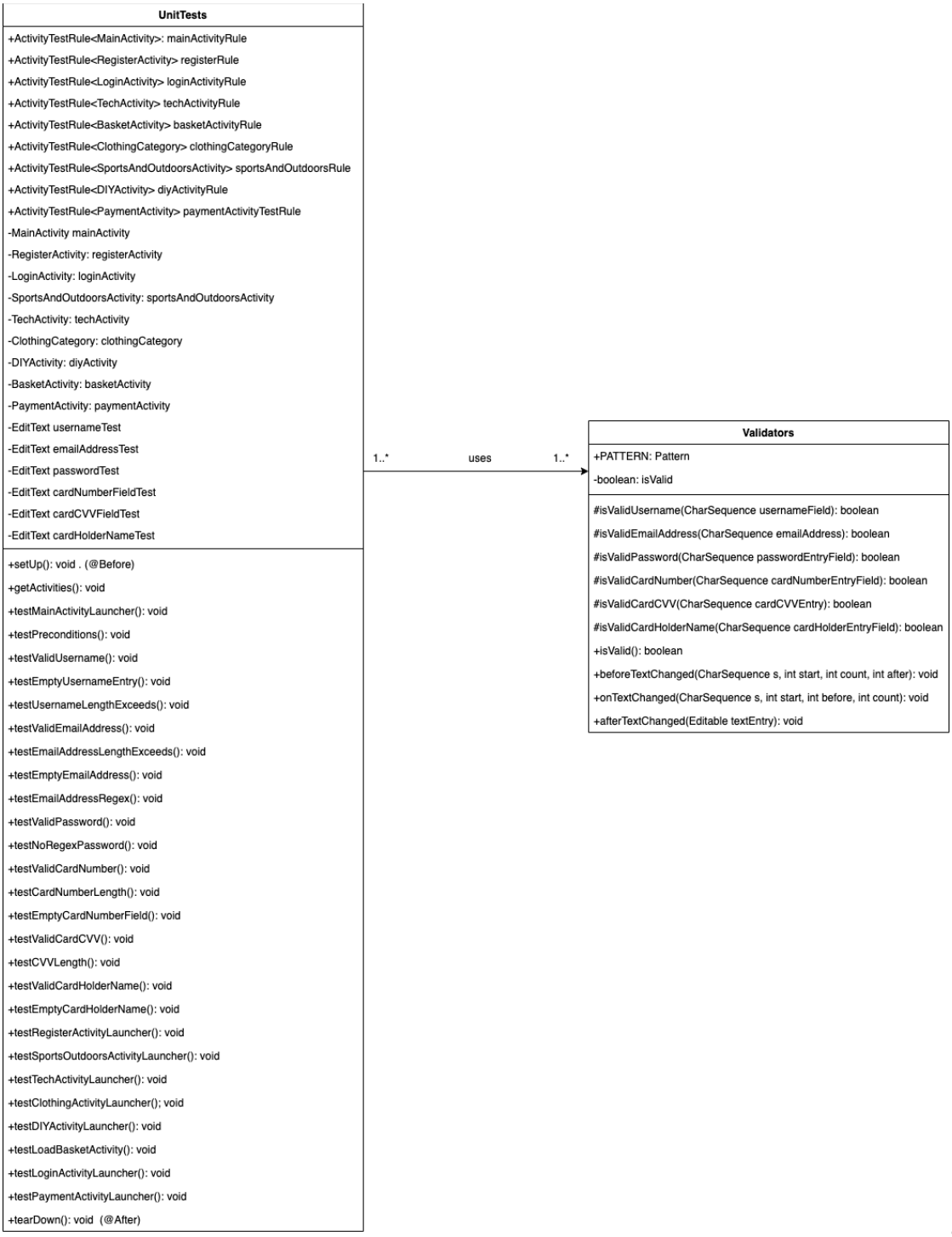
2020

We Shop E-Commerce Shopping Application – Mobile Applications Development Project

SABIN CONSTANTIN LUNGU
40397517

Sabin Constantin Lungu
Matriculation Number: 40397517
Mobile Applications Development Project Software Document

Table of Contents



We Shop E-Commerce Android Application

Project Proposal (Introduction) – 1.0

I am planning on designing and implementing an e-Commerce shopping application in Android that will allow end-users to register, login to the app and browse for products in different categories just like **Amazon**.

This application will currently have 4 pages of product categories that users can browse through and they are **Sports & Outdoors**, **Tech, Clothing** and **DIY** and once the users have chosen the products that they wish to purchase from category **X**, they can add it to their basket and proceed to the checkout. Customers can view the products added to the basket.

Before browsing for products, users are required to **Register** and **Login** into their account, however users can also browse the products before logging in just to see if they would like to register after or not.

The main scope of this application is to allow end-users to shop for products in different categories, very similar to Amazon's application.

After registration customers will receive a notification saying that they have registered successfully and the data from the registration will get sent to a back-end database. I will be making use of the **Firebase Service** to store the user's registration data.

When customers have decided on what they want to purchase they can add the products to their basket and the users can view their basket in the form of a list view.

Users will be presented with a payment form where they are prompted to fill out their payment information and pay for the products chosen. After the payment is complete and processed users will receive an invoice through e-mail.

Each category with products will be shown on different activity pages on the application. For example, on page 1 the sports and outdoors category section will be displayed with the products available in stock along with the available sizes, colors and quantities.

I will be making use of version control using **Git** in order to track the changes made to the project and to also commit and push new changes when developing the application. In case I lose my work, I can revert back to previous commits to start from that checkpoint.

My initial inspiration for choosing this type of application was from the **Amazon** application.

The 3-Tier Software Architecture that consists of the Application Layer, Business Layer and Data Layer will be used to develop this application.

Feasibility Study – 1.1

In this section I will be writing about the different kinds of feasibility studies that will impact this project. These will determine whether this project is feasible enough to develop or not.

Legal Feasibility – 1.2

Under the **Copyright Design and Patents Act**, all of the images and source code that I will be using throughout this project I will reference in an index at the end of the project.

Under the **Data Protection Act** all of the registration data will get stored in a secure back-end database using **Firebase**.

Under the **Computer Misuse Act,** I will not any malware that will harm and render other people's computer unavailable.

Schedule Feasibility – 1.3

The total time taken for each task will be stated in a Gantt Chart and also the start and end dates for that task. The Gantt Chart will also show whether or not that task has been completed or if it's still pending.

Economic Feasibility – 1.4

I will not be purchasing any software in order to achieve project completion, all of the software and tools that I will be using to develop this project will be free to use. No costs will be incurred.

Technical Feasibility – 1.5

I will be making use of my MacBook Pro to develop my project. The software that will be used are:

- **Android Studio:** Where I will implement the User Interface
- **Draw.io**: Used to design the class diagrams using UML.
- **Zube**: Used to implement Kanban Boards & implement the sprints for the project.
- **Firebase**: Used to store the registration details of the customers.
- **Adobe XD CC**: Used to design the User Interface of the application.
- **Survey Monkey**: Used to gather data from end-users by creating a survey of 10 questions.

Gantt Chart – 1.2

The chart below will show how long each development task will take.

Task	Start Time	End Time	Completed
Analysis	14th January 2020	18th January 2020	**Completed**
Design	18th January 2020	1st February 2020	**Completed**
Implementation	2nd February 2020	29th February 2020	**Completed**
Testing	1st March 2020	4th March 2020	**Completed**
Evaluation	5th March 2020	6th March 2020	**Completed**

From the table above, the total time taken to complete this Android Application will be **41 Days.**

Before writing the Requirements Specification, I will do some data gathering in order to find out what end-users would like to see and have in the application, then I can turn that data into a meaningful and feasible requirements specification.

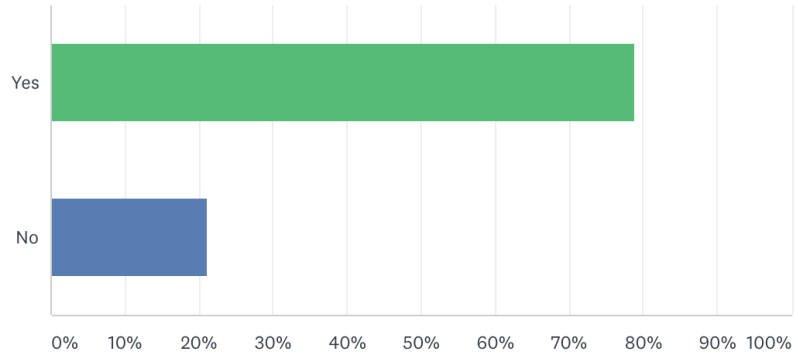
Survey Data Gathering – Analysis

Q1

Customize Save as

Would you like to see a homepage with a welcome image?

Answered: 19 Skipped: 0



ANSWER CHOICES	RESPONSES	
Yes	78.95%	15
No	21.05%	4
Total Respondents: 19		

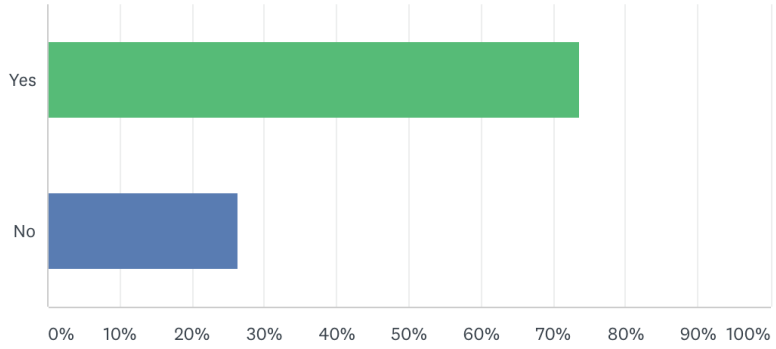
It is clear that the majority of the end-users would like to see a welcome image on the homepage, hence this will be a **Requirement Feature**.

Q2

Customize Save as

Would you like to see the homepage with some text below the image that describes this We Shop application?

Answered: 19 Skipped: 0



ANSWER CHOICES	RESPONSES	
Yes	73.68%	14
No	26.32%	5
Total Respondents: 19		

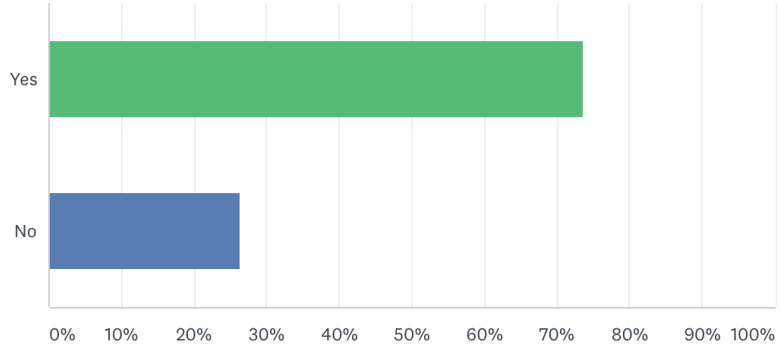
It is clear that most end-users would like this feature in the application, hence this will be another **Requirement Feature** for the application.

Q3

Customize Save as

Would you like the main activity page to have a register and a login button underneath the text?

Answered: 19 Skipped: 0



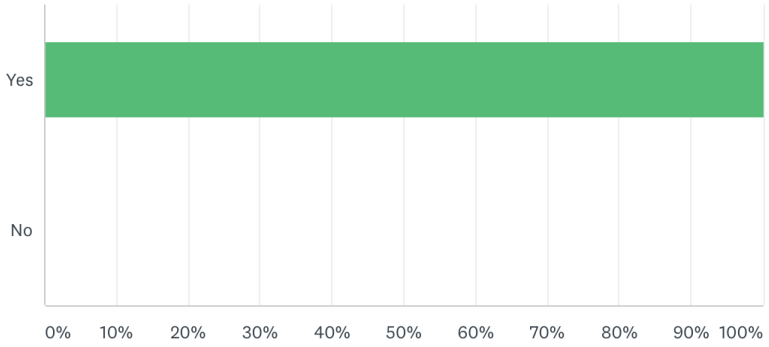
ANSWER CHOICES	RESPONSES	
Yes	73.68%	14
No	26.32%	5
Total Respondents: 19		

Q4

Customize Save as

Would you like the registration page to have entries for Username, E-mail Address, password and a terms and conditions check box?

Answered: 19 Skipped: 0



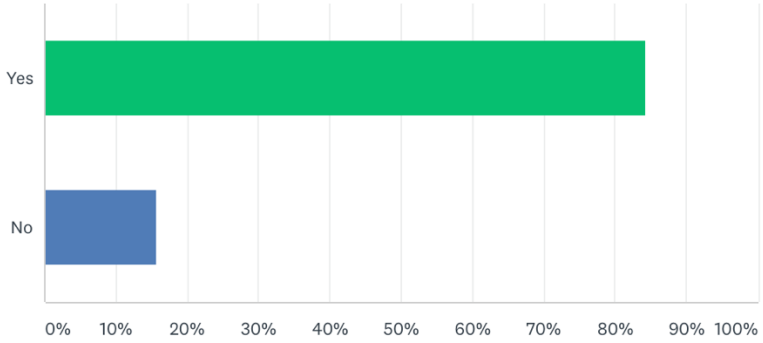
ANSWER CHOICES	RESPONSES	
Yes	100.00%	19
No	0.00%	0
Total Respondents: 19		

Q5

Customize Save as

Would you like to login into the application using your E-mail Address and password only?

Answered: 19 Skipped: 0



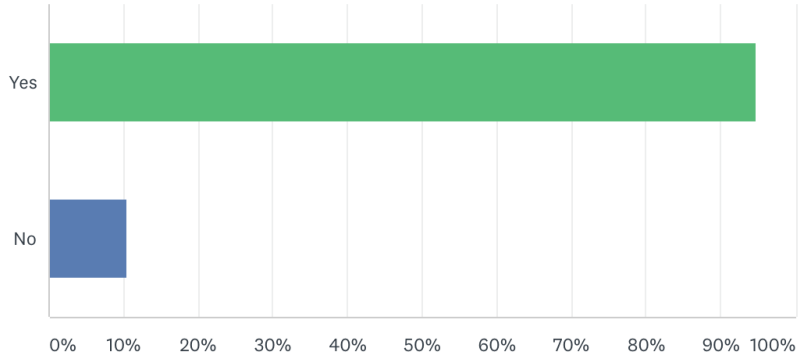
ANSWER CHOICES	RESPONSES	
Yes	84.21%	16
No	15.79%	3
Total Respondents: 19		

Q6

Customize Save as

Would you like the homepage to have a drop-down menu that allows the customers to choose what type of category to browse on?

Answered: 19 Skipped: 0



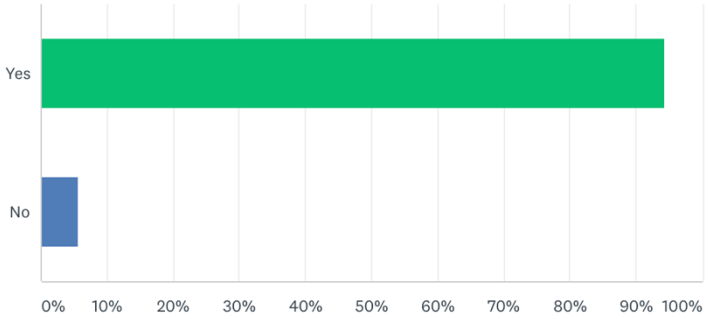
ANSWER CHOICES	RESPONSES	
Yes	94.74%	18
No	10.53%	2
Total Respondents: 19		

Q7

Customize Save as

Would you like your registration data to get stored in a Firebase database back-end?

Answered: 18 Skipped: 1



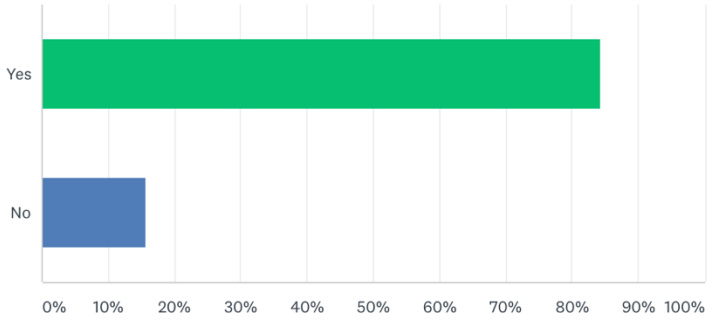
ANSWER CHOICES	RESPONSES	
Yes	94.44%	17
No	5.56%	1
TOTAL		18

Q8

Customize Save as

Would you like to see the products that are currently available in stock on the Sports & Outdoors activity?

Answered: 19 Skipped: 0



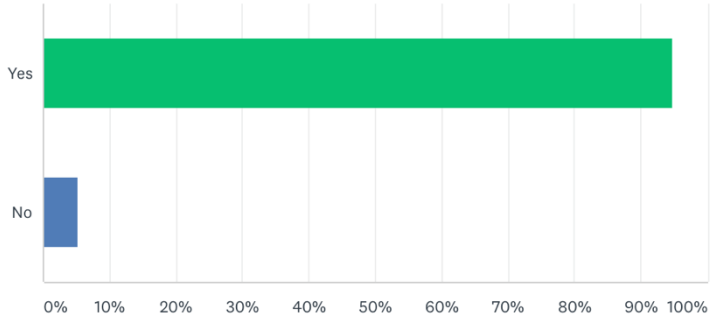
ANSWER CHOICES	RESPONSES	
Yes	84.21%	16
No	15.79%	3
Total Respondents: 19		

Q9

Customize Save as

Would you like to see a drop-down menu where you can choose the colour of the product available?

Answered: 19 Skipped: 0



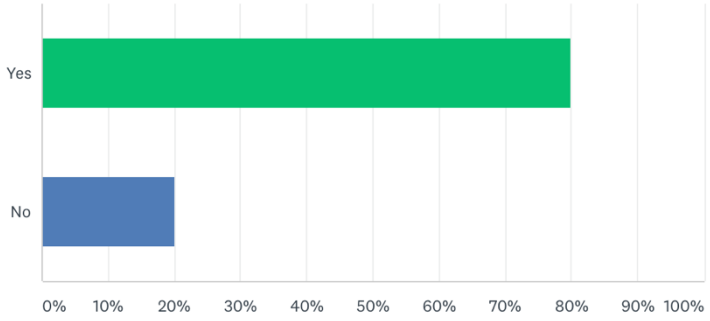
ANSWER CHOICES	RESPONSES	
Yes	94.74%	18
No	5.26%	1
Total Respondents: 19		

Q10

Customize Save as

Would you like to receive a notification after registration to ensure that you have registered successfully?

Answered: 20 Skipped: 0



ANSWER CHOICES	RESPONSES	
Yes	80.00%	16
No	20.00%	4
Total Respondents: 20		

(*) Initially there should have been 20 questions, however Survey Monkey only allows 10 questions on a basic free plan, I would've had to upgrade to go beyond 10 questions.

Functional Requirements Specification

From the survey data that has been gathered and the responses that the end-users have given, the requirements of this application are:

1. The main activity will have a **Register Button** and a **Login Button** below the text.
2. The registration activity will have entry fields for Username, E-mail Address, Password and a terms and conditions check box that must be ticked before registering successfully.
3. The registration activity entry fields will get validated against:

- Username **must not be empty.**
- Username **must contain some numbers.**
- Username length must not exceed 10 characters.
- The password field must: **contain numbers, special characters, characters and must not be left empty and also start with an upper-case character.**
- The e-mail address field must: **have an @ symbol and should not be left empty, ensure it is within the range of a-z, A-Z, 0-9 and contains characters from A-Z, 0-9 inclusively.**
- The Terms & Conditions check box should be ticked to register successfully.

4. The main activity will have a drop-down menu that allows users to choose which category of products they would like to choose from. The options are:

- Sports & Outdoors
- Tech
- Clothing
- DIY

The main activity will have a Contact Us button that will allow users to fill out a form if they are having any issues with the application. SQLite will be used as the database to store any complaints from users.

5. Users will receive a notification saying that they have registered successfully after clicking the register button.
6. The login activity will allow users to login with their E-mail Address and password only.
7. After users register with their unique account, **Firebase** will be used to store the credentials in a back-end database.

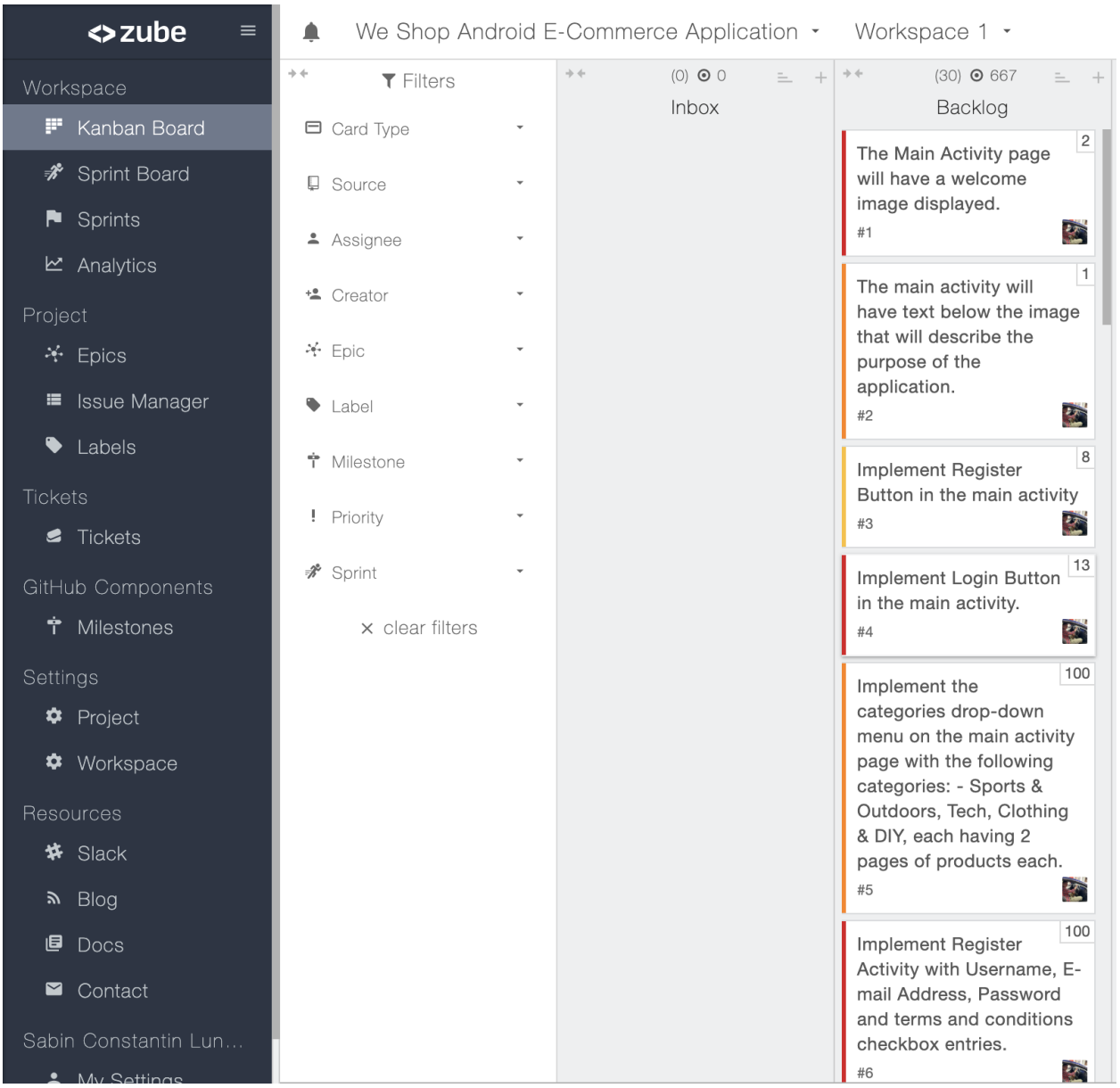
8. A toast message will appear saying “You are logged in as X” when the user has logged in successfully.
9. The **Sports & Outdoors** activity will show the images of the products that are currently in stock. There will be 2 pages of sports & outdoors products.
10. The **Tech** category activity will show the images of the products that are currently in stock.
11. The **Clothing** category activity will show the images of the products that are currently in stock.
12. The **DIY** category activity will show the images of the products that are currently in stock.
13. The cost of each product will be displayed next to the images in GBP.
14. Users have the option to choose the quantity of each product, hence that will affect the total cost of the product.
15. There will be an **“Add to Basket”** button underneath the products which allows the users to add the chosen product(s) to their basket (adds it to the menu items)
16. The Clothing products will have an option where users can choose the size, color and quantity they would like to pick.
17. Users have the option to view the products in their basket by clicking “**View Basket”** which will get placed in a menu on the activity.
18. On each of the categories, users will have the ability to choose the color of the product that they would like.
19. The application will have a **Payment** activity that will have the following entry fields when checking out:

- The type of payment the users may wish to pay with: **VISA, PayPal, Mastercard**
- **Card number**: XXXX XXXX XXXX XXXX (will get validated to ensure that only numbers are present) and no characters are inputted.
- Card **CVV** (* this is the 3-digit code at the back of the card), will get validated to ensure that no more than 3 numbers are entered.
- **Cardholder’s Name.**
- **Expiry Date (Month & Year).**

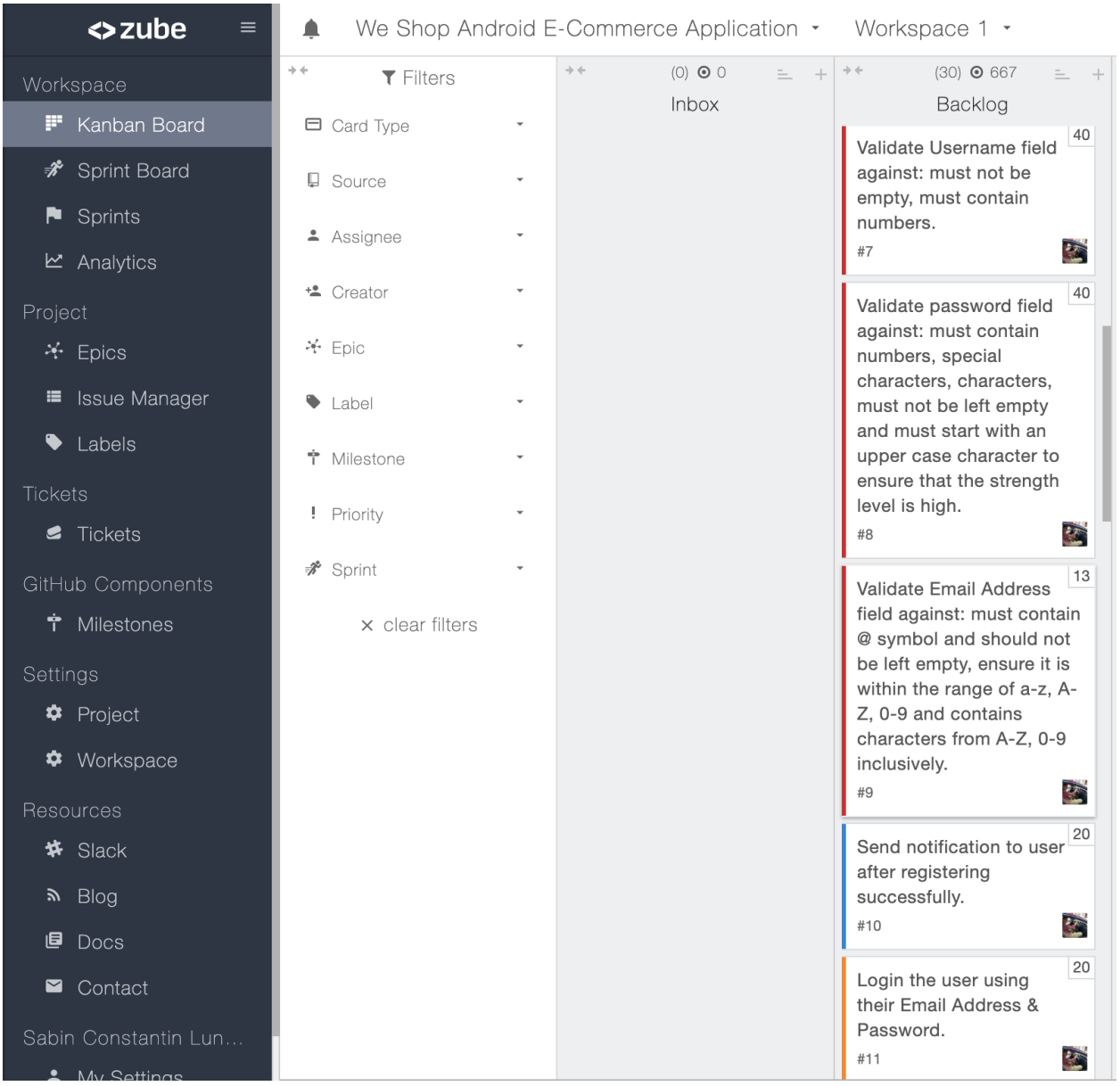
20. The payment activity will have a “Confirm Payment” button that the users will click when they are ready to pay for the products.
21. A rotating circular progress bar will be shown after the button has been clicked to indicate that the payment is being processed and after it has been complete, an alert dialogue will get displayed saying that the payment has been successful.
22. After the payment has been completed, the users will get re-directed back to the home page.

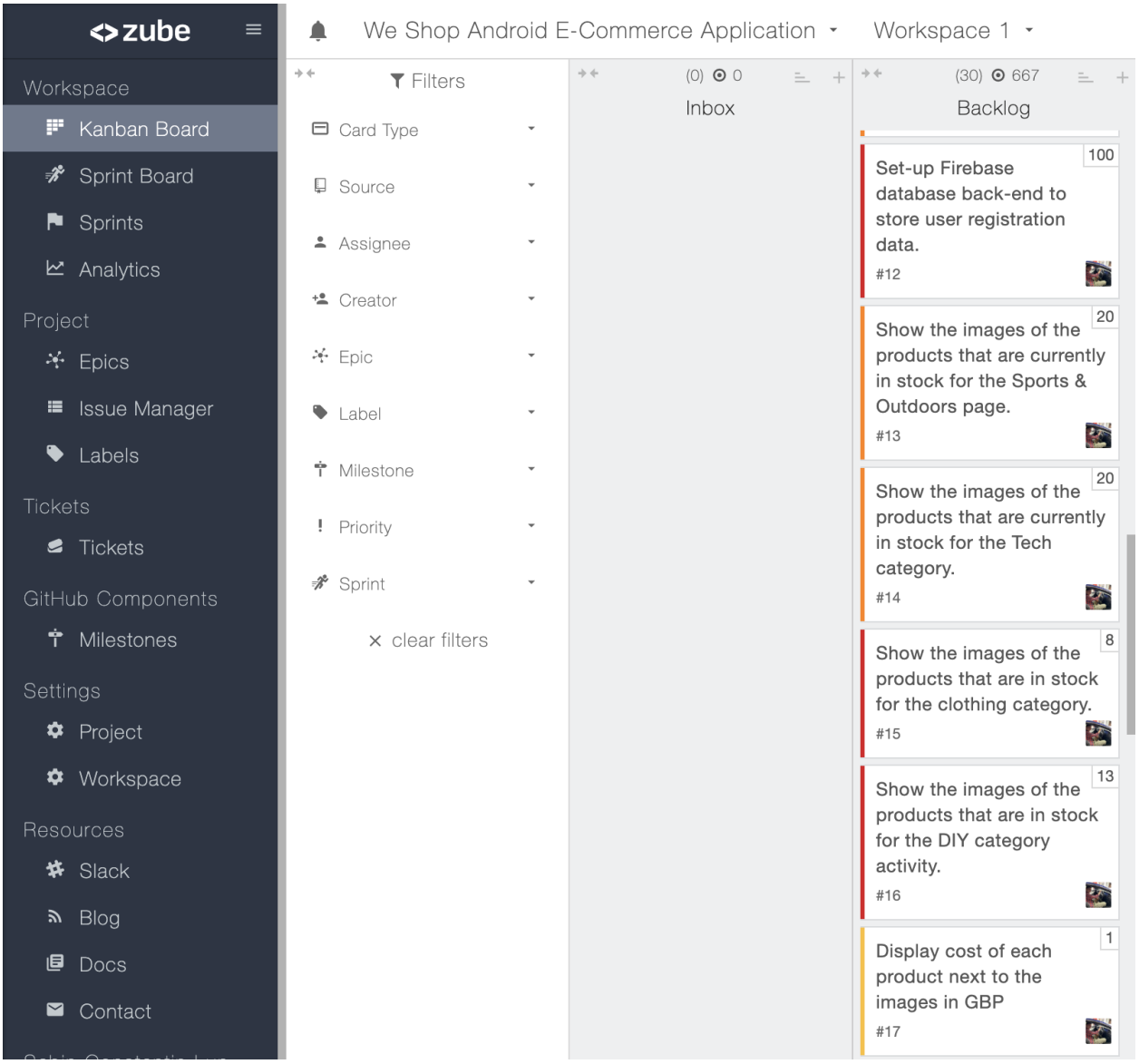
23. Users have the option to log out of their accounts if they wish to no longer be logged in.
24. Users have the option to pay with PayPal.
25. After the payment has been successful, users will receive an e-mail with an invoice.
26. Version control using **Git** will be used to track changes that will be made to different parts of the project. Branches will also be implemented that partitions each task, for example: “feature/implement-register-page” or “feature/implement-login-page”.
27. The login E-mail Address & Password will get validated to ensure no wrong data is entered.
28. Junit tests will be implemented that will test different aspects of the application, for example testing if the activities load as required and feeding in sample inputs for the registration fields to see if the validation is functioning correctly.

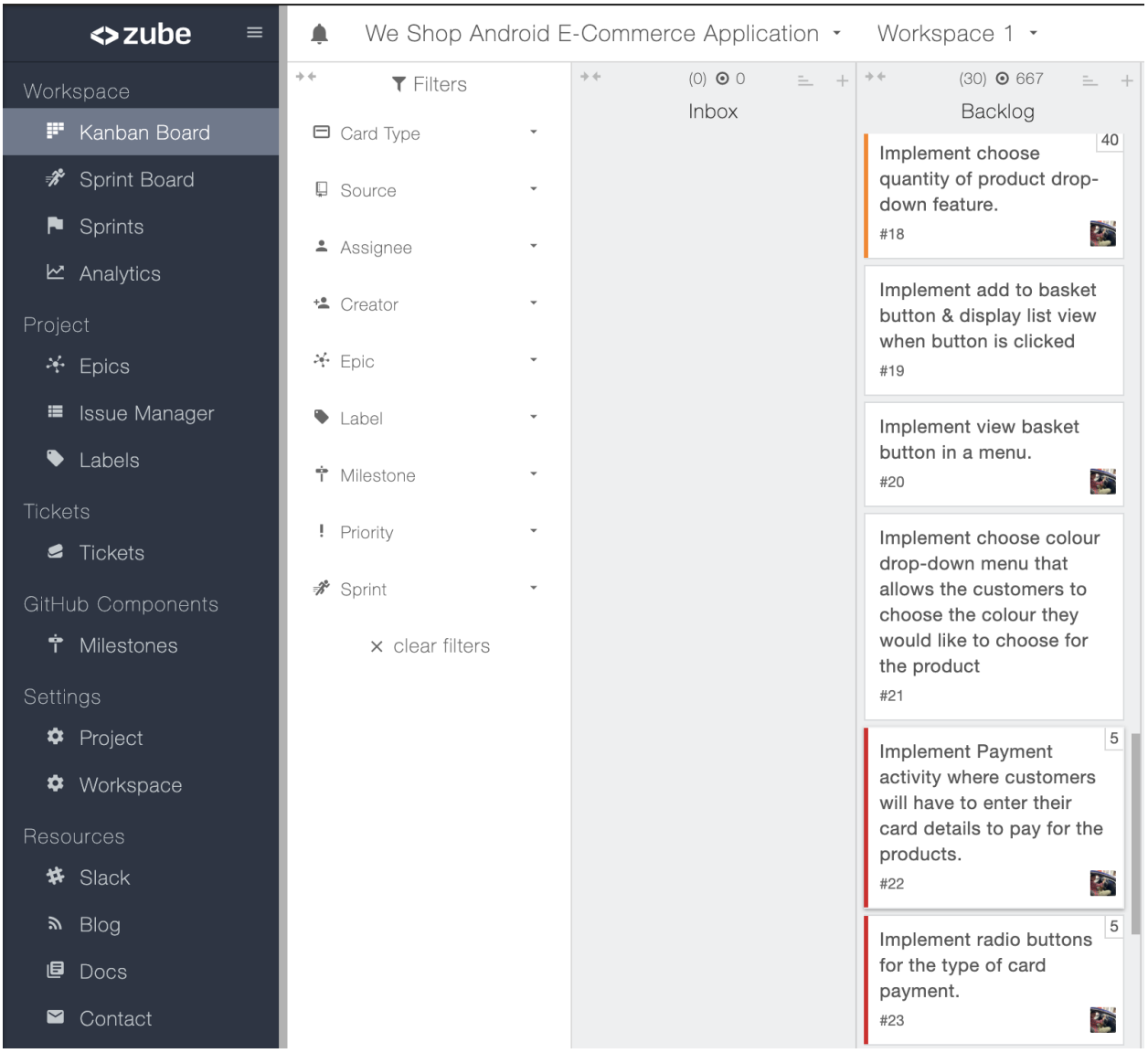
Product Backlog

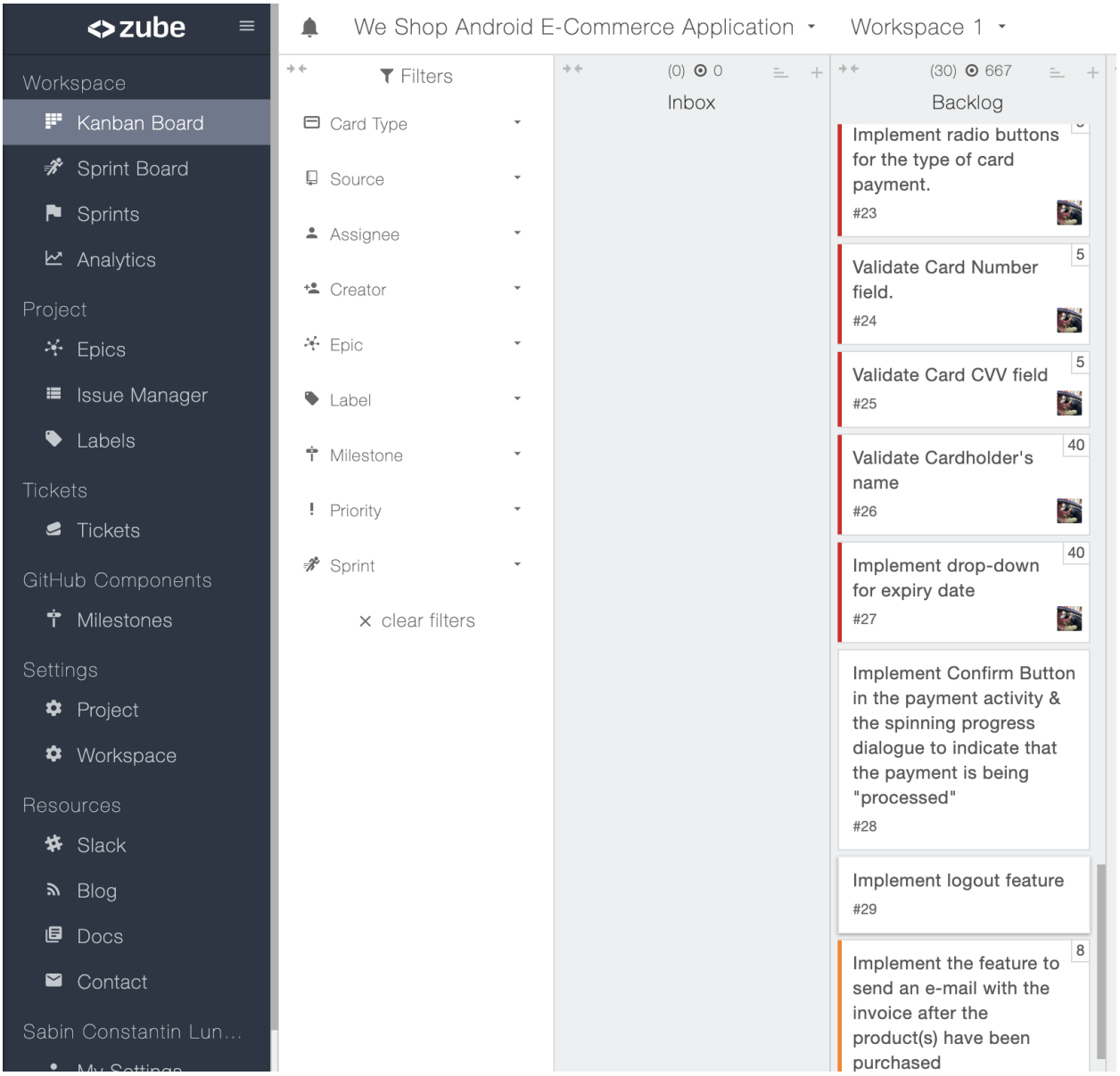


This product backlog shows the tasks that will be worked on in order. I will be assigning a sprint (time-frame) for each task in order after each one has been completed.

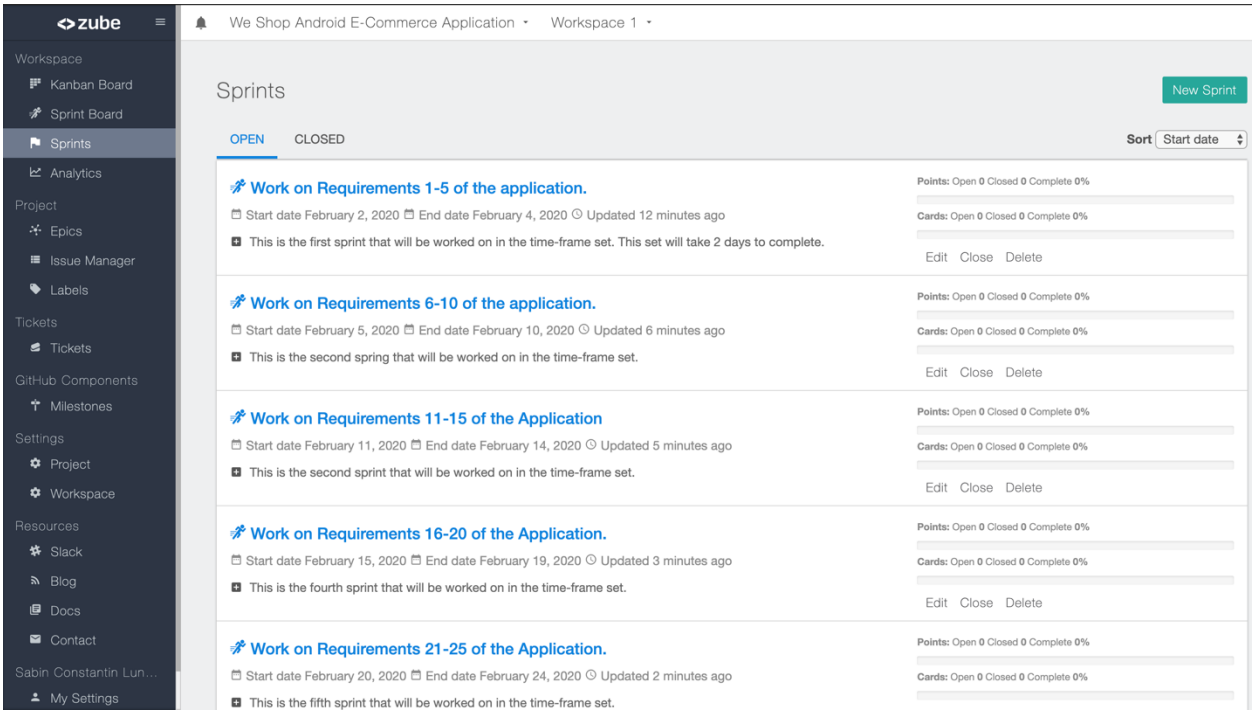

zube
We Shop Android E-Commerce Application
Workspace 1
Workspace
Kanban Board
Sprint Board
Sprints
Analytics
Project
Epics
Issue Manager
Labels
Tickets
Tickets
GitHub Components
Milestones
Settings
Project
Workspace
Resources
Slack
Blog
Docs
Contact
Sabin Constantin Lun...
Filters
Card Type
Source
Assignee
Creator
Epic
Label
Milestone
Priority
Sprint
clear filters
(0) 0
Inbox
(30) 667
Backlog
Validate Username field against: must not be empty, must contain numbers.
#7
40
Validate password field against: must contain numbers, special characters, characters, must not be left empty and must start with an upper case character to ensure that the strength level is high.
#8
40
Validate Email Address field against: must contain @ symbol and should not be left empty, ensure it is within the range of a-z, A-Z, 0-9 and contains characters from A-Z, 0-9 inclusively.
#9
13
Send notification to user after registering successfully.
#10
20
Login the user using their Email Address & Password.
#11
20


zube
We Shop Android E-Commerce Application
Workspace 1
Workspace
Kanban Board
Sprint Board
Sprints
Analytics
Project
Epics
Issue Manager
Labels
Tickets
Tickets
GitHub Components
Milestones
Settings
Project
Workspace
Resources
Slack
Blog
Docs
Contact
Filters
Card Type
Source
Assignee
Creator
Epic
Label
Milestone
Priority
Sprint
clear filters
(0) 0
Inbox
(30) 667
Backlog
100
Set-up Firebase database back-end to store user registration data.
#12
20
Show the images of the products that are currently in stock for the Sports & Outdoors page.
#13
20
Show the images of the products that are currently in stock for the Tech category.
#14
8
Show the images of the products that are in stock for the clothing category.
#15
13
Show the images of the products that are in stock for the DIY category activity.
#16
1
Display cost of each product next to the images in GBP
#17


zube
We Shop Android E-Commerce Application
Workspace 1
Workspace
Kanban Board
Sprint Board
Sprints
Analytics
Project
Epics
Issue Manager
Labels
Tickets
Tickets
GitHub Components
Milestones
Settings
Project
Workspace
Resources
Slack
Blog
Docs
Contact
Filters
Card Type
Source
Assignee
Creator
Epic
Label
Milestone
Priority
Sprint
clear filters
(0) 0
Inbox
(30) 667
Backlog
Implement choose quantity of product drop-down feature.
40
#18
Implement add to basket button & display list view when button is clicked
#19
Implement view basket button in a menu.
#20
Implement choose colour drop-down menu that allows the customers to choose the colour they would like to choose for the product
#21
Implement Payment activity where customers will have to enter their card details to pay for the products.
5
#22
Implement radio buttons for the type of card payment.
5
#23


zube
We Shop Android E-Commerce Application
Workspace 1
Workspace
Kanban Board
Sprint Board
Sprints
Analytics
Project
Epics
Issue Manager
Labels
Tickets
Tickets
GitHub Components
Milestones
Settings
Project
Workspace
Resources
Slack
Blog
Docs
Contact
Sabin Constantin Lun...
Filters
Card Type
Source
Assignee
Creator
Epic
Label
Milestone
Priority
Sprint
clear filters
(0) 0
Inbox
(30) 667
Backlog
Implement radio buttons for the type of card payment.
#23
Validate Card Number field.
5
#24
Validate Card CVV field
5
#25
Validate Cardholder's name
40
#26
Implement drop-down for expiry date
40
#27
Implement Confirm Button in the payment activity & the spinning progress dialogue to indicate that the payment is being "processed"
#28
Implement logout feature
#29
Implement the feature to send an e-mail with the invoice after the product(s) have been purchased
8

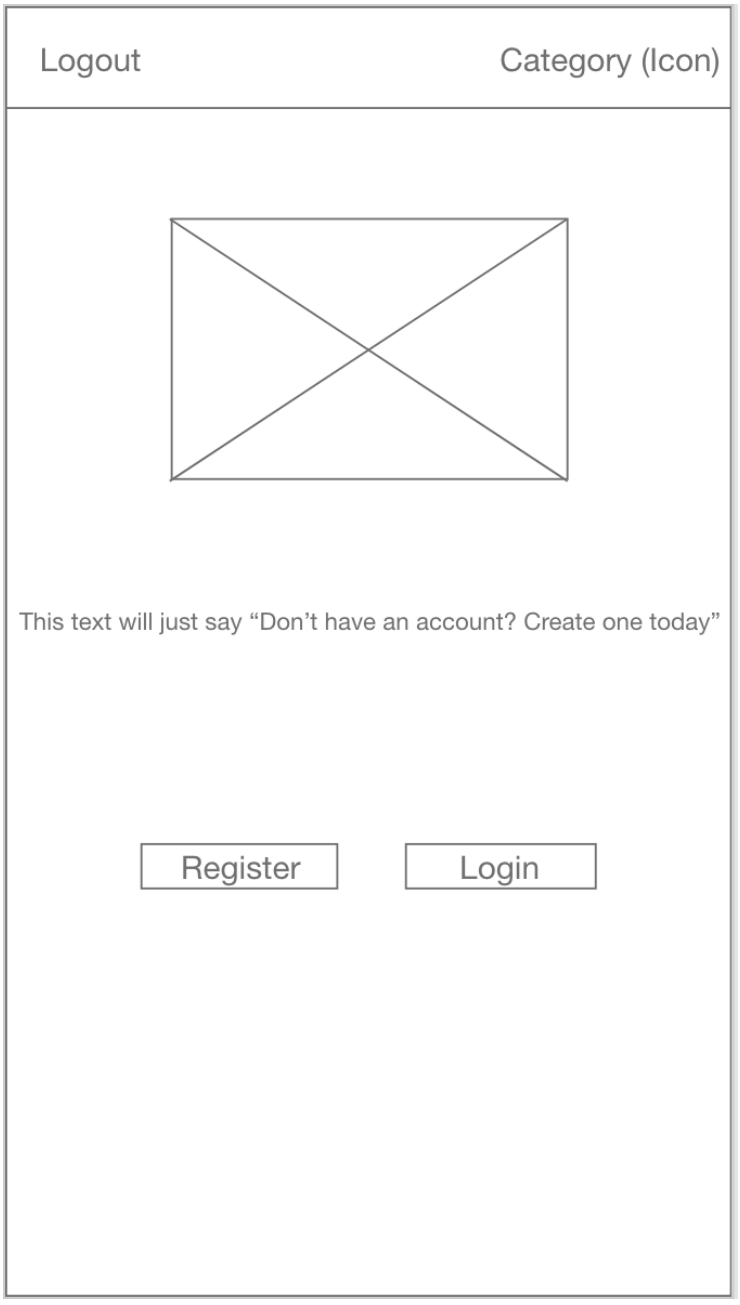
Sprint Boards



These are the sprints that will be worked one by one

Note: * the squared boxes in the User Interface design represent drop-down menus, **NOT** check boxes because the Adobe XD CC software that I use for the design doesn't have a drop-down box option, only a square shape.

