DSA SEMESTER PROJECT

Submitted to: Mr. Shakeel Ahmed

Submitted by: 4782-FOC/BSSE/F23B-Fahad Imran
Safeerullah Khan 4815 FOC BSSE F23 B
Hamza Zeeshan 4809 FOC BSSE F23 B
Reshail 4785 FOC BSSE F23 B

International Islamic University Islamabad

Department of Software Engineering

Project Report

Course: Data Structures and Algorithms (BSSE F23)
Institute: International Islamic University Islamabad (IIUI)
Project Title: Food Delivery System

Team Members:

- Fahad Imran 4782 FOC BSSE F23 B
- Safeerullah Khan 4815 FOC BSSE F23 B
- Hamza Zeeshan 4809 FOC BSSE F23 B
- Reshail 4785 FOC BSSE F23 B

1. Project Overview

The Food Delivery System is a console-based application written in C++, designed to provide a simplified simulation of a food ordering and delivery platform. Users can view the menu, search and sort food items, place orders, and track deliveries. This project integrates core concepts of Data Structures and Algorithms (DSA), focusing on real-world application, input validation, and data handling.

2. Objectives

- To implement core DSA concepts in a practical project.
- To build a functional console-based application using C++.
- To apply searching and sorting algorithms effectively.
- To demonstrate the use of custom data structures like linked lists, queues, and stacks.
- To perform proper input validation and error handling.

3. System Architecture

The system architecture is modular, consisting of the following components:
- Menu Management Module: Handles adding, displaying, and sorting food items.
- Order Processing Module: Manages placing, updating, and tracking orders.

- User Interface: A menu-driven console interface for user interaction.
- Data Structures: Custom implementations of arrays, linked lists, queues, and optional stacks.

4. Key Features

- Add new food items to the menu with name, price, and category.
- Display the full menu in a formatted manner.
- Search for food items by name (Linear or Binary Search).
- Sort items by name or price using Bubble Sort, Selection Sort, or Merge Sort.
- Place an order and track it using a queue.
- Display recent actions using a stack (optional).
- Input validation with appropriate error messages, including checks for empty input.

5. Data Structures Used

Data Structure	Purpose
Linked List	Used to manage a dynamic list of food orders.
Array / Vector	Stores the static menu items.
Struct / Class	Represents entities such as FoodItem, Customer, and Order.
Queue	Maintains order processing and delivery tracking.
Stack (optional)	Tracks user actions or recently placed orders.

6. Searching Algorithms Used

- Linear Search: Traverses the menu to locate a food item by name.
- Binary Search: Efficiently finds items by name in a sorted list.

7. Sorting Algorithms Used

- Bubble Sort: Arranges menu items by price.
- Selection Sort: Sorts by alphabetical order of names.
- Merge Sort: Used for larger datasets to optimize performance.

8. Input Validation and Error Handling

Robust input validation is implemented across all input points:
- Validates that numeric inputs (like price and quantity) are correct.
- Handles invalid options in menus and prevents crashes from unexpected input.

9. User Interface

The system is entirely text-based, using a menu-driven console approach. All user prompts are clear, and the layout is designed to enhance readability.

10. Challenges Faced

- Implementing custom sorting algorithms without STL functions.
- Ensuring dynamic memory safety and preventing leaks.
- Designing modular code to separate logic for menu, order, and search features.
- Implementing efficient search/sort functions without performance bottlenecks.

11. Learning Outcomes

- Reinforced understanding of how different data structures behave.
- Practical experience with pointers and dynamic memory management.
- Improved ability to debug and validate complex input scenarios.
- Gained insights into algorithmic efficiency and the importance of optimization.

12. Future Enhancements

- Add persistent storage using file I/O or databases.
- Introduce a graphical interface for enhanced usability.
- Implement user authentication and role-based access (admin/customer).
- Enable delivery tracking with estimated times and mapping.
- Create a web-based version using modern frameworks.

13. Conclusion

This project served as a comprehensive exercise in applying the principles of Data Structures and Algorithms in a meaningful way. The Food Delivery System demonstrates how these core concepts can power real-life applications while reinforcing error handling, modular coding, and user interface design.

