Software System for Food Delivery System

UCS2265 – Fundamentals and Practice of Software Development

A PROJECT REPORT

Submitted By

VIGNESHAR K-3122235001156
VISHNU KUMAR S- 3122235001164

THARUN KUMAR S-3122235001146

Department of Computer Science and Engineering Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University) Kalavakkam – 603110

June 2024

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this project report titled “Software System for Railway Reservation” is the Bonafide work of “Vishnukumar S (3122235001164), Vigneshwar K (3122235001156) and Tharun Kumar S (3122235001146)” who carried out the project work in the UCS226– Fundamentals and Practice of Software Development during the academic year 2023-24.

Internal Examiner External Examiner

Date:

TABLE OF CONTENTS

1. INTRODUCTION:

The online food ordering system establishes a comprehensive food menu online, enabling customers to easily place orders as per their preferences. The management maintains customer records and continuously works to enhance food delivery services. This system includes a feedback mechanism where users can rate food items. Additionally, the system recommends hotels and food based on user ratings, informing hotel staff about areas for improvement, including quality. Payments can be made online, by cash, or through a pay-on-delivery system. For enhanced security, each user maintains a separate account with a unique ID and password. Our application ensures an efficient and user-friendly ordering experience, utilizing advanced algorithms to address our problem statement effectively.

2. PROBLEM STATEMENT:

- Develop a responsive application enabling customers to view nearby hotel availability and receive recommendations for quality establishments close to them. By inputting their address, and providing feedback, users can see the closest hotels based on ratings, feedback, price range, and available discounts.
- Additionally, the application utilizes past purchase history to suggest personalized food recommendations, ensuring a tailored dining experience. This system enhances convenience by allowing seamless access to nearby dining options while prioritizing user preferences and feedback to deliver optimal service and satisfaction.
- Recommend the food according to the past purchase history.
- Input: Address of the customer, Login credentials, Feedback from the customer.
- Output: Customer could be able to see the nearest available hotels Based on ratings, feedback, price range and discounts

3. EXTENDED EXPLORATION OF PROBLEM STATEMENT:

We researched many food delivery applications and surfed web to identify the major algorithms used in different modules of the food delivery application.
The major algorithms that we found useful and used in our food delivery application are,
1)Haversine formula
2) Binary insertion sort

HARVERSINE FORMULA:

In a food delivery system, the Haversine formula plays a crucial role in calculating distances between two points on the Earth's surface defined by their latitude and longitude coordinates. This formula is particularly useful for determining the proximity of customers to restaurants or delivery addresses. By leveraging the Haversine formula, the application can:

1. Optimize Delivery Routes: Calculate the shortest distance between the restaurant and the customer's location, facilitating efficient delivery route planning.

2. Display Nearby Options: Show customers nearby restaurants based on their current location, ensuring they have access to relevant dining choices.

3. Real-time Tracking: Use distance calculations to provide accurate estimates of delivery times and real-time tracking of delivery progress.

4. Geofencing: Implement geofencing techniques to define delivery areas based on distance thresholds from restaurants, ensuring timely and efficient service.

Overall, integrating the Haversine formula enhances the operational efficiency and user experience of the food delivery system by enabling precise location-based functionalities and optimizing logistical operations.

BINARY INSERTION SORT:

- Binary insertion sort is instrumental in optimizing task management within a food delivery management system by efficiently sorting tasks based on criteria like delivery time and customer location. Unlike traditional insertion sort, it uses binary search to find the correct position for new tasks in a sorted list, reducing the number of comparisons needed and improving efficiency.

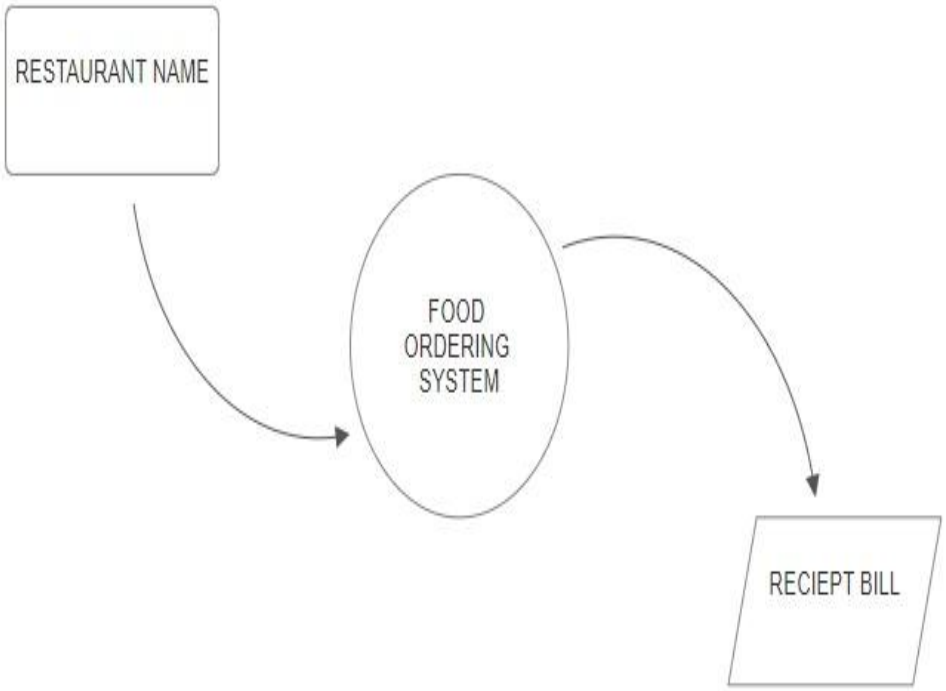
- In the context of food delivery, binary insertion sort ensures that new orders are seamlessly integrated into the existing sequence of tasks without the need for a full re-sort. This allows for quick processing of incoming orders based on priority, delivery proximity, or customer preferences. For instance, as new orders arrive, the algorithm swiftly determines where each order fits based on criteria such as delivery urgency or geographical proximity to optimize delivery routes.

