

**VISVESVARAYA TECHNOLOGICAL
UNIVERSITY BELAGAVI-590018**



**A PROJECT REPORT
ON
CHOICE MART**

BY

SATHVIK B RAO

4SF20CS114

DINAKARA

4SF21CS405

In the partial fulfillment of the requirement for VI Sem. B. E. (CSE)

**MOBILE APPLICATION DEVELOPMENT
LABORATORY WITH MINIPROJECT (18CSMP68)**

Under the guidance of

Ms. PRAPULLA G

Assistant Professor, Dept. of CS&E



**Department of Computer Science & Engineering
SAHYADRI COLLEGE OF ENGINEERING & MANAGEMENT**

Adyar, Mangaluru-575007

2022-23

SAHYADRI
COLLEGE OF ENGINEERING & MANAGEMENT
(Affiliated to Visvesvaraya Technological University, BELAGAVI)
Adyar, Mangaluru – 575007

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the project entitled “**CHOICEMART**” is submitted in partial fulfillment for the requirement of VI sem. B. E. (Computer Science & Engineering), “**MOBILE APPLICATION DEVELOPMENT LABORATORY WITH MINI PROJECT**” during the year 2022 – 23 is a result of bonafide work carried out by,

SATHVIK B RAO

4SF20CS114

DINAKARA

4SF21CS405

.....
Ms. PRAPULLA G

Asst. Prof, Dept. of CS&E

SCEM, MANGALURU

.....
Dr. NAGESH H R

HOD, Dept. of CS&E

SCEM, MANGALURU

Signature of the Examiners

1.

2.

ABSTRACT

The ChoiceMart Android project is an ecommerce mobile application developed using Android Studio and Firebase. The app provides a platform for users to explore and purchase a wide range of products conveniently. It integrates various APIs, including Razorpay and Fakestore API, to enable secure payment transactions and offer a diverse selection of products.

The project aims to deliver a user-friendly and visually appealing interface, allowing customers to browse through different categories and discover products of their interest. Users can create an account, sign in securely, and manage their profile information within the app. The integration with Firebase ensures seamless authentication, data storage, and real-time updates.

ChoiceMart leverages the power of Razorpay API to facilitate secure and reliable payment processing. Users can add products to their cart, proceed to checkout, and complete transactions using different payment methods supported by Razorpay, such as credit/debit cards, net banking, and digital wallets.

To provide a wide variety of products, the app utilizes the Fakestore API, which serves as a source of product data. This integration allows users to explore a diverse catalog of products, view detailed product information, and make informed purchase decisions.

Overall, the ChoiceMart Android project combines the features of an ecommerce platform with the capabilities of Android Studio and Firebase. By integrating APIs like Razorpay and Fakestore, it offers a secure, convenient, and extensive shopping experience for users, making it a comprehensive solution for ecommerce on mobile devices.

ACKNOWLEDGEMENT

Before we get in-depth with the project, we want to include few expressions of appreciation for the people who has been a part of this project from its inception. The written work of this project has been one of the huge academic challenges we have faced and without the help, patience and guidance of the people involved, this assignment would not have been completed satisfactorily.

It gives us immense pleasure in presenting this project report on **CHOICE MART**. It has been our privilege to have a project guide who had assisted us from the commencement of this project. The success of this project is a sheer diligent work, and determination put in by us with the help of our project guide.

We hereby take this chance to include a special note of much obliged for guide **Ms.Prapulla G**, Assistant Professor, and Department of Computer Science who guided us in our project.

We are additionally grateful to **Dr. Nagesh H R**. Head of the Department, Computer Science and Engineering for furnishing us with the correct academic atmosphere in the department, whose encouragement and support made our entire undertaking appreciable.

We are extremely thankful to our beloved Principal **Dr. Rajesha S** for encouraging us to come up with new ideas and to express them in a systematic manner. We would also like to thank all our non-teaching staffs who also were very much supportive to us in building this project.

Last but not the least we want to extend our gratitude to our parents for their continuous love and support for which we are always indebted to them and also thank each and every one of those who helped or provided a helping hand in this project to be carried out.

SATHVIK B RAO(4SF20CS114)

DINAKARA (4SF21CS405)

PAGE INDEX

Chapter No.	Topic	Page No.
1.	Introduction 1.1 Objective 1.2 Introduction to Android Studio 1.3 Introduction to Firebase 1.4 Project Overview	1-2
2.	System Requirements 2.1 Project Requirements 2.2 Hardware Requirements 2.3 Software Requirements	3
3	System Design 3.1 Basic Layout 3.2 Flowchart	4-5
4	Implementation 4.1 Algorithm – Introduction 4.2 Functionality 4.2.1 User Login Page (JAVA) 4.2.2 User Sign Up Page (JAVA) 4.2.3 Admin Add product Page (JAVA) 4.2.4 User Home Page (JAVA) 4.2.5 Cart Page (JAVA) 4.2.6 Payment Page (JAVA)	6-21
5	Results 5.1 User Login Page 5.2 User Sign Up Page 5.3 User Home Page 5.4 Cart Page 5.5 Product Detail Page 5.6 Buy Now Page 5.7 Payment Detail Page 5.8 Orders Page	22-29
6	Conclusion References	30

LIST OF FIGURES

Figure No.	Title	Page No.
3.2.1.2	Flow Diagram – Staff	5
5.1.1	User Login Page	22
5.2.1	User Sign Up Page	23
5.3.1	User Home Page	24
5.4.1	Product Detail page	25
5.5.1	Cart Page	26
5.6.1	Buy now page	27
5.7.1	Payment Detail Page	28

CHAPTER 1

INTRODUCTION

1.1 OBJECTIVE

Choicemart aims to provide a seamless and satisfactory experience to its customers. This involves offering a user-friendly interface, a wide selection of products, competitive pricing, and efficient delivery services. Choicemart strives to maintain its position as a leading e-commerce platform in India and compete on a global scale. Choicemart the importance of its sellers and aims to empower them by providing a reliable and transparent marketplace. The company offers tools, resources, and support to help sellers grow their businesses and reach a wider customer base. Choicemart aims to expand its reach and penetrate into different segments of the market. Choicemart is committed to leveraging technology and driving innovation in the e-commerce industry. The company invests in research and development to enhance its platform, improve customer experience, and optimize logistics and supply chain operations.

1.2 INTRODUCTION TO ANDROID STUDIO

Android Studio is an authority incorporated advancement condition (IDE) for an android application improvement, in the light of the android Studio is planned explicitly for the android improvement accessibility for the download on Windows & Linux, android application being Google's essential development environment for the local android app advancement. Android Studio also offers adaptable Gradle-based form framework, the code formats to enable the user to fabricate the regular android app highlights & the rich format editorial manager with the help for the intuitive topic altering & also worked help for the Google Cloud Platform, making it simple to coordinate with the Google Cloud Messaging android app Engine & considerably more android Studio as good as ever interface structure point of view where you can see the interface you are taking a shot at and its related segments. android Studio give away different UI instruments to help you with making designs, executing style subjects, & making realistic or content assets for your android application. The android manufacture framework the toolbox you use to assemble, test, run & bundle your android applications. The construct framework can keep running as a coordinated android application from the android studio menu & freely from the direction line.

