K.S.RANGASAMY COLLEGE OF TECHNOLOGY
(Autonomous)
TIRUCHENGODE-637215

A MINI PROJECT REPORT ON

PERSONAL EXPENSE TRACKER

60 IT L04 – C# AND .NET FRAMEWORK

BACHELOR OF ENGINEERING
in
COMPUTER SCIENCE AND ENGINEERING

Submitted by

SANDHIYA R (73772214191)
SANTHIYA K (73772214196)

K.S.RANGASAMY COLLEGE OF TECHNOLOGY

(Autonomous)

TIRUCHENGODE-637215

BONAFIDE CERTIFICATE

Certified that this project report titled "**PERSONAL EXPENSE TRACKER**" is the bonafide work of **SANDHIYA R (73772214191)** and **SANTHIYA K (73772214196)** who carried out the project under my guidance.

ABSTRACT

The Personal Expense Tracker is a Windows-based application developed in C# for the user interface and MYSQL for data storage. This project aims to assist users in managing and tracking their daily expenses by recording financial transactions, categorizing them into different spending types, and generating comprehensive monthly financial reports. By allowing users to monitor income and expenses efficiently, the application helps in understanding spending patterns, setting budgets, and achieving financial goals.

The app supports a range of essential features such as adding, editing, and deleting transactions, viewing detailed transaction lists, and generating visual reports through bar and pie charts that highlight spending distribution across various categories. It offers customizable categories to accommodate unique user needs, enhancing the user's ability to organize their finances according to personal preferences. With built-in security measures, users' data remains protected while ensuring a user-friendly, responsive interface that delivers a seamless experience. This tool is designed to simplify financial tracking, reduce manual errors, and empower users with insights to make informed financial decisions.

TABLE OF CONTENT

CHAPTER 1
INTRODUCTION

1.1 OVERVIEW

The Managing personal finances is crucial in today’s world, where financial discipline ensures a secure and stress-free life. However, tracking daily expenses manually is time-consuming, error-prone, and can lead to a lack of accurate insights into spending habits. Many individuals struggle with budgeting due to inefficient record-keeping methods or the inability to visualize their financial status effectively. The Personal Expense Tracker addresses these challenges by offering a simple and efficient way to record transactions, categorize them, and analyze spending patterns.

With this application, users can gain a clear overview of their finances through detailed reports and visual graphs, helping them make informed decisions about budgeting and savings. The tool supports not only basic functionalities, such as adding and editing transaction entries, but also advanced features like categorizing expenses, generating monthly summaries, and presenting visualizations like bar and pie charts. The seamless integration of these elements enhances users' ability to identify areas where they can cut back on spending, track their progress toward financial goals, and ultimately achieve better financial stability and control.

The Personal Expense Tracker is designed with an intuitive user interface that caters to both tech-savvy and non-technical users, ensuring accessibility and ease of use. Whether used for personal budgeting or as a supplementary tool for financial planning, this application serves as a valuable resource for anyone looking to maintain a disciplined approach to their finances and achieve long-term financial wellness.

CHAPTER 2

REQUIREMENTS ANALYSIS

2.1 Functional Requirements

- Add income and expense entries.
- Edit or delete existing transactions.
- Categorize expenses (e.g., Food, Travel, Bills).
- Generate monthly reports displaying categorized expenses.
- Display graphs (bar or pie charts) to visualize spending trends.

2.2 Non - Functional Requirements

- Responsive and intuitive user interface.
- Efficient data handling for fast loading and processing.
- Secure data storage to protect user information.

CHAPTER 3

SYSTEM DESIGN

Class Diagram

Classes:

1. **User:** Contains user details (optional, for multi-user systems).
   - Attributes: UserID, Name, Email
   - Methods: Register(), Login()

2. **Transaction:** Manages income and expense entries.
   - Attributes: TransactionID, Date, Amount, Category, Type (Income/Expense), Description
   - Methods: AddTransaction(), EditTransaction(), DeleteTransaction(), ViewTransactions()

3. **Report:** Generates financial reports.
   - Attributes: ReportID, Month, Year, TotalIncome, TotalExpense, ExpenseBreakdown
   - Methods: GenerateMonthlyReport(), DisplayGraph()

