D Y Patil Agiculture & Technical University Talsande, Kolhapur

A

Project Report On

"ANALYSIS OF ONLINE FOOD ORDER & DELIVERY(ZOMATO)"

Submitted in the partial fulfilment of the requirements Under University Projects

Bachelor of Engineering in CSE (Data Science)

Submitted By

Avishkar Reneuse	PRN NO-2023011052038
Sakshi Nvase	PRN NO-2023011052021
Sayali Mandhare	PRN NO-2023011052019
Rutuja Adurkar	PRN NO-2023011052001
Shubham Savant	PRN NO-2023011052031

Under the Support and Guidence of

Dr.N.A Patil Sir

Department of CSE

Analysis of online food order & delivery(zomato) problem before use of this application:

Zomato, the popular Indian food delivery app, encountered a significant drop in orders during the festive season in india. This decline was unexpected, given that the festive period typically sees a surge in demand for food delivery services. Here are the key points:

1)Festive Season Decline:

During the festive season (usually in octomber), people order food delivery services more frequently due to celebrations and gatherings.

Despite this expected increase in orders, Zomato experienced a decrease, impacting revenue and market share.

2)Impact on Zomato:

- **Revenue Decrease:** The drop in orders affected zomato's revenue during a high-demand period.

- **Market Share Loss:** Competitors capitalized on festive season demand, potentially causing Zomato to lose market share.

- **Customer Satisfaction:** The decline could negatively impact customer satisfaction.

- **Brand Image:** Unaddressed, it might harm zomato's brand image and long-term prospects.

3)Scope of the Problem:

- Zomato's revenue and market share were directly affected.
- The company failed to effectively capture the festive season surge in demand.
- Loss of customer trust could harm its reputation.

4)Broader impact:

The problem extends to the entire Indian food delivery industry, which is highly competitive.

Study of existing system:

Background: Zomato, a popular Indian food delivery app, revolutionized the way people order food. Here are some key aspects to consider:

1)Emergence of Online Food Ordering:

- Zomato emerged during a time when the food industry was rapidly evolving.
- The rise urbanization and busy lifestyles led to increased demand for convenient food delivery services.

2)Features of Zomato's System:

- **Restaurant Listings**: Zomatos listed various restaurants, chefs, and canteens.

- **Menus and Privacy**: Consumers could explore menus without carrying physical pamphlets
- **Convenience:** Ordering food was just a click away, with no human intervention.
- **Payment Options:** Multiple payment model(credit cards, debit cards, cashless accounts) were available.
- **App Accessibility:** Consumers could download the app directly to their smartphones.

3)Drivers of Success:

- **Urbanization**: As India urbanized, Zomato's user base grew.
- **Facilities and Comfort**: Use-friendly features contributed to Zomato's success.
- **Consumer Satisfaction:** Convenience and privacy attracted users.

4)Market Potential:

- The Indian food industry was projected to reach $420 by 2020.
- Zomato capitalized on this growth.

SDLC model using for this application design & development:

Zomato, initially named Foodiebay, started in 2008 by Mr.Deepinder Goyal. It evolved into a restaurant searching platfrom providing in-depth details, autonomous reviews, and ratings. Here are relevant methodologies:

1)Waterfall Methodology:

- The Waterfall model is a sequential, linear process of project management**.**
- It consists of several discrete phases, where each phase begins only after the prior one is complete.
- Waterfall does not allow revisiting previous phases; any changes require starting over from phase one.
- It suits scenarios with known requirements and less complexity.
- However, it delays testing until the entire SDLC process is finished.

2)Agile Methodology:

- Agile emphasizes adaptability, customer satisfaction, and rapid delivery of working software products.
- It breaks development into small incremental builds provided in iterations.
- Agile allows flexibility, evolutionary development, early delivery, and continual improvement.
- Self-organizing, cross-functional teams collaborate closely customers/end users.
- Zomato likely adopted Agile to respond quickly to market changes and user needs.

<u>SRS document section -1</u>

Sure, here's a sample outline for the "online food Delivery " section of a Software Requirements Specification (SRS) document: online food Delivery

1.) Introduction

- Overview of the online food Delivery system.
- Purpose of this section in the SRS document.

2.) Functional Requirements

- **User Registration**

  Requirement 2.1: User registration is a necessary step for customers to create an account on the online food ordering system.
- **Menu Management**

  Requirement 2.2: Menu management is an essential function of an online food ordering system.
- **Order Placement**

  Requirement 2.3: The order placement function is the core of an online food ordering system.
- **Payment Processing**

  Requirement 2.4: Payment processing is an important function of an online food ordering system.
- **Delivery Management**

  Requirement 2.5: Delivery management is the final step in the online food ordering process.

3.) Functional Requirements

- **Usability**

  Requirement 2,1: Usability is an important factor in the success of an online food ordering system.
- **Performance**

  Requirement 2.2: Performance is an important factor in the success of an online food ordering system.
- **Reliability**

  Requirement 2.3: Reliability is another important factor in the success of an online food ordering system.
- **Scalability**

  Requirement 2.4: Scalability is an important consideration for an online food ordering system.

4.) Interface Requirements

- **User Interface**

  Requirement 4.1: The user interface should be intuitive and user-friendly.

  Requirement 4.2: The interface should be accessible on multiple devices (desktop, mobile).

- **Integration**
  Requirement 4.3: The system should integrate with payment gateways for premium payments

5)Constraints

- **Shifting Customer Preferences**
  Requirement 5.1: The ultimate aim of any food delivery service would be to grow its market share by offering the best possible value to customers at the least possible cost.
- **Unstable Market Prices**
  Requirement 5.2: Apart from the customer base, food costs are also highly volatile. Several factors affect the prices in the food industry.

Problem statement

Certainly! Here's a refined problem statement for online food delivery system.

The online food ordering system sets up a food menu online and customers can easily place the order as per they like. Also, the online customers can easily track their orders. This system stem also provides a feedback system in which user can rate the food items. Also, the proposed system can recommend hotels, food, based on the ratings given by the user, the hotel staff will be informed for the improvements along with the quality. The payment can be made online or cash or pay-on-delivery system.

Feasibility study :-Cost vs benefits tangible/ intangible) operational and social

1) **Technical feasibility:** The technical requirement for the system is economic and it does not use any other additional Hardware and software.

2) **Economic Feasibility:** Analyze the costs associated with setting up and maintaining the food delivery system compared to the benefits it brings, both tangible (direct financial gains) and intangible (improved safety, peace of mind).

3) **Operational Feasibility:** Assess how smoothly the food delivery system an be integrated into existing processes and operations, considering factors like staff training, customer service.

4) **Social Feasibility**: Consider the societal impact of food delivery, including its role in promoting safer driving habits, mitigating financial risks for individuals, and contributing to overall economic stability.

Costs:

- **Development Costs:** Estimate the expenses involved in building the online food delivery platform, including software development, infrastructure setup, and

integration of third-party services.
- **Operational Costs:** Calculate ongoing expenses such as hosting fees, maintenance, customer support, and transaction processing fees.
- **Marketing and Promotion Costs:** Assess the investment required for advertising, promotional offers, SEO, and content marketing to attract users and drive engagement.
- **Personnel Costs:** Determine the costs associated with hiring and retaining staff for system development, maintenance, customer support, and administration.
- **Regulatory and Compliance Costs:** Factor in expenses for compliance audits, legal consultation, and insurance coverage to ensure regulatory compliance and risk mitigation.

Benefit:

- **Increased Revenue**: Estimate the potential increase in revenue from online orders, considering factors such as order volume, average order value, and customer retention.
- **Expanded Market Reach:** Evaluate the opportunity to reach a broader customer base by offering online ordering and delivery services, including potential for growth in new geographic markets.
- **Improved Customer Experience:** Assess the benefits of providing a convenient and user-friendly ordering platform, leading to higher customer satisfaction, loyalty, and repeat business.
- **Operational Efficiency:** Consider the efficiencies gained from streamlining order management, reducing manual errors, and optimizing resource allocation through automation.
- **Competitive Advantage:** Analyze the competitive landscape and assess how offering online delivery services can differentiate the business and attract customers away from competitors.

Tangible:

1. Increased Revenue:

- **Order Volume:** Online food delivery systems often lead to an increase in the number of orders, as they offer convenience and accessibility to customers.
- **Higher Average Order Value:** Customers tend to spend more when ordering online, leading to an increase in average order value compared to in-person dining.
- **New Customer Acquisition:** Online platforms can attract new customers who prefer the convenience of ordering food from home.

2. Cost Savings:

- **Reduced Overhead:** Online ordering eliminates the need for additional staff, tables, and space required for in-person dining, resulting in cost savings on overhead expenses.
- **Optimized Inventory Management**: With real-time tracking of orders and ingredients, restaurants can optimize inventory levels, minimize waste, and reduce food costs.

- **Decreased Marketing Costs:** Online platforms provide cost-effective marketing opportunities through targeted promotions, reducing the need for traditional advertising methods.

3. Operational Efficiency:

- **Streamlined Order Management**: Automated order processing and integration with POS systems streamline order management, reducing manual errors and improving efficiency.
- **Faster Turnaround Time:** Online ordering systems enable faster order processing and delivery, leading to shorter wait times for customers and increased order throughput.
- **Improved Customer Service:** Features such as order tracking and real-time communication enhance the customer experience, leading to higher satisfaction and loyalty.

Intangible

1. Convenience and Accessibility:

- **Enhanced Customer Experience**: Online food delivery systems offer convenience and accessibility, allowing customers to order food from anywhere, at any time, using their preferred devices.
- **Reduced Effort:** Customers appreciate the convenience of not having to travel to a restaurant or wait in line to place an order, saving time and effort.

2. Customer Satisfaction and Loyalty:

- **Improved Convenience:** Providing customers with the option to order food online enhances their overall satisfaction with the restaurant's service, leading to increased loyalty and repeat business.
- **Personalized Experience:** Customizable ordering options and personalized recommendations based on past orders contribute to a more tailored and enjoyable dining experience.

3. Brand Perception and Image:

- **Tech-Savvy Image:** Restaurants with online food delivery systems are perceived as modern and technologically advanced, appealing to tech-savvy consumers and enhancing the restaurant's brand image.
- **Innovation and Adaptability:** Embracing digital platforms demonstrates a restaurant's ability to adapt to changing consumer preferences and market trends, positioning it as an innovative and forward-thinking brand.

4. Competitive Advantage:

- **Market Differentiation:** Offering online food delivery services sets restaurants apart from competitors and gives them a competitive edge in the crowded restaurant industry.

- **Expanded Reach:** Restaurants can reach new customer segments, including those who prefer the convenience of online ordering over traditional dining experiences.

Social:

The social aspects of an online food delivery system encompass its impact on individuals, communities, and society as a whole. Here are several key social aspects:

1. Convenience and Time Savings:

- **Busy Lifestyles:** Online food delivery systems cater to individuals with hectic schedules who may not have the time or energy to cook or dine out.
- **Work-Life Balance:** By providing convenient access to meals, these systems support individuals in balancing work, family, and other commitments.

2. Social Interaction and Connectivity:

- **Shared Experiences:** Ordering food online allows friends, family, and colleagues to enjoy meals together, even when physically apart.
- **Virtual Gatherings:** Online food delivery facilitates virtual gatherings and social events, fostering connections and maintaining relationships across distances.

3. Inclusivity and Accessibility:

- **Accessibility:** Online food delivery systems increase access to restaurant meals for individuals with mobility limitations, disabilities, or limited transportation options.
- **Diverse Cuisine:** These platforms offer a wide variety of cuisines, catering to diverse tastes and dietary preferences, including vegetarian, vegan, and gluten-free options.

4. Support for Local Businesses:

- **Local Restaurants:** Online food delivery systems support local restaurants and food vendors by providing them with a platform to reach a broader customer base.
- **Economic Impact:** By patronizing local establishments, consumers contribute to the local economy and help sustain small businesses in their communities.

5. Employment Opportunities:

- **Delivery Drivers:** Online food delivery systems create job opportunities for delivery drivers, providing flexible employment options and supplemental income for individuals.
- **Restaurant Staff:** Increased demand for food delivery may lead to job creation within restaurants, such as hiring additional kitchen staff or order coordinators.

6. Food Waste Reduction:

- **Portion Control**: Online ordering allows customers to order only what they need, reducing food waste associated with oversized portions at restaurants.
- **Inventory Management**: Restaurants can optimize food inventory and minimize waste by accurately predicting demand through online orders.

Requirements Gathering

1.User Registration:

- Users should be able to register an account with the platform.
- Registration should require basic information such as name, email, password, and address.

2.Restaurant Registration:

- Restaurants should be able to register their establishment on the platform.
- Registration should require details such as restaurant name, address, contact information.

3.User Profile Management:

- Users should be able to update their profile information, including name, address, contact details, and password.
- Users should have the option to delete their account if they wish to.

4.Restaurant Profile Management:

- Restaurants should be able to update their profile information, including contact details, menu items, and opening hours.
- Restaurants should have the option to delete their profile if needed.

5.Browsing and Ordering:

- Users should be able to browse restaurants based on location, cuisine type, ratings, etc.
- Users should be able to view restaurant details, including menu items, prices, and ratings.
- Users should be able to add items to their cart and place an order.

6.Placing an Order:

Description: Users should be able to select items from the menu of a restaurant and place an order.
Requirements:

- Users should be able to add items to their cart.
- Users should specify the quantity of each item they wish to order.

7.Cancelling an Order:

Description: Users should have the option to cancel an order after placing it.
Requirements:

- Users should be able to cancel an order before it is confirmed by the restaurant.
- Users should receive a confirmation prompt when attempting to cancel an order.

8.Payment System:

- The platform should support online payments for orders.Users should be able to pay using various payment methods such as credit/debit cards, digital wallets (e.g., PayPal), or cash on delivery.

9.Order Tracking:

- Users should be able to track the status of their orders in real-time, from order placement to delivery.

10.Support System:

- The platform should have a support system in place to handle user queries, complaints, and feedback.

<u>Requirements specification - Functional</u>

1) Registration

- If customer wants to order the food then he/she must be registered, unregistered user can't go for ordering

2) Login

- The customer login to the system by entering valid user id and password for ordering

3) Display the menu

- In the system all the items are displayed with their rates.

4) Modify menu

- System can make changings in menu like adding or removing food items which are not avaibble.

5) Select food item's

- Items are selected customer feel free to order.

6) Changes to order

- Changes to order means the customer can make changings in order like he/she can delete or add food item in order.

7) Review the order before submitting

- Before submitting the complete order is reviewed to the customer. Customer name, phone number, location (address) and placed order, hen finally order is submitted.

8) **Payment**

- For customer there are many type of secure billing will be prepaid as debit or credit card, postpaid as after delivering, check or bank draft.

9) **Provide delivery and payment details**

- Here bill is generated, order no, and payment is given and confirmation of delivery is done.

10) **Logout**

- After the payment or surf the product the customer will log out.

Requirements Specification Non-Functional

1) **Portability**

- System running on one platform can easily be converted to run on another platform

2) **Reliability**

- The ability of the system to behave consistently in a user-acceptable manner when operating within the environment for which the system was intended.

3) **Availability**

- The system should be available at all times, meaning the user can access it using a web browser, only restricted by the down time of the server on which the system runs.

4) **Maintainability**

- A commercial database is used for maintaining the database and the application server takes care of the site.

5) **Security**

- Secure access of confidential data (customer information).

6) **User friendly**

- Systern should be easily used by the customer.

7) **Performance**

- Performance should be fast.