1.3 INTRODUCTION TO FIREBASE

Firebase is a Backend-as-a-Service (BaaS) which started as a YC11 start-up. It grew up into a next-generation app-development platform on Google Cloud Platform. Firebase is a real-time database that allows storing a list of objects in the form of a tree. We can synchronize data between different devices. It is a software which allows developers to develop Android, IOS, and Web apps. For reporting and fixing app crashes, tracking analytics, creating marketing and product experiments, firebase provides several tools. Firebase has three main services, i.e., a real-time database, user authentication, and hosting. Firebase evolved from Envolv. Envolv is a prior start-up founded by James Tamplin and Andrew Lee in 2011. Firebase Real-time Database was the first product of firebase. It is an API which syncs application data across Android, iOS, and Web devices. It gets stored on Firebase's cloud. Then the firebase real-time database helps the developers to build real-time, collaborative applications. Firebase manages real-time data in the database. So, it easily and quickly exchanges the data to and from the database. Hence, for developing mobile apps such as live streaming, chat messaging, etc., Firebase allows syncing real-time data across all devices - iOS, Android, and Web - without refreshing the screen.

1.4 PROJECT OVERVIEW

The The necessity of Choicemart arises from the need to address the challenges faced by both users and traditional grocery shopping methods. Users often encounter difficulties in finding time to visit physical stores, browsing through aisles, and carrying heavy grocery bags. Additionally, the lack of personalized recommendations and the inconvenience of payment processes further hinder the shopping experience.

By developing Choicemart, a user-centric grocery e-commerce application, these challenges can be overcome. The project aims to provide users with a convenient and hassle-free shopping experience by offering an extensive range of grocery products and services online. The application's easy-to-navigate interface ensures a smooth shopping experience, allowing users to browse through products, add them to their carts, and complete purchases securely using a payment gateway.

CHAPTER 2

REQUIREMENT SPECIFICATION

2.1 PROJECT REQUIRMENTS

The package is designed such that users with an Android phone having minimum configuration can also use it. It does not require complex computing.

The App requires a simple daily use android phone in which this app can be installed and then user can run it. For a developer it is required to install android studio or IntelliJ Application software which is used to design the app using Xml and Java. SQLite database is used as backend to store the data.

2.2 HARDWARE REQUIRMENTS

- Hard Disk: 500 GB
- Processor: Ryzen 9 5000Hx
- RAM: 16 GB
- Graphics: RADEON / NVIDIA RTX
- Monitor resolution - A colour monitor with a minimum resolution

2.3 SOFTWARE REQUIRMENTS

- Operating System: Windows 10
- User Interface: XML
- Programming Language: Java
- Application: Android Studio / IntelliJ
- Database: Firebase(Real Time DataBase)

CHAPTER 3

SYSTEM DESIGN

3.1 BASIC LAYOUT

1. User login page
2. User home page
3. Products (Category) page
4. Order and Payment page
5. Cart & Buy page
6. Admin login page
7. Admin home page
8. Add Product Page

3.2 ARCHITECTURE

The flow diagram shown in Figure 3.2.1 illustrates the step-by-step process of using the app. It begins with the user opening the app, which then displays the login page. If the user already has an account, they can proceed to enter their registered username and password to login. However, if they are new users, they have the option to go to the Signup page and create a new account. Once successfully logged in, users are directed to the home screen, where they can access various features and functionalities.

If users encounter any issues or need to report a problem, they can click on the plus button, which allows them to add the required details and upload any relevant files. For quick access to emergency numbers, the app provides a contacts page where users can find important contact numbers. By clicking on a specific number, the app opens the default calling app on the user's device with the number already dialed for immediate assistance.

Additionally, the app offers a news page that users can navigate to in order to stay updated with the latest news and information. This feature provides a convenient way for users to access current news within the app's interface.

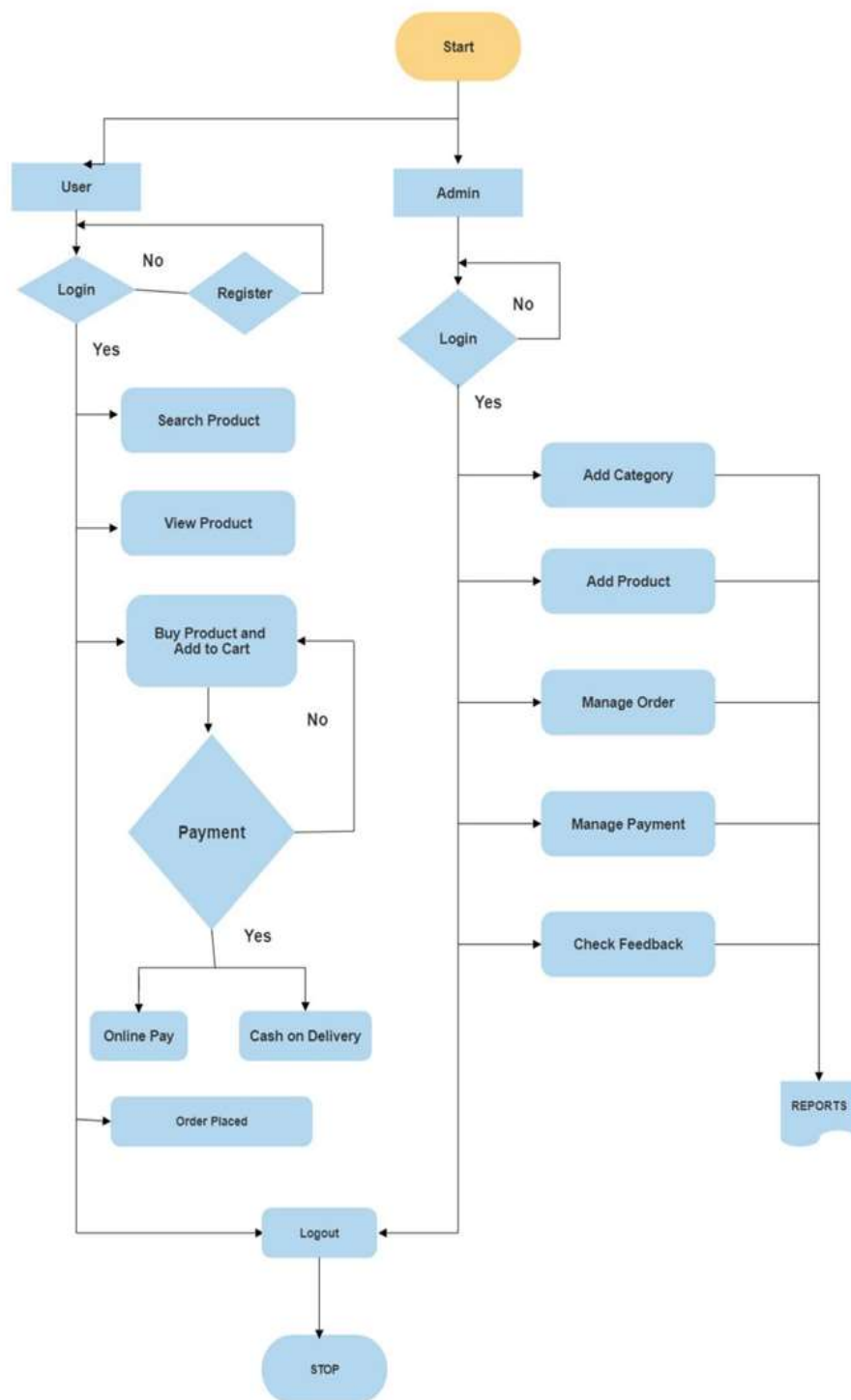


Fig 3.2.1.1 Flow Diagram

CHAPTER 4

IMPLEMENTATION

4.1 ALGORITHM – INTRODUCTION

Recommendation Algorithms: These algorithms analyze user data, such as browsing history, purchase history, and demographic information, to provide personalized product recommendations. They are commonly used to suggest products to customers based on their preferences and behavior patterns.

Search Algorithms: E-commerce platforms often employ search algorithms to help users find products efficiently. These algorithms take into account various factors like relevance, popularity, and user preferences to display the most relevant search results.

Pricing Algorithms: Pricing algorithms are used by e-commerce businesses to dynamically adjust product prices based on factors like demand, competition, inventory levels, and customer behavior. These algorithms aim to optimize pricing strategies for maximizing sales and profits.

