

**To Analyse Data Structures and Algorithms  
in the ZOMATO application**

**A PROJECT ACTIVITY REPORT**

**21CSC201J – DATA STRUCTURES AND ALGORITHMS**

**LABORATORY**

**(2021 Regulation)**

**II Year/ III Semester**

**Academic Year: 2023 -2024**

**By**

**Shayan Dey – RA2211028010135**

**Lavanya Pakhale – RA2211028010132**

**T. Vinay Kumar – RA2211028010123**



**DEPARTMENT OF NETWORKING AND COMMUNICATIONS**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR - 603203**

**NOVEMBER 2023**



**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY  
KATTANKULATHUR**

**603203**

**BONAFIDE**

Certified that this Project activity report titled “**To analyse Data Structures and Algorithms in the Zomato Application**” for the course **21CSC201J – DATA STRUCTURES AND ALGORITHMS LABORATORY** is the bonafide work of **Shayan Dey (RA2211028010135), Lavanya Pakhale (RA2211028010132) and T.Vinay Kumar (RA2211028010123)** who undertook the task of completing the project activity within the allotted time.

**Signature**

Dr. K. Deepa Thilak

**Course Faculty**

Associate Professor

Department of NWC

**Signature**

Dr. Annapurani K

**Head of the Department**

Professor

Department of NWC

## TABLE OF CONTENTS

Section No	Title	Page No.
1	Introduction	4
2	Problem Definition	5
3	Problem Explanation with diagram and example	6-15
4	Data structure used	16-21
5	Demo/ Simulation Output Screenshots	22-25
6	Conclusion	26

## Introduction

The Zomato app has become an integral part of the modern dining experience, revolutionizing how individuals and groups discover, order, and enjoy food. In a rapidly evolving food delivery and restaurant discovery landscape, the efficient functioning of the Zomato app is crucial. As we delve into the realm of data structures and algorithms, it becomes evident that these computational elements play a pivotal role in enhancing the user experience, improving efficiency, and ensuring the seamless operation of the app.

Analyzing data structures and algorithms within the Zomato app is not merely an academic exercise; it is a practical exploration of how cutting-edge technology enhances our daily lives. The importance of this analysis lies in its potential to uncover the hidden mechanisms that power this widely used platform, providing insights into the intricacies of its features and functions.

Objectives of the Project:

To comprehend the underlying data structures and algorithms that drive Zomato's core functionalities.

To dissect specific features of the Zomato app and illustrate how data structures and algorithms are applied to address real-world challenges.

To highlight the significance of these computational elements in creating a user-friendly and efficient platform.

To explore the hypothetical feature of "Group Order and Split Bill" and understand how data structures and algorithms might be integrated into such a feature.

By achieving these objectives, this project seeks to shed light on the intricate workings of a leading food delivery app and demonstrate the central role played by data structures and algorithms in our daily interactions with technology.

## **Problem Definition**

The primary problem addressed by this project revolves around the intricate technological ecosystem that underlies the Zomato app, and specifically, how data structures and algorithms are harnessed to tackle real-world challenges. This investigation delves into the complexities of a multifaceted platform used by millions daily, aiming to identify and elucidate the following key issues:

**Efficient Functionality:** The Zomato app is a dynamic platform where users interact with an extensive database of restaurants, menus, and orders. The project aims to define the problem of maintaining efficient functionality while catering to diverse user preferences and demands.

**User Experience:** Ensuring a seamless and intuitive user experience is central to the Zomato app's success. The problem is to maintain high user satisfaction and ease of use in the face of a vast array of data and complex algorithms.

**Real-Time Decision-Making:** With orders streaming in and restaurant data constantly changing, the app must make quick, real-time decisions. The problem is to address the need for instant decision-making and maintain an up-to-date user experience.

**Scalability:** As the user base grows and more restaurants and menu items are added, the problem is to create a scalable platform that can handle increased data volume without compromising performance.

By defining and addressing these problems, this project will provide insights into how data structures and algorithms are applied to optimize the Zomato app's functionality, enhance user experience, and ensure efficient real-time decision-making in a highly dynamic environment.

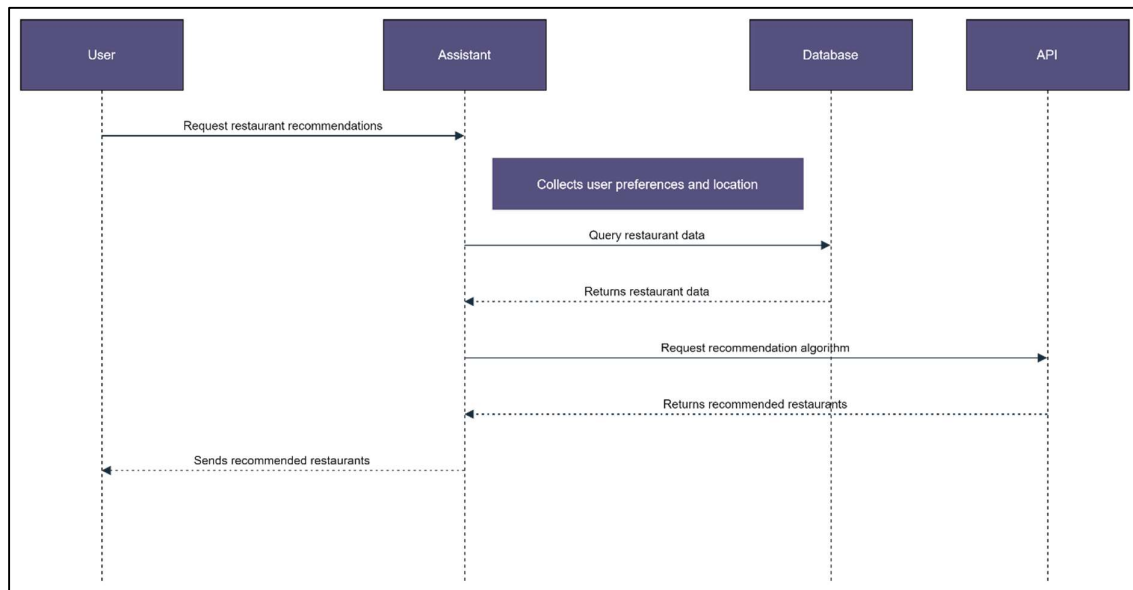
## Restaurant Search and Recommendations:

### Problem Explanation:

In the realm of food delivery apps, one of the foremost challenges is helping users discover and select suitable restaurants from a vast and ever-growing list of options. This challenge is compounded by the diverse and often unique preferences of users, making it essential to streamline and personalize the restaurant discovery process.