- Moreover, binary insertion sort supports real-time updates, making it adaptable to changing conditions like traffic or customer address changes. By integrating with mapping services, the algorithm enhances route planning by organizing delivery tasks in an optimal sequence, thereby improving delivery efficiency and customer satisfaction in the food delivery service.

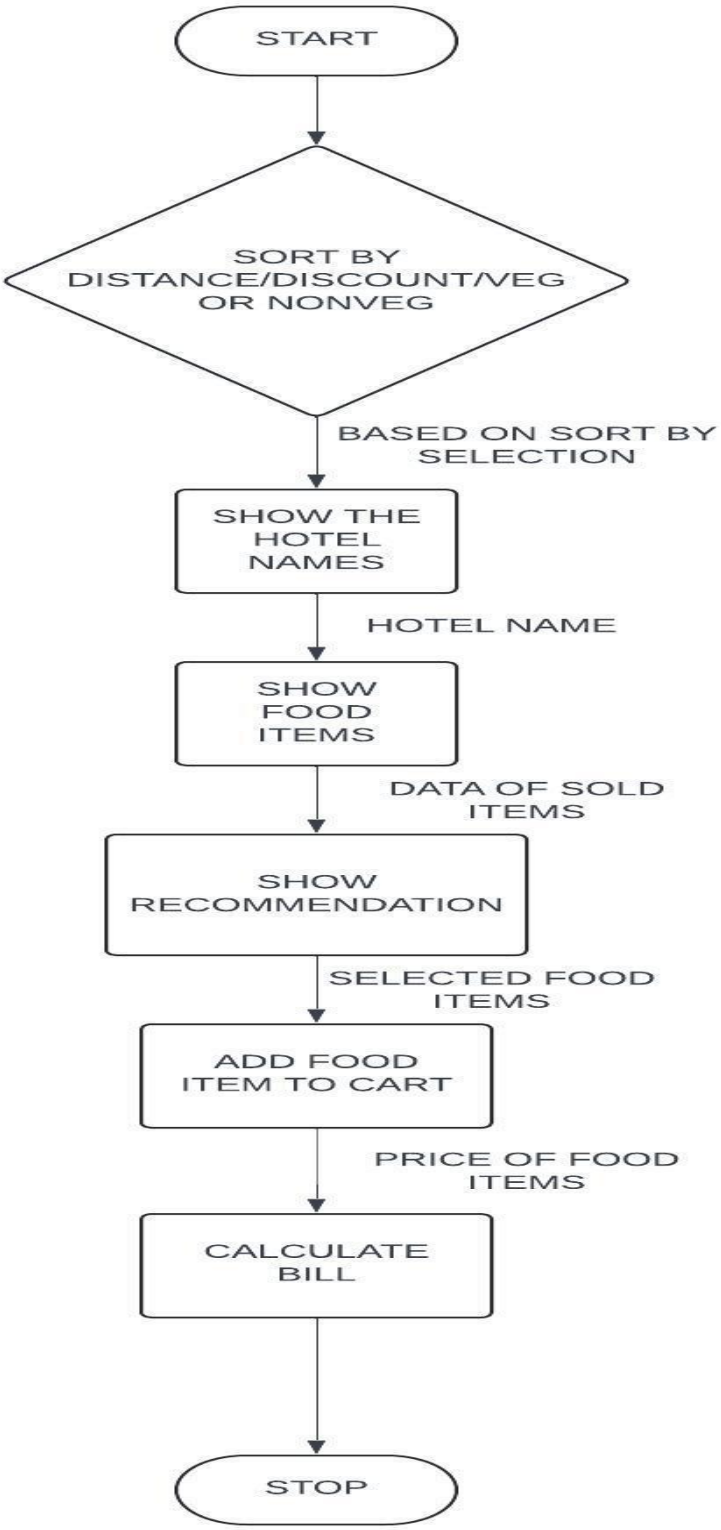
4. ANALYSIS USING DATA FLOW DIAGRAM

Level 0:

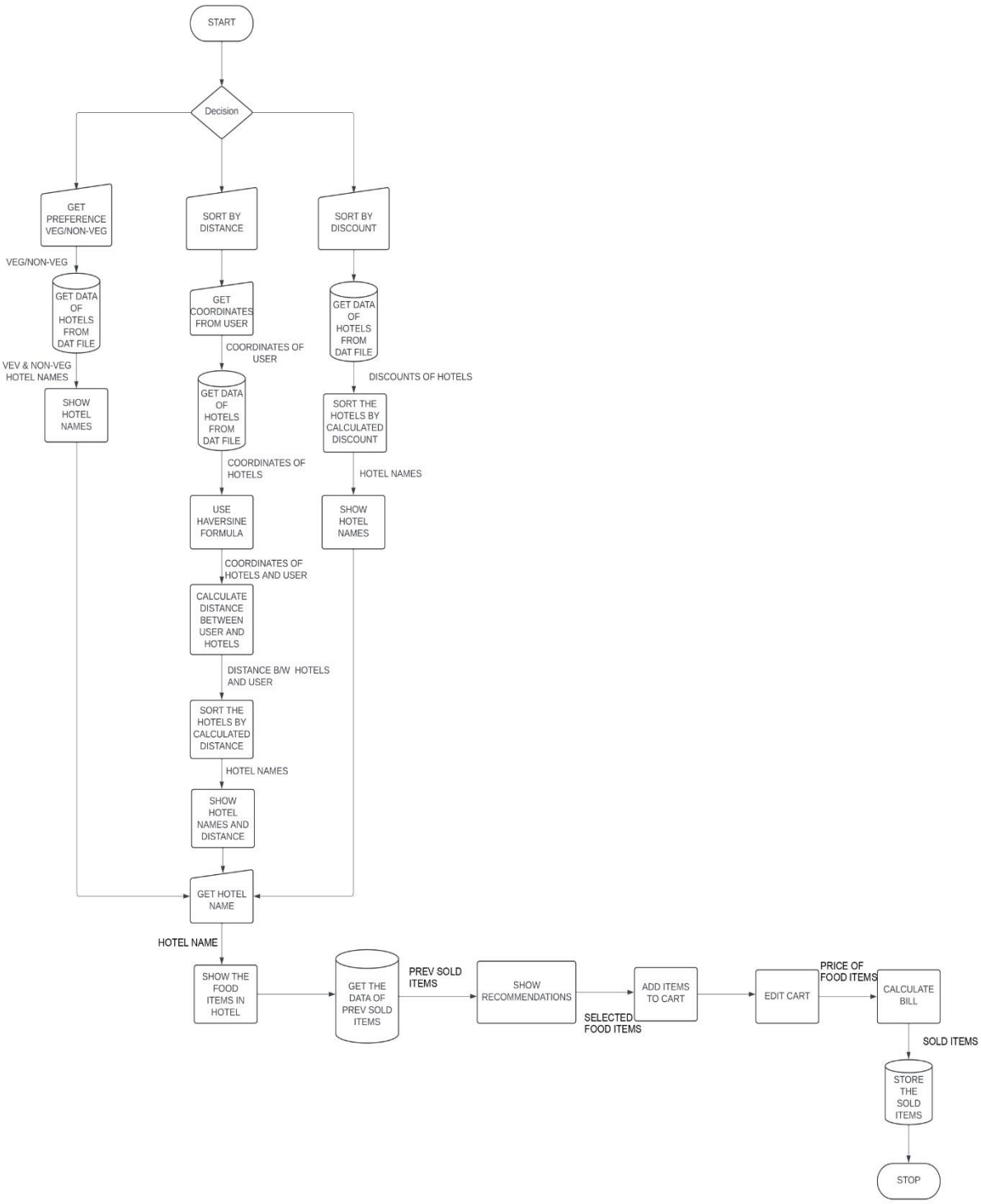
At this level, the Input and Output of the system are shown. The system is designed and established across the world with input and output at this level.



LEVEL 1:


START
SORT BY DISTANCE/DISCOUNT/VEG OR NONVEG
BASED ON SORT BY SELECTION
SHOW THE HOTEL NAMES
HOTEL NAME
SHOW FOOD ITEMS
DATA OF SOLD ITEMS
SHOW RECOMMENDATION
SELECTED FOOD ITEMS
ADD FOOD ITEM TO CART
PRICE OF FOOD ITEMS
CALCULATE BILL
STOP