8) **Efficient**

- System should be efficient that it won't get hang if heavy traffic of order is placed.

9) Safety

- Data in the database of system should not loss or damage.

10) Privacy

- Personal data of the system should not disclose to anyone.

Requirements Specification - Input, output and process

Input requirements for online food delivery system

Certainly! Input requirements for an online food delivery system typically involve various types of data and information that users provide or interact with during their usage of the system. Here's a breakdown of input requirements:

1) User Registration and Authentication

Input Data:

- User's full name
- Contact information (email address, phone number)
- Delivery address
- Username
- Password

2) Search Queries

Input Data:

- Keywords for food items (e.g., pizza, sushi, burger)
- Location for restaurant search (optional)
- Filters (e.g., cuisine type, price range, dietary preferences)

3) Order Placement

Input Data:

- Selected food items
- Quantity of each item
- Special instructions (e.g., no onions, extra cheese)
- Preferred delivery time (if applicable)
- Payment method

4) Payment Information

Input Data:

- Credit/debit card details (card number, expiry date, CVV)
- Billing address
- Digital wallet information (e.g., PayPal, Apple Pay, Google Pay)

5) Feedback and Reviews

Input Data:

- Rating (numeric or star-based)
- Review comments
- Optional: User details (for authenticated feedback)

6) Account Management

Input Data:

- Profile updates (e.g., change of address, phone number)
- Password changes
- Subscription preferences (if applicable)

7) Admin Functions (for system administrators)

Input Data:

- Restaurant/menu updates
- User management (e.g., adding new users, modifying permissions)
- Analytics parameters (if the system provides analytics dashboards)

8) Real-Time Tracking (for users to track their orders)

Input Data:

- Order ID or reference number
- Location tracking permissions (if using GPS for real-time tracking)

9) Customer Support

Input Data:

- User queries or complaints
- Contact details for response (email, phone number)

Output requirements for online food delivery system

Output requirements for an online food delivery system pertain to the information and feedback provided to users and administrators throughout their interactions with the platform. Here's a breakdown of output requirements:

1) User Confirmation and Notifications

Output:

- Account creation confirmation (upon successful registration)
- Login authentication status (success/failure)
- Order confirmation details (items, quantity, delivery time)
- Payment confirmation (successful transaction)
- Delivery status updates (order dispatched, estimated arrival time)
- Receipts and invoices (sent via email or accessible in-app)

2) Search Results

Output:

- List of food items/restaurants matching search queries
- Detailed information about each item/restaurant (name, description, price, ratings)

3) Order Tracking

Output:

- Real-time order tracking information (order status, delivery driver details)
- Estimated time of arrival (ETA) updates
- Notifications for order delays or changes in status

4. **Feedback and Reviews**

Output:

- Acknowledgment of feedback submission
- Display of user reviews and ratings for restaurants and food items
- Responses to user feedback (if provided by the system or restaurant)

5. **Account Management**

- Output:
- Profile updates confirmation (changes saved successfully)
- Password change confirmation
- Subscription status updates (if applicable)

6. **Admin Notifications and Reports**

Output:.

- Notifications for new orders, cancellations, or refunds
- Reports on sales, revenue, and customer feedback
- Alerts for system maintenance or updates

7. **Customer Support Responses**

Output:

- Acknowledgment of user queries or complaints
- Resolution status (resolved, pending, escalated)
- Responses to user inquiries with relevant information or solutions

8. **Recommendations and Promotions**

Output:

- Personalized food recommendations based on past orders or preferences
- Promotional offers and discounts notifications

9. **System Errors and Alerts**

Output:

- Error messages for failed actions (e.g., payment declined, item out of stock)
- Alerts for technical issues or system downtime
- Guidance on how to resolve common issues (FAQs, troubleshooting tips)

10**. Analytics and Insights**

Output:

- Visualizations and reports on user engagement, order trends, and revenue
- Insights on popular food items, peak ordering times, and customer demographics

Process requirements for online food delivery system

1) User Registration and Authentication Process

Process:

- User initiates registration by providing necessary details.
- System validates input data (e.g., email format, password strength).
- Upon successful validation, user account is created.
- User logs in using registered credentials for subsequent access.

2) Food Search and Selection Process

Process:

- User enters search criteria (e.g., food item, cuisine, location).
- System retrieves relevant results from the database.
- User selects desired items or restaurants from the search results.
- Detailed information (e.g., menu, prices, ratings) is displayed for selected items/restaurants.

3) Order Placement Process

Process:

- User adds selected items to the cart.
- User provides additional instructions (e.g., customization, delivery time).
- System calculates the total order amount including taxes and delivery charges.
- User proceeds to checkout and selects preferred payment method.

4) Payment Processing

Process:

- User enters payment details (credit/debit card, digital wallet).
- System securely processes the payment transaction.
- Payment gateway communicates with relevant financial institutions for authorization.
- Upon successful payment, confirmation is sent to the user and order is placed.

5) Order Fulfillment and Delivery Process

Process:

- Restaurant receives order notification and prepares the items.
- Delivery personnel are assigned to pick up and deliver the order.
- User receives notifications on order status and estimated delivery time.
- Delivery personnel navigate to the user's location using GPS or provided address.

6) Order Tracking and Updates

Process:

- User can track the status of their order in real-time.
- System provides updates on order preparation, pickup, and delivery stages.

- Estimated time of arrival (ETA) is continuously updated based on delivery progress.

7) Feedback and Review Process

Process:

- After order completion, user is prompted to provide feedback and ratings.
- System collects and stores user reviews for restaurants and food items.
- Restaurants may respond to user feedback to address concerns or express gratitude.

8) Account Management Process

Process:

- User can update profile information (e.g., address, contact details).
- Password reset functionality is available in case of forgotten passwords.
- User can manage subscription preferences or opt-out of promotional communications.

9) Admin Functions and Management Process

Process:

- Administrators can manage restaurant listings, menus, and pricing.
- They can monitor and analyze order data, sales performance, and customer feedback.
- Admins handle customer support inquiries and address technical issues promptly.

10)Customer Support Process

Process:

- Users can reach out to customer support via various channels (e.g., chat, email, phone).
- Support agents triage and respond to user queries, issues, and complaints.
- Escalation procedures are in place for unresolved or escalated cases.

Requirements Specification – Performance (static & Dynamic)

Static Performance Specifications: for online food delivery system Static performance specifications outline the quantitative criteria and benchmarks that the online food delivery system must meet in terms of its performance characteristics. Here are some static performance specifications for an online food delivery system:

1) Response Time

- **Specification:** The system should respond to user interactions within a maximum of 3 seconds.
- **Rationale:** Quick response times enhance user experience and satisfaction, ensuring seamless navigation and interaction with the platform.

2) System Availability

- **Specification:** The system should have an uptime of at least 99.9%.

- **Rationale:** High availability ensures that users can access the platform whenever they need it, minimizing disruptions and downtime.

3) Concurrent User Capacity

- **Specification:** The system should be able to handle a minimum of 1000 concurrent users.
- **Rationale:** Scalability is crucial to accommodate peak usage periods and prevent performance degradation during high traffic.

4) Data Throughput

- **Specification:** The system should support a minimum data throughput of 100 requests per second.
- **Rationale:** High throughput ensures efficient processing of user requests, reducing latency and improving overall system performance.

5) Search Speed

- **Specification:** Search queries for food items/restaurants should return results within a maximum of 1 second.
- **Rationale:** Fast search speeds enable users to quickly find relevant options, enhancing user satisfaction and engagement.

6) Payment Processing Time

- Specification: Payment transactions should be processed and confirmed within 5 seconds.
- Rationale: Swift payment processing is essential for a smooth checkout experience, reducing user wait times and potential order abandonment**.**

7) Order Placement Time

- **Specification:** The time taken to place an order should not exceed 2 minutes.
- **Rationale:** Streamlined order placement processes ensure efficient transactions and reduce user friction, leading to higher conversion rates.

8) Order Delivery Time Accuracy

- **Specification:** The estimated delivery time provided to users should be accurate within a margin of ±5 minutes.
- **Rationale:** Accurate delivery time estimates improve user satisfaction and trust in the system, reducing instances of dissatisfaction due to delays.

9) System Reliability

- **Specification:** The system should have a Mean Time Between Failures (MTBF) of at least 30 days.
- **Rationale:** High reliability ensures minimal disruptions and downtimes, fostering user trust and confidence in the system's performance**.**

10)Security Compliance

- **Specification:** The system should comply with industry-standard security protocols (e.g., PCI DSS) to protect user data and payment information.
- **Rationale:** Robust security measures safeguard user privacy and prevent unauthorized access, enhancing trust in the platform**.**

Dynamic Performance Specifications: for online food delivery system

Dynamic performance specifications focus on the system's behavior under varying workloads or conditions. Here are dynamic performance specifications for an online food delivery system:

1) Scalability

- **Specification:** The system should scale dynamically to handle increasing user loads during peak hours, such as lunch and dinner times.
- **Scenario:** Simulate a sudden surge in user traffic by increasing the number of concurrent users by 50% and observe the system's ability to maintain response times within acceptable limits.

2) Load Balancing

- **Specification:** The system should distribute incoming requests evenly across multiple servers to prevent overloading and ensure optimal resource utilization.
- **Scenario:** Introduce varying levels of simulated traffic to the system and monitor how effectively requests are distributed among server instances, ensuring no single server becomes a bottleneck.

3) Fault Tolerance

- **Specification:** The system should gracefully handle failures in individual components or services without affecting overall functionality.
- **Scenario:** Introduce simulated failures in backend services or databases and observe how the system reroutes requests, maintains data integrity, and continues to serve users without interruption.

4) Response Time under Load

- **Specification:** The system should maintain consistent response times even under heavy load conditions, ensuring a smooth user experience.
- **Scenario:** Gradually increase the number of concurrent users and observe how response times are affected, ensuring that they remain within acceptable thresholds even at peak loads.

5) Recovery Time

- **Specification:** In the event of a system failure or downtime, the system should recover and resume normal operation within a specified timeframe.
- **Scenario:** Simulate a server crash or network outage and **measure the time taken for** the system to detect the failure, initiate recovery procedures, and restore full functionality**.**

6) **Session Management**

- **Specification:** The system should efficiently manage user sessions to prevent session timeouts or loss of user data.
- **Scenario:** Test user sessions under varying conditions, such as prolonged inactivity or concurrent sessions from multiple devices, ensuring that users remain authenticated and their data is preserved.

7) **Database Performance**

- **Specification:** The system's database should efficiently handle read and write operations without degradation in performance, even with increasing data volume.
- **Scenario:** Generate large volumes of data and simulate concurrent read and write operations to assess database performance and scalability.

8) **Transaction Processing**

- **Specification:** The system should process payment transactions accurately and reliably, without errors or discrepancies.
- **Scenario:** Conduct stress tests by processing a high volume of payment transactions simultaneously, ensuring that all transactions are processed correctly and funds are transferred securely.

9) **Geographic Load Distribution**

- **Specification:** The system should efficiently serve users located in different geographical regions, minimizing latency and ensuring consistent performance worldwide.
- **Scenario:** Simulate user access from various locations and measure response times, ensuring that users experience similar performance regardless of their geographic location.

10) **Resource Utilization**

- **Specification:** The system should optimize resource utilization to minimize costs while maintaining performance.
- **Scenario:** Monitor CPU, memory, and bandwidth usage under varying workloads and optimize resource allocation to ensure efficient utilization without unnecessary overhead.

Requirement specifications-User Interface:

A requirements specification for the user interface (UI) defines the necessary features and functionalities of the UI for a software application. It outlines how the UI should look and behave, ensuring that the application is user-friendly, accessible, and aligned with users' needs. Here are some key points to consider when creating a requirements specification for the UI:

1) General Requirements:

- o The UI should be intuitive, user-friendly, and easy to navigate. - The design should be visually appealing, with consistent colours, fonts, and styles throughout the

application.
- The UI should be responsive and compatible with different devices and screen sizes.

2) User Authentication:
- The UI should include a secure login and registration process.
- Provide options for password recovery and account management (e.g., changing password, updating profile).

3) Dashboard and Navigation:
- The UI should have a dashboard for users to access key features and information at a glance.
- Include a clear and organized navigation menu to help users move between different sections of the application.

4) Policy Management:
- The UI should allow users to create, view, edit, and delete insurance policies.
- Provide an easy-to-use form for entering policy details, such as coverage type, policy term, and premium amount.

5) Claims Processing:
- The UI should enable users to submit claims with necessary details and supporting documents.
- Include a claims tracking feature so users can monitor the status of their claims.

6) Premium Calculation and Payment:
- The UI should offer a calculator for users to estimate premiums based on factors such as age, vehicle type, and driving history.
- Provide options for online payments and payment history tracking.

7) Notifications and Alerts:
- Include notifications for important events, such as policy expiration, claim status updates, and payment reminders.
- Allow users to customize notification preferences.

8) Search and Filtering:
- Provide search and filtering options to help users quickly find policies, claims, or other relevant information.

9) Help and Support:
- Include a help section with FAQs, guides, and contact information for customer support.
- Consider implementing a chatbot for instant support and assistance.

10) Accessibility and Inclusivity:
- Ensure the UI is accessible to users with disabilities, following web accessibility standards.

Requirement specifications-Any other:

Requirements specifications are essential for defining the features and functionalities of different aspects of a software application. Here are some additional areas where requirements specifications are important:

1) Security Requirements:

- The application must protect user data with encryption, secure authentication, and access control measures.
- Follow industry best practices for secure coding to prevent vulnerabilities such as SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF).
- Ensure compliance with data protection laws and regulations such as GDPR.

2)Performance Requirements:

- The application should respond quickly to user interactions and maintain high availability.
- Define acceptable response times for various operations such as loading pages, submitting forms, and processing transactions.
- Implement caching and optimization techniques to improve performance.

3) Scalability Requirements:

- The application should be able to handle an increasing number of users, data, and transactions without degradation in performance.
- Use scalable architectures such as microservices or serverless computing to support growth.

4) Data Requirements:

- Specify the types of data the application will handle, such as user profiles, transactions, and logs.
- Define data validation rules and data integrity constraints to ensure data accuracy and consistency.
- Outline data storage and backup strategies to protect data from loss.

5) Integration Requirements:

- Define the systems and services the application needs to integrate with, such as payment gateways, CRM systems, or APIs.
- Specify data formats, protocols, and authentication methods for integration.
- Ensure compatibility and seamless data exchange with other systems.

6) Usability Requirements:

- Focus on creating an intuitive and user-friendly experience with clear instructions and labels.
- Allow users to customize settings and preferences for a personalized experience.
- Conduct usability testing to gather feedback and improve the user experience.

7) Compliance Requirements:

- Ensure the application complies with relevant industry standards, laws, and regulations.
- Maintain documentation and logs to demonstrate compliance in case of audits.

8) Testing Requirements:
- Define the types of tests the application will undergo, such as unit tests, integration tests, and user acceptance tests.
- Specify testing tools and frameworks to be used.
- Outline criteria for test success and failure.

9) Maintenance Requirements:
- Plan for ongoing maintenance, including bug fixes, updates, and feature enhancements.
- Establish a process for handling user support requests and incidents.

10) Deployment Requirements:
- Specify the deployment process and environments (e.g., development, staging, production).
- Define the release management strategy and versioning approach.
- Plan for rollbacks and contingencies in case of deployment issues.

