UNIVERSITY OF TEXAS ARLINGTON

CSE 1320 Project Documentation

ORION

Students names, surnames, IDs:
1. Dikshya Laudari, 1002227124
2. Madhurima Shrestha, 1002209924
3. Kripa Neupane, 1002203947
4. Badrinath Dalvai, 1002216983

Mentor: Marika Apostolova

PROJECT LOGO

Intermediate programming CSE 1320

Student declaration:

We declare that:

- *We understand what is meant by plagiarism*
- *The implication of plagiarism has been explained to me by our professor*
- *This assignment is all team own work and we have acknowledged any use of the published and unpublished works of other people.*

1 Student name, surname, ID and **signature: ……………………......**
2 Student name, surname, ID and **signature: ……………………......**
3 Student name, surname, ID and **signature: ……………………......**
4 Student name, surname, ID and **signature: ……………………......**

04-20-2025

Date:……..

	Total number of pages including this cover page	9
Class Code / Group	CSE 1320	
Lecturer's Name	MARIKA APOSTOLOVA	

Table of Contents

1. Project Introduction

Orion is a terminal-based Application in C that will prompt the users for their current expenses and help them manage their monthly expenses. Based on the input data, it will provide personalized suggestions to reduce or increase their expenses, it allows users to create personal accounts, enter and sort expenses by priority. It utilizes basic concepts of C like file handling, file interlinking, loops, sorting and conditionals to achieve its goals of budget allocation, and to minimize overspending. It generates personalized suggestions using AI which was obtained by triggering the google Gemini API through python and executing in C using bash file.

2. Project Description

Goals and objectives

- It enables users to manage their finances based on priority.
- It allows users to create an account and helps to keep track of their expenses.
- It provides suggestions to the user regarding their expenses using google ai.

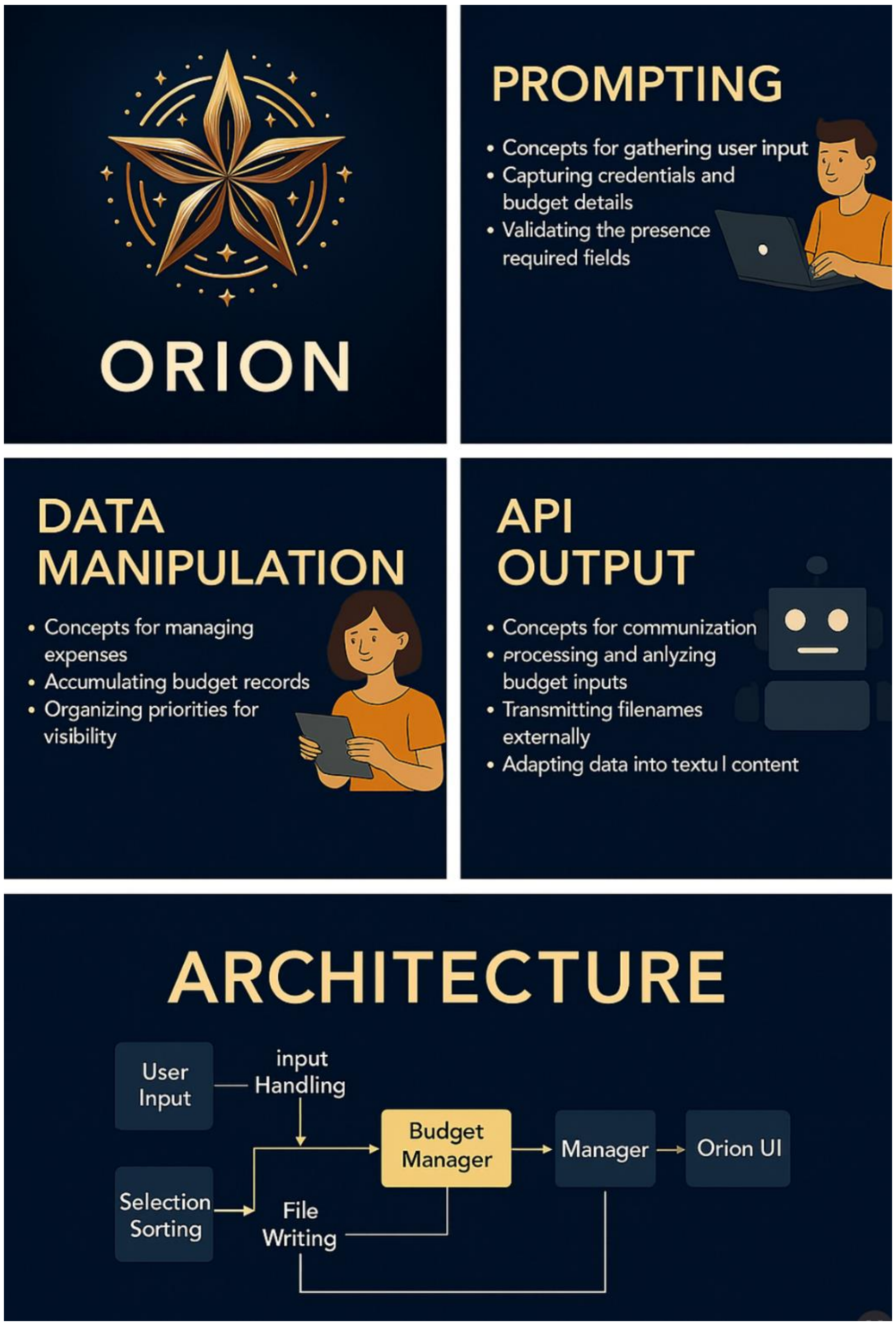
Need for the project:

- Orion can be very useful for someone who struggles to manage their money or someone who wants to save money and wants to keep track of the expenses. Orion addresses the issue of money management by providing simple yet effective budgeting solutions with AI advice.

Phases of the project:

- Account creation or login for existing account
- Takes monthly budget input for income and expenses
- Ask user for sorting the expenses based on their priority
- Then the budget data will be saved in a local file
- Ask user if they need Orion AI Assistant for budget analysis
- Generates AI advice using python script

3. Project Architecture


ORION

PROMPTING

- Concepts for gathering user input
- Capturing credentials and budget details
- Validating the presence required fields

DATA MANIPULATION

- Concepts for managing expenses
- Accumulating budget records
- Organizing priorities for visibility

API OUTPUT

- Concepts for communization
- processing and anlyzing budget inputs
- Transmitting filenames externally
- Adapting data into textu l content

ARCHITECTURE

4. Programming Concepts Used

- We used concepts like sorting, file handling, system calls, structures, functions, inter-process communication.

- **Functions** – We assigned separatre functions while performing each task, which made our code easy to use and understand. Function helped in logical breakdown of tasks like taking user input, saving data, sorting, and authentication.
- **File handling** – We used file handling for storing user data, user persistence and data tracking. Files were used in reading and writing mode.
- **Structures** – Concepts of structures were used to store data of user and budget.
- **Sorting** – We used the concept of sorting to give the user choice to select their expenses based on priority. Priority based selection sort was implemented to generate the suggestion for user based on the priority.
- **Systems calls** – Orion generates suggestions using google API (Application Programming Interface), which was used in python to integrate AI capabilities in our program. We then used system (“python3 orion_adivce.py”) to trigger python from within C.
- **Inter-process communication** – We used .current_user file to pass state from C to Python which took the user input data to python and then the suggestion was generated in python using google Api.

5. Code Description

Our three keys' features include user authentication, priority-based sorting, and python AI suggestion Integration

- **User authentication**

```c
FILE *userfile = fopen(filename, "r");
if (!userfile) {
printf("Error opening user file.\n");
return;
}

//Read user data from user file
fscanf(userfile, "%s %s %s", user.username, user.password, user.fullname);
fclose(userfile);

printf("\nEnter your password: ");
fgets(password, sizeof(password), stdin);
password[strcspn(password, "\n")] = '\0'; //replace new line with null character for accurate comparison

if (strcmp(password, user.password) != 0) { //if stored password and user-input password doesn't match, don't allow access
printf("Incorrect password.\n");
input();
return;
}

//if program reaches here, password has matched, access allowed
printf("\nWelcome to Orion, %s!\n", user.fullname);
InputChoice(filename); //asks user what they want to do today
```

User account details and passwords are handled securely using file-based storage. Each user's information is stored in a dedicated text file (e.g., xyz.txt). When a user who already has an account logs in to the C program validates the credentials before proceeding to the next step. This part ensures that unauthorized users are blocked from accessing user's data.

- **Priority-Based Budget Input and Sorting**

```c
//selection sort method (ascending order)
void sortCategories(struct ExpenseCategory categories[], int count) {
for (int i = 0; i < count; i++) {
int smallest_priority = i; //Assume the current priority is the smallest
//search for the smaller priority in the unsorted part
for (int j = i + 1; j < count; j++) {
if (categories[j].priority < categories[smallest_priority].priority) {
smallest_priority = j; //if smaller priority is found, update smallest_priority
}
}
if (smallest_priority != i) { //if initially assumed priority was not smallest, swap
struct ExpenseCategory temp = categories[i]; //temporary space to store current priority
categories[i] = categories[smallest_priority]; //current priority replaced by smaller priority
categories[smallest_priority] = temp; //in the initial place of smaller priority, current priority is stored
}
}
}
```

When a user inputs their expenses like rent, groceries, insurance etc; each of these categories is assigned a unique priority from 1 to 6, with one being the highest priority. User gets to sort their expenses based on their need. Selection sort algorithm arranges the user's expenses in order of priority and helps user in providing idea of their expenses.