Fraud Detection Algorithms: E-commerce platforms employ fraud detection algorithms to identify and prevent fraudulent activities such as unauthorized transactions, fake accounts, and identity theft. These algorithms analyze patterns, anomalies, and user behavior to flag potentially fraudulent activities.

Inventory Management Algorithms: Efficient inventory management is crucial for e-commerce businesses. Inventory management algorithms help in optimizing inventory levels, reordering products, and forecasting demand based on historical sales data, seasonality, and other factors.

Ad Targeting Algorithms: E-commerce platforms often use algorithms to target and personalize advertisements to individual users based on their browsing and purchase history, demographics, and interests. These algorithms aim to deliver relevant ads to potential customers, increasing the likelihood of conversions.

4.2 FUNCTIONALITY

4.2.1 USER LOGIN PAGE (JAVA)

```

class SignupActivity extends AppCompatActivity {
    EditText nameEditText
    EditText phoneEditText
    EditText passwordEditText
    Button signupButton
    DatabaseReference databaseReference

    method onCreate(savedInstanceState: Bundle)
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_signup)

    // Initialize views
    nameEditText = findViewById(R.id.nameEditText)
    phoneEditText = findViewById(R.id.phoneEditText)
    passwordEditText = findViewById(R.id.passwordEditText)
    signupButton = findViewById(R.id.signupButton)

    // Initialize Firebase database reference
    databaseReference = FirebaseDatabase.getInstance().getReference().child("Users")

    // Set click listener for signup button
    signupButton.setOnClickListener(new View.OnClickListener() {
    method onClick(v: View)
    signUp()
    })

    method signUp()
    // Get user input values
    String name = nameEditText.getText().toString().trim()
    String phone = phoneEditText.getText().toString().trim()
    String password = passwordEditText.getText().toString().trim()

    // Validate input values
    if (TextUtils.isEmpty(name) or TextUtils.isEmpty(phone) or TextUtils.isEmpty(password)) then
    Toast.makeText(this, "Please fill in all fields", Toast.LENGTH_SHORT).show()
    return
    end if
    if (!TextUtils.isDigitsOnly(phone)) then
    Toast.makeText(this, "Phone number should only contain numbers",
    Toast.LENGTH_SHORT).show()
    return
    end if

    // Check if user already exists
    databaseReference.child(phone).addListenerForSingleValueEvent(new ValueEventListener() {
    method onDataChange(dataSnapshot: DataSnapshot)
    if (dataSnapshot.exists()) then

```

```

Toast.makeText(SignupActivity.this, "User already exists. Please log in.",
    Toast.LENGTH_SHORT).show()
else
// Create user object
User user = new User(name, phone, password)

// Save user to Firebase database
databaseReference.child(phone).setValue(user).addOnCompleteListener(new
    OnCompleteListener<Void>() {
        method onComplete(task: Task<Void>)
        if (task.isSuccessful()) then
            Toast.makeText(SignupActivity.this, "User registered successfully",
                Toast.LENGTH_SHORT).show()
            Intent intent = new Intent(SignupActivity.this, HomePage.class)
            UserData.getInstance().setUserName(name)
            intent.putExtra("userId", phone)
            startActivity(intent)
            // Perform any additional actions after successful registration
        else
            Toast.makeText(SignupActivity.this, "Registration failed. Please try again.",
                Toast.LENGTH_SHORT).show()
        end if
    })
end if
})

method onCancelled(databaseError: DatabaseError)
Log.e("SignupActivity", "Database error: " + databaseError.getMessage())
end method
})
}
}

```

4.2.2 USER SIGNUP PAGE (JAVA)

```

class LoginActivity extends AppCompatActivity {
    EditText phoneEditText
    EditText passwordEditText
    TextView adminTextView
    TextView notAdminTextView
    Button loginButton
    boolean isAdmin = false

    method onCreate(savedInstanceState: Bundle)
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_login)

    // Initialize views
    phoneEditText = findViewById(R.id.phoneEditText)
    passwordEditText = findViewById(R.id.passwordEditText)
    adminTextView = findViewById(R.id.adminTextView)
    notAdminTextView = findViewById(R.id.notAdminTextView)
}

```

```

loginButton = findViewById(R.id.loginButton)
// Set click listener for adminTextView
adminTextView.setOnClickListener(new View.OnClickListener() {
method onClick(v: View)
isAdmin = true
adminTextView.setVisibility(View.GONE)
notAdminTextView.setVisibility(View.VISIBLE)
})

// Set click listener for notAdminTextView
notAdminTextView.setOnClickListener(new View.OnClickListener() {
method onClick(v: View)
isAdmin = false
notAdminTextView.setVisibility(View.GONE)
adminTextView.setVisibility(View.VISIBLE)
})

// Set click listener for loginButton
loginButton.setOnClickListener(new View.OnClickListener() {
method onClick(v: View)
login()
})

method login()
String phoneNumber = phoneEditText.getText().toString().trim()
String password = passwordEditText.getText().toString().trim()

// Check if the phone number contains only numbers
if (!phoneNumber.matches("\\d+")) then
Toast.makeText(this, "Please enter a valid phone number", Toast.LENGTH_SHORT).show()
return
end if

// Perform the login process based on admin status
if (isAdmin) then
DatabaseReference adminRef = FirebaseDatabase.getInstance().getReference().child("Admin")

if (phoneNumber.isEmpty() || password.isEmpty()) then
Toast.makeText(this, "Invalid admin credentials", Toast.LENGTH_SHORT).show()
else
adminRef.child(phoneNumber).addListenerForSingleValueEvent(new ValueEventListener() {
method onDataChange(dataSnapshot: DataSnapshot)
if (dataSnapshot.exists()) then
String storedPassword = dataSnapshot.child("password").getValue(String.class)

if (password.equals(storedPassword)) then
Toast.makeText(LoginActivity.this, "Admin login successful", Toast.LENGTH_SHORT).show()
Intent intent = new Intent(LoginActivity.this, AdminHomePage.class)
intent.putExtra("phone", phoneNumber)
startActivity(intent)
else
Toast.makeText(LoginActivity.this, "Invalid admin credentials",
Toast.LENGTH_SHORT).show()
end if
else

```

```

Toast.makeText(LoginActivity.this, "Invalid admin credentials",
Toast.LENGTH_SHORT).show()
end if
end method

method onCancelled(databaseError: DatabaseError)
// Error occurred while fetching admin data
Log.e("LoginActivity", "Database error: " + databaseError.getMessage())
end method
})
end if
else
DatabaseReference usersRef = FirebaseDatabase.getInstance().getReference().child("Users")

if (phoneNumber.isEmpty() || password.isEmpty()) then
Toast.makeText(this, "Invalid user credentials", Toast.LENGTH_SHORT).show()
else
usersRef.child(phoneNumber).addListenerForSingleValueEvent(new ValueEventListener() {
method onDataChange(dataSnapshot: DataSnapshot)
if (dataSnapshot.exists()) then
String storedPassword = dataSnapshot.child("password").getValue(String.class)
String username = dataSnapshot.child("name").getValue(String.class)
System.out.println(username)

if (password.equals(storedPassword)) then
Toast.makeText(LoginActivity.this, "User login successful", Toast.LENGTH_SHORT).show()
Intent intent = new Intent(LoginActivity.this, HomePage.class)
UserData.getInstance().setUserName(username)
intent.putExtra("userId", phoneNumber)
startActivity(intent)
else
Toast.makeText(LoginActivity.this, "Invalid user credentials", Toast.LENGTH_SHORT).show()
end if
else
Toast.makeText(LoginActivity.this, "Account does not exist", Toast.LENGTH_SHORT).show()
end if
end method

method onCancelled(databaseError: DatabaseError)
// Error occurred while fetching user data
Log.e("LoginActivity", "Database error: " + databaseError.getMessage())
end method
})
end if
end if
end method
}

