Student Name: Daniel Eshkeri

Student ID: 21534117

Course: BSc Information Technology Management for Business

Module: Mobile Web Application Development

Module Leader: Cain Kazimoglu

Video Link: https://youtu.be/CG1ant0gw4E

Word Count Minus Title Page, Table of Contents/Figures, References and Appendix: 1307

Table of Contents

Introduction 5

Wireframing, UI Design and Development 5

- Main Activity 5
- Product Activity 6
- Bottom Sheet 7
- Basket Activity 8
- Checkout Activtiy 9
- Order Success Activity 10
- Orders Activity 10
- Login Activity 11
- Register Activity 12
- Profile Activity 13
- Contact Activity 14
- Admin Web Panel 15

Backend Development 17

- Home Page 17
- Registration 18
- Product Page 19
- Checkout, Order Success Page 20

Reflection and Discussion 21

Conclusion and Future Work 21

Appendix/Code 22

- Account Activity Part 1 22
- Account Activity Part 2 23
- Account Activity Part 3 24
- Account Activity Part 4 25
- Basket Activity 26
- Basket Adapter Part 1 27
- Basket Adapter Part 2 28
- Basket Adapter Part 3 29
- BasketDB Part 1 30
- BasketDB Part 2 31
- BasketDB Part 3 32
- Checkout Activity Part 1 33
- Checkout Activity Part 2 34

Checkout Activity Part 3 35
Contact Activity 36
DBHelper 37
Image Adapter 38
Image View Activity 39
Login Activity Part 1 40
Login Activity Part 2 41
Main Activity Part 1 42
Main Activity Part 2 43
Main Activity Part 3 44
Main Activity Part 4 45
Order Confirmed Activity Part 1 46
Order Confirmed Activity Part 2 47
Order Confirmed Activity Part 3 48
Orders Activity 49
Orders Adapter Part 1 50
Orders Adapter Part 2 51
Product 52
Product Activity Part 1 53
Product Activity Part 2 54
Product Activity Part 3 55
Product Activity Part 4 56
Product Activity Part 5 57
Product Adapter Part 1 58
Product Adapter Part 2 59
Product Adapter Part 3 60
Register Activity Part 1 61
Register Activity Part 2 62
Register Activity Part 3 63
Register Activity Part 4 64

Table of Figures

Figure 1.1: Main Wireframe Old........5
Figure 1.2: Main Wireframe New........5
Figure 1.3: Main Activity Final........5
Figure 2.1: Product Wireframe........6
Figure 2.2: Product Activity........6
Figure 2.3: Product Activity Out of Stock........6
Figure 3.1: Bottom Sheet Signed Out Wireframe........7
Figure 3.2: Bottom Sheet Signed In Wireframe........7
Figure 3.3: Bottom Sheet Signed Out App........7
Figure 3.4: Bottom Sheet Signed In App........7
Figure 4.1: Basket Wireframe........8
Figure 4.2: Basket Activity........8
Figure 5.1: Checkout Wireframe........9
Figure 5.2: Checkout Activity........9
Figure 6.1: Order Success Wireframe........10
Figure 6.2: Order Success Activity........10
Figure 7.1: Orders Activity........10
Figure 8.1: Login Wireframe........11
Figure 8.2: Login Activity........11
Figure 9.1: Register Wireframe 1........12
Figure 9.2: Register Wireframe 2........12
Figure 9.3: Register Wireframe 3........12
Figure 9.4: Register Activity 1........12
Figure 9.5: Register Activity 2........12
Figure 9.6: Register Activity 3........12
Figure 10.1: Profile Wireframe........13
Figure 10.2: Profile Activity 1........13
Figure 10.3: Profile Activity 2........13
Figure 10.4: Profile Activity 3........13
Figure 11.1: Contact Wireframe........14
Figure 11.2: Contact Activity........14
Figure 11.3: Send pressed........14
Figure 11.4: Email client pressed........14
Figure 12.1: Lucky Merch Admin Panel........15
Figure 12.2: Price Sort........15
Figure 12.3: Quantity Sort........15
Figure 12.4: Category Filter........15
Figure 12.5: Add New Product........16
Figure 12.6: Edit Product........16
Figure 12.7: Delete Product........16
Figure 12.8: Orders Tab........17
Figure 13.1: Get Collection Category Function – Main Activity........18
Figure 14.1: Sign Up Function – Registration Activity........18
Figure 15.1: Add to Basket Button – Product Activity........19
Figure 16.1: Add Order to Server Function – Order Success Activity........20
Figure 16.2: Stock Update Function – Order Success Activity........20

Introduction

In this report I clearly articulate the creation and reflection of the application I created for this assessment. Lucky Merch is a celebrity merchandise store with a web app component for the administration interface. I used Firebase for the database and Figma for the wireframes as I've had previous experience with both.

This report is split into 5 sections:

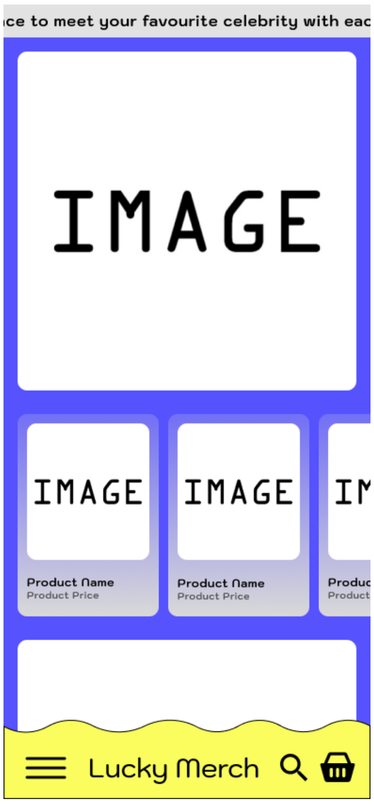
1. Wireframing, UI Design and Development
2. Back End Development
3. Reflection and Discussion
4. Conclusion and Future Work
5. Appendix/Code

Please enjoy this report and thank you for the opportunity to present my work.

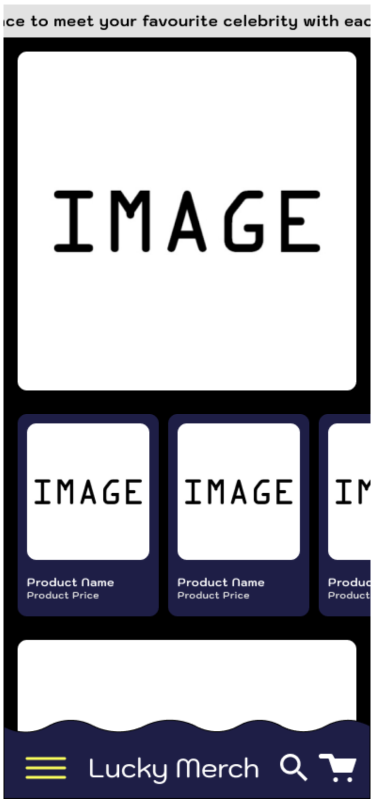
Wireframing, UI Design and Development

My application had several design stages. Originally, I tried a bold colour palette. After receiving feedback from my peers and lecturers I decided to tone down the colours but keep the original design language. In addition, some further changes were made during development.

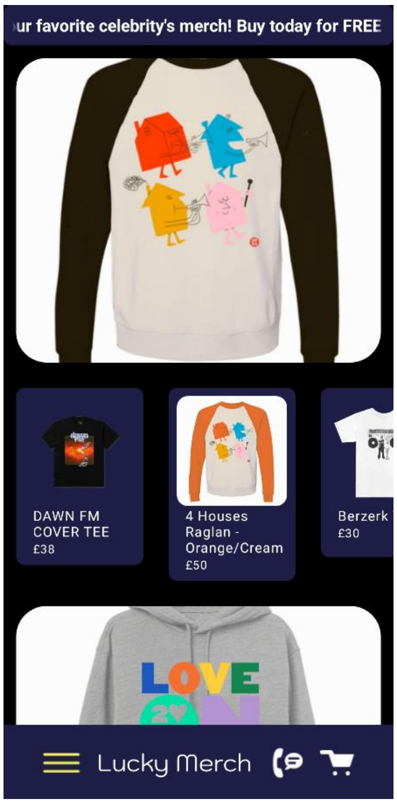
Main Activity



(Figure 1.1: Main Wireframe Old)

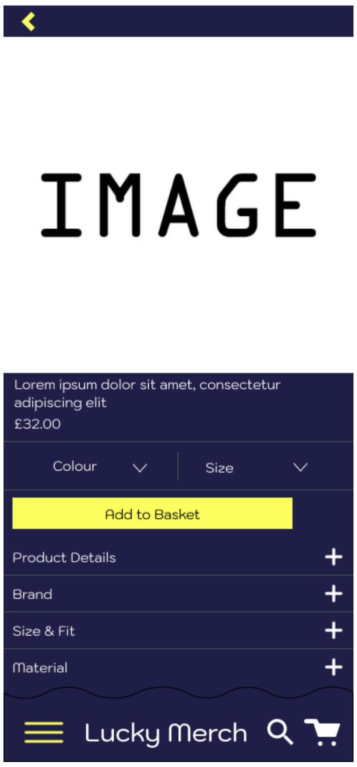


(Figure 1.2: Main Wireframe New)

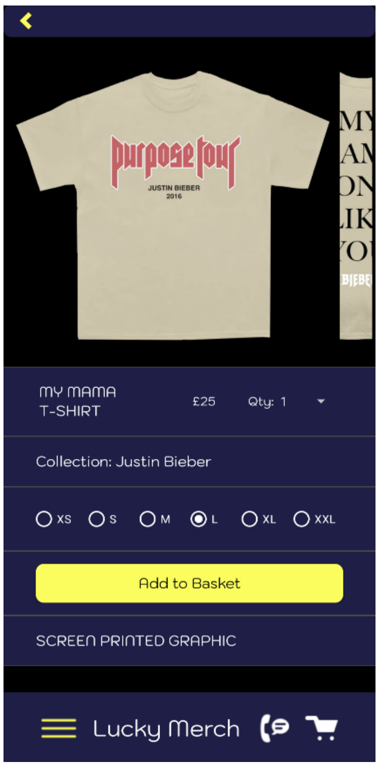

ur favorite celebrity's merch! Buy today for FREE
DAWN FM COVER TEE
£38
4 Houses Raglan - Orange/Cream
£50
Berzerk
£30
Lucky Merch

(Figure 1.3: Main Activity Final)

Product Activity

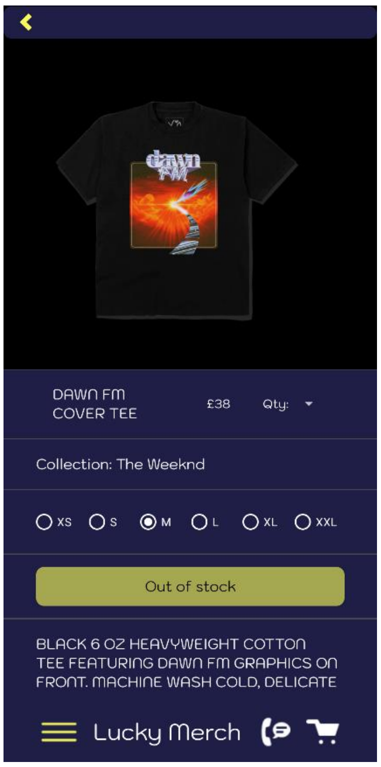


(Figure 2.1: Product Wireframe)



SCREEN PRINTED GRAPHIC

(Figure 2.2: Product Activity)

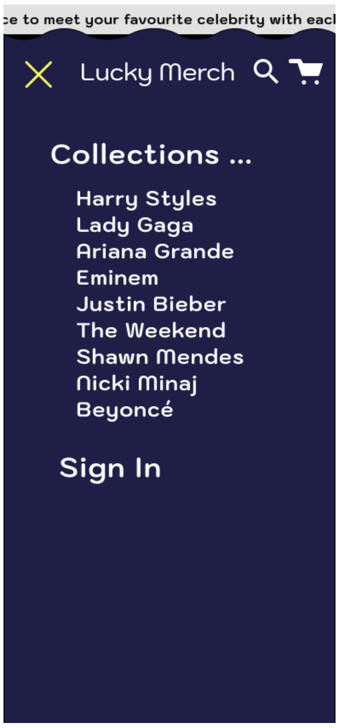


BLACK 6 OZ HEAVYWEIGHT COTTON TEE FEATURING DAWN FM GRAPHICS ON FRONT. MACHINE WASH COLD, DELICATE

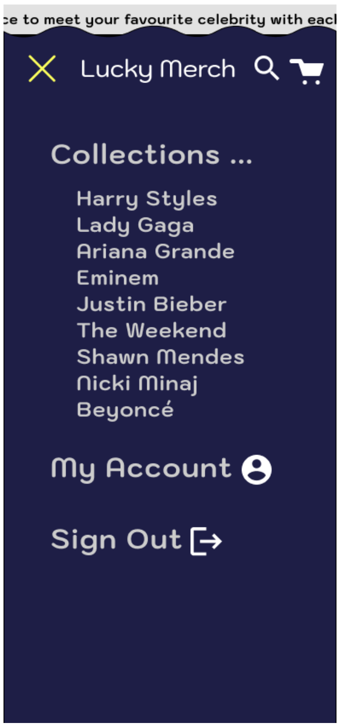
(Figure 2.3: Product Activity Out of Stock)

Upon click of a product, this page will open, showing more information about the product. Again, this page went through minor changes during development.

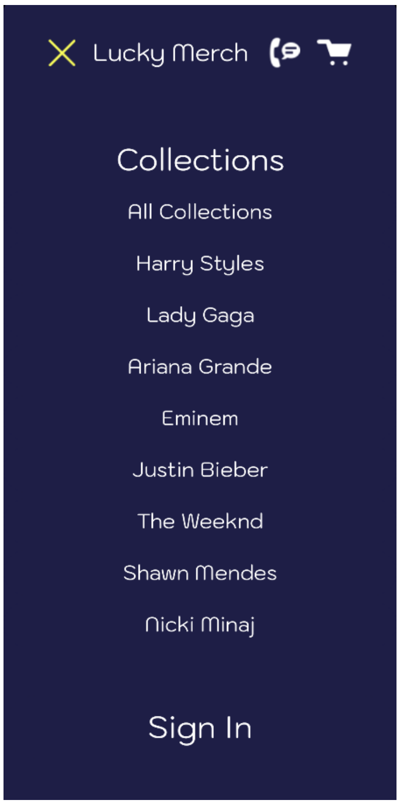
Bottom Sheet



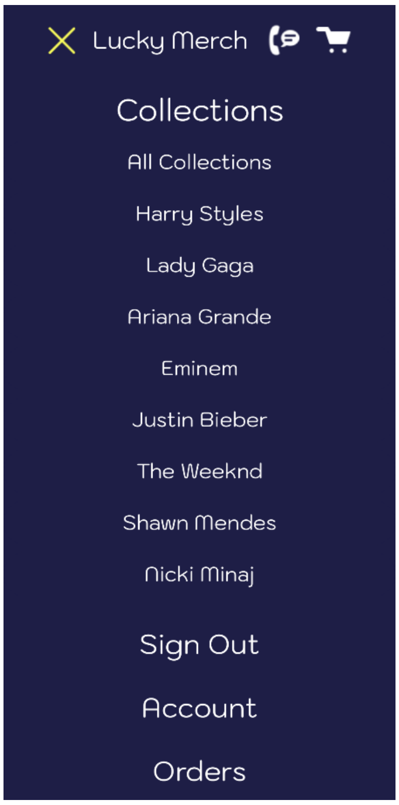
(Figure 3.1: Bottom Sheet Signed Out Wireframe)



(Figure 3.2: Bottom Sheet Signed In Wireframe)



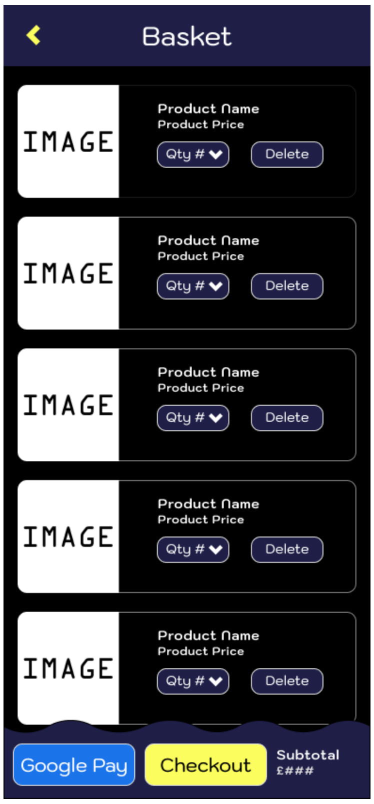
(Figure 3.3: Bottom Sheet Signed Out App)



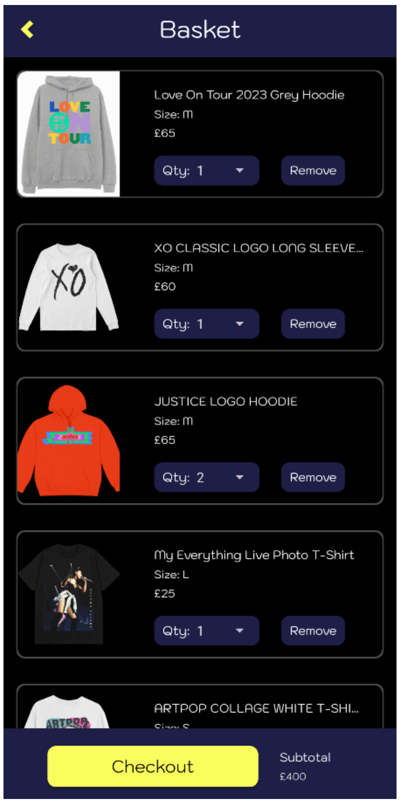
(Figure 3.4: Bottom Sheet Signed In App)

The bottom sheet appears when the user presses the menu button or swipes up on the menu bar.

Basket Activity



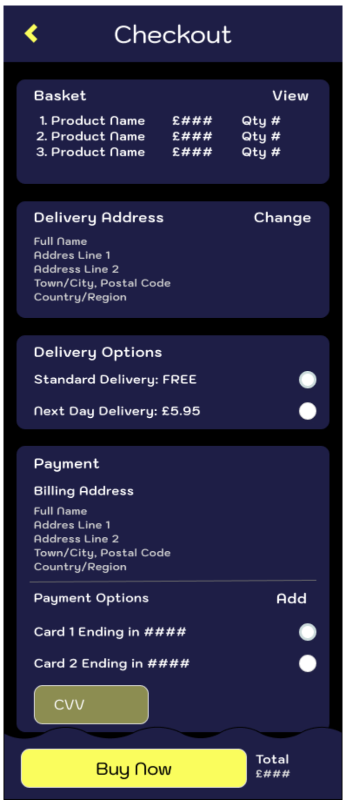
(Figure 4.1: Basket Wireframe)



(Figure 4.2: Basket Activity)

The main changes here was the removal of the google pay button and the addition of size information for each product.

Checkout Activtiy


Checkout

Basket View
1. Product Name £### Qty #
2. Product Name £### Qty #
3. Product Name £### Qty #

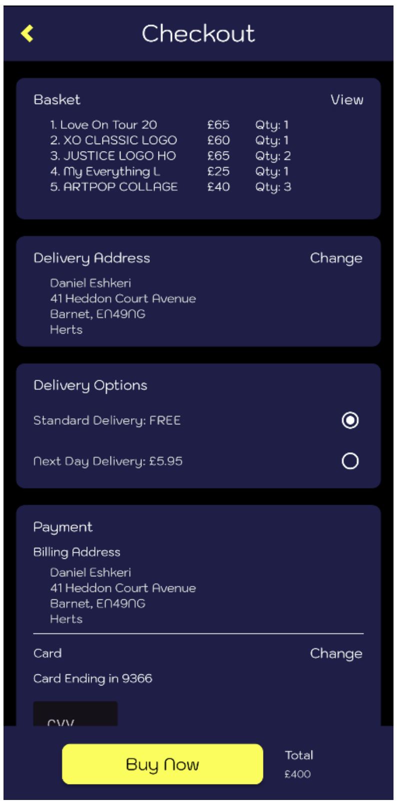
Delivery Address Change
Full Name
Addres Line 1
Address Line 2
Town/City, Postal Code
Country/Region

Delivery Options
Standard Delivery: FREE
Next Day Delivery: £5.95

Payment
Billing Address
Full Name
Addres Line 1
Address Line 2
Town/City, Postal Code
Country/Region

Payment Options Add
Card 1 Ending in ####
Card 2 Ending in ####
CVV

Buy Now Total £###


Checkout

Basket View
1. Love On Tour 20 £65 Qty: 1
2. XO CLASSIC LOGO £60 Qty: 1
3. JUSTICE LOGO HO £65 Qty: 2
4. My Everything L £25 Qty: 1
5. ARTPOP COLLAGE £40 Qty: 3

Delivery Address Change
Daniel Eshkeri
41 Heddon Court Avenue
Barnet, EN49NG
Herts

Delivery Options
Standard Delivery: FREE ●
Next Day Delivery: £5.95 ○

Payment
Billing Address
Daniel Eshkeri
41 Heddon Court Avenue
Barnet, EN49NG
Herts

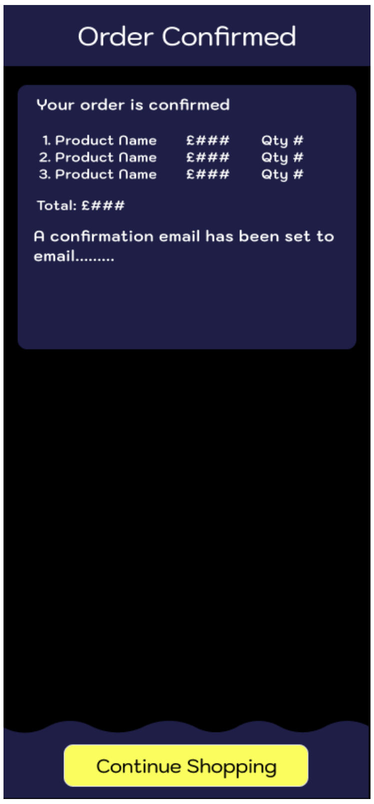
Card Change
Card Ending in 9366
CVV

Buy Now Total £400

(Figure 5.1: Checkout Wireframe) (Figure 5.2: Checkout Activity)

The user can only access the checkout activity if they are logged in, else they will be directed to the login page. Once they click buy now, if their CVV has been entered, they are taken to the order success activity.

Order Success Activity


Order Confirmed

Your order is confirmed

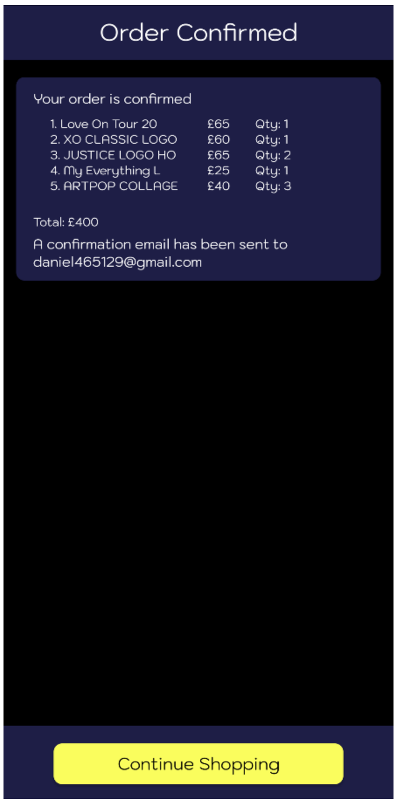
1. Product Name £### Qty #
2. Product Name £### Qty #
3. Product Name £### Qty #

Total: £###

A confirmation email has been set to email.........

Continue Shopping

(Figure 6.1: Order Success Wireframe)


Order Confirmed

Your order is confirmed

1. Love On Tour 20	£65	Qty: 1
2. XO CLASSIC LOGO	£60	Qty: 1
3. JUSTICE LOGO HO	£65	Qty: 2
4. My Everything L	£25	Qty: 1
5. ARTPOP COLLAGE	£40	Qty: 3

Total: £400

A confirmation email has been sent to daniel465129@gmail.com

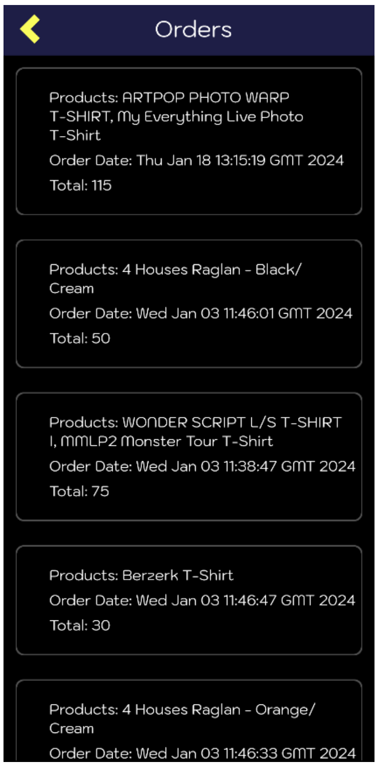
Continue Shopping

(Figure 6.2: Order Success Activity)

This page confirms their order and allows the user to go back to the home page with the continue shopping button.

