VISVESVARAYA TECHNOLOGICAL UNIVERSITY

“JNANA SANGAMA”, BELGAUM-590018.

A
Mini Project Report
On

“EXPENSE TRACKER APPLICATION”

SUBMITTED IN FULFILLMENT OF THE REQUIREMENT IN
6TH SEMESTER MOBILE APPLICATION DEVELOPMENT MINI PROJECT WORK
OF BACHELOR OF ENGINEERING
IN

COMPUTER SCIENCE AND ENGINEERING

By

Varshitha Y S [1JB20CS134]
Veekshitha B N [1JB20CS135]

UNDER THE GUIDANCE OF

Mr. Dhananjaya M
Assistant Professor
Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SJB INSTITUTE OF TECHNOLOGY

#67, BGS HEALTH AND EDUCATION CITY, Dr. Vishnuvardhana Road,
Kengeri, Bangalore-560060.

2022-2023

||Jai Sri Gurudev||

Sri Adichunchanagiri Shikshana Trust ®

SJB INSTITUTE OF TECHNOLOGY

#67, BGS HEALTH AND EDUCATION CITY, Dr. Vishnuvardhana Road, Kengeri, Bangalore-560060.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the mini project report entitled **"Expense Tracker Application"** is a bonafide work carried out **Varshitha Y S [1JB20CS134]** and **Veekshitha B N [1JB20CS135],** in partial fulfillment of mini project of 6th semester Mobile Application Development Laboratory (18CSMP68) in **COMPUTER SCIENCE AND ENGINEERING** of **VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM,** during the academic year **2022-2023**. It is certified that all corrections/suggestions have been incorporated in the report deposited in the department library. The mini project report has been approved as it satisfies the academic requirements.

Signature of the guide	Signature of the HOD
Mr. Dhananjaya M	**Dr. Krishna A. N**
Asst. Professor	**Professor & Head**
Dept. of CSE	**Dept. of CSE**

Name of the Examiner:	**Signature of the Examiner:**
1. ________________________	________________________
2. ________________________	________________________

ACKNOWLEDGEMENT

We would like to express our profound gratefulness to His Divine Soul **Jagadguru Padmabhushan Sri Sri Sri Dr.Balagangadharanatha Mahaswamiji** and His Holiness **Jagadguru Sri Sri Sri Dr.Nirmalanandanatha Mahaswamiji** for providing us with an opportunity to study in this esteemed Institution.

We would like to express our profound thanks to **Revered Sri Sri Dr. Prakashnath Swamiji,** Managing Director, SJB Institute of Technology, for his continuous support in providing amenities to carry out this mini project in this admired institution.

We express our gratitude to **Dr. K V Mahendra Prashanth**, Principal, SJB Institute of Technology, for providing us with excellent facilities and academic ambience; which have helped us in satisfactory completion of this mini project.

We extend our sincere thanks to **Dr. Krishna A N,** Head of the Department of Computer Science and Engineering, for providing us with invaluable support throughout the period of our mini project.

We wish to express our heartfelt gratitude to our guides, **Mr. Dhananjaya M,** Assistant professor, Dept. of CSE for their valuable guidance, suggestions and cheerful encouragement during the entire period of this mini project work.

Finally, we take this opportunity to extend our earnest gratitude and respect to our parents, Teaching & Non-teaching staffs of the department, the library staff and all our friends, who have directly or indirectly supported us during the period of our mini project work.

Regards,

Varshitha Y S [1JB20CS134]
Veekshitha B N [1JB20CS135]

ABSTRACT

The Project “Expense Tracker” is a user-friendly web application that allows individuals to effectively manage their income and expenses. It offers a convenient way for users to create an account, log in securely, and access their personal financial data. The application utilizes “SQLite” as its database to store and manage transaction records. Users can view their transactions based on a “calendar interface”, enabling them to track their spending habits and budget more effectively. The Expense Tracker offers comprehensive features to record income, categorize expenses, and perform real-time calculations to provide users with a clear overview of their financial status. With its intuitive interface and powerful functionality, the Expense Tracker aims to “simplify personal finance management” and empower individuals to make informed financial decisions. The Expense Tracker offers a range of features to assist users in maintaining their financial records. Users can add income sources and record expenses, assigning appropriate categories to each transaction. The calendar-based interface allows users to view their transactions on specific dates, making it easier to analyze spending patterns over time. Real-time calculations are performed to provide users with an accurate balance, enabling them to stay on top of their financial situation. The application also provides graphical representations, such as expense charts and income breakdowns, to enhance data visualization and aid in financial analysis.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
INTRODUCTION

Android is the Linux-based open-source operating system for mobile devices like smartphones & tablets. However, nowadays, many other devices are incorporating android in them to turn them into smart devices such as Smart TVs, Smart car interface for GPS, electrical appliances, etc. This software was unveiled in 2007 & the first Android Device was launched in September 2008. Since then Google, the sponsor of Android has been releasing its software updates, versions almost every year.

Android also offers several features:

- NFC (Near Field Communications): NFC Allows electric devices to easily interact across short distances.
- Alternate keywords: It supports multiple keyboards & makes them easy to install.
- Beautiful and Interactive UI
- Storage: SQLite a lightweight relational DB is used for data storage.
- Multi lang: Supports single direction & Bi-Directional.

Android offers a unified approach to application development for mobile devices which means developers need only develop for Android, and their applications should be able to run on different devices powered by Android.The source code for Android is available under free and open source software licenses.