LEVEL 2

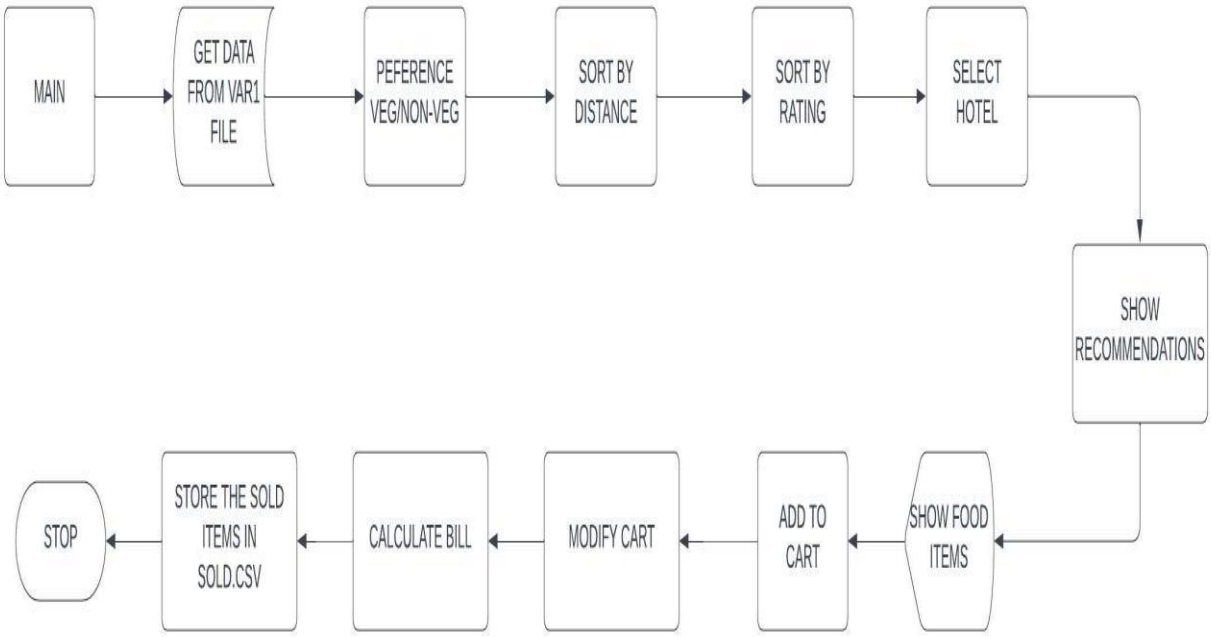

START
Decision
GET PREFERENCE VEG/NON-VEG
SORT BY DISTANCE
SORT BY DISCOUNT
VEG/NON-VEG
GET DATA OF HOTELS FROM DAT FILE
GET COORDINATES FROM USER
GET DATA OF HOTELS FROM DAT FILE
VEV & NON-VEG HOTEL NAMES
COORDINATES OF USER
DISCOUNTS OF HOTELS
SHOW HOTEL NAMES
GET DATA OF HOTELS FROM DAT FILE
SORT THE HOTELS BY CALCULATED DISCOUNT
COORDINATES OF HOTELS
HOTEL NAMES
USE HAVERSINE FORMULA
SHOW HOTEL NAMES
COORDINATES OF HOTELS AND USER
CALCULATE DISTANCE BETWEEN USER AND HOTELS
DISTANCE B/W HOTELS AND USER
SORT THE HOTELS BY CALCULATED DISTANCE
HOTEL NAMES
SHOW HOTEL NAMES AND DISTANCE
GET HOTEL NAME
HOTEL NAME
SHOW THE FOOD ITEMS IN HOTEL
GET THE DATA OF PREV SOLD ITEMS
PREV SOLD ITEMS
SHOW RECOMMENDATIONS
SELECTED FOOD ITEMS
ADD ITEMS TO CART
EDIT CART
PRICE OF FOOD ITEMS
CALCULATE BILL
SOLD ITEMS
STORE THE SOLD ITEMS
STOP

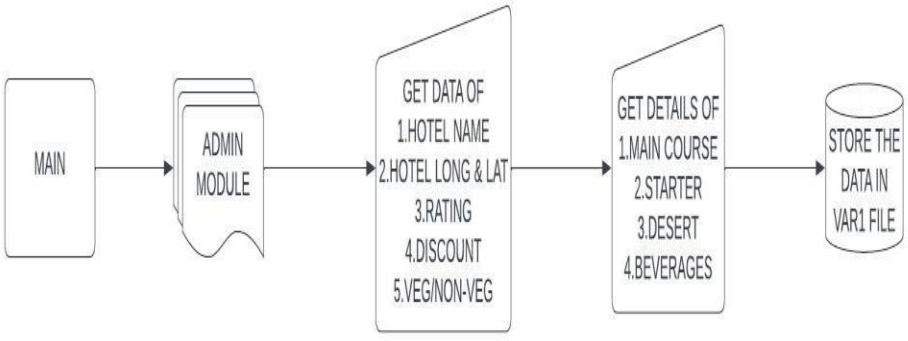
Description For DFD

Here are the main steps involved in the system:

1. Get user's preferences (vegetarian or non-vegetarian) and location (coordinates).
2. Retrieve data on hotels from a database, including their names, locations (coordinates), and discounts.
3. Calculate the distance between each hotel and the user using the Haversine formula.
4. Sort the hotels by distance and then by discount.
5. Show the user a list of hotels sorted by distance and discount.
6. Get the user's selection of hotel.
7. Retrieve data on the food items sold at the selected hotel from a database.
8. Sort the food items by price.
9. Show the user a list of recommended food items.
10. Allow the user to add items to their cart, edit their cart, and checkout.
11. Store the sold items.

Architecture Diagram





Description For Architecture Diagram

- **Start:** The system begins by gathering data from the user.
- **Get Data:** This involves collecting the user's location (through their device most likely) and any specific preferences they have for their stay (e.g., price range, amenities, hotel type).
- **Filter by Preferences:** The system retrieves data on hotels from a database and filters the options based on the user's preferences. This ensures only relevant hotels are considered.
- **Sort by Distance:** Similar to the restaurant scenario, the system calculates the distance between each remaining hotel and the user's location. Hotels are then sorted by distance, with the closest hotel first.
- **Sort by Additional Criteria (Optional):** This step might involve sorting by another factor, such as price (low to high) or user rating (high to low). This depends on the system's design.
- **Show Recommendations:** The system presents the user with a list of recommended hotels based on the sorting criteria. This list prioritizes hotels that meet the user's preferences and are closest to their location.

Admin Module: This suggests a backend for managing hotel data (updating information, adding new hotels).

6.Modules and their description:

ADMIN

- Includes `stdio.h` for standard input/output functions and defines `MAX_ITEMS` (maximum number of items per food category).
- Defines structures as described above.
- The `main` function starts by prompting the user to enter the number of hotels.
- Opens a file named "var1.dat" in binary write mode (`"wb"`) for writing hotel data.
- Checks if the file opened successfully.
- Iterates for the entered number of hotels (`num_hotels`):
    - Prompts the user to enter details for each hotel (name, location, rating, discount, veg/non-veg, number of items in each food category).
    - Reads user input for hotel details and stores them in the `var1` variable of type `struct hotel`.
    - For each food category (starters, main courses, etc.):
        - Reads the number of items from the user.
        - Loops for the number of items and reads name and price for each item, storing them in the corresponding `food` structure.
    - Writes the entire `var1` structure (containing all hotel information) to the file using `fwrite`.
- Closes the file.
- Prints a success message.

Food Ordering System

1. Get user's preferences (vegetarian or non-vegetarian) and location (coordinates).
2. Retrieve data on hotels from a database, including their names, locations (coordinates), and discounts.
3. Calculate the distance between each hotel and the user using the Haversine formula.
4. Sort the hotels by distance and then by discount.
5. Show the user a list of hotels sorted by distance and discount.
6. Get the user's selection of hotel.
7. Retrieve data on the food items sold at the selected hotel from a database.
8. Sort the food items by price.
9. Show the user a list of recommended food items.
10. Allow the user to add items to their cart, edit their cart, and checkout.

Store the sold items

7.Implementation:

The program is a food ordering system with a graphical user interface (GUI) developed using the GTK library in C. The system allows users to log in or register, view and select hotels based on preferences, sort hotels by distance or discount, view available food items from selected hotels, add items to a cart, view and manage the cart, and calculate the total bill. Additionally, it provides recommendations for popular food items based on past orders.

