

VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“Jnana Sangama”, Belgaum 590014, KARNATAKA, INDIA



A Mini-Project Report On

“EXPENSE MANAGER”

A Mini-project report submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belgaum.

Submitted by:

SABERI YOUNUS	[1AM20CS165]
NEELES S	[1AM20CS127]
NISCHITH PM	[1AM20CS129]
NANDA KUMAR V	[1AM20CS124]

Under the Guidance of:

Mr. Praveen Kumar B
Assistant Professor
Department of CSE



Department of Computer Science and Engineering

AMC Engineering College,

18th K.M, Bannerghatta Main Road, Bangalore-560 083

2022-2023

AMC Engineering College,

18th K.M, Bannerghatta Main Road, Bangalore-560 083

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CERTIFICATE

This is to certify that the mini-project work entitled “**EXPENSE MANAGER**” has been successfully carried out by **SABERI YOUNUS (1AM20CS165)**, **NEELESH S (1AM20CS127)**, **NANDA KUMAR (1AM20CS124)** and **NISCHITH PM (1AM20CS129)**, bonafide students of AMC Engineering College in partial fulfilment of the requirements for the award of degree in Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belgaum during academic year 2022-2023. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of project work for the said degree.

Project Guide

Mr. Praveen Kumar B
Assistant Professor
Department of CSE

HOD

Dr. V Mareeswari
Professor & Head
Department of CSE

Principal

Dr. Nagaraja R
Principal,
AMCEC

External Examiners:

1

2

Signature with Date

ACKNOWLEDGEMENT

First and foremost, I would like to thank GOD, the Almighty for being so merciful on me I am grateful to my guide **Mr. Praveen Kumar B**, Asst. Prof., Department of CSE, AMC Engineering College, Bengaluru for his unfailing and constant motivational & timely help, encouragement and suggestion, given to me in the course of my project work.

I would like to extend my special thanks to **Dr. V. Mareeswari** Professor and HOD, Department of CSE, for her support and encouragement and suggestions given to me in the course of my project work.

I express my sincere thanks and gratitude to our Principal **Dr. R Nagaraja** for providing me all the necessary facilities for successful completion of my project.

I have a great pleasure in expressing my deep sense of gratitude to founder Chairman **Dr. K.R. Paramahamsa** for having provided me with a great infrastructure and well-furnished labs for successful completion of my seminar.

Last but not the least, I wish to thank all the teaching & non-teaching staffs of department of Computer Science and Engineering, for their support, patience and endurance shown during the preparation of this project report.

SABERI YOUNUS: (1AM20CS165)

NEELESH S: (1AM20CS127)

NANDA KUMAR V: (1AM20CS124)

NISCHITH P M: (1AM20CS129)

ABSTRACT

The expense manager is a software application designed to help individuals and businesses track and manage their expenses effectively. It incorporates various features such as expense categorization, budget tracking, and data visualization to provide users with a comprehensive solution for financial management. The expense manager offers a user-friendly interface that allows users to easily input and categorize their expenses. By assigning expenses to specific categories such as food, transportation, entertainment, and more, users can organize their financial data efficiently. This categorization feature enables users to gain insights into their spending patterns and identify areas where adjustments can be made to improve financial planning and savings. Budget tracking is another essential feature of the expense manager. Users can set budgets for different expense categories and receive notifications or warnings when they approach or exceed their predefined limits. This functionality promotes responsible spending habits and helps users stay within their financial means, contributing to better financial management and stability. The expense manager also includes data visualization tools that present financial information in a clear and concise manner. Through charts, graphs, and reports, users can analyze their spending patterns, identify trends, and make informed decisions. This visual representation of data helps users understand their financial health and make adjustments to their spending habits or financial strategies accordingly. Overall, the development of an expense manager as a mini project provides a practical solution for individuals and businesses seeking to manage their expenses effectively. By incorporating features such as expense categorization, budget tracking, and data visualization, the expense manager empowers users to take control of their finances, make informed decisions, and work towards achieving their financial goals.

CONTENTS

CHAPTER	TITLE	Page No
1	INTRODUCTION	1
	1.1 Overview of the Project	1
	1.2 Purpose of the Project	2
	1.3 Problem Definition	2
	1.4 Scope	3
	1.5 Objective of the System	3
	1.6 Features of the System	3
2	SYSTEM SPECIFICATIONS	4
	2.1 Hardware Requirements	4
	2.2 Software Requirements	4
	2.3 About the technology, Tools, Language used in the project	4
	2.4 Architecture Diagram	6
	2.5 Description	6
3	DESIGN	7
	3.1 Design principle of Android Application	7
	3.2 Function and Structure design of android system	7
	3.3 Requirement analysis of system	8
	3.4 Use Case Diagram	8
4	IMPLEMENTATION AND CODING	10
	4.1 XML Code Snippet	10
	4.2 Java Code Snippet	16
5	SNAPSHOTS	24
6	CONCLUSIONS	29
	REFERENCES	30

LIST OF FIGURES

FIGURE No	TITLE	Page No
2.1	Architecture Diagram	6
3.1	Use-Case Diagram	9
5.1	Home Page	24
5.2	Transaction Details Page	24
5.3	Income Transaction Page	25
5.4	Expense Transaction Page	25
5.5	Recent Transaction Page	26
5.6	Budget Page	26
5.7	Dashboard Page	27
5.8	Budget Limit Page	27
5.9	Pie Chart Analysis	28
5.10	Dashboard Viewing Page	28

CHAPTER 1

INTRODUCTION

The Expense Manager project is an Android application developed in Java using Android Studio. It aims to provide users with a comprehensive solution for managing their personal finances. In today's fast-paced world, managing personal finances can be a challenging task. Many individuals struggle to keep track of their expenses, often leading to overspending and financial instability. The Expense Manager project addresses this need by providing a convenient and efficient tool

for expense tracking and management. Managing personal finances is crucial for financial stability and achieving financial goals. However, traditional methods of expense tracking, such as pen and paper or spreadsheets, can be time-consuming and prone to errors. Additionally, these methods often lack real-time insights into spending patterns, making it difficult for individuals to control their expenses effectively. The Expense Manager project aims to address these challenges by providing a user-friendly and intuitive mobile application that simplifies the process of tracking expenses. The purpose of the Expense Manager project is to help individuals effectively manage their expenses by providing them with a user-friendly and intuitive mobile application. The project aims to simplify the process of tracking expenses, enabling users to have a clear understanding of their financial situation. By categorizing expenses and setting budgets, users can gain insights into their spending habits and make necessary adjustments to achieve their financial goals. The Expense Manager project offers a range of features to facilitate expense tracking and management.

