



Flutter Course

# Shopify App with Flutter

**Final**



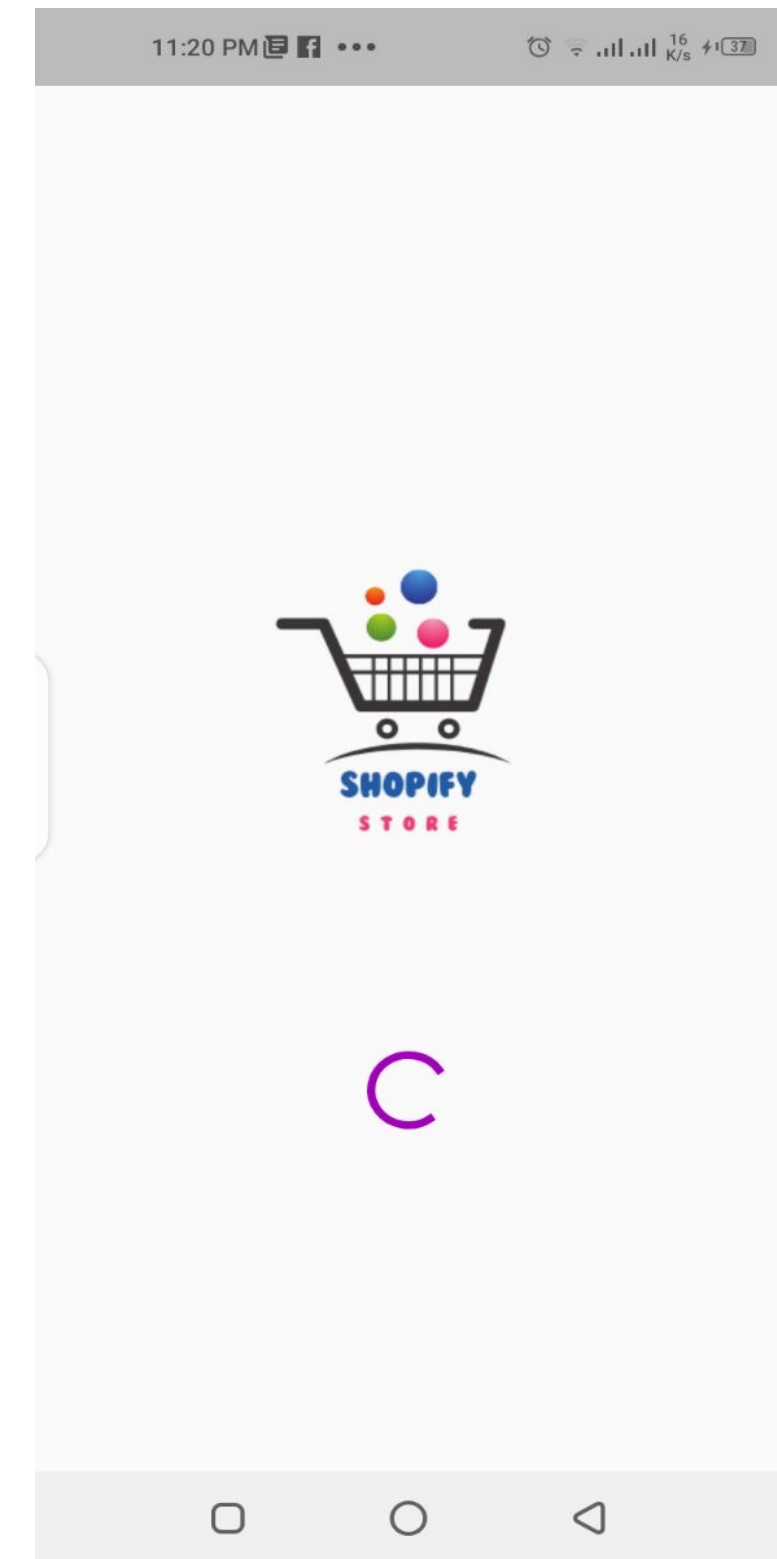
Fatma Abdelghany Mohammed

# Splash Screen

In Splash Screen check if user first time to use app .

If first time show onboard Screens.  
Then start Registration Pages.

And check if already register then  
Go to master page.



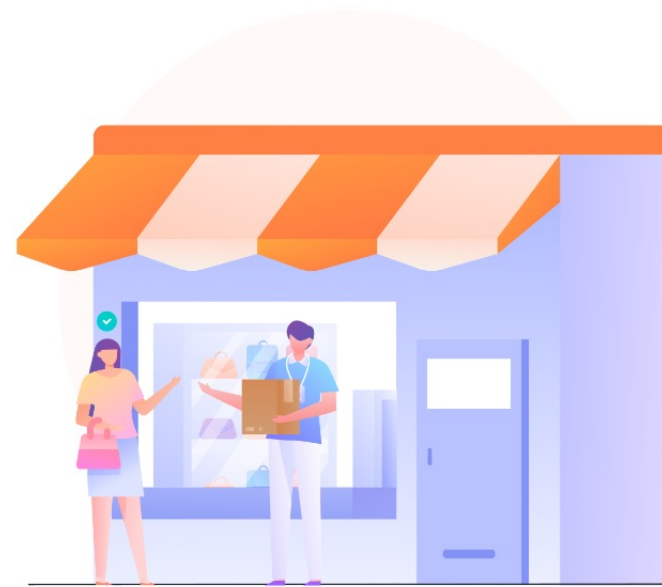
# Introduction Screen



## New Collection

Welcome to Shpify App, Let's shop!

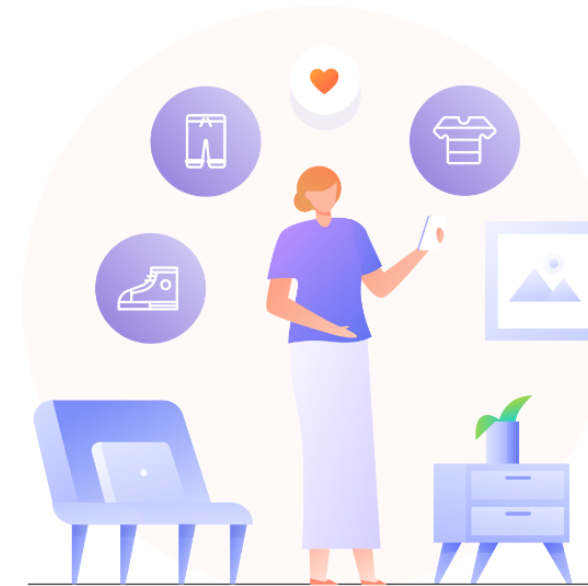
Skip



## Easy To Buy

We help people connect with store  
around Egypt

Skip



## Free Delivery

We show the easy way to shop.  
Just stay at home with us



Done

# Introduction Screen

We can use an introduction screen to explain our software, for example, at the launcher. This widget has a nice design and is quite adaptable.

Simply add the `introduction_screen` to your `pubspec.yaml` file as a dependency.

```
1  dependencies:
2    cupertino_icons: ^1.0.2
3    flutter:
4      sdk: flutter
5
6    introduction_screen: ^3.0.2
```

# Introduction Screen

intro Screen takes list of Pages then add PageViewModel with the title, body, an image, and page decoration.

```
1    pages: [  
2        PageViewModel(),  
3        PageViewModel()  
4    ]
```



# Introduction Screen

intro Screen takes list of Pages then add PageViewModel with the title, body, an image, and page decoration.

```
1    pages: [  
2        PageViewModel(),  
3        PageViewModel()  
4    ]
```



# Introduction Screen

```
PageViewModel(  
  title: 'Title of 1st Page',  
  body: 'Body of 1st Page',  
  
  image: Center(  
    child: Image.asset("images/image_1.png", width: 450, height: 200,)  
  ),  
  //getPageDecoration, a method to customise the page style  
  decoration: const PageDecoration(  
    imagePadding: EdgeInsets.only(top: 120),  
    pageColor: Colors.white,  
    bodyPadding: EdgeInsets.only(top: 8, left: 20, right: 20),  
    titlePadding: EdgeInsets.only(top: 50),  
    bodyTextStyle: TextStyle(color: Colors.black54, fontSize: 15),  
  )  
)
```

# Introduction Screen

to save status if you just want to show IntroductionScreen once (at the start of your application) (already display or not). We use **SharedPreferences**

```
setInSharedPreference() async {  
    await PreferenceService.getPrefsInstance!.setBool("firstOpened", true);  
}
```



# ScreenUtilInit

One of Flutter's primary goals is to create a framework that allows you to develop apps from a single codebase that looks and feels great on any platform.

When it comes to layout in Flutter, design can be grouped into two:

**Responsive:** This is an app that has its layout tuned for the available screen size.

**Adaptive:** Running an app on different device types.

Package Name: [flutter\\_screenutil](#)



# ScreenUtilInit

We wrap our `MaterialApp` with `ScreenUtilInt` Widget and you will have to pass the following parameters

- ✓ `designSize` : This is very important to set our default fit size, you can find this in your UI design, by looking at the dimensions of the device screen and filling it in.
- ✓ `Builder`: Use builder only if you need to use a library outside `ScreenUtilInit` context



# ScreenUtilnit

For Example: if we want to display a square based on width and height: put **.w** , **.h**

```
Container(  
  width: 50.w,  
  height:200.h  
)
```



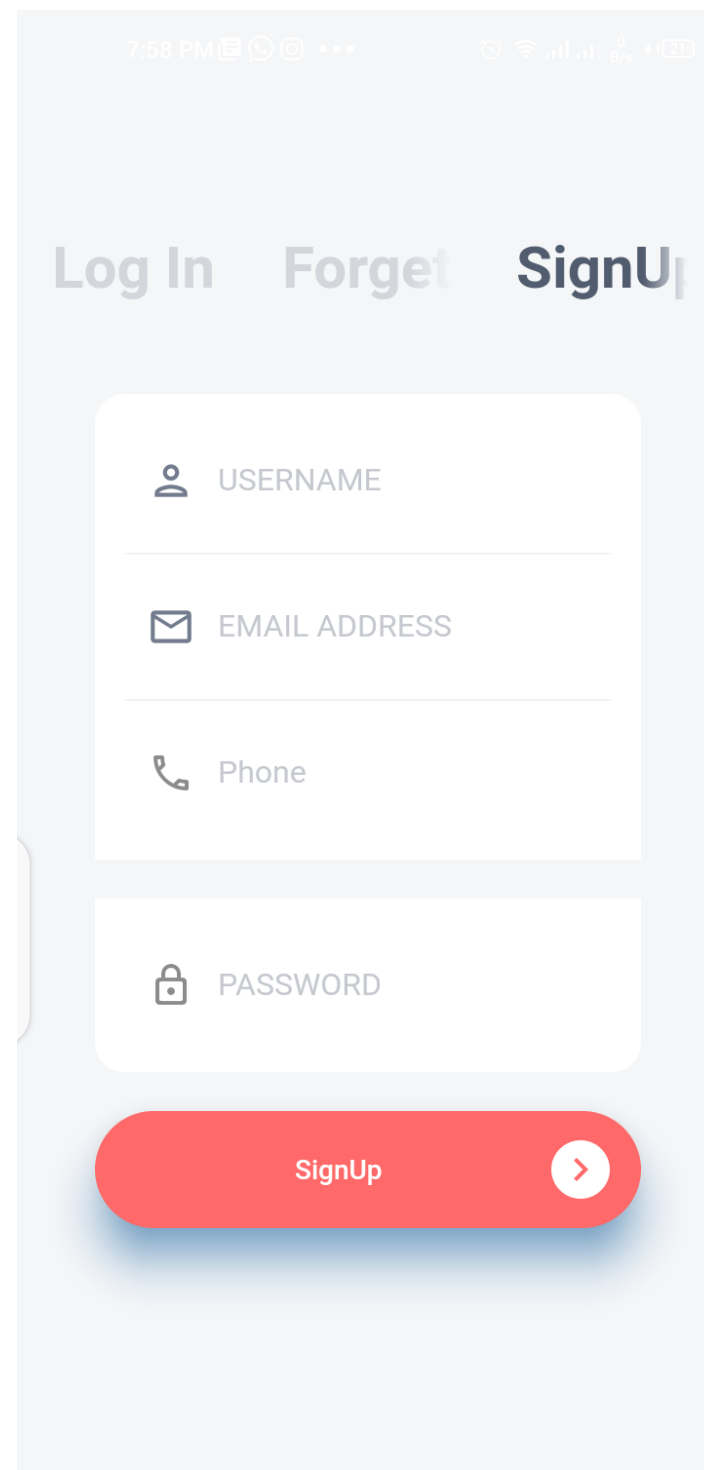
# ScreenUtilInit

Adapter font: put **.sp** for font size

```
Text(  
  '16sp,if data is not set in MediaQuery,my font size will change with  
  style: TextStyle(  
    color: Colors.black,  
    fontSize: 16.sp,  
  ),
```