Structure and Components:

User Authentication

The program starts with a login system where users are prompted to enter their username and password. If the credentials match an existing user in the `users.dat` file, the user is logged in. If the user doesn't exist, the program allows for the creation of a new account, saving the new user's credentials to the file.

Data Structures

Several data structures are defined to manage different aspects of the program

User: Stores username and password.

Distance: Stores latitude and longitude for distance calculations.

Starters, Main Courses, Deserts, Beverages: Each struct stores the name and price of the respective food item.

Food: Contains arrays of starters, main courses, deserts, and beverages along with their counts.

Hotel: Stores hotel details including name, distance, rating, food items, discount,

Cart: Contains an array of cart items and the total number of items.Frequent Item: Used to track popular items based on order frequency.

and a preference tag.

Cart Item: Represents an item added to the cart with its name, price, and associated hotel.

GUI Elements

The interface consists of several pages in a notebook widget, each serving a different purpose:

1. Preferences Page: Allows users to enter a preference tag to filter and display hotels matching the preference.

2. Distance Page: Users can input their latitude and longitude to sort hotels by proximity.

3. Discount Page: Displays hotels sorted by discount percentages.

4. Cart Page: Shows items in the user's cart and allows for viewing and managing the cart.

5. Food Items Page: Displays food items available at a selected hotel and allows users to add items to the cart.

Functionality:

Sorting and Filtering

The program provides functionality to sort hotels by distance and discount:

Sorting by Distance: Users input their latitude and longitude, and the program calculates the distance to each hotel using the Haversine formula. The hotels are then sorted by distance.

Sorting by Discount: Hotels are sorted by their discount percentages in descending order.

Viewing and Managing Cart:

Users can view the contents of their cart, add new items, and remove items. The cart page displays each item’s name, price, and the total bill.

Food Items Display

When a user selects a hotel, the program reads the hotel’s food items from `var1.dat` and displays them. This includes starters, main courses, deserts, and beverages along with their prices.

Recommendations

The system tracks the popularity of food items based on past orders recorded in `sold.csv`. It calculates the frequency of each item sold and displays the top five recommended items.

Implementation Details:

Distance Calculation

The program uses the Haversine formula to calculate the great-circle distance between two points on the Earth given their latitudes and longitudes. This is essential for sorting hotels by distance.

File Handling:

User Data: Stored in `users.dat`, accessed in binary mode for reading and writing user credentials.

Hotel Data: Stored in `var1.dat`, contains hotel and food item details.

Order Data: Stored in `sold.csv`, used for tracking sold items to generate recommendations.

GTK Integration:

GTK is used for the GUI components:

GtkNotebook: Manages multiple pages (preferences, distance, discount, cart, food items).

GtkEntry: Input fields for user data, preferences, coordinates, and food item names.

GtkTextView: Displays lists of hotels, cart items, and recommendations.

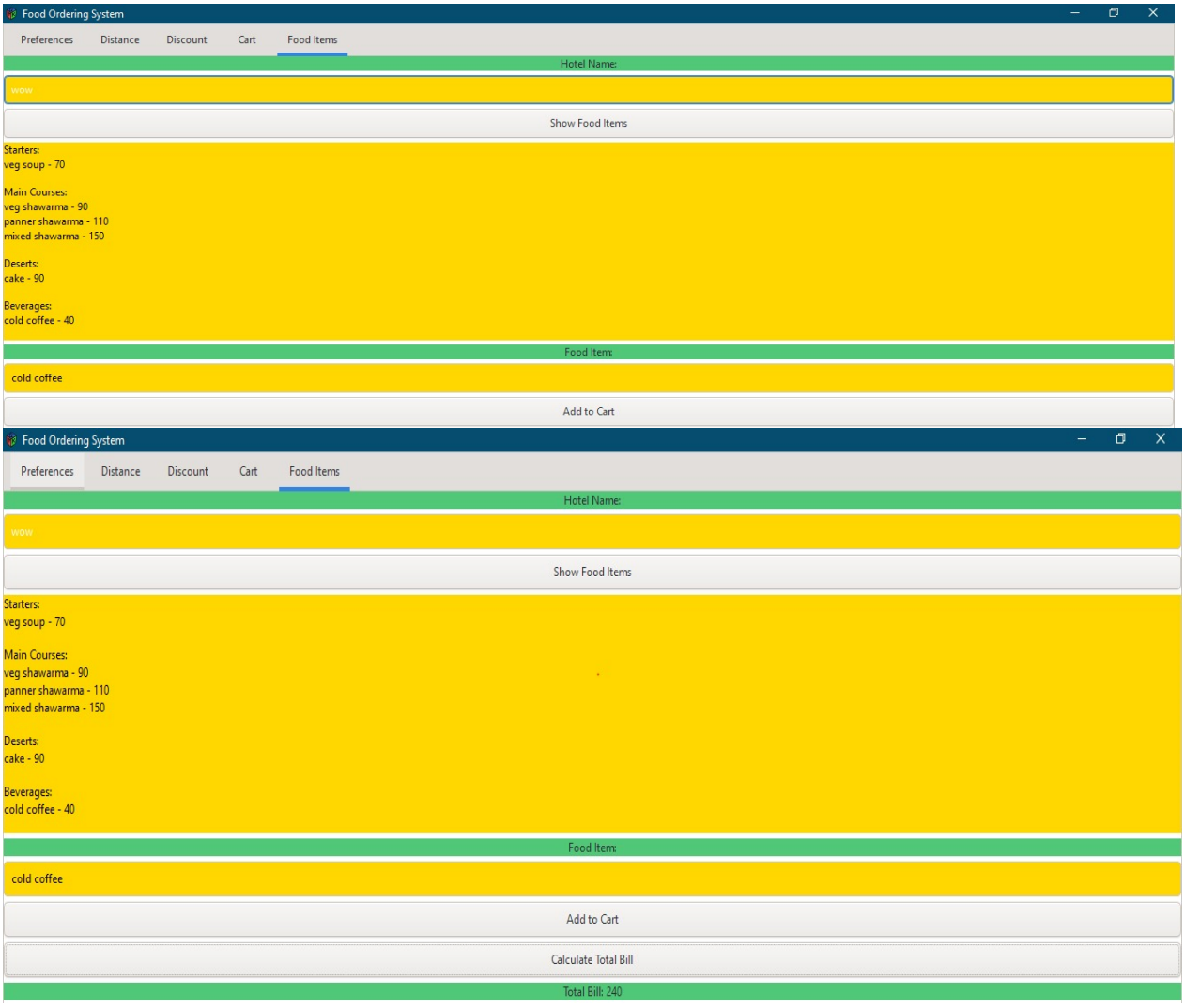
GtkButton: Triggers actions like sorting, adding to cart, and calculating bills.

