EXPENSE TRACKING SYSTEM

PF SEMESTER PROJECT

Course Title: Programming Fundamentals

Course Instructor: Mansoor Ali

Semester/Batch: 1st Semester, Batch: Fall 2024,

Computer Science Department

University: Salim Habib University

Submission Date: 28th January 2025

Group Members:

- Daniyal Ali (F24CSC019)
- Ayesha Zafar (F24CSC015)
- Syed Abdul Muiz (F24CSC001)

INTRODUCTION

➢ OVERVIEW

The Expense Tracking System is a user-friendly application designed to help individuals manage their personal finances efficiently. Built in C++, this system provides robust features for registering users, logging expenses, and analyzing financial data. It is an ideal solution for users seeking a simple yet effective tool to track and manage their spending habits.

➢ PURPOSE

This project was chosen to demonstrate and apply the core principles of programming in a practical and meaningful way. The **Expense Tracking System** addresses a common real-world problem—managing personal finances—while integrating essential programming concepts, making it both relevant and valuable as a learning experience.

➢ OBJECTIVES

1. **To Apply Fundamental Programming Concepts**
    - Gain hands-on experience with basic programming constructs such as loops, functions, conditionals, and enumerations.

    - Understand and implement file handling techniques for data storage and retrieval.
2. **To Develop a Practical, User-Friendly Application**
    - Create a simple yet functional program that allows users to efficiently manage their personal expenses.
    - Ensure a clear and intuitive menu-driven interface for seamless user interaction.
3. **To Foster Problem-Solving and Modular Design Skills**
    - Break down a real-world problem into manageable components, such as expense addition, deletion, and reporting.
    - Design modular functions to encourage code reusability, maintainability, and scalability.
4. **To Enhance Error Handling and Input Validation Skills**

- Implement robust error-handling mechanisms to manage invalid inputs and unexpected user actions.
- Ensure the program gracefully handles edge cases, such as missing files or incorrect data formats.

5. **To Practice Persistent Data Management**

- Learn how to use file input/output operations to store user data and expenses persistently.
- Enable users to access their saved data across multiple sessions, ensuring a reliable experience.

6. **To Organize and Categorize Data Efficiently**

- Allow users to categorize their expenses (e.g., food, transport, utilities) for better financial insights.
- Provide structured and formatted output for easy analysis of expenses.

7. **To Build Real-World Problem-Solving Skills**

- Develop a solution to a practical and relatable problem that individuals face in everyday life.
- Focus on creating a project that demonstrates the relevance of programming in solving real-world challenges.

➢ TOOLS AND TECHNOLOGIES

1. **Programming Language**:
    - The project is developed using **C++**, a versatile and widely-used programming language known for its efficiency and performance.
2. **Integrated Development Environment (IDE)**:
    - The project is coded and tested using **Code::Blocks** or **Dev-C++**, popular IDEs for C++ development, ensuring a smooth coding and debugging experience.
3. **Libraries Used**:
    - **<iostream>**: For input and output operations.
    - **<fstream>**: For file handling to store and retrieve data persistently.
    - **<string>**: For managing user input and processing text data.
    - **<iomanip>**: For formatted output to enhance readability.

- **<stdexcept>**: For implementing error handling with exceptions.
- **<limits>**: To handle input validation effectively.

4. **Operating System**:
    - The program can be developed and executed on any operating system, including **Windows**, **Linux**, or **macOS**, as long as a C++ compiler is available.
5. **Compiler**:
    - The project uses the **GNU Compiler Collection (GCC)** or **MinGW**, ensuring compatibility with standard C++ features.

These tools and technologies were chosen to provide an efficient and flexible environment for developing a console-based application that demonstrates core programming skills.

6. **Version Control System**:
    - **Git**): For tracking changes and maintaining version control during the development process.
    - **GitHub**: Used for storing and sharing the project code in an online repository.

➢ PSEUDOCODE

STARTING MENU

- Display the main menu:
  - Register
  - Login
  - Helpline
  - Exit
- Take user input and navigate to the selected option.

---

2. **Registration Process**
   - Prompt the user for their **username** and **password**.
   - Check if the username already exists.
     - If it exists, display: "User already exists. Please login."
     - Otherwise, create a file for the user and save credentials.
   - Confirm registration.

---

3. **Login Process**
   - Prompt the user for their **username** and **password**.
   - Validate the credentials from the stored file.
     - If valid, navigate to the **User Dashboard**.
     - If invalid, display: "Incorrect credentials. Try again."

---

4. **Add Expense**
    - Prompt the user to enter:
        - **Description**: A brief text for the expense (e.g., "Lunch").
        - **Amount**: The cost of the expense.
        - **Category**:
            - Display category options:
                1. Food
                2. Transport
                3. Utilities
                4. Entertainment
                5. Other
            - Take user input for category selection.
    - Validate inputs (e.g., amount > 0, valid category).
    - Save the expense details (description, amount, category) into the user's expense file.
    - Confirm that the expense has been added.

