UE24CS252A DSA
Mini-Project

Problem Statement: Food Delivery Platform Simulation– Model orders, restaurant preparation, and delivery assignments to minimize delivery time

Objective:

This project simulates real-world food delivery operations - from order placement to final delivery - using efficient data structures.

Name	SRN
Ibrahim khaleel	PES2UG25CS809
Deeraj S	PES2UG24CS806
Gokul	PES2UG25CS246

Data Structures

1. Queue (Pending & Ready Orders)

Used For:

Pending orders waiting for preparation

Ready orders waiting for agent assignment

Why Queue?

First-Come-First-Serve (FCFS) matches real food order workflow

O(1) enqueue and dequeue O(1)

Natural fit for maintaining order sequence

2. Binary Search Tree (In-Transit Orders)

Used For:

Active delivery orders stored by order_id

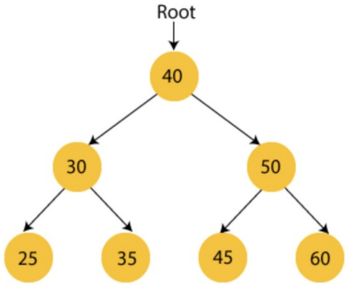
Searching and displaying sorted data by ETA

Why BST?

Efficient search (O(log n)), insert (O(log n))

Inorder traversal provides sorted listing of deliveries by ETA

Ideal for tracking active deliveries efficiently



Abstract Data Type: Queue (Linear Data Structure)

Structures

```
typedef struct Order {
    int order_id;
    char restaurant[32];
    char customer[32];
    int prep_time;
} Order;

typedef struct QNode {
    Order data;
    struct QNode *next;
} QNode;

typedef struct Queue {
    QNode *front;
    QNode *rear;
    int size;
} Queue;
```

Operations

1. queue_create()

Description: Creates an empty queue and initializes front/rear pointers to NULL

Time Complexity: O(1)

2. queue_enqueue(Queue q, Order data)

Description: Adds an order to the rear of the queue and updates size

3. queue_dequeue(Queue q)

Description: Removes and returns order from the front of queue, updates size

Time Complexity: O(1)

4. queue_is_empty(Queue q)

Description: Checks if queue has no elements

Time Complexity: O(1)

Abstract Data Type: Binary Search Tree

Structures

```
typedef struct Delivery {
    int order_id;
    int agent_id;
    int eta;  // Estimated Time of Arrival (in minutes)
    char status[16];
} Delivery;
typedef struct BSTNode {
    Delivery data;
    struct BSTNode *left;
    struct BSTNode *right;
} BSTNode;
typedef struct BST {
    BSTNode *root; }
```

Operations

1. bst_create()

Description: Creates an empty Binary Search Tree.

Time Complexity: O(1)

2. bst_insert(tree, data)

Description: Inserts a delivery record while maintaining BST order (order_id as key).

Time Complexity: O(log n) average, O(n) worst case

3. bst_search(tree, order_id)

Description: Searches for delivery by order_id.

Time Complexity: O(log n) average, O(n) worst case

4. bst_inorder(tree)

Description: Traverses the BST in sorted order (Left → Root → Right).

Time Complexity: O(n)

Platform Operations

1. paceOrder()

Purpose: Adds a new order to the system.

Outcome: Order waits to be assigned and cooked.

2. prepareOrder()

Purpose: Moves to ready queue.

Outcome: Order waits to be assigned .

3. assignDeliveryAgent()

Purpose: Moves an order from ready queue to the in-transit BST by assigning an available agent.