Orders Activity

This was a last-minute addition, so I did not complete a wireframe for this activity. This activity shows the user all their past orders.


Orders

Products: ARTPOP PHOTO WARP T-SHIRT, My Everything Live Photo T-Shirt
Order Date: Thu Jan 18 13:15:19 GMT 2024
Total: 115

Products: 4 Houses Raglan - Black/ Cream
Order Date: Wed Jan 03 11:46:01 GMT 2024
Total: 50

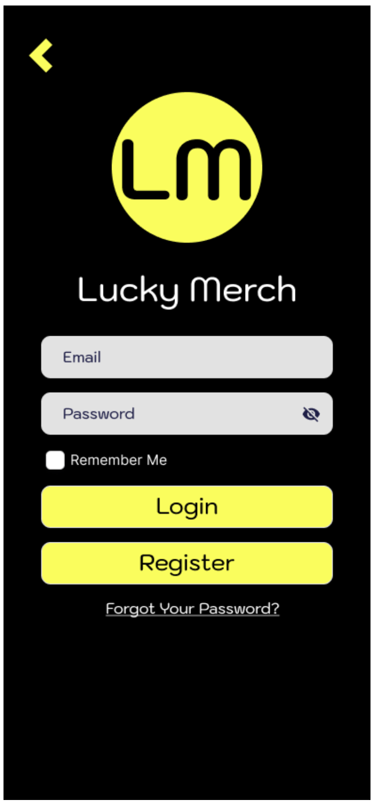
Products: WONDER SCRIPT L/S T-SHIRT I, MMLP2 Monster Tour T-Shirt
Order Date: Wed Jan 03 11:38:47 GMT 2024
Total: 75

Products: Berzerk T-Shirt
Order Date: Wed Jan 03 11:46:47 GMT 2024
Total: 30

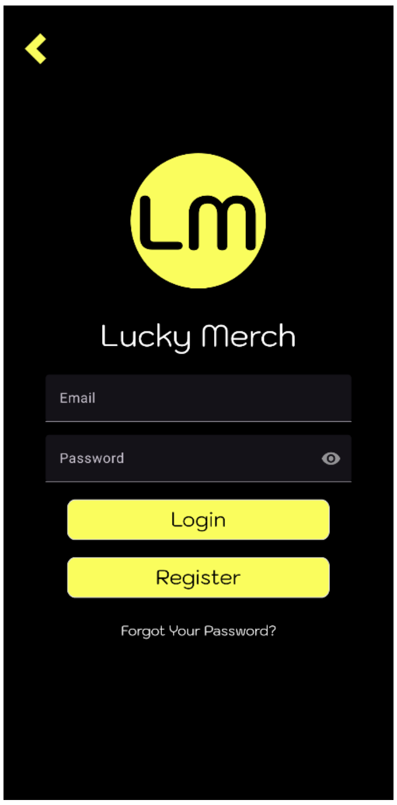
Products: 4 Houses Raglan - Orange/ Cream
Order Date: Wed Jan 03 11:46:33 GMT 2024

(Figure 7.1: Orders Activity)

Login Activity



(Figure 8.1: Login Wireframe)

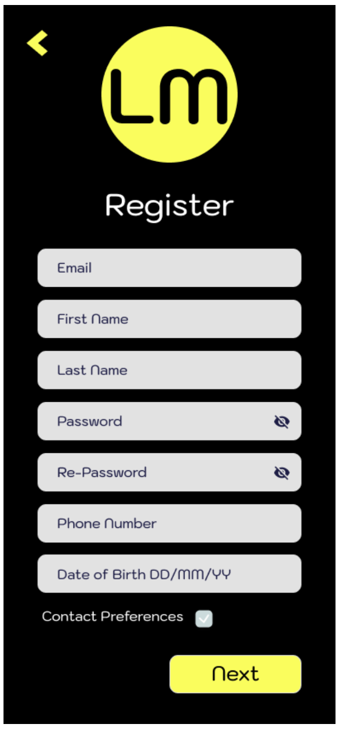


(Figure 8.2: Login Activity)

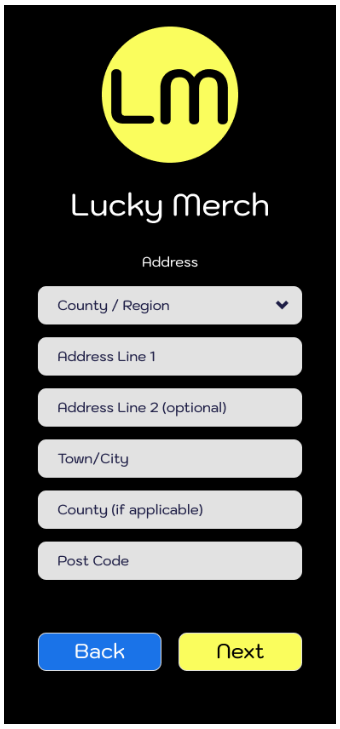
In my final application the user's credentials are saved automatically as this felt more natural and realistic to production applications.

Register Activity

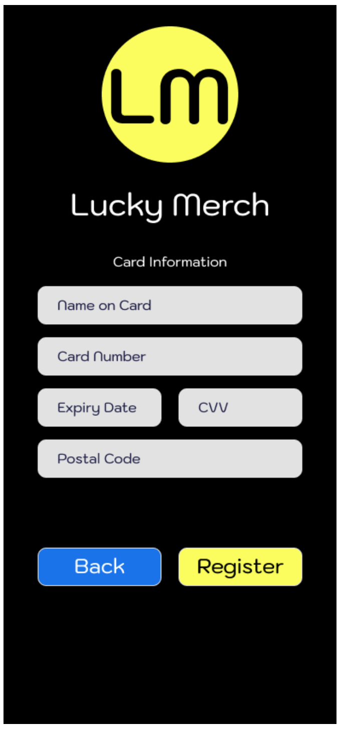
Originally, I split the register page into three different activities. However, I realised this added to the complexity, so in the application I put it into one activity.



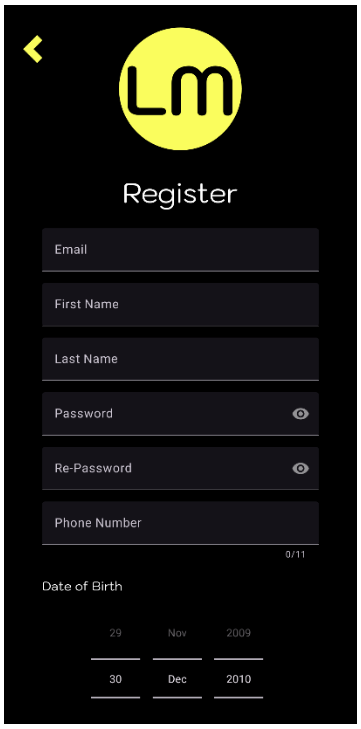
(Figure 9.1: Register Wireframe 1)



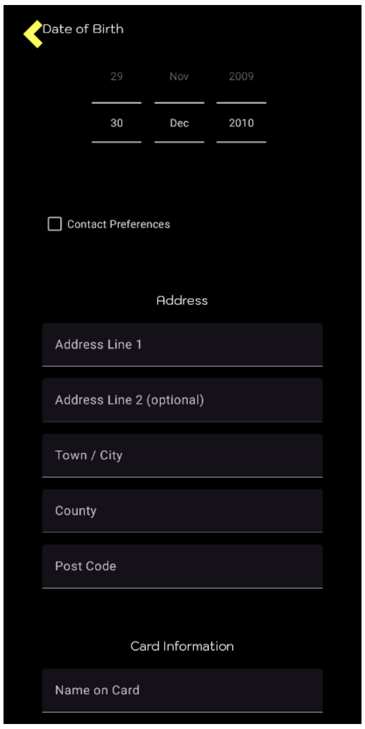
(Figure 9.2: Register Wireframe 2)



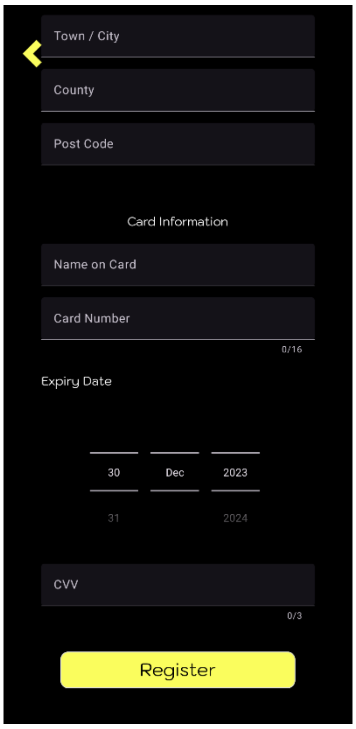
(Figure 9.3: Register Wireframe 3)



(Figure 9.4: Register Activity 1)



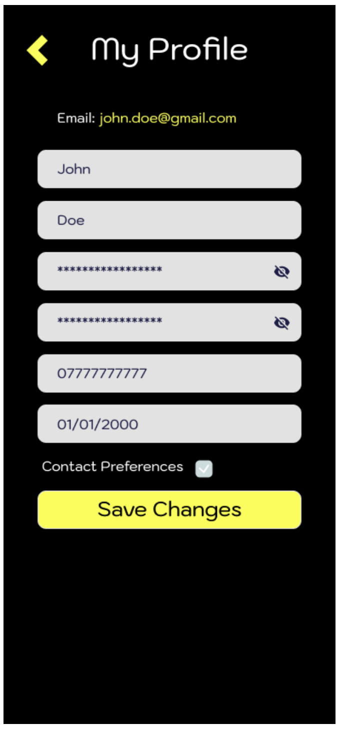
(Figure 9.5: Register Activity 2)



(Figure 9.6: Register Activity 3)

Profile Activity

When I created the profile page in my wireframing, I forgot to add the address and card information. In my actual application I did include these edit fields.



My Profile

Email: john.doe@gmail.com

John

Doe

****

****

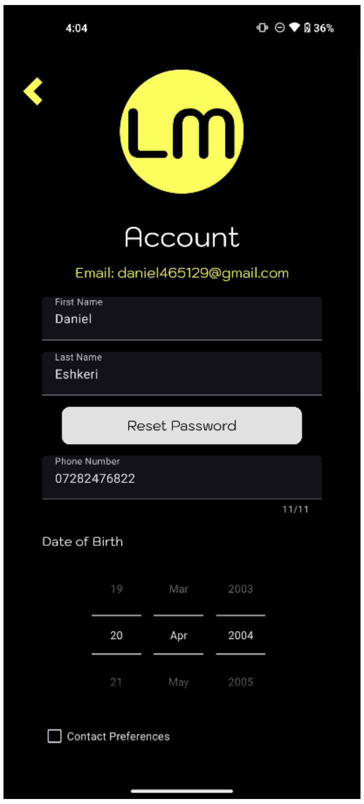
07777777777

01/01/2000

Contact Preferences ✓

Save Changes

(Figure 10.1: Profile Wireframe)



4:04 36%

Account

Email: daniel465129@gmail.com

First Name: Daniel

Last Name: Eshkeri

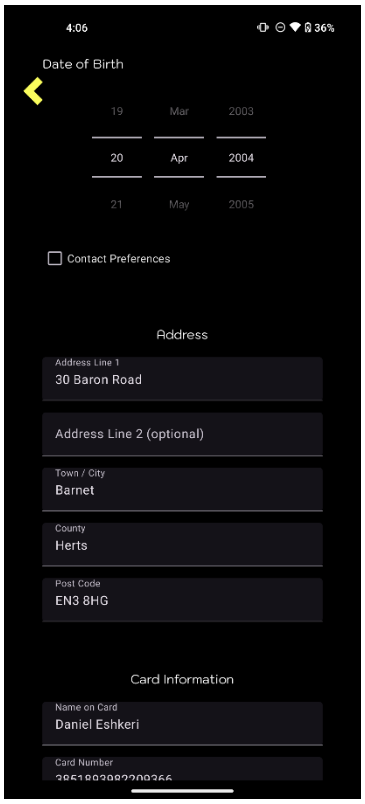
Reset Password

Phone Number: 07282476822 (11/11)

Date of Birth

19	Mar	2003
20	Apr	2004
21	May	2005

☐ Contact Preferences



4:06 36%

Date of Birth

19	Mar	2003
20	Apr	2004
21	May	2005

☐ Contact Preferences

Address

Address Line 1: 30 Baron Road

Address Line 2 (optional)

Town / City: Barnet

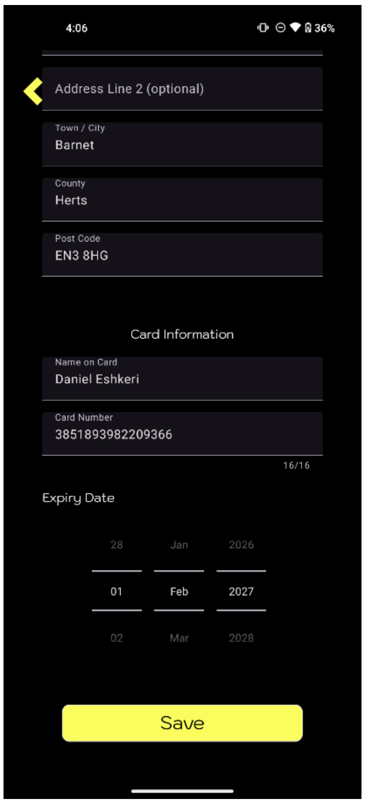
County: Herts

Post Code: EN3 8HG

Card Information

Name on Card: Daniel Eshkeri

Card Number



4:06 36%

Address Line 2 (optional)

Town / City: Barnet

County: Herts

Post Code: EN3 8HG

Card Information

Name on Card: Daniel Eshkeri

Card Number: 3851893982209366 (16/16)

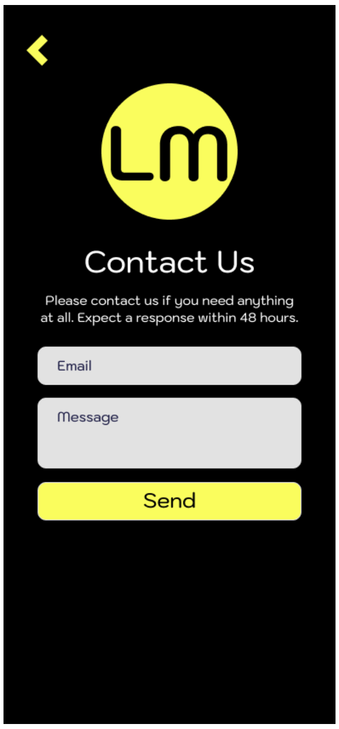
Expiry Date

28	Jan	2026
01	Feb	2027
02	Mar	2028

Save

(Figure 10.2: Profile Activity 1) (Figure 10.3: Profile Activity 2) (Figure 10.4: Profile Activity 3)

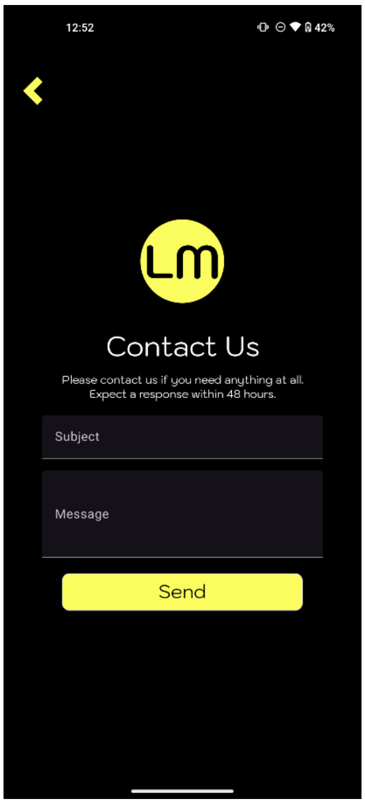
Contact Activity



Contact Us

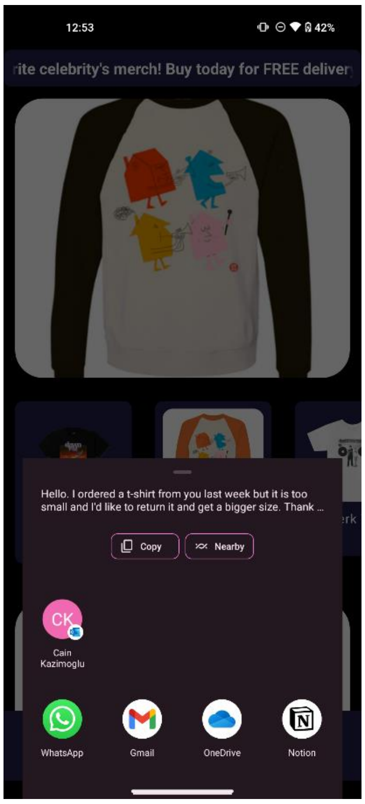
Please contact us if you need anything at all. Expect a response within 48 hours.

(Figure 11.1: Contact Wireframe)



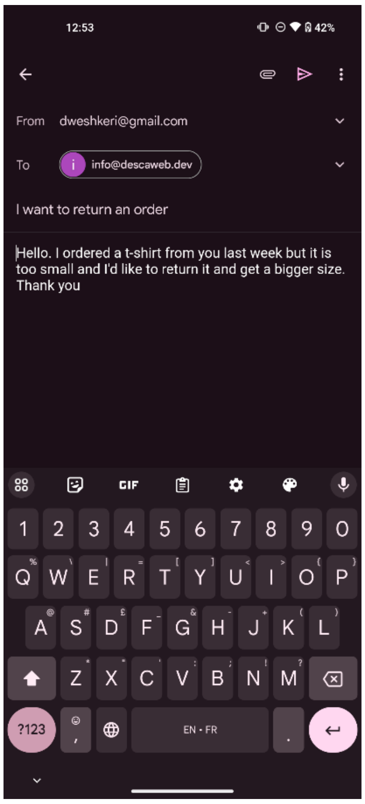
Contact Us

Please contact us if you need anything at all. Expect a response within 48 hours.



rite celebrity's merch! Buy today for FREE deliver

Hello. I ordered a t-shirt from you last week but it is too small and I'd like to return it and get a bigger size. Thank ...



From dweshkeri@gmail.com

To info@descaweb.dev

I want to return an order

Hello. I ordered a t-shirt from you last week but it is too small and I'd like to return it and get a bigger size. Thank you

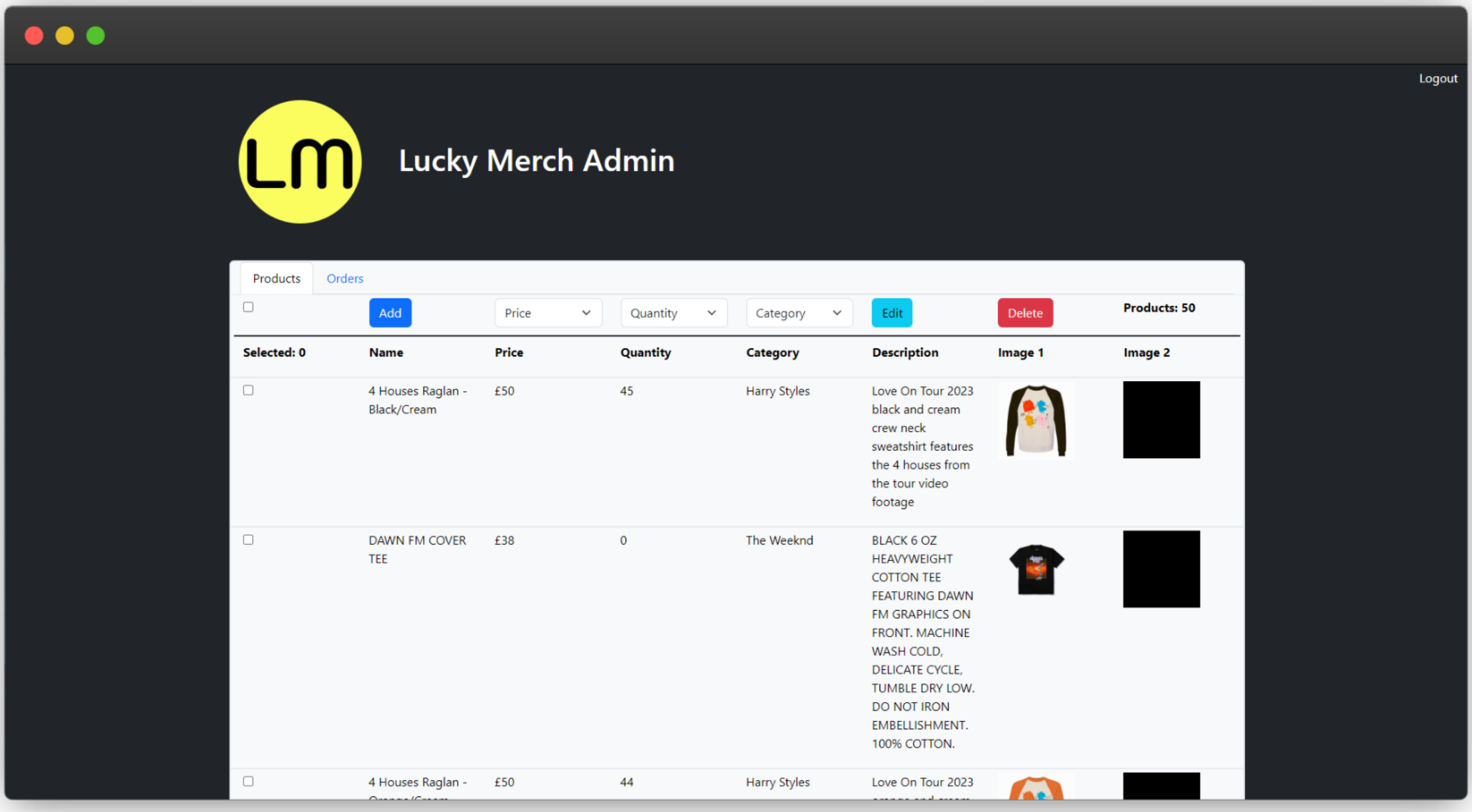
(Figure 11.2: Contact Activity) (Figure 11.3: Send pressed) (Figure 11.4: Email client pressed)

The contact activity is a convenient way for the user to send us an email with anything they might need support with.

Admin Web Panel

In addition to the application, I decided to make a web app to support the store. Users with the admin role can login and add, edit, and delete products, in addition to viewing customer orders. The web application interacts with the same backend as the android app, so they are synchronized.

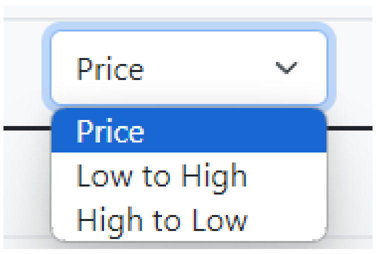
You can find the admin panel at https://lucky-merch.web.app/. Login with the email: lucky@merchin.com. Use the password: Password1@.

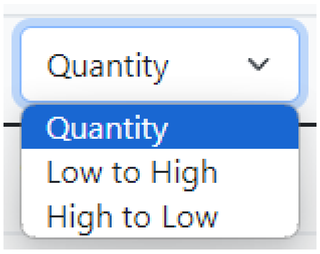


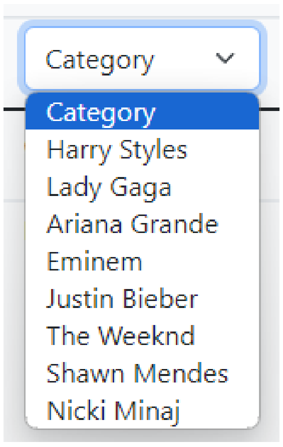
Selected: 0	Name	Price	Quantity	Category	Description	Image 1	Image 2
	4 Houses Raglan - Black/Cream	£50	45	Harry Styles	Love On Tour 2023 black and cream crew neck sweatshirt features the 4 houses from the tour video footage		
	DAWN FM COVER TEE	£38	0	The Weeknd	BLACK 6 OZ HEAVYWEIGHT COTTON TEE FEATURING DAWN FM GRAPHICS ON FRONT. MACHINE WASH COLD, DELICATE CYCLE, TUMBLE DRY LOW. DO NOT IRON EMBELLISHMENT. 100% COTTON.		
	4 Houses Raglan -	£50	44	Harry Styles	Love On Tour 2023		

(Figure 12.1: Lucky Merch Admin Panel)

I included sorting and filters such as price, quantity, and category to make management easier.



