3: Apps should be able to generate and scan QR codes

6.2.1.4.1 Description & Priority

It is important that the android application can generate and scan QR codes, this is how payments are made from a customer to a merchant.

6.2.1.4.2 Use Case

Generate and scan QR code

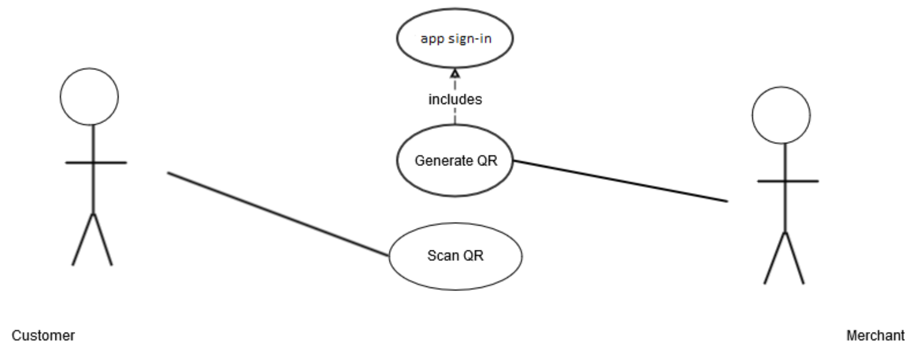
Scope

This use case shows how customers make payments to merchants using QR codes.

Description

This use case describes how a customer makes a payment and how a merchant can receive the payment.

Use Case Diagram



Flow Description

Precondition

The customer and merchant must be in the process of completing a transaction.

Activation

The use case is initialised when a merchant selects the option to accept a payment on the merchant's application.

Main flow

1. The application displays a screen prompting the merchant to input the amount they wish to request for the transaction.
2. The app generates a unique QR code containing the merchant's PayPal information and displays it on the screen for a limited time.
3. The customer selects the option to accept a payment on their app, a QR scanner will then be displayed.
4. The customer scans the QR code displayed on the merchant's device.
5. When the QR has been recognized by the customer's app the information is sent to the Payd database. The application then displays the PayPal login interface which informs them of the transaction price.
6. After the customer logs into their PayPal account and accept the payment the device sends another message to the Payd database to acknowledge the payment.

Alternate flow

A1: Incorrect credentials

1. The PayPal interface alerts the customer that the credentials they have submitted is incorrect.
2. The may cancel the payment process or re-enter their credentials.

A2: PayPal Rejection

1. If the customer does not have enough funds in their PayPal account the payment request will be rejected.
2. The customer is notified of the problem and the payment process is cancelled.

A3: Invalid QR

1. If the payment information on the QR is invalid or has been used before the customer is informed.

Exceptional flow

E1: Internet Failure

1. The app alerts the Merchant that their device cannot access the internet and they are prompted to retry the payment process after the problem has been fixed. The Payd payment process is cancelled and the use case ends.

E1: Semantics Error

1. If the data passed from the merchant's app to the customer's app contains semantics errors the customer is prompted to repeat the QR scan or request a new QR.

Termination

The merchant receives a notification that the payment has be successful.

Post condition

Both the customer and merchant can view details of the transaction.

6.2.1.5 *Requirement 4: Merchant’s websites should be able generate Payd QR code.*

6.2.1.5.1 Description & Priority

Merchant websites should be able generate QR codes, this is how ecommerce payment will be made.

6.2.1.5.2 Use Case

Generate website QR code

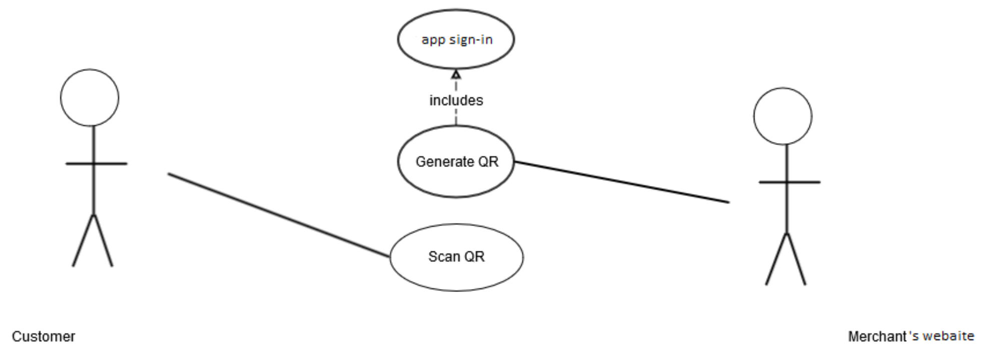
Scope

This use case shows how customers make ecommerce payments to merchants using QR codes.