1.1 Overview of the Project

The Expense Manager project is an Android application developed in Java using Android Studio. It aims to provide users with a comprehensive solution for managing their personal finances. In today's fast-paced world, managing personal finances can be a challenging task. Many individuals struggle to keep track of their expenses, often leading to overspending and financial instability. The Expense Manager project addresses this need by providing a convenient and efficient tool for expense tracking and management. The Expense Manager application offers a user-friendly interface that allows users to record their expenses, categorize them, set budgets, and generate reports. With its intuitive design and powerful features, the application simplifies the process of tracking expenses, enabling users to have a clear understanding of their financial situation. The primary objective of the Expense Manager project is to help individuals effectively manage their expenses. By providing a user-friendly and intuitive mobile application, the project aims to simplify the process of tracking expenses and enable users to make informed financial decisions. The application allows users to record their expenses on the go, ensuring that no expense is overlooked. Users can categorize their expenses

based on different categories such as food, transportation, entertainment, etc., which provides insights into their spending patterns.

1.2 Purpose of the Project

The purpose of the Expense Manager project is to provide individuals with a simple and efficient tool for managing their personal finances. In today's fast-paced world, managing personal finances can be a challenging task. Many individuals struggle to keep track of their expenses, often leading to overspending and financial instability. The Expense Manager project aims to address this issue by providing a convenient and efficient solution for expense tracking and management.

The primary purpose of the Expense Manager project is to help individuals effectively manage their expenses. By offering a user-friendly and intuitive mobile application, the project aims to simplify the process of tracking expenses and enable users to make informed financial decisions. The application allows users to record their expenses on the go, ensuring that no expense is overlooked. This feature is particularly useful for individuals who have busy lifestyles and need a convenient way to track their expenses. Another purpose of the Expense Manager project is to provide users with a clear understanding of their spending habits. By categorizing expenses based on different categories such as food, transportation, entertainment, etc., the application provides insights into how users allocate their funds. This categorization allows users to identify areas where they may be overspending and make necessary adjustments to their budgets.

1.3 Problem Definition

In today's fast-paced world, managing personal finances can be a challenging task. Many individuals struggle to keep track of their expenses, often leading to overspending and financial instability. Traditional methods of expense tracking, such as pen and paper or spreadsheets, can be time-consuming and prone to errors. Additionally, these methods often lack real-time insights into spending patterns, making it difficult for individuals to control their expenses effectively. The Expense Manager project aims to address these challenges by providing a convenient and efficient solution for expense tracking and management.

One of the main problems individuals face in managing their personal finances is the difficulty in tracking expenses accurately. With numerous transactions happening daily, it becomes challenging to keep a record of every expense. This leads to incomplete or inaccurate expense tracking, making it difficult for individuals to have a clear understanding of their spending habits. The Expense Manager project aims to solve this problem by providing a user-friendly mobile application that allows users to easily record their expenses on the go. By offering a convenient and efficient expense tracking solution, the project enables users to have a comprehensive overview of their spending patterns.

Another problem individuals encounter is the lack of real-time insights into their spending habits. Traditional methods of expense tracking often require manual calculations and analysis, which can be time-consuming

and prone to errors. This makes it difficult for individuals to identify areas where they may be overspending or to track their progress towards financial goals. The Expense Manager project addresses this problem by providing users with real-time updates on their expenses. The application categorizes expenses and provides visual representations of spending patterns through charts and graphs, allowing users to analyze their expenses easily and make necessary adjustments to their budgets.

1.4 Scope

PERFORMANCE: The application is designed to perform efficiently even with large amounts of data.

EFFICIENCY: The application provides quick and accurate expense tracking and management.

SECURITY: The application ensures the security and privacy of user data.

1.5 Objective of the System

- To provide a user-friendly interface for recording and managing expenses.
- To categorize expenses for better understanding of spending patterns.
- To allow users to set and track budgets.
- To generate detailed expense reports for financial analysis.
- To ensure the security and privacy of user data.

1.6 Features of the System

- User-friendly interface for expense recording and management.
- Expense categorization for better understanding of spending habits.
- Budget setting and tracking feature.
- Detailed expense report generation.
- Robust security measures to protect user data.

CHAPTER 2

SYSTEM SPECIFICATIONS

2.1 Hardware Requirements

System Requirements:

- Processor: Intel
- RAM: 8GB
- Storage: 512GB

2.2 Software Requirements

- Operating System: Windows 7/8/10. The Android Emulator only supports 64-bit Windows.
- Development Environment: Android Studio 4.2
- API: Java Development Kit (JDK) 7
- Core Language: Java, XML for Front-end

2.3 About the technology, Tools, Language used in the project

Platform: Android Studio

Android Studio is the official integrated development environment (IDE) for Google's Android OS, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. It is available for download on Windows, macOS and Linux based operating systems or as a subscription-based service in 2020. It is a replacement for the Eclipse Android Development Tools (E-ADT) as the primary IDE for native Android application development. Android Studio was announced on May 16, 2013 at the Google I/O conference. It was in early access preview stage starting from version 0.1 in May 2013, then entered beta stage starting from version 0.8 which was released in June 2014. The first stable build was released in December 2014, starting from version 1.0. On May 7, 2019, Kotlin replaced Java as Google's preferred language for Android app development. Java is still supported, as is C++.

Features of Android Studio

A specific feature of the Android Studio is an absence of the possibility to switch autosave feature off. The following features are provided in the current stable version:

- Gradle-based build support
- Android-specific refactoring and quick fixes
- Lint tools to catch performance, usability, version compatibility and other problems

- ProGuard integration and app-signing capabilities
- Template-based wizards to create common Android designs and components
- A rich layout editor that allows users to drag-and-drop UI components, option to preview layouts on multiple screen configurations
- Support for building Android Wear apps
- Built-in support for Google Cloud Platform, enabling integration with Firebase Cloud Messaging (Earlier 'Google Cloud Messaging') and Google App Engine.
- Android Virtual Device (Emulator) to run and debug apps in

Android Studio supports all the same programming languages of IntelliJ. e.g. Java, C++, and more with extensions, such as Go; and Android Studio 3.0 or later supports Kotlin and "all Java 7 language features and a subset of Java 8 language features that vary by platform version." While IntelliJ states that Android Studio supports all released Java versions, and Java 12, it's not clear to what level Android Studio supports Java versions up to Java 12. Once an app has been compiled with Android Studio, it can be published on the Google Play Store. The application has to be in line with the Google Play Store. The application has to be in line with the Google Play Store developer content policy.

The Android Emulator has additional requirements beyond the basic system requirements for Android Studio, which are described below:

- SDK Tools 26.1.1 or higher.
- 64-bit processor
- Windows: CPU with UG (unrestricted guest) support;
- Intel Hardware Accelerated Execution Manager (HAXM) 6.2.1 or later (HAXM 7.2.0 or later recommended).

Programming Language: Java

Java is a high-level, class-based, object-oriented programming language that is designed to have as few implementation dependencies as possible. It is a general-purpose programming language intended to let application developers write once, run anywhere (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte-code that can run on any Java virtual machine (JVM) regardless of the underlying computer architecture. The syntax of Java is similar to C and C++, but has fewer low-level facilities than either of them. The Java runtime provides dynamic capabilities (such as reflection and runtime code modification) that are typically not available in traditional compiled languages.

