Final Year Project

IntelliScan

Conor Kenny

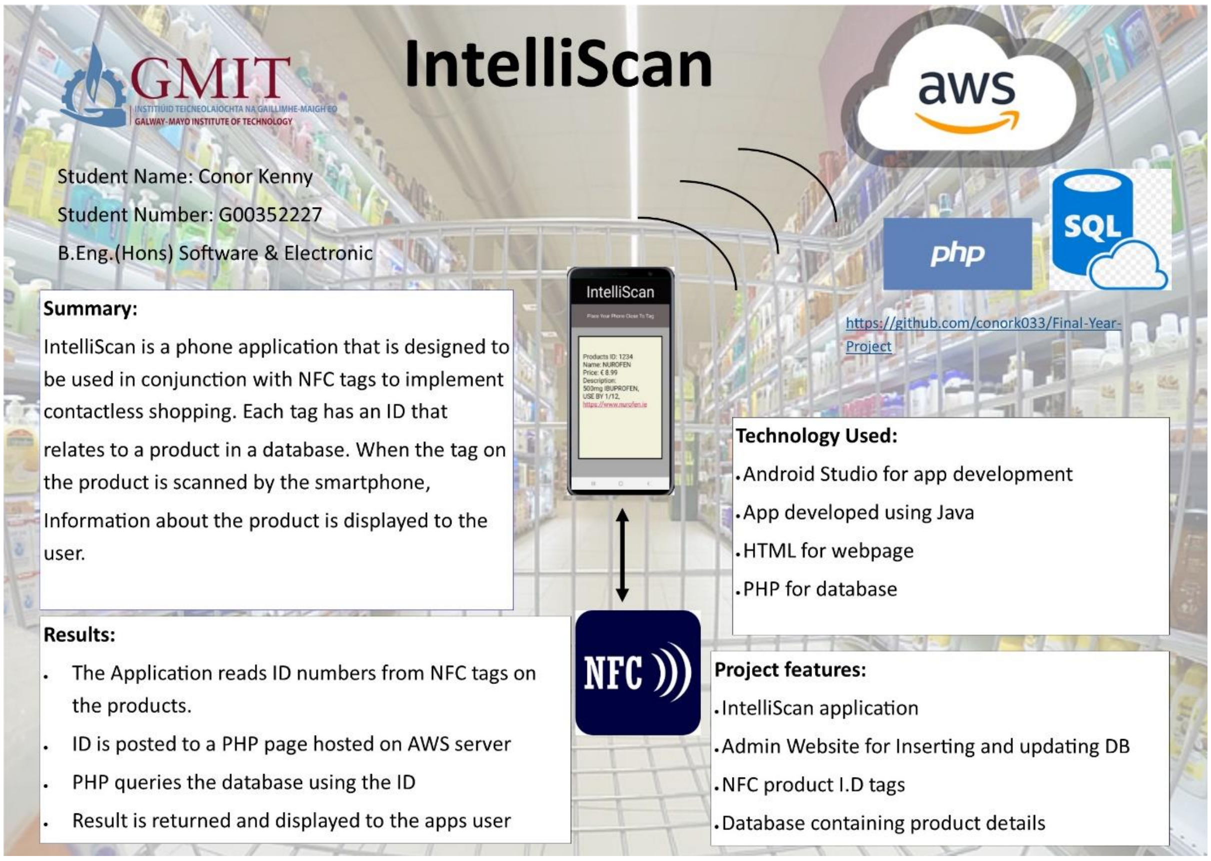
G00352227

Bachelor of Software & Electronic Engineering

Galway-Mayo Institute of Technology

2019/2020

Poster



IntelliScan

Student Name: Conor Kenny

Student Number: G00352227

B.Eng.(Hons) Software & Electronic

Summary:

IntelliScan is a phone application that is designed to be used in conjunction with NFC tags to implement contactless shopping. Each tag has an ID that relates to a product in a database. When the tag on the product is scanned by the smartphone, Information about the product is displayed to the user.

Results:

- The Application reads ID numbers from NFC tags on the products.
- ID is posted to a PHP page hosted on AWS server
- PHP queries the database using the ID
- Result is returned and displayed to the apps user

https://github.com/conork033/Final-Year-Project

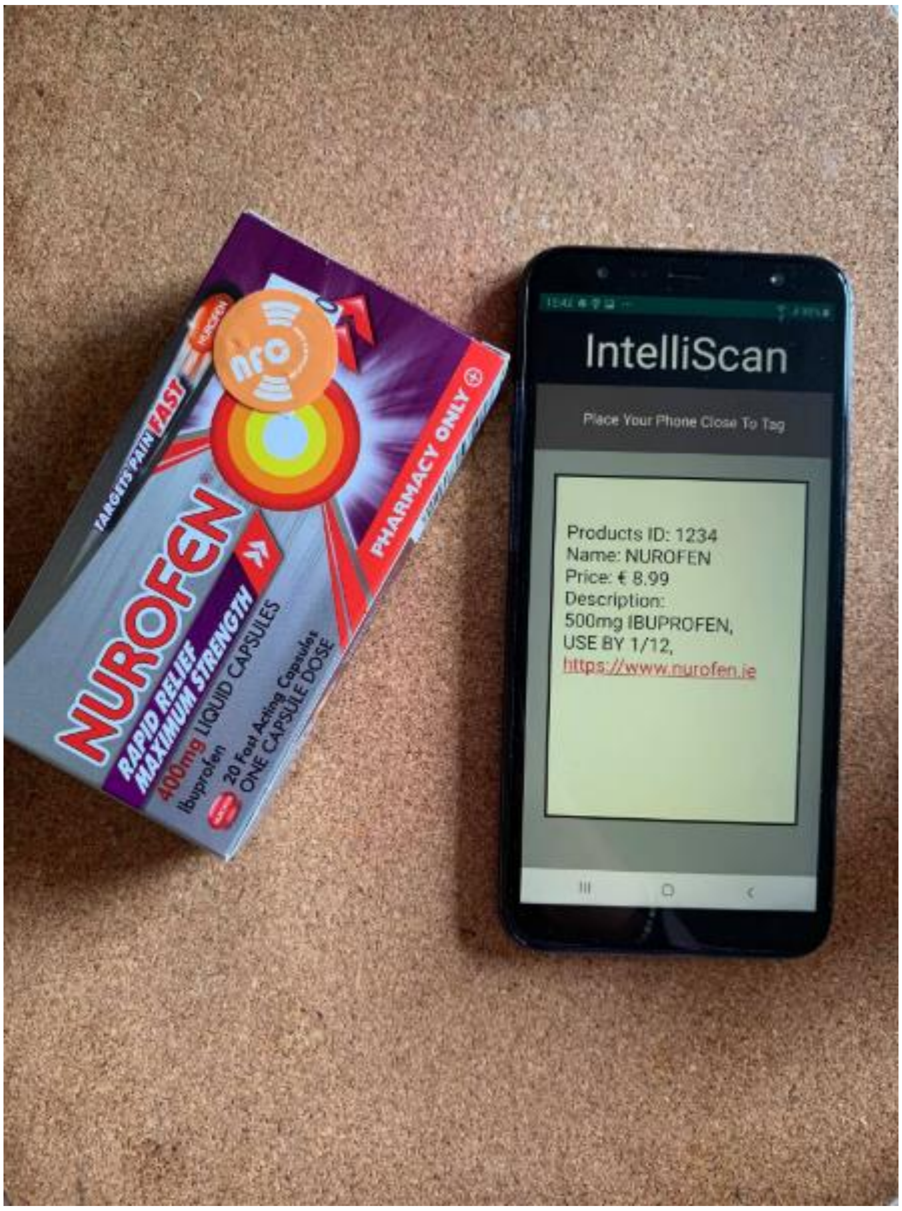
Technology Used:

- Android Studio for app development
- App developed using Java
- HTML for webpage
- PHP for database

Project features:

- IntelliScan application
- Admin Website for Inserting and updating DB
- NFC product I.D tags
- Database containing product details

Project Graphic



Products ID: 1234
Name: NUROFEN
Price: € 8.99
Description:
500mg IBUPROFEN,
USE BY 1/12,
https://www.nurofen.ie

Declaration

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering in Software & Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

Acknowledgments

I would like to thank all my lectures for their help throughout my project and in particular, I would like to my supervisor Mark Sherlock for his encouragement and guidance throughout the project.

Table of Contents

Summary

The goal of my project was to make a mobile phone application that could be used to speed up the users shopping experience in a supermarket. The scope of the application was to be able to scan each product and as you picked it up, the application would then display details about the product including its price and details. You could then choose to add these items to a virtual cart which would then total up the cost of each item. When leaving the shop, you could just pay for your items using the application.

In order to get information from the product, the smartphone application needed something on the product to interact with. Having some minor experience with using NFC tags before, I decided them to be the best option. By placing an ID on each tag that relates to a product in a database the smartphone could then get information about the product by placing it near the tag.

An additional feature of the project is two admin pages hosted on an AWS server. The functionality of the pages is to give the supermarket the ability to insert new products to their database and the option of updating existing products details such as its price or description. This eliminates the need to go around the shop and manually change the prices on each item when it needs updating. It is also useful in the case that the supermarket gets a new product.

Additional features that I had hoped to have in the project but wasn't able to implement, was to be able to save each product that was scanned on the smartphone so you could display all the products that had been scanned and display the total cost of their shop, in order to put this feature in my application I would have used SQLite to store the items. But in the process of developing this feature, I discovered that my smartphone had a factory permissions feature which wouldn't allow me to edit the phones file system meaning I couldn't create a table. So, because of this issue that was out of my control, I couldn't implement the feature.