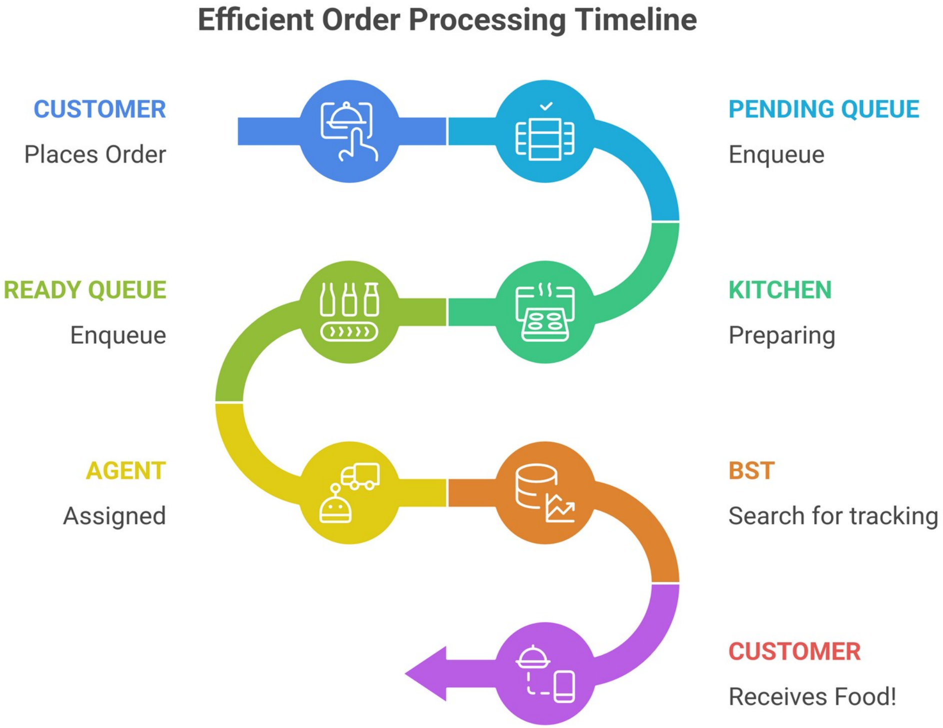
Outcome: Order status becomes “In Transit”.

3. trackOrder(order_id)

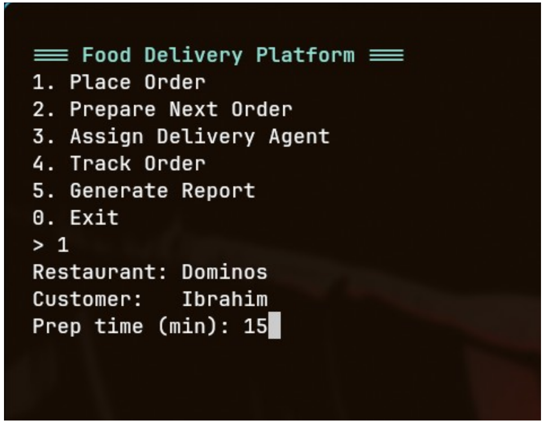
Purpose: Shows current status of an order.

4. generateReport()

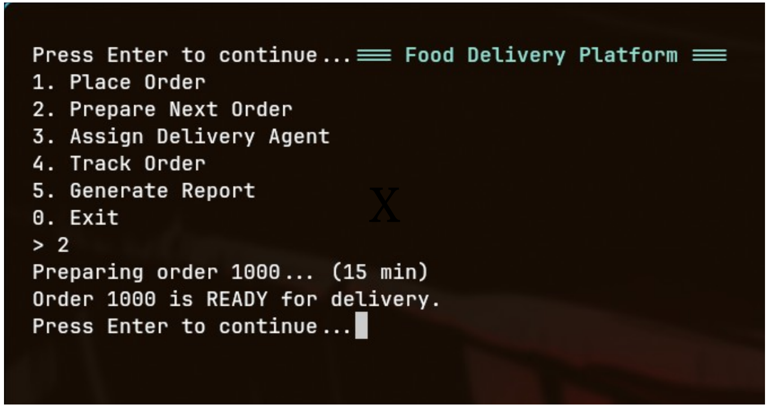
Purpose: Gives a complete operational overview.