- **AI powered financial feedback**

In C:

```c
char choice;
printf("\nWould you like to get personalized financial advice? (Y/N): ");
scanf("%c", &choice);
```

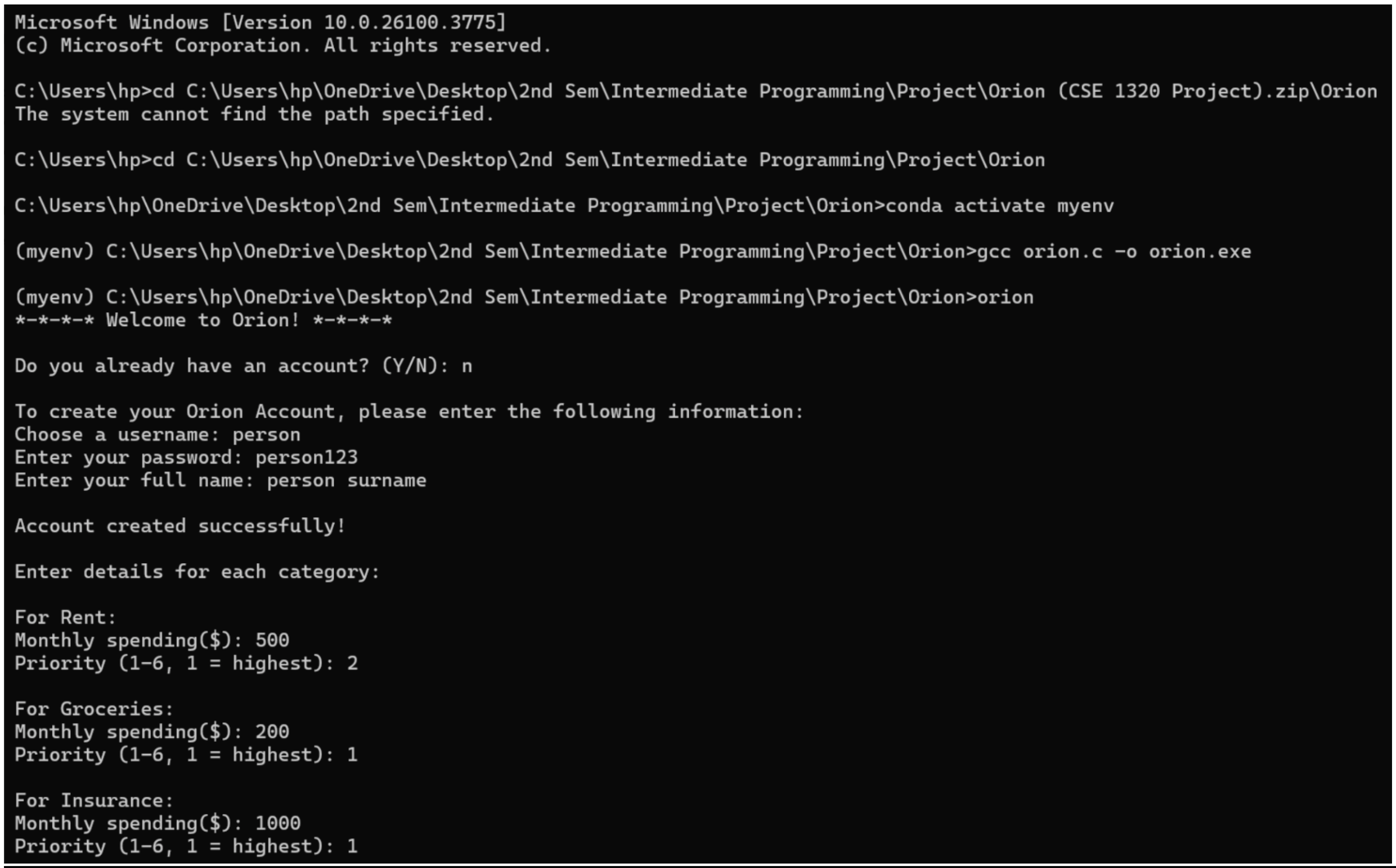
```
while (getchar() != '\n'); //consumes newline

if (tolower(choice) == 'y') {
printf("\nGetting personalized financial advice from Orion AI...\n\n");
system("python ai_advice.py"); //integration of AI for personalized budget vs expense response
}
else if(tolower(choice) == 'n'){
printf("\nHave a great day!\n");
}
else{
printf("\nInvalid choice.\n");
return;
}
```