Introduction

My final year project IntelliScan is a mobile phone application developed to make shopping an easier and faster experience for users. As someone who has a lot of experience in the retail sector due to my part time job in a supermarket, I have noticed that supermarkets are constantly looking for ways to improve their customers experience while shopping. Most recently the supermarket I work in installed an express self-service checkout for shoppers to scan their own items before leaving the store. This was done in order to reduce lines at checkouts and ultimately increase the number of customers that can be serviced at a time. The smaller the queues the smaller the wait time which in turn gives people an incentive to shop in one supermarket over another. I began wondering as to ways in which the process could be streamlined even further.

The conclusion that I came to is that there must be a way in which consumers can pay for their products without having to put all their items into a trolley bring them to a checkout, unload their trolley, scan their items and then pack their items once again. So rather than having to do this what if customers could scan their items while they shop using an App on their smartphone.

To implement my idea, I decided to build an application using android studio which scans NFC tags that have been placed on each product much like a barcode would be. The tag on each product has an I.D which relates to the product in a database. When the product is scanned by the phone the products information such as price and description is displayed on the Apps user interface. Initially I had planned for the user to be able to keep scanning items and save the scanned data on their phone and continue placing them in their cart. Customers could pay for them using their smartphone without having to unload and scan their shopping again at the checkout.

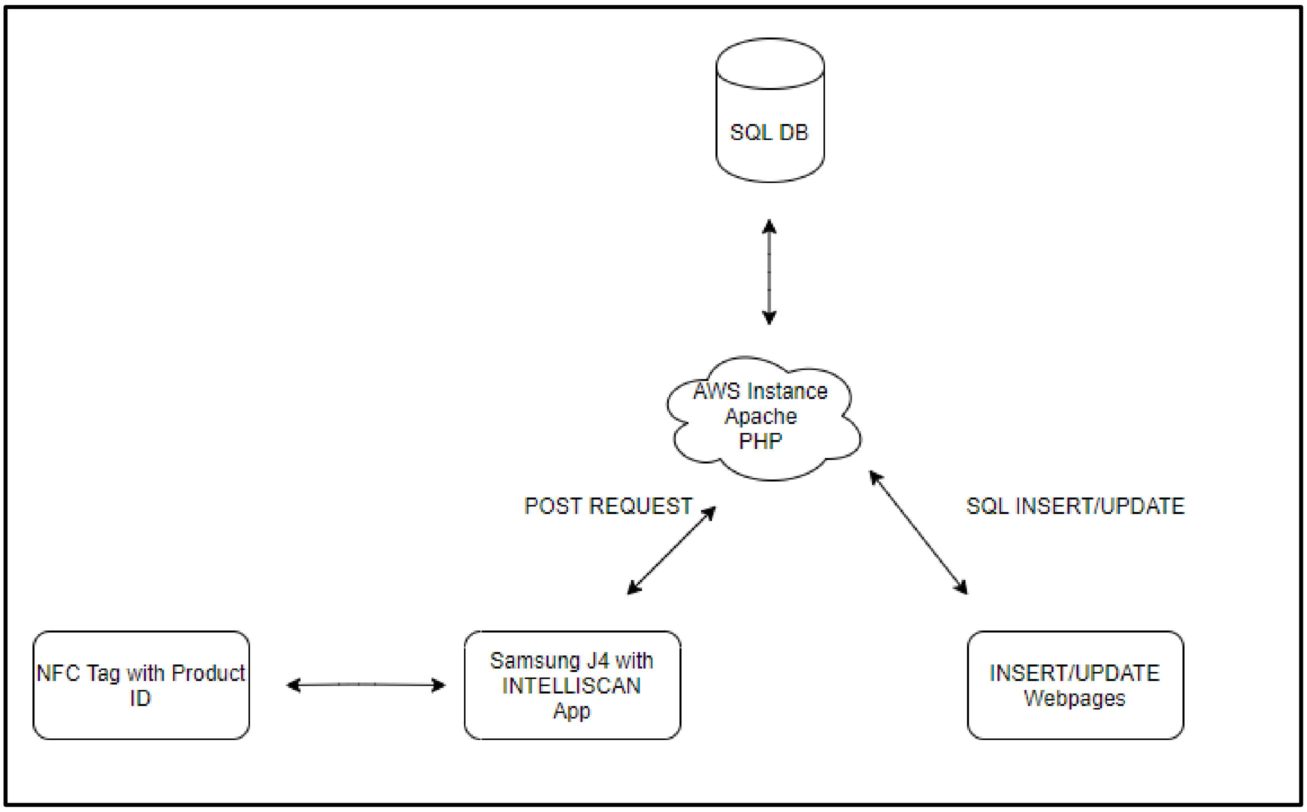
How my Application Could Be Beneficial

When I first conceived my project idea back in September, I never could've imagined how relevant contactless interactions would become in our daily lives due to the coronavirus outbreak through the world at the beginning of 2020. We now live in a world where social distancing is now a part and parcel of our daily lives, especially when we are shopping. With supermarkets implementing social distancing restrictions of 2 metres in their stores, it has provided them with a new challenge of having to reduce the number of people in their supermarkets at a time and causing queues to form outside their stores and at checkouts. This is something I have become greatly acquainted as I work in a supermarket.

I think that my application even in its current development phase could be very useful in combatting these challenges. Imagine if shoppers could find out information about a product that they are interested in purchasing without having to pick it up and read the packaging. Currently browsing through products slows down the shopping experience for others as well possibly as causing social distancing rules to be broken if more than one customer wants to look at the same product at the same time. Not to mention the possible ramifications if a person unwittingly carrying the covid-19 infection picks up an item and then decides to put it back. It is possible the next shopper to handle that item could possibly become infected as there is current evidence to say the virus stays on surfaces and possibly on products.

If a shopper was using my application, they could just scan the product to find out its details without having to pick it up and read its packaging. Further still if I had been able to implement the final features of my project of being able to save the scanned items and pay for them without having to interact with a cashier, which is also a challenge facing supermarkets with most of them installing Perspex glass in front of cashiers and their tills, it would solve a lot of problems and help shoppers get their shopping done a lot faster and reduce the time they could be possibly be exposed to infection during their shop.

Project Architecture Diagram



Architecture Explanation

Each element comes together to form the finished article. Starting with NFC tags which are stickers placed on the products, each sticker has a unique ID written onto it using a third party NFC write(*this is a feature I considered adding to my project but felt I mostly wanted to focus on the customer side when developing the application*) the ID on it.

The smartphone is NFC enabled so it has the functionality to read the tags. The tags are read, and the ID number is extracted. The extracted ID number is sent to a php page by the application using a volley post request. The php is running on a tomcat server which is hosted on an AWS instance.

The php receives the ID and performs an SQL SELECT command to find out the details of the product on that row in the MySQL database. The information is then retuned as a json object and parsed by the application. The information about the product is displayed on the applications user interface. This information includes ID number, name, price and description. The description includes details such as use by dates, ingredients and in some cases hyperlinks to the products website.

On the supermarkets admin side there are two webpages. The webpages functionalities are for adding new products and also updating existing products. This is done using two webpages written in html. The users enter details into the form on the webpage. These details are then passed to php pages which perform SQL commands.

The php page for inserting data to the MYSQL contains an SQL insert command and passes the details entered to the database. For the update php it updates the details of the ID entered with the new product details entered.

NFC Technology

What NFC is and How NFC Works

NFC stands for near field communication, there are two main types of NFC device. They are passive communication and active communication devices; active devices have a power source and can send and receive a data. Passive devices have no power sources and are powered by the active device. In my project my smartphone acts the as the active and the NFC sticker tags act as the passive device. The smartphone can also act as a passive device so if I were to implement the payment functionality in the future that could also be done using NFC.

NFC technology uses magnetic coupling to send and receive signals, when two NFC enabled devices are close enough, which is generally between 10cm and touching off each other. The electromagnetic field allows for the active device to power up the passive device and communicate with it. The active device then picks up the signal deviations from the passive tag. A detector and decoder on the active device are used to understand the signal and take the information from the signal generated. In my case the information is the ID number. Figure 1.1 demonstrates the electromagnetic signal passing from the active device to the passive device and powering it [1], [7].