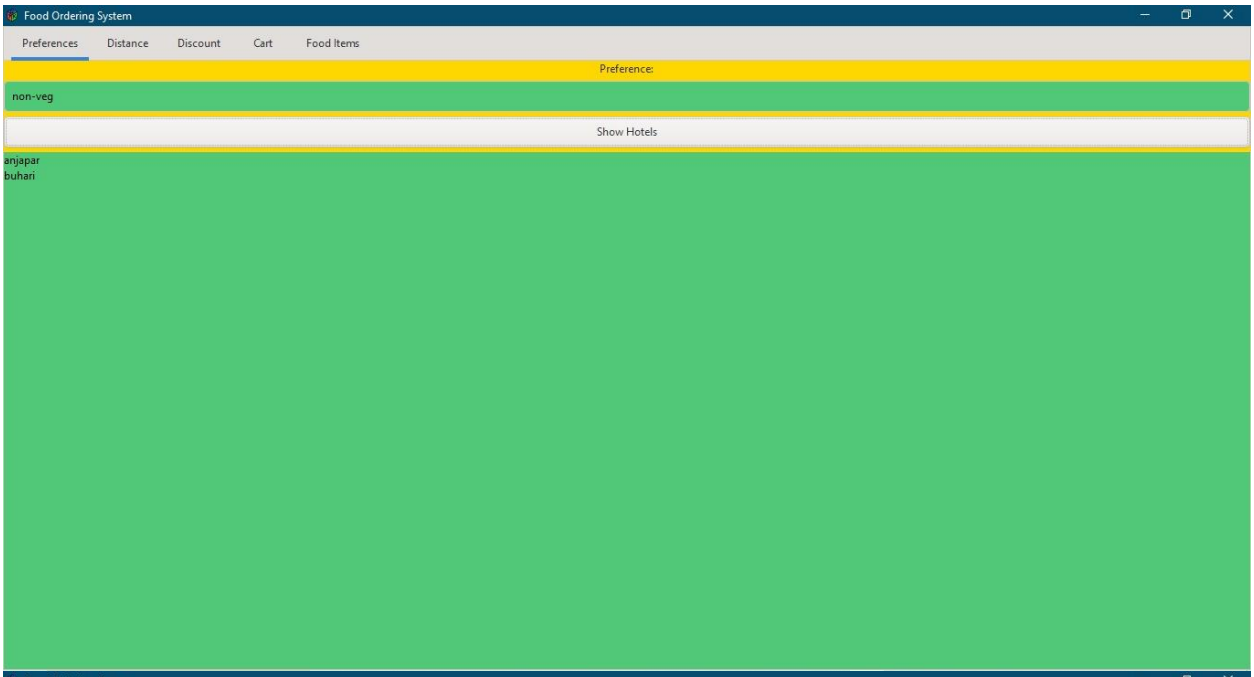
Adding and Removing Cart Items:

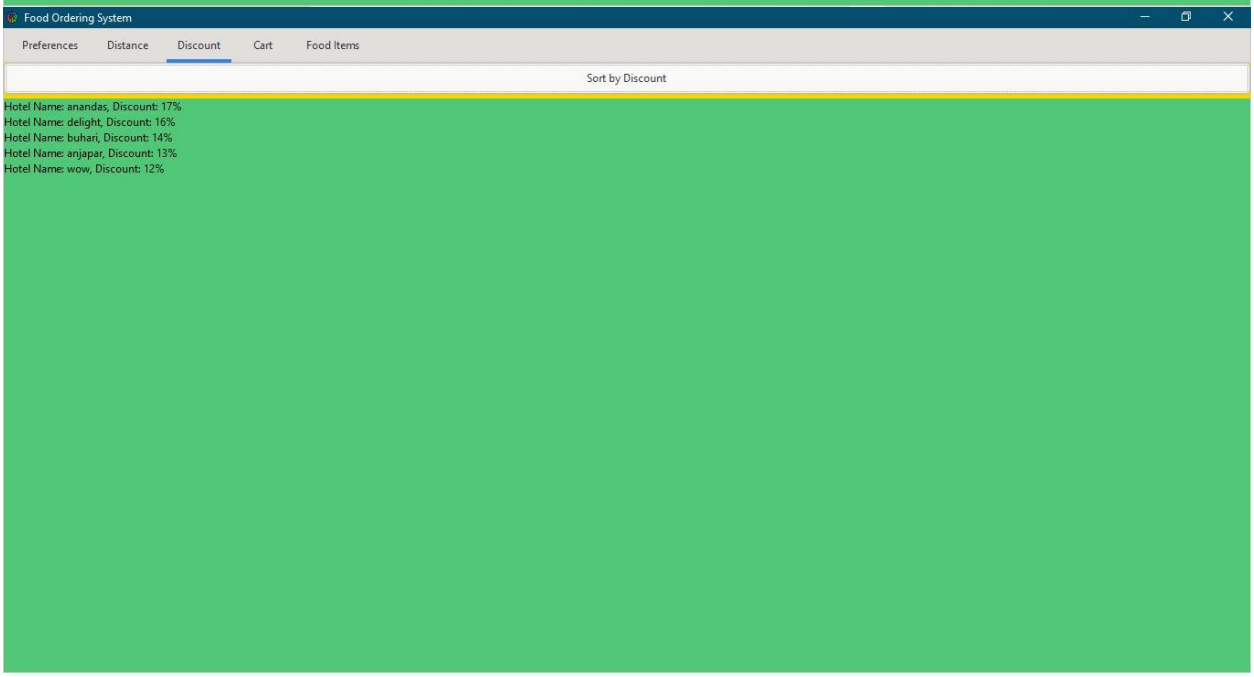
Users can add items to their cart from the food items page. When an item is added, it's appended to the cart's item array. For removing items, the specified item is searched in the cart, and if found, it's removed by shifting subsequent items.