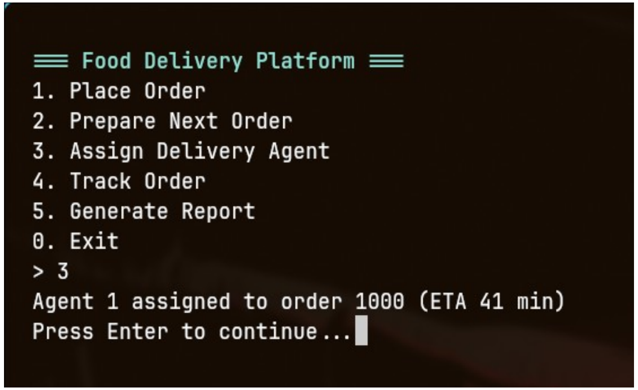
Inputs & OutPuts



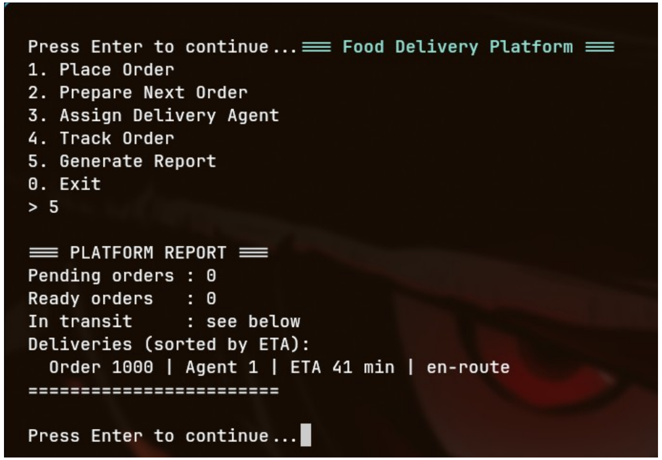
```
══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 1
Restaurant: Dominos
Customer:   Ibrahim
Prep time (min): 15
```



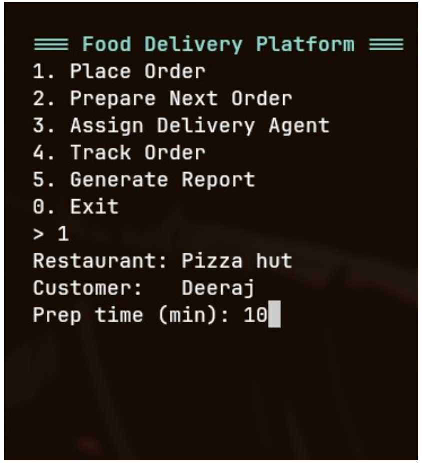
```
Press Enter to continue...══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 2
Preparing order 1000... (15 min)
Order 1000 is READY for delivery.
Press Enter to continue...
```



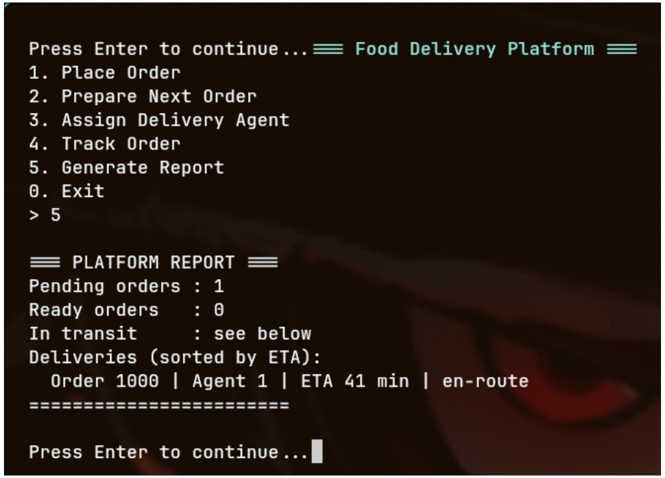
```
══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 3
Agent 1 assigned to order 1000 (ETA 41 min)
Press Enter to continue...
```



```
Press Enter to continue...══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 5

══ PLATFORM REPORT ══
Pending orders : 0
Ready orders   : 0
In transit     : see below
Deliveries (sorted by ETA):
  Order 1000 | Agent 1 | ETA 41 min | en-route
========================

Press Enter to continue...
```