Design – Main Activity User Interface Prototype

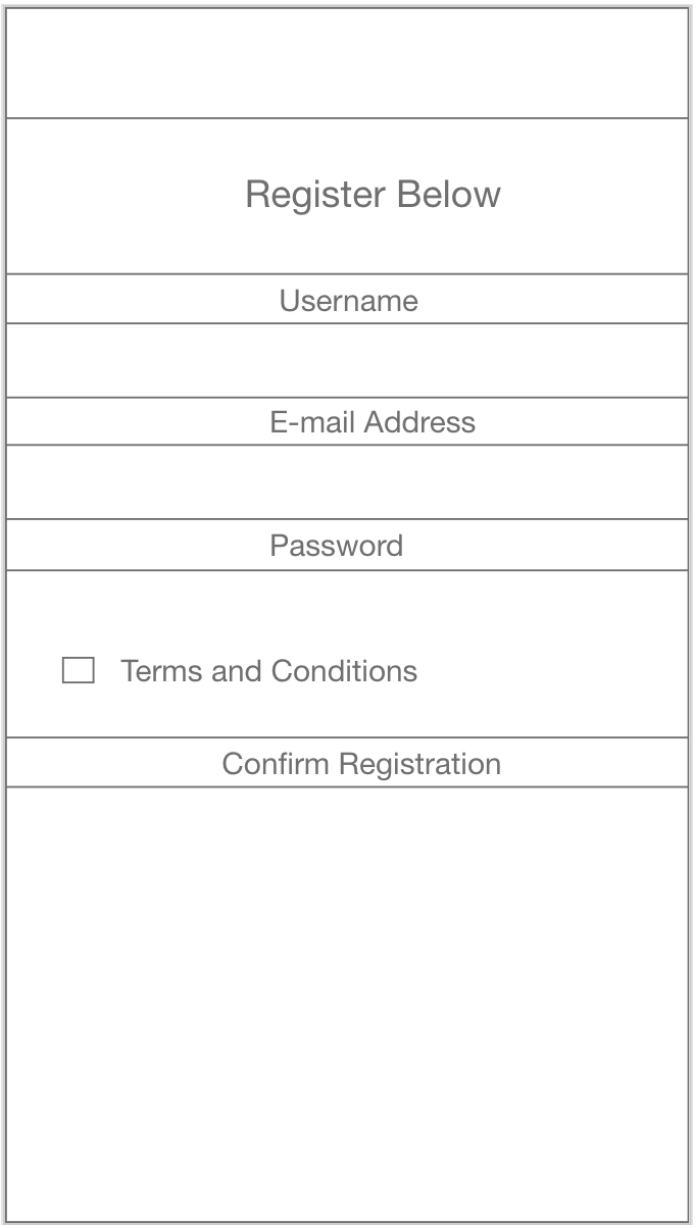


This is how the main activity (homepage) will look like. The top navigation bar will have the drop-down menu with the categories that the users can choose from when they want to browse for products.

There will be an image in the middle that will welcome the users to the application and the text below the image will just say “Don’t have an account? Register today”

The button on the left is the Register button which will allow users to register an account. The button next to it is the Login button where users can log in with their account.

Register Activity User Interface Prototype

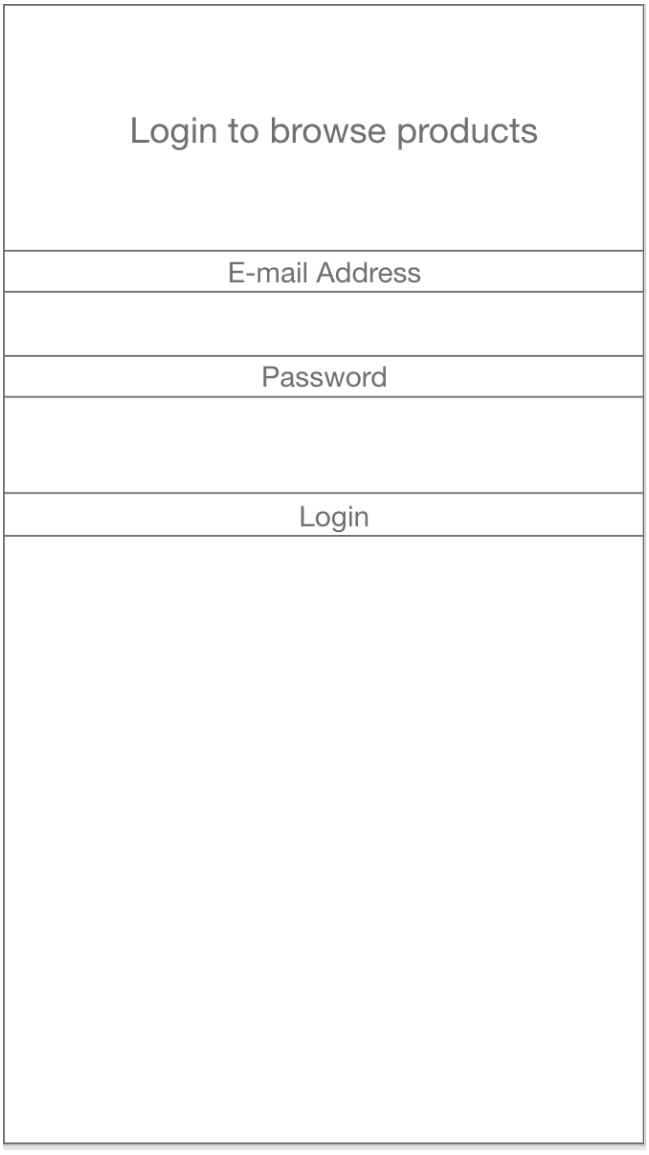


This is how the Register activity will look like. The horizontal rectangle at the top will be where users can choose the category of products to browse on, I chose to implement this in the design just to keep it **consistent** throughout all the activities.

Once the users are presented with the registration page, they will be prompted to enter their desired Username, E-mail Address and password. They must tick the terms and conditions box before clicking the register button.

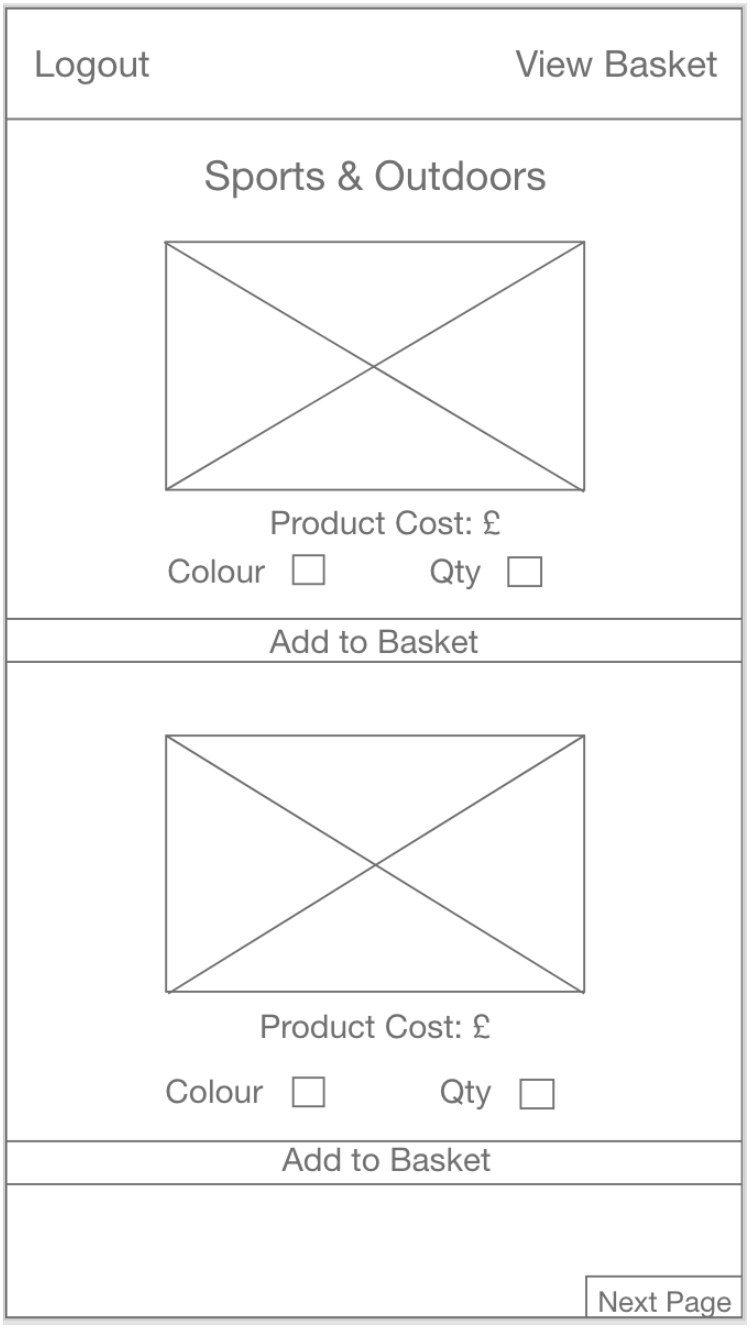
All of the inputs will be validated to ensure that the correct data is entered only. The validation that will be implemented is stated in the requirements specification.

Login User Interface Prototype



After registration, users will have to log in with their E-mail Address and password.

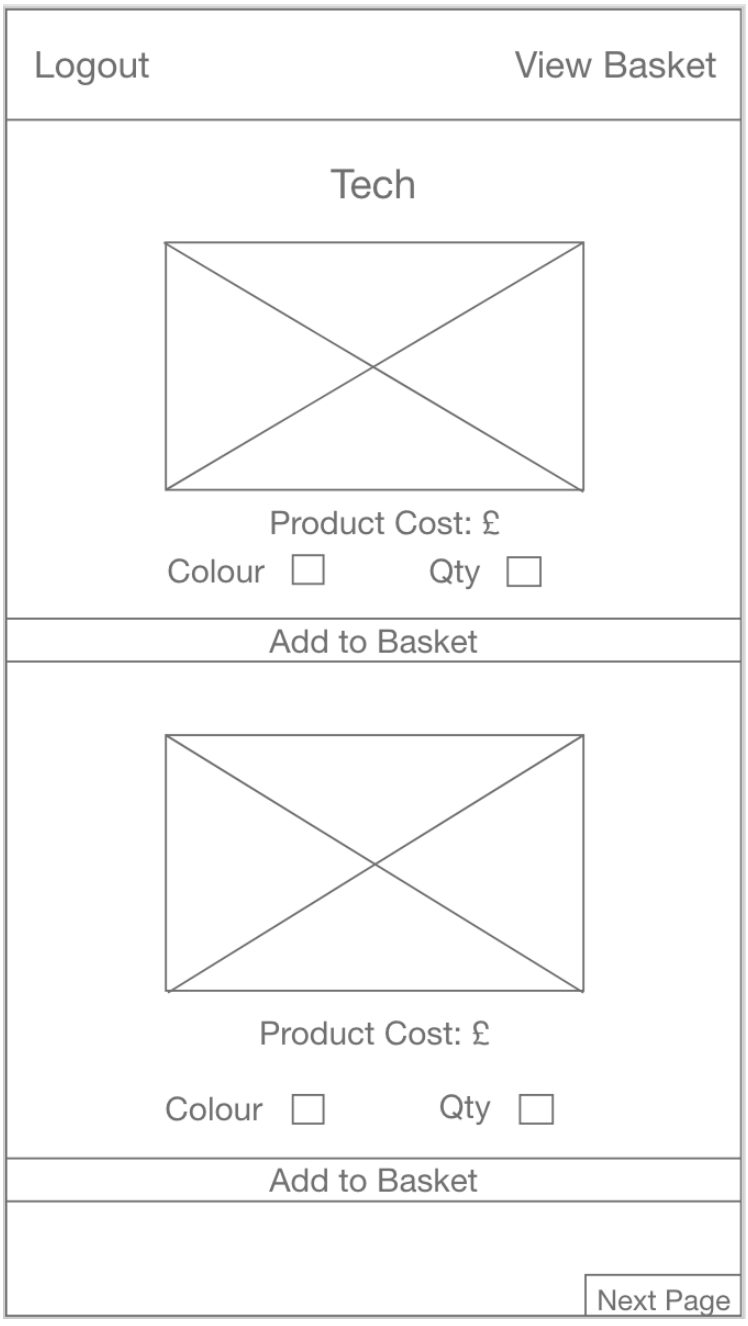
Sports & Outdoors User Interface Prototype



This is how the Sports & Outdoors page will look like. At the top of the activity there will be a logout button incase the user no longer wants to be logged in, if the button is clicked then the users will get sent back to the home page.

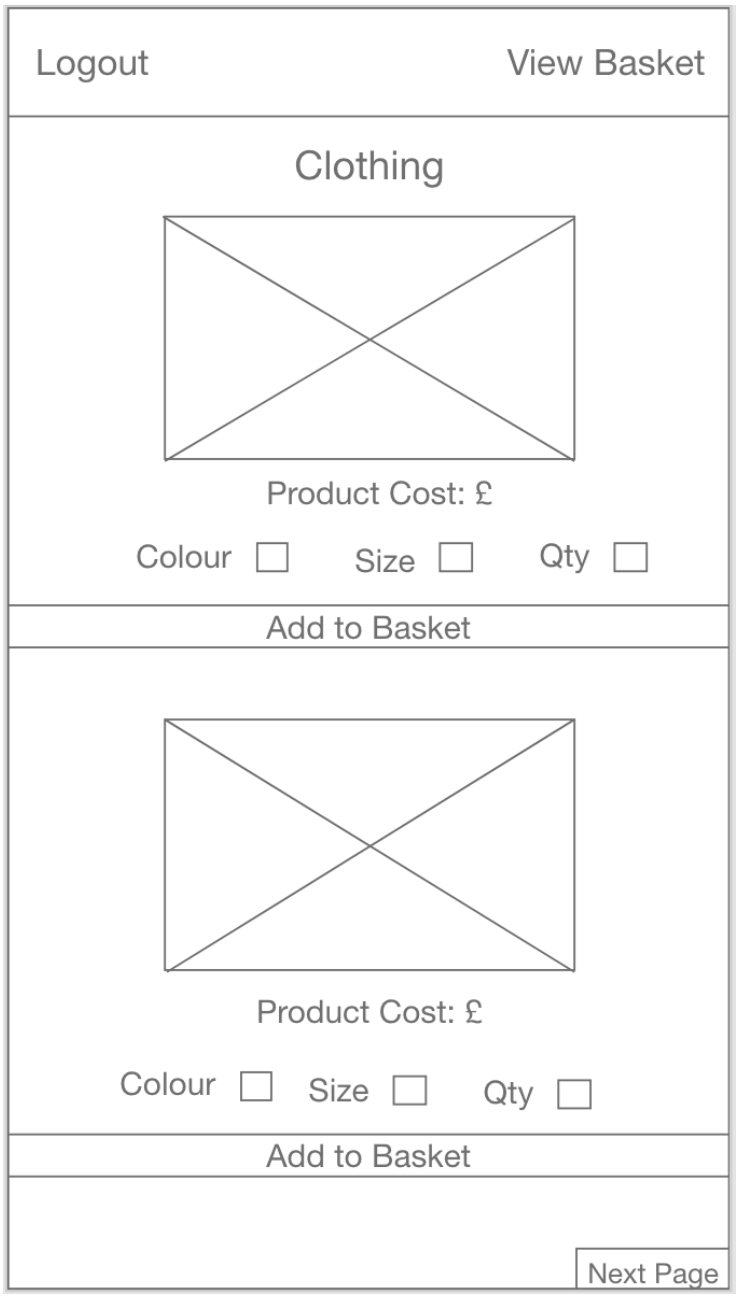
The top of the activity also has a View Basket button that when it is clicked, it will show the products that the user has added to the basket. The Sports & Outdoors category will have 2 pages of products, so there is no need to show another User Interface with the same design.

Tech User Interface Prototype



The User Interface for the **Tech** category will look the same as the Sports & Outdoors.

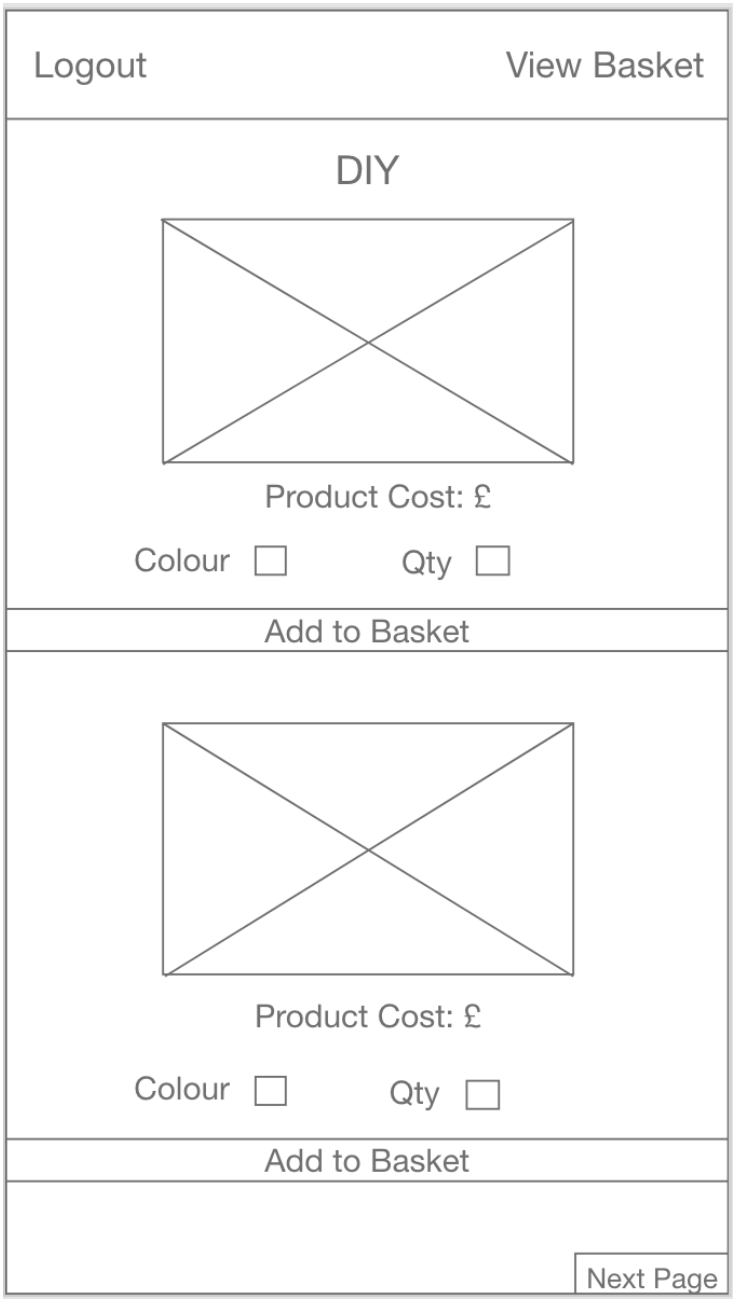
Clothing User Interface Prototype



This is how the Clothing category will look like. It is the exact same as the others, however there will be a size-drop down menu that will allow users to choose the size they wish for the product.

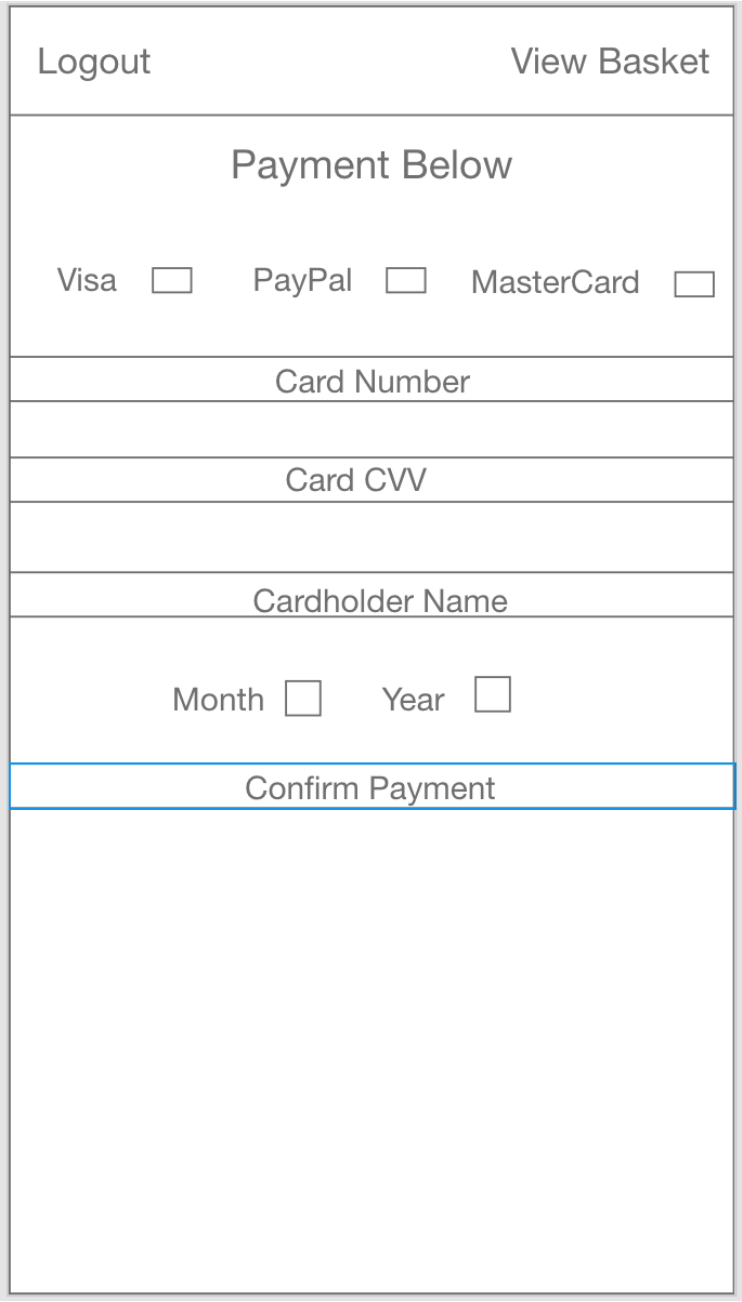
Users will choose the color and quantity of the product they would like, then they can add it to the basket.

DIY User Interface Prototype



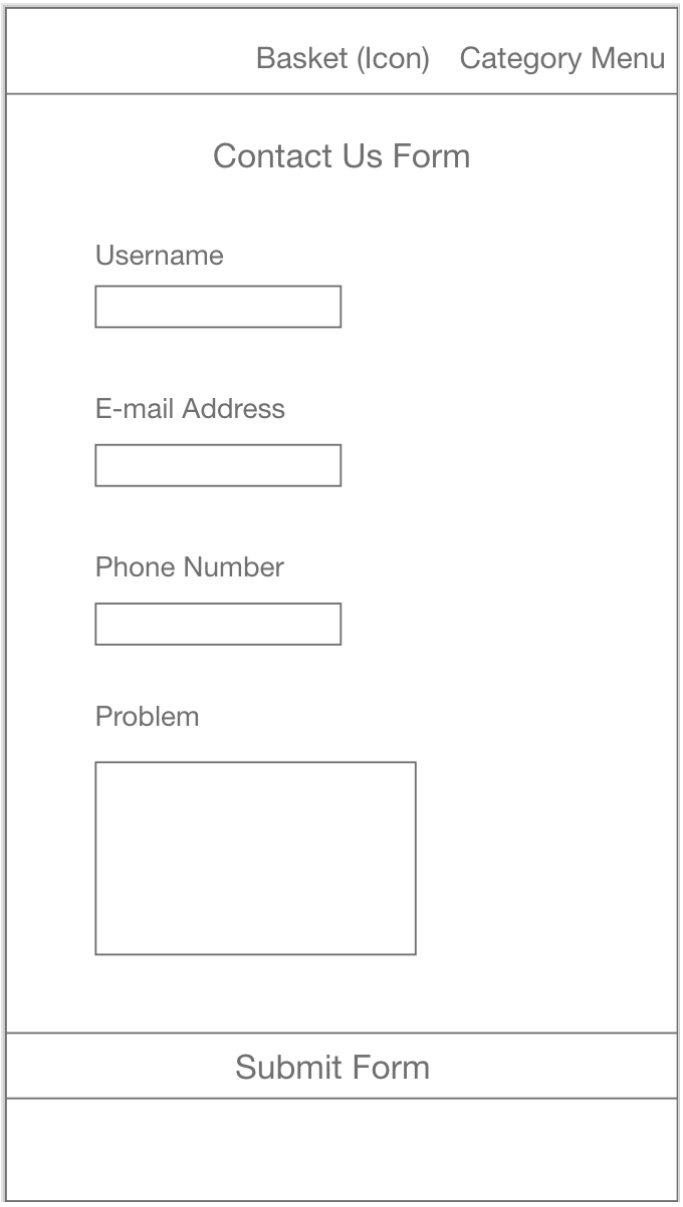
Same UI as the other categories.

Payment User Interface Prototype



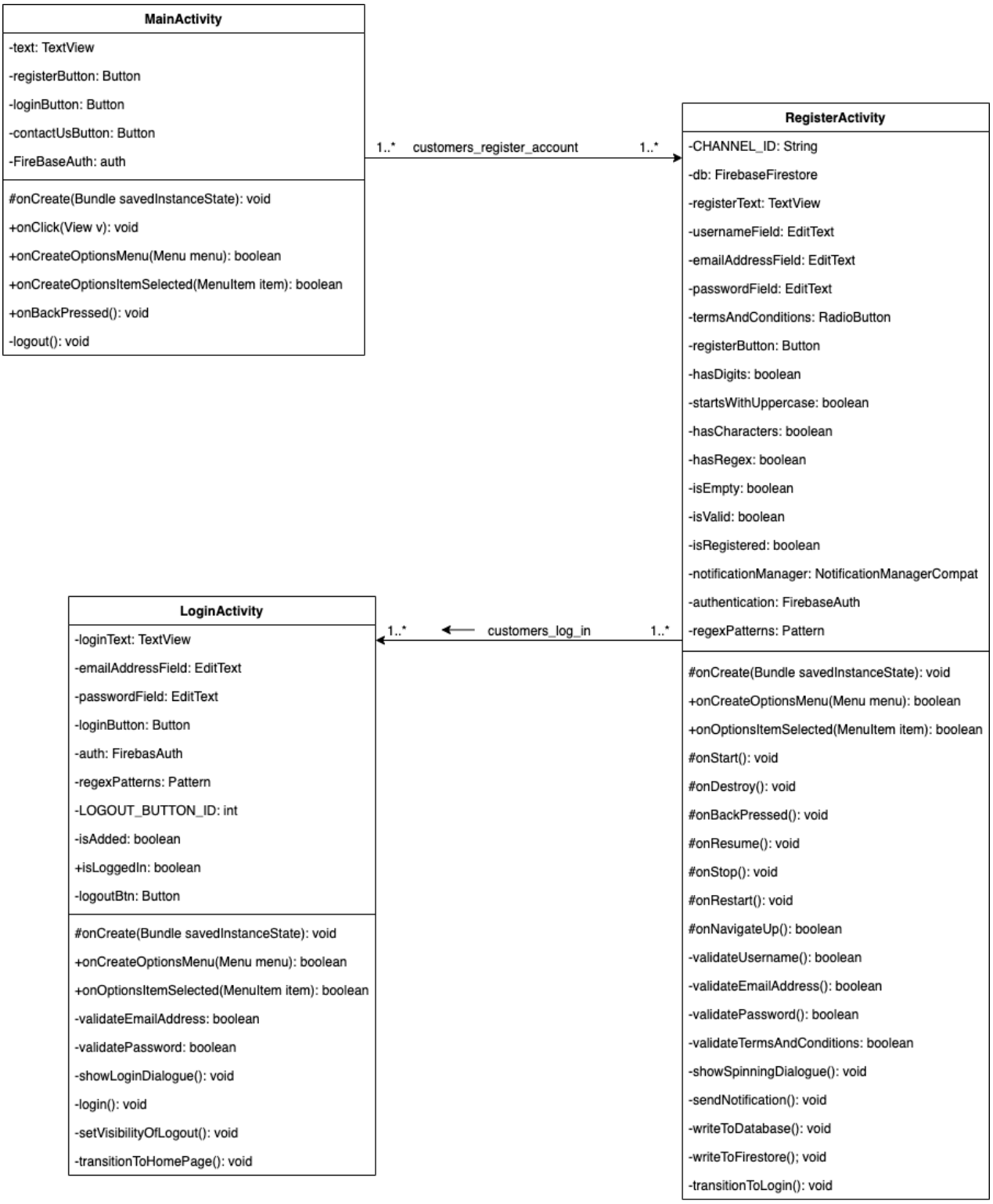
This is the payment form that the customers will get presented with when they are ready to purchase the products. They will have to choose which type of card they wish to pay with, then they are required to fill out their details. The expiry date will be a drop-down menu.

Submit Complaint Form User Interface Design



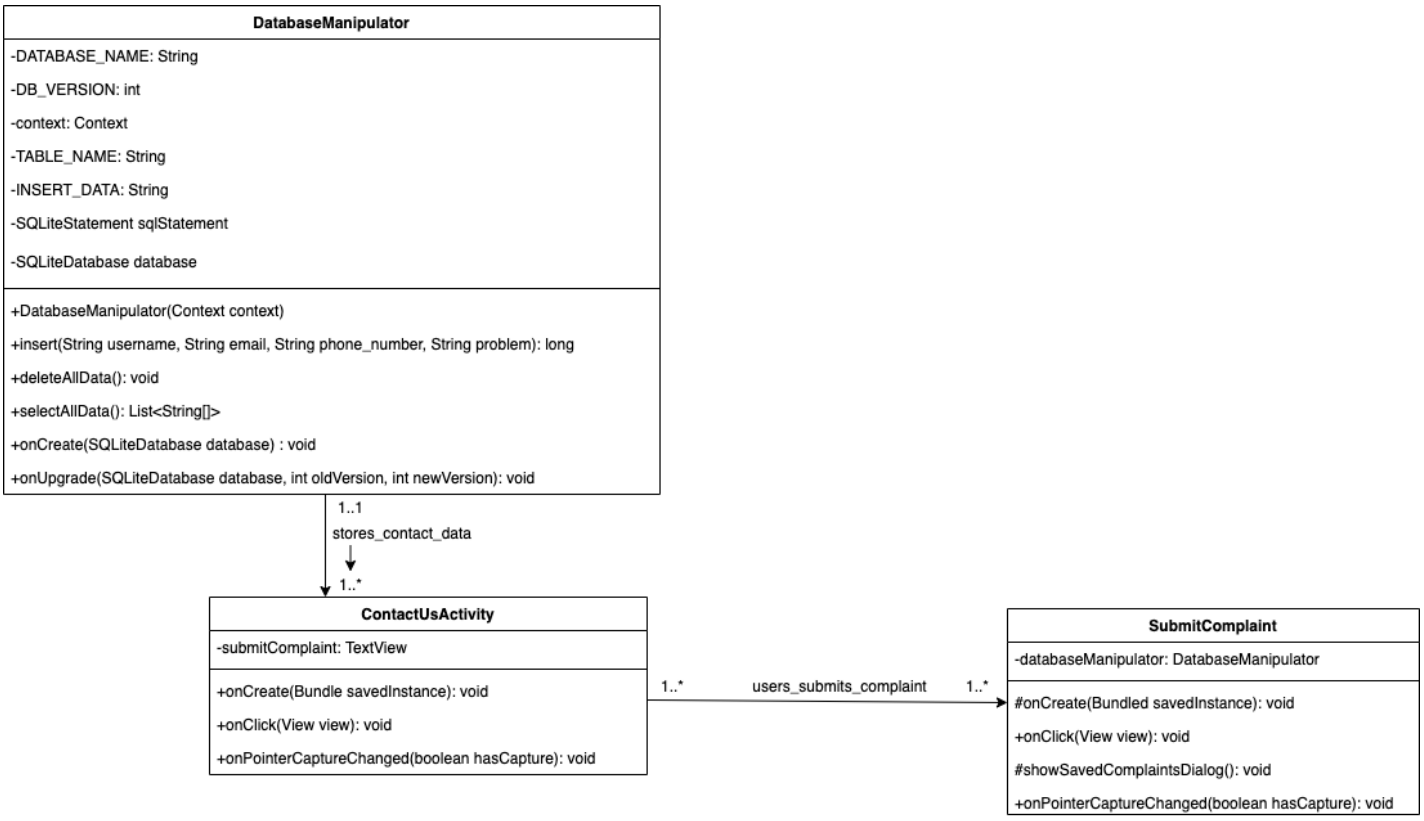
This is how the Contact Us form will look like. The users are required to fill out the following fields in order to successfully submit the form. An SQL database will be used to store the details of the complaints and based on the complaints I will be taking into consideration what needs to be improved for future updates.

Register & Login Class Diagram



Contact Us Class Diagram (Back-End)

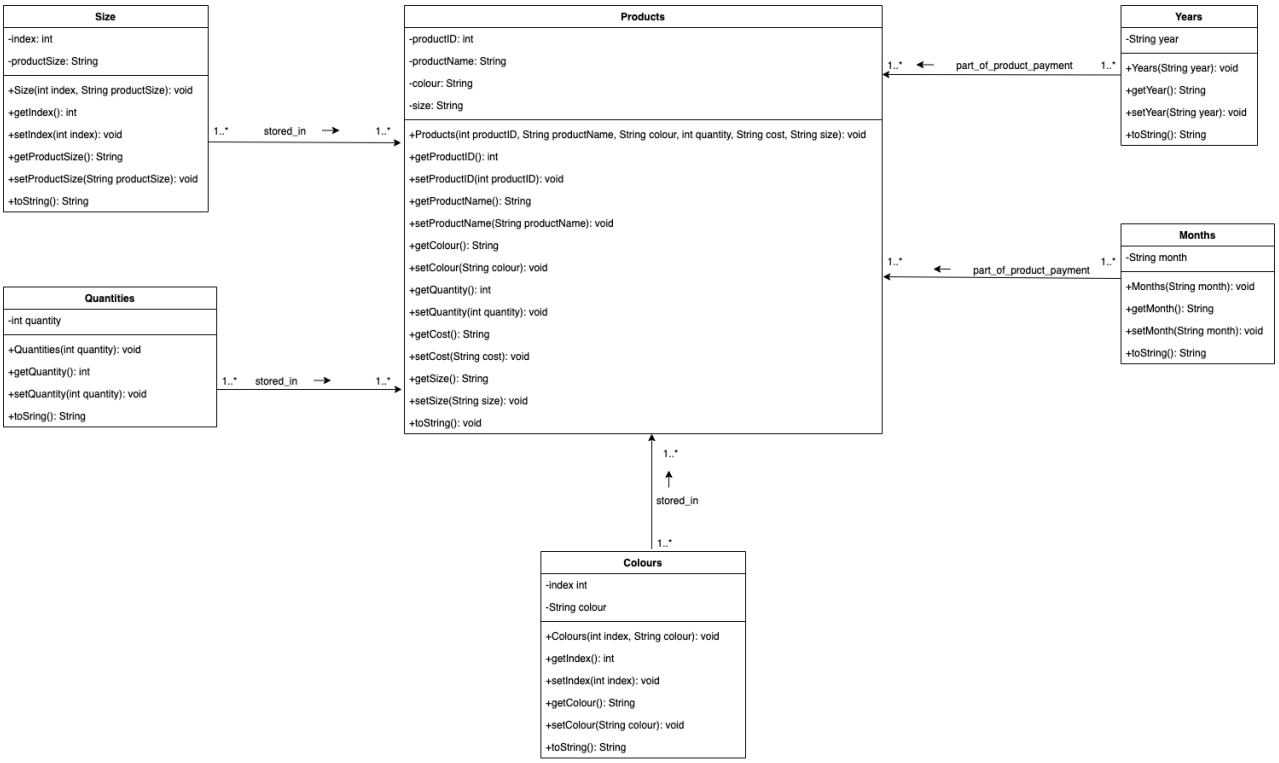
#Class Diagram 1



The Class Diagram shows how the database manipulator (SQLite) is used as a back-end service to store the complaints that the customer's might/will make if they encounter any issues on the application.

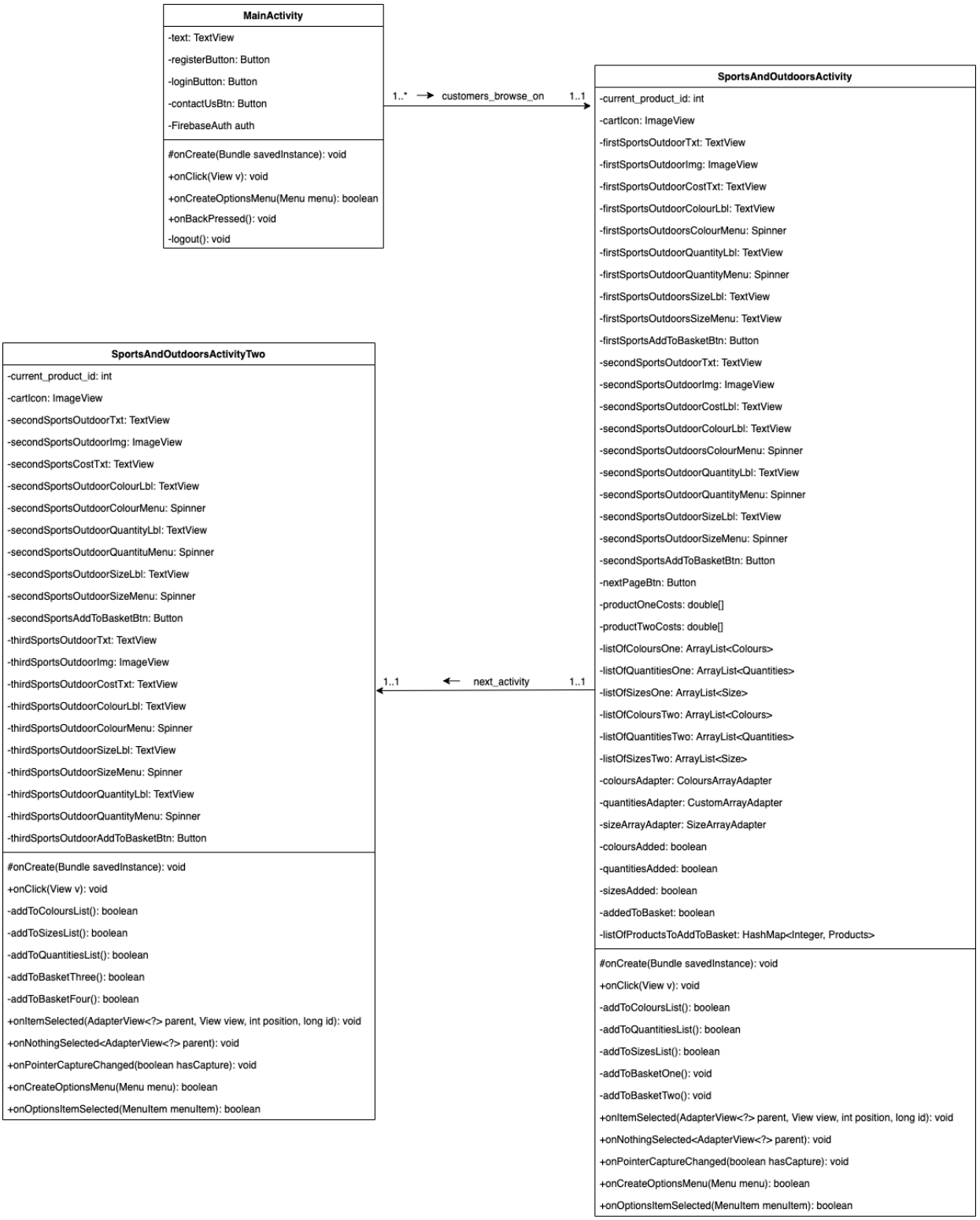
One back-end service will be used to store contact data that the customer submits. Many customers can make many complaints and the data will get stored in an SQL database table. The structure of the table will be shown further in the design.

Business Layer Class Diagram



This Class Diagram shows the business objects that will be used in this project. Each product will have a colors, quantities and size instance that will store the required color, quantity and size for the product.

Application Layer Class Diagram – Sports and Outdoors



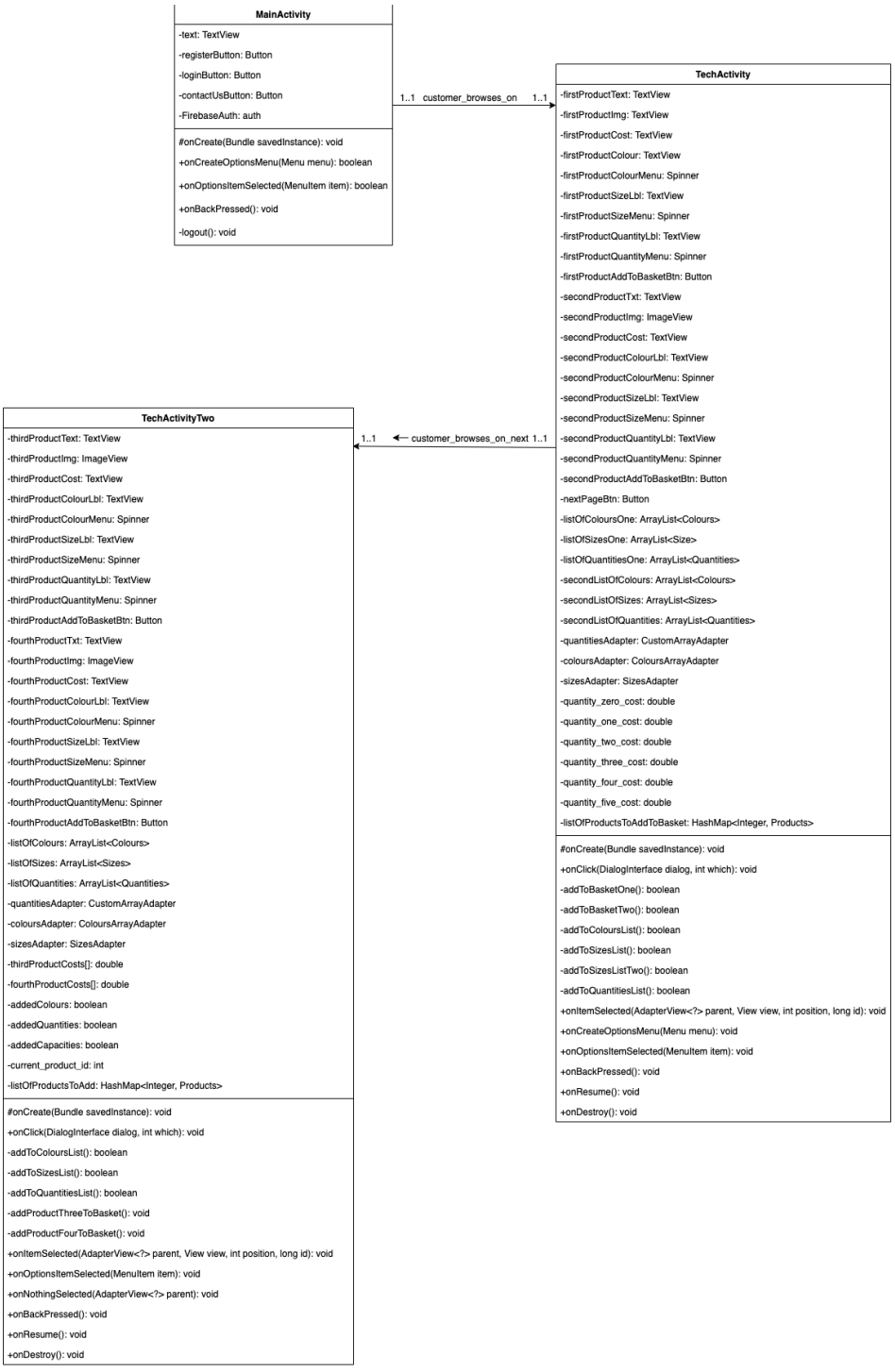
The class diagram above shows the relationship between the homepage (Main Activity) and the sports and outdoors activity.

The reason why the class diagrams are being split up into separate ones is because if I make one with all the classes it would be too big and would be very hard to read.