2.4 Architecture Diagram

Figure 2.1 shows the architecture diagram for an Expense Manager that has been implemented in the application.

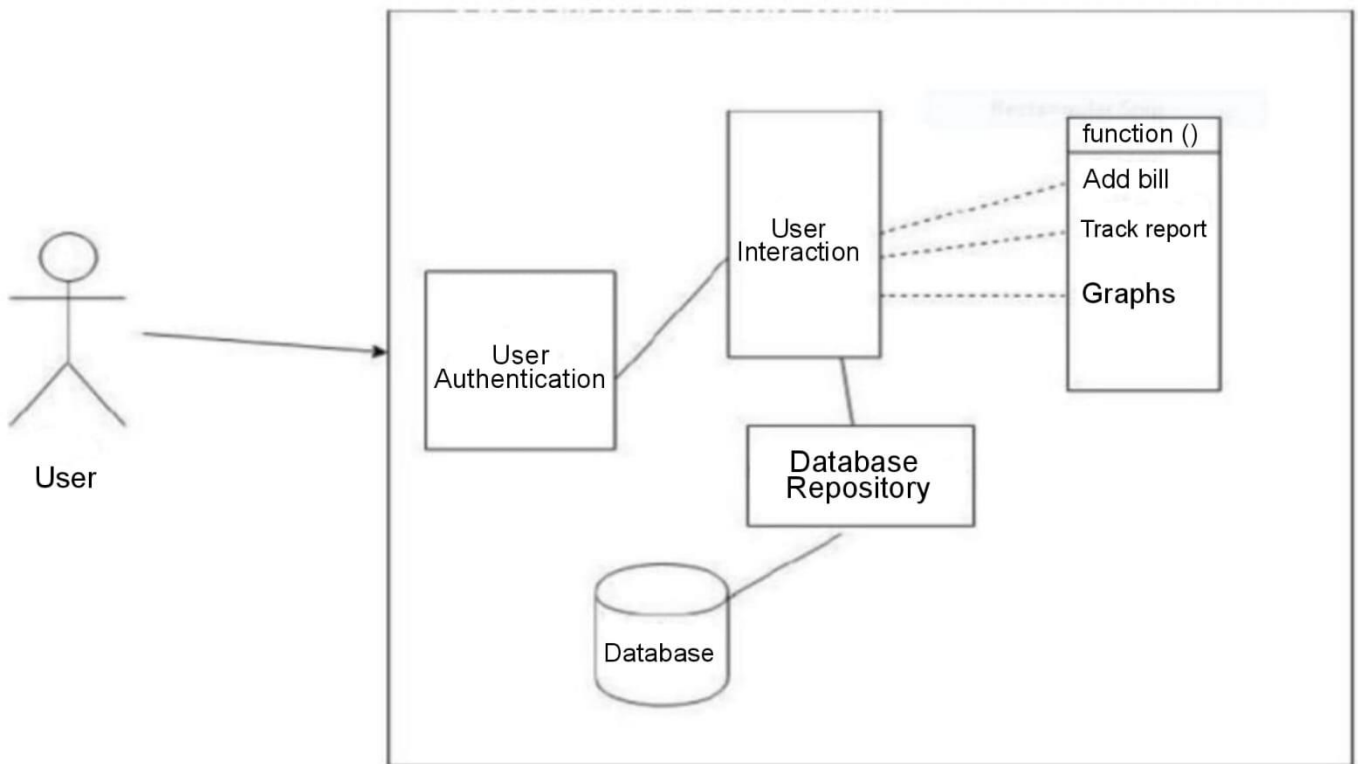


Fig 2.1: Architecture Diagram

2.5 Description

The architecture diagram has various components with the first component to interact with the user being the user authentication. The user is given access to use the application only after it has been verified that the user is a registered user. User authentication component then interacts with the user interface where the user is directed after verifying its authenticity. The user interface or simply an "interface", is the means in which the user controls the application. It provides a "user-friendly" experience, allowing the user to interact with the application in a natural and intuitive way. Depending upon the user's need and selection, it either calls the class function for various functionalities or it accesses the database by interacting to it via data repository component. Data repository refers to the destination designated for data storage.

CHAPTER 3

DESIGN

3.1 Design principle of Android Application

Twice the result with half the effort will get if an overall study of the principles done before the design and follow them in the operation. The principle of software design mainly includes the following points:

- **Reliability**

The reliability of the software design must be determined. The reliability of the software system refers to the ability to avoid fault occurred in the process of system running, as well as the ability to remedy troubles once the fault occurs.

- **Reusability**

Look for commonness of similar codes, and come out new method abstractly and reasonably. Pay attention to the generic design.

- **Understandability**

The understandability of software not only require clear and read able document, but the simplified structure of software itself, which requires the designer possess keen insight and creativity, and know well about the design objects.

- **Simple program**

To keep the program simple and clear, good programmers can use simple program to solve complex problems.

- **Testability**

Testability means that the created system has a proper data collection to conduct a comprehensive test of the entire system.

- **The Open-Closed Principal**

Module is extensible but cannot be modified. That is to say, extension is open to the existing code in order to adapt to the new requirements.

3.2 Function and Structure design of android system

This system adopts the modularized program design, and system function is correspondingly divided into function modules, the main modules include:

- **UI function module design of mobile terminal:**

The home screen, settings screen page is realized.

- **Backstage function module design of mobile terminal:**

The specific function, Database function and other function.

3.3 Requirement analysis of system

The feasibility analysis: This section verified that it is feasible to add music player on the Android system from the aspects of economic, technical and social feasibility.

Economic feasibility: To design Android mobile phone music player as long as a computer has the Android development and the application development of Android is free. In addition, mobile phone radio music player is basic needs for public. And a lot of research is eliminated, thus saved the spending. Therefore, the whole process of development doesn't need to spend any money that is economic feasibility.

Technical feasibility: To design a radio music player which meets the basic requirements, a deep understand of JAVA language, the Android system architecture, application of framework and other technical knowledge are needed.

Social Feasibility: With the rapid development of the mobile phone market, all kinds of news resources are widely circulated on the Internet. These resources seem ordinary, but have gradually become an indispensable part of people life, which derived the development of all kinds of mobile phone player. But a lot of news apps devoted to fancy appearance, strong function causing a lot of wasted resources to the user's mobile phone and bringing a lot of inconvenience to the user as multi tasking operation is needed. Some functions are useless to ordinary people.

Saturation Overview:

This section describes requirements of the system based on basic control functions of types, and system setup function of the app according to research results of the project demand. According to the research results of project demand, the basic requirements of project system and its function structure are presented.

3.4 Use Case Diagram

A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. It models the tasks, services, and functions required by a system/subsystem of an application. It depicts the high-level functionality of a system and also tells how the user handles a system.

