**Pets Chewy**

Software Requirements Specifications

* Android Studio – v3.6
* GitHub
* Firebase

Hardware Requirements Specifications

* Windows 7
* Intel Core i5 Processor
* Android Mobile Phone

9th June 2020

Team Members

Priyanka Bodapati

Deepthi Tejaswani Chokka

Nikitha Kethireddy

Suma Soma

Sushma Yedugani

Submitted in partial fulfilment

Of the requirements of

CSIS 44-691 Graduate Directed Project 1

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# Document Approval

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

1. **Table of Contents**

|  |  |
| --- | --- |
| **Table of Contents** | **Page Number** |
| 1. Introduction | 5 |
| * 1. Purpose | 5 |
| * 1. Scope | 5 |
| * 1. Definitions, Acronyms, and Abbreviations | 6 |
| * 1. References | 6 |
| * 1. Overview | 6 |
| 1. General Description | 7 |
| * 1. Product Perspective | 7 |
| * 1. Product Functions | 7 |
| * 1. User Characteristics | 7 |
| * 1. General Constraints | 7 |
| * 1. Assumptions and Dependencies | 8 |
| 1. Specific Requirements | 9 |
| * 1. External Interface Requirements | 9 |
| * + 1. User Interfaces | 9 |
| * + 1. Hardware Interfaces | 9 |
| * + 1. Software Interfaces | 9 |
| * + 1. Communications Interface | 10 |
| * 1. Functional Requirements | 10 |
| * 1. Use Cases | 13 |
| * 1. Class/Objects |  |
| * 1. Non-Functional Requirements | 14 |
| * + 1. Performance | 14 |
| * + 1. Reliability | 14 |
| * + 1. Availability | 14 |
| * + 1. Availability Requirements | 14 |
| * + 1. Security | 14 |
| * + 1. Portability | 14 |
| * 1. Inverse Requirements |  |
| * 1. Design Constraints |  |
| * 1. Logical Database Requirements |  |
| * 1. Other Requirements |  |
| 3.10. Prototypes (for complete project) |  |
| 3.11. Use Case Diagrams |  |
| 1. Design | 15 |
| * 1. ER diagram | 15 |
| * 1. GUI | 16 |
| 1. Analysis Models |  |
| * 1. Data Flow Diagram |  |
| * 1. Sequence Diagram |  |
| 1. Technical Manual | 18 |
| * 1. Purpose | 18 |
| * 1. Scope | 18 |
| * 1. GUI of the Project | 18 |
| * + 1. Customer’s flow | 19 |
| * + 1. Admin’s Flow | 20 |
| * 1. Back-end of the Project | 22 |
| * 1. References | 25 |
| 1. End-user Manual | 26 |

****1. Introduction****

# Purpose

E-commerce is fast gaining ground as an accepted and used business paradigm. More and more business houses are implementing mobile apps providing functionality for performing commercial transactions over the web. It is reasonable to say that the process of shopping on the web is becoming commonplace.

The prime objective of this project is to develop a general-purpose e-commerce store where products for pets like food and their toys can be bought from the comfort of home through the Internet. An online store is a virtual store on the Internet where customers can browse the category and select products of interest.

The selected items may be collected in a shopping cart. At checkout time, the items in the shopping cart will be presented as an order. At that time, more information will be needed to complete the transaction. Usually, the customer will be asked to fill or select a billing address, a shipping address, a shipping option, and payment information such as credit card number. An e-mail notification is sent to the customer as soon as the order is placed.

# ****Scope****

10 years ago, when the idea of having a website was in its infancy, only a bunch of academics went ahead to launch their websites and eventually made the most of its ability to reach to the masses. History is repeating itself with the advent of smartphone apps. Yet again, only a small percentage of E-commerce academics have their apps built so far while the rest is continuing to lose on the opportunity.

With mobile applications, customers can interact in the simplest and easiest of the way with enormous products and services. The products and services on sale can be easily viewed along with their descriptions and all other details. All users would have to do is touch, speak or type on their smartphones to connect to these apps. With mobile applications, users are always connected to diverse applications, options that in a very easy manner.

With e-commerce mobile applications, it is estimated that the sales might increase by 30 times in the next five years. Ecommerce mobile applications offer notifications and other important details that send out alerts to customers as soon as a particular event takes place. These notifications could be everything from product news, sale offers, events, or things that a company implements for taking on its strategies.

There are many benefits with E-commerce Mobile Application for customers which are as follows.

* Easier Registration process.
* Few clicks away to purchase an online product.
* More reliable navigation than a web application.
* Fastened Push notifications.
* Excellent user experience.
* Wish list option to shortlist the products.
* Easy to check out.
* Easy to filter and search for products.
* Offline mode.

# Definitions, Acronyms, and Abbreviations

# References

* <http://www.w3schools.com/>
* <http://msdn.microsoft.com/>
* <http://agilemodeling.com/>
* <https://opus.govst.edu/cgi/viewcontent.cgi?article=1079&context=capstones>(image)
* <https://www.v-softinc.com/importance-e-commerce-mobile-applications/>

# ****Overview****

**This document provides a high-level description of Pets Chewy. This documentation is used to develop a general purpose e-commerce store where products for pets like food, and their toys can be bought from the comfort of home through the Internet. It identifies the involved users and helps to explain their roles. The document then describes general software and hardware constraints as well as any assumptions and dependencies concerning the system.**

**The majority of this document focuses on the specific requirement list. The external interface requirements are addressed in the subsequent section. External interface requirements are requirements involving user hardware, software, and communications interfaces. This requirements document concludes with general design constraints specified by the customer along with the requirements this software must meet.**

1. General Description

# Product Perspective

Any member can register and view available products. An only registered member can purchase multiple products regardless of quantity. Contact us page is available to contact Admin for queries.

There are three roles available namely visitor, user and admin. Visitor can view available products but not be able to purchase the products. User can view and purchase products only after his successful login. An Admin has some extra privilege including all privilege of visitor and user.

Admin can add products, edit product information and add/remove a product. Admin can ship an order to the user based on the order placed by sending confirmation mail.