```


4.2.3 ADMIN ADD PRODUCT PAGE (JAVA)

```

class AdminAddProduct extends AppCompatActivity {
static final int PICK_IMAGE_REQUEST = 1
Uri imageUri
ImageView productImage
EditText productName, productPrice, productDescription, productCategory
Button addProductButton
DatabaseReference productsRef
StorageReference productImageRef

method onCreate(savedInstanceState: Bundle)
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_admin_add_product)

// Initialize the Firebase database and storage references
productsRef = FirebaseDatabase.getInstance().getReference().child("Products")
productImageRef = FirebaseStorage.getInstance().getReference().child("ProductImages")

productImage = findViewById(R.id.admin_product_image)
productName = findViewById(R.id.admin_product_name)
productPrice = findViewById(R.id.admin_product_price)
productDescription = findViewById(R.id.admin_product_description)
productCategory = findViewById(R.id.admin_product_category)
addProductButton = findViewById(R.id.addAdminProductBtn)

productImage.setOnClickListener(view -> openImageGallery())

addProductButton.setOnClickListener(view -> {
if (imageUri != null) then
uploadImageAndProduct()
else
Toast.makeText(this, "Please select an image", Toast.LENGTH_SHORT).show()
end if
})

method openImageGallery()
Intent intent = new Intent()
intent.setType("image/*")
intent.setAction(Intent.ACTION_GET_CONTENT)
startActivityForResult(intent, PICK_IMAGE_REQUEST)

method uploadImageAndProduct()
String name = productName.getText().toString()
String price = productPrice.getText().toString()
String description = productDescription.getText().toString()
String category = productCategory.getText().toString()

// Generate a unique key for the product
DatabaseReference categoryRef = productsRef.child(category)
DatabaseReference productRef = categoryRef.push()
String productKey = productRef.getKey()

// Upload the image to Firebase Storage
StorageReference imageRef = productImageRef.child(productKey + ".jpg")
UploadTask uploadTask = imageRef.putFile(imageUri)

```

```

uploadTask.addOnCompleteListener(task -> {
    if (task.isSuccessful()) then
        // Image upload successful, get the download URL
        imageRef.getDownloadUrl().addOnSuccessListener(uri -> {
            // Save the image URL in the product details
            String imageUrl = uri.toString()
            productRef.child("ProductImage").setValue(imageUrl)

            // Save other product details
            productRef.child("ProductName").setValue(name)
            productRef.child("ProductPrice").setValue(price)
            productRef.child("ProductDescription").setValue(description)
            productRef.child("ProductCategory").setValue(category)

            Toast.makeText(this, "Product added successfully", Toast.LENGTH_SHORT).show()
            finish() // Finish the activity after adding the product
        }).addOnFailureListener(e -> {
            // Handle failure to get the image download URL
            Toast.makeText(this, "Failed to upload image: " + e.getMessage(), Toast.LENGTH_SHORT).show()
        })
    else
        // Handle failure to upload the image
        Toast.makeText(this, "Failed to upload image: " + task.getException().getMessage(),
            Toast.LENGTH_SHORT).show()
    end if
})

method onActivityResult(requestCode: int, resultCode: int, data: Intent)
super.onActivityResult(requestCode, resultCode, data)
if (requestCode == PICK_IMAGE_REQUEST && resultCode == RESULT_OK && data != null &&
    data.getData() != null) then
    imageUrl = data.getData()
    productImage.setImageURI(imageUri)
end if
end method
}

```

4.2.4 HOME PAGE (JAVA)

```

class AdminAddProduct extends AppCompatActivity {
static final int PICK_IMAGE_REQUEST = 1
Uri imageUri
ImageView productImage
EditText productName, productPrice, productDescription, productCategory
Button addProductButton
DatabaseReference productsRef
StorageReference productImageRef

method onCreate(savedInstanceState: Bundle)
super.onCreate(savedInstanceState)
setContentView(R.layout.activity_admin_add_product)

// Initialize the Firebase database and storage references
productsRef = FirebaseDatabase.getInstance().getReference().child("Products")
productImageRef = FirebaseStorage.getInstance().getReference().child("ProductImages")

productImage = findViewById(R.id.admin_product_image)
productName = findViewById(R.id.admin_product_name)
productPrice = findViewById(R.id.admin_product_price)
productDescription = findViewById(R.id.admin_product_description)
productCategory = findViewById(R.id.admin_product_category)
addProductButton = findViewById(R.id.addAdminProductBtn)

productImage.setOnClickListener(view -> openImageGallery())

addProductButton.setOnClickListener(view -> {
if (imageUri != null) then
uploadImageAndProduct()
else
Toast.makeText(this, "Please select an image", Toast.LENGTH_SHORT).show()
end if
})

method openImageGallery()
Intent intent = new Intent()
intent.setType("image/*")
intent.setAction(Intent.ACTION_GET_CONTENT)
startActivityForResult(intent, PICK_IMAGE_REQUEST)

method uploadImageAndProduct()
String name = productName.getText().toString()
String price = productPrice.getText().toString()
String description = productDescription.getText().toString()
String category = productCategory.getText().toString()

// Generate a unique key for the product
DatabaseReference categoryRef = productsRef.child(category)
DatabaseReference productRef = categoryRef.push()
String productKey = productRef.getKey()

// Upload the image to Firebase Storage
StorageReference imageRef = productImageRef.child(productKey + ".jpg")
UploadTask uploadTask = imageRef.putFile(imageUri)

```

```

uploadTask.addOnCompleteListener(task -> {
    if (task.isSuccessful()) then
        // Image upload successful, get the download URL
        imageRef.getDownloadUrl().addOnSuccessListener(uri -> {
            // Save the image URL in the product details
            String imageUrl = uri.toString()
            productRef.child("ProductImage").setValue(imageUrl)

            // Save other product details
            productRef.child("ProductName").setValue(name)
            productRef.child("ProductPrice").setValue(price)
            productRef.child("ProductDescription").setValue(description)
            productRef.child("ProductCategory").setValue(category)

            Toast.makeText(this, "Product added successfully", Toast.LENGTH_SHORT).show()
            finish() // Finish the activity after adding the product
        }).addOnFailureListener(e -> {
            // Handle failure to get the image download URL
            Toast.makeText(this, "Failed to upload image: " + e.getMessage(), Toast.LENGTH_SHORT).show()
        })
    else
        // Handle failure to upload the image
        Toast.makeText(this, "Failed to upload image: " + task.getException().getMessage(),
            Toast.LENGTH_SHORT).show()
    end if
})

method onActivityResult(requestCode: int, resultCode: int, data: Intent)
super.onActivityResult(requestCode, resultCode, data)
if (requestCode == PICK_IMAGE_REQUEST && resultCode == RESULT_OK && data != null &&
    data.getData() != null) then
    imageUrl = data.getData()
    productImage.setImageURI(imageUri)
end if
end method
}