Description

After an online transaction the merchant’s website should display a QR allowing Payd customers to scan and make the payment.

Use Case Diagram



Flow Description

Precondition

The customer must be in the process of completing an online transaction.

Activation

The use case is initialised when a customer selects to make a payment on the merchant website.

Main flow

1. The merchant's website uses the Payd service to generate a QR code which is displayed for customers to scan.
2. The customer selects the option to accept a payment on their app, a QR scanner will then be displayed.
3. The customer scans the QR code displayed on the merchant's device.
4. When the QR has been recognized by the customer's app the information is sent to the Payd database. The application then displays the PayPal login interface which informs them of the transaction price.
5. The application then uses the Onyx service to request and validate the customers fingerprint.
6. After the customer logs into their PayPal account and accept the payment the device sends another message to the Payd database to acknowledge the payment.
   (The merchant may choose to have separate functionality to save other transaction information.)

Alternate flow

A1: Incorrect credentials

1. The PayPal interface alerts the customer that the credentials they have submitted is incorrect.
2. The may cancel the payment process or re-enter their credentials.

A2: PayPal Rejection

1. If the customer does not have enough funds in their PayPal account the payment request will be rejected.
2. The customer is notified of the problem and the payment process is cancelled.

A3: Invalid QR

1. If the payment information on the QR is invalid or has been used before the customer is informed.

Exceptional flow

E1: Internet Failure

2. The app alerts the Merchant that their device cannot access the internet and they are prompted to retry the payment process after the problem has been fixed. The Payd payment process is cancelled and the use case ends.

E1: Semantics Error

2. If the data passed from the merchant's app to the customer's app contains semantics errors the customer is prompted to repeat the QR scan or request a new QR.

Termination

The merchant receives a notification that the payment has be successful.

Post condition

Both the customer and merchant can view details of the transaction.

6.2.1.6 *Requirement 5: The apps must allow users to sign in to their android.*

6.2.1.6.1 Description & Priority

For users to be able make and receive payments through the Payd system they must be able to login to their Payd account on any android device.

6.2.1.6.2 Use Case

Payd android sign-in

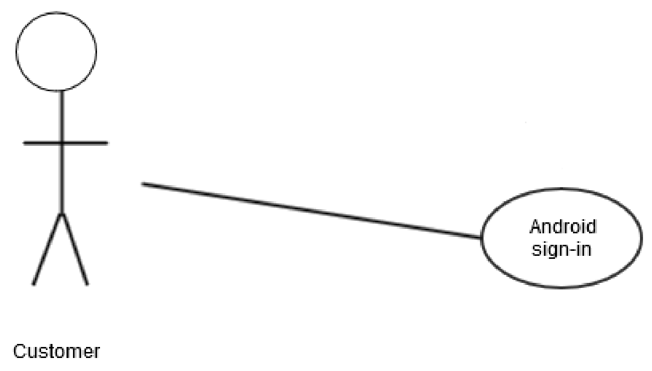
Scope

This use case shows how users sign into their Payd account on their android device.

Description

This use case describes how users sign into their account using the android mobile devices. Users may sign into more than one device at a time.

Use Case Diagram



Flow Description

Precondition

The Payd app is installed on the android device and has no account assigned to it.

Activation

This use case starts when a merchant completes and submits the Payd login form or a customer selects the option to sign-in using PayPal.

Main flow

1. The app sends the login credentials to be verified on the server or uses the PayPal API login interface to request the user's information.
2. If the credentials are correct the user's information is returned as a response from the server.
3. If the users is a customer the application then uses the Onyx service to request and validate the customers fingerprint.
4. The user's information is then stored on the device and a personalized account is created on the android device.

