THE APOLLO UNIVERSITY

Established Under Section 3 of the Andhra Pradesh Private University
(Establishment and Regulation Act ,2016)

PERSONAL EXPENSE TRACKER WITH BUDGET INSIGHTS

Android Application Development – BTCL3503

Submitted By

1.U.Jenifer(122210602102)

2. M.Yasawini(162310602101)

3. T.Sathvika(122210602101)

Submitted to

Faculty Coordinator

Dr.A.B.Manju

Assistant Professor

The Apollo University

Date Of Submission: 27/03/2025

TABLE OF CONTENTS **Page No**

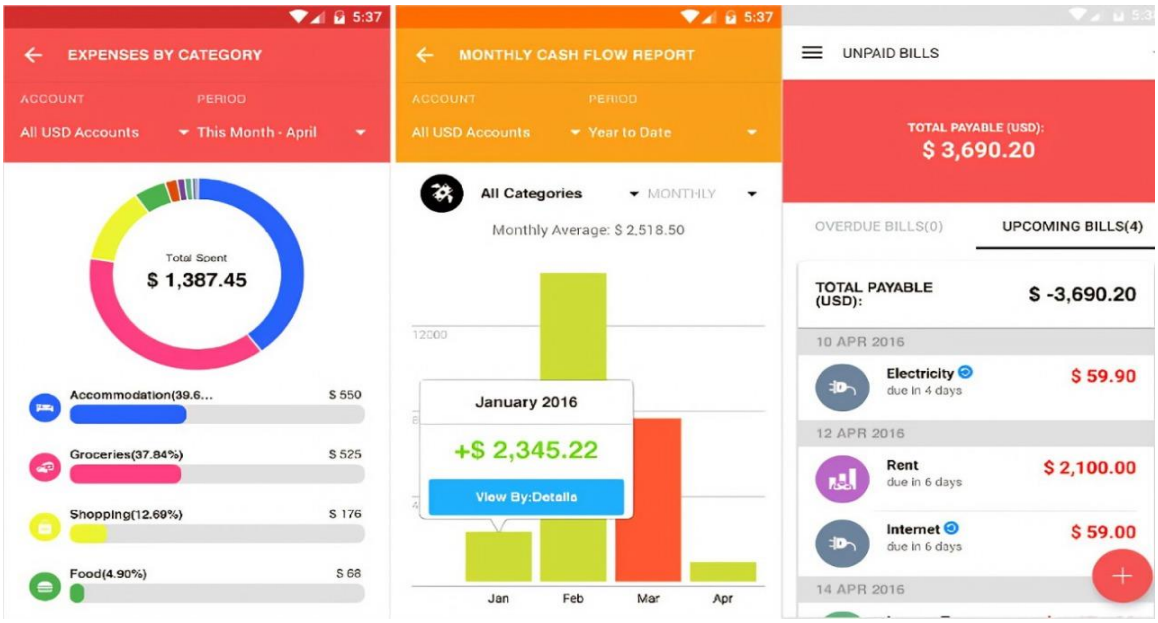
ABSTRACT

The Personal Expense Tracker with Budget Insights is an intuitive Android app designed to assist users in managing their finances effectively. It helps track daily expenses, set monthly budgets, and provides real-time alerts when users are approaching or exceeding their budget limits. The app offers essential features such as expense logging, budget management, and categorized expense analysis. Users can generate detailed reports (CSV/PDF) to review their spending patterns.

With the integration of the MPAndroidChart library, the app displays dynamic charts to visualize spending trends, enabling users to understand their financial habits better. It also leverages AI-based predictive analytics, which forecasts future spending based on historical data, helping users adjust their budgets for the upcoming months.

The app securely stores data using SQLite and Firebase, ensuring privacy and reliable access. Developed with Java/Kotlin in Android Studio, it features a user-friendly interface for all technical levels. Its goal is to reduce unnecessary spending, improve financial discipline, and promote saving.

Future versions may include cloud synchronization for cross-device access, voice-activated tracking, and multi-user support for shared budgeting, making the app more versatile and accessible for users seeking better financial control.