Google publishes most of the code under the Apache **Android Studio** is the official[8] integrated development environment (IDE) for Google's Android operatingsystem, built on JetBrains' IntelliJ IDEA software and designed specifically for Android development. Android Studio is a powerful integrated development environment (IDE) specifically designed for mobile application development on the Android platform. With its comprehensive set of tools and intuitive user interface, Android Studio enables developers to create high-quality mobile applications efficiently. It offers a wide range of features, including code editing, debugging, and testing, along with a built-in emulator for simulating various Android devices. Android Studio supports multiple programming languages such as Java and Kotlin, allowing developers to leverage their preferred language. Additionally, it provides access to a vast array of libraries, APIs, and development resources, facilitating the creation of visually appealing and feature-rich mobile applications.

1.1 Scope

The main purpose of a Expense tracker can vary depending on the specific goals and features anyone want to include. Seems simple, right? However, here common features and functionalities that we can consider is User registration and Authentication. This User registration and Authentication will allow the users to create accounts and log in securely and implement authentication mechanisms to protect user data. The Expense Tracking enables users to record their expenses by entering details such as amount, date, category, and description. It also supports multiple categories for expenses such as food, transportation, entertainment etc. It will also provide the options to the users to add tags or labels to expenses for better organization. Allows the user to edit or delete the recorded expenses which is user friendly. It will also implement the features for recurring expenses or scheduled payments.

It will also include functionality to track and record user income from various sources which enables the users to categorize their income sources.

Expenses tracker provides tools for user to set budgets for different expense categories or overall spending. It will display progress and alerts when users analyze their spending habits. It contains Expense Analytics which generate charts, graphs or reports to display spending patterns over time or through category. Has special feature called Receipt Management which allows users to capture and sort images of receipts for future reference or expense verification. It has good Data backup and Synchronization which enables users to back up their data securely and restore it if needed. It also provides the options for data synchronization across multiple devices.

It important to note that it will depend on factors such as target audience, platform and development resources. It’s recommended to priorities features based on user needs and focus on delivering a polished and user-friendly environment.

1.2 OBJECTIVE

The purpose of a Expense tracker is to help users manage their finances efficiently by tracking and categorizing their expenses. The main goal of the Expense tracker the app should allow users to easily record their expenses by entering details such as the amount spent, date, and category of the expense. It should provide a user-friendly interface for quickly inputting and saving expense information. The app should help the users set and manage budgets for different expense categories such as groceries, transportation, entertaining etc. It should provide visual representations of budget limits and notifications when approaching or exceeding those limits.

The app should maintain a history of past expenses and provide tools for users to review their spending over time. This could include features like filtering expenses by date range, viewing monthly or yearly summaries, and identifying spending trends. The app should have an intuitive and user-friendly interface that makes it easy for users to navigate, input expenses and access relevant features. It should also consider accessibility guidelines to ensure that people with disabilities can use the app efficiently.

The overall objective of an Expense tracker app is to empower users to take control of their finances, make informed spending decisions, and work towards their financial goals.

1.3 Challenges

Developing a mobile application for Expense tracker can present several challenges. Firstly, obtaining reliable and up-to-date expense data of the users. Ensuring that users can easily navigate and understand the app's features can be a challenge. Implementing encryption, secure authentication, and protecting user data from unauthorized access are vital challenges that could be faced. If the app supports multiple devices or platforms, syncing data across different devices can be challenging. Ensuring that all user data remains consistent and up to date across devices requires careful synchronization mechanisms.

Integrating with financial institutions or payment gateways to automatically fetch transaction data can be complex. Each service may have different API's, authentication methods and data formats, requiring through integration efforts.

As the user base and data grow, the app needs to maintain good performance and handle increase traffic and data storage. Building Expense tracker app that runs smoothly across multiple platforms requires platform-specific development knowledge and addressing compatibility issues.

Addressing all these challenges requires careful planning, strong development skills and effective collaboration with designers, developers, and stakeholders. Regular user feedback and iterative development can help refine the app and overcome these challenges over time.

CHAPTER 2

LITERATURE SURVEY

The purpose of this literature survey is to explore the existing research and literature related to the development of mobile applications for Expense Tracker. The survey aims to identify key concepts, technologies, and methodologies utilized in similar applications and provide insights into the challenges faced and solutions proposed in this domain.

2.1 Expense Tracker Applications:

Numerous mobile applications have been developed for Expense Tracker, catering to the growing demand for user-generated content and recommendations. Existing literature discusses various aspects of these applications, including their features, user interfaces, user experience, and backend systems. Researchers have highlighted the importance of personalized recommendations, social interaction features, and real-time updates to enhance user engagement and satisfaction.

2.2 User-generated Content and Sentiment Analysis:

Studies have focused on analyzing user-generated content, particularly Expense Tracker, to extract sentiment and opinion. Natural Language Processing (NLP) techniques, such as sentiment analysis and topic modeling, have been employed to understand users' sentiments towards movies. These analyses assist in generating aggregated ratings and sentiment-based recommendations for personalized user experiences.

2.3 Collaborative Filtering and Recommendation Systems:

Literature highlights the significance of collaborative filtering and recommendation systems in Expense Tracker applications. Researchers have explored different algorithms and approaches, including user-based and item-based collaborative filtering, matrix factorization, and hybrid recommendation systems. These techniques leverage user preferences and behavior to provide accurate movie recommendations, improving user satisfaction and engagement.

2.4 User Experience and Interface Design:

Studies emphasize the importance of intuitive and user-friendly interfaces for Expense

Tracker applications. User experience (UX) design principles, such as information architecture, usability, and visual design, play a crucial role in enhancing user engagement and retention. Existing literature suggests incorporating features like search filters, personalized watchlists, and easy-to-use rating systems to improve the overall user experience.