# Registration Pages



7:58 PM

Log In Forget **SignUp**

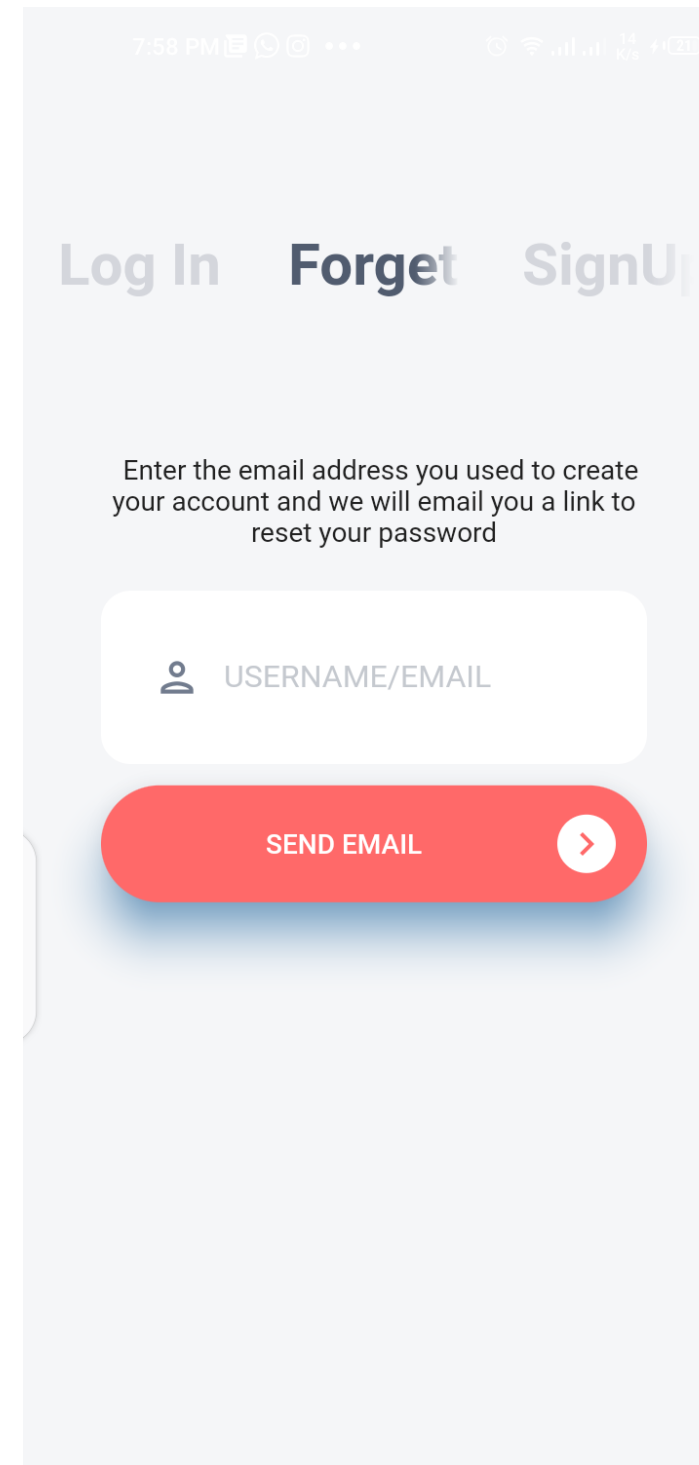
USERNAME

EMAIL ADDRESS

Phone

PASSWORD

SignUp



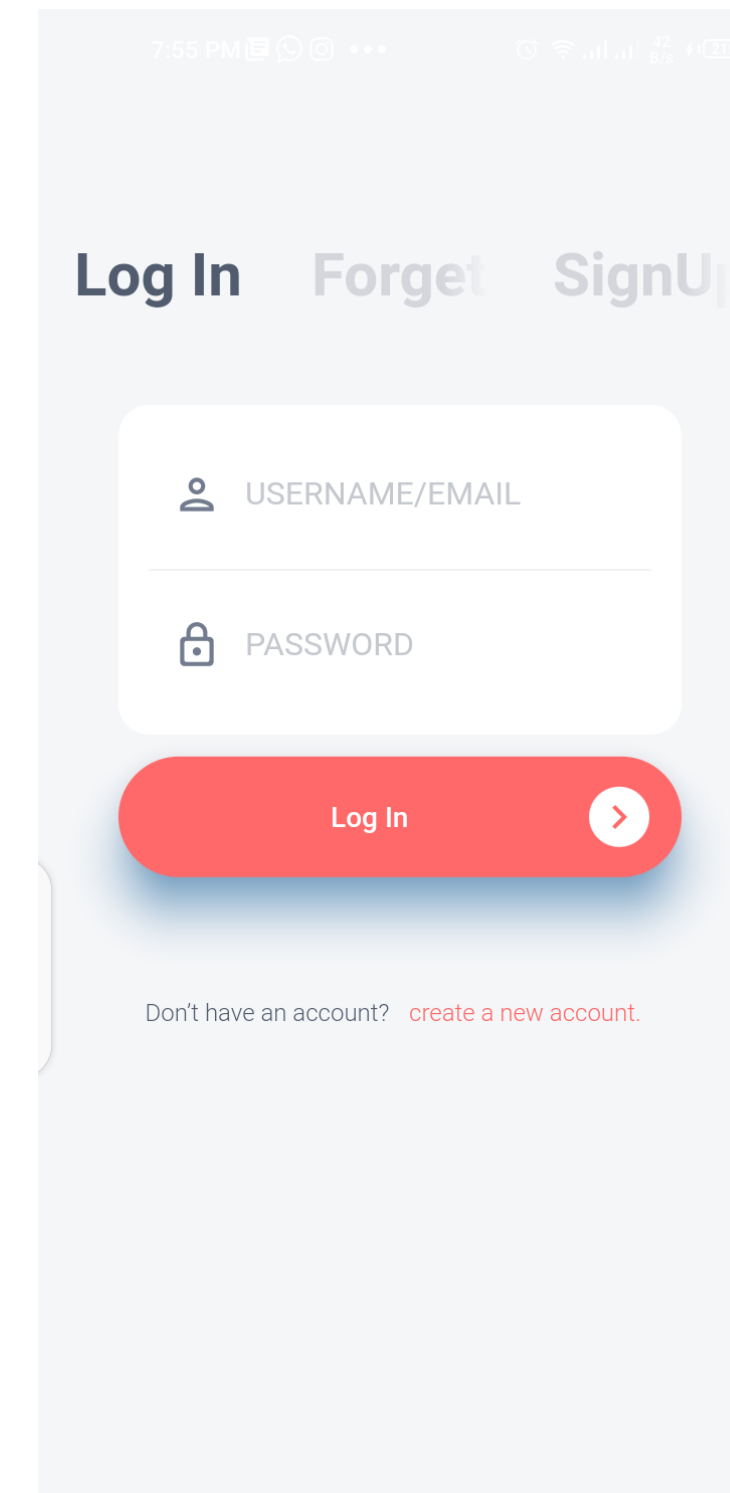
7:58 PM

Log In **Forget** SignUp

Enter the email address you used to create your account and we will email you a link to reset your password

USERNAME/EMAIL

SEND EMAIL



7:55 PM

Log In **Forget** SignUp

USERNAME/EMAIL

PASSWORD

Log In

Don't have an account? [create a new account.](#)

# Registration Pages

## Tab Bar:

By using Tab Bar widget, I handle login, SignUp, Forget Screens

Create and assign `TabController`:

```
TabController _controller;  
int _selectedIndex = 0;  
@override  
void initState() {  
  super.initState();  
  _controller = TabController(length: 5, vsync: this);  
}
```

# Registration Pages

## Tab Bar:

Tab Bar take List of tabs like this.

```
TabBar(  
  tabs: [...],  
)
```

```
tabs: const [  
  // first tab  
  Tab(  
    text: StringsConstants.login,  
  ), // Tab  
  
  // second tab  
  Tab(  
    text: StringsConstants.forgetPassword,  
  ), // Tab  
  
  // third tab  
  Tab(  
    text: StringsConstants.signUP,  
  ), // Tab  
,
```

# Registration Pages

## Tab Bar:

Set the Design of screens in **TabBarView**:  
TabBarView takes controller and list of children

```
body: TabBarView(  
  controller: _controller,  
  children: <Widget>[... ],  
  //controller: _tabController,  
),
```

```
Expanded buildTabBarPages() {  
  return Expanded(  
    child: TabBarView(  
      controller: _tabController,  
      children: const [  
        // first tab bar view widget  
        LoginPage(),  
  
        // second tab bar view widget  
        ForgetPassword(),  
        // third tab bar view widget  
        SignUpPage()  
      ],  
    ), // TabBarView  
  ); // Expanded
```



# Registration Pages

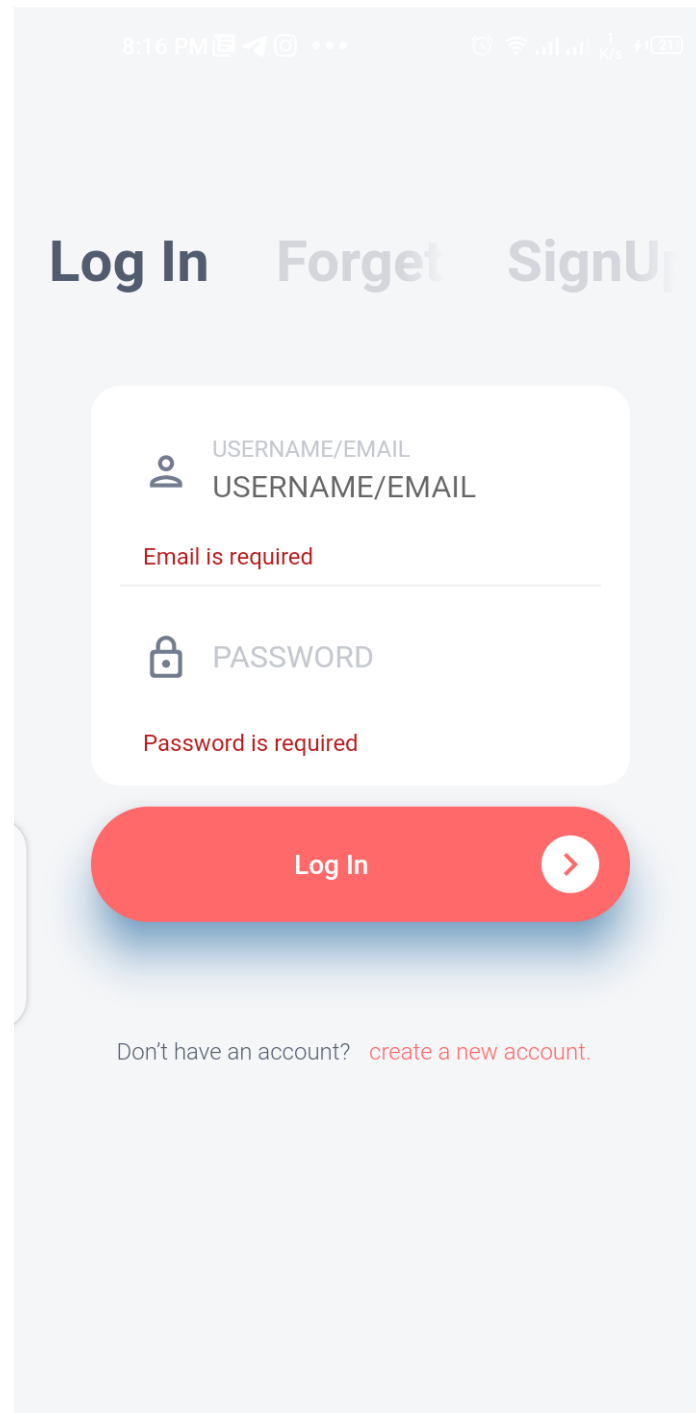
**For Validation :**

I use `email_validator`: package

To check if user entered valid email address



# Registration Pages



8:16 PM

Log In Forget Sign Up

USERNAME/EMAIL  
USERNAME/EMAIL

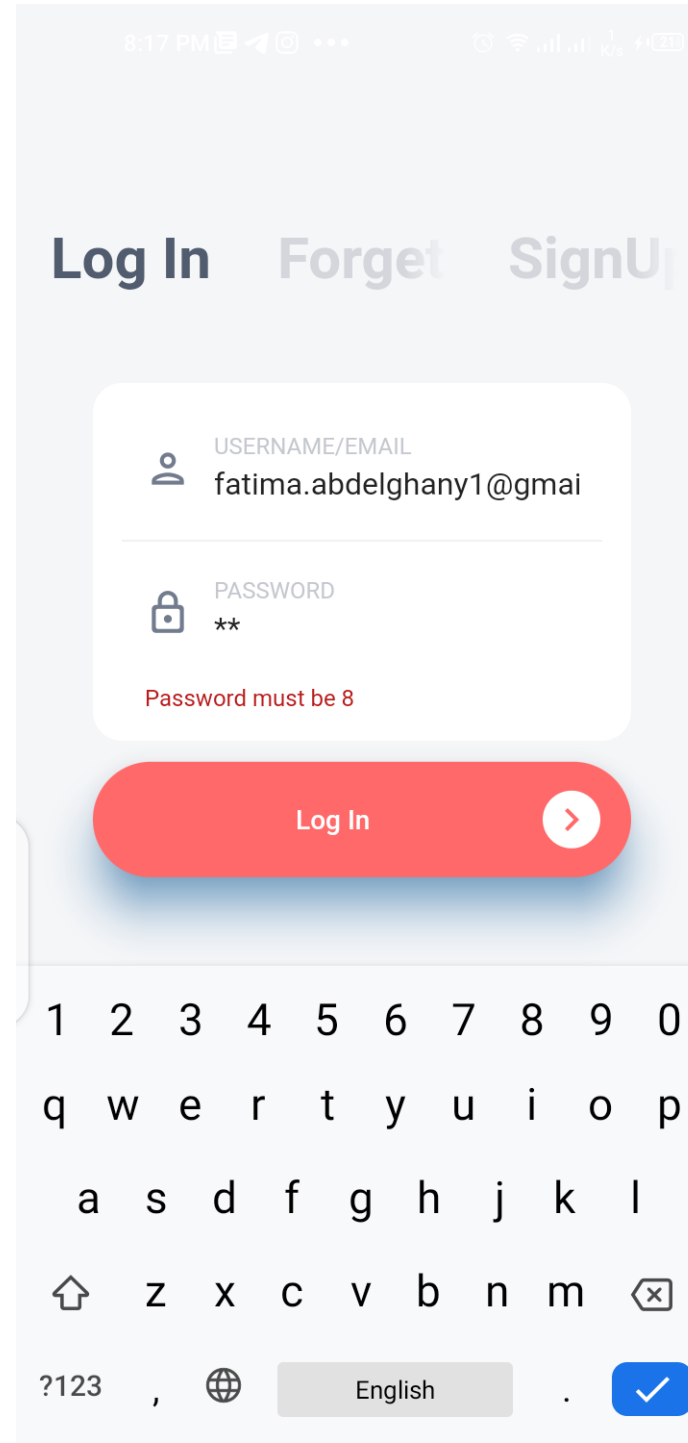
Email is required

PASSWORD  
PASSWORD

Password is required

Log In

Don't have an account? [create a new account.](#)



8:17 PM

Log In Forget Sign Up

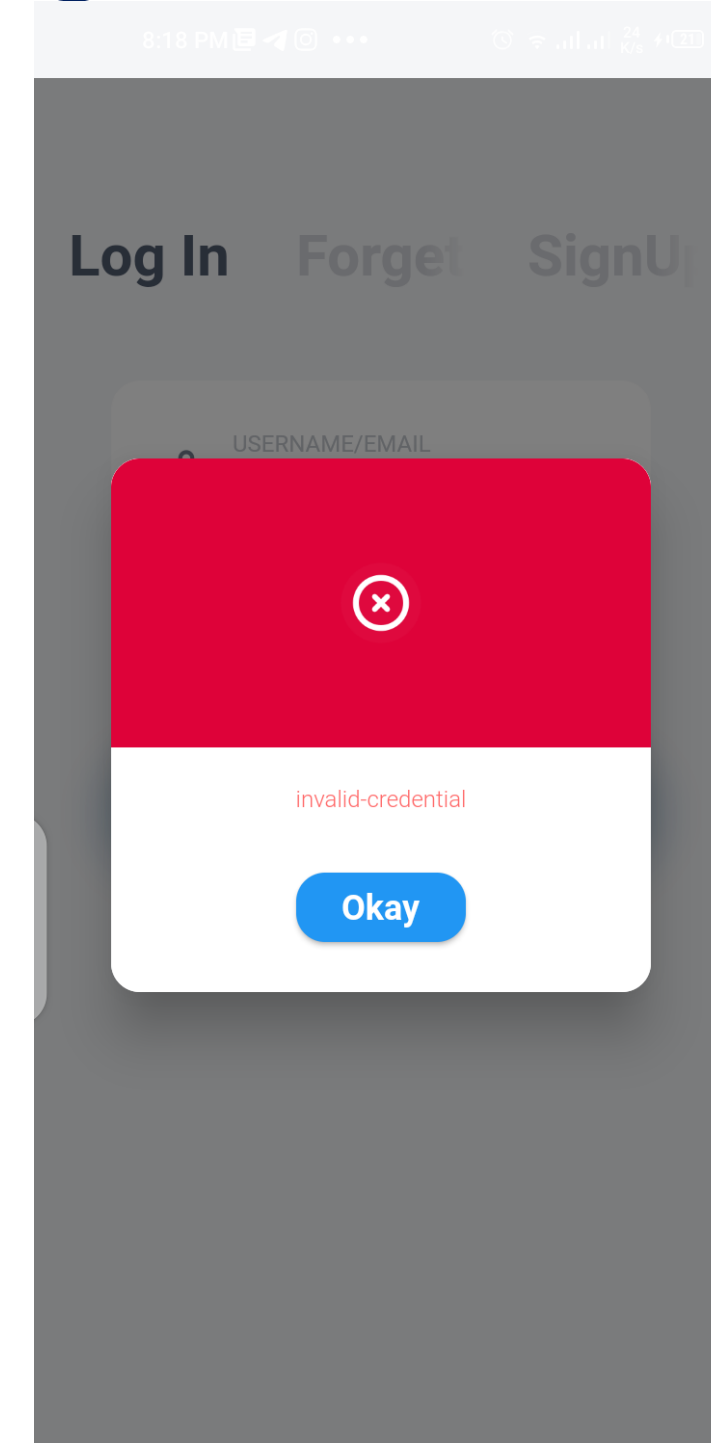
USERNAME/EMAIL  
fatima.abdelghany1@gmai