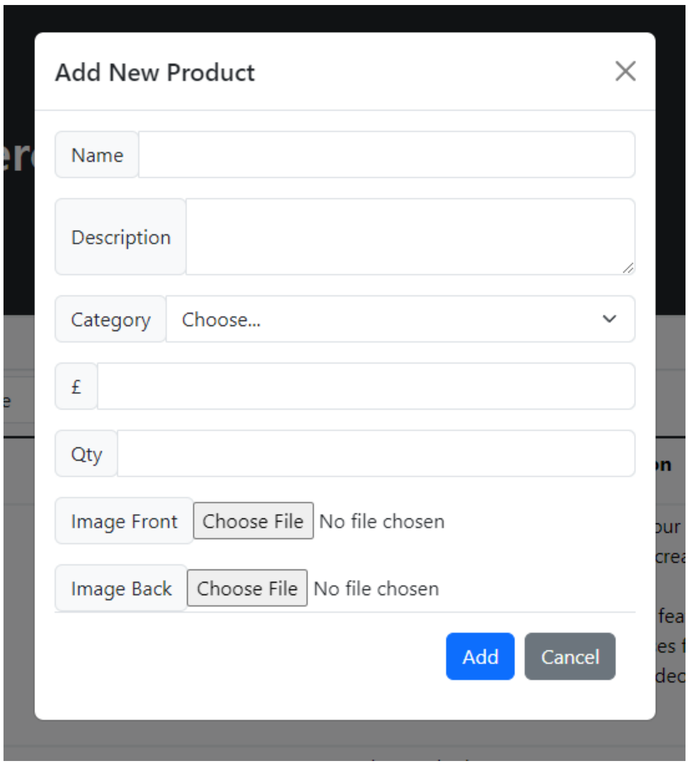




(Figure 12.2: Price Sort)

(Figure 12.3: Quantity Sort)

(Figure 12.4: Category Filter)


Add New Product

Name

Description

Category: Choose...

£

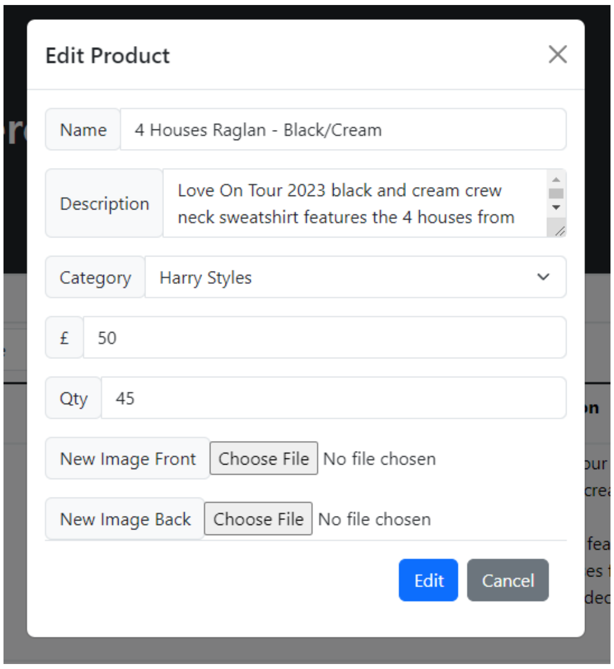
Qty

Image Front: Choose File No file chosen

Image Back: Choose File No file chosen

(Figure 12.5: Add New Product)

In this modal you can add a new product. The web app will not allow you to add a product unless every field is populated.


Edit Product

Name: 4 Houses Raglan - Black/Cream

Description: Love On Tour 2023 black and cream crew neck sweatshirt features the 4 houses from

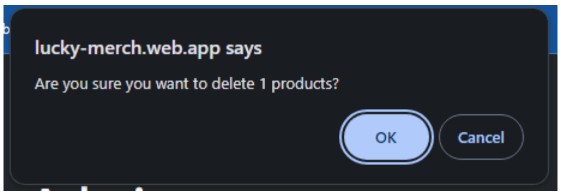
Category: Harry Styles

£: 50

Qty: 45

New Image Front: Choose File No file chosen

New Image Back: Choose File No file chosen

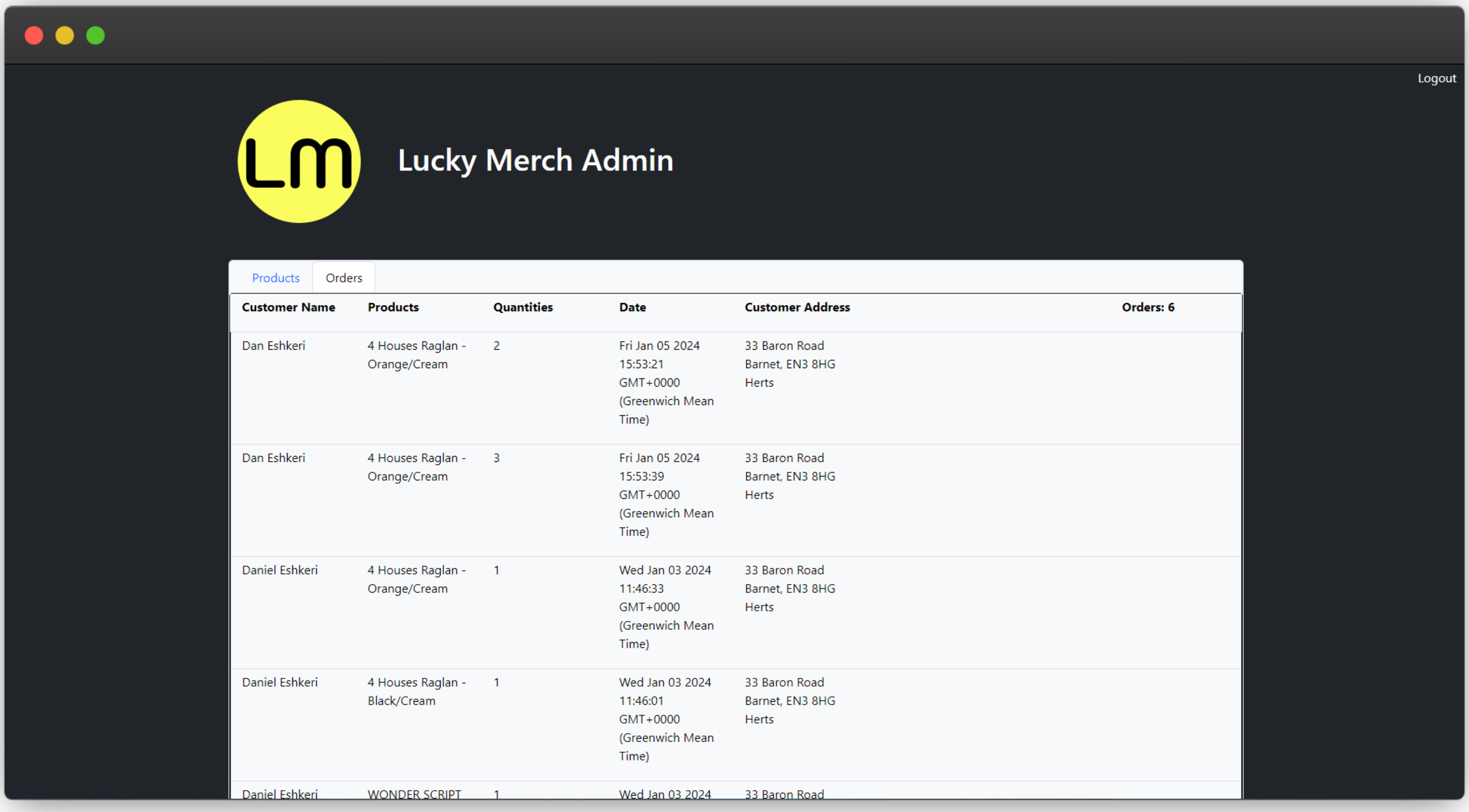

lucky-merch.web.app says

Are you sure you want to delete 1 products?

(Figure 12.6: Edit Product)

(Figure 12.7: Delete Product)

The edit modal allows you to edit each product individually and the delete button will delete all the selected products after you confirm deletion with the alert in figure 12.7.



Customer Name	Products	Quantities	Date	Customer Address	Orders: 6
Dan Eshkeri	4 Houses Raglan - Orange/Cream	2	Fri Jan 05 2024 15:53:21 GMT+0000 (Greenwich Mean Time)	33 Baron Road Barnet, EN3 8HG Herts	
Dan Eshkeri	4 Houses Raglan - Orange/Cream	3	Fri Jan 05 2024 15:53:39 GMT+0000 (Greenwich Mean Time)	33 Baron Road Barnet, EN3 8HG Herts	
Daniel Eshkeri	4 Houses Raglan - Orange/Cream	1	Wed Jan 03 2024 11:46:33 GMT+0000 (Greenwich Mean Time)	33 Baron Road Barnet, EN3 8HG Herts	
Daniel Eshkeri	4 Houses Raglan - Black/Cream	1	Wed Jan 03 2024 11:46:01 GMT+0000 (Greenwich Mean Time)	33 Baron Road Barnet, EN3 8HG Herts	
Daniel Eshkeri	WONDER SCRIPT	1	Wed Jan 03 2024	33 Baron Road	

(Figure 12.8: Orders Tab)

My justification for creating a web interface instead of building the admin panel into the mobile app is that trying to add, edit and delete products on my smartphone would be time consuming and tedious. Furthermore, it is my understanding that many industry grade applications use this method of administration.

Backend Development

Home Page

I wanted a way for the users to filter the products based on category, so I built this function to get the products from the Firebase database queried by category and create a new adapter to populate the recycler view with the products.

```
public void getCollectionCategory(String category, BottomSheetBehavior<View> sheetBehavior, LinearLayout
mBottomSheetLayout) {
    // get the products where category equals selected category
    RecyclerView rvProducts = findViewById(R.id.productRecyclerView);
    TextView titleText = mBottomSheetLayout.findViewById(R.id.lucky_merch);
    titleText.setText(category);
    Toast.makeText(this, "Collection: " + category, Toast.LENGTH_SHORT).show();
    sheetBehavior.setState(BottomSheetBehavior.STATE_COLLAPSED);
    products.clear();
    FirebaseFirestore db = FirebaseFirestore.getInstance();
    db.collection("products").whereEqualTo("category", category).get().addOnCompleteListener(task -> {
        if (task.isSuccessful()) {
            for (QueryDocumentSnapshot document : task.getResult()) {
                String id = Objects.requireNonNull(document.getId());
                String name = Objects.requireNonNull(document.getData().get("name")).toString();
                String description = Objects.requireNonNull(document.getData().get("description")).toString();
                String price = Objects.requireNonNull(document.getData().get("price")).toString();
                String imageFront = Objects.requireNonNull(document.getData().get("imageFront")).toString();
                String imageBack = Objects.requireNonNull(document.getData().get("imageBack")).toString();
                String quantity = Objects.requireNonNull(document.getData().get("quantity")).toString();

                products.add(new Product(id, name, description, price, imageFront, imageBack, quantity));

            }
            // Create adapter passing in the data
            ProductAdapter adapter = new ProductAdapter(products);
            // Attach the adapter to the recyclerview to populate items
            rvProducts.setAdapter(adapter);
            // Set layout manager to position the items
            rvProducts.setLayoutManager(new LinearLayoutManager(this));
        }
    });
}
```

(Figure 13.1: Get Collection Category Function – Main Activity)

Registration

The sign-up function takes all the entered information after it's been validated and creates a new Firebase user and adds and new user with all their information to the fire store database.

```
public void signUp(String email, String password, Map<String, Object> data) {
    mAuth.createUserWithEmailAndPassword(email, password)
            .addOnCompleteListener(this, task -> {
                if (task.isSuccessful()) {
                    // Sign up success, update UI with the signed-in user's information
                    Log.d("RegisterActivity", "createUserWithEmail:success");
                    FirebaseUser user = mAuth.getCurrentUser();
                    FirebaseFirestore db = FirebaseFirestore.getInstance();

                    db.collection("users")
                            .add(data)
                            .addOnSuccessListener(documentReference -> Log.d("TAG", "DocumentSnapshot written with
ID: " + documentReference.getId()))
                            .addOnFailureListener(e -> Log.w("TAG", "Error adding document", e));
                    finish();
                    startActivity(new Intent(RegisterActivity.this, MainActivity.class));
                } else {
                    // If sign up fails, display a message to the user.
                    Log.w("RegisterActivity", "createUserWithEmail:failure", task.getException());
                    Toast.makeText(RegisterActivity.this, "Authentication failed.",
                            Toast.LENGTH_SHORT).show();
                    finish();
                    startActivity(new Intent(RegisterActivity.this, MainActivity.class));
                }
            });
}
```

(Figure 14.1: Sign Up Function – Registration Activity)

Product Page

The add to basket button will only be active if the product is not out of stock. Furthermore, if the product is in stock and the user clicks it, the program will get the size and quantity they have selected and will save that to an SQL lite database along with the id of the product. However, it will only add the product if the product is not already in the basket. There is a small oversight here because if the user wishes to add the same product twice but in different sizes, the program will not let them. That is something that could be improved in a later version.

```java
//add to basket button
Button addToBasketBtn = findViewById(R.id.addToBasketBtn);
addToBasketBtn.setOnClickListener(v -> {
    String quantitySelected = quantitySpinner.getSelectedItem().toString();

    RadioGroup radioGroupSize = findViewById(R.id.radioGroupSize);
    int sizeID =  radioGroupSize.indexOfChild(findViewById(radioGroupSize.getCheckedRadioButtonId()));
    String size;
    switch (sizeID) {
        case 0:
            size = "XS";
            break;
        case 1:
            size = "S";
            break;
        case 2:
            size = "M";
            break;
        case 3:
            size = "L";
            break;
        case 4:
            size = "XL";
            break;
        case 5:
            size = "XXL";
            break;
        default:
            size = "M";
            break;
    }

    ArrayList<String> productIDs = new ArrayList<>();
    basketDB.openDatabaseConnection();
    for (int i = 0; i < basketDB.getRowCount(); i++) {
        for (String row : basketDB.getDataArray()) {
            String[] rowArray = row.split(":");
            productIDs.add(rowArray[1]);
        }
    }
    if (!productIDs.contains(productID)) {
        basketDB.insertRecords(productID, quantitySelected, price, size);
        basketDB.closeDatabaseConnection();
        addToBasketBtn.setText("Added to basket");
        Toast.makeText(this, "Added to basket. Qty: " + quantitySelected, Toast.LENGTH_SHORT).show();
    } else {
        addToBasketBtn.setText("Item already in basket");
        Toast.makeText(this, "Item already in basket", Toast.LENGTH_SHORT).show();
        basketDB.closeDatabaseConnection();
    }
});
```

(Figure 15.1: Add to Basket Button – Product Activity)

Checkout, Order Success Page

The function to add a successful order runs after the user has completed checkout. This function will loop through the products, put them into a HashMap, and add them to the Firebase database as references to the products and the user.

```
public void addOrderToServer(ArrayList<String> idList, ArrayList<String> quantityList, FirebaseUser currentUser, String
total) {

    FirebaseFirestore db = FirebaseFirestore.getInstance();
    if (currentUser != null) {
        email = currentUser.getEmail();
        Query docRef = db.collection("users").whereEqualTo("email", email);
        docRef.get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                String id = null;
                for (QueryDocumentSnapshot document : task.getResult()) {
                    id = Objects.requireNonNull(document.getId());
                }
                if (id != null) {
                    DocumentReference userDocRef = db.collection("users").document(id);

                    Map<String, Object> data = new HashMap<>();
                    data.put("customer", userDocRef);
                    for(int i = 0; i < idList.size(); i++) {
                        String product = "product" + (i + 1);
                        String quantity = "quantity" + (i + 1);
                        data.put(product, db.collection("products").document(idList.get(i)));
                        data.put(quantity, quantityList.get(i));
                        data.put("orderDate", Timestamp.now());
                        data.put("total", total);
                    }
                    db.collection("orders")
                            .add(data)
                            .addOnSuccessListener(documentReference -> Log.d("TAG", "DocumentSnapshot written with ID:
" + documentReference.getId()))
                            .addOnFailureListener(e -> Log.w("TAG", "Error adding document", e));
                }
            }
        });
    }
}
```

(Figure 16.1: Add Order to Server Function – Order Success Activity)

The stock update function looks at what products have been ordered and the quantity of each product and will adjust the products collection in the database to reflect those stock numbers.

```
public void stockUpdate(ArrayList<String> idList, ArrayList<String> quantityList) {
    BasketDB basketDB = new BasketDB(this);
    basketDB.clearAllRecords();

    FirebaseFirestore db = FirebaseFirestore.getInstance();
    for (int i = 0; i < idList.size(); i++) {
        int quantityToMinus = Integer.parseInt(quantityList.get(i));
        db.collection("products").document(idList.get(i)).get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                String quantity = Objects.requireNonNull(task.getResult().getData().get("quantity")).toString();
                int quantityInt = Integer.parseInt(quantity);

                db.collection("products").document(task.getResult().getId()).update("quantity", quantityInt -
quantityToMinus);
            }
        });
    }
}
```

(Figure 16.2: Stock Update Function – Order Success Activity)

Reflection and Discussion

I am incredibly happy with how this application turned out. However, there are two things I would like to change. Firstly, there is a bug where you cannot add the same product twice with different sizes. I would like to amend this in the future. Secondly, if a user tried to register with an email that has already been registered, the program will only tell them that authentication has failed and not why it failed. To improve this, I would add more specific failure message to better inform the user of the issue. Aside from those two changes, this application has exceeded my own expectations, and I am impressed with the result.

Conclusion and Future Work

Overall, I am satisfied with my project. I believe it is high quality and even something I could develop into a production application without many changes. I satisfied all the requirements including users account, baskets, stock updates, and transactions. In addition, I went beyond the requirements with the addition of an admin panel to manage products, stock, and orders. In the future, I would like to add user management to the admin panel to expand the admin's capabilities.

References

I used some code from Cain's lectures and some from the Firebase Documentation.

Google Firebase (2023) *Developer documentation for Firebase.* Available at: https://firebase.google.com/docs (Accessed: 20/12/23)

Appendix/Code

Account Activity Part 1

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.Timestamp;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.Query;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import java.util.Objects;

public class AccountActivity extends AppCompatActivity {
    private FirebaseAuth mAuth;

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

        // get all buttons, text views, edit texts, date pickers, and checkboxes
        Button saveBtn = findViewById(R.id.saveBtn);
        TextView emailTextView = findViewById(R.id.emailTextView);
        EditText fNameField = findViewById(R.id.fNameEditText);
        EditText lNameField = findViewById(R.id.lNameEditText);
        Button resetPasswordBtn = findViewById(R.id.resetPasswordBtn);
        EditText phoneField = findViewById(R.id.phoneEditText);
        DatePicker DOBField = findViewById(R.id.DOBField);
        CheckBox contractPreferences = findViewById(R.id.contractPreferences);
        EditText addressLineOneField = findViewById(R.id.addressLineOneEditText);
        EditText addressLineTwoField = findViewById(R.id.addressLineTwoEditText);
        EditText townCityField = findViewById(R.id.townCityEditText);
        EditText countyField = findViewById(R.id.countyEditText);
        EditText postCodeField = findViewById(R.id.postCodeEditText);
        EditText nameOnCardField = findViewById(R.id.nameOnCardEditText);
        EditText cardNumberField = findViewById(R.id.cardNumberEditText);
        DatePicker expiryDateField = findViewById(R.id.expiryDateField);

        mAuth = FirebaseAuth.getInstance();

        //back btn
        final Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            this.finish();
        });

        // set min and max dates for date pickers
        DOBField.setMaxDate(System.currentTimeMillis());
        expiryDateField.setMinDate(System.currentTimeMillis());

        FirebaseUser currentUser = mAuth.getCurrentUser();
```

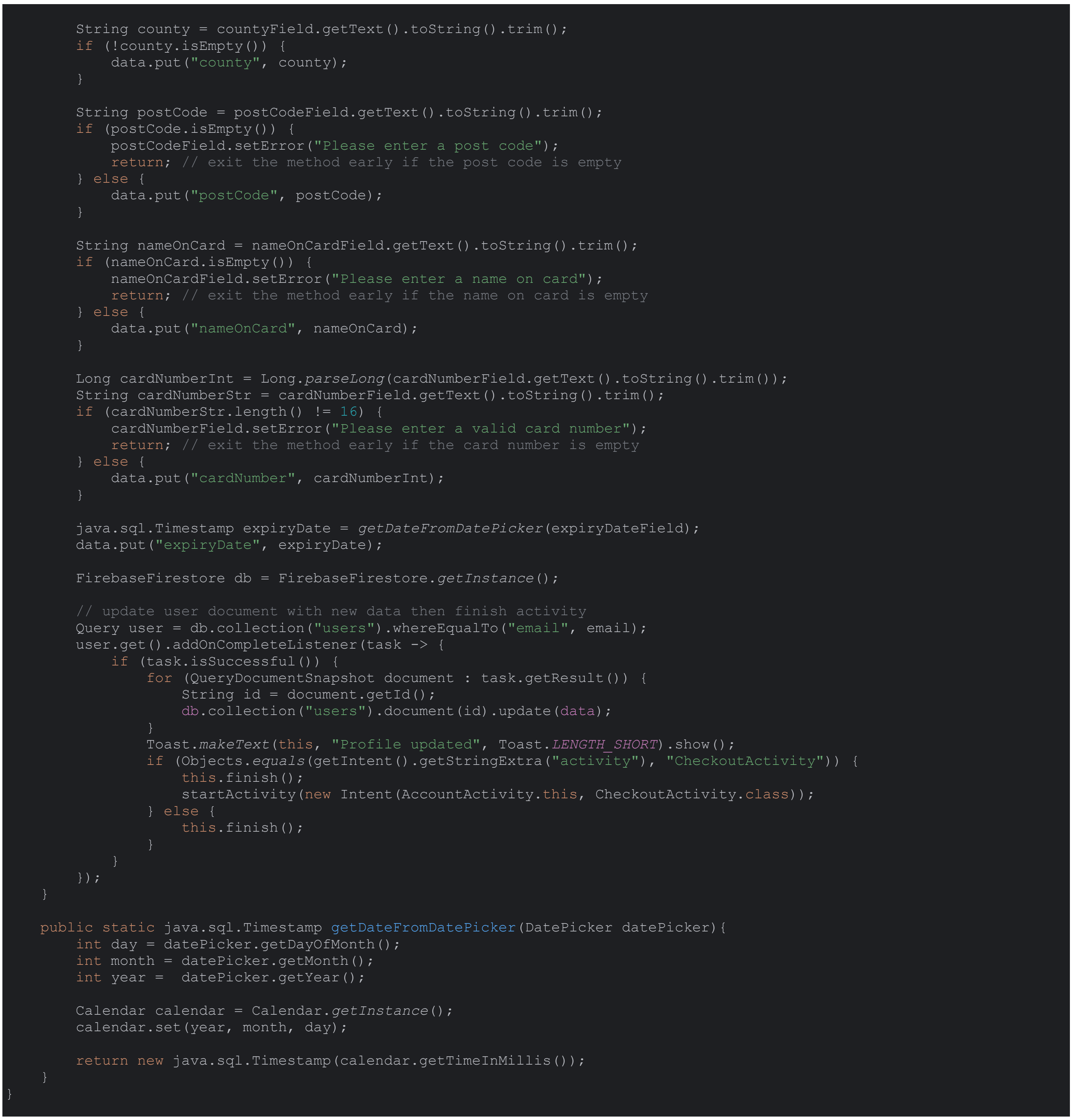
Account Activity Part 2

```
    String email;

    // get user data from firestore and set edit texts and date pickers to user data
    FirebaseFirestore db = FirebaseFirestore.getInstance();
    if (currentUser != null) {
        email = currentUser.getEmail();
        emailTextView.setText("Email: " + email);
        Query docRef = db.collection("users").whereEqualTo("email", email);
        docRef.get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                for (QueryDocumentSnapshot document : task.getResult()) {
                    String fName = Objects.requireNonNull(document.getData().get("fName")).toString();
                    String lName = Objects.requireNonNull(document.getData().get("lName")).toString();
                    String phone = Objects.requireNonNull(document.getData().get("phone")).toString();
                    Timestamp DOB = (Timestamp) Objects.requireNonNull(document.getData().get("DOB"));
                    Date DOBDate = DOB.toDate();
                    Boolean contactPreferences = (Boolean)
Objects.requireNonNull(document.getData().get("contactPreferences"));
                    String addressLineOne = Objects.requireNonNull(document.getData().get("addressLineOne")).toString();
                    String addressLineTwo = null;
                    if (document.getData().get("addressLineTwo") != null) {
                        addressLineTwo = Objects.requireNonNull(document.getData().get("addressLineTwo")).toString();
                    }
                    String townCity = Objects.requireNonNull(document.getData().get("townCity")).toString();
                    String county = Objects.requireNonNull(document.getData().get("county")).toString();
                    String postCode = Objects.requireNonNull(document.getData().get("postCode")).toString();
                    String nameOnCard = Objects.requireNonNull(document.getData().get("nameOnCard")).toString();
                    String cardNumber = Objects.requireNonNull(document.getData().get("cardNumber")).toString();
                    Timestamp expiryDate = (Timestamp) Objects.requireNonNull(document.getData().get("expiryDate"));
                    Date DateexpiryDate = expiryDate.toDate();
                    fNameField.setText(fName);
                    lNameField.setText(lName);
                    phoneField.setText(phone);
                    Calendar calendar1 = Calendar.getInstance();
                    calendar1.setTime(DOBDate);
                    int year1 = calendar1.get(Calendar.YEAR);
                    int month1 = calendar1.get(Calendar.MONTH);
                    int day1 = calendar1.get(Calendar.DAY_OF_MONTH);
                    DOBField.updateDate(year1, month1, day1);
                    contractPreferences.setChecked(contactPreferences);
                    addressLineOneField.setText(addressLineOne);
                    addressLineTwoField.setText(addressLineTwo);
                    townCityField.setText(townCity);
                    countyField.setText(county);
                    postCodeField.setText(postCode);
                    nameOnCardField.setText(nameOnCard);
                    cardNumberField.setText(cardNumber);
                    Calendar calendar2 = Calendar.getInstance();
                    calendar2.setTime(DateexpiryDate);
                    int year2 = calendar2.get(Calendar.YEAR);
                    int month2 = calendar2.get(Calendar.MONTH);
                    int day2 = calendar2.get(Calendar.DAY_OF_MONTH);
                    expiryDateField.updateDate(year2, month2, day2);

                    saveBtn.setOnClickListener(v -> {
                        // on save run update profile method
                        updateProfile();
                    });
                }
            }
        });