2.5 Backend Infrastructure and Data Management:

The backend infrastructure of Expense Tracker applications involves managing movie data, user profiles, and Tracking. Researchers have discussed various approaches, including data scraping, API integration with movie databases, and cloud-based storage solutions. Scalability, security, and real-time updates are key considerations in designing an efficient backend architecture for Expense Tracker applications.

2.6 Mobile Development Frameworks and Technologies:

Literature surveys highlight the popular mobile development frameworks, such as Android Studio and iOS development tools, along with programming languages like Java, Kotlin, and Swift. Cross-platform frameworks like React Native and Flutter are also gaining attention due to their ability to develop applications for multiple platforms simultaneously.

The literature survey provides a comprehensive overview of the existing research and literature related to Expense Tracker mobile applications. It covers key aspects such as user-generated content analysis, collaborative filtering, user experience design, backend infrastructure, and mobile development frameworks. The insights gained from this survey will serve as a foundation for the development of a Expense Tracker mobile application, considering the challenges and solutions discussed in the literature.

CHAPTER 3
SYSTEM REQUIREMENTS

System requirements play a crucial role in the development of mobile applications. They define the necessary hardware, software, and infrastructure components that a mobile device or platform must have in order to run the application smoothly and effectively. System requirements ensure that the application functions optimally, meets performance expectations, and provides a seamless user experience across a wide range of devices.

3.1 Hardware Requirements

- Processor: Intel CORE i3 or higher
- Ram: 8GB
- Hard Disk: 1TB

3.2 Software Requirements

- Android Studio version 2.3.3
- Internet Connection

Chapter 4
System Design

4.1 Architecture of the Project [Expense Tracker]

4.1.1 Flow Chart

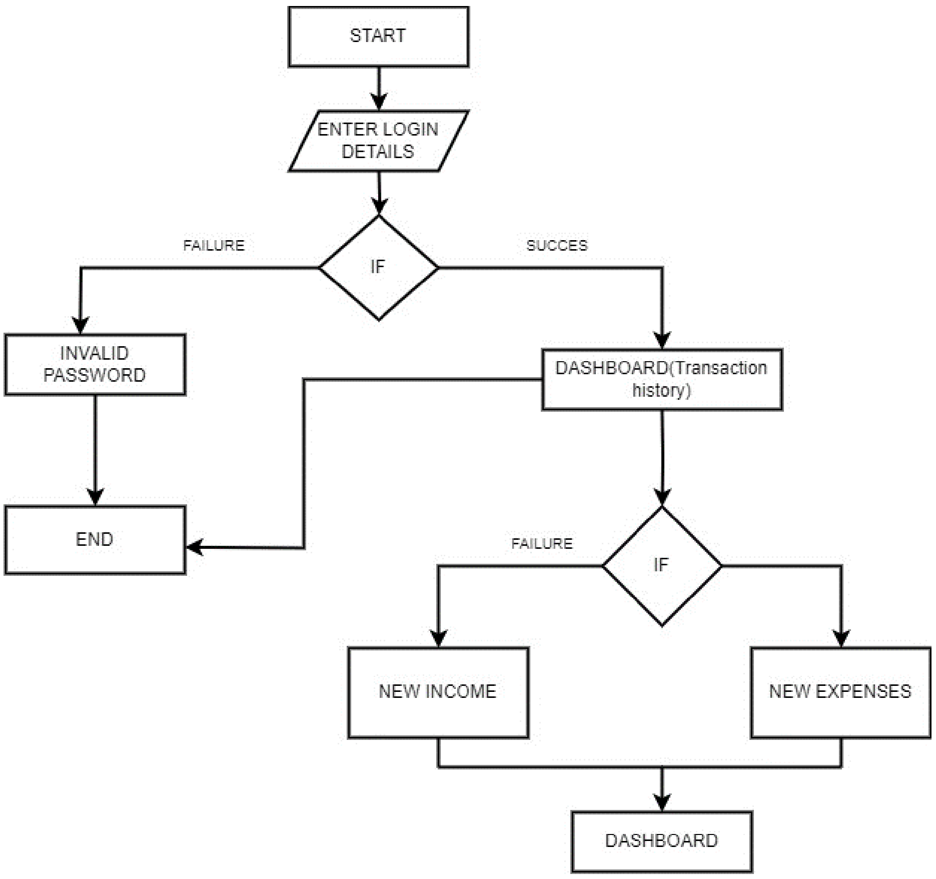


Fig 4.1 Flow Diagram of the Expense Tracker

4.1.2 Command

In computing, a command is **a directive to a computer program to perform a specifictask**. It may be issued via a command-line interface, such as a shell, or as input to a network service as part of a network protocol, or as an event in a graphical user interfacetriggered by the user selecting an option in a menu.

4.1.3 Files Used

- MainActivity.java
- activity_main.xml
- activity_sign_in.xml
- activity_update_transaction.xml
- DataSaver.java
- New_Expense.java
- Sign_up.java
- TransactionModel.java
- activity_income.xml

4.1.4 Description of Function

Sl.No	Functions	Description
1.	Text View	A TextView displays text to the user and optionally allows them to edit it.A TextView is a complete text editor, however the basic class is configured to not allow editing.
2.	Edit Text	A EditText is an overlay over TextView that configures itself to be editable. It is the predefined subclass of TextView that includes richediting capabilities.
3.	Button	In Android, Button represents a push button. A Push buttons can beclicked, or pressed by the user to perform an action.
4.	Radio Button	Radio buttons allow the user to select one option from a set. You should use radio buttons for optional sets that are mutually exclusive if you thinkthat the user needs to see all available options side-by-side. If it's not necessary to show all options side-by-side, use a spinner instead.
5.	Widgets	widget is a small gadget or control of the android application placed on home screen. Widgets can be very handy as they allow you to put your favorite applications on your home screen in order to quickly access them.