PASSWORD  
\*\*

Password must be 8

Log In

1 2 3 4 5 6 7 8 9 0  
q w e r t y u i o p  
a s d f g h j k l  
⬆ z x c v b n m ⬆  
?123 , . English ✓



8:18 PM

Log In Forget Sign Up

USERNAME/EMAIL

invalid-credential

Okay

# Registration Pages

**For Send and save Data :**

I use firebase packages

```
firebase_core: ^2.24.2  
firebase_auth: ^4.15.3  
cloud_firestore: ^4.13.6
```

First use firebase auth with email and password then save data to database of firebase and create collection of **users**

---

# Registration Pages

## Firestore signup:

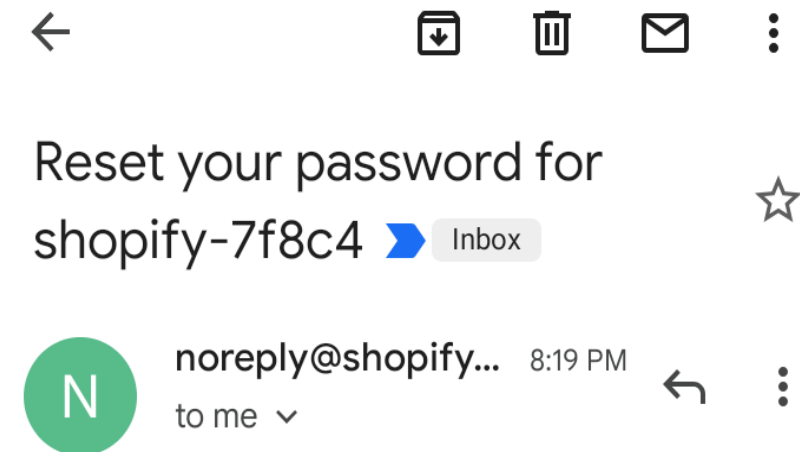
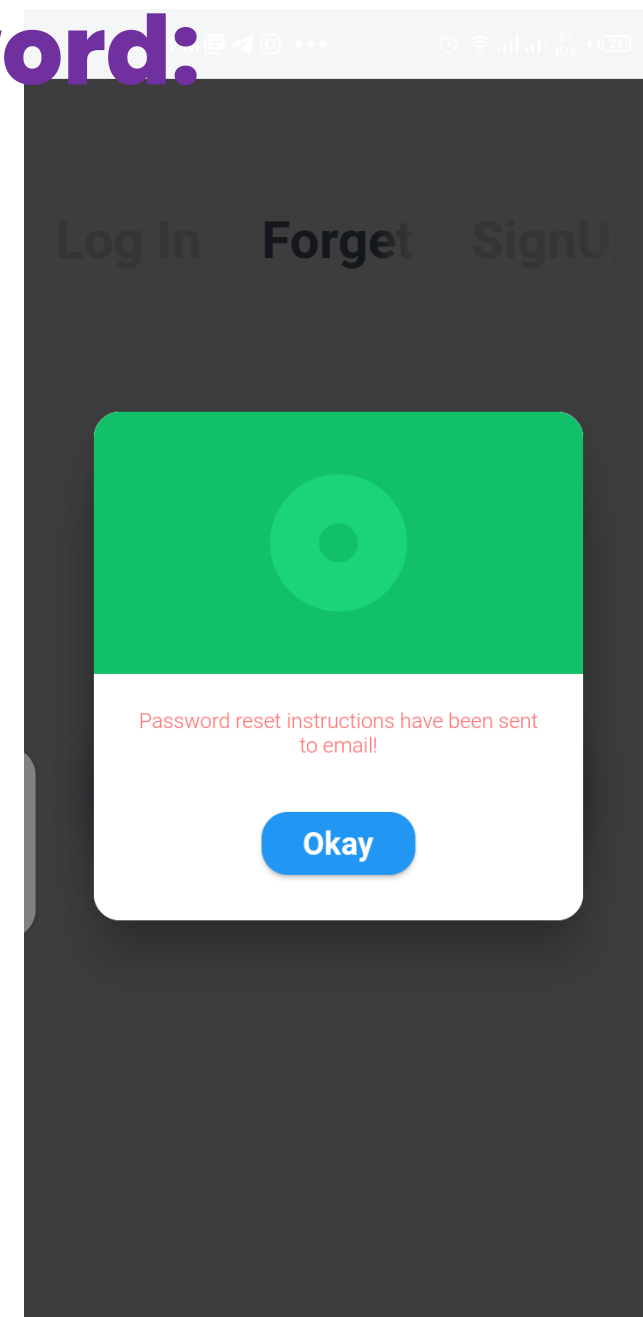
```
var credentials = await FirebaseAuth.instance
    .createUserWithEmailAndPassword(
      email: emailController.text, password: passwordController.text);
```

If user is auth set data in collection users

```
Navigator.pop(context);
if (credentials.user != null) {
  await FirebaseFirestore.instance
    .collection(CollectionsUtils.users.name)
    .doc(currentUser?.uid)
    .set({
      "name": nameController.text,
      "mail": emailController.text,
      "Address": "not now",
      "phone": phoneController.text,
      "image": "any",
    });
}
```

# Registration Pages

## Firestore ForgetPassword:



Hello,

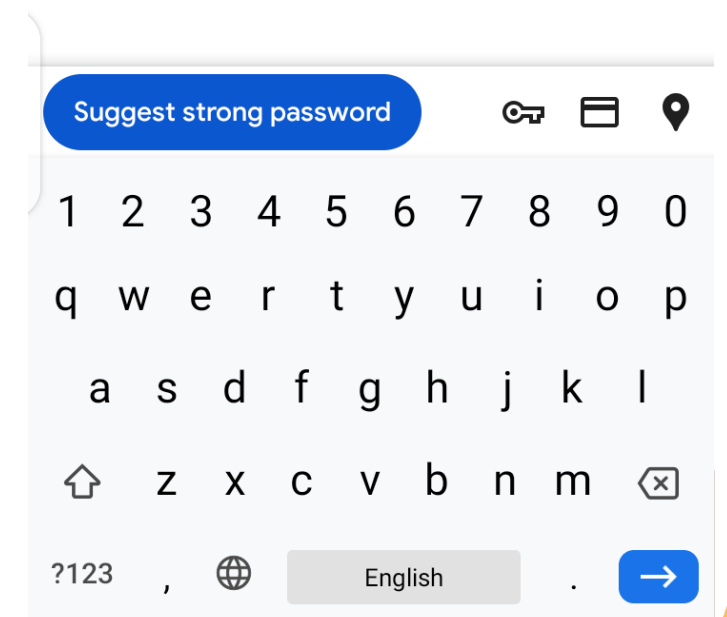
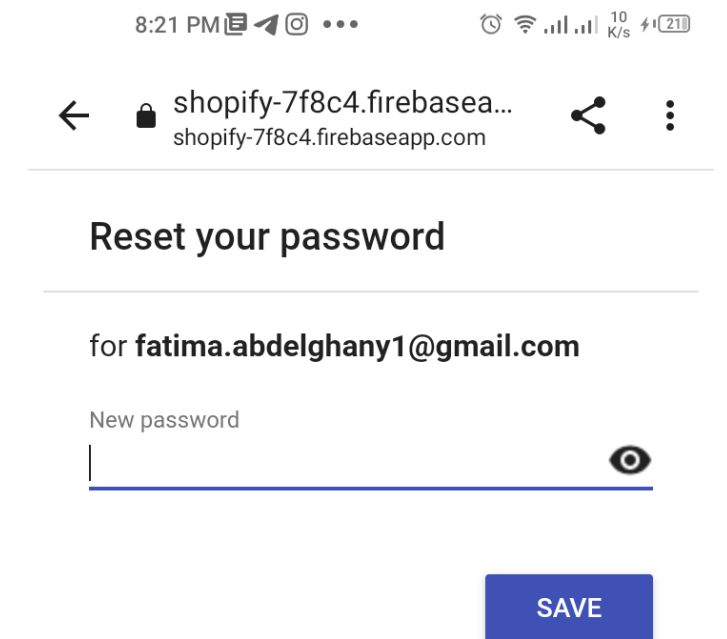
Follow this link to reset your shopify-7f8c4 password for your [fatima.abdelghany1@gmail.com](mailto:fatima.abdelghany1@gmail.com) account.

[https://shopify-7f8c4.firebaseio.com/\\_/auth/action?mode=resetPassword&oobCode=nCCxSlZ6FY0vEv7aYe16XVryzqKEayj2y\\_5Pjajy\\_9UAAAGM-cBbBw&apiKey=AlzaSyAJMLBgDoszvfblFTZ2Y6afWol-QHJ7I&lang=en](https://shopify-7f8c4.firebaseio.com/_/auth/action?mode=resetPassword&oobCode=nCCxSlZ6FY0vEv7aYe16XVryzqKEayj2y_5Pjajy_9UAAAGM-cBbBw&apiKey=AlzaSyAJMLBgDoszvfblFTZ2Y6afWol-QHJ7I&lang=en)

If you didn't ask to reset your password, you can ignore this email.

Thanks,

Your shopify-7f8c4 team



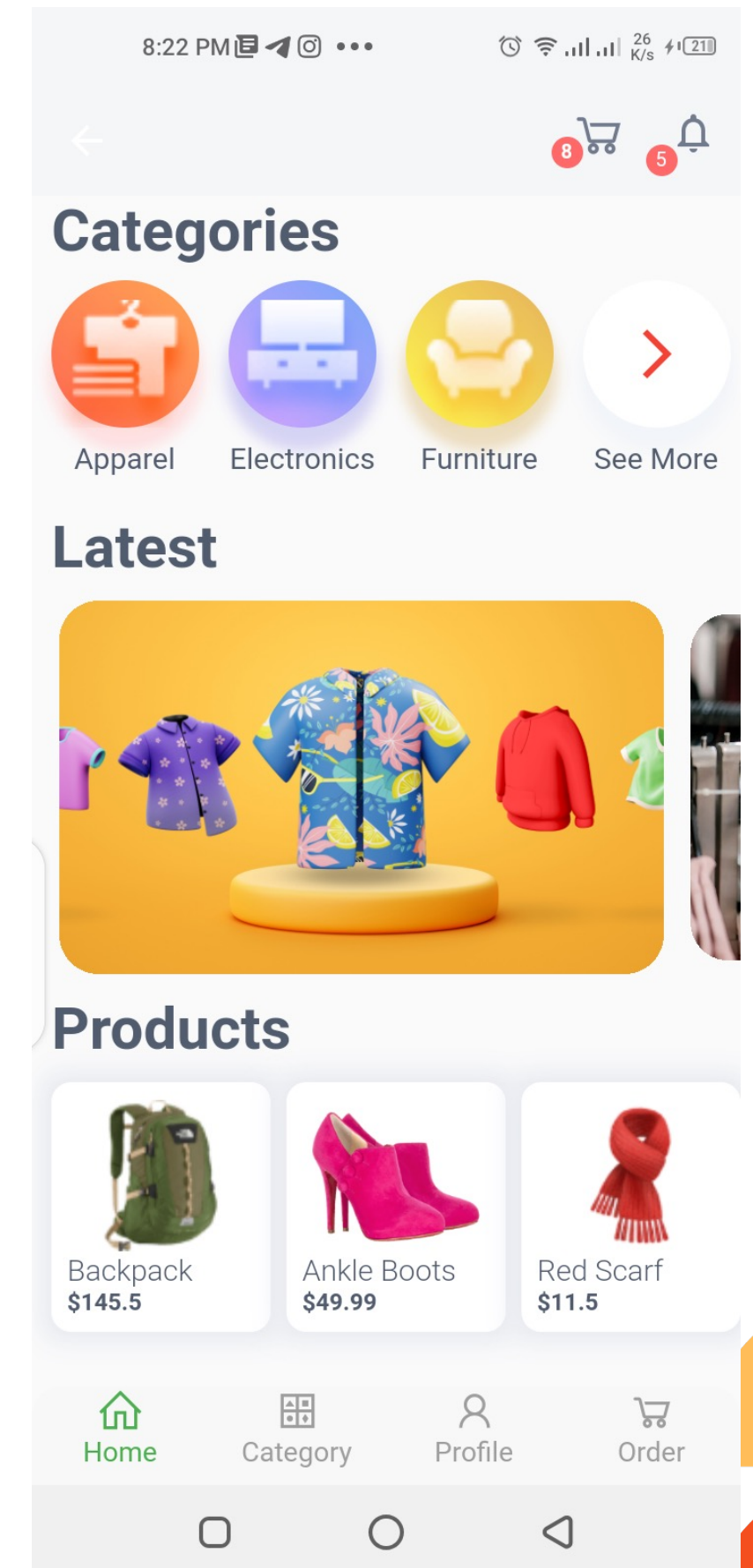
```
await FirebaseAuth.instance
    .sendPasswordResetEmail(email: emailController.text);
```

# Registration Pages

## Firestore Login:

If user entered login data correctly then open home screen.