```

4.2.5 CART PAGE (JAVA)

```

public class CartFragment extends Fragment {

    // Declare instance variables
    private LinearLayout cartProductsLayout;
    private TextView cartEmptyTextView;
    private List<Product> selectedProducts = new ArrayList<>();

    // Default constructor
    public CartFragment() {
        // Required empty public constructor
    }

    // Static factory method to create a new instance of CartFragment
    public static CartFragment newInstance() {
        return new CartFragment();
    }

    // Override the onCreateView method to inflate the layout for the fragment
    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState) {
        return inflater.inflate(R.layout.fragment_cart, container, false);
    }

    // Override the onViewCreated method to initialize the fragment's views and listeners
    @Override
    public void onViewCreated(@NonNull View view, @Nullable Bundle savedInstanceState) {
        super.onViewCreated(view, savedInstanceState);

        // Initialize views
        cartProductsLayout = view.findViewById(R.id.cartProductsLayout);
        cartEmptyTextView = view.findViewById(R.id.cartEmptyTextView);

        // Display cart products
        displayCartProducts();

        // Set onClickListener for the "Buy Now" button
        Button buyNowButton = view.findViewById(R.id.buy_now_button);
        buyNowButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Check if any products are selected
                if (selectedProducts.isEmpty()) {
                    Toast.makeText(requireContext(), "No products selected", Toast.LENGTH_SHORT).show();
                } else {
                    // Proceed to the payment activity
                    Intent intent = new Intent(requireContext(), PaymentActivity.class);
                    intent.putExtra("selectedProducts", (Serializable) selectedProducts);
                    startActivity(intent);
                }
            }
        });
    }
}

```

```

// Method to display cart products
private void displayCartProducts() {
// Retrieve the user's phone number
String phone = UserData.getInstance().getUserId();

// Get a reference to the Users node in the Firebase database
DatabaseReference usersRef = FirebaseDatabase.getInstance().getReference().child("Users");

// Query the database to find the user with the given phone number
usersRef.orderByChild("phone").equalTo(phone).addListenerForSingleValueEvent(new
 ValueEventListener() {
@Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
if (dataSnapshot.exists()) {
// User exists, get the user's ID
DataSnapshot userSnapshot = dataSnapshot.getChildren().iterator().next();
String userId = userSnapshot.getKey();

// Get a reference to the user's cart
DatabaseReference cartRef = usersRef.child(userId).child("Cart");

cartRef.addListenerForSingleValueEvent(new ValueEventListener() {
@Override
public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
if (dataSnapshot.exists()) {
// Iterate over the products in the cart
for (DataSnapshot productSnapshot : dataSnapshot.getChildren()) {
// Retrieve product details from the database snapshot
String productDescription = productSnapshot.child("ProductDescription").getValue(String.class);
String productImageUrl = productSnapshot.child("ProductImageUrl").getValue(String.class);
String productName = productSnapshot.child("ProductName").getValue(String.class);
String productPrice = productSnapshot.child("ProductPrice").getValue(String.class);

// Create a new product view and populate it with data
View productView = LayoutInflater.from(requireContext())
.inflate(R.layout.cart_product, cartProductsLayout, false);

// Set product details in the view
ImageView productImageView = productView.findViewById(R.id.product_image);
TextView productNameTextView = productView.findViewById(R.id.product_name);
TextView productPriceTextView = productView.findViewById(R.id.product_price);
TextView quantityTextView = productView.findViewById(R.id.quantity_text);

// Load product image using Glide or any other image loading library
Glide.with(requireContext())
.load(productImageUrl)
.into(productImageView);

// Set product name and price
productNameTextView.setText(productName);

// Convert the product price to the desired currency format
// and set it in the productPriceTextView

// Set up increment and decrement buttons for quantity

// Set up long click and click listeners for selecting/deselecting products

```

```

// Set up remove button listener to remove the product from the cart

// Add the product view to the cartProductsLayout
}
} else {
// Cart is empty, display appropriate message
cartEmptyTextView.setVisibility(View.VISIBLE);
}
}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
// Handle any errors that occur during data retrieval
}
});
} else {
// User does not exist or is not logged in, display cart empty message
cartEmptyTextView.setVisibility(View.VISIBLE);
}
}

@Override
public void onCancelled(@NonNull DatabaseError databaseError) {
// Handle any errors that occur during data retrieval
}
});
}

// Method to retrieve the Product object associated with a product view
private Product getProductFromView(View productView) {
return (Product) productView.getTag();
}
}

```

4.2.6 PAYMENT PAGE (JAVA)

```

public class PaymentActivity extends AppCompatActivity implements PaymentResultListener {
    private Checkout razorpayCheckout;
    private AutoReadOtpHelper autoReadOtpHelper;
    private String formattedTotalAmount;
    private NavController navController;
    private Product product = new Product();

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

        // Preload Checkout
        Checkout.preload(getApplicationContext());

        // Get the selected products from the intent
        List<Product> selectedProducts = (List<Product>) getIntent().getSerializableExtra("selectedProducts");

        if (selectedProducts != null && !selectedProducts.isEmpty()) {
            // Calculate the total price
            TextView productNameTextView = findViewById(R.id.productNameTextView);
            TextView quantityTextView = findViewById(R.id.quantityTextView);
            TextView totalPriceTextView = findViewById(R.id.totalPriceTextView);

            double totalAmount = 0.0;
            for (Product product : selectedProducts) {
                totalAmount += product.getPrice() * product.getQuantity();

                // Append product name, quantity, and total price to respective TextViews
                String productName = product.getName();
                if (productName != null) {
                    productNameTextView.append(productName.toString() + "\n\n");
                }
                int productQuantity = product.getQuantity();
                quantityTextView.append(String.valueOf(productQuantity) + "\n\n");
                totalPriceTextView.append(String.valueOf(product.getPrice() * product.getQuantity()) + "\n\n");
            }

            // Display the grand total amount
            TextView grandTotalTextView = findViewById(R.id.grandTotalTextView);
            DecimalFormat decimalFormat = new DecimalFormat("#0.00");
            formattedTotalAmount = decimalFormat.format(totalAmount);
            String grandTotalText = getString(R.string.rupee_symbol, formattedTotalAmount);
            grandTotalTextView.setText(grandTotalText);
        }

        // Set OnClickListener for the "Proceed to Checkout" button
        Button proceedToCheckoutButton = findViewById(R.id.proceedToCheckoutButton);
        proceedToCheckoutButton.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Retrieve the phone number from the EditText field
                EditText phoneNumberEditText = findViewById(R.id.shippment_phone_number);
                String phoneNumber = phoneNumberEditText.getText().toString();
            }
        });
    }
}

```



```

// Retrieve the address from the EditText field
EditText addressEditText = findViewById(R.id.addressEditText);
String address = addressEditText.getText().toString();

if (phoneNumber.isEmpty() || address.isEmpty()) {
    Toast.makeText(PaymentActivity.this, "Please enter the required information",
        Toast.LENGTH_SHORT).show();
    return;
} else {
    // Proceed with payment
    startPayment(formattedTotalAmount);
}
});
}

// Method to start the payment process
public void startPayment(String grandTotal) {
    Checkout checkout = new Checkout();
    checkout.setKeyID("rzp_test_5ljbfulRar9Z9P");
    checkout.setImage(R.drawable.applogo);

    final Activity activity = this;
    try {
        JSONObject options = new JSONObject();

        // Set payment options
        options.put("name", "Choice Mart");
        options.put("description", "Reference No. #123456");
        options.put("image", "https://s3.amazonaws.com/rzp-mobile/images/rzp.jpg");
        options.put("theme.color", "#3399cc");
        options.put("currency", "INR");
        options.put("amount", String.valueOf(Double.parseDouble(grandTotal) * 100));
        options.put("prefill.email", "gaurav.kumar@example.com");
        options.put("prefill.contact", "6360169868");

        JSONObject retryObj = new JSONObject();
        retryObj.put("enabled", true);
        retryObj.put("max_count", 4);
        options.put("retry", retryObj);

        checkout.open(activity, options);

    } catch (Exception e) {
        Log.e("TAG", "Error in starting Razorpay Checkout", e);
    }
}

// Callback method invoked on successful payment
@Override
public void onPaymentSuccess(String s) {
    // Retrieve the phone number from the EditText field
    EditText phoneNumberEditText = findViewById(R.id.shippment_phone_number);
    String phoneNumber = phoneNumberEditText.getText().toString();

    // Retrieve the address from the EditText field
    EditText addressEditText = findViewById(R.id.addressEditText);