6.	Image View	Displays image resources, for example Bitmap or Drawable resources. ImageView is also commonly used to apply tints to an image and handleimage scaling.
7.	Hint	android: hint is more like a placeholder that sort of explains what type ofinput the EditText is asking for. i.e. If an EditText is asking for posting astatus on social media, the hint like What's on your mind? will be suitable.
8.	Toast	A toast provides simple feedback about an operation in a small popup. It only fills the amount of space required for the message and the currentactivity remains visible and interactive. Toasts automatically disappear after a timeout.

4.1.5 Code Snippets

XML CODE

activity_sign_in.xml

```
<?xml version="1.0" encoding="utf-8"?>
    <androidx.constraintlayout.widget.ConstraintLayout
    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/bg"
    tools:context=".Sign_in">

    <EditText
        android:id="@+id/signinPasswordid"
        android:layout_width="match_parent"
        android:layout_height="48dp"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="30dp"
        android:layout_marginRight="15dp"
        android:background="@color/semiTransparent"
        android:backgroundTint="#CC4572"
        android:drawableRight="@drawable/ic_baseline_info_24"
        android:ems="10"
        android:hint="Password"
        android:inputType="textPassword"
        android:paddingLeft="16dp"
        android:paddingRight="10dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.517"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/signInEmailid" />

    <EditText
        android:id="@+id/signInEmailid"
        android:layout_width="match_parent"
        android:layout_height="48dp"
        android:layout_marginLeft="15dp"
        android:layout_marginTop="76dp"
        android:layout_marginRight="15dp"
        android:background="@color/semiTransparent"
        android:backgroundTint="#B8496F"
        android:drawableRight="@drawable/ic_baseline_email_24"
        android:ems="10"
        android:hint="Email"
        android:inputType="textEmailAddress"
        android:paddingLeft="16dp"
        android:paddingRight="10dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.533"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView3" />

    <Button
        android:id="@+id/login"
        android:layout_width="120dp"
        android:layout_height="50dp"
        android:layout_marginTop="36dp"
        android:backgroundTint="@color/Mauve"
        android:text="Login"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/signinPasswordid" />

    <TextView
        android:id="@+id/Tosignupid"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="28dp"
        android:layout_marginBottom="160dp"
        android:text="Create an account"
        android:textColor="@color/black"
        android:textSize="16sp"
        android:textStyle="bold"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.516"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintVertical_bias="0.0" />
    <TextView
        android:id="@+id/textView3"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="88dp"
        android:background="@drawable/circle"
        android:gravity="center"
        android:textSize="20sp"
        android:textColor="@color/white"
```

```
        android:text="LOGIN"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.498"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

JAVA CODE

MainActivity.java

```
package com.example.practiceexptracker;
import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import com.example.practiceexptracker.databinding.ActivityMainBinding;
import java.util.ArrayList;
import java.util.List;
import java.util.Objects;

public class MainActivity extends AppCompatActivity {
   ActivityMainBinding binding;
   UltAdapter myAdapter;
   List<TransactionModel> list;
```

```
RecyclerView rv;
DataSaver myData;
String N_pos;
int totalIncome =0;
int totalExpense = 0;
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    binding = ActivityMainBinding.inflate(getLayoutInflater());
    setContentView(binding.getRoot());
    getSupportActionBar().hide();
    Toolbar toolbarTop = (Toolbar) findViewById(R.id.toolbar_top);
    toolbarTop.inflateMenu(R.menu.menu_items);
    TextView mTitle = (TextView) toolbarTop.findViewById(R.id.toolbar_title);
    Bundle data = getIntent().getExtras();
    myData = new DataSaver(this);
    list = RetrieveData();
    if(data != null)
    {
        N_pos = data.getString("ItemPos");
    }
    rv = binding.recyclerView;
    rv.setLayoutManager(new LinearLayoutManager(this));
    TransactionModel t = new TransactionModel("1","2000","lunch","22/2/22","income");
    myAdapter = new UltAdapter(list,this);
    rv.setAdapter(myAdapter);
    binding.incomebtnid.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View view) {
            Intent i = new Intent(MainActivity.this,New_Income.class);
            startActivity(i);
    binding.expensebtnid.setOnClickListener(new View.OnClickListener() {
        @Override
```