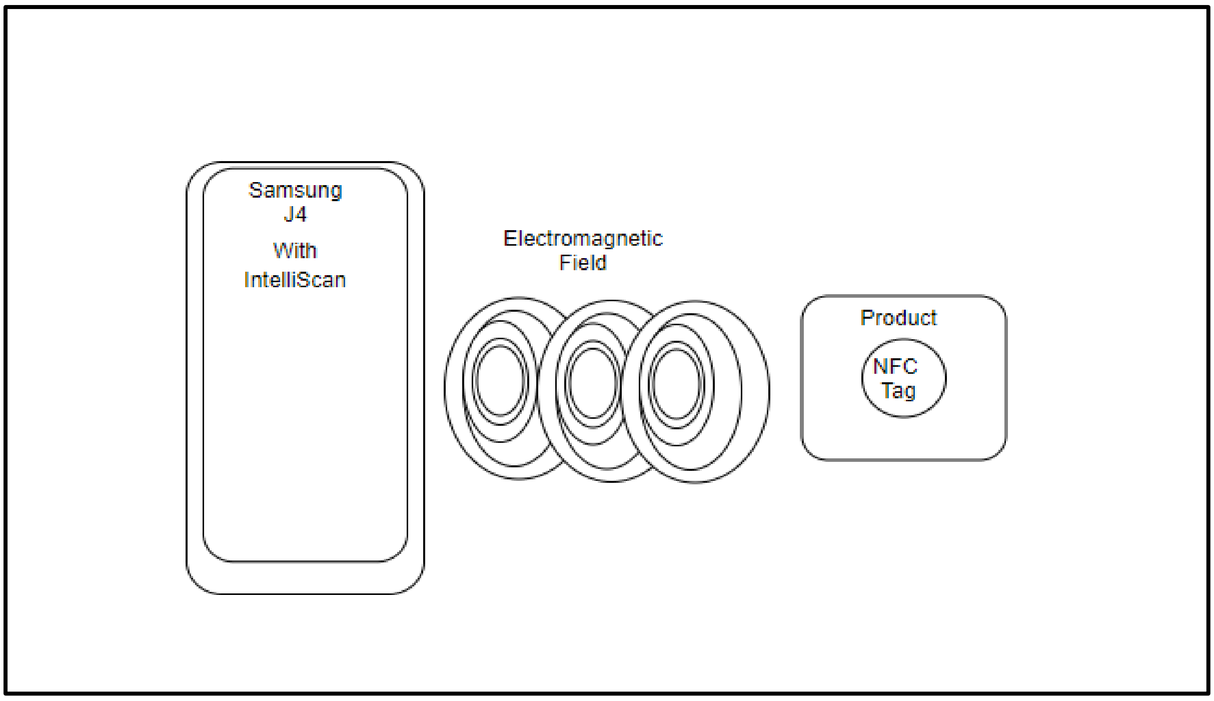


fig.1.1

NFC Sticker Internal Circuit

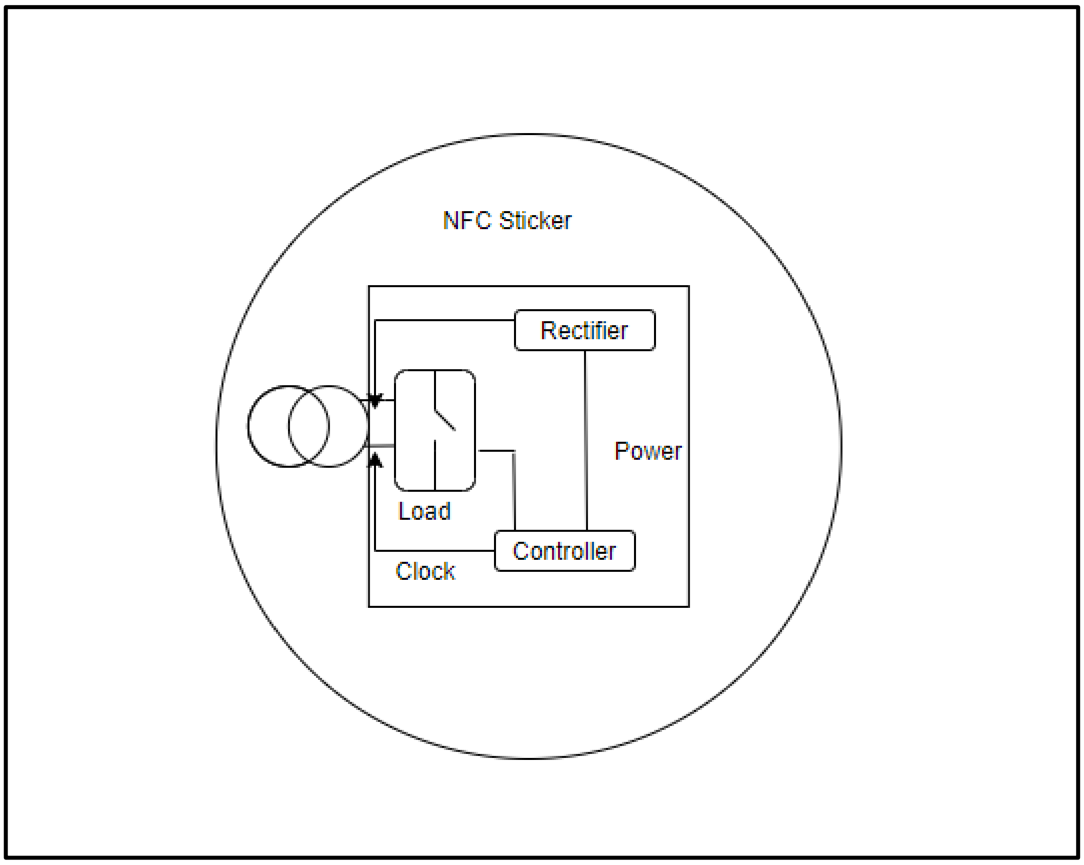


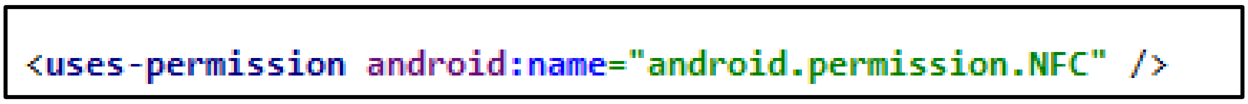
Fig 1.2

In figure 1.2 the internal structure of the NFC sticker can be seen.

The process that happens when the smartphone powers the NFC sticker is called electromagnetic induction. The smartphone emits the radio to form the electromagnetic field at a frequency of 13.56MHz, the waves generated get coupled to the antenna of the active tag and it receives power. Voltage is generated in the circuit of the tag; the voltage is rectified using the rectifier on the chip. Now that the chip within the tag is powered up, it can respond to the smartphone using load modulation. The clock within the chip is used and a frequency of 848KHz is generated by the chip. The load within the chip is then turned on and off which sends the data stored on the chip to the smartphone [1],[7].

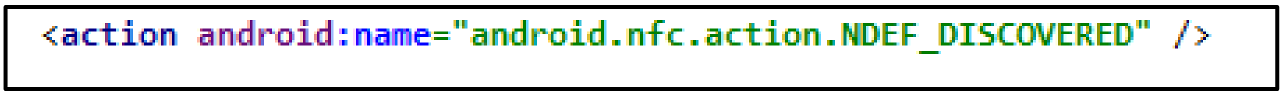
NFC Code Explanation

Firstly, in order for the application to be allowed use the smartphones NFC technology it must be given permissions in the Android studio projects manifest file, as can be seen on figure 2.1.



```
<uses-permission android:name="android.permission.NFC" />
```

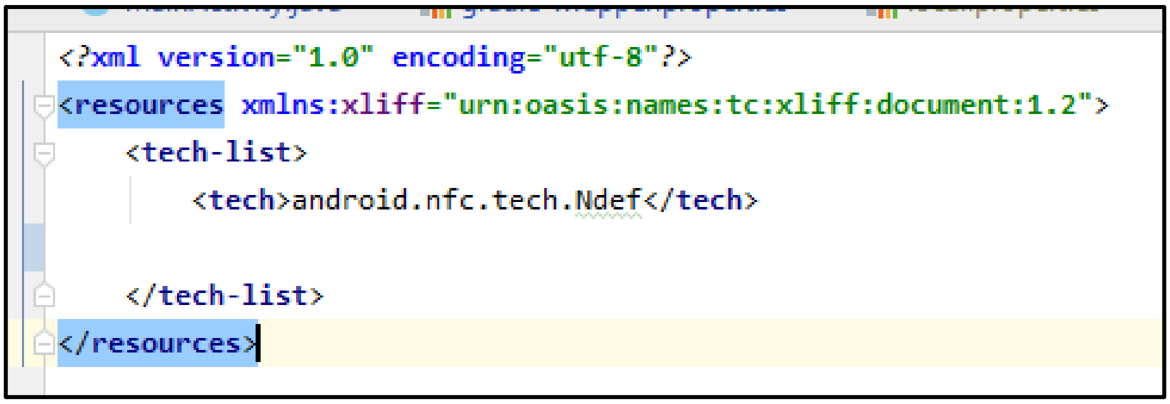
Fig 2.1



```
<action android:name="android.nfc.action.NDEF_DISCOVERED" />
```

