

Dharmsinh Desai University , Nadiad
Faculty of Technology
Department of Computer Engineering



B. Tech. CE Semester – VI

Subject: System Design Practice

Project Title: Simple to use app for managing earnings
and expenses

By: Tejas Dobariya , CE035, 20CEUOS048

Maharshi Mistry, CE004, 20CEUBS043

Guided by:
Prof. Brijesh Bhatt
Professor,
CE Dept, DDIT

Dharmsinh Desai University

College Road, NADIAD - 387001. (Gujarat)



Certificate

This is to certify that the practical / term work carried out in the subject of System Design Practice and recorded in this journal is the bonafide work of

Tejas Dobariya , CE035, 20CEUOS048

Maharshi Mistry, CE004, 20CEUBS043

of B.Tech semester VI in the branch of Computer Engineering
during the academic year 2022-2023.

Table of Content

1.<u>Abstract</u>	4
2.<u>Introduction</u>	5
<u>General Brief:-</u>	5
<u>Technology used to build the application are as follows</u>	6
3.<u>Software Requirement specification</u>	7
<u>Functional Requirement</u>	7
4.<u>Designs</u>	11
5.<u>Implementation Detail</u>	14
6.Testing	21
7.<u>Screen-shots</u>	23
8.<u>Conclusion</u>	33
9.<u>Limitation and Future Extension</u>	33
10.<u>Bibliography</u>	34

1. Abstract

The Purpose of this project is to develop a solution to everyday problem which is management of budget. To solve this problem, we need to keep in mind that whatever solution is developed it needs to be trackable, user friendly and easily accessible. With the data of solution to be secure and accessible to the rightful data owner.

2. Introduction

General Brief

The project is about to develop an application as a solution to the budget management problem. This application needs to contain all attributes of a budget management solution. These attributes are mainly trackability, security and data depiction.

The traditional solution lacks the trackability as it cannot manage to list all the expenses of the same type in one page and which may create a problem of loss of data which need to be taken into account. An untimely data in a traditional solution may create a problem of tracking of data while analyzing.

Security is one of the most important attributes of the solution. As it gives a sense of relief to the data owner that there wouldn't be any stealing of information, accessed by other than the rightful owner of the data. The traditional solution can solve this problem as it's all physical. Any unknown person can see the data of the owner. Can use that data wrongfully. This can be solve by placing it on modern devices which can give security to application

We come to one of the most important attributes; data depiction. The Traditional solution lacks the feature to produce as it has to be done manually which might be a tedious job to do. Data depiction use the data and create a graphical representation of data which help user to make better decision about the expenses and budgeting

Technology used to build the application are as follows:-

Android Studio

Github - To manage application version

Firebase- As database of application

3. Software Requirement specification

1. Manage Account

R.1.1 :- Register user

- Description: If a new user wants to Track of his expense, user needs to register himself/herself first.

R.1.1.1 :- Sign up

- input : Required Details like Username , email id, contact, number , password etc

- output : Account created

R.1.1.2 :- Sign in

-input : Username , password

-output : Logged in

R.1.1.3 :- display user information

-output : Display user's profile

2. Manage Earnings

R.2.1 :- Add INCOME

-Description: User will be able to add income

- input : Insert income value
- output : Value will be added.

R.2.2 :- Manage category

R.2.2.1 : Select Existing Category

-Description :- User can select one or more than one categories from the given list of categories.

-Input :-Select Categories

-Output:- Categories will be added to home page

R.2.3 Add Expense to categories

-Description :- Add the maximum limit of expense of each categories selected by User .

-Input:- set maximum limit of categories

-Output :- show limit for categories on home page

3. Manage Expense

R.3.1 Add Expense

R.3.1.1 expense details

-Description :- user can add his expense amount for the day in the categories of his choice.

-Input :- expense amount , category and date

-Output: Amount added

R.3.2 Alert Message

-Description: User receives notification every day. That reminds him to add your today's expenses.

-Output: show alert message everyday.

R.3.3 :- View Expenses

-Description: User can view their expense category-wise on daily as well as monthly basis.

R.3.3.1 :- Recent expenses

-Description:- user can see all the recent expenses.

R.3.3.2 :- Chart representation

- Description: view the expenses made on particular categories on daily and monthly basis.

-Output:- show chart

R.3.4 :- Delete Expense

- Description: User can delete particular transaction.

4 :- View Balance

- Description: View the total balance left after total expenses.

5:- Report

-Description:- Generates a report to give a detailed insight about budgets, income, balance.

Non-functional Requirements:

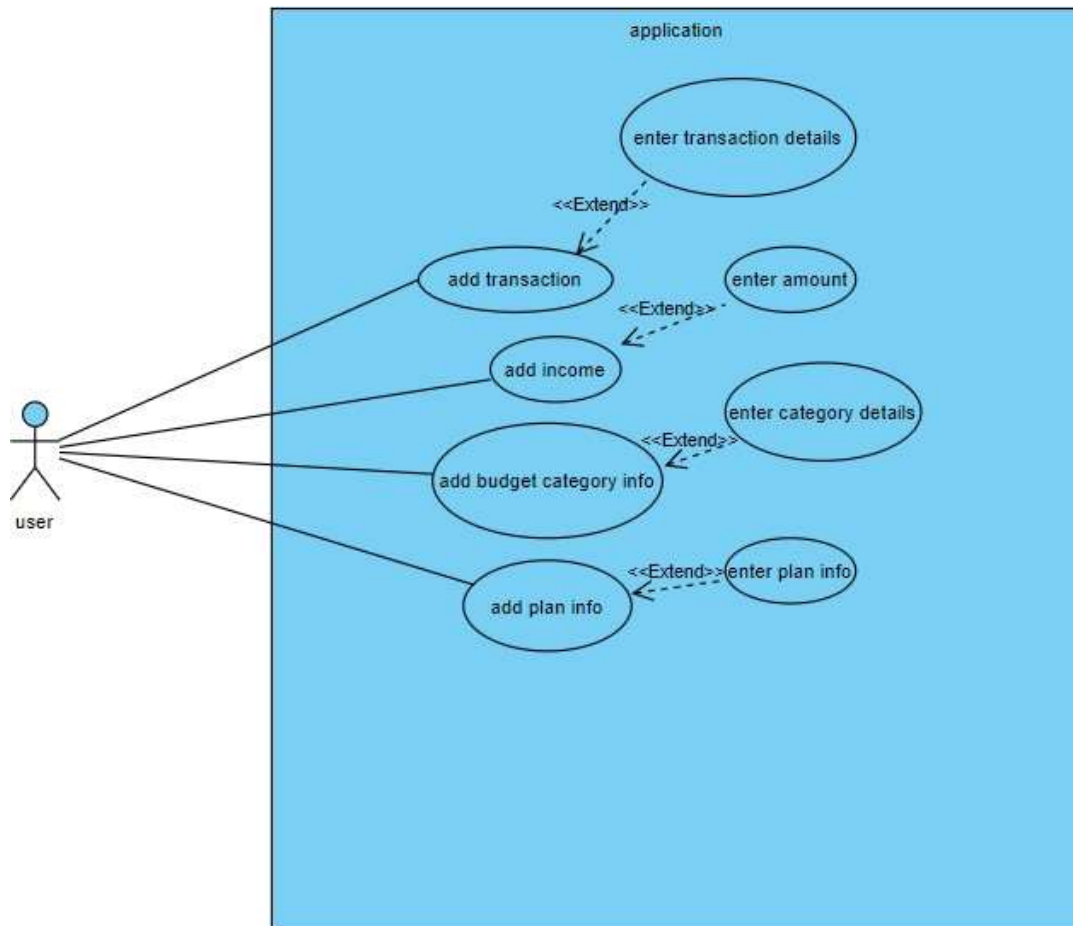
N.1 :- Database :

A database management system that is available free of cost in public domain should be used.

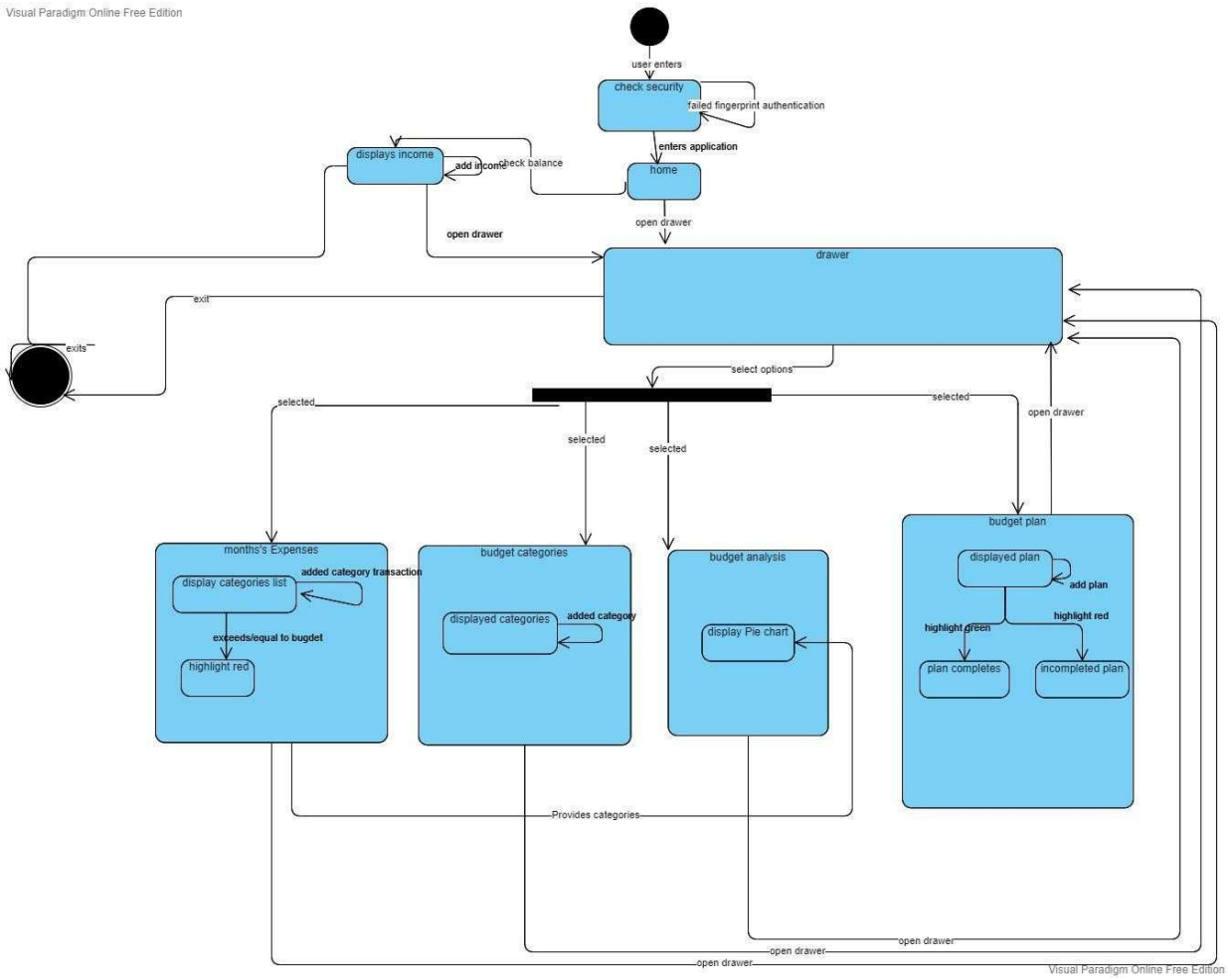
N.2 :- Platform :

android version of software need to be developed.