Alternate flow

A1: Invalid credentials

1. The user is alerted that the credentials they have entered are incorrect.
2. The use case ends at this point.

Exceptional flow

E1: Database Error / Malfunction

1. The user is alerted of the database problems and asked to try at another time. The use case ends at this point.

Termination

The application saves the retrieved information on the android device.

Post condition

The application allows users to either accept or make in-store or online payments. The application also allows the user to view the Payd account details.

6.2.1.7 Requirement 6: Merchants should be able to view transaction statistics

6.2.1.7.1 Description & Priority

The purpose of this function is allow a merchant to view a report or analysis of their transactions, the items they've sold and their customers.

6.2.1.7.2 Use Case

Transaction report

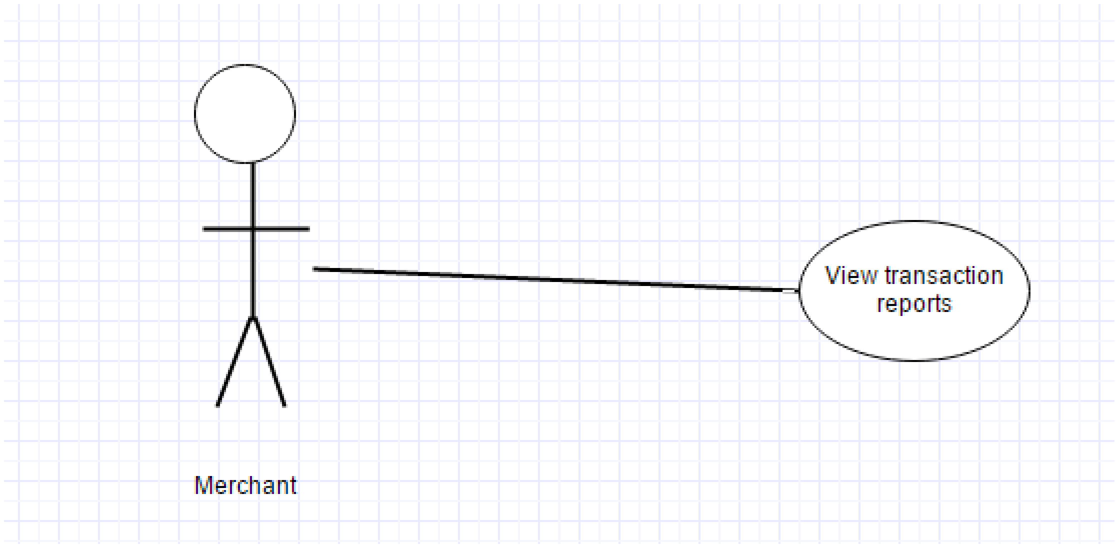
Scope

This use case describes how a merchant can view a report of their performance.

Description

After merchant have made multiple transactions on the Payd system, the application uses the collection of data to provide merchant with weekly/monthly reports.

Use Case Diagram



Flow Description

Precondition

The Merchant should have completed at least 10 transactions using the Payd payment system.

Activation

This use case starts when the merchant selects the option to view transaction reports.

Main flow

1. The system arranges all the customers into groups based on their age, the type of products they have bought in the store and other factors.
2. The merchant can then select time interval they want the report to show.

Exceptional flow

E1: Database Error / Malfunction

1. The User is alerted of the database problems and asked to try at another time. The use case ends at this point.

Termination

The merchant is able to view the request repost.

Post condition

The customer can make decisions based on the details reports generate

6.3 REFLECTIVE JOURNALS

6.3.1 SEPTEMBER

My Achievements

So far I have been able to put together the required proposal for my project. I've been able to decide exactly what I want my final year project to be based on, mobile payment. My objective is to develop an in-store mobile payment system. The system will use QR codes technologies to send payment information and will allow users to transfer funds. My project will also implement elements of data analysis. I've been able to create dataflow and use-case diagrams to help me understand what approach I need to take in the development of the project. I have put together a list of all the resources, languages and APIs I will need in the development of this project.

My Reflection

Now that I have found a library that implements QR Codes I feel a lot more confident in my project idea. I feel I have gathered the two core elements that will make up this project, QR implementation and a connection to the server.