Hardware & Software Requirements:

- **Hardware Requirements:**

  1) Server Requirements:
  - **Processor:** Specify the type and speed of the processor (e.g., Intel Xeon, AMD EPYC).
  - **Memory:** Define the minimum and recommended RAM for optimal performance (e.g., 16GB, 32GB).
  - **Storage:** Determine the type and capacity of storage (e.g., SSDs, HDDs, RAID configuration).
  - **Network:** Specify network bandwidth and connectivity requirements (e.g., 1Gbps, 10Gbps).
  - **Graphics:** For applications with heavy graphical content, consider specifying the type of graphics card.

  2) Client Requirements:
  - **Processor:** Specify the minimum and recommended processor speed for client devices.
  - **Memory:** Define the minimum and recommended RAM for client devices.
  - **Storage:** Specify storage capacity for client devices, if applicable.
  - **Display:** Define the screen resolution and size for optimal viewing.
  - **Internet Connection:** Specify minimum and recommended network speed for users to access the application.

- **Software Requirements:**

1) **Operating System:**
   - Define supported operating systems for servers (e.g., Windows Server, Linux) and clients (e.g., Windows, macOS, Linux).
   - Specify the version or distribution if applicable.

2) **Software Dependencies:**
   - **Server:** List server-side dependencies, such as web servers (e.g., Apache, Nginx), databases (e.g., MySQL, PostgreSQL), and programming languages (e.g., Python, Java, Node.js).
   - **Client:** Specify client-side dependencies, such as browsers (e.g., Chrome, Firefox), frameworks (e.g., React, Vue.js), and plugins.

3) **Development Tools:**
   - Specify the tools and IDEs for development (e.g., Visual Studio Code, PyCharm, Eclipse).
   - List version control tools (e.g., Git) and collaboration tools (e.g., GitHub, GitLab).

4) **Database:**
   - Define the type and version of the database (e.g., MySQL, PostgreSQL, MongoDB).
   - Specify backup and recovery tools, if applicable.

5) **Middleware:**
   - Specify any middleware or libraries required for communication between components (e.g., message queues, caching layers).

6) **Testing Tools:**
   - Define testing tools for unit, integration, and end-to-end testing (e.g., Jest, Selenium).

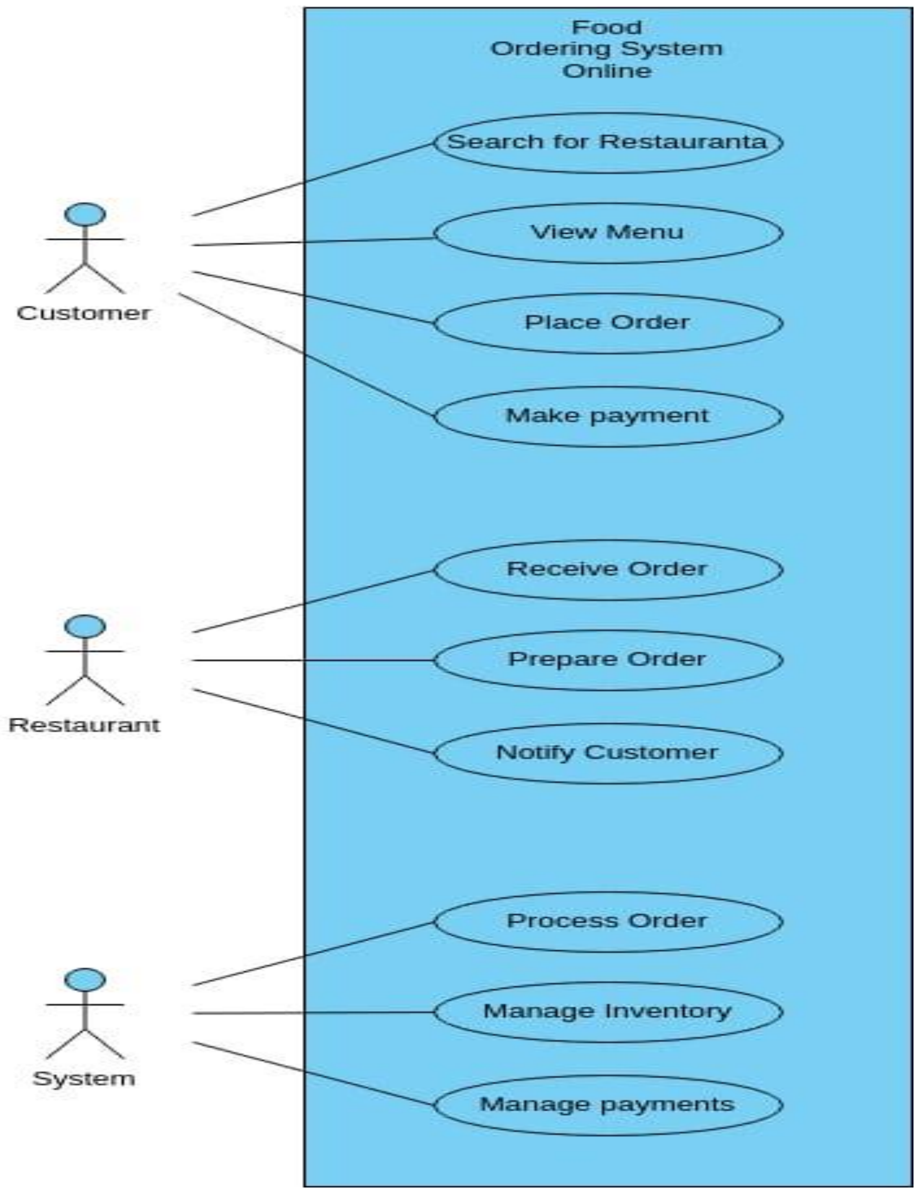
7) **Monitoring and Logging:**
   - Specify tools for monitoring application performance and logs (e.g., Prometheus, ELK stack).

8) **Deployment Tools:**
   - Define deployment and continuous integration tools (e.g., Jenkins, Docker, Kubernetes).

Use Case	Actor	Description
Register	Customer	Allows a new customer to create an account on the platform by providing necessary details such as name, email, and password.
Login	Customer	Enables registered customers to log in to their account using their credentials.
Search Restaurants	Customer	Allows customers to search for restaurants based on various criteria such as location, cuisine, and ratings.
Browse Menu	Customer	Permits customers to view the menu of a selected restaurant, displaying available food items with descriptions, prices, and images.
Place Order	Customer	Enables customers to add selected items to their cart, specify quantity, provide delivery details, and proceed to checkout for order placement.
Make Payment	Customer	Facilitates payment processing for orders, allowing customers to securely enter payment details and complete transactions using various methods.
Track Order	Customer	Allows customers to track the status of their orders in real-time, providing updates on order preparation, dispatch, and estimated delivery time.
View Order History	Customer	Permits customers to view their order history, displaying details of past orders including items, total amount, and delivery status.
Provide Feedback	Customer	Enables customers to provide feedback and ratings for restaurants and food items, contributing to the overall rating and reputation of the platform.
Manage Account	Customer	Allows customers to manage their account settings, including profile information, password changes, and subscription preferences.
Manage Restaurants	Restaurant Owner	Allows restaurant owners to manage their restaurant profile, including updating menu items, prices, descriptions, and business hours.
Accept/Reject Orders	Restaurant Owner	Permits restaurant owners to accept or reject incoming orders, providing flexibility to manage order volume and availability.
View Sales Analytics	Administrator	Enables administrators to view sales analytics and reports, including order trends, revenue, popular items, and customer demographics, for performance analysis and decision-making.
Manage Users	Administrator	Allows administrators to manage user accounts, including adding new users, modifying permissions, and handling account-related issues.
Provide Support	Customer Support	Facilitates customer support interactions, allowing support agents to respond to user inquiries, resolve issues, and escalate unresolved cases as needed.

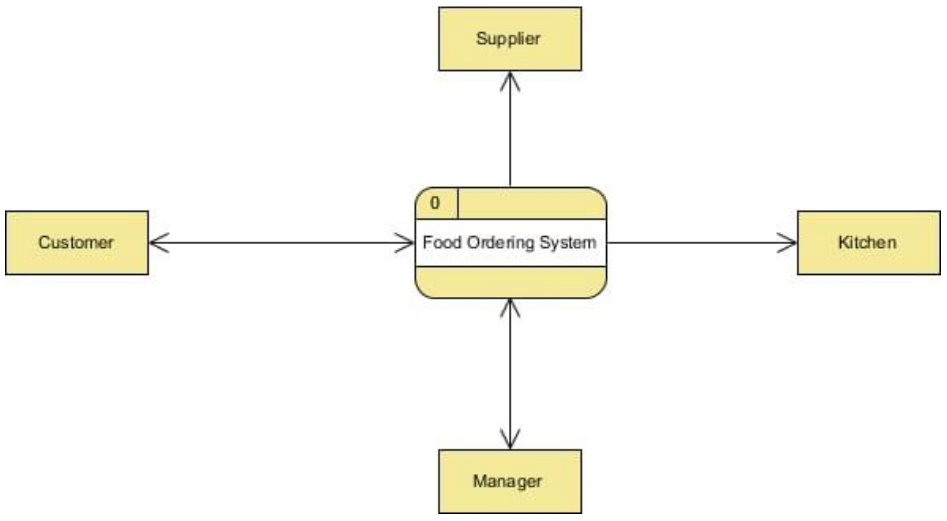
Use Case Table & Diagram:

- **Use Case Table:**
  - Diagram:

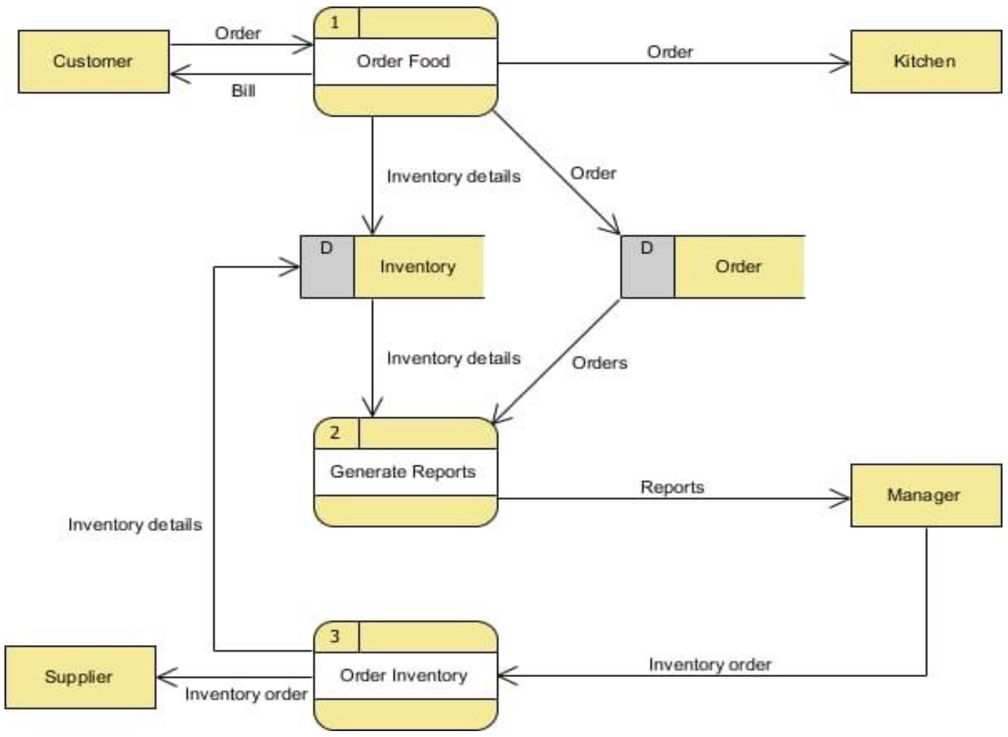


Data Flow Diagram 0-level,1-level & 2-level

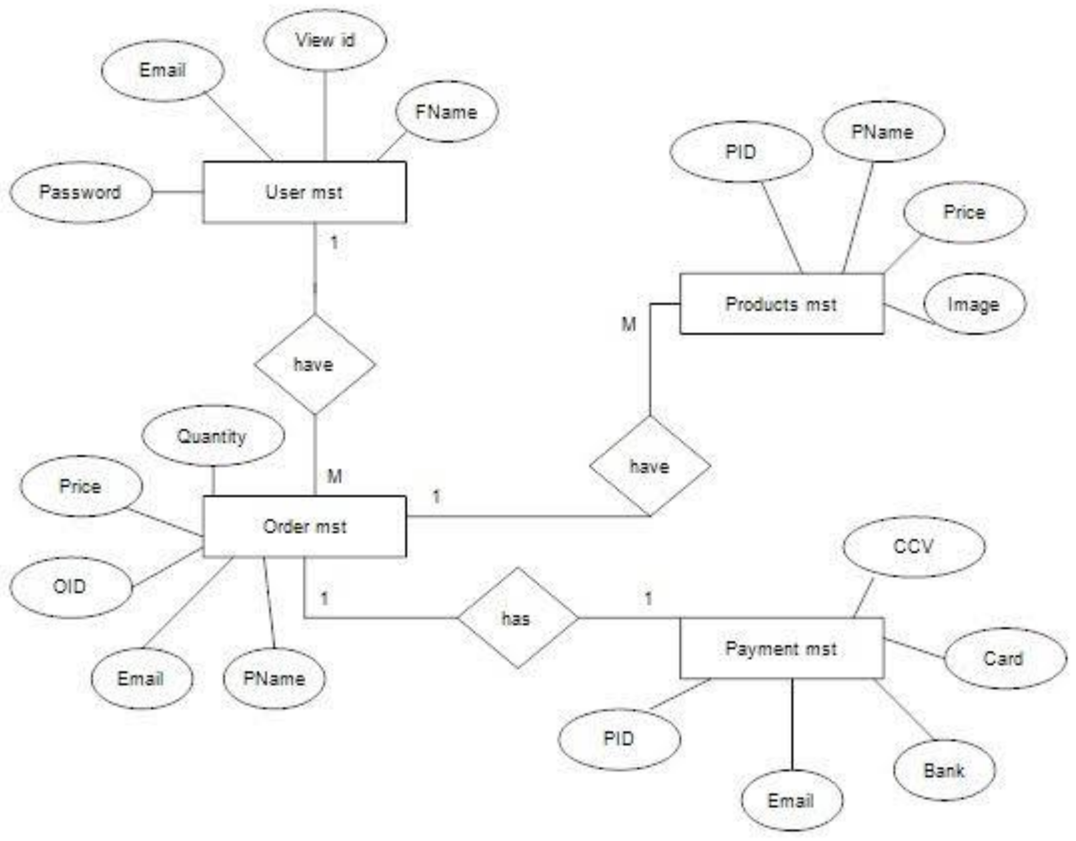
- **Data flow Diagram 0 level:-**



- **Data flow Diagram 1 &2 level:-**



Er diagram



Requirement Reviews:

Review Submission:

- Validate that users can submit reviews for restaurants and food items.
- Ensure that the submission process is intuitive and straightforward.

Rating System:

- Validate that users can assign ratings to restaurants and food items.
- Ensure that the rating system accurately reflects user satisfaction.

Text Input:

- Validate that users can provide textual feedback along with ratings.
- Ensure that there are no restrictions on the length or format of the text input.

Authentication and Permissions:

- Validate that only registered users can submit reviews.
- Ensure that users can only edit or delete their own reviews.

Review Criteria:

- Validate that users can review restaurants based on factors like food quality, delivery time, etc.
- Ensure that users can review food items based on factors like taste, portion size, etc.

Rating Aggregation:

- Validate that aggregate ratings accurately represent the overall user sentiment.
- Ensure that the rating calculation takes into account factors like the number of reviews and recency.