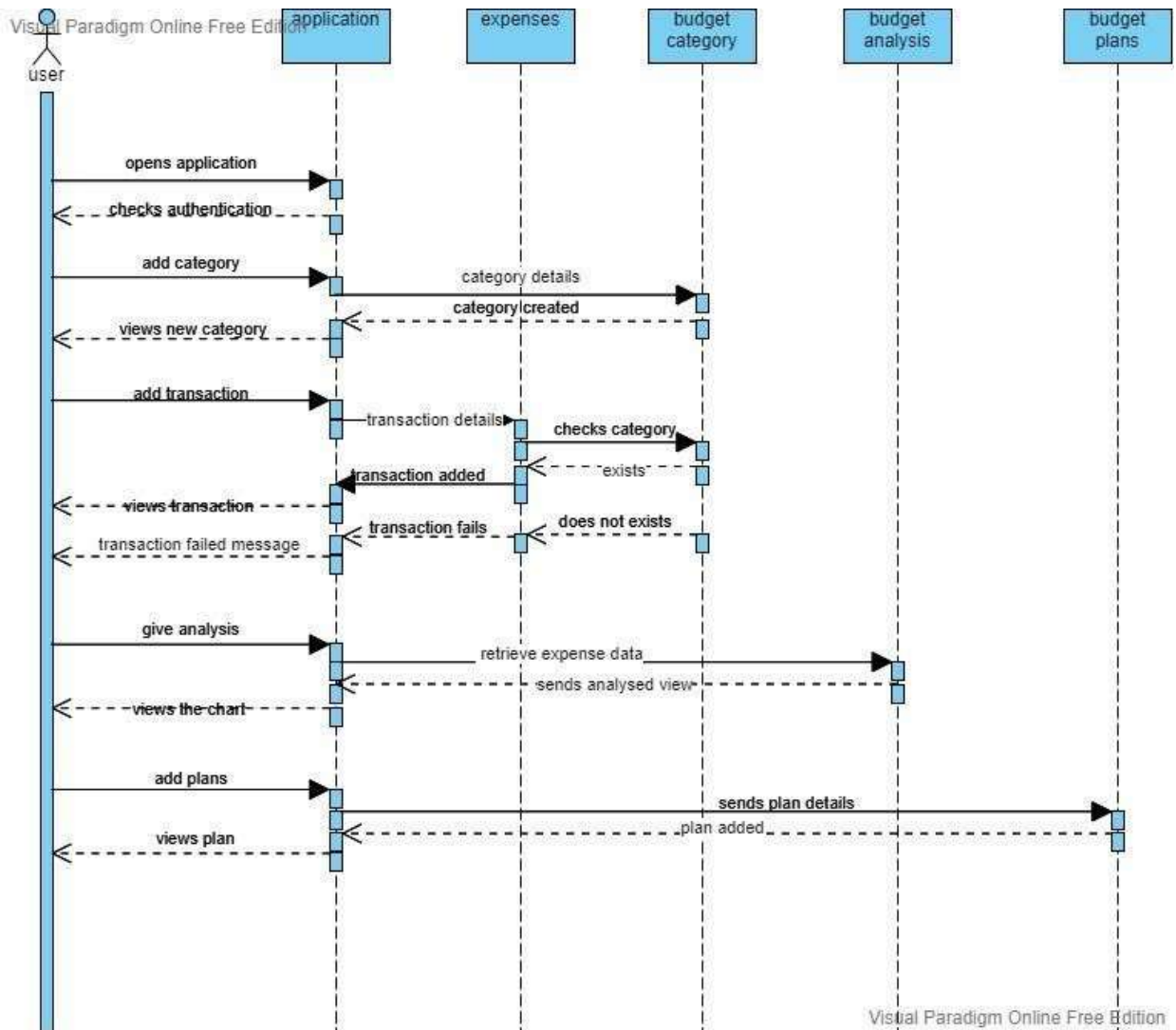
4. Designs



4.1 Use Case Diagram



4.2 State diagram



4.3 sequence diagram

5. Implementation Detail

1. Add Transaction

Details module contains details of the transaction. First user selects the category of transaction and then enters the amount of the transaction and then enters the details and selects the date of transaction. Detail module is used for saving the transaction details.

```
class Details {  
    late String id;  
    late String category;  
    late int amount;  
    late String note;  
    late String type;  
    late DateTime selectedDate;  
    late String email;  
    late String txnid;  
    late String account;  
  
    Details(  
        {required this.category,  
        required this.amount,  
        required this.note,  
        required this.type,  
        required this.selectedDate,  
        required this.email,  
        required this.txnid,  
        required this.id,  
        required this.account});  
}
```

```

String txid(DateTime td)
{
    String id="S";
    id+=td.day.toString()+td.month.toString()+td.year.toString();
    id+=td.second.toString();
    return id;
}

int amount;
String email,category;
_AddTransactionState(this.amount,this.email,this.category);
String note = "";
String type = "Income";
DateTime selectedDate = DateTime.now();
String txnid ="" ;

TextEditingController amo = new TextEditingController();
TextEditingController nt = new TextEditingController();