Using changeNotifierProvider to send and get data from design to firebase and vice versa

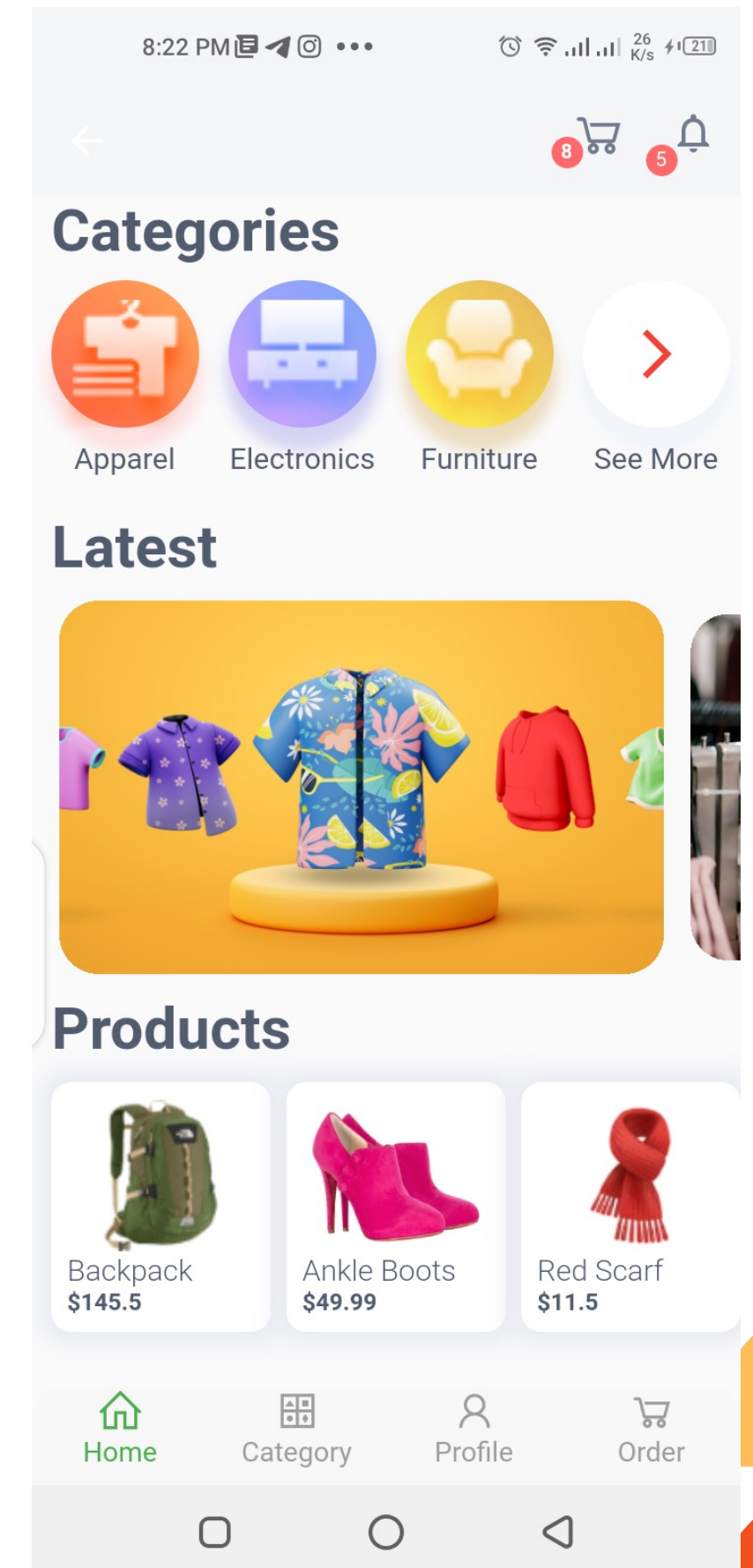




# Home Screen

## bottom Navigation Bar :

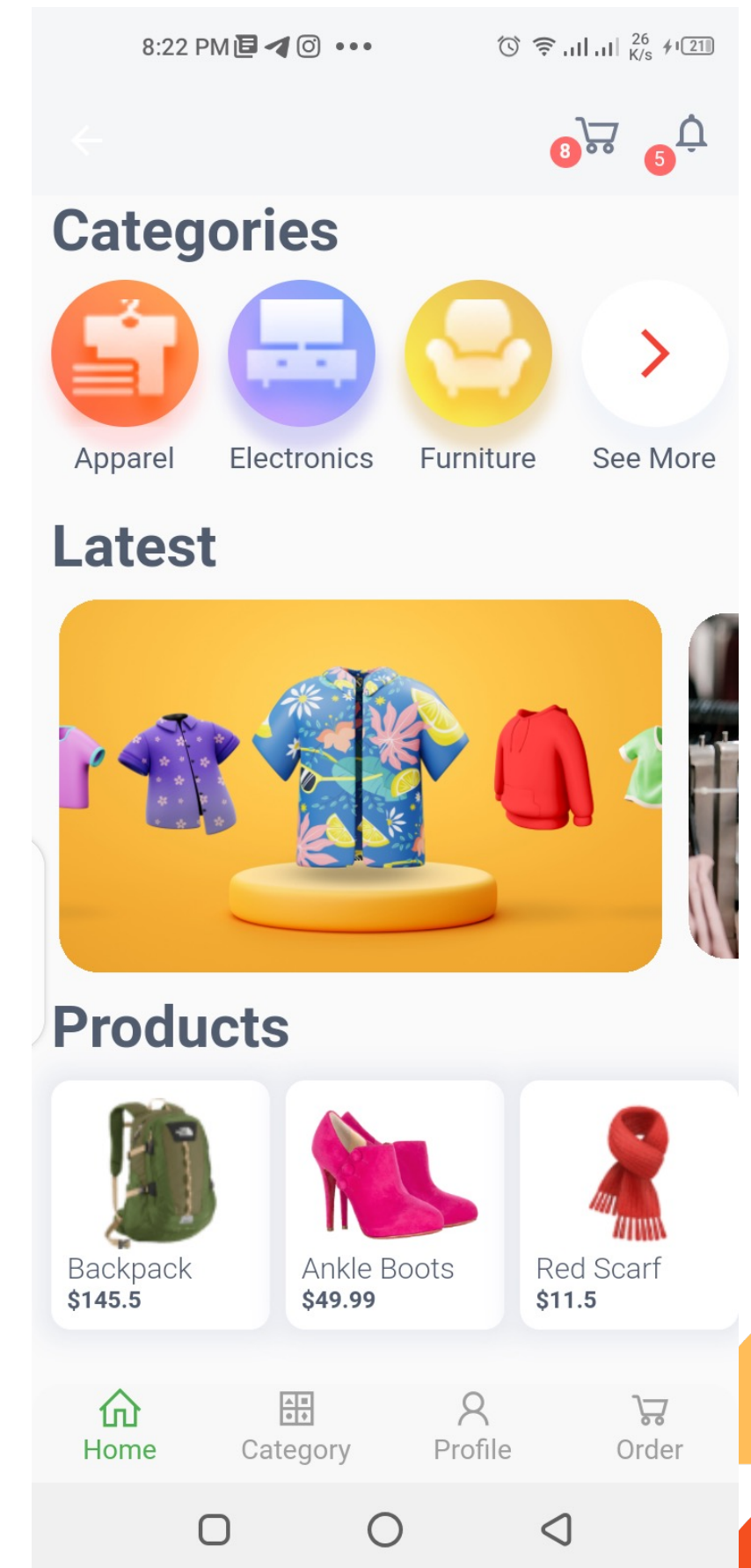
Bottom Navigation Bar is a navigation component displaying three to five destinations at the bottom of a screen. Each destination is usually represented by an icon and an optional text label. When a bottom navigation icon is tapped, the user is taken to the top-level navigation destination associated with that icon. Bottom Tab Bar is another navigation component used to switch between different screens in an app. It is typically used with a TabBar and a TabBarController.



# Home Screen

bottom Navigation Bar :

Using  
`animated_bottom_navigation_bar: ^1.3.0`  
Package For bottom Navigation Bar



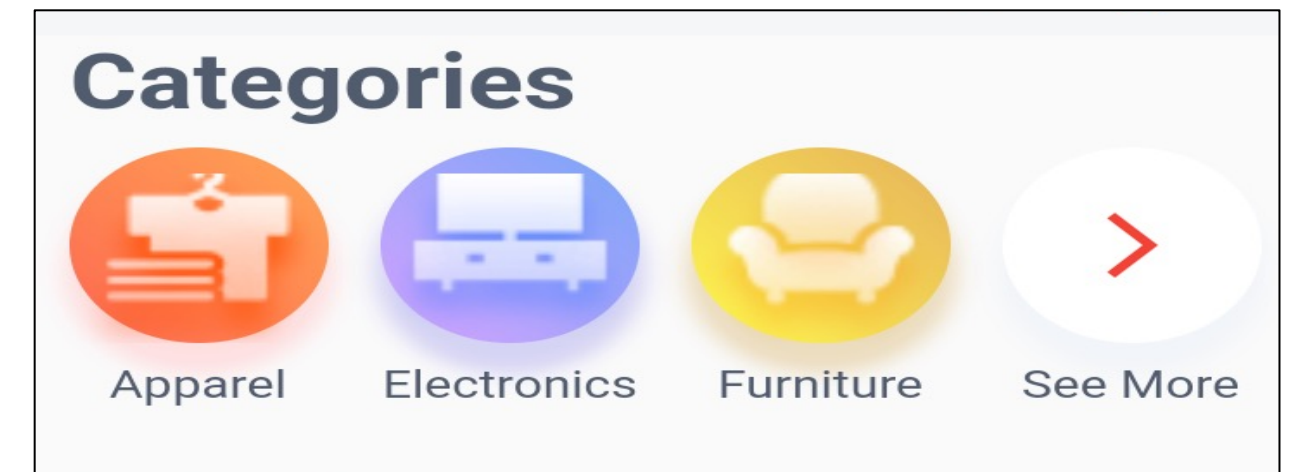


# Home Screen

## Categories:

In Home Screen get just 3 Categories By using `getCategories` method in `CategoryProvider` if user clicked in `SeeMore` get all Categories

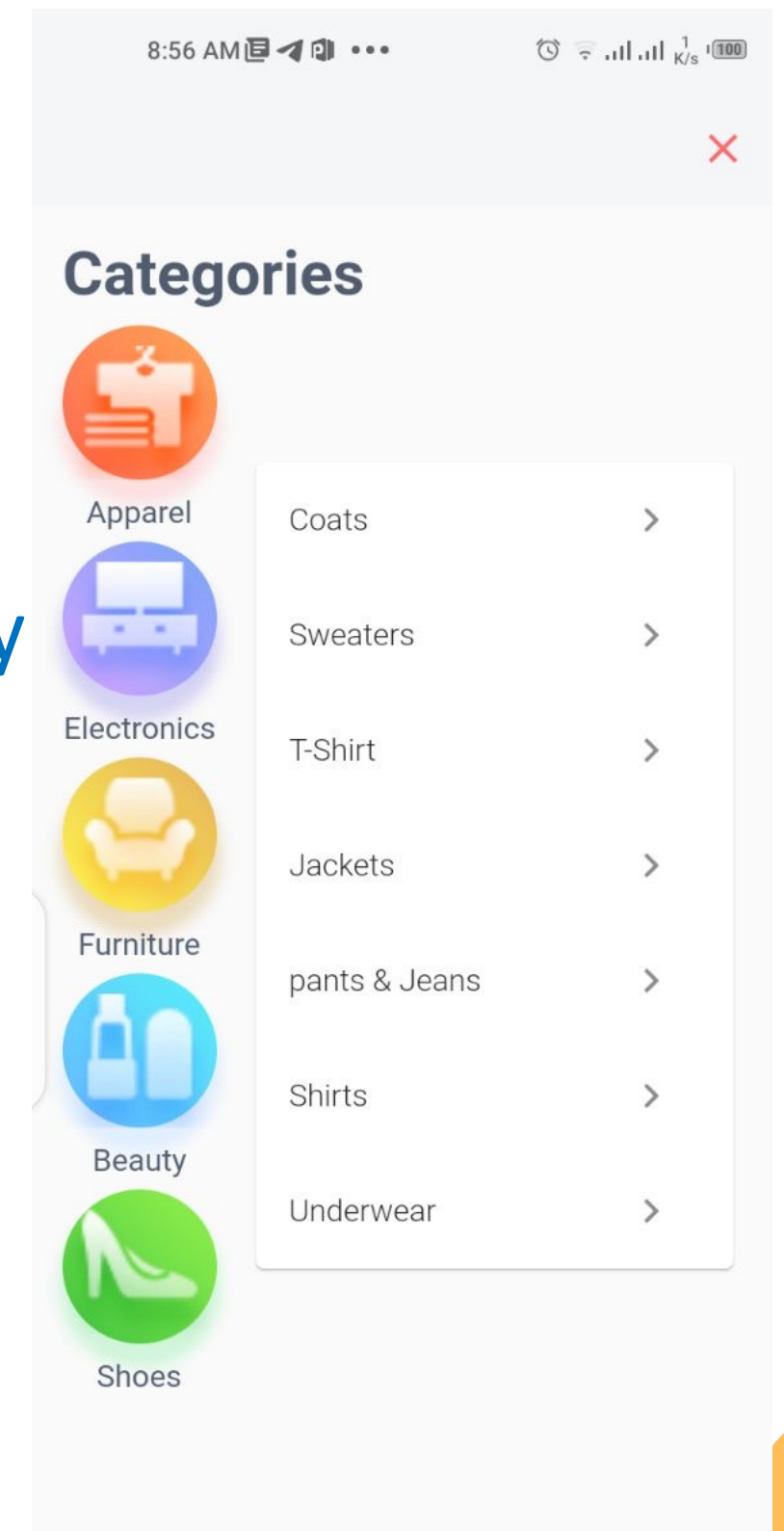
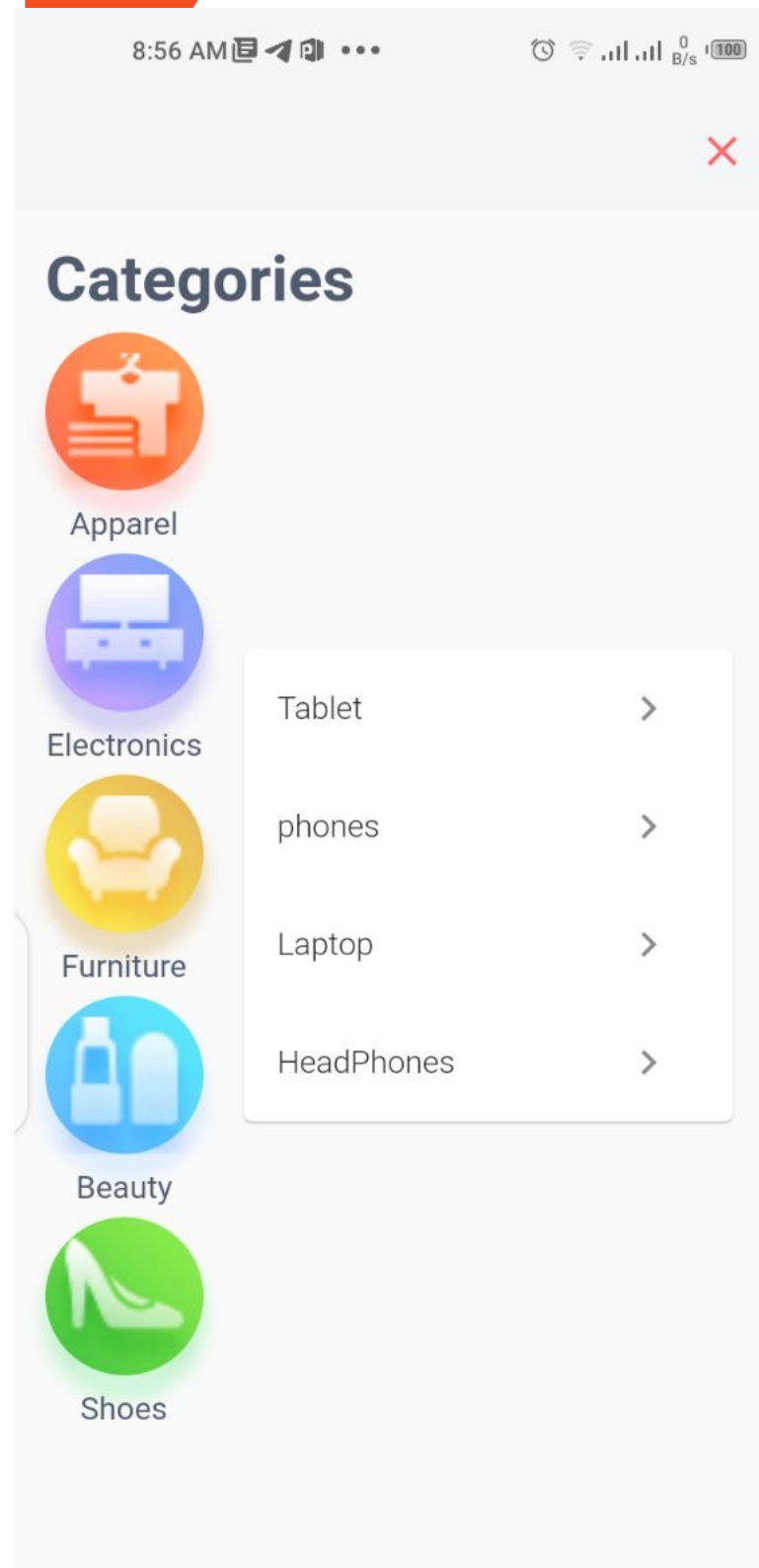
```
[limit: limit]) async {  
  try {  
    QuerySnapshot<Map<String, dynamic>>? result;  
    if (limit != null) {  
      result = await FirebaseFirestore.instance  
        .collection(CollectionsUtils.categories.name)  
        .limit(limit)  
        .get();  
    } else {  
      result = await FirebaseFirestore.instance  
        .collection(CollectionsUtils.categories.name)  
        .get();  
    }  
  }  
}
```