        // reset password btn to send user reset password email
        resetPasswordBtn.setOnClickListener(v -> {
            mAuth.sendPasswordResetEmail(email)
                    .addOnCompleteListener(task -> {
                        if (task.isSuccessful()) {
                            Log.d("FirebaseAuth", "Email sent.");
                            Toast.makeText(this, "Password Reset Email Sent to: " + email, Toast.LENGTH_LONG).show();
                            this.finish();
                            startActivity(new Intent(AccountActivity.this, MainActivity.class));
                        }
                    });
        });
    }

}
```

Account Activity Part 3

```java
public void updateProfile() {
    // get all buttons, text views, edit texts, date pickers, and checkboxes
    EditText fNameField = findViewById(R.id.fNameEditText);
    EditText lNameField = findViewById(R.id.lNameEditText);
    EditText phoneField = findViewById(R.id.phoneEditText);
    DatePicker DOBField = findViewById(R.id.DOBField);
    CheckBox contractPreferences = findViewById(R.id.contractPreferences);
    EditText addressLineOneField = findViewById(R.id.addressLineOneEditText);
    EditText addressLineTwoField = findViewById(R.id.addressLineTwoEditText);
    EditText townCityField = findViewById(R.id.townCityEditText);
    EditText countyField = findViewById(R.id.countyEditText);
    EditText postCodeField = findViewById(R.id.postCodeEditText);
    EditText nameOnCardField = findViewById(R.id.nameOnCardEditText);
    EditText cardNumberField = findViewById(R.id.cardNumberEditText);
    DatePicker expiryDateField = findViewById(R.id.expiryDateField);

    // create hashmap to save data to
    Map<String, Object> data = new HashMap<>();

    FirebaseUser currentUser = mAuth.getCurrentUser();

    String email = currentUser.getEmail();

    String fName = fNameField.getText().toString().trim();
    if (fName.isEmpty()) {
        fNameField.setError("Please enter a first name");
        return; // exit the method early if the first name is empty
    } else {
        data.put("fName", fName);
    }

    String lName = lNameField.getText().toString().trim();
    if (lName.isEmpty()) {
        lNameField.setError("Please enter a last name");
        return; // exit the method early if the last name is empty
    } else {
        data.put("lName", lName);
    }

    String phoneStr = phoneField.getText().toString().trim();
    if (phoneStr.length() != 11) {
        phoneField.setError("Please enter a valid phone number");
        return; // exit the method early if the phone number is empty
    } else {
        data.put("phone", phoneStr);
    }

    java.sql.Timestamp DOB = getDateFromDatePicker(DOBField);
    data.put("DOB", DOB);

    Boolean contractPreferencesBool = contractPreferences.isChecked();
    data.put("contactPreferences", contractPreferencesBool);

    String addressLineOne = addressLineOneField.getText().toString().trim();
    if (addressLineOne.isEmpty()) {
        addressLineOneField.setError("Please enter an address");
        return; // exit the method early if the address is empty
    } else {
        data.put("addressLineOne", addressLineOne);
    }

    String addressLineTwo = addressLineTwoField.getText().toString().trim();
    if (!addressLineTwo.isEmpty()) {
        data.put("addressLineTwo", addressLineTwo);
    }

    String townCity = townCityField.getText().toString().trim();
    if (townCity.isEmpty()) {
        townCityField.setError("Please enter a town/city");
        return; // exit the method early if the town/city is empty
    } else {
        data.put("townCity", townCity);
    }
```

Account Activity Part 4



```
        String county = countyField.getText().toString().trim();
        if (!county.isEmpty()) {
            data.put("county", county);
        }

        String postCode = postCodeField.getText().toString().trim();
        if (postCode.isEmpty()) {
            postCodeField.setError("Please enter a post code");
            return; // exit the method early if the post code is empty
        } else {
            data.put("postCode", postCode);
        }

        String nameOnCard = nameOnCardField.getText().toString().trim();
        if (nameOnCard.isEmpty()) {
            nameOnCardField.setError("Please enter a name on card");
            return; // exit the method early if the name on card is empty
        } else {
            data.put("nameOnCard", nameOnCard);
        }

        Long cardNumberInt = Long.parseLong(cardNumberField.getText().toString().trim());
        String cardNumberStr = cardNumberField.getText().toString().trim();
        if (cardNumberStr.length() != 16) {
            cardNumberField.setError("Please enter a valid card number");
            return; // exit the method early if the card number is empty
        } else {
            data.put("cardNumber", cardNumberInt);
        }

        java.sql.Timestamp expiryDate = getDateFromDatePicker(expiryDateField);
        data.put("expiryDate", expiryDate);

        FirebaseFirestore db = FirebaseFirestore.getInstance();

        // update user document with new data then finish activity
        Query user = db.collection("users").whereEqualTo("email", email);
        user.get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                for (QueryDocumentSnapshot document : task.getResult()) {
                    String id = document.getId();
                    db.collection("users").document(id).update(data);
                }
                Toast.makeText(this, "Profile updated", Toast.LENGTH_SHORT).show();
                if (Objects.equals(getIntent().getStringExtra("activity"), "CheckoutActivity")) {
                    this.finish();
                    startActivity(new Intent(AccountActivity.this, CheckoutActivity.class));
                } else {
                    this.finish();
                }
            }
        });
    }

    public static java.sql.Timestamp getDateFromDatePicker(DatePicker datePicker){
        int day = datePicker.getDayOfMonth();
        int month = datePicker.getMonth();
        int year =  datePicker.getYear();

        Calendar calendar = Calendar.getInstance();
        calendar.set(year, month, day);

        return new java.sql.Timestamp(calendar.getTimeInMillis());
    }
}
```

Basket Activity

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

import java.util.ArrayList;

public class BasketActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;

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

        mAuth = FirebaseAuth.getInstance();

        final BasketDB basketDB = new BasketDB(this);

        // get array of basket items
        ArrayList<String> basketList;
        basketList = basketDB.getDataArray();

        String activity = getIntent().getStringExtra("activity");

        //back btn
        final Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            this.finish();
        });

        // Lookup the recyclerview in activity layout
        RecyclerView rvBasket = findViewById(R.id.basketRecyclerView);
        // Create adapter passing in the data
        BasketAdapter adapter = new BasketAdapter(basketList);
        // Attach the adapter to the recyclerview to populate items
        rvBasket.setAdapter(adapter);
        // Set layout manager to position the items
        rvBasket.setLayoutManager(new LinearLayoutManager(this));
    }

    public void onStart() {
        super.onStart();

        // Check if user is signed in (non-null) and update checkout button accordingly
        FirebaseUser currentUser = mAuth.getCurrentUser();
        if(currentUser != null){
            BasketDB  basketDB = new BasketDB(this);
            if (basketDB.getRowCount() != 0) {
                final Button checkoutBtn = findViewById(R.id.checkoutBtn);
                checkoutBtn.setOnClickListener(v -> {
                    startActivity(new Intent(BasketActivity.this, CheckoutActivity.class));
                });
            } else {
                final Button checkoutBtn = findViewById(R.id.checkoutBtn);
                checkoutBtn.setOnClickListener(v -> {
                    Toast.makeText(this, "Your basket is empty", Toast.LENGTH_SHORT).show();
                });
            }
        } else {
            final Button checkoutBtn = findViewById(R.id.checkoutBtn);
            checkoutBtn.setOnClickListener(v -> {
                startActivity(new Intent(BasketActivity.this, LoginActivity.class));
            });
        }
    }
}
```

Basket Adapter Part 1

```java
package com.example.luckymerch;

import android.app.Activity;
import android.content.Context;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.Spinner;
import android.widget.TextView;

import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.recyclerview.widget.RecyclerView;

import com.bumptech.glide.Glide;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.List;
import java.util.Objects;

public class BasketAdapter extends RecyclerView.Adapter<BasketAdapter.ViewHolder> {
    // Provide a direct reference to each of the views within a data item
    // Used to cache the views within the item layout for fast access
    public class ViewHolder extends RecyclerView.ViewHolder {
        // Your holder should contain a member variable
        // for any view that will be set as you render a row
        public TextView nameTextView, priceTextView, sizeTextView;
        public ImageView imageView;
        public Spinner quantitySpinner;
        public Button deleteBtn;


        // We also create a constructor that accepts the entire item row
        // and does the view lookups to find each subview
        public ViewHolder(View itemView) {
            // Stores the itemView in a public final member variable that can be used
            // to access the context from any ViewHolder instance.
            super(itemView);

            nameTextView = itemView.findViewById(R.id.productNameView);
            priceTextView = itemView.findViewById(R.id.productPriceView);
            sizeTextView = itemView.findViewById(R.id.productSizeView);
            imageView = itemView.findViewById(R.id.imageView);
            quantitySpinner = itemView.findViewById(R.id.quantitySpinner);
            deleteBtn = itemView.findViewById(R.id.deleteBtn);
        }
    }

    // Store a variable
    private List<String> mBasket;

    // Pass in the array into the constructor
    public BasketAdapter(List<String> basket) {
        mBasket = basket;
    }

    @Override
    public BasketAdapter.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        Context context = parent.getContext();
        LayoutInflater inflater = LayoutInflater.from(context);

        // Inflate the custom layout
        View basketView = inflater.inflate(R.layout.basket_recycler_layout, parent, false);

        // Return a new holder instance
        BasketAdapter.ViewHolder viewHolder = new BasketAdapter.ViewHolder(basketView);
        return viewHolder;
    }
```

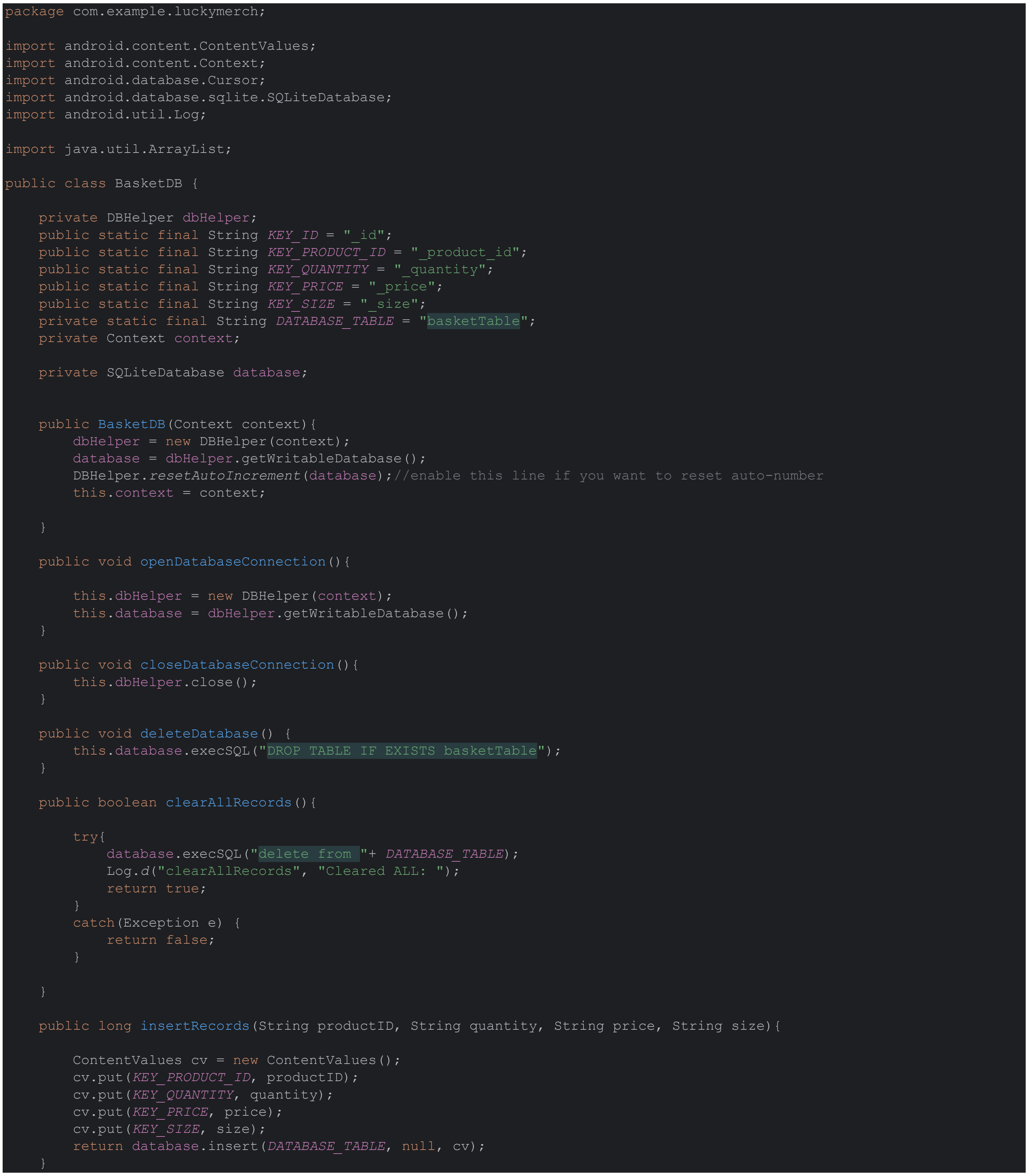
Basket Adapter Part 2

```
@Override
public void onBindViewHolder(BasketAdapter.ViewHolder holder, int position) {
    String item = mBasket.get(position);

    String[] itemArray = item.split(":");

    String rowID = itemArray[0];
    String productID = itemArray[1];
    String quantitySelected = itemArray[2];
    String sizeSelected = itemArray[4];

    //get product documents
    FirebaseFirestore db = FirebaseFirestore.getInstance();

    DocumentReference docRef = db.collection("products").document(productID);
    docRef.get().addOnCompleteListener(task -> {
        if (task.isSuccessful()) {
            DocumentSnapshot document = task.getResult();
            if (document.exists()) {
                Log.d("BasketAdapterMsg", "DocumentSnapshot data: " + document.getData());
                String name = Objects.requireNonNull(document.getData().get("name")).toString();
                String price = Objects.requireNonNull(document.getData().get("price")).toString();
                String imageFront = Objects.requireNonNull(document.getData().get("imageFront")).toString();
                String quantity = Objects.requireNonNull(document.getData().get("quantity")).toString();

                ImageView imageView = holder.imageView;
                Glide.with(imageView.getContext()).load(imageFront).into(imageView);

                TextView nameView = holder.nameTextView;
                nameView.setText(name);

                TextView priceView = holder.priceTextView;
                priceView.setText("£" + price);

                TextView sizeView = holder.sizeTextView;
                sizeView.setText("Size: " + sizeSelected);

                String[] quantityArray = new String[Integer.parseInt(quantity)];
                for (int i = 0; i < Integer.parseInt(quantity); i++) {
                    quantityArray[i] = String.valueOf(i + 1);
                }

                ArrayAdapter<String> dataAdapter = new ArrayAdapter<>(holder.itemView.getContext(),
android.R.layout.simple_spinner_item, quantityArray);
                dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
                holder.quantitySpinner.setAdapter(dataAdapter);
                holder.quantitySpinner.setSelection(Integer.parseInt(quantitySelected) - 1);
                // quantity spinner change
                holder.quantitySpinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
                    @Override
                    public void onItemSelected(AdapterView<?> parent, View view, int position, long id) {
                        final BasketDB basketDB = new BasketDB(holder.itemView.getContext());
                        basketDB.updateRecord(rowID, productID, String.valueOf(position + 1), price, sizeSelected);
                        mBasket = basketDB.getDataArray();

                        Activity activity = (Activity) holder.itemView.getContext();
                        ConstraintLayout activity_basket = activity.findViewById(R.id.activityBasket);

                        TextView basketTotal = activity_basket.findViewById(R.id.totalTextView);

                        basketTotal.setText("£" + basketDB.getTotal());
                    }

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

Basket Adapter Part 3

```java
                    // delete btn
                    holder.deleteBtn.setOnClickListener(v -> {
                        final BasketDB basketDB = new BasketDB(holder.itemView.getContext());
                        Log.d("holder.deleteBtn.setOnClickListener", "ROWID: " + rowID);
                        Log.d("holder.deleteBtn.setOnClickListener", "POSITION: " + position);
                        basketDB.deleteRecord(rowID);
                        mBasket.remove(position);
                        notifyItemRemoved(position);
                        notifyItemRangeChanged(0, basketDB.getRowCount());
                        if (mBasket.isEmpty()) {
                            Activity activity = (Activity) holder.itemView.getContext();
                            ConstraintLayout activity_basket = activity.findViewById(R.id.activityBasket);

                            TextView basketTotal = activity_basket.findViewById(R.id.totalTextView);

                            basketTotal.setText("£0");
                        }
                    });

                } else {
                    Log.d("BasketAdapterMsg", "No such document");
                }
            } else {
                Log.d("BasketAdapterMsg", "get failed with ", task.getException());
            }
        });
    }

    @Override
    public int getItemCount() {
        return mBasket.size();
    }
}
```

BasketDB Part 1



```java
package com.example.luckymerch;

import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.util.Log;

import java.util.ArrayList;

public class BasketDB {

    private DBHelper dbHelper;
    public static final String KEY_ID = "_id";
    public static final String KEY_PRODUCT_ID = "_product_id";
    public static final String KEY_QUANTITY = "_quantity";
    public static final String KEY_PRICE = "_price";
    public static final String KEY_SIZE = "_size";
    private static final String DATABASE_TABLE = "basketTable";
    private Context context;

    private SQLiteDatabase database;


    public BasketDB(Context context){
        dbHelper = new DBHelper(context);
        database = dbHelper.getWritableDatabase();
        DBHelper.resetAutoIncrement(database);//enable this line if you want to reset auto-number
        this.context = context;

    }

    public void openDatabaseConnection(){

        this.dbHelper = new DBHelper(context);
        this.database = dbHelper.getWritableDatabase();
    }

    public void closeDatabaseConnection(){
        this.dbHelper.close();
    }

    public void deleteDatabase() {
        this.database.execSQL("DROP TABLE IF EXISTS basketTable");
    }

    public boolean clearAllRecords(){

        try{
            database.execSQL("delete from "+ DATABASE_TABLE);
            Log.d("clearAllRecords", "Cleared ALL: ");
            return true;
        }
        catch(Exception e) {
            return false;
        }

    }

    public long insertRecords(String productID, String quantity, String price, String size){

        ContentValues cv = new ContentValues();
        cv.put(KEY_PRODUCT_ID, productID);
        cv.put(KEY_QUANTITY, quantity);
        cv.put(KEY_PRICE, price);
        cv.put(KEY_SIZE, size);
        return database.insert(DATABASE_TABLE, null, cv);
    }
```

BasketDB Part 2

```java
public String getData(){

    String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};

    // create a cursor object to move the cursor and return data

    Cursor c = database.query(DATABASE_TABLE, columns, null, null, null, null, null );

    String result = "";

    int columnID = c.getColumnIndex(KEY_ID);
    int columnProductID = c.getColumnIndex(KEY_PRODUCT_ID);
    int columnQuantity = c.getColumnIndex(KEY_QUANTITY);
    int columnPrice = c.getColumnIndex(KEY_PRICE);
    int columnSize = c.getColumnIndex(KEY_SIZE);

    for (c.moveToFirst();!c.isAfterLast();c.moveToNext()){

        result+= "\n"+ c.getString(columnID)+":"+
c.getString(columnProductID)+":"+c.getString(columnQuantity)+":"+c.getString(columnPrice)+":"+c.getString(columnSize);

    }

    c.close();

    return result;

}

public ArrayList<String> getDataArray() {
    String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};
    Cursor c = database.query(DATABASE_TABLE, columns, null, null, null, null, null );
    ArrayList<String> data = new ArrayList<String>();
    if (c != null) {
        c.moveToFirst();
        while (!c.isAfterLast()) {
            String row = c.getString(0) + ":" + c.getString(1) + ":" + c.getString(2) + ":" + c.getString(3) + ":" +
c.getString(4); // return product id and quantity and price and size
            data.add(row);
            c.moveToNext();
        }
        c.close();
        return data;
    } else {
        return null;
    }
}

public int getRowCount() {
    // return row count
    String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};
    Cursor c = database.query(DATABASE_TABLE, columns, null, null, null, null, null );
    int count = c.getCount();
    c.close();
    return count;
}

public long deleteRecord (String rowId)
{
    return database.delete(DATABASE_TABLE, KEY_ID + "=?",new String[] {rowId} );
}
```

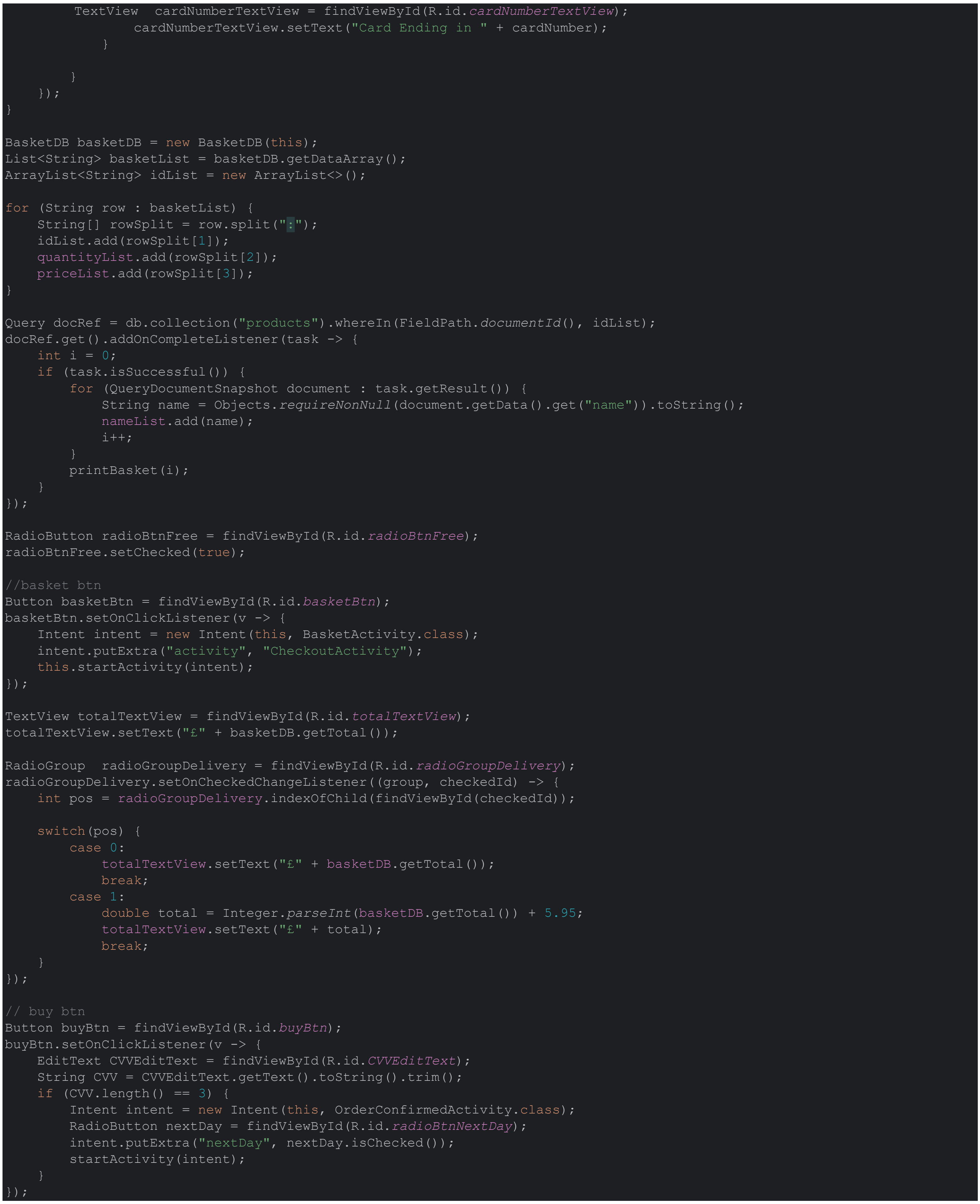
BasketDB Part 3

```java
    public boolean deleteLastRecord(){

        String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};
        String lastID="";