INTRODUCTION

Managing personal finances is a critical aspect of daily life. Many individuals struggle with keeping track of their spending habits, which often leads to overspending, financial stress, and difficulties in saving money. The rise of digital transactions has made it even harder for people to monitor their expenses effectively, as small transactions accumulate without immediate realization.

The Personal Expense Tracker with Budget Insights is designed to address this challenge by providing users with an intuitive and efficient way to record, categorize, and analyze their expenses. This Android application helps individuals gain better control over their financial activities by allowing them to set monthly budgets, receive real-time alerts, and generate expense reports for better planning.

Need for an Expense Tracker

Many people find it difficult to track their expenses manually using notebooks, spreadsheets, or traditional methods. The lack of real-time expense tracking results in poor budgeting decisions and difficulty in managing finances. A dedicated mobile application provides a convenient and automated approach to tracking spending, eliminating the need for manual calculations.

Key Features and Benefits

The Personal Expense Tracker with Budget Insights offers several key features to enhance financial discipline:

1. **Expense Logging**: Users can add daily expenses, specifying categories such as food, transportation, entertainment, and utilities.
2. **Budget Management**: The app allows users to set a monthly budget, ensuring controlled spending.
3. **Real-Time Alerts**: Notifications are triggered when spending reaches 80% of the budget and warnings are issued if the budget is exceeded.
4. **Data Visualization**: Users can view graphical charts and reports summarizing their spending patterns.
5. **Report Generation**: Expenses can be exported in CSV or PDF format for offline review.
6. **Predictive Analysis** *(Optional)*: Machine learning algorithms provide insights into future spending patterns based on past financial behavior.

Technological Advancements in Financial Management

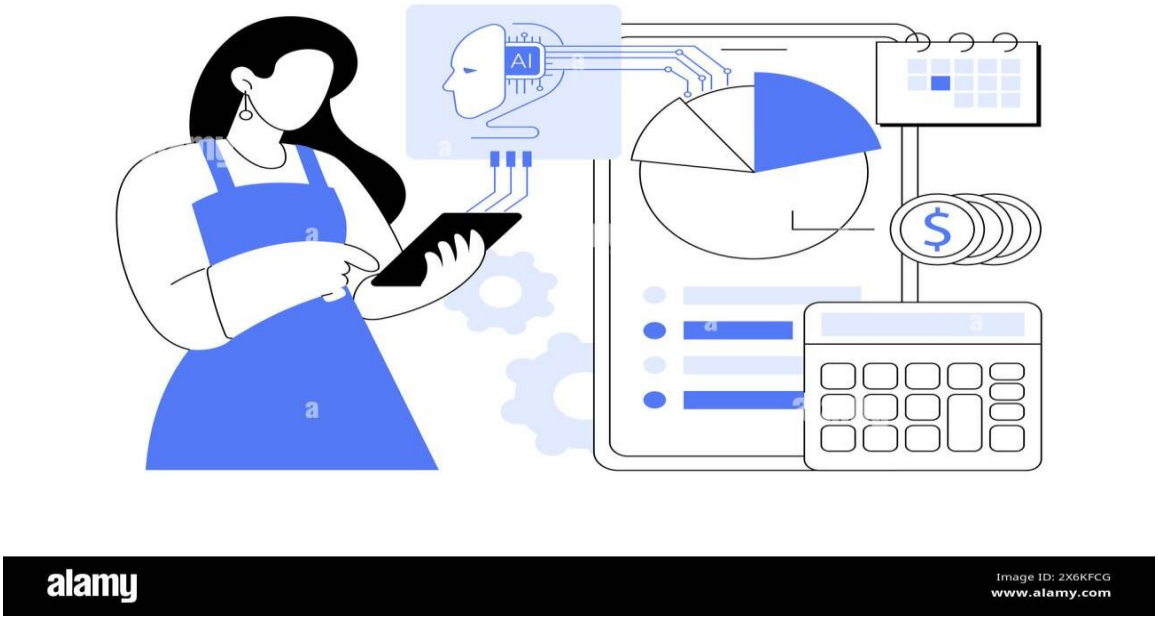
With advancements in mobile technology, cloud computing, and artificial intelligence, financial management applications have evolved significantly. The use of SQLite or Firebase ensures secure and efficient storage of financial data, while libraries like MPAndroidChart provide interactive charts for better visualization. The optional machine learning model predicts spending trends, helping users make data-driven budgeting decisions.