The problem at hand can be summarized as follows: How can we efficiently help users find restaurants that match their specific tastes and requirements, making their dining experience not just convenient, but also delightful?

### Diagrams:



### Example Scenario:

Let's delve into an example scenario to understand this problem better. Imagine a Zomato user in a bustling city, craving a delightful Italian meal. The user opens the app and, through the app's user-friendly interface, enters their preferences: "Italian cuisine," "within a 5-mile radius," and "open now." In response, the app's algorithms swiftly sift through a vast database of restaurants, filtering those that match the user's criteria. The user is then presented with a list of Italian restaurants that meet their preferences, complete with ratings, reviews, and real-time availability.

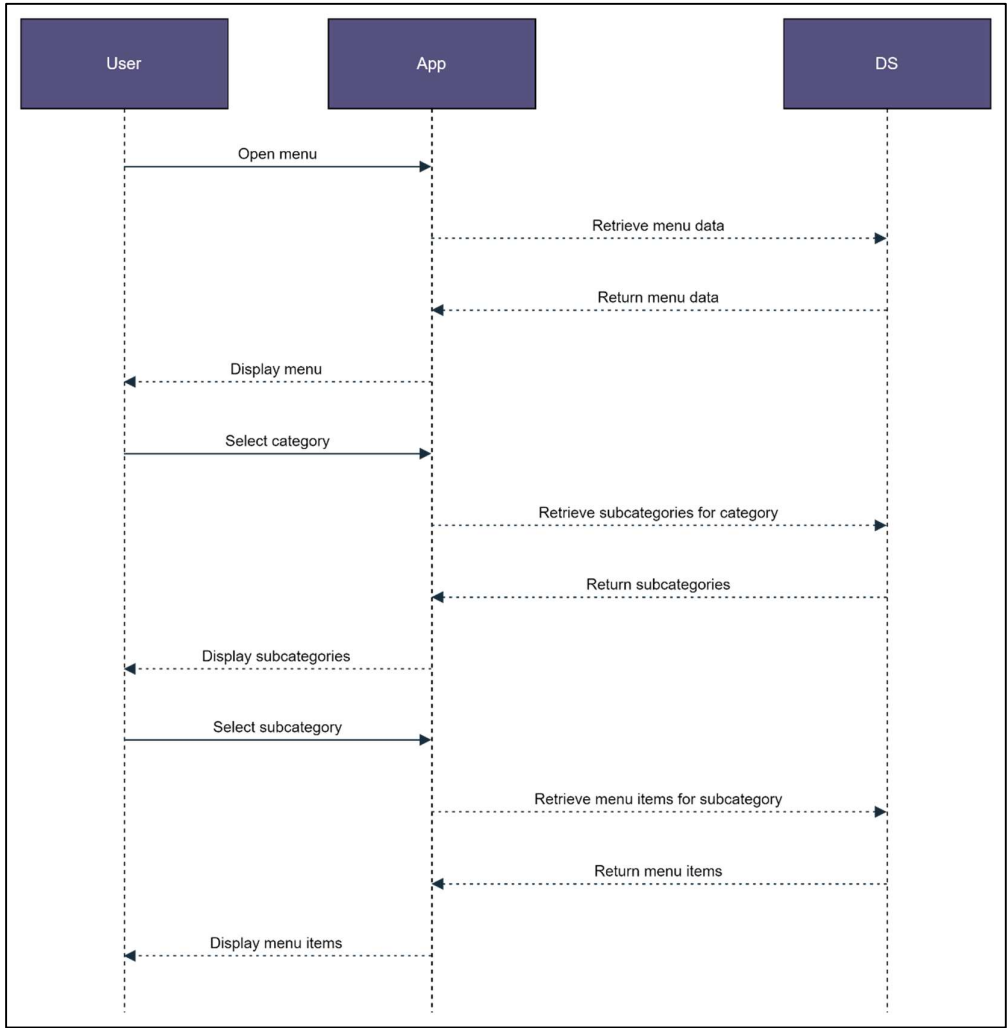
**Menu Browsing:**

**Problem Explanation:**

Efficiently navigating and exploring restaurant menus is a central challenge in food delivery apps. Users need to effortlessly browse through a multitude of menu items, making it essential to structure and present menus in a user-friendly and organized manner.

The problem at hand can be summarized as follows: How can we optimize the process of menu browsing, ensuring that users can easily explore menu categories, subcategories, and individual items within a restaurant's menu?

**Diagrams:**



### Example Scenario:

Let's explore an example scenario to understand this problem better. Imagine a Zomato user looking for a specific type of pizza at a local pizzeria. The user opens the app and navigates to the restaurant's menu. Instead of facing a long, unorganized list of items, the user is presented with a structured menu that starts with categories like "Appetizers," "Pizzas," "Pastas," and "Desserts." Within the "Pizzas" category, there are subcategories like "Vegetarian" and "Non-Vegetarian." The user selects the "Vegetarian" subcategory and quickly finds the desired pizza item. This efficient menu organization enhances the user's experience.



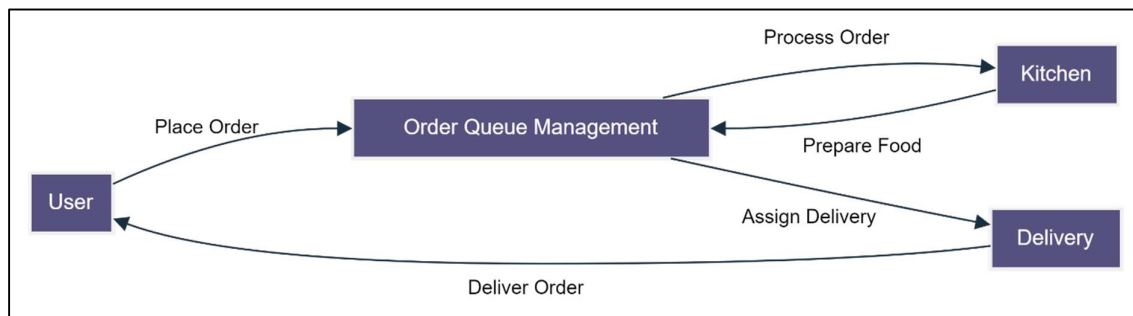
## Order Queue:

### Problem Explanation:

Efficiently managing user orders and organizing them in a queue is a critical challenge in food delivery apps like Zomato. As numerous users place orders simultaneously, it becomes essential to prioritize and streamline the process of order processing. The primary problem revolves around effectively managing the influx of orders and ensuring that each order is processed promptly and accurately.

