Python Personal Finance Manager (Internship-Level Project)

📌Project Overview

A complete **Personal Finance Management System** built using **Python**, designed with **Object-Oriented Programming**, **CSV-based data persistence**, **modular architecture**, and a **menu-driven CLI**. This project is suitable for **internship submission**, demonstrating clean code structure, error handling, reporting, and backup functionality.

🧱Project Structure

```
finance_manager/
│
├── main.py                # Entry point
├── expense.py             # Expense class
├── file_manager.py        # File handling & backup
├── menu.py                # CLI menu system
├── reports.py             # Reports & analytics
├── utils.py               # Validation utilities
│
├── data/
│   └── expenses.csv       # Expense storage
│
├── backups/               # Backup files
└── reports/               # Generated reports
```

🧰expense.py

```
class Expense:
    def __init__(self, amount, category, date, description):
        self.amount = float(amount)
        self.category = category
        self.date = date
        self.description = description

    def to_list(self):
        return [self.date, self.category, self.amount, self.description]

    def __str__(self):
        return f"{self.date} | {self.category} | ₹{self.amount:.2f} |
{self.description}"
```

📂file_manager.py

```
import csv
import os
import shutil
from expense import Expense

DATA_FILE = "data/expenses.csv"
BACKUP_DIR = "backups"


def load_expenses():
    expenses = []
    if not os.path.exists(DATA_FILE):
        return expenses

    with open(DATA_FILE, "r") as file:
        reader = csv.reader(file)
        next(reader, None)
        for row in reader:
            expenses.append(Expense(*row))
    return expenses


def save_expenses(expenses):
    os.makedirs("data", exist_ok=True)
    with open(DATA_FILE, "w", newline="") as file:
        writer = csv.writer(file)
        writer.writerow(["Date", "Category", "Amount", "Description"])
        for exp in expenses:
            writer.writerow(exp.to_list())


def backup_data():
    os.makedirs(BACKUP_DIR, exist_ok=True)
    shutil.copy(DATA_FILE, f"{BACKUP_DIR}/expenses_backup.csv")


def restore_data():
    shutil.copy(f"{BACKUP_DIR}/expenses_backup.csv", DATA_FILE)
```

🛠️utils.py

```
from datetime import datetime
```

```
def validate_amount(amount):
    try:
        return float(amount) > 0
    except ValueError:
        return False


def validate_date(date_text):
    try:
        datetime.strptime(date_text, "%Y-%m-%d")
        return True
    except ValueError:
        return False


def pause():
    input("\nPress Enter to continue...")
```

📊 reports.py

```
from collections import defaultdict


def category_summary(expenses):
    summary = defaultdict(float)
    for exp in expenses:
        summary[exp.category] += exp.amount
    return summary


def monthly_report(expenses, month):
    return [e for e in expenses if e.date.startswith(month)]


def total_and_average(expenses):
    total = sum(e.amount for e in expenses)
    avg = total / len(expenses) if expenses else 0
    return total, avg
```

📋 menu.py

```
from expense import Expense
from utils import validate_amount, validate_date, pause
from reports import category_summary, total_and_average
```

```
def show_menu():
    print("=" * 40)
    print("   PERSONAL FINANCE MANAGER")
    print("=" * 40)
    print("1. Add Expense")
    print("2. View Expenses")
    print("3. Category Summary")
    print("4. Backup Data")
    print("5. Exit")


def add_expense(expenses):
    amount = input("Amount: ")
    if not validate_amount(amount):
        print("❌ Invalid amount")
        return

    category = input("Category: ")
    date = input("Date (YYYY-MM-DD): ")
    if not validate_date(date):
        print("❌ Invalid date")
        return

    desc = input("Description: ")
    expenses.append(Expense(amount, category, date, desc))
    print("✅ Expense added")
```

📩main.py

```
from file_manager import load_expenses, save_expenses, backup_data
from menu import show_menu, add_expense
from reports import category_summary, total_and_average
from utils import pause


def main():
    expenses = load_expenses()

    while True:
        show_menu()
        choice = input("Enter choice: ")

        if choice == "1":
            add_expense(expenses)
            save_expenses(expenses)
            pause()
```

```
        elif choice == "2":
            for e in expenses:
                print(e)
            pause()

        elif choice == "3":
            summary = category_summary(expenses)
            for cat, amt in summary.items():
                print(f"{cat}: ₹{amt:.2f}")
            total, avg = total_and_average(expenses)
            print(f"Total: ₹{total:.2f}, Average: ₹{avg:.2f}")
            pause()

        elif choice == "4":
            backup_data()
            print("☢️ Backup completed")
            pause()

        elif choice == "5":
            save_expenses(expenses)
            print("Goodbye ⬇️")
            break

        else:
            print("❌ Invalid choice")


if __name__ == "__main__":
    main()
```

🕐Internship Evaluation Highlights

- ✅OOP & Modular Design
- ✅File Handling with CSV
- ✅Input Validation & Error Handling
- ✅Reports & Analytics
- ✅Backup & Restore System
- ✅Industry-level folder structure

📌Future Enhancements (Optional)

- PDF/Excel report export
- GUI using Tkinter
- Charts using Matplotlib
- Login & authentication

⬜ **Perfect for:** Python Internship • College Mini Project • Resume/GitHub Portfolio