Objective of the Project

The primary objective of this project is to empower individuals with financial awareness and discipline by providing an easy-to-use, automated expense tracking solution. The app is designed to cater to:

- Students managing limited budgets.
- Working professionals looking for efficient financial planning.
- Families seeking better household expense management.

By integrating budget alerts, category-wise expense tracking, and predictive analytics, this Android application aims to improve financial habits, reduce unnecessary expenses, and promote better savings.



Project Statement

The absence of a structured expense management system can cause several financial challenges:

1. **Lack of Organization:** Without an organized system, users often struggle to remember where their money has been spent, making budgeting and financial planning difficult.

2. **Human Errors:** Manual calculations can lead to miscalculations, forgotten updates, or lost data, resulting in inaccurate financial records.

3. **No Instant Access:** Paper-based records or spreadsheets are not always easily accessible, delaying financial decisions and leaving users uncertain about their financial status.

4. **Limited Insights:** Traditional methods don't provide detailed analysis, making it hard for users to identify spending patterns or areas for improvement.

To overcome these challenges, the Expense Tracker app offers a digital solution that allows users to:

- **Automatically Track and Categorize Expenses:** No more manual updates or guessing where the money went.

- **Prevent Errors:** Automated calculations ensure accurate, up-to-date data.

- **Instant Access:** Access financial records anytime, anywhere, on any device.

- **Gain Insights:** Visual charts and reports provide detailed analysis to help users make informed financial decisions.

Existing Work and Disadvantages

Existing Expense Tracking Methods

1. Manual Note-Keeping – One of the most basic and traditional methods, this involves recording expenses by hand in notebooks or journals. While it can provide a visual record, it is extremely time-consuming, and prone to human errors such as miscalculations, lost records, or incomplete entries. Keeping track of expenses manually can also be cumbersome when it comes to summarizing and categorizing spending.

2. Spreadsheets (Excel, Google Sheets) – While spreadsheets are a popular tool for expense tracking, they still require manual data entry and are vulnerable to errors due to improper formulas or missed updates. Additionally, spreadsheets are often static and lack the mobility and automation of more advanced solutions, making it challenging to track finances on the go. Sharing spreadsheets or accessing them remotely can also become inconvenient without proper setup.

3. Bank Statements – Bank statements provide a retrospective view of expenses, detailing transactions made over a specific period. However, they are not real-time tools and do not give users an immediate understanding of their current financial situation. Furthermore, bank statements lack any sort of categorization or analysis, which means users must manually filter and interpret their spending patterns. This makes it harder to get actionable insights on the fly.

4. Third-Party Expense Tracker Apps – There are many apps designed to track expenses, but many of them come with their own disadvantages. They can be overly complicated, featuring a multitude of options and functionalities that overwhelm users rather than simplifying the process. Some of these apps also require paid subscriptions for full functionality, leading to extra costs for users. In addition, concerns about data privacy often arise, as some free apps collect and sell user data or require an active internet connection to function properly.

Disadvantages of Existing Systems

1. Lack of Automation – Whether using manual note-keeping or spreadsheets, these systems rely on the user to manually input and track every transaction, which can be time-consuming and prone to errors. Users must continuously update their records, which leads to a lack of real-time accuracy. With no automation, the process is inefficient, and there's a high risk of missing or incorrectly recorded transactions.

2. High Complexity in Some Apps – While third-party expense tracking apps are designed to make the process easier, many of them are loaded with too many features. This can make the app overly complex, confusing users, especially those who are looking for a simple, streamlined solution. The excess features might cause users to feel overwhelmed and fail to use the app effectively for basic expense tracking. For example, some apps may include budgeting, investment tracking, and debt management tools that go beyond what's needed for simple expense monitoring.