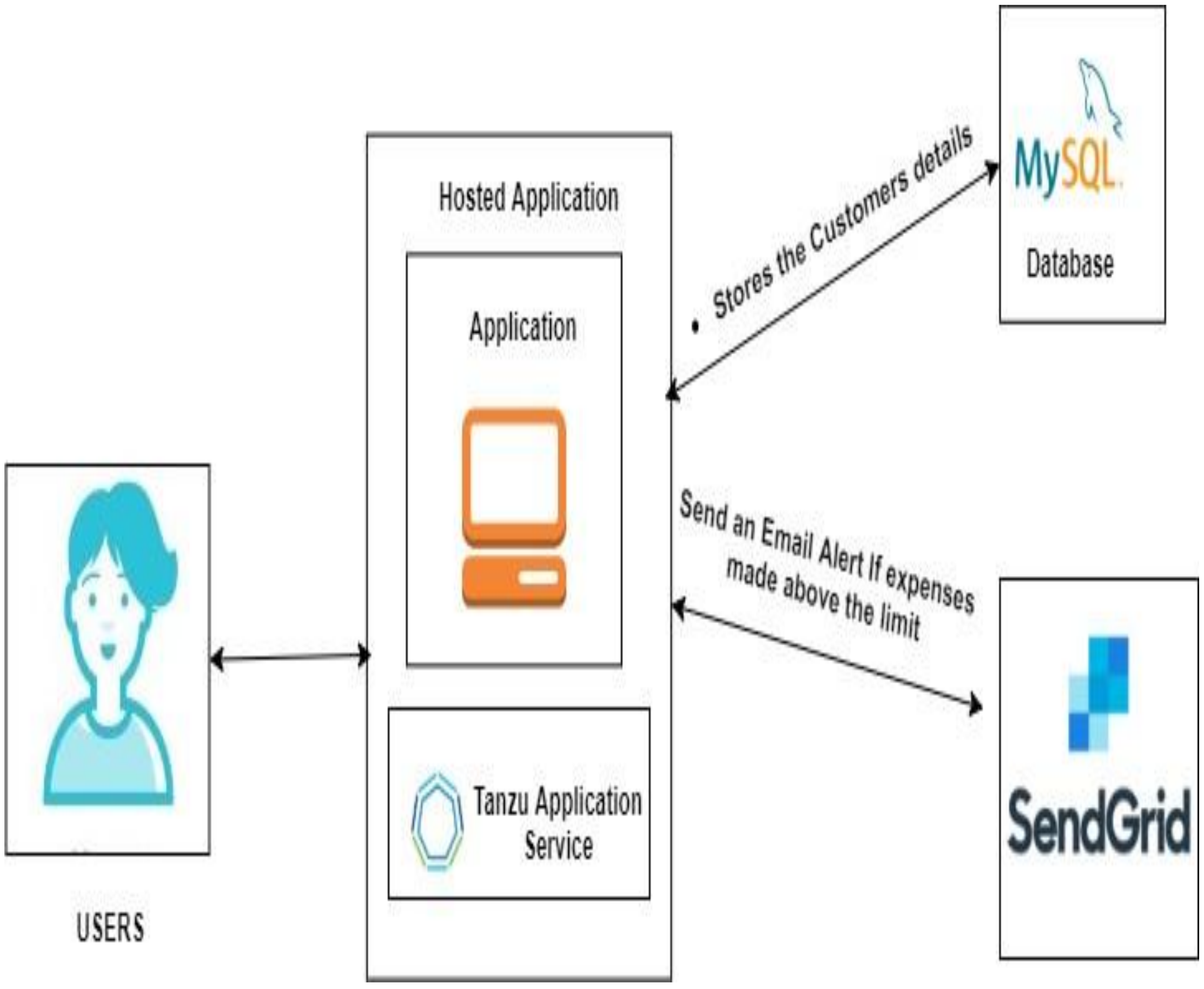
Database Schema

Tables:

1. **Users**
   - UserID (Primary Key)
   - Name
   - Email
   - Password

2. **Transactions**
   - TransactionID (Primary Key)
   - UserID (Foreign Key)
   - Date
   - Amount
   - Category
   - Type
   - Description

3. **Reports**
    - ReportID (Primary Key)
    - UserID (Foreign Key)
    - Month
    - Year
    - TotalIncome
    - TotalExpense
    - ExpenseBreakdown



Hosted Application
Application
Tanzu Application Service
USERS
• Stores the Customers details
MySQL
Database
Send an Email Alert If expenses made above the limit
SendGrid

CHAPTER 4

IMPLEMENTATION

Main Features

1. **Adding Transactions:** Users can add income or expense entries by specifying the amount, category, and date. An optional description field allows adding notes for transactions.
2. **Categorizing Expenses:** Transactions are categorized into predefined groups like Food, Travel, Bills, or custom categories specified by the user.
3. **Reports and Graphs:** Monthly reports are generated, summarizing total income and expenses. Bar and pie charts visually represent the distribution of spending across categories.