The main purpose of a use case diagram is to portray the dynamic aspect of a system and it accumulates the system's requirement, which includes both internal as well as external influences. It invokes persons and use cases, and several things that invoke the actors and elements accountable for the implementation of use case diagrams. It represents how an entity from the external environment can interact with a part of the system.

Following are the purposes of a use case diagram given below:

- It gathers the system's needs.
- It depicts the external view of the system.
- It recognizes the internal as well as external factors that influence the system.
- It represents the interaction between the actors.

Figure 3.1 shows the use case diagram for an Expense Manager that has been implemented in the application.

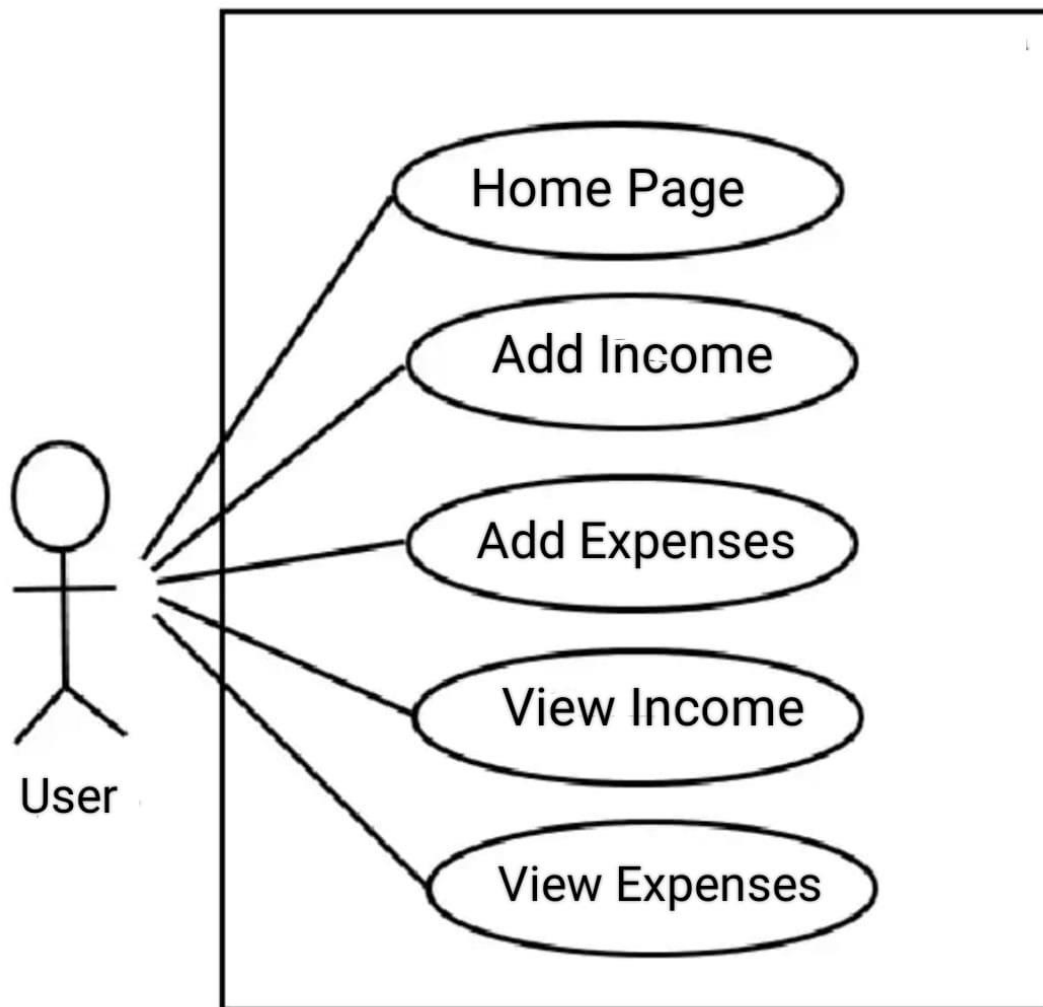


Fig.3.1 Use-Case Diagram

CHAPTER 4

IMPLEMENTATION AND CODING

4.1 XML Code Snippet

activity_main.xml

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

<androidx.drawerlayout.widget.DrawerLayout 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:id="@+id/drawer_layout"
android:layout_width="match_parent"
android:layout_height="match_parent"
tools:context=".MainActivity">

    <ScrollView
        android:layout_width="fill_parent"
        android:layout_height="wrap_content"
        android:id="@+id/scrollView"
        android:background="#EEEEEE">

        <LinearLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:orientation="vertical"
            android:background="#EEEEEE">

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
                android:orientation="vertical">

                <include layout="@layout/main_toolbar" />

            <LinearLayout
                android:layout_width="match_parent"
                android:layout_height="match_parent"
```



```
android:orientation="horizontal"
android:padding="8dp">
<Button
android:id="@+id/Expense_Amount"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="20dp"
android:layout_marginTop="25dp"
android:layout_marginRight="20dp"
android:layout_weight="1"
android:background="@drawable/red_rounded_button"
android:drawableLeft="@drawable/arrow_upward"
android:drawablePadding="10dp"
android:text=""
android:paddingRight="20dp"
android:textAlignment="textStart"
android:textSize="20dp"
android:textStyle="normal"
app:backgroundTint="#DF4358"></Button>
```

```
<Button
android:id="@+id/Income_amount"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginLeft="20dp"
android:paddingLeft="20dp"
android:layout_marginTop="25dp"
android:layout_weight="1"
android:background="@drawable/green_rounded_button"
android:drawableLeft="@drawable/arrow_downward"
android:text=""
android:textAlignment="textStart"
android:textSize="20dp"
android:textStyle="normal"
android:layout_marginRight="20dp"
app:backgroundTint="#74D93E"></Button>
```

</LinearLayout>

</LinearLayout>

Transaction XML PAGE:

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

<androidx.drawerlayout.widget.DrawerLayout 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:id="@+id/drawer_layout"

android:layout_height="match_parent"

tools:context=".AllTransactions">

<LinearLayout

android:layout_width="match_parent"

android:layout_height="match_parent"

android:orientation="vertical"

android:background="#DEDEDE">

<include layout="@layout/main_toolbar"/>

<androidx.recyclerview.widget.RecyclerView

android:layout_width="match_parent"

android:layout_height="wrap_content"

tools:itemCount="5"

android:padding="8dp"

android:id="@+id/transaction"

tools:listitem="@layout/transactions_item"

app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"

/>

</LinearLayout>

<LinearLayout

android:layout_width="match_parent"

android:layout_height="match_parent"

android:gravity="bottom|right">

<com.google.android.material.floatingactionbutton.FloatingActionButton

```

android:id="@+id/addtrans"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:clickable="true"
android:layout_marginBottom="30dp"
android:layout_marginRight="25dp"
android:src="@drawable/ic_add"
/>
</LinearLayout>
<LinearLayout
android:layout_width="match_parent"
android:layout_height="match_parent"
android:orientation="vertical"
app:layout_constraintBottom_toBottomOf="parent">