3. Data Privacy Concerns – Many free expense tracker apps collect a significant amount of personal data from users. This could include sensitive financial information, spending habits, and even personal identifiers. While these apps may promise to safeguard your data, the risk of data breaches or unauthorized access is still a concern. Moreover, the apps may require internet access, which can lead to users' data being stored or transmitted online without full transparency.

4. No Offline Support – Many modern apps rely heavily on internet connectivity for syncing data, updating records, and accessing features. This creates a major limitation for users who do not always have access to a stable internet connection. If users are in remote locations, traveling, or simply experiencing connectivity issues, they may not be able to access or update their expense records, which can delay financial tracking and disrupt budget planning. This lack of offline support can hinder the flexibility of such applications.

Proposed Work and Its Advantages

Proposed Solution

The Expense Tracker Android App is designed to be a lightweight, easy-to-use, and efficient solution that allows users to:

Add expenses by entering a description and amount.

View recorded expenses in an organized list.

Delete entries that are no longer needed.

Advantages of the Proposed System

- User-Friendly Design – Simple and easy-to-navigate interface.
- Real-Time Tracking – Instant updates when a new expense is added.
- Offline Accessibility – Can be used without an internet connection.
- Customization – Users can modify or delete entries easily.
- Secure Data Handling – The app does not require internet access, ensuring data privacy.
- Lightweight and Fast – The application is optimized for smooth performance.

This application serves as a practical and reliable tool for students, professionals, and households who need a simple way to manage their finances.

Implementation Details and Source Code

The Expense Tracker application is designed to be user-friendly, with a clear separation between the user interface (UI), backend logic, and database components. Below is a breakdown of the core components used in the application's current implementation, along with plans for future enhancements:

Application Components

1. **User Interface (UI) – XML Files**

The **User Interface (UI)** defines how the app looks and interacts with the user. The layout and structure of the app are managed using XML files to create a responsive, visually appealing, and intuitive design.

- **activity_main.xml** – This is the layout file for the main screen of the app, where the user can view and interact with their expenses. It defines the overall structure, such as buttons, lists, and input fields. The layout elements are arranged in a way that allows users to easily add, delete, and view expenses.
- **expense_item.xml** – This file controls the appearance of individual expense items in the list. Each item displayed in the main screen is structured and styled using this XML. It defines how the expense data (such as amount, category, and date) is presented in the list view.

- **input_box.xml –** This layout file manages the input fields for entering new expenses. It defines how the input fields, such as the amount, category, and description, are styled and arranged within the interface. The layout ensures that users can easily add expenses without cluttering the screen.

2. **Backend Logic – Java Files**

   The **backend logic** of the app is responsible for handling user interactions, performing calculations, and updating the UI. It is written in Java, which handles events such as adding, deleting, and displaying expenses.

   - **`MainActivity.java`** – This is the main activity of the app, where the core functionality takes place. It controls the logic for adding new expenses, deleting existing ones, and displaying them on the main screen. This file also defines how the app interacts with the user, managing button clicks, data input, and list updates. The logic inside `MainActivity.java` ensures that users can manage their expenses efficiently.
   - **`ExpenseAdapter.java`** – This Java class is responsible for managing the dynamic list of expenses. It acts as a bridge between the data (expenses) and the user interface, updating the displayed list whenever changes occur. The `ExpenseAdapter` class ensures that expenses are displayed in real time as users add, modify, or delete entries. It also provides efficient handling of the list by reusing views, improving performance.

3. **Database (For Future Enhancements)**

   Currently, the app uses **temporary in-memory storage** to hold expense data. While this is sufficient for the initial version of the app, it has limitations such as data loss when the app is closed or restarted. To overcome this, future versions of the app will integrate a **persistent storage solution**, ensuring that data is saved and retained across app sessions.

   - **SQLite Database –** One of the future enhancements will involve implementing **SQLite**, a lightweight and reliable relational database system. This will allow the app to store expenses in a local database, providing persistent storage even when the app is closed. Users will be able to retrieve and update their data seamlessly.
   - **Firebase Database –** Another option for future integration is **Firebase**, a cloud-based database that offers real-time syncing and data storage. With Firebase, expenses can be stored in the cloud, allowing users to access their data across multiple devices and ensuring data is securely backed up.
   - The integration of SQLite or Firebase will enhance the app's functionality by enabling persistent storage, improving data retrieval speed, and providing cloud backup, all of which will make the app more robust and scalable.