Technologies Used

1. **Programming Language:** C#
2. **Framework:** WPF for user interface design.
3. **Database:** MYSQL for advanced implementations.
4. **Libraries:**
    - OxyPlot or LiveCharts for creating graphs.
    - Entity Framework for database interaction.

CODE

Adding a Transaction

```
using System;
using System.Data.SQLite;

class ExpenseTracker
{
    private string connectionString = "Data Source=ExpenseTracker.db;Version=3;";

    public void AddTransaction(string date, double amount, string category, string type, string description)
    {
        using (SQLiteConnection connection = new SQLiteConnection(connectionString))
        {
            connection.Open();
            string query = "INSERT INTO Transactions (Date, Amount, Category, Type, Description)
VALUES (@Date, @Amount, @Category, @Type, @Description)";
            using (SQLiteCommand command = new SQLiteCommand(query, connection))
            {
                command.Parameters.AddWithValue("@Date", date);
                command.Parameters.AddWithValue("@Amount", amount);
                command.Parameters.AddWithValue("@Category", category);
                command.Parameters.AddWithValue("@Type", type);
                command.Parameters.AddWithValue("@Description", description);
                command.ExecuteNonQuery();
            }
        }
    }
}
```

Generating a Report

```
public void GenerateReport(string month, string year)
{
    using (SQLiteConnection connection = new SQLiteConnection(connectionString))
    {
        connection.Open();
        string query = "SELECT Category, SUM(Amount) as Total FROM Transactions WHERE
strftime('%m', Date) = @Month AND strftime('%Y', Date) = @Year GROUP BY Category";
        using (SQLiteCommand command = new SQLiteCommand(query, connection))
        {
            command.Parameters.AddWithValue("@Month", month);
            command.Parameters.AddWithValue("@Year", year);
            using (SQLiteDataReader reader = command.ExecuteReader())
            {
                while (reader.Read())
                {
                    Console.WriteLine($"Category: {reader["Category"]}, Total: {reader["Total"]}");
                }
            }
        }
    }
}
```

CHAPTER 5
OUTPUT

TEST CASES

1. Add Transaction Test
    - Input: Date, Amount, Category, Type, Description.
    - Expected Output: Transaction added successfully.
2. Delete Transaction Test
    - Input: TransactionID.
    - Expected Output: Transaction removed from the database.
3. Generate Report Test
    - Input: Month and Year.
    - Expected Output: Display of monthly income, expenses, and graphs.
4. Category Filtering Test
    - Input: Select category.
    - Expected Output: Display transactions specific to the selected category.

OUTPUT

Adding a Transaction:

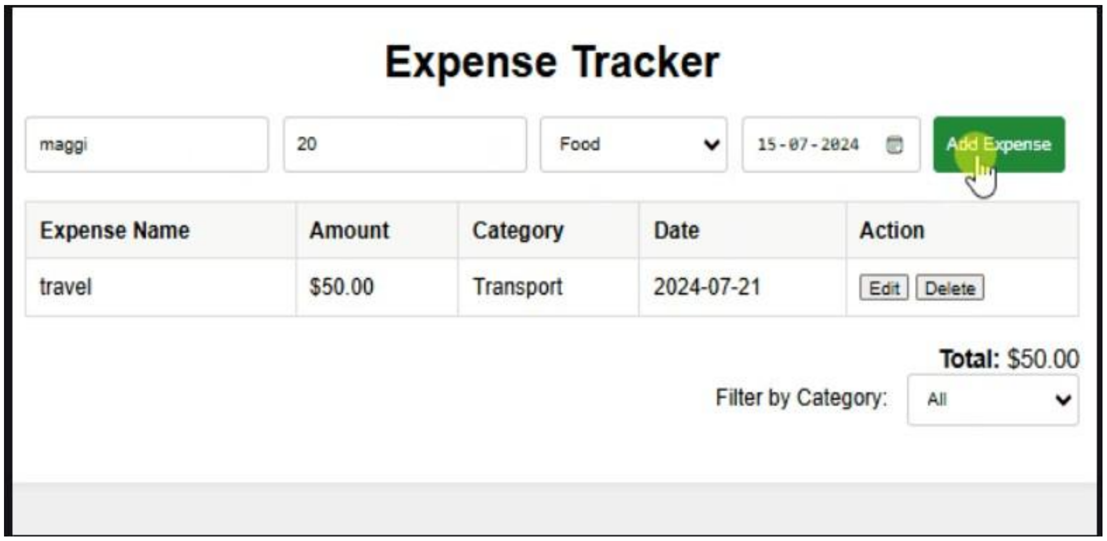
Transaction Added Successfully.