Intended Changes

I intend to do some more research on if and how the Zxing library can contribute to my project. The Zxing library provides QR functionalities for Android developers. I will also do some research in to Android and database communication. This aspect of the project is essential in order to synchronize information between the database and android devices.

6.3.2 *OCTOBER*

My Achievements

I was able to find out a lot about QR Codes and how they work. I did a lot of research on how and when QR codes can be implemented. From my research I was able to find one of the Java libraries that provide resources for implementing QR codes on android, the library is called Zxing. I was also able to find samples projects which helped me understand how to utilize the Zxing library. The sample projects I found contained functionalities to scan/decode QR codes and to generate QR codes. I modified the sample projects so I could get a sense of how and where they will be useful in my project.

To make sure the Zxing library did exactly what it was intended for I decided to carry out some small test. I wanted to check and make sure that the Zxing library could generate and decode all the characters on a QWERTY keyboard. Using the sample projects, I created a string that contained every character on the QWERTY keyboard and then encoded it. After the QR was generated I decoded it and checked if the input was the same as the output. The input was the same as the output, this means the project encoded and decoded the string correctly.

I've also been doing some research on how android devices maybe connected to a database. I found that there is a Java class called HttpURLConnection. This method takes a URL as an input and then returns the content of the URL. Using this method I am able to post to a PHP file which will save and return information from the database.

My Reflection

Now that I have found a library that implements QR Codes I feel a lot more confident in my project idea. I feel I have gathered the two core elements that will make up this project, QR implementation and a connection to the server.

Intended Changes

I intend to begin working on my actual project and to begin implementing the things I have been working on from sample projects.

6.3.3 NOVEMBER

My Achievements

Using the ZXING library I have been able send messages from one Android device to another. To make sure the information displayed by the QR cannot be read by unauthorized users or apps I created a Java class to cipher and decipher payments and users information. The class replaces each letter of the alphabet with a specific string; the class can also reverse the action allowing the information to be deciphered. I also created PHP methods that carry out the same functions as the Java class. This way all the payment information being sent to the Payd server can be ciphered by the devices and will only be deciphered by the server.

I have also begun to work on the Payd website which will allow users to top-up their Payd balance. From my web development experience I have found bootstrap to be very useful and easy to work with. I have viewed and sampled a number of bootstrap templates and have chosen the one I feel will suit the functionalities that I will be implementing on the website. I have re-arranged some elements on the webpages and redesigned others completely in preparation for the PHP functions I will be creating.

My Reflection

I feel that the project is slowly coming together. The more task I complete for the system the more I understand how I can implement more features. I feel the next steps in this project will be much easier than previous steps as I now have more of an understanding of what I am doing.

Intended Changes

I intend to find or create a method that generates random numbers that will never be repeated. This function is crucial within my project as it will ensure the integrity of the system and will prevent false transaction information being sent to the Payd server. I also intend to create login and registration functionalities on the website.

6.3.4 *DECEMBER*

My Achievements

After a meeting with my project supervisor I decided to change some of the logic behind the project. The original plan was that the Payd system would collect and send funds on behalf of merchants and customers. It would have essentially acted as a bank, after reviewing this with my supervisor I decided that it would be much better to use an existing payment system to transfer money between merchants and customers. Using an existing payment system would ensure user confidentiality, integrity and security. I spent some time doing some research on various payment systems that could be used within android applications. One of the payment systems I came across was Bit-Coin. I downloaded and tested some sample application to help me understand how the payment process was conducted; I was able to find how it could be integrated with my project. After further research and requirements gathering I found that Bit-Coin had a very bad reputation which would be a huge disadvantage to my project. After much considerations I decided to use PayPal's android API. Using the PayPal API meant that I would have to make changes to the way users made payments. Initially during a payment the customer would generate a QR using their Payd app and the merchant would scan the generated QR, now it is the merchant that generates the QR while the customer scans. The merchant's PayPal details will be contained within the QR, when the customer scans it they will be prompted to sign into PayPal and accept the payment. Since I was already using the PayPal's API I decided to use their service to retrieve customer's information so that customers will no longer need to manually submit their details.

My Reflection

I’ve had to re-structure my project to facilitate the use of PayPal. Although I was reluctant to do so before I now feel the system is a lot more secure and capable of accomplishing its objectives and fulfilling the user and functional requirements.