Review Management:

- Validate that administrators can moderate reviews effectively.
- Ensure that users can report inappropriate or fake reviews.
- Integration with Other Features:
- Validate that reviews influence search results and recommendations.
- Ensure that restaurants receive notifications of new reviews.

Test Case Development:

Review Submission:

- Test submitting reviews for restaurants and food items.
- Verify that the submission process is smooth and error-free.

Rating System:

- Test assigning ratings to restaurants and food items.
- Verify that ratings are accurately reflected in the system.

Text Input:

- Test providing textual feedback along with ratings.
- Verify that there are no restrictions on the length or format of the text input.

Authentication and Permissions:

- Test submitting reviews as both registered and unregistered users.
- Verify that only registered users can edit or delete their own reviews.

SRS Checklist with tick :

Introduction:

o [✓] Overview of the project
o [✓] Purpose of the document
o [✓] Scope of the system

General Description:
o [✓] Background information
o [✓] Objectives of the system
o [✓] Assumptions and constraints

Specific Requirements:
o [✓] Functional requirements (what the system should do)
o [✓] Nonfunctional requirements (how the system should perform) o [✓] External interfaces (how the system interacts with other systems)
o [✓] System features (specific functionalities)

Use Cases:
o [✓] List of all possible interactions with the system
o [✓] Use case diagrams illustrating system behavior
o [✓] Detailed descriptions of each use case

User Stories:
o [✓] Narratives describing system features from an end-user perspective
o [✓] Acceptance criteria for each user story

Data Requirements:
o [✓] Data entities used in the system [✓] Data attributes for each entity
o [✓] Relationships between data entities

System Models:
o [✓] Diagrams (e.g., class diagrams, sequence diagrams) illustrating system structure and behavior
o [✓] State transition diagrams for relevant system components

Quality Attributes:
o [✓] Performance requirements (response times, throughput)
o [✓] Reliability requirements (system availability, fault tolerance)
o [✓] Security requirements (access control, data • encryption)
o [✓] Usability requirements (user interface design, accessibility)

Constraints:
o [✓] Regulatory constraints (compliance with insurance regulations) o [✓] Technical constraints (compatibility with existing systems)
o [✓] Budget and time constraints

Appendices:
o [✓] Glossary of terms used in the document
o [✓] References to external documents or standards

Feel free to use this checklist as a guide to ensure that all necessary information is included in your SRS document for the vehicle insurance project.

Project Estimation-Software size:

1)Expert Judgment: In this technique, a group of experts in the relevant field estimates the project size based on their experience and expertise. It's useful when there is limited information available about the project.

2)Analogous Estimation: Estimate the project size based on similarities between the current project(Zomato) and previously completed projects. Historical data from similar systems can guide the estimation.

3)Bottom-up Estimation: Divide the project into smaller modules or tasks, estimate each task separately, and the aggregate the estimates to arrive at the overall project size.

4)Three-point Estimation: Use three values(optimistic, pessimistic, and most likely) to estimate the project size. These values are then used to calculate the expected project size using a formula like thr PERT formula.

5)Function Points: Estimate the project size based on the functionality provided by the software. Factors such as inputs, outputs, inquiries, and files are considered.

6)Use Case Points: Estimate the project size based on the number of use cases that the software must support. Factors include the complexity of each use case, the number of actors involved, and the total number of use cases.

7)Parametric Estimation: Use mathematical modes founded on project parameters and historical data for precise size estimation.

8)COCOMO(Constructive Cost Model): An algorithmic model that estimates effort, time, and cost in software development projects by considering various elements.

9)Wideband Delphi: A consensus-based estimating method the combines expert estimates from anonymous experts with cooperative conversations.

10)Monte Carlo Simulation: Useful for complicated and unpredictable projects, this technique estimates project size and analyzes risks using statistical methods and random sampling.

Project Estimation-Effort Estimation

1)Functional Requirements:

- Identify the features and functionalities needed for the system. These include user registration, menu browsing, order placement, payment processing, delivery tracking, and notifications.
- Estimate the effort required for each feature based on its complexity, dependencies, and integration points.

2)Technology Stack:

- Choose the technology stack (programming languages, frameworks, databases, etc.) for development.
- Consider the team’s expertise with the chosen stack and any learning curve.

3)Development Effort:

- Estimate the coding effort for each module or component.
- Consider factors like data modeling, API development, UI design, and integration with external services (e.g., payment gateways, maps).

4)Testing Effort:

- Estimate the time needed for unit testing, integration testing, and user acceptance testing.
- Include bug fixing and regression testing efforts.

5)Deployment and Infrastructure:

- Estimate the effort for deploying the system to production servers.
- Consider setting up cloud infrastructure, configuring load balancers, and ensuring scalability.

6)Data Migration and Seeding:

- Estimate the effort for migrating existing data (if any) and seeding initial data (restaurants, menus, etc.).

7)Delivery Partner Integration:

- Estimate the effort for integrating with delivery partners (DPs).
- Consider real-time tracking, notifications, and route optimization.

8)Food Preparation Time (FPT) Prediction:

- Zomato uses predictive models to estimate FPT12.
- Effort estimation includes developing and training these models.

9)Documentation and Training:

- Estimate the effort for creating user manuals, technical documentation, and training materials.
-

10)Project Management and Communication:

- Consider effort for project management, meetings, and communication within the team and with stakeholders.

Time Estimation

1) Food Preparation Time (FPT):

- FPT refers to the time it takes for the restaurant to prepare the food once an order is placed.
- Factors affecting FPT include the complexity of the dish, kitchen capacity, and optimization of food preparation processes.
- For example, a restaurant specializing in biryanis may have a shorter FPT for chicken biryani compared to a multi-cuisine restaurant.

2)Delivery Partner (DP) Pick-Up Time:

- Once the food is ready, the delivery partner (DP) arrives at the restaurant to pick up the order.
- DP pick-up time depends on the distance between the restaurant and the DP's location, traffic conditions, and other external factors.

3)DP Drop Time:

- After picking up the order, the DP delivers it to the customer's address.
- DP drop time considers the distance between the restaurant and the customer, traffic, and any unforeseen delays.

4)Predictable and Unpredictable Factors:

- Predictable factors include restaurant load, order volume, and historical data.
- Unpredictable factors may include traffic accidents, weather conditions, or unexpected delays.

5)Machine Learning Models:

- Zomato uses machine learning models to predict FPT and optimize delivery time.
- These models consider various features, including restaurant type, cuisine, location, and historical data.

Cost Estimation

1)Development Team:

- The size and expertise of the development team impact costs. A larger team with specialized roles (backend developers, frontend developers, UI/UX designers, testers) may increase expenses.
- Outsourcing vs. in-house development also affects costs.

2)Features and Complexity:

- The more features you want to include (searching menus, order placement, real-time tracking, payment integration, etc.), the higher the cost.
- Complex features like predictive food delivery time estimation may require additional effort.

3)Technology Stack:

- Choosing the right technology stack (backend, frontend, database, cloud services) affects development costs.
- Integration with map providers (for delivery route optimization) and payment gateways adds to expenses.

4)Design and User Experience:

- UI/UX design impacts user satisfaction. Investing in a user-friendly interface is essential.
- Design costs depend on the level of customization and visual appeal desired.

5)Testing and Quality Assurance:

- Rigorous testing ensures a bug-free system. QA efforts contribute to costs.
- Security testing, load testing, and performance testing are crucial.

6)Deployment and Infrastructure:

- Deploying the system to servers (cloud or on-premises) involves setup costs.
- Scalability considerations affect infrastructure costs.

7)Maintenance and Support:

- Post-launch maintenance, updates, and customer support require ongoing investment.

8)Legal and Compliance:

- Legal aspects (licenses, terms of use, privacy policies) have associated costs.

9)Predictive Models (Optional):

- If you want to predict food delivery time using machine learning models, development and training costs apply.

10)Location and Development Partner:

- Development costs vary by region. For example, Indian development rates may differ from those in the United States.

- **Cost Range:**
  - **Basic App:** A simple food delivery app with essential features may cost around $10,000 to $20,0001.
  - **Advanced Features**: If you add more advanced features, such as predictive models, the cost could go up to $40,000 or beyond.

Hardware Requirement

1) Servers and Hosting:

- **Web Servers:** These handle HTTP requests from users and serve web pages.
- **Database Servers**: Required for storing user data, restaurant details, menus, and order history.

- **Cloud Hosting:** Consider using cloud platforms (e.g., AWS, Azure, Google Cloud) for scalability and reliability.

2)Network Infrastructure:

- **Load Balancers:** Distribute incoming traffic across multiple servers to prevent overload.
- **Firewalls and Security Appliances:** Protect against unauthorized access and attacks.
- **Content Delivery Networks (CDNs):** Improve content delivery speed by caching static assets.

3)Storage Solutions:

- **Database Storage:** Choose a robust database system (e.g., MySQL, PostgreSQL, MongoDB) based on scalability and data consistency requirements.
- **File Storage:** Store images, menus, and other files efficiently (e.g., Amazon S3).

4)Processing Power:

- **CPU and RAM:** Sufficient processing power to handle concurrent requests during peak hours.
- **Caching Mechanisms:** Implement caching (e.g., Redis) to reduce database load.

5)Networking Components:

- **Routers and Switches:** Ensure reliable communication between servers.
- **Internet Service Providers (ISPs):** High-speed internet connectivity for seamless user experience.

6)Mobile Devices and Browsers:

- Optimize for various devices (smartphones, tablets) and browsers (Chrome, Safari, Firefox).

7)Payment Gateways:

- Integration with payment providers (e.g., Stripe, PayPal) requires secure communication channels.

8)Geolocation Services:

- APIs for location-based services (finding nearby restaurants, tracking orders).

9)Monitoring and Analytics Tools:

- **Monitoring Software:** Monitor server health, response times, and resource utilization.
- **Analytics Tools:** Gather insights on user behavior, order patterns, and performance.

10)Backup and Disaster Recovery:

- Regular backups and a disaster recovery plan are crucial for data safety.

Gantt Chart

Introduction:

Online food ordering apps have become increasingly popular due to the rise in working professionals and busy lifestyles. These apps allow local hotels, restaurants, chefs, and canteens to deliver take-away and food parcels directly to consumers' doorsteps. Zomato, along with other platforms like Swiggy, plays a significant role in this industry.

Key Features of Online Food Ordering Apps:

1)Convenience: Consumers can order food with just a click of a button, eliminating the need for physical menus or pamphlets.

2)Privacy: Since there is no huma intervention, online ordering apps provide privacy for users.
3)Accessibility: Apps can be easily downloaded to smartphones, making them accessible anytime and anywhere.

4)Payment Options: Various payment methods, including credit cards, debit cards, cashless accounts, and free home delivery, enhance user experience.

5)Additional Benefits: Apps often offer discounts, order history, palette suggestions, and customer reviews on restaurants and dishes.

Zomato's Impact

Zomato, one of the most popular food delivery apps, has revolutionized the way people order food. Its marketing mix, positioning strategies, and competitive analysis contribute to its success2. As urbanization continues in India, online food delivery applications are thriving, and the future of online food ordering websites looks promising.

Gantt Chart for Online Food Ordering System

Creating a Gantt chart can help visualize the project timeline for developing an online food ordering system. Here's a simplified example of tasks involved:

1)Requirement Gathering: Understand user needs and system requirements.

2)Design and Development:

- Design the app interface.
- Develop the backend system.
- Implement payment gateways.

3)Testing and Quality Assurance:

- Test the app thoroughly.
- Fix any issues.

4)Deployment and Launch:

- Deploy the app.
- Launch it for users.

5)Ongoing Maintenance and Updates:

- Regularly update the app.
- Address user feedback.

<u>Pert Chart</u>

Zomato, an Indian-based multinational company, started in 2008 as “Foodiebay” and later rebranded to its current name. It serves as a restaurant search and discovery platform, providing users with information about restaurants, menus, user ratings, and food delivery options. Here are some key details about Zomato:

- **Founders**: Deepinder Goyal, Pankaj Chaddah
- **Headquarters:** Gurgaon, Haryana, India
- **Parent Company:** Infoedge
- **Services:** Restaurant aggregation, food delivery, and user reviews
- **Market Cap:** $13.3 billion
- **Annual Revenue:** $280 million
- **Net Profit:** -$110 million (as of the latest data)

SWOT Analysis of Zomato

Let’s explore Zomato’s strengths, weaknesses, opportunities, and threats (SWOT):

1) **Strengths:**
   - **Brand Presence**: Zomato has established a strong brand in the restaurant industry.
   - **Extensive Network:** It boasts an extensive network of restaurants and cafes.
   - **Quality Content:** Zomato stands out due to its high-quality content and user-generated reviews.

2) **Weaknesses:**
   - **Discount Dependency:** Zomato heavily relies on discounts and offers, which may impact profitability.
   - **Operational Challenges:** Occasionally, operational issues affect service quality.
   - **Competition:** Intense competition from other food delivery platforms.

3) **Opportunities:**
   - **Technological Innovation**: Zomato can continue to innovate in terms of technology and user experience.
   - **Global Expansion:** Expanding to new markets beyond India.
   - **Diversification:** Exploring related services (e.g., grocery delivery).

4) **Threats:**
   - **Competition:** Rivalry with platforms like Swiggy and Uber Eats.
   - **Regulatory Changes:** Evolving regulations in the food delivery industry.
   - **Economic Factors**: Economic downturns affecting consumer spending.

Resource Histogram

1) **Resource Histograms:**
    - A resource histogram is a graphical representation that shows the distribution of resources (such as time, manpower, or computing power) over a specific period.
    - In the context of Zomato, we can create resource histograms to analyze various aspects related to food delivery and restaurant operations.

2) **Zomato Data Analysis**:
    - Researchers and data enthusiasts have explored Zomato's data to gain insights into restaurant performance, user preferences, and delivery efficiency.
    - Here are some ways resource histograms could be useful:

        - **Delivery Time Distribution:**
        - Create a histogram showing the distribution of delivery times for different restaurants.
        - Identify peak delivery hours and allocate resources accordingly (e.g., more delivery personnel during busy hours).

        - **Restaurant Ratings vs. Resources:**
        - Analyze the relationship between restaurant ratings and the resources allocated (e.g., staff, kitchen equipment).
        - Are highly-rated restaurants more efficient in resource utilization?

        - **User Reviews and Response Time:**
        - Investigate how quickly Zomato responds to user reviews.
        - Plot a histogram of response times to identify bottlenecks or areas for improvement.

        - **Cuisine Popularity:**
        - Create histograms for different cuisines to understand their popularity.
        - Allocate resources based on demand (e.g., more resources for popular cuisines).

3) **Example Project:**
    - Some data enthusiasts have created Power BI dashboards for Zomato analysis.
    - One such project involved data extraction, cleaning, and modeling from Zomato's API.
    - The resulting dashboards provided actionable insights for Zomato stakeholders.

4) **ER Diagrams:**
    - Designing an Entity-Relationship (ER) diagram for Zomato involves identifying entities (e.g., restaurants, users, orders), defining attributes, and establishing relationships.
    - ER diagrams help streamline database desing and improve system efficiency.

Staff Required

1) Back-end Service Agent:

- Responsible for maintaining the restaurant listings and ensuring accurate information.
- Handles updates related to restaurant menus, availability, and operational details.

2) Restaurant Owners:

- These are the food providers (restaurants, cafes, etc.).
- They manage their food listings, pricing, and availability on the platform.
- Collaborate with Zomato to ensure seamless operations.