# Categories Screen

## Categories:

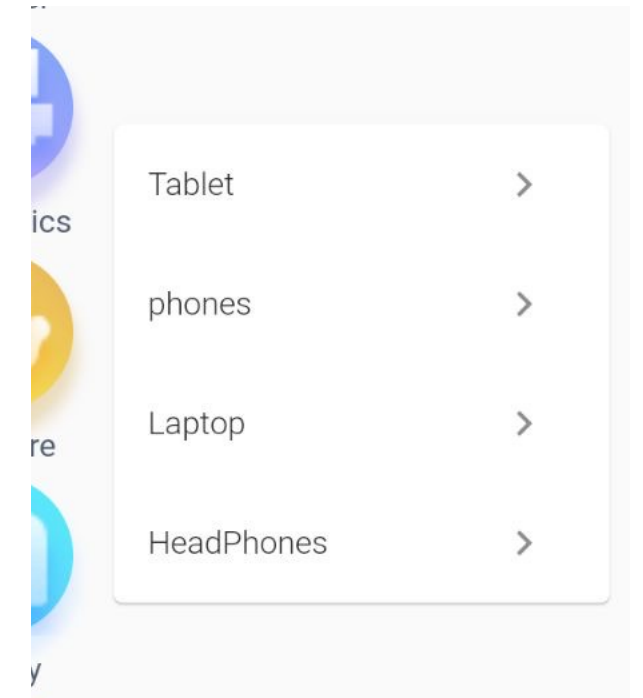
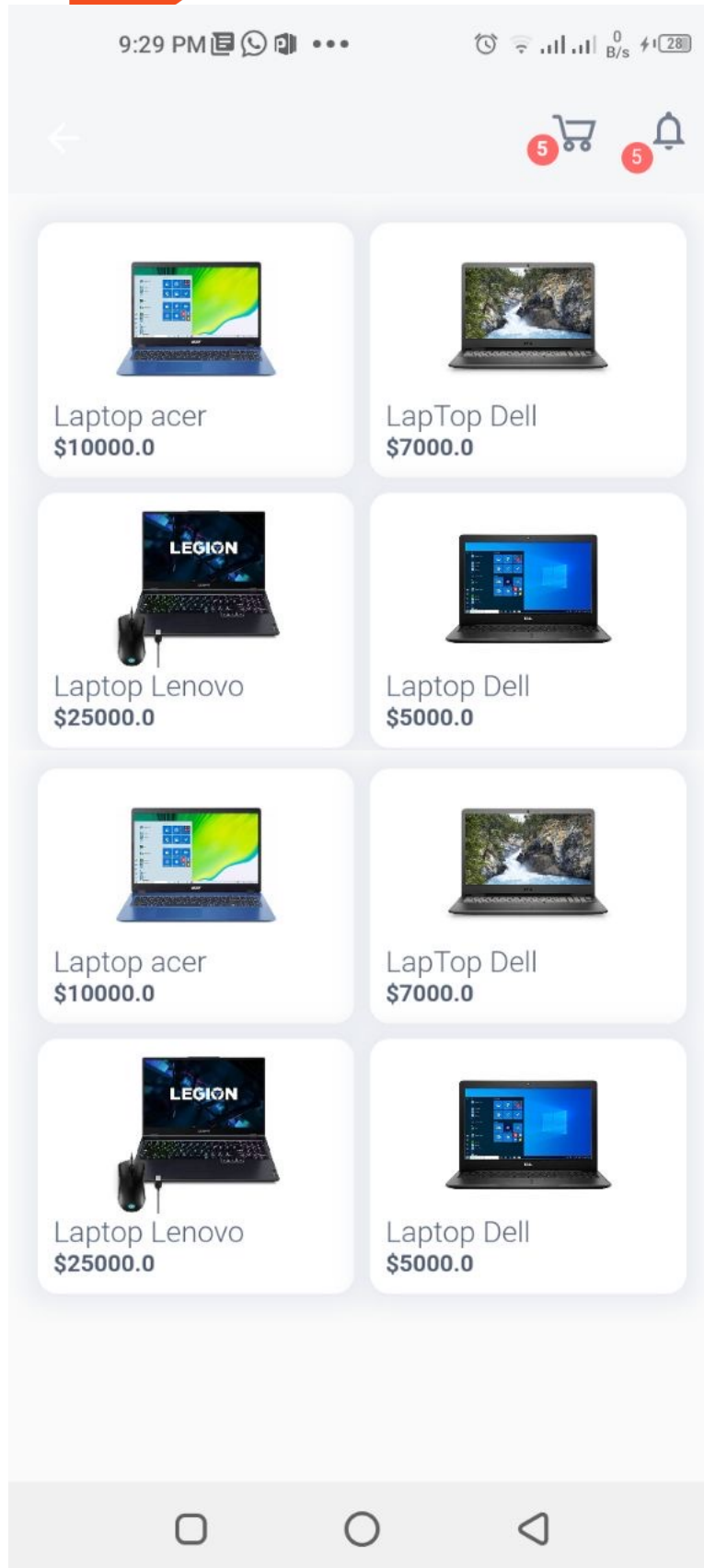
If Clicked on Category sub Category changes according to CategoryId so I add getSubCategory By Category Id to get Data in CategoryProvider



# Categories Screen

## Categories:

If select Category Electronics and then choose sub category Laptop will getting all Laptops

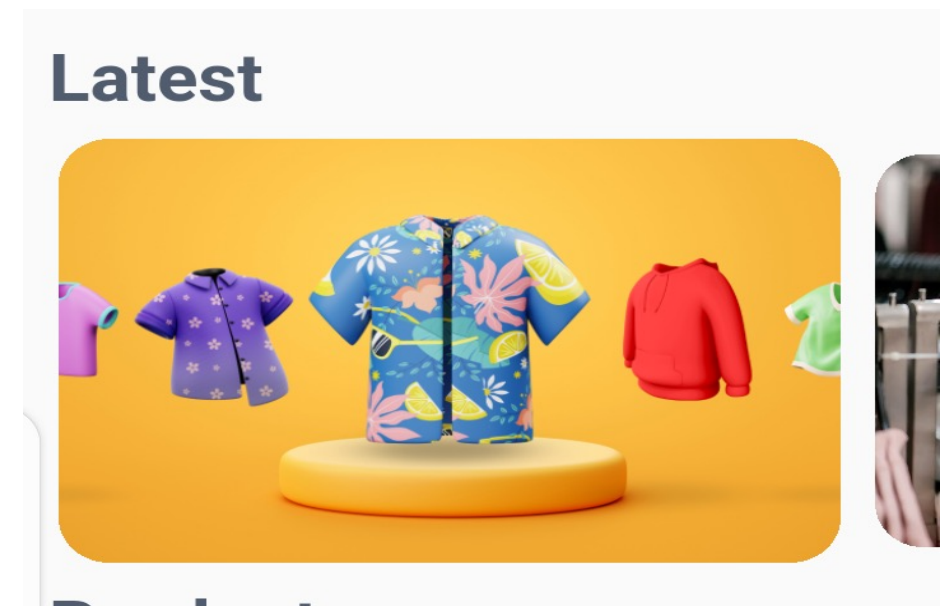


# Home Screen

## Ads:

For Ads I use `carousel_slider: ^4.2.1`  
`packget`

The page view widget is required to implement image sliding functionality in the app and the image views to show actual images. Other than that, you need a container widget to implement a page indicator on the bottom of the slider.



# Home Screen

## Ads:

Getting List of image urls from Firebase Ads collection then pass this list to Carousel slider and set some properties to CarouselOptions like scrollDirection, reverse, autoPlay and onPageChanged to know which image is selected

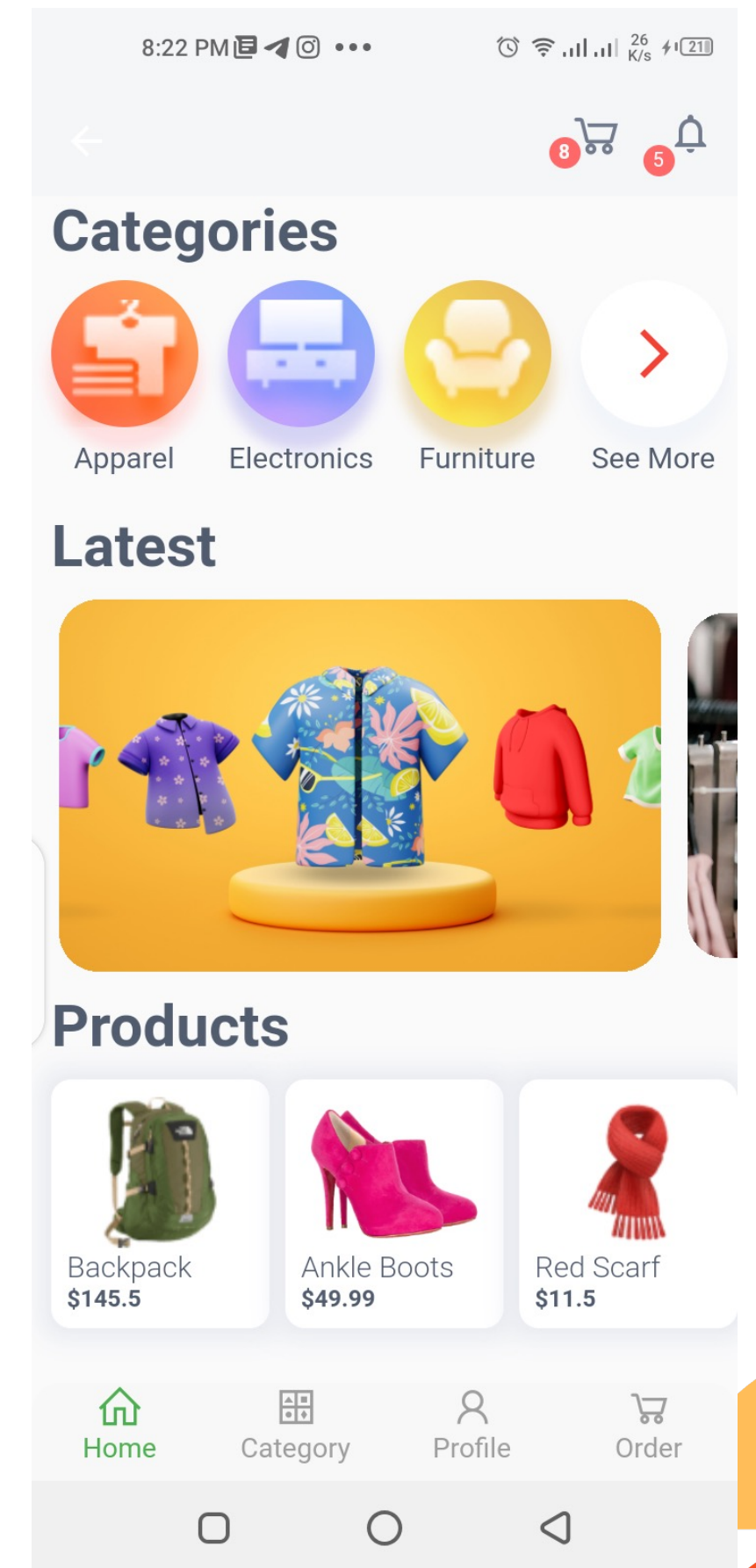
```
CarouselSlider.builder(  
  itemCount: widget.imageUrls.length,  
  itemBuilder: (BuildContext context, int index, int pageViewIndex)  
    => buildSliderImage(index),  
  options: CarouselOptions(  
    height: 190,  
    viewportFraction: .9,  
    padEnds: false,  
    initialPage: 0,  
    enableInfiniteScroll: true,  
    reverse: false,  
    autoPlay: true,  
    autoPlayInterval: const Duration(seconds: 3),  
    autoPlayAnimationDuration: const Duration(milliseconds: 800),  
    autoPlayCurve: Curves.fastOutSlowIn,  
    enlargeCenterPage: true,  
    enlargeFactor: 0.15,  
    onPageChanged: (index, _) {  
      index = index;  
      setState(() {});  
    },  
    scrollDirection: Axis.horizontal,  
  ),  
);
```



# Home Screen

## Products:

Using FutureBuilder to get Products from Firebase Collection products and using **FlexibleGridView** package to build product design if specific product click push to ProductDetails Page