Fig 2.2

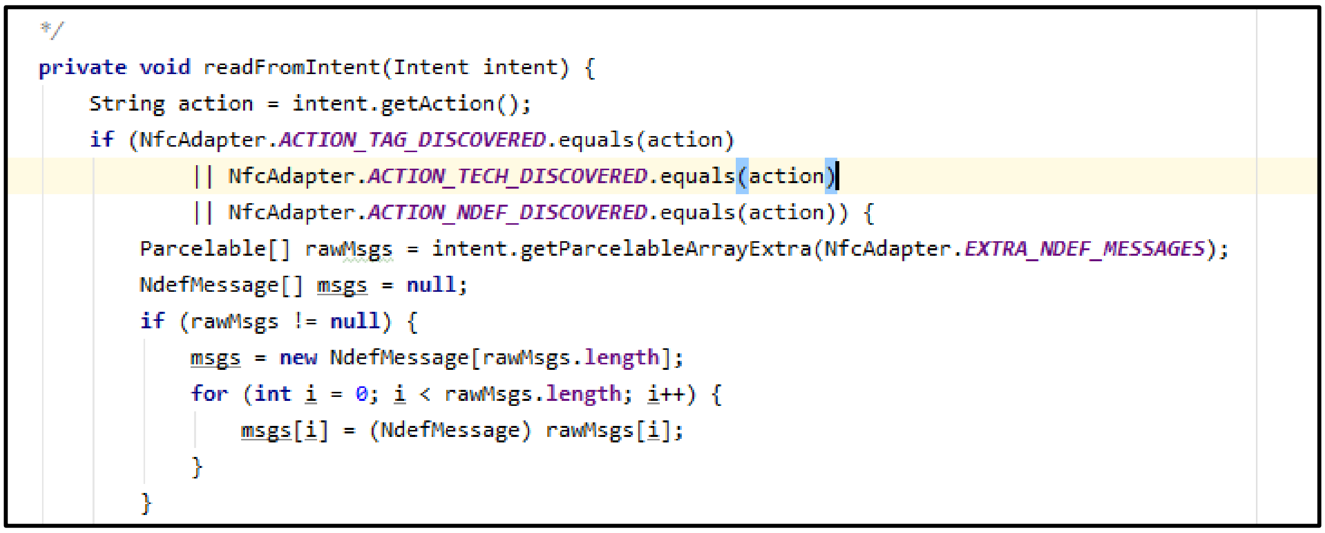
Figure 2.2 is also a piece of code from the manifest file, this piece of code is placed inside on an intent and its functionality is to start the application once it encounters NFC technology. This will cause the application to start no matter what application a user may be interacting with, once the smartphone comes into contact with an NFC tag.



```xml
<?xml version="1.0" encoding="utf-8"?>
<resources xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
    <tech-list>
        <tech>android.nfc.tech.Ndef</tech>

    </tech-list>
</resources>
```

Fig 2.3

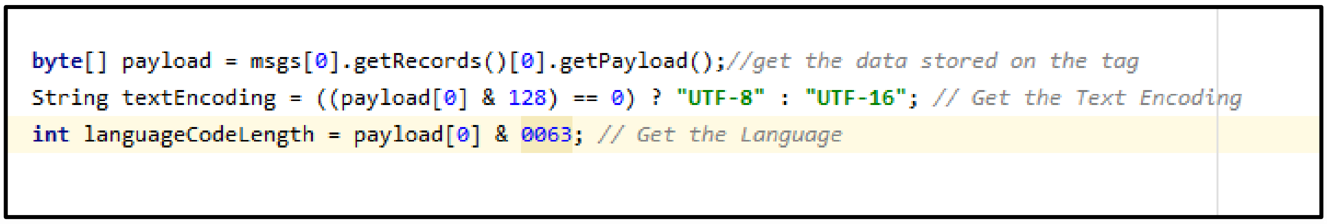
Figure 2.3 shows an xml file, which is used to filter the list of NFC tag technology that is readable by the application. Ndef is the standardised method of communication between a reader and a tag in NFC. Additional technologies could be added here but for my project Ndef was sufficient.



```java
*/
private void readFromIntent(Intent intent) {
    String action = intent.getAction();
    if (NfcAdapter.ACTION_TAG_DISCOVERED.equals(action)
            || NfcAdapter.ACTION_TECH_DISCOVERED.equals(action)
            || NfcAdapter.ACTION_NDEF_DISCOVERED.equals(action)) {
        Parcelable[] rawMsgs = intent.getParcelableArrayExtra(NfcAdapter.EXTRA_NDEF_MESSAGES);
        NdefMessage[] msgs = null;
        if (rawMsgs != null) {
            msgs = new NdefMessage[rawMsgs.length];
            for (int i = 0; i < rawMsgs.length; i++) {
                msgs[i] = (NdefMessage) rawMsgs[i];
            }
        }
```

Fig 2.4

The code in figure 2.4 is checking to see what kind of tag has been read and see if it matches what has been declared in the manifest file in fig 2.3 and also gets the information from the tag.



```
byte[] payload = msgs[0].getRecords()[0].getPayload();//get the data stored on the tag
String textEncoding = ((payload[0] & 128) == 0) ? "UTF-8" : "UTF-16"; // Get the Text Encoding
int languageCodeLength = payload[0] & 0063; // Get the Language
```

Fig 2.5

The code in figure 2.5 on line one is reading the data that is stored on the tag, on line two it is converting it to a string and on the third line it is figuring what language it is e.g. English. In my case it is numerical.

Android studio

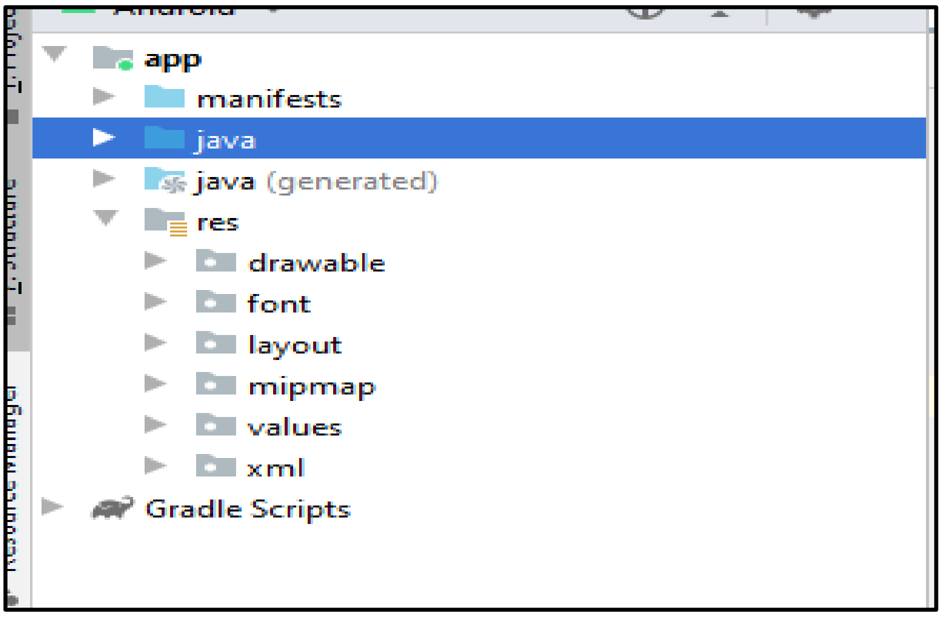
Android Studio is the IDE that I used to develop my application in java. Android Studio is the official IDE for Googles Android operating system built on JetBrains IntelliJ IDEA; it has been designed specifically for Android development. Android studio makes it easier to build the user interface and functionality of your application simultaneously.

Gradle

Gradle is a build automation tool and is specific to Android since Android Studio has been released. Each module in a project has its own Gradle. The benefit of Gradle is that it knows which part of the build is up to date so that parts that have already been built do not have to be executed again.

Gradle is also used for configuring project details such as the SDK version and dependencies [2].

Project Structure

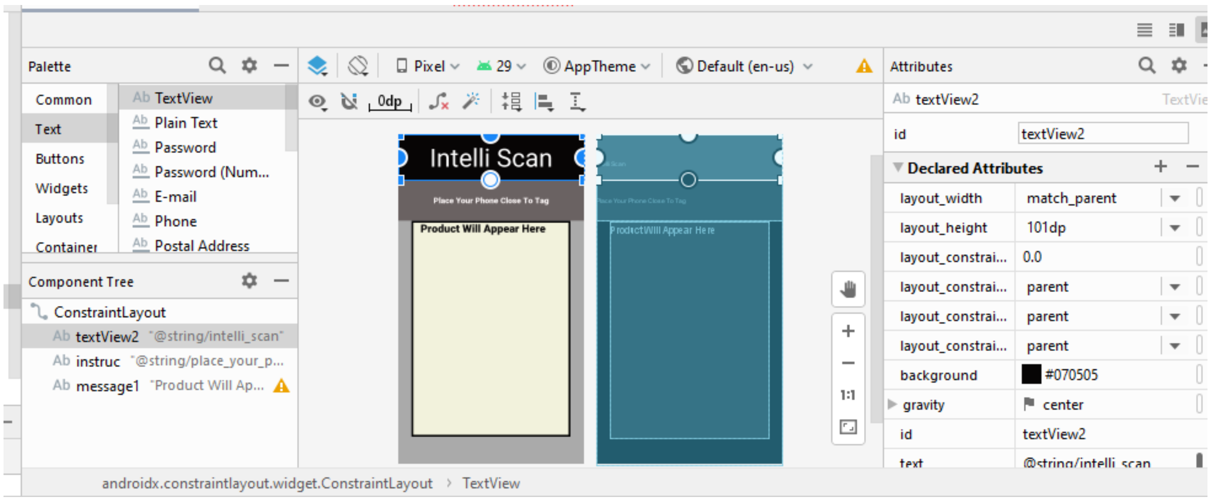


3.1

From fig 3.1 the project structure of Android Studio can be seen. The manifest file defines the project structure and metadata of the application and its components and requirements. For example, in my project permissions for using internet and using the smartphones NFC is included in the manifest file.

The Java folder includes the activity files of the application. The res/layout folder includes the main activities layout files. These xml files are the design of how the project will be displayed to the applications user.

Layout files in android studio can either be edited using xml text or by using the built-in layout editor. The layout editor is useful for understanding what the app user will be seeing when they are using the application.



Component Tree
- ConstraintLayout
  - textView2 "@string/intelli_scan"
  - instruc "@string/place_your_p...
  - message1 "Product Will Ap...

Attributes: textView2 (TextView)

id	textView2

Declared Attributes

layout_width	match_parent
layout_height	101dp
layout_constrai...	0.0
layout_constrai...	parent
layout_constrai...	parent
layout_constrai...	parent
background	#070505
gravity	center
id	textView2
text	@string/intelli_scan

Fig 3.2

Figure 3.2 shows the layout view on android studio, included is my applications design for the user. On the right-hand side of the editor multiple attributes of the textbox can edited to fit the design I wanted.