   Future Features and Enhancements

- **User Authentication**: Allow users to create accounts and log in to keep track of their expenses across multiple devices.
- **Analytics & Reporting**: Implement data analysis tools that provide insights into spending patterns and help users set budgets.
- **Custom Categories**: Let users create their own categories for expense tracking.
- **Recurring Expenses**: Allow users to add recurring expenses and set reminders.

Source Code

Activity_main.xml:--

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  android:background="#FFE0B2"
  android:gravity="center"
  android:orientation="vertical"
  android:padding="30dp"> <!-- Light Orange Background -->

  <!-- Title -->

  <TextView
    android:id="@+id/tv_title"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignParentTop="true"
    android:layout_centerHorizontal="true"
    android:paddingTop="30dp"
    android:text="Expense Tracker"
    android:textColor="#1976D2"
    android:textSize="30sp"
    android:textStyle="bold" />

  <EditText
    android:id="@+id/et_description"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:background="@drawable/input_box"
    android:hint="Enter Description"
    android:padding="12dp"
    android:textSize="16sp" />

  <EditText
    android:id="@+id/et_amount"
    android:layout_width="match_parent"
    android:layout_height="50dp"
    android:layout_marginTop="10dp"
    android:background="@drawable/input_box"
    android:hint="Enter Amount"
    android:inputType="numberDecimal"
    android:padding="12dp"
    android:textSize="16sp" />
```

```
    <!-- Add Expense Button -->
    <Button
        android:id="@+id/btn_add"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:layout_marginTop="20dp"
        android:backgroundTint="#6A1B9A"
        android:padding="10dp"
        android:text="Add Expense"
        android:textColor="#FFFFFF"
        android:textSize="16sp" />

    <!-- Table Layout for Expenses -->
    <TableLayout
        android:id="@+id/table_layout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="20dp"
        android:background="@android:color/white"
        android:stretchColumns="*" />

</LinearLayout>
```

expense_item.xml:--

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:orientation="horizontal"
    android:padding="10dp"
    android:background="@android:color/white"
    android:gravity="center_vertical">

    <!-- Expense Text -->
    <TextView
        android:id="@+id/tv_expense"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:layout_weight="1"
        android:textSize="16sp"
        android:textColor="#000000"/>

    <!-- Delete Button -->
    <Button
```

```
    android:id="@+id/btn_delete"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Delete"
    android:textSize="14sp"
    android:backgroundTint="#FF0000"
    android:textColor="#FFFFFF"/>
</LinearLayout>
```

In path app/res/drawable/input_box.xml:--

```
<?xml version="1.0" encoding="utf-8"?>
<shape xmlns:android="http://schemas.android.com/apk/res/android"
  android:shape="rectangle">

  <solid android:color="#FFFFFF"/> <!-- White background -->
  <corners android:radius="10dp"/> <!-- Rounded corners -->
  <stroke android:width="2dp" android:color="#BDBDBD"/> <!-- Gray border -->
</shape>
```