Intended Changes

I hope to implement even more security features within the project to make customers and merchants even more confident and secure when using the system.

Supervisor Meetings

Date of Meeting: 14 / 12 / 2015

Items discussed:

Using existing payment system

System security

Data mining options

6.3.5 JANUARY

My Achievements

Over the course of the month I put together a list of questions about payments to help me better understand the users’ requirement. I carefully phrased and selected these questions and put them into a survey. The survey was sent to as many people as I could send it to. I used Survey-monkey to create the survey. I intend to leave the survey for a period of about two months; I feel it has really helped me in my requirements gathering for this project. The survey has also helped me to understand customer views on QR, Biometrics and PayPal.

From the research I’ve been doing around my project’s domain I discovered that there was one key problem for online merchants. Online merchants are unable to verify that the person making a transaction is who they claim to be, passwords and PIN can be stolen.

This problem has discouraged and caused a lot of troubles for merchant trying to move into ecommerce. After consulting with my project supervisor I decided to add a new aspect to my project. The project was incremented so that merchants are now able to use the Payd system to generate QR codes on their website. This new feature means that the system can cater for both online and in-store payments. To tackle the problem I identified during my research I decided to add the Onyx PassPrint API. This API collects stores and validates fingerprints using the android device. When customers are making online transactions they will be required to authenticate with both their PayPal credentials and their fingerprint. This new implementation means that merchants can now accept payment confident that the user making the transaction is the real customer.

My Reflection

Although it's been quite hard implementing the new features I mentioned into the system I feel it has been worth the effort. I think the implementation of the fingerprint scanner makes my project more secure and innovative.

Intended Changes

I aim develop the merchant's side of the project a bit more. I also intend to start looking for and sampling data analytic tools. These tools will be very helpful in developing transaction reports for merchants.

Supervisor Meetings

Date of Meeting: 19 / 1 / 2016

Items discussed:

Requirements gathering survey

Required diagrams, Collaborative diagram etc.

6.3.6 FEBRUARY

My Achievements

Over the course of the month I was able to create the activity that would be responsible for displaying user's transaction history. I decided to use a WebView activity to display the user's content because it would be faster to design. A WebView is an android view that allows apps to display web contents. The WebView component is powered by google and is pre-installed on every android device meaning there would be very little compatibility issues. Using Bootstrap I designed a webpage that displayed all the user's transactions, the webpage requires the user's key to be stored as a cookie. After I was satisfied with the layout and contents of the webpage I loaded the webpage into the WebView. I was able to retrieve the user's primary key from the device and set it as a cookie for the webpage using the methods provided by the WebSettings and CookieManager classes. I built the WebView activity in both the merchant and customer application.

I have been able to gather and generate some sample data to use in generating reports analysis for merchants. I will be developing this part of the project using the R programming language

My Reflection

I found that creating WebView was a much faster approach to display contents because I had a lot more experience in creating content using html and CSS. I feel I saved a lot of time and effort by using the WebView approach.

Intended Changes

I intend to start gathering the resources and packages I will need to develop in R. I also intend to do a little bit of research into the algorithms and methods that will best suit what I'm trying to accomplish using the R language.

Supervisor Meetings

Date of Meeting: 6 / 02 / 2016

Items discussed:

Datamining algorithms to be implemented.

6.3.7 MARCH

My Achievements

Over the course of the month I did a lot of research on datamining tools in R, and how they could be implemented into my project.

The Payd system collects and stores all the details of transactions made using the system. I decided to use the Apriori algorithm to process the information stored by the system. The Apriori Algorithm is an influential algorithm for mining frequent item-sets for Boolean association rules. The Eclat algorithm is another algorithm that can be used to identify associations within a dataset. I chose the Apriori algorithm over the Eclat algorithm because although the Apriori is slower, it works well with large datasets while Eclat works well with small and medium sized datasets.

From my research I also found that the Apriori uses a "bottom up" approach, where frequent subsets are extended one item at a time and groups of candidates are tested against the data. The algorithm terminates when no further successful extensions are found.