        Cursor c = database.query(DATABASE_TABLE, columns, null,
                null, null, null, null );

        int columnID = c.getColumnIndex(KEY_ID);

        for (c.moveToFirst();!c.isAfterLast();c.moveToNext())
            lastID = c.getString(columnID);


        int value = database.delete(DATABASE_TABLE, KEY_ID + "=?",new String[] {lastID});

        if (value == 1)
            return true;
        else
            return false;
    }

    public boolean updateLastRecord (String productID, String quantity, String price, String size){
        String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};
        String lastID="";
        Cursor c = database.query(DATABASE_TABLE, columns, null,
                null, null, null, null );
        int columnID = c.getColumnIndex(KEY_ID);
        for (c.moveToFirst();!c.isAfterLast();c.moveToNext())
            lastID = c.getString(columnID);
        ContentValues cv = new ContentValues();
        cv.put(KEY_PRODUCT_ID, productID);
        cv.put(KEY_QUANTITY, quantity);
        cv.put(KEY_PRICE, price);
        cv.put(KEY_SIZE, size);
        int value = database.update(DATABASE_TABLE, cv, KEY_ID + "=?",new String[] {lastID});
        if (value == 1)
            return true;
        else
            return false;
    }

    public long updateRecord (String rowId, String productID, String quantity, String price, String size){
        ContentValues cv = new ContentValues();
        cv.put(KEY_PRODUCT_ID, productID);
        cv.put(KEY_QUANTITY, quantity);
        cv.put(KEY_PRICE, price);
        cv.put(KEY_SIZE, size);
        return database.update(DATABASE_TABLE, cv, KEY_ID + "=?", new String[]{rowId});
    }

    public String getTotal() {
        String [] columns = new String[]{KEY_ID, KEY_PRODUCT_ID, KEY_QUANTITY, KEY_PRICE, KEY_SIZE};
        Cursor c = database.query(DATABASE_TABLE, columns, null, null, null, null, null );
        ArrayList<String> data = new ArrayList<String>();

        int total = 0;
        if (c != null) {
            c.moveToFirst();
            while (!c.isAfterLast()) {

                String quantity = c.getString(2);
                String price = c.getString(3);

                total += Integer.parseInt(quantity) * Integer.parseInt(price);
                c.moveToNext();
            }
            c.close();
            return String.valueOf(total);
        } else {
            return null;
        }
    }

}
```

Checkout Activity Part 1

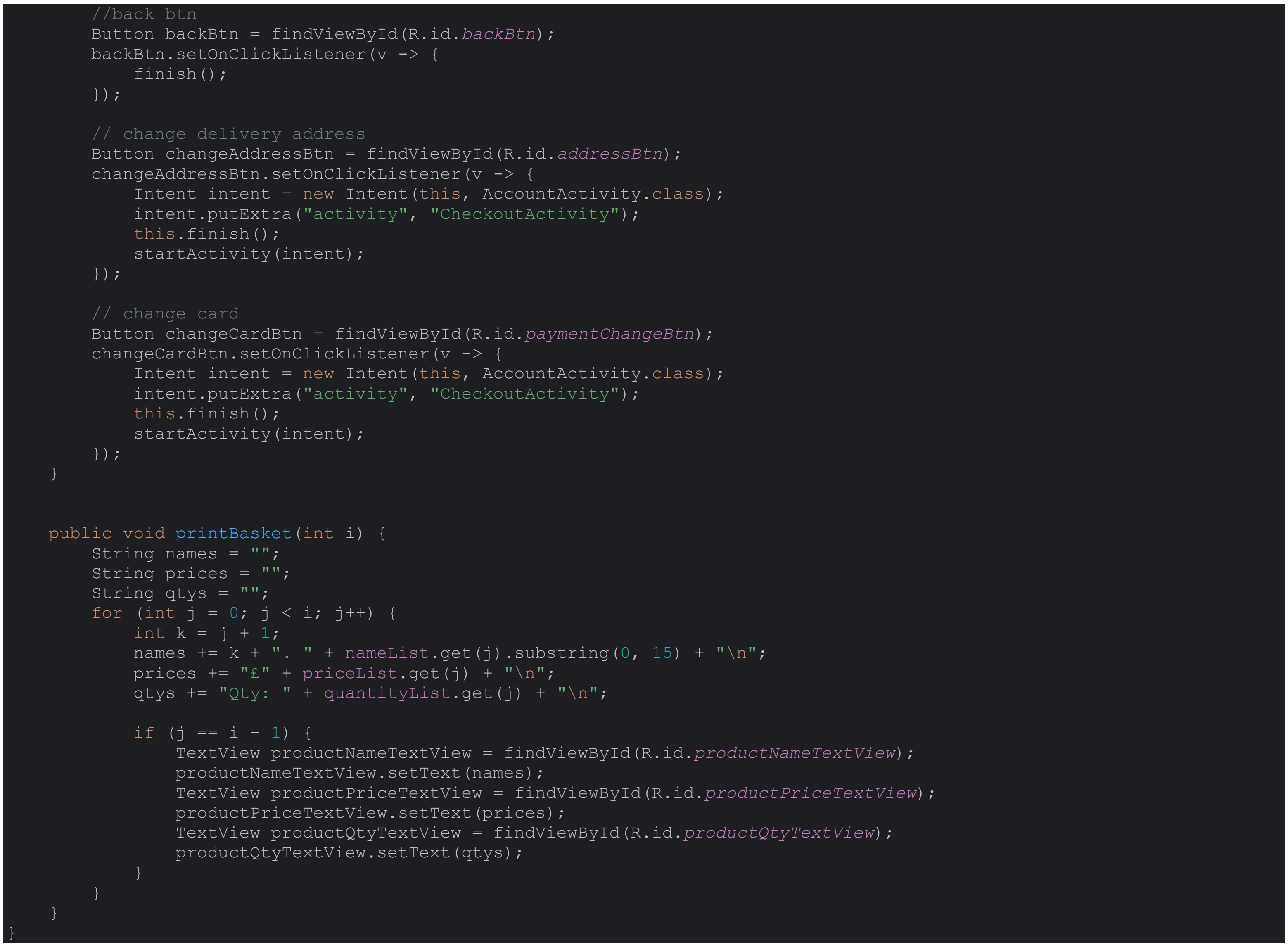
```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.FieldPath;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.Query;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import java.util.ArrayList;
import java.util.List;
import java.util.Objects;

public class CheckoutActivity extends AppCompatActivity {
    private FirebaseAuth mAuth;
    ArrayList<String> quantityList = new ArrayList<>();
    ArrayList<String> priceList = new ArrayList<>();
    ArrayList<String> nameList = new ArrayList<>();
    String email;
    String addressFull;

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

        FirebaseFirestore db = FirebaseFirestore.getInstance();

        mAuth = FirebaseAuth.getInstance();
        FirebaseUser currentUser = mAuth.getCurrentUser();

        if (currentUser != null) {
            email = currentUser.getEmail();
            Query docRef = db.collection("users").whereEqualTo("email", email);
            docRef.get().addOnCompleteListener(task -> {
                if (task.isSuccessful()) {
                    for (QueryDocumentSnapshot document : task.getResult()) {
                        String fName = Objects.requireNonNull(document.getData().get("fName")).toString();
                        String lName = Objects.requireNonNull(document.getData().get("lName")).toString();
                        String addressLineOne =
Objects.requireNonNull(document.getData().get("addressLineOne")).toString();
                        String addressLineTwo = null;
                        if (document.getData().get("addressLineTwo") != null) {
                            addressLineTwo =
Objects.requireNonNull(document.getData().get("addressLineTwo")).toString();
                        }
                        String townCity = Objects.requireNonNull(document.getData().get("townCity")).toString();
                        String postCode = Objects.requireNonNull(document.getData().get("postCode")).toString();
                        String county = Objects.requireNonNull(document.getData().get("county")).toString();
                        String cardNumber =
Objects.requireNonNull(document.getData().get("cardNumber")).toString().substring(12);

                        addressFull = fName + " " + lName + "\n" + addressLineOne + "\n";
                        if (addressLineTwo != null) {
                            addressFull += addressLineTwo + "\n";
                        }
                        addressFull += townCity + ", " + postCode + "\n" + county;

                        TextView addressTextView = findViewById(R.id.deliveryAddressTextView);
                        addressTextView.setText(addressFull);

                        TextView billingTextView = findViewById(R.id.billingAddressTextView);
                        billingTextView.setText(addressFull);
```

Checkout Activity Part 2



```
        TextView  cardNumberTextView = findViewById(R.id.cardNumberTextView);
                cardNumberTextView.setText("Card Ending in " + cardNumber);
            }

        }
    });
}

BasketDB basketDB = new BasketDB(this);
List<String> basketList = basketDB.getDataArray();
ArrayList<String> idList = new ArrayList<>();

for (String row : basketList) {
    String[] rowSplit = row.split(":");
    idList.add(rowSplit[1]);
    quantityList.add(rowSplit[2]);
    priceList.add(rowSplit[3]);
}

Query docRef = db.collection("products").whereIn(FieldPath.documentId(), idList);
docRef.get().addOnCompleteListener(task -> {
    int i = 0;
    if (task.isSuccessful()) {
        for (QueryDocumentSnapshot document : task.getResult()) {
            String name = Objects.requireNonNull(document.getData().get("name")).toString();
            nameList.add(name);
            i++;
        }
        printBasket(i);
    }
});

RadioButton radioBtnFree = findViewById(R.id.radioBtnFree);
radioBtnFree.setChecked(true);

//basket btn
Button basketBtn = findViewById(R.id.basketBtn);
basketBtn.setOnClickListener(v -> {
    Intent intent = new Intent(this, BasketActivity.class);
    intent.putExtra("activity", "CheckoutActivity");
    this.startActivity(intent);
});

TextView totalTextView = findViewById(R.id.totalTextView);
totalTextView.setText("£" + basketDB.getTotal());

RadioGroup  radioGroupDelivery = findViewById(R.id.radioGroupDelivery);
radioGroupDelivery.setOnCheckedChangeListener((group, checkedId) -> {
    int pos = radioGroupDelivery.indexOfChild(findViewById(checkedId));

    switch(pos) {
        case 0:
            totalTextView.setText("£" + basketDB.getTotal());
            break;
        case 1:
            double total = Integer.parseInt(basketDB.getTotal()) + 5.95;
            totalTextView.setText("£" + total);
            break;
    }
});

// buy btn
Button buyBtn = findViewById(R.id.buyBtn);
buyBtn.setOnClickListener(v -> {
    EditText CVVEditText = findViewById(R.id.CVVEditText);
    String CVV = CVVEditText.getText().toString().trim();
    if (CVV.length() == 3) {
        Intent intent = new Intent(this, OrderConfirmedActivity.class);
        RadioButton nextDay = findViewById(R.id.radioBtnNextDay);
        intent.putExtra("nextDay", nextDay.isChecked());
        startActivity(intent);
    }
});
```

Checkout Activity Part 3



```
        //back btn
        Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            finish();
        });

        // change delivery address
        Button changeAddressBtn = findViewById(R.id.addressBtn);
        changeAddressBtn.setOnClickListener(v -> {
            Intent intent = new Intent(this, AccountActivity.class);
            intent.putExtra("activity", "CheckoutActivity");
            this.finish();
            startActivity(intent);
        });

        // change card
        Button changeCardBtn = findViewById(R.id.paymentChangeBtn);
        changeCardBtn.setOnClickListener(v -> {
            Intent intent = new Intent(this, AccountActivity.class);
            intent.putExtra("activity", "CheckoutActivity");
            this.finish();
            startActivity(intent);
        });
    }


    public void printBasket(int i) {
        String names = "";
        String prices = "";
        String qtys = "";
        for (int j = 0; j < i; j++) {
            int k = j + 1;
            names += k + ". " + nameList.get(j).substring(0, 15) + "\n";
            prices += "£" + priceList.get(j) + "\n";
            qtys += "Qty: " + quantityList.get(j) + "\n";

            if (j == i - 1) {
                TextView productNameTextView = findViewById(R.id.productNameTextView);
                productNameTextView.setText(names);
                TextView productPriceTextView = findViewById(R.id.productPriceTextView);
                productPriceTextView.setText(prices);
                TextView productQtyTextView = findViewById(R.id.productQtyTextView);
                productQtyTextView.setText(qtys);
            }
        }
    }
}
```

Contact Activity

```
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

public class ContactActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_contact);

        //back btn
        Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            finish();
        });

        EditText subjectEditText = findViewById(R.id.subjectEditText);
        EditText messageEditText = findViewById(R.id.messageEditText);

        //send btn
        Button sendBtn = findViewById(R.id.sendBtn);
        sendBtn.setOnClickListener(v -> {
            if (subjectEditText.getText().toString().trim().isEmpty()) {
                subjectEditText.setError("Please enter a subject");
            } else if (messageEditText.getText().toString().trim().isEmpty()) {
                messageEditText.setError("Please enter a message");
            } else {
                Intent emailIntent = new Intent(Intent.ACTION_SEND);
                emailIntent.setType("message/rfc822");
                emailIntent.putExtra(Intent.EXTRA_EMAIL, new String[]{"info@descaweb.dev"});
                emailIntent.putExtra(Intent.EXTRA_SUBJECT, subjectEditText.getText().toString());
                emailIntent.putExtra(Intent.EXTRA_TEXT, messageEditText.getText().toString());
                try {
                    startActivity(Intent.createChooser(emailIntent, "Send mail..."));
                } catch (android.content.ActivityNotFoundException ex) {
                    Toast.makeText(ContactActivity.this, "There is no email client installed.",
Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}
```

DBHelper

```java
package com.example.luckymerch;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

public class DBHelper extends SQLiteOpenHelper {

    public static final String KEY_ID = "_id";
    public static final String KEY_PRODUCT_ID = "_product_id";
    public static final String KEY_QUANTITY = "_quantity";
    public static final String KEY_PRICE = "_price";
    public static final String KEY_SIZE = "_size";

    private static final String DATABASE_NAME = "basket";
    private static final String DATABASE_TABLE = "basketTable";
    private static final int DATABASE_VERSION = 3;//version of database

    @Override
    public void onCreate(SQLiteDatabase sqLiteDatabase) {

//              CREATE TABLE basketTable(_id INTEGER PRIMARY KEY AUTOINCREMENT,
//                      _product_id TEXT NOT NULL.
//                      _quantity TEXT NOT NULL);

        String sqlCode = "CREATE TABLE IF NOT EXISTS " + DATABASE_TABLE + " (" +
                KEY_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
                KEY_PRODUCT_ID + " TEXT NOT NULL, " +
                KEY_QUANTITY + " TEXT NOT NULL, " +
                KEY_PRICE + " TEXT NOT NULL, " +
                KEY_SIZE + " TEXT NOT NULL ); ";

        sqLiteDatabase.execSQL(sqlCode);

    }

    @Override
    public void onUpgrade(SQLiteDatabase sqLiteDatabase, int toVersion, int fromVersion) {

        //only called when database file exists, the stored version number is lower than requested.
        sqLiteDatabase.execSQL("DROP TABLE IF EXISTS "+ DATABASE_TABLE);

        onCreate(sqLiteDatabase);

    }


    public DBHelper(Context context){

        super(context, DATABASE_NAME,null, DATABASE_VERSION);

    }

    public static void resetAutoIncrement(SQLiteDatabase db) {

        db.execSQL("DELETE FROM SQLITE_SEQUENCE WHERE NAME = '" + DATABASE_TABLE + "'");
    }
}
```

Image Adapter

```java
package com.example.luckymerch;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;

import androidx.annotation.NonNull;
import androidx.recyclerview.widget.RecyclerView;

import com.bumptech.glide.Glide;

import java.util.ArrayList;

public class ImageAdapter extends RecyclerView.Adapter<ImageAdapter.ViewHolder> {
    Context context;
    ArrayList<String> arrayList;
    OnItemClickListener onItemClickListener;

    public ImageAdapter(Context context, ArrayList<String> arrayList) {
        this.context = context;
        this.arrayList = arrayList;
    }

    @NonNull
    @Override
    public ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View view = LayoutInflater.from(context).inflate(R.layout.image_list_item, parent, false);
        return new ViewHolder(view);
    }

    @Override
    public void onBindViewHolder(@NonNull ViewHolder holder, int position) {
        Glide.with(context).load(arrayList.get(position)).into(holder.imageView);
        holder.itemView.setOnClickListener(view -> onItemClickListener.onClick(holder.imageView,
arrayList.get(position)));
    }

    @Override
    public int getItemCount() {
        return arrayList.size();
    }

    public static class ViewHolder extends RecyclerView.ViewHolder {
        ImageView imageView;
        public ViewHolder(@NonNull View itemView) {
            super(itemView);
            imageView = itemView.findViewById(R.id.list_item_image);
        }
    }

    public void setOnItemClickListener(OnItemClickListener onItemClickListener) {
        this.onItemClickListener = onItemClickListener;
    }

    public interface OnItemClickListener {
        void onClick(ImageView imageView, String path);
    }
}
```

Image View Activity

```java
package com.example.luckymerch;

import androidx.appcompat.app.AppCompatActivity;

import android.os.Bundle;
import android.widget.ImageView;

import com.bumptech.glide.Glide;

public class ImageViewActivity extends AppCompatActivity {

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

        ImageView imageView = findViewById(R.id.imageView);

        Glide.with(ImageViewActivity.this).load(getIntent().getStringExtra("image")).into(imageView);
    }
}
```

Login Activity Part 1

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;

public final class LoginActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;

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

        mAuth = FirebaseAuth.getInstance();

        //back btn
        final Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            this.finish();
        });

        //login btn
        final Button loginBtn = findViewById(R.id.login);
        EditText usernameField = findViewById(R.id.usernameEditText);
        EditText passwordField = findViewById(R.id.passwordEditText);

        loginBtn.setOnClickListener(v -> {
            signIn(usernameField.getText().toString().trim(), passwordField.getText().toString());
        });

        //register btn
        final Button registerBtn = findViewById(R.id.register);
        registerBtn.setOnClickListener(v -> {
            this.startActivity(new Intent(this, RegisterActivity.class));
        });

        //forgot password btn
        final Button forgotPasswordBtn = findViewById(R.id.forgotPassword);
        forgotPasswordBtn.setOnClickListener(v -> {
           if (usernameField.getText().toString().trim().isEmpty()){
               usernameField.setError("Enter email adn try again");
           } else {
               mAuth.sendPasswordResetEmail(usernameField.getText().toString().trim())
                   .addOnCompleteListener(task -> {
                       if (task.isSuccessful()) {
                           Log.d("FirebaseAuth", "Email sent.");
                           Toast.makeText(LoginActivity.this, "Password reset email sent to: " +
usernameField.getText().toString().trim(), Toast.LENGTH_SHORT).show();
                       } else {
                           usernameField.setError("Email not found");
                       }
                   });
           }

        });

    }
```

Login Activity Part 2

```
    private void signIn(String email, String password) {
        mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, task -> {
                    if (task.isSuccessful()) {
                        // Sign in success, update UI with the signed-in user's information
                        Log.d("LoginActivity", "signInWithEmail:success");
                        FirebaseUser user = mAuth.getCurrentUser();
                        finish();
                        startActivity(new Intent(LoginActivity.this, MainActivity.class));
                    } else {
                        // If sign in fails, display a message to the user.
                        Log.w("LoginActivity", "signInWithEmail:failure", task.getException());
                        Toast.makeText(LoginActivity.this, "Authentication failed.",
                                Toast.LENGTH_SHORT).show();
                    }
                });
    }
}
```

Main Activity Part 1

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.google.android.material.bottomsheet.BottomSheetBehavior;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import org.jetbrains.annotations.NotNull;

import java.util.ArrayList;
import java.util.Objects;


public final class MainActivity extends AppCompatActivity {
    ArrayList<Product> products = new ArrayList<>();
    private FirebaseAuth mAuth;


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

        mAuth = FirebaseAuth.getInstance();

        // set up bottom sheet depending on logged in state
        onStart();

        // set marquee to scroll horizontally
        TextView txtMarquee = findViewById(R.id.marqueeText);
        txtMarquee.setSelected(true);

        final BasketDB basketDB = new BasketDB(this);
        // used to reset basket database
//        basketDB.openDatabaseConnection();
//        basketDB.clearAllRecords();
//        basketDB.deleteDatabase();
//        basketDB.closeDatabaseConnection();

        // load recycler view for products
        getAllCollections(null, null);

        // used for adding product items to database
//        Map<String, Object> data = new HashMap<>();
//        data.put("name", "XO CLASSIC LOGO LONG SLEEVE WHITE");
//        data.put("category", "The Weeknd");
//        data.put("description", "WHITE 6.5 OZ HEAVYWEIGHT COTTON LONGSLEEVE FEATURING DAWN FM XO GRAPHICS ON FRONT. MACHINE WASH COLD, DELICATE CYCLE, TUMBLE DRY LOW. DO NOT IRON EMBELLISHMENT. 100% COTTON.");
//        data.put("price", "60");
//        data.put("imageFront", "https://firebasestorage.googleapis.com/v0/b/lucky-merch.appspot.com/o/XO-CLASSIC-LOGO-LONGSLEEVE-WHITE_800x.webp?alt=media&token=c161ff3d-fd0f-40a5-890e-8dc60dc9b1d3");
//        data.put("imageBack", "https://firebasestorage.googleapis.com/v0/b/lucky-merch.appspot.com/o/black.webp?alt=media&token=bac2c049-28f4-449b-b2c8-cf4c4aa31f24");
//        data.put("quantity", "50");
//
//        //https://firebasestorage.googleapis.com/v0/b/lucky-merch.appspot.com/o/black.webp?alt=media&token=bac2c049-28f4-449b-b2c8-cf4c4aa31f24
//        db.collection("products")
//                .add(data)
//                .addOnSuccessListener(documentReference -> Log.d("TAG", "DocumentSnapshot written with ID: " + documentReference.getId()))
//                .addOnFailureListener(e -> Log.w("TAG", "Error adding document", e));
```

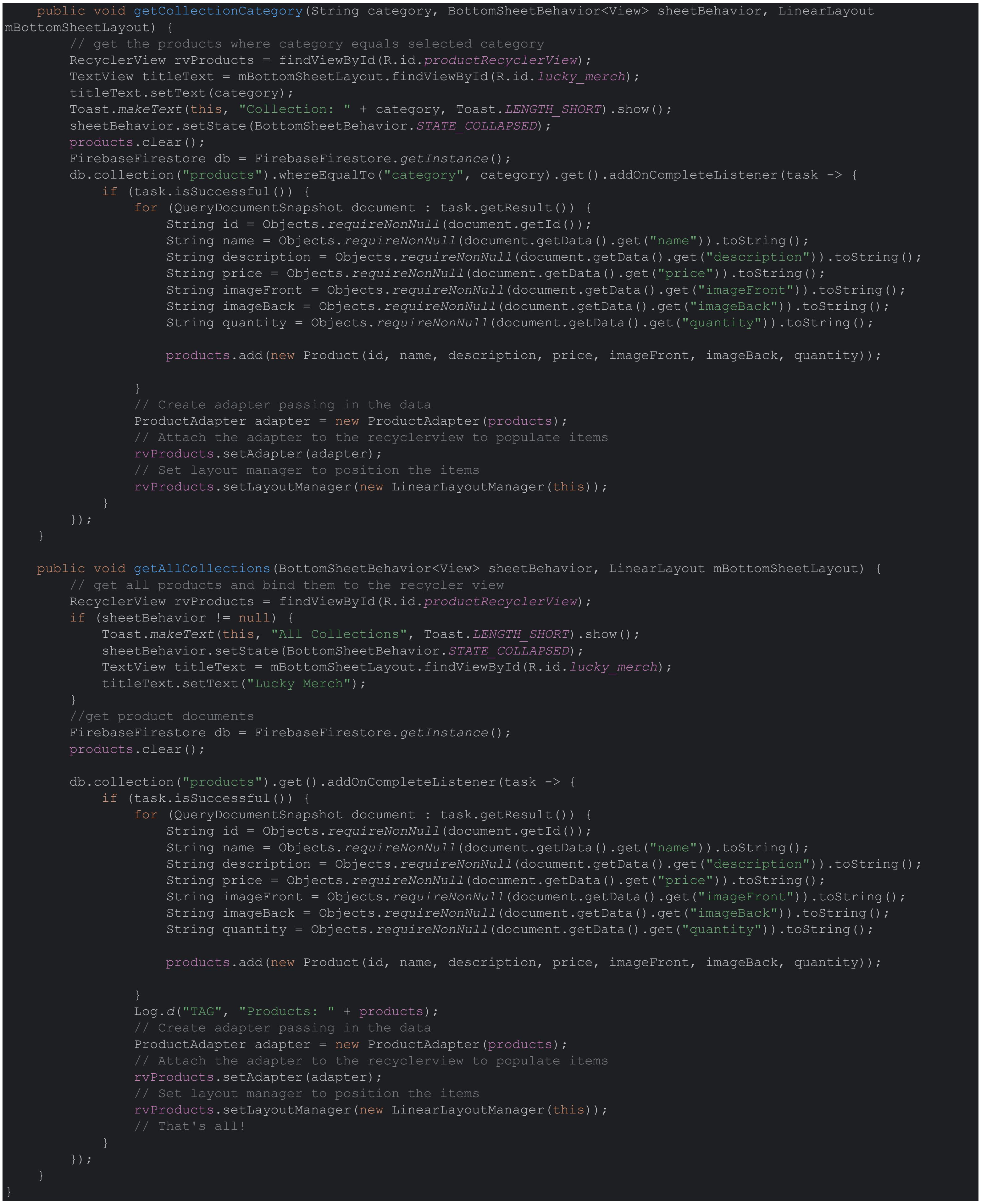
Main Activity Part 2