# Product Functions

**The android application functions in two different ways based on the person logged in. If admin provider is logged in, their functionalities will be enabled like adding, editing, deleting the products and tracking the orders placed by customers, view order history, view complaints (if any). And if the customer logged in, their functionalities are like finding products and searching for the products by giving some filter options. Then to add them to the cart/wish list and order online.**

# ****User Characteristics****

**Two types of users are there for this app:**

* **Admin needs to fill license id, email id, name, mobile number, address and password for signup.**
* **Customer: User needs to fill name, email password and phone.**

# ****General Constraint****

**The following constraints are required for this application:**

* **User should have android mobile phone to access the app.**
* **Internet is required.**
* **Should have a mail id to login to the app.**

# ****Assumptions and Dependencies:****

****Assumptions:****

* **Users can easily identify the product by product brand which is mentioned in the description.**
* **Users can accept online orders and payments.**
* **Users can purchase the product easily by searching.**
* **Users can do online payments.**
* **Users can check the status of the order.**

**Dependencies:**

* **Customers need to purchase different products from different vendors**

1. ****Specific Requirements****
   1. External Interface Requirements

External interface requirements specify the way the user shall interact with the system as well as define the necessary hardware interface and communication interfaces required by the software to store and retrieve data.

* + 1. User Interfaces

We will be utilizing android studio to develop the front-end of the application. Within this application, we will try to portrait all the useful applications which are pretty much useful to ensure that the user gets what they need from our application. Help and live chat are instances of such functionalities which serve as an interface amid the user and the software utilized in our project. These functionalities authorize us to gather suggestions and dilemmas that customers have which further provides ideas to the developers in developing the application someday. We will be utilizing firebase at the backend for collecting all the essential data. 

* + 1. Hardware Interfaces

For this project, we require android mobile with a minimum API level of 5.0 which is commonly known as Lollipop. We also require a laptop with Windows XP, or Windows 7, or Windows 8 or Windows 10 Operating system with Intel Core i5 processor and also a stable internet connection for smooth process of program. The points stated below are the hardware requirements which are important for the project.

* Windows XP, or Windows 7, or Windows 8 or Windows 10
* Memory of 4 GB RAM or more
* Monitor resolution of 1024 x 768 or higher
* Intel Pentium 4 or AMD Athlon 2 GHz (or faster)
* 1 GB (or more) available hard disk space
  + 1. Software Interfaces

The points stated below are the software requirements which are useful for successful completion of the project.

* VMware Player to run the Simulator on Windows
* Genymotion to run the simulator in Android Studio
* Android SDK tools 22.0.5 or (latest version)
* Android 4.2.2 platform API 17
* Android Studio
* GitHub for collaborating with tram members with an ease
* Firebase for back-end development
* Visual Studio Code: Provides all the tools that the developer requires for building code.
  + 1. Communications Interfaces

This project supports all types of android devices. We are using this e-commerce application for showing the products to the customers, allowing customers to buy those products, etc.

# Functional Requirements:

# Customer:

1. **Welcome page:** This is the initial screen which is displayed as soon as the app runs. It displays three buttons namely sign in, signup and skip buttons wherein sign in and signup buttons, further have their own individual functionality and skip button enables the user to redirect to the homepage so as to view the items.
2. **Login Page:** As soon as the user clicks upon the sign-in button, entering username and password so as to log in successfully after validating.
3. **Sign-up Page:** As soon as the user clicks on signup button, the user has to register into the app wherein they have to type certain basic information about them.
4. **Reset Password:** If user forgets his password. He can reset his password by requesting to reset password link to his registered email.
5. **Profile Page:** This page shows the basic details of the customer who has logged-in. These basic information is provided by the customer itself. We do not provides any authorization to these basic information.
6. **Home Page:** This page displays all of the products list in grid format.

* **Search bar:** There is a search bar to type for product name or some related information,
* **Filter for searching a Product:** The process of filtration for a particular product is done based upon the category.

1. **Wish list:** User can shortlist the products to wish list further can be added to cart.

* **Delete:** User can delete the shortlisted products from the wish list.

1. **Add to Cart (Cart Management):** This functionality is used to add products to their cart from any of their product collection lists on their homepage and from wish list page.

* **Quantity**: User can change the quantity of the product.
* **Delete**: User can delete the product from cart list placing final order.

1. **Order Confirmation:** This page is displayed when the order which has been placed by the customer is a success.
2. **Checkout:** This is used to quickly buy the product they are viewing. It gives the customers the opportunity to enter the payment details and prompts the customer to complete their order accordingly.
3. **Offers:** There is an option to apply **promo code**. This provides a discount to the overall items in the cart.
4. **Order history**: Order history is a convenient way for a customer to keep track of all current and past orders and their status in the order process. It allows the customer to get up-to-date information on each order placed online, including shipping updates, delivery, and payment details.
5. **Order management:** An individual can **return** or **cancel** the **order** which has already been placed earlier.
6. **Live Chat:** Allows an individual to give quick answers to questions about a products, solve problems faster, and assure customers that someone is there when they are in need. This improves both customer service and loyalty.
7. **Sign-Out:** This button sign out the currently signed-in user. When no user is signed in, pressing this button has no effect.
8. **Contact us:** This is used to provide a set of contacts so as to reach people for assistance. The set of contacts include email address, telephone number, a postal address, links to social media, etc.
9. **Share App:** This functionality is used to share our current application to another pre-existing one. Thus this enables sharing of application to another user.
10. **About Us:** This gives detailed information of about the application. This further informs an individual who uses that application about the company and their respective operations. This provides a visitor with a glimpse of the company and provides more insight into the business that the application makes.
11. **FAQ’s:** Provides information about frequently asked questions or concerns that the customer usually has when an individual uses the application. This usually comprises of text which further consists of most prominently asked questions along with their respective answers to resolve their doubt.

# Admin:

1. **Login Page:** As soon as the admin gives his login details and clicks upon the sign-in button, it validates the details to log in successfully.
2. **Live Chat:** The questions which were being asked by the customers are being answered by the admin.
3. **Profile Page:** This consists of the basic details of the admin who has logged in.
4. **Home page for admin:** After login, this page will show the list of all categories.

* **Add new products:** Whenever the admin clicks on category, it show a form with fields name, price and description of the product.