```

```

String address = addressEditText.getText().toString();

// Store the order details in the Firebase Realtime Database
String orderId = generateOrderId(); // Implement a method to generate a random order ID

// Get product name and grand total
TextView productNameTextView = findViewById(R.id.productNameTextView);
String productName = productNameTextView.getText().toString();

TextView grandTotalTextView = findViewById(R.id.grandTotalTextView);
String grandTotal = grandTotalTextView.getText().toString();

DatabaseReference ordersRef = FirebaseDatabase.getInstance().getReference("Orders");
String phone = UserData.getInstance().getUserId();

// Check if the user exists or create a new user ID and push data under that node
Query query = ordersRef.child(phone).orderByKey().limitToFirst(1);
query.addListenerForSingleValueEvent(new ValueEventListener() {
    @Override
    public void onDataChange(DataSnapshot dataSnapshot) {
        DatabaseReference userOrderRef;

        if (dataSnapshot.exists()) {
            // User exists, get the existing node reference
            DataSnapshot userSnapshot = dataSnapshot.getChildren().iterator().next();
            String orderId = generateOrderId();
            userOrderRef = ordersRef.child(phone).child(orderId);
        } else {
            // User doesn't exist, create a new user ID and push data under that node
            String orderId = generateOrderId();
            userOrderRef = ordersRef.child(phone).child(orderId);
        }

        // Push order data to the database
        pushOrderData(userOrderRef, orderId, productName, grandTotal, phoneNumber, address);
    }
});

private void pushOrderData(DatabaseReference userOrderRef, String orderId, String productName, String
grandTotal, String phoneNumber, String address) {
    userOrderRef.child("productName").setValue(productName);
    userOrderRef.child("grandTotalPrice").setValue(grandTotal);
    userOrderRef.child("orderPhoneNo").setValue(phoneNumber);
    userOrderRef.child("shipmentAddress").setValue(address);

    // Get the current date and set it as the order date
    String currentDate = new SimpleDateFormat("dd-MM-yyyy").format(new Date());
    userOrderRef.child("orderDate").setValue(currentDate);

    DatabaseReference cartRef = FirebaseDatabase.getInstance().getReference()
        .child("Users")
        .child(UserData.getInstance().getUserId())
        .child("Cart");

    String[] productsArray = productName.split("\n\n");

    for (String product : productsArray) {
        product = product.trim();
    }
}

```

```

Query productQuery = cartRef.orderByChild("ProductName").equalTo(product);
removeProductFromCart(productQuery);
}
}

private void removeProductFromCart(Query productQuery) {
productQuery.addListenerForSingleValueEvent(new ValueEventListener() {
@Override
public void onDataChange(DataSnapshot dataSnapshot) {
for (DataSnapshot productSnapshot : dataSnapshot.getChildren()) {
// Get the key and delete the product node
String productKey = productSnapshot.getKey();
productSnapshot.getRef().removeValue();
}
}

@Override
public void onCancelled(DatabaseError databaseError) {
// Error occurred while querying the product
}
});

@Override
public void onCancelled(DatabaseError databaseError) {
// Handle any error during database read
}
});

// Send an SMS message
String message = "Payment successful. Your payment id is " + s;
try {
SmsManager smsManager = SmsManager.getDefault();
smsManager.sendTextMessage(phoneNumber, null, message, null, null);
Toast.makeText(this, "Payment Successful. SMS sent to " + phoneNumber, Toast.LENGTH_SHORT).show();
} catch (Exception e) {
e.printStackTrace();
}

finish();
}

// Callback method invoked on payment failure
@Override
public void onPaymentError(int code, String response) {
Toast.makeText(this, "Payment Failed and cause is " + response, Toast.LENGTH_SHORT).show();
}

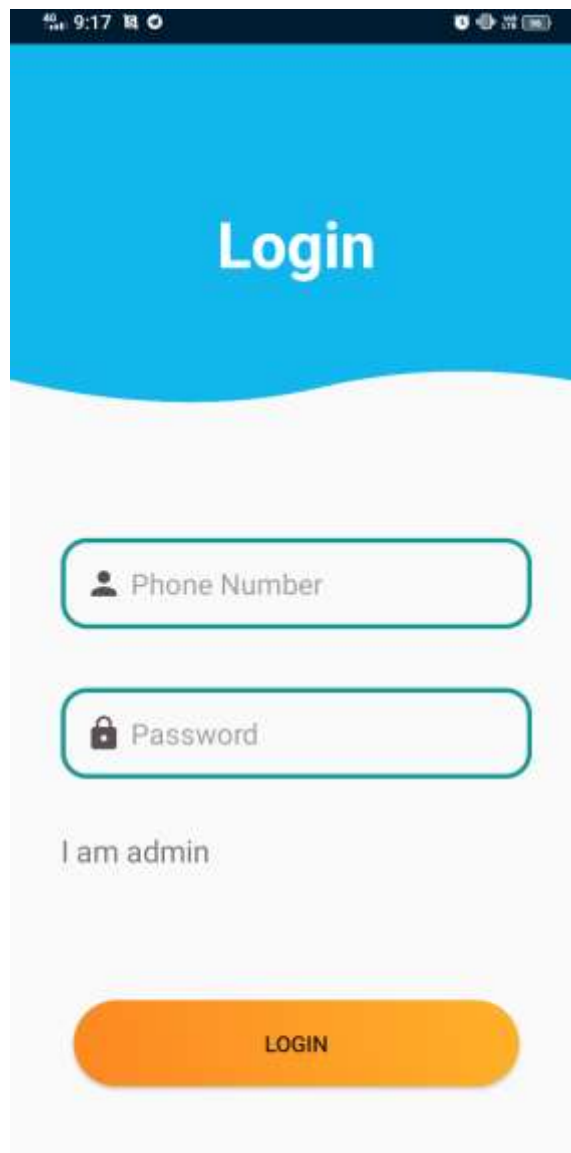
@Override
protected void onDestroy() {
super.onDestroy();
}
}