In python:

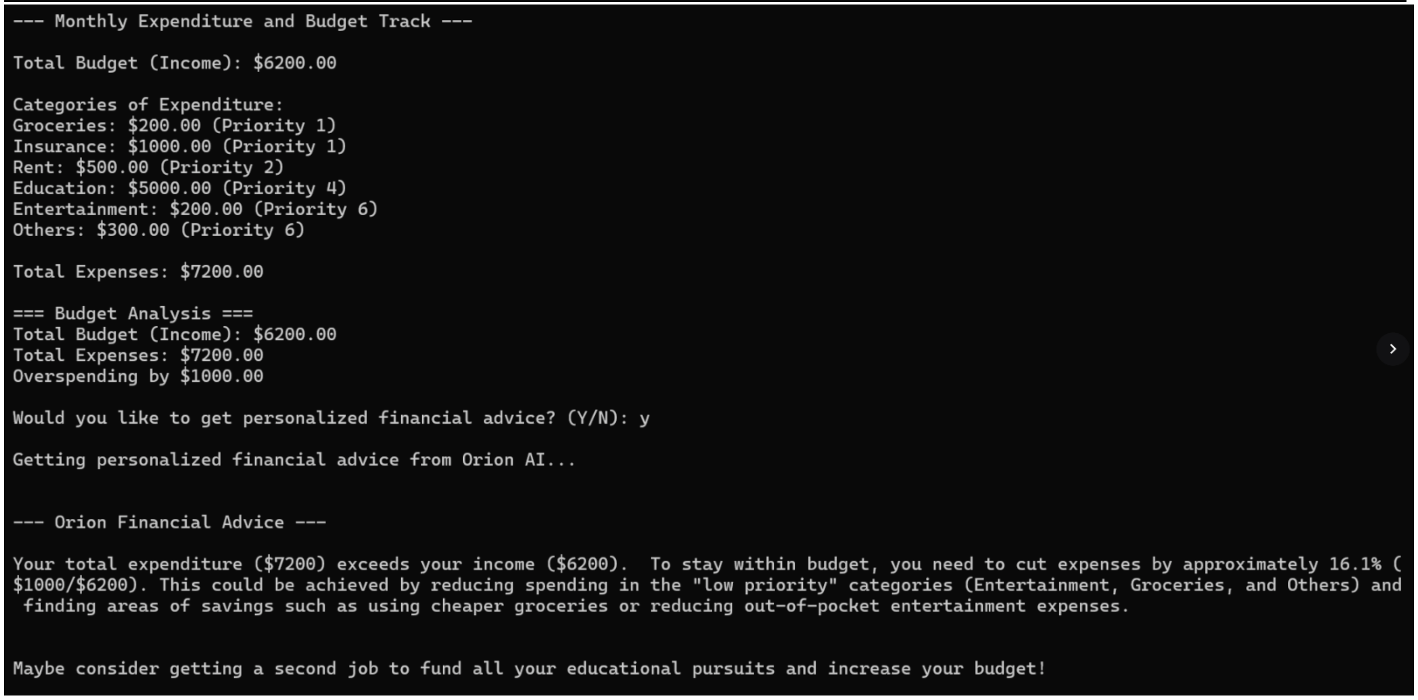
```
import google.generativeai as genai

# Set your API key
genai.configure(api_key="AIzaSyDoG1s9n2V8etcrULDOs8t0YpX7RWdDu6I")

data = {}

try:
with open("ai_input.txt", "r") as f:
for line in f:
if ":" in line:
key, value = line.strip().split(": ")
data[key.lower()] = float(value)
except FileNotFoundError:
print("❌ ai_input.txt not found. Run the C program first.")
exit()

prompt = f"""
Here is my monthly financial data:

- Income: ${data.get('income', 0)}
- Education: ${data.get('education', 0)}
- Insurance: ${data.get('insurance', 0)}
- Groceries: ${data.get('groceries', 0)}
- Rent: ${data.get('rent', 0)}
- Entertainment: ${data.get('entertainment', 0)}
- Others: ${data.get('others', 0)}
- Total Expenditure: ${data.get('total expenditure', 0)}

Based on this, analyze my spending and suggest me in three short sentences line-by-line: 1) only if expenditure exceeds budget,
analyze how to and by what percentage should I cut off expenses from 'low priority (3-6)' category and keep things under budget
2)else, only if budget exceeds expenditure, tell me how to improve my quality of life by increasing my spending by what percentage
on 'high priority (1-3)' category and keep aside what percentage on savings for future. Make one of the two choices I gave you
based on budget amount and expenses and discard the other. At the end, give one line of sarcastic suggestion just for fun.
"""


# Call Gemini
```

```
model = genai.GenerativeModel("models/gemini-1.5-flash")
response = model.generate_content(prompt)

print("\n--- Orion Financial Advice ---\n")
print(response.text)
```