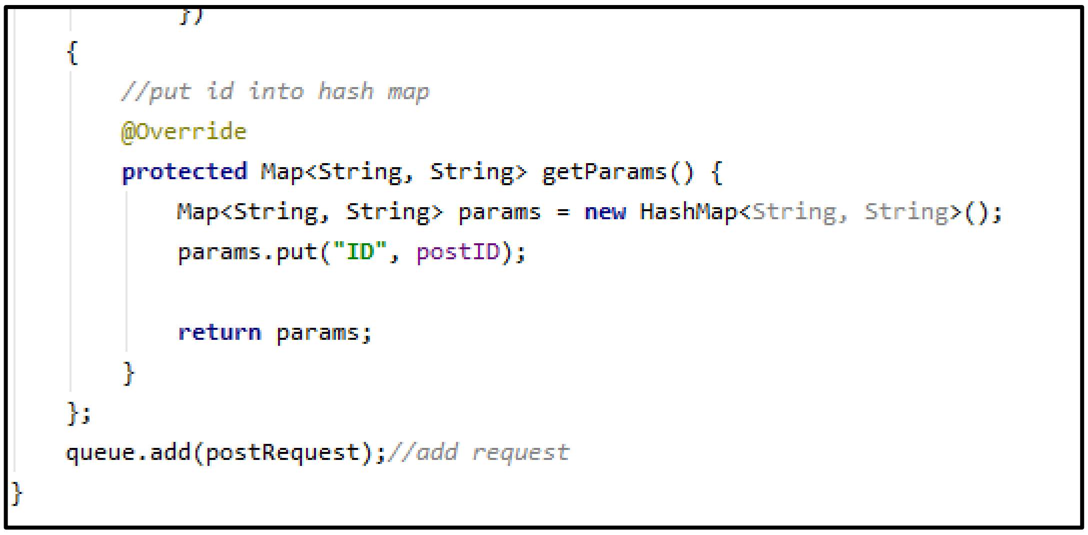
Volley Library

Volley is a HTTP library that is used for making networking easier and faster in android. Volley is suitable for fetching small files images and JSON responses. Volley is what I used in this project for sending a post request with my products ID in order to receive information about the product from the SQL database. In order to use Volley it must be added to the GRADLE file dependencies as well as the internet permissions in the manifest files.

Volley uses a request queue for dispatching requests to the network. A request has all the necessary information in it for making a web call stored in it. In my project I used a String post request to post the product ID String to the PHP page that is handling my database requests.

To send a request, a request is constructed and added to the request queue using add( ).Volley runs one cache processing thread and a pool of network dispatch threads. When you add a request to the queue, it is picked up by the cache thread and its priority is determined [3].

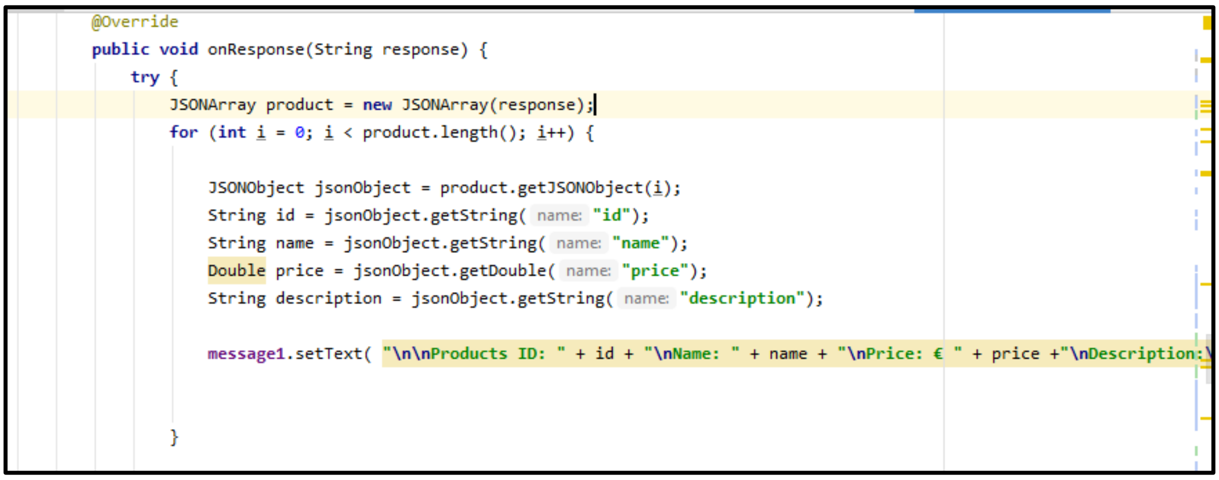
If the request can be serviced from the cache, the cached response is parsed on the cache thread and the parsed response is delivered on the main thread. If the request cannot be serviced from cache it is placed on the network queue. The first available network thread takes the request from the queue and performs the HTTP transaction. It then parses the response on the main worker thread and writes the response to the cache and posts the response back to the main thread for delivery [3].



```
            })
    {
        //put id into hash map
        @Override
        protected Map<String, String> getParams() {
            Map<String, String> params = new HashMap<String, String>();
            params.put("ID", postID);

            return params;
        }
    };
    queue.add(postRequest);//add request
}
```

Fig 4.1

The code in figure 4.1 is adding the id attained from the NFC tag to a hash map. The ID will then be added to the request and sent to the server.



```
        @Override
        public void onResponse(String response) {
            try {
                JSONArray product = new JSONArray(response);
                for (int i = 0; i < product.length(); i++) {

                    JSONObject jsonObject = product.getJSONObject(i);
                    String id = jsonObject.getString( name: "id");
                    String name = jsonObject.getString( name: "name");
                    Double price = jsonObject.getDouble( name: "price");
                    String description = jsonObject.getString( name: "description");

                    message1.setText( "\n\nProducts ID: " + id + "\nName: " + name + "\nPrice: € " + price +"\nDescription:\

                }
```

Fig 4.2

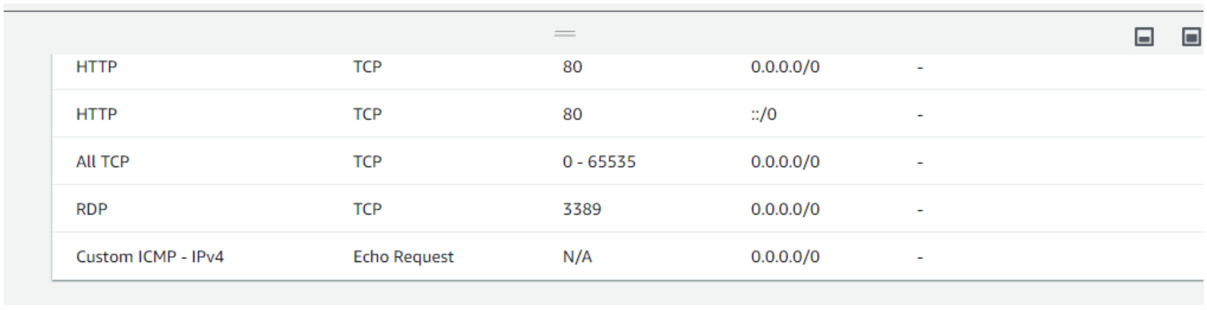
The code shown in figure 4.2 is showing the response from the server after the database query has been completed. The response is a JSON object so it must be first parsed and then it can be displayed in the message1 text view.

AWS

In order to access my database from my application I need to run it on a server. AWS stands for amazon web services and this is what I ran my server on. AWS has an educate package for students which provides one-hundred-euro worth of free credit. This and that fact that there is a huge amount of documentation provided on how to use AWS is one of the main reasons I chose to use it.

An EC2 instance stands for Elastic Compute Cloud and is used for running applications on AWS infrastructure. I chose to launch a windows instance from the instance AMI options.

Before launching an instance a security group must be configured or one can be chosen from the existing list. A security group acts as a virtual firewall for the instance to control the incoming and outgoing traffic [4].



HTTP	TCP	80	0.0.0.0/0	-
HTTP	TCP	80	::/0	-
All TCP	TCP	0 - 65535	0.0.0.0/0	-
RDP	TCP	3389	0.0.0.0/0	-
Custom ICMP - IPv4	Echo Request	N/A	0.0.0.0/0	-

Fig 5.1

Figure 5.1 shows the inbound rules that I configured on my AWS instance.

WAMP

WAMP stands for Windows Apache MySQL and PHP. A WAMP is a software bundle which I installed on my windows instance. Apache is the most important aspect of the WAMP package as it is used to run the server within the windows virtual machine.

WAMP also includes MySQL and PHP which are two of the most common technologies used in creating dynamic websites. MySQL is a high-speed database which I used to store my products ID numbers and information. PHP is a scripting language that is used for to access data in the database.

In my project my application sent a post request to the php pages to interact with the database. On the admin side, the PHP pages were used for inserting or updating info into the database.

WAMP stacks are more suitable for being used on prototype projects such as min rather than in production environments.