```

```

body: ListView(
  padding: EdgeInsets.all(
    12.0,
  ), // EdgeInsets.all
  children: [
    Row(
      children: [
        Container(
          decoration: BoxDecoration(
            color: Colors.grey,
            borderRadius: BorderRadius.circular(16.0),
          ), // BoxDecoration
          padding: EdgeInsets.all(
            12.0,
          ), // EdgeInsets.all
          child: Icon(
            Icons.currency_rupee,
            size: 20.0,
            color: Colors.white,
          ), // Icon
        ), // Container
        SizedBox(
          width: 12.0,
        ), // SizedBox
        Text(
          amount.toString(),
          style: TextStyle(fontSize: 25),
        ), // Text
      ],
    ),
  ],

```

```

        decoration: BoxDecoration(
          color: Colors.grey,
          borderRadius: BorderRadius.circular(16.0),
        ), // BoxDecoration
        padding: EdgeInsets.all(
          12.0,
        ), // EdgeInsets.all
        child: Icon(
          Icons.description,
          size: 20.0,
          color: Colors.white,
        ), // Icon
      ), // Container
    ), // SizedBox
    Expanded(
      child: TextField(
        controller: nt,
        decoration: InputDecoration(
          hintText: "Note",
          border: InputBorder.none,
        ), // InputDecoration
        style: TextStyle(
          fontSize: 23.0,
        ), // TextStyle
        onChanged: (val) {
          note = val;
        },
      ), // TextField
    ),
  ),

```

```

    ChoiceChip(
      label: Text(
        "Income",
        style: TextStyle(
          fontSize: 14.0,
          color: type == "Income" ? Colors.white : Colors.black,
        ), // TextStyle
      ), // Text
      selectedColor: Colors.indigo,
      selected: type == "Income" ? true : false,
      onSelect: (val) {
        if (val) {
          setState(() {
            type = "Income";
          });
        }
      },
    ), // ChoiceChip
    SizedBox(
      width: 12.0,
    ), // SizedBox
    ChoiceChip(
      label: Text(
        "Expense",
        style: TextStyle(
          fontSize: 14.0,
          color: type == "Expense" ? Colors.white : Colors.black,
        ), // TextStyle
      ), // Text
    ),
  ),

```


5.2 Expense Analysis

This module is used to represent the pie chart of the data so that user can easily analysis the data. Also user can see how much amount of data he has spent from his budget.

```
body: Container(  
  child: PieChart(  
    dataMap: dataMap,  
    animationDuration: Duration(milliseconds: 3000),  
    chartLegendSpacing: 20,  
    chartRadius: MediaQuery.of(context).size.width / 1.2,  
    // colorList: colorList,  
    initialAngleInDegree: 180,  
    chartType: ChartType.disc,  
    ringStrokeWidth: 32,  
    // centerText: "HYBRID",  
    legendOptions: const LegendOptions(  
      showLegendsInRow: true,  
      legendPosition: LegendPosition.bottom,  
      showLegends: true,  
      legendShape: BoxShape.rectangle,  
      legendTextStyle: TextStyle(  
        fontWeight: FontWeight.bold,  
      ), // TextStyle  
    ), // LegendOptions  
    chartValuesOptions: const ChartValuesOptions(  
      showChartValueBackground: true,  
      showChartValues: true,  
      showChartValuesInPercentage: true,  
      showChartValuesOutside: true,  
      decimalPlaces: 1,  
    ), // ChartValuesOptions  
  ) // PieChart  
, // Container  
  // Scaffold
```

```
Container(  
  child: CircularPercentIndicator(  
    radius: 200,  
    lineWidth: 25,  
    percent: perc,  
    center: Text(  
      pe.toStringAsFixed(0)+'%',  
      style: TextStyle(fontWeight: FontWeight.bold, fontSize: 30),  
    ), // Text  
    footer: Padding(  
      padding: const EdgeInsets.all(8.0),  
      // child: Text(  
      //   'Point Score This week',  
      //   style: TextStyle(fontWeight: FontWeight.bold, fontSize: 20),  
      // ),  
    ), // Padding  
    circularStrokeCap: CircularStrokeCap.round,  
    progressColor: color,  
    backgroundColor: Colors.blue,  
    animation: true,  
    animationDuration: 5000,  
  ), // CircularPercentIndicator  
) // Container
```

5.3 Set limits for each category

User can set maximum limits for each category. In future this feature is extended as if the user crosses the maximum limit then he receives a notification about spend less in that category.

```
Row(  
  mainAxisAlignment: MainAxisAlignment.spaceAround,  
  children: <Widget>[  
    Flexible(  
      child: TextField(  
        controller: foodcontroller,  
        textAlign: TextAlign.center,  
        decoration: const InputDecoration(  
          alignLabelWithHint: true,  
          border: OutlineInputBorder(),  
          labelText: "Food",  
          hintText: "0",  
          contentPadding: EdgeInsets.all(10)  
        ), // InputDecoration  
        onChanged: (val){  
          try{  
            food=int.parse(val);  
          }catch(e){}  
        },  
        keyboardType: TextInputType.number,  
        inputFormatters: <TextInputFormatter>[  
          FilteringTextInputFormatter.digitsOnly  
        ], // <TextInputFormatter>[]  
      ), // TextField  
    ), // Flexible  
    SizedBox(  
      width: 12.0,  
    ), Flexible( // SizedBox
```