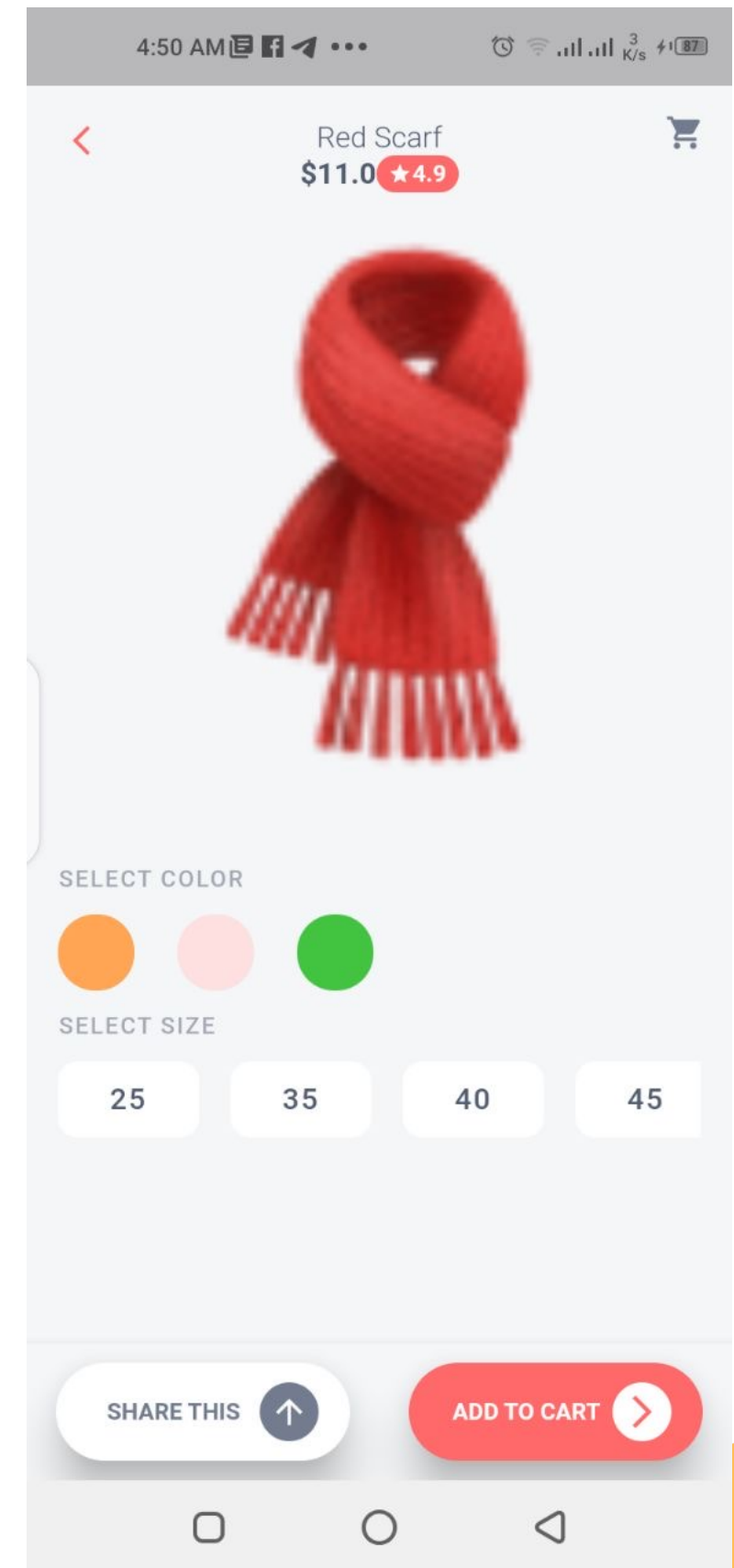
# ProductDetails Screen

## Product Details:

Using FutureBuilder to get Products from Firebase Collection products and using **FlexibleGridView** package to build product design if specific product clicked push to ProductDetails Page to see all details and select color or size

Using **CachedNetworkImage** package

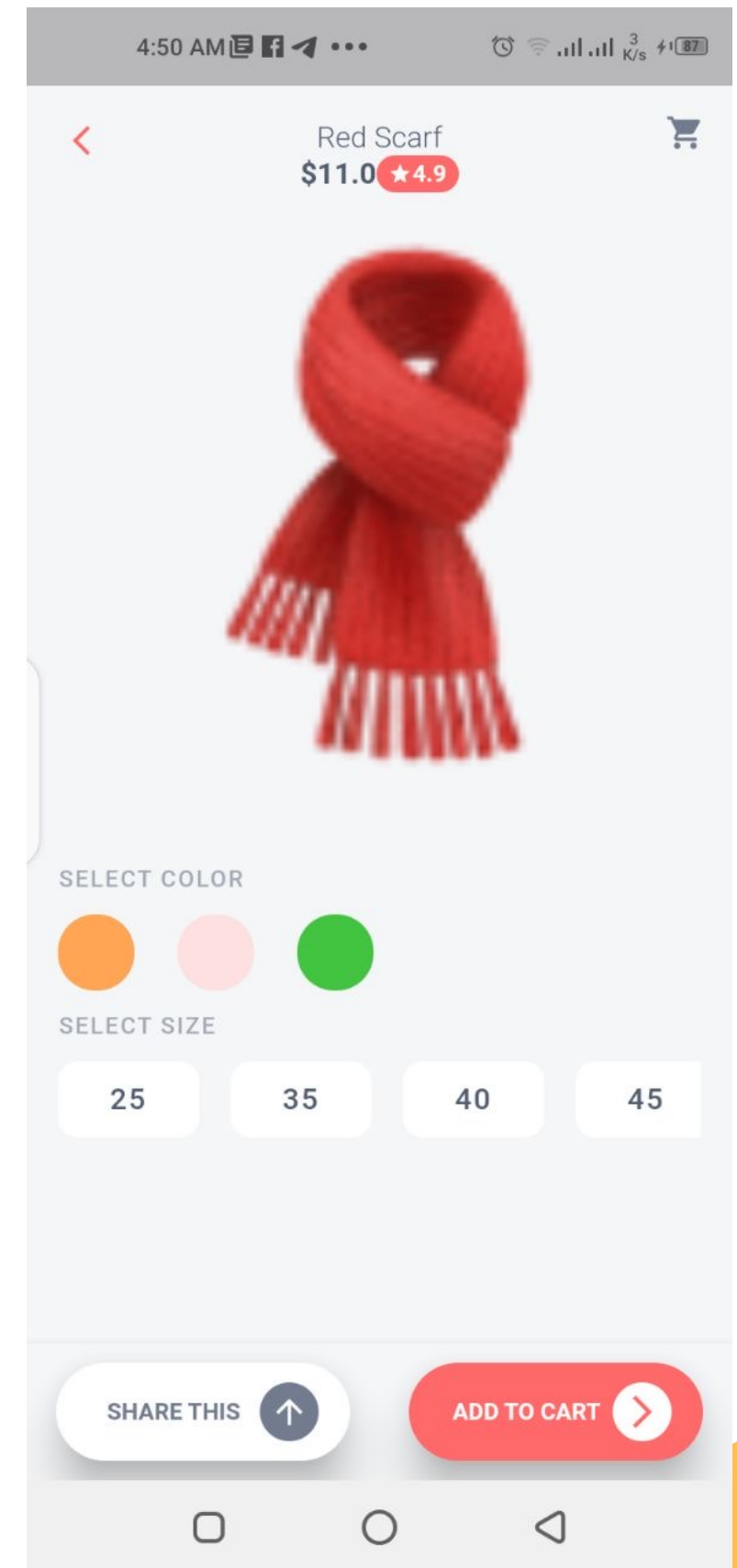
```
CachedNetworkImage(  
  imageUrl: "http://via.placeholder.com/350x150",  
  progressIndicatorBuilder: (context, url, downloadProgress) =>  
    CircularProgressIndicator(value: downloadProgress.progress),  
  errorWidget: (context, url, error) => Icon(Icons.error),  
),
```



# ProductDetails Screen

## Product Details:

Using `BottomAppBar` to build add to cart if  
add this product to cart pushing to Cart  
Page

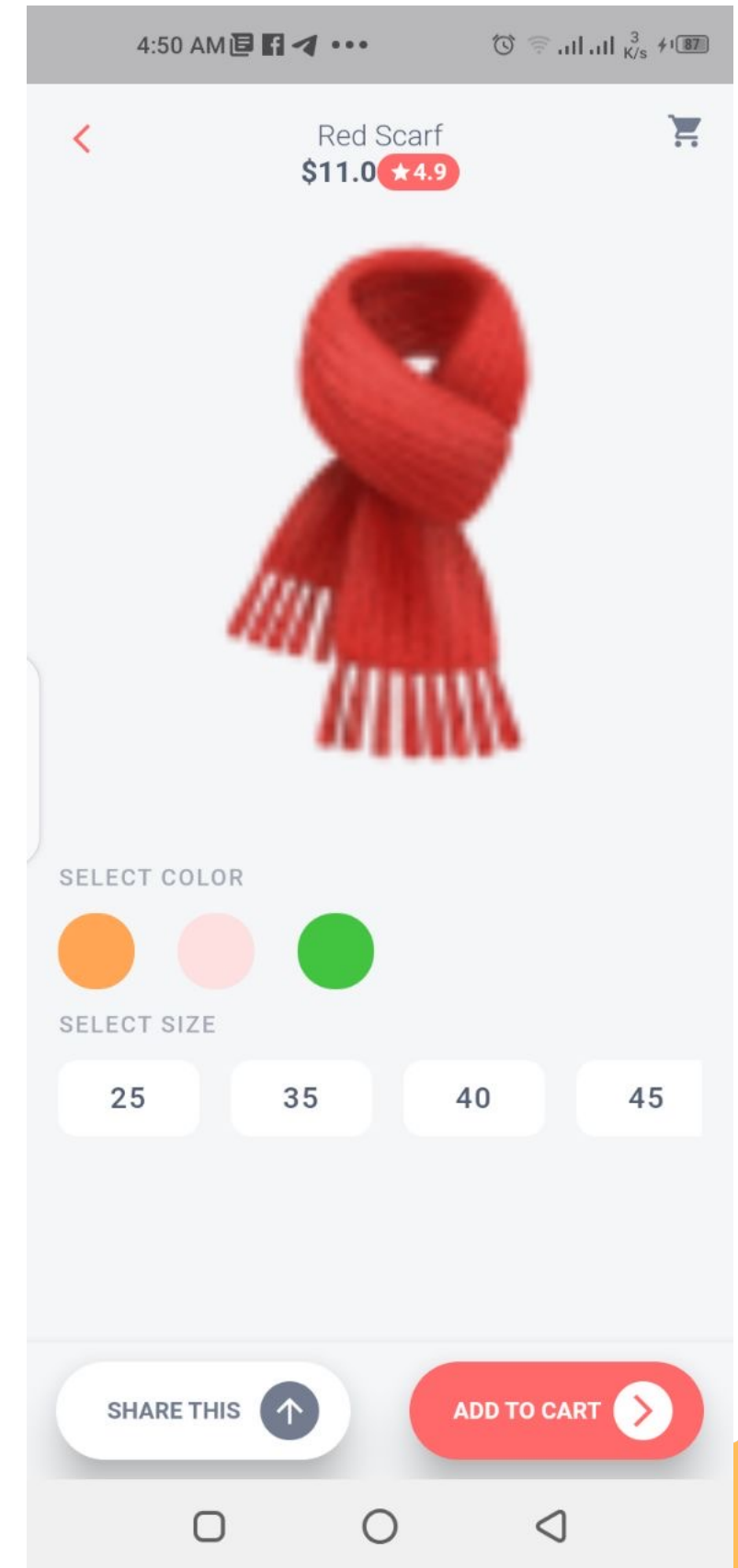




# Cart Screen

Using **BottomAppBar** to build add to cart if add this product to cart pushing to Cart Page .

Using **CartProvider** to add product to firebase

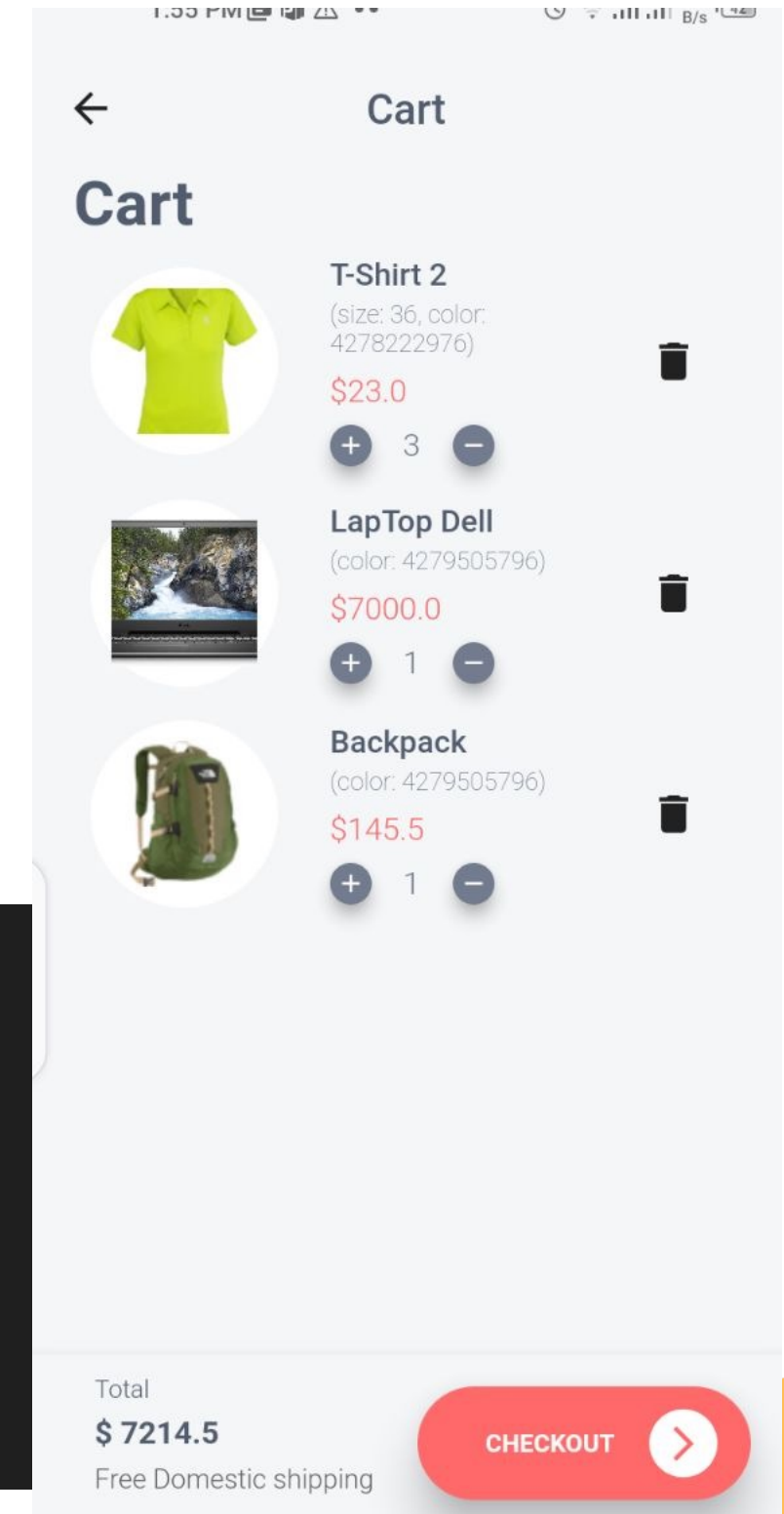


# Cart Screen

Get all products in cart:

Open Stream on cart collection and get item of user by his email

```
Stream<DocumentSnapshot<Map<String, dynamic>>> get cartStream =>
  FirebaseFirestore.instance
    .collection(CollectionsUtils.cart.name)
    .doc(FirebaseAuth.instance.currentUser?.email ?? '')
    .snapshots();
```

