HBMS

HOUSEHOLD BUDGET MANAGEMENT SYSTEM

21CSS101J – PROGRAMMING FOR PROBLEM-SOLVING

Mini Project Report

Submitted by

PRANJAL MISHRA [RA2311047010020]
ADHIRAJ CHOUDHURY [RA2311056010280]

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be **University** *u/s 3 of UGC Act, 1956)*

SCHOOL OF COMPUTING
COLLEGE OF ENGINEERING AND TECHNOLOGY
SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

November 2023

SRM
INSTITUTE OF SCIENCE & TECHNOLOGY
(Deemed to be University u/s 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that Mini project report titled _HOUSEHOLD BUDGET MANAGEMENT SYSTEM is the bonafide work of Reg.No RA23110560102 of CSE DATA SCIENCE_and Reg.No RA2311047010020__ of BTECH AI_who carried out the minor project under my supervision. Certified further, that to the best of my knowledge, the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE	**SIGNATURE**
Prof. J. JAYASUDHA	**HEAD / CINTEL**

TABLE OF CONTENTS

CHAPTER 1
PROBLEM STATEMENT OF HBMS

Problem Statement:

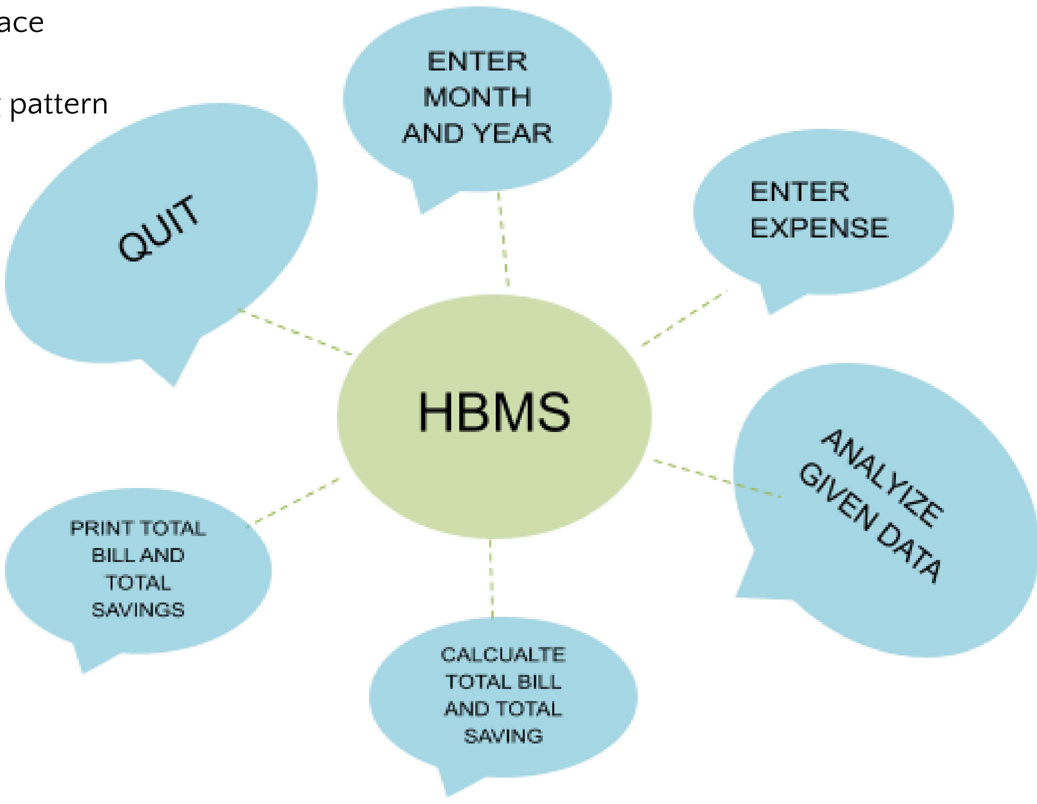
The problem statement for a household budget management system involves the need for an efficient and user friendly tool that helps individuals or families track , manage and optimize their finances. This system should address challenges such as expense

tracking , income management and provide insights to promote better financial decision
making. It should be accessible, secure and capable of generating reports to record spending patterns and ultimately support users in achieving their financial goals.