```
            public void onClick(View view) {
                Intent i = new Intent(MainActivity.this,New_Expense.class);
                    startActivity(i);
                }});}
     public List<TransactionModel> RetrieveData()
    {
        List<TransactionModel> newlist = new ArrayList<>();
        newlist = myData.ReadAllData();
        for(int i=0;i<newlist.size();i++)
        {
            TransactionModel Tm = newlist.get(i);
            int amount = 0;
            try
            {
                amount = Integer.parseInt(Tm.getAmount());
            }
            catch (Exception e)
            {
                amount = 0;
            }
            if (Objects.equals(Tm.getType(), "Income")) {
                    totalIncome += amount;
            } else {
                    totalExpense += amount;
            }
        }
        binding.IncomeAmountid.setText(String.valueOf(totalIncome));
        binding.ExpensesAmountid.setText(String.valueOf(totalExpense));
        binding.BalanceAmountid.setText(String.valueOf(totalIncome - totalExpense));
        return  newlist;
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_items,menu);
        return true;
    }}
```

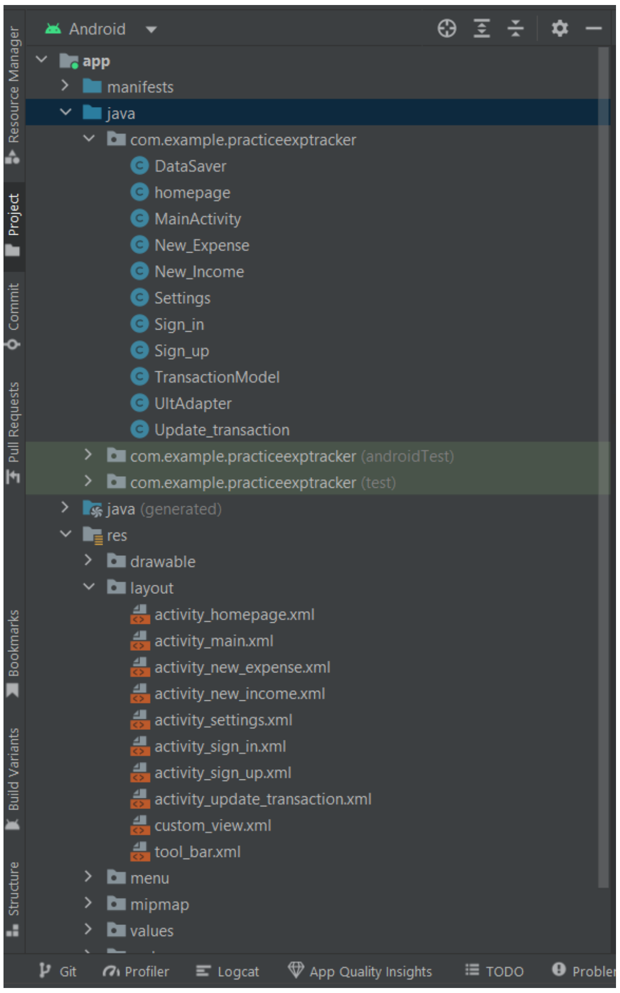
DataSaver.java

```
package com.example.practiceexptracker;
import android.content.ContentValues;
import android.content.Context;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;

import java.util.ArrayList;

public class DataSaver extends SQLiteOpenHelper {

    private static final String DbName = "tansacdb";
    private static final int version = 1;
    private static final String TableName = "TransTable";
    private static final String ID = "id";
    private static final String Amount = "amount";
    private static final String Note = "note";
    private static final String type = "type";
    private static final String date = "date";

    public DataSaver(@Nullable Context context) {
        super(context, DbName, null, version);
    }
```

```
@Override
public void onCreate(SQLiteDatabase db) {
   String theQuery = "CREATE TABLE " + TableName +
         " (" + ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " +
         Amount + " TEXT, " + Note +
         " TEXT, " + type + " TEXT, " +  date + " TEXT)";
   db.execSQL(theQuery);
}

@Override
public void onUpgrade(SQLiteDatabase sqLiteDatabase, int i, int i1) {
   sqLiteDatabase.execSQL("DROP TABLE IF EXISTS " + TableName);
   onCreate(sqLiteDatabase);
}

public void AddData(String amount, String note,String Type,String Date)
{
   SQLiteDatabase db = this.getWritableDatabase();

   ContentValues values = new ContentValues();
   values.put(Amount,amount);
   values.put(Note,note);
   values.put(type,Type);
   values.put(date,Date);
   db.insert(TableName,null,values);
   db.close();

}

public ArrayList<TransactionModel> ReadAllData()
{
   SQLiteDatabase db = this.getReadableDatabase();
   Cursor cur = db.rawQuery("SELECT * FROM " + TableName,null);
```

```
        ArrayList<TransactionModel> list = new ArrayList<>();

        if(cur.moveToFirst())
        {
            do
            {
                int index = cur.getColumnIndexOrThrow(ID);
                String id = cur.getString(index);
                index = cur.getColumnIndexOrThrow(Amount);
                String amount = cur.getString(index);
                index = cur.getColumnIndexOrThrow(Note);
                String note = cur.getString(index);
                index = cur.getColumnIndexOrThrow(type);
                String mytype = cur.getString(index);
                index = cur.getColumnIndexOrThrow(date);
                String Date = cur.getString(index);
                TransactionModel transactionModel =
                        new TransactionModel(id,amount,note,Date,mytype);
                list.add(transactionModel);
            }
            while(cur.moveToNext());
        }
        cur.close();
        return list;
    }

    public void deleteTransaction(int id)
    {
        SQLiteDatabase db = this.getWritableDatabase();
        db.delete(TableName, "id=?", new String[]{String.valueOf(id)});
        db.close();
    }
    public void updateTrans(String amount,String note,int id) {
        SQLiteDatabase db = this.getWritableDatabase();
        ContentValues values = new ContentValues();
        values.put(Amount, amount);
        values.put(Note, note);
        db.update(TableName, values, "id=?", new String[]{String.valueOf(id)});
        db.close();
    }
}
```

4.1.6 Folders used