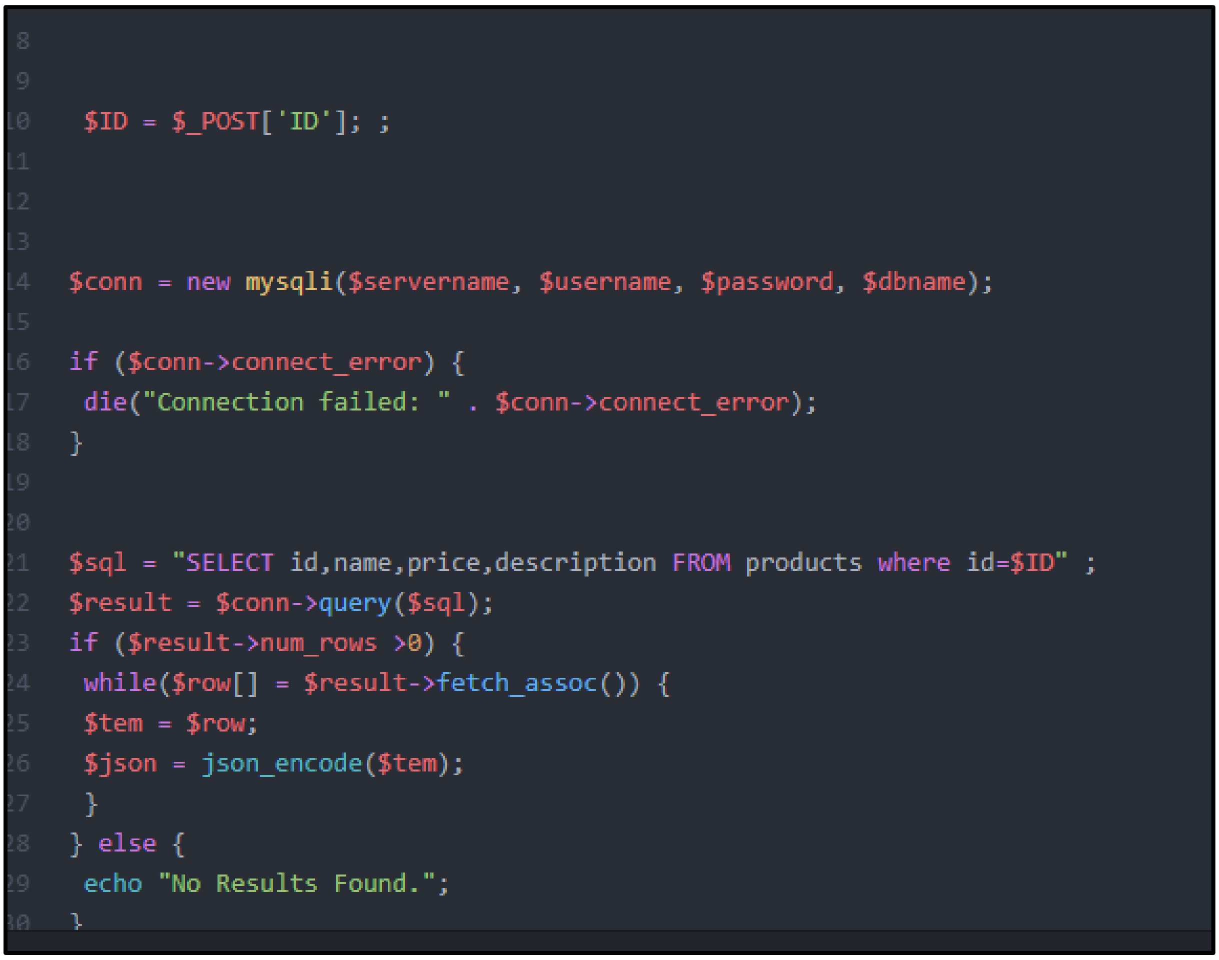
For my project I used a third party WAMP package recommended by AWS called Bitnami. In order for Bitnami to be installed the security groups must be set as stated in the AWS chapter of my report. Also, IE Enhanced Security Configurations on the instance must be turned off for the web browser on the windows virtual machine to be useable. Then I went to the Bitnami website and downloaded and installed it from there.

Once installed, Bitnami has a user interface where the user can control whether the server is running or not and can also gain access to the PHP MyAdmin login page for the SQL database. In order to test if the server is running, I could just enter the public ip of the instance followed by a file name that was in the htdocs folder of the WAMP file structure, e.g. index. If the page is accessible it was working [5].

PHP pages

PHP is a server-side scripting language that is used to develop Static websites, Dynamic websites and web applications. PHP can only be used on servers that have PHP installed.

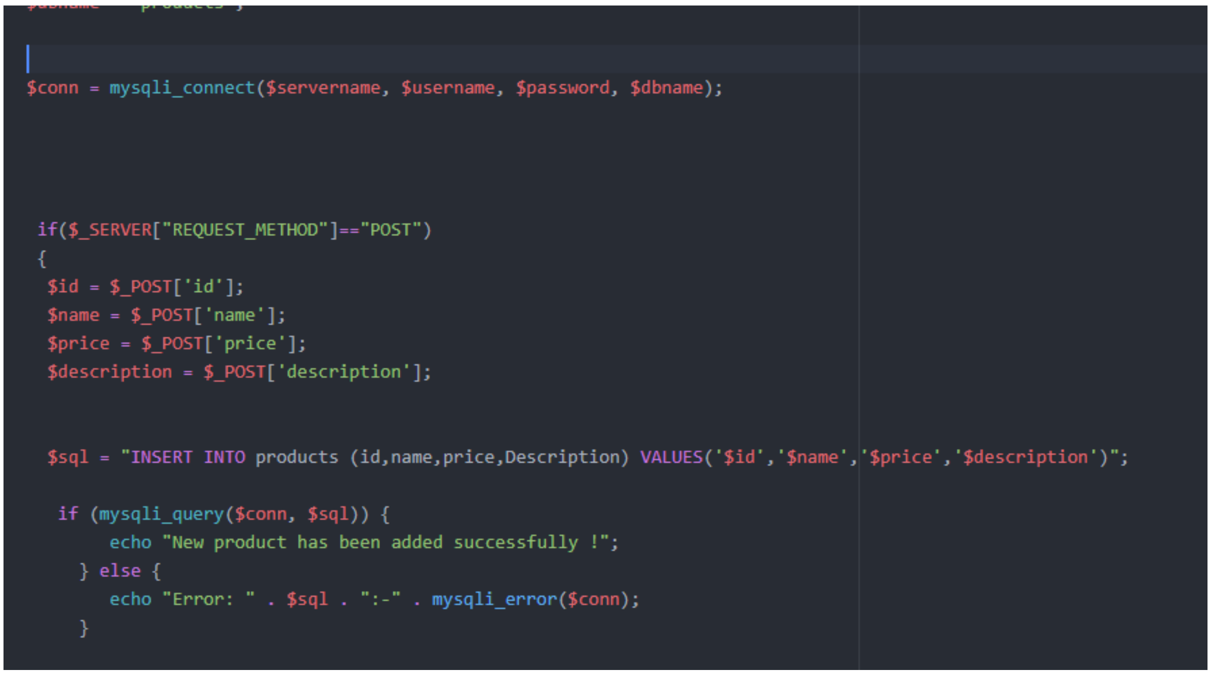
In the case of my project I used three PHP scripts one for interacting with my application and the other two for interacting with my admin pages.



```
8
9
10   $ID = $_POST['ID']; ;
11
12
13
14  $conn = new mysqli($servername, $username, $password, $dbname);
15
16  if ($conn->connect_error) {
17   die("Connection failed: " . $conn->connect_error);
18  }
19
20
21  $sql = "SELECT id,name,price,description FROM products where id=$ID" ;
22  $result = $conn->query($sql);
23  if ($result->num_rows >0) {
24   while($row[] = $result->fetch_assoc()) {
25   $tem = $row;
26   $json = json_encode($tem);
27   }
28  } else {
29   echo "No Results Found.";
30  }
```

Fig 6.1

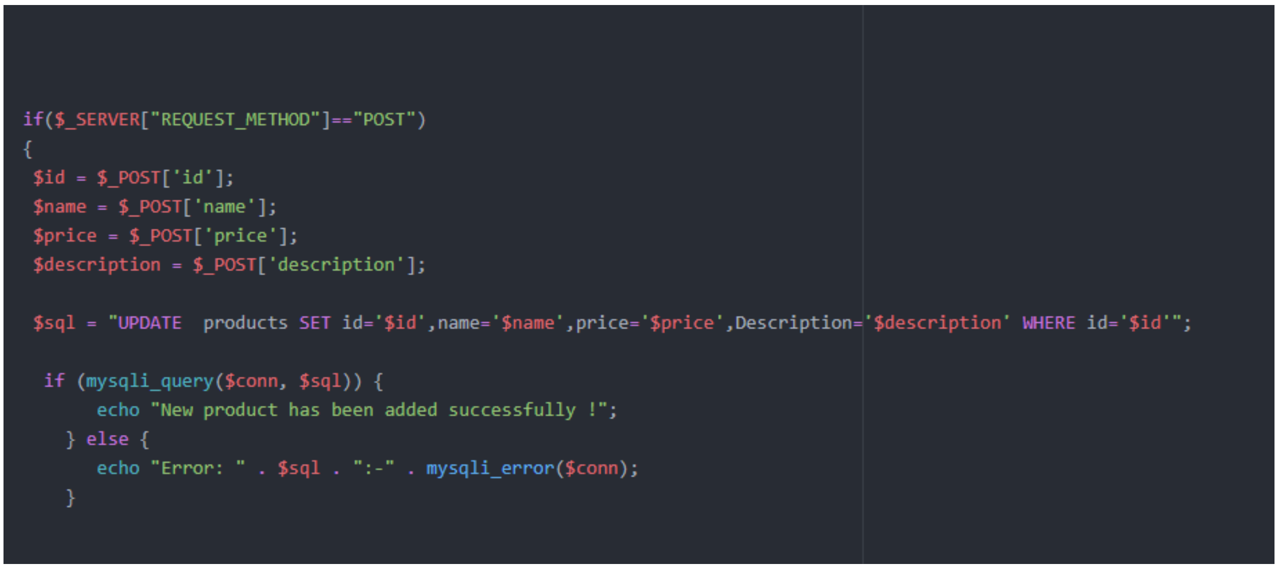
The PHP code shown in figure 6.1 is the script that the application interacts with when it posts the products id number. The code written on line is checking for the connection to the database. I had my login values as being the variable names shown at the beginning of this script. So, this piece of code is effectively verifying if the connection to the database using my credentials is successful. Line 21 is the SQL command that will be sent to the database whereby the is id will be the id that I have sent to the php page from my application. The code following this line is getting the result of the query and encoding it as a JSON object to be sent back to the users application.



```
$conn = mysqli_connect($servername, $username, $password, $dbname);



 if($_SERVER["REQUEST_METHOD"]=="POST")
 {
  $id = $_POST['id'];
  $name = $_POST['name'];
  $price = $_POST['price'];
  $description = $_POST['description'];


  $sql = "INSERT INTO products (id,name,price,Description) VALUES('$id','$name','$price','$description')";

   if (mysqli_query($conn, $sql)) {
        echo "New product has been added successfully !";
     } else {
        echo "Error: " . $sql . ":-" . mysqli_error($conn);
     }
```

