Project Report: Expense Tracker Application in C++

1. Introduction

The Expense Tracker application is designed to help users manage their daily expenses efficiently. This application allows users to record expenses under various categories, view stored expenses, and generate reports to monitor spending patterns.

2. Objective

The primary objective of this project is to create a user-friendly, console-based expense tracker in C++. The tracker should:

- Record expenses with details like date, category, and amount.
- Display a summary of all recorded expenses.
- Generate expense reports to aid budgeting and financial management.

3. System Requirements

Software:

- Operating System: Windows/Linux/Mac
- C++ Compiler (e.g., GCC, MinGW)

Libraries Used:

- <iostream> for console I/O
- <fstream> for file handling
- <vector> for managing dynamic expense records
- < iomanip> for formatted output

4. Project Design

4.1 Architecture

The project uses a modular design with two main classes:

- Expense: Represents each expense record with properties like date, category, and amount.
- Expense Tracker: Manages the collection of expenses, including adding, viewing, and generating reports.

4.2 Class Diagram

Classes:

Expense

- Attributes: date, category, amount.
- Methods: getDate(), getCategory(), getAmount(), displayExpense().

Expense Tracker

- Attributes: expenses (vector of Expense objects).
- Methods: addExpense(), viewExpenses(), generateReport(), saveToFile(), loadFromFile().

5. Implementation

5.1 Core Functionalities

1. Adding Expenses: Users input date, category, and amount, which are stored in an Expense object and added to the expenses vector.
2. Viewing Expenses: The viewExpenses() function lists all expenses with details.
3. Generating Reports: The generateReport() function calculates and displays the total expenditure.

5.2 Code Snippet

Example code for the addExpense() function:

```
void ExpenseTracker::addExpense() {

string date, category;

double amount;

cout << "Enter date (DD-MM-YYYY): ";
```

```
cin >> date;

cout << "Enter category (Food, Transport, etc.): ";

cin >> category;

cout << "Enter amount: ";

cin >> amount; expenses.emplace_back(date, category, amount); saveToFile();

cout << "Expense added successfully!\n";

}
```

6. Testing and Validation

The Expense Tracker was tested for:

- Correctness: Verifying that all functionalities (adding, viewing, and reporting) work as expected.
- File Persistence: Ensuring that expenses are saved to and loaded from a file correctly.
- Edge Cases: Handling invalid inputs (e.g., negative amounts) and large data sets.

7. Challenges Faced

Some challenges encountered during development:

- File Handling: Ensuring data persistence and managing file I/O operations.
- Dynamic Memory Management: Efficiently managing memory for a dynamically growing list of expenses.
- User Input Validation: Validating inputs to prevent incorrect data entries.

8. Future Enhancements

Potential future improvements include:

1. Graphical User Interface (GUI): Transitioning from console-based to a GUI for better user experience.

2. Database Integration: Using a database instead of file storage for scalability and faster data retrieval.
3. Enhanced Reporting: Adding detailed reports with categories and trends, along with graphical summaries.

9. Conclusion

The Expense Tracker project successfully meets the objective of managing and categorizing personal expenses through a simple, console-based C++ application. This project enhanced our understanding of C++ concepts such as file handling, object-oriented programming, and vector operations, while demonstrating practical applications of programming in financial management.