5.3 Daily reminder

User get daily notification that whether he has add today's expenses or not. The user receives daily reminder with some note.

```
class NotificationApi{
  final FlutterLocalNotificationsPlugin flutterLocalNotificationsPlugin = FlutterLocalNotificationsPlugin();

  final AndroidInitializationSettings _androidInitializationSettings = AndroidInitializationSettings('logo');
  void initialiseNotifications() async{
    InitializationSettings initializationSettings = InitializationSettings(
      android: _androidInitializationSettings,
    );

    await flutterLocalNotificationsPlugin.initialize(initializationSettings);
  }

  void sendNotification(String title, String body) async{
    AndroidNotificationDetails androidNotificationDetails = const AndroidNotificationDetails(
      'channelId',
      'channelName',
      importance: Importance.max,
      priority: Priority.high,
    );

    NotificationDetails notificationDetails=NotificationDetails(
      android: androidNotificationDetails,
    );
```

```
void scheduleNotification(String title, String body) async{
  print("hello1");
  AndroidNotificationDetails androidNotificationDetails = const AndroidNotificationDetails(
    'channelId',
    'channelName',
    importance: Importance.max,
    priority: Priority.high,
  );
  NotificationDetails notificationDetails=NotificationDetails(
    android: androidNotificationDetails,
  );
  await flutterLocalNotificationsPlugin.periodicallyShow(
    0,
    title,
    body,
    RepeatInterval.daily,
    notificationDetails
```

6. Testing

6.1 Months expenses

Sr. No	Test Scenario	Expected Result	Actual Result	Status
1	Open expense activity from drawer	Open and shows information	Opened and showed information	success
2	Add Transaction	Shows on home fragment	Showed on home fragment	success
3	Transaction information	Shows info of transaction	showed	success

6.2 Budget Analysis

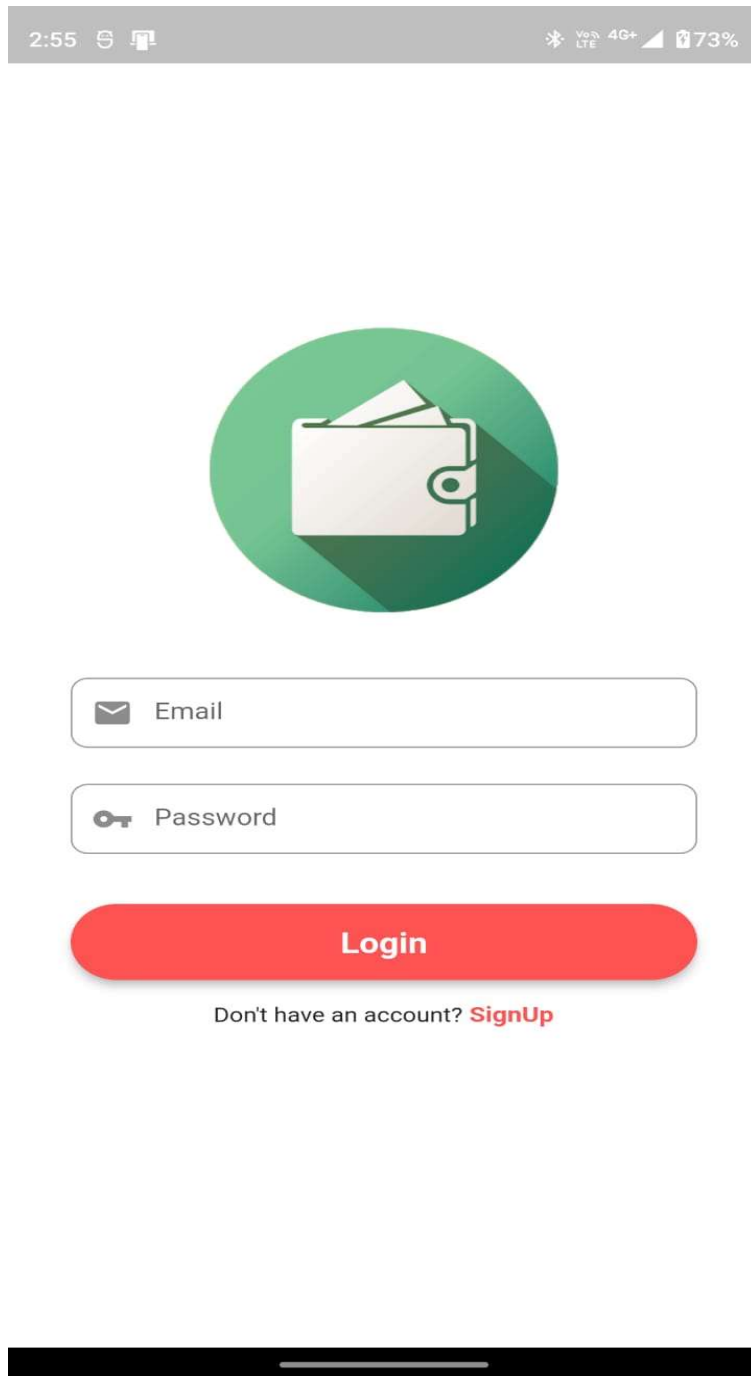
Sr. No	Test Scenario	Expected result	Actual result	Status
1	Show pie chart of expenses	Displays Pie chart	Displayed Pie chart	success

6.3 Income

Sr.No	Test scenario	Expected result	Actual result	Status
1	Add money	Shows in wallet	Showed in wallet	success