14. Pseudocode for Key Functionalities

Below is the pseudocode for some of the essential operations used in the Food Delivery System.

a. Binary Search (Search by Name)

```
function binarySearch(foodItems, targetName):
    low = 0
    high = foodItems.length - 1

    while low <= high:
        mid = (low + high) / 2
        if foodItems[mid].name == targetName:
            return mid
        else if foodItems[mid].name < targetName:
            low = mid + 1
        else:
            high = mid - 1
    return -1
```

b. Bubble Sort (Sort by Price)

```
function bubbleSort(foodItems):
    for i from 0 to foodItems.length - 1:
        for j from 0 to foodItems.length - i - 1:
            if foodItems[j].price > foodItems[j + 1].price:
                swap(foodItems[j], foodItems[j + 1])
```

15. Screenshots and Interface (To Be Added)

Screenshots of the application interface.
MENU

```
=== Food Delivery System Menu ===
1. Display Menu
2. Place Order
3. Track Order
4. View Order History
5. Process Orders
6. Exit
-------------------------------------
Enter your choice:
```

Food menu

ID	Name	Price	Category	Prep Time
DF6	Butter Naan	2.99	Desi Food	10 mins
DFB2	Chai	2.99	Desi Food Beverage	5 mins
BV3	Water	2.99	Beverage	5 mins
BV4	Soda	3.99	Beverage	5 mins
DFB1	Lassi	4.99	Desi Food Beverage	5 mins
D2	Ice Cream	4.99	Dessert	5 mins
BV2	Lemonade	4.99	Beverage	5 mins
DFD1	Gulab Jamun	5.99	Desi Food Dessert	5 mins
D3	Apple Pie	5.99	Dessert	5 mins
DFD2	Kheer	6.99	Desi Food Dessert	15 mins
D1	Chocolate Cake	6.99	Dessert	5 mins
BV1	Mango Juice	6.99	Beverage	5 mins
CH4	Spring Rolls	7.99	Chinese Food	15 mins
IT5	Tiramisu	8.99	Italian Dessert	10 mins
MX5	Guacamole & Chips	8.99	Mexican Food	10 mins
S1	Caesar Salad	8.99	Salad	10 mins
B1	Classic Burger	9.99	Burger	15 mins
S2	Greek Salad	9.99	Salad	10 mins
B3	Veggie Burger	10.99	Burger	15 mins
CH5	Fried Rice	11.99	Chinese Food	20 mins
MX4	Nachos Supreme	11.99	Mexican Food	15 mins
B2	Cheese Burger	11.99	Burger	15 mins
MX1	Beef Tacos	12.99	Mexican Food	20 mins
DF4	Seekh Kabab	12.99	Desi Food	15 mins
P1	Margherita Pizza	12.99	Pizza	20 mins
CH3	Chow Mein	13.99	Chinese Food	20 mins
MX2	Chicken Quesadilla	13.99	Mexican Food	20 mins
SF5	Calamari	13.99	Seafood	20 mins
P4	Vegetarian Pizza	13.99	Pizza	20 mins
IT1	Spaghetti Carbonara	14.99	Italian Food	25 mins
MX3	Beef Burrito	14.99	Mexican Food	25 mins
DF3	Chicken Tikka	14.99	Desi Food	20 mins
P2	Pepperoni Pizza	14.99	Pizza	20 mins
CH1	Kung Pao Chicken	15.99	Chinese Food	25 mins
IT2	Fettuccine Alfredo	15.99	Italian Food	25 mins
P3	BBQ Chicken Pizza	15.99	Pizza	20 mins
CH2	Sweet & Sour Pork	16.99	Chinese Food	25 mins
IT3	Lasagna	16.99	Italian Food	30 mins
SF2	Fish & Chips	16.99	Seafood	20 mins
DF1	Chicken Biryani	16.99	Desi Food	25 mins
IT4	Risotto	17.99	Italian Food	30 mins
DF5	Nihari	17.99	Desi Food	35 mins
SF3	Shrimp Scampi	18.99	Seafood	25 mins
DF2	Beef Karahi	18.99	Desi Food	30 mins
SF1	Grilled Salmon	19.99	Seafood	25 mins
SF4	Lobster Tail	29.99	Seafood	35 mins