```
app
    manifests
    java
        com.example.practiceexptracker
            DataSaver
            homepage
            MainActivity
            New_Expense
            New_Income
            Settings
            Sign_in
            Sign_up
            TransactionModel
            UltAdapter
            Update_transaction
        com.example.practiceexptracker (androidTest)
        com.example.practiceexptracker (test)
    java (generated)
    res
        drawable
        layout
            activity_homepage.xml
            activity_main.xml
            activity_new_expense.xml
            activity_new_income.xml
            activity_settings.xml
            activity_sign_in.xml
            activity_sign_up.xml
            activity_update_transaction.xml
            custom_view.xml
            tool_bar.xml
        menu
        mipmap
        values
```

Fig 4.1 Folders Used in Expense Tracker

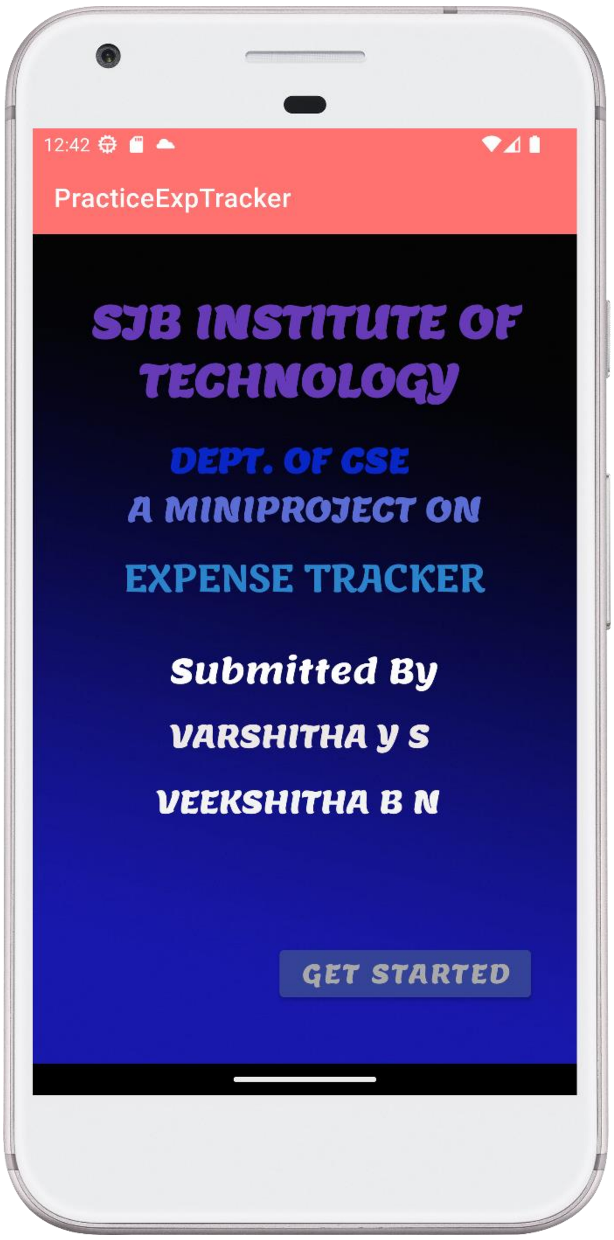
4.2 Implementation