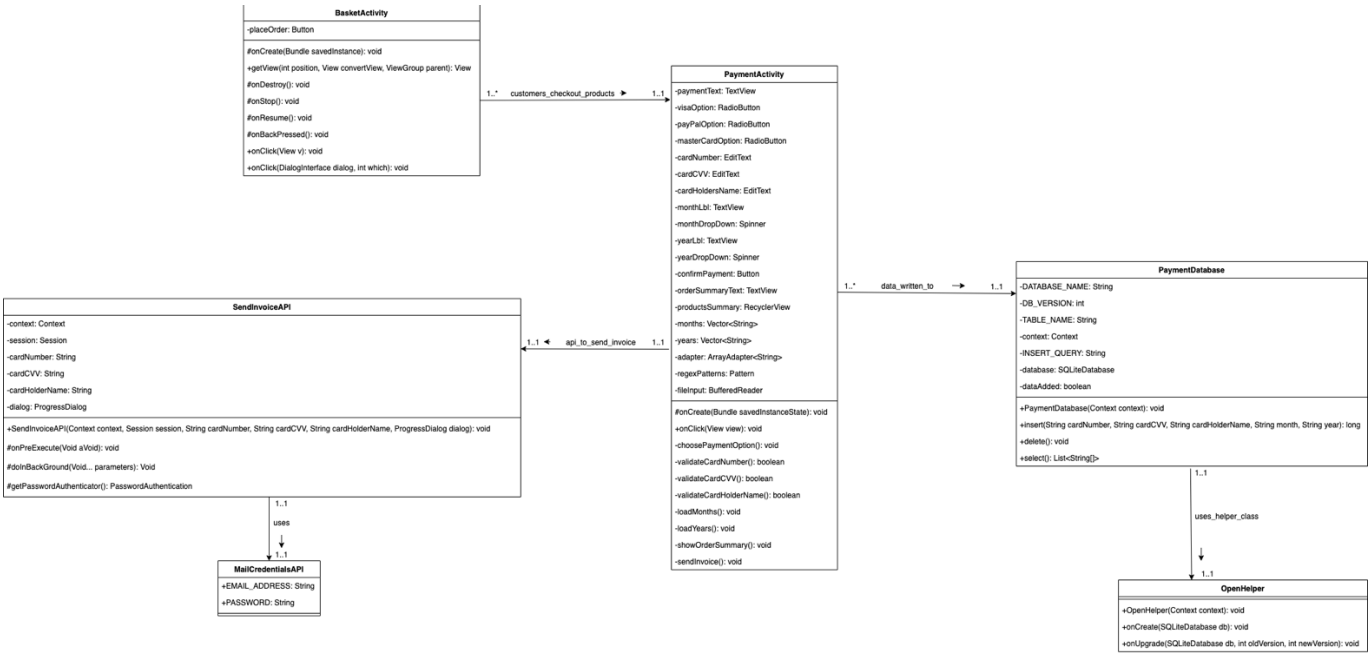
The diagram shows that many customers on the homepage can browse the sports and outdoor activity, as well as the second page.

Sabin Constantin Lungu
Matriculation Number: 40397517

Application Layer Class Diagram – Tech Activity



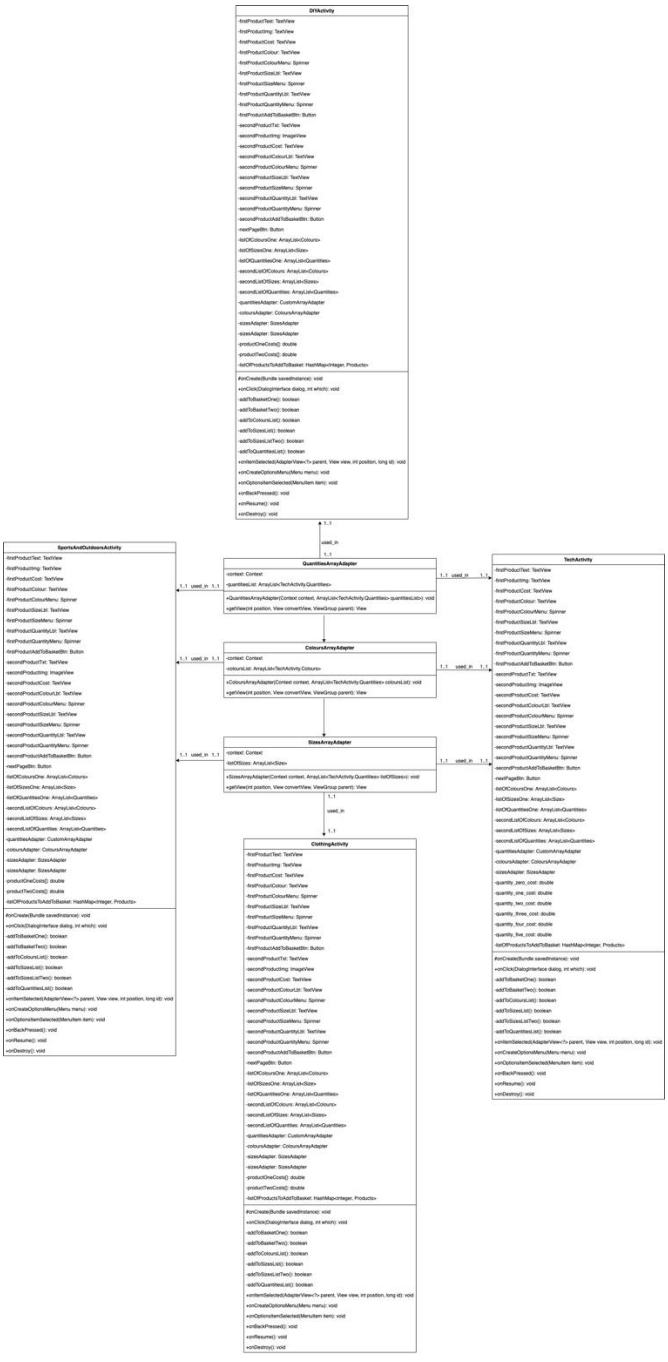
Basket Activity & Payment Activity Class Diagram – Data Layer



The class diagram is a bit hard to see, it might need a bit of zooming in. It shows how the Basket Activity class is related to the Payment Activity class that enables many customers to purchase the products they have chosen, then their payment information will get written to an SQL database to be stored in.

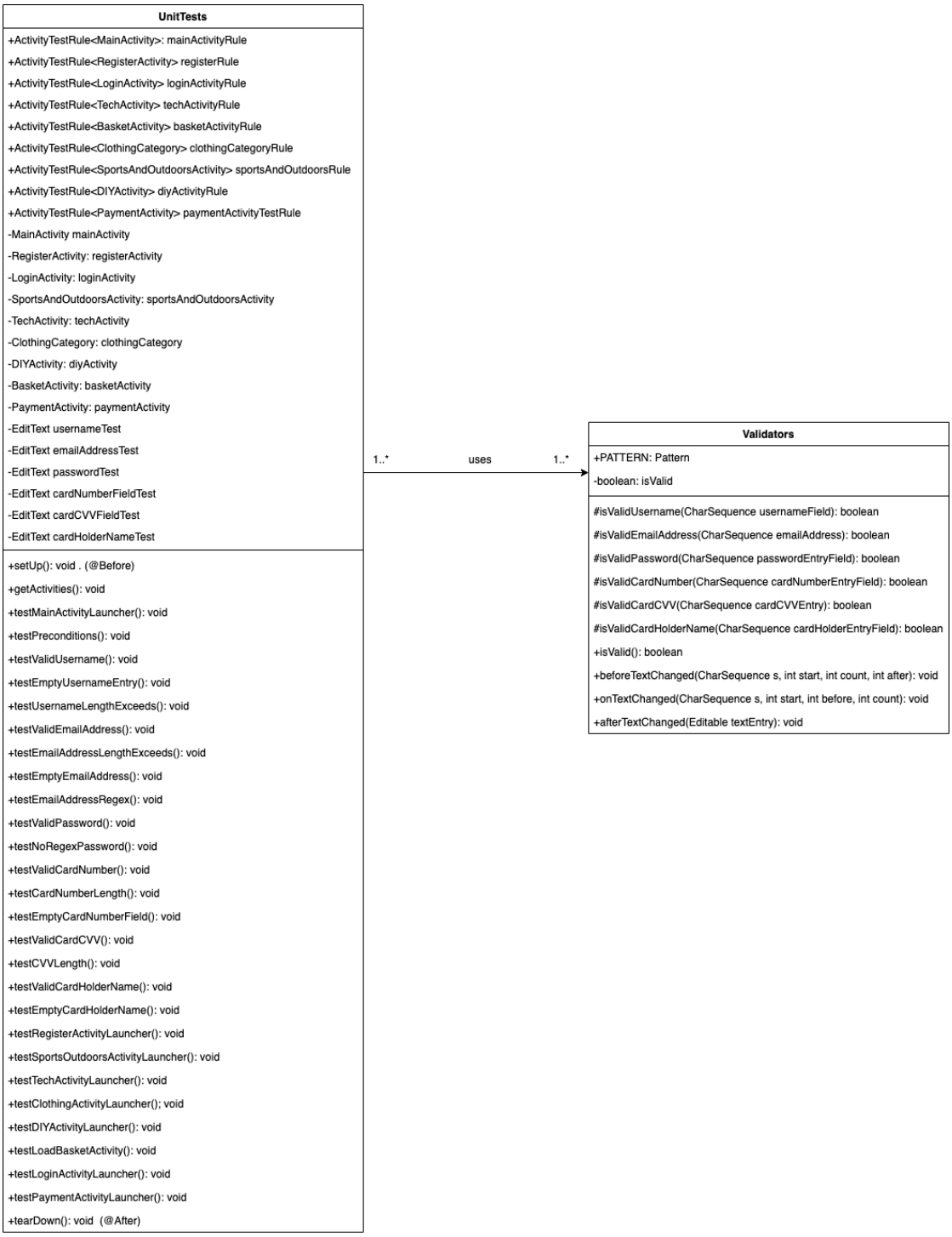
At the same time an e-mail with an invoice will be sent to them, the invoice will contain their payment details and a message saying that the order has been confirmed.

Adapters Class Diagram – Business Layer 2

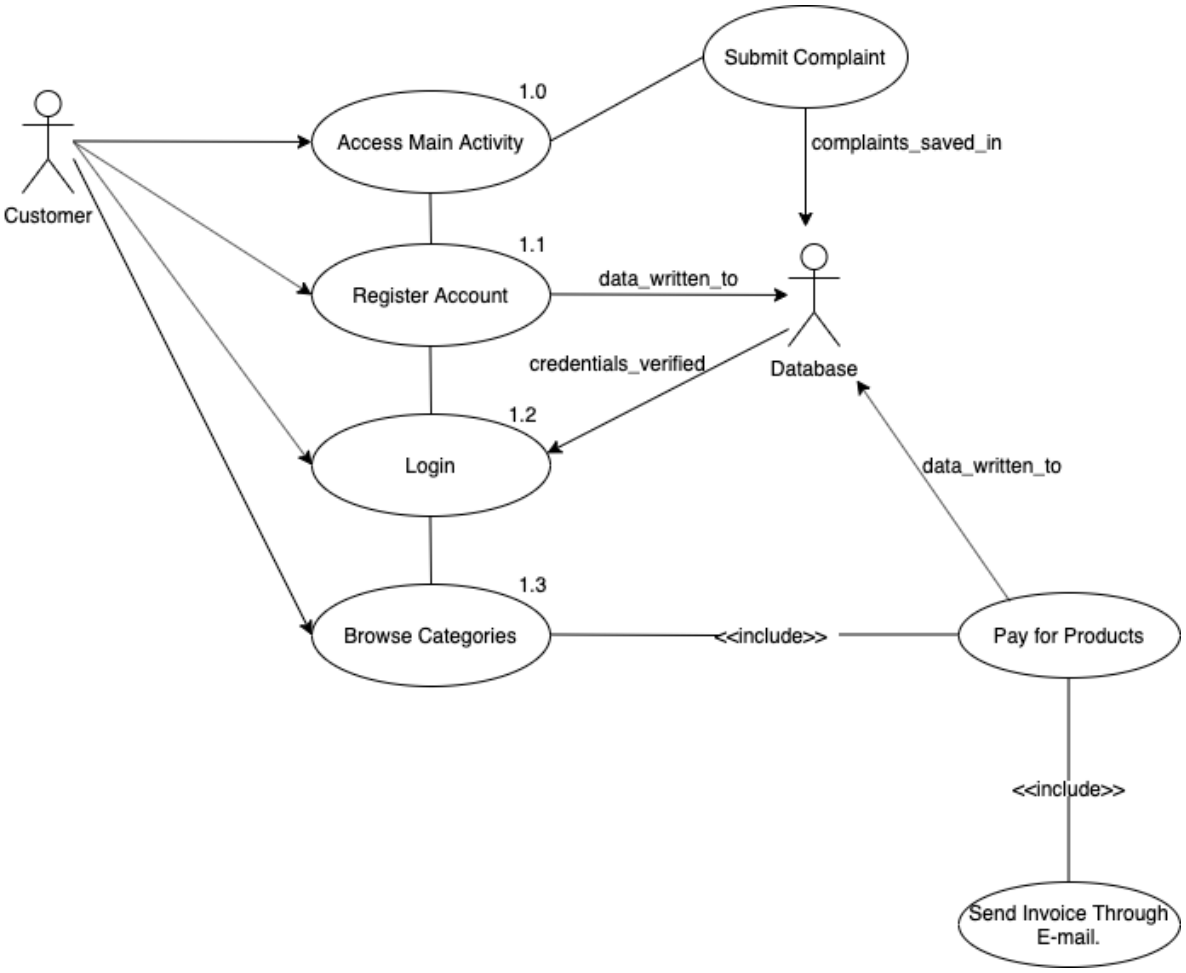


This diagram might need a little bit of zooming in to see it clearly.

Unit Tests Class Diagram



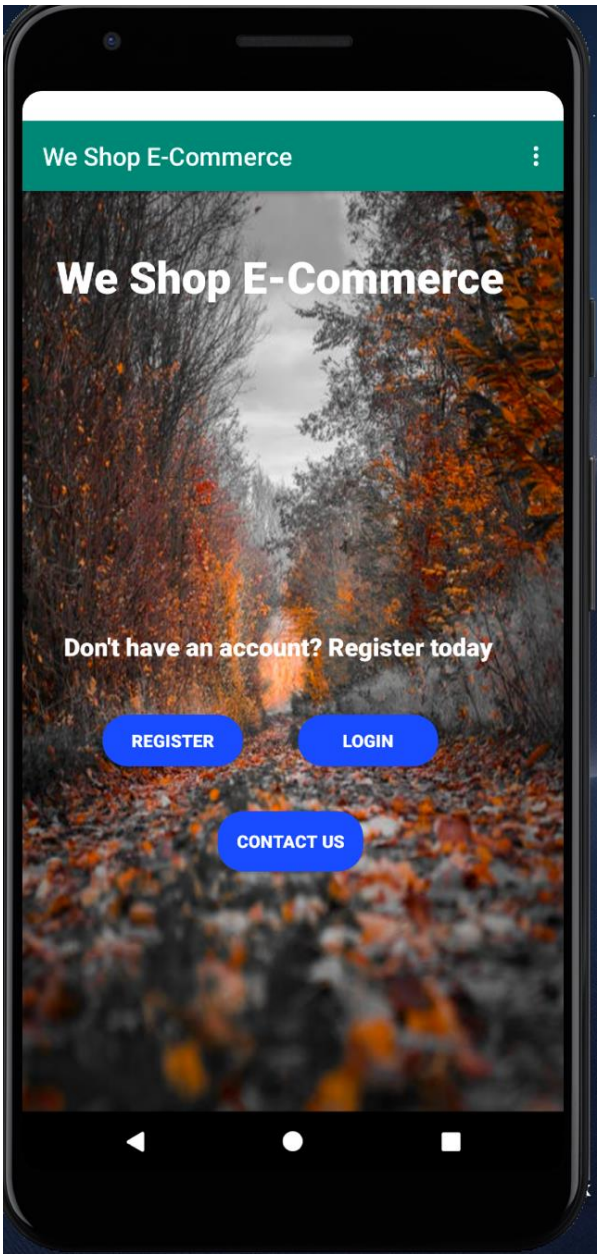
Use Case Diagram



The Use Case diagram above shows how the customers using the application will interact with it and how the back-end database is used between each process.

Application Homepage Implementation

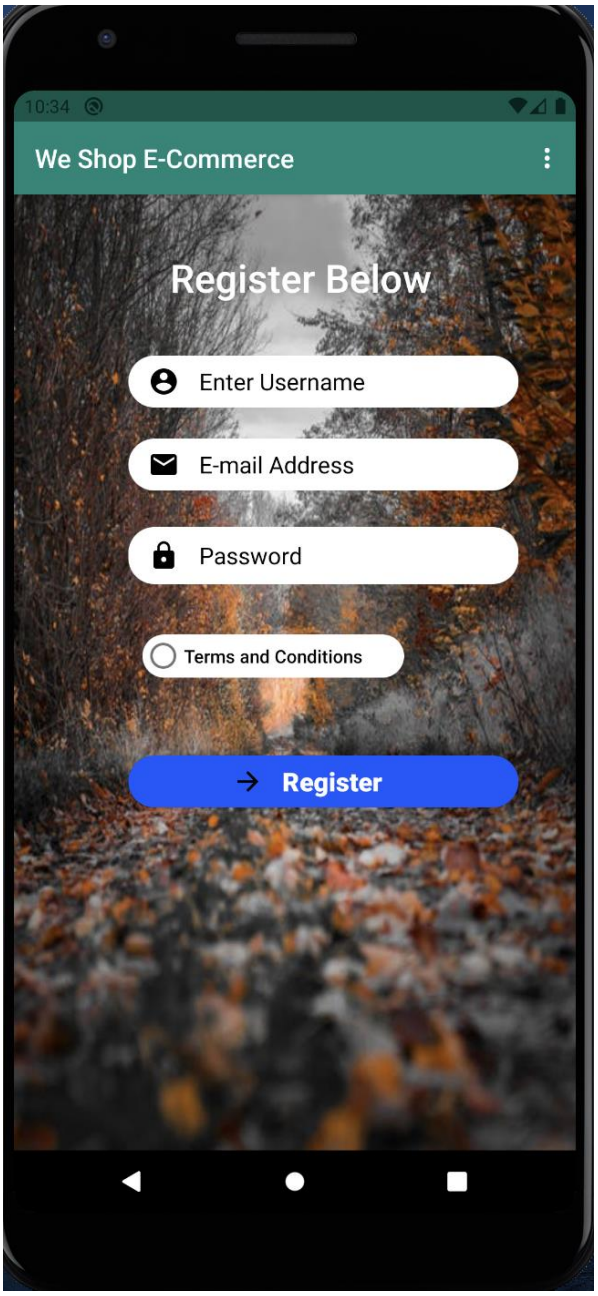
 # Image 1 – Main Activity



This is how the Homepage (Main Activity) looks. It has a Register Button, Login Button and a drop-down menu on the action bar that allows users to choose a category to browse on. User's register first before logging in. The Implementation of the Homepage matches the design.

Register Activity Implementation

 # Image 2 – Register Activity

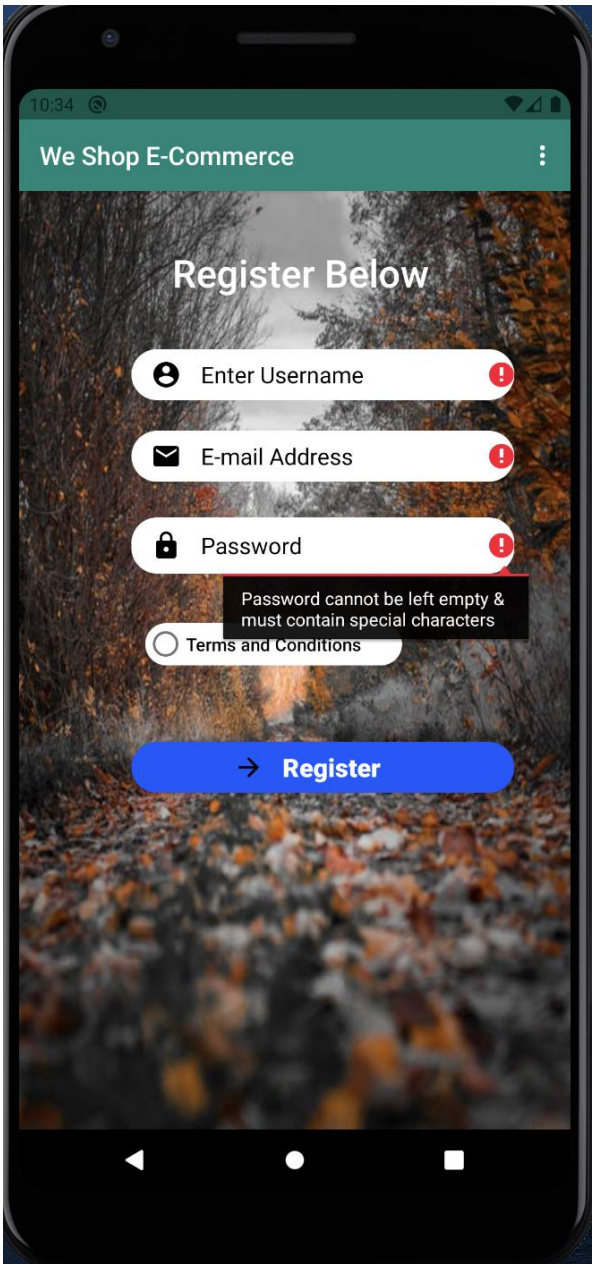


This is the Implementation of the Register Activity. Before users can register an account, they must fill out the fields shown in the activity, if they leave the fields blank or enter incorrect data then they will get presented with errors as shown in Image 3 below.

After all the fields are entered will correct data, the registration data will get stored in a Firebase Database.

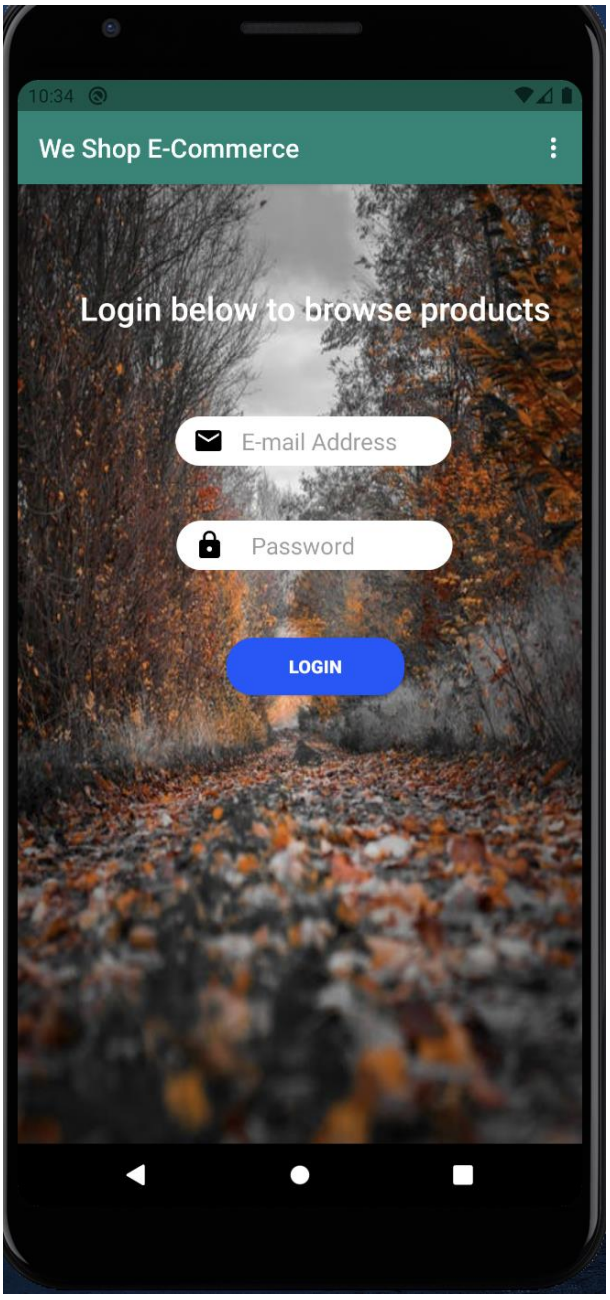
Register Activity Validation

 # Image 3 – Validation of Register Activity



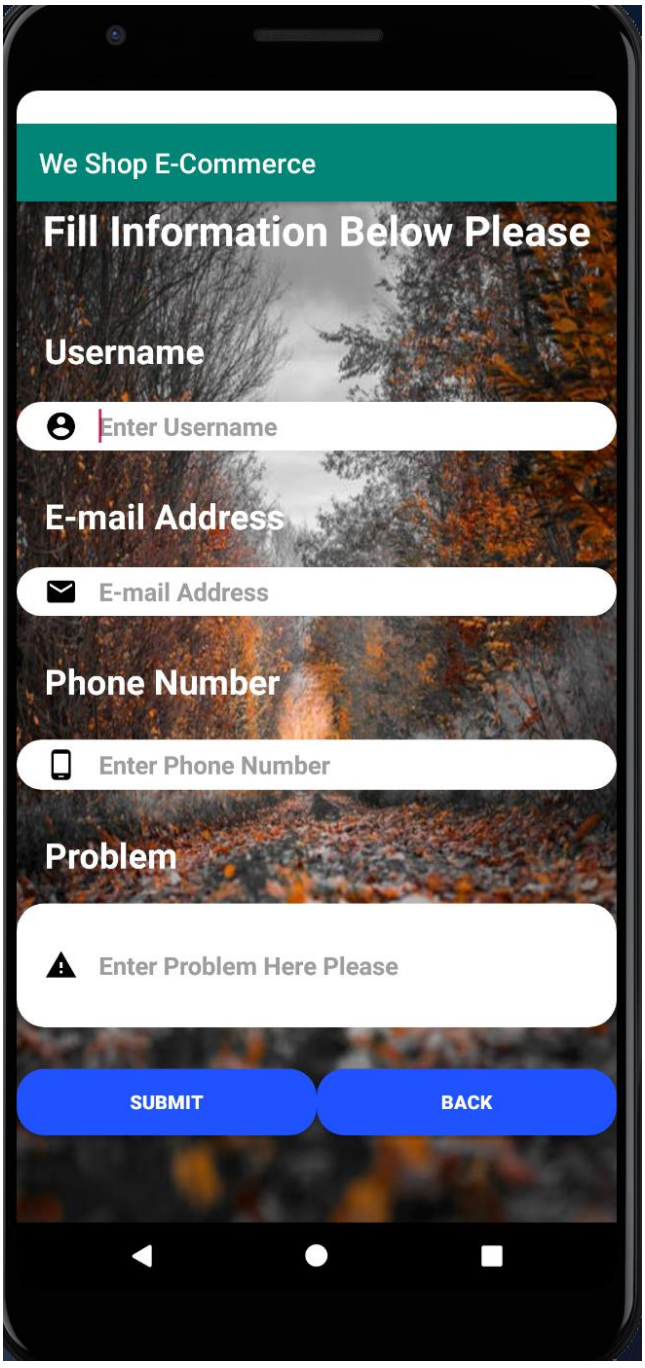
Login Activity

 # Image 4 – Login Activity



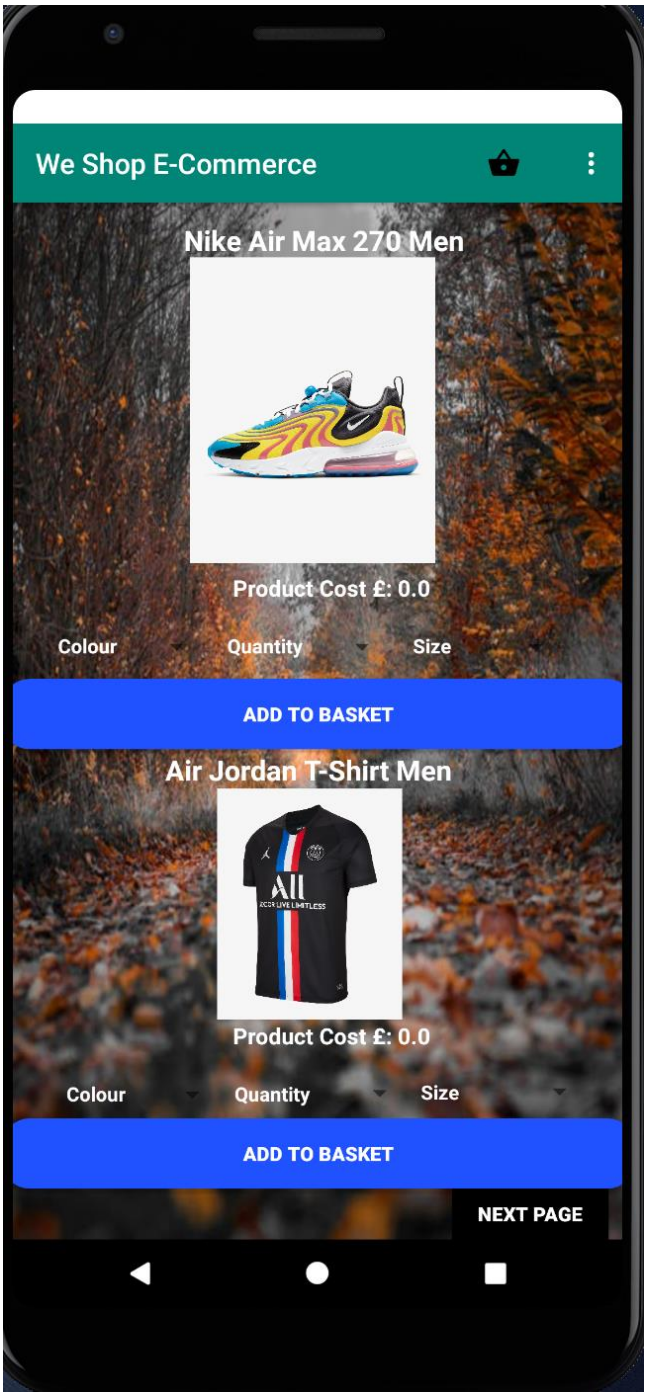
This is how the Login Activity looks. The user will log in with their registered account, if they enter the wrong credentials then they will receive an error saying that the details are incorrect. If the credentials entered are correct and match the database, then they are taken back to the homepage.

Submit Complaint Implementation



Customers that are experiencing problems with the application can submit a complaint by filling out the form above. The details of the customer and the corresponding complaint will get stored in a back-end SQLite Database in which it will be used in order to see what is wrong and the problem will be taken into consideration for future updates.

Sports and Outdoors Activity Implementation

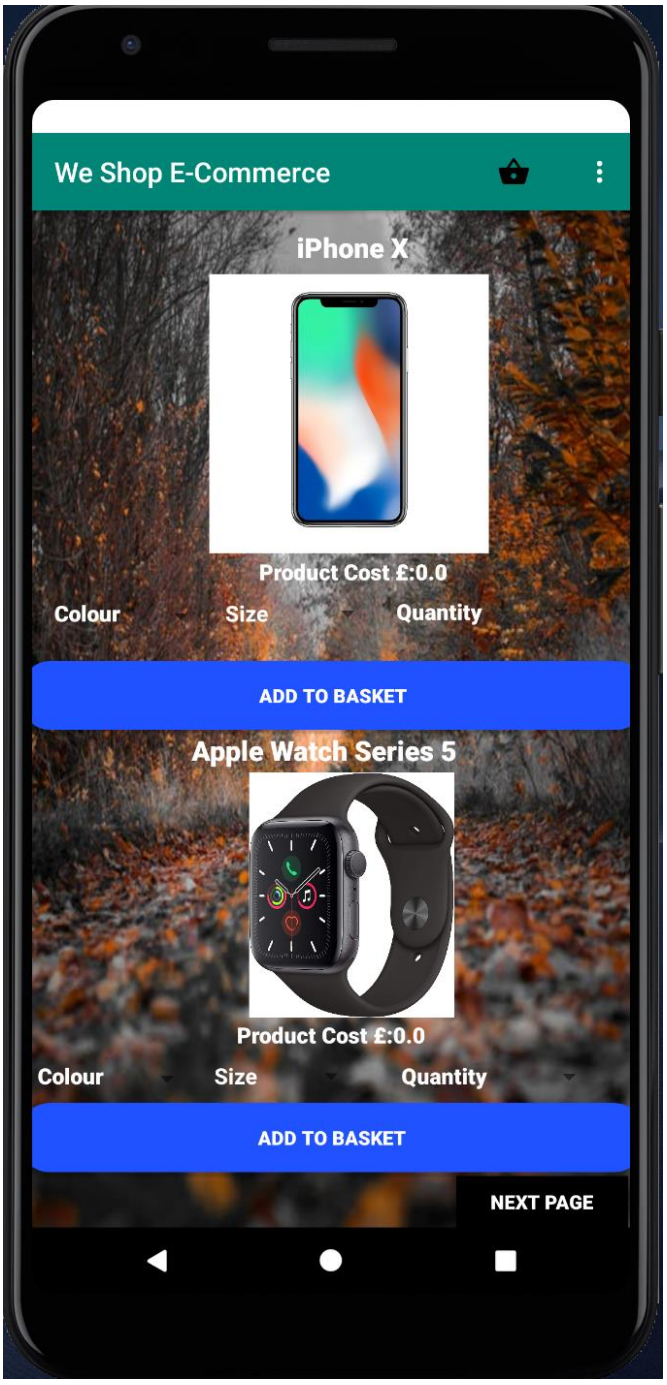


This is how the Sports and Outdoors Activity User Interface looks like. Customers can choose the color, quantity and size they would like for each product and then they can add it to the basket.

Basket Activity Interface Implementation

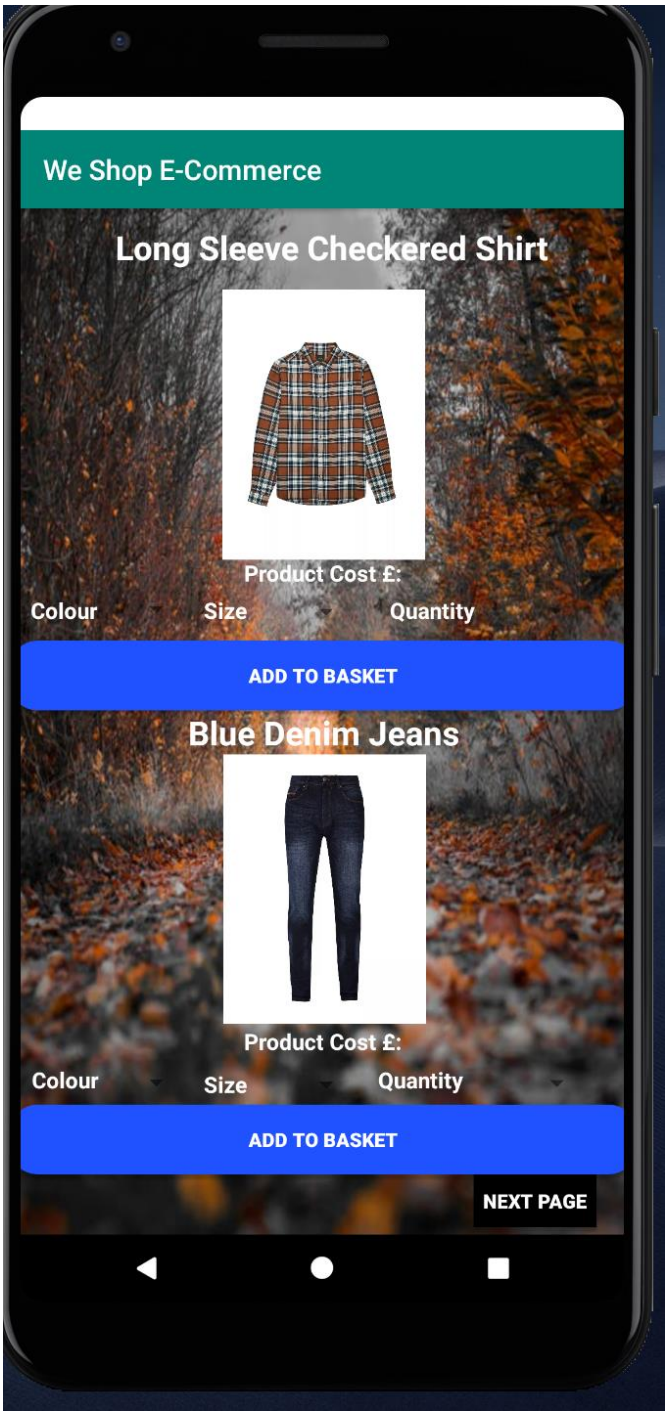
This is how the basket looks like when customers add the products they wish to the basket, this applies for all of the other categories, Tech, Clothing etc. If the customers are happy with the products in their basket, then they can go ahead and click the place order button which will take them to the **Payment Activity**.

Tech User Interface Implementation



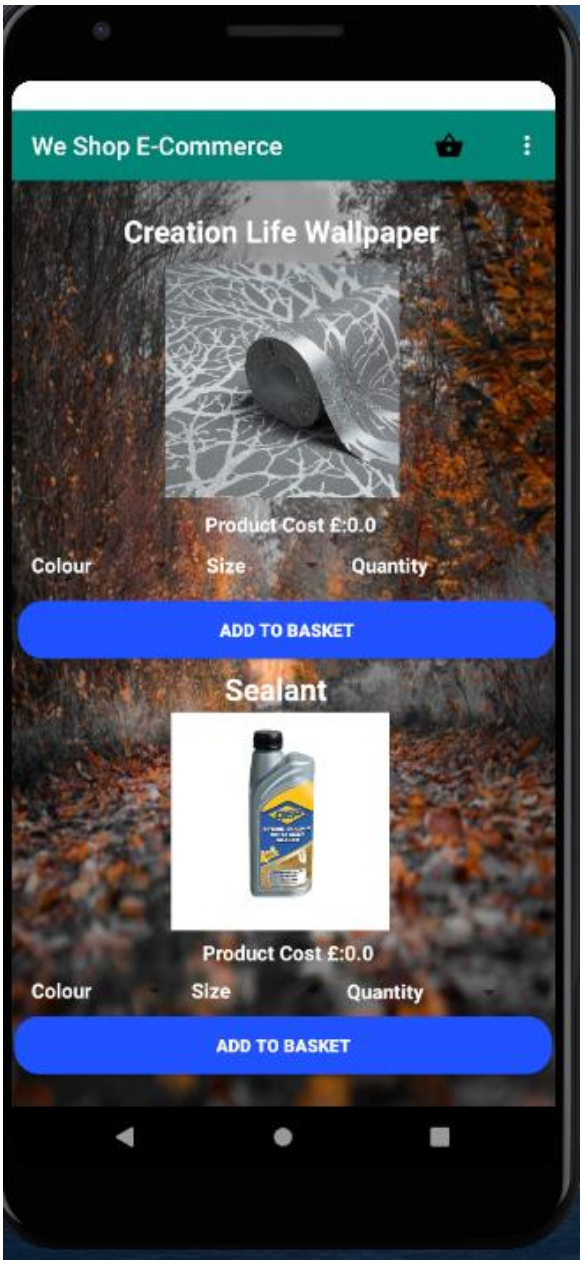
This is how the Tech Activity User Interface looks like. It is pretty minimalistic and simple to use. Customers can choose the color, size and quantity of the product they would like and then they can add the product(s) to the basket. Same applies to the next page of the tech activity.

Clothing User Interface Implementation



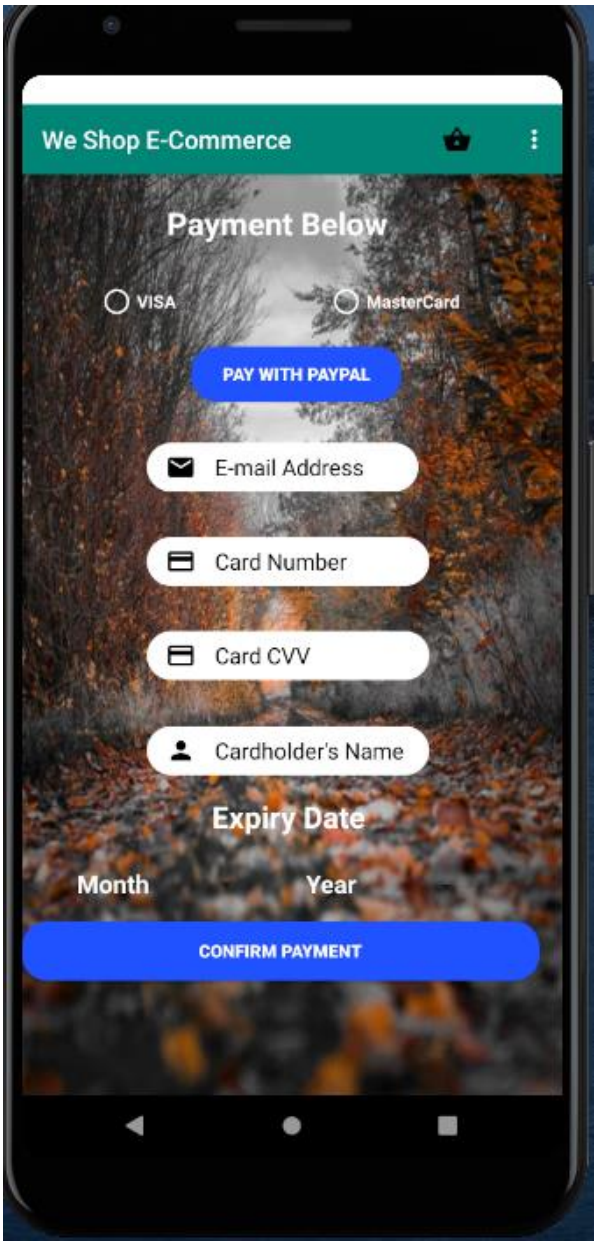
This is how the Clothing User Interface looks like. Same applies here, customers choose the color, size and quantity of the product(s) they would like to purchase then they can add it to the basket.

DIY Activity Implementation



This is how the DIY User Interface looks like implemented. It has the same functionality as the other categories.

Payment User Interface Implementation



This is how the Payment Activity looks like implemented. After customers have chosen the products they would like to purchase, they are taken to this activity in order to purchase the products.

Customers have the option to pay with PayPal if they don't have VISA or MasterCard. After the payment process is successful, the customers will receive a confirmation e-mail sent to them saying that the order has been successful.

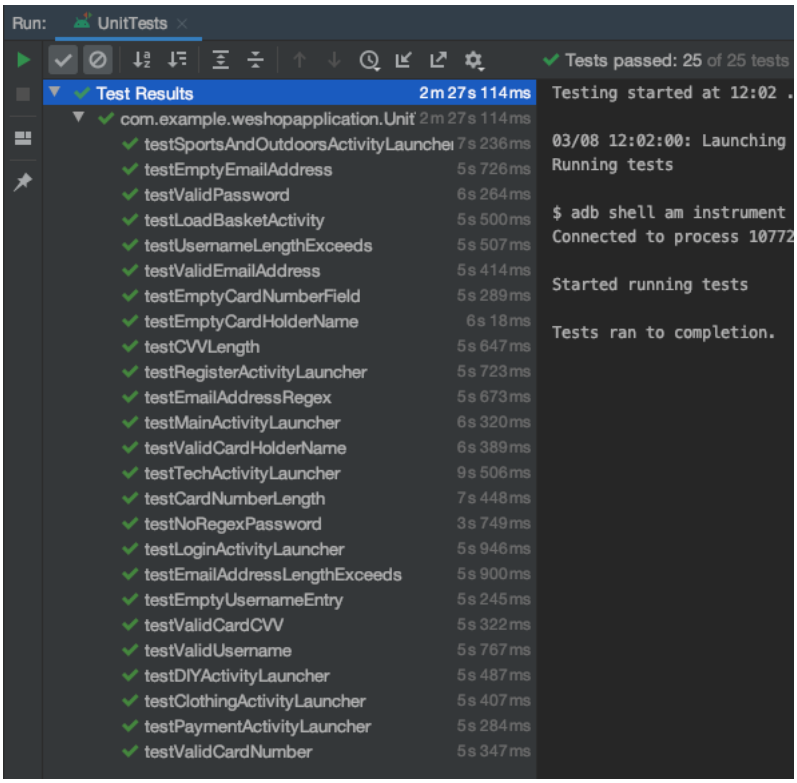
Test Plan

The test plan table below shows which features of the application will be tested using Java Unit tests.

There will be a total of 25 Tests.

Test Feature	Test Input	Test Passed	Comments
Launch Main Activity.	Welcome Text View	Yes	**Test Passes**
Test Valid Username.	Sabin2000	Yes	**Test Passes.**
Test Empty Username.	Empty string (null)	Yes	**Test Passes.**
Test Username Length.	Kwefgowijfgoifgjwoif09wufwgf9w	Yes	**Test Passes.**
Test Valid E-mail Address	Sabinlungu293@gmail.com	Yes	**Test Passes.**
Test E-mail Address Length	jadoijdfijaoifjsif@gmail.com	Yes	**Test Passes.**
Test Empty E-mail Address	Empty string (null)	Yes	**Test Passes.**
Test Regex E-mail Address	Sabinlungu293gmail.com	Yes	**Test Passes.**
Test Valid Password	Sabin2000*@(	**Yes**	**Test Passes.**
Test No Regex Password	Sabin2000	**Yes**	**Test Passes**
Test Valid Card Number	1234000090991234	**Yes**	**Test Passes.**
Test Card Number Length	23947234823748932478293476829346 7	**Yes**	**Test Passes**
Test Empty Card Number	Empty string (null)	**Yes**	**Test Passes.**
Test Valid Card CVV	218	**Yes**	**Test Passes.**
Test Card CVV Length	1234	**Yes**	**Test Passes.**
Test Valid Card Holder Name	Sabin Lungu	**Yes**	**Test Passes.**
Test Empty Card Holder Name	Empty string (null)	**Yes**	**Test Passes.**

Test Register Activity Launcher	Looks for the Register Text View	Yes	**Test Passes.**
Test Sports and Outdoors Activity Launcher	Looks for the Product Cost Label.	Yes	**Test Passes.**
Test Tech Activity Launcher	Looks for a Tech Product Image	Yes	**Test Passes.**
Test Clothing Activity Launcher	Looks for the Clothing Product Cost Label	Yes	**Test Passes.**
Test DIY Activity Launcher	Looks for the product cost label	Yes	**Test Passes.**
Test Basket Activity Launcher	Looks for the Place Order Button	Yes	**Test Passes.**
Test Login Activity Launcher	Looks for the Login Button.	Yes	**Test Passes.**
Test Payment Activity Launcher	Looks for the Payment Text View	Yes	**Test Passes.**



Payment Database Implementation

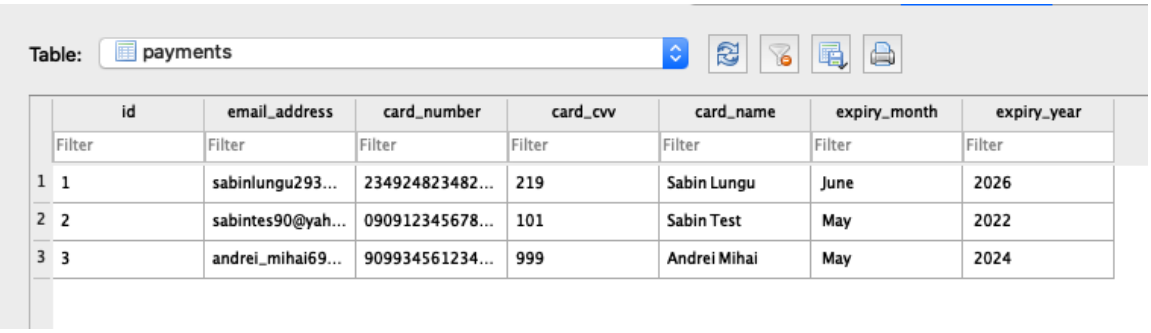


Table: payments

	id	email_address	card_number	card_cvv	card_name	expiry_month	expiry_year
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	sabinlungu293...	234924823482...	219	Sabin Lungu	June	2026
2	2	sabintes90@yah...	090912345678...	101	Sabin Test	May	2022
3	3	andrei_mihai69...	909934561234...	999	Andrei Mihai	May	2024

This image shows some test payment data being stored in a MySQLite database.