1. **Managing products page:**

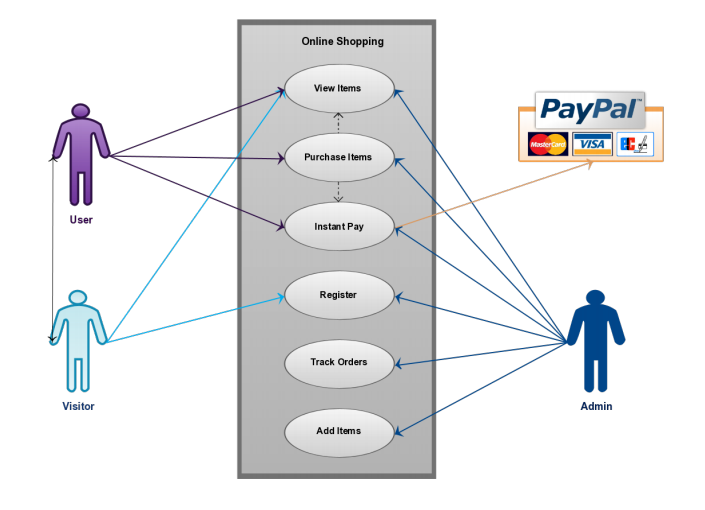
This shows the list of products already added.

* **Edit product details:** Whenever admin clicks on the product it allows to edit the price, name, and description.
* **Delete Product:** There will be a delete button below every product, whenever the admin clicks on delete, it asks for confirmation to delete.

1. **Check for orders placed:**

This page shows all the orders so far ordered by different customers.

# Use Cases



The image reference is given in the references.

* 1. Non- Functional Requirements

A non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behavior. This should be contrasted with functional requirements that define specific behavior or functions. In general, functional requirements define what a system is supposed to do whereas non-functional requirements define how a system is supposed to be.

* + 1. **Performance requirements:**

The mobile application will be developed using Android Studio which will, thanks to the good algorithms that will be used, guarantee a high execution speed and a minimized response time.

* + 1. **Reliability requirements:**

Reliability refers to the probability and or the likelihood that a given product will perform in the way and or manner it was intended to perform in the efforts that have been deemed required of that given product within or under a specific period of time required.

* + 1. **Availability Requirements:**

The system must be always available for use. Except for special cases (in case of backup), it will announced beforehand through push notifications.

* + 1. **Security Requirements:**

The application must be very secure because it deals with the private information of the users. This should be performed using the right encryption of data only accessed by the administrator. More than that, the system must follow these main security rules: Confidentiality: Only the admin has access to user personal info and orders. Integrity: Only the users can modify their personal info. Authenticity: No one can access, modify or delete other accounts’ information.

* + 1. **Portability Requirements:**

Portability specifies the ease with which the software can be installed on all necessary platforms, and the platforms on which it is expected to run. By using appropriate server versions released for different platforms our project can be easily operated on any operating system, hence can be said highly portable.

4. Design

# 4.1. ER diagram:

For this app we need to store admin information, product information, customer information, order information, payment information. Admin Table contains all the employees who has right to edit the website information. Customer table contains all the necessary information of customer. We can back track the payment and order information by placing CheckNum and Order ID in the customer table.

All the products in the website will be there in the product table. Product table contain attributes ID, SKU, Name, Price, Weight, Qty, Image, Description. Order\_Product contains orderID and productID. So that we can tract both order information and product information. Order table contains order id and customer ID. So, we track which customer has which order.