```
import androidx.appcompat.app.AppCompatActivity;
import android.content.Intent;
import android.os.Bundle;
import java.util.Calendar;
import android.view.View;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Button;
import android.widget.EditText;
import java.util.regex.Matcher;
import android.database.Cursor;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;
import androidx.recyclerview.widget.LinearLayoutManager;
import androidx.recyclerview.widget.RecyclerView;
```

CHAPTER 5
RESULTS

Home Page:

- Fig 5.1 shows the Home page where it contains the details about mini project title.
- It has one button which leads to the login page.



12:42

PracticeExpTracker

SJB INSTITUTE OF TECHNOLOGY

DEPT. OF CSE

A MINIPROJECT ON

EXPENSE TRACKER

Submitted By

VARSHITHA Y S

VEEKSHITHA B N

Fig 5.1 Home Page of Expense Tracker

Login Page:

- Fig 5.2 shows the Login page where the user can enter his credentials or signup using create account.
- There are 2 input text boxes, one for email and another for password.
- It contains one button to login , text view to create an account leads to signup page.

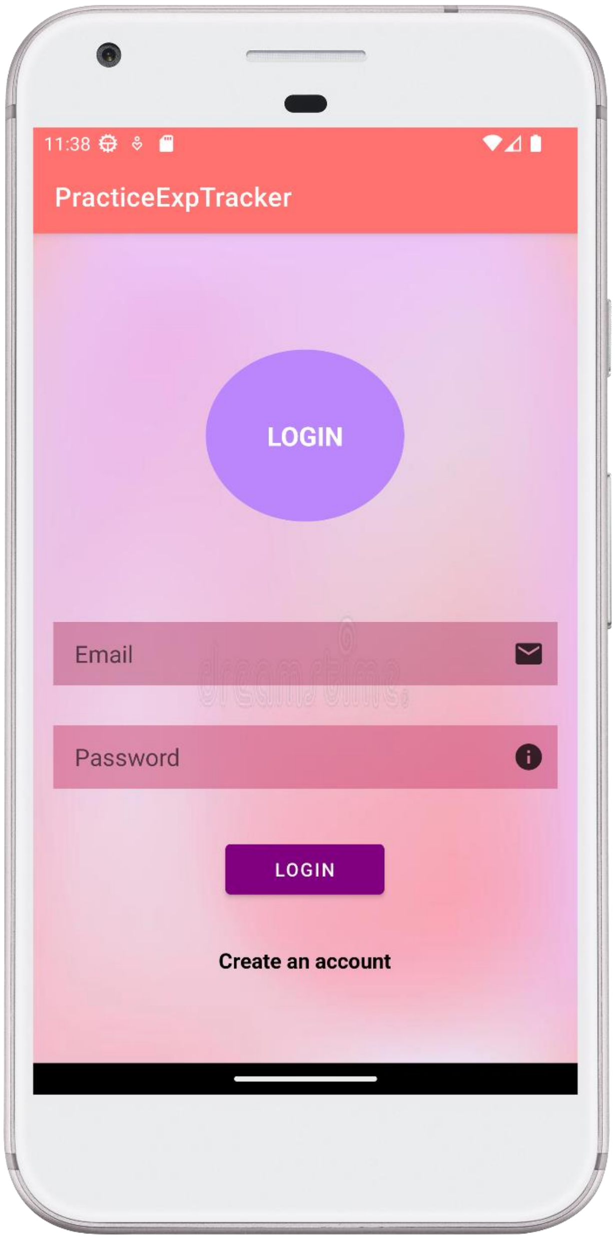
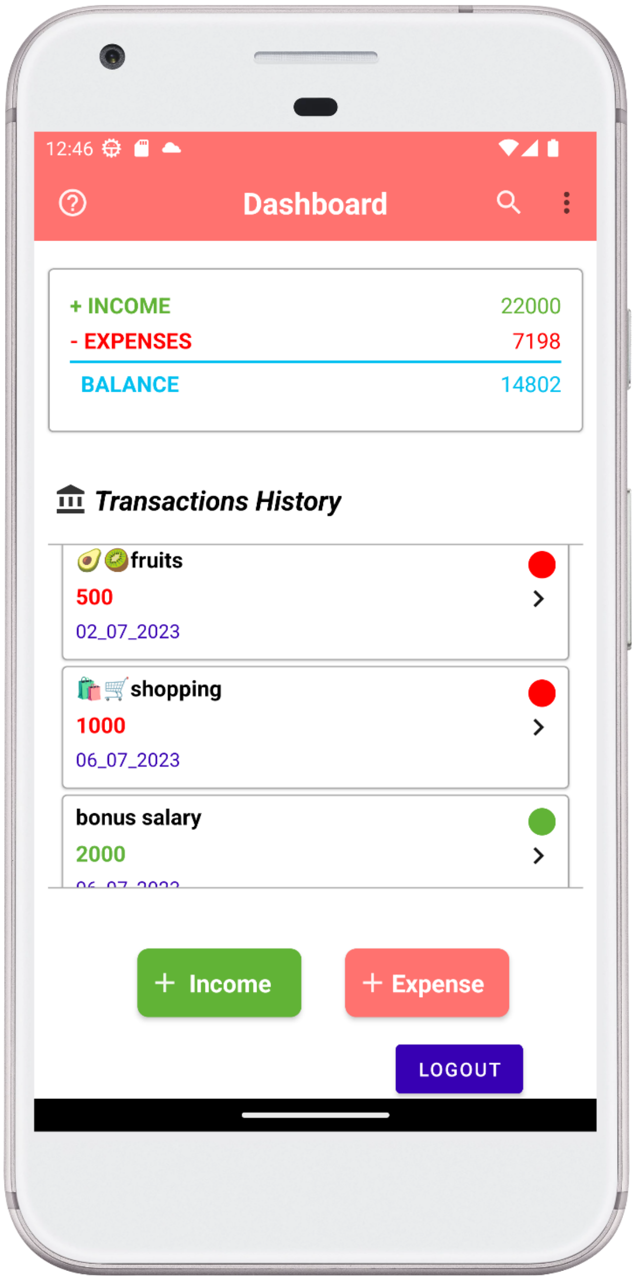


Fig 5.2 Login Page of Expense Tracker

Dashboard:

- Fig 5.3 shows the Income where the user can update his/her expenses and income.
- Its shows the transaction history along with that it has the logout button.


12:46

Dashboard

+ INCOME	22000
- EXPENSES	7198
BALANCE	14802

Transactions History

🥑🥝fruits
500
02_07_2023

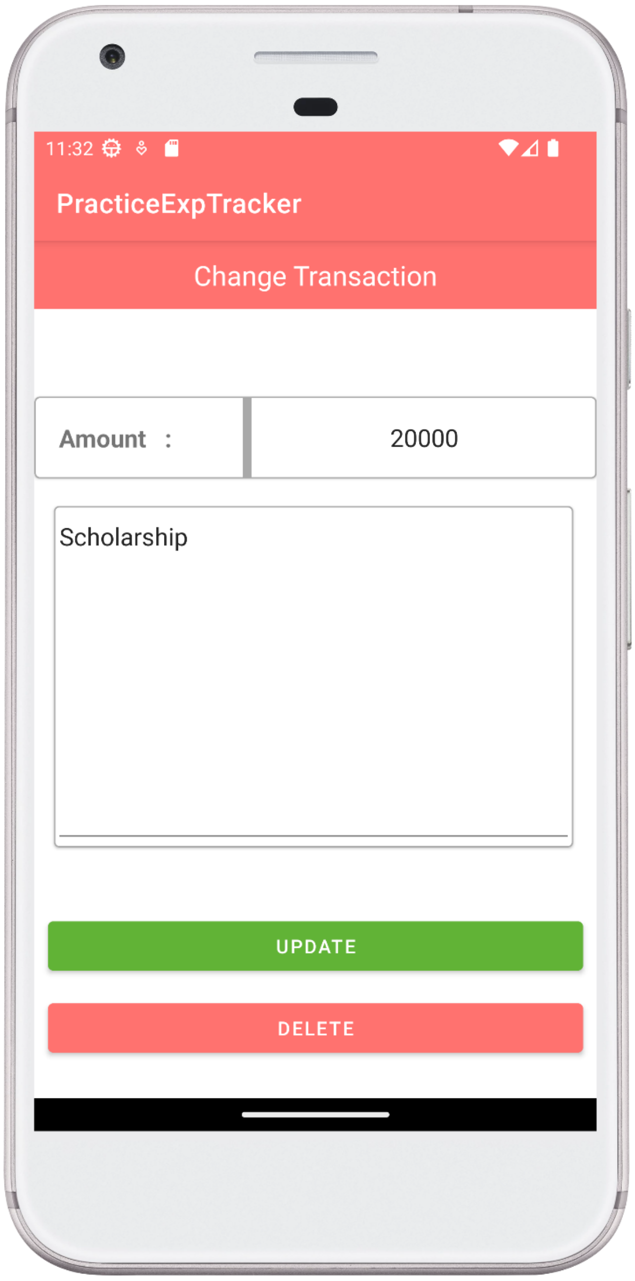
🛍️🛒shopping
1000
06_07_2023

bonus salary
2000

Fig 5.3 Transaction history of Expense Tracker

Expense Details:

- Fig 5.4 shows the Expense amount entered in text box along with the detail of the Expense entered in the other text box which can be updated and deleted.



Amount : 20000

Scholarship

Fig 5.4 Expense Details of Expense Tracker

Review Page:

- Fig 5.5 shows the page that contains the entry of amount and note for new income.