<ImageView
android:id="@+id/imgtrans"
android:layout_width="100dp"
android:layout_height="140dp"
android:layout_gravity="center"
android:layout_marginTop="120dp"
android:visibility="gone"
app:srcCompat="@drawable/no_trans" />

<TextView
android:id="@+id/txttrans"
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:text="No Transcation "
android:textAlignment="center"
android:textSize="30sp"
android:textStyle="bold"
android:textColor="@color/black"
android:visibility="gone"
/>
</LinearLayout>

```

```

<RelativeLayout
    android:layout_width="300dp"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:background="@android:color/white">

    <include layout="@layout/main_nav_drawer"/>
</RelativeLayout>
</androidx.drawerlayout.widget.DrawerLayout>

```

Budget XML PAGE:

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout 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:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".Budget">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical">

        <include layout="@layout/main_toolbar" />
        <androidx.recyclerview.widget.RecyclerView
            android:id="@+id/rv"
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            tools:listitem="@layout/rv_budget"
            app:layoutManager="androidx.recyclerview.widget.LinearLayoutManager"
            tools:itemCount="2" />

        <LinearLayout
            android:id="@+id/hidel_budget"

```

```
android:layout_width="match_parent"
android:layout_height="wrap_content"
android:layout_marginTop="20dp"
android:orientation="vertical">
```

```
<ImageView
android:id="@+id/budget_empty_icon"
android:visibility="gone"
android:layout_width="80dp"
android:src="@drawable/budget"
android:gravity="center"
android:layout_height="150dp"
android:layout_marginLeft="150dp"
/>
```

```
<TextView
android:id="@+id/no_budget"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:text="No Budget set"
android:visibility="gone"
android:textSize="20sp"
android:textStyle="bold"
android:gravity="center"
/>
```

```
<TextView
android:id="@+id/tap_budget"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:text="Tap + to add one"
android:visibility="gone"
android:textSize="15sp"
android:gravity="center"
/>
</LinearLayout>
```

```

</LinearLayout>

<LinearLayout
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:gravity="bottom|right">

    <com.google.android.material.floatingactionbutton.FloatingActionButton
        android:id="@+id/addbugetclick"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginRight="25dp"
        android:layout_marginBottom="30dp"
        android:clickable="true"
        android:src="@drawable/ic_add" />

</LinearLayout>

<RelativeLayout
    android:layout_width="300dp"
    android:layout_height="match_parent"
    android:layout_gravity="start"
    android:background="@android:color/white">

    <include layout="@layout/main_nav_drawer" />

</RelativeLayout>

</androidx.drawerlayout.widget.DrawerLayout>

```

4.2 Java Code Snippet

MainActivity.java

```

package com.example.expensify;

import androidx.appcompat.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;

```

```

import android.widget.LinearLayout;
import android.widget.TextView;
import java.util.ArrayList;

public class MainActivity extends AppCompatActivity implements View.OnClickListener {
    Animation fapopen,fapclose;
    TextView textView;
    Boolean isOpen=false;
    RecyclerView recyclerView,show_data_budget;
    ArrayList<Integer> amount;
    ArrayList<String> date;
    ArrayList<String> type;
    Button expense, income;
    int expans_amt = 0;
    int income_amt = 0;
    LinearLayout linearLayout,HideLayout_budget;
    ArrayList<Integer> image;
    ArrayList<String> paymentmode,transaction_id,spinner,note;
    TextView getTextView,view,addhomebudget,no_budget,tap_budget;
    CardView month_card,week_Card,year_card;
    ArrayList<String> budgetdate,budgetamount,budgetid;
    ArrayList<Integer> totalamount;
    SharedPreferences sharedPreferences;
    SharedPreferences.Editor myEdit;
    int status = 0;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        dl = findViewById(R.id.drawer_layout);
        iv = findViewById(R.id.menu1);
        fap1=findViewById(R.id.flincome);
        helper_category = new dbHelper(this);
        sharedPreferences = getSharedPreferences("MySharedPref",MODE_PRIVATE);
        myEdit = sharedPreferences.edit();
        int id = sharedPreferences.getInt("Id",0);

```

```

if(id == 0){
helper_category.AddCategoryAll();
myEdit.putInt("Id", 1);
myEdit.commit();
} else {
Toast.makeText(this, ""+id, Toast.LENGTH_SHORT).show();
}
expense = findViewById(R.id.Expense_Amount);
income = findViewById(R.id.Income_amount);
empty_content = findViewById(R.id.empty_content);
linearLayout = findViewById(R.id.HideLayout);
view = findViewById(R.id.tap_trans);
getTextView = findViewById(R.id.no_trans);
recyclerView = findViewById(R.id.show_data_home);
show_data_budget = findViewById(R.id.show_data_budget);
textView = findViewById(R.id.SeeAllData);

HideLayout_budget = findViewById(R.id.HideLayout_budget);
empty_budget = findViewById(R.id.empty_budget);
no_budget = findViewById(R.id.no_budget);
tap_budget = findViewById(R.id.tap_budget);

addhomebudget = findViewById(R.id.addhomebudget);
addhomebudget.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
Intent intent = new Intent();
intent.setClass(MainActivity.this,Budget.class);
startActivity(intent);
}
});

month_card = findViewById(R.id.month_card);
week_Card = findViewById(R.id.week_card);
year_card = findViewById(R.id.year_card);
textView.setOnClickListener(new View.OnClickListener() {

```


@Override

```
public void onClick(View v) {  
    Intent intent = new Intent();  
    intent.setClass(MainActivity.this, AllTransactions.class);  
    startActivity(intent);  
}  
});
```