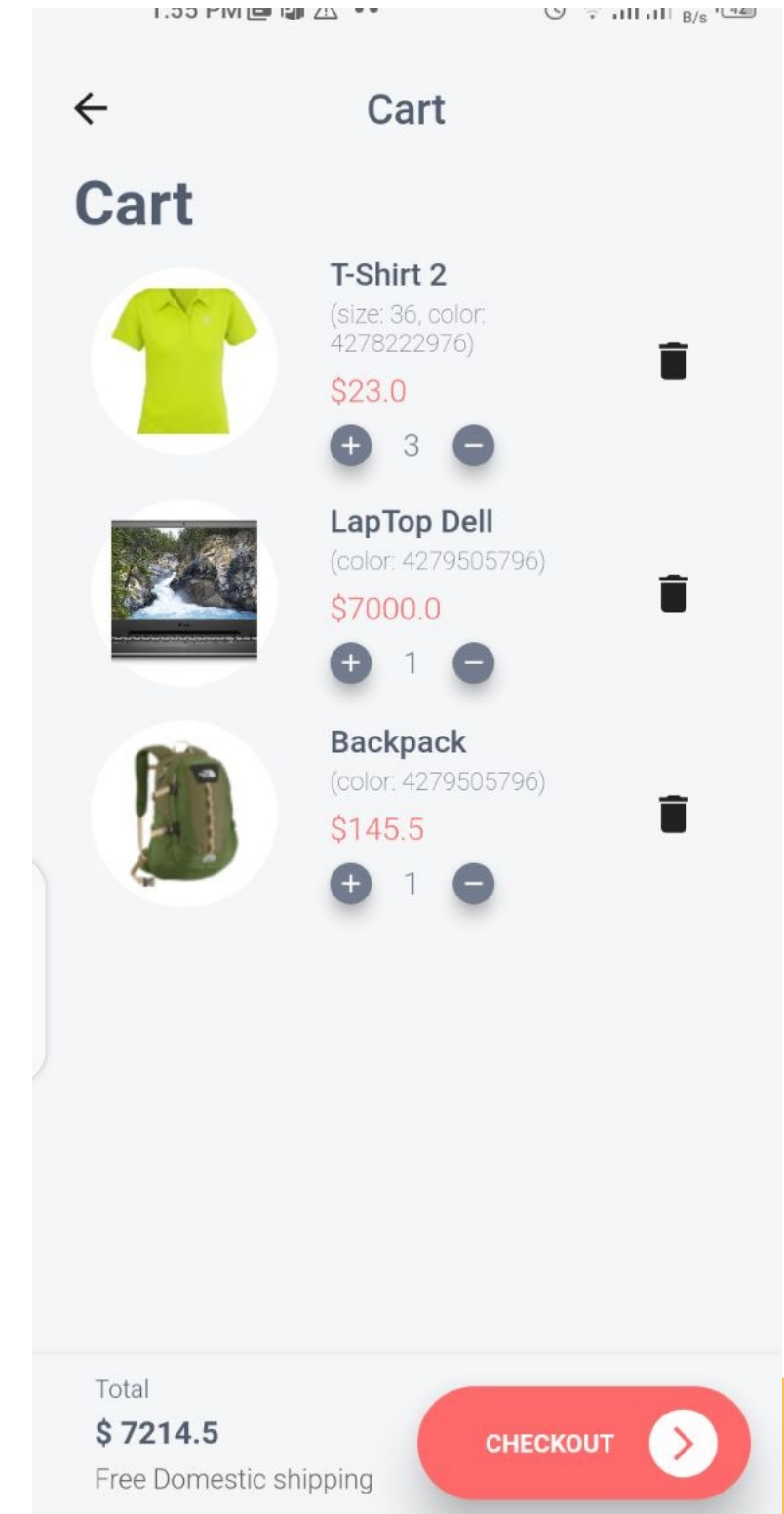


# Cart Screen

Increase product and decrease:

To increase or decrease item get this product quantity and change this Quantity then update this product in collection cart

```
await FirebaseFirestore.instance
  .collection(CollectionsUtils.cart.name)
  .doc(FirebaseAuth.instance.currentUser?.email ?? '')
  .update(cart.toJson())
  .then((value) => Navigator.pop(context));
calculateTotal(cart);
```



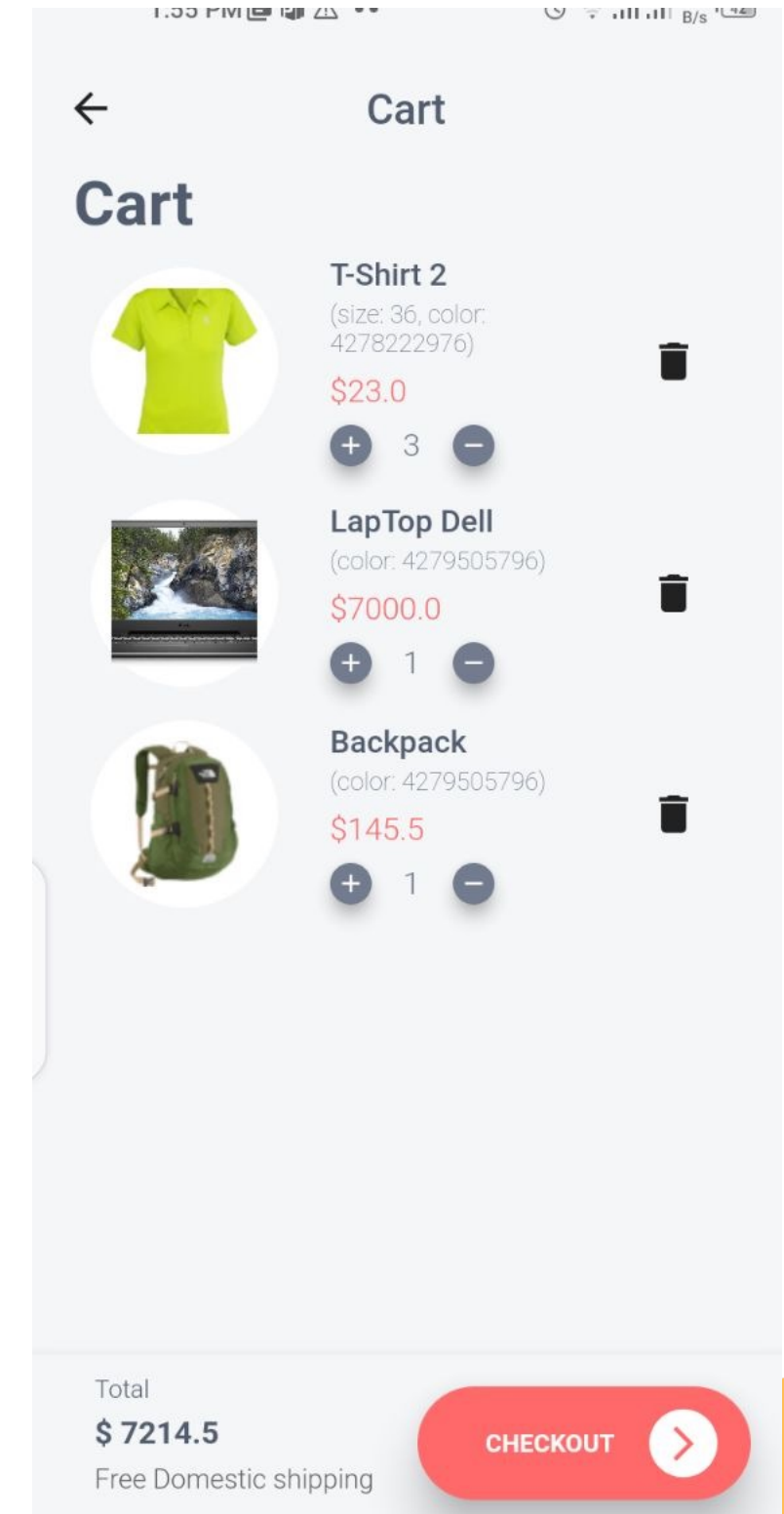
# Cart Screen

## Delete Product:

To remove item in cart showing alert Dialog to confirm deleting or not then select item and remove it then update the cart

```
QuickAlert.show(context: context, type: QuickAlertType.loading);
cart.items?.removeWhere((element) => element.itemId == itemId);

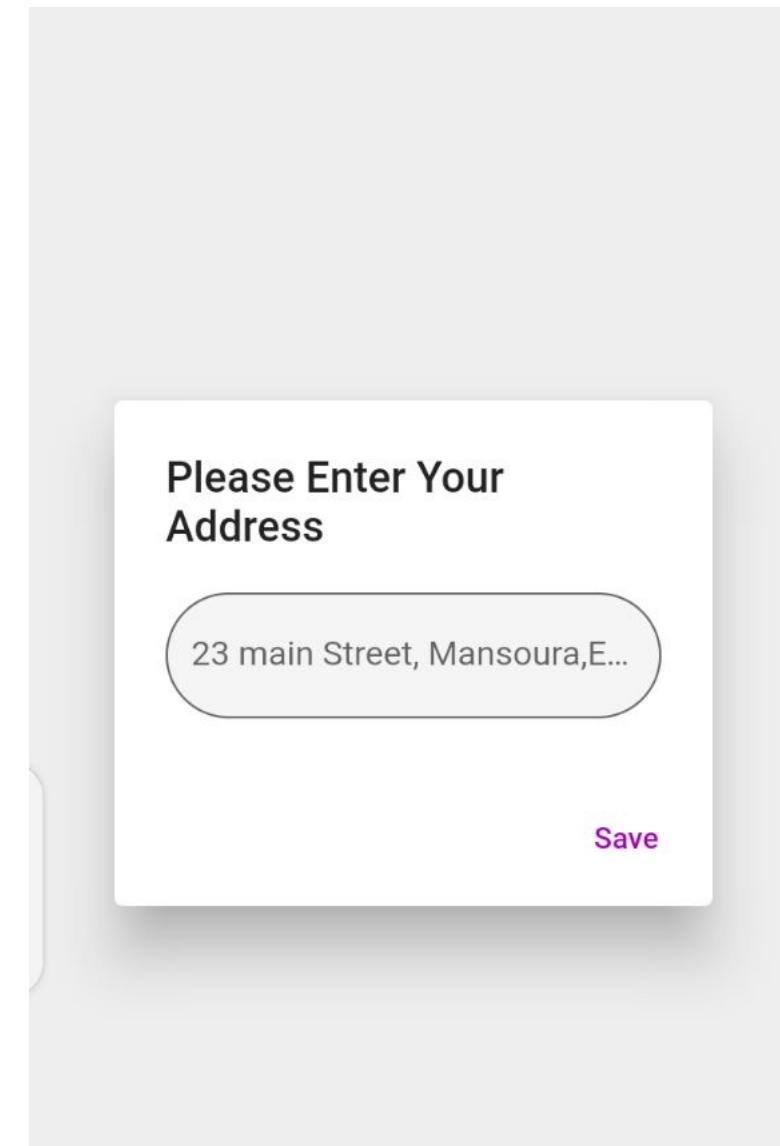
await FirebaseFirestore.instance
  .collection(CollectionsUtils.cart.name)
  .doc(FirebaseAuth.instance.currentUser?.email ?? '')
  .update(cart.toJson());
```



# Cart Screen

CheckOut Order:

Show **AlertDialog** to add address when save  
update User Data from SignUPProvider