7. Screen-shots


1. Home





7.2 Sign up page


2:43 5G+ 4G+ 72%


<





 First Name

 Second Name

 Email

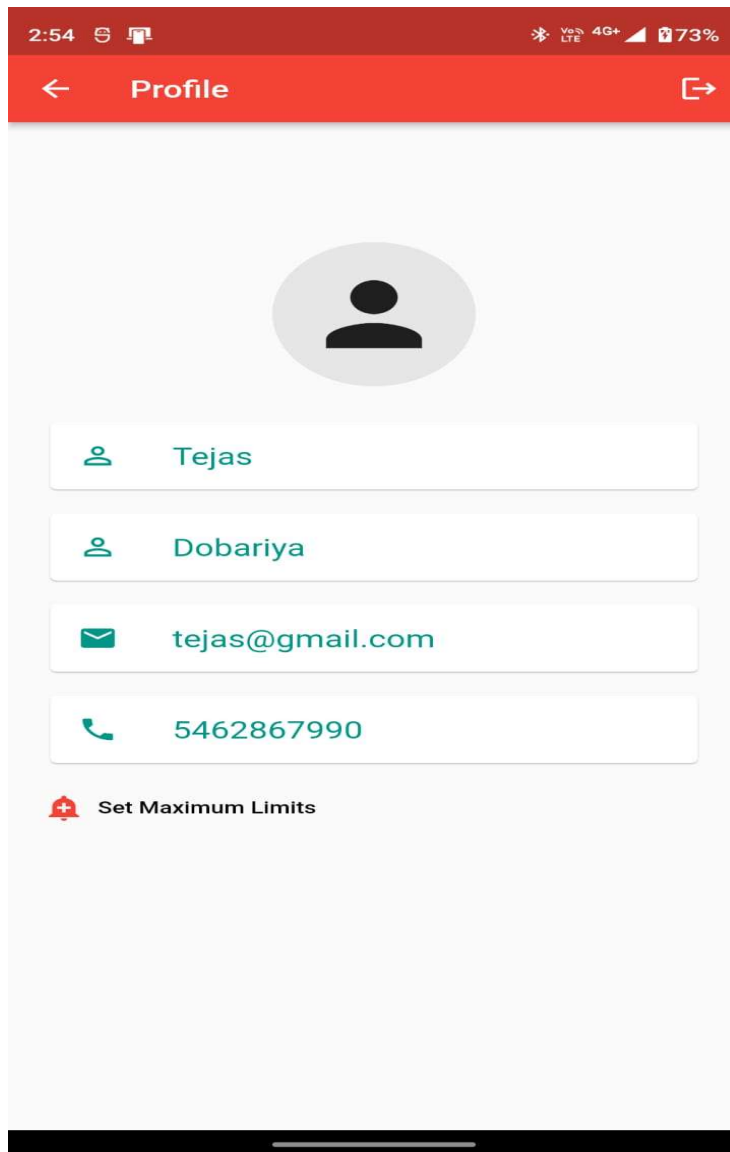
 Mobile

 Password

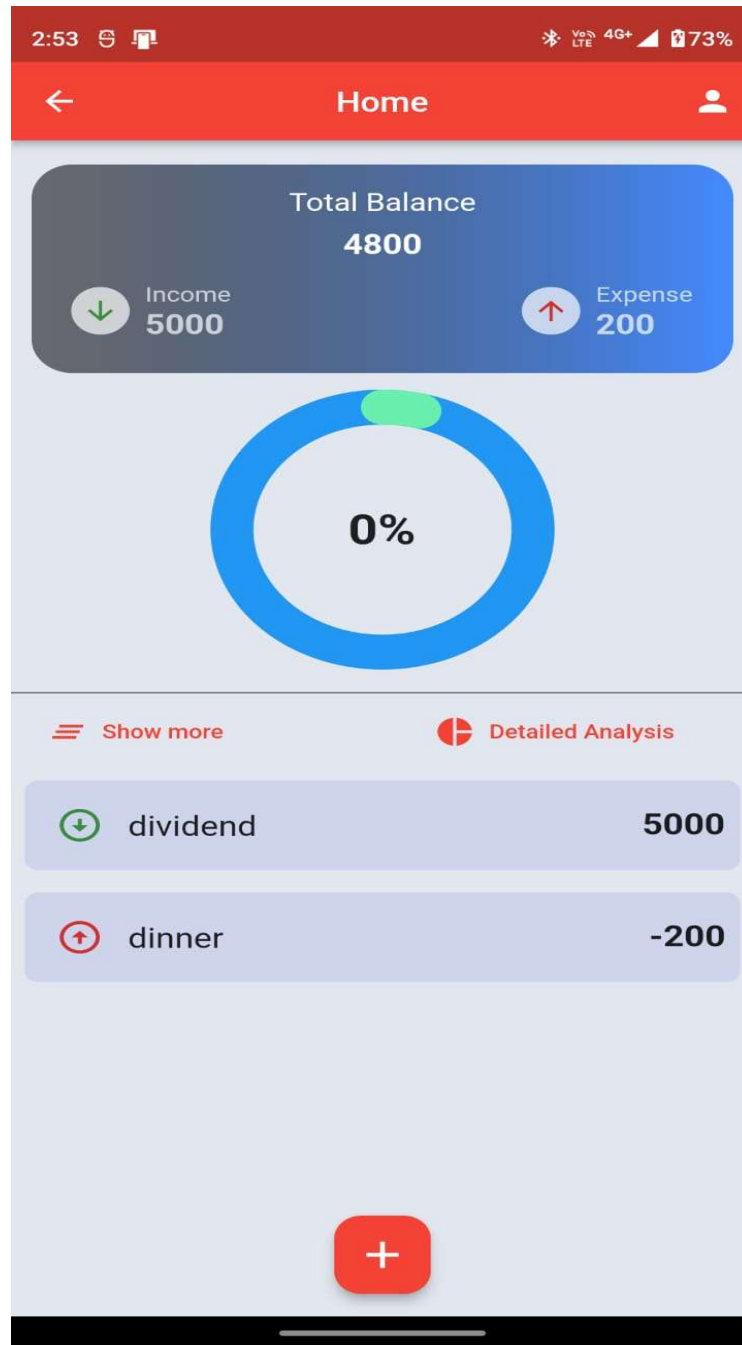
 Confirm Password

SignUp

7.3 My Profile



7.4 Home Page



7.5 Add expense amount

The screenshot shows a mobile application interface for adding an expense. At the top, there is a red header bar with a back arrow on the left, the word "Food" in the center, and a forward arrow on the right. Below this header, there is a light red bar containing the text "Category Food" on the left and "Account CASH" on the right. The main area of the screen is white and contains a single text input field with a thin red border. Inside the field, the placeholder text "Enter Amount" is visible. At the very bottom of the screen, there is a black bar with a white horizontal line, which is a standard feature of an iPhone's home indicator.