Problem statement description:

Create a simple console-based household budget management system that provide a convenient way for users to monitor their financial activities. The system should support the following features:

1. Tracking income
2. Expenses
3. Analyze spending pattern
4. Record various Expenses
5. Viewing Summary
6. Menu-driven interface
7. Data storage
8. Analyzing spending pattern
9. Graceful exit


ENTER MONTH AND YEAR
ENTER EXPENSE
QUIT
HBMS
ANALYIZE GIVEN DATA
PRINT TOTAL BILL AND TOTAL SAVINGS
CALCUALTE TOTAL BILL AND TOTAL SAVING

CHAPTER 2
METHODOLOGY OF HBMS

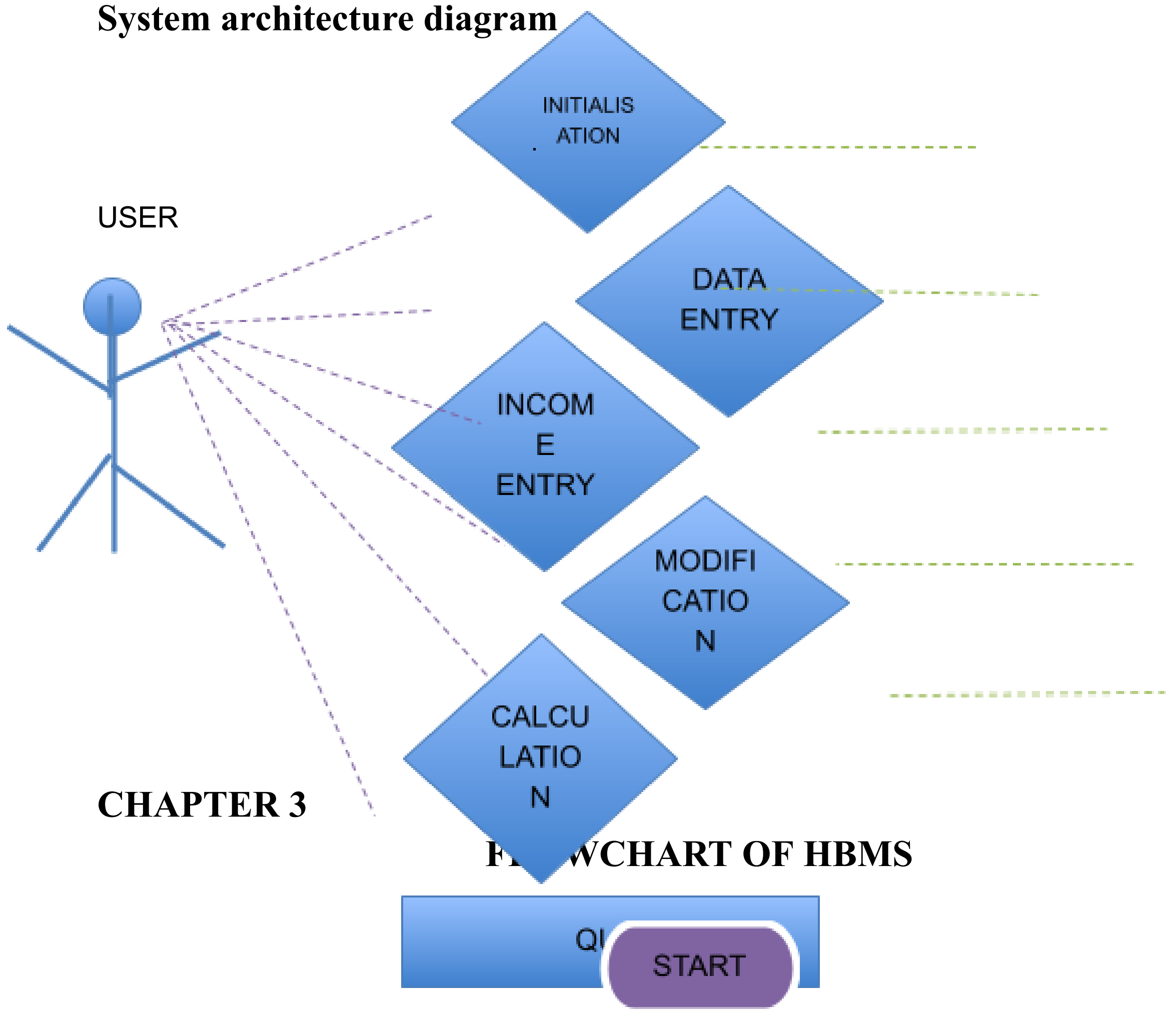
The given code is an interactive household budget management system that utilizes a JSON file to store account information. It provides options to initialize, data entry , monthly income entry , modification , and calculation of budget . Below is the algorithmic representation of the code:

Algorithm:

1.Import necessary libraries

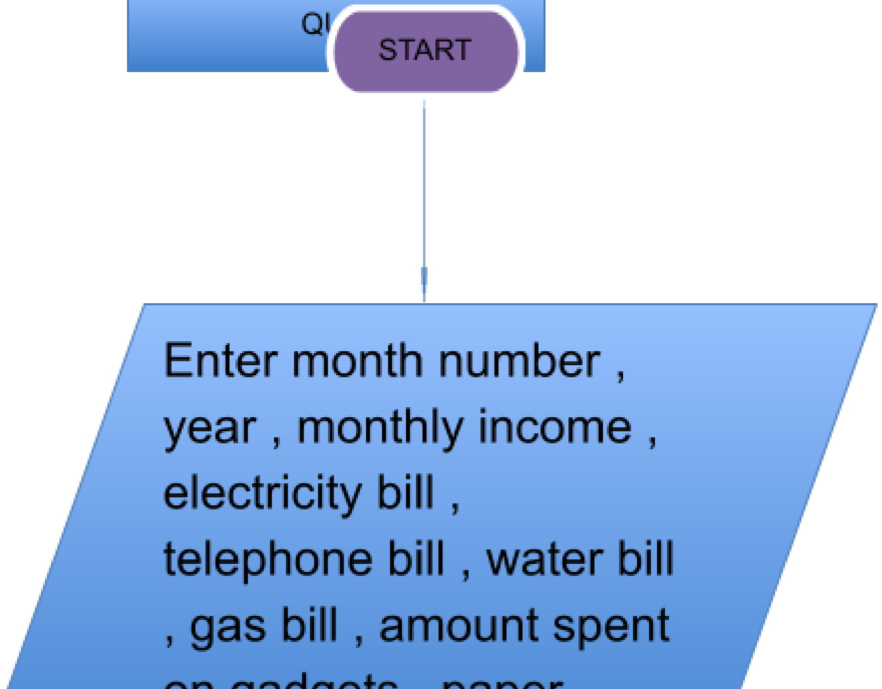
2.Define variables
3.Define Functions
4.Main Program
5.Operations within the main loop
6.Exit the loop when the user chooses to quit (6).
This algorithm forms the structure of the given code, allowing users to perform various operations related to household management through an interactive command-line interface.