Contact Us Database Implementation

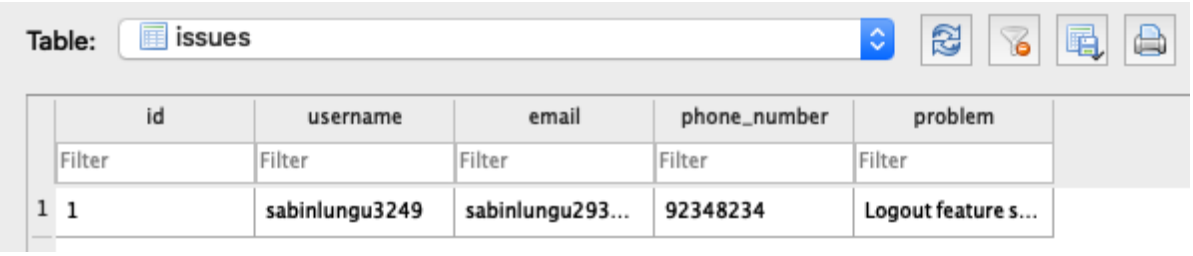


Table: issues

	id	username	email	phone_number	problem
	Filter	Filter	Filter	Filter	Filter
1	1	sabinlungu3249	sabinlungu293...	92348234	Logout feature s...

Application Evaluation

In this section I will be writing about my implemented application, I will be making a comparison against the application inspiration (Amazon) that I decided to inspire myself from in order to implement the We-Shop Shopping Application. In this section I will be also taking into account user feedback and from that feedback I will write a section on what can be improved.

Overall, I think that my implementation of the application went really well, however I did encounter a lot of bugs and issues throughout the process, but in the end I managed to successfully implement a fully working application with a lot of functionality, but of course there are still bits in the application that are a bit flawed and will require a bit of maintenance.

Evaluation – Application Comparison

After completing the implementation of my application I revisited the Amazon application from where I got my inspiration from and I can come to a conclusion that my application has similar functionality to the Amazon one, however it is clear that the Amazon application has more sections in which users can choose to buy products from, of course I did not implement more than 4 sections in my app because it would be far too big and would take up a lot of time to complete.

Evaluation – User Feedback

There have been multiple people who have looked at my application and have evaluated it and provided me useful feedback. I have taken this feedback into account and will include this in any upcoming maintenance that I will make. Below are the people who have provided me feedback.

- **Sadeem Rashid:** “The User Interface of the shopping application looks very nice; it is consistent, easy to navigate around on and has a high level of functionality. There are a couple of issues with the application, one is that the logout button is visible once the application loads up, this should not happen and should be invisible upon application launch”

- **Taylor Courtney:** “The application has a nice look and feel, the buttons look nice, there is a consistent navigation that makes it easier for users to use. There are a couple of problems that I have noticed, one is that when users choose a size or color from the drop-down menu, it doesn’t actually display the text, it is not major, but it doesn’t look as well if it doesn’t display it. Also, the logout by default should not be visible on the home page.”

- **Jonathan Sung:** “In my opinion the application looks fantastic, it has really good functionality, it is easy to use and consistent. I have pinpointed a couple of issues in the application that can be fixed in the future, but they are quite minor. First of all, a toast message or alert box could be shown when adding a product to the basket to identify that the product has been added successfully, also a counter on the basket icon can be shown to identify how many items are currently in the basket with a red circle.”

Evaluation – What could be improved?

From the feedback that I have received from my colleagues, I will compile a list of things that could be improved.

- **Logout Feature**: The logout button should not be visible when the user first launches the application, I will try to make it visible after the user logs in and bot before.

- **Show Text on Drop-Down Menu**: Currently for some reason the quantity, color or size does not show, this is not major but will be tried to be fixed.

- **Basket Icon Counter**: Whenever a customer adds a product to the basket, it currently doesn’t show how many products are in the basket, I could improve my application by adding a red counter after every product added to the basket to identify the number of products in the basket.

- **Basket Add Product Toast Message:** A toast message can be displayed to show that the product has successfully been added to the basket.

Overall in conclusion I am very happy with the work that I have put into my application and there are a lot of things that can be improved in the future.

Appendix

Here I will be listing all of the source code and images that I have gotten from the internet.

Appendix – Application Layer Main Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.content.ActivityNotFoundException;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.example.weshopapplication.R;
import com.google.firebase.auth.FirebaseAuth;

// Author of Application: Sabin Constantin Lungu
// Purpose of Class: Main Activity implements homepage functionality of application.
// Last modified: 26 January 2020
// Any Bugs?: N/A

public class MainActivity extends AppCompatActivity {
    // Encapsulated variables
    private TextView text; //

    private Button registerButton; // Variable Button to register
    private Button loginButton; // Variable to store the login button
    private Button contactUsBtn;
    private FirebaseAuth auth;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        // Initialise components
        this.text = findViewById(R.id.welcomeTxt);

        this.registerButton = findViewById(R.id.registerBtn);
        this.loginButton = findViewById(R.id.loginBtn);
        this.contactUsBtn = findViewById(R.id.contactUsBtn);
```

```
    this.auth = FirebaseAuth.getInstance(); // Get instance of fire base

    this.registerButton.setOnClickListener(new View.OnClickListener() { // Listener added to register button
        @Override
        public void onClick(View v) {
            try {

                if (v.getId() == R.id.registerBtn) { // If the register button is clicked
                    Intent registerIntent = new Intent(MainActivity.this, RegisterActivity.class);
                    startActivity(registerIntent); // Take user to the register page
                }

            } catch (ActivityNotFoundException act) { // Catch exception if the activity is not found
                act.printStackTrace();
                Log.d("Cause : ", act.getMessage());
            }
        }
    });

    this.loginButton.setOnClickListener(new View.OnClickListener() { // Listener for login button
        @Override
        public void onClick(View v) {
            try {

                Intent loginIntent = new Intent(MainActivity.this, LoginActivity.class);
                startActivity(loginIntent);

            } catch (ActivityNotFoundException exc) {
                Log.d("Error : ", exc.getMessage());
            }
        }
    });

    this.contactUsBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            try {

                Intent contactUsIntent = new Intent(MainActivity.this, ContactUsActivity.class);
                startActivity(contactUsIntent);

            } catch (ActivityNotFoundException exc) {
                Log.d("Error", exc.toString());
            }
        }
    });
}


public boolean onCreateOptionsMenu(Menu menu) { // Overidden method that creates the menu
```

```
        // Inflate the activities
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.homepagemenu, menu);

        // Inflate the logout button
        MenuInflater logoutInflater = getMenuInflater();
        logoutInflater.inflate(R.menu.logout_menu, menu);

        MenuItem logout = menu.findItem(R.id.logout_button);
        logout.setVisible(true); // Set the logout button to be initially true. Will be changed to false

        return true;
    }



    public boolean onOptionsItemSelected(MenuItem item) { // Determines which item is selected from the menu
        try {
            if (item == null) { // If there is no item to choose
                return false; // Return false
            }

            // Determines which activity is chosen
            switch (item.getItemId()) {
                case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors category is chosen
                    Intent sportsIntent = new Intent(MainActivity.this, SportsAndOutdoorsActivity.class); // Take user to the
sports and outdoors activity
                    startActivity(sportsIntent);

                    return true; // Return true.

                case R.id.techCategory:
                    Intent techIntent = new Intent(MainActivity.this, TechActivity.class);
                    startActivity(techIntent);

                    return true;


                case R.id.clothingCategory: // If the clothing category is chosen

                    Intent clothingIntent = new Intent(MainActivity.this, ClothingCategory.class);
                    startActivity(clothingIntent); // Go to the clothing intent

                    return true;

                case R.id.diyCategory:

                    Intent intent = new Intent(MainActivity.this, DIYActivity.class);
                    startActivity(intent);

                    return true; // Return true;
```

```
            case R.id.logout_button:

                logout(); // Logs the user out
                return true;

            default:

                return super.onOptionsItemSelected(item); // Return the base item

        }

    } catch (ActivityNotFoundException exc) { // Catch exception
        exc.printStackTrace();
    }

    return true;
}

public void onBackPressed() { // Routine that determines if the back button is pressed
    moveTaskToBack(true); // Move back a task
}

private void logout() { // Logout the user
    auth.signOut();
    finish();
    startActivity(new Intent(MainActivity.this, LoginActivity.class)); // Take user to login activity
}
}
```

Appendix – Application Layer Sports and Outdoors Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
```

```
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;

import java.util.ArrayList;
import java.util.HashMap;


// Author of Application & Class: Sabin Constantin Lungu
// Purpose of Class: Stores the code needed to implement the Sports and Outdoors Activity 1.
// Date of Last Modification: 13/02/2020
// Any Errors? Currently None.


public class SportsAndOutdoorsActivity extends AppCompatActivity implements
AdapterView.OnItemSelectedListener {
    private int current_product_id = 1;
    private ImageView cartIcon;

    private TextView firstSportsOutdoorTxt;
    private ImageView firstSportsOutdoorImg;

    private TextView firstSportsOutdoorCostTxt;
    private TextView firstSportsOutdoorColourLbl;
    private Spinner firstSportsOutdoorsColourMenu;

    private TextView firstSportsOutdoorQuantityLbl;
    private Spinner firstSportsOutdoorQuantityMenu;

    private TextView firstSportsOutdoorsSizeLbl;
    private Spinner firstSportsOutdoorsSizeMenu;
    private Button firstSportsAddToBasketBtn;

    private TextView secondSportsOutdoorTxt;
    private ImageView secondSportsOutdoorImg;

    private TextView secondSportsOutdoorCostLbl;
    private TextView secondSportsOutdoorColourLbl;
    private Spinner secondSportsOutdoorsColourMenu;

    private TextView secondSportsOutdoorQuantityLbl;
    private Spinner secondSportsOutdoorQuantityMenu;

    private TextView secondSportsOutdoorSizeLbl;
    private Spinner secondSportsOutdoorSizeMenu;
```

```
    private Button secondSportsAddToBasketBtn;
    private Button nextPageBtn;

    // The costs of the products
    private double[] productOneCosts = {0.00, 90.00, 180.00, 360.00, 720.00, 1440.00};
    private double[] productTwoCosts = {0.00, 50.00, 100.00, 150.00, 200.00, 250.00};

    private ArrayList<TechActivity.Colours> listOfColoursOne; // An array list of colours
    private ArrayList<TechActivity.Quantities> listOfQuantitiesOne; // An array list of quantities
    private ArrayList<Size> listOfSizesOne;

    private ArrayList<TechActivity.Colours> listOfColoursTwo;
    private ArrayList<TechActivity.Quantities> listOfQuantitiesTwo;
    private ArrayList<Size> listOfSizesTwo;

    private ColourArrayAdapter coloursAdapter; // A colours adapter is needed to store objects in a drop-down menu
(spinner)
    private QuantitiesArrayAdapter quantitiesAdapter;

    private QuantitiesArrayAdapter secondQuantitiesAdapter;
    private SizeArrayAdapter sizeArrayAdapter;

    private boolean coloursAdded; // Flag to determine if the colours have been added to the drop-down list
    private boolean quantitiesAdded;

    private boolean sizesAdded;
    private boolean addedToBasket; // True or false to determine if the products have been added to the basket after
adding to the hash map

    private HashMap<Integer, Products> listOfProductsToAddToBasket = new HashMap<>(); // A HashMap of
products to add to the basket with a corresponding ID that each product will have.

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sports_and_outdoors);

        // INITIALISE COMPONENTS
        this.firstSportsOutdoorTxt = findViewById(R.id.firstSportsOutdoorTxt);
        this.firstSportsOutdoorImg = findViewById(R.id.firstSportsOutdoorImg);
        this.firstSportsOutdoorCostTxt = findViewById(R.id.firstSportsOutdoorCostLbl);
        this.firstSportsOutdoorColourLbl = findViewById(R.id.firstSportsColourLbl);

        this.firstSportsOutdoorsColourMenu = findViewById(R.id.firstSportsOutdoorColourMenu);
        this.firstSportsOutdoorQuantityLbl = findViewById(R.id.firstSportsOutdoorQuantityLbl);
        this.firstSportsOutdoorQuantityMenu = findViewById(R.id.firstSportsOutdoorQuantityMenu);

        this.firstSportsOutdoorsSizeLbl = findViewById(R.id.firstSportsOutdoorSizeLbl);
        this.firstSportsOutdoorsSizeMenu = findViewById(R.id.firstSportsOutdoorSizeMenu);

        this.firstSportsAddToBasketBtn = findViewById(R.id.firstAddToBasketBtn); // Button for the first product to add to
the basket.
```

```
this.secondSportsOutdoorTxt = findViewById(R.id.secondSportsOutdoorsProductTxt);
this.secondSportsOutdoorImg = findViewById(R.id.secondSportsOutdoorsImg);

this.secondSportsOutdoorCostLbl = findViewById(R.id.secondSportsOutdoorProductCostLbl);
this.secondSportsOutdoorColourLbl = findViewById(R.id.secondSportsOutdoorsColourLbl);
this.secondSportsOutdoorsColourMenu = findViewById(R.id.secondSportsOutdoorsColourMenu);

this.secondSportsOutdoorQuantityLbl = findViewById(R.id.secondsSportsOutdoorQuantityLbl);
this.secondSportsOutdoorQuantityMenu = findViewById(R.id.secondSportsOutdoorsQuantityMenu);

this.secondSportsOutdoorSizeLbl = findViewById(R.id.secondSportsOutdoorsSizeLbl);
this.secondSportsOutdoorSizeMenu = findViewById(R.id.secondSportsOutdoorsSizeMenu);

this.secondSportsAddToBasketBtn = findViewById(R.id.secondAddToBasketBtn);
this.nextPageBtn = findViewById(R.id.sportsNextPage); // Button for taking the user to the next page.

// Create the array lists
this.listOfColoursOne = new ArrayList<>();
this.listOfQuantitiesOne = new ArrayList<>();
this.listOfSizesOne = new ArrayList<>();

this.listOfColoursTwo = new ArrayList<>();
this.listOfQuantitiesTwo = new ArrayList<>();
this.listOfSizesTwo = new ArrayList<>();

// Method calls to add to the specific array lists
addToColoursList();
addToColoursListTwo(); // Method call to add to the colours list two.

addToQuantitiesListOne();
addToQuantitiesListTwo();
addToSizesList();

// Set up the colours adapter
this.coloursAdapter = new ColourArrayAdapter(SportsAndOutdoorsActivity.this, listOfColoursOne);
coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

firstSportsOutdoorsColourMenu.setAdapter(coloursAdapter);
firstSportsOutdoorsColourMenu.setOnItemSelectedListener(this);

// Create array adapter for the quantities for product 1
this.quantitiesAdapter = new QuantitiesArrayAdapter(SportsAndOutdoorsActivity.this, listOfQuantitiesOne);
quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

firstSportsOutdoorQuantityMenu.setAdapter(quantitiesAdapter);
firstSportsOutdoorQuantityMenu.setOnItemSelectedListener(this);

// Create array adapter for the sizes
this.sizeArrayAdapter = new SizeArrayAdapter(SportsAndOutdoorsActivity.this, listOfSizesOne);
sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
```

```
        firstSportsOutdoorsSizeMenu.setAdapter(sizeArrayAdapter);
        firstSportsOutdoorsSizeMenu.setOnItemSelectedListener(this);

        this.coloursAdapter = new ColourArrayAdapter(SportsAndOutdoorsActivity.this, listOfColoursTwo);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        secondSportsOutdoorsColourMenu.setAdapter(coloursAdapter);
        secondSportsOutdoorsColourMenu.setOnItemSelectedListener(this);

        // Create the Array Adapter for the quantities for the second product
        this.secondQuantitiesAdapter = new QuantitiesArrayAdapter(SportsAndOutdoorsActivity.this,
listOfQuantitiesTwo);
        secondQuantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        secondSportsOutdoorQuantityMenu.setAdapter(secondQuantitiesAdapter);
        secondSportsOutdoorQuantityMenu.setOnItemSelectedListener(this);

        // Create the sizes array adapter for the second product
        this.sizeArrayAdapter = new SizeArrayAdapter(SportsAndOutdoorsActivity.this, listOfSizesTwo);
        sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        secondSportsOutdoorSizeMenu.setAdapter(sizeArrayAdapter);
        secondSportsOutdoorSizeMenu.setOnItemSelectedListener(this);

        this.nextPageBtn = findViewById(R.id.sportsNextPage); // The next page button

        this.nextPageBtn.setOnClickListener(new View.OnClickListener() { // Add an action listener to the next page
button to take the user to the next page
            @Override
            public void onClick(View v) {
                try {

                    Intent nextSportsActivity = new Intent(SportsAndOutdoorsActivity.this,
SportsAndOutdoorsActivityTwo.class);
                    startActivity(nextSportsActivity);

                } catch (ActivityNotFoundException exc) { // Catch the error if the activity has not been found

                    Log.d(String.valueOf(R.string.error), exc.toString()); // Log the error to the console
                }
            }
        });

        firstSportsAddToBasketBtn.setOnClickListener(new View.OnClickListener() { // Add action listener to the first
button
            @Override
            public void onClick(View v) {

                if (v.getId() == R.id.firstAddToBasketBtn) {

                    if (firstSportsOutdoorsColourMenu.getSelectedItemPosition() == 0 ||
firstSportsOutdoorsSizeMenu.getSelectedItemPosition() == 0 || firstSportsOutdoorsSizeMenu.getSelectedItemId() ==
```

```
0) {

                AlertDialog.Builder error = new AlertDialog.Builder(SportsAndOutdoorsActivity.this) // Create an alert
dialogue for the user to see.
                        .setTitle(R.string.error)
                        .setMessage(R.string.errorMsg)
                        .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {

                            @Override

                            public void onClick(DialogInterface dialog, int which) {
                                if (dialog != null) { // If there is a dialog
                                    dialog.dismiss(); // Dismiss it
                                }
                            }
                        });

                error.show();
                error.setCancelable(true);

            }

            else {

                addToBasketOne(); // Otherwise add the product to the basket one
            }
        }

    }
});

    secondSportsAddToBasketBtn.setOnClickListener(new View.OnClickListener() { // Adds an action listener to the
second add to basket button
        @Override
        public void onClick(View v) {
            Context context = getApplicationContext();
            String[] tempResources = new String[]{context.getString(R.string.chooseOption)};

            if (v.getId() == R.id.secondAddToBasketBtn) { // If the second add to basket button is clicked

                if (secondSportsOutdoorsColourMenu.getSelectedItemPosition() == 0 ||
secondSportsOutdoorQuantityMenu.getSelectedItemPosition() == 0 ||
secondSportsOutdoorSizeMenu.getSelectedItemPosition() == 0) { // If index 0 is chosen for the colour menu

                    AlertDialog.Builder errorMenu = new AlertDialog.Builder(SportsAndOutdoorsActivity.this)
                            .setTitle(R.string.error)

                            .setMessage(tempResources[0])
                            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                @Override

                                public void onClick(DialogInterface dialog, int which) {
```

```
                            if (dialog != null) {
                                dialog.dismiss();
                            }
                        }
                    });

                errorMenu.show();
                errorMenu.setCancelable(true);
            } else {

                addToBasketTwo(); // Otherwise if the above condition is false, add to the basket two.
            }
        }
    }
});
}

private boolean addToColoursList() { // Routine returns true or false if the colours have been added to the array list
or not.

    Context context = getApplicationContext();
    String[] coloursResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.colourYellow), context.getString(R.string.colourBlack),
            context.getString(R.string.colourRed), context.getString(R.string.skyBlue)};

    TechActivity.Colours[] colours = {new TechActivity.Colours(0, coloursResources[0]), new TechActivity.Colours(1,
coloursResources[1]),
            new TechActivity.Colours(2, coloursResources[2]), new TechActivity.Colours(3, coloursResources[3]),
            new TechActivity.Colours(4, coloursResources[4])};

    for (TechActivity.Colours productColours : colours) { // For each colours in the array
        listOfColoursOne.add(productColours); // Add it to the array list
        coloursAdded = true; // Colours have been added
    }

    return true;
}

private boolean addToColoursListTwo() {
    Context context = getApplicationContext();
    String[] coloursResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.blue), context.getString(R.string.red), context.getString(R.string.limeGreen)};

    TechActivity.Colours[] colours = {new TechActivity.Colours(0, coloursResources[0]), new TechActivity.Colours(1,
coloursResources[1]),
            new TechActivity.Colours(2, coloursResources[2]), new TechActivity.Colours(3, coloursResources[3])};

    for (TechActivity.Colours theColours : colours) {
        listOfColoursTwo.add(theColours);
        coloursAdded = true;
    }
```

```
        return true;
    }

    private boolean addToQuantitiesListOne() {

        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2])
                , new TechActivity.Quantities(quantitiesResources[3]), new
TechActivity.Quantities(quantitiesResources[4]), new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities quantitiesArray : quantities) {
            listOfQuantitiesOne.add(quantitiesArray);
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean addToQuantitiesListTwo() { // Add to quantities list two.

        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2])
                , new TechActivity.Quantities(quantitiesResources[3]), new
TechActivity.Quantities(quantitiesResources[4]), new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities quantitiesArray : quantities) { // For every quantity in the object array
            listOfQuantitiesTwo.add(quantitiesArray); // Add it to the array list
            quantitiesAdded = true; // The quantities have been added to the list is true.
        }

        return true;
    }

    private boolean addToSizesList() {
        Context context = getApplicationContext();
        String[] sizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.smallSize), context.getString(R.string.mediumSize), context.getString(R.string.largeSize),
context.getString(R.string.extraLargeSize)};

        Size[] sizes = {new Size(0, sizesResources[0]), new Size(1, sizesResources[1]), new Size(2,
```

```
sizesResources[2]),
            new Size(3, sizesResources[3]), new Size(4, sizesResources[4])}; // Creates an array of object of type size.

        for (Size theSizes : sizes) { // For each of the sizes in the array
            listOfSizesOne.add(theSizes);
            listOfSizesTwo.add(theSizes);

            sizesAdded = true;
        }

        return true;
    }

    private boolean addToBasketOne() { // Adds the first product to the basket
        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(SportsAndOutdoorsActivity.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false); // The dialog can be cancelled by the user by clicking out of bounds.

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() { // Routine that runs the thread.

                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show(); // Shows the dialog

        // Create an instance for the first product and adds it to the hash map.
        Products firstSportsProduct = new Products(current_product_id, firstSportsOutdoorTxt.getText().toString(),
firstSportsOutdoorsColourMenu.getSelectedItem().toString(), (int)
firstSportsOutdoorQuantityMenu.getSelectedItemId(), firstSportsOutdoorCostTxt.getText().toString(),
firstSportsOutdoorsSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id, firstSportsProduct); // Add the product instance to the
hash map.

        return true; // Returns true.
```

```
    }

    private boolean addToBasketTwo() { // Adds the second product to the basket
        Context context = getApplicationContext();

        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};
        final ProgressDialog dialog = new ProgressDialog(SportsAndOutdoorsActivity.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog.

        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products secondSportsProduct = new Products(current_product_id++,
secondSportsOutdoorTxt.getText().toString(), secondSportsOutdoorsColourMenu.getSelectedItem().toString(), (int)
secondSportsOutdoorQuantityMenu.getSelectedItemId(), secondSportsOutdoorCostLbl.getText().toString(),
secondSportsOutdoorSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, secondSportsProduct);

        return true;
    }

    @SuppressLint({"DefaultLocale", "SetTextI18n"})
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) { // Method that determines
which item has been selected and at which index
        boolean valueAppended = false;
        int[] indexes = {0, 1, 2, 3, 4};

        Context context = getApplicationContext();
        String[] productResources = new String[]{context.getString(R.string.productCost)};
```

```
if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[0]))) {
    firstSportsOutdoorCostTxt.setText(null);
    firstSportsOutdoorCostTxt.setText(productResources[0] + (productOneCosts[0]));
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[1]))) {
    firstSportsOutdoorCostTxt.setText(productResources[0] + (productOneCosts[1]));
    valueAppended = true; // Value is appended

}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[2]))) {
    firstSportsOutdoorCostTxt.setText(null);
    firstSportsOutdoorCostTxt.append(productResources[0] + productOneCosts[2]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[3]))) {
    firstSportsOutdoorCostTxt.setText(null);
    firstSportsOutdoorCostTxt.append(productResources[0] + productOneCosts[3]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[4]))) {
    firstSportsOutdoorCostTxt.setText(null);
    firstSportsOutdoorCostTxt.append(productResources[0] + productOneCosts[4]);
    valueAppended = true;
}


if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[0]))) {
    secondSportsOutdoorCostLbl.setText(null);
    secondSportsOutdoorCostLbl.append(productResources[0] + productTwoCosts[0]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[1]))) {
    secondSportsOutdoorCostLbl.setText(null);
    secondSportsOutdoorCostLbl.append(productResources[0] + productTwoCosts[1]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[2]))) {
    secondSportsOutdoorCostLbl.setText(null);
    secondSportsOutdoorCostLbl.append(productResources[0] + productTwoCosts[2]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[3]))) {
    secondSportsOutdoorCostLbl.setText(null);
    secondSportsOutdoorCostLbl.append(productResources[0] + productTwoCosts[3]);
```

```
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[4]))) {
            secondSportsOutdoorCostLbl.setText(null);
            secondSportsOutdoorCostLbl.append(productResources[0] + productTwoCosts[4]);
            valueAppended = true;
        }
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) { // Nothing selected

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
        // Inflate the activities menu
        MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
        activityInflater.inflate(R.menu.homepagemenu, menu);

        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.basket_action_button, menu);

        View view = menu.findItem(R.id.cart_menu).getActionView();

        cartIcon = view.findViewById(R.id.cart_icon);

        cartIcon.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent basketIntent = new Intent(SportsAndOutdoorsActivity.this, BasketActivity.class); // Create a basket
intent
                basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
                startActivity(basketIntent); // Start the intent
            }
        });

        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) {

        try {
            switch (item.getItemId()) {
```

```
            case R.id.sportsAndOutdoorsCategory:
                Intent sportsCategory = new Intent(SportsAndOutdoorsActivity.this, SportsAndOutdoorsActivity.class);
                startActivity(sportsCategory);

                break;

            case R.id.techCategory:
                Intent techActivity = new Intent(SportsAndOutdoorsActivity.this, TechActivity.class);
                startActivity(techActivity);
                break;

            case R.id.clothingCategory:
                Intent clothingActivity = new Intent(SportsAndOutdoorsActivity.this, ClothingCategory.class);
                startActivity(clothingActivity);
                break;

            case R.id.diyCategory:
                Intent diyActivity = new Intent(SportsAndOutdoorsActivity.this, DIYActivity.class);
                startActivity(diyActivity);
                break;

            default:
                super.onOptionsItemSelected(item);

        }

    } catch (ActivityNotFoundException exc) {
        Log.d(String.valueOf(R.string.error), exc.toString());
    }

    return true;
  }
}
```

Appendix – Application Layer Sports and Outdoors Activity Two Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
```

```
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;

import java.util.ArrayList;
import java.util.HashMap;

// Author of Application & Class: Sabin Constantin Lungu
// Purpose of Class: Stores the code needed to implement the Sports and Outdoors Activity 2.
// Date of Last Modification: 13/02/2020
// Any Errors? Currently None

public class SportsAndOutdoorsActivityTwo extends AppCompatActivity implements
AdapterView.OnItemSelectedListener {
    private int current_product_id = 1;
    private ImageView cartIcon;

    private TextView sportsOutdoorsTxtTwo;
    private ImageView thirdSportsOutdoorsImg;

    private TextView thirdSportsOutdoorsCostLbl;
    private TextView thirdSportsOutdoorsColoursLbl;
    private Spinner thirdSportsOutdoorsColoursMenu;

    private TextView thirdSportsOutdoorsSizeLbl;
    private Spinner thirdSportsOutdoorsSizeMenu;
    private TextView thirdSportsOutdoorsQuantityLbl;

    private Spinner thirdSportsOutdoorsQuantityMenu;
    private Button thirdSportsOutdoorsAddToBasketBtn;

    private TextView fourthSportsOutdoorsTxt;
    private TextView fourthSportsOutdoorsCostLbl;
    private ImageView fourthSportsOutdoorsImg;

    private TextView fourthSportsOutdoorsColourLbl;
    private Spinner fourthSportsOutdoorsColourMenu;
```

```
    private TextView fourthSportsOutdoorsSizeLbl;
    private Spinner fourthSportsOutdoorsSizeMenu;

    private TextView fourthSportsOutdoorsQuantityLbl;

    private Spinner fourthSportsOutdoorsQuantityMenu;
    private Button fourthAddToBasketBtn;

    private double[] thirdProductCosts = {0.00, 60.00, 120.00, 240.00, 480.00, 1020.00}; // The costs for the third
product
    private double[] fourthProductCosts = {0.00, 100.00, 200.00, 400.00, 800.00, 1600.00}; // Costs for the fourth
product

    private boolean coloursAdded; // Determines if the colours have been added to the array list successfully.
    private boolean quantitiesAdded;

    private boolean sizesAdded;
    private boolean addedToBasket;

    private ArrayList<TechActivity.Colours> listOfColoursOne; // An Array list of colours
    private ArrayList<TechActivity.Quantities> listOfQuantitiesOne;
    private ArrayList<Size> listOfSizesOne;

    private ArrayList<TechActivity.Colours> listOfColoursTwo;
    private ArrayList<Size> listOfSizesTwo;
    private ArrayList<TechActivity.Quantities> listOfQuantitiesTwo;

    private QuantitiesArrayAdapter quantitiesAdapter;
    private SizeArrayAdapter sizesAdapter;
    private ColourArrayAdapter coloursAdapter;

    private HashMap<Integer, Products> listOfProductsToAddToBasket; // A HashMap to store the products when
adding to the basket

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_sports_and_outdoors_two);

        // Initialise components
        this.sportsOutdoorsTxtTwo = findViewById(R.id.sportsOutdoorsTxtTwo);
        this.thirdSportsOutdoorsImg = findViewById(R.id.thirdSportsOutdoorsImg);
        this.thirdSportsOutdoorsCostLbl = findViewById(R.id.thirdSportsOutdoorsCostLbl);

        this.cartIcon = findViewById(R.id.cart_icon);

        this.thirdSportsOutdoorsColoursLbl = findViewById(R.id.thirdSportsOutdoorsColourLbl);
        this.thirdSportsOutdoorsColoursMenu = findViewById(R.id.thirdSportsOutdoorsColourMenu);

        this.thirdSportsOutdoorsSizeLbl = findViewById(R.id.thirdSportsOutdoorsSizeLbl);
        this.thirdSportsOutdoorsSizeMenu = findViewById(R.id.thirdSportsOutdoorsSizeMenu);
```

```
this.thirdSportsOutdoorsQuantityLbl = findViewById(R.id.thirdQuantityLbl);
this.thirdSportsOutdoorsQuantityMenu = findViewById(R.id.thirdSportsOutdoorsQuantityMenu);

this.thirdSportsOutdoorsAddToBasketBtn = findViewById(R.id.thirdSportsOutdoorsAddToBasketBtn);

this.fourthSportsOutdoorsTxt = findViewById(R.id.fourthSportsOutdoorsTxt);
this.fourthSportsOutdoorsCostLbl = findViewById(R.id.fourthSportsOutdoorsCostLbl);
this.fourthSportsOutdoorsImg = findViewById(R.id.fourthSportsOutdoorsImg);

this.fourthSportsOutdoorsColourLbl = findViewById(R.id.fourthSportsOutdoorsColourLbl);
this.fourthSportsOutdoorsColourMenu = findViewById(R.id.fourthSportsOutdoorsColourMenu);

this.fourthSportsOutdoorsSizeLbl = findViewById(R.id.fourthProductSizeLbl);
this.fourthSportsOutdoorsSizeMenu = findViewById(R.id.fourthSportsOutdoorsSizeMenu);

this.fourthSportsOutdoorsQuantityLbl = findViewById(R.id.fourthSportsOutdoorsQuantityLbl);
this.fourthSportsOutdoorsQuantityMenu = findViewById(R.id.fourthSportsOutdoorsQuantityMenu);

this.fourthAddToBasketBtn = findViewById(R.id.fourthSportsOutdoorsAddToBasketBtn);

this.listOfColoursOne = new ArrayList<>();
this.listOfColoursTwo = new ArrayList<>();

this.listOfSizesOne = new ArrayList<>();
this.listOfSizesTwo = new ArrayList<>();

this.listOfQuantitiesOne = new ArrayList<>();
this.listOfQuantitiesTwo = new ArrayList<>();

this.listOfProductsToAddToBasket = new HashMap<>();

addToColoursListOne();
addToColoursListTwo();

addToQuantitiesList();
addToQuantitiesListTwo();
addToSizesListOne();

this.quantitiesAdapter = new QuantitiesArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfQuantitiesOne);
quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

thirdSportsOutdoorsQuantityMenu.setAdapter(quantitiesAdapter);
thirdSportsOutdoorsQuantityMenu.setOnItemSelectedListener(this);

this.sizesAdapter = new SizeArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfSizesOne);
sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

thirdSportsOutdoorsSizeMenu.setAdapter(sizesAdapter);
thirdSportsOutdoorsSizeMenu.setOnItemSelectedListener(this);

this.coloursAdapter = new ColourArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfColoursOne);
```

```
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        thirdSportsOutdoorsColoursMenu.setAdapter(coloursAdapter);
        thirdSportsOutdoorsColoursMenu.setOnItemSelectedListener(this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfQuantitiesTwo);
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        fourthSportsOutdoorsQuantityMenu.setAdapter(quantitiesAdapter);
        fourthSportsOutdoorsQuantityMenu.setOnItemSelectedListener(this);

        this.coloursAdapter = new ColourArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfColoursTwo);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        fourthSportsOutdoorsColourMenu.setAdapter(coloursAdapter);
        fourthSportsOutdoorsColourMenu.setOnItemSelectedListener(this);

        this.sizesAdapter = new SizeArrayAdapter(SportsAndOutdoorsActivityTwo.this, listOfSizesTwo);
        sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        fourthSportsOutdoorsSizeMenu.setAdapter(sizesAdapter);
        fourthSportsOutdoorsSizeMenu.setOnItemSelectedListener(this);

        // Initialise adapters

        this.thirdSportsOutdoorsAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                if (v.getId() == R.id.thirdSportsOutdoorsAddToBasketBtn) {

                    if(thirdSportsOutdoorsColoursMenu.getSelectedItemPosition() == 0 ||
thirdSportsOutdoorsSizeMenu.getSelectedItemPosition() == 0 ||
thirdSportsOutdoorsQuantityMenu.getSelectedItemPosition() == 0) {
                        AlertDialog.Builder chooseError = new AlertDialog.Builder(SportsAndOutdoorsActivityTwo.this)
                                .setTitle(R.string.error)
                                .setMessage(R.string.error2)
                                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                    @Override

                                    public void onClick(DialogInterface dialog, int which) {
                                        if(dialog != null) {
                                            dialog.dismiss();
                                        }
                                    }
                                });

                        chooseError.show();
                        chooseError.setCancelable(false);
                    }

                    else {
```

```
                addToBasketThree();
            }
        }
    }
});

this.fourthAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        if(fourthSportsOutdoorsColourMenu.getSelectedItemPosition() == 0 ||
fourthSportsOutdoorsSizeMenu.getSelectedItemPosition() == 0 ||
fourthSportsOutdoorsQuantityMenu.getSelectedItemPosition() == 0) {
            AlertDialog.Builder chooseError = new AlertDialog.Builder(SportsAndOutdoorsActivityTwo.this)
                    .setTitle(R.string.error)

                    .setMessage(R.string.error2)
                    .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                        @Override

                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) {
                                dialog.dismiss();
                            }
                        }
                    });

            chooseError.show();
            chooseError.setCancelable(false);
        }

        else {
            addToBasketFour();
        }
    }
});
}

private boolean addToColoursListOne() { // Adds the colours for the third product to the array list
    boolean coloursAdded = false;

    Context context = getApplicationContext();
    String[] colourResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.white), context.getString(R.string.darkRed),
            context.getString(R.string.skyBlue), context.getString(R.string.darkYellow),
context.getString(R.string.bloodOrange)};

    TechActivity.Colours[] colours = {new TechActivity.Colours(0, colourResources[0]),
            new TechActivity.Colours(1, colourResources[1]), new TechActivity.Colours(2, colourResources[2])
            , new TechActivity.Colours(3, colourResources[3]), new TechActivity.Colours(4, colourResources[4])};

    for(TechActivity.Colours theColours : colours) {
```

```
            listOfColoursOne.add(theColours);
            coloursAdded = true;
        }

        return true;
    }

    private boolean addToColoursListTwo() { // Adds the colours for the fourth product to the array list

        Context context = getApplicationContext();
        String[] resources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.spaceGray), context.getString(R.string.black),
                context.getString(R.string.darkRed), context.getString(R.string.bloodOrange),
context.getString(R.string.white)};

        boolean coloursAdded = false;

        TechActivity.Colours[] colours = {new TechActivity.Colours(0, resources[0]),
                new TechActivity.Colours(1, resources[1]), new TechActivity.Colours(2, resources[2])
                , new TechActivity.Colours(3, resources[3]), new TechActivity.Colours(4, resources[4])};

        for(TechActivity.Colours theColours : colours) {
            listOfColoursTwo.add(theColours);
            coloursAdded = true;
        }

        return true;
    }

    private boolean addToQuantitiesList() {
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};


        TechActivity.Quantities[] quantities = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfQuantitiesOne.add(theQuantities);
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean addToQuantitiesListTwo() {
        Context context = getApplicationContext();
```

```
        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};


        TechActivity.Quantities[] quantities = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfQuantitiesTwo.add(theQuantities);
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean addToSizesListOne() { // Adds the sizes for the third product to the array list
        Context context = getApplicationContext();

        String[] sizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.smallSize), context.getString(R.string.mediumSize), context.getString(R.string.largeSize),
                context.getString(R.string.extraLargeSize)};

        boolean sizesAdded = false;
        Size[] sizes = {new Size(0, sizesResources[0]), new Size(1, sizesResources[1]), new Size(2,
sizesResources[2]),
                new Size(3, sizesResources[3]), new Size(4, sizesResources[4])};

        for (Size theSize : sizes) {

            listOfSizesOne.add(theSize);
            listOfSizesTwo.add(theSize);
            sizesAdded = true;
        }

        return true;
    }

    private void addToBasketThree() {

        Context context = getApplicationContext();

        String[] resources = new String[]{context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(SportsAndOutdoorsActivityTwo.this); // Spinning progress
dialog
        dialog.setTitle(R.string.addingBasket); // Set the title of the dialog
        dialog.setMessage(resources[0]);
```

```
        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products thirdProduct = new Products(current_product_id, sportsOutdoorsTxtTwo.getText().toString(),
thirdSportsOutdoorsColoursMenu.getSelectedItem().toString(), (int)
thirdSportsOutdoorsQuantityMenu.getSelectedItemId(), thirdSportsOutdoorsCostLbl.getText().toString(),
thirdSportsOutdoorsSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id, thirdProduct);
    }

    private void addToBasketFour() {
        Context context = getApplicationContext();
        String[] resources = new String[]{context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(SportsAndOutdoorsActivityTwo.this); // Spinning progress
dialog
        dialog.setTitle(R.string.addingBasket); // Set the title of the dialog
        dialog.setMessage(resources[0]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }
```

```
            dialog.dismiss();
        }
    }).start(); // Starts the thread

    dialog.show();

    // Create an instance for the first product and adds it to the hash map.
    Products fourthProduct = new Products(current_product_id, fourthSportsOutdoorsTxt.getText().toString(),
fourthSportsOutdoorsColourMenu.getSelectedItem().toString(), (int)
fourthSportsOutdoorsQuantityMenu.getSelectedItemId(), fourthSportsOutdoorsCostLbl.getText().toString(),
fourthSportsOutdoorsSizeMenu.getSelectedItem().toString());
    listOfProductsToAddToBasket.put(current_product_id, fourthProduct);
}

@Override
public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
    // Inflate the activities menu
    MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
    activityInflater.inflate(R.menu.homepagemenu, menu);

    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.basket_action_button, menu);

    View view = menu.findItem(R.id.cart_menu).getActionView();

    cartIcon = view.findViewById(R.id.cart_icon);

    cartIcon.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            Intent basketIntent = new Intent(SportsAndOutdoorsActivityTwo.this, BasketActivity.class); // Create a
basket intent
            basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
            startActivity(basketIntent); // Start the intent
        }
    });

    return true;
}

@SuppressLint("SetTextI18n")
@Override
public void onItemSelected(AdapterView<?> parent, View view, int position, long id) { // Routine to determine which
item has been selected
    int[] indexes = {0, 1, 2, 3, 4};
    boolean valueAppended = false;

    Context context = getApplicationContext();
    String[] productResources = new String[]{context.getString(R.string.productCost)};
```

```
if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[0]))) {
    thirdSportsOutdoorsCostLbl.setText(null);
    thirdSportsOutdoorsCostLbl.append(productResources[0] + (thirdProductCosts[0]));
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[1]))) {
    thirdSportsOutdoorsCostLbl.setText(null);

    thirdSportsOutdoorsCostLbl.setText(productResources[0] + (thirdProductCosts[1]));
    valueAppended = true; // Value is appended

}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[2]))) {
    thirdSportsOutdoorsCostLbl.setText(null);
    thirdSportsOutdoorsCostLbl.append(productResources[0] + thirdProductCosts[2]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[3]))) {
    thirdSportsOutdoorsCostLbl.setText(null);
    thirdSportsOutdoorsCostLbl.append(productResources[0] + thirdProductCosts[3]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesOne.get(indexes[4]))) {
    thirdSportsOutdoorsCostLbl.setText(null);
    thirdSportsOutdoorsCostLbl.append(productResources[0] + thirdProductCosts[4]);
    valueAppended = true;
}


if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[0]))) {
    fourthSportsOutdoorsCostLbl.setText(null);
    fourthSportsOutdoorsCostLbl.append(productResources[0] + fourthProductCosts[0]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[1]))) {
    fourthSportsOutdoorsCostLbl.setText(null);
    fourthSportsOutdoorsCostLbl.append(productResources[0] + fourthProductCosts[1]);
    valueAppended = true;
}

else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[2]))) {
    fourthSportsOutdoorsCostLbl.setText(null);
    fourthSportsOutdoorsCostLbl.append(productResources[0] + fourthProductCosts[2]);
    valueAppended = true;
}
```

```
        else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[3]))) {
            fourthSportsOutdoorsCostLbl.setText(null);
            fourthSportsOutdoorsCostLbl.append(productResources[0] + fourthProductCosts[3]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantitiesTwo.get(indexes[4]))) {
            fourthSportsOutdoorsCostLbl.setText(null);
            fourthSportsOutdoorsCostLbl.append(productResources[0] + fourthProductCosts[4]);
            valueAppended = true;
        }
    }

    public boolean onOptionsItemSelected(MenuItem item) {

        try {

            switch (item.getItemId()) {
                case R.id.sportsAndOutdoorsCategory:
                    Intent sportsCategory = new Intent(SportsAndOutdoorsActivityTwo.this,
SportsAndOutdoorsActivity.class);
                    startActivity(sportsCategory);

                    break;

                case R.id.techCategory:
                    Intent techActivity = new Intent(SportsAndOutdoorsActivityTwo.this, TechActivity.class);
                    startActivity(techActivity);
                    break;

                case R.id.clothingCategory:
                    Intent clothingActivity = new Intent(SportsAndOutdoorsActivityTwo.this, ClothingCategory.class);
                    startActivity(clothingActivity);
                    break;

                case R.id.diyCategory:
                    Intent diyActivity = new Intent(SportsAndOutdoorsActivityTwo.this, DIYActivity.class);
                    startActivity(diyActivity);
                    break;

                default:
                    super.onOptionsItemSelected(item);