```
    //sign in button
    final Button signInBtn = findViewById(R.id.SignInBtn);
    signInBtn.setOnClickListener(v -> {
        this.startActivity(new Intent(this, LoginActivity.class));
    });

    //sign out button
    final Button signOutBtn = findViewById(R.id.SignOutBtn);
    signOutBtn.setOnClickListener(v -> {
        mAuth.signOut();
        onStart();
    });

    //account btn
    Button accountBtn = findViewById(R.id.accountBtn);
    accountBtn.setOnClickListener(v -> {
        startActivity(new Intent(MainActivity.this, AccountActivity.class));
    });

    //orders btn
    Button ordersBtn = findViewById(R.id.ordersBtn);
    ordersBtn.setOnClickListener(v -> {
        startActivity(new Intent(MainActivity.this, OrdersActivity.class));
    });

    //basket btn signed out
    ImageButton basketBtnSignedOut = findViewById(R.id.basketBtnSignedOut);
    if(basketDB.getRowCount() > 0) {
        basketBtnSignedOut.setImageResource(R.drawable.cart_icon_full);
    }
    basketBtnSignedOut.setOnClickListener(v -> {
        this.startActivity(new Intent(this, BasketActivity.class));
    });

    //basket btn signed in
    ImageButton basketBtnSignedIn = findViewById(R.id.basketBtnSignedIn);
    if(basketDB.getRowCount() > 0) {
        basketBtnSignedIn.setImageResource(R.drawable.cart_icon_full);
    }
    basketBtnSignedIn.setOnClickListener(v -> {
        this.startActivity(new Intent(this, BasketActivity.class));
    });

    //contact btn signed out
    ImageButton contactBtnSignedOut = findViewById(R.id.contactBtnSignedOut);
    contactBtnSignedOut.setOnClickListener(v -> {
        this.startActivity(new Intent(this, ContactActivity.class));
    });

    //contact btn signed in
    ImageButton contactBtnSignedIn = findViewById(R.id.contactBtnSignedIn);
    contactBtnSignedIn.setOnClickListener(v -> {
        this.startActivity(new Intent(this, ContactActivity.class));
    });
}
```

Main Activity Part 3

```java
public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update bottom sheet and filters
    FirebaseUser currentUser = mAuth.getCurrentUser();
    if(currentUser != null){
        LinearLayout mBottomSheetLayoutLoggedIn = findViewById(R.id.bottom_sheet_layout_logged_in);
        mBottomSheetLayoutLoggedIn.setVisibility(View.VISIBLE);

        LinearLayout mBottomSheetLayoutLoggedOut = findViewById(R.id.bottom_sheet_layout_logged_out);
        mBottomSheetLayoutLoggedOut.setVisibility(View.INVISIBLE);

        SetUpBottomSheet(mBottomSheetLayoutLoggedIn);
        setUpFilters(mBottomSheetLayoutLoggedIn);
    } else {
        LinearLayout mBottomSheetLayoutLoggedOut = findViewById(R.id.bottom_sheet_layout_logged_out);
        mBottomSheetLayoutLoggedOut.setVisibility(View.VISIBLE);

        LinearLayout mBottomSheetLayoutLoggedIn = findViewById(R.id.bottom_sheet_layout_logged_in);
        mBottomSheetLayoutLoggedIn.setVisibility(View.INVISIBLE);

        SetUpBottomSheet(mBottomSheetLayoutLoggedOut);
        setUpFilters(mBottomSheetLayoutLoggedOut);
    }
}

public void SetUpBottomSheet(LinearLayout mBottomSheetLayout){
    // set up state behaviour and menu button
    BottomSheetBehavior<View> sheetBehavior = BottomSheetBehavior.from(mBottomSheetLayout);
    ImageButton mButton = mBottomSheetLayout.findViewById(R.id.menuBtn);
    mButton.setOnClickListener(it -> {
        if (sheetBehavior.getState() != BottomSheetBehavior.STATE_EXPANDED) {
            sheetBehavior.setState(BottomSheetBehavior.STATE_EXPANDED);
            mButton.setBackgroundResource(R.drawable.close_icon);
        } else {
            sheetBehavior.setState(BottomSheetBehavior.STATE_COLLAPSED);
            mButton.setBackgroundResource(R.drawable.menu_icon);
        }

    });
    sheetBehavior.addBottomSheetCallback(new BottomSheetBehavior.BottomSheetCallback() {
        public void onStateChanged(@NonNull View bottomSheet, int newState) {
            if (sheetBehavior.getState() == BottomSheetBehavior.STATE_EXPANDED) {
                mButton.setBackgroundResource(R.drawable.close_icon);
            } else {
                mButton.setBackgroundResource(R.drawable.menu_icon);
            }

        }
        public void onSlide(@NotNull View bottomSheet, float slideOffset) {
        }
    });
}

public void setUpFilters(LinearLayout mBottomSheetLayout) {
    // on click of filter button run get collection category with correct category filter
    BottomSheetBehavior<View> sheetBehavior = BottomSheetBehavior.from(mBottomSheetLayout);
    Button allCollectionsBtn = mBottomSheetLayout.findViewById(R.id.allCollectionsBtn);
    allCollectionsBtn.setOnClickListener(v -> getAllCollections(sheetBehavior, mBottomSheetLayout));
    Button harryStylesBtn = mBottomSheetLayout.findViewById(R.id.harryStylesBtn);
    harryStylesBtn.setOnClickListener(v -> getCollectionCategory("Harry Styles", sheetBehavior, mBottomSheetLayout));
    Button ladyGagaBtn = mBottomSheetLayout.findViewById(R.id.ladyGagaBtn);
    ladyGagaBtn.setOnClickListener(v -> getCollectionCategory("Lady Gaga", sheetBehavior, mBottomSheetLayout));
    Button arianaGrandeBtn = mBottomSheetLayout.findViewById(R.id.arianaGrandeBtn);
    arianaGrandeBtn.setOnClickListener(v -> getCollectionCategory("Ariana Grande", sheetBehavior, mBottomSheetLayout));
    Button eminemBtn = mBottomSheetLayout.findViewById(R.id.eminemBtn);
    eminemBtn.setOnClickListener(v -> getCollectionCategory("Eminem", sheetBehavior, mBottomSheetLayout));
    Button justinBieberBtn = mBottomSheetLayout.findViewById(R.id.justinBieberBtn);
    justinBieberBtn.setOnClickListener(v -> getCollectionCategory("Justin Bieber", sheetBehavior, mBottomSheetLayout));
    Button theWeekndBtn = mBottomSheetLayout.findViewById(R.id.theWeekndBtn);
    theWeekndBtn.setOnClickListener(v -> getCollectionCategory("The Weeknd", sheetBehavior, mBottomSheetLayout));
    Button shawnMendesBtn = mBottomSheetLayout.findViewById(R.id.shawnMendesBtn);
    shawnMendesBtn.setOnClickListener(v -> getCollectionCategory("Shawn Mendes", sheetBehavior, mBottomSheetLayout));
    Button nickiMinajBtn = mBottomSheetLayout.findViewById(R.id.nickiMinajBtn);
    nickiMinajBtn.setOnClickListener(v -> getCollectionCategory("Nicki Minaj", sheetBehavior, mBottomSheetLayout));
}
```

Main Activity Part 4

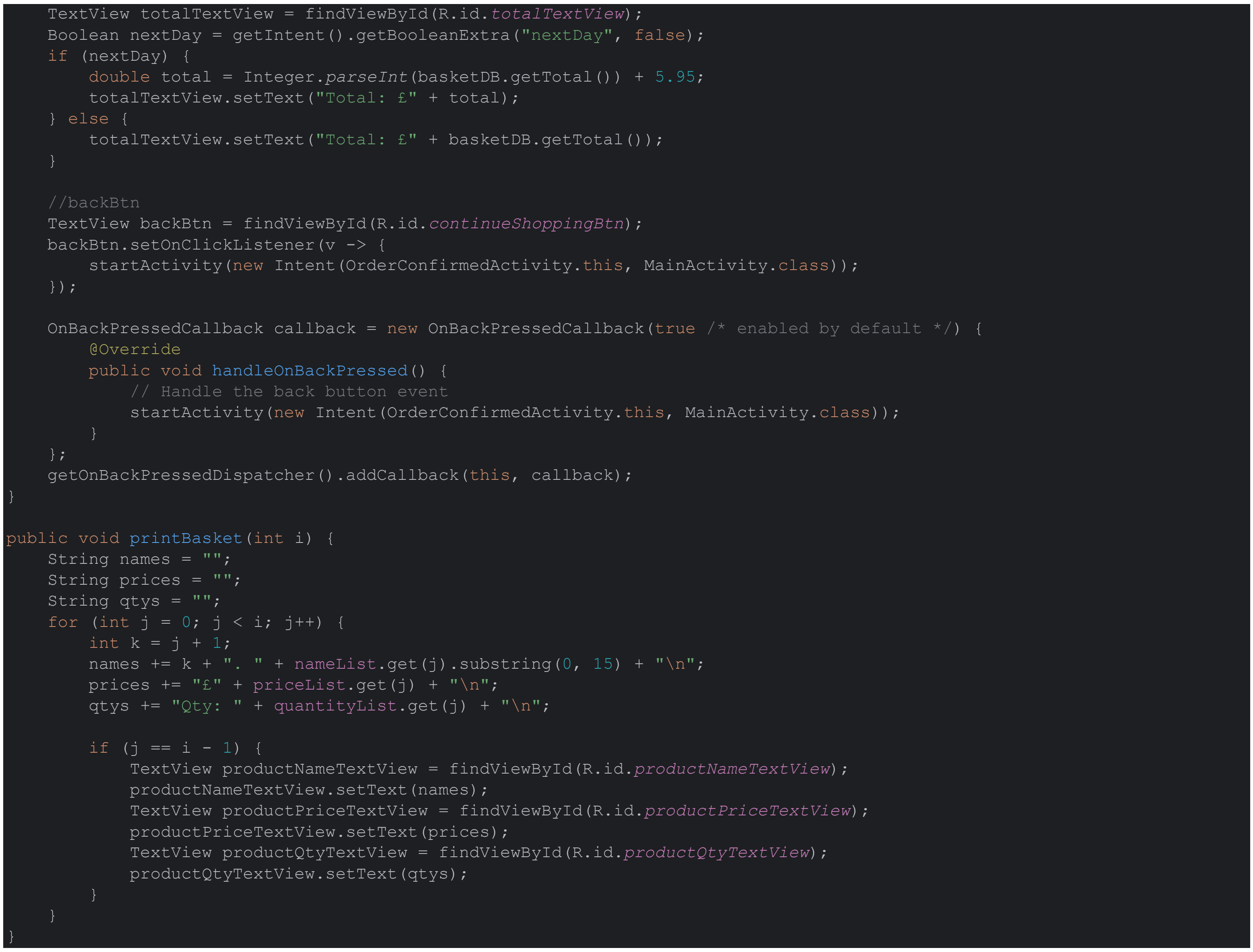


```java
    public void getCollectionCategory(String category, BottomSheetBehavior<View> sheetBehavior, LinearLayout 
mBottomSheetLayout) {
        // get the products where category equals selected category
        RecyclerView rvProducts = findViewById(R.id.productRecyclerView);
        TextView titleText = mBottomSheetLayout.findViewById(R.id.lucky_merch);
        titleText.setText(category);
        Toast.makeText(this, "Collection: " + category, Toast.LENGTH_SHORT).show();
        sheetBehavior.setState(BottomSheetBehavior.STATE_COLLAPSED);
        products.clear();
        FirebaseFirestore db = FirebaseFirestore.getInstance();
        db.collection("products").whereEqualTo("category", category).get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                for (QueryDocumentSnapshot document : task.getResult()) {
                    String id = Objects.requireNonNull(document.getId());
                    String name = Objects.requireNonNull(document.getData().get("name")).toString();
                    String description = Objects.requireNonNull(document.getData().get("description")).toString();
                    String price = Objects.requireNonNull(document.getData().get("price")).toString();
                    String imageFront = Objects.requireNonNull(document.getData().get("imageFront")).toString();
                    String imageBack = Objects.requireNonNull(document.getData().get("imageBack")).toString();
                    String quantity = Objects.requireNonNull(document.getData().get("quantity")).toString();

                    products.add(new Product(id, name, description, price, imageFront, imageBack, quantity));

                }
                // Create adapter passing in the data
                ProductAdapter adapter = new ProductAdapter(products);
                // Attach the adapter to the recyclerview to populate items
                rvProducts.setAdapter(adapter);
                // Set layout manager to position the items
                rvProducts.setLayoutManager(new LinearLayoutManager(this));
            }
        });
    }

    public void getAllCollections(BottomSheetBehavior<View> sheetBehavior, LinearLayout mBottomSheetLayout) {
        // get all products and bind them to the recycler view
        RecyclerView rvProducts = findViewById(R.id.productRecyclerView);
        if (sheetBehavior != null) {
            Toast.makeText(this, "All Collections", Toast.LENGTH_SHORT).show();
            sheetBehavior.setState(BottomSheetBehavior.STATE_COLLAPSED);
            TextView titleText = mBottomSheetLayout.findViewById(R.id.lucky_merch);
            titleText.setText("Lucky Merch");
        }
        //get product documents
        FirebaseFirestore db = FirebaseFirestore.getInstance();
        products.clear();

        db.collection("products").get().addOnCompleteListener(task -> {
            if (task.isSuccessful()) {
                for (QueryDocumentSnapshot document : task.getResult()) {
                    String id = Objects.requireNonNull(document.getId());
                    String name = Objects.requireNonNull(document.getData().get("name")).toString();
                    String description = Objects.requireNonNull(document.getData().get("description")).toString();
                    String price = Objects.requireNonNull(document.getData().get("price")).toString();
                    String imageFront = Objects.requireNonNull(document.getData().get("imageFront")).toString();
                    String imageBack = Objects.requireNonNull(document.getData().get("imageBack")).toString();
                    String quantity = Objects.requireNonNull(document.getData().get("quantity")).toString();

                    products.add(new Product(id, name, description, price, imageFront, imageBack, quantity));

                }
                Log.d("TAG", "Products: " + products);
                // Create adapter passing in the data
                ProductAdapter adapter = new ProductAdapter(products);
                // Attach the adapter to the recyclerview to populate items
                rvProducts.setAdapter(adapter);
                // Set layout manager to position the items
                rvProducts.setLayoutManager(new LinearLayoutManager(this));
                // That's all!
            }
        });
    }
}
```

Order Confirmed Activity Part 1

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.widget.TextView;

import androidx.activity.OnBackPressedCallback;
import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.Timestamp;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FieldPath;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.Query;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Objects;

public class OrderConfirmedActivity extends AppCompatActivity {
    private FirebaseAuth mAuth;
    ArrayList<String> quantityList = new ArrayList<>();
    ArrayList<String> priceList = new ArrayList<>();
    ArrayList<String> nameList = new ArrayList<>();
    String email;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_order_confirmed);

        FirebaseFirestore db = FirebaseFirestore.getInstance();

        mAuth = FirebaseAuth.getInstance();
        FirebaseUser currentUser = mAuth.getCurrentUser();

        if (currentUser != null) {
            email = currentUser.getEmail();

            TextView confirmationEmailText = findViewById(R.id.confirmationEmailTextView);
            confirmationEmailText.setText("A confirmation email has been sent to " + email);
        }

        BasketDB basketDB = new BasketDB(this);
        List<String> basketList = basketDB.getDataArray();
        ArrayList<String> idList = new ArrayList<>();

        for (String row : basketList) {
            String[] rowSplit = row.split(":");
            idList.add(rowSplit[1]);
            quantityList.add(rowSplit[2]);
            priceList.add(rowSplit[3]);
        }

        Query docRef = db.collection("products").whereIn(FieldPath.documentId(), idList);
        docRef.get().addOnCompleteListener(task -> {
            int i = 0;
            if (task.isSuccessful()) {
                for (QueryDocumentSnapshot document : task.getResult()) {
                    String name = Objects.requireNonNull(document.getData().get("name")).toString();
                    nameList.add(name);

                    i++;
                }
                printBasket(i);
                addOrderToServer(idList, quantityList, currentUser, basketDB.getTotal());
                stockUpdate(idList, quantityList);
            }
        });
```

Order Confirmed Activity Part 2



```
    TextView totalTextView = findViewById(R.id.totalTextView);
    Boolean nextDay = getIntent().getBooleanExtra("nextDay", false);
    if (nextDay) {
        double total = Integer.parseInt(basketDB.getTotal()) + 5.95;
        totalTextView.setText("Total: £" + total);
    } else {
        totalTextView.setText("Total: £" + basketDB.getTotal());
    }

    //backBtn
    TextView backBtn = findViewById(R.id.continueShoppingBtn);
    backBtn.setOnClickListener(v -> {
        startActivity(new Intent(OrderConfirmedActivity.this, MainActivity.class));
    });

    OnBackPressedCallback callback = new OnBackPressedCallback(true /* enabled by default */) {
        @Override
        public void handleOnBackPressed() {
            // Handle the back button event
            startActivity(new Intent(OrderConfirmedActivity.this, MainActivity.class));
        }
    };
    getOnBackPressedDispatcher().addCallback(this, callback);
}

public void printBasket(int i) {
    String names = "";
    String prices = "";
    String qtys = "";
    for (int j = 0; j < i; j++) {
        int k = j + 1;
        names += k + ". " + nameList.get(j).substring(0, 15) + "\n";
        prices += "£" + priceList.get(j) + "\n";
        qtys += "Qty: " + quantityList.get(j) + "\n";

        if (j == i - 1) {
            TextView productNameTextView = findViewById(R.id.productNameTextView);
            productNameTextView.setText(names);
            TextView productPriceTextView = findViewById(R.id.productPriceTextView);
            productPriceTextView.setText(prices);
            TextView productQtyTextView = findViewById(R.id.productQtyTextView);
            productQtyTextView.setText(qtys);
        }
    }
}
```

Order Confirmed Activity Part 3

```java
    public void addOrderToServer(ArrayList<String> idList, ArrayList<String> quantityList, FirebaseUser currentUser,
String total) {

        FirebaseFirestore db = FirebaseFirestore.getInstance();
        if (currentUser != null) {
            email = currentUser.getEmail();
            Query docRef = db.collection("users").whereEqualTo("email", email);
            docRef.get().addOnCompleteListener(task -> {
                if (task.isSuccessful()) {
                    String id = null;
                    for (QueryDocumentSnapshot document : task.getResult()) {
                        id = Objects.requireNonNull(document.getId());
                    }
                    if (id != null) {
                        DocumentReference userDocRef = db.collection("users").document(id);

                        Map<String, Object> data = new HashMap<>();
                        data.put("customer", userDocRef);
                        for(int i = 0; i < idList.size(); i++) {
                            String product = "product" + (i + 1);
                            String quantity = "quantity" + (i + 1);
                            data.put(product, db.collection("products").document(idList.get(i)));
                            data.put(quantity, quantityList.get(i));
                            data.put("orderDate", Timestamp.now());
                            data.put("total", total);
                        }
                        db.collection("orders")
                                .add(data)
                                .addOnSuccessListener(documentReference -> Log.d("TAG", "DocumentSnapshot written with
ID: " + documentReference.getId()))
                                .addOnFailureListener(e -> Log.w("TAG", "Error adding document", e));
                    }
                }
            });
        }
    }

    public void stockUpdate(ArrayList<String> idList, ArrayList<String> quantityList) {
        BasketDB basketDB = new BasketDB(this);
        basketDB.clearAllRecords();

        FirebaseFirestore db = FirebaseFirestore.getInstance();
        for (int i = 0; i < idList.size(); i++) {
            int quantityToMinus = Integer.parseInt(quantityList.get(i));
            db.collection("products").document(idList.get(i)).get().addOnCompleteListener(task -> {
                if (task.isSuccessful()) {
                    String quantity = Objects.requireNonNull(task.getResult().getData().get("quantity")).toString();
                    int quantityInt = Integer.parseInt(quantity);

                    db.collection("products").document(task.getResult().getId()).update("quantity", quantityInt -
quantityToMinus);
                }
            });
        }
    }
}
```

Orders Activity

```java
package com.example.luckymerch;

import android.os.Bundle;
import android.widget.Button;

import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.Query;
import com.google.firebase.firestore.QueryDocumentSnapshot;

import java.util.ArrayList;

public class OrdersActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;

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

        mAuth = FirebaseAuth.getInstance();

        //back btn
        final Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            this.finish();
        });

        FirebaseUser currentUser = mAuth.getCurrentUser();

        if (currentUser != null) {
            FirebaseFirestore db = FirebaseFirestore.getInstance();
            String email = currentUser.getEmail();

            Query user = db.collection("users").whereEqualTo("email", email);
            user.get().addOnCompleteListener(task -> {
                if(task.isSuccessful()) {
                    for (QueryDocumentSnapshot documentSnapshot : task.getResult()) {
                        DocumentReference userReference = documentSnapshot.getReference();
                        Query orders = db.collection("orders").whereEqualTo("customer", userReference);
                        orders.get().addOnCompleteListener(task1 -> {

                            ArrayList<String> ordersList = new ArrayList<>();
                            if (task1.isSuccessful()) {
                                for (QueryDocumentSnapshot documentSnapshot1 : task1.getResult()) {
                                    ordersList.add(documentSnapshot1.getId());
                                }
                                // Lookup the recyclerview in activity layout
                                RecyclerView rvOrders = findViewById(R.id.ordersRecyclerView);
                                // Create adapter passing in the data
                                OrdersAdapter adapter = new OrdersAdapter(ordersList);
                                // Attach the adapter to the recyclerview to populate items
                                rvOrders.setAdapter(adapter);
                                // Set layout manager to position the items
                                rvOrders.setLayoutManager(new LinearLayoutManager(this));
                            }
                        });
                    }
                }
            });

        }
    }
}
```

Orders Adapter Part 1

```java
package com.example.luckymerch;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

import androidx.recyclerview.widget.RecyclerView;

import com.google.firebase.Timestamp;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.Date;
import java.util.List;

public class OrdersAdapter extends RecyclerView.Adapter<OrdersAdapter.ViewHolder> {
    // Provide a direct reference to each of the views within a data item
    // Used to cache the views within the item layout for fast access
    public class ViewHolder extends RecyclerView.ViewHolder {
        // Your holder should contain a member variable
        // for any view that will be set as you render a row
        public TextView productsNameView, productDateView, productTotalView;


        // We also create a constructor that accepts the entire item row
        // and does the view lookups to find each subview
        public ViewHolder(View itemView) {
            // Stores the itemView in a public final member variable that can be used
            // to access the context from any ViewHolder instance.
            super(itemView);

            productsNameView = itemView.findViewById(R.id.productsNameView);
            productDateView = itemView.findViewById(R.id.productDateView);
            productTotalView = itemView.findViewById(R.id.productTotalView);
        }
    }

    // Store a variable
    private List<String> mOrders;

    // Pass in the array into the constructor
    public OrdersAdapter(List<String> orders) {
        mOrders = orders;
    }

    @Override
    public OrdersAdapter.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        Context context = parent.getContext();
        LayoutInflater inflater = LayoutInflater.from(context);

        // Inflate the custom layout
        View ordersView = inflater.inflate(R.layout.orders_recycler_layout, parent, false);

        // Return a new holder instance
        OrdersAdapter.ViewHolder viewHolder = new OrdersAdapter.ViewHolder(ordersView);
        return viewHolder;
    }

    @Override
    public void onBindViewHolder(OrdersAdapter.ViewHolder holder, int position) {
        String item = mOrders.get(position);

        FirebaseFirestore db = FirebaseFirestore.getInstance();

        DocumentReference docRef = db.collection("orders").document(item);
        docRef.get().addOnCompleteListener(task -> {
           if(task.isSuccessful()) {
               DocumentSnapshot documentSnapshot = task.getResult();
               if (documentSnapshot.exists()) {
                   int numOfProducts = (documentSnapshot.getData().size() - 3) / 2;
```

Orders Adapter Part 2

```
                    for (int i = 0; i < numOfProducts; i++) {
                        int j = i + 1;
                        String tempProd = "product" + j;
                        DocumentReference documentReference = (DocumentReference)
documentSnapshot.getData().get(tempProd);
                        documentReference.get().addOnCompleteListener(task1 ->  {
                            if (task1.isSuccessful()) {
                                DocumentSnapshot documentSnapshot1 = task1.getResult();
                                if (documentSnapshot1.exists()) {
                                    TextView productsNameView = holder.productsNameView;
                                    if (productsNameView.getText().equals("Products: ")) {
                                        productsNameView.setText(productsNameView.getText() +
documentSnapshot1.getData().get("name").toString());
                                    } else {
                                        productsNameView.setText(productsNameView.getText() + ", " +
documentSnapshot1.getData().get("name").toString());
                                    }
                                }
                            }
                        });
                    }
                    Timestamp firebaseTimestamp = (Timestamp) documentSnapshot.getData().get("orderDate");
                    Date javaDate = firebaseTimestamp.toDate();
                    TextView productDateView = holder.productDateView;
                    productDateView.setText(productDateView.getText() + javaDate.toString());

                    String total = (String) documentSnapshot.getData().get("total");
                    TextView productTotalView = holder.productTotalView;
                    productTotalView.setText(productTotalView.getText() + total);
                }
            }
        });
    }

    @Override
    public int getItemCount() {
        return mOrders.size();
    }
}
```