Fig 6.2

Figure 6.2 is the PHP that the product add HTML page interacts with when a new product needs to be inserted into the database. Much the same as previously the connection using my login details is done at the beginning. Then the PHP receives the information posted from the HTML page. The details entered on the webpage are then inserted into the INSERT INTO SQL command and the command is executed.



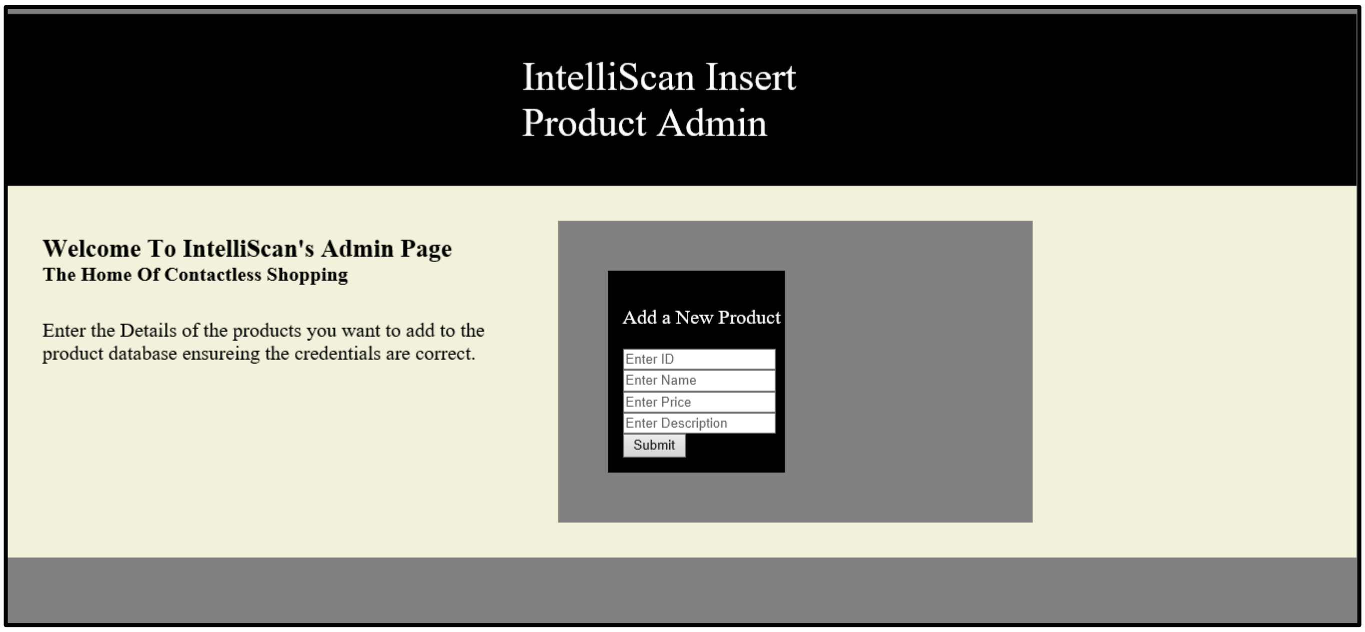
```
if($_SERVER["REQUEST_METHOD"]=="POST")
{
 $id = $_POST['id'];
 $name = $_POST['name'];
 $price = $_POST['price'];
 $description = $_POST['description'];

 $sql = "UPDATE  products SET id='$id',name='$name',price='$price',Description='$description' WHERE id='$id'";

  if (mysqli_query($conn, $sql)) {
       echo "New product has been added successfully !";
    } else {
       echo "Error: " . $sql . ":-" . mysqli_error($conn);
    }
```

Fig 6.3

The PHP script in figure in 6.3 is for updating an existing product in the database. The script is mostly the same as the one for inserting a new product the only difference being the SQL command. The SQL command will update any new details entered for the ID number that is entered.

HTML

HYML stands for Hypertext Markup Language and is used for writing webpages. I used HTML to design the webpages for the admin side of my project.



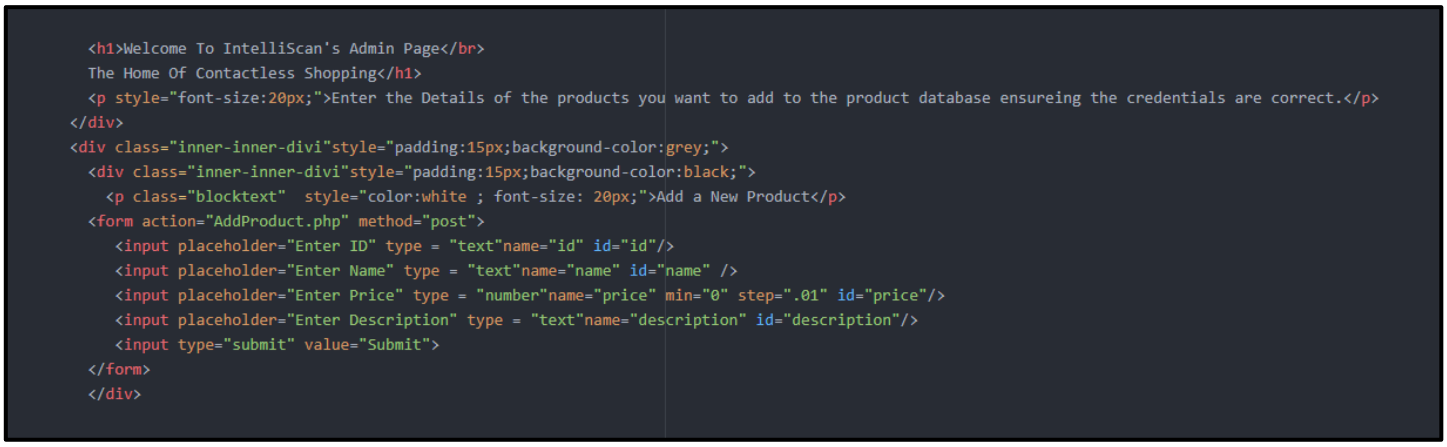
IntelliScan Insert
Product Admin

Welcome To IntelliScan's Admin Page
The Home Of Contactless Shopping

Enter the Details of the products you want to add to the product database ensureing the credentials are correct.

Fig 7.1

The image in figure 7.1 is what the user sees when they visit the new user webpage.