            }
        } catch (ActivityNotFoundException exc) {
            Log.d(String.valueOf(R.string.error), exc.toString());
        }

        return true;
    }

    @Override
```

```
    public void onNothingSelected(AdapterView<?> parent) {

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }
}
```

Appendix – Application Layer Tech Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;

import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;

import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.SizesAdapter;
import com.example.weshopapplication.R;

import java.util.ArrayList;
import java.util.HashMap;

// Author: Sabin Constantin Lungu
// Purpose of Activity: Shows the products in stock for the tech activity along with the colour to choose from and
quantities.
// Date of Last Modified: 4/2/2020
// Any Bugs?: Currently none. Unit tested recently. 11/11 Tests completed

public class TechActivity extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
    public int current_product_id = 1;
```

```
private TextView firstProductText;
private Thread firstActivityThread;
private ImageView firstProductImg;
private TextView productCost;

private TextView firstProductColour;
private Button firstAddToBasketButton;
private TextView firstProductSizes;
private Spinner firstProductSizesMenu;

private TextView secondProductText;
private ImageView secondProductImg;
private TextView secondProductCost;

private TextView secondProductColour;
private Button secondAddToBasketButton;

private TextView secondProductSizes;
private Spinner secondProductSizesMenu;

private ImageView cartIcon; // Cart Icon should be red once a product is added
private Spinner firstProductColourOptions;
private Spinner firstProductQuantityOptions;

private Spinner secondProductColourOptions;
private Spinner secondProductQuantityOptions;

private ArrayList<Colours> listOfColours = null;
private ArrayList<Quantities> listOfQuantities = null;
private ArrayList<Size> listOfSizes = null;

private QuantitiesArrayAdapter quantitiesCustomAdapter = null;
private ColourArrayAdapter colourArrayAdapter = null;
private SizesAdapter sizesAdapter = null;

private ArrayList<Colours> secondListOfColours = null;
private ArrayList<Quantities> secondListOfQuantities = null;
private ArrayList<Size> secondListOfSizes = null;

private Button nextPageBtn;
private double[] techProductOneCosts = new double[]{0.00, 500.00, 1000.00, 2000.00, 4000.00, 8000.00};
private double[] techProductTwoCosts = new double[]{0.00, 300.00, 600.00, 1200.00, 2400.00, 4800.00};

private HashMap<Integer, Products> listOfProductsToAddToBasket = new HashMap<>();

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tech);

    // INITIALISE COMPONENTS FOR FIRST PRODUCT
```

```
        this.firstActivityThread = new Thread();
        this.firstProductText = findViewById(R.id.firstProductText);
        this.firstProductImg = findViewById(R.id.firstProductImg);
        this.productCost = findViewById(R.id.firstProductCost);

        this.firstProductColourOptions = findViewById(R.id.firstColourSpinner);
        this.firstProductColour = findViewById(R.id.firstProductColorText); // Text View for Product Cost £: (1)

        this.firstProductSizes = findViewById(R.id.firstTechProductSizeLabel);
        this.firstProductSizesMenu = findViewById(R.id.firstProductSizeDropDownMenu);

        this.firstProductColourOptions = findViewById(R.id.firstColourSpinner); // Spinner 1. -> COLOURS
        this.firstProductQuantityOptions = findViewById(R.id.firstQuantitySpinner); // Spinner 2 -> QUANTITIES
        this.firstAddToBasketButton = findViewById(R.id.thirdAddToBasketBtn); // Button: -> ADD TO BASKET
BUTTON 1

        this.secondProductColourOptions = findViewById(R.id.secondColourSpinner);
        this.secondProductQuantityOptions = findViewById(R.id.secondQuantitySpinner);

        this.secondProductSizes = findViewById(R.id.secondProductSizeLabel);
        this.secondProductSizesMenu = findViewById(R.id.secondProductSizeDropDownMenu);

        this.listOfColours = new ArrayList<>();
        this.listOfQuantities = new ArrayList<>();
        this.listOfSizes = new ArrayList<>();

        // Create 2nd array list
        this.secondListOfColours = new ArrayList<>();
        this.secondListOfQuantities = new ArrayList<>();
        this.secondListOfSizes = new ArrayList<>();

        addToColoursList();
        addToQuantitiesList();

        addToSizesList();
        addToSizesTwoList();

        this.colourArrayAdapter = new ColourArrayAdapter(TechActivity.this, listOfColours); // Create an array adapter
for the colours drop down menu
        colourArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        firstProductColourOptions.setAdapter(colourArrayAdapter); // Set its adapter

        this.quantitiesCustomAdapter = new QuantitiesArrayAdapter(TechActivity.this, listOfQuantities);
        quantitiesCustomAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        firstProductQuantityOptions.setAdapter(quantitiesCustomAdapter);

        // Add action listener for first product colour and first product quantity
        firstProductColourOptions.setOnItemSelectedListener(this);
        firstProductQuantityOptions.setOnItemSelectedListener(this);

        this.sizesAdapter = new SizesAdapter(TechActivity.this, listOfSizes);
        sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
```

```
sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

firstProductSizesMenu.setAdapter(sizesAdapter);
firstProductSizesMenu.setOnItemSelectedListener(this);

secondProductSizesMenu.setAdapter(sizesAdapter);
secondProductSizesMenu.setOnItemSelectedListener(this);

secondProductColourOptions.setOnItemSelectedListener(this);
secondProductQuantityOptions.setOnItemSelectedListener(this);

// Initialise components for SECOND PRODUCT
this.secondProductText = findViewById(R.id.secondProductTxt);
this.secondProductImg = findViewById(R.id.appleWatchImg);
this.secondProductCost = findViewById(R.id.secondProductCost);

this.secondProductColour = findViewById(R.id.secondProductColourTxt);
this.secondAddToBasketButton = findViewById(R.id.secondAddToBasketBtn);

// Create adapters for the 2nd product
this.quantitiesCustomAdapter = new QuantitiesArrayAdapter(TechActivity.this, secondListOfQuantities);
this.colourArrayAdapter = new ColourArrayAdapter(TechActivity.this, secondListOfColours);

quantitiesCustomAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
colourArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

secondProductQuantityOptions.setAdapter(quantitiesCustomAdapter);
secondProductColourOptions.setAdapter(colourArrayAdapter);

this.sizesAdapter = new SizesAdapter(TechActivity.this, secondListOfSizes);
sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

secondProductSizesMenu.setAdapter(sizesAdapter);
secondProductSizesMenu.setOnItemSelectedListener(this);

this.nextPageBtn = findViewById(R.id.sportsNextPage);

this.nextPageBtn.setOnClickListener(new View.OnClickListener() { // Add a listener for the next page button
    @Override
    public void onClick(View v) {
        // Create intent for next Tech Activity
        try {

            // Start the next intent
            Intent techActivityTwo = new Intent(TechActivity.this, TechActivityTwo.class);
            startActivity(techActivityTwo);

        } catch (ActivityNotFoundException not) {
            Log.d(String.valueOf(R.string.error), not.toString());
        }
    }
```

```
        });


        firstAddToBasketButton.setOnClickListener(new View.OnClickListener() { // Adds a listener for the first add to
basket button
            @Override
            public void onClick(View v) {

                if (v.getId() == R.id.thirdAddToBasketBtn) { // If the first add to basket button is clicked

                    if (firstProductColourOptions.getSelectedItemPosition() == 0 ||
firstProductQuantityOptions.getSelectedItemPosition() == 0 || firstProductSizesMenu.getSelectedItemPosition() == 0)
{ //

                        AlertDialog.Builder colourErrorOne = new AlertDialog.Builder(TechActivity.this)
                                .setTitle(R.string.error)
                                .setMessage(R.string.errorMsg).setNegativeButton(R.string.ok, new
DialogInterface.OnClickListener() {
                                    @Override

                                    public void onClick(DialogInterface dialog, int which) {

                                        if (dialog != null) {

                                            dialog.dismiss();
                                        }
                                    }
                                });

                        colourErrorOne.show();
                        colourErrorOne.setCancelable(true);
                    }

                    else {

                        addToBasketOne();
                    }
                }
            }
        });

        secondAddToBasketButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View secondButton) {

                if (secondButton.getId() == R.id.secondAddToBasketBtn) {

                    if (secondProductColourOptions.getSelectedItemPosition() == 0 ||
secondProductQuantityOptions.getSelectedItemPosition() == 0 ||
secondProductSizesMenu.getSelectedItemPosition() == 0) {

                        AlertDialog.Builder secondProductColourError = new AlertDialog.Builder(TechActivity.this)
```

```
                        .setTitle(R.string.error)
                        .setMessage(R.string.errorMsg)
                        .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                            @Override

                            public void onClick(DialogInterface dialog, int which) {
                                if (dialog != null) {
                                    dialog.dismiss();
                                }
                            }
                        });

                secondProductColourError.show();
                secondProductColourError.setCancelable(true); // User can click outside the Window to cancel the
dialogue
            } else {
                addToBasketTwo();
            }
        }
    }
});
}

private boolean addToBasketOne() {

    Context context = getApplicationContext();
    String[] temp = new String[]{context.getString(R.string.wait), context.getString(R.string.addingBasket)};

    final ProgressDialog dialog = new ProgressDialog(TechActivity.this);
    dialog.setTitle(temp[1]);
    dialog.setMessage(temp[0]);

    dialog.setCancelable(false);

    dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

    new Thread(new Runnable() { // Create a new thread

        @Override
        public void run() {
            try {

                Thread.sleep(1900);
            } catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();
        }
    }).start();

    dialog.show();
```

```
        Products firstProduct = new Products(current_product_id, firstProductText.getText().toString(),
firstProductColourOptions.getSelectedItem().toString(), (int) firstProductQuantityOptions.getSelectedItemId(),
productCost.getText().toString(), firstProductSizesMenu.getSelectedItem().toString());

        listOfProductsToAddToBasket.put(current_product_id, firstProduct); // Add the product data to the hash map

        return true;
    }

    private boolean addToBasketTwo() { // Adds the second product on the First Tech Activity to the basket

        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.wait), context.getString(R.string.addingBasket)};

        final ProgressDialog dialog = new ProgressDialog(TechActivity.this);
        dialog.setTitle(temp[0]);
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);
        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {
                    Thread.sleep(1900);
                }

catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start();

        dialog.show();

        Products secondProduct = new Products(current_product_id++, secondProductText.getText().toString(),
secondProductColourOptions.getSelectedItem().toString(), (int) secondProductQuantityOptions.getSelectedItemId(),
secondProductCost.getText().toString(), secondProductSizesMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, secondProduct);

        return true;

    }

    private boolean addToColoursList() { // Routine to add the first tech product colours to an array list
        boolean addedColours = false;
```

```
        Context context = getApplicationContext();

        String[] colourResources = new String[]{context.getString(R.string.colour),
context.getString(R.string.spaceGray), context.getString(R.string.secondColour),
context.getString(R.string.thirdColour)};

        Colours[] coloursArray = {new Colours(0, colourResources[0]), new Colours(1, colourResources[1]), new
Colours(2, colourResources[2]), new Colours(3, colourResources[3])};

        for (Colours colours : coloursArray) { // For each colour in the array
            listOfColours.add(colours); // Add it to the array list

            secondListOfColours.add(colours); // Add the second colours
            addedColours = true;
        }

        return true;
    }

    private boolean addToSizesList() {
        boolean addedSizes = false;

        Context context = getApplicationContext();

        String[] sizeResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.sixtyFourGB), context.getString(R.string.oneTwoEight),
context.getString(R.string.twoFiveSix), context.getString(R.string.fiveTwelve)};

        Size[] techSizes = {new Size(0, sizeResources[0]), new Size(1, sizeResources[1]), new Size(2,
sizeResources[2]), new Size(3, sizeResources[3]),
                new Size(4, sizeResources[4])};

        for (Size sizes : techSizes) { // For every size in the object array
            listOfSizes.add(sizes); // Add it to the array list
            addedSizes = true; // Sizes have been added
        }

        return true;
    }

    private boolean addToSizesTwoList() {
        boolean addedSizes = false;

        Context context = getApplicationContext();
        String[] watchResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.fourtyMM), context.getString(R.string.fourtyTwoMM)};


        Size[] watchSizes = {new Size(0, watchResources[0]), new Size(1, watchResources[1]), new Size(2,
watchResources[2])};

        for (Size size : watchSizes) {
```

```
            secondListOfSizes.add(size);
            addedSizes = true;
        }

        return true;
    }

    private boolean addToQuantitiesList() { // Routine to add the quantities to the array list
        boolean addedQuantities = false;

        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};

        Quantities[] firstProductQuantities = { // Create quantities array of objects
                new Quantities(quantitiesResources[0]), new Quantities(quantitiesResources[1]), new
Quantities(quantitiesResources[2])
                , new Quantities(quantitiesResources[3]), new Quantities(quantitiesResources[4]), new
Quantities(quantitiesResources[5])};

        Quantities[] secondProductQuantities = {new Quantities(quantitiesResources[0]), new
Quantities(quantitiesResources[1]), new Quantities(quantitiesResources[2]),
                new Quantities(quantitiesResources[3]), new Quantities(quantitiesResources[4]), new
Quantities(quantitiesResources[5])};

        for (Quantities quantities : firstProductQuantities) { // For each quantity in the array
            listOfQuantities.add(quantities); // Add it to the array list

            addedQuantities = true;
        }

        for (Quantities secondQuantities : secondProductQuantities) { // For each quantities in the second list of
quantities
            secondListOfQuantities.add(secondQuantities); // Add it to the list

            addedQuantities = true;
        }

        return true;
    }

    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
        boolean valueAppended = false;
        int[] indexes = {0, 1, 2, 3, 4, 5}; // Array of indexes

        Context context = getApplicationContext();
        String[] productResource = new String[]{context.getString(R.string.product_cost)};

        if (parent.getItemAtPosition(position).equals(listOfQuantities.get(indexes[0]))) {
```

```
            productCost.setText(null);
            productCost.append(productResource[0] + techProductOneCosts[0]);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(indexes[1]))) { // If the value at index 1 is
chosen
            productCost.setText(null); // Flush out the product cost
            productCost.append(productResource[0] + techProductOneCosts[1]); // Append the cost

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(indexes[2]))) {
            productCost.setText(null);
            productCost.append(productResource[0] + techProductOneCosts[2]);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(indexes[3]))) {
            productCost.setText(null);
            productCost.append(productResource[0] + techProductOneCosts[3]);
            valueAppended = true;

        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(indexes[4]))) {
            productCost.setText(null);
            productCost.append(productResource[0] + techProductOneCosts[4]);
            valueAppended = true;

        }

        if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[0]))) {
            secondProductCost.setText(null);
            secondProductCost.append(productResource[0] + techProductTwoCosts[0]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[1]))) {
            secondProductCost.setText(null);
            secondProductCost.append(productResource[0] + techProductTwoCosts[1]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[2]))) {
            secondProductCost.setText(null);
            secondProductCost.append(productResource[0] + techProductTwoCosts[2]);
```

```
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[3]))) { // If the item at index
3 is chosen
            secondProductCost.setText(null); // Set text to null
            secondProductCost.append(productResource[0] + techProductTwoCosts[3]); // Then append the cost +
string.
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[4]))) {
            secondProductCost.setText(null);
            secondProductCost.append(productResource[0] + techProductTwoCosts[4]);
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(indexes[5]))) {
            secondProductCost.setText(null);
            secondProductCost.append(productResource[0] + techProductTwoCosts[5]);
        }

        else {
            valueAppended = false;
        }
    }

    public boolean onOptionsItemSelected(MenuItem item) { // Routine that gets an item selected from the menu

        try {

            switch (item.getItemId()) {

                case R.id.sportsAndOutdoorsCategory: // When the sports and activity category is chosen
                    Intent sportsAndOutdoors = new Intent(TechActivity.this, SportsAndOutdoorsActivity.class); // Create
intent for sports and outdoors
                    Thread.sleep(1); // Sleep for 1ms
                    startActivity(sportsAndOutdoors); // Start the activity

                    return true; // Returns true

                case R.id.techCategory: // If the tech category is chosen
                    Intent techCategory = new Intent(TechActivity.this, TechActivity.class); // Go to the tech category
                    Thread.sleep(1);
                    startActivity(techCategory); // Start that activity

                    return true;

                case R.id.clothingCategory:
                    Intent clothingCategory = new Intent(TechActivity.this, ClothingCategory.class);
                    Thread.sleep(1);
                    startActivity(clothingCategory);

                    return true;
```

```
            case R.id.diyCategory:
                Intent diyCategory = new Intent(TechActivity.this, DIYActivity.class);
                Thread.sleep(1);
                startActivity(diyCategory);

                return true;

            default:
                return super.onOptionsItemSelected(item); // Return base item if no option is chosen

        }

    } catch (ActivityNotFoundException act) { // Catch error

        Log.d(String.valueOf(R.string.error), act.toString()); // Log error if there is a problem.


    } catch (InterruptedException act) {

        Log.d(String.valueOf(R.string.error), act.toString());
    }

    return true;
  }

  public static class Quantities {
    private String quantity; // Quantity variable

    public Quantities(String quantity) { // Parameterised constructor that creates the object and the data when this is called.
        this.quantity = quantity;
    }

    public String getQuantity() { // Gets the quantity
        return this.quantity;
    }

    public void setQuantity(String quantity) {
        this.quantity = quantity;
    }

    public String toString() { // Returns the data.
        return " " + this.quantity + " ";
    }
  }

  // Anonymous inner class Quantities that is used to add the quantities to the drop-down menu. This class will be reused throughout other activities in order to retrieve specific data.

  @Override
```

```
public void onBackPressed() {
    super.onBackPressed();
}

public void onDestroy() { // On destroy method
    super.onDestroy();
}

public void onResume() { // On resume method when the activity resumes
    super.onResume();
}

@Override
public void onNothingSelected(AdapterView<?> parent) {

}

public boolean onCreateOptionsMenu(Menu menu) {

    // Inflate the activities menu
    MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
    activityInflater.inflate(R.menu.homepagemenu, menu);

    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.basket_action_button, menu);

    View view = menu.findItem(R.id.cart_menu).getActionView();

    cartIcon = view.findViewById(R.id.cart_icon);

    cartIcon.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            Intent basketIntent = new Intent(TechActivity.this, BasketActivity.class);
            basketIntent.putExtra("map", listOfProductsToAddToBasket);
            startActivity(basketIntent);
        }
    });

    return true;
}

// Anonymous inner classes that will be used later on in the basket activity and the payment form

public static class Colours { // Anonymous inner class of colours
    private int index;
    private String colour;

    public Colours(int index, String colour) {
        this.index = index;
        this.colour = colour;
```

```
    }

    public int getIndex() { // Returns the index
        return this.index;
    }

    public void setIndex(int index) {
        this.index = index;
    }

    public String getColour() {
        return this.colour;
    }

    public void setColour(String colour) {
        this.colour = colour;
    }

    @Override
    public String toString() {
        return " " + this.colour;
    }
}

public static class Size {
    private int index;
    private String theSize;

    public Size(int index, String theSize) {
        this.index = index;
        this.theSize = theSize;
    }

    public int getIndex() {
        return index;
    }

    public void setIndex(int index) {
        this.index = index;
    }

    public String getTheSize() {
        return theSize;
    }

    public void setTheSize(String theSize) {
        this.theSize = theSize;
    }

    @Override
    public String toString() {
        return " " + this.theSize;
```

```
    }
  }
}
```

Appendix – Application Layer Tech Activity 2 Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.HashMap;

public class TechActivityTwo extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
    // Components for the Third Product
    private TextView thirdProductTextView;
    private ImageView thirdProductImage;

    private TextView thirdProductCostTxt;
    private TextView thirdProductColourLbl;

    private ImageView cartIcon;
    private Spinner thirdProductDropDown;
```

```
    private TextView thirdQuantityLabel;
    private Spinner thirdQuantityDropDown;
    private Button thirdAddToBasketButton;

    private TextView thirdSizeLbl;
    private Spinner thirdSizeDropDownMenu;

    // Components for the Fourth Product to sell
    private TextView fourthProductTextView;
    private ImageView fourthProductImage;

    private TextView fourthProductCost;
    private TextView fourthProductColourLbl;
    private Spinner fourthProductColourSpinner;

    private TextView fourthProductQuantity;
    private Spinner fourthProductQuantityDropDown;
    private TextView fourthProductSizeLbl;

    private Spinner fourthProductMenu;
    private Button fourthAddToBasketButton;

    // An array list of colours, quantities and capacity
    private ArrayList<TechActivity.Colours> listOfColours;
    private ArrayList<TechActivity.Quantities> listOfQuantities;
    private ArrayList<Size> listOfSizes;

    private ArrayList<TechActivity.Colours> secondListOfColours;
    private ArrayList<TechActivity.Quantities> secondListOfQuantities;
    private ArrayList<Size> secondListOfSizes;

    private double quantity_zero_cost = 0.0;
    private double quantity_one_cost = 249.99;

    // Formulae to calculate price & Capacity
    // COST FOR THE FIRST PRODUCT
    private double quantity_two_cost = 2 * quantity_one_cost; // Quantity 2 is 3 times the price of 1 quantity.
    private double quantity_three_cost = 3 * quantity_one_cost;

    private double quantity_four_cost = 4 * quantity_one_cost;
    private double quantity_five_cost = 5 * quantity_one_cost;

    private double product_four_zero_cost = 0.00;
    private double product_four_one_cost = 750.00;
    private double product_four_two_cost = 2 * product_four_one_cost;

    private double product_four_three_cost = 3 * product_four_one_cost;
    private double product_four_four_cost = 4 * product_four_one_cost;

    // Array adapters to aid the addition of colours, capacity and colours to the array list
    private QuantitiesArrayAdapter quantitiesAdapter;
    private ColourArrayAdapter colourArrayAdapter;
```

```
private SizeArrayAdapter sizeArrayAdapter;

// Boolean variables that holds either true or false
private boolean addedColours = false;
private boolean addedQuantities = false;
private boolean addedCapacities = false;

public int current_product_id = 1;
private HashMap<Integer, Products> listOfProductsToAdd = new HashMap<>();

@Override
protected void onCreate(Bundle savedInstanceState) { // On create method
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_tech_two);

    // Initialise components for the THIRD PRODUCT

    this.thirdProductTextView = findViewById(R.id.thirdProductTextView);
    this.thirdProductImage = findViewById(R.id.thirdProductImg);
    this.thirdProductCostTxt = findViewById(R.id.thirdProductCostTxt);
    this.thirdProductColourLbl = findViewById(R.id.thirdColourLbl);
    this.thirdProductDropDown = findViewById(R.id.thirdColourDropDownMenu);

    this.thirdSizeDropDownMenu = findViewById(R.id.thirdProductSizeDropDown);
    this.thirdSizeLbl = findViewById(R.id.thirdProductSizeLbl);

    this.thirdQuantityLabel = findViewById(R.id.thirdQuantityLbl);
    this.thirdQuantityDropDown = findViewById(R.id.thirdQuantityDropDownMenu);
    this.thirdAddToBasketButton = findViewById(R.id.thirdAddToBasketBtn);

    // INITIALISE COMPONENTS FOR THE FOURTH PRODUCT

    this.fourthProductTextView = findViewById(R.id.fourthProductTitleLbl);
    this.fourthProductImage = findViewById(R.id.fourthProductImg);
    this.fourthProductCost = findViewById(R.id.fourthProductImgCost);
    this.fourthProductColourLbl = findViewById(R.id.fourthProductColourLabel);

    this.fourthProductSizeLbl = findViewById(R.id.fourthProductSizeLbl);

    this.fourthProductColourSpinner = findViewById(R.id.fourthProductDropDownMenu);
    this.fourthProductQuantity = findViewById(R.id.fourthProductQuantityLbl);
    this.fourthAddToBasketButton = findViewById(R.id.fourthAddToBasketButton);

    this.fourthProductQuantityDropDown = findViewById(R.id.fourthProductQuantityDropDown);

    this.fourthProductSizeLbl = findViewById(R.id.fourthProductSizeLbl);
    this.fourthProductMenu = findViewById(R.id.fourthProductSizeMenu);

    // Put array list on the heap
    this.listOfColours = new ArrayList<>();
    this.listOfQuantities = new ArrayList<>();
    this.listOfSizes = new ArrayList<>();
```

```
this.secondListOfColours = new ArrayList<>();
this.secondListOfQuantities = new ArrayList<>();
this.secondListOfSizes = new ArrayList<>();

addToColoursList(); // Method to add to the colours array list
addToQuantitiesList();

addThirdProductSizes();
addFourthProductSizes();

// SET UP THE THIRD PRODUCT QUANTITIES DROP DOWN MENU TO SHOW
this.quantitiesAdapter = new QuantitiesArrayAdapter(TechActivityTwo.this, listOfQuantities);
quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
thirdQuantityDropDown.setAdapter(quantitiesAdapter);
thirdQuantityDropDown.setOnItemSelectedListener(TechActivityTwo.this);

// SET UP THE THIRD PRODUCT COLOUR SPINNER DROP DOWN MENU TO SHOW
this.colourArrayAdapter = new ColourArrayAdapter(TechActivityTwo.this, listOfColours);
colourArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

thirdProductDropDown.setAdapter(colourArrayAdapter);
thirdProductDropDown.setOnItemSelectedListener(TechActivityTwo.this);

// SET UP THE FOURTH PRODUCT COLOUR DROP DOWN MENU TO SHOW
this.colourArrayAdapter = new ColourArrayAdapter(TechActivityTwo.this, secondListOfColours);
colourArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

fourthProductColourSpinner.setAdapter(colourArrayAdapter);
fourthProductColourSpinner.setOnItemSelectedListener(TechActivityTwo.this);

// SET UP QUANTITY FOR FOURTH PRODUCT
this.quantitiesAdapter = new QuantitiesArrayAdapter(TechActivityTwo.this, secondListOfQuantities);
quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

fourthProductQuantityDropDown.setAdapter(quantitiesAdapter);
fourthProductQuantityDropDown.setOnItemSelectedListener(TechActivityTwo.this);

this.sizeArrayAdapter = new SizeArrayAdapter(TechActivityTwo.this, listOfSizes);
sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

thirdSizeDropDownMenu.setAdapter(sizeArrayAdapter);
thirdSizeDropDownMenu.setOnItemSelectedListener(TechActivityTwo.this);

this.sizeArrayAdapter = new SizeArrayAdapter(TechActivityTwo.this, secondListOfSizes);
sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

fourthProductMenu.setAdapter(sizeArrayAdapter);
fourthProductMenu.setOnItemSelectedListener(TechActivityTwo.this);

this.thirdAddToBasketButton.setOnClickListener(new View.OnClickListener() { // Add Listener for third button
    @Override
```

```
        public void onClick(View v) {

            if (v.getId() == R.id.thirdAddToBasketBtn) {

                if (thirdProductDropDown.getSelectedItemPosition() == 0 ||
thirdQuantityDropDown.getSelectedItemPosition() == 0 || thirdSizeDropDownMenu.getSelectedItemPosition() == 0) {

                    AlertDialog.Builder colourError = new AlertDialog.Builder(TechActivityTwo.this).setTitle(R.string.error)
                            .setMessage(R.string.errorMsg)
                            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {

                                @Override
                                public void onClick(DialogInterface dialog, int which) {
                                    if (dialog != null) {

                                        dialog.dismiss();
                                    }
                                }
                            });

                    colourError.show(); // Show the error
                    colourError.setCancelable(true); // Set cancelable to true
                } else {

                    addProductThreeToBasket();
                }

            }
        }
    });

    this.fourthAddToBasketButton.setOnClickListener(new View.OnClickListener() { // Add action listener to the
fourth button
        @Override

        public void onClick(View view) {
            if (view.getId() == R.id.fourthAddToBasketButton) {

                if (fourthProductColourSpinner.getSelectedItemPosition() == 0 ||
fourthProductQuantityDropDown.getSelectedItemPosition() == 0 || fourthProductMenu.getSelectedItemPosition() ==
0) {
                    AlertDialog.Builder error = new AlertDialog.Builder(TechActivityTwo.this).setTitle(R.string.error)

                            .setMessage(R.string.errorMsg)

                            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface dialog, int which) {
                                    if (dialog != null) {
                                        dialog.dismiss();
                                    }
```

```
                        }
                    });

                error.show();
                error.setCancelable(true);
            } else {
                addProductFourToBasket();
            }
        }
    });
}

private void addProductThreeToBasket() { // Adds the third product to basket

    Context context = getApplicationContext();
    String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

    final ProgressDialog dialog = new ProgressDialog(TechActivityTwo.this);
    dialog.setTitle(temp[0]);
    dialog.setMessage(temp[1]);

    dialog.setCancelable(false);

    dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

    new Thread(new Runnable() { // Create a new thread

        @Override
        public void run() {
            try {

                Thread.sleep(1900);
            }

            catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();

        }
    }).start();

    dialog.show(); // Show the progress bar

    Products thirdTechProduct = new Products(current_product_id, thirdProductTextView.getText().toString(),
thirdProductDropDown.getSelectedItem().toString(), (int) thirdQuantityDropDown.getSelectedItemId(),
thirdProductCostTxt.getText().toString(), thirdSizeDropDownMenu.getSelectedItem().toString());
    listOfProductsToAdd.put(current_product_id, thirdTechProduct);
}
```

```
    private void addProductFourToBasket() { // Adds the fourth product to the basket

        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(TechActivityTwo.this);
        dialog.setTitle(temp[0]);
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900);
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start();
        dialog.show(); // Show the progress bar

        Products fourthTechProduct = new Products(current_product_id++, fourthProductTextView.getText().toString(),
fourthProductColourSpinner.getSelectedItem().toString(), (int) fourthProductQuantityDropDown.getSelectedItemId(),
fourthProductCost.getText().toString(), fourthProductMenu.getSelectedItem().toString());
        listOfProductsToAdd.put(current_product_id++, fourthTechProduct);

    }

    private boolean addToColoursList() { // Routine that adds the colours to the array list

        Context context = getApplicationContext();
        String[] techTwoColourResources = new String[]{context.getString(R.string.colour),
context.getString(R.string.white), context.getString(R.string.black), context.getString(R.string.salmon),
context.getString(R.string.limeGreen), context.getString(R.string.rubyRed),
context.getString(R.string.sportsThirdColour)};


        TechActivity.Colours[] firstColoursArray = {new TechActivity.Colours(0, techTwoColourResources[0]), new
TechActivity.Colours(1, techTwoColourResources[1]), new TechActivity.Colours(2, techTwoColourResources[2])};

        TechActivity.Colours[] secondColoursArray = {new TechActivity.Colours(0, techTwoColourResources[0]), new
TechActivity.Colours(1, techTwoColourResources[3]), new TechActivity.Colours(2, techTwoColourResources[4]),
                new TechActivity.Colours(3, techTwoColourResources[5]), new TechActivity.Colours(4,
techTwoColourResources[6])};
```

```
        for (TechActivity.Colours colours : firstColoursArray) {
            listOfColours.add(colours);
            addedColours = true;
        }

        for (TechActivity.Colours secondColours : secondColoursArray) {
            secondListOfColours.add(secondColours);
            addedColours = true;
        }

        return true;
    }

    private boolean addThirdProductSizes() { // Adds the required sizes to the third product
        boolean addedSizes = false;
        Context context = getApplicationContext();
        String[] techTwoSizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.small), context.getString(R.string.medium), context.getString(R.string.large)};

        Size[] sizes = {new Size(0, techTwoSizesResources[0]), new Size(1, techTwoSizesResources[1]), new Size(2,
techTwoSizesResources[2]), new Size(3, techTwoSizesResources[3])};

        for (Size theSizes : sizes) {
            listOfSizes.add(theSizes);
            addedSizes = true;
        }

        return true;
    }

    private boolean addFourthProductSizes() {
        boolean addedSizes = false;

        Context context = getApplicationContext();

        String[] techSizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.sixtyFourGB), context.getString(R.string.oneTwoEight),
context.getString(R.string.twoFiveSix), context.getString(R.string.fiveTwelve)};

        Size[] fourthProdSizes = {new Size(0, techSizesResources[0]), new Size(1, techSizesResources[1]), new
Size(2, techSizesResources[2]), new Size(3, techSizesResources[3]), new Size(4, techSizesResources[4])};

        for (Size sizes : fourthProdSizes) {
            secondListOfSizes.add(sizes);
            addedSizes = true;
        }

        return true;
    }

    private boolean addToQuantitiesList() { // Routine that adds the quantities to the array list
```

```
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two), context.getString(R.string.three), context.getString(R.string.four),
context.getString(R.string.five)};


        TechActivity.Quantities[] quantitiesArray = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        TechActivity.Quantities[] secondProductQuantities = {new TechActivity.Quantities(quantitiesResources[0]), new
TechActivity.Quantities(quantitiesResources[1]), new TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};


        for (TechActivity.Quantities qty : quantitiesArray) { // For each quantity in the array
            listOfQuantities.add(qty);
            addedQuantities = true; // Quantities
        }

        for (TechActivity.Quantities qty2 : secondProductQuantities) {
            secondListOfQuantities.add(qty2);
            addedQuantities = true;
        }

        return true;
    }

    @Override
    protected void onDestroy() { // End the activity
        super.onDestroy();
    }

    @Override
    public void onBackPressed() { // Click back button
        super.onBackPressed();
    }

    @Override
    protected void onResume() { // Resumes the activity
        super.onResume();
    }

    @Override
    protected void onRestart() { // When restarted.
        super.onRestart();
    }
```

```
    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) { // Routine that determines
which item has been selected
        boolean valueAppended = false;

        int[] quantityIndexes = {0, 1, 2, 3, 4, 5};
        Context context = getApplicationContext();

        String[] productResource = new String[]{context.getString(R.string.product_cost)};


        if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[0]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_zero_cost);

            valueAppended = true;
        }

        // If the quantity at index 1 is chosen
        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[1]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_one_cost);
            valueAppended = true; // Value is appended

        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[2]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_two_cost);

            valueAppended = true;

        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[3]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_three_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[4]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_four_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfQuantities.get(quantityIndexes[5]))) {
            thirdProductCostTxt.setText(null);
            thirdProductCostTxt.append(productResource[0] + quantity_five_cost);
```

```
            valueAppended = true;
        }

        if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[0]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + product_four_zero_cost);
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[1]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + product_four_one_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[2]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + product_four_two_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[3]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + quantity_three_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[4]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + quantity_four_cost);

            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(secondListOfQuantities.get(quantityIndexes[5]))) {
            fourthProductCost.setText(null);
            fourthProductCost.append(productResource[0] + quantity_five_cost);
        }

    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }
```

```
@Override
public boolean onCreateOptionsMenu(Menu menu) { // Creates the menu bar
    MenuInflater activityInflater = getMenuInflater();
    activityInflater.inflate(R.menu.homepagemenu, menu);

    MenuInflater basketButtonInflater = getMenuInflater();
    basketButtonInflater.inflate(R.menu.basket_action_button, menu);

    View cartView = menu.findItem(R.id.cart_menu).getActionView(); // Get the action view for the cart

    cartIcon = cartView.findViewById(R.id.cart_icon);

    cartIcon.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            try {

                Intent basketIntent = new Intent(TechActivityTwo.this, BasketActivity.class);
                basketIntent.putExtra("map", listOfProductsToAdd);
                startActivity(basketIntent); // Start the basket intent

            } catch (ActivityNotFoundException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }
        }
    });

    return true;
}

@Override
public boolean onOptionsItemSelected(@NonNull MenuItem item) {

    try {

        switch (item.getItemId()) {

            case R.id.sportsAndOutdoorsCategory:
                Intent sportsAndOutdoorsActivity = new Intent(TechActivityTwo.this, SportsAndOutdoorsActivity.class);
                startActivity(sportsAndOutdoorsActivity);

                return true;

            case R.id.techCategory:
                Intent techCategory = new Intent(TechActivityTwo.this, TechActivity.class);
                startActivity(techCategory);

                return true;

            case R.id.clothingCategory:
```

```
                Intent clothingCategory = new Intent(TechActivityTwo.this, ClothingCategory.class);
                startActivity(clothingCategory);

                return true;

            case R.id.diyCategory:

                Intent diyCategory = new Intent(TechActivityTwo.this, DIYActivity.class);
                startActivity(diyCategory);

                return true;

            default:
                return super.onOptionsItemSelected(item);

        }

    } catch (ActivityNotFoundException exc) {

        Log.d(String.valueOf(R.string.error), exc.toString());
    }

    return true;
  }
}
```

Appendix – Application Layer Clothing Category Code One

```
package com.example.weshopapplication.ApplicationLayer;

import android.annotation.SuppressLint;
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
```

```
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;

import java.util.ArrayList;
import java.util.HashMap;

// Author of Application: Sabin Constantin Lungu
// Purpose of Application: Displays the clothing category products.
// Date of Last Modification: 02/03/2020
// Any Errors? None Yet.

public class ClothingCategory extends AppCompatActivity implements AdapterView.OnItemSelectedListener { //
Class implements the item selected listener methods from the adapter view class.
    private int current_product_id = 1;
    private TextView clothingFirstProductTxt;
    private ImageView clothingFirstProductImg;
    private TextView clothingFirstProductCostLbl;

    private TextView clothingFirstProductColourLbl;
    private Spinner clothingFirstProductColourMenu;

    private TextView clothingFirstProductSizeLbl; // The size label of the first clothing product.
    private Spinner clothingFirstProductSizeMenu; // The size menu of the clothing activity.

    private TextView clothingFirstProductQuantityLbl;
    private Spinner clothingFirstProductQuantityMenu;

    private Button clothingFirstProductAddToBasketBtn;

    private TextView clothingSecondProductTxt;
    private ImageView clothingSecondProductImg;
    private TextView clothingSecondProductCostLbl;

    private TextView clothingSecondProductColourLbl;
    private Spinner clothingSecondProductColourMenu;

    private TextView clothingSecondProductSizeLbl;
    private Spinner clothingSecondProductSizeMenu;

    private TextView clothingSecondProductQuantityLbl;
    private Spinner clothingSecondProductQuantityMenu;

    private Button clothingSecondProductAddToBasketBtn;

    private double[] clothingProductOneCosts = new double[]{0.00, 25.00, 50.00, 150.00, 450.00, 1350.00}; // Clothing
product one costs.
```

```
    private double[] clothingProductTwoCosts = new double[]{0.00, 30.00, 60.00, 120.00, 240.00, 480.00};

    private QuantitiesArrayAdapter quantitiesAdapter;
    private ColourArrayAdapter coloursAdapter;
    private SizeArrayAdapter sizeArrayAdapter;

    // Array List of colours, clothing and sizes.
    private ArrayList<TechActivity.Colours> listOfClothingColoursOne = null;
    private ArrayList<Size> listOfClothingSizesOne = null;
    private ArrayList<TechActivity.Quantities> listOfClothingQuantitiesOne = null;

    private ArrayList<TechActivity.Colours> listOfClothingColoursTwo = null;
    private ArrayList<Size> listOfClothingSizesTwo = null;
    private ArrayList<TechActivity.Quantities> listOfClothingQuantitiesTwo = null;

    private ImageView cartIcon; // The cart icon that represents the basket
    private transient boolean coloursAdded = false; // Determines if the colours have been added.

    private Button nextPageBtn;
    private HashMap<Integer, Products> listOfProductsToAddToBasket; // A Hash Map that stores a list of products to
add to the basket.

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_clothing_category);

        this.clothingFirstProductTxt = findViewById(R.id.clothingFirstProductTxt);
        this.clothingFirstProductImg = findViewById(R.id.clothingFirstProductImg);
        this.clothingFirstProductCostLbl = findViewById(R.id.clothingFirstProductCostLbl);

        this.clothingFirstProductColourLbl = findViewById(R.id.clothingFirstProductColourLbl);
        this.clothingFirstProductColourMenu = findViewById(R.id.clothingFirstProductColourMenu);

        this.clothingFirstProductSizeLbl = findViewById(R.id.clothingFirstProductSizeLbl);
        this.clothingFirstProductSizeMenu = findViewById(R.id.clothingFirstProductSizeMenu);

        this.clothingFirstProductQuantityLbl = findViewById(R.id.clothingFirstProductQuantityLbl);
        this.clothingFirstProductQuantityMenu = findViewById(R.id.clothingFirstProductQuantityMenu);
        this.clothingFirstProductAddToBasketBtn = findViewById(R.id.clothingFirstProductAddToBasketBtn);

        this.clothingSecondProductTxt = findViewById(R.id.clothingSecondProductTxt);
        this.clothingSecondProductImg = findViewById(R.id.clothingSecondProductImg);
        this.clothingSecondProductCostLbl = findViewById(R.id.clothingSecondProductCostLbl);

        this.clothingSecondProductColourLbl = findViewById(R.id.clothingSecondProductColourLbl);
        this.clothingSecondProductColourMenu = findViewById(R.id.clothingSecondProductColourMenu);

        this.clothingSecondProductSizeLbl = findViewById(R.id.clothingSecondProductSizeLbl);
        this.clothingSecondProductSizeMenu = findViewById(R.id.clothingSecondProductSizeMenu);

        this.clothingSecondProductQuantityLbl = findViewById(R.id.clothingSecondProductQuantityLbl);
```

```
this.clothingSecondProductQuantityMenu = findViewById(R.id.clothingSecondProductQuantityMenu);
this.clothingSecondProductAddToBasketBtn = findViewById(R.id.clothingSecondProductAddToBasketBtn);

this.nextPageBtn = findViewById(R.id.clothingNextPageBtn);

this.listOfClothingColoursOne = new ArrayList<>();
this.listOfClothingSizesOne = new ArrayList<>();
this.listOfClothingQuantitiesOne = new ArrayList<>();

this.listOfClothingColoursTwo = new ArrayList<>();
this.listOfClothingSizesTwo = new ArrayList<>();
this.listOfClothingQuantitiesTwo = new ArrayList<>();

this.listOfProductsToAddToBasket = new HashMap<Integer, Products>(); // Creates a new hash map.

addToColoursList();
addToSizesList();

addToQuantitiesList();
addToQuantitiesListTwo();

// Set-up Adapters.
this.coloursAdapter = new ColourArrayAdapter(ClothingCategory.this, listOfClothingColoursOne);
coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

clothingFirstProductColourMenu.setAdapter(coloursAdapter);
clothingFirstProductColourMenu.setOnItemSelectedListener(ClothingCategory.this);

this.sizeArrayAdapter = new SizeArrayAdapter(ClothingCategory.this, listOfClothingSizesOne);
sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

clothingFirstProductSizeMenu.setAdapter(sizeArrayAdapter);
clothingFirstProductSizeMenu.setOnItemSelectedListener(this);

this.quantitiesAdapter = new QuantitiesArrayAdapter(ClothingCategory.this, listOfClothingQuantitiesOne);
quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

clothingFirstProductQuantityMenu.setAdapter(quantitiesAdapter);
clothingFirstProductQuantityMenu.setOnItemSelectedListener(this);

this.coloursAdapter = new ColourArrayAdapter(ClothingCategory.this, listOfClothingColoursTwo);
coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

clothingSecondProductColourMenu.setAdapter(coloursAdapter);
clothingSecondProductColourMenu.setOnItemSelectedListener(ClothingCategory.this);

this.sizeArrayAdapter = new SizeArrayAdapter(ClothingCategory.this, listOfClothingSizesTwo);
sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

clothingSecondProductSizeMenu.setAdapter(sizeArrayAdapter);
clothingSecondProductSizeMenu.setOnItemSelectedListener(this);
```

```
        this.quantitiesAdapter = new QuantitiesArrayAdapter(ClothingCategory.this, listOfClothingQuantitiesTwo);
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingSecondProductQuantityMenu.setAdapter(quantitiesAdapter);
        clothingSecondProductQuantityMenu.setOnItemSelectedListener(this);

        this.clothingFirstProductAddToBasketBtn.setOnClickListener(new View.OnClickListener() { // Add action listener
to the first clothing add to product button
            @Override

            public void onClick(View firstButton) {
                if (firstButton.getId() == R.id.clothingFirstProductAddToBasketBtn) {

                    if (clothingFirstProductColourMenu.getSelectedItemPosition() == 0 ||
clothingFirstProductSizeMenu.getSelectedItemPosition() == 0 ||
clothingFirstProductQuantityMenu.getSelectedItemPosition() == 0) {
                        AlertDialog.Builder error = new AlertDialog.Builder(ClothingCategory.this)
                                .setTitle(R.string.error)
                                .setMessage(R.string.errorMsg)

                                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int which) {
                                        if (dialog != null) {
                                            dialog.dismiss();
                                        }
                                    }
                                });

                        error.show(); // Show the error.
                        error.setCancelable(true);
                    }

                    else {
                        clothingAddToBasketOne();
                    }
                }
            }
        });

        this.clothingSecondProductAddToBasketBtn.setOnClickListener(new View.OnClickListener() { // Adds an action
listener to the second product add to basket button.
            @Override
            public void onClick(View secondButton) {
                if (secondButton.getId() == R.id.clothingSecondProductAddToBasketBtn) {

                    if (clothingSecondProductColourMenu.getSelectedItemPosition() == 0 ||
clothingSecondProductSizeMenu.getSelectedItemPosition() == 0 ||
clothingSecondProductQuantityMenu.getSelectedItemPosition() == 0) {
                        AlertDialog.Builder error = new AlertDialog.Builder(ClothingCategory.this)
                                .setTitle(R.string.error)
                                .setMessage(R.string.errorMsg)
```

```
                        .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {
                                if (dialog != null) {
                                    dialog.dismiss();
                                }
                            }
                        });

                error.show();
                error.setCancelable(true);
            }

            else {
                clothingAddToBasketTwo(); // Otherwise if an option is selected, add the product to the basket.
            }
        }
    });

    this.nextPageBtn.setOnClickListener(new View.OnClickListener() { // Adds an action listener for the next page
button.
        @Override

        public void onClick(View v) {
            try {
                Intent intent = new Intent(ClothingCategory.this, ClothingActivityTwo.class);
                startActivity(intent);

            }

            catch (ActivityNotFoundException exc) { // Catch the error if the activity does not exist.
                Log.d(String.valueOf(R.string.error), exc.toString());
            }
        }
    });
}

@SuppressLint("SetTextI18n")
@Override
public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
    boolean valueAppended = false;

    int[] indexes = new int[]{0, 1, 2, 3, 4};

    Context context = getApplicationContext();
    String[] productResources = new String[]{context.getString(R.string.productCost)};

    if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[0]))) {
        clothingFirstProductCostLbl.setText(null);
        clothingFirstProductCostLbl.append(productResources[0] + clothingProductOneCosts[0]);
        valueAppended = true;
```

```
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[1]))) {
        clothingFirstProductCostLbl.setText(null);
        clothingFirstProductCostLbl.append(productResources[0] + clothingProductOneCosts[1]);
        valueAppended = true; // Value is appended

    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[2]))) {
        clothingFirstProductCostLbl.setText(null);
        clothingFirstProductCostLbl.append(productResources[0] + clothingProductOneCosts[2]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[3]))) {
        clothingFirstProductCostLbl.setText(null);
        clothingFirstProductCostLbl.append(productResources[0] + clothingProductOneCosts[3]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[4]))) {
        clothingFirstProductCostLbl.setText(null);
        clothingFirstProductCostLbl.append(productResources[0] + clothingProductOneCosts[4]);
        valueAppended = true;
    }


    if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[0]))) {
        clothingSecondProductCostLbl.setText(null);
        clothingSecondProductCostLbl.append(productResources[0] + clothingProductTwoCosts[0]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[1]))) {
        clothingSecondProductCostLbl.setText(null);
        clothingSecondProductCostLbl.append(productResources[0] + clothingProductTwoCosts[1]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[2]))) {
        clothingSecondProductCostLbl.setText(null);
        clothingSecondProductCostLbl.append(productResources[0] + clothingProductTwoCosts[2]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[3]))) {
        clothingSecondProductCostLbl.setText(null);
        clothingSecondProductCostLbl.append(productResources[0] + clothingProductTwoCosts[3]);
        valueAppended = true;
    }
```

```
        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[4]))) {
            clothingSecondProductCostLbl.setText(null);
            clothingSecondProductCostLbl.append(productResources[0] + clothingProductTwoCosts[4]);
            valueAppended = true;
        }
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }

    private boolean addToColoursList() {
        Context context = getApplicationContext();

        String[] clothingResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.brown), context.getString(R.string.navyBlue),
                context.getString(R.string.darkRed), context.getString(R.string.checkeredBlack)};

        TechActivity.Colours[] colours = new TechActivity.Colours[]{new TechActivity.Colours(0, clothingResources[0]),
new TechActivity.Colours(1, clothingResources[1]), new TechActivity.Colours(2, clothingResources[2]),
                new TechActivity.Colours(3, clothingResources[3]), new TechActivity.Colours(4, clothingResources[4])};

        for (TechActivity.Colours theColours : colours) { // For every colour in the object array

            listOfClothingColoursOne.add(theColours);
            listOfClothingColoursTwo.add(theColours);

            coloursAdded = true;
        }

        return true;
    }

    private boolean addToSizesList() {
        boolean sizes_added = false;
        Context context = getApplicationContext();

        String[] clothingSizes = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.thirtyFourRegular), context.getString(R.string.thirtyEightRegular)
                , context.getString(R.string.forty), context.getString(R.string.fortyTwo)};

        Size[] sizes = new Size[]{new Size(0, clothingSizes[0]), new Size(1, clothingSizes[1]), new Size(2,
clothingSizes[2]), new Size(3, clothingSizes[3]), new Size(4, clothingSizes[4])};

        for (Size theSizes : sizes) {
```

```
            listOfClothingSizesOne.add(theSizes);
            listOfClothingSizesTwo.add(theSizes);
            sizes_added = true;

        }

        return true;
    }

    private boolean addToQuantitiesList() {
        boolean quantitiesAdded = false;
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two),
                context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantitiesResources[0]), new TechActivity.Quantities(quantitiesResources[1]), new
TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfClothingQuantitiesOne.add(theQuantities); // Add the quantities to the first array list
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean addToQuantitiesListTwo() { // Adds to the quantitites list.
        boolean quantitiesAdded = false;
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two),
                context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantitiesResources[0]), new TechActivity.Quantities(quantitiesResources[1]), new
TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfClothingQuantitiesTwo.add(theQuantities); // Add the quantities to the first array list
            quantitiesAdded = true;
        }

        return true;
    }
```

```
    private boolean clothingAddToBasketOne() {
        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(ClothingCategory.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products clothingFirstProduct = new Products(current_product_id, clothingFirstProductTxt.getText().toString(),
clothingFirstProductColourMenu.getSelectedItem().toString(), (int)
clothingFirstProductQuantityMenu.getSelectedItemId(), clothingFirstProductCostLbl.getText().toString(),
clothingFirstProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id, clothingFirstProduct);

        return true;
    }

    private boolean clothingAddToBasketTwo() { // Adds the second clothing product to the basket.

        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(ClothingCategory.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar
```

```
        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() { // Run the thread.
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products clothingSecondProduct = new Products(current_product_id++,
clothingSecondProductTxt.getText().toString(), clothingSecondProductColourMenu.getSelectedItem().toString(), (int)
clothingSecondProductQuantityMenu.getSelectedItemId(), clothingSecondProductCostLbl.getText().toString(),
clothingSecondProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, clothingSecondProduct); // Add the product instance to
the hash map

        return true;
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
        // Inflate the activities menu
        MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
        activityInflater.inflate(R.menu.homepagemenu, menu);

        MenuInflater menuInflater = getMenuInflater(); // Get the menu inflater to inflate the menu.
        menuInflater.inflate(R.menu.basket_action_button, menu); // Inflates the menu.