Product

```java
package com.example.luckymerch;

import androidx.annotation.NonNull;

public class Product {

    private String id;
    private String name;
    private String description;
    private String price;
    private String imageFront;
    private String imageBack;
    private String quantity;

    public Product(String id, String name, String description, String price, String imageFront, String imageBack, String
quantity) {
        this.id = id;
        this.name = name;
        this.description = description;
        this.price = price;
        this.imageFront = imageFront;
        this.imageBack = imageBack;
        this.quantity = quantity;
    }

    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
    public String getPrice() {
        return price;
    }
    public void setPrice(String price) {
        this.price = price;
    }
    public String getImageFront() {
        return imageFront;
    }
    public void setImageFront(String imageFront) {
        this.imageFront = imageFront;
    }
    public String getImageBack() {
        return imageBack;
    }
    public void setImageBack(String imageBack) {
        this.imageBack = imageBack;
    }
    public String getQuantity() {
        return quantity;
    }
    public void setQuantity(String quantity) {
        this.quantity = quantity;
    }
    @NonNull
    public String toString() {
        return name + " " + description + " " + price + " " + imageFront + " " + imageBack + " " + quantity;
    }

//    public static ArrayList<Product> createProductsList() {
//
//
//        return products;
//    }
}
```

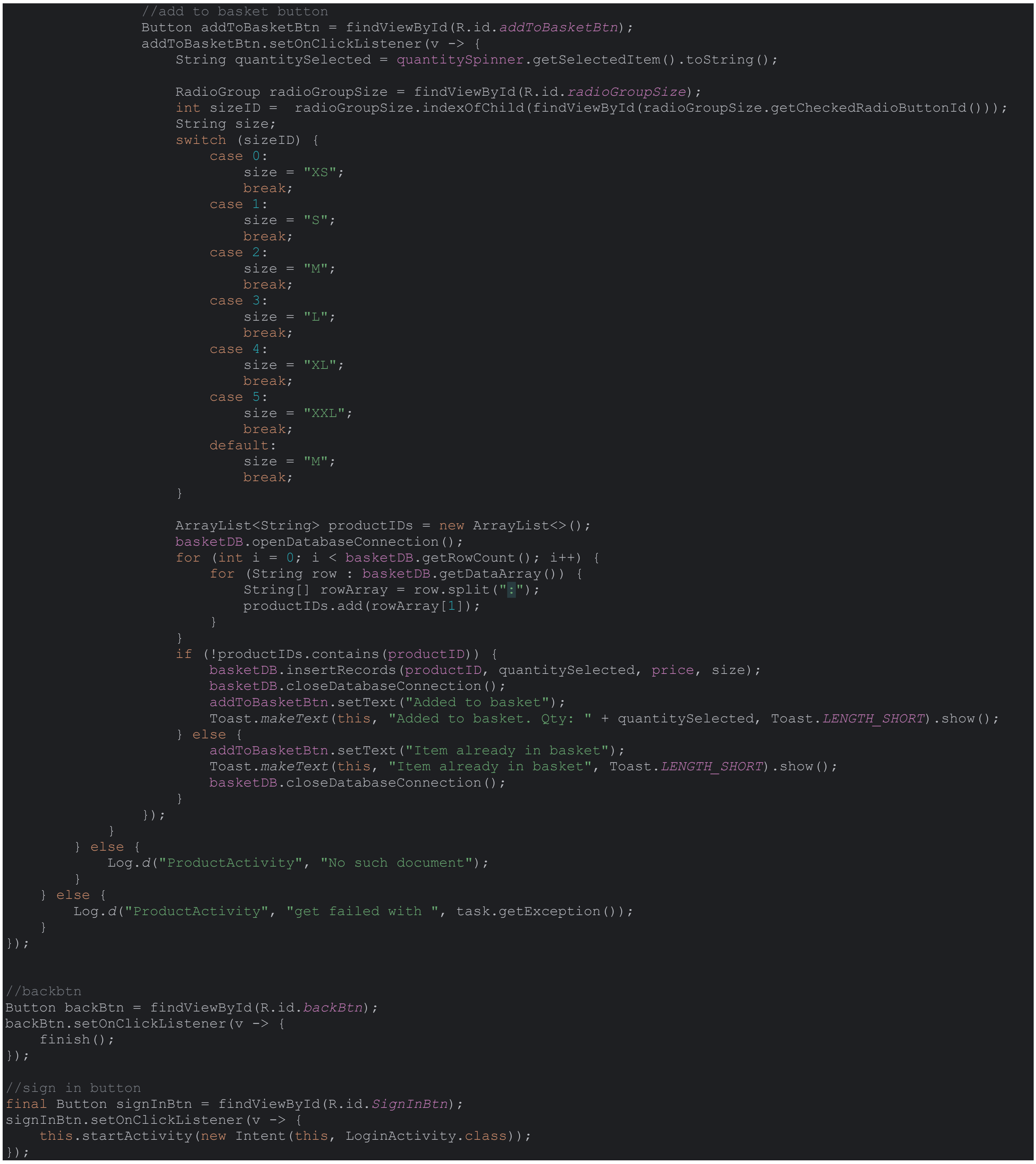
Product Activity Part 1

```java
package com.example.luckymerch;

import android.app.ActivityOptions;
import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.view.View;
import android.widget.ArrayAdapter;
import android.widget.Button;
import android.widget.ImageButton;
import android.widget.LinearLayout;
import android.widget.RadioButton;
import android.widget.RadioGroup;
import android.widget.Spinner;
import android.widget.TextView;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;
import androidx.recyclerview.widget.RecyclerView;

import com.google.android.material.bottomsheet.BottomSheetBehavior;
import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.FirebaseFirestore;

import org.jetbrains.annotations.NotNull;

import java.util.ArrayList;
import java.util.Objects;


public class ProductActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;

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

        mAuth = FirebaseAuth.getInstance();

        final BasketDB basketDB = new BasketDB(this);

        onStart();

        TextView nameTextView = findViewById(R.id.productNameView);
        TextView priceTextView = findViewById(R.id.productPriceView);
        TextView collectionTextView = findViewById(R.id.productCollectionView);
        TextView descriptionTextView = findViewById(R.id.productDescriptionView);
        Spinner quantitySpinner = findViewById(R.id.quantitySpinner);
        RadioButton radioButtonM = findViewById(R.id.radioButton3);
        radioButtonM.setChecked(true);

        String productID = getIntent().getStringExtra("id");

        //get product documents
        FirebaseFirestore db = FirebaseFirestore.getInstance();
```

Product Activity Part 2

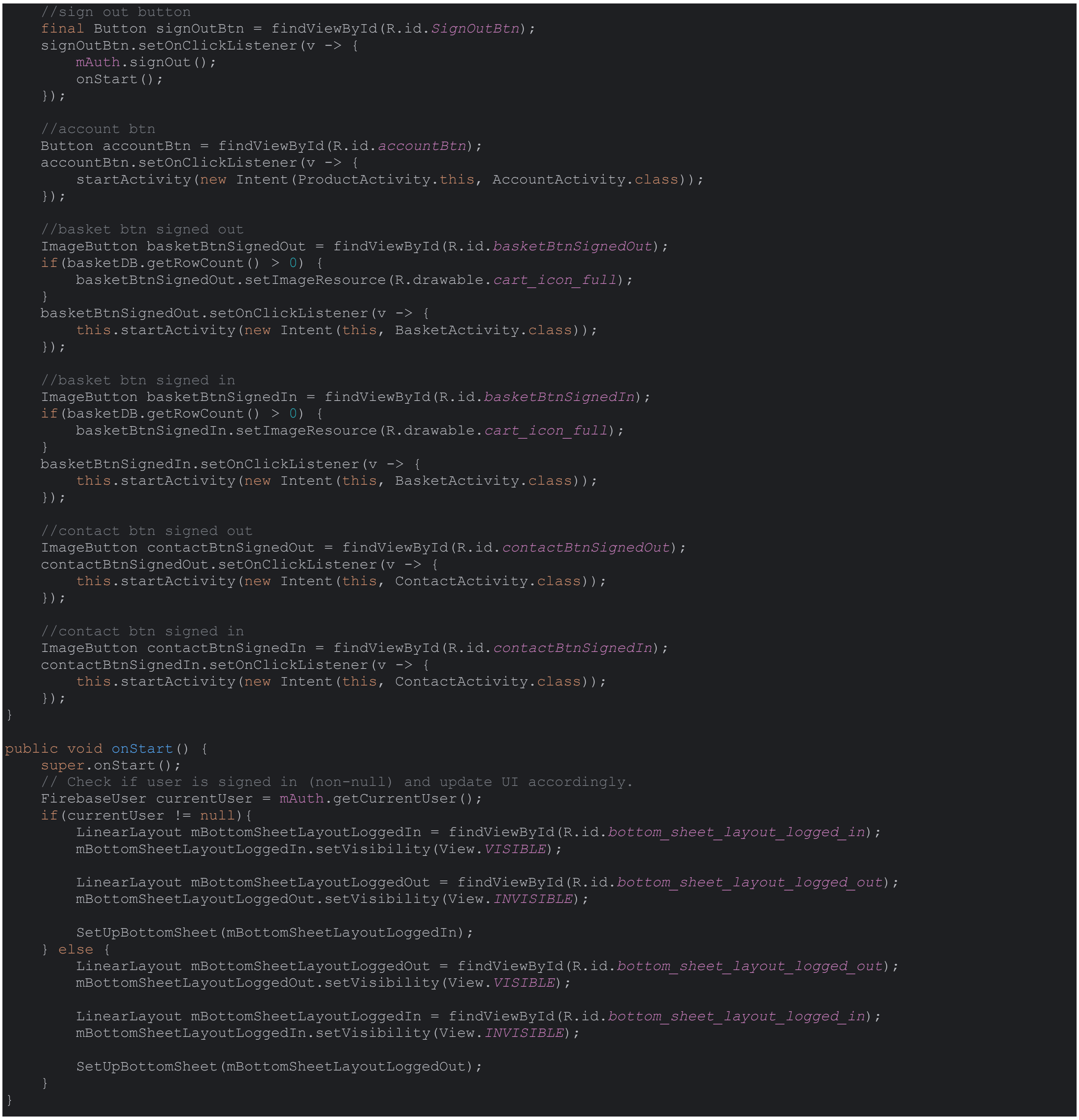
```java
DocumentReference docRef = db.collection("products").document(Objects.requireNonNull(productID));
docRef.get().addOnCompleteListener(task -> {
    if (task.isSuccessful()) {
        DocumentSnapshot document = task.getResult();
        if (document.exists()) {
            Log.d("ProductActivity", "DocumentSnapshot data: " + document.getData());
            String id = Objects.requireNonNull(document.getId());
            String name = Objects.requireNonNull(document.getData().get("name")).toString();
            String description = Objects.requireNonNull(document.getData().get("description")).toString();
            String price = Objects.requireNonNull(document.getData().get("price")).toString();
            String imageFront = Objects.requireNonNull(document.getData().get("imageFront")).toString();
            String imageBack = Objects.requireNonNull(document.getData().get("imageBack")).toString();
            String quantity = Objects.requireNonNull(document.getData().get("quantity")).toString();
            String category = Objects.requireNonNull(document.getData().get("category")).toString();

            RecyclerView recyclerView = findViewById(R.id.recycler);
            ArrayList<String> arrayList = new ArrayList<>();

            arrayList.add(imageFront);
            arrayList.add(imageBack);

            ImageAdapter adapter = new ImageAdapter(ProductActivity.this, arrayList);
            recyclerView.setAdapter(adapter);

            adapter.setOnItemClickListener((imageView, path) -> startActivity(new Intent(ProductActivity.this,
ImageViewActivity.class).putExtra("image", path), ActivityOptions.makeSceneTransitionAnimation(ProductActivity.this,
imageView, "image").toBundle()));

            nameTextView.setText(name);
            priceTextView.setText("£" + price);
            collectionTextView.setText("Collection: " + category);
            descriptionTextView.setText(description);

            if (Integer.parseInt(quantity) == 0) {
                String[] quantityArray = new String[0];
                ArrayAdapter<String> dataAdapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item,
quantityArray);
                dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
                quantitySpinner.setAdapter(dataAdapter);

                Button addToBasketBtn = findViewById(R.id.addToBasketBtn);
                addToBasketBtn.setText("Out of stock");
                addToBasketBtn.setBackground(getDrawable(R.drawable.rectangle_8));
                addToBasketBtn.setOnClickListener(v -> {
                    Toast.makeText(this, "Out of stock", Toast.LENGTH_SHORT).show();
                });
            } else {
                String[] quantityArray = new String[Integer.parseInt(quantity)];
                for (int i = 0; i < Integer.parseInt(quantity); i++) {
                    quantityArray[i] = String.valueOf(i + 1);
                }
                ArrayAdapter<String> dataAdapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item,
quantityArray);
                dataAdapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
                quantitySpinner.setAdapter(dataAdapter);
```

Product Activity Part 3



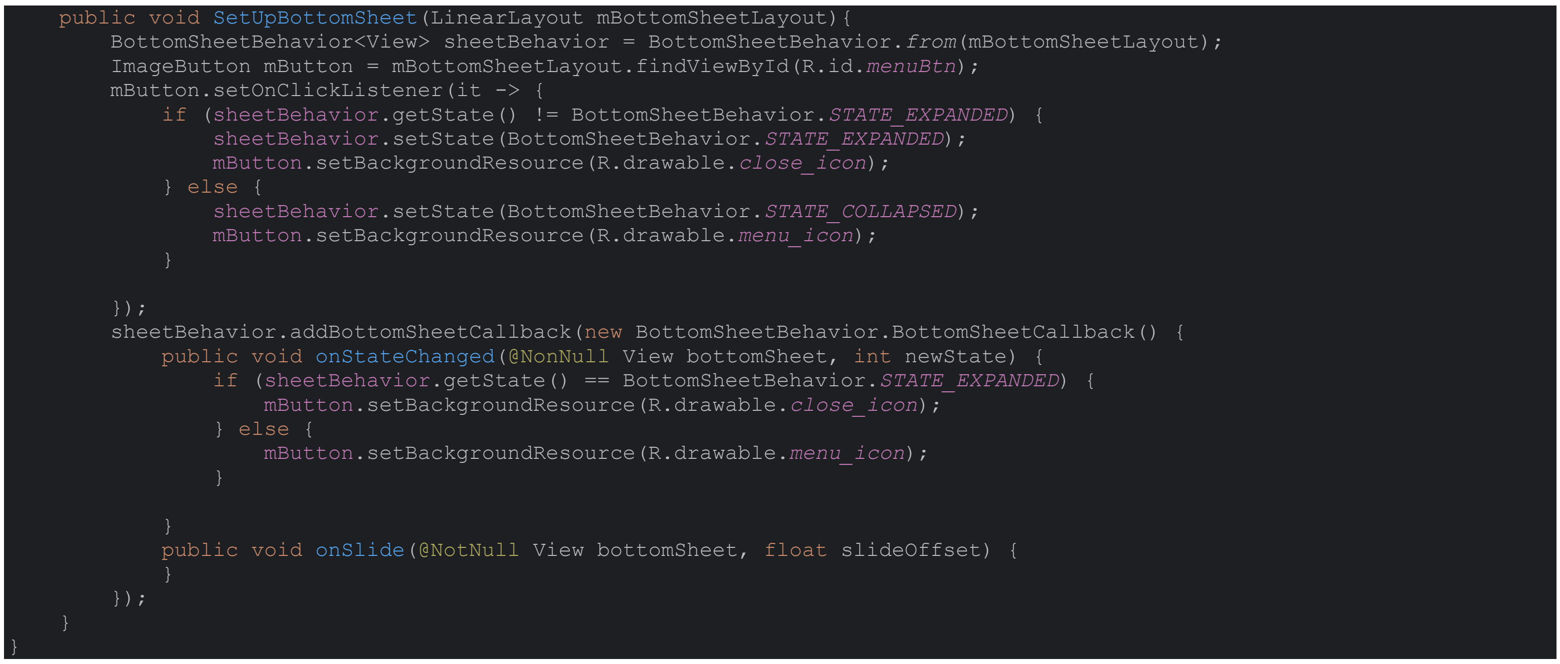
```java
                //add to basket button
                Button addToBasketBtn = findViewById(R.id.addToBasketBtn);
                addToBasketBtn.setOnClickListener(v -> {
                    String quantitySelected = quantitySpinner.getSelectedItem().toString();

                    RadioGroup radioGroupSize = findViewById(R.id.radioGroupSize);
                    int sizeID =  radioGroupSize.indexOfChild(findViewById(radioGroupSize.getCheckedRadioButtonId()));
                    String size;
                    switch (sizeID) {
                        case 0:
                            size = "XS";
                            break;
                        case 1:
                            size = "S";
                            break;
                        case 2:
                            size = "M";
                            break;
                        case 3:
                            size = "L";
                            break;
                        case 4:
                            size = "XL";
                            break;
                        case 5:
                            size = "XXL";
                            break;
                        default:
                            size = "M";
                            break;
                    }

                    ArrayList<String> productIDs = new ArrayList<>();
                    basketDB.openDatabaseConnection();
                    for (int i = 0; i < basketDB.getRowCount(); i++) {
                        for (String row : basketDB.getDataArray()) {
                            String[] rowArray = row.split(":");
                            productIDs.add(rowArray[1]);
                        }
                    }
                    if (!productIDs.contains(productID)) {
                        basketDB.insertRecords(productID, quantitySelected, price, size);
                        basketDB.closeDatabaseConnection();
                        addToBasketBtn.setText("Added to basket");
                        Toast.makeText(this, "Added to basket. Qty: " + quantitySelected, Toast.LENGTH_SHORT).show();
                    } else {
                        addToBasketBtn.setText("Item already in basket");
                        Toast.makeText(this, "Item already in basket", Toast.LENGTH_SHORT).show();
                        basketDB.closeDatabaseConnection();
                    }
                });
            }
        } else {
            Log.d("ProductActivity", "No such document");
        }
    } else {
        Log.d("ProductActivity", "get failed with ", task.getException());
    }
});


//backbtn
Button backBtn = findViewById(R.id.backBtn);
backBtn.setOnClickListener(v -> {
    finish();
});

//sign in button
final Button signInBtn = findViewById(R.id.SignInBtn);
signInBtn.setOnClickListener(v -> {
    this.startActivity(new Intent(this, LoginActivity.class));
});
```

Product Activity Part 4



```java
    //sign out button
    final Button signOutBtn = findViewById(R.id.SignOutBtn);
    signOutBtn.setOnClickListener(v -> {
        mAuth.signOut();
        onStart();
    });

    //account btn
    Button accountBtn = findViewById(R.id.accountBtn);
    accountBtn.setOnClickListener(v -> {
        startActivity(new Intent(ProductActivity.this, AccountActivity.class));
    });

    //basket btn signed out
    ImageButton basketBtnSignedOut = findViewById(R.id.basketBtnSignedOut);
    if(basketDB.getRowCount() > 0) {
        basketBtnSignedOut.setImageResource(R.drawable.cart_icon_full);
    }
    basketBtnSignedOut.setOnClickListener(v -> {
        this.startActivity(new Intent(this, BasketActivity.class));
    });

    //basket btn signed in
    ImageButton basketBtnSignedIn = findViewById(R.id.basketBtnSignedIn);
    if(basketDB.getRowCount() > 0) {
        basketBtnSignedIn.setImageResource(R.drawable.cart_icon_full);
    }
    basketBtnSignedIn.setOnClickListener(v -> {
        this.startActivity(new Intent(this, BasketActivity.class));
    });

    //contact btn signed out
    ImageButton contactBtnSignedOut = findViewById(R.id.contactBtnSignedOut);
    contactBtnSignedOut.setOnClickListener(v -> {
        this.startActivity(new Intent(this, ContactActivity.class));
    });

    //contact btn signed in
    ImageButton contactBtnSignedIn = findViewById(R.id.contactBtnSignedIn);
    contactBtnSignedIn.setOnClickListener(v -> {
        this.startActivity(new Intent(this, ContactActivity.class));
    });
}

public void onStart() {
    super.onStart();
    // Check if user is signed in (non-null) and update UI accordingly.
    FirebaseUser currentUser = mAuth.getCurrentUser();
    if(currentUser != null){
        LinearLayout mBottomSheetLayoutLoggedIn = findViewById(R.id.bottom_sheet_layout_logged_in);
        mBottomSheetLayoutLoggedIn.setVisibility(View.VISIBLE);

        LinearLayout mBottomSheetLayoutLoggedOut = findViewById(R.id.bottom_sheet_layout_logged_out);
        mBottomSheetLayoutLoggedOut.setVisibility(View.INVISIBLE);

        SetUpBottomSheet(mBottomSheetLayoutLoggedIn);
    } else {
        LinearLayout mBottomSheetLayoutLoggedOut = findViewById(R.id.bottom_sheet_layout_logged_out);
        mBottomSheetLayoutLoggedOut.setVisibility(View.VISIBLE);

        LinearLayout mBottomSheetLayoutLoggedIn = findViewById(R.id.bottom_sheet_layout_logged_in);
        mBottomSheetLayoutLoggedIn.setVisibility(View.INVISIBLE);

        SetUpBottomSheet(mBottomSheetLayoutLoggedOut);
    }
}
```

Product Activity Part 5

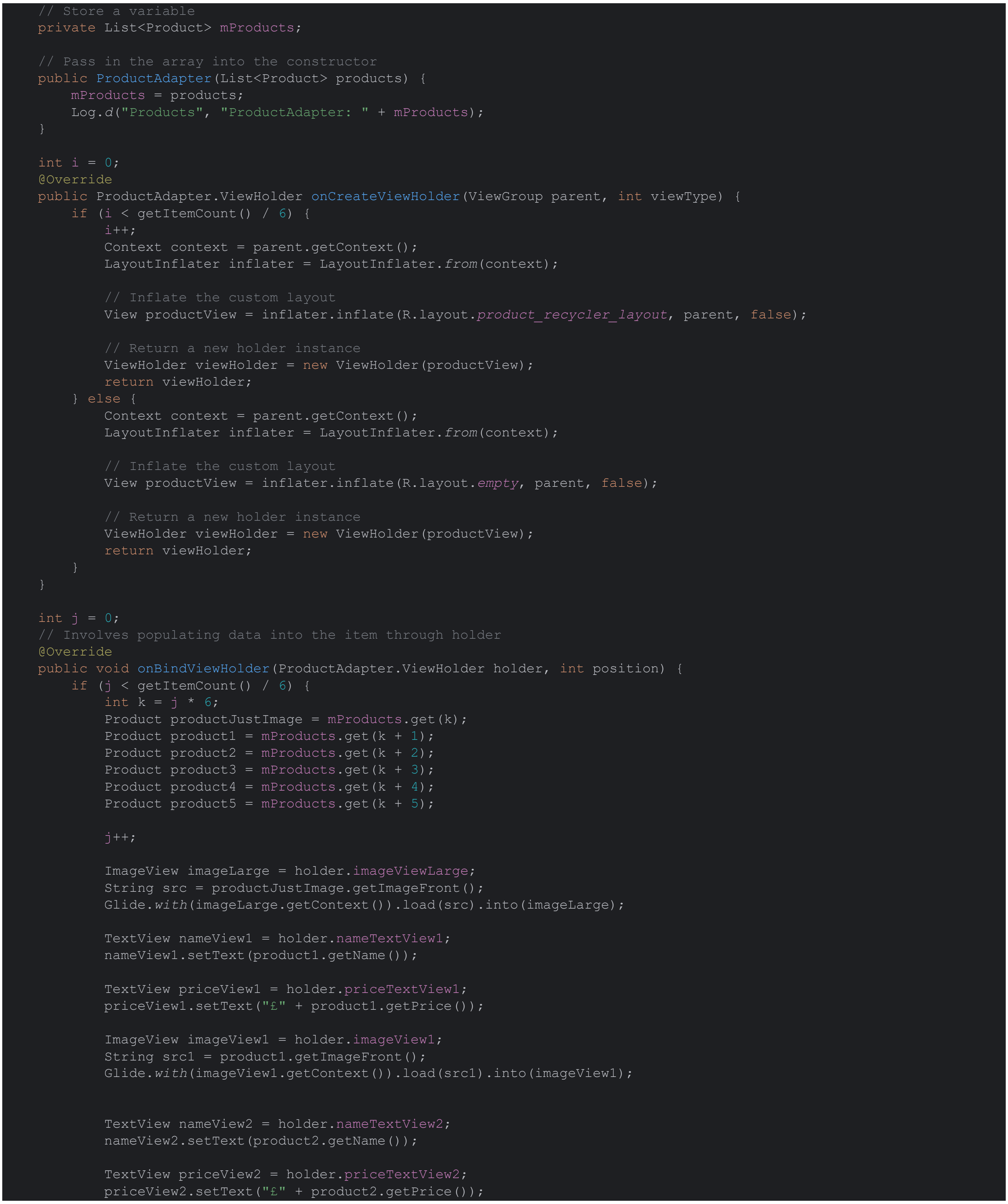


```java
    public void SetUpBottomSheet(LinearLayout mBottomSheetLayout){
        BottomSheetBehavior<View> sheetBehavior = BottomSheetBehavior.from(mBottomSheetLayout);
        ImageButton mButton = mBottomSheetLayout.findViewById(R.id.menuBtn);
        mButton.setOnClickListener(it -> {
            if (sheetBehavior.getState() != BottomSheetBehavior.STATE_EXPANDED) {
                sheetBehavior.setState(BottomSheetBehavior.STATE_EXPANDED);
                mButton.setBackgroundResource(R.drawable.close_icon);
            } else {
                sheetBehavior.setState(BottomSheetBehavior.STATE_COLLAPSED);
                mButton.setBackgroundResource(R.drawable.menu_icon);
            }

        });
        sheetBehavior.addBottomSheetCallback(new BottomSheetBehavior.BottomSheetCallback() {
            public void onStateChanged(@NonNull View bottomSheet, int newState) {
                if (sheetBehavior.getState() == BottomSheetBehavior.STATE_EXPANDED) {
                    mButton.setBackgroundResource(R.drawable.close_icon);
                } else {
                    mButton.setBackgroundResource(R.drawable.menu_icon);
                }

            }
            public void onSlide(@NotNull View bottomSheet, float slideOffset) {
            }
        });
    }
}
```