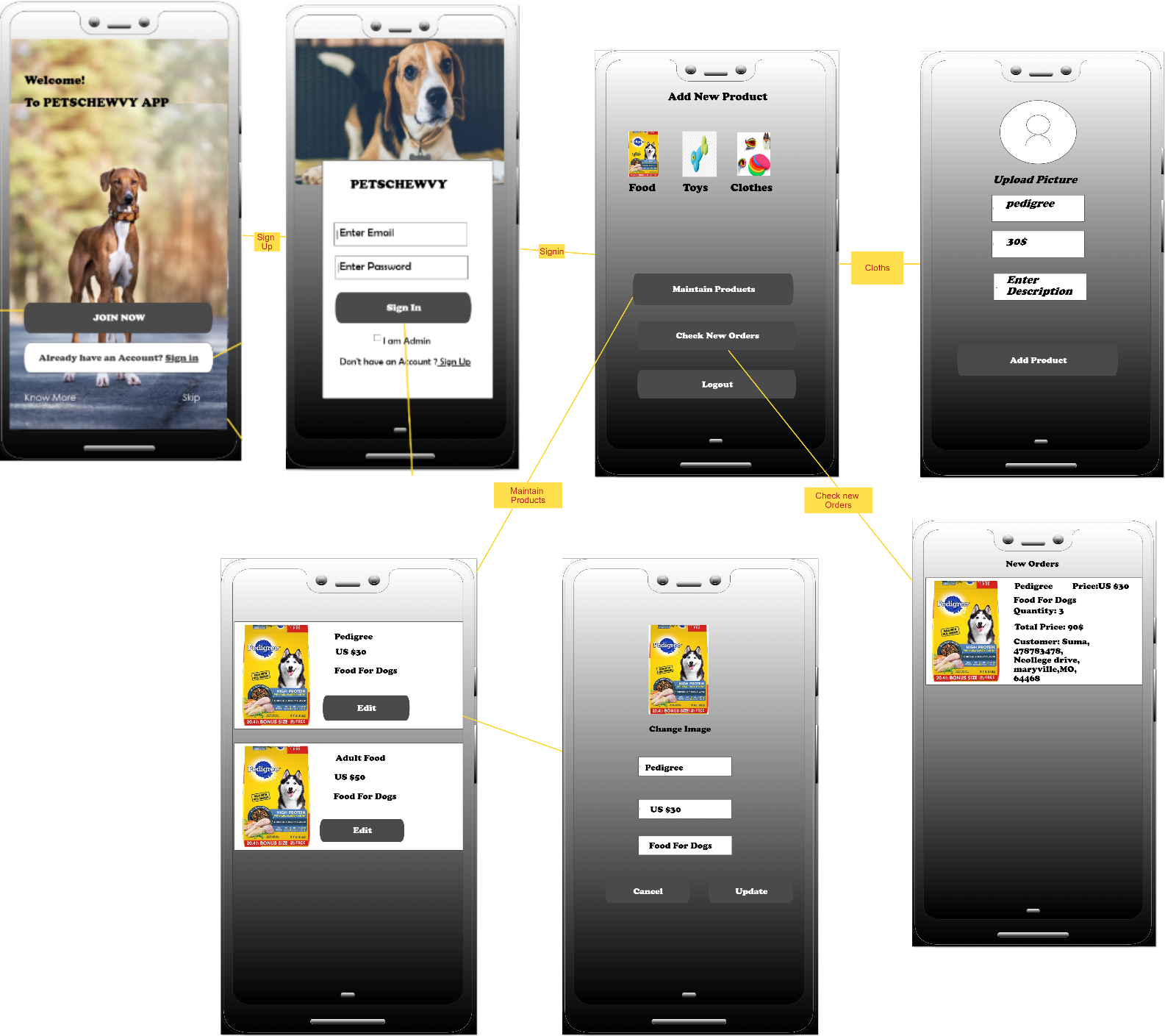
## ER-Diagram

# 4.2. GUI

# Customer’s Flow:



# Admin’s Flow



1. **Technical Manual**

# 6.1. Purpose:

The purpose of this manual is to provide the gist of the project as to how a part of the project works accordingly. The manual is intended to provide assistance to the people who are using the application. This allows us to provide the detailed description of the application to another person in case if we have to hand over our job to complete or modify our project as and when required.

# 6.2. Scope:

This manual describes about the GUI and backend of the Pets Chewy Project which is used as an e-commerce website for buying the products which are related to pets.

The GUI of the project comprises of the front-end screens. Wherein it depicted in the form of a flowchart. It show a flow of continuity like what exactly happens when a button on a screen is clicked or which page it gets redirected when another button on that page is clicked. The GUI of the project varies whether the person who decides to login is a customer or an admin. The screens which is both common to customer as well as admin will be the welcome and login screens. The flow of the screens will be explained in detail as we go further.

The back-end of the project is related to the storage of data and real-time database. These are used to store all the credentials of the customer or the admin. The credentials include name, phone number, mail id., etc. The data thus stored is to be used for authorization during login. Which means that certain data is used so as to check and validate with the data typed in the GUI with that of the data within the firebase. If the data is correct, then the access is granted otherwise it will display a message stating that the data typed or the login credential are incorrect. To acquire this, we need to follow certain steps and these steps will be explained further.

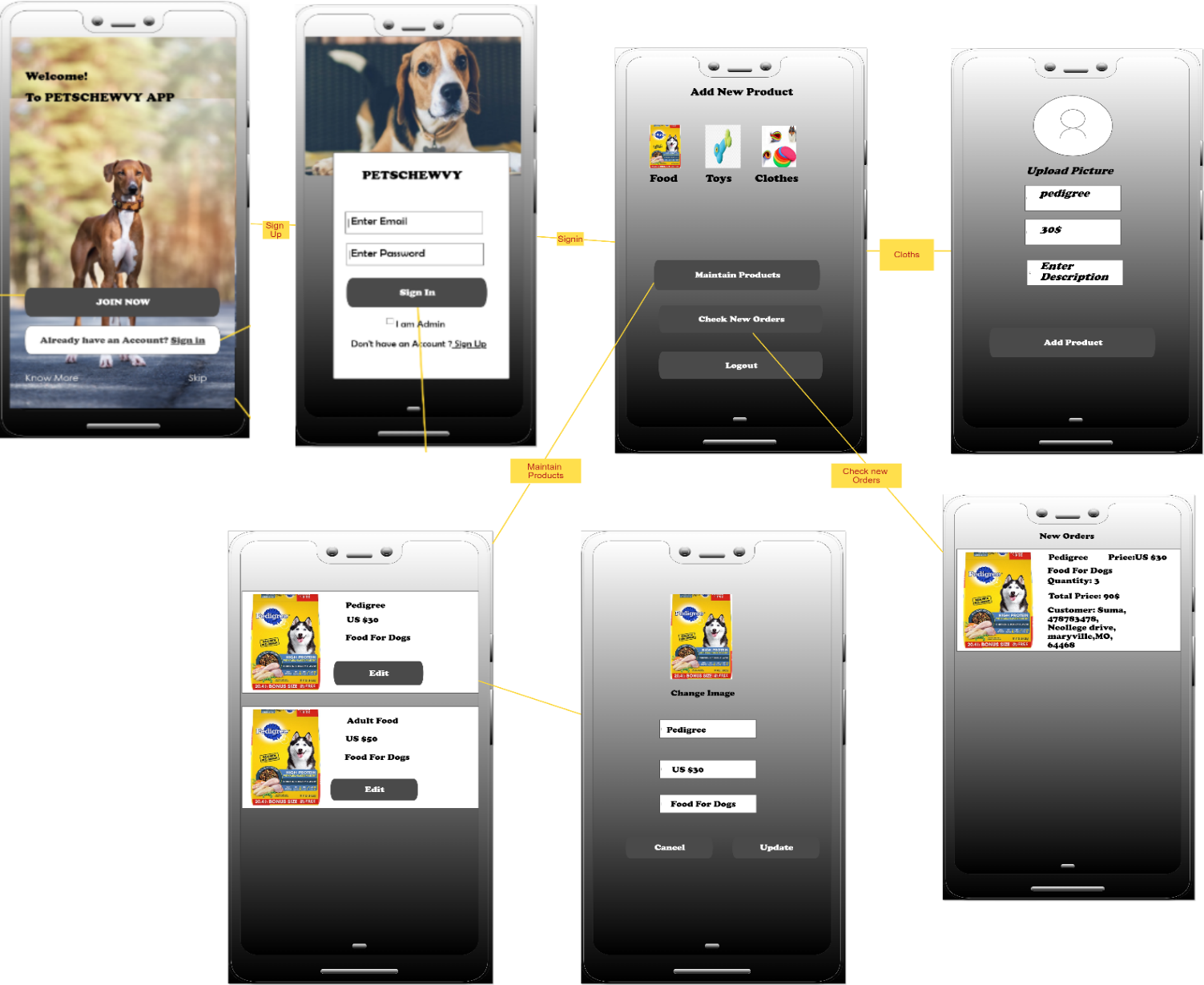
# 6.3. GUI of the project:

The GUI of the project varies whether the person who decides to login is a customer or an admin. The screens which is both common to customer as well as admin will be the welcome and login screens. The flow of the screens will be explained in detail as we go further.

**6.3.1. Customer’s flow:** 

The figure displayed above show the screens with respect to the customer. The screens displayed are just the sample screens and not the actual screens which are utilized in the application. This is to ensure that the user manual provides a clear description of the front-end or the GUI of the application which is used for a customer.