```

CHAPTER 5

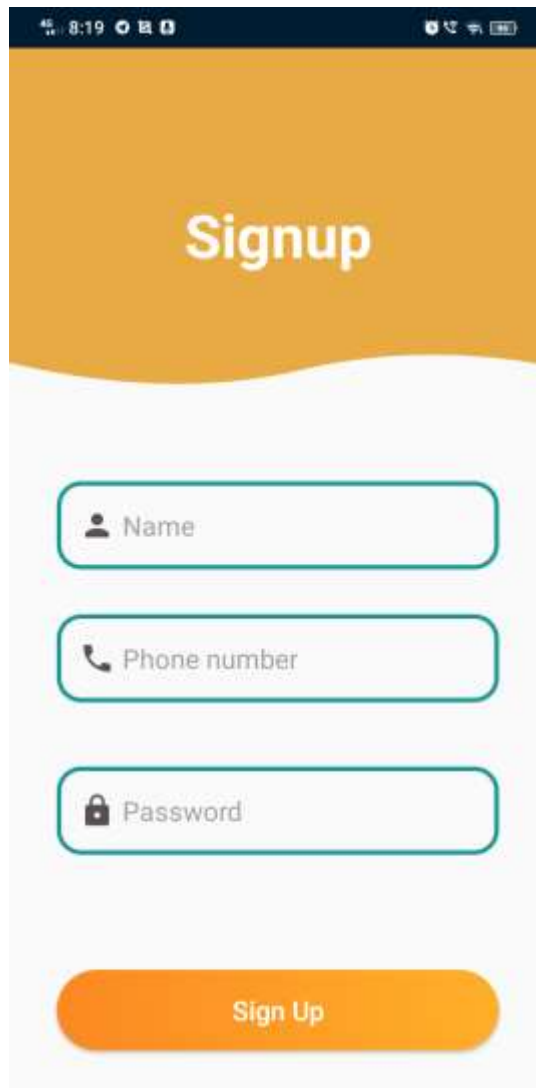
RESULTS

5.1 USER LOGIN PAGE



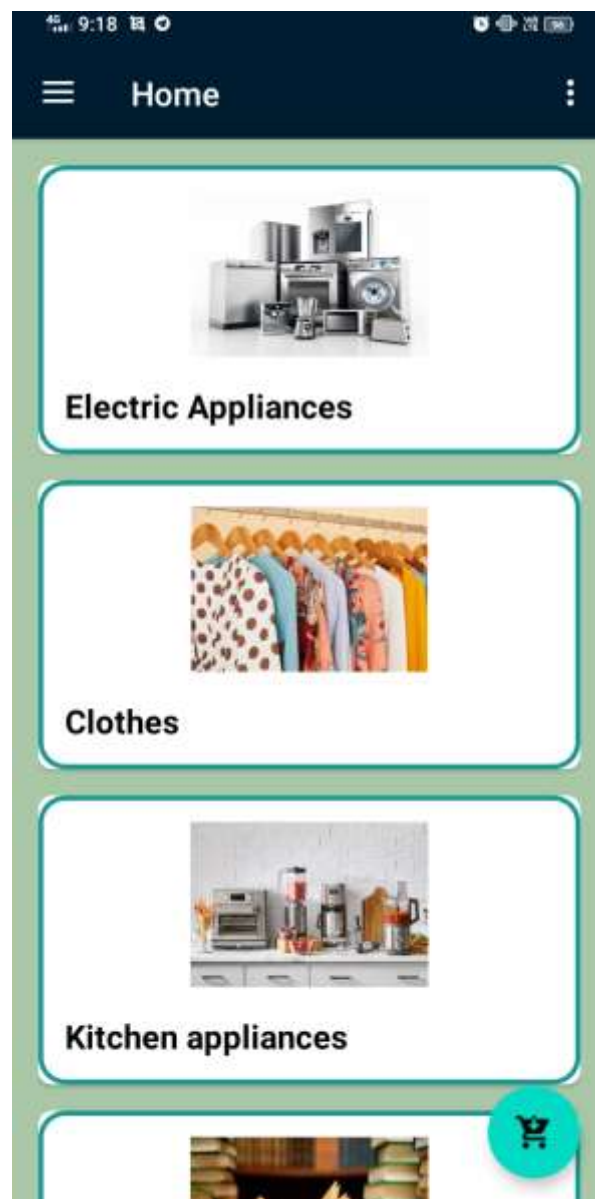
The above figure represents the login page, which allows users to access their accounts by entering their username and password. If users do not have an existing account, they can choose to sign up for a new account.

5.2 USER SIGN UP PAGE



The above figure shows a Sign Up page where people can create an account by giving the necessary details. If someone already has an account, they can log in instead.

5.3 USER HOME PAGE



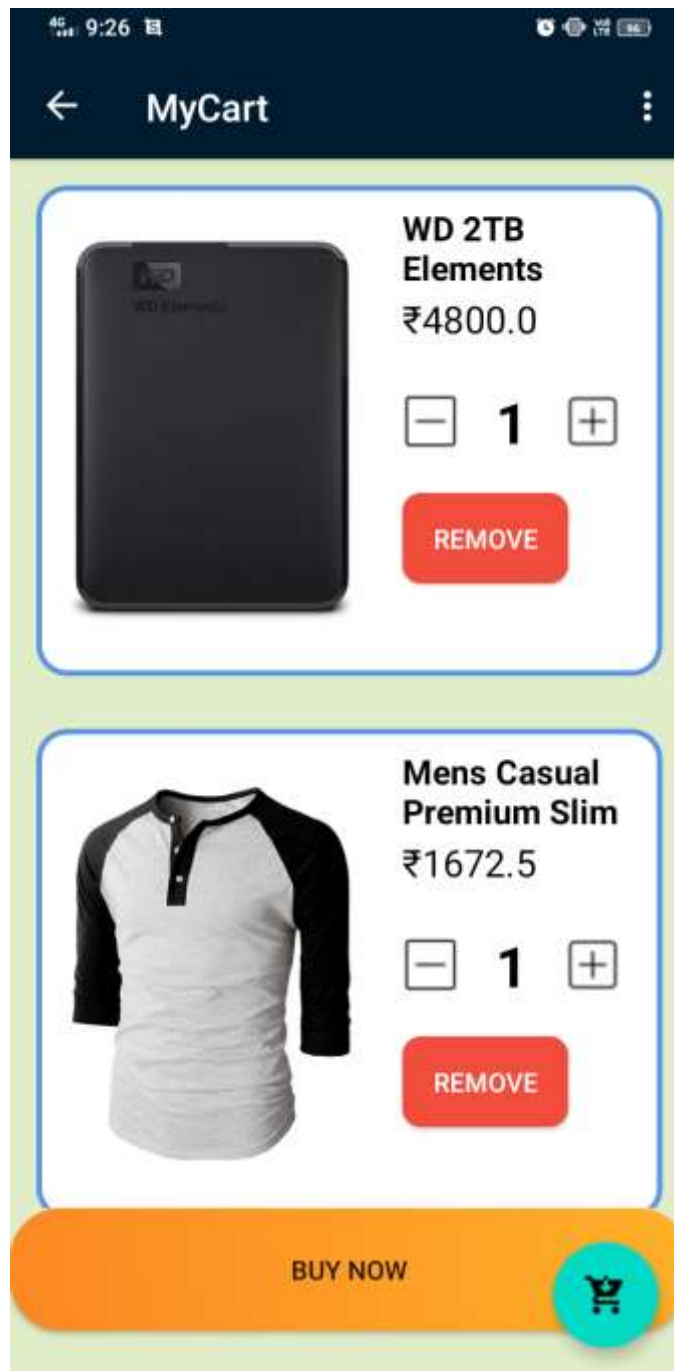
The above figure is the home page, where users can view all products and the number of products add to the cart.

5.4 PRODUCT DETAILS PAGE



The above figure is the Product Detail page, where user can see the product details

5.5 CART DETAILS PAGE



The above figure is the Cart page, where user can add product in cart page the buy the product.