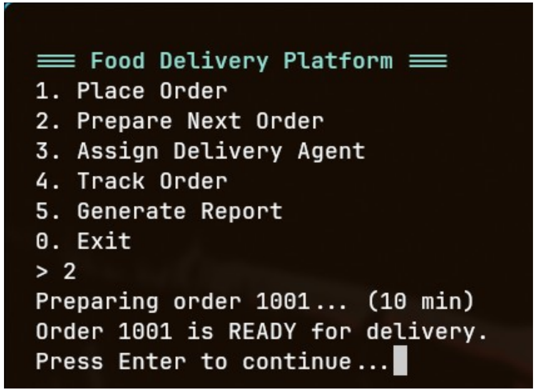
```
══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 1
Restaurant: Pizza hut
Customer:   Deeraj
Prep time (min): 10
```



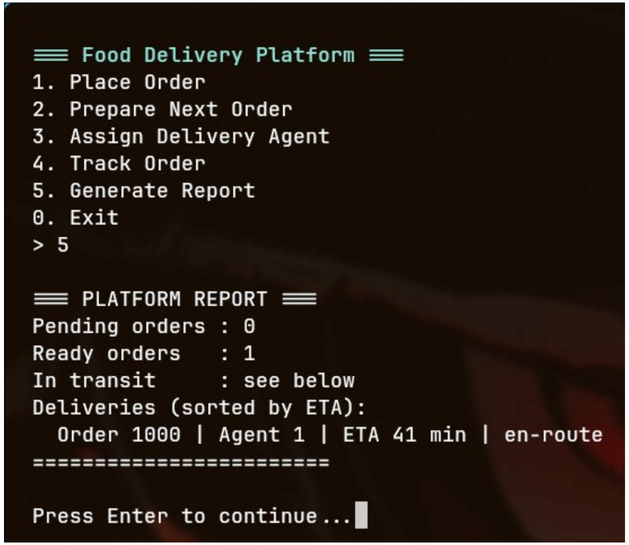
```
Press Enter to continue...══ Food Delivery Platform ══
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 5

══ PLATFORM REPORT ══
Pending orders : 1
Ready orders   : 0
In transit     : see below
Deliveries (sorted by ETA):
  Order 1000 | Agent 1 | ETA 41 min | en-route
========================

Press Enter to continue...
```

Food Delivery Platform Simulation

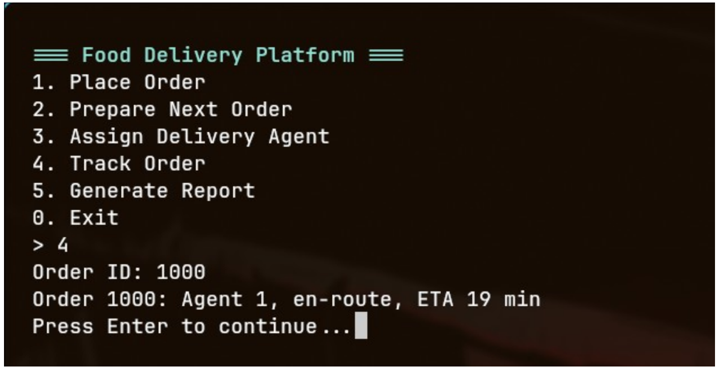
Inputs & OutPuts



```
=== Food Delivery Platform ===
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 2
Preparing order 1001... (10 min)
Order 1001 is READY for delivery.
Press Enter to continue...
```



```
=== Food Delivery Platform ===
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 5

=== PLATFORM REPORT ===
Pending orders : 0
Ready orders   : 1
In transit     : see below
Deliveries (sorted by ETA):
  Order 1000 | Agent 1 | ETA 41 min | en-route
========================

Press Enter to continue...
```



```
=== Food Delivery Platform ===
1. Place Order
2. Prepare Next Order
3. Assign Delivery Agent
4. Track Order
5. Generate Report
0. Exit
> 4
Order ID: 1000
Order 1000: Agent 1, en-route, ETA 19 min
Press Enter to continue...
```

THANK YOU!

Division of Work:

Ibrahim Khaleel: BST implementaion + PPT
Deeraj: Queue implementaion
Gokul: Functions