Displaying previous orders:

```
Displaying all orders...
Order ID: ORD5026
Customer Name: Fahad
Total Amount: $0.00
Status: Pending
------------------------------
Order ID: ORD8713
Customer Name: Nashit
Total Amount: $12.99
Status: Pending
------------------------------
Order ID: ORD7717
Customer Name: fadi
Total Amount: $18.99
Status: Pending
------------------------------
Order ID: ORD7730
Customer Name: Fahad
Total Amount: $11.99
Status: Pending
------------------------------
Order ID: ORD6280
Customer Name: Fahad
Total Amount: $0.00
Status: Pending
------------------------------
Order ID: ORD9307
```

DIAGRAMS

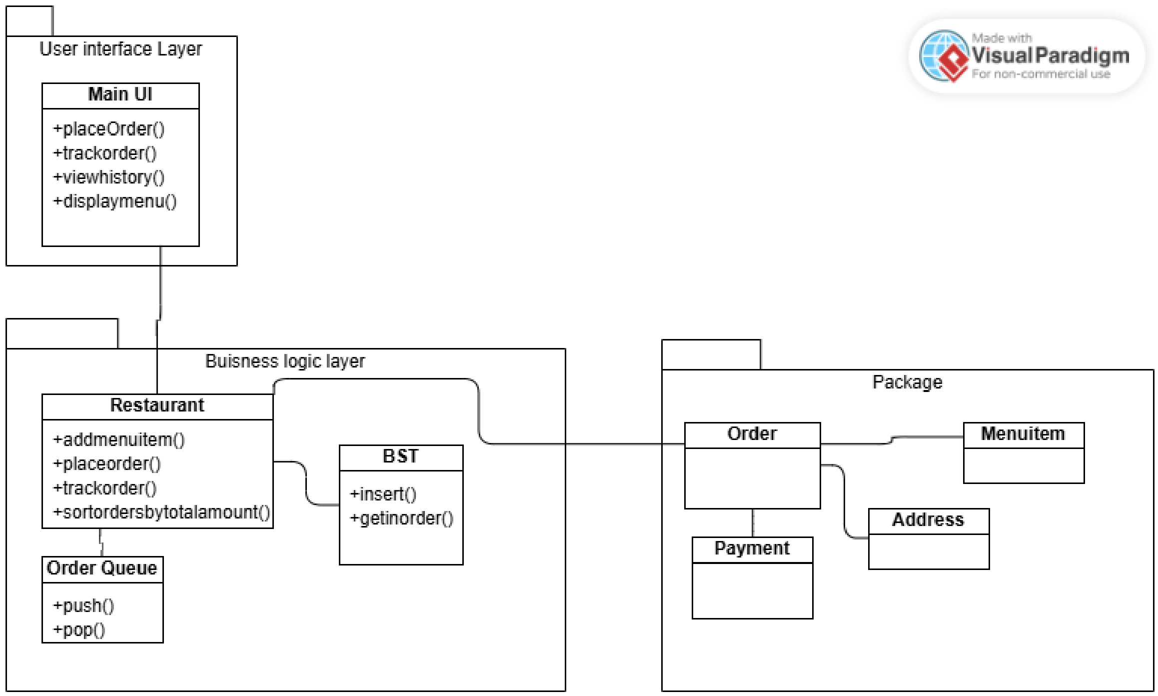


Figure 1 OOP DESIGN

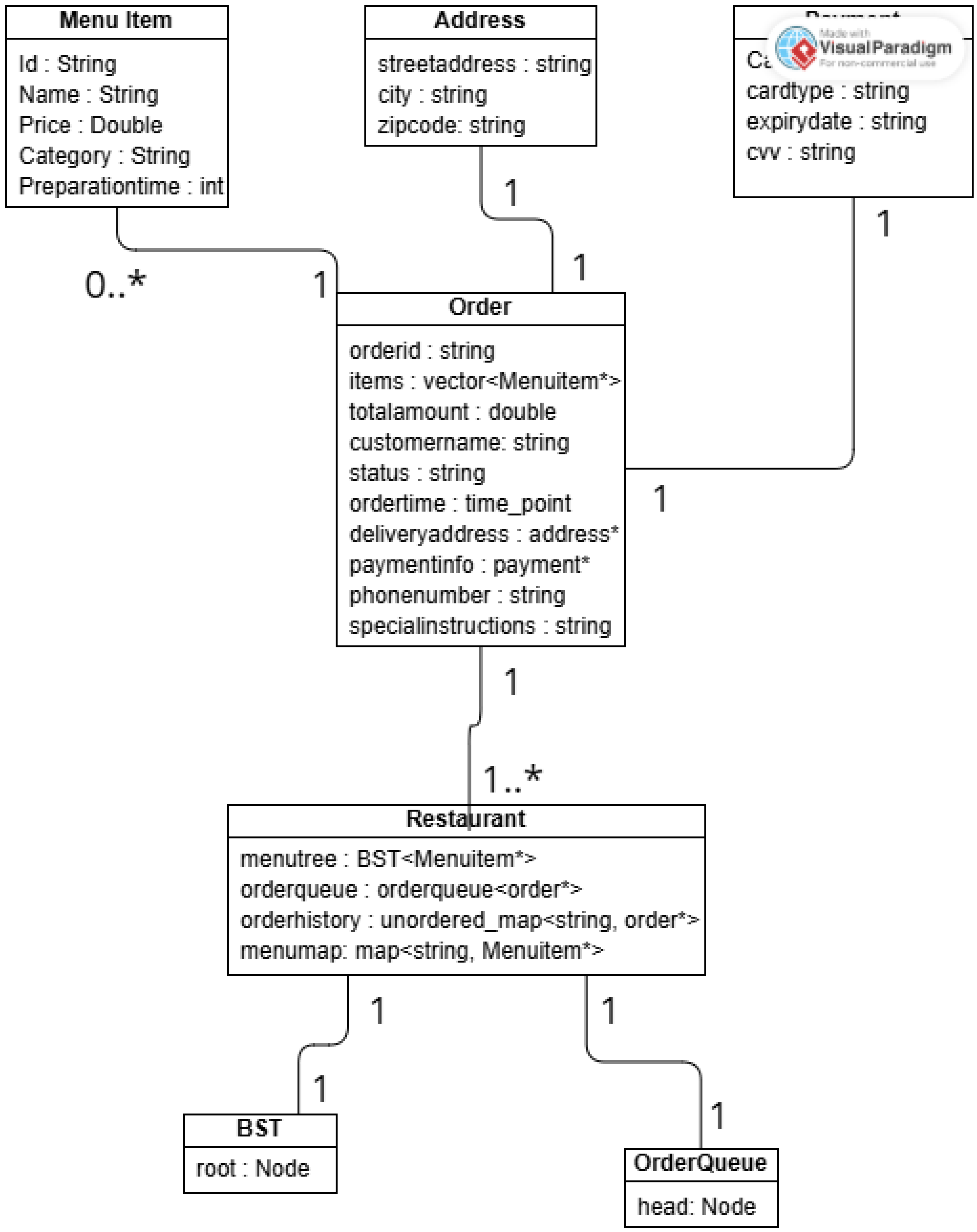


Figure 2 CLASS DIAGRAM

DSA PROJECT : STRUCTURE ANALYSIS

Tables & Chart Description

Our project uses different data structures and algorithms. The tables and chart help compare them in terms of speed and usefulness.

What the Tables Show:

- **Data Structures Table** shows how fast each structure is for adding and finding data.