        View view = menu.findItem(R.id.cart_menu).getActionView(); // Get the action view for the cart menu

        cartIcon = view.findViewById(R.id.cart_icon);

        cartIcon.setOnClickListener(new View.OnClickListener() { // Add a listener to the cart icon when clicked
            @Override
            public void onClick(View v) {
                Intent basketIntent = new Intent(ClothingCategory.this, BasketActivity.class); // Create a basket intent
                basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
                startActivity(basketIntent); // Start the intent.
            }
        });
```

```
        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which menu item has been
selected.

        try {
            switch (item.getItemId()) {
                case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors category is chosen
                    Intent sportsCategory = new Intent(ClothingCategory.this, SportsAndOutdoorsActivity.class);
                    startActivity(sportsCategory); // Start that activity.

                    break;

                case R.id.techCategory: // If the tech activity is chosen
                    Intent techActivity = new Intent(ClothingCategory.this, TechActivity.class);
                    startActivity(techActivity); // Start that activity
                    break;

                case R.id.clothingCategory:
                    Intent clothingActivity = new Intent(ClothingCategory.this, ClothingCategory.class);
                    startActivity(clothingActivity);
                    break;

                case R.id.diyCategory: // If the diy category is chosen
                    Intent diyActivity = new Intent(ClothingCategory.this, DIYActivity.class);
                    startActivity(diyActivity); // Start the DIY Activity.
                    break;

                default:
                    super.onOptionsItemSelected(item);

            }
        }

        catch (ActivityNotFoundException exc) {
            Log.d(String.valueOf(R.string.error), exc.toString());
        }

        return true;
    }
}
```

Appendix – Application Layer Clothing Category Two Code

```
package com.example.weshopapplication.ApplicationLayer;
```

```
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.HashMap;

// Author of Application/Class: Sabin Constantin Lungu
// Purpose of Application Layer Class: To show customers the clothing category two activity.
// Date of Last Modification: 02/03/2020
// Any Errors? None

public class ClothingActivityTwo extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
    private int current_product_id = 1; // The current product id.
    private ImageView cartIcon; // The Basket Icon.

    private TextView clothingThirdProductTxt; // The text for the third Clothing Product
    private ImageView clothingThirdProductImg;

    private TextView clothingThirdProductColourLbl;

    private TextView clothingThirdProductCostLbl;
    private Spinner clothingThirdProductColourMenu; // The drop-down menu for the third clothing product colour

    private TextView clothingThirdProductSizeLbl; // The size for the clothing third product.
    private Spinner clothingThirdProductSizeMenu;
    private TextView clothingFourthProductCostLbl;

    private TextView clothingThirdProductQuantityLbl;
    private Spinner clothingThirdProductQuantityMenu;
    private Button clothingThirdAddToBasketBtn; // Button for adding the third product to the basket.
```

```
    private TextView clothingFourthProductTxt;
    private ImageView clothingFourthProductImageView;

    private TextView clothingFourthProductColourLbl;
    private Spinner clothingFourthProductColourMenu;

    private TextView clothingFourthProductSizeLbl;
    private Spinner clothingFourthProductSizeMenu;

    private TextView clothingFourthProductQuantityLbl;
    private Spinner clothingFourthProductQuantityMenu;
    private Button clothingFourthProductAddToBasketBtn;

    private double[] clothingThirdProductCosts = new double[]{0.00, 30.00, 60.00, 120.00, 240.00, 480.00}; // This
double array stores the costs for the third product
    private double[] clothingFourthProductCosts = new double[]{0.00, 40.00, 80.00, 160.00, 320.00, 640.00};

    private QuantitiesArrayAdapter quantitiesAdapter;
    private ColourArrayAdapter coloursAdapter;
    private SizeArrayAdapter sizesAdapter;

    private ArrayList<TechActivity.Colours> listOfClothingColoursOne = null; // An Array lsit of colours.
    private ArrayList<Size> listOfClothingSizesOne = null;
    private ArrayList<TechActivity.Quantities> listOfClothingQuantitiesOne = null; // An Array list of quantities.

    private ArrayList<TechActivity.Colours> listOfClothingColoursTwo = null;
    private ArrayList<Size> listOfClothingSizesTwo = null;
    private ArrayList<TechActivity.Quantities> listOfClothingQuantitiesTwo = null;

    private boolean coloursAdded = false; // Boolean variable that stores either true or false if the colours have been
added to the array list
    private boolean quantitiesAdded = false;
    private boolean sizesAdded = false; // Boolean variable that stores true or false if the sizes have been added.

    private HashMap<Integer, Products> listOfProductsToAddToBasket; // A hash map that stores the product id and
the product instance.

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_clothing_two);

        // Initialises the components
        this.clothingThirdProductTxt = findViewById(R.id.clothingThirdProductTxt); // Set up the clothing third product
text view.
        this.clothingThirdProductImg = findViewById(R.id.clothingThirdProductImg); // Set-up the image component

        this.clothingThirdProductColourLbl = findViewById(R.id.clothingThirdProductColourLbl);
        this.clothingThirdProductColourMenu = findViewById(R.id.clothingThirdProductColourMenu);
        this.clothingThirdProductCostLbl = findViewById(R.id.clothingThirdProductCost);

        this.clothingThirdProductQuantityLbl = findViewById(R.id.clothingThirdProductQuantityLbl);
```

```
        this.clothingThirdProductQuantityMenu = findViewById(R.id.clothingThirdProductQuantityMenu);

        this.clothingThirdProductSizeLbl = findViewById(R.id.clothingThirdProductSizeLbl);
        this.clothingThirdProductSizeMenu = findViewById(R.id.clothingThirdProductSizeMenu);
        this.clothingThirdAddToBasketBtn = findViewById(R.id.clothingThirdProductAddToBasketBtn);

        this.clothingFourthProductTxt = findViewById(R.id.clothingFourthProductTxt);
        this.clothingFourthProductCostLbl = findViewById(R.id.clothingFourthProductCost);
        this.clothingFourthProductImageView = findViewById(R.id.clothingFourthProductImg);

        this.clothingFourthProductColourLbl = findViewById(R.id.clothingFourthProductColourLbl);
        this.clothingFourthProductColourMenu = findViewById(R.id.clothingFourthProductColourMenu);

        this.clothingFourthProductSizeLbl = findViewById(R.id.clothingFourthProductSizeLbl);
        this.clothingFourthProductSizeMenu = findViewById(R.id.clothingFourthProductSizeMenu);

        this.clothingFourthProductQuantityLbl = findViewById(R.id.clothingFourthProductQuantityLbl);
        this.clothingFourthProductQuantityMenu = findViewById(R.id.clothingFourthProductQuantityMenu);

        this.clothingFourthProductAddToBasketBtn = findViewById(R.id.clothingFourthProductAddToBasketBtn);

        this.listOfClothingColoursOne = new ArrayList<>();
        this.listOfClothingSizesOne = new ArrayList<>();
        this.listOfClothingQuantitiesOne = new ArrayList<>();

        this.listOfClothingColoursTwo = new ArrayList<>();
        this.listOfClothingSizesTwo = new ArrayList<>();
        this.listOfClothingQuantitiesTwo = new ArrayList<>();

        this.listOfProductsToAddToBasket = new HashMap<>();

        addToColoursList(); // Routine that adds the colours to the array list
        addToSizesList(); // Adds the sizes to the array list

        addToQuantitiesListOne();
        addToQuantitiesListTwo();

        // Set-up Adapters.
        this.coloursAdapter = new ColourArrayAdapter(ClothingActivityTwo.this, listOfClothingColoursOne);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item); // Set the drop
down view for the colours.

        clothingThirdProductColourMenu.setAdapter(coloursAdapter);
        clothingThirdProductColourMenu.setOnItemSelectedListener(ClothingActivityTwo.this);

        this.sizesAdapter = new SizeArrayAdapter(ClothingActivityTwo.this, listOfClothingSizesOne);
        sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingThirdProductSizeMenu.setAdapter(sizesAdapter);
        clothingThirdProductSizeMenu.setOnItemSelectedListener(this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(ClothingActivityTwo.this, listOfClothingQuantitiesOne);
```

```
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingThirdProductQuantityMenu.setAdapter(quantitiesAdapter);
        clothingThirdProductQuantityMenu.setOnItemSelectedListener(this);

        this.coloursAdapter = new ColourArrayAdapter(ClothingActivityTwo.this, listOfClothingColoursTwo);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingFourthProductColourMenu.setAdapter(coloursAdapter);
        clothingFourthProductColourMenu.setOnItemSelectedListener(ClothingActivityTwo.this);

        this.sizesAdapter = new SizeArrayAdapter(ClothingActivityTwo.this, listOfClothingSizesTwo);
        sizesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingFourthProductSizeMenu.setAdapter(sizesAdapter);
        clothingFourthProductSizeMenu.setOnItemSelectedListener(this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(ClothingActivityTwo.this, listOfClothingQuantitiesTwo);
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        clothingFourthProductQuantityMenu.setAdapter(quantitiesAdapter);
        clothingFourthProductQuantityMenu.setOnItemSelectedListener(this); // Add an on click listener for the clothing
product.

        this.clothingThirdAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View thirdBtn) {
                if (thirdBtn.getId() == R.id.clothingThirdProductAddToBasketBtn) { // If the third button is clicked

                    if (clothingThirdProductColourMenu.getSelectedItemPosition() == 0 ||
clothingThirdProductSizeMenu.getSelectedItemPosition() == 0 ||
clothingThirdProductQuantityMenu.getSelectedItemPosition() == 0) { // If no colour, size and quantity is chosen
                        AlertDialog.Builder error = new AlertDialog.Builder(ClothingActivityTwo.this) // Create an alert dialogue
to display an error.
                                .setTitle(R.string.error) // Sets the title of the alert dialogue.
                                .setMessage(R.string.errorMsg) // Shows the message.

                                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int which) {
                                        if (dialog != null) {
                                            dialog.dismiss();
                                        }
                                    }
                                });

                        error.show(); // Show the error.
                        error.setCancelable(true);
                    } else {
                        clothingAddToBasketThree(); // Otherwise add the product to the basket.
                    }
                }
```

```
        }
    });

    this.clothingFourthProductAddToBasketBtn.setOnClickListener(new View.OnClickListener() { // Adds an on click
listener for the fourth product.
        @Override
        public void onClick(View fourthBtn) {
            if (fourthBtn.getId() == R.id.clothingFourthProductAddToBasketBtn) {

                if (clothingFourthProductColourMenu.getSelectedItemPosition() == 0 ||
clothingFourthProductSizeMenu.getSelectedItemPosition() == 0 ||
clothingFourthProductQuantityMenu.getSelectedItemPosition() == 0) {
                    AlertDialog.Builder error = new AlertDialog.Builder(ClothingActivityTwo.this)
                            .setTitle(R.string.error)
                            .setMessage(R.string.errorMsg)

                            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface dialog, int which) {
                                    if (dialog != null) {
                                        dialog.dismiss();
                                    }
                                }
                            });

                    error.show();
                    error.setCancelable(true);
                }

                else {
                    clothingAddToBasketFour();
                }
            }
        }
    });

}

@Override
public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
    // Inflate the activities menu
    MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
    activityInflater.inflate(R.menu.homepagemenu, menu);

    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.basket_action_button, menu);

    View view = menu.findItem(R.id.cart_menu).getActionView();

    cartIcon = view.findViewById(R.id.cart_icon);

    cartIcon.setOnClickListener(new View.OnClickListener() { // Add an action listener to the cart icon
```

```
        @Override
        public void onClick(View v) {

            Intent basketIntent = new Intent(ClothingActivityTwo.this, BasketActivity.class); // Create a basket intent
            basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
            startActivity(basketIntent); // Start the intent
        }
    });

    return true;
}

public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which item has been selected
from the menu

    try {
        switch (item.getItemId()) {
            case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors activity is chosen
                Intent sportsCategory = new Intent(ClothingActivityTwo.this, SportsAndOutdoorsActivity.class);
                startActivity(sportsCategory); // Start that activity

                break;

            case R.id.techCategory:
                Intent techActivity = new Intent(ClothingActivityTwo.this, TechActivity.class);
                startActivity(techActivity);
                break;

            case R.id.clothingCategory:
                Intent clothingActivity = new Intent(ClothingActivityTwo.this, ClothingCategory.class);
                startActivity(clothingActivity);
                break;

            case R.id.diyCategory:
                Intent diyActivity = new Intent(ClothingActivityTwo.this, DIYActivity.class);
                startActivity(diyActivity);
                break;

            default:
                super.onOptionsItemSelected(item);

        }
    }

    catch (ActivityNotFoundException exc) {
        Log.d(String.valueOf(R.string.error), exc.toString());
    }

    return true;
}
```

```
    private boolean addToColoursList() { // Routine that adds to the colours array list

        Context context = getApplicationContext();

        String[] clothingResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.salmonPink), context.getString(R.string.skyBlue),
                context.getString(R.string.rubyRed), context.getString(R.string.cityGray)}; // Creates a string array of
resources to add

        TechActivity.Colours[] colours = new TechActivity.Colours[]{new TechActivity.Colours(0, clothingResources[0]),
new TechActivity.Colours(1, clothingResources[1]), new TechActivity.Colours(2, clothingResources[2]),
                new TechActivity.Colours(3, clothingResources[3]), new TechActivity.Colours(4, clothingResources[4])};

        for (TechActivity.Colours theColours : colours) { // For every colour in the colours array
            listOfClothingColoursOne.add(theColours); // Add it to the array list
            listOfClothingColoursTwo.add(theColours);

            coloursAdded = true; // Colours added is now true.
        }

        return true;
    }

    private boolean addToSizesList() { // Adds to the sizes list.

        boolean sizes_added = false;
        Context context = getApplicationContext();

        String[] clothingSizes = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.smallSize), context.getString(R.string.mediumSize)
                , context.getString(R.string.largeSize), context.getString(R.string.extraLargeSize)};

        Size[] sizes = new Size[]{new Size(0, clothingSizes[0]), new Size(1, clothingSizes[1]), new Size(2,
clothingSizes[2]), new Size(3, clothingSizes[3]), new Size(4, clothingSizes[4])};

        for (Size theSizes : sizes) {
            listOfClothingSizesOne.add(theSizes);
            listOfClothingSizesTwo.add(theSizes);
            sizes_added = true;
        }

        return true;
    }

    private boolean addToQuantitiesListOne() { // Adds to the quantities array list

        boolean quantitiesAdded = false;
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two),
                context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};
```

```
        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantitiesResources[0]), new TechActivity.Quantities(quantitiesResources[1]), new
TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfClothingQuantitiesOne.add(theQuantities); // Add the quantities to the first array list
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean addToQuantitiesListTwo() {

        boolean quantitiesAdded = false;
        Context context = getApplicationContext();

        String[] quantitiesResources = new String[]{context.getString(R.string.zero), context.getString(R.string.one),
context.getString(R.string.two),
                context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantitiesResources[0]), new TechActivity.Quantities(quantitiesResources[1]), new
TechActivity.Quantities(quantitiesResources[2]),
                new TechActivity.Quantities(quantitiesResources[3]), new TechActivity.Quantities(quantitiesResources[4]),
new TechActivity.Quantities(quantitiesResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            listOfClothingQuantitiesTwo.add(theQuantities); // Add the quantities to the first array list
            quantitiesAdded = true;
        }

        return true;
    }

    private boolean clothingAddToBasketThree() { // Adds the third clothing product to the basket
        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(ClothingActivityTwo.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread
```

```
        @Override
        public void run() {
            try {

                Thread.sleep(1900); // Sleep for 1.9 seconds.
            } catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();
        }
    }).start(); // Starts the thread

    dialog.show();

    // Create an instance for the first product and adds it to the hash map.
    Products clothingThirdProduct = new Products(current_product_id, clothingThirdProductTxt.getText().toString(),
clothingThirdProductColourMenu.getSelectedItem().toString(), (int)
clothingThirdProductQuantityMenu.getSelectedItemId(), clothingThirdProductCostLbl.getText().toString(),
clothingThirdProductSizeMenu.getSelectedItem().toString());
    listOfProductsToAddToBasket.put(current_product_id, clothingThirdProduct);

    return true;
}

private boolean clothingAddToBasketFour() {

    Context context = getApplicationContext();
    String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

    final ProgressDialog dialog = new ProgressDialog(ClothingActivityTwo.this); // Spinning progress dialog
    dialog.setTitle(temp[0]); // Set the title of the dialog
    dialog.setMessage(temp[1]);

    dialog.setCancelable(false);

    dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

    new Thread(new Runnable() { // Create a new thread

        @Override
        public void run() {
            try {

                Thread.sleep(1900); // Sleep for 1.9 seconds.
            }

            catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();
```

```
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products clothingThirdProduct = new Products(current_product_id++,
clothingFourthProductTxt.getText().toString(), clothingFourthProductColourMenu.getSelectedItem().toString(), (int)
clothingFourthProductQuantityMenu.getSelectedItemId(), clothingFourthProductCostLbl.getText().toString(),
clothingThirdProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, clothingThirdProduct);

        return true;
    }

    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) { // Routine that will determine
which item has been selected in the spinner.
        boolean valueAppended = false;

        int[] indexes = new int[]{0, 1, 2, 3, 4}; // An array of indexes

        Context context = getApplicationContext();
        String[] productResources = new String[]{context.getString(R.string.productCost)};

        if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[0]))) { // If the first index is
chosen in the drop-down list of quantities.
            clothingThirdProductCostLbl.setText(null); // Empty the text by default.
            clothingThirdProductCostLbl.append(productResources[0] + clothingThirdProductCosts[0]); // Append the
product cost. Product Cost £: 0.00
            valueAppended = true; // Value has been appended.

        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[1]))) {
            clothingThirdProductCostLbl.setText(null);
            clothingThirdProductCostLbl.append(productResources[0] + clothingThirdProductCosts[1]);
            valueAppended = true; // Value is appended

        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[2]))) {
            clothingThirdProductCostLbl.setText(null);
            clothingThirdProductCostLbl.append(productResources[0] + clothingThirdProductCosts[2]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[3]))) {
            clothingThirdProductCostLbl.setText(null);
            clothingThirdProductCostLbl.append(productResources[0] + clothingThirdProductCosts[3]);
            valueAppended = true;
        }
```

```
        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesOne.get(indexes[4]))) {
            clothingThirdProductCostLbl.setText(null);
            clothingThirdProductCostLbl.append(productResources[0] + clothingThirdProductCosts[4]);
            valueAppended = true;
        }

        if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[0]))) {
            clothingFourthProductCostLbl.setText(null);
            clothingFourthProductCostLbl.append(productResources[0] + clothingFourthProductCosts[0]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[1]))) {
            clothingFourthProductCostLbl.setText(null);
            clothingFourthProductCostLbl.append(productResources[0] + clothingFourthProductCosts[1]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[2]))) {
            clothingFourthProductCostLbl.setText(null);
            clothingFourthProductCostLbl.append(productResources[0] + clothingFourthProductCosts[2]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[3]))) {
            clothingFourthProductCostLbl.setText(null);
            clothingFourthProductCostLbl.append(productResources[0] + clothingFourthProductCosts[3]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(listOfClothingQuantitiesTwo.get(indexes[4]))) {
            clothingFourthProductCostLbl.setText(null);
            clothingFourthProductCostLbl.append(productResources[0] + clothingFourthProductCosts[4]);
            valueAppended = true;
        }
    }

    @Override
    public void onNothingSelected(AdapterView<?> parent) { // Routine that determines if nothing is selected.

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }
}
```

Appendix – Application Layer DIY Category Code One

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.HashMap;

// Author of Application: Sabin Constantin Lungu.
// Purpose of Application / Class: Contains the Java code for the DIY activity that corresponds to the DIY XML code.
// Date of Last Modification: 03/02/2020
// Any errors? None

public class DIYActivity extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
    private int current_product_id = 1; // The current product id to add to basket (will be incremented)
    private TextView diyFirstProductTxt; // The first product text

    private ImageView diyFirstProductImg; // Image of the first DIY product
    private TextView diyFirstProductCost;

    private TextView diyFirstProductColourLbl;
    private Spinner diyFirstProductColourMenu;

    private TextView diyFirstProductSizeLbl;
    private Spinner diyFirstProductSizeMenu;

    private TextView diyFirstProductQuantityLbl;
```

```
    private Spinner diyFirstProductQuantityMenu;
    private Button diyFirstProductToAddToBasketBtn;

    private TextView diySecondProductTxt;
    private ImageView diySecondProductImg;

    private TextView diySecondProductCost;

    private TextView diySecondProductColourLbl;
    private Spinner diySecondProductColourMenu;

    private TextView diySecondProductSizeLbl;
    private Spinner diySecondProductSizeMenu;

    private TextView diySecondProductQuantityLbl;
    private Spinner diySecondProductQuantityMenu;

    private Button diySecondProductAddToBasketBtn;

    private double[] diyFirstProductCosts = new double[]{0.00, 40.00, 80.00, 160.00, 320.00, 640.00}; // A double array
of product costs for the first DIY product
    private double[] diySecondProductCosts = new double[]{0.00, 20.00, 40.00, 80.00, 160.00, 320.00};

    private boolean coloursAdded = false;
    private boolean sizesAdded = false;
    private boolean quantitiesAdded = false;

    // Adapters for the objects to add to the list
    private QuantitiesArrayAdapter quantitiesAdapter;
    private SizeArrayAdapter sizeArrayAdapter;
    private ColourArrayAdapter coloursAdapter;

    private ArrayList<TechActivity.Colours> diyListOfColoursOne = null; // An array list of colours
    private ArrayList<Size> diyListOfSizesOne = null;
    private ArrayList<TechActivity.Quantities> diyListOfQuantitiesOne = null; // An Array list of quantities for the first diy
product

    // Creates the array lists for the second DIY product.
    private ArrayList<TechActivity.Colours> diyListOfColoursTwo = null;
    private ArrayList<Size> diyListOfSizesTwo = null;
    private ArrayList<TechActivity.Quantities> diyListOfQuantitiesTwo = null;

    private ImageView cartIcon;
    private HashMap<Integer, Products> listOfProductsToAddToBasket = new HashMap<Integer, Products>(); //
Creates a new hash map of products with an associated ID
    private Button nextPageBtn;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_diy);
```

```
// Initialise components
this.diyFirstProductTxt = findViewById(R.id.diyFirstProductTxt);
this.diyFirstProductImg = findViewById(R.id.diyFirstProductImg);

this.diyFirstProductCost = findViewById(R.id.diyFirstProductCostTxt);
this.diyFirstProductColourLbl = findViewById(R.id.diyFirstProductColourLbl);
this.diyFirstProductColourMenu = findViewById(R.id.diyFirstProductColourMenu);

this.diyFirstProductSizeLbl = findViewById(R.id.diyFirstProductSizeLbl);
this.diyFirstProductSizeMenu = findViewById(R.id.diyFirstProductSizeMenu);

this.diyFirstProductQuantityLbl = findViewById(R.id.diyFirstProductQuantityLbl);
this.diyFirstProductQuantityMenu = findViewById(R.id.diyFirstProductQuantityMenu);
this.diyFirstProductToAddToBasketBtn = findViewById(R.id.diyFirstProductAddToBasketBtn);

this.diySecondProductTxt = findViewById(R.id.diySecondProductTxt);
this.diySecondProductImg = findViewById(R.id.diySecondProductImg);

this.diySecondProductCost = findViewById(R.id.diySecondProductCostLbl);
this.diySecondProductColourLbl = findViewById(R.id.diySecondProductColourLbl);
this.diySecondProductColourMenu = findViewById(R.id.diySecondProductColourMenu);
this.diySecondProductQuantityMenu = findViewById(R.id.diySecondProductQuantityMenu);

this.diySecondProductSizeLbl = findViewById(R.id.diySecondProductSizeLbl);
this.diySecondProductSizeMenu = findViewById(R.id.diySecondProductSizeMenu);

this.diySecondProductAddToBasketBtn = findViewById(R.id.diySecondProductAddToBasketBtn);

this.diyListOfColoursOne = new ArrayList<>();
this.diyListOfColoursTwo = new ArrayList<>();

this.diyListOfSizesOne = new ArrayList<>();
this.diyListOfSizesTwo = new ArrayList<>();

this.diyListOfQuantitiesOne = new ArrayList<>();
this.diyListOfQuantitiesTwo = new ArrayList<>();

addToDIYColourList();
addToDIYSizesList();

addToDIYQuantitiesListOne();
addToDIYQuantitiesListTwo();

// Set-up adapters for the firsts Array List

this.coloursAdapter = new ColourArrayAdapter(DIYActivity.this, diyListOfColoursOne);
coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

diyFirstProductColourMenu.setAdapter(coloursAdapter);
diyFirstProductColourMenu.setOnItemSelectedListener(DIYActivity.this);

this.quantitiesAdapter = new QuantitiesArrayAdapter(DIYActivity.this, diyListOfQuantitiesOne); // Creates the
```

```
quantities adapter for the first list of quantities.
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyFirstProductQuantityMenu.setAdapter(quantitiesAdapter);
        diyFirstProductQuantityMenu.setOnItemSelectedListener(DIYActivity.this);

        this.sizeArrayAdapter = new SizeArrayAdapter(DIYActivity.this, diyListOfSizesOne);
        sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
        diyFirstProductSizeMenu.setAdapter(sizeArrayAdapter);
        diyFirstProductSizeMenu.setOnItemSelectedListener(DIYActivity.this);

        // Set-up adapters for the second ArrayList

        this.coloursAdapter = new ColourArrayAdapter(DIYActivity.this, diyListOfColoursTwo);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diySecondProductColourMenu.setAdapter(coloursAdapter);
        diySecondProductColourMenu.setOnItemSelectedListener(this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(DIYActivity.this, diyListOfQuantitiesTwo);
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diySecondProductQuantityMenu.setAdapter(quantitiesAdapter);
        diySecondProductQuantityMenu.setOnItemSelectedListener(this);

        this.sizeArrayAdapter = new SizeArrayAdapter(DIYActivity.this, diyListOfSizesTwo);
        sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diySecondProductSizeMenu.setAdapter(sizeArrayAdapter);
        diySecondProductSizeMenu.setOnItemSelectedListener(this);

        this.diyFirstProductToAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                if (diyFirstProductToAddToBasketBtn.getId() == R.id.diyFirstProductAddToBasketBtn) {

                    if (diyFirstProductColourMenu.getSelectedItemPosition() == 0 ||
diyFirstProductSizeMenu.getSelectedItemPosition() == 0 || diyFirstProductQuantityMenu.getSelectedItemPosition()
== 0) {
                        AlertDialog.Builder error = new AlertDialog.Builder(DIYActivity.this)
                                .setTitle(R.string.error)
                                .setMessage(R.string.errorMsg)
                                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int which) {
                                        if (dialog != null) {
                                            dialog.dismiss();
                                        }
                                    }
                                });

                        error.show();
```

```
                error.setCancelable(true);
            } else {
                diyAddToBasketOne();
            }
        }
    }
});

this.diySecondProductAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View secondButton) {
        if (secondButton.getId() == R.id.diySecondProductAddToBasketBtn) {

            if (diySecondProductColourMenu.getSelectedItemPosition() == 0 ||
diySecondProductSizeMenu.getSelectedItemPosition() == 0 ||
diySecondProductQuantityMenu.getSelectedItemPosition() == 0) {
                AlertDialog.Builder error = new AlertDialog.Builder(DIYActivity.this)

                        .setTitle(R.string.error)
                        .setMessage(R.string.errorMsg)
                        .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {

                                if (dialog != null) {

                                    dialog.dismiss();
                                }
                            }
                        });

                error.show();
                error.setCancelable(true);
            } else {
                diyAddToBasketTwo();
            }
        }
    }
});

this.nextPageBtn = findViewById(R.id.diyNextPageBtn);

this.nextPageBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {

            if (v.getId() == R.id.diyNextPageBtn) {
                Intent nextDiyIntent = new Intent(DIYActivity.this, DIYActivityTwo.class);
                startActivity(nextDiyIntent);
            }
        } catch (ActivityNotFoundException exc) {
```

```
                Log.d(String.valueOf(R.string.error), exc.toString());
            }
        }
    });
}

private boolean addToDIYColourList() {
    Context context = getApplicationContext();
    String[] diyColoursResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.gallantGray), context.getString(R.string.darkBlack),
context.getString(R.string.strawberryRed), context.getString(R.string.gardenGreen)};

    TechActivity.Colours[] colours = new TechActivity.Colours[]{new TechActivity.Colours(0,
diyColoursResources[0]), new TechActivity.Colours(1, diyColoursResources[1]), new TechActivity.Colours(2,
diyColoursResources[2]), new TechActivity.Colours(3, diyColoursResources[3]),
            new TechActivity.Colours(4, diyColoursResources[4])};

    for (TechActivity.Colours theColours : colours) {
        diyListOfColoursOne.add(theColours);
        diyListOfColoursTwo.add(theColours);
        coloursAdded = true;
    }

    return true;
}

private boolean addToDIYSizesList() {
    Context context = getApplicationContext();
    String[] diySizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.smallSize), context.getString(R.string.mediumSize), context.getString(R.string.largeSize),
context.getString(R.string.extraLargeSize)};

    Size[] sizes = new Size[]{new Size(0, diySizesResources[0]), new Size(1, diySizesResources[1]), new Size(2,
diySizesResources[2]), new Size(3, diySizesResources[3]), new Size(4, diySizesResources[4])};

    for (Size theSizes : sizes) {
        diyListOfSizesOne.add(theSizes);
        diyListOfSizesTwo.add(theSizes);

        sizesAdded = true;
    }

    return true;
}

private boolean addToDIYQuantitiesListOne() {

    Context context = getApplicationContext();
    String[] quantityResources = new String[]{context.getString(R.string.quantitiesPrompt),
context.getString(R.string.zero), context.getString(R.string.one), context.getString(R.string.two),
context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};
```

```
        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantityResources[0]), new TechActivity.Quantities(quantityResources[1]), new
TechActivity.Quantities(quantityResources[2]), new TechActivity.Quantities(quantityResources[3]), new
TechActivity.Quantities(quantityResources[4]),
                new TechActivity.Quantities(quantityResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            diyListOfQuantitiesOne.add(theQuantities);

            sizesAdded = true;
        }

        return true;
    }

    private boolean addToDIYQuantitiesListTwo() {
        Context context = getApplicationContext();
        String[] quantityResources = new String[]{context.getString(R.string.quantitiesPrompt),
context.getString(R.string.zero), context.getString(R.string.one), context.getString(R.string.two),
context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantityResources[0]), new TechActivity.Quantities(quantityResources[1]), new
TechActivity.Quantities(quantityResources[2]), new TechActivity.Quantities(quantityResources[3]), new
TechActivity.Quantities(quantityResources[4]),
                new TechActivity.Quantities(quantityResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            diyListOfQuantitiesTwo.add(theQuantities);
            sizesAdded = true;
        }


        return true;
    }


    private boolean diyAddToBasketOne() { // Routine that adds the first DIY product to the basket list view.
        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(DIYActivity.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
```

```
            public void run() {
                try {
                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                }

                catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread

        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products diyFirstProduct = new Products(current_product_id, diyFirstProductTxt.getText().toString(),
diyFirstProductColourMenu.getSelectedItem().toString(), (int) diyFirstProductQuantityMenu.getSelectedItemId(),
diyFirstProductCost.getText().toString(), diyFirstProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id, diyFirstProduct);

        return true;
    }

    private boolean diyAddToBasketTwo() {

        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(DIYActivity.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
                try {

                    Thread.sleep(1900); // Sleep for 1.9 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss();
            }
        }).start(); // Starts the thread
```

```
        dialog.show();

        // Create an instance for the first product and adds it to the hash map.
        Products diySecondProduct = new Products(current_product_id++, diySecondProductTxt.getText().toString(),
diySecondProductColourMenu.getSelectedItem().toString(), (int)
diySecondProductQuantityMenu.getSelectedItemId(), diySecondProductCost.getText().toString(),
diySecondProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, diySecondProduct);

        return true;
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
        // Inflate the activities menu
        MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
        activityInflater.inflate(R.menu.homepagemenu, menu);

        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.basket_action_button, menu);

        View view = menu.findItem(R.id.cart_menu).getActionView();

        cartIcon = view.findViewById(R.id.cart_icon);

        cartIcon.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent basketIntent = new Intent(DIYActivity.this, BasketActivity.class); // Create a basket intent
                basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
                startActivity(basketIntent); // Start the intent
            }
        });

        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which item has been selected.

        try {
            switch (item.getItemId()) {

                case R.id.mainActivity:
                    Intent mainActivity = new Intent(DIYActivity.this, MainActivity.class);
                    startActivity(mainActivity);
                    break;

                case R.id.sportsAndOutdoorsCategory:
                    Intent sportsCategory = new Intent(DIYActivity.this, SportsAndOutdoorsActivity.class);
                    startActivity(sportsCategory);
```

```
            break;

        case R.id.techCategory:
            Intent techActivity = new Intent(DIYActivity.this, TechActivity.class);
            startActivity(techActivity);
            break;

        case R.id.clothingCategory:
            Intent clothingActivity = new Intent(DIYActivity.this, ClothingCategory.class);
            startActivity(clothingActivity);
            break;

        case R.id.diyCategory:
            Intent diyActivity = new Intent(DIYActivity.this, DIYActivity.class);
            startActivity(diyActivity);
            break;

        default:
            super.onOptionsItemSelected(item);

      }

    }

    catch (ActivityNotFoundException exc) {
        Log.d(String.valueOf(R.string.error), exc.toString());
    }

    return true;
}

@Override
public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
    boolean valueAppended = false;

    int[] indexes = new int[]{0, 1, 2, 3, 4};

    Context context = getApplicationContext();
    String[] productResources = new String[]{context.getString(R.string.productCost)};

    if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[0]))) {
        diyFirstProductCost.setText(null);
        diyFirstProductCost.append(productResources[0] + diyFirstProductCosts[0]);
        valueAppended = true;

    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[1]))) {
        diyFirstProductCost.setText(null);
        diyFirstProductCost.append(productResources[0] + diyFirstProductCosts[1]);
        valueAppended = true; // Value is appended
```

```
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[2]))) {
        diyFirstProductCost.setText(null);
        diyFirstProductCost.append(productResources[0] + diyFirstProductCosts[2]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[3]))) {
        diyFirstProductCost.setText(null);
        diyFirstProductCost.append(productResources[0] + diyFirstProductCosts[3]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[4]))) {
        diyFirstProductCost.setText(null);
        diyFirstProductCost.append(productResources[0] + diyFirstProductCosts[4]);
        valueAppended = true;
    }


    if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[0]))) {
        diySecondProductCost.setText(null);
        diySecondProductCost.append(productResources[0] + diySecondProductCosts[0]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[1]))) {
        diySecondProductCost.setText(null);
        diySecondProductCost.append(productResources[0] + diySecondProductCosts[1]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[2]))) {
        diySecondProductCost.setText(null);
        diySecondProductCost.append(productResources[0] + diySecondProductCosts[2]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[3]))) {
        diySecondProductCost.setText(null);
        diySecondProductCost.append(productResources[0] + diySecondProductCosts[3]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[4]))) {
        diySecondProductCost.setText(null);
        diySecondProductCost.append(productResources[0] + diySecondProductCosts[4]);
        valueAppended = true;
    }
}
```

```
    @Override
    public void onNothingSelected(AdapterView<?> parent) {

    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }
}
```

Appendix – Application Layer DIY Activity Two Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.ColourArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.QuantitiesArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Size;
import com.example.weshopapplication.BusinessObjects.SizeArrayAdapter;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.HashMap;

// Author of Application: Sabin Constantin Lungu
// Purpose of Application / Class: Contains the Java code for the DIY activity 2.
// Date of Last Modification : 04/03/2020
// Any Errors? None

public class DIYActivityTwo extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
```

```
    private int current_product_id = 1; // The current product id to add to basket (will be incremented)
    private TextView diyThirdProductTxt; // The first product text

    private ImageView diyThirdProductImg; // Image of the first DIY product
    private TextView diyThirdProductCost;

    private TextView diyThirdProductColourLbl;
    private Spinner diyThirdProductColourMenu;

    private TextView diyThirdProductSizeLbl;
    private Spinner diyThirdProductSizeMenu;

    private TextView diyThirdProductQuantityLbl;
    private Spinner diyThirdProductQuantityMenu;
    private Button diyThirdProductToAddToBasketBtn;

    private TextView diyFourthProductTxt;
    private ImageView diyFourthProductImg;

    private TextView diyFourthProductCost;

    private TextView diyFourthProductColourLbl;
    private Spinner diyFourthProductColourMenu;

    private TextView diyFourthProductSizeLbl;
    private Spinner diyFourthProductSizeMenu;

    private TextView diyFourthProductQuantityLbl;
    private Spinner diyFourthProductQuantityMenu;

    private Button diyFourthProductAddToBasketBtn;

    private double[] diyThirdProductCosts = new double[]{0.00, 50.00, 100.00, 200.00, 400.00, 800.00}; // A double
array of product costs for the first DIY product
    private double[] diyFourthProductCosts = new double[]{0.00, 15.00, 30.00, 45.00, 60.00, 75.00};

    private boolean coloursAdded = false;
    private boolean sizesAdded = false;
    private boolean quantitiesAdded = false;

    // Adapters for the objects to add to the list
    private QuantitiesArrayAdapter quantitiesAdapter;
    private SizeArrayAdapter sizeArrayAdapter;
    private ColourArrayAdapter coloursAdapter;

    private ArrayList<TechActivity.Colours> diyListOfColoursOne = null; // An array list of colours
    private ArrayList<Size> diyListOfSizesOne = null;
    private ArrayList<TechActivity.Quantities> diyListOfQuantitiesOne = null; // An Array list of quantities for the first diy
product

    // Creates the array lists for the second DIY product.
    private ArrayList<TechActivity.Colours> diyListOfColoursTwo = null;
```

```
    private ArrayList<Size> diyListOfSizesTwo = null;
    private ArrayList<TechActivity.Quantities> diyListOfQuantitiesTwo = null;

    private ImageView cartIcon;
    private HashMap<Integer, Products> listOfProductsToAddToBasket = new HashMap<Integer, Products>(); //
Creates a new hash map of products with an associated ID

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_d_i_y_two);

        this.diyThirdProductTxt = findViewById(R.id.diyThirdProductTxt);
        this.diyThirdProductImg = findViewById(R.id.diyThirdProductImg);

        this.diyThirdProductCost = findViewById(R.id.diyThirdProductCostTxt);
        this.diyThirdProductColourLbl = findViewById(R.id.diyThirdColourLbl);
        this.diyThirdProductColourMenu = findViewById(R.id.diyThirdColourMenu);

        this.diyThirdProductSizeLbl = findViewById(R.id.diyThirdSizeLbl);
        this.diyThirdProductSizeMenu = findViewById(R.id.diyThirdSizeMenu);

        this.diyThirdProductQuantityLbl = findViewById(R.id.diyThirdQuantityLbl);
        this.diyThirdProductQuantityMenu = findViewById(R.id.diyThirdQuantityMenu);
        this.diyThirdProductToAddToBasketBtn = findViewById(R.id.diyThirdAddToBasketBtn);

        this.diyFourthProductTxt = findViewById(R.id.diyFourthProductTxt);
        this.diyFourthProductImg = findViewById(R.id.diyFourthProductImg);

        this.diyFourthProductCost = findViewById(R.id.diyFourthProductCostTxt);
        this.diyFourthProductColourLbl = findViewById(R.id.diyFourthProductColourLbl);

        this.diyFourthProductColourMenu = findViewById(R.id.diyFourthColourMenu);
        this.diyFourthProductQuantityMenu = findViewById(R.id.diyFourthProductQuantityMenu);

        this.diyFourthProductSizeLbl = findViewById(R.id.fourthProductSizeLbl);
        this.diyFourthProductSizeMenu = findViewById(R.id.diyFourthProductSizeMenu);

        this.diyFourthProductAddToBasketBtn = findViewById(R.id.diyFourthProductAddToBasketBtn);

        this.diyListOfColoursOne = new ArrayList<>();
        this.diyListOfColoursTwo = new ArrayList<>();

        this.diyListOfSizesOne = new ArrayList<>();
        this.diyListOfSizesTwo = new ArrayList<>();

        this.diyListOfQuantitiesOne = new ArrayList<>();
        this.diyListOfQuantitiesTwo = new ArrayList<>();

        addToDIYColourListThree();
        addToDIYSizesListThree();
```

```
        addToDIYQuantitiesListThree();
        addToDIYQuantitiesListFour();

        // Set-up adapters for the firsts Array List

        this.coloursAdapter = new ColourArrayAdapter(DIYActivityTwo.this, diyListOfColoursOne);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyThirdProductColourMenu.setAdapter(coloursAdapter);
        diyThirdProductColourMenu.setOnItemSelectedListener(DIYActivityTwo.this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(DIYActivityTwo.this, diyListOfQuantitiesOne); // Creates
the quantities adapter for the first list of quantities.
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyThirdProductQuantityMenu.setAdapter(quantitiesAdapter);
        diyThirdProductQuantityMenu.setOnItemSelectedListener(DIYActivityTwo.this);

        this.sizeArrayAdapter = new SizeArrayAdapter(DIYActivityTwo.this, diyListOfSizesOne);
        sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyThirdProductSizeMenu.setAdapter(sizeArrayAdapter);
        diyThirdProductSizeMenu.setOnItemSelectedListener(DIYActivityTwo.this);

        // Set-up adapters for the second ArrayList

        this.coloursAdapter = new ColourArrayAdapter(DIYActivityTwo.this, diyListOfColoursTwo);
        coloursAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyFourthProductColourMenu.setAdapter(coloursAdapter);
        diyFourthProductColourMenu.setOnItemSelectedListener(this);

        this.quantitiesAdapter = new QuantitiesArrayAdapter(DIYActivityTwo.this, diyListOfQuantitiesTwo);
        quantitiesAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyFourthProductQuantityMenu.setAdapter(quantitiesAdapter);
        diyFourthProductQuantityMenu.setOnItemSelectedListener(this);

        this.sizeArrayAdapter = new SizeArrayAdapter(DIYActivityTwo.this, diyListOfSizesTwo);
        sizeArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        diyFourthProductSizeMenu.setAdapter(sizeArrayAdapter);
        diyFourthProductSizeMenu.setOnItemSelectedListener(this);

        this.diyThirdProductToAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View thirdBtn) {
                if (diyThirdProductToAddToBasketBtn.getId() == R.id.diyThirdAddToBasketBtn) {

                    if (diyThirdProductColourMenu.getSelectedItemPosition() == 0 ||
diyThirdProductSizeMenu.getSelectedItemPosition() == 0 || diyThirdProductQuantityMenu.getSelectedItemPosition()
== 0) {
```

```
                AlertDialog.Builder error = new AlertDialog.Builder(DIYActivityTwo.this)
                        .setTitle(R.string.error)
                        .setMessage(R.string.errorMsg)
                        .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {

                                if (dialog != null) {
                                    dialog.dismiss();
                                }
                            }
                        });

                error.show();
                error.setCancelable(true);
            }

            else {
                diyAddToBasketThree();
            }
        }
    });

    this.diyFourthProductAddToBasketBtn.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View fourthBtn) {
            if (diyFourthProductAddToBasketBtn.getId() == R.id.diyFourthProductAddToBasketBtn) {

                if (diyFourthProductColourMenu.getSelectedItemPosition() == 0 ||
diyFourthProductSizeMenu.getSelectedItemPosition() == 0 ||
diyFourthProductQuantityMenu.getSelectedItemPosition() == 0) {
                    AlertDialog.Builder error = new AlertDialog.Builder(DIYActivityTwo.this)
                            .setTitle(R.string.error)
                            .setMessage(R.string.errorMsg)
                            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface dialog, int which) {
                                    if (dialog != null) {
                                        dialog.dismiss();
                                    }
                                }
                            });

                    error.show();
                    error.setCancelable(true);
                } else {
                    diyAddToBasketFour();
                }
            }
        }
    });
```

```
    }

    private boolean addToDIYColourListThree() {
        Context context = getApplicationContext();
        String[] diyColoursResources = new String[]{context.getString(R.string.colourPrompt),
context.getString(R.string.gallantGray), context.getString(R.string.darkBlack),
context.getString(R.string.strawberryRed), context.getString(R.string.gardenGreen)};

        TechActivity.Colours[] colours = new TechActivity.Colours[]{new TechActivity.Colours(0,
diyColoursResources[0]), new TechActivity.Colours(1, diyColoursResources[1]), new TechActivity.Colours(2,
diyColoursResources[2]), new TechActivity.Colours(3, diyColoursResources[3]),
                new TechActivity.Colours(4, diyColoursResources[4])};

        for (TechActivity.Colours theColours : colours) {
            diyListOfColoursOne.add(theColours);
            diyListOfColoursTwo.add(theColours);
            coloursAdded = true;
        }

        return true;
    }

    private boolean addToDIYSizesListThree() {
        Context context = getApplicationContext();
        String[] diySizesResources = new String[]{context.getString(R.string.sizePrompt),
context.getString(R.string.smallSize), context.getString(R.string.mediumSize), context.getString(R.string.largeSize),
context.getString(R.string.extraLargeSize)};

        Size[] sizes = new Size[]{new Size(0, diySizesResources[0]), new Size(1, diySizesResources[1]), new Size(2,
diySizesResources[2]), new Size(3, diySizesResources[3]), new Size(4, diySizesResources[4])};

        for (Size theSizes : sizes) {
            diyListOfSizesOne.add(theSizes);
            diyListOfSizesTwo.add(theSizes);

            sizesAdded = true;
        }

        return true;
    }

    private boolean addToDIYQuantitiesListThree() {

        Context context = getApplicationContext();
        String[] quantityResources = new String[]{context.getString(R.string.quantitiesPrompt),
context.getString(R.string.zero), context.getString(R.string.one), context.getString(R.string.two),
context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantityResources[0]), new TechActivity.Quantities(quantityResources[1]), new
TechActivity.Quantities(quantityResources[2]), new TechActivity.Quantities(quantityResources[3]), new
```

```
TechActivity.Quantities(quantityResources[4]),
                new TechActivity.Quantities(quantityResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            diyListOfQuantitiesOne.add(theQuantities);

            sizesAdded = true;
        }


        return true;
    }

    private boolean addToDIYQuantitiesListFour() {
        Context context = getApplicationContext();
        String[] quantityResources = new String[]{context.getString(R.string.quantitiesPrompt),
context.getString(R.string.zero), context.getString(R.string.one), context.getString(R.string.two),
context.getString(R.string.three), context.getString(R.string.four), context.getString(R.string.five)};

        TechActivity.Quantities[] quantities = new TechActivity.Quantities[]{new
TechActivity.Quantities(quantityResources[0]), new TechActivity.Quantities(quantityResources[1]), new
TechActivity.Quantities(quantityResources[2]), new TechActivity.Quantities(quantityResources[3]), new
TechActivity.Quantities(quantityResources[4]),
                new TechActivity.Quantities(quantityResources[5])};

        for (TechActivity.Quantities theQuantities : quantities) {
            diyListOfQuantitiesTwo.add(theQuantities);

            sizesAdded = true;
        }


        return true;
    }

    private boolean diyAddToBasketThree() {

        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

        final ProgressDialog dialog = new ProgressDialog(DIYActivityTwo.this); // Spinning progress dialog
        dialog.setTitle(temp[0]); // Set the title of the dialog
        dialog.setMessage(temp[1]);

        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

        new Thread(new Runnable() { // Create a new thread

            @Override
            public void run() {
```

```
            try {

                Thread.sleep(1900); // Sleep for 1.9 seconds.
            } catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();
        }
    }).start(); // Starts the thread

    dialog.show();

    // Create an instance for the first product and adds it to the hash map.
    Products diyThirdProduct = new Products(current_product_id, diyThirdProductTxt.getText().toString(),
diyThirdProductColourMenu.getSelectedItem().toString(), (int) diyThirdProductQuantityMenu.getSelectedItemId(),
diyThirdProductCost.getText().toString(), diyThirdProductSizeMenu.getSelectedItem().toString());
    listOfProductsToAddToBasket.put(current_product_id, diyThirdProduct);

    return true;
  }

  private boolean diyAddToBasketFour() {
    Context context = getApplicationContext();
    String[] temp = new String[]{context.getString(R.string.addingBasket), context.getString(R.string.wait)};

    final ProgressDialog dialog = new ProgressDialog(DIYActivityTwo.this); // Spinning progress dialog
    dialog.setTitle(temp[0]); // Set the title of the dialog
    dialog.setMessage(temp[1]);

    dialog.setCancelable(false);

    dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER); // Sets the style of the progress bar

    new Thread(new Runnable() { // Create a new thread

        @Override
        public void run() {
            try {

                Thread.sleep(1900); // Sleep for 1.9 seconds.
            } catch (InterruptedException exc) {
                Log.d(String.valueOf(R.string.error), exc.toString());
            }

            dialog.dismiss();
        }
    }).start(); // Starts the thread

    dialog.show();