**6.3.2. Admin’s flow:**



The figure displayed above show the screens with respect to the admin of the application. The screens displayed are just the sample screens and not the actual screens which are utilized in the application. This is to ensure that the user manual provides a clear description of the front-end or the GUI of the application which is used for an admin.

When the admin decides to login, they need to type their respective username and password and it is compulsory that they need to tick-mark the check-box so as to ensure that it is admin who is logging in and not another person.

**Sample Code:**

The code stated below is a sample code for the page activity\_welcome.xml

*<?***xml version="1.0" encoding="utf-8"***?>*

*<!--the layout of this entire page will be in the form of relative layout  
 wherein all the tags used like background, etc. are applicable to this layout -->*

<**RelativeLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 android:background="@drawable/azawakhdog"  
 tools:context=".MainActivity"**>  
  
*<!--Here, we are going to use the image(logo of our app) which is located in another -->  
<!-- folder named drawable and the name of the image is applogo and we are adjusting -->  
<!-- the app logo to our convenience like height, width, etc., This is to ensure that the applogo -->  
<!-- is clearly visible-->*

<**ImageView  
 android:id="@+id/app\_logo"  
 android:layout\_width="300dp"  
 android:layout\_height="100dp"  
 android:layout\_centerHorizontal="true"  
 android:layout\_marginTop="60dp"  
 android:src="@drawable/applogo"** />  
 …………………………

…………………………

…………………………

*<!-- This is a code for a button with id main\_login\_btn for -->  
<!-- redirecting to login page when the button is pressed-->  
<!-- The button is customized as desired so as to ensure that -->  
<!-- the look and feel of the page is as pleasant and as-->  
<!-- understandable as possible-->*  
 <**Button  
 android:id="@+id/main\_login\_btn"  
 android:layout\_width="match\_parent"  
 android:layout\_height="wrap\_content"  
 android:layout\_alignParentStart="true"  
 android:layout\_alignParentBottom="true"  
 android:layout\_marginStart="11dp"  
 android:layout\_marginLeft="15dp"  
 android:layout\_marginRight="15dp"  
 android:layout\_marginBottom="100dp"  
 android:background="@drawable/buttons"  
 android:padding="20dp"  
 android:text="Already have an Account? Login"**

**android:textAllCaps="false"  
 android:textColor="@android:color/white"  
 android:textSize="18sp"  
 android:layout\_alignParentLeft="true"** />  
 …………………………

…………………………

…………………………

*<!--This is the closing tag of the relative layout and thus-->  
<!-- this marks the end of the xml file -->*

</**RelativeLayout**>

The end result of the code(stated above) is the layout screen which is displayed below.



**6.4. Back-end of the Project:**

We are currently using Firebase for the back-end of our project. The back-end of the project is related to the storage of data and real-time database. These are used to store all the credentials of the customer or the admin. The credentials include name, phone number, mail id., etc. The data thus stored is to be used for authorization during login. Which means that certain data is used so as to check and validate with the data typed in the GUI with that of the data within the firebase. If the data is correct, then the access is granted otherwise it will display a message stating that the data typed or the login credential are incorrect.

This lets us build rich, collaborative applications by allowing secure access to the database directly from client-side code. Data is persisted locally, and even while offline, real-time events continue to fire, giving the end user a responsive experience. When the device regains connection, the Real-time Database synchronizes the local data changes with the remote updates that occurred while the client was offline, merging any conflicts automatically.

The Real-time Database is a NoSQL database and as such has different optimizations and functionality compared to a relational database. The Real-time Database API is designed to only allow operations that can be executed quickly. This enables you to build a great real-time experience that can serve millions of users without compromising on responsiveness. Because of this, it is important to think about how users need to access your data and then structure it accordingly.

In order to install and setup on android, we need to follow the necessary steps:

* Connect your app to firebase
* Create a database
* Add the real-time database SDK to your app
* Configure real-time database rules
* Prepare for launch

**Sample Code:**

The code stated below is the code which is used for registering a customer i.e. for saving the data which has been typed by the customer.

…………………………………………….

…………………………………………….

…………………………………………….

*<!—If the text view of email and password is empty then it will provide a toast message stating that the signup was unsuccessful, please try again-->  
<!—otherwise it will start another activity and we will be redirected to the welcome page -->*

if(!(email.isEmpty() && pwd.isEmpty())){

mFirebaseAuth.createUserWithEmailAndPassword(email, pwd).addOnCompleteListener(RegisterActivity.this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task) {

if(!task.isSuccessful()){

Toast.makeText(RegisterActivity.this,"SignUp Unsuccessful, Please Try Again",Toast.LENGTH\_SHORT).show();

}

else {

startActivity(new Intent(RegisterActivity.this,HomeActivity.class));

}

}

});

}

…………………………………………….

…………………………………………….

…………………………………………….

The code stated below is the code which is used for login of a user i.e. for data retrieval and comparing it with the data typed in the text fields.

…………………………………………….

…………………………………………….

…………………………………………….

*<!—If the text view of email and password is empty then it will provide a toast message stating that login error, please login again-->  
<!—otherwise it will start another activity and we will be redirected to the welcome page -->*

if(!(email.isEmpty() && pwd.isEmpty())){

mFirebaseAuth.signInWithEmailAndPassword(email, pwd).addOnCompleteListener(LoginActivity.this, new OnCompleteListener<AuthResult>() {

@Override

public void onComplete(@NonNull Task<AuthResult> task) {

if(!task.isSuccessful()){

Toast.makeText(LoginActivity.this,"Login Error, Please Login Again",Toast.LENGTH\_SHORT).show();

}

else{

Intent intToHome = new Intent(LoginActivity.this,HomeActivity.class);

startActivity(intToHome);

}

}

});

}

…………………………………………….

…………………………………………….

…………………………………………….

### Reference:

For further guidelines, an individual can refer to the websites stated below:

* <https://firebase.google.com/docs/storage>
* <https://firebase.google.com/docs/database/android/start>
* <https://firebase.google.com/docs/storage/android/start>

### Team member’s responsibilities:

* **Priyanka Bodapati**
* **Deepthi Chokka**
  + **Search Functionality:**

In order to implement the search functionality first we need to obtain the keyword and this keyword will be used to obtain the necessary items which are related to the keyword. The next process will be to retrieve the data from the database based upon the keyword that the customer uses.