In essence, the problem can be defined as: How can we optimize the handling of user orders, organize them in a queue, and ensure that orders are processed in a timely and organized manner?

### Diagrams:



### Example Scenario:

Let's consider an example scenario to better comprehend this problem. During the busy evening hours, a popular restaurant partnered with Zomato receives a surge of food orders from users. These orders are placed for a variety of dishes and are continuously coming in. The restaurant employs an order queue system where each order is added to the queue as it's received. The orders are prioritized based on factors like order time, proximity to delivery locations, and the complexity of the dishes. As the restaurant's kitchen staff prepare each order, it is marked as "in progress" and then "completed" when ready for delivery. This organized queue ensures that every order is processed efficiently, and users receive their food on time.

**Payment Processing:**

**Problem Explanation:**

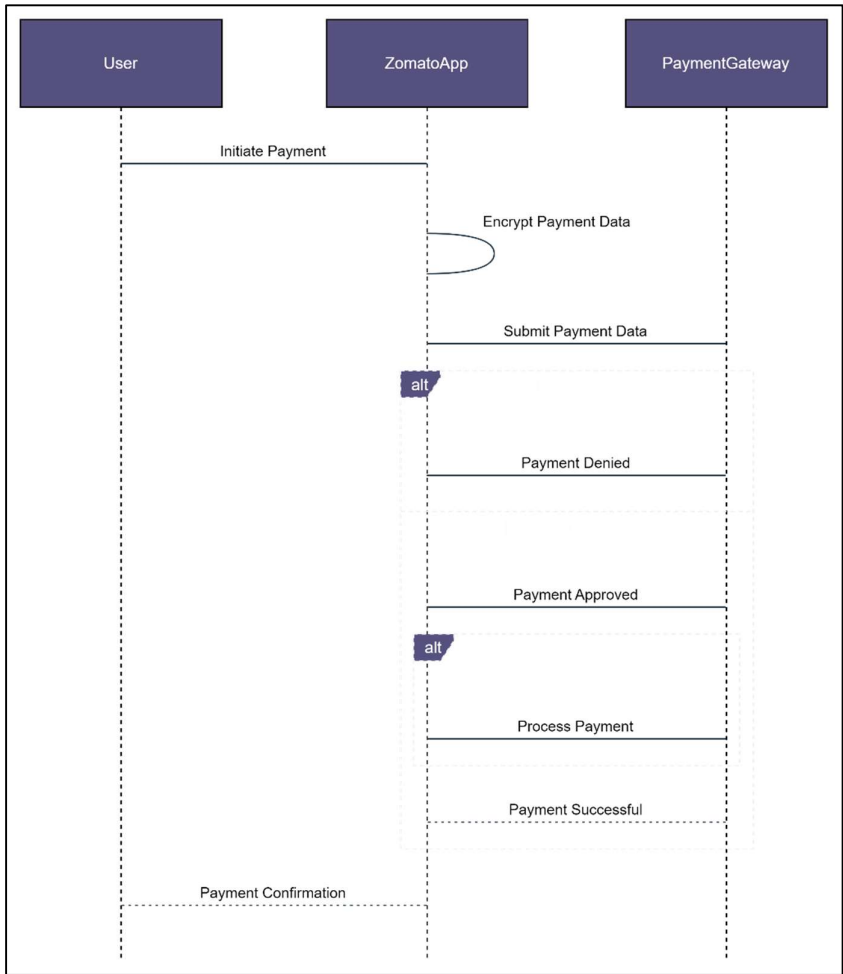
Efficiently handling payments in a secure manner is a crucial concern for Zomato and other food delivery apps. Ensuring that user payments are processed smoothly and securely is essential to provide a seamless and trustworthy user experience.

The primary problem can be defined as follows: How can Zomato handle payments securely and efficiently, guaranteeing that financial transactions are protected from potential risks?

**Payment Algorithms:**

To address this issue, Zomato implements advanced payment algorithms. They can include encryption techniques, fraud detection mechanisms, and secure payment gateways.

**Diagrams:**



### Example Scenario:

Consider an example scenario to illustrate this problem. A Zomato user has selected their favorite restaurant, added items to the cart, and is ready to make a payment. They proceed to the checkout and enter their payment details, including credit card information. The payment algorithm within the Zomato app encrypts this sensitive data, ensuring that it is transmitted securely. The algorithm also checks for any potential fraudulent activities during the transaction, such as unusual spending patterns. Once the payment is approved, the algorithm ensures that the funds are transferred correctly to the restaurant and sends a payment confirmation to the user. This scenario showcases how payment algorithms within Zomato work to process payments securely and efficiently.