    // Create an instance for the first product and adds it to the hash map.
```

```
        Products diyFourthProduct = new Products(current_product_id++, diyFourthProductTxt.getText().toString(),
diyFourthProductColourMenu.getSelectedItem().toString(), (int) diyFourthProductQuantityMenu.getSelectedItemId(),
diyFourthProductCost.getText().toString(), diyFourthProductSizeMenu.getSelectedItem().toString());
        listOfProductsToAddToBasket.put(current_product_id++, diyFourthProduct);

        return true;
    }

    @Override
    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
        boolean valueAppended = false;

        int[] indexes = new int[]{0, 1, 2, 3, 4, 5};

        Context context = getApplicationContext();
        String[] productResources = new String[]{context.getString(R.string.productCost)};

        if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[0]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[0]);
            valueAppended = true;

        }

        else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[1]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[1]);
            valueAppended = true; // Value is appended

        }

        else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[2]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[2]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[3]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[3]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[4]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[4]);
            valueAppended = true;
        }

        else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesOne.get(indexes[5]))) {
            diyThirdProductCost.setText(null);
            diyThirdProductCost.append(productResources[0] + diyThirdProductCosts[5]);
```

```
    }

    if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[0]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[0]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[1]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[1]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[2]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[2]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[3]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[3]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[4]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[4]);
        valueAppended = true;
    }

    else if (parent.getItemAtPosition(position).equals(diyListOfQuantitiesTwo.get(indexes[5]))) {
        diyFourthProductCost.setText(null);
        diyFourthProductCost.append(productResources[0] + diyFourthProductCosts[5]);
    }
}

@Override
public void onNothingSelected(AdapterView<?> parent) {

}

@Override
public void onPointerCaptureChanged(boolean hasCapture) {

}

@Override
public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
    // Inflate the activities menu
    MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
```

```
        activityInflater.inflate(R.menu.homepagemenu, menu);

        MenuInflater menuInflater = getMenuInflater();
        menuInflater.inflate(R.menu.basket_action_button, menu);

        View view = menu.findItem(R.id.cart_menu).getActionView();

        cartIcon = view.findViewById(R.id.cart_icon);

        cartIcon.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {

                Intent basketIntent = new Intent(DIYActivityTwo.this, BasketActivity.class); // Create a basket intent
                basketIntent.putExtra("map", listOfProductsToAddToBasket); // Transit over the hash map data to the
basket
                startActivity(basketIntent); // Start the intent
            }
        });

        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which item has been selected.

        try {
            switch (item.getItemId()) {

                case R.id.mainActivity:
                    Intent homePage = new Intent(DIYActivityTwo.this, MainActivity.class);
                    startActivity(homePage);
                    break;

                case R.id.sportsAndOutdoorsCategory:
                    Intent sportsCategory = new Intent(DIYActivityTwo.this, SportsAndOutdoorsActivity.class);
                    startActivity(sportsCategory);

                    break;

                case R.id.techCategory:
                    Intent techActivity = new Intent(DIYActivityTwo.this, TechActivity.class);
                    startActivity(techActivity);
                    break;

                case R.id.clothingCategory:
                    Intent clothingActivity = new Intent(DIYActivityTwo.this, ClothingCategory.class);
                    startActivity(clothingActivity);
                    break;

                case R.id.diyCategory:
                    Intent diyActivity = new Intent(DIYActivityTwo.this, DIYActivity.class);
                    startActivity(diyActivity);
```

```
                break;

            default:
                super.onOptionsItemSelected(item);

        }

    }

    catch (ActivityNotFoundException exc) {
        Log.d(String.valueOf(R.string.error), exc.toString());
    }

    return true;
  }
}
```

Appendix – Application Layer Register Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.AlertDialog;
import android.app.NotificationChannel;
import android.app.NotificationManager;
import android.app.PendingIntent;
import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.media.RingtoneManager;
import android.net.Uri;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.core.app.NotificationCompat;
```

```
import androidx.core.app.NotificationManagerCompat;
import com.example.weshopapplication.R;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.OnFailureListener;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;
import java.util.HashMap;
import java.util.regex.Pattern;

// Author: Sabin Constantin Lungu.
// Matriculation Number: 40397517.
// Date of Last Modification: 08/03/2020
// Purpose of Activity: To allow users to register an account and write their registration data to a Firebase database.
// Any errors? Pending testing..

public class RegisterActivity extends AppCompatActivity { // Register class
    private static final String CHANNEL_ID = "register_channel"; // The channel to send the notification on
    private FirebaseFirestore db = FirebaseFirestore.getInstance(); // Get an instance of Firebase
    private EditText usernameField;

    private TextView registerText; // The register text
    private EditText passwordField; // The password entry field.
    private RadioButton termsAndConditions;

    private Button registerButton; // Register button
    private EditText emailAddressField;

    private boolean hasDigits; // True or false if the inputs have numbers
    private boolean startsWithUppercase; // True or false if the inputs start with an upper case.

    private boolean hasCharacters; // True or false if the input has characters
    private boolean hasRegex;

    private boolean isEmpty; // Variable that determines if any of the
    private boolean isValid;
    private boolean isRegistered;


    private NotificationManagerCompat notificationManager; // Notification manager variable
    private FirebaseAuth authentication = FirebaseAuth.getInstance();
    private Pattern regexPatterns = Pattern.compile("[$&+,:;=\\\\?@#|/'<>.^*()%!-]"); // Regex patterns that will be
checked.

    @Override
    protected void onCreate(Bundle savedInstanceState) { // Android Lifecycle method 1
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_register);
```

```
        // Initialise components
        this.usernameField = findViewById(R.id.usernameField);
        this.emailAddressField = findViewById(R.id.emailAddressField);

        this.registerText = findViewById(R.id.registerTxt);
        this.passwordField = findViewById(R.id.passwordField);

        this.termsAndConditions = findViewById(R.id.termsAndConditionsBox);
        this.registerButton = findViewById(R.id.registerBtn);
        notificationManager = NotificationManagerCompat.from(this); // Register the notification manager


        this.registerButton.setOnClickListener(new View.OnClickListener() { // Add listener to the button
            @Override
            public void onClick(View buttonView) {

                validateUsername(); // Call method to validate username
                validateEmailAddress(); // Validate the e-mail address

                validatePassword(); // Call method to validate the password
                validateTermsAndConditions();
            }
        });
    }

    public boolean onCreateOptionsMenu(Menu menu) { // Routine that creates the menu
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.register_menu, menu); // Inflate the menu

        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which menu item is chosen

        try {
            switch (item.getItemId()) {

                case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors category is clicked on
                    Intent sportsActivity = new Intent(RegisterActivity.this, SportsAndOutdoorsActivity.class); // Create intent
for sports activity
                    startActivity(sportsActivity);

                    return true;

                case R.id.techCategory:
                    Intent techActivity = new Intent(RegisterActivity.this, TechActivity.class);
                    startActivity(techActivity);

                    return true;

                case R.id.clothingCategory:
                    Intent clothingCategory = new Intent(RegisterActivity.this, ClothingCategory.class);
```

```
                startActivity(clothingCategory);

                return true;

            case R.id.diyCategory:
                Intent diyCategory = new Intent(RegisterActivity.this, DIYActivity.class);
                startActivity(diyCategory);

                return true;

            default:

                return super.onOptionsItemSelected(item); // Return the base item selected
        }
    }

    catch (ActivityNotFoundException act) {
        Log.d(String.valueOf(R.string.error), act.toString()); // Get the cause of the error.
    }

    return true;
}

public void onStart() { // Android Lifecycle method 2.
    super.onStart();
}

@Override
protected void onDestroy() {
    super.onDestroy();
    finish();
}

@Override
public void onBackPressed() {
    super.onBackPressed();
    moveTaskToBack(true); // Move back a activity
}

@Override
protected void onResume() {
    super.onResume();
}

@Override
protected void onStop() {
    super.onStop();
}

@Override
protected void onRestart() {
    super.onRestart();
```

```
    }

    @Override
    public boolean onNavigateUp() {
        return super.onNavigateUp();
    }

    private boolean validateUsername() { // Routine that validates the username entered by the user against specific
criteria
        String usernameInputField = usernameField.getText().toString().trim();

        Context context = getApplicationContext();
        String[] usernameValidationResource = new String[]{context.getString(R.string.empty),
context.getString(R.string.usernameLength), context.getString(R.string.usernameReEnter),
context.getString(R.string.usernameRegex),
                context.getString(R.string.usernameReEnter), context.getString(R.string.usernameRegexWarning)};

        if (usernameInputField.isEmpty()) { // If the input field is left empty

            AlertDialog.Builder emptyDialog = new
AlertDialog.Builder(RegisterActivity.this).setTitle(R.string.usernameError)
                    .setMessage(R.string.usernameErrorMsg).setNegativeButton(R.string.ok, new
DialogInterface.OnClickListener() {

                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) {
                                dialog.dismiss(); // Dismiss the dialogue
                            }
                        }
                    });

            emptyDialog.show();
            usernameField.setError(usernameValidationResource[0]); // Set the error.

            usernameField.setText(""); // Flush the empty field out
            isEmpty = true; // The field is empty

            isValid = false; // Is valid is false.

        }

        for (int i = 0; i < usernameInputField.length(); i++) { // Loop over the username

            if (!Character.isDigit(usernameInputField.charAt(i)) && usernameInputField.length() > 20) { // If the username
has no digits or the length is bigger than 20

                usernameField.setError(usernameValidationResource[1]);

                AlertDialog.Builder usernameError = new
AlertDialog.Builder(RegisterActivity.this).setMessage(usernameValidationResource[2])
                        .setTitle(usernameValidationResource[0]).setNegativeButton(R.string.ok, new
```

```
DialogInterface.OnClickListener() {
                    @Override

                    public void onClick(DialogInterface dialog, int which) {
                        if (dialog != null) {
                            dialog.dismiss();
                        }
                    }
                });

            usernameError.show(); // Show the dialogue
            usernameField.setText(""); // Flush out the data

            hasDigits = false; // Has digits is false.
            isValid = false;
            break;
        }

        if (Character.isDigit(usernameInputField.charAt(i)) && usernameInputField.length() > 20) { // If the username
field contains characters and the length is not 20
            isValid = true;
        }

        if (regexPatterns.matcher(usernameInputField).find()) { // If the username has a regex character.
            usernameField.setError(usernameValidationResource[3]);

            AlertDialog.Builder regexWarning = new
AlertDialog.Builder(RegisterActivity.this).setMessage(usernameValidationResource[4])
                    .setTitle(usernameValidationResource[5]).setNegativeButton(R.string.ok, new
DialogInterface.OnClickListener() {
                    @Override

                    public void onClick(DialogInterface dialog, int which) {
                        if (dialog != null) {

                            dialog.dismiss();
                        }
                    }
                });

            regexWarning.show();
            usernameField.setText("");
            isValid = false;

        } else {

            isValid = true;
            usernameField.setError(null); // Set username error to null if no errors occur
        }

    }
```

```
        return true;
    }

    private boolean validateEmailAddress() { // Routine that validates the e-mail address.
        Context context = getApplicationContext();
        String[] emailAddressResources = new String[]{context.getString(R.string.emailError),
context.getString(R.string.reEnterEmail), context.getString(R.string.emailEmpty),
context.getString(R.string.emailLength)
                , context.getString(R.string.emailAtSymbol), context.getString(R.string.emailRegexWarning)};

        String emailAddressInputField = emailAddressField.getText().toString().trim(); // Get the input for the
emailAddress

        if (emailAddressInputField.isEmpty()) {

            AlertDialog.Builder emailError = new
AlertDialog.Builder(RegisterActivity.this).setTitle(emailAddressResources[0]).setMessage(emailAddressResources[1]
)
                    .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                        @Override

                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) {

                                dialog.dismiss(); // Dismiss the dialogue if the dialog is not shown
                            }
                        }
                    });

            emailError.show();
            emailError.setCancelable(true); // User can click outside the window to cancel the error dialogue

            emailAddressField.setError(emailAddressResources[2]);
            isEmpty = true;
            isValid = false;

            return true;
        }

        if (emailAddressInputField.length() > 30) { // If the e-mail length is bigger than 25 characters
            emailAddressField.setError(emailAddressResources[3]); // Display error

            isValid = false; // Not valid
            return false;
        }

        if (!regexPatterns.matcher(emailAddressInputField).find()) { // If there is no regex characters matched including
the @ symbol that is needed

            emailAddressField.setError(emailAddressResources[4]);
            AlertDialog.Builder emailRegexWarning = new
AlertDialog.Builder(RegisterActivity.this).setTitle(emailAddressResources[5]).setMessage(emailAddressResources[4]
```

```
)
                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {

                    @Override
                    public void onClick(DialogInterface dialog, int which) {
                        if (dialog != null) { // If the dialog is not empty

                            dialog.dismiss();
                            dialog.cancel();
                        }
                    }
                });

            emailRegexWarning.show();
            emailAddressField.setText(""); // Set the field to empty
            isValid = false;

            return false;

        } else {

            isValid = true; // Otherwise the
            return true;
        }
    }

    private boolean validatePassword() { // Routine to validate the password
        Context context = getApplicationContext();

        String[] passwordResources = new String[]{context.getString(R.string.passwordWarning),
context.getString(R.string.passwordReEnter), context.getString(R.string.passwordRegexEmpty),
context.getString(R.string.passwordError),
                context.getString(R.string.passwordUppercase)};
        String passwordEntryField = passwordField.getText().toString().trim(); // Get the password input and trim it

        if (passwordEntryField.isEmpty() && !regexPatterns.matcher(passwordEntryField).matches()) { // If the password
is empty and there are no regex characters found

            AlertDialog.Builder passwordWarning = new
AlertDialog.Builder(RegisterActivity.this).setTitle(passwordResources[0])
                    .setMessage(passwordResources[1]).setNegativeButton(R.string.ok, new
DialogInterface.OnClickListener() {
                        @Override

                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) { // If the dialog is empty
                                dialog.dismiss(); // Dismiss it
                            }
                        }
                    });

            passwordWarning.show();
```

```
            passwordField.setText("");

            passwordField.setError(passwordResources[2]);
            isEmpty = true; // Is empty is true
            hasRegex = false;

            isValid = false; // Not valid
            return false; // Return false
        }

        for (int i = 0; i < passwordEntryField.length(); i++) { // Loop over the password entry

            if (!Character.isUpperCase(passwordEntryField.charAt(0))) { // If the password does not start with an upper
case character.

                AlertDialog.Builder pwUpperCase = new
AlertDialog.Builder(RegisterActivity.this).setTitle(passwordResources[3])

                        .setMessage(passwordResources[1]).setNegativeButton(R.string.ok, new
DialogInterface.OnClickListener() {
                            @Override
                            public void onClick(DialogInterface dialog, int which) {
                                if (dialog != null) {
                                    dialog.dismiss();
                                }
                            }
                        });

                pwUpperCase.show();
                passwordField.setText(""); // Set the field empty

                passwordField.setError(passwordResources[4]);
                startsWithUppercase = false;

                isValid = false;
                break;

            } else {
                isValid = true;
            }
        }

        return true;
    }

    private boolean validateTermsAndConditions() { // Validates the terms and conditions box
        Context context = getApplicationContext();

        String[] termsAndConditionsResources = new String[]{context.getString(R.string.termsAndConditionsError),
context.getString(R.string.tickTermsAndConditions),
                context.getString(R.string.termsAndConditionsMust)};
```

```
        if (!termsAndConditions.isChecked()) { // If the terms and conditions box is not checked

            AlertDialog.Builder boxError = new
AlertDialog.Builder(RegisterActivity.this).setTitle(termsAndConditionsResources[0])
                    .setMessage(termsAndConditionsResources[1])

                    .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                        @Override

                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) {
                                dialog.dismiss();
                            }
                        }
                    });

            boxError.show(); // Show the error
            termsAndConditions.setError(termsAndConditionsResources[2]);
            isValid = false;
        }

        if (termsAndConditions.isChecked() && isValid) { // If the terms and conditions box is checked and the validation
is all valid
            sendNotification(); // CALL METHOD TO SEND NOTIFICATION
            writeToDatabase(); // Write registration data to database

            writeToFirestore(); // Writes to firestore database
            showSpinningDialogue();
            transitionToLogin(); // Take user to login page

            isRegistered = true;

        }

        else {

            isRegistered = false;
            termsAndConditions.setError(null); // Otherwise set no error

            return false; // Otherwise return false
        }

        return true; // Fallback onto previous statement and return true
    }

    private void showSpinningDialogue() { // Routine that shows the spinning dialogue when register button is clicked
        // Create the progress dialogue
        final ProgressDialog dialog = new ProgressDialog(RegisterActivity.this); // The progress dialogue that will be
shown.
        Context context = getApplicationContext();

        String[] temp = new String[]{context.getString(R.string.creatingAccount), context.getString(R.string.wait)};
```

```
        dialog.setTitle(temp[0]);

        dialog.setMessage(temp[1]);
        dialog.setCancelable(false);

        dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

        new Thread(new Runnable() { // Create a new thread
            @Override
            public void run() {

                try {
                    Thread.sleep(4000); // Sleep for 4 seconds.
                } catch (InterruptedException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }

                dialog.dismiss(); // Dismiss the dialogue
            }
        }).start();

        dialog.show(); // Show the progress bar
    }


    private void sendNotification() { // Routine to send notification after registration
        int channelID = 12345; // The channel ID to send the notification on.
        NotificationManager notificationManager = (NotificationManager)
getSystemService(Context.NOTIFICATION_SERVICE);

        // create channel in new versions of android
        if (android.os.Build.VERSION.SDK_INT >= android.os.Build.VERSION_CODES.O) {
            int importance = NotificationManager.IMPORTANCE_HIGH;

            NotificationChannel notificationChannel = new NotificationChannel(CHANNEL_ID, CHANNEL_ID,
importance);
            notificationChannel.enableLights(true); // Enable the lights
            notificationChannel.setLightColor(Color.RED); // Sets the light colour

            notificationChannel.enableVibration(true);

            notificationChannel.setVibrationPattern(new long[]{100, 200, 300, 400, 500, 400, 300, 200, 400});
            notificationManager.createNotificationChannel(notificationChannel);
        }

        // Code below shows the notification
        Intent intent = new Intent(this, RegisterActivity.class); // Get the intent
        intent.addFlags(Intent.FLAG_ACTIVITY_CLEAR_TOP);
        // 0 is request code
        PendingIntent pendingIntent = PendingIntent.getActivity(this, 0, intent, PendingIntent.FLAG_ONE_SHOT); // Get
the pending intent activity
```

```
        Uri defaultSoundUri = RingtoneManager.getDefaultUri(RingtoneManager.TYPE_NOTIFICATION); // Creates a
uniform resource identifier for the sound.
        NotificationCompat.Builder notificationBuilder =
                new NotificationCompat.Builder(this, CHANNEL_ID) // Creates a new channel.
                        .setSmallIcon(R.drawable.ic_message_black_24dp)

                        .setContentTitle(getString(R.string.app_name))
                        .setContentText("Register Success") // Sets the content of the notification.
                        .setTicker("Success")
                        .setAutoCancel(true) // Auto cancels itself.
                        //.setSound(defaultSoundUri)
                        .setContentIntent(pendingIntent);

        notificationManager.notify(0, notificationBuilder.build()); // Shows the notification
    }

    private void writeToDatabase() { // Writes to database.

        // Get the user inputs
        String emailEntry = emailAddressField.getText().toString();
        String passwordEntry = passwordField.getText().toString();

        authentication.createUserWithEmailAndPassword(emailEntry, passwordEntry).addOnCompleteListener(new
OnCompleteListener<AuthResult>() { // Create user account with e-mail and password.
            @Override
            public void onComplete(@NonNull Task<AuthResult> task) {

                if (task.isSuccessful()) { // If the register task is successful

                    Toast.makeText(RegisterActivity.this, "Data written to DB", Toast.LENGTH_LONG).show(); // Debug
code.

                } else {

                    Toast.makeText(RegisterActivity.this, "Could not write to DB", Toast.LENGTH_LONG).show();
                }
            }
        });
    }

    private void writeToFirestore() { // Routine to write to Fire Store database.
        Context context = getApplicationContext();
        String[] registrationResources = new String[]{context.getString(R.string.basket_username),
context.getString(R.string.basket_email_address), context.getString(R.string.basket_password),
                context.getString(R.string.collection_name)};

        // Get the entries
        String usernameEntry = usernameField.getText().toString(); // Get the username entry
        String emailEntry = emailAddressField.getText().toString();
        String passwordEntry = passwordField.getText().toString();

        HashMap<String, Object> user_data = new HashMap<>(); // HashMap for the user data
```

```
        user_data.put(registrationResources[0], usernameEntry); // Add the Username Entry to the hash map
        user_data.put(registrationResources[1], emailEntry);
        user_data.put(registrationResources[2], passwordEntry);

        db.collection(registrationResources[3]).add(user_data).addOnSuccessListener(new
OnSuccessListener<DocumentReference>() {
            @Override

            public void onSuccess(DocumentReference documentReference) {
                Toast.makeText(RegisterActivity.this, "Added Data To Fire Store", Toast.LENGTH_LONG).show(); //
Debug toast to make sure that the data is added to fire store.
            }

        }).addOnFailureListener(new OnFailureListener() {
            @Override
            public void onFailure(@NonNull Exception e) {
                Log.d(String.valueOf(R.string.error), e.toString());
            }
        });
    }

    private void transitionToLogin() { // Take the user to the login page after registration

        try {
            // Take user to login
            Intent loginIntent = new Intent(RegisterActivity.this, LoginActivity.class);
            startActivity(loginIntent); // Start the login activity

        } catch (ActivityNotFoundException act) { // Catch the error if the activity is not found.

            Log.d(String.valueOf(R.string.error), act.toString()); // Log the error.
        }
    }
}
```

Appendix – Application Layer Login Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ProgressDialog;
import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
```

```
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.R;
import com.google.android.gms.tasks.OnCompleteListener;
import com.google.android.gms.tasks.Task;
import com.google.firebase.auth.AuthResult;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.firestore.FirebaseFirestore;
import java.util.regex.Pattern;

// Author of Application: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Activity: To login a registered user
// Date of last modified: 08/03/2020
// Any Bugs?: N/A

public class LoginActivity extends AppCompatActivity {
    private TextView loginText; // The login text at the top of the application
    private EditText emailAddressField;
    private EditText passwordField;

    private Button loginButton;
    private FirebaseFirestore firebaseFirestore;

    private FirebaseAuth auth; // Firebase authentication variable
    private Pattern regexPatterns = Pattern.compile("[$&+,:;=\\\\?@#|/'<>.^*()%!-]"); // Regex patterns
    private final int LOGOUT_BUTTON_ID = 101;

    private boolean isAdded = false;
    public boolean isLoggedIn = false;

    private Button logoutBtn;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);

        this.emailAddressField = findViewById(R.id.emailAddressField);
        this.passwordField = findViewById(R.id.passwordField);
        this.loginButton = findViewById(R.id.loginBtn);

        this.firebaseFirestore = FirebaseFirestore.getInstance();
```

```
    this.auth = FirebaseAuth.getInstance();

    this.logoutBtn = findViewById(R.id.logout_button);

    this.loginButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            validateEmailAddress();
            validatePassword();
        }
    });
}

@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater categoriesMenu = getMenuInflater();
    categoriesMenu.inflate(R.menu.homepagemenu, menu);

    return true;
}

public boolean onOptionsItemSelected(MenuItem item) {

    try {

        switch (item.getItemId()) {

            case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors category is chosen

                Intent sportsIntent = new Intent(LoginActivity.this, SportsAndOutdoorsActivity.class);
                startActivity(sportsIntent);

            case R.id.techCategory:

                Intent techCategoryIntent = new Intent(LoginActivity.this, TechActivity.class);
                startActivity(techCategoryIntent);

                return true;

            case R.id.clothingCategory:

                Intent clothingCategory = new Intent(LoginActivity.this, ClothingCategory.class);
                startActivity(clothingCategory);

                return true;

            case R.id.diyCategory:

                Intent diyIntent = new Intent(LoginActivity.this, DIYActivity.class);
                startActivity(diyIntent);

                return true;
```

```
            default:

                return super.onOptionsItemSelected(item);
            }
        }

        catch (ActivityNotFoundException act) {
            Log.d(String.valueOf(R.string.error), act.toString()); // Log the error as to why it occurred.
        }

        return true;
    }

    private boolean validateEmailAddress() { // Routine that validates the e-mail address when logging in.
        String emailEntry = emailAddressField.getText().toString().trim(); // Get the email address entry
        Context context = getApplicationContext();

        String[] tempResources = new String[]{context.getString(R.string.emailError),
context.getString(R.string.emailAtSymbol)};

        if (!regexPatterns.matcher(emailEntry).find() && emailEntry.isEmpty()) { // If there is no @ symbol and the email
field is empty

            AlertDialog.Builder emailError = new AlertDialog.Builder(LoginActivity.this).setTitle(R.string.error)
                    .setMessage(tempResources[0]).setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface dialog, int which) {
                            if (dialog != null) {
                                dialog.dismiss();
                            }
                        }
                    });

            emailError.show();
            emailAddressField.setError(tempResources[1]);

            emailAddressField.setText("");
            return false;

        } else {

            emailAddressField.setError(null); // Set no error otherwise
            login();
            return true;
        }
    }

    private boolean validatePassword() { // Routine that validates the password when logging in.

        String passwordEntry = passwordField.getText().toString().trim(); // Get the password entry
        String flushedString = "";
```

```
    if (passwordEntry.isEmpty()) { // If the password field is left empty
        AlertDialog.Builder passwordError = new AlertDialog.Builder(LoginActivity.this).setTitle("Password Error")
                .setMessage("Password should not be left empty")
                .setNegativeButton("OK", new DialogInterface.OnClickListener() {
                    @Override

                    public void onClick(DialogInterface dialog, int which) {
                        if (dialog != null) {

                            dialog.dismiss();
                        }
                    }
                });

        passwordError.show();
        passwordField.setError("Password cannot be left empty");
        passwordField.setText(flushedString);

        return false;

    } else {
        passwordField.setError(null);
        login();

        showLoginDialogue();
        return true;
    }
}

private void showLoginDialogue() {
    Context context = getApplicationContext();
    String[] tempResources = new String[]{context.getString(R.string.loggingIn), context.getString(R.string.wait)};
    final ProgressDialog dialog = new ProgressDialog(LoginActivity.this);

    dialog.setTitle(tempResources[0]);
    dialog.setMessage(tempResources[1]);

    dialog.setCancelable(false);
    dialog.setProgressStyle(ProgressDialog.STYLE_SPINNER);

    new Thread(new Runnable() { // Create a new thread
        @Override
        public void run() {
            try {

                Thread.sleep(2400);

            } catch (InterruptedException exc) {

                Log.d(String.valueOf(R.string.error), exc.toString());
            }
```

```
            dialog.dismiss();
        }
    }).start();

    dialog.show(); // Show the progress bar
}

private void login() { // Logs the user in
    final String emailInput = emailAddressField.getText().toString(); // Get the e-mail input
    String passwordInput = passwordField.getText().toString(); // Get the password input

    auth.signInWithEmailAndPassword(emailInput, passwordInput).addOnCompleteListener(LoginActivity.this, new
OnCompleteListener<AuthResult>() {
        @Override
        public void onComplete(@NonNull Task<AuthResult> task) {
            if (task.isSuccessful()) { // If the task is successful
                isLoggedIn = true;

                Toast.makeText(LoginActivity.this, "You are logged in as " + emailInput, Toast.LENGTH_LONG).show();
                transitionToHomepage(); // Take user to homepage
                setVisibilityOfLogout();
            }

            else if (!task.isSuccessful()) { // If the task is not successful, i.e the credentials do not match
                Toast.makeText(LoginActivity.this, "Invalid credentials", Toast.LENGTH_LONG).show(); // Show error
message
                isLoggedIn = false;
            }
        }
    });
}

private void setVisibilityOfLogout() {

}

public void transitionToHomepage() { // Routine that takes user to home page
    try {
        Intent homeIntent = new Intent(LoginActivity.this, MainActivity.class);
        startActivity(homeIntent);

    } catch (ActivityNotFoundException act) {
        Log.d(String.valueOf(R.string.error), act.toString());
    }
}
}
```

Appendix – Basket Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.graphics.Color;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ListView;
import android.widget.TextView;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.R;
import org.jetbrains.annotations.NotNull;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;

// Author of Application: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Application & Class: To store the products added to the basket in a List View.
// Date of Last Modification: 13/02/2020.
// Any Errors: N/A

public class BasketActivity extends AppCompatActivity implements View.OnClickListener { // The basket activity
inherits from the AppCompat Activity and implements the methods for the view on click listener.
    private Button placeOrderBtn; // Variable for the place order button
    private HashMap<Integer, Products> listOfProductsToAddToBasket = new HashMap<>();

    protected void onCreate(Bundle savedInstanceState) { // Android LifeCycle method. onCreate()
        super.onCreate(savedInstanceState);

        setContentView(R.layout.activity_basket); // Set the content view of the android app.

        this.placeOrderBtn = findViewById(R.id.placeOrderBtn); // Initialise component for the place order button
        this.placeOrderBtn.setOnClickListener(this); // Add an on click listener for the place order button.

        Intent intent = getIntent(); // Get the current intent.
        HashMap<Integer, Products> hashMap = (HashMap<Integer, Products>) intent.getSerializableExtra("map"); //
Get the hash map from the tech activity
        ArrayList<String> products = new ArrayList<>(); // Creates a temporary array list of products.
```

```
        ArrayAdapter<String> arrayAdapter = new ArrayAdapter<String>(BasketActivity.this,
android.R.layout.simple_list_item_1, products) { // Create a new array adapter for the basket activity.

            public View getView(int position, View convertView, @NotNull ViewGroup parent) {
                View view = super.getView(position, convertView, parent);

                TextView tv = view.findViewById(android.R.id.text1);

                tv.setTextColor(Color.WHITE); // Change the colour of the list view.

                return view; // Return the view
            }
        };

        ListView view = findViewById(R.id.listViewBasket); // Find the list view component
        view.setAdapter(arrayAdapter); // Set its adapter

        for (Map.Entry<Integer, Products> entry : hashMap.entrySet()) { // Loop over the hash map of products.
            arrayAdapter.add(entry.toString()); // Add the entries to the adapter list.
        }
    }

    protected void onDestroy() { // Android LifeCycle method. onDestroy() that destroys the current activity.
        super.onDestroy();
    }

    protected void onStop() { // Android LifeCycle Method to stop() the current activity.
        super.onStop();
        finish();
    }

    public void onBackPressed() { // Android LifeCycle Method that is called when the back button is clicked.
        super.onBackPressed();
        moveTaskToBack(true);
    }

    protected void onResume() { // Android LifeCycle method onResume() that resumes to the current activity if
another activity intervenes.
        super.onResume();
    }

    @Override
    public void onClick(View v) {
        Context context = getApplicationContext();
        String[] temp = new String[]{context.getString(R.string.finish)}; // Creates a string array of string values.

        try {
            if (v.getId() == R.id.placeOrderBtn) { // if the button is clicked

                AlertDialog.Builder builder = new AlertDialog.Builder(BasketActivity.this)
                        .setTitle(R.string.checkout) // Sets the title of the alert dialogue.
                        .setMessage(temp[0]) // Set the message to the first index in the array
```

```
                .setNegativeButton(R.string.no, new DialogInterface.OnClickListener() {
                    @Override

                    public void onClick(DialogInterface dialog, int which) {
                        if (dialog != null) { // If there is no dialogue.

                            dialog.dismiss(); // Close it.
                            finish(); // Finish the activity,
                        }
                    }
                }).setPositiveButton(R.string.yes, new DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface dialog, int which) {

                        Intent checkOutActivity = new Intent(BasketActivity.this, PaymentActivity.class); // Create an
intent to switch from the basket activity to the payment activity if the customer is happy with their basket.
                        checkOutActivity.putExtra("map", listOfProductsToAddToBasket); // Put the hash map in the list
of products basket view.
                        startActivity(checkOutActivity); // Starts the activity
                    }
                });

            builder.show(); // Show the dialogue.
            builder.setCancelable(true); // User can click outside the alert dialogue to close it.
        }
    }
    catch (ActivityNotFoundException exc) { // Catch the error if there is no activity.
        Log.d(String.valueOf(R.string.error), exc.toString()); // Log the error to the console.
    }
  }

  @Override
  public void onPointerCaptureChanged(boolean hasCapture) {

  }
}
```

Appendix – Application Layer Payment Activity Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.content.ActivityNotFoundException;
import android.content.Context;
import android.content.DialogInterface;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.AdapterView;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageView;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Spinner;
import android.widget.TextView;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.BusinessObjects.Months;
import com.example.weshopapplication.BusinessObjects.MonthsArrayAdapter;
import com.example.weshopapplication.BusinessObjects.Products;
import com.example.weshopapplication.BusinessObjects.SendPaymentInvoiceAPI;
import com.example.weshopapplication.BusinessObjects.Years;
import com.example.weshopapplication.BusinessObjects.YearsArrayAdapter;
import com.example.weshopapplication.DataLayer.PaymentDatabase;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.regex.Pattern;

// Author of Application/Class: Sabin Constantin Lungu
// Purpose of Application/Class: Allows Customers to pay for the products chosen
// Date of Last Modification: 22/02/2020
// Any Bugs? None

public class PaymentActivity extends AppCompatActivity implements AdapterView.OnItemSelectedListener {
    private RadioGroup paymentGroup;
    private RadioButton visaPayment;
    private int cardCVVLength = 3;
    private PaymentDatabase paymentDatabase;

    private RadioButton masterCardPayment;

    private EditText emailAddressField;
    private EditText cardNumber;
    private EditText cardCVV;
    private EditText cardholdersName;
```

```
    private ImageView cartIcon;
    private Button paypalBtn;

    private TextView expiryMonthLbl;
    private Spinner monthMenu;
    private Spinner yearsMenu;

    private MonthsArrayAdapter monthsArrayAdapter;
    private YearsArrayAdapter yearsArrayAdapter;

    private ArrayList<Months> listOfMonths = null; // An array list of months to add to the spinner
    private ArrayList<Years> listOfYears = null;

    private boolean isEmpty = false;
    private boolean isValid = false;
    private boolean paymentOptionChosen = false;

    private boolean isMonthChosen;
    private boolean isYearChosen;

    private boolean exceedsLength;
    private boolean hasRegex;
    private boolean hasSpace;

    private boolean hasDigits;

    private Button confirmPaymentBtn;
    private Pattern regexPatterns = Pattern.compile("[$&+,:;=\\\\?@#|/'<>.^*()%!-]"); // Regex patterns
    private HashMap<Integer, Products> orderSummary = new HashMap<>(); // A Hash Map data structure that stores
the order summary.

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_payment);
        this.paymentGroup = findViewById(R.id.paymentGroup);

        this.listOfMonths = new ArrayList<>();
        this.listOfYears = new ArrayList<>();

        this.visaPayment = findViewById(R.id.visaOption);
        this.masterCardPayment = findViewById(R.id.masterCardOption);
        this.paypalBtn = findViewById(R.id.paypalBtn);

        this.emailAddressField = findViewById(R.id.emailAddressPaymentField);

        this.cardNumber = findViewById(R.id.creditCardNumberField);
        this.cardCVV = findViewById(R.id.cardCVVField);
        this.cardholdersName = findViewById(R.id.cardNameField);

        this.monthMenu = findViewById(R.id.monthMenu);
        this.yearsMenu = findViewById(R.id.yearMenu);
```

```
        this.cartIcon = findViewById(R.id.cart_icon);

        this.expiryMonthLbl = findViewById(R.id.monthLbl);
        this.confirmPaymentBtn = findViewById(R.id.confirmPaymentBtn);

        addToMonthsList(); // Routine to add to the months list
        addToYearsList(); // Routine to add to the years array list.

        this.monthsArrayAdapter = new MonthsArrayAdapter(PaymentActivity.this, listOfMonths);
        monthsArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        monthMenu.setAdapter(monthsArrayAdapter);
        monthMenu.setOnItemSelectedListener(this);

        this.yearsArrayAdapter = new YearsArrayAdapter(PaymentActivity.this, listOfYears);
        yearsArrayAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);

        yearsMenu.setAdapter(yearsArrayAdapter);
        yearsMenu.setOnItemSelectedListener(this);

        this.paypalBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {

                    Intent paypalGateway = new Intent(PaymentActivity.this, PaypalPaymentGateway.class);
                    startActivity(paypalGateway);

                } catch (ActivityNotFoundException exc) {
                    Log.d(String.valueOf(R.string.error), exc.toString());
                }
            }
        });

        this.confirmPaymentBtn.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View confirmPaymentBtn) {

                if (confirmPaymentBtn.getId() == R.id.confirmPaymentBtn) {

                    if (monthMenu.getSelectedItemId() == 0 || yearsMenu.getSelectedItemId() == 0) { // If the index is 0 for
the month menu or the years menu

                        AlertDialog.Builder paymentError = new AlertDialog.Builder(PaymentActivity.this)
                                .setTitle(R.string.paymentErrorTitle)
                                .setMessage(R.string.expiryDateError)
                                .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                                    @Override
                                    public void onClick(DialogInterface dialog, int which) {
                                        if (dialog != null) {
                                            dialog.dismiss();
                                        }
```

```
                        }
                    });

                paymentError.show(); // Show the payment error
                paymentError.setCancelable(true);

                isMonthChosen = false;
                isYearChosen = false;
            } else {
                isMonthChosen = true;
                isYearChosen = true;
            }

            if (isMonthChosen && isYearChosen) { // If the month is chosen and the year is chosen by the customer
                validatePaymentOptions(); // Call method to validate the payment options
            }
        }
    }
});
}

private boolean validatePaymentOptions() { // Routine to validate the payment options Radio Buttons

    if (paymentGroup.getCheckedRadioButtonId() == -1) { // If the visa payment or paypal or the mastercard
payment are not checked.
        AlertDialog.Builder paymentError = new AlertDialog.Builder(PaymentActivity.this)

            .setTitle(R.string.paymentErrorTitle)
            .setMessage(R.string.paymentChoose)
            .setNegativeButton(R.string.ok, new DialogInterface.OnClickListener() {
                @Override
                public void onClick(DialogInterface dialog, int which) {
                    if (dialog != null) {
                        dialog.dismiss();
                    }
                }
            });

        paymentError.show();
        paymentError.setCancelable(true);

        paymentOptionChosen = false;
        isValid = false;

        return true;
    }

    else {
        isValid = true;
        paymentOptionChosen = true;
    }
```

```
        if (isValid && paymentOptionChosen) {
            validateEmailAddress();
        }

        return true;
    }

    private boolean validateEmailAddress() { // Routine that will validate the e-mail address input
        Context context = getApplicationContext();
        String[] emailResources = new String[]{context.getString(R.string.emailError),
context.getString(R.string.flushPaymentField)};

        String emailInput = emailAddressField.getText().toString();

        if (emailInput.isEmpty()) { // If the e-mail address is empty
            emailAddressField.setText(emailResources[1]); // Set an error message.
            emailAddressField.setError(emailResources[0]);

            isEmpty = true; // Is empty is true.
            isValid = false;
        } else {
            isEmpty = false;
            isValid = true;
        }

        if (!isEmpty && isValid) { // If the field is not empty has is valid
            validateCardNumber(); // Move on to validating the card number
        }

        return true;

    }

    private boolean validateCardNumber() {
        int cardLength = 20;
        String cardInput = cardNumber.getText().toString();
        Context context = getApplicationContext();

        String[] paymentErrors = new String[]{context.getString(R.string.cardEmpty)
                , context.getString(R.string.cardLength), context.getString(R.string.flushPaymentField),
context.getString(R.string.cardDigitsOnly),
        };

        if (cardInput.isEmpty()) {
            cardNumber.setError(paymentErrors[0]);
            cardNumber.setText(paymentErrors[2]);

            isEmpty = true;
            isValid = false;
        }

        else {
```

```
            isEmpty = false;
            isValid = true;
        }

        for (int i = 0; i < cardInput.length(); i++) {

            if (!Character.isDigit(cardInput.charAt(i))) { // If there are no digits or there is no space in between the digits

                cardNumber.setText("");
                cardNumber.setError("Card Number Must Only Contain Digits");

                hasDigits = false;
                isValid = false;
                break; // Break out the loop
            }

            if (cardInput.length() > cardLength) { // If the card input length exceeds 20 digits

                cardNumber.setError("Card Number Should Not Exceed 16 Digits"); // Show the error message
                cardNumber.setText(""); // Set the field to empty

                hasDigits = true; // Has digits is true
                exceedsLength = true; // Exceeds length is true

                isValid = false;
                break;
            }

            else {
                hasDigits = true;
                exceedsLength = false;
                isValid = true;
            }

            if (hasDigits && isValid && !isEmpty && !exceedsLength) {
                validateCardCVV();
                break;
            }
        }

        return true;
    }


    private boolean validateCardCVV() {
        Context context = getApplicationContext();

        String[] cardCVVResources = new String[]{context.getString(R.string.cardCVVError),
context.getString(R.string.flushPaymentField), context.getString(R.string.cardCVVError)};
        String cardCVVInput = cardCVV.getText().toString();

        if (cardCVVInput.isEmpty()) {
```

```
            cardCVV.setText(cardCVVResources[1]);
            cardCVV.setError(cardCVVResources[0]);

            isEmpty = true;
            isValid = false;
        }

        if (cardCVVInput.length() > cardCVVLength) { // If the length of the Card CVV is > 20
            cardCVV.setText(cardCVVResources[1]); // Set the error
            cardCVV.setError(cardCVVResources[2]);

            exceedsLength = true; // Exceeds length condition is no true
            isValid = false;
        }

        if (!isEmpty && isValid && !exceedsLength) { // If the field is not empty, if it's valid and does not exceed the
length
            validateCardHolderName(); // Call method to validate the card holder's name entry
        }

        return true;
    }

    private boolean validateCardHolderName() {
        Context context = getApplicationContext();

        String[] cardHolderNameResources = new String[]{context.getString(R.string.cardHolderNameEmpty),
context.getString(R.string.flushPaymentField), context.getString(R.string.cardHolderRegex),
context.getString(R.string.cardHolderDigits)};

        String cardHolderNameInput = cardholdersName.getText().toString(); // Get the user input.

        if (cardHolderNameInput.isEmpty()) {
            cardholdersName.setText(cardHolderNameResources[1]);
            cardholdersName.setError(cardHolderNameResources[0]);

            isEmpty = true;
            isValid = false;
        }

        if (regexPatterns.matcher(cardHolderNameInput).find()) { // If there is a special character found in the card
holder name input
            cardholdersName.setText(cardHolderNameResources[1]);
            cardholdersName.setError(cardHolderNameResources[2]);

            hasRegex = true; // Has regex flag is true.
            isValid = false;
        }

        else {
            isEmpty = false;
            isValid = true;
```

```
            hasRegex = false;
        }

        if (!hasRegex && isValid && !isEmpty) { // If the field does not have special characters, has no digits and is not
empty
            sendPaymentInvoice();
        }

        return true;
    }

    private void sendPaymentInvoice() {
        String mail = emailAddressField.getText().toString().trim();
        String subject = "Order Confirmation";
        String message = "Your Order Has Been Confirmed";

        SendPaymentInvoiceAPI sendPaymentInvoiceAPI = new SendPaymentInvoiceAPI(PaymentActivity.this, mail,
subject, message);
        sendPaymentInvoiceAPI.execute();

        writeToDatabase();
    }

    private void writeToDatabase() {
        String email_address = ((EditText) findViewById(R.id.emailAddressPaymentField)).getText().toString();
        String card_number = ((EditText) findViewById(R.id.creditCardNumberField)).getText().toString();

        String cardCVV = ((EditText) findViewById(R.id.cardCVVField)).getText().toString();
        String card_name = ((EditText) findViewById(R.id.cardNameField)).getText().toString();

        String expiry_month = ((Spinner) findViewById(R.id.monthMenu)).getSelectedItem().toString();
        String expiry_year = ((Spinner) findViewById(R.id.yearMenu)).getSelectedItem().toString();

        this.paymentDatabase = new PaymentDatabase(this);
        this.paymentDatabase.insert(email_address, card_number, cardCVV, card_name, expiry_month, expiry_year);

        transitionToHomePage();
    }

    public void checkButton(View view) { // Routine attached to the radio group to determine which radio button has
been selected.
        int optionChecked = paymentGroup.getCheckedRadioButtonId();
        visaPayment = findViewById(optionChecked);
    }

    private void transitionToHomePage() { // Routine to transition the customer to the home page after the payment
has been successful.
        try {

            Intent homeActivity = new Intent(PaymentActivity.this, MainActivity.class);
            startActivity(homeActivity);
```

```
        }

        catch (ActivityNotFoundException exc) { // Catch the error if the activity does not exist.
            Log.d(String.valueOf(R.string.error), exc.toString());
        }
    }

    private boolean addToMonthsList() {
        boolean month_added = false;
        Context context = getApplicationContext();

        String[] monthsResources = new String[]{context.getString(R.string.monthPrompt),
context.getString(R.string.januaryMonth),
                context.getString(R.string.februaryMonth),
                context.getString(R.string.marchMonth),
                context.getString(R.string.aprilMonth), context.getString(R.string.mayMonth),
context.getString(R.string.juneMonth), context.getString(R.string.julyMonth),
                context.getString(R.string.augustMonth), context.getString(R.string.septemberMonth),
context.getString(R.string.octoberMonth), context.getString(R.string.novemberMonth),
context.getString(R.string.decemberMonth)};


        Months[] theMonths = new Months[]{new Months(monthsResources[0]), new Months(monthsResources[1]), new
Months(monthsResources[2]), new Months(monthsResources[3]), new Months(monthsResources[4]),
                new Months(monthsResources[5]), new Months(monthsResources[6]), new Months(monthsResources[7]),
new Months(monthsResources[8]), new Months(monthsResources[9]), new Months(monthsResources[10]), new
Months(monthsResources[11]),
                new Months(monthsResources[12])};

        for (Months month : theMonths) {
            listOfMonths.add(month);
            month_added = true;
        }

        return true;
    }

    private boolean addToYearsList() {
        boolean years_added = false;
        Context context = getApplicationContext();

        String[] yearsResources = new String[]{context.getString(R.string.yearsPrompt),
context.getString(R.string.firstYear), context.getString(R.string.secondYear),
                context.getString(R.string.thirdYear), context.getString(R.string.fourthYear),
context.getString(R.string.fifthYear),
                context.getString(R.string.sixthYear), context.getString(R.string.seventhYear),
context.getString(R.string.eighthYear)};

        Years[] years = new Years[]{new Years(yearsResources[0]), new Years(yearsResources[1]), new
Years(yearsResources[2]), new Years(yearsResources[3]),
                new Years(yearsResources[4]), new Years(yearsResources[5]), new Years(yearsResources[6]), new
Years(yearsResources[7]), new Years(yearsResources[8])};
```

```
    for (Years theYears : years) {
        listOfYears.add(theYears);
        years_added = true;
    }

    return true;
}

@Override
public boolean onCreateOptionsMenu(Menu menu) { // Add the toolbar menu
    // Inflate the activities menu
    MenuInflater activityInflater = getMenuInflater(); // Get the activity inflater
    activityInflater.inflate(R.menu.homepagemenu, menu);

    MenuInflater menuInflater = getMenuInflater();
    menuInflater.inflate(R.menu.basket_action_button, menu);