- Here, user can add new income and it will be added to the transaction.

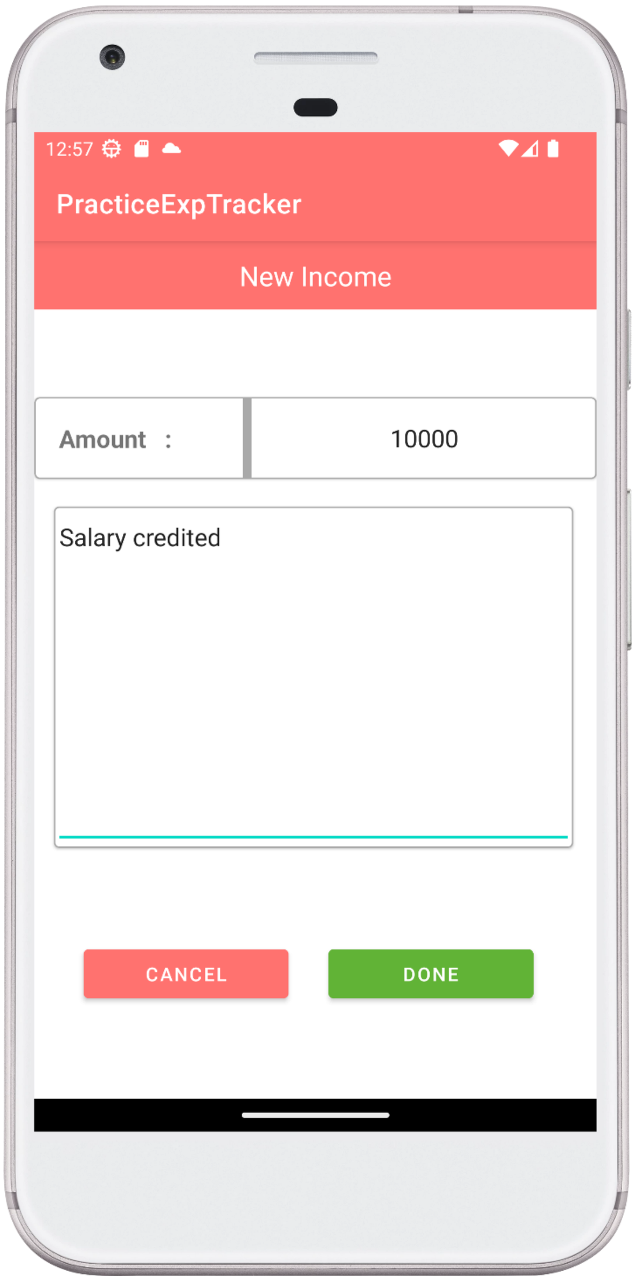


Fig 5.5 New Income page of Expense Tracker

CONCLUSION AND FUTURE ENHANCEMENTS

In conclusion, the Expense Tracker app offers a user-friendly solution for managing income and expenses. With its intuitive interface and functionality, users can easily input transactions, categorize them, and view a summary of their financial situation. Future enhancements could include the addition of expense graphs and visual representations to provide users with a visual overview of their spending patterns. Implementing a budgeting feature with reminders and notifications would help users stay on track with their financial goals. Integrating advanced analysis features, such as trend analysis and expense forecasting, would provide deeper insights into financial habits. Syncing with financial institutions and enabling automatic transaction syncing would streamline the tracking process. Multi-platform versions and cloud synchronization would enhance accessibility and data security. Overall, these enhancements would empower users to make informed decisions and achieve their financial goals more effectively.

Other challenges include :-

Implementing the Expense Tracker app poses challenges in data accuracy, security, user experience design, and integration.

Ensuring accurate data entry, implementing robust security measures, designing a user-friendly interface, and integrating the app with different platforms and devices are key challenges. Addressing these challenges through testing, user feedback, and continuous improvement will lead to a successful and efficient Expense Tracker app.

REFERENCES

[1]. “Programming Android Java Programming for the NewGeneration of Mobile Devices” by Zigurd Mennieks.

[2]. “Android Cookbook” by Ian F Darwin.

[3]. https://www.youtube.com

[4]. https://developer.android.com

[5]. https://www.udemy.com

[6]. https://www.edureka.com

[7]. https://app.diagrams.net