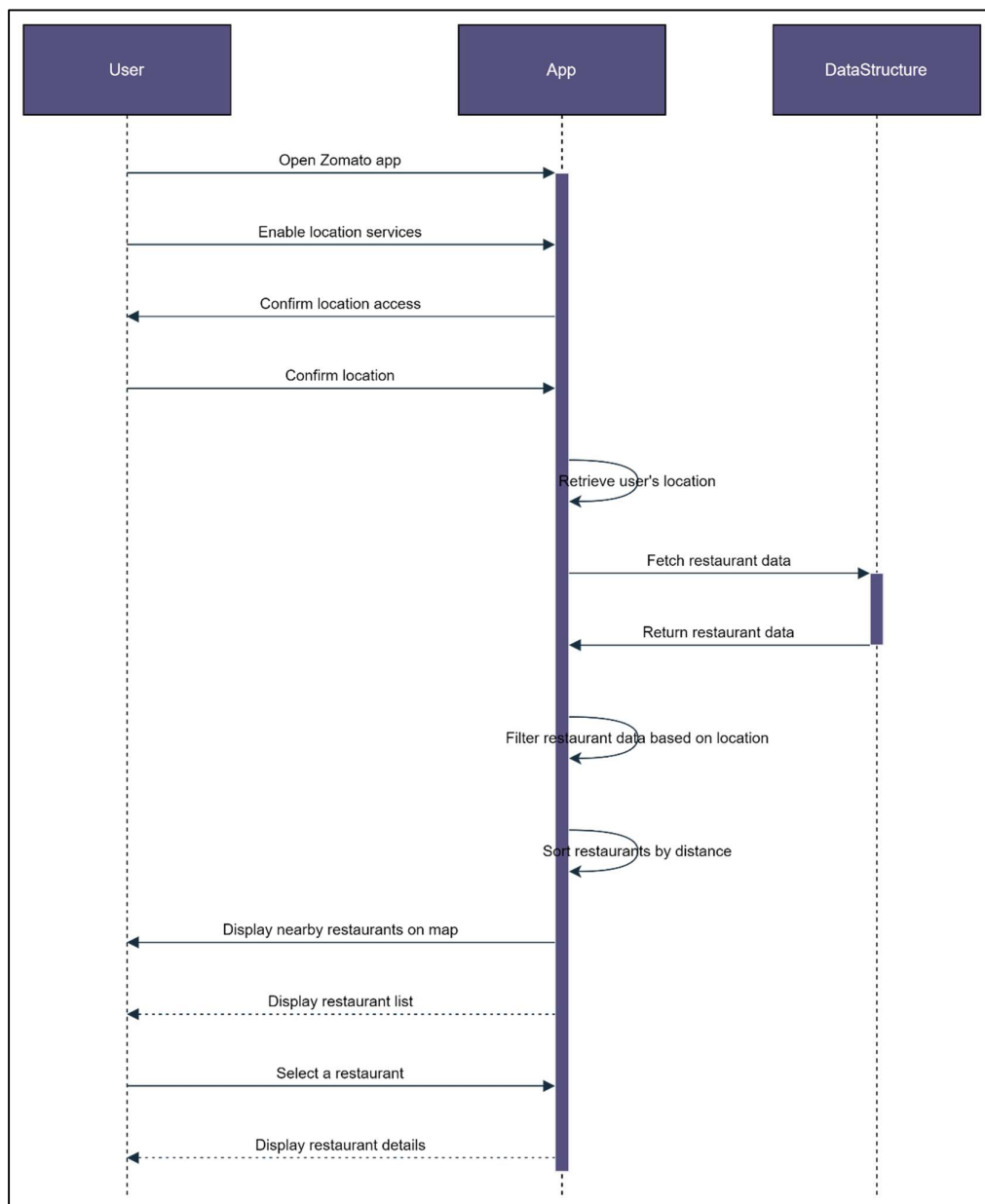
## Location-Based Searching:

### Problem Explanation:

Finding restaurants based on a user's location is a fundamental challenge for Zomato. Users often seek nearby dining options, and the app must efficiently and accurately match user locations with relevant restaurants.

The problem can be succinctly defined as follows: How can Zomato effectively locate and recommend restaurants to users based on their current location?

### Diagrams:



### Example Scenario:

To better understand this problem, let's consider an example scenario. A Zomato user opens the app while in a new city. The app detects the user's current location using GPS or user input. The location-based data structure within Zomato quickly identifies restaurants nearby and presents them to the user. In this scenario, the user is delighted to discover local dining options without needing to search extensively. This showcases how Zomato effectively addresses the challenge of location-based searching.

## **New Hypothetical Feature: Group Order and Split Bill:**

### **Introduction:**

Innovation and user-centric features are at the core of Zomato's mission. Introducing a new hypothetical feature, "Group Order and Split Bill," aims to further enhance the user experience by simplifying the process of ordering food with friends and fairly splitting the bill. This feature brings social dining to the digital realm, enabling users to place group orders and seamlessly divide the costs among friends.

### **Problem Explanation:**

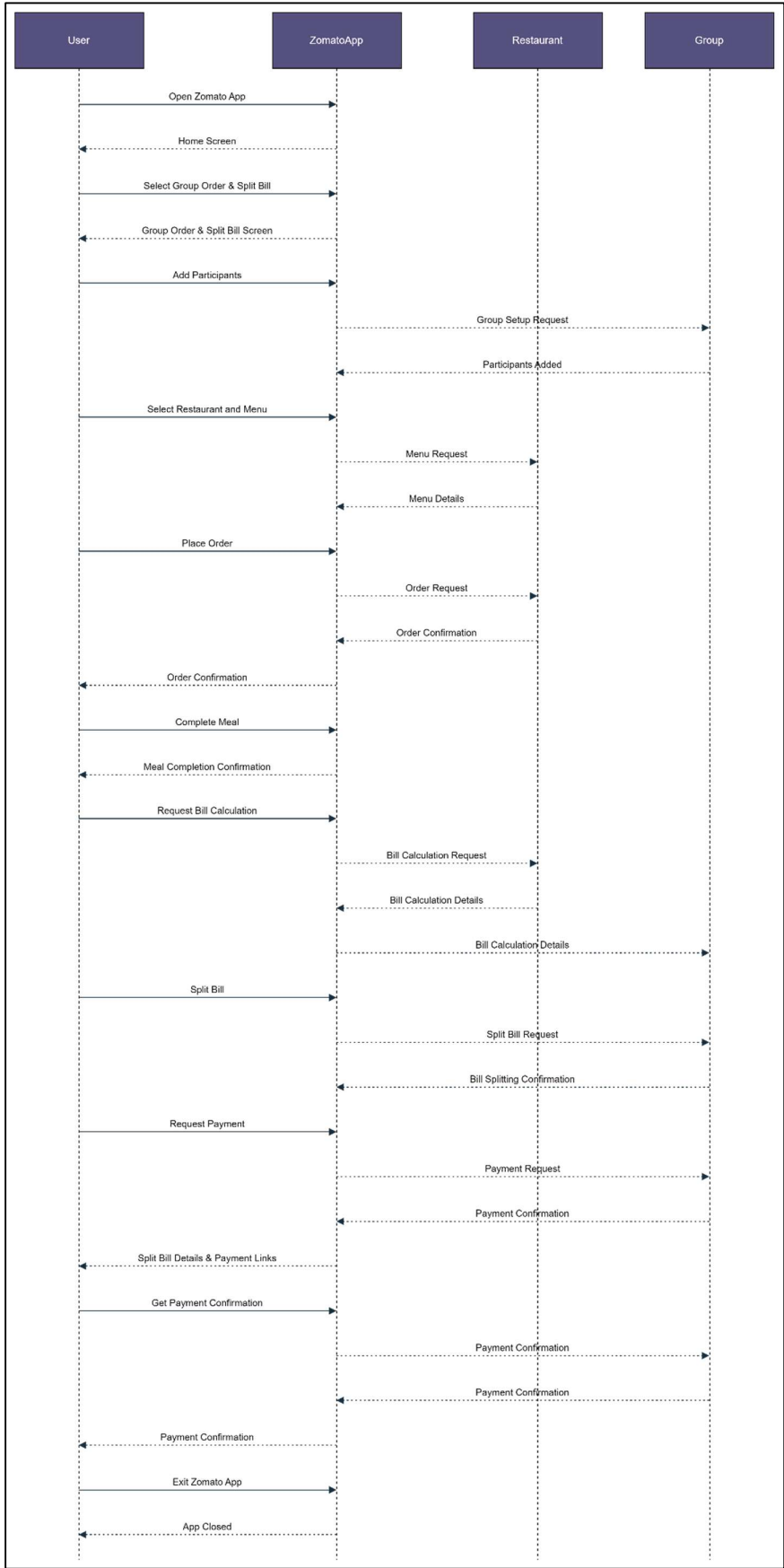
Managing group orders and effectively splitting bills is often a complex and time-consuming endeavor, both in dining establishments and within food delivery apps.