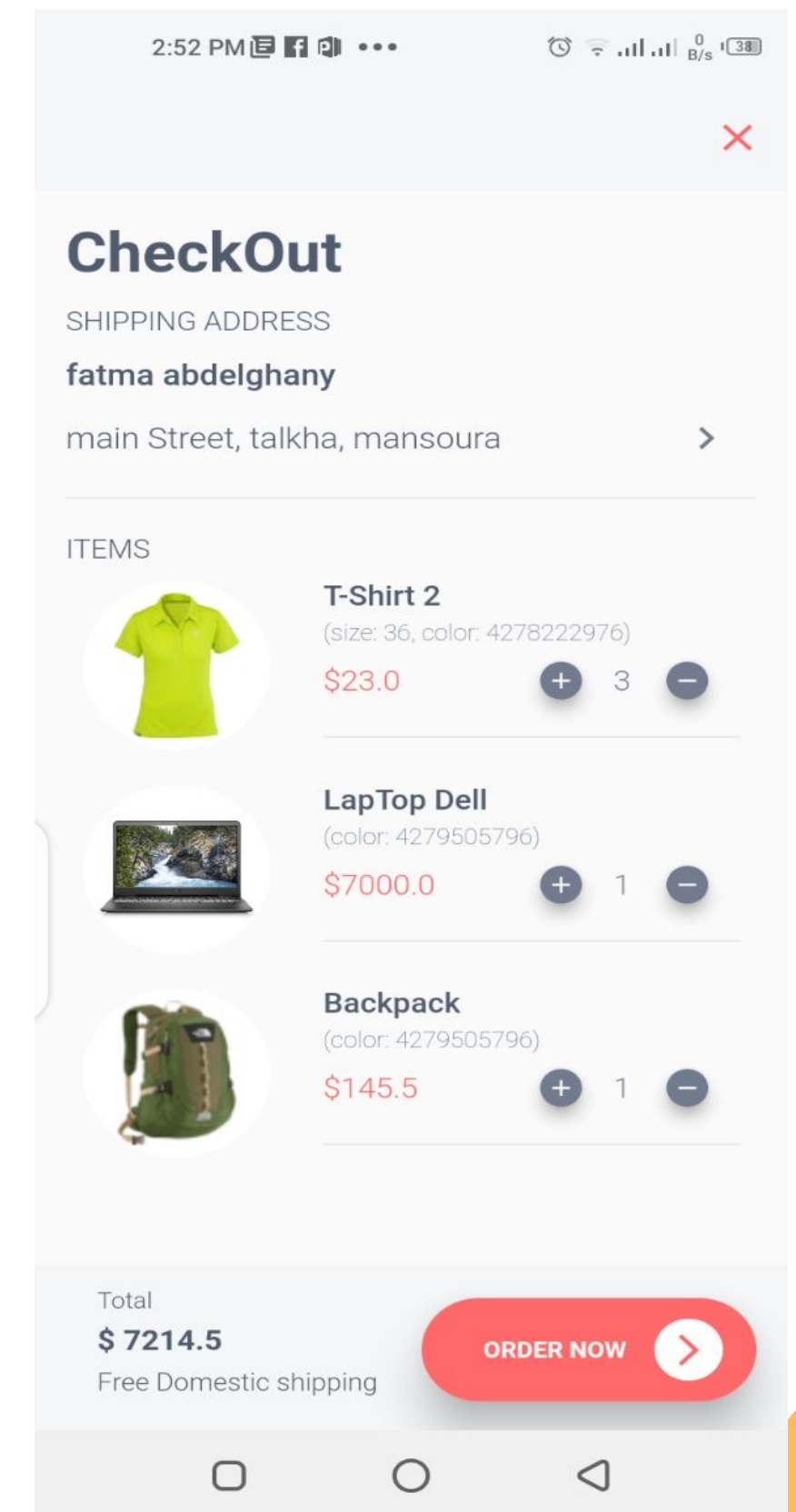




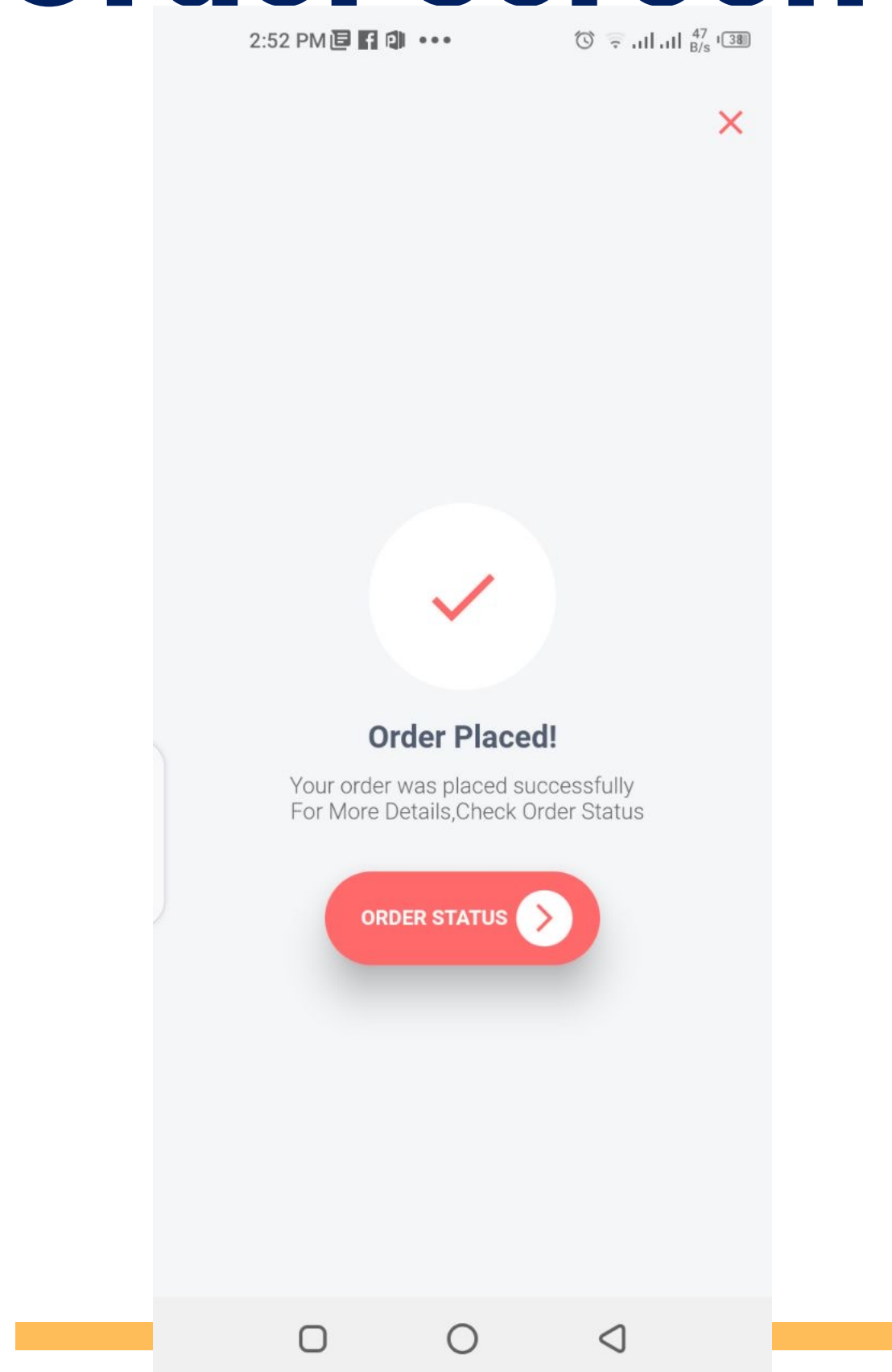
# Order Screen

## CheckOut Order:

Get User Data from SignUPProvider then save Order

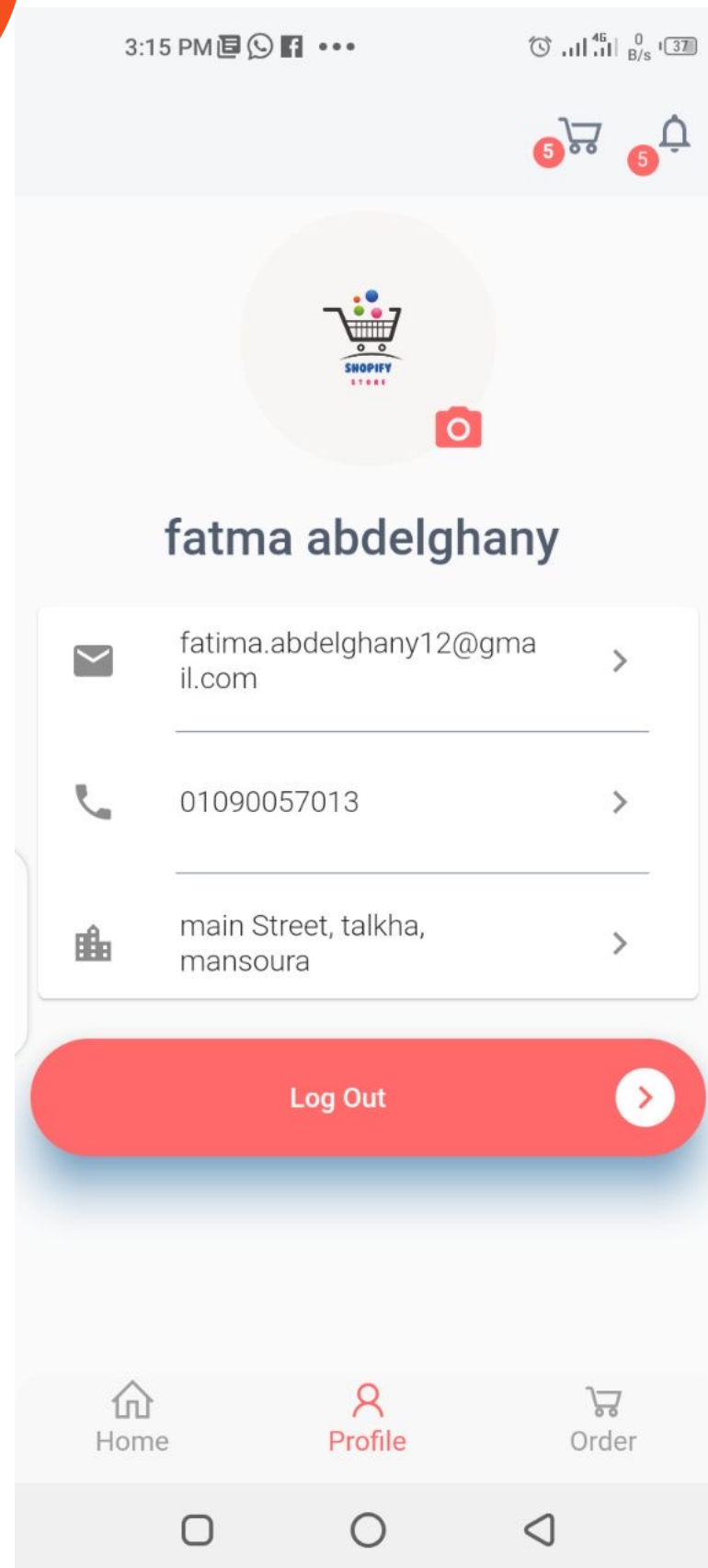


# Order Screen



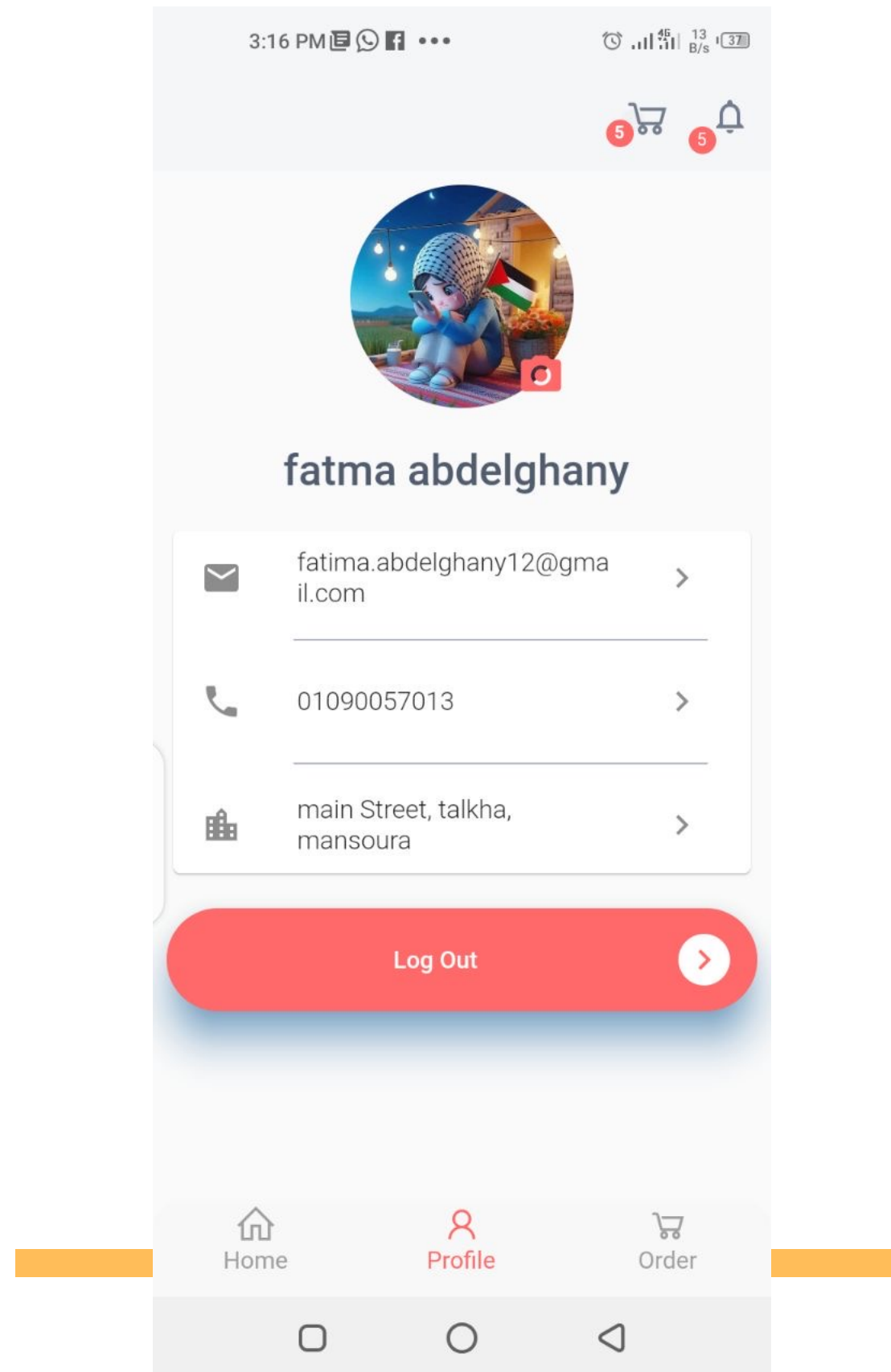
# profile Screen

show User Data from `SignUpProvider` and update his image and Logout from app



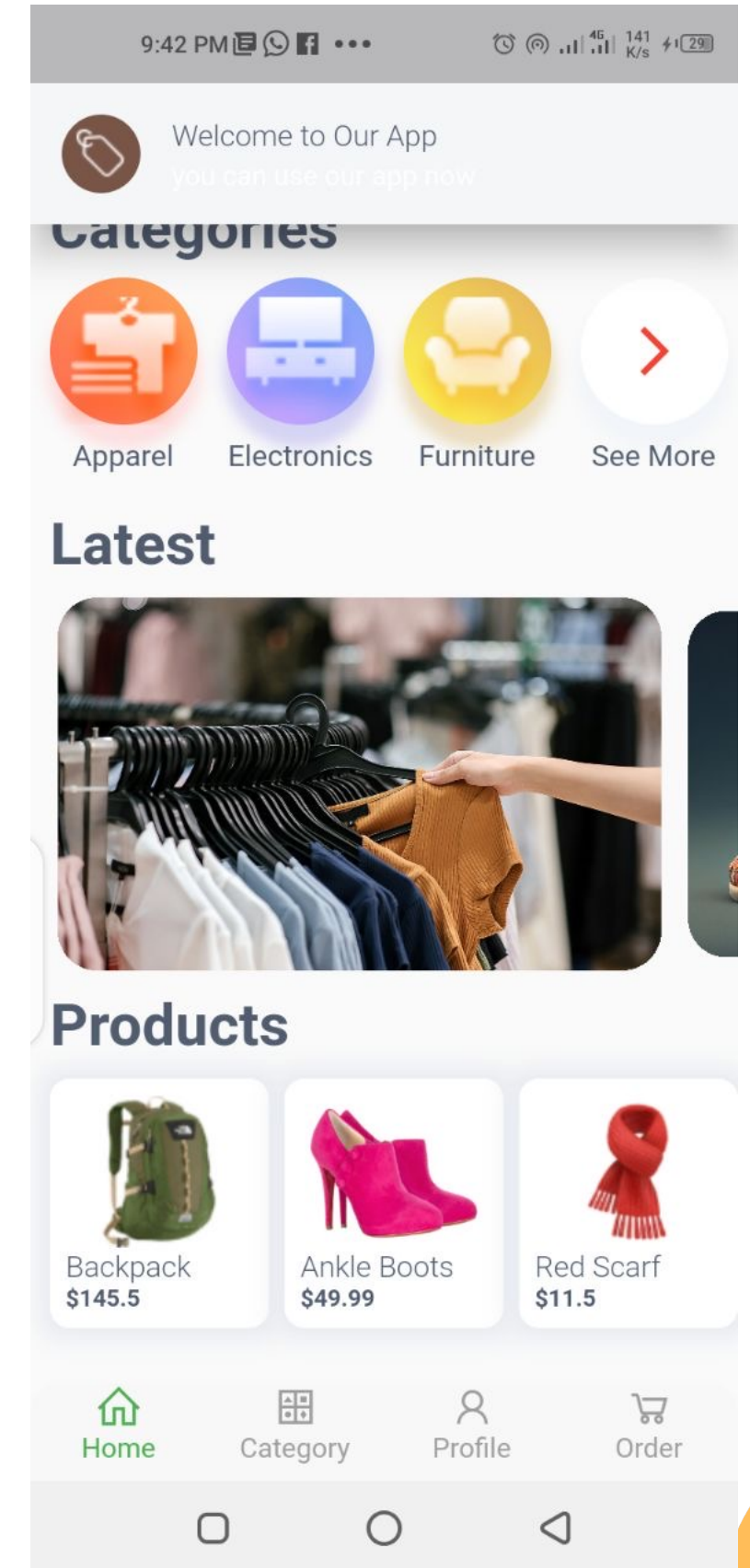


# profile Screen



# Push Notification Screen

When I receive notification add it to firebase collection Notification ,and see last Notifications in Notification screen



# Push Notification Screen

When I receive notification add it to firebase collection Notification ,and see last Notifications in Notification screen