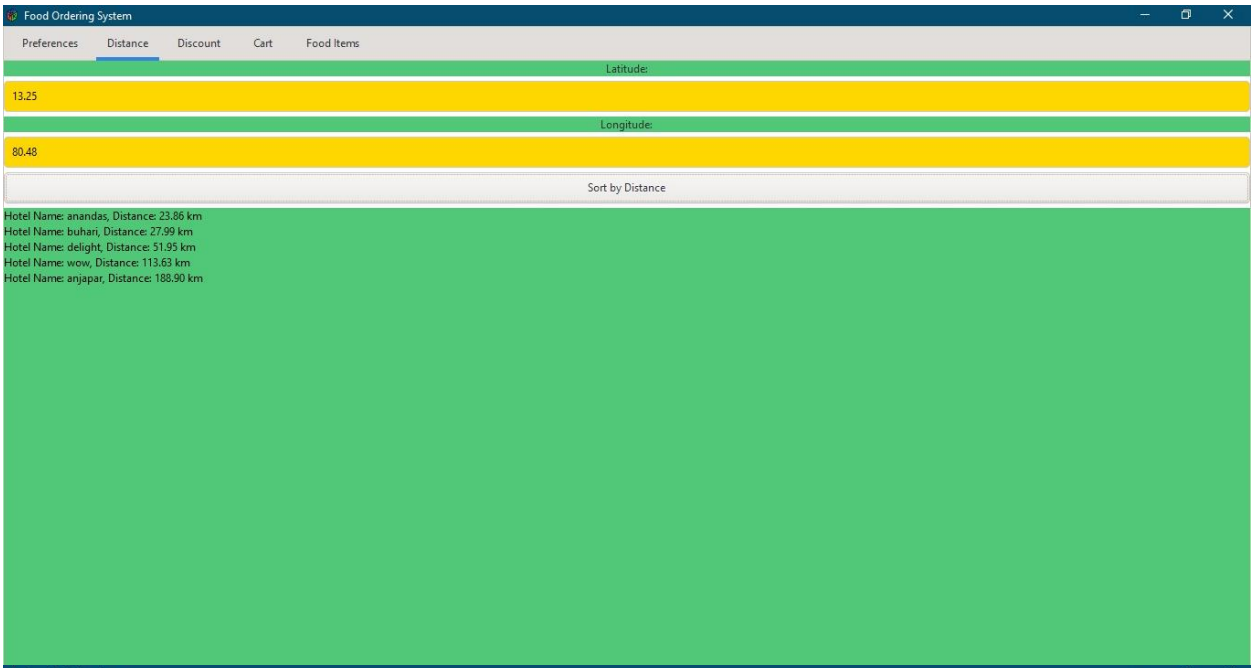
This food ordering system integrates several functionalities into a seamless user experience using GTK for the GUI and C for the backend logic. It efficiently handles user authentication, data management, sorting, and recommendations. The use of structured data storage and file handling ensures the persistence of user and order data. The recommendation system adds value by suggesting popular items, potentially enhancing user satisfaction and sales. Overall, the program exemplifies a comprehensive approach to building a functional and user-friendly food ordering system.