**Sample Code:**

**Front-end Code:**

*<?*xml version="1.0" encoding="utf-8"*?>*<RelativeLayout  
 xmlns:android="http://schemas.android.com/apk/res/android"  
 xmlns:app="http://schemas.android.com/apk/res-auto"  
 xmlns:tools="http://schemas.android.com/tools"  
 android:layout\_width="match\_parent"  
 android:layout\_height="match\_parent"  
 tools:context=".SearchProductsActivity">  
  
 <RelativeLayout  
 android:id="@+id/r5"  
 android:layout\_width="match\_parent"  
 android:layout\_height="60dp"  
 android:background="@color/colorPrimaryDark"  
 android:layout\_alignParentTop="true"  
 android:layout\_centerHorizontal="true">

………………………………………………………………

………………………………………………………………

………………………………………………………………

**Back-end Code:**

protected void onCreate(Bundle savedInstanceState)  
{  
 super.onCreate(savedInstanceState);  
 setContentView(R.layout.*activity\_search\_products*);  
 inputText = findViewById(R.id.*search\_product\_name*);  
 SearchBtn = findViewById(R.id.*search\_btn*);  
 searchList = findViewById(R.id.*search\_list*);  
 searchList.setLayoutManager(new LinearLayoutManager(SearchProductsActivity.this));  
 SearchBtn.setOnClickListener(new View.OnClickListener() {  
 @Override  
 public void onClick(View view)  
 {  
 SearchInput = inputText.getText().toString().toLowerCase();  
 onStart();  
 }  
 });  
}  
@Override  
protected void onStart()  
{  
 super.onStart();  
 DatabaseReference reference = FirebaseDatabase.*getInstance*().getReference().child("Products");  
  
 FirebaseRecyclerOptions<Products> options =  
 new FirebaseRecyclerOptions.Builder<Products>() .setQuery(reference.orderByChild("pname").startAt(SearchInput).endAt(SearchInput+"\uf8ff"), Products.class)  
 .build();

…………………………………………….

…………………………………………….

…………………………………………….

* + Contact-us Page

This is a static page which is used when the customer wants to ask any questions.

The inner implementation is

* + Maintain Products(Admin)

# End-user Manual

The end-user manual is used to show or direct as to how the user should use the application. This ensures that the user of this application know what shall the next page be if an individual clicks upon a button. Thus clarifying the doubts and all queries that arise when using the application.

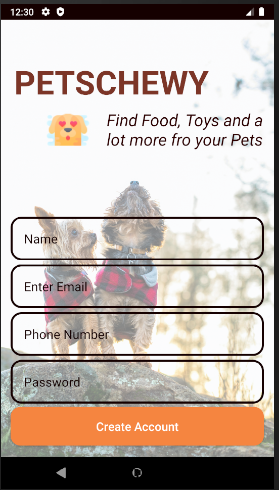
1. **Home screen:**



If the app runs successfully, then the above screen will be displayed to the user of the application. There are two buttons namely join now and login. These buttons are used by the customer accordingly. Admin does not have the functionality to register but admin shares the same login page with that of the customer in order to login as an admin.

If an individual customer clicks upon “Join Now” button, then the customer gets redirected to the registration screen.

1. **Registration Screen:**



In this screen the customer have to enter their name, phone number and the password for the account. When they click on create account button, the application checks whether the credential are already used. If they are already used then the application gives a popup message stating that user has already created an account with these credential. Else, the customer will register successfully and a popup message will be displayed likewise.

1. **Login Screen(For Customer):**

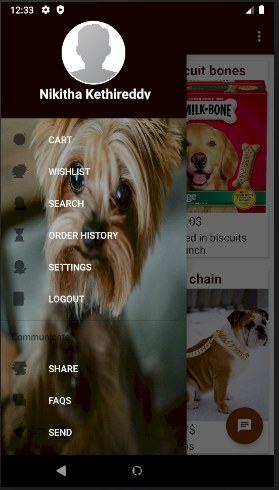
If the customer has an existing account, then they can login to the application. All they have to do is to click upon the” Login” button in the home screen. By clicking upon this button, they will be redirected to the screen as shown below.



The customer needs to type their phone number and the password which has been provided during their registration. If the credentials match with the credentials present within the database, then they can login successfully. An individual can view a popup stating whether the login was a success or a failure. If login was a success, then the customer will be re-directed to the welcome screen designed for the customers.

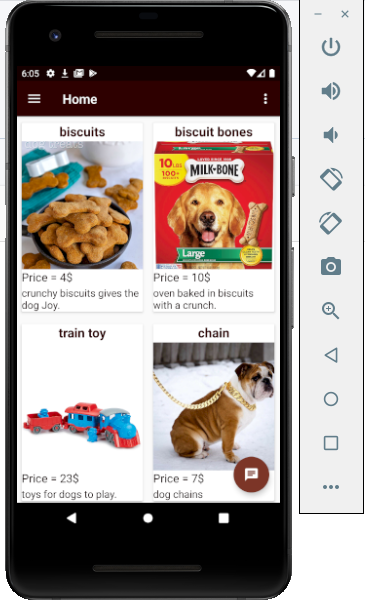
1. **Welcome Screen(For Customer):**

The figure stated below is the screen which is displayed after a successful login of the customer. An individual gets to view their corresponding name, email and will be able to use all the functionalities present in the navigation bar as shown in the image.



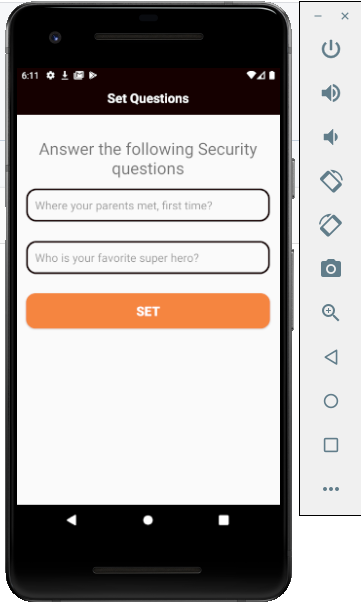
1. **Home Screen(Products List):**

Immediately after Login, customer can see all list of products, when customer long presses on the product, he can see two options to view the full details of the product and adding them to Wish list for future purchase.