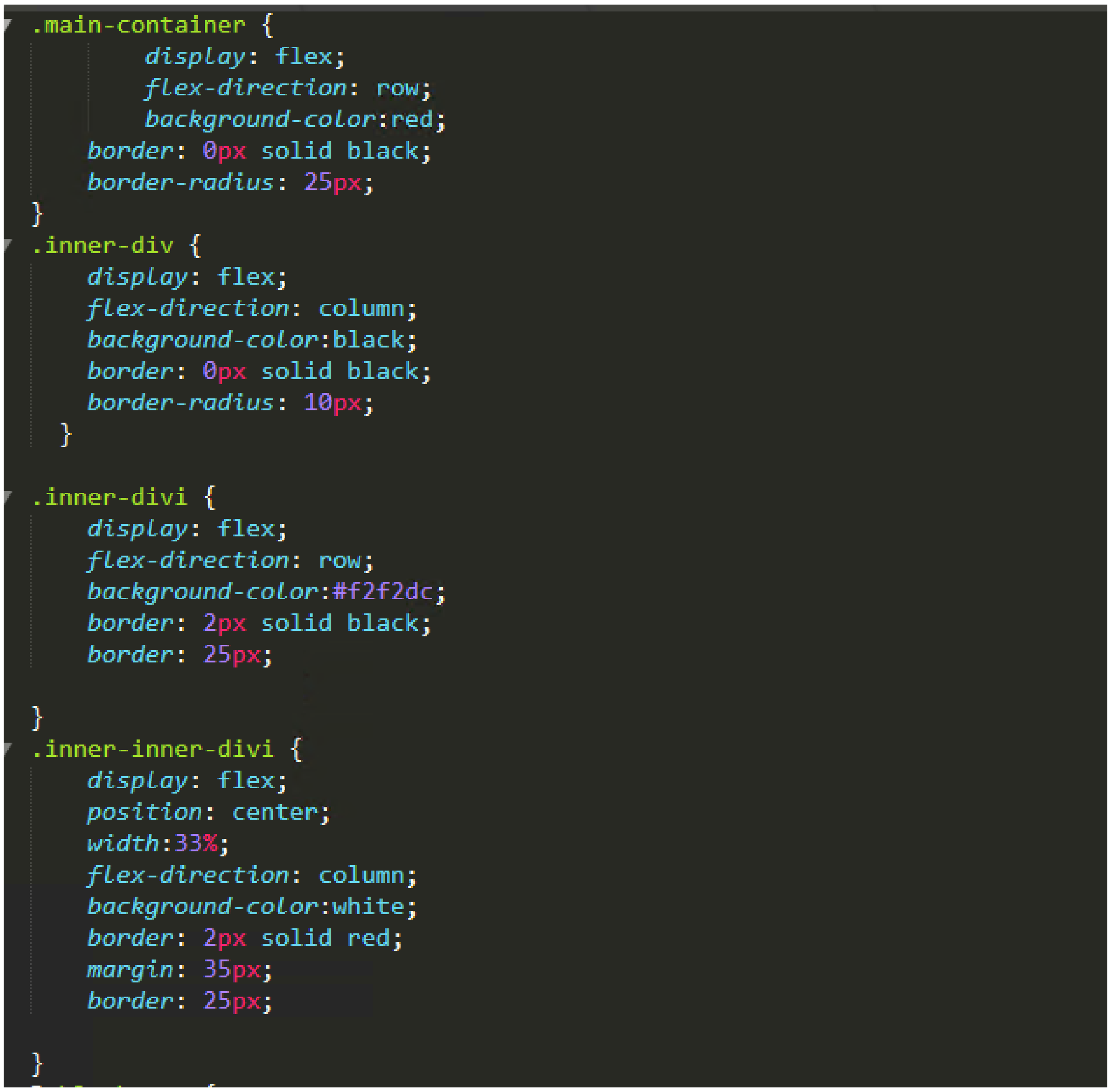


```
  <h1>Welcome To IntelliScan's Admin Page</br>
  The Home Of Contactless Shopping</h1>
  <p style="font-size:20px;">Enter the Details of the products you want to add to the product database ensureing the credentials are correct.</p>
</div>
<div class="inner-inner-divi"style="padding:15px;background-color:grey;">
  <div class="inner-inner-divi"style="padding:15px;background-color:black;">
    <p class="blocktext"  style="color:white ; font-size: 20px;">Add a New Product</p>
  <form action="AddProduct.php" method="post">
     <input placeholder="Enter ID" type = "text"name="id" id="id"/>
     <input placeholder="Enter Name" type = "text"name="name" id="name" />
     <input placeholder="Enter Price" type = "number"name="price" min="0" step=".01" id="price"/>
     <input placeholder="Enter Description" type = "text"name="description" id="description"/>
     <input type="submit" value="Submit">
  </form>
  </div>
```

Fig 7.2

The HTML snippet in figure 7.2 is showing the how the placeholder for entering details to be inserted is formed. The placeholders are placed within a form which is how the HTML page is linked to the PHP script for inserting the data. The layout for the product update page is the same with the only difference being that the form is linked to the update PHP script.

In order to make the building of the HTML page I used a CSS file which globally declares the style values of the divisions in the HTML page.



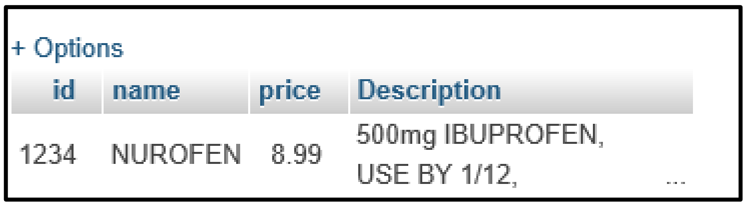
```
.main-container {
        display: flex;
        flex-direction: row;
        background-color:red;
    border: 0px solid black;
    border-radius: 25px;
}
.inner-div {
    display: flex;
    flex-direction: column;
    background-color:black;
    border: 0px solid black;
    border-radius: 10px;
  }

.inner-divi {
    display: flex;
    flex-direction: row;
    background-color:#f2f2dc;
    border: 2px solid black;
    border: 25px;

}
.inner-inner-divi {
    display: flex;
    position: center;
    width:33%;
    flex-direction: column;
    background-color:white;
    border: 2px solid red;
    margin: 35px;
    border: 25px;

}
```

Fig 7.3

The snippet in figure 7.3 is how I set the values of each division used in my HTML. As can be seen characteristics such as margin width and the direction of the flex. Flex meaning the direction that multiples of the version appear when they are placed beside each other.

MySQL

MySQL is a relational database management system based on SQL- Structured Query Language. In my project I used MySQL as the database for storing my products and their details. I used PhpMyAdmin for updating my database in the early stages of my project, before I created an admin page for doing so.

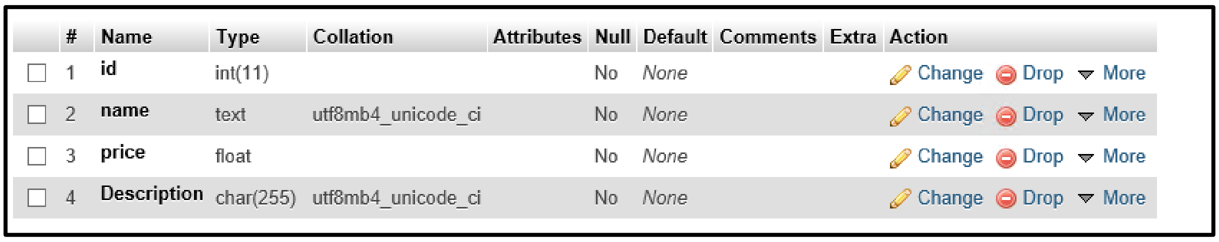


+ Options

id	name	price	Description
1234	NUROFEN	8.99	500mg IBUPROFEN, USE BY 1/12, ...

Fig 8.1

Figure 8.1 shows my database table with the ID name price and description of a product shown.



	#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
☐	1	id	int(11)			No	*None*			Change Drop More
☐	2	name	text	utf8mb4_unicode_ci		No	*None*			Change Drop More
☐	3	price	float			No	*None*			Change Drop More
☐	4	Description	char(255)	utf8mb4_unicode_ci		No	*None*			Change Drop More

Fig 8.2

Figure 8.2 shows the structure of my table and the type of data that is in each row. E.g. ID is an int.

Future Additional features

As I previously stated in earlier parts of my report there are certain additional features that I would have liked to add to my project but weren’t possible due permissions issues on my Samsung J4+. The main feature I was interested in adding was to be able to save each item that has been scanned to an SQLite database. The products stored in that database could then be displayed at the end of the users shopping experience, using a button to call the information from the table.

An additional feature that I would like to add in the future is possibly integrating the product with PayPal as PayPal provide an sdk for android. PayPal is the largest payment gateway in the world and enables seamless payment transaction in e-commerce and mobile applications. If I could implement PayPal with my application that would mean that the shopping experience while using my application would be fully contactless and the complete scope that I set out to achieve would be fulfilled [6].

I feel that if I could integrate both of these features in the future, my application could be greatly beneficial to numerous people including shoppers and supermarkets if they chose to use it.

Conclusion

When I began my project, I had no understanding of how Android Studio worked or I how I could use it to implement the concept that I had for my project. But over the months of developing my application and after multiple periods of elation and disappointment, I feel as if I benefited greatly from the experience. It gave me a real understanding of how to use the IDE and of how applications work.

As well as android I got a very good understanding of how NFC technology works. It is a real emerging technology and to have worked closely with it over the past few months, provided me with a real learning curve but without having done so, I wouldn't understand what I know about it today. I think having worked with NFC in my project will benefit me in my future career endeavours.

Ultimately my final project didn't fully meet the targets that I had set out to achieve at the conception of my idea. Having said that I think what I did complete is quite close to what I wanted to achieve. The application could prove to be very useful in its current format such as finding out allergen information on a product or finding out the price it. It could also be useful in current supermarket environments whereby customers wouldn't have to pick up a product to find out information thus helping to prevent the spread of COVID-19.

References

[1] Androidauthority.com(2019,June,30). What is NFC and how does it work. Available: https://www.androidauthority.com/what-is-nfc-270730/

[2] Developers.android.com(2020.May.01).Configure your build. Available: https://developer.android.com/studio/build

[3] Developer.android.com(2019.Dec,27). Send a Simple Request. Available: https://developer.android.com/training/volley/simple

[4] Doc.aws.amazon.com(n/a)Amazon EC2 security groups. Available: https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

[5] Docs.aws.amazon.com(n/a) Tutorial: Installing a WAMP Server on an Amazon EC2 Instance Running Windows Server. Available: https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/install-WAMP.html

[6] Licio Lentimo Medium.com(2019.August.2019) PayPal Integration on Android. Available: https://medium.com/@lentimo/paypal-integration-on-android-c655480ae6b2

[7] YouTube(2018,Mar,11).NFC Explained: What is NFC? How NFC Works? Applications of NFC. Available: https://www.youtube.com/watch?v=eWPtt2hLnJk