7.6 Add Expense

The screenshot shows a mobile application interface for adding a new transaction. At the top, a red header bar contains a back arrow and the text "New Transaction". Below this, the form fields are as follows:

- Amount:** A grey button with the Indian Rupee symbol (₹) is followed by the text "200".
- Category:** A grey button with a fork and knife icon is followed by the text "Food".
- Payment Method:** A grey button with a wallet icon is followed by the text "Cash".
- Description:** A grey button with a document icon is followed by the text "dinner".
- Type:** A grey button with an upward arrow icon is followed by two buttons: "Income" (light grey) and "Expense" (dark blue).
- Date:** A grey button with a calendar icon is followed by the text "31 Mar".

At the bottom of the form is a large, dark blue button with the text "Add". The status bar at the very top shows the time as 2:51, signal strength, 4G+ network, and 73% battery.

7.7 Update Limit

2:54 5G+ 4G+ 73%

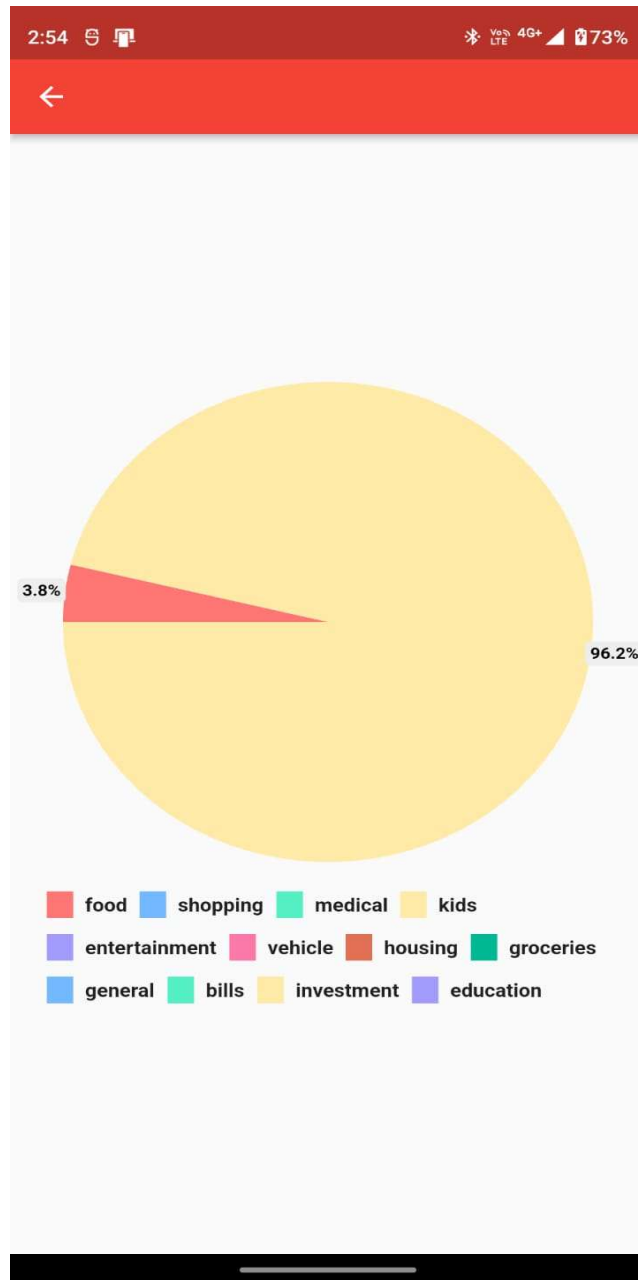
←

Food	Shopping 3500	Medical
Kids	Entertainment	Vehicle
Housing	Groceries	General
Bills	Investment	Education