I used the R language to implement the Apriori algorithm in the development of this module. I began by creating and cleansing some sample data. After making sure the data was correct and properly formatted I imported the dataset as a CSV into RStudio. The data contained information on over 8000 store transactions including the time of purchase and the items purchased. In order to provide merchants with information that will help them improve their store the system will show them different graphs that represent the information in the data. The purpose of mining the data was to answer the following questions for merchants:

What time of the day the most money is made?

What day of the week is most money made?

What items do different age-ranges buy together?

What items are most frequently bought together?

My Reflection

I was able to find a lot of helpful resources online for developing in R. Using some of the resources I found I answered all the questions I aimed to answer for the merchants. I was also able to develop a good understanding of datamining tasks and learn a new programming language at the same time.

Intended Changes

I intend to deploy what I have developed in R using a web service. Over the coming month I aim to begin the testing phase for the project.

Supervisor Meetings

Date of Meeting: 11/ 03 / 2016 & 18/ 03 / 2016

Items discussed:

- Testing Approaches
- Deployment methods

6.3.8 *APRIL*

My Achievements

Over the course of the month I encountered a minor challenge in deploying the project I developed using R to an online server. The reason behind this problem was that I could not find a free hosting service.

I did some research on Clover devices and discovered that I could use a Clover emulator to develop and test android applications that can be used on real Clover devices. The app will be able to retrieve products names, prices and other transaction information from a Clover device, making the app much easier to use in a store environment. Using resources I found online I was able to implement the Clover API on the merchant app so that merchant could select items to be bought and immediately generate a QR with all the transaction information. To make sure the application worked on real Clover devices I made attempt to get one, unfortunately I was unable to do so.

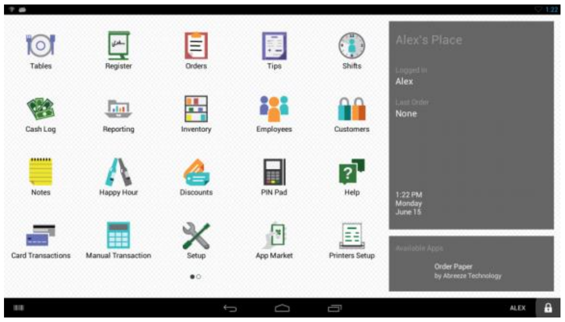
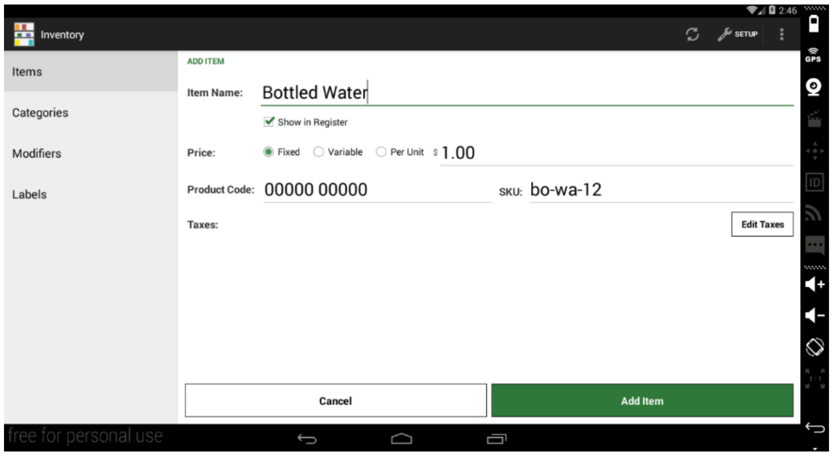


Figure 26 - Clover device layout



ADD ITEM

Item Name: Bottled Water

[✓] Show in Register

Price: [● Fixed] ○ Variable ○ Per Unit $ 1.00

Product Code: 00000 00000 SKU: bo-wa-12

Taxes:

Figure 27 Clover Retrieve item

My Reflection

Although I was unable to deploy the R project I developed it works well using a local server. I feel implementing the Clover API on the merchant application makes it more appealing and easy to use for merchants who already own Clover device.

Intended Changes

I intend to begin planning and implementing tests for the application. I will be performing System, Integration, Stress, load and usability tests.

Supervisor Meetings

Date of Meeting: 28 / 04 / 2016

Items discussed:

Types of testing and documentations required.