The problem to be tackled by this new feature can be outlined as follows: How can Zomato simplify and streamline the process of placing group orders, sharing the costs, and ensuring that each participant pays their fair share?

### **Real-World Scenario:**

To illustrate the importance of this feature, let's delve into a real-world scenario. A group of friends gathers for a fun evening and decides to order a variety of dishes from different restaurants via the Zomato app. Each friend selects their preferred items, and the order is placed collectively. With the "Group Order and Split Bill" feature, the app automatically calculates the total bill and divides it equally among all participants. The friends receive payment requests for their respective shares, and they can settle the bill promptly using their preferred payment methods. This scenario exemplifies how the new feature simplifies the process of group dining, making it convenient and equitable for all involved.

Diagram:



**Data Structures used:**

1) Feature: Restaurant Search and Recommendation

2) Data Structures: Hash Tables

3) Brief info about this Data Structure:

A hash table, also known as a hash map, is a data structure used for efficient data retrieval. It stores key-value pairs and allows you to quickly access values associated with a given key.

4) Implementation:

**Restaurant Information Management:** Hash tables can be used to store essential information about each restaurant, such as its name, location, cuisine type, ratings, and customer reviews. For example, each restaurant's name can be used as a key in the hash table, and the corresponding value might be an object or a data structure containing detailed information about the restaurant.

**Quick Retrieval of Restaurant Details:** When a user searches for a specific restaurant by name or location, the application can use a hash function to quickly locate the relevant entry in the hash table, thereby enabling fast retrieval of the restaurant's details. This ensures that users can access comprehensive information about a restaurant, including its menu, customer reviews, and location, in a timely manner.

**Customized Recommendations:** Hash tables can also be employed to store user preferences and historical data related to their restaurant choices and reviews. The application can use these stored preferences to generate customized recommendations for users based on their previous restaurant selections, utilizing the hash table to efficiently access and process this personalized information.

**Facilitating Search Functionality:** Hash tables can support search functionality by enabling the application to index restaurants based on specific criteria, such as cuisine type, location, or price range. This allows users to filter and search for restaurants that meet their specific preferences, with the hash table providing a means to quickly access and present the relevant search results.



### **Data Structures used:**

1) Feature: Menu Browsing

2) Data structures: Trees

3) Algorithms:

Depth-First Search (DFS) and Breadth-First Search (BFS): These graph traversal algorithms can be used to navigate through the hierarchical tree structure of menus. They enable efficient exploration of menu categories and subcategories, ensuring that users can easily browse through the menu options.

Recursive Algorithms for Tree Traversal: Recursive algorithms can be implemented to traverse the tree structure of menus, allowing for systematic exploration of various menu categories and subcategories. Recursive approaches simplify the process of displaying menu items and categories in a hierarchical and organized manner.

4) Brief info about this data structure:

Tree data structures are hierarchical data structures that consist of nodes connected by edges. They are used to represent hierarchical relationships between elements or to organize data in a way that supports efficient search, insertion, and deletion operations. Trees have a root node at the top, and every node in the tree has zero or more child nodes.

5)Implementation:

Zomato can use tree data structures to organize restaurant menus hierarchically. Each node in the tree can represent a specific menu category, such as appetizers, main courses, desserts, or beverages. Subcategories within each menu category can be represented as child nodes, ensuring a clear and intuitive structure for users to navigate and browse through the restaurant menus.

Efficient Menu Search and Retrieval: By utilizing tree structures, Zomato can facilitate quick and efficient menu search and retrieval. Users can easily browse through different menu categories and subcategories, enabling them to access the desired information swiftly and seamlessly.

Dynamic Menu Updates and Modifications: The tree structure allows for dynamic updates and modifications to the menus of various restaurants. This feature enables restaurant owners to easily add new items, modify existing ones, or remove outdated options, ensuring that the menu information remains up-to-date and accurate for users.

Enhanced User Experience: By implementing tree structures for menu browsing, Zomato can offer users an enhanced browsing experience, allowing them to explore restaurant menus in a structured and user-friendly manner. The hierarchical representation of menus facilitates intuitive navigation and simplifies the process of finding specific food items or categories within the menus.

## **Data Structures used:**

1) Feature: ORDER QUEUE

2) Data structures: FIFO QUEUE OR PRIORITY QUEUE

3) Brief info about this data structure:

A FIFO queue, also known as a "queue" in computer science, follows a simple rule: the first element added to the queue is the first one to be removed. It's similar to a real-world queue, like people waiting in line at a store or a bank.

A priority queue is a data structure where elements have an associated priority, and the element with the highest priority is always at the front. This is in contrast to a simple FIFO queue, where order of insertion determines processing order.

4) Algorithms:

**Synchronization Algorithms:** In scenarios where multiple users are simultaneously accessing the queue, synchronization algorithms are employed to manage access to shared resources and prevent data races. Techniques such as locks, semaphores, or mutexes might be utilized to ensure the orderly processing of orders and avoid conflicts between concurrent operations.

5) Implementation:

### **FIFO QUEUE:**

**Order Placement:** When a customer places an order, the application stores the order details in a queue data structure. This could be an array-based implementation or a linked list, depending on the specific requirements and constraints of the application.

**Real-Time Order Processing:** As orders come in, they are added to the back of the queue. The application continuously checks the queue and processes orders from the front of the queue based on the first-come-first-served principle.