    View view = menu.findItem(R.id.cart_menu).getActionView();

    cartIcon = view.findViewById(R.id.cart_icon);

    cartIcon.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {

            Intent basketIntent = new Intent(PaymentActivity.this, BasketActivity.class); // Create a basket intent
            basketIntent.putExtra("map", orderSummary); // Transit over the hash map data to the basket
            startActivity(basketIntent); // Start the intent
        }
    });

    return true;
}

@Override
public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {

}

@Override
public void onNothingSelected(AdapterView<?> parent) {

}

@Override
public void onPointerCaptureChanged(boolean hasCapture) {

}

public boolean onOptionsItemSelected(MenuItem item) { // Routine that determines which menu item is chosen
```

```
        try {
            switch (item.getItemId()) {

                case R.id.sportsAndOutdoorsCategory: // If the sports and outdoors category is clicked on
                    Intent sportsActivity = new Intent(PaymentActivity.this, SportsAndOutdoorsActivity.class); // Create intent
for sports activity
                    startActivity(sportsActivity);

                    return true;

                case R.id.techCategory:
                    Intent techActivity = new Intent(PaymentActivity.this, TechActivity.class);
                    startActivity(techActivity);

                    return true;

                case R.id.clothingCategory:
                    Intent clothingCategory = new Intent(PaymentActivity.this, ClothingCategory.class);
                    startActivity(clothingCategory);

                    return true;

                case R.id.diyCategory:
                    Intent diyCategory = new Intent(PaymentActivity.this, DIYActivity.class);
                    startActivity(diyCategory);

                    return true;

                default:

                    return super.onOptionsItemSelected(item); // Return the base item selected
            }
        }

        catch (ActivityNotFoundException act) {
            Log.d(String.valueOf(R.string.error), act.toString()); // Get the cause of the error.
        }

        return true;
    }

}
```

Appendix – Application Layer PayPal Gateway Activity Code

```
package com.example.weshopapplication.ApplicationLayer;
```

```
import android.graphics.Bitmap;
import android.os.Bundle;
import android.view.View;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.ProgressBar;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.R;

// Author of Application/Class: Sabin Constantin Lungu
// Purpose of Application/Class: Allows customers to pay with PayPal
// Date of Last Modification: 08/03/2020
// Any Bugs? None

public class PaypalPaymentGateway extends AppCompatActivity {
    private WebView paypalView;
    private ProgressBar progressBar;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_paypal_payment_gateway);

        this.paypalView = findViewById(R.id.webView);
        this.progressBar = findViewById(R.id.progressBar);

        paypalView.getSettings().setJavaScriptEnabled(true);
        paypalView.getSettings().setJavaScriptCanOpenWindowsAutomatically(true);

        paypalView.setWebViewClient(new WebViewClient() {

            public void onPageStarted(WebView view, String url, Bitmap favicon) {
                super.onPageStarted(view, url, favicon);

                paypalView.setVisibility(View.GONE);
                progressBar.setVisibility(View.VISIBLE);

                if (url.contains("https://www.google.co.uk/")) {
                    Toast.makeText(PaypalPaymentGateway.this, "Payment Cancelled", Toast.LENGTH_LONG).show();
                    finish();
                }

                else if (url.contains("https://www.facebook.com/")) {
                    Toast.makeText(PaypalPaymentGateway.this, "Payment Successful", Toast.LENGTH_LONG).show();
                }

            }

            public void onPageFinished(WebView view, String url) {
                super.onPageFinished(view, url);
                paypalView.setVisibility(View.VISIBLE);
```

```
            progressBar.setVisibility(View.GONE); // Progress Bar is gone now.
        }
    });

    paypalView.loadUrl("https://www.paypal.com/cgi-bin/webscr?cmd=_s-
xclick&hosted_button_id=C8KCZ98RCKUQW");

  }
}
```

Appendix – Application Layer Submit Complaint Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.content.DialogInterface;
import android.os.Bundle;
import android.view.View;
import android.widget.EditText;
import androidx.appcompat.app.AlertDialog;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.DataLayer.ContactUsDatabase;
import com.example.weshopapplication.R;

// Author of Application: Sabin Constantin Lungu
// Purpose of Class: Allows customers to Submit a complaint through the Contact Us Form if they are experiencing
any issues with the app.
// Date of Last Modification: 22/02/2020
// Any Bugs? None.

public class SubmitComplaint extends AppCompatActivity implements View.OnClickListener {
    private ContactUsDatabase databaseManipulator;

    private EditText usernameField;
    private EditText emailAddressField;
    private EditText phoneNumberField;

    private EditText problemField;
    private boolean isValidated = false;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_submit_complaint);

        View submitComplaintBtn = findViewById(R.id.btnSubmit);
        submitComplaintBtn.setOnClickListener(this);

        View backButton = findViewById(R.id.btnBack);
```

```
        backButton.setOnClickListener(this);

        this.usernameField = findViewById(R.id.add_usernameField);
        this.emailAddressField = findViewById(R.id.add_complaintEmailField);

        this.phoneNumberField = findViewById(R.id.add_complaint_phoneNumberField);
        this.problemField = findViewById(R.id.add_complaint_fieldProblem);
    }

    @Override
    public void onClick(View v) {
        switch (v.getId()) {

            case R.id.btnBack:
                this.finish();
                break;

            case R.id.btnSubmit: // When the submit button is clicked

                String username = ((EditText) findViewById(R.id.add_usernameField)).getText().toString();
                String email = ((EditText) findViewById(R.id.add_complaintEmailField)).getText().toString();

                String phone_number = ((EditText)
findViewById(R.id.add_complaint_phoneNumberField)).getText().toString();
                String problem = ((EditText) findViewById(R.id.add_complaint_fieldProblem)).getText().toString();

                this.databaseManipulator = new ContactUsDatabase(this);
                this.databaseManipulator.insert(username, email, phone_number, problem);

                showSavedComplaintsDialog();

                break;
        }
    }

    protected void showSavedComplaintsDialog() {
        AlertDialog.Builder builder;

        builder = new AlertDialog.Builder(SubmitComplaint.this);

        builder.setMessage(R.string.add_next_dialog_message);
        builder.setCancelable(false);

        builder.setNegativeButton("No", new DialogInterface.OnClickListener() {
            @Override

            public void onClick(DialogInterface dialog, int which) {
                SubmitComplaint.this.finish();
            }
        });

        builder.setPositiveButton(R.string.add_next_dialog_confirm_yes, new DialogInterface.OnClickListener() {
```

```
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.cancel();
        }
    });

    AlertDialog alert = builder.create();
    alert.show();
}

@Override
public void onPointerCaptureChanged(boolean hasCapture) {
}
}
```

Appendix – Application Layer Check Complaints Code

```
package com.example.weshopapplication.ApplicationLayer;

import android.app.ListActivity;
import android.os.Bundle;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import com.example.weshopapplication.DataLayer.ContactUsDatabase;
import com.example.weshopapplication.R;
import java.util.ArrayList;
import java.util.List;

// Author of Application Layer Class: Sabin Constantin Lungu
// Purpose of Application Layer Class: Allows users to view the complaints made. Useful for maintenance.
// Date of Last Modification: 13/02/2020.
// Any Errors: None.

public class CheckComplaints extends ListActivity { // Class Check Complaints inherits from the List Activity class
    public int idToModify;
    private TextView selection;
    private ContactUsDatabase manipulator;

    private List<String[]> listOfComplaints = new ArrayList<>(); // Creates a new list of complaints list.
    private List<String[]> listOfUsernames = null; // A list of Username.

    private String[] displayDataStrings; // A string array to display the strings in the list view

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_check_complaints);
```

```
        manipulator = new ContactUsDatabase(this); // Creates a new instance of the contact us database
        listOfUsernames = manipulator.selectAllData();

        displayDataStrings = new String[listOfUsernames.size()]; // Creates a new array of strirngs.
        int index = 0; // The index
        String temp;

        for (String[] usernames : listOfUsernames) { // For each username in the list of usernames
            temp = usernames[1] + " - " + usernames[2] + " - " + usernames[3] + " - " + usernames[4];
            displayDataStrings[index] = temp; // Display the strings
            index++; // Add the username to the next index.
        }

        ArrayAdapter<String> adapter = new ArrayAdapter<>(CheckComplaints.this,
android.R.layout.simple_list_item_1, displayDataStrings); // Create an array adapter to display the complaints in the
list view.
        this.setListAdapter(adapter);
        selection = findViewById(R.id.check_selection);
    }

    public void onListItemClick(ListView parent, View view, int position, long id) {
        selection.setText(displayDataStrings[position]);
    }
}
```

Appendix – Application Layer Contact Us Activity Code.

```
package com.example.weshopapplication.ApplicationLayer;

import android.content.ActivityNotFoundException;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import androidx.appcompat.app.AppCompatActivity;
import com.example.weshopapplication.R;

// Author of Application: Sabin Constantin Lungu
// Purpose of Class: To allow user's to fill out a contact us form if they are having issues with the app
// Date of last modification: 15/02/2020
// Any Errors? No

public class ContactUsActivity extends AppCompatActivity implements View.OnClickListener {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contact_us);
```

```
        View submitComplaint = findViewById(R.id.submitComplaintBtn);
        submitComplaint.setOnClickListener(this);

        View checkComplaint = findViewById(R.id.checkComplaintsBtn);
        checkComplaint.setOnClickListener(this);

    }

    @Override
    public void onClick(View v) {
        try {
            switch (v.getId()) {
                case R.id.submitComplaintBtn:
                    Intent submitComplaintIntent = new Intent(ContactUsActivity.this, SubmitComplaint.class);
                    startActivity(submitComplaintIntent);
                    break;

                case R.id.checkComplaintsBtn:
                    Intent checkComplaintsIntent = new Intent(ContactUsActivity.this, CheckComplaints.class);

                    startActivity(checkComplaintsIntent);
                    break;
            }

        } catch (ActivityNotFoundException exc) {

            Log.d(String.valueOf(R.string.error), exc.toString());
        }
    }

    @Override
    public void onPointerCaptureChanged(boolean hasCapture) {

    }
}
```

Appendix – Business Layer Size Class Code

```
package com.example.weshopapplication.BusinessObjects;

// Author of Business Layer Class: Sabin Constantin Lungu.
// Purpose of Business Layer Class: To store the size data.
```

```
// Matriculation Number: 40397517
// Date of Last Modification: 08/03/2020

public class Size {
    private int index;
    private String productSize;

    public Size(int index, String productSize) { // Size constructor
        this.index = index;
        this.productSize = productSize;
    }

    public int getIndex() { // Returns the required index.
        return this.index;
    }

    public void setIndex(int index) {
        this.index = index;
    }

    public String getProductSize() {
        return productSize;
    }

    public void setProductSize(String productSize) {
        this.productSize = productSize;
    }

    @Override
    public String toString() { // To string method to return the data
        return " " + this.productSize;
    }
}
```

Appendix – Business Layer Months Class Code

```
package com.example.weshopapplication.BusinessObjects;

// Author of Application: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Business Layer Class: To store the data for the Expiry Date Month that will be used in the Payment
Activity Class.
// Date of Last Modification: 08/03/2020
// Any Bugs? None

public class Months { // Months Class
    private String month; // The month variable

    public Months(String month) { // Constructor for the months that will set the data for the instantiated object
```

```
        this.month = month;
    }

    public String getMonth() { // Returns the month
        return this.month;
    }

    public void setMonth(String month) {
        this.month = month;
    }

    @Override
    public String toString() { // Method that returns the month data.
        return " " + this.month + " ";
    }
}
```

Appendix – Business Layer Years Class Code

```
package com.example.weshopapplication.BusinessObjects;

// Author of Application: Sabin Constantin Lungu
// Purpose of Business Layer Class: To store the data for the Years that will be used in the Payment Activity
// Date of Last Modification: 08/03/2020
// Any Bugs? None

public class Years { // Years class
    private String year; // Year variable

    public Years(String year) { // Constructor
        this.year = year;
    }

    public String getYear() { // Method to get the year
        return this.year;
    }

    public void setYear(String year) {
        this.year = year;
    }

    @Override
    public String toString() { // To string method will return the value of the year.
        return " " + year + " ";
    }
}
```

Appendix – Business Layer Products Class Code

```
package com.example.weshopapplication.BusinessObjects;

// Author of Application: Sabin Constantin Lungu 40397517
// Purpose of Application: To store data regarding the products when they are added to basket, the data will get displayed in a list view
// Date of Last Modification: 07/02/2020
// Any Errors? No

import java.io.Serializable;

public class Products implements Serializable { // Products Class
    private int productID; // The Product ID
    private String productName; // The Product Name
    private String colour; // Product Colour

    private int quantity;
    private String cost;
    private String size;

    public Products(int productID, String productName, String colour, int quantity, String cost, String size) { // Constructor for products class that stores the data necessary to represent a product
        this.productID = productID;
        this.productName = productName;
        this.colour = colour;
        this.quantity = quantity;
        this.cost = cost; // Set the cost.
        this.size = size;
    }

    public int getProductID() { // Returns the product ID
        return productID;
    }

    public void setProductID(int productID) { // Set the product id
        this.productID = productID;
    }

    public String getProductName() { // Returns the product name.
        return this.productName;
    }

    public void setProductName(String productName) {
        this.productName = productName;
    }
```

```
    public String getColour() { // Returns the product colour if called.
        return this.colour;
    }

    public void setColour(String colour) {
        this.colour = colour;
    }

    public int getQuantity() { // Returns the product quantity
        return this.quantity;
    }

    public void setQuantity(int quantity) { // Routine that will set the quantity of a product
        this.quantity = quantity;
    }

    public String getCost() { // Returns the product cost
        return this.cost;
    }

    public void setCost(String cost) { // Sets the cost of a product.
        this.cost = cost;
    }

    public String getSize() { // Returns the product size
        return this.size;
    }

    public void setSize(String size) { // Routine that sets the size of a product.
        this.size = size;
    }

    @Override
    public String toString() { // To string method returns all of the data from the instance
        return "Product Name : " + this.productName + "\n " + "Product Colour " + colour + "\n " + "Product Quantity : " +
this.quantity + "\n " + this.cost
                + "\n Product Size : " + this.size;
    }
}
```

Appendix – Business Layer Colors Array Adapter Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.content.Context;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import androidx.annotation.NonNull;
import com.example.weshopapplication.ApplicationLayer.TechActivity;
import java.util.ArrayList;

// Author of Application / Class: Sabin Constantin Lungu
// Purpose of Application / Class: A helper class that allows objects of type colour to be stored in the drop-down menu in the product categories.
// Date of Last Modification: 08/03/2020
// Any Errors? None

public class ColourArrayAdapter extends ArrayAdapter<TechActivity.Colours> { // The colours array adapter class inherits from the array adapter base class
    private Context context;
    private ArrayList<TechActivity.Colours> listOfColours = null; // The array list of colours to be stored.

    public ColourArrayAdapter(Context context, ArrayList<TechActivity.Colours> listOfColours) { // Constructor for  the colour array adapter.
        super(context, 0, listOfColours); // Inherit the context and list of colours
        this.context = context;
        this.listOfColours = listOfColours;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull ViewGroup parent) { // Routine that gets the list view at a specific position
        View listOfItems = convertView;

        if (listOfItems == null) { // If there is no list of items
            listOfItems = LayoutInflater.from(context).inflate(android.R.layout.simple_spinner_dropdown_item, parent, false); // Inflate a drop-down menu with the colours

            TechActivity.Colours colours = listOfColours.get(position);
        }

        return listOfItems; // Returns the list of items.
    }
}
```

Appendix – Business Layer Quantities Array Adapter Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.content.Context;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import androidx.annotation.NonNull;
import com.example.weshopapplication.ApplicationLayer.TechActivity;
import java.util.ArrayList;

// Author of Application / Class: Sabin Constantin Lungu
// Purpose of Application / Class: A helper class that allows objects of type Quantities to be stored in the drop-down menu in the product categories.
// Date of Last Modification: 08/03/2020
// Any Errors? None

public class QuantitiesArrayAdapter extends ArrayAdapter<TechActivity.Quantities> { // The quantities array adapter class inherits from the base array adapter class.

    private Context context;
    private ArrayList<TechActivity.Quantities> quantitiesList = null; // Array list of quantities

    public QuantitiesArrayAdapter(Context context, ArrayList<TechActivity.Quantities> quantitiesList) {
        super(context, 0, quantitiesList); // Super() used to inherit the features from the base class
        this.context = context;
        this.quantitiesList = quantitiesList;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull ViewGroup parent) {
        View listItems = convertView;

        if (listItems == null) { // If there is no list of items
            listItems = LayoutInflater.from(context).inflate(android.R.layout.simple_spinner_dropdown_item, parent, false);
// Inflate the drop-down menu with this type of instance Quantity

            TechActivity.Quantities quantities = quantitiesList.get(position);
        }

        return listItems; // Returns the quantities list of items.
    }
}
```

Appendix – Business Layer Months Array Adapter Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.content.Context;
```

```
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import androidx.annotation.NonNull;
import java.util.ArrayList;

// Author of Application / Class: Sabin Constantin Lungu
// Purpose of Application / Class: A helper class that allows objects of type Month to be stored in the drop-down menu in the product categories.
// Date of Last Modification: 08/03/2020
// Any Errors? None

public class MonthsArrayAdapter extends ArrayAdapter<Months> { // Array adapter class to allow the months to be stored as instances which will be added to an array list of months
    private Context context; // The context
    private ArrayList<Months> listOfMonths = null; // The array list of months

    public MonthsArrayAdapter(Context context, ArrayList<Months> listOfMonths) { // Constructor for months array adapter.
        super(context, 0, listOfMonths); // Inherit the features
        this.context = context;
        this.listOfMonths = listOfMonths;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull ViewGroup parent) { // Gets the view.
        View listOfItems = convertView;

        if (listOfItems == null) { // If there is no items
            listOfItems = LayoutInflater.from(context).inflate(android.R.layout.simple_spinner_dropdown_item, parent, false); // Inflate the list of items

            Months theMonths = listOfMonths.get(position); // Get the item position of the month
        }

        return listOfItems; // Return back the list of items
    }
}
```

Appendix – Business Layer Sizes Array Adapter Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.content.Context;
import android.view.LayoutInflater;
```

```
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import androidx.annotation.NonNull;
import com.example.weshopapplication.ApplicationLayer.TechActivity;
import java.util.ArrayList;

// Author: Sabin Constantin Lungu
// Purpose of Activity: Allows the sizes to be stored in a drop-down menu.
// Date of Last Modified: 4/2/2020
// Any Bugs?: Currently none. Unit tested recently. 11/11 Tests completed

public class SizesAdapter extends ArrayAdapter<TechActivity.Size> {
    private Context context;
    private ArrayList<TechActivity.Size> listOfSizes;

    public SizesAdapter(Context context, ArrayList<TechActivity.Size> listOfSizes) {
        super(context, 0, listOfSizes);
        this.context = context;
        this.listOfSizes = listOfSizes;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull ViewGroup parent) {
        View listOfItems = convertView;

        if (listOfItems == null) {
            listOfItems = LayoutInflater.from(context).inflate(android.R.layout.simple_spinner_dropdown_item, parent,
false);

            TechActivity.Size size = listOfSizes.get(position);
        }

        return listOfItems;
    }
}
```

Appendix – Business Layer Years Array Adapter Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
```

```
import androidx.annotation.NonNull;
import java.util.ArrayList;

// Author: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Activity: Allows the years to be stored in a drop-down menu.
// Date of Last Modified: 4/2/2020
// Any Bugs?: Currently none. Unit tested recently. 25/25 Tests completed

public class YearsArrayAdapter extends ArrayAdapter<Years> {
    private Context context;
    private ArrayList<Years> listOfYears = null;

    public YearsArrayAdapter(Context context, ArrayList<Years> listOfYears) {
        super(context, 0, listOfYears);
        this.context = context;
        this.listOfYears = listOfYears;
    }

    @NonNull
    @Override
    public View getView(int position, View convertView, @NonNull ViewGroup parent) { // Gets the view.
        View listOfItems = convertView;

        if (listOfItems == null) { // If there is no items
            listOfItems = LayoutInflater.from(context).inflate(android.R.layout.simple_spinner_dropdown_item, parent,
false); // Inflate the list of items

            Years theYears = listOfYears.get(position); // Get the item position of the month
        }

        return listOfItems; // Return back the list of items
    }
}
```

Appendix – Business Layer Send Payment Invoice API Class Code

```
package com.example.weshopapplication.BusinessObjects;

import android.app.ProgressDialog;
import android.content.Context;
import android.os.AsyncTask;
import android.widget.Toast;

import java.util.Properties;
```

```
import javax.mail.Message;
import javax.mail.MessagingException;
import javax.mail.PasswordAuthentication;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeMessage;

// Author of Application/Class: Sabin Constantin Lungu
// Purpose of Application/Class: This class uses the JavaMail API that automatically sends an e-mail to the user after they have purchased a specific product(s)
// Date of Last Modification: 19/02/2020
// Any Bugs? None

public class SendPaymentInvoiceAPI extends AsyncTask<Void, Void, Void> { // The class inherits from an Asynchronous Task Class. with 3 parameters initially Void
    private Context context; // The context.
    private Session session; // The current session.

    private String mail; // The mail instance.
    private String subject; // The subject variable.
    private String theMessage; // The message that will be sent will be stored in this variable.

    private ProgressDialog dialog; // The progress dialog

    public SendPaymentInvoiceAPI(Context context, String mail, String subject, String theMessage) { // Constructor for the send payment invoice that stores the context, email, subject and message
        this.context = context;
        this.mail = mail;
        this.subject = subject;
        this.theMessage = theMessage;
    }

    protected void onPreExecute() { // Method that is called on pre-execution.
        super.onPreExecute(); // Call the base method to pre execute.
        dialog = ProgressDialog.show(context, "Processing..", "Please wait..", false, false); // When the Confirm Payment button is clicked, this dialogue is shown.
    }

    protected void onPostExecute(Void aVoid) { // Method overridden after post-execution.
        super.onPostExecute(aVoid);

        dialog.dismiss(); // The dialogue is gone
        Toast.makeText(context, "Payment Invoice Sent Successfully", Toast.LENGTH_SHORT).show(); // Displays a toast message saying that the order is confirmed.
    }

    protected Void doInBackground(Void... params) { // A protected routine that performs the following tasks below in the background
        Properties properties = new Properties(); // Create a new properties instance

        properties.put("mail.smtp.host", "smtp.gmail.com");
```

```
        properties.put("mail.smtp.socketFactory.port", "465");
        properties.put("mail.smtp.socketFactory.class", "javax.net.ssl.SSLSocketFactory");
        properties.put("mail.smtp.auth", "true");
        properties.put("mail.smtp.port", "465");

        session = Session.getDefaultInstance(properties, new javax.mail.Authenticator() { // Gets the default instance of
the current session

            protected PasswordAuthentication getPasswordAuthentication() { // Routine to get the password
authentication
                return new PasswordAuthentication(MailCredentialsAPI.EMAIL_ADDRESS,
MailCredentialsAPI.PASSWORD); // Returns a new e-mail address and password each time.
            }
        });

        try {
            MimeMessage message = new MimeMessage(session); // A new message instance.
            message.setFrom(new InternetAddress(MailCredentialsAPI.EMAIL_ADDRESS));
            message.addRecipient(Message.RecipientType.TO, new InternetAddress(mail)); // Creates the recipient to
send the e-mail.

            message.setSubject(subject); // Sets the subject of the e-mail
            message.setText(theMessage); // Sets the text to be sent

            Transport.send(message); // Send the e-mail through the transport layer.
        } catch (MessagingException exc) { // Catch the exception if there is no recipient.
            exc.printStackTrace(); // Print the stack trace.
        }

        return null; // Return nothing otherwise
    }
}
```

Appendix – Business Layer Mail Credentials API Class Code

```
package com.example.weshopapplication.BusinessObjects;

// Author of Application / Class: Sabin Constantin Lungu
// Purpose of Application / Class: A helper class that stores the default E-mail Address and password to send the
invoice to.
// Date of Last Modification: 08/03/2020
// Any Errors? None

public class MailCredentialsAPI {
    public static final String EMAIL_ADDRESS = "sabinlungu293@gmail.com"; // The test e-mail address to send to.
    public static final String PASSWORD = "sphysiyjrfcjkpnt"; // The password.
}
```

Appendix – Data Layer Contact Us Database Class Code

```
package com.example.weshopapplication.DataLayer;

import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.database.sqlite.SQLiteStatement;
import java.util.ArrayList;
import java.util.List;

// Author of Application: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Application: To create a SQLite database to store the data filled in the contact us form activity
// Date of Last Modification: 15/02/2020
// Any Bugs? No. 25/25 Tests Passed.

public class ContactUsDatabase {

    private static final String DATABASE_NAME = "complaints.db"; // The Database to create
    private static int DB_VERSION = 1; // Database version.

    private static final String TABLE_NAME = "issues"; // The table name
    private static Context context; // The current context.

    private static final String INSERT_DATA = "INSERT INTO " + TABLE_NAME
            + " (username, email, phone_number, problem) VALUES (?,?,?,?)"; // Insert Query (DML) that inserts data
into the contacts DB
    private SQLiteStatement sqlStatement; // The SQL statement

    private static SQLiteDatabase db; // The SQL database

    public ContactUsDatabase(Context context) { // Constructor for the database manipulator
        ContactUsDatabase.context = context; // Sets the current context
        OpenHelper helper = new OpenHelper(ContactUsDatabase.context);
        ContactUsDatabase.db = helper.getWritableDatabase();

        this.sqlStatement = ContactUsDatabase.db.compileStatement(INSERT_DATA); // Sets the sql statement to
compile the INSERT DATA query to insert into the DB.

    }

    // Routine that inserts data into the table
    public long insert(String username, String email, String phone_number, String problem) { // Routine to insert data
into the table
        this.sqlStatement.bindString(1, username);
        this.sqlStatement.bindString(2, email);
```

```
        this.sqlStatement.bindString(3, phone_number); // Bind the
        this.sqlStatement.bindString(4, problem);

        return this.sqlStatement.executeInsert(); // Return the execution of the statement
    }


    public void deleteAllData() { // Routine to delete all the data from the DB
        db.delete(TABLE_NAME, null, null); // Deletes the table if required
    }

    public List<String[]> selectAllData() { // Routine to select all the data from the db
        List<String[]> listOfComplaints = new ArrayList<>(); // An array list of complaints

        Cursor cursor = db.query(TABLE_NAME, new String[]{"id", "username", "email", "phone_number", "problem"},
null, null, null, null, "username ASC");

        int index = 0;

        if (cursor.moveToFirst()) {

            do {
                String[] complaints_data = new String[]{cursor.getString(0), cursor.getString(1), cursor.getString(2),
cursor.getString(3), cursor.getString(4)};

                listOfComplaints.add(complaints_data); // Add the retrieved data to the array list
                index++; // Increment the index

            } while (cursor.moveToNext()); // While loop to go to the next row.
        }

        if (cursor != null && !cursor.isClosed()) {
            cursor.close(); // Close cursor
        }

        cursor.close();

        return listOfComplaints;
    }

    public static class OpenHelper extends SQLiteOpenHelper { // A Helper Class that inherits from
SQLiteOpenHelper.
        public OpenHelper(Context context) {
            super(context, DATABASE_NAME, null, DB_VERSION); // Inherit the features using super()
        }

        public void onCreate(SQLiteDatabase db) { // Creates the DB. Method overridden
            db.execSQL("CREATE TABLE " + TABLE_NAME + " (id INTEGER PRIMARY KEY, username TEXT, email
TEXT, phone_number TEXT, problem TEXT)"); // Creates the database table
        }
```

```
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
        DB_VERSION = newVersion; // Set the DB version to the newest version

        db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME); // DROP The specific table if it exists.
        onCreate(db); // Execute method
    }
  }
}
```

Appendix – Data Layer Payment Database Class Code

```
package com.example.weshopapplication.DataLayer;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import android.database.sqlite.SQLiteStatement;

// Author of Data Layer / Class: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Data Layer Class: The Payment Database Data Layer class is used to store the payment data of
customers in a MYSQLite Database.
// Date of Last Modification: 08/03/2020
// Any Errors? None

public class PaymentDatabase {
  private static final String DATABASE_NAME = "payments.db"; // The payment database to create.
  private static final String TABLE_NAME = "payments";

  private static final String INSERT_DATA = "INSERT INTO " + TABLE_NAME
      + " (email_address, card_number, card_cvv, card_name, expiry_month, expiry_year) VALUES (?,?,?,?,?,?)"; //
Insert Query (DML) that inserts data into the contacts DB
  private static int DATABASE_VERSION = 1;

  private static Context context; // The current context.
  private static SQLiteDatabase db; // The SQL database
  private SQLiteStatement sqlStatement; // The SQL statement

  public PaymentDatabase(Context context) { // Constructor for the database manipulator
    PaymentDatabase.context = context;
    PaymentDatabase.OpenHelper helper = new OpenHelper(PaymentDatabase.context);
    PaymentDatabase.db = helper.getWritableDatabase();

    this.sqlStatement = PaymentDatabase.db.compileStatement(INSERT_DATA); // Compiles the Insert query.
  }

  // Routine that inserts data into the table
```

```
    public long insert(String email_address, String card_number, String card_cvv, String card_name, String
expiry_month, String expiry_year) { // Routine to insert data into the table
        this.sqlStatement.bindString(1, email_address); // Binds the email_address string value into the database table.
        this.sqlStatement.bindString(2, card_number);

        this.sqlStatement.bindString(3, card_cvv);
        this.sqlStatement.bindString(4, card_name);
        this.sqlStatement.bindString(5, expiry_month);
        this.sqlStatement.bindString(6, expiry_year);

        return this.sqlStatement.executeInsert(); // Return the execution of the statement
    }

    public void deleteData() { // Routine is only called when the database needs to be deleted.
        db.delete(TABLE_NAME, null, null);
    }

    public static class OpenHelper extends SQLiteOpenHelper { // A Helper Class that inherits from SQLiteOpenHelper
        public OpenHelper(Context context) {
            super(context, DATABASE_NAME, null, DATABASE_VERSION); // Inherit the features from the base
SQLiteOpenHelper class. It inherits the context, database name and version.
        }

        public void onCreate(SQLiteDatabase db) { // Creates the DB. Method overridden
            // This line of code will create the table.
            db.execSQL("CREATE TABLE " + TABLE_NAME + " (id INTEGER PRIMARY KEY, email_address TEXT,
card_number TEXT, card_cvv TEXT, card_name TEXT, expiry_month TEXT, expiry_year TEXT)");
        }

        public void onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) {
            DATABASE_VERSION = newVersion; // Set the DB version to the newest version

            db.execSQL("DROP TABLE IF EXISTS " + TABLE_NAME); // DROP The specific table if it exists
            onCreate(db); // Execute method
        }
    }
}
```

Appendix – Unit Test Validators

```
package com.example.weshopapplication;

import android.text.Editable;
import android.text.TextWatcher;
import java.util.regex.Pattern;

// Author of Validators Unit Testing Class: Sabin Constantin Lungu
// Matriculation Number: 40397517
// Purpose of Validators Unit Testing Class: To provide helper methods that will be used in the UnitTests class to test
```

```
certain aspects of the software.
// Date of Last Modification: 08/03/2020
// Any bugs? No. 25/25 Tests Passed.

public class Validators implements TextWatcher { // Validator class implements the Android Text Watcher methods

    public static final Pattern PATTERN = Pattern.compile( // Regex patterns to use.
            "[a-zA-Z0-9\\+\\.\\_\\%\\-\\+]{1,256}" +
                    "\\@" +
                    "[a-zA-Z0-9][a-zA-Z0-9\\-]{0,64}" +
                    "(" +
                    "\\." +
                    "[a-zA-Z0-9][a-zA-Z0-9\\-]{0,25}" +
                    ")+"
    );

    private boolean isValid = false; // Valid field is false by default

    protected static boolean isValidUsername(CharSequence usernameField) { // Routine to determine if the username
is valid

        return usernameField != null && !PATTERN.matcher(usernameField).matches() && !(usernameField.length() >
20); // Is valid when the username field is not empty and does not have regex characters and also the length is not
bigger than 20
    }

    protected static boolean isValidEmailAddress(CharSequence emailAddress) { // Helper method that determines if
the E-mail Address input is valid or not.
        assert emailAddress != null;
        return PATTERN.matcher(emailAddress).matches() && !(emailAddress.length() > 30);
    }

    protected static boolean isValidPassword(CharSequence passwordEntryField) { // Determines if the password is
valid.
        return passwordEntryField != null && !PATTERN.matcher(passwordEntryField).matches() &&
Character.isUpperCase(passwordEntryField.charAt(0));
    }

    protected static boolean isValidCardNumber(CharSequence cardNumberEntryField) { // Determines if the card
number entry is valid.
        return cardNumberEntryField != null && !(cardNumberEntryField.length() > 20);
    }

    protected static boolean isValidCardCVV(CharSequence cardCVVEntry) {
        return cardCVVEntry != null && !(cardCVVEntry.length() > 3);
    }

    protected static boolean isValidCardHolderName(CharSequence cardHolderEntryField) {
        return cardHolderEntryField != null && PATTERN.matcher(cardHolderEntryField).matches();
    }

    boolean isValid() {
```

```
        return isValid;
    }

    @Override
    public void beforeTextChanged(CharSequence s, int start, int count, int after) {

    }

    @Override
    public void onTextChanged(CharSequence s, int start, int before, int count) {

    }

    @Override
    public void afterTextChanged(Editable textEntry) {
        isValid = isValidEmailAddress(textEntry);
        isValid = isValidUsername(textEntry);
        isValid = isValidPassword(textEntry);
    }
}
```

Appendix – Unit Tests Class Code

```
package com.example.weshopapplication;

import android.view.View;
import android.widget.EditText;
import androidx.test.ext.junit.runners.AndroidJUnit4;
import androidx.test.filters.LargeTest;
import androidx.test.rule.ActivityTestRule;
import com.example.weshopapplication.ApplicationLayer.BasketActivity;
import com.example.weshopapplication.ApplicationLayer.ClothingCategory;
import com.example.weshopapplication.ApplicationLayer.DIYActivity;
import com.example.weshopapplication.ApplicationLayer.LoginActivity;
import com.example.weshopapplication.ApplicationLayer.MainActivity;
import com.example.weshopapplication.ApplicationLayer.PaymentActivity;
import com.example.weshopapplication.ApplicationLayer.RegisterActivity;
import com.example.weshopapplication.ApplicationLayer.SportsAndOutdoorsActivity;
import com.example.weshopapplication.ApplicationLayer.TechActivity;
import org.junit.After;
import org.junit.Before;
import org.junit.Rule;
import org.junit.Test;
import org.junit.runner.RunWith;
import static org.junit.Assert.assertFalse;
import static org.junit.Assert.assertNotNull;
import static org.junit.Assert.assertTrue;
```

```
// Purpose of Test: To test if the MainActivity loads
// Matriculation Number: 40397517
// Author of Test: Sabin Constantin Lungu
// Date of Last Modification: 4/2/2020
// Tests Pass? : Yes. 25/25 Tests Pass.

@RunWith(AndroidJUnit4.class)
@LargeTest
public class UnitTests {

    // Rules created for each test
    @Rule
    public ActivityTestRule<MainActivity> activityRule = new ActivityTestRule<>(MainActivity.class);

    @Rule
    public ActivityTestRule<RegisterActivity> registerRule = new ActivityTestRule<>(RegisterActivity.class);

    @Rule
    public ActivityTestRule<LoginActivity> loginActivityRule = new ActivityTestRule<>(LoginActivity.class);

    @Rule
    public ActivityTestRule<TechActivity> techActivityActivityTestRule = new ActivityTestRule<>(TechActivity.class);

    @Rule
    public ActivityTestRule<BasketActivity> basketActivityActivityTestRule = new
ActivityTestRule<>(BasketActivity.class);

     @Rule
     public ActivityTestRule<ClothingCategory> clothingCategoryActivityTestRule = new
ActivityTestRule<>(ClothingCategory.class);

    @Rule
    public ActivityTestRule<SportsAndOutdoorsActivity> sportsAndOutdoorsActivityActivityTestRule = new
ActivityTestRule<>(SportsAndOutdoorsActivity.class);

    @Rule
    public ActivityTestRule<DIYActivity> diyActivityActivityTestRule = new ActivityTestRule<>(DIYActivity.class);

    @Rule
    public ActivityTestRule<PaymentActivity> paymentActivityActivityTestRule = new
ActivityTestRule<>(PaymentActivity.class);

    // Activities to be tested
    private MainActivity mainActivity = null; // The main activity is null by default.
    private RegisterActivity registerActivity = null;
    private LoginActivity loginActivity = null;

    private SportsAndOutdoorsActivity sportsAndOutdoorsActivity = null;
    private TechActivity techActivity = null;

    private ClothingCategory clothingCategory = null;
```

```
    private DIYActivity diyActivity = null;

    private BasketActivity productsBasket = null;
    private PaymentActivity paymentActivity = null;

    // Retrieve the data from the fields
    private EditText usernameTest;
    private EditText emailAddressTest;
    private EditText passwordTest;

    // Test Payment Input Fields
    private EditText cardNumberFieldTest; // The card number input field to be tested against certain conditions.
    private EditText cardCVVFieldTest;
    private EditText cardHolderNameTest;

    @Before
    public void setUp() { // Sets up the tests
        getActivities();
        // Get the edit text fields
        usernameTest = registerActivity.findViewById(R.id.usernameField);
        emailAddressTest = registerActivity.findViewById(R.id.emailAddressField);
        passwordTest = registerActivity.findViewById(R.id.passwordField);

        cardNumberFieldTest = paymentActivity.findViewById(R.id.creditCardNumberField);
        cardCVVFieldTest = paymentActivity.findViewById(R.id.cardCVVField);
        cardHolderNameTest = paymentActivity.findViewById(R.id.cardNameField);
    }

    public void getActivities() { // Retrieves the activities
        mainActivity = activityRule.getActivity(); // Get the activity
        registerActivity = registerRule.getActivity();
        clothingCategory = clothingCategoryActivityTestRule.getActivity(); // Get the clothing activity

        loginActivity = loginActivityRule.getActivity(); // Get the login activity
        techActivity = techActivityActivityTestRule.getActivity(); // Get the tech activity
        productsBasket = basketActivityActivityTestRule.getActivity();
        paymentActivity = paymentActivityActivityTestRule.getActivity();

        diyActivity = diyActivityActivityTestRule.getActivity();

        sportsAndOutdoorsActivity = sportsAndOutdoorsActivityActivityTestRule.getActivity(); // Get the sports and
outdoors activity
    }

    @Test
    public void testMainActivityLauncher() { // Routine that tests if the main activity launches.
        View view = mainActivity.findViewById(R.id.welcomeTxt);

        assertNotNull(view);
    }

    public void testPreconditions() { // Test preconditions
```

```
        assertNotNull(usernameTest);
        assertNotNull(emailAddressTest);
        assertNotNull(passwordTest);

        assertNotNull(cardNumberFieldTest);
        assertNotNull(cardCVVFieldTest);
        assertNotNull(cardHolderNameTest);
    }

    @Test
    public void testValidUsername() { // Test Routine to test a valid Username Entry field.
        assertTrue(usernameTest.getText().toString(), Validators.isValidUsername("sabin2000")); // Test should pass
because username is valid
    }

    @Test
    public void testEmptyUsernameEntry() { // Test routine to test if a Username is empty. Test should fail because it
has special characters.
        assertTrue(usernameTest.getText().toString(), Validators.isValidUsername(" "));
    }

    @Test
    public void testUsernameLengthExceeds() { // Test Routine to test if a Username entry exceeds 20 characters.
Test should fail because it exceeds 20.
        assertFalse(usernameTest.getText().toString(),
Validators.isValidUsername("sabinOafjdfhiusdfhsdiufhuAIUFhiufsflsdkl"));
    }

    @Test
    public void testValidEmailAddress() { // Test stub to test if the E-mail Address entered is the one to expect. Test
should pass
        assertTrue(emailAddressTest.getText().toString(),
Validators.isValidEmailAddress("sabinlungu293@gmail.com"));
    }

    @Test
    public void testEmailAddressLengthExceeds() { // Test Method to test if the E-mail Address length is > 30. Test
Should Pass.
        assertFalse(emailAddressTest.getText().toString(),
Validators.isValidEmailAddress("bobmichaeltesemailaddressparkinsonapplescarstobuy@yahoo.com"));
    }

    @Test
    public void testEmptyEmailAddress() { // Test Stub to test if the email address entered is an empty string. This test
should pass as the entry field is empty
        assertFalse(emailAddressTest.getText().toString(), Validators.isValidEmailAddress(" "));
    }

    @Test
    public void testEmailAddressRegex() { // Test Routine to test if the E-mail Address contains an @ symbol. Test will
pass.
        assertFalse(emailAddressTest.getText().toString(), Validators.isValidEmailAddress("sabinlungu293gmail.com"));
```

```
    }

    @Test
    public void testValidPassword() { // Test Routine to test if the Password entry field is valid: Starts with Uppercase,
has regex characters and not empty. This test should pass
        assertTrue(passwordTest.getText().toString(), Validators.isValidPassword("Sabin2000*@("));
    }

    @Test
    public void testNoRegexPassword() { // Test routine to check if the password entry has special characters.
        assertTrue(passwordTest.getText().toString(), Validators.isValidPassword("Sabin2000"));
    }

    @Test
    public void testValidCardNumber() { // Test routine to check if the card number entry is valid or not.
        assertTrue(cardNumberFieldTest.getText().toString(), Validators.isValidCardNumber("1234000090991234"));
    }

    @Test
    public void testCardNumberLength() { // Test routine to check if the card number of the payment activity is > 20.
Test fails as length is bigger than 20
        assertFalse(cardNumberFieldTest.getText().toString(),
Validators.isValidCardNumber("943274837429384638746237468723482734723764"));
    }

    @Test
    public void testEmptyCardNumberField() {
        assertTrue(cardNumberFieldTest.getText().toString(), Validators.isValidCardNumber(" "));
    }

    @Test
    public void testValidCardCVV() {
        assertTrue(cardCVVFieldTest.getText().toString(), Validators.isValidCardCVV("218"));
    }

    @Test
    public void testCVVLength() {
        assertFalse(cardCVVFieldTest.getText().toString(), Validators.isValidCardCVV("1234"));
    }

    @Test
    public void testValidCardHolderName() { // Tests to see if the card holder name is valid.
        assertFalse(cardHolderNameTest.getText().toString(), Validators.isValidCardHolderName("Sabin Lungu")); //
Test will pass because the name is valid.
    }

    @Test
    public void testEmptyCardHolderName() {
        assertFalse(cardHolderNameTest.getText().toString(), Validators.isValidCardHolderName(" "));
    }

    @Test
```

```
public void testRegisterActivityLauncher() {
    View registerView = registerActivity.findViewById(R.id.registerTxt);
    assertNotNull(registerView);
}

@Test
public void testSportsAndOutdoorsActivityLauncher() {
    View sportsView = sportsAndOutdoorsActivity.findViewById(R.id.firstSportsOutdoorCostLbl);
    assertNotNull(sportsView);
}

@Test
public void testTechActivityLauncher() { // Tests to see if the tech activity loads correctly.
    View activityView = techActivity.findViewById(R.id.firstProductImg);
    assertNotNull(activityView);
}

@Test
public void testClothingActivityLauncher() { // Tests to see if the clothing activity is loaded
    View activityView = clothingCategory.findViewById(R.id.clothingFirstProductCostLbl);
    assertNotNull(activityView);
}

@Test
public void testDIYActivityLauncher() {
    View diyView = diyActivity.findViewById(R.id.diyFirstProductCostTxt);
    assertNotNull(diyView);
}

@Test
public void testLoadBasketActivity() { // Tests to see if the basket loads correctly.
    View basketView = productsBasket.findViewById(R.id.placeOrderBtn);
    assertNotNull(basketView);
}

@Test
public void testLoginActivityLauncher() { // Test stub that tests to see if the login activity launches
    View loginView = loginActivity.findViewById(R.id.loginBtn); // Finds the login button
    assertNotNull(loginView); // Check condition
}

@Test
public void testPaymentActivityLauncher() {
    View paymentView = paymentActivity.findViewById(R.id.paymentTxt);
    assertNotNull(paymentView);
}

@After
public void tearDown() { // After testing
    // Empty activities after testing
    mainActivity = null;
    registerActivity = null;
```

```
        loginActivity = null;

        diyActivity = null;

        techActivity = null;
        productsBasket = null;
    }
}
```

Appendix – List of Resources Used

- I got the MySQLite Database Code from the Mobile Application Development Practical Lab.

https://www.udemy.com/course/java-android-complete-guide/

https://dzone.com/articles/cloud-firestore-read-write-update-and-delete

http://www.myappwiz.com/home/appdetail?platform=android&appID=com.MidCenturyMedia.Shopper.light&refer=fromSimilar&name=Shopper%3A+Grocery+Shopping+List

https://www.youtube.com/watch?v=I-I67MNRQZM

https://www.colorhexa.com/2052ff

https://stackoverflow.com/questions/29949501/android-show-notification-with-a-popup-on-top-of-any-application

https://stackoverflow.com/questions/38436799/android-unit-test-start

https://stackoverflow.com/questions/4792260/how-do-you-change-text-to-bold-in-android

https://stackoverflow.com/questions/44305206/ask-permission-for-push-notification

https://github.com/Musfick/JavaMailAPIDemo/blob/master/app/src/main/java/com/teamcreative/javamailapidemo/JavaMailAPI.java

https://stackoverflow.com/questions/2020088/sending-email-in-android-using-javamail-api-without-using-the-default-built-in-a -

https://www.apple.com/uk/shop/product/FQAG2B/A/Refurbished-iPhone-X-256GB-Silver

https://www.apple.com/uk/apple-watch-series-5/

https://www.apple.com/uk/shop/product/MWP22ZM/A/airpods-pro

https://www.samsung.com/uk/smartphones/galaxy-s10-sm-g973-hybrid-sim/SM-G973FZWDBTU/buy/?cid=UK_PPC_8324597618_Evaluate_Brand+%2B+Product+(Tease)_S10_Exact+-+cclearly+experiment&gclid=CjwKCAiAzJLzBRAZEiwAmZb0agx-s4ggYrS_PtBP4qVTsfkge6pk3kP_vc3zdghQgBUUX2j3-WqWjBoCQ0wQAvD_BwE&gclsrc=aw.ds

https://www.nike.com/gb/w/air-max-shoes-a6d8hzy7ok

https://medium.com/mindorks/custom-array-adapters-made-easy-b6c4930560dd

https://android--code.blogspot.com/2015/08/android-listview-text-color.html

https://www.adidas.co.uk/predator-mutator-20.1-firm-ground-boots/EF1992.html

https://www.adidas.co.uk/helionic-down-jacket/BQ1935.html

https://www.adidas.co.uk/predator-mutator-20.1-firm-ground-boots/EF2206.html

https://www.adidas.co.uk/helionic-down-jacket/BQ1935.html

https://www.burton.co.uk/en/bruk/product/clothing-281559/mens-clothing-view-all-1865858/rust-long-sleeve-check-shirt-9364496

https://www.burton.co.uk/en/bruk/product/clothing-281559/mens-jeans-281570/blue-raw-denim-carter-tapered-fit-jeans-8856060

https://www.matalan.co.uk/product/detail/s2760741_c323/long-sleeve-henley-top-pink

https://www.matalan.co.uk/product/detail/s2725941_c101/longline-hoodie-black

https://www.diy.com/departments/dulux-weathershield-gallant-grey-satin-metal-wood-paint-2-5l/1923943_BQ.prd

https://www.diy.com/departments/diall-paint-brush-set-pack-of-3/1600728_BQ.prd

https://www.diy.com/departments/a-s-creation-life-grey-tree-silver-effect-textured-wallpaper/1212827_BQ.prd

https://www.diy.com/departments/qep-colour-enhancing-tile-sealant-1l/655795_BQ.prd

https://www.youtube.com/watch?v=D1brwtjjOLk

https://stackoverflow.com/questions/16586409/how-can-i-create-tests-in-android-studio