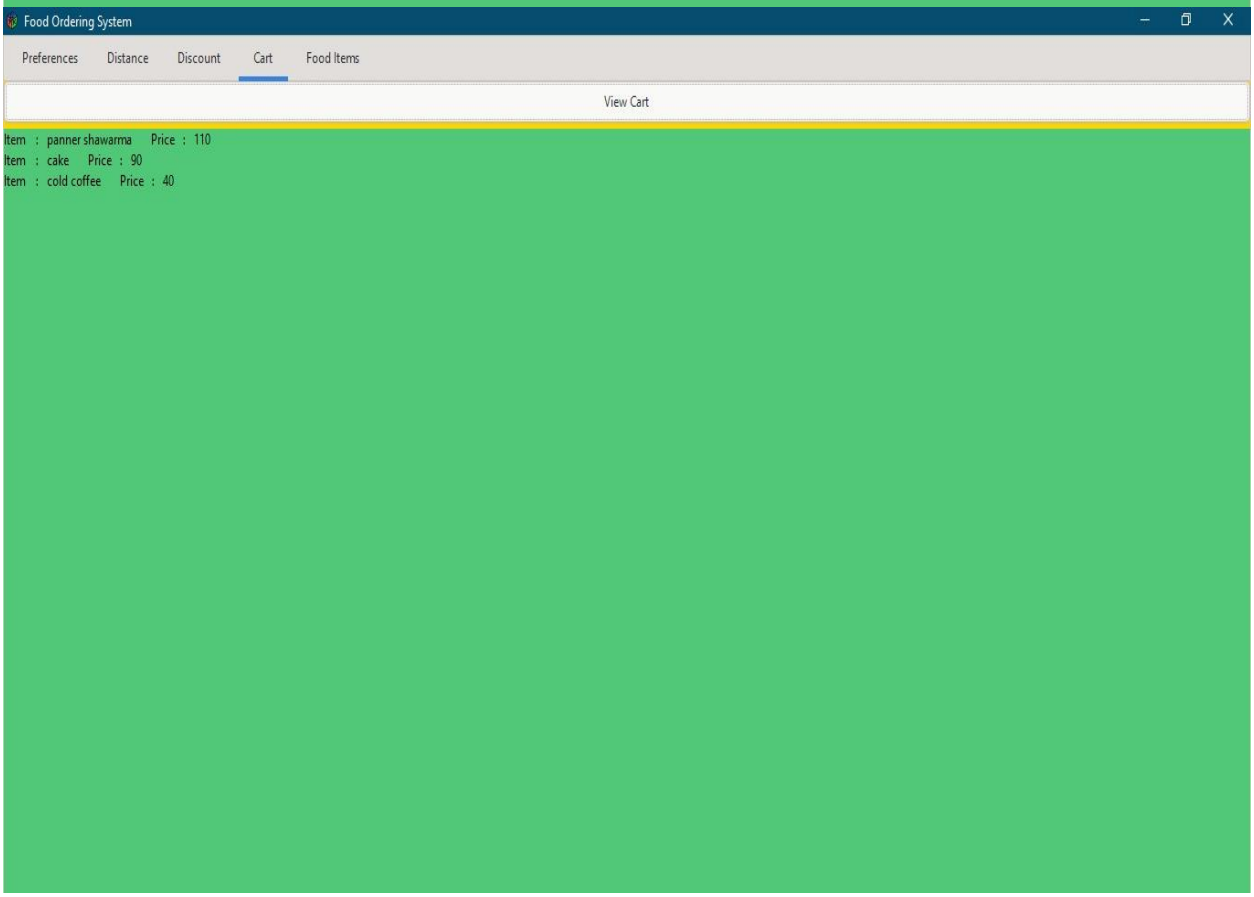
8.Validation through detailed test cases for various Scenarios:




Food Ordering System
Preferences
Distance
Discount
Cart
Food Items
Preference:
non-veg
Show Hotels
anjapar
buhari


Food Ordering System
Preferences
Distance
Discount
Cart
Food Items
Sort by Discount
Hotel Name: anandas, Discount: 17%
Hotel Name: delight, Discount: 16%
Hotel Name: buhari, Discount: 14%
Hotel Name: anjapar, Discount: 13%
Hotel Name: wow, Discount: 12%


Food Ordering System
Preferences
Distance
Discount
Cart
Food Items
Latitude:
13.25
Longitude:
80.48
Sort by Distance
Hotel Name: anandas, Distance: 23.86 km
Hotel Name: buhari, Distance: 27.99 km
Hotel Name: delight, Distance: 51.95 km
Hotel Name: wow, Distance: 113.63 km
Hotel Name: anjapar, Distance: 188.90 km


Food Ordering System
Preferences
Distance
Discount
Cart
Food Items
View Cart
Item : panner shawarma Price : 110
Item : cake Price : 90
Item : cold coffee Price : 40

10.Observation from the social, Legal, environmental and Ethical perspectives:

Social Perspective

- Access Issues: Restaurant owners' inability to directly manage their data may hinder operational efficiency.
- Payment Limitations: Credit-based payment system limits user options, impacting convenience.
- Order Tracking: Lack of real-time order status updates may frustrate users expecting transparency.

Legal Perspective

- Data Protection: Compliance with laws like GDPR is crucial to protect user information.
- Financial Regulations: Even credit systems require adherence to financial laws to prevent fraud.
- Contractual Clarity: Clear agreements are needed between admins and restaurants on data management.

Environmental Perspective

- Resource Efficiency: Minimizing energy use and promoting paperless operations can reduce environmental impact.
- Sustainable Practices: Encouraging eco-friendly delivery methods supports environmental sustainability.

Ethical Perspective

- Transparency: Admin actions regarding data updates should be transparent to avoid bias.
- Fairness in Payment: Ethical concerns arise with credit-based systems that limit payment choices.
- User Rights: Ensuring informed consent, data access, and fair dispute resolution are essential ethical considerations.

Addressing these aspects will enhance the system's fairness, compliance, and sustainability, fostering trust and user satisfaction.

11.Learning Outcomes:

i. We able to understand how to implement and develop an application for food delivery that facilitates ordering for food from restaurants of different categories at different locations. A user can order for the food through this application to any preferred delivery address.
ii. We know now how to read and write on a file using c program and also to store this information in array of structures to manipulate the data to build a effective application that has good user-friendly interface
iii. We also learnt to handle the error and input validation is carried out properly.
iv. We learnt how to integrate file and module to build a full-fledged application.

12.Limitation:

1) The login interface is command-line based, while the main application uses GTK+, creating an inconsistent user experience.

There's no option to log out or switch users within the application.

2) The application doesn't support multiple languages or locales.

3)Only cash on delivery option available

12.Refference:

FOR MAKING DFDS:

https://www.smartdraw.com/data-flow-diagram/data-flow-diagram-software.htm

FOR RESEARCHING CONSTARINTS:

https://www.geeksforgeeks.org/

FOR UNDERSTADING SOCIAL IMPACT:

https://iaeme.com/MasterAdmin/Journal_uploads/JOM/VOLUME_5_ISSUE_5/JOM_05_05_0 15.pdf

FOR ODER FOOD:

https://www.zomato.com/mobile https://www.swiggy.com/city/chennai