**Dynamic Updates:** In real-time, the application continuously updates the queue to reflect new incoming orders and processed orders. It ensures that the processing of orders follows a FIFO approach, thereby maintaining the sequence in which the orders were received.

**Status Updates:** Customers might receive real-time notifications about their order status, including details about when their order is received, being processed, out for delivery, or delivered.

### **PRIORITY QUEUE:**

**Order Prioritization:** The priority queue assigns a priority level to each incoming order based on various factors such as delivery distance, customer preferences, restaurant capacity, or delivery time. For instance, orders with a shorter delivery distance or premium customer status might be assigned higher priorities.

**Real-Time Priority Updates:** As new orders arrive and priorities change dynamically, the priority queue is continuously updated to ensure that the highest priority order is processed and delivered first.

**Data Structures used:**

1) Feature: Payment Processing

2) Data structures: Linked list

3) Brief info about this data structure:

A linked list is a linear data structure where each element, known as a node, is a separate object. Each node contains a data field and a reference (link) to the next node in the sequence.

4) Implementation:

**Transaction Record Management:** Zomato can use a linked list to maintain a record of payment transactions made by customers for their food orders. Each node in the linked list can represent a specific transaction, storing details such as transaction ID, amount, date, and customer information.

**Dynamic Transaction Updates:** The linked list structure allows for dynamic updates, enabling the addition of new payment transactions as customers place orders and make payments through the platform. New transactions can be easily added to the end of the linked list, ensuring that the payment records remain up-to-date and comprehensive.

**Efficient Transaction Retrieval:** Zomato can utilize the linked list to efficiently retrieve and display transaction details when required, allowing administrators to access specific payment information for auditing or analysis purposes. Traversing through the linked list enables quick access to transaction details, facilitating efficient data retrieval for various payment processing operations.

**Order of Payment Processing:** The sequential nature of the linked list ensures that payment transactions are processed and recorded in the order in which they occur, providing a clear and accurate history of all payment activities on the platform.

### **Data Structures used:**

- 1) Feature: Location Based Services
- 2) Data structures: Weighted Graph and Dijkstra's Algorithm
- 3) Brief info about these data structures:

Graph: A graph is a data structure that consists of nodes (vertices) and edges. It's used to represent the connections or relationships between various entities. In the context of location-based searching, a graph can represent the map of a geographical area, where nodes are locations, and edges represent the paths or roads connecting them.

#### 4) Algorithms:

Dijkstra's Algorithm: Dijkstra's algorithm is employed to find the shortest path from the user's location to nearby restaurants. It considers factors like distance and road conditions to determine the optimal path. The algorithm computes the route efficiently, ensuring that users can access nearby dining options quickly.

#### 5) Implementation:

##### Graph:

Node Representation: Each node in the graph represents a geographical location or point of interest. Nodes can store information like latitude and longitude coordinates, location names, and relevant attributes.

Edge Representation: Edges in the graph represent the connections between locations, typically roads or pathways. Edges may contain data such as distance, estimated travel time, or road conditions.

Graph Creation: The graph is created based on real geographical data, incorporating information about roads, landmarks, and restaurant locations.

##### Dijkstra's Algorithm:

Source Node: The user's current location is set as the source node in the graph.

Weight Calculation: The algorithm calculates the weight (distance, time, or other relevant factors) of edges between the source node and other nodes, considering the user's location as the starting point.

Shortest Path Determination: Dijkstra's algorithm identifies the shortest path from the user's location to nearby restaurant nodes by minimizing the total edge weight.

Real-Time Updates: As the user's location changes or new restaurants are added, the graph and algorithm are dynamically updated to provide the most up-to-date information.

## **Data Structures used:**

1) Feature: Group Ordering and Splitting Bills

2) Data structures: Dictionary/Lists/Arrays

3) Brief info about these data structures:

Dictionary: A dictionary is a data structure that stores data in key-value pairs. Each key is unique, and it maps to a specific value. In group orders, a dictionary can be used to associate items with participants, allowing for easy tracking of who ordered what.

4) Implementation:

How Group Ordering and Bill Splitting Works:

1. Initiating a Group Order:

- A user initiates a group order, which can be done when placing an order through the app. They may invite friends to join the order.

2. Inviting Participants:

- The user sends invitations to friends to join the group order. Invitations can be sent via the app, email, or SMS.

3. Adding Items to the Order:

- Participants can add their desired items to the shared order. These items are added to a collective cart.

4. Assigning Items to Participants:

- Each item in the shared order is assigned to a specific participant. Users can choose which items they want to cover.

5. Automatic Bill Calculation:

- The app uses algorithms to calculate the total bill based on the assigned items. It also considers taxes, delivery fees, and any applicable discounts.

6. Splitting Costs:


- The app divides the total bill equally among all participants or according to their assigned items. It displays each participant's share.

7. Payment Options:

- Participants can choose their preferred payment methods to cover their share of the bill. The app may facilitate payment through various options, including credit cards, digital wallets, or in-app credits.

## Demo/Simulation Output:

### Restaurant Search-



**Domino's**

Hash function= searches and returns the address of the restaurant

#### Hash table

Restaurant Names	Location
1 La Plisir	Potheri
2 Domino's	Chetpet
3 Wow Mamos	Tambaram

Restaurant Names	Location
1 La Plisir	Potheri
2 Domino's	Chetpet
3 Wow Mamos	Tambaram

Here, keys = address, restaurant names, location  
values = corresponding values of keys