3) Customers:

- The end-users who place food orders through the app.
- They explore menus, select dishes, and make payments.
- Provide feedback and ratings.

4) Delivery Executives:

- The backbone of the delivery process.
- Responsible for picking up orders from restaurants and delivering them to customers.
- Optimal staffing ensures timely deliveries.

5) Restaurant Staff:

- Within the restaurant, staff members handle food preparation, packaging, and order coordination.
- Chefs, kitchen staff, and order dispatchers play crucial roles.

6) Customer Support Agents:

- Address customer queries, resolve issues, and provide assistance.
- Ensure a positive user experience.

Quality Policy

1) Quality Policy Overview:

- A quality policy outlines an organization’s commitment to delivering high-quality products or services. For Zomato, this would relate to ensuring a seamless and delightfu.
- l experience for users, restaurants, and delivery partners.
- Key aspects of a quality policy include reliability, security, performance, and user satisfaction.

2) User Experience:

- Zomato's quality policy likely emphasizes a user-centric approach:
  - **Reliability:** Ensuring the app is available 24/7, with minimal downtime.
  - **Usability**: Intuitive navigation, clear menus, and easy order placement.
  - **Performance:** Fast loading times, quick search results, and smooth checkout.
  - **Security:** Protecting user data (payment details, addresses) and preventing fraud.
  - **Feedback:** Regularly collecting user feedback to improve the platform**.**

3) Restaurant Partners:

- Zomato's quality policy would extend to its restaurant partners:
  - **Menu Accuracy:** Ensuring that menu items, prices, and availability are up-to-date.
  - **Timely Updates:** Restaurants should promptly update their status (open/closed).
  - **Hygiene Standards:** Encouraging restaurants to maintain cleanliness and food safety.

4)Delivery Partners:

- Quality policy for delivery partners:
  - **Timeliness**: Ensuring timely deliveries.
  - **Professionalism:** Courteous behavior and adherence to delivery guidelines.
  - **Safety**: Safe driving practices and following traffic rules.

1) Continuous Improvement:

- Zomato likely emphasizes continuous improvement:
  - **Monitoring Metrics**: Regularly tracking key performance indicators (KPIs).
  - **Root Cause Analysis**: Investigating issues and addressing them promptly.
  - **Training:** Providing training to delivery partners and restaurant staff.

2) Compliance and Legal Aspects:

- Zomato's quality policy would align with legal requirements and industry standards.
- Compliance with food safety regulations, data privacy laws, and consumer protection laws is crucial.

<u>Scope Statement</u>

Scope Statement for Analyzing Zomato's Online Food Delivery System

1) Objective:

- The primary objective is to analyze Zomato's online food delivery platform from a software engineering perspective.
- We aim to understand its architecture, functionality, user experience, and underlying technologies.

2) Inclusions:

- **Functional Scope:**
  - User registration and login

        - Restaurant search and menu browsing
        - Order placement and payment processing
        - Delivery tracking

    - **Non-Functional Scope:**
        - Performance (response time, scalability)
        - Security (data protection, authentication)
        - Usability (user interface, navigation)
        - Reliability (system uptime, error handling)

3) Exclusions:

- Detailed financial analysis (revenue, costs)
- Marketing strategies (customer acquisition, promotions)
- Legal aspects (regulatory compliance, contracts)

4) Constraints:

- We'll focus on Zomato's public-facing features and avoid internal processes.
- Our analysis will be based on available information (publicly accessible data, research papers, etc.).

Product Description

1) Seamless Ordering Experience: Zomato Gold

- **What is it?** Zomato Gold, launched in 2017, disrupted the traditional dining experience by offering subscribers exclusive deals at partner restaurants.

- **Integration:** Zomato seamlessly integrated Zomato Gold into its existing app, making it a one-stop solution for both restaurant discovery and dining. Users could browse reviews, make reservations, and avail discounts—all within a few taps.
- **Data-Driven Decisions:** Zomato leveraged user data to curate personalized recommendations and offers, enhancing value for users and restaurants.

2) Hyper-Localized Solutions: Zomato Market

- **COVID-19 Pivot:** In response to the pandemic, Zomato launched Zomato Market in 2020. It leveraged its delivery network to offer groceries and household essentials.

- **Agility in Response:** Zomato's ability to adapt its offerings underscores its agility and customer-centric approach.

- **Ecosystem Expansion:** Zomato Market showcased potential beyond food delivery, diversifying revenue streams and onboarding new restaurant partners.

3) Enhanced User Engagement: Zomato Stories & Gen-Z Notifaction

- **Zomato Stories:** Introduced in 2019, this feature allowed users to share dining experiences through photos and short reviews, fostering a sense of community.

- **Gen-Z Lingo:** Zomato adapted Gen-Z language to attract a younger audience, enhancing engagement and making the platform more relatabl

Standards and regulations

1) Food Safety and Hygiene Standards:

- Zomato must adhere to local and national food safety regulations.
- Compliance with hygiene standards ensures safe food handling, storage, and delivery.
- Regular inspections and audits are essential.

2) Data Privacy and Security:

- Zomato collects user data (personal information, payment details, etc.).
- Compliance with data protection laws (e.g., GDPR, CCPA) is vital.
- Secure data storage, encryption, and user consent are key.

3) Delivery Vehicle Regulations:

- Zomato’s delivery fleet (bikes, cars) must meet safety and emission standards.
- Licensing, insurance, and roadworthiness are critical.

4) Labor Laws and Employment Practices:

- Fair treatment of delivery executives (wages, working hours, benefits).
- Compliance with labor laws and employment contracts**.**

5) Platform Liability and Dispute Resolution:

- Zomato’s terms of service, liability clauses, and dispute resolution mechanisms.
- Clear communication with users regarding refunds, cancellations, and complaints.

6) Accessibility and Inclusivity:

- Ensure the app is accessible to all users (including those with disabilities).
- Compliance with accessibility guidelines (e.g., WCAG).

Risk Management

a) Technology Risk

1) Threat of New Entrants:

- Zomato faces varying levels of threat from new entrants. As Amazon expands into food delivery, the competition intensifies.
- To counter this, Zomato can focus on expanding its market share by penetrating deeper into Indian cities, especially tier 3, 4, and 5 cities. Strengthening partnerships with restaurants and cafes is crucial.
- Collaborations (like Dunzo and Biryani Blues) and leveraging existing assets (such as Amazon Pay) can help Zomato maintain its competitive edge.

2) Threat of Substitution:

- The high availability of food delivery apps on the same platform poses a significant threat of substitution for Zomato.
- To address this, Zomato should differentiate itself by offering unique features and building emotional connections with customers. For instance, direct restaurant-to-customer deliveries or innovative app features can set it apart.

3) Bargaining Power of Customers:

- Zomato faces medium bargaining power from customers due to the availability of similar products.
- Acquiring more customers is essential to reduce their bargaining power. However, Zomato must ensure customer retention beyond promotional offers.

In addition to these risks, Zomato operates in 24 countries, subject to economic, political, and legal risks. Regulatory changes and market requirements may necessitate flexible adaptations

b) People Risk

1) Delivery Personnel Risks:

- Zomato relies heavily on its delivery partners (riders) to ensure timely and accurate food delivery.

- **Risk:** High attrition rates among delivery personnel can disrupt operations and impact customer satisfaction**.**
- **Mitigation:** Implement robust recruitment, training, and retention strategies. Regularly assess rider satisfaction and address their concerns.

2) Quality Control Risks:

- Ensuring consistent food quality is crucial for customer trust.

- **Risk:** Inadequate quality checks during food preparation or packaging can lead to customer complaints**.**
- **Mitigation**: Regular audits, hygiene standards, and clear guidelines for

restaurants and delivery partners are essential.

3) Customer Service Risks:

- Zomato's customer service team handles queries, complaints, and refunds.

- **Risk:** Inefficient or unresponsive customer service can harm Zomato's reputation.
- **Mitigation:** Invest in well-trained customer support staff, efficient ticket resolution, and proactive communication with users.

4) Restaurant Partner Risks:

- Zomato collaborates with restaurants for food supply.

- **Risk**: Disputes, delays, or quality issues with restaurant partners can affect Zomato's service.
- **Mitigation:** Clear contracts, performance monitoring, and regular feedback loops with restaurants are essentia**.**

5) Technology Risks:

- Zomato's app and platform rely on technology for seamless ordering and delivery.

- **Risk:** System failures, cyber threats, or data breaches can disrupt operations**.**
- **Mitigation:** Robust cybersecurity measures, regular testing, and backup systems are critical**.**

6) Regulatory and Compliance Risks:

- Zomato operates in multiple countries, each with its regulations.

- **Risk:** Non-compliance with local laws can lead to fines or legal issues.
- **Mitigation:** Legal teams, compliance checks, and proactive adaptation to changing regulations are necessary

c) <u>Organizational Risk</u>

1) Cybersecurity Threats:

With a vast amount of user data and financial transactions, Zomato faces the risk of cyberattacks, data breaches, and hacking attempts, which could compromise customer information and damage trust.

2) Food Safety and Quality Control:

Ensuring that partner restaurants adhere to food safety standards is crucial for Zomato to maintain customer satisfaction and prevent incidents related to foodborne illnesses or quality issues.

3) **Logistical Challenges:**
Timely delivery is essential for customer satisfaction. Any disruptions in the supply chain, such as traffic congestion, weather conditions, or driver shortages, could impact Zomato's ability to deliver orders promptly.

4) **Regulatory Compliance:**
Zomato operates in multiple jurisdictions, each with its own regulations regarding food safety, labor practices, and data protection. Compliance with these regulations is necessary to avoid legal issues and penalties.

5)Market Competition:
The online food delivery industry is highly competitive, with numerous players vying for market share. Zomato faces the risk of losing customers to competitors who offer better services, prices, or incentives**.**

6)Dependency on Partners:
Zomato relies on a network of partner restaurants, delivery partners, and technology providers. Any issues with these partners, such as disputes, service interruptions, or quality concerns, could affect Zomato's operations**.**

7)Brand Reputation:
Negative publicity, whether due to food safety incidents, poor customer service, or ethical lapses, can significantly damage Zomato's brand reputation and erode consumer trust.

8)Economic Factors:
Economic downturns, inflation, or fluctuations in consumer spending patterns can impact Zomato's revenue and profitability, as consumers may cut back on discretionary spending, including ordering food delivery.

Addressing these risks requires a comprehensive risk management strategy that includes proactive measures to mitigate vulnerabilities, robust contingency plans for potential crises, and ongoing monitoring and evaluation to adapt to changing conditions.

d) <u>Tool Requirement Risk</u>

1) **Dependency on Technology:**
Zomato relies heavily on its website, mobile app, and backend systems to facilitate orders, payments, and logistics. Any disruptions or failures in these systems, whether due to technical issues, server outages, or software bugs, could adversely impact operations and customer experience.

2) **Scalability Challenges:**
As Zomato expands its user base and geographic reach, it needs to ensure that its technology infrastructure can scale accordingly to handle increased traffic, order

volumes, and data processing requirements. Failure to scale effectively could lead to performance issues and service disruptions.

3) Integration Complexity:
Zomato's platform integrates various components, including restaurant management systems, payment gateways, and delivery tracking software. Ensuring seamless integration and compatibility among these components can be challenging, especially as new features and functionalities are introduced.

4)Data Security and Privacy:
With the collection and processing of sensitive customer data, Zomato must prioritize data security and privacy. Any breaches or vulnerabilities in its systems could result in unauthorized access to personal information, financial data, or proprietary business data, leading to legal and reputational consequences.

5)Vendor Reliability:
Zomato may rely on third-party vendors for hosting, cloud services, and software solutions. The reliability and performance of these vendors can impact Zomato's operations and uptime. Issues such as service outages, contract disputes, or vendor bankruptcies could disrupt services and require contingency plans.

To mitigate tool requirement risks, Zomato should invest in robust technology infrastructure, including redundant systems for high availability and disaster recovery, ongoing monitoring and maintenance to identify and address issues proactively, regular security assessments and audits to safeguard data, and strategic partnerships with reliable technology vendors. Additionally, investing in talent and expertise to manage and innovate the technology stack is essential for long-term success.

e) <u>Estimation Risk</u>

1) Demand Forecasting:
Zomato must estimate future demand for its food delivery services accurately to optimize inventory management, staffing, and resource allocation. Factors such as seasonal fluctuations, changing consumer preferences, and competitive dynamics can make demand forecasting challenging.

2) Financial Projections:
Estimating future revenue, expenses, and profitability is crucial for Zomato's financial planning and decision-making. However, uncertainties in market conditions, regulatory changes, and operational performance can affect the accuracy of financial projections.

3) Expansion Planning:
Zomato's growth strategy relies on expanding into new markets and diversifying its services. Estimating the potential success and profitability of new ventures, partnerships, or acquisitions involves inherent risks due to market variability, competitive pressures, and unforeseen challenges.

4) **Technology Investments:**
Estimating the costs, benefits, and timeline of technology investments, such as developing new features or upgrading infrastructure, requires careful analysis. Changes in technology trends, project scope, and implementation challenges can impact the accuracy of these estimates.

5) **Risk Assessment:**
Assessing and quantifying risks, such as cybersecurity threats, operational disruptions, or regulatory compliance issues, involves estimating the likelihood and potential impact of adverse events. However, uncertainties in risk assessment methodologies and evolving risk landscapes can introduce estimation risks.

To mitigate estimation risks, Zomato can employ several strategies:

- Conduct thorough research and analysis to gather relevant data and insights for informed decision-making.
- Use scenario planning and sensitivity analysis to evaluate the impact of different assumptions and variables on outcomes.
- Implement agile and iterative approaches to adapt to changing circumstances and refine estimates over time.
- Foster a culture of transparency and open communication to encourage collaboration and collective problem-solving.
- Monitor key performance indicators (KPIs) and performance metrics regularly to track actual results against projected estimates and adjust strategies accordingly.

By addressing estimation risks proactively and adopting a flexible and adaptive mindset, Zomato can enhance its ability to navigate uncertainties and achieve its business objectives effectively.

❖ sub systems of ZOMATO and their relationship with each other.

1. User Interface (UI):

- The front-end interface that users interact with. It includes screens for browsing restaurants, viewing menus, placing orders, tracking deliveries, and managing user profiles.
- Interfaces with the backend to retrieve and display relevant data such as restaurant information, menu items, order status, etc.

2. Restaurant Management System:

- Manages restaurant profiles, including details like menu items, pricing, operating hours, and availability.
- Allows restaurants to update their information, manage orders, and track performance metrics.
- Interfaces with the main system to receive incoming orders and update order status.

3. Order Management System:

- Handles the entire lifecycle of an order, from placement to delivery.
- Tracks order status, communicates with users and delivery partners regarding order updates.
- Interfaces with the restaurant management system to receive menu and availability information and with the delivery system to assign delivery partners.

4. Delivery System:

- Manages the logistics of delivering orders from restaurants to customers.
- Assigns delivery partners to orders based on factors like proximity, availability, and delivery preferences
- Provides real-time tracking of delivery status to users and handles communication between users, restaurants, and delivery partners.

5. Payment Gateway:

- Facilitates secure online payments for orders.
- Interfaces with banking systems or third-party payment processors to process transactions securely.
- Ensures smooth and reliable payment processing for both users and restaurants.

6. Rating and Review System:

- Allows users to rate and review restaurants based on their experience.
- Provides feedback to both users and restaurants to improve service quality.
- Interfaces with user profiles and restaurant profiles to display ratings and reviews appropriately.