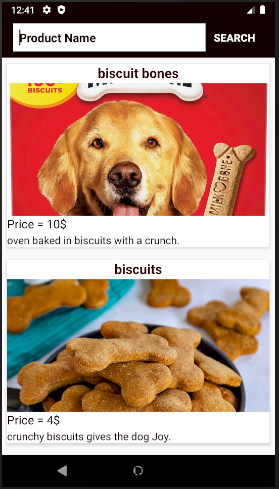
1. **Set Security Questions:**

After Login, if there are no security questions given by the user, it will show the security questions page where he can give the answers to the questions. These answers are used to reset password if customer forgets his password.

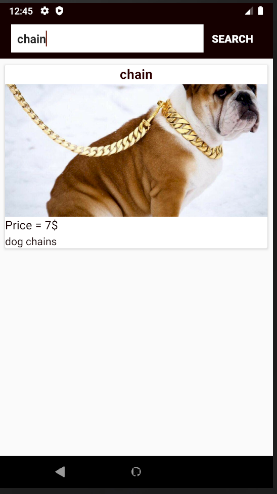


1. **Search Screen:**

This is one of the functionalities present for customer. When the customer requires an item, they can use the search functionality so as to search and find out whether the product they require is present or not. Search functionality enables the customer to look out for a product with an ease. By clicking upon search option in the navigation bar, customer will be redirected to the search page as shown below.

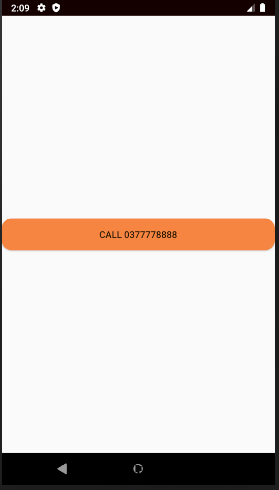


Here customer has to type the name of the product and press the search button. When the button is pressed, the products related to the name of the product will be displayed as shown in the screenshot below.



1. **Contact-Us Screen:**

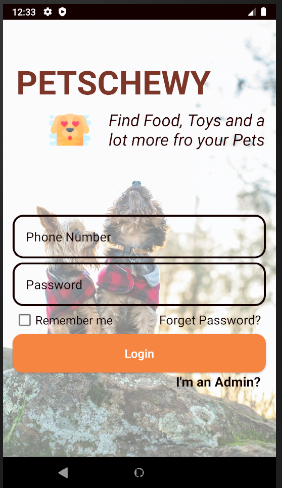
This is used to contact a concerned person for any further information or to clarify the customer’s doubts, etc. By clicking upon contact-us option in the navigation bar, customer will be redirected to the contact-us page as shown below.



By clicking upon the button shown in this page, the customer will be redirected to call the concerned person.

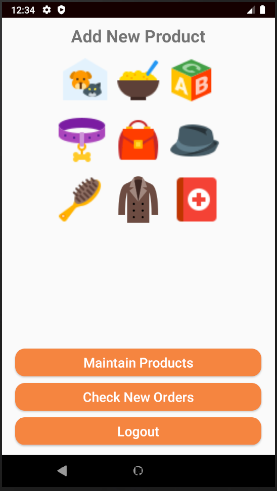
1. **Login Screen(For Admin):**

Now, coming to the admin, we can see from the image below that both admin and the customer share the same login screen. As soon as the admin gets redirected from the home screen to the login screen, admin now has to type their corresponding credentials. By clicking upon the login button, if the login was a success then the admin will be redirected to the welcome screen exclusively designed for the admin.



1. **Welcome Screen(For Admin):**

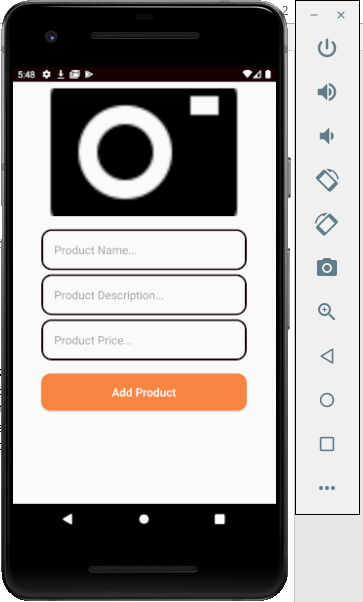
The image stated below is the welcome screen which is displayed for the admin.



1. **Add new product:**

Admin can add products to the Application by giving the fields like Product Name, Description, and price of the product.

He can also add image for the reference purpose. After adding the products here, customers can browse for products for future purchase.



1. **Maintain products:**

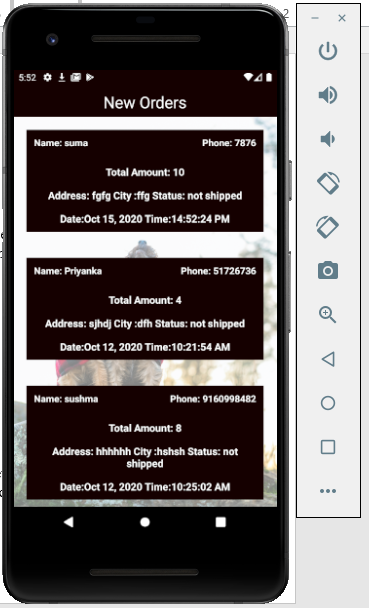
If the admin decides to modify/delete the current existing products due to various reasons like stock unavailable, etc. The admin can modify the product accordingly. This can be done by clicking upon the maintain products button and selecting the particular product the admin wants to make the changes. By doing that, the page will be redirected to the necessary page as shown in the below screenshot.



By clicking on apply changes, changes made will be saved in the firebase. Admin can also delete the product so as to remove an item from the database completely.

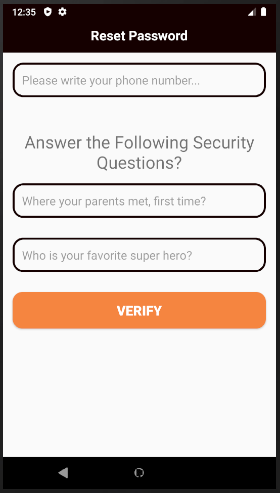
1. **Check new orders:**

This page will have all the purchases list of customers and their details like order date, order total, shipping address and time of order.