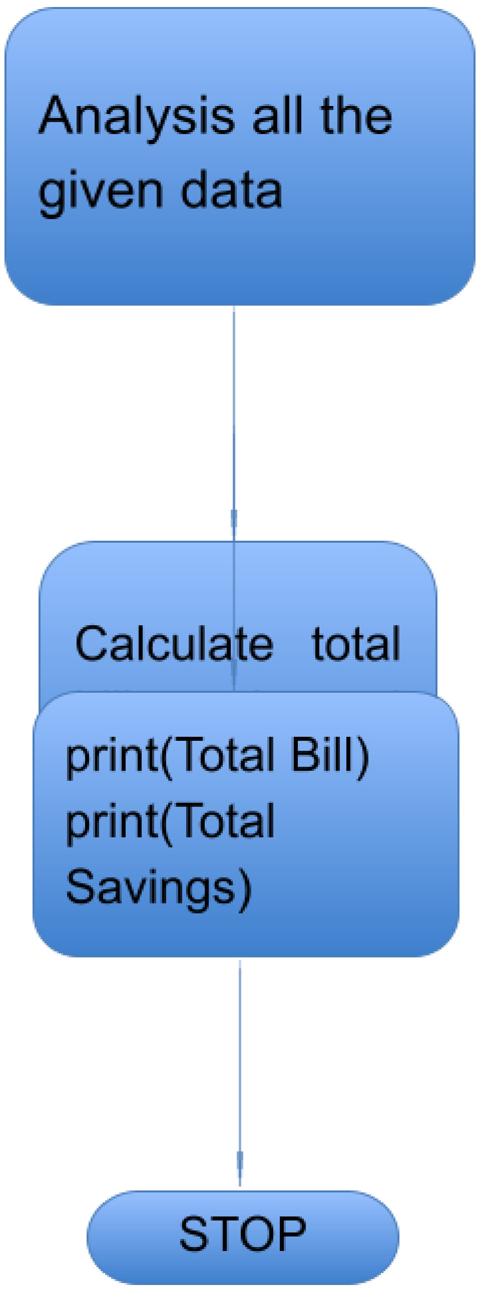
System architecture diagram



CHAPTER 3

FLOWCHART OF HBMS


START
Enter month number , year , monthly income , electricity bill , telephone bill , water bill , gas bill , amount spent


Analysis all the given data
Calculate total
print(Total Bill)
print(Total Savings)
STOP