Generating a Report:

Category: Food, Total: 500

Category: Travel, Total: 300

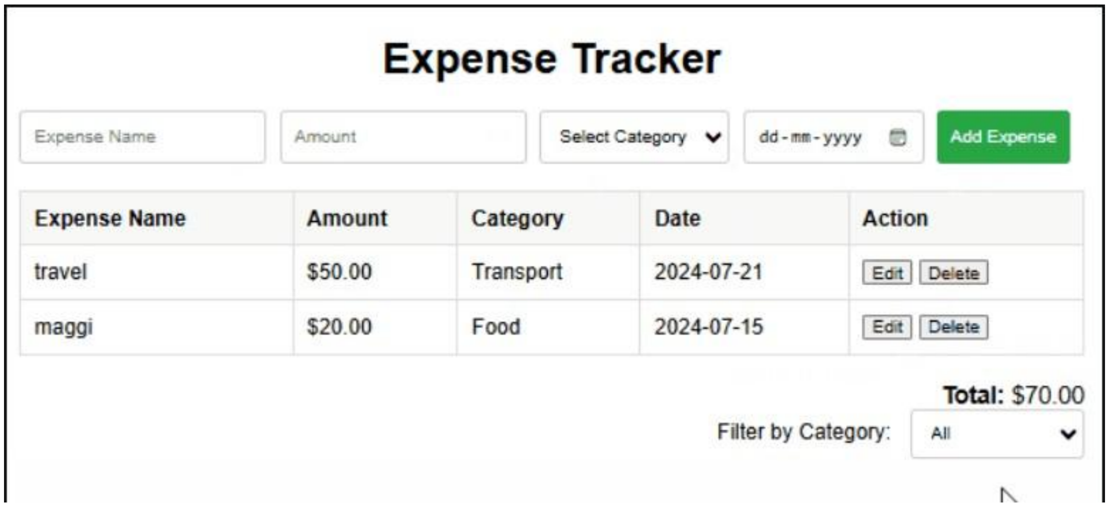
Category: Bills, Total: 200



Expense Tracker

Expense Name	Amount	Category	Date	Action
travel	$50.00	Transport	2024-07-21	Edit Delete

Total: $50.00



Expense Tracker

Expense Name	Amount	Category	Date	Action
travel	$50.00	Transport	2024-07-21	Edit Delete
maggi	$20.00	Food	2024-07-15	Edit Delete

Total: $70.00

CHAPTER 6
CONCLUSION

In Conclusion, The Personal Expense Tracker offers a practical and comprehensive solution for managing daily finances, empowering users to take control of their spending habits and make informed decisions about their financial future. By combining an intuitive user interface with robust features such as transaction recording, categorization, and report generation, the application provides users with the tools needed to monitor their income and expenses effectively. The inclusion of graphical reports, like bar and pie charts, simplifies the understanding of spending trends, making financial management more accessible and engaging for users of all backgrounds. This project not only addresses the inefficiencies of manual expense tracking but also encourages better financial discipline, enabling users to achieve their budgeting and savings goals more effectively.

CHAPTER 7

REFERENCES

[1] J. Albahari, B. Albahari. *C# 10 in a Nutshell: The Definitive Reference*. O'Reilly Media, 2022. ISBN: 978-1098121990.

[2] A. Troelsen, P. Japikse. *Pro C# 10 with .NET 6: Foundational Principles and Practices in Programming*. Apress, 2022. https://doi.org/10.1007/978-1-4842-7983-4.

[3] J. Skeet. *C# in Depth*. Manning Publications, 2019. ISBN: 978-1617294532.

[4] R. Freeman. *Essential .NET Core and ASP.NET Core*. Addison-Wesley, 2021. ISBN: 978-0135972266.

[5] Microsoft. "Introduction to C# and .NET." Microsoft Learn, 2023. https://learn.microsoft.com/en-us/dotnet/csharp/