MainActivity.java:--

```
package com.example.expensetracker;

import android.graphics.Color;
import android.os.Bundle;
import android.view.Gravity;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TableLayout;
import android.widget.TableRow;
import android.widget.TextView;
import android.widget.Toast;

import androidx.appcompat.app.AppCompatActivity;

import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.Locale;

public class MainActivity extends AppCompatActivity {
  private EditText etDescription, etAmount;
  private Button btnAdd;
  private TableLayout tableLayout;

  @Override
  protected void onCreate(Bundle savedInstanceState) {
```

```
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // Initialize UI elements
    etDescription = findViewById(R.id.et_description);
    etAmount = findViewById(R.id.et_amount);
    btnAdd = findViewById(R.id.btn_add);
    tableLayout = findViewById(R.id.table_layout);

    // Add Table Header Row
    addTableHeader();

    // Add Expense Button Click Listener
    btnAdd.setOnClickListener(v -> {
        String description = etDescription.getText().toString().trim();
        String amount = etAmount.getText().toString().trim();

        if (description.isEmpty() || amount.isEmpty()) {
            Toast.makeText(MainActivity.this, "Please enter all details", Toast.LENGTH_SHORT).show();
            return;
        }

        // Get current date
        String currentDate = new SimpleDateFormat("dd/MM/yyyy", Locale.getDefault()).format(new Date());

        // Add expense to table
        addTableRow(description, amount, currentDate);

        // Clear input fields
        etDescription.setText("");
        etAmount.setText("");
    });
}

// Method to add table header
private void addTableHeader() {
    TableRow headerRow = new TableRow(this);

    String[] headers = {"Description", "Amount (₹)", "Date", "Delete"};
    for (String header : headers) {
        TextView tvHeader = new TextView(this);
        tvHeader.setText(header);
        tvHeader.setPadding(16, 8, 16, 8);
        tvHeader.setTextSize(16);
        tvHeader.setGravity(Gravity.CENTER);
        tvHeader.setTextColor(Color.BLACK);
```

```
            tvHeader.setBackgroundColor(Color.LTGRAY);
            headerRow.addView(tvHeader);
        }

        tableLayout.addView(headerRow);
    }

    // Method to add a new row
    private void addTableRow(String description, String amount, String date) {
        TableRow row = new TableRow(this);

        // Create TextViews for data
        TextView tvDescription = createTextView(description);
        TextView tvAmount = createTextView("₹" + amount); // Updated to Rupees (₹)
        TextView tvDate = createTextView(date);

        // Create Delete Button
        Button btnDelete = new Button(this);
        btnDelete.setText("Delete");
        btnDelete.setBackgroundColor(Color.RED);
        btnDelete.setTextColor(Color.WHITE);
        btnDelete.setPadding(16, 8, 16, 8);

        // Set Delete Functionality
        btnDelete.setOnClickListener(v -> tableLayout.removeView(row));

        // Add views to the row
        row.addView(tvDescription);
        row.addView(tvAmount);
        row.addView(tvDate);
        row.addView(btnDelete);

        // Add row to the table
        tableLayout.addView(row);
    }

    // Helper method to create TextViews for table
    private TextView createTextView(String text) {
        TextView textView = new TextView(this);
        textView.setText(text);
        textView.setPadding(16, 8, 16, 8);
        textView.setTextSize(16);
        textView.setGravity(Gravity.CENTER);
        return textView;
    }
}
```