Product Adapter Part 1

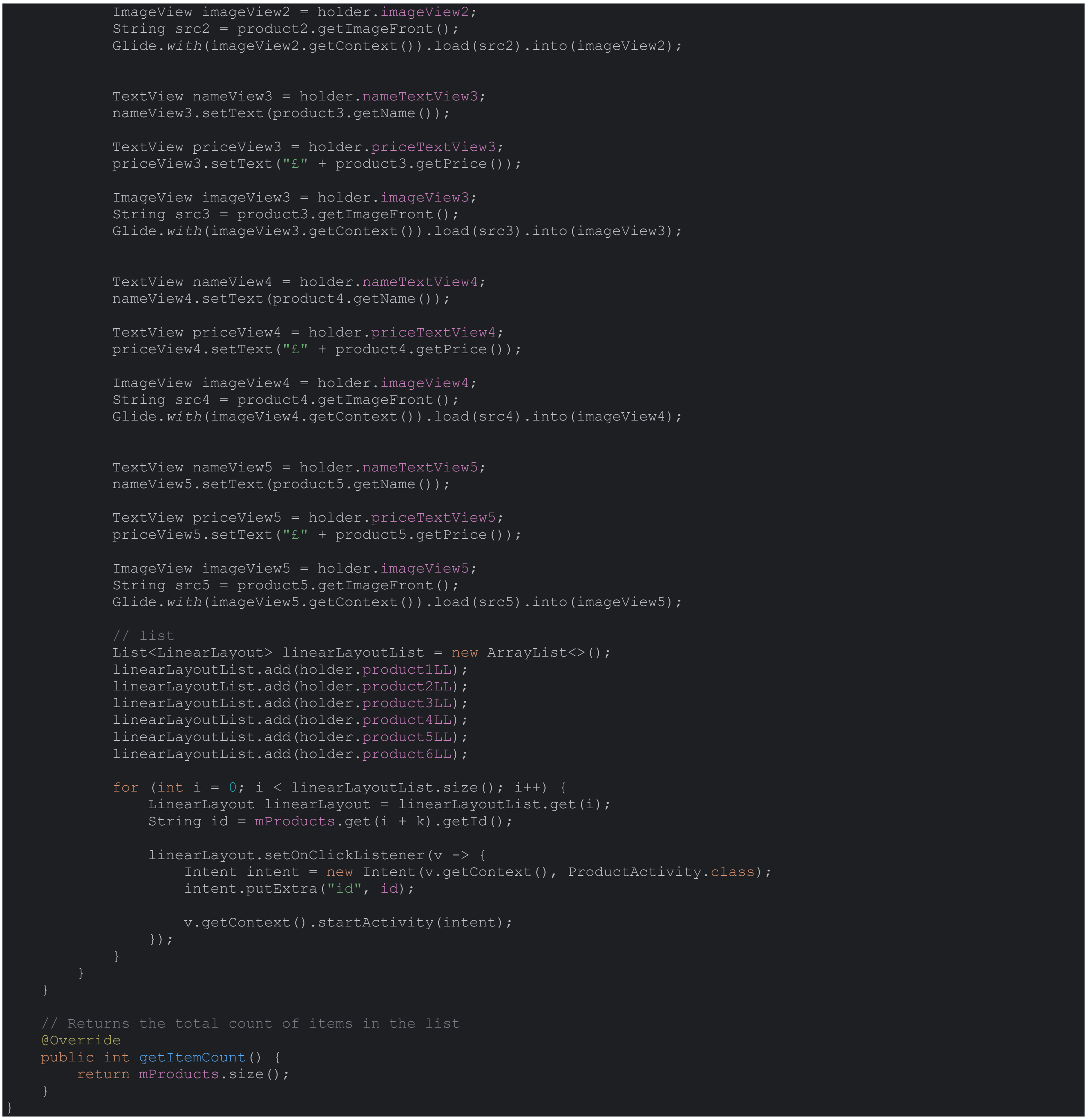
```java
package com.example.luckymerch;

import android.content.Context;
import android.content.Intent;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;

import androidx.recyclerview.widget.RecyclerView;

import com.bumptech.glide.Glide;

import java.util.ArrayList;
import java.util.List;

// Create the basic adapter extending from RecyclerView.Adapter
// Note that we specify the custom ViewHolder which gives us access to our views
public class ProductAdapter extends RecyclerView.Adapter<ProductAdapter.ViewHolder> {


    // Provide a direct reference to each of the views within a data item
    // Used to cache the views within the item layout for fast access
    public class ViewHolder extends RecyclerView.ViewHolder {
        // Your holder should contain a member variable
        // for any view that will be set as you render a row
        public TextView nameTextView1, nameTextView2, nameTextView3, nameTextView4, nameTextView5;
        public TextView priceTextView1, priceTextView2, priceTextView3, priceTextView4, priceTextView5;
        public ImageView imageViewLarge, imageView1, imageView2, imageView3, imageView4, imageView5;

        public LinearLayout product1LL, product2LL, product3LL, product4LL, product5LL, product6LL;


        // We also create a constructor that accepts the entire item row
        // and does the view lookups to find each subview
        public ViewHolder(View itemView) {
            // Stores the itemView in a public final member variable that can be used
            // to access the context from any ViewHolder instance.
            super(itemView);

            imageViewLarge = (ImageView) itemView.findViewById(R.id.imageViewLarge);


            nameTextView1 = (TextView) itemView.findViewById(R.id.nameText1);
            priceTextView1 = (TextView) itemView.findViewById(R.id.priceText1);
            imageView1 = (ImageView) itemView.findViewById(R.id.imageView1);

            nameTextView2 = (TextView) itemView.findViewById(R.id.nameText2);
            priceTextView2 = (TextView) itemView.findViewById(R.id.priceText2);
            imageView2 = (ImageView) itemView.findViewById(R.id.imageView2);

            nameTextView3 = (TextView) itemView.findViewById(R.id.nameText3);
            priceTextView3 = (TextView) itemView.findViewById(R.id.priceText3);
            imageView3 = (ImageView) itemView.findViewById(R.id.imageView3);

            nameTextView4 = (TextView) itemView.findViewById(R.id.nameText4);
            priceTextView4 = (TextView) itemView.findViewById(R.id.priceText4);
            imageView4 = (ImageView) itemView.findViewById(R.id.imageView4);

            nameTextView5 = (TextView) itemView.findViewById(R.id.nameText5);
            priceTextView5 = (TextView) itemView.findViewById(R.id.priceText5);
            imageView5 = (ImageView) itemView.findViewById(R.id.imageView5);

            product1LL = itemView.findViewById(R.id.product1);
            product2LL = itemView.findViewById(R.id.product2);
            product3LL = itemView.findViewById(R.id.product3);
            product4LL = itemView.findViewById(R.id.product4);
            product5LL = itemView.findViewById(R.id.product5);
            product6LL = itemView.findViewById(R.id.product6);
        }
    }
```

Product Adapter Part 2



```java
    // Store a variable
    private List<Product> mProducts;

    // Pass in the array into the constructor
    public ProductAdapter(List<Product> products) {
        mProducts = products;
        Log.d("Products", "ProductAdapter: " + mProducts);
    }

    int i = 0;
    @Override
    public ProductAdapter.ViewHolder onCreateViewHolder(ViewGroup parent, int viewType) {
        if (i < getItemCount() / 6) {
            i++;
            Context context = parent.getContext();
            LayoutInflater inflater = LayoutInflater.from(context);

            // Inflate the custom layout
            View productView = inflater.inflate(R.layout.product_recycler_layout, parent, false);

            // Return a new holder instance
            ViewHolder viewHolder = new ViewHolder(productView);
            return viewHolder;
        } else {
            Context context = parent.getContext();
            LayoutInflater inflater = LayoutInflater.from(context);

            // Inflate the custom layout
            View productView = inflater.inflate(R.layout.empty, parent, false);

            // Return a new holder instance
            ViewHolder viewHolder = new ViewHolder(productView);
            return viewHolder;
        }
    }

    int j = 0;
    // Involves populating data into the item through holder
    @Override
    public void onBindViewHolder(ProductAdapter.ViewHolder holder, int position) {
        if (j < getItemCount() / 6) {
            int k = j * 6;
            Product productJustImage = mProducts.get(k);
            Product product1 = mProducts.get(k + 1);
            Product product2 = mProducts.get(k + 2);
            Product product3 = mProducts.get(k + 3);
            Product product4 = mProducts.get(k + 4);
            Product product5 = mProducts.get(k + 5);

            j++;

            ImageView imageLarge = holder.imageViewLarge;
            String src = productJustImage.getImageFront();
            Glide.with(imageLarge.getContext()).load(src).into(imageLarge);

            TextView nameView1 = holder.nameTextView1;
            nameView1.setText(product1.getName());

            TextView priceView1 = holder.priceTextView1;
            priceView1.setText("£" + product1.getPrice());

            ImageView imageView1 = holder.imageView1;
            String src1 = product1.getImageFront();
            Glide.with(imageView1.getContext()).load(src1).into(imageView1);


            TextView nameView2 = holder.nameTextView2;
            nameView2.setText(product2.getName());

            TextView priceView2 = holder.priceTextView2;
            priceView2.setText("£" + product2.getPrice());
```

Product Adapter Part 3



```java
            ImageView imageView2 = holder.imageView2;
            String src2 = product2.getImageFront();
            Glide.with(imageView2.getContext()).load(src2).into(imageView2);


            TextView nameView3 = holder.nameTextView3;
            nameView3.setText(product3.getName());

            TextView priceView3 = holder.priceTextView3;
            priceView3.setText("£" + product3.getPrice());

            ImageView imageView3 = holder.imageView3;
            String src3 = product3.getImageFront();
            Glide.with(imageView3.getContext()).load(src3).into(imageView3);


            TextView nameView4 = holder.nameTextView4;
            nameView4.setText(product4.getName());

            TextView priceView4 = holder.priceTextView4;
            priceView4.setText("£" + product4.getPrice());

            ImageView imageView4 = holder.imageView4;
            String src4 = product4.getImageFront();
            Glide.with(imageView4.getContext()).load(src4).into(imageView4);


            TextView nameView5 = holder.nameTextView5;
            nameView5.setText(product5.getName());

            TextView priceView5 = holder.priceTextView5;
            priceView5.setText("£" + product5.getPrice());

            ImageView imageView5 = holder.imageView5;
            String src5 = product5.getImageFront();
            Glide.with(imageView5.getContext()).load(src5).into(imageView5);

            // list
            List<LinearLayout> linearLayoutList = new ArrayList<>();
            linearLayoutList.add(holder.product1LL);
            linearLayoutList.add(holder.product2LL);
            linearLayoutList.add(holder.product3LL);
            linearLayoutList.add(holder.product4LL);
            linearLayoutList.add(holder.product5LL);
            linearLayoutList.add(holder.product6LL);

            for (int i = 0; i < linearLayoutList.size(); i++) {
                LinearLayout linearLayout = linearLayoutList.get(i);
                String id = mProducts.get(i + k).getId();

                linearLayout.setOnClickListener(v -> {
                    Intent intent = new Intent(v.getContext(), ProductActivity.class);
                    intent.putExtra("id", id);

                    v.getContext().startActivity(intent);
                });
            }
        }
    }

    // Returns the total count of items in the list
    @Override
    public int getItemCount() {
        return mProducts.size();
    }
}
```

Register Activity Part 1

```java
package com.example.luckymerch;

import android.content.Intent;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.CheckBox;
import android.widget.DatePicker;
import android.widget.EditText;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import com.google.firebase.auth.FirebaseAuth;
import com.google.firebase.auth.FirebaseUser;
import com.google.firebase.firestore.FirebaseFirestore;

import java.sql.Timestamp;
import java.util.Calendar;
import java.util.HashMap;
import java.util.Map;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

public final class RegisterActivity extends AppCompatActivity {

    private FirebaseAuth mAuth;

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

        mAuth = FirebaseAuth.getInstance();

        //back btn
        final Button backBtn = findViewById(R.id.backBtn);
        backBtn.setOnClickListener(v -> {
            this.finish();
        });

        // btns and fields
        final Button registerBtn = findViewById(R.id.register);
        EditText emailField = findViewById(R.id.emailEditText);
        EditText fNameField = findViewById(R.id.fNameEditText);
        EditText lNameField = findViewById(R.id.lNameEditText);
        EditText passwordField = findViewById(R.id.passwordEditText);
        EditText rePasswordField = findViewById(R.id.rePasswordEditText);
        EditText phoneField = findViewById(R.id.phoneEditText);
        DatePicker DOBField = findViewById(R.id.DOBField);
        DOBField.setMaxDate(System.currentTimeMillis() - Long.parseLong("410240038000")); // current time - 13 years
        CheckBox contactPreferences = findViewById(R.id.contactPreferences);
        EditText addressLineOneField = findViewById(R.id.addressLineOneEditText);
        EditText addressLineTwoField = findViewById(R.id.addressLineTwoEditText);
        EditText townCityField = findViewById(R.id.townCityEditText);
        EditText countyField = findViewById(R.id.countyEditText);
        EditText postCodeField = findViewById(R.id.postCodeEditText);
        EditText nameOnCardField = findViewById(R.id.nameOnCardEditText);
        EditText cardNumberField = findViewById(R.id.cardNumberEditText);
        DatePicker expiryDateField = findViewById(R.id.expiryDateField);
        expiryDateField.setMinDate(System.currentTimeMillis());
        EditText cvvField = findViewById(R.id.cvvEditText);
```

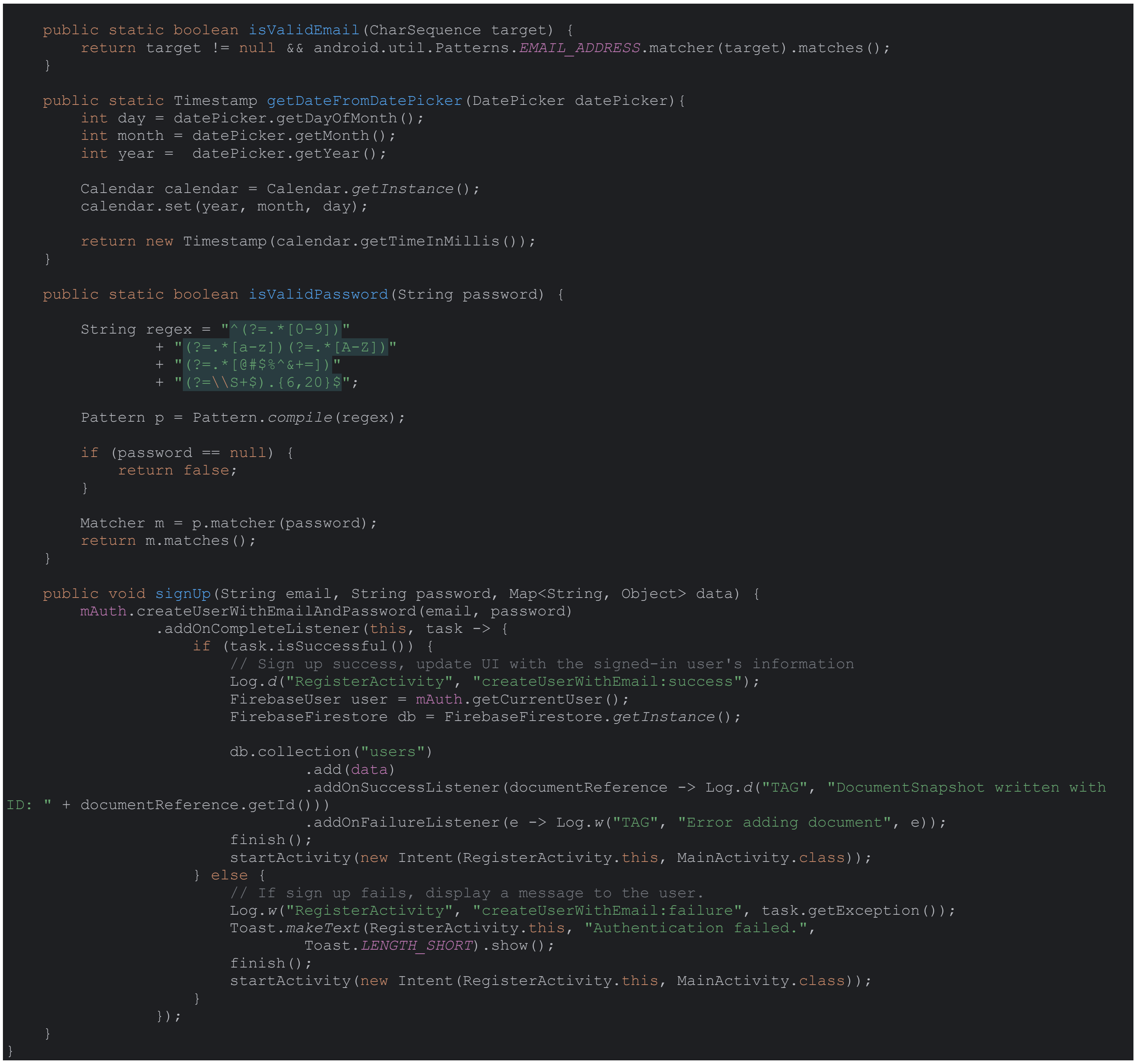
Register Activity Part 2

```
registerBtn.setOnClickListener(v -> {
    Map<String, Object> data = new HashMap<>();

    String email = emailField.getText().toString().trim();
    if (!isValidEmail(email)) {
        emailField.setError("Please enter a valid email");
        Toast.makeText(this, "Please enter a valid email", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the email is invalid
    } else {
        data.put("email", email);
    }

    String fName = fNameField.getText().toString().trim();
    if (fName.isEmpty()) {
        fNameField.setError("Please enter a first name");
        Toast.makeText(this, "Please enter a first name", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the first name is empty
    } else {
        data.put("fName", fName);
    }

    String lName = lNameField.getText().toString().trim();
    if (lName.isEmpty()) {
        lNameField.setError("Please enter a last name");
        Toast.makeText(this, "Please enter a last name", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the last name is empty
    } else {
        data.put("lName", lName);
    }

    String password = passwordField.getText().toString().trim();
    if (password.isEmpty()) {
        passwordField.setError("Please enter a password");
        Toast.makeText(this, "Please enter a password", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the password is empty
    }
    if (!isValidPassword(password)) {
        passwordField.setError("Password must be between 6 and 20 characters. Must include one uppercase letter, one
number, and one special character");
        Toast.makeText(this, "Password must be between 6 and 20 characters. Must include one uppercase letter, one
number, and one special character", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the password is not valid
    }

    String rePassword = rePasswordField.getText().toString().trim();
    if (rePassword.isEmpty()) {
        rePasswordField.setError("Please enter a password");
        Toast.makeText(this, "Please enter a password", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the password is empty
    }
    if (!password.equals(rePassword)) {
        rePasswordField.setError("Passwords do not match");
        Toast.makeText(this, "Passwords do not match", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the passwords do not match
    }

    String phoneStr = phoneField.getText().toString().trim();
    if (phoneStr.length() != 11) {
        phoneField.setError("Please enter a valid phone number");
        Toast.makeText(this, "Please enter a valid phone number", Toast.LENGTH_SHORT).show();
        return; // exit the method early if the phone number is empty
    } else {
        data.put("phone", phoneStr);
    }

    Timestamp DOB = getDateFromDatePicker(DOBField);
    data.put("DOB", DOB);

    Boolean contactPreferencesBool = contactPreferences.isChecked();
    data.put("contactPreferences", contactPreferencesBool);
```

Register Activity Part 3

```
        String addressLineOne = addressLineOneField.getText().toString().trim();
        if (addressLineOne.isEmpty()) {
            addressLineOneField.setError("Please enter an address");
            Toast.makeText(this, "Please enter an address", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the address is empty
        } else {
            data.put("addressLineOne", addressLineOne);
        }

        String addressLineTwo = addressLineTwoField.getText().toString().trim();
        if (!addressLineTwo.isEmpty()) {
            data.put("addressLineTwo", addressLineTwo);
        }

        String townCity = townCityField.getText().toString().trim();
        if (townCity.isEmpty()) {
            townCityField.setError("Please enter a town/city");
            Toast.makeText(this, "Please enter a town/city", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the town/city is empty
        } else {
            data.put("townCity", townCity);
        }

        String county = countyField.getText().toString().trim();
        if (!county.isEmpty()) {
            data.put("county", county);
        }

        String postCode = postCodeField.getText().toString().trim();
        if (postCode.isEmpty()) {
            postCodeField.setError("Please enter a post code");
            Toast.makeText(this, "Please enter a post code", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the post code is empty
        } else {
            data.put("postCode", postCode);
        }

        String nameOnCard = nameOnCardField.getText().toString().trim();
        if (nameOnCard.isEmpty()) {
            nameOnCardField.setError("Please enter a name on card");
            Toast.makeText(this, "Please enter a name on card", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the name on card is empty
        } else {
            data.put("nameOnCard", nameOnCard);
        }

        Long cardNumberInt = Long.parseLong(cardNumberField.getText().toString().trim());
        String cardNumberStr = cardNumberField.getText().toString().trim();
        if (cardNumberStr.length() != 16) {
            cardNumberField.setError("Please enter a valid card number");
            Toast.makeText(this, "Please enter a valid card number", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the card number is empty
        } else {
            data.put("cardNumber", cardNumberInt);
        }

        Timestamp expiryDate = getDateFromDatePicker(expiryDateField);
        data.put("expiryDate", expiryDate);

        String cvvStr = cvvField.getText().toString().trim();
        if (cvvStr.length() != 3) {
            cvvField.setError("Please enter a valid CVV");
            Toast.makeText(this, "Please enter a valid CVV", Toast.LENGTH_SHORT).show();
            return; // exit the method early if the CVV is empty
        }

        signUp(email, password, data);
    });

}
```

Register Activity Part 4



```java
    public static boolean isValidEmail(CharSequence target) {
        return target != null && android.util.Patterns.EMAIL_ADDRESS.matcher(target).matches();
    }

    public static Timestamp getDateFromDatePicker(DatePicker datePicker){
        int day = datePicker.getDayOfMonth();
        int month = datePicker.getMonth();
        int year =  datePicker.getYear();

        Calendar calendar = Calendar.getInstance();
        calendar.set(year, month, day);

        return new Timestamp(calendar.getTimeInMillis());
    }

    public static boolean isValidPassword(String password) {

        String regex = "^(?=.*[0-9])"
                + "(?=.*[a-z])(?=.*[A-Z])"
                + "(?=.*[@#$%^&+=])"
                + "(?=\\S+$).{6,20}$";

        Pattern p = Pattern.compile(regex);

        if (password == null) {
            return false;
        }

        Matcher m = p.matcher(password);
        return m.matches();
    }

    public void signUp(String email, String password, Map<String, Object> data) {
        mAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(this, task -> {
                    if (task.isSuccessful()) {
                        // Sign up success, update UI with the signed-in user's information
                        Log.d("RegisterActivity", "createUserWithEmail:success");
                        FirebaseUser user = mAuth.getCurrentUser();
                        FirebaseFirestore db = FirebaseFirestore.getInstance();

                        db.collection("users")
                                .add(data)
                                .addOnSuccessListener(documentReference -> Log.d("TAG", "DocumentSnapshot written with 
ID: " + documentReference.getId()))
                                .addOnFailureListener(e -> Log.w("TAG", "Error adding document", e));
                        finish();
                        startActivity(new Intent(RegisterActivity.this, MainActivity.class));
                    } else {
                        // If sign up fails, display a message to the user.
                        Log.w("RegisterActivity", "createUserWithEmail:failure", task.getException());
                        Toast.makeText(RegisterActivity.this, "Authentication failed.",
                                Toast.LENGTH_SHORT).show();
                        finish();
                        startActivity(new Intent(RegisterActivity.this, MainActivity.class));
                    }
                });
    }
}
```