7. Search and Recommendation Engine:

- Helps users discover restaurants and dishes based on various criteria like cuisine type, location, popularity, and user preferences.
- Utilizes algorithms to provide personalized recommendations and improve user experience.
- Interfaces with the main system to fetch restaurant and menu data and with user profiles to tailor recommendations.

❖ an abstract specification of subsystems for a Zomato-like food delivery application:

1. **User Management System:**

   - **Authentication:** Handles user registration, login, and authentication processes.
   - **Profile Management:** Allows users to view and update their profiles, including personal information, delivery addresses, and payment methods.
   - **Notifications:** Manages notifications to users regarding order updates, promotions, and other relevant information.

2. **Restaurant Management System:**

   - **Menu Management:** Enables restaurants to add, update, and remove menu items, including prices, descriptions, and images.
   - **Order Management:** Facilitates the processing of incoming orders, order status updates, and communication with delivery partners.
   - **Analytics:** Provides insights to restaurants regarding order volumes, popular items, and other relevant metrics to optimize their operations.

3. **Order Management System:**

   - **Order Placement:** Allows users to browse restaurants, select items, customize orders, and place orders.
   - **Payment Processing:** Integrates with payment gateways to securely process payments from users.
   - **Order Tracking:** Enables users to track the status of their orders in real-time, from confirmation to delivery.

4. **Delivery Management System:**

   - **Dispatching:** Assigns delivery personnel to incoming orders based on proximity, availability, and workload.
   - **Route Optimization:** Optimizes delivery routes to minimize delivery time and distance.
   - **Tracking:** Allows users to track the location of their delivery in real-time and provides estimated delivery times.

Rating and Review System:

- Allows users to rate and review restaurants and delivery experiences.
- Aggregates and displays ratings and reviews to help users make informed decisions.
- Provides feedback to restaurants and delivery partners for performance improvement.

5. Admin Dashboard:

- Centralized management interface for platform administrators.
- Provides tools for user and restaurant management, order monitoring, and system configuration.
- Generates reports and analytics to monitor platform performance and identify areas for improvement.

6. Customer Support System:

- Enables users to contact customer support for assistance with orders, payments, or other inquiries.
- Provides customer support agents with tools for managing and resolving user issues efficiently.
- Integrates with other subsystems to access order and user information for troubleshooting.

❖ Interface design

User Interface (UI):

- Abstract: Provides a graphical interface for users to interact with the app.
- Responsibilities:
  - Display restaurant listings, menus, and order options.
  - Facilitate user registration, login, and profile management.
  - Present order tracking, payment options, and customer support access.
- Interfaces:
  - Receives user input for orders, registration, and profile updates.
  - Communicates with other subsystems to fetch and display relevant data.

Authentication and User Management:

- Abstract: Manages user authentication and profile information.
- Responsibilities:
  - Authenticate users during login and registration processes.
  - Manage user profiles, including personal information and order history.
  - Ensure data security and privacy compliance.
- Interfaces:
  - Authenticates user credentials from the UI.
  - Provides user profile data to other subsystems upon request.

Restaurant Management:

- Abstract: Stores and provides information about restaurants available on the platform.
- Responsibilities:
  - Maintain a database of restaurants, including menus, locations, and ratings.
  - Update restaurant information based on user feedback and partner updates.
  - Ensure data accuracy and consistency.
- Interfaces:
  - Supplies restaurant data to the UI for display.
  - Receives updates from the UI regarding user interactions and feedback.

Order Management:

- Abstract: Handles the processing and tracking of user orders.
- Responsibilities:
  - Receive and validate user orders.
  - Coordinate with restaurants for order preparation and confirmation.
  - Provide real-time order tracking updates to users.
- Interfaces:
  - Receives order requests from the UI.
  - Communicates with the Restaurant Management subsystem for order fulfillment.

Payment Gateway:

- Abstract: Facilitates secure payment processing for user orders.
- Responsibilities:
  - Process payment transactions using various payment methods.
  - Ensure transaction security and compliance with payment regulations.
  - Confirm payment status to users and order management subsystem.
- Interfaces:
  - Receives payment requests from the UI.
  - Communicates with the Order Management subsystem for order validation.

Delivery Management:

- Abstract: Coordinates the delivery process from restaurant to user.
- Responsibilities:
  - Assign delivery personnel to fulfill orders.
  - Optimize delivery routes and timings.
  - Provide real-time delivery tracking updates to users.
- Interfaces:
  - Receives order information from the Order Management subsystem.
  - Communicates with delivery personnel and updates delivery status in the UI.

Rating and Review System:

- Abstract: Manages user feedback and ratings for restaurants.
- Responsibilities:
  - Collect user reviews and ratings for restaurants.
  - Aggregate and analyze feedback to generate restaurant ratings.
  - Display reviews and ratings on the UI for user reference.
- Interfaces:

    - Receives review submissions from the UI.
    - Provides restaurant ratings and reviews to the UI for display.

Notification System:

- Abstract: Sends notifications to users regarding order status and updates.
- Responsibilities:
    - Generate and send notifications for order confirmations, status changes, and promotions.
    - Support multiple notification channels (e.g., push notifications, emails).
- Interfaces:
    - Triggers notifications based on events from other subsystems.
    - Communicates with the UI to display notifications to users.

Customer Support:

- Abstract: Provides assistance and resolves user queries and issues.
- Responsibilities:
    - Offer customer support channels such as live chat, FAQs, and help articles.
    - Handle user inquiries, complaints, and feedback.
- Interfaces:
    - Receives user queries and feedback from the UI.
    - Provides support responses and updates to the UI.

Analytics and Insights:

- Abstract: Analyzes user data to derive insights for business decision-making.
- Responsibilities:
    - Collect user interaction data from various subsystems.
    - Analyze data to identify trends, user behaviors, and business opportunities.
    - Generate reports and recommendations for business optimization.
- Interfaces:
    - Retrieves data from other subsystems for analysis.
    - Communicates insights and recommendations to stakeholders.

❖ Software Engineering Report on Zomato Database Design

Zomato, like many other modern web-based platforms, likely employs a complex and scalable database design to handle its vast amount of data efficiently. While I don't have access to Zomato's specific internal architecture, I can outline a hypothetical database design that aligns with the typical requirements of a food delivery and restaurant discovery platform like Zomato.

1. User Data:

- This includes information about users such as their usernames, email addresses, passwords (ideally hashed for security), profile pictures, contact information, and any other relevant details. This data is crucial for user authentication, personalization, and communication.

2. Restaurant Data:

- This comprises details about restaurants listed on the platform. Information such as restaurant names, addresses, contact details, operational hours, cuisines served, average ratings, reviews, menus, and images are stored here. Each restaurant entry might also have a unique identifier for easy retrieval.

3. Menu Items:

- This table contains details about the various items available on the menus of different restaurants. It includes attributes such as item names, descriptions, prices, and possibly dietary information or tags (e.g., vegetarian, gluten-free).

4. Orders:

- When a user places an order, the details of that order are stored in this table. This includes the user who placed the order, the restaurant from which the order was placed, the items ordered, quantities, prices, order status, delivery address, and timestamps for order creation and fulfillment.

5. Reviews and Ratings:

- Users can leave reviews and ratings for restaurants and possibly individual menu items. This data is essential for users to make informed decisions and for restaurants to gather feedback. Each review typically includes the user who left the review, the restaurant being reviewed, the rating, and optional text feedback.

6. Geolocation Data:

- Given the importance of location-based services, Zomato likely maintains data related to geographic locations. This can include latitude and longitude coordinates for restaurants and users, as well as mapping information to facilitate search and discovery based on proximity.

7. Payment Information:

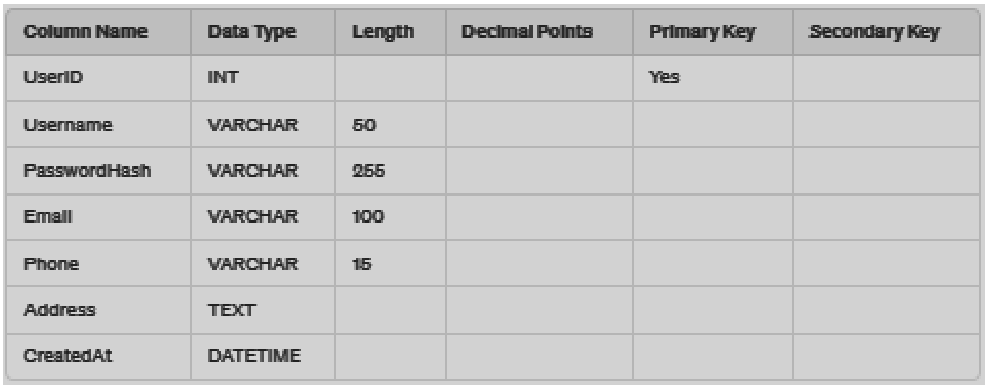
- Since Zomato facilitates transactions, it needs to securely store payment information. This includes details of users' payment methods (such as credit card numbers or digital wallets) and transaction records.

8. Analytics and Logs:

- To monitor platform usage, performance, and user behavior, Zomato likely maintains logs and analytics data. This can include information on website traffic, user interactions, popular restaurants, peak ordering times, and more.

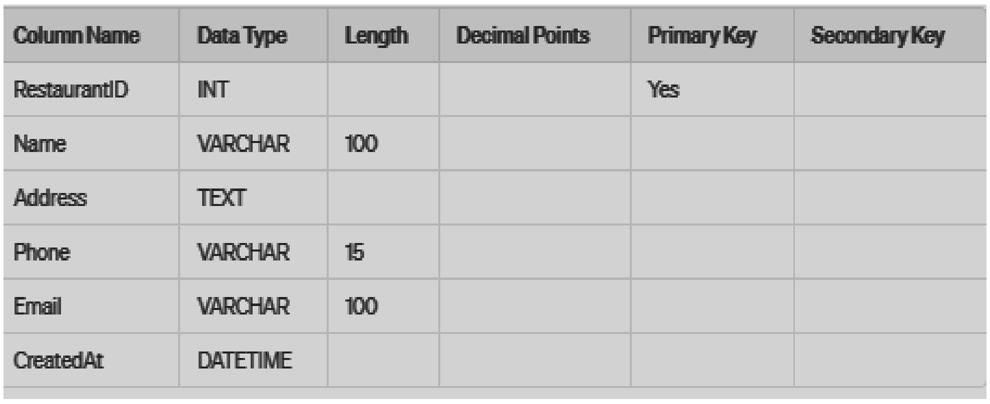
❖ **Data structure design - Database Tables. Columns (Type, length and decimal points). For each table primary key secondary key if any.**

- **User table.**



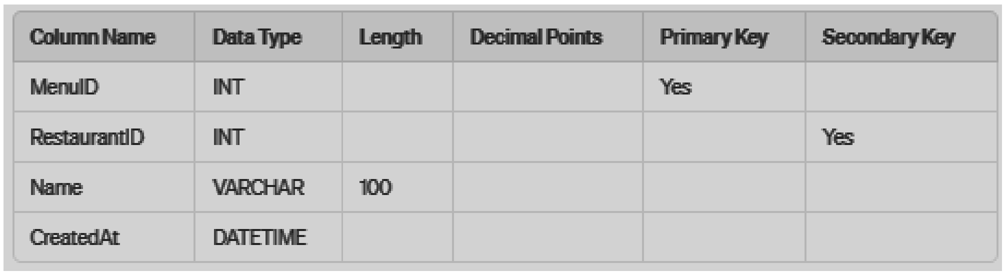
Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
UserID	INT			Yes	
Username	VARCHAR	50			
PasswordHash	VARCHAR	255			
Email	VARCHAR	100			
Phone	VARCHAR	15			
Address	TEXT				
CreatedAt	DATETIME				

- **Restaurants Table.**



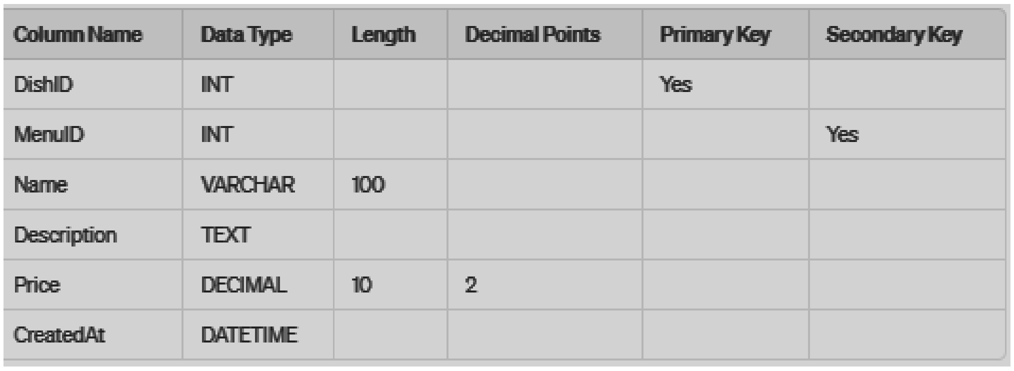
Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
RestaurantID	INT			Yes	
Name	VARCHAR	100			
Address	TEXT				
Phone	VARCHAR	15			
Email	VARCHAR	100			
CreatedAt	DATETIME				

- **Menus Table.**



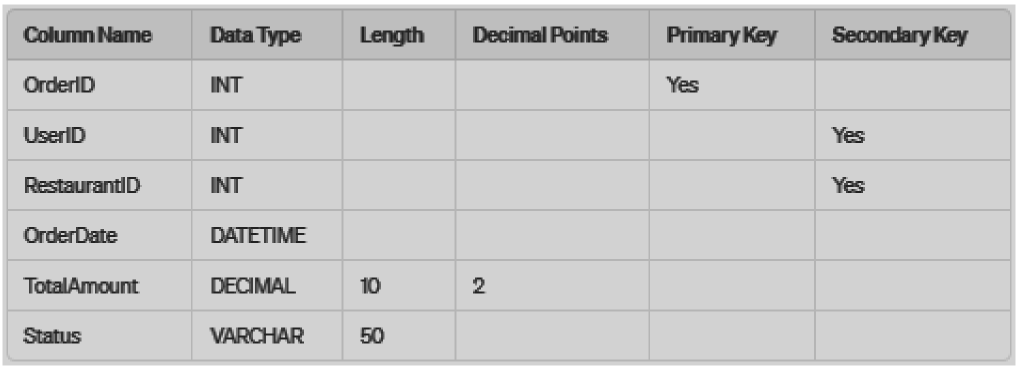
Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
MenuID	INT			Yes	
RestaurantID	INT				Yes
Name	VARCHAR	100			
CreatedAt	DATETIME				

- **Dishes Table.**



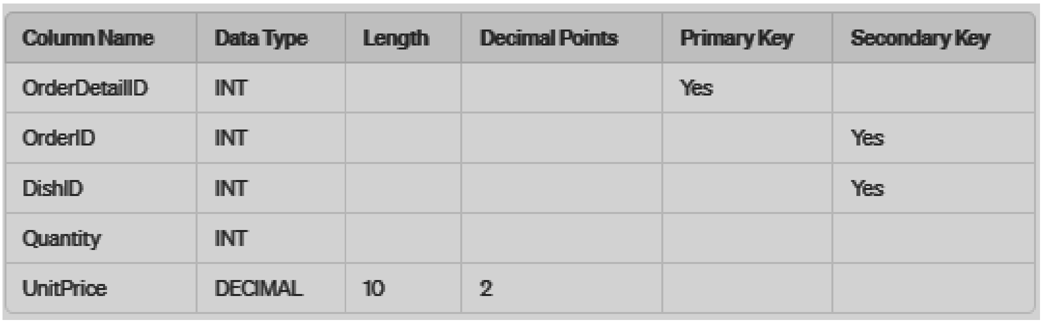
Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
DishID	INT			Yes	
MenuID	INT				Yes
Name	VARCHAR	100			
Description	TEXT				
Price	DECIMAL	10	2		
CreatedAt	DATETIME				