CHAPTER 4
CODING OF HBMS

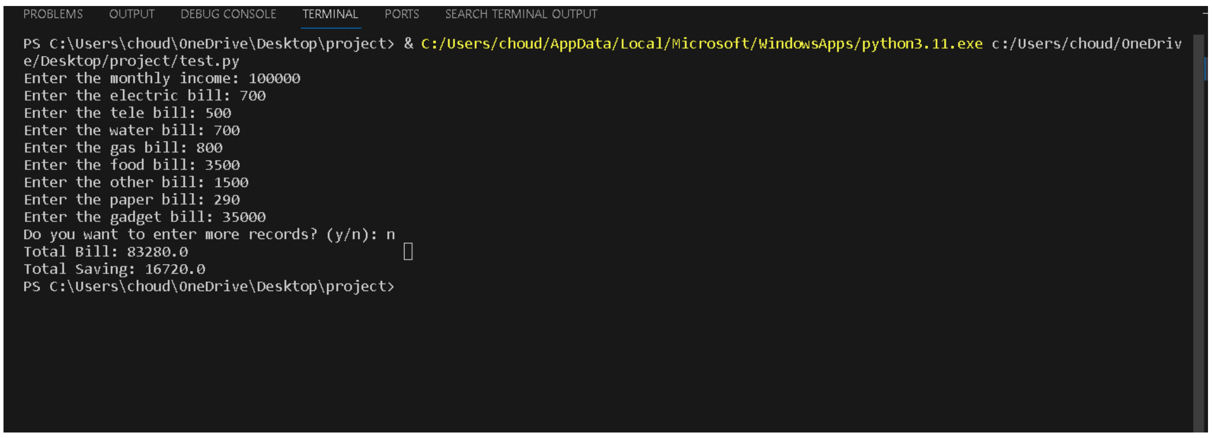
```
import json

class Budget:
    def __init__(self):
        # Initialize expense categories and financial metrics
        self.expenses = {
            'electric': 0,
            'tele': 0,
            'water': 0,
            'gas': 0,
            'food': 0,
            'other': 0,
            'paper': 0,
            'gadget': 0
        }
        self.avail = 0
        self.total_bill = 0
        self.total = 0
        self.savings = 0

    def modify(self):
        # Allow users to modify expenses
        for expense in self.expenses:
            new_value = float(input(f"Enter the new {expense} bill (for the old one enter '-1'): "))
            if new_value != -1:
                self.expenses[expense] = new_value

        # Recalculate financial metrics
        self.calculate()

    def getdata(self):
        # Get user input for monthly income and expenses
        self.avail = float(input("Enter the monthly income: "))
        for expense in self.expenses:
            self.expenses[expense] = float(input(f"Enter the {expense} bill: "))

        # Calculate financial metrics
        self.calculate()
    def putdata(self):
        # Display financial summary
        print("Bills:")
        for expense, value in self.expenses.items():
            print(f"{expense.capitalize()}: {value}")
        print(f"Total cost of bills: {self.total_bill}")
        print(f"Total expenditure: {self.total}")
        print(f"Total savings: {self.savings}")

    def calculate(self):
        # Calculate total bills, total expenditure, and savings
        self.total_bill = sum(self.expenses.values())
        self.total = self.total_bill + sum([self.expenses['food'], self.expenses['other'],
self.expenses['paper'], self.expenses['gadget']])

        try:
            self.savings = self.avail - self.total
        except ZeroDivisionError:
            # Handle division by zero error
            print("Error: Monthly income cannot be zero.")

        return [self.total, self.savings]

    def analyse(self):
        # Provide financial analysis based on spending patterns
        max_category = max(self.expenses, key=self.expenses.get)
        max_expenditure = self.expenses[max_category]

        print("\nThe maximum expenditure of the month is for:")
        print(f"{max_category.capitalize()}. Consider reducing spending in this category.")

        if self.savings == 0:
            print("Your savings for the month are zero. Please be very careful from here on.")
        elif self.savings < 0:
            print("You have run into debt. Be penny-wise the next time!")

def create_records():
        …

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

MODULES OF HBMS

In the provided code for the simple HOUSEHOLD BUDGET MANAGEMENT system, you can enhance the organization and modularity by using separate modules. This can make the code more maintainable and easier to understand. Here's a basic suggestion on how you could structure your modules:
1: budget.py(Main Module)
2: record_management.py(Record Management Module)
3: main.py(Main execution)
By organizing your code into modules, you can achieve a better separation of concerns and make the codebase more modular and maintainable. Each module has a specific responsibility, making it easier to understand and extend the functionality of the system.

CHAPTER 6

RESULT OF HBMS



```
PS C:\Users\choud\OneDrive\Desktop\project> & C:/Users/choud/AppData/Local/Microsoft/WindowsApps/python3.11.exe c:/Users/choud/OneDriv
e/Desktop/project/test.py
Enter the monthly income: 100000
Enter the electric bill: 700
Enter the tele bill: 500
Enter the water bill: 700
Enter the gas bill: 800
Enter the food bill: 3500
Enter the other bill: 1500
Enter the paper bill: 290
Enter the gadget bill: 35000
Do you want to enter more records? (y/n): n
Total Bill: 83280.0
Total Saving: 16720.0
PS C:\Users\choud\OneDrive\Desktop\project>
```

CHAPTER 7
CONCLUSION OF HBMS

This project involves creating a simple console-based household budget management system using object-oriented programming principles in python. Here are some of the key learning points from this project:

1. Data validation
2. Data storage using dictionary
3. Menu driven user interface
4. User input and output
5. Flow control
6. Exception handling
7. Code organisation
8. Code reusability

9. User guidance and feedback
10. Security considerations
11. Graceful program termination

Overall, this project serves as a practical example of applying programming concepts to create a functional and interactive system. It can be a starting point for more advanced features, improvements and even the development of more sophisticated household budget management systems.

CHAPTER 8
REFERENCES OF HBMS

Some of the references are:

“THE TOTAL MONEY MAKEOVER” by dave ramsey
“The complete refresh python” by Martin C Browd
“Python crash course” by Eric Matthes
W3Schools Python Tutorial
Hackerrank Python Practice
THE SIMPLE DOLLAR(BLOG)
MR. MONEY MUSTACHE(BLOG)