---

5. **View Expenses**
    - Open the user's expense file.
    - Display expenses in a **table format** with the following columns:
        - **Description**
        - **Category**
        - **Amount**
    - If no expenses are found, display: "No expenses recorded."

---

6. **Delete Expense**
    - Prompt the user to enter the **description** of the expense to delete.
    - Open the user's expense file and search for the matching description.
    - If found, delete the expense from the file.
        - Confirm: "Expense deleted successfully."
    - If not found, display: "Expense not found."

---

7. **Total Expenses**
    - Open the user's expense file.
    - Sum the **amount** of all expenses.
    - (Optional) Show a **category-wise breakdown** (e.g., total for Food, Transport, etc.).
    - Display the total amount.

---

8. **Logout**
    - Return to the main menu.

---

9. **Exit Program**
    - Terminate the program with a goodbye message.

➢ IMPLEMENTATION

The **Expense Tracking System** is a console-based application designed to enable users to manage and track their daily expenses effectively. The system allows users to register and log in, add and view expenses, delete an expense, and calculate the total sum of their expenses. The expenses are categorized, and each expense is saved in a file for persistence, which means the data will remain even if the program is closed.:

1. **Starting Menu**:

   The program begins by displaying a main menu with options for the user. The choices include registering as a new user, logging in, viewing help, or exiting the program. If the user opts for registration, they will be prompted to provide their credentials (username and password). If the user opts for login, they will be asked to enter their existing credentials. The system also allows users to view help contact details, and it will exit if selected.

2. **User Registration**:

   When a user chooses to register, the system asks for their username and password. The program checks if a file with the same name already exists. If not, a new file is created with the user's credentials. This allows each user to have a unique file where their login credentials and expenses are stored.

3. **User Login**:

   For login, the user must provide a username and password. The system checks whether the entered credentials match what is stored in the corresponding file. If successful, the user is given access to a dashboard that allows them to perform tasks such as adding, viewing, deleting, or calculating expenses.

4. **Expense Management**:
   - **Add Expense**:

     The user can add an expense by providing a description, amount, and selecting a category (e.g., Food, Transport, Utilities). This information is then saved to the user's expense file for future reference.
   - **View Expenses**:

     The user can choose to view their saved expenses. The system reads from the user’s expense file and displays the details in a tabular format, showing the description, amount, and category of each recorded expense.
   - **Delete Expense**:

     If the user wants to delete an expense, they simply enter the description of the expense they wish to remove, and the system updates the expense file by deleting that particular record.
   - **Total Expenses**:

     The user can request the total sum of all their expenses. The system calculates the sum of all recorded amounts and displays it for the user’s reference.

5. **Category Management**:

   Each expense is categorized, making it easier for users to track where their money is going. The categories include Food, Transport, Utilities, Entertainment, and Other. The user is prompted to select a category when adding an expense.

6. **File Handling**:

   User data (e.g., login credentials) and expenses are stored in text files. Each user has a file containing their credentials (e.g., username.txt), and their expenses are saved in a separate file (e.g., username_expenses.txt). These files are read and written using file input/output streams (ifstream and ofstream), ensuring that data is saved persistently between program sessions.

➢ TESTING

INPUT	EXPECTED OUTPUT	ACTUAL OUTPUT
Register User: "Daniyal", Password: "1234"	User Registered Successfully	Success
Login: "Daniyal", Password: "1234"	Login Successful	Success
Add Expense: "Food", Amount: "500", Category: FOOD	Expense Added Successfully	Success
View Expenses	List of Expenses	Displayed Correctly
TOTAL Expenses	Sum of Expenses	Correct Total

➢ RESULTS

- Successfully allows users to **register** and **log in** with unique usernames.
- Provides a user-friendly **main menu** with clear options for navigation.
- Allows adding expenses with **categories** (e.g., Food, Transport).
- Facilitates **viewing expenses** in a well-formatted table.
- Enables users to **delete specific expenses** by description.
- Computes and displays **total expenses** for the logged-in user.
- Includes robust **error handling** for invalid inputs and file operations.
- Displays **helpline contact details** for user assistance.

➢ CONCLUSION

This project provided valuable hands-on experience in advanced C++ programming and deepened our understanding of key concepts:

- **Modular Programming**: We learned the importance of breaking the program into smaller, manageable functions. This improved code readability, maintainability, and allowed easier debugging and future extensions.
- **File Handling for Data Storage**: We implemented file handling to store user data and expenses persistently, simulating real-world scenarios where data persists between program runs.

- **Debugging and Testing**: We honed our debugging skills by using error messages and exception handling. Rigorous testing ensured the program worked correctly, including adding, deleting, and totaling expenses.

In summary, this project enhanced our C++ skills, especially in modularity, file handling, and debugging, providing us with a solid foundation for future programming tasks.