- **Orders Table.**



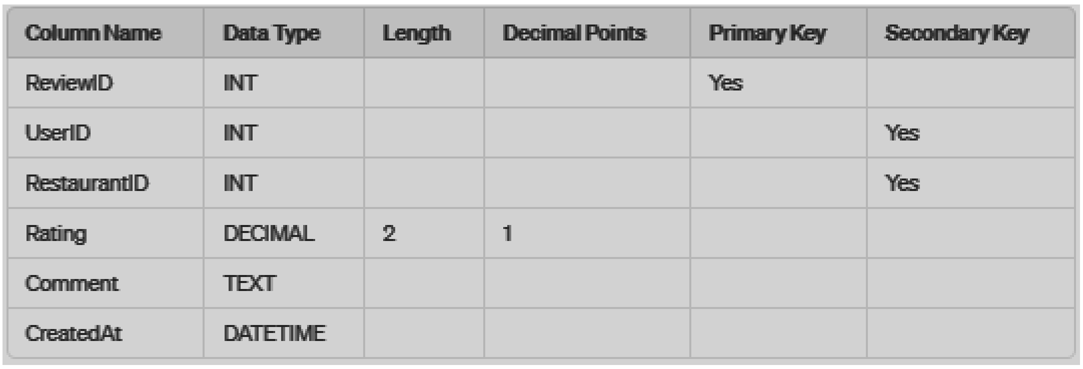
Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
OrderID	INT			Yes	
UserID	INT				Yes
RestaurantID	INT				Yes
OrderDate	DATETIME				
TotalAmount	DECIMAL	10	2		
Status	VARCHAR	50			

- **OrderDetails Table.**



Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
OrderDetailID	INT			Yes	
OrderID	INT				Yes
DishID	INT				Yes
Quantity	INT				
UnitPrice	DECIMAL	10	2		

- **Review table**

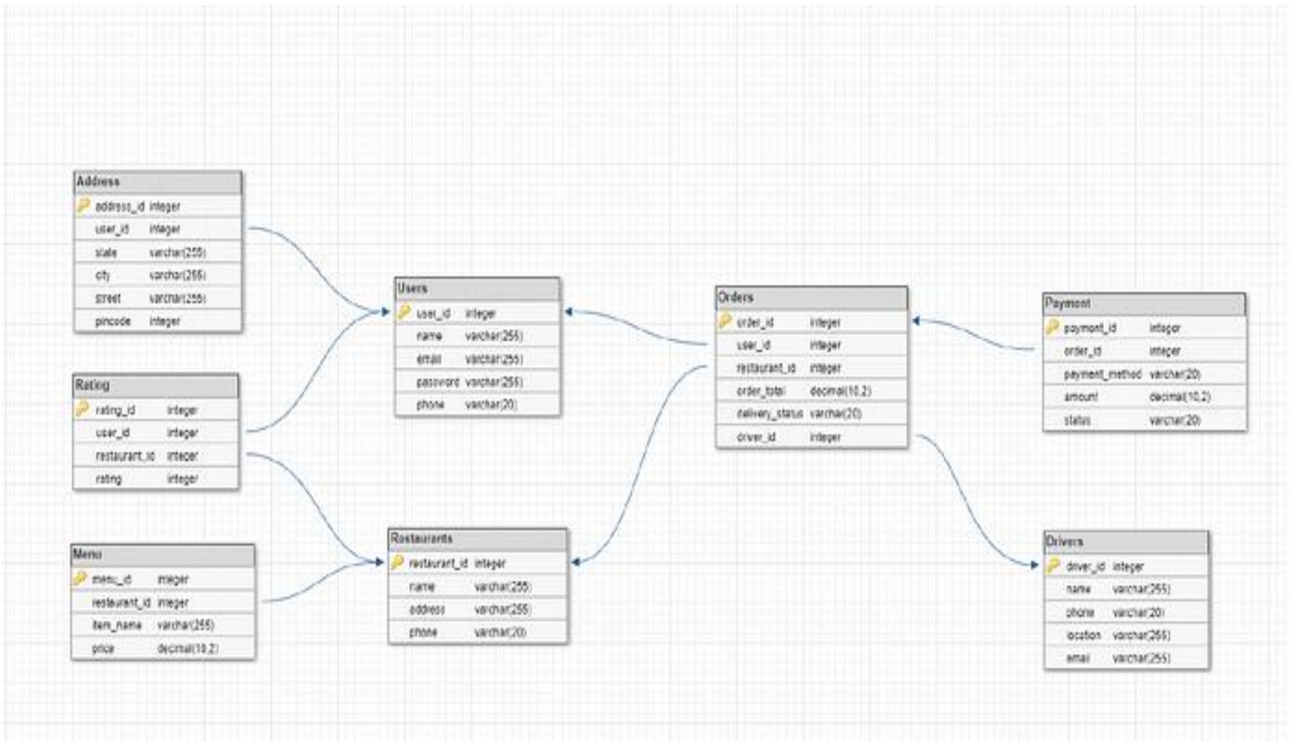


Column Name	Data Type	Length	Decimal Points	Primary Key	Secondary Key
ReviewID	INT			Yes	
UserID	INT				Yes
RestaurantID	INT				Yes
Rating	DECIMAL	2	1		
Comment	TEXT				
CreatedAt	DATETIME				

- **Explanation of Tables and Relationships**

  - **Users:** Stores user information. UserID is the primary key.
  - **Restaurants:** Stores restaurant information. RestaurantID is the primary key.
  - **Menus:** Stores menus for each restaurant. MenuID is the primary key, RestaurantID is a foreign key referencing Restaurants.
  - **Dishes:** Stores individual dishes in each menu. DishID is the primary key, MenuID is a foreign key referencing Menus.
  - **Orders:** Stores order information. OrderID is the primary key, UserID is a foreign key referencing Users, and RestaurantID is a foreign key referencing Restaurants.
  - **OrderDetails:** Stores details of each order, including dish quantities and prices. OrderDetailID is the primary key, OrderID is a foreign key referencing Orders, and DishID is a foreign key referencing Dishes.
  - **Reviews:** Stores user reviews for restaurants. ReviewID is the primary key, UserID is a foreign key referencing Users, and RestaurantID is a foreign key referencing Restaurants

- **Data Integrity and Constraints**
  - Primary Keys ensure each record is uniquely identifiable**.**
  - Foreign Keys maintain the relationships between different entities, ensuring referential integrity.
  - Data Types and Lengths are chosen to balance storage efficiency and accommodate expected data sizes.
  - Timestamps (CreatedAt) are used to track record creation dates.



❖ Structure charts for application.

```
Zomato Online Food Delivery App
|
|-- USER MANAGEMENT
|   |-- USER REGISTRATION
|   |-- USER LOGIN
|   |-- USER PROFILE
|
|-- RESTAURANT MANAGEMENT
|   |-- RESTAURANT SEARCH
|   |-- RESTAURANT LISTING
|   |-- RESTAURANT DETAILS
|   |-- MENU MANAGEMENT
|
|-- ORDERING SYSTEM
|   |-- BROWSE MENU
|   |-- ADD TO CART
|   |-- CHECKOUT
|   |-- PAYMENT PROCESSING
|   |-- ORDER TRACKING
|
|-- DELIVERY MANAGEMENT
|   |-- DRIVER ASSIGNMENT
|   |-- ORDER PICKUP
|   |-- ORDER DELIVERY
|
|-- RECOMMENDATION SYSTEM
|   |-- PERSONALIZED RECOMMENDATIONS
|   |-- POPULAR DISHES
|   |-- TRENDING RESTAURANTS
|
|-- ADMIN DASHBOARD
|   |-- USER ANALYTICS
|   |-- RESTAURANT MANAGEMENT
|   |-- ORDER MANAGEMENT
|   |-- DRIVER MANAGEMENT
```

❖ High-Level Design for Zomato Online Food Delivery Application

➢ Table of Contents

1. Introduction

The high-level design for the Zomato online food delivery application provides an overview of the system architecture, key components, data flow, and security considerations. This design aims to ensure the application is scalable, maintainable, and secure.

2. System Architecture

The system architecture for the Zomato online food delivery application is based on a microservices model, which ensures modularity, scalability, and ease of maintenance. The architecture includes the following layers:

- **Presentation Layer:** Handles the user interface (UI) for web and mobile applications.
- **Application Layer:** Contains the business logic and microservices.
- **Data Layer:** Manages data storage and retrieval using databases.
- **Integration Layer:** Handles communication with external services, such as payment gateways and notification services.

- **High-Level Architectural Diagram**

```
 -------------------------------------------
|         Presentation Layer               |
|------------------------------------------|
|  Web App (React.js) | Mobile App |
|  (React Native)                          |
 -------------------------------------------
                     |
                     V
 -------------------------------------------
|         Application Layer                |
|------------------------------------------|
|  User Service      |  Order Service     |
|  Restaurant        |  Payment Service |
|  Service           |  Review Service   |
|  Menu Service      |    Notification     |
|------------------------------------------|
                     |
                     V
 ------------------------------------------
|              Data Layer                 |
|--------------------------------=-------|
|  MySQL Database  | Redis Cache |
 ------------------------------------------
                     |
                     V
 -------------------------------------------
|        Integration Layer                 |
|------------------------------------------|
|  Payment Gateway | Email Service  |
|  (Stripe, PayPal) | SMS Gateway     |
 -------------------------------------------
```

3. Component Design

Each microservice in the application layer is designed to handle specific business functions. Below are the key components and their responsibilities:

- **User Service:**
  - User registration and authentication
  - Profile management
  - Order history retrieval
- **Restaurant Service:**
  - Restaurant registration and management
  - Fetching restaurant details
- **Menu Service:**
  - Adding and updating menus
  - Retrieving menu items
- **Order Service:**
  - Order placement and tracking
  - Managing order status
- **Payment Service:**
  - Processing payments
  - Handling payment confirmations and failures
- **Review Service:**
  - Adding, viewing, and updating reviews
- **Notification Service:**
  - Sending email and SMS notifications
  - Push notifications

4. Data Flow Design

Data flows between various components to ensure seamless operation of the application. Here is a high-level view of the data flow:

Data Flow Diagram

[User] --(Register/Login)--> [User Service] --(Authenticate)--> [MySQL Database]
[User] --(View Restaurants)--> [Restaurant Service] --(Fetch Data)--> [MySQL Database]
[User] --(Place Order)--> [Order Service] --(Save Order)--> [MySQL Database]
[Order Service] --(Notify)--> [Notification Service] --(Send Email/SMS)--> [External Services]
[Order Service] --(Process Payment)--> [Payment Service] --(Validate)--> [Payment Gateway]
[User] --(Add Review)--> [Review Service] --(Save Review)--> [MySQL Database]

5. User Interface Design

The user interface (UI) is designed to be intuitive and user-friendly, providing seamless navigation and interaction for users. Key UI components include:

- **Homepage:**
  - Search bar for finding restaurants and dishes
  - Featured restaurants and deals
  - User login and registration options

- **Restaurant Page:**
  - Restaurant details (name, address, contact)
  - Menu items with descriptions and prices
  - Reviews and ratings

- **Order Page:**
  - Order summary with selected dishes
  - Delivery address and payment options
  - Order confirmation and tracking

- **Profile Page:**
  - User details and edit options
  - Order history and past reviews

6. Security Design

Security is paramount in the design of the Zomato online food delivery application. Key security measures include:

- **Authentication and Authorization:**
  - JWT (JSON Web Tokens) for secure user sessions
  - Role-based access control (RBAC) to restrict access to sensitive operations

- **Data Encryption:**
  - HTTPS for secure data transmission between the client and server
  - Encryption of sensitive data (e.g., passwords, payment details) in the database

- **Input Validation:**
  - Server-side validation of all input data to prevent SQL injection and XSS attacks
  - Sanitization of user inputs

- **Monitoring and Logging:**
  - Real-time monitoring of application activities
  - Logging of important events for audit and debugging purposes

- **Backup and Recovery:**
  - Regular database backups to prevent data loss
  - Disaster recovery plans to handle unexpected failures

❖ Detailed design-Input and Output

1. User Management System

- **Inputs:**
  - **User Registration:**
    - User details (name, email, phone number, address)
    - User credentials (username, password)
    - Payment information (credit/debit card details, digital wallets)
  - **User Login:**
    - Username and password or biometric authentication (fingerprint, face ID)
    - Social media login credentials (Google, Facebook)
- **Outputs:**
  - **User Account:**

- Confirmation emails/SMS for registration and login
- User profile page with account details
- Authentication tokens for session management
- Payment confirmation and transaction history

2. Restaurant Management System

- **Inputs:**
  - **Restaurant Registration:**
    - Restaurant details (name, address, cuisine type)
    - Owner/manager contact information
    - Menu items with descriptions, prices, and images
  - **Menu Updates:**
    - New menu items, modifications to existing items
    - Availability status (e.g., sold out, special offers)
- **Outputs:**
  - **Restaurant Profile:**
    - Restaurant listing on the platform
    - Menu displayed on user interface
    - Order notifications and summaries to restaurant dashboard
    - Analytics and performance reports for restaurant owners

3. Order Management System

- **Inputs:**
  - **Order Placement:**
    - User selections from the menu
    - Special instructions (e.g., no onions, extra spicy)
    - Payment method and confirmation
  - **Order Updates:**
    - Status changes (e.g., order received, preparing, out for delivery)
- **Outputs:**
  - **Order Confirmation:**
    - Order summary and receipt sent to user
    - Order details forwarded to the restaurant
    - Real-time order tracking information to user
    - Notifications to delivery personnel with delivery instructions

4. Delivery Management System

- **Inputs:**
  - **Delivery Assignment:**
    - Order details (pickup location, delivery address, expected delivery time)
    - Delivery personnel availability and location
    - Real-Time Tracking:
    - GPS data from delivery personnel's device
    - Traffic and route information
- **Outputs:**
  - **Delivery Instructions:**
    - Optimized delivery route to delivery personnel
    - Delivery status updates to the user (e.g., on the way, delivered)
    - Estimated time of arrival updates

5. Payment Processing System

- **Inputs:**
  - **Payment Information:**
    - User's payment method details
    - Payment amount and transaction details
    - Payment Confirmation:
    - Confirmation from payment gateway or bank
- **Outputs:**
  - **Transaction Records:**
    - Payment receipts sent to user and restaurant
    - Transaction history updated in user's account
    - Financial reports for Zomato's accounting system

6. Customer Support System

- **Inputs:**
    - **Support Requests:**
        - User queries and complaints via email, chat, or phone
        - Feedback and ratings for orders and restaurants
- **Outputs:**
    - **Support Responses:**
        - Automated acknowledgments and ticket numbers
        - Resolutions and follow-ups with users
        - Internal reports for quality assurance and improvement

7. Data Analytics and Reporting System

- **Inputs:**
    - **Usage Data:**
        - User activity logs (searches, clicks, orders)
        - Restaurant performance data (order volumes, ratings)
        - Delivery efficiency metrics (time, success rates)
- **Outputs:**
    - **Analytical Reports:**
        - User behavior insights and trends
        - Sales and revenue reports for Zomato and partner restaurants
        - Performance dashboards for operational monitoring

❖ Logic/Algorithm Design

The Logic/Algorithm Design of the Zomato Online Food Delivery App encompasses the systematic organization of algorithms and logical processes that drive the various functionalities of the application. This report provides an overview of the key components, algorithms, and design considerations involved in creating a robust and efficient system.