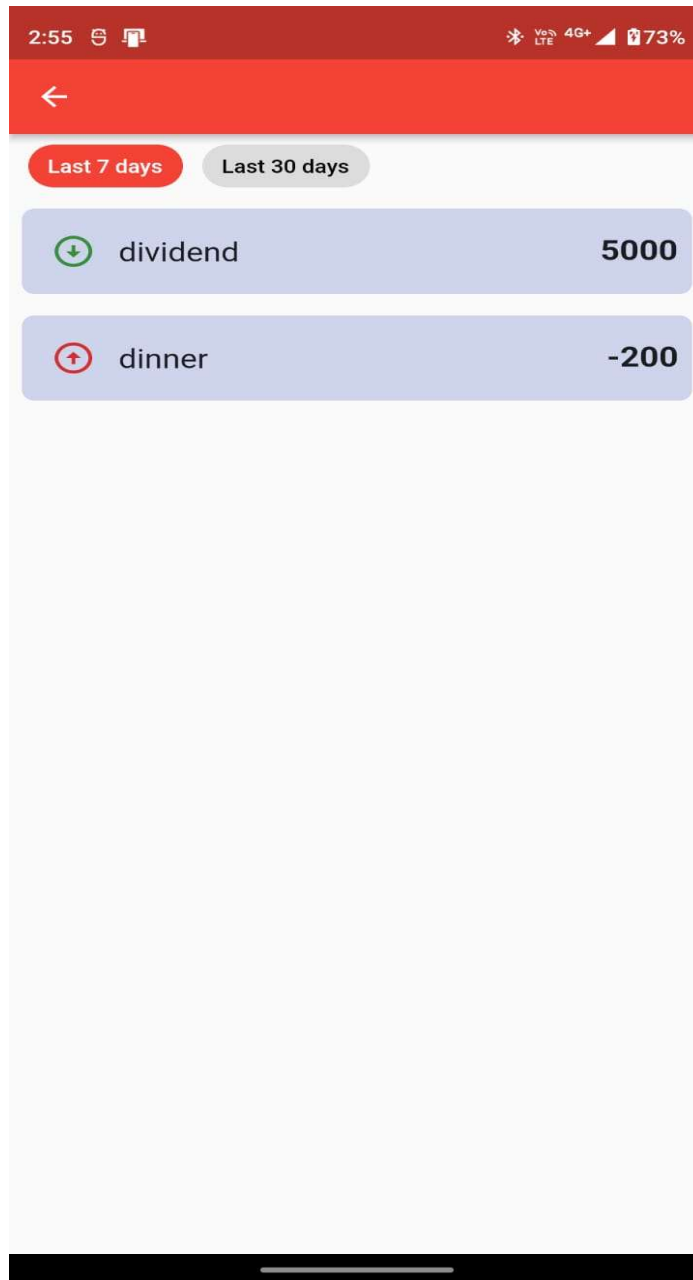
Update

1 2 3 4 5 6 7 8 9 0
@ # ₹ & _ - () = %
{&= " * ' : / ! ? + ✕
abc , _ . ←

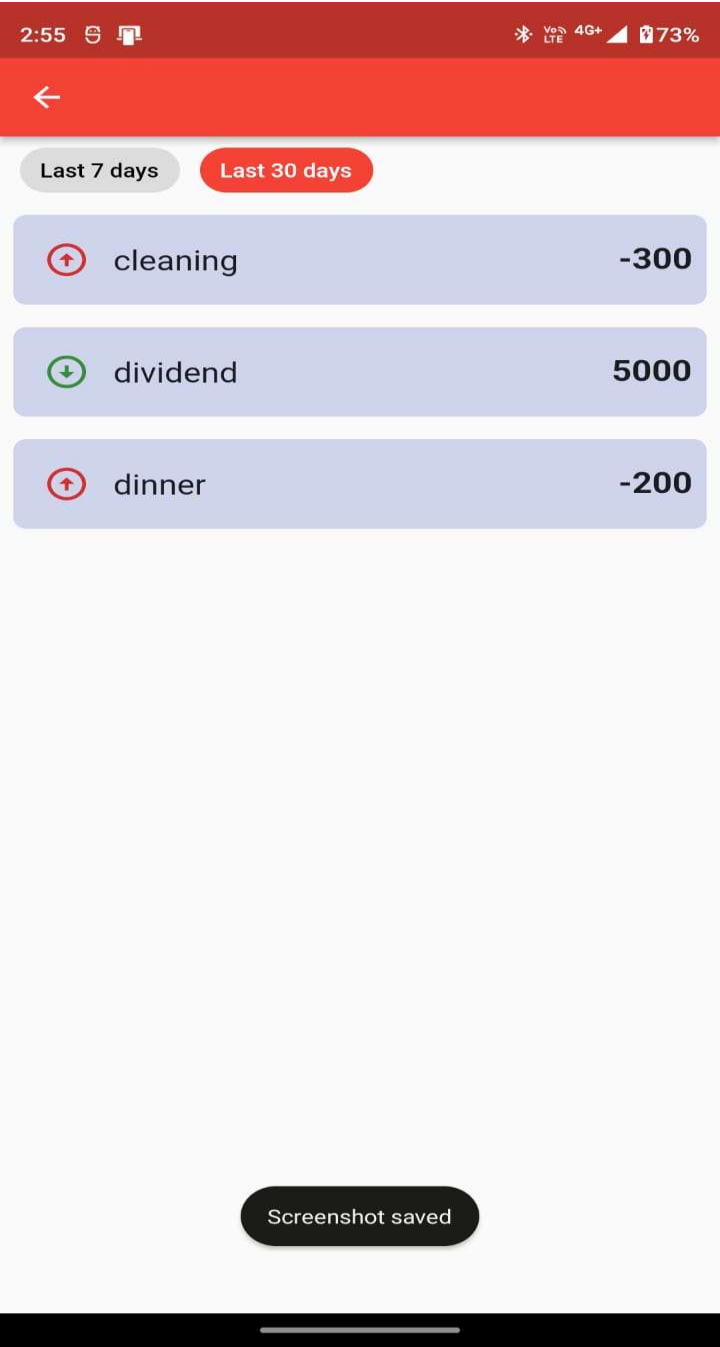
7.8 View PieChart



7.9 View Expenses



7.10 View Expenses



8. Conclusion

As per the Software Requirement Specification, we were able to make all major features in our application.

The Add transaction was done with precision as we also want to set the categories. We set the add transaction in such a way that it gets associated with its category. We also took the date of the transaction into account. Previous dates transaction can also be added in later on days

The Add Income is functionality in which user add money. This will get added to the wallet of the user.

9. Limitation and Future Extension

The Limitation to the projects are as follows:-

- The user has to first add the income and then he can able to add expenses.
- User can't change his initial income.

Future Extension of the application can be as follows:-

- User gets notification on reaching the maximum limit.
- User can download monthly report of his expenses.

10. Bibliography

Smyth, Neil. Android Studio 3.0 Development Essentials - Android 8 Edition. Payload Media, Inc., 2017.

“Documentation.” Android Developers,
<https://developer.android.com/docs>. Accessed 12 February 2022.

Stack Overflow - Where Developers Learn, Share, & Build Careers,
<http://www.stackoverflow.com>. Accessed 28 March 2022.

“Android - Studio.” Tutorialspoint,
https://www.tutorialspoint.com/android/android_studio.htm. Accessed 28 March 2022.