Transaction Java Code:

```
package com.example.expensify;  
import androidx.annotation.NonNull;  
import androidx.appcompat.app.AppCompatActivity;  
import androidx.cardview.widget.CardView;  
import androidx.drawerlayout.widget.DrawerLayout;  
import androidx.recyclerview.widget.LinearLayoutManager;  
import android.view.View;  
import android.widget.ImageView;  
import android.widget.TextView;  
import android.widget.Toast;  
  
public class AllTransactions extends AppCompatActivity {  
    DrawerLayout dl;  
    RecyclerView recyclerView;  
    ArrayList<String> dates, amount, typeoftransaction, spinner, note, paymentmode;  
    String dates1, amount1, typeoftransaction1, spinner1, note1, paymentmode1, bd_date;  
    int transactionid1, image1, remain_bef_delete, remain_aft_delete, total_budget, budgetid;  
    ArrayList<Integer> transactionid, image, bid, remain_amt, total_amt;  
    ArrayList<String> bt_date;  
    MyTransactionItemAdapter myTransactionItemAdapter;  
    @Override  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_all_transactions);  
        dl = findViewById(R.id.drawer_layout);  
        recyclerView = findViewById(R.id.transaction);  
        fab1 = findViewById(R.id.addtrans);
```

```

imageView = findViewById(R.id.imgtrans);
textView = findViewById(R.id.txttrans);
myDB = new dbHelper(AllTransactions.this);
displayData();
fab1.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
Bundle bundle = new Bundle();
bundle.putBoolean("key", true);
Intent i = new Intent();
i.putExtras(bundle);
i.setClass(AllTransactions.this,addincome.class);
i.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK | Intent.FLAG_ACTIVITY_CLEAR_TASK);
startActivity(i);
}
});

myTransactionItemAdapter = new
MyTransactionItemAdapter(this,transactionid,dates,amount,typeoftransaction,spinner,note,paymentmode,image);
recyclerView.setAdapter(myTransactionItemAdapter);
recyclerView.setLayoutManager(new LinearLayoutManager(AllTransactions.this));

new ItemTouchHelper(new ItemTouchHelper.SimpleCallback(0,ItemTouchHelper.LEFT ) {
@Override
public boolean onMove(@NonNull RecyclerView recyclerView, @NonNull RecyclerView.ViewHolder
viewHolder, RecyclerView.ViewHolder target) {
return false;
}

@Override
public void onSwiped(@NonNull RecyclerView.ViewHolder viewHolder, int direction) {
int data = (Integer) viewHolder.itemView.getTag();
final int position = viewHolder.getLayoutPosition();
flag = true;
transactionid1 = transactionid.remove(position);

```

```

dates1 = dates.remove(position);
amount1 = amount.remove(position);
typeoftransaction1 = typeoftransaction.remove(position);
spinner1 = spinner.remove(position);
note1 = note.remove(position);
myTransactionItemAdapter.setCards(transactionid,dates,amount,typeoftransaction,spinner,note,paymentmode,image);
myTransactionItemAdapter.notifyItemRemoved(position);
Snackbar snackbar = Snackbar.make(recyclerView,"Transaction is Deleted",Snackbar.LENGTH_LONG);
snackbar.setAction("UNDO", v -> { flag = false;
transactionid.add(position,transactionid1);
dates.add(position,dates1);
amount.add(position,amount1);
typeoftransaction.add(position,typeoftransaction1);
spinner.add(position,spinner1);
note.add(position,note1);
paymentmode.add(position,paymentmode1);
image.add(position,image1);
myTransactionItemAdapter.setCards(transactionid,dates,amount,typeoftransaction,spinner,note,paymentmode,image);
myTransactionItemAdapter.notifyItemInserted(position);
});
snackbar.addCallback(new Snackbar.Callback() {
@Override
public void onDismissed(Snackbar transientBottomBar, int event) {
super.onDismissed(transientBottomBar, event);
if(flag) {
deleteData(data);
}}
});
snackbar.setActionTextColor(Color.RED);
snackbar.show();

```

Budget Java Code:

```

package com.example.expensify;

import androidx.drawerlayout.widget.DrawerLayout;
import androidx.recyclerview.widget.LinearLayoutManager;
import android.content.Intent;
import android.database.Cursor;
import android.os.Bundle;
import android.view.View;
import android.widget.ImageView;
import android.widget.LinearLayout;
import android.widget.TextView;
import java.util.ArrayList;
import java.util.Date;

```

```

public class Budget extends AppCompatActivity {
    FloatingActionButton fbbgt1;
    RecyclerView rv;
    LinearLayout hidel_budget;
    TextView no_budget,tap_budget;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_budget);
        dl = findViewById(R.id.drawer_layout);
        fbbgt1 = findViewById(R.id.addbugetclick);
        displayData();
        fbbgt1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Bundle bundle = new Bundle();
                bundle.putBoolean("key", true);
                Intent i = new Intent();
                i.putExtras(bundle);
                i.setClass(Budget.this,AddBudget.class);
                startActivity(i);
            }
        });
    }
}

```

```
}  
void displayData()  
{  
    Cursor cursor = myDB.GetBudget();  
    while (cursor.moveToNext())  
    {  
        budgetid.add(cursor.getString(0));  
        date.add(cursor.getString(1));  
        amount.add(cursor.getString(2));  
        total_amount.add(Integer.valueOf(cursor.getString(3)));  
    }  
}
```