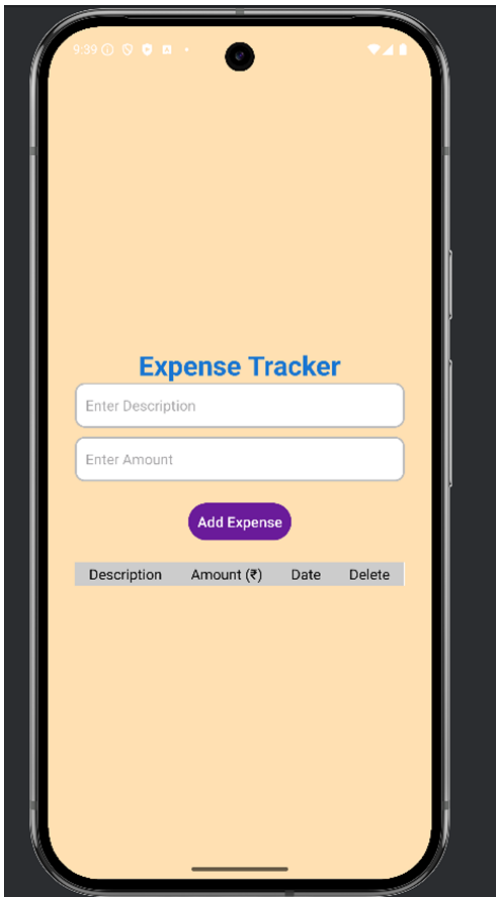
ExpenseAdaptor.java

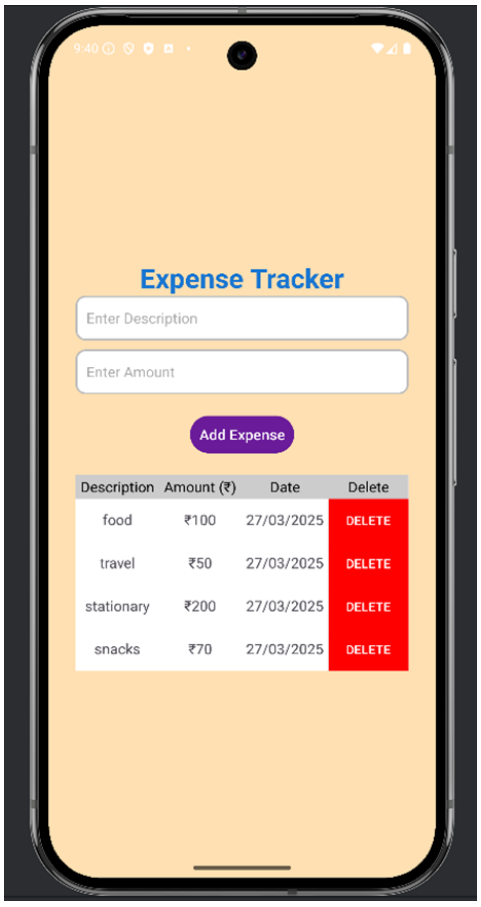
```
package com.example.expensetracker;

import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import android.widget.ArrayAdapter;
import java.util.ArrayList;
public class ExpenseAdapter extends ArrayAdapter<String> {
   private Context context;
   private ArrayList<String> expenses;

   public ExpenseAdapter(Context context, ArrayList<String> expenses) {
      super(context, R.layout.expense_item, expenses);
      this.context = context;
      this.expenses = expenses;
   }
   @NonNull
   @Override
   public View getView(int position, @Nullable View convertView, @NonNull ViewGroup parent) {
      if (convertView == null) {
         convertView = LayoutInflater.from(context).inflate(R.layout.expense_item, parent, false);return convertView;
   }
}
```

Application Output :





Contribution of Team Members

- **Sathvika** – Designed the UI, focusing on creating an intuitive, user-friendly interface and also handled **form validation** to ensure correct data input and contributed to basic app functionalities.
- **U. Jenifer** – Developed the logic for adding and deleting expenses, ensuring smooth data management and display within the app.
- **M. Yasaswini** – Worked on **performance optimization** and improved **UI responsiveness**, ensuring the app operates smoothly on various devices and screen sizes.

CONCLUSION

The Personal Expense Tracker with Budget Insights provides a practical solution for managing daily expenses and budgeting effectively. With features like real-time budget alerts, spending insights, and predictive analytics, the application helps users take control of their financial habits. Future enhancements may include cloud synchronization and multi-user functionality. The Personal Expense Tracker with Budget Insights is a powerful, user-friendly, and intelligent financial management tool. By integrating budget tracking, expense categorization, alerts, and AI-based predictions, it helps users maintain better financial discipline and avoid overspending.

FUTURE SCOPE

1. Cloud Syncing: Enable users to sync their data across multiple devices.
2. Multi-User Support: Allow family members or teams to collaborate on expense tracking.
3. AI-based Suggestions: Provide personalized financial advice based on spending habits.
4. Voice Input for Expense Entry: Enable users to add expenses using voice com

REFERENCES

- Android Development Documentation
  Official guide for building Android applications.
  Available at: https://developer.android.com/

- Firebase Documentation
  Details on Firebase Authentication, Realtime Database, and Cloud Firestore for data storage.
  Available at: https://firebase.google.com/docs/database

- SQLite Database Documentation
  Information on using SQLite for offline data storage.
  Available at: https://www.sqlite.org/docs.html

- MPAndroidChart Library
  Open-source library for creating expense visualization charts.
  Available at: https://github.com/PhilJay/MPAndroidChart

- Machine Learning for Expense Prediction
  Used for predicting future expenses with Scikit-learn models.
  Available at: https://scikit-learn.org/

- Books & Study Materials
  "Android Programming: The Big Nerd Ranch Guide" – Bill Phillips & Chris Stewart