PROJECT REPORT

Project Title:
STUDENT FINANCE MANAGEMENT SYSTEM

Course: B.Tech – Computer Science Engineering
Subject: C Programming Major Project
Submitted By: Akshit Chibber

SAP ID: 590022153
Submitted To: Dr. Tanu Singh, SOCS

Institution: University of Petroleum and Energy Studies

Submission Date: 30-11-25

1. ABSTRACT

Managing money wisely is one of the biggest challenges for students today. Most of us receive pocket money, stipends, or part-time earnings, but due to a lack of proper planning, we often end up overspending without realizing where the money went. To solve this problem, I have developed a **Student Finance Management System** in C language.

This project allows a student to create an account, track income, record expenses, and even allocate a percentage of their balance towards savings. The data is saved permanently using file storage, so even after closing the program, the student's financial details remain intact. The system is fully menu-driven and easy to use, making it a practical tool for students to learn basic financial discipline.

---

2. PROBLEM DEFINITION

Many students struggle with money management. They forget how much they spent, how much they saved, or how much balance remains. Without a proper system, it becomes difficult to build good spending habits.

The problem can be summarized as follows:

- Students have no organized place to track finances
- Manual tracking leads to errors
- There is no habit of saving
- Financial awareness is low at a young age

Objective of this project:

To design a simple and user-friendly system that helps students:

- Create an account
- Log in securely
- Add income and expenses
- Automatically calculate savings based on a chosen percentage
- View updated balance anytime
- Store all details even after program exit

---

3. SYSTEM DESIGN

3.1 How the System Works

The program begins with a home screen where a user can sign up or log in. Once logged in, the student enters the main dashboard, where they can perform finance-related actions. All operations immediately update the stored values.

The system is divided into multiple C files such as `auth.c`, `finance.c`, `utils.c`, etc., which makes the code modular and easy to understand.

3.2 Flowchart

```
START
  |
  |--> Home Page (1 Login / 2 Signup / 0 Exit)
  |
  |--> If Signup → Create Account → Save Details
  |
  |--> If Login → Verify Credentials
  |          |
  |          v
  |     Finance Dashboard
  |     1 Income
  |     2 Expense
  |     3 Savings %
  |     0 Logout
  |
  |--> Save Data → Exit Program
  |
END
```

3.3 System Components

Component	Description
Login / Signup	Handles user accounts
Finance Module	Adds income, expense, savings
Storage Module	Saves and loads the user data
Helper Module	Input validation and utility functions

---

4. IMPLEMENTATION DETAILS

The project is written entirely in **C language** using modular programming. Each feature is placed in a separate `.c` file with a corresponding `.h` file.

Data Structure Used

```
typedef struct {
    char username[50];
    int password;
    float totalBalance;
    float savings;
    int isActive;
} User;
```

Savings Logic (Human Explanation)

If a user has ₹2000 and wants to save 10%, the system:

- Takes 10% of 2000 = 200
- Moves that amount to **Savings**
- Reduces balance to 1800
- Stores the updated values in the database

This encourages real-life habit of saving money regularly.

---

5. TESTING & RESULTS

Several test cases were executed to verify the working of the system:

Action Tested	Result
Signup	Successfully creates new user
Login	Works only with correct username and password
Income	Adds amount correctly to balance
Expense	Deducts amount and prevents negative balance
Savings	Correctly calculates and transfers percentage
Restart Program	Loads data correctly from file

The program worked as expected for all test cases, proving reliability.

---

6. CONCLUSION & FUTURE WORK

This project helped me understand the importance of modular programming, structures, and file handling in C. It also showed how programming can solve real-life problems—something as common as managing student pocket money.

Future improvements may include:

- Encrypting passwords for better security
- Adding monthly reports and charts
- Developing a mobile or GUI version
- Adding notifications and budgeting tips

---

7. REFERENCES

- *The C Programming Language* by Dennis Ritchie
- Classroom notes and lab guidance
- Online resources and C documentation
- StackOverflow discussions for debugging

END OF REPORT