### Restaurant Recommendations-

**Search by cuisine type: ITALIAN**

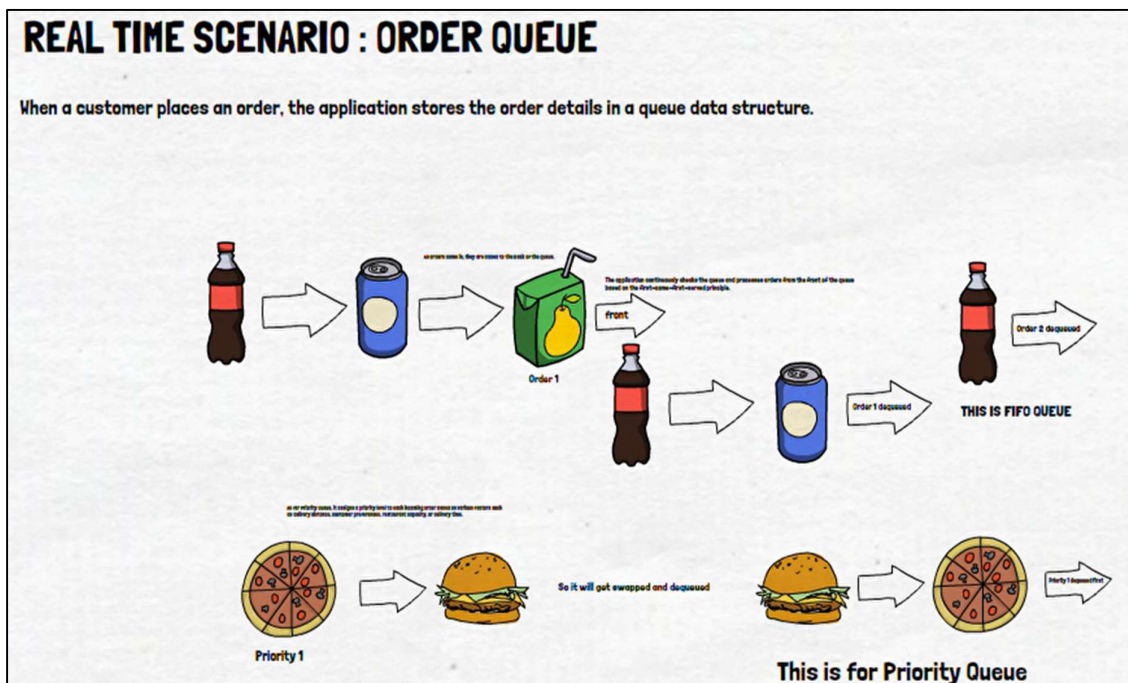
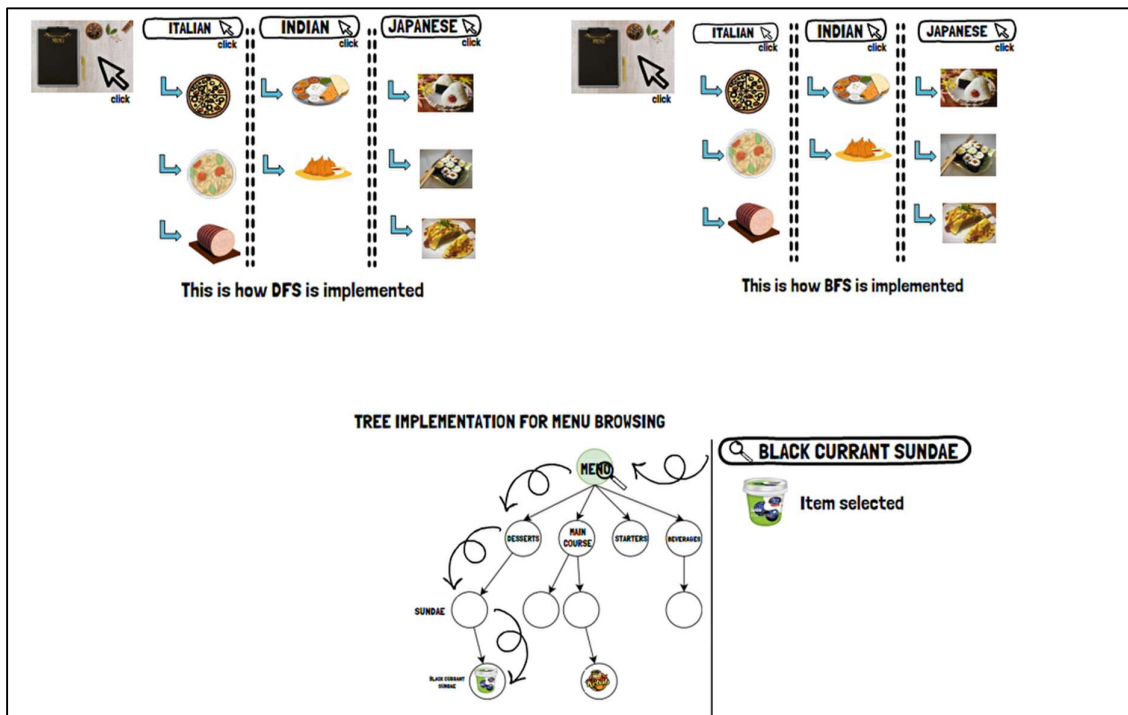
#### Hash table

Cuisine type	Food name
10 Italian	Pasta
20 Chinese	Noodles
30 Asian	Dim-sum

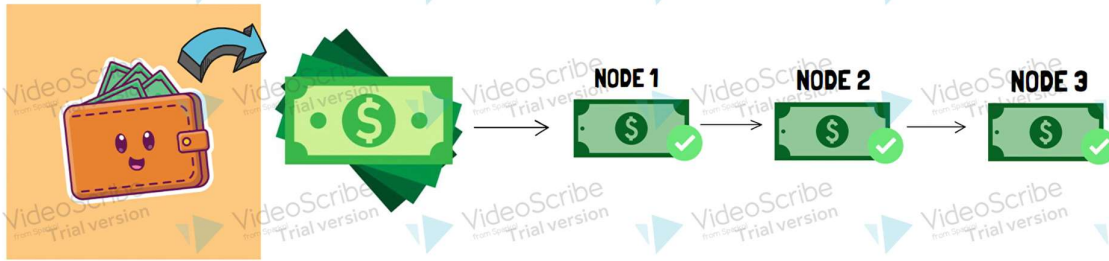
#### List of all Italian restaurants: Hash values

Restaurant name	Location
1 La Plisir	Potheri
2 PastaBowl	Crampet
3 Pizza Hut	Tirusulam