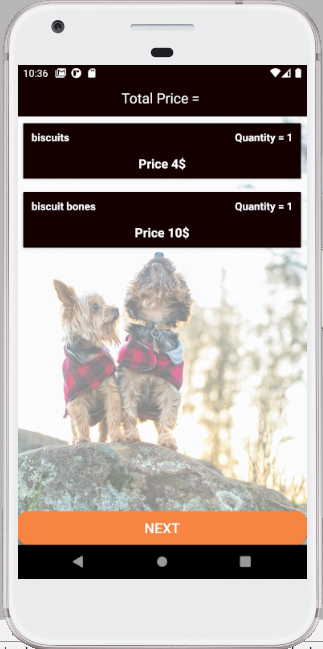
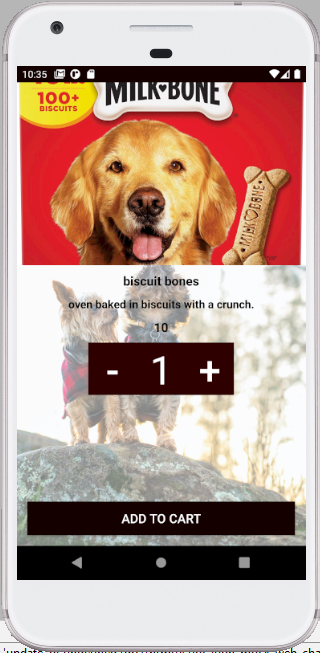
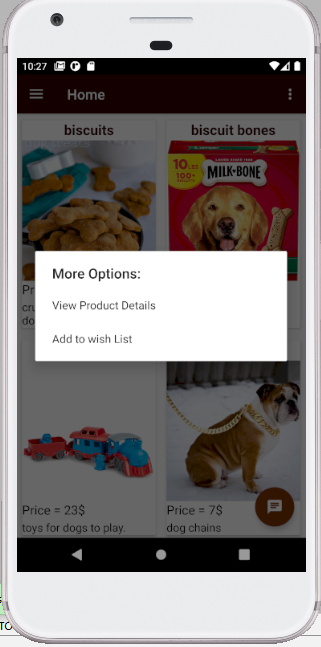
1. **Forget password screen:**

If the user of the application (customer or admin) forgets their password during the process of login, then the user can click upon the “forgot password” button present in the login screen. Then they will be re-directed to the screen which is displayed below. All the user has to do is to type their email id for which their password of the account is to be sent and click upon the “Reset Password” button.



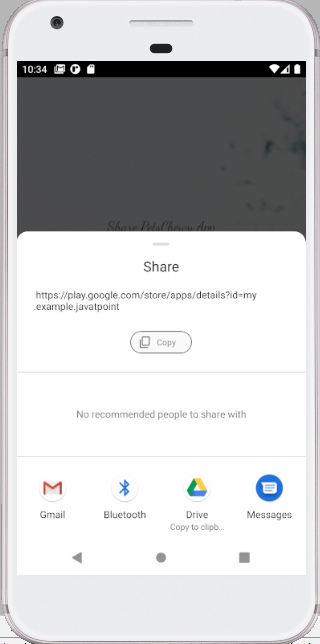
Sushma Yedugani Full Responsibilities:

1. Cart
2. Share
3. Checkout(Payment page)
4. Cart Page.



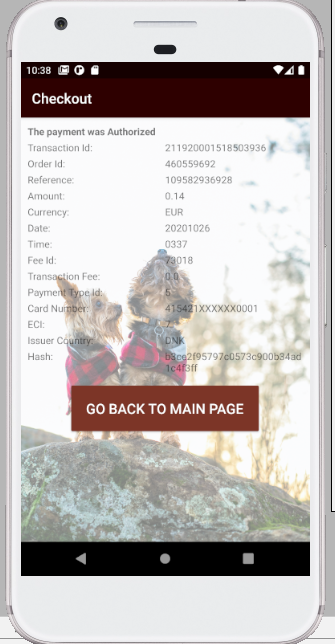
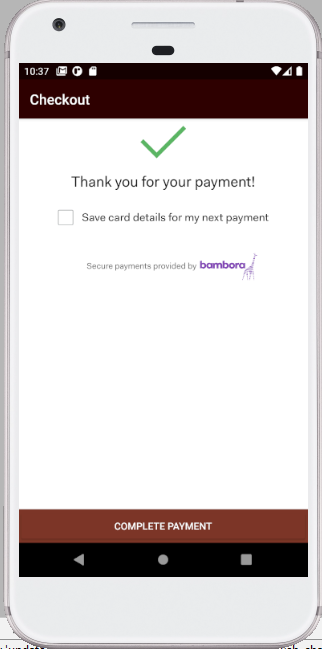
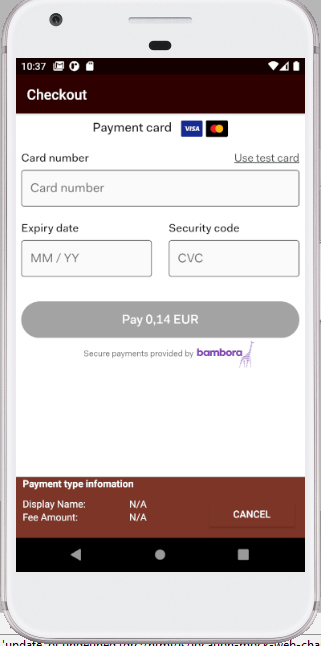
As we can see from the above images, We have three screens related to cart functionality, First screen allows customer to redirect to the product details page by clicking on the View Product details option and then he/she will be redirected to the second screen where they can add the quantity for the particular product and add it to the cart. Once the customer clicks on the add to cart button they’ll be redirected to the third screen where the all the products added to the cart are being displayed in the Cart Page.

1. Share



The above two screens are related to the Share page where when you open the Share page the above first screen will appear and when you click on the share image you’ll be redirected to the second screen above, then you’ll have options displayed below on the screen where you can share your app with your friends, family or anyone via the above given options.

1. Payment



The above screenshots belong to the Payment related pages where after you select the payment method credit/debit card payment, you’ll be redirected to the first page and then after entering all the details asked in the first screen and clicking on pay button will redirect to second page and once you click on the complete payment button we’ll be redirected to the third page where it gives us the payment details summary.