CHAPTER 5

SNAPSHOTS

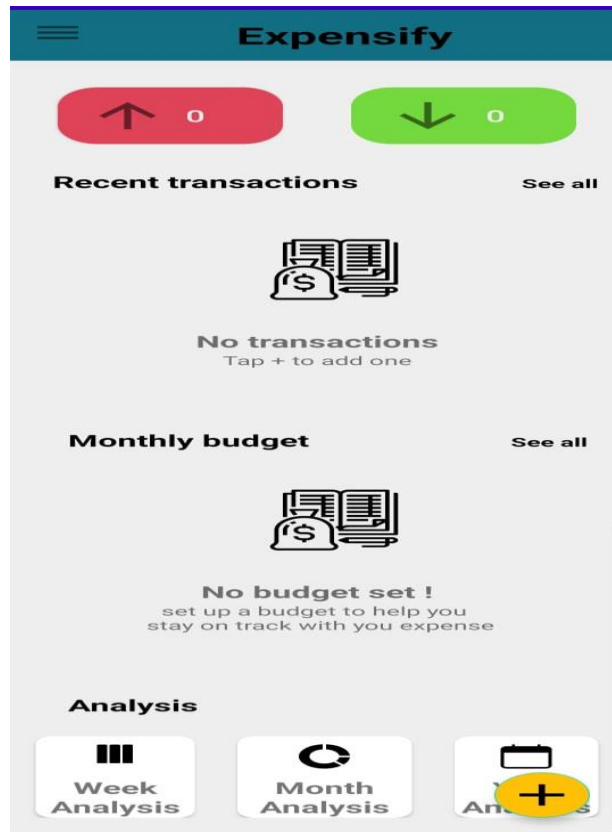


Fig 5.1: Home Page

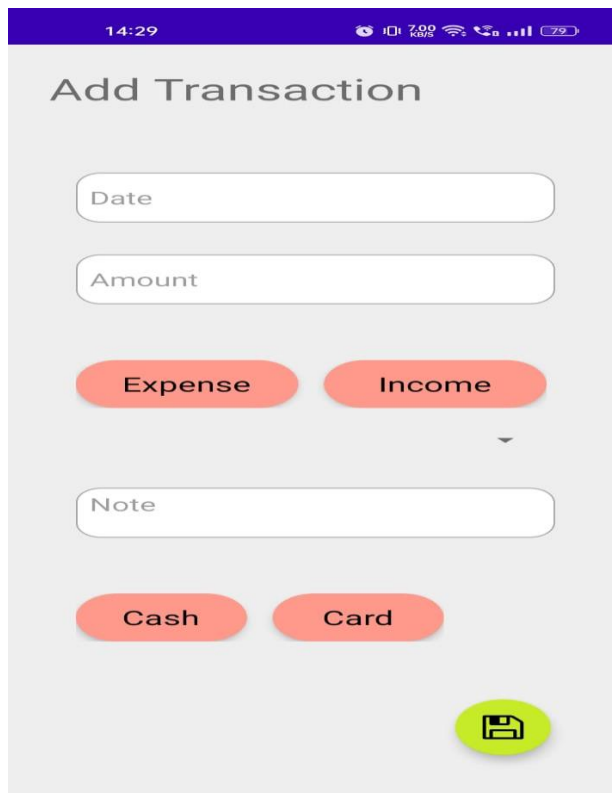


Fig 5.2: Transaction Details Page


14:30 0.00 79

Add Transaction

2023/07/09

50000

Expense Income

 Salary

salary credited

Cash Card




Fig 5.3: Income Transaction Page


14:31 0.00 79


Add Transaction

2023/07/08

1500

Expense Income

 Food

Food 

Cash Card




Fig 5.4: Expense Transaction Page

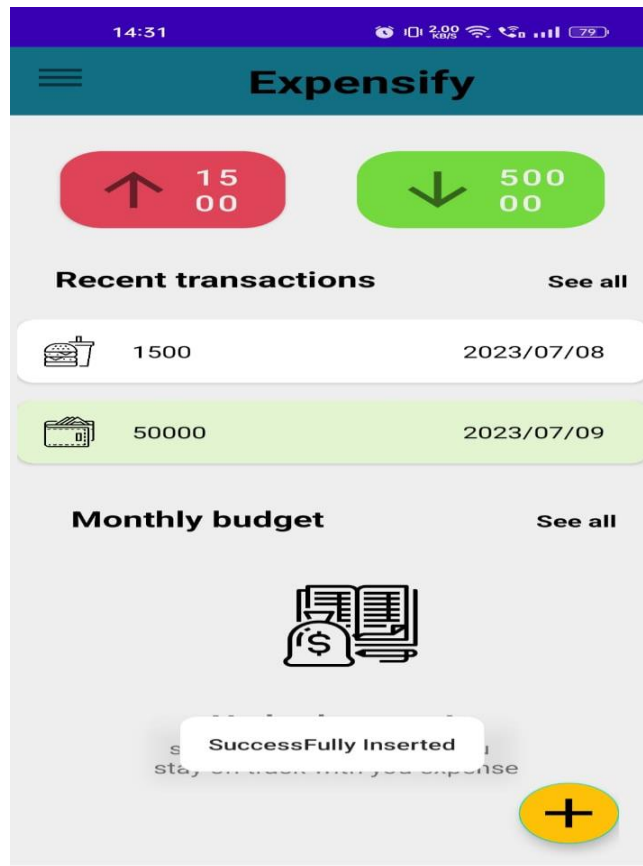


Fig 5.5: Recent Transaction Page

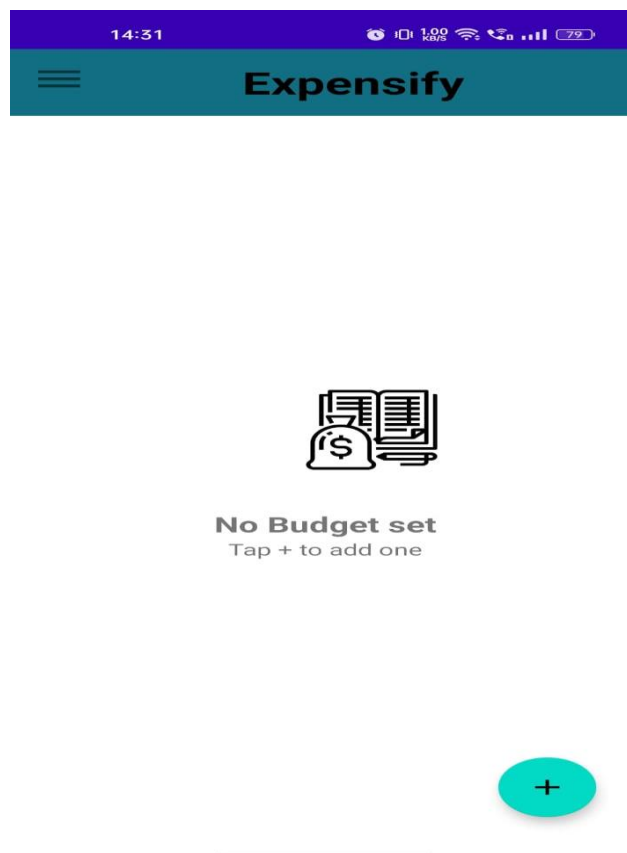


Fig 5.6: Budget Page

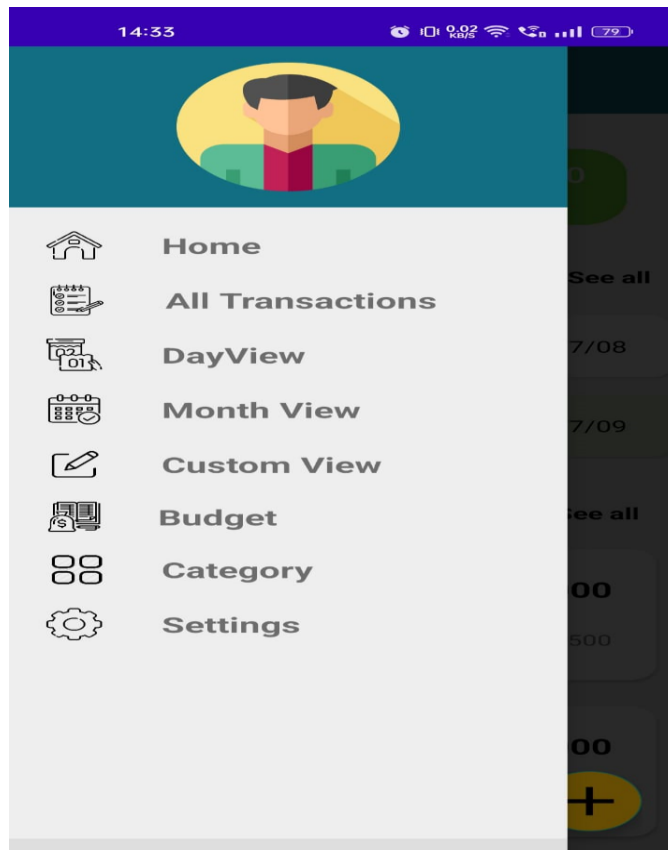


Fig 5.7: Dashboard Page

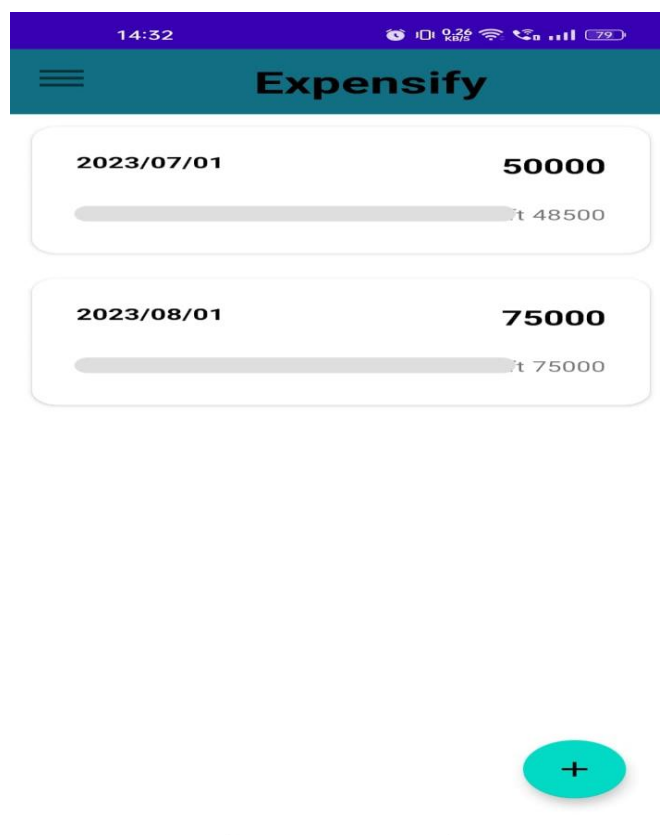


Fig 5.8: Budget Limit Page

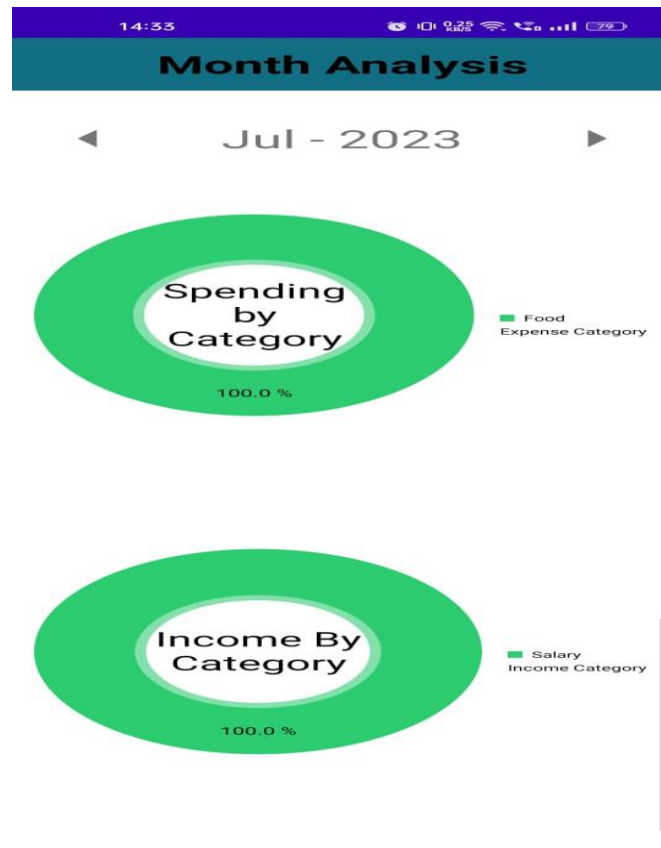


Fig 5.9: Pie Chart Analysis

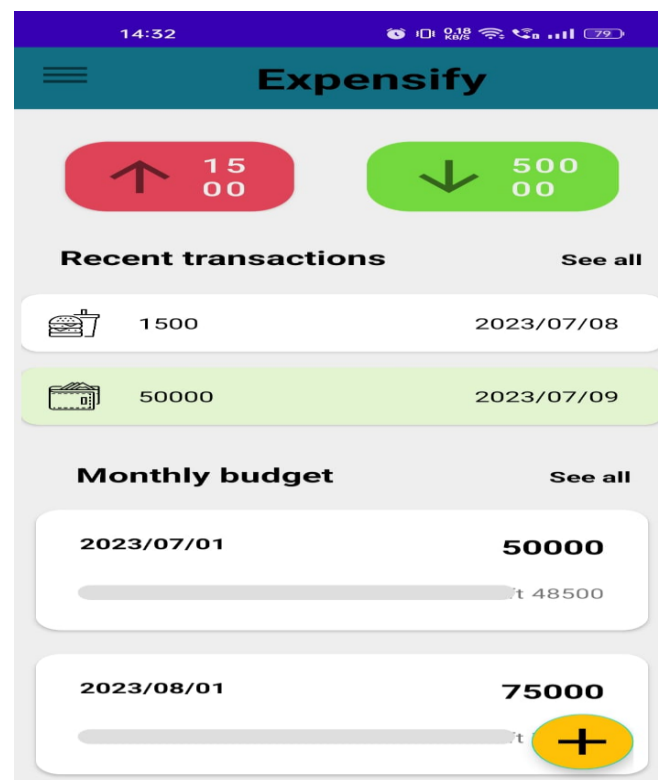


Fig 5.10: Dashboard Viewing Page

CHAPTER 6

CONCLUSION

The expense manager provides individuals and businesses with an efficient and user-friendly tool to track and manage their expenses effectively. By incorporating various features such as expense categorization, budget tracking, and data visualization, the expense manager offers a comprehensive solution for maintaining financial stability and making informed decisions. Firstly, the expense manager allows users to categorize their expenses, enabling them to organize their financial data efficiently. This feature enables users to allocate expenses to specific categories such as food, transportation, entertainment, and more. Categorization facilitates the identification of spending patterns and areas