Here, keys = Cuisine and food name  
values = list of cuisines and food

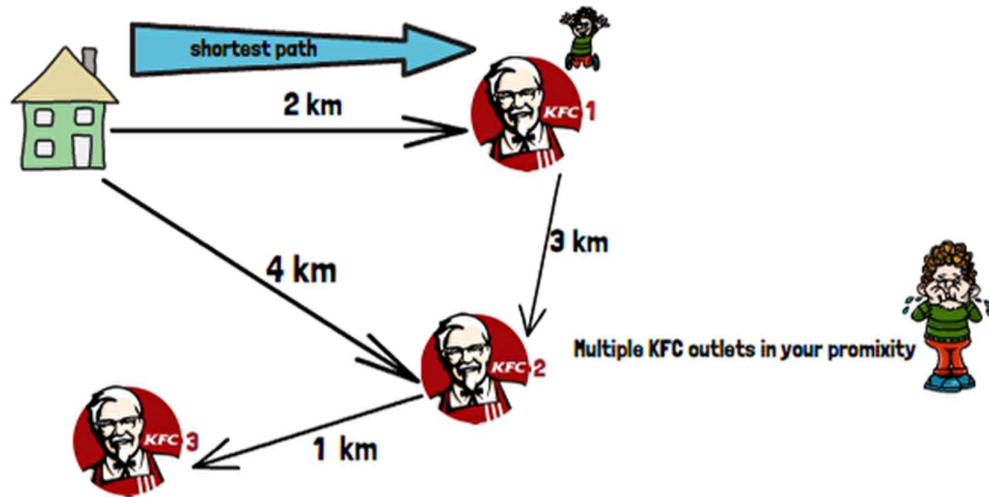


Zomato can utilize a linked list to maintain a chronological log of all payment transactions processed through the platform.

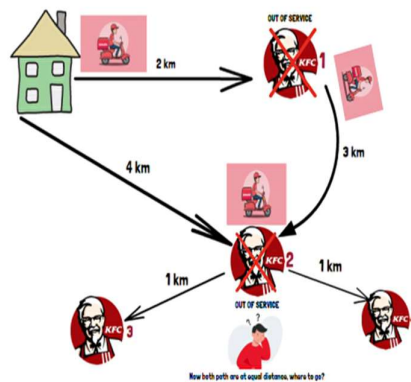


## Algorithm used: DIJKSTRA'S ALGORITHM

Brief info about the algorithm: algorithm to find the shortest path between nodes in a weighted graph.



## IMPLEMENTATION OF DIJKSTRA ALGORITHM AND WEIGHTED GRAPHS



No. of orders in KFC 3 = 6

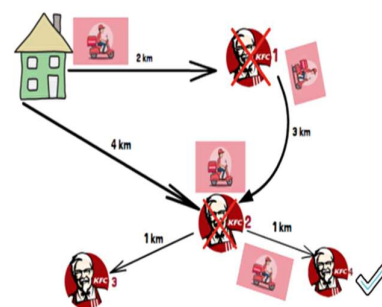
Order 6  
Order 5  
Order 4  
Order 3  
Order 2  
Order 1

No. of orders in KFC 4 = 3

Order 3  
Order 2  
Order 1

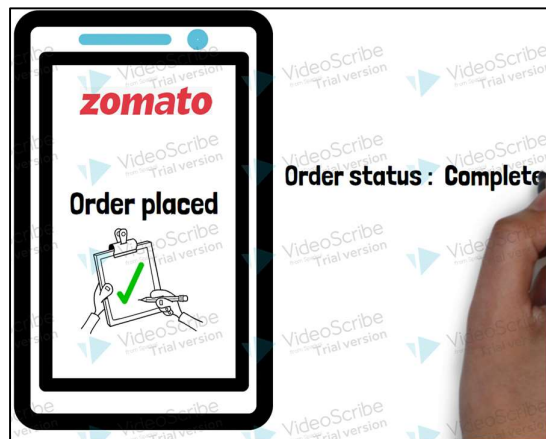
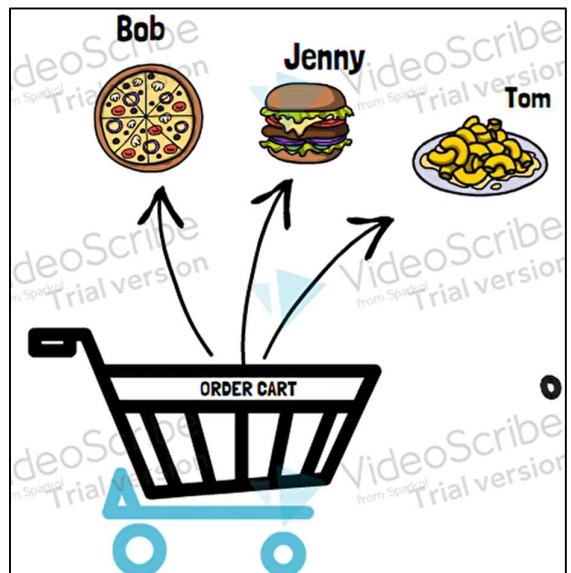
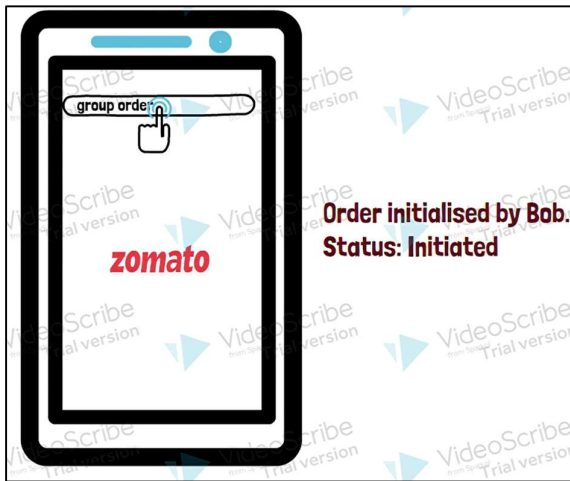
**KFC 4 < KFC 3**

In real time scenarios, when multiple short paths are available, path is selected based on next priority.



Order is placed from KFC 4





## **Conclusion:**

In conclusion, the analysis of data structures and algorithms in the Zomato app has unveiled the vital role they play in enhancing the overall user experience and app functionality. The significance of these elements is evident in the app's ability to efficiently provide restaurant recommendations, streamline menu browsing, manage order queues, ensure secure payment processing, and facilitate location-based restaurant searches.

Through the use of data structures such as trees, queues, and dictionaries, and algorithms like Dijkstra's algorithm for location-based searching, Zomato has demonstrated a commitment to optimizing the user experience. These data structures enable organized menu hierarchies, efficient order processing, and equitable group order and bill splitting.

The insights gained from this analysis emphasize the integral role of data structures and algorithms in creating a seamless and user-friendly platform. Zomato's successful implementation of these elements showcases their importance in navigating the complex world of food delivery and restaurant discovery. The app's efficiency in connecting users to their culinary desires hinges on the intelligent use of data structures and algorithms, making it a leading choice for food enthusiasts worldwide.