5.6 BUY NOW DETAILS PAGE

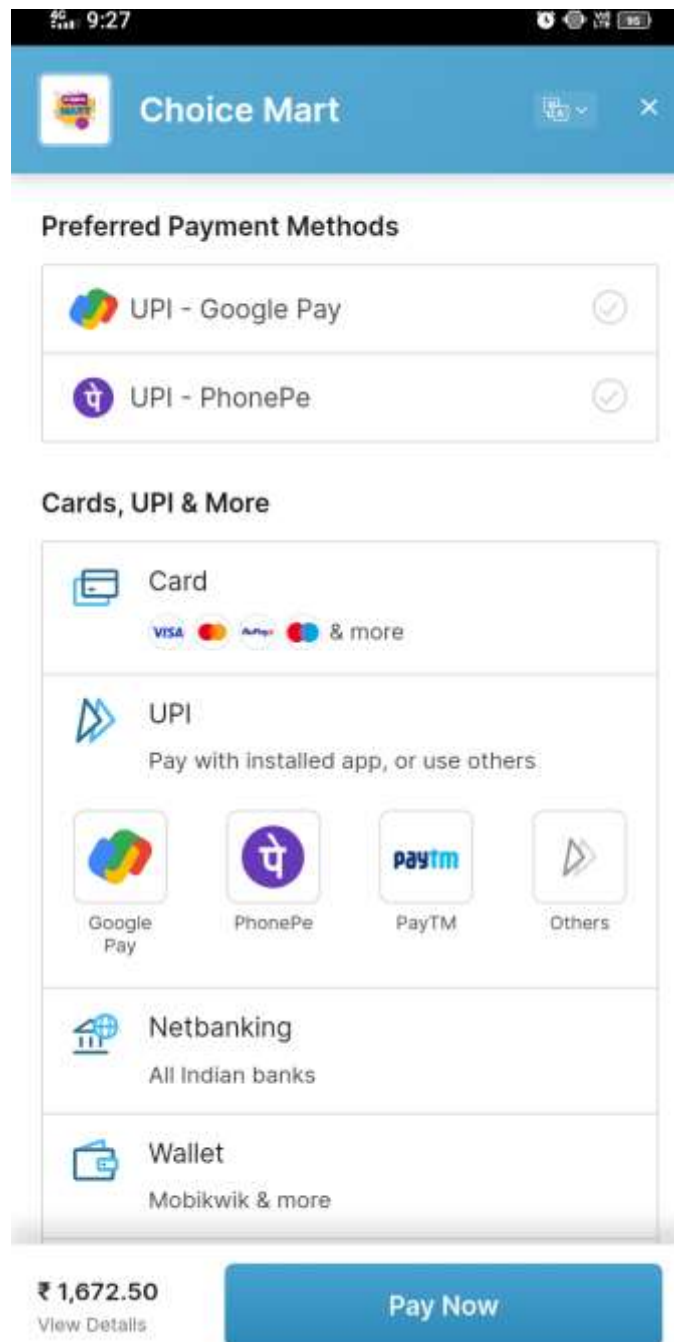
The screenshot displays the 'Checking' page of the CHOICE MART app. At the top, a dark blue header contains the title 'Checking'. Below this, a table lists the product details:

ProductName	Qty	TotalPrice
Mens Casual Premium Slim Fit T-Shirts	1	1672.5

Below the table, the total price is displayed as ₹1672.50. Underneath, there are two input fields: 'Your Phone Number' and 'Enter Address'. At the bottom, there is a green button labeled 'PROCEED TO CHECKOUT'.

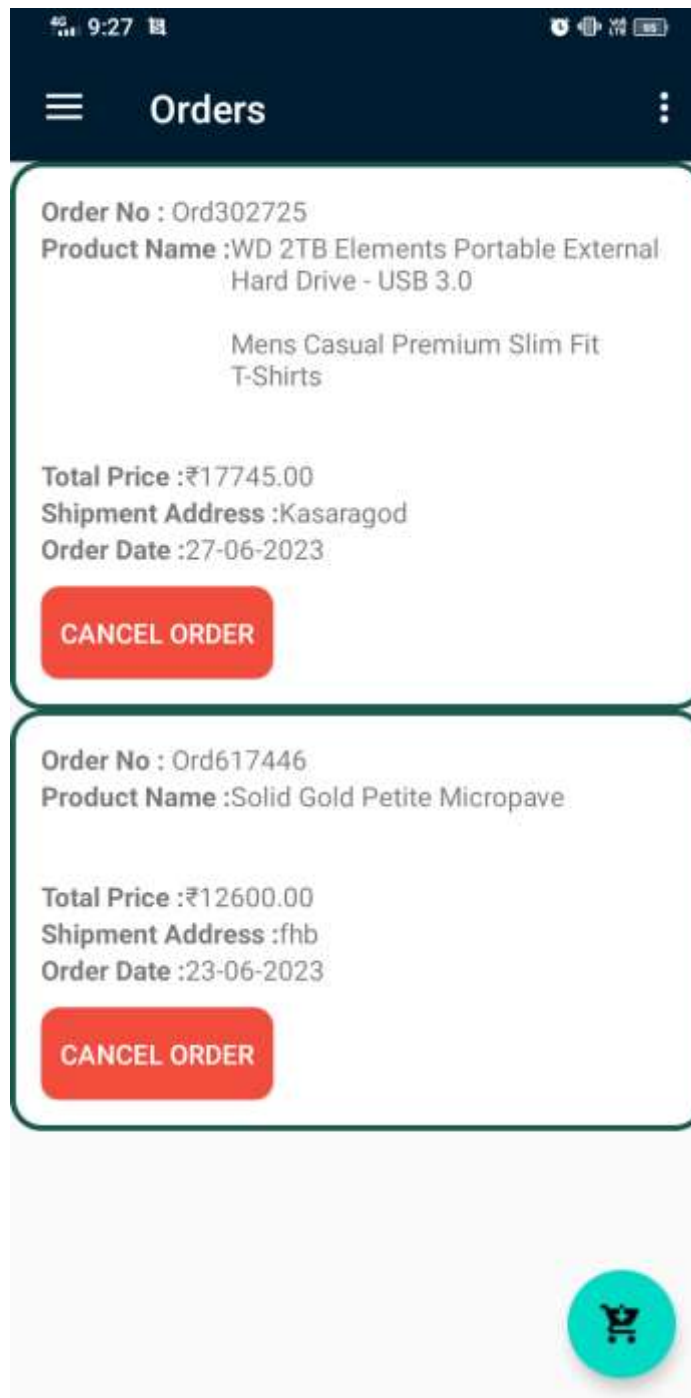
The above figure is the Buy Now page, where user can Buy product and that product price should be displayed.

5.7 PAYMENT DETAILS PAGE



The above figure is Payment Detail page.

5.8 ORDER DETAILS PAGE



The above figure is Order Detail page.

CHAPTER 6

CONCLUSION

In conclusion, our eCommerce Android app project, ChoiceMart, encompasses a wide range of features that enhance the shopping experience for users. With its robust functionality, including buying and selling goods, seamless integration with various payment gateways, and user-friendly interface, ChoiceMart aims to provide a convenient and secure platform for online transactions.

Throughout the development process, we prioritized the needs and preferences of our target users, ensuring that the app offers a smooth browsing experience, comprehensive product listings, and efficient search and filter options. ChoiceMart's intuitive design facilitates easy navigation and effortless product discovery, resulting in a satisfying shopping journey.

By incorporating multiple payment gateways, such as credit/debit cards, digital wallets, and other popular methods, we have strived to accommodate the diverse payment preferences of our users. This variety of options enhances convenience and trust, contributing to a higher rate of successful transactions.

Furthermore, we have implemented robust security measures to safeguard user data and ensure secure transactions. By utilizing industry-standard encryption protocols and adhering to best practices, we prioritize user privacy and data protection, promoting trust and reliability.

ChoiceMart's success relies on a well-rounded approach, combining powerful features, a user-friendly interface, secure payment options, and a commitment to continuous improvement. As we move forward, we will actively gather user feedback, analyze market trends, and iterate on the app to further enhance its functionality and address evolving user needs.

In conclusion, ChoiceMart represents a comprehensive eCommerce solution, providing a seamless and secure platform for buying and selling goods. With its range of features and commitment to user satisfaction, we believe that ChoiceMart will become a preferred choice for online shopping, delivering convenience, reliability, and an enjoyable shopping experience.

REFERENCES

- [1] Google Developer Training, "Android Develop FundamentalsCourse-Concept Reference",Google Developer Training Team, 2017.
<https://www.gitbook.com/book/googledeveloper-training/android-developer-fundamentals-course-concepts/details>.
- [2] Erik Hellman, "Android Programming-Pushing the limits", 1st Edition, Wiley IndiaPvt Ltd, 2014. ISBN-13: 978-8126547197.
- [3] Dawn Griffiths and David Griffiths, "Head First Android Development", 1st SPDPublishers, 2015. ISBN-13: 978-9352131341.
- [4] Bill Phillips, Chris Stewart and Kristin Marsicano, "Android Programming: The Big Berd Ranch Guide", 3rd Edition, Big Nerd Ranch Guides, 2017. ISBN-13: 978- 0134706054.