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

# 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

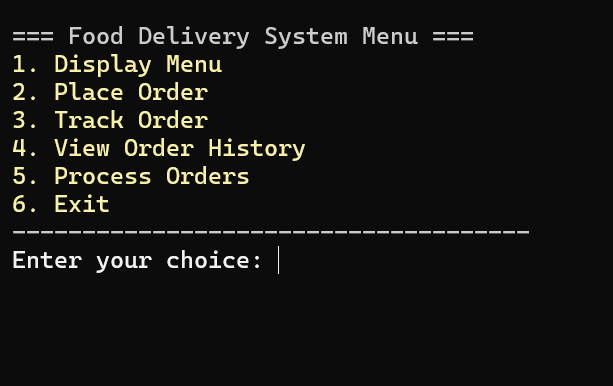
## b. Bubble Sort (Sort by Price)

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 menu



Displaying previous orders:

A screenshot of a computer

AI-generated content may be incorrect.