After the budget is saved in a file, C program triggers a Python script, orion_advice.py, to analyze and provide suggestion based on the user input. The C passes the current user files to the python code using .current_user. When python gets the user input from C it uses the values for total budget and total expenses, then calculates whether user is overspending or not. If the user is overspending, then it generates personalized suggestions for managing the expenses in different categories based on the priority. This integration gives human-like feedback to the C-based system using API.



```
Microsoft Windows [Version 10.0.26100.3775]
(c) Microsoft Corporation. All rights reserved.

C:\Users\hp>cd C:\Users\hp\OneDrive\Desktop\2nd Sem\Intermediate Programming\Project\Orion (CSE 1320 Project).zip\Orion
The system cannot find the path specified.

C:\Users\hp>cd C:\Users\hp\OneDrive\Desktop\2nd Sem\Intermediate Programming\Project\Orion

C:\Users\hp\OneDrive\Desktop\2nd Sem\Intermediate Programming\Project\Orion>conda activate myenv

(myenv) C:\Users\hp\OneDrive\Desktop\2nd Sem\Intermediate Programming\Project\Orion>gcc orion.c -o orion.exe

(myenv) C:\Users\hp\OneDrive\Desktop\2nd Sem\Intermediate Programming\Project\Orion>orion
*-*-*-* Welcome to Orion! *-*-*-*

Do you already have an account? (Y/N): n

To create your Orion Account, please enter the following information:
Choose a username: person
Enter your password: person123
Enter your full name: person surname

Account created successfully!

Enter details for each category:

For Rent:
Monthly spending($): 500
Priority (1-6, 1 = highest): 2

For Groceries:
Monthly spending($): 200
Priority (1-6, 1 = highest): 1

For Insurance:
Monthly spending($): 1000
Priority (1-6, 1 = highest): 1
```



```
--- Monthly Expenditure and Budget Track ---

Total Budget (Income): $6200.00

Categories of Expenditure:
Groceries: $200.00 (Priority 1)
Insurance: $1000.00 (Priority 1)
Rent: $500.00 (Priority 2)
Education: $5000.00 (Priority 4)
Entertainment: $200.00 (Priority 6)
Others: $300.00 (Priority 6)

Total Expenses: $7200.00

=== Budget Analysis ===
Total Budget (Income): $6200.00
Total Expenses: $7200.00
Overspending by $1000.00

Would you like to get personalized financial advice? (Y/N): y

Getting personalized financial advice from Orion AI...


--- Orion Financial Advice ---

Your total expenditure ($7200) exceeds your income ($6200).  To stay within budget, you need to cut expenses by approximately 16.1% (
$1000/$6200). This could be achieved by reducing spending in the "low priority" categories (Entertainment, Groceries, and Others) and
 finding areas of savings such as using cheaper groceries or reducing out-of-pocket entertainment expenses.


Maybe consider getting a second job to fund all your educational pursuits and increase your budget!
```

6. Group members

- **Kripa Neupane** - Input handling, authentication, and initial data manipulation
- **Madhurima Shrestha** – Core data manipulation and code clarity through comments
- **Dikshya Laudari** – AI integration, AI execution, and report writing
- **Badrinath Dalvai** – AI integration, debugging, and optimization

7. Conclusion and future works

It is concluded that the main objective of the application is to help computer science students understand the fundamentals of C. By browsing through the application and looking at the code for each graphical interpretation, students should be able to easily understand the implementation. Future work could involve adding more categories which are more detailed and have additional items. This part of the documentation is reserved for conclusions and possible future improvements to the project itself.

Orion successfully meets the objective of helping in money management by tracking user expenses and income while providing financial suggestions based on the user's priority. It demonstrates the use of fundamental C programming and python in creating terminal-based programs to help people in managing their expenses.

Future Enhancements:

- Implement monthly tracking using user's expenses history and provide suggestion along with comparison
- Use of user-friendly Graphical User Interface (GUI) to replace terminal based interaction making it more accessible to non-technical users.
- Allow users to create and name their own category rather than predefined ones.