- ### Components of Logic/Algorithm Design:

1. **User Authentication and Profile Management:**
   - Algorithms for user registration, login, and profile management ensure secure access to the application and personalized user experiences.
2. **Restaurant and Menu Management:**
   - Algorithms facilitate the listing of nearby restaurants, fetching restaurant menus, and managing menu items' availability and details.
3. **Order Processing:**
   - Order placement algorithms handle the creation, validation, and processing of orders, including calculating total costs, initiating payment processing, and updating order statuses.
4. **Delivery Management:**
   - Algorithms for assigning delivery personnel, optimizing delivery routes, and tracking order delivery ensure timely and efficient delivery of food orders to customers.
5. **Notification System:**
   - Algorithms manage the sending of notifications to users at various stages of the order lifecycle, providing updates on order confirmation, status changes, and delivery notifications.
6. **Review and Rating System:**
   - Algorithms handle the submission, storage, and retrieval of user reviews and ratings for restaurants and delivery experiences, contributing to the overall user feedback mechanism.

- ## Design Considerations:

1. **Scalability:**

The design ensures scalability to accommodate a growing user base and increased demand for food delivery services.

2. **Security:**

Robust authentication mechanisms and data encryption techniques are implemented to safeguard user data and transactions.

3. **Efficiency:**

Algorithms are optimized for performance to minimize response times and ensure smooth user experiences, even during peak hours.

4. **Reliability:**

Error handling and fault tolerance mechanisms are incorporated to handle exceptions and ensure the application's reliability and availability.

5. **User Experience:**

The design prioritizes user-centric features and intuitive interfaces to enhance the overall user experience and encourage user engagement.

❖ Verify statement coverage

- **Code Instrumentation:** Use a code coverage tool or profiler to instrument the code. This tool tracks which statements are executed during testing.
- **Test Suite Creation:** Develop a comprehensive test suite covering various scenarios and functionalities of the Zomato system. This suite should include positive and negative test cases, boundary cases, and edge cases.
- **Test Execution:** Run the test suite against the Zomato system. Ensure that all test cases are executed, covering different parts of the system.
- **Coverage Analysis:** After test execution, analyze the coverage report generated by the code coverage tool. The report typically shows which statements were covered (executed) and which were not.
- **Review Uncovered Statements:** Review the uncovered statements and identify why they were not executed during testing. It could be due to missing test cases, conditional branches that were not triggered, or unreachable code.
- **Adjust Test Suite:** Modify the test suite to include additional test cases or improve existing ones to cover the remaining statements.
- **Repeat Test Execution:** Run the updated test suite again and analyze the coverage report. Ensure that the changes in the test suite result in improved statement coverage.
- **Verification:** Once all statements have been covered, verify that the system behaves as expected under various conditions and scenarios.
- **Documentation:** Document the statement coverage achieved and any areas where coverage could not be attained, along with the reasons.
- **Continuous Monitoring:** As the codebase evolves, regularly monitor statement coverage to ensure that new changes are adequately tested.

❖ Verify sequence of control flow for zomato

- **User Registration/Login Flow:**

User opens the Zomato app.
If the user is not logged in, they are directed to the login or registration screen.
User enters their credentials or registers as a new user.
Upon successful authentication, the user is logged in and directed to the home screen.

- **Restaurant and Menu Browsing Flow:**
    - On the home screen, the user can browse nearby restaurants or search for specific ones.
    - User selects a restaurant to view its menu.
    - The app fetches and displays the menu items for the selected restaurant.
    - User selects items from the menu to add to their order.

- **Order Placement Flow:**
    - After selecting items, the user proceeds to place the order.
    - The app calculates the total cost including taxes and delivery charges.
    - User provides delivery address and any special instructions.
    - The order is submitted, and the app initiates payment processing.
- **Payment Processing Flow:**
    - The app communicates with the payment gateway to process the payment.
    - Upon successful payment authorization, the order status is updated to 'Paid'.
    - If payment fails, appropriate error handling is performed, and the user is notified.

- **Order Fulfillment Flow:**
  - The restaurant receives the order and prepares the food.
  - Once ready, a delivery person is assigned to pick up the order.
  - The delivery person navigates to the restaurant to collect the order.
  - The delivery person delivers the order to the user's specified address.
- **Order Tracking Flow:**
  - Throughout the process, the user can track the status of their order in real-time.
  - Notifications are sent to the user at key stages (order confirmed, out for delivery, delivered).
- **Review and Rating Flow:**
  - After order delivery, the user may provide a review and rating for the restaurant and delivery experience.
  - The app prompts the user to rate and review the order.
  - User submits their feedback, which is stored and used to calculate restaurant ratings.
- **Profile Management Flow:**
  - At any time, the user can access and update their profile information.
  - This includes personal details, delivery addresses, and payment methods.
- **Logout Flow:**
  - When the user chooses to log out, their session is terminated, and they are redirected to the login screen.

❖ Verify Decision coverage

- **Identify Decision Points:** Review the codebase to identify all decision points. These are typically conditional statements such as if-else, switch-case, and ternary operators.
- **Create Test Cases:** Develop test cases that cover each decision point with both true and false conditions. Ensure that the test cases cover all possible branches of the decision points.
- **Execute Test Cases:** Run the test suite against the Zomato application. Ensure that all test cases are executed, covering different paths through the codebase.
- **Check Decision Coverage:** Analyze the coverage report generated by the testing framework. The report should indicate which decision points have been evaluated with both true and false conditions.
- **Review Uncovered Decision Points:** Review the coverage report to identify any decision points that were not evaluated with both true and false conditions. These points indicate gaps in decision coverage.
- **Adjust Test Suite:** Modify the test suite to include additional test cases or improve existing ones to cover the uncovered decision points. Ensure that all decision points are evaluated with both true and false conditions.
- **Repeat Test Execution:** Run the updated test suite again and analyze the coverage report to verify that decision coverage has been improved.
- **Verification:** Once decision coverage is achieved, verify that the application behaves as expected under various conditions and scenarios.
- **Documentation:** Document the decision coverage achieved and any decision points that could not be covered, along with the reasons.

❖ List the Programs required for application –for each program write input process and output. Write Algorithm of program.

- Registration
- Login
- Changes to cart
- Payment
- Logout
- Report Generation

- **Registration**– If customer wants to buy the product then he/she must be registered, unregistered user can't go to the shopping cart.
- **Login**-Customer logins to the system by entering valid user id and password for the shopping.
- **Changes to Cart**-Changes to cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.
- **Payment**-For customer there are many types of secure billing will be prepaid as debit or credit card, postpaid as after shipping, check or bank draft. The security will provide by the third party like Pay-Pal etc.
- **Logout**-After the payment or surf the product the customer will logged out.
- **Report Generation**–After all transaction the system can generate the portable document file (.pdf) and then sent one copy to the customer's Email-address and another one for the system data base to calculate the monthly transaction.

Algorithm of Program:

The simulation first starts with the customer entering his/her credentials (name, ID and password). Once that has been verified, the customer can place an order specifying the quantity of the food required. Now we get a window that displays the order number, customer ID, food name, price and quantity. Once the customer finalizes his/her order, they are redirected to the payment window where the total price is displayed and the customer can select the payment method of their choice and then the customer gets a message of confirmation of order. The above mentioned simulation flow is with respect to the customer's point of view. Now if you are an admin, you can select the normal login option and enter the admin credentials (email ID and password). Once you enter the admin portal, you get the option of adding food, deleting food or updating food.

Method for unit testing

Functional Testing of Food Delivery Applications:

Functional testing ensures that all features and functionalities of the food delivery application work as intended. Testers check things like menu navigation, ordering process, payment processing, account management, and notifications to ensure they function correctly.

Performance Testing of Food Delivery App:

Performance testing assesses the speed, responsiveness, and scalability of the food delivery application. Testers simulate different levels of user traffic and orders to evaluate how the app performs under various conditions, ensuring it can handle peak loads without crashing or slowing down.

Usability Testing of Food Delivery Applications:

Usability testing focuses on the user experience of the food delivery application. Testers assess factors like ease of use, intuitive design, clear navigation, and accessibility to ensure that users can easily find restaurants, browse menus, place orders, and track deliveries without confusion or frustration.

Compatibility Testing of Food Delivery Applications:

Compatibility testing checks how well the food delivery application performs on different devices, operating systems, and web browsers. Testers ensure that the app is compatible with a wide range of devices and platforms to reach a broader audience and provide a consistent user experience across various devices.

Security Testing for Food Delivery Applications:

Security testing evaluates the security measures implemented in the food delivery application to protect user data, payment information, and sensitive information from unauthorized access or cyber-attacks. Testers identify potential vulnerabilities and ensure that proper security protocols are in place to safeguard user privacy and prevent data breaches.

❖ Test Cases for unit testing:

We will investigate the 6 main components of any food delivery app. The below can be used as **test cases for Zomato**

1. **Search Functionality**

- Validation that the restaurant name is searchable in the search text box.
- Validation that the cuisine name is searchable in the search text box.
- Validation that the dish name is searchable in the search text box.
- Validation that matching suggestions should be shown when search items do not match any of the relevant records.
- Validation that the appropriate search results should be shown when an item is searched.

2. **Home Page**

- Validation that billing discounts should be displayed on the home page.
- Validation that past orders should be shown on the home page for quick delivery.
- Validation that restaurants should be shown on the home page as per distance from the delivery location.
- Validation that the right delivery location should be shown.
- Validation that the filter and sort options should be provided on the home page. Filters options such as Cuisines and rating should be shown. Sorting of restaurants should be done as per sort criteria such as rating, delivery time, cost
- Validation that history and donate options should be shown on the home page.

3.Ordering Page

- Verify restaurant name and rating along with photos should be displayed clearly.
- Verify cuisines served by restaurants should be shown under restaurant names such as north Indian, Chinese, and others.
- Verify reviews should be clearly shown so that people can order based on those.
- Verify reviews can be filtered based on various filters such as detailed reviews, latest, delivery reviews, and my reviews.
- Mode of delivery, approximate time of delivery, and offers should be shown clearly.
- Verification that the veg and non-veg toggle options should be present.
- Verify add option should add the item to the cart.
- Verify all items should be listed clearly along with their prices.
- Verify the best offer of the restaurant should be shown in the lower section of the page.
- Verify users can customize the order for the customizable items.
- Verify green and red dots are marked against items as per veg and non-veg.
- Verify rating of food items should be displayed clearly.
- Verify recommended section should be displayed.
- Verify menu hamburger option should be shown and upon clicking, should open the menu based on umbrella categories for easy ordering.
- Verify images should be correctly shown against the item.

4. **Cart Page**

- Verification that delivery location should be shown on the cart page.
- Verify right items are displayed on the cart page.
- Verify that option is available to increment or decrement the number of items.
- Verify the option to delete the item is available.
- Verify offers section should be displayed.
- Verify offer should be applied successfully when the user applies one offer.
- Verify option to tip your valet should be available.
- Verify the invoice should be generated correctly.
- Verify there is an option to ad voice directions.
- Verify delivery location can be changed.
- Verify when the delivery location is changed and if the delivery location is not serviceable then an error message should be shown.
- Verify your details should be shown.
- Verify there should be an option to order for someone else.
- Verify pay using option should be shown.
- Verify upon clicking of pay now, different payments options should open.
- Verify different payment options should work correctly.
- Verify cart page can be minimized.
- Verify cart should retain the products even if the app is closed.
- Verify upon adding another item from the different restaurant the earlier item should be removed automatically.

5. **Account Section**

- Verify different account options should be shown for the user,
- Verify past orders, favorite orders, and address book can be accessed from the account section.
- Verify help option should be shown In case any help is required.
- Verify about, send feedback, log out options should be displayed correctly.

6. **Registration and Login**

- Verify users can log in to a Zomato account using a mobile number.
- Verify users can log in via different options such as email and password, Facebook, or Google sign-in.
- Verify users can create an account username and email.
- Verify users can choose delivery locations upon successful registration.
- Verify authentication is performed correctly when OTP is sent to the customer.

❖ Performance Testing of Food Delivery App:

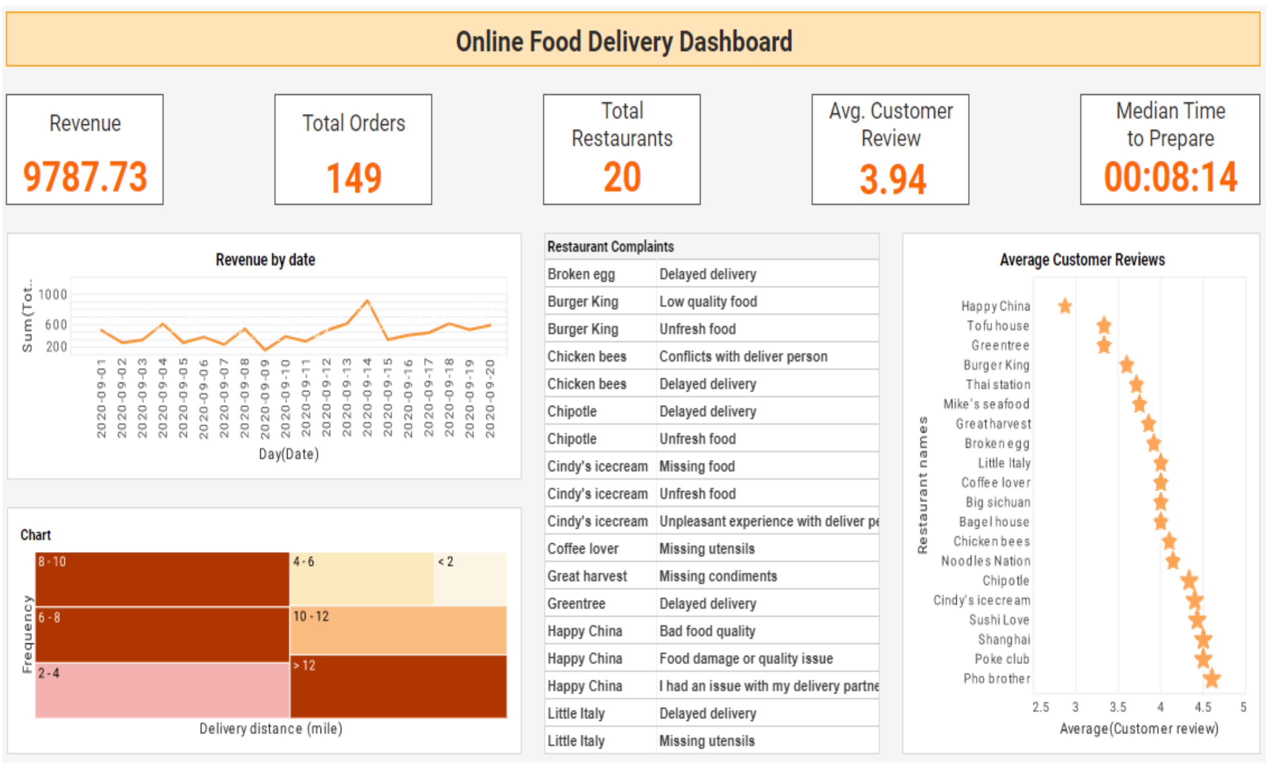
Performance testing is a very crucial part of testing food delivery applications. We need to evaluate the behaviour and stability of food delivery