  Example: Hash Map is very fast for searching orders.
- **Sorting Algorithms Table** compares how sorting methods work.
  Example: Merge Sort is stable and safe for big data, Quick Sort is faster but riskier.
- **Searching Algorithms Table** explains when to use Linear Search or Binary Search.
  Example: Linear Search works on any list, Binary Search only works on sorted lists but is faster.

Data Structure	Average Time	Worst Time	Space Complexity
BST<MenuItem*>	O(log n)	O(n)	O(n)
OrderQueue<Order*>	Push: O(n), Pop: O(1)	Push: O(n), Pop: O(1)	O(n)
unordered_map<string, Order*>	O(1)	O(n)	O(n)
map<string, MenuItem*>	O(log n)	O(log n)	O(n)
list<DeliveryDriver*>	O(1)	O(n)	O(n)

Algorithm	Average Time	Worst Time	Space	Stable	In-Place
Merge Sort	O(n log n)	O(n log n)	O(n)	Yes	No
Quick Sort	O(n log n)	$O(n^2)$	O(log n)	No	Yes
std::sort()	O(n log n)	O(n log n)	O(log n)	No	Yes

Algorithm	Time Complexity	Sorted Required	Use Case
Linear Search	O(n)	No	Unsorted Order ID Search
Binary Search	O(log n)	Yes	Sorted Order ID Search

Conceptual Efficiency Comparison

What the Chart Shows:

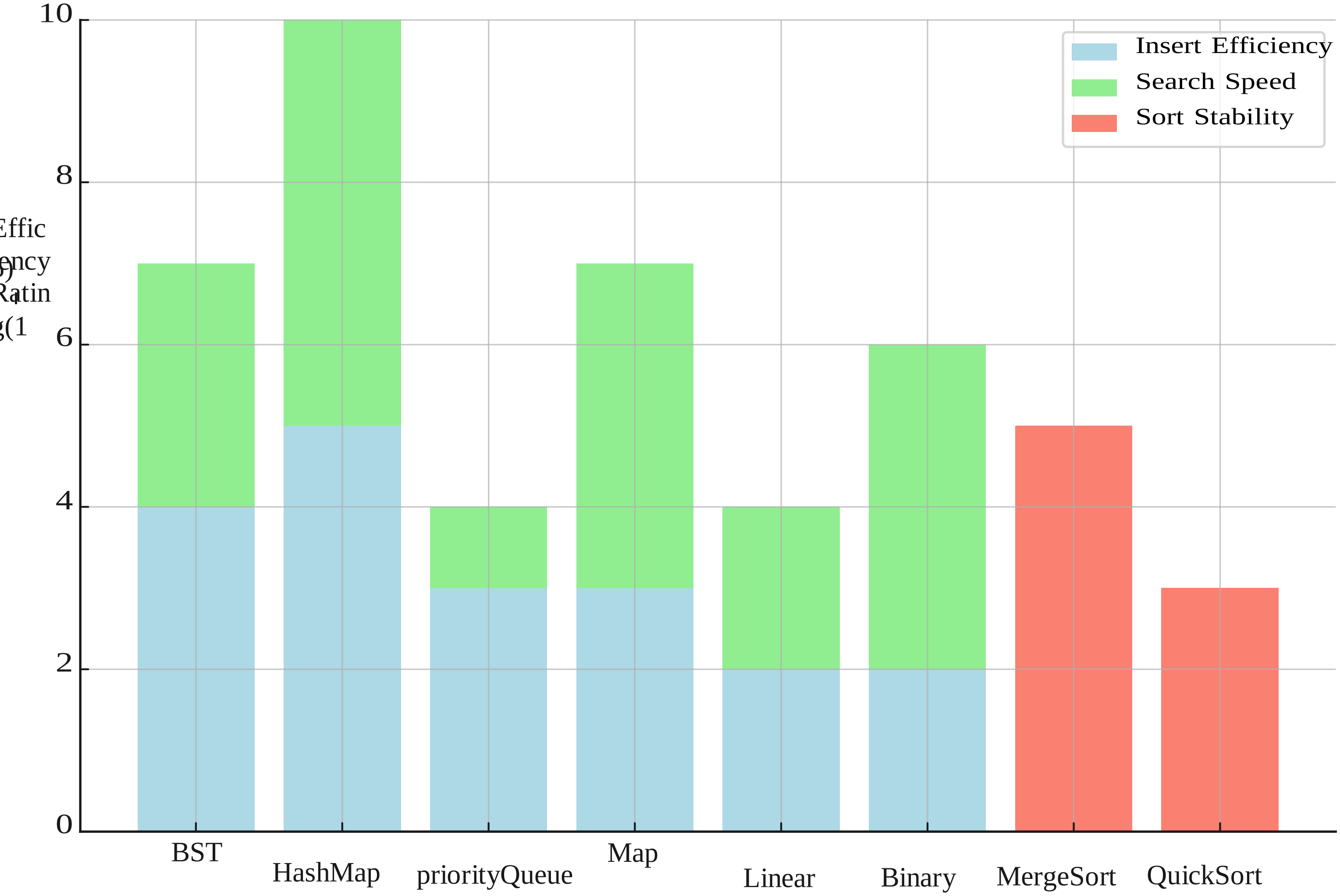
- It compares how each method performs in three ways:
  - Adding new data (Insert)
  - Searching data (Speed)
  - Keeping order while sorting (Stability)

Summary:

We chose each method because it fits the job:

- Fast search = Hash Map
- Sorted menu = BST
- Stable sorting = Merge Sort

These choices make the system faster and easier to use.


Effic
ency
Ratin
g(1
10
8
6
4
2
0
Insert Efficiency
Search Speed
Sort Stability
BST
HashMap
priorityQueue
Map
Linear
Binary
MergeSort
QuickSort