where adjustments can be made, promoting better financial planning and saving strategies. Secondly, the budget tracking functionality of the expense manager proves to be indispensable for individuals and businesses aiming to maintain fiscal discipline. Users can set budgets for various expense categories and receive notifications or warnings when they approach or exceed their predefined limits. This capability promotes responsible spending habits and helps prevent overspending, ultimately contributing to better financial management.

Moreover, the expense manager's data visualization tools provide users with a clear and comprehensive overview of their financial situation. Through charts, graphs, and reports, users can analyze their spending patterns, identify trends, and gain insights into their financial health. This visual representation of data facilitates decision-making by highlighting areas of improvement or success and allows users to adjust their financial strategies accordingly.

REFERENCES

Textbooks:

- Expense Manager: A User-Friendly Android App for Personal Finance Management By Smith, J (2020) - Conceptual Reference
- Simplifying Personal Finance Management with Expense Manager by Johnson, A. (2019) - Implementation Guide

Websites:

1. Android Developer Documentation. (<https://developer.android.com/docs>)
2. Java Documentation. (<https://docs.oracle.com/en/java/>)
3. Android Architecture Patterns(<https://www.geeksforgeeks.org/android-architecture-patterns/>)
4. <https://www.tutorialspoint.com/java/index.html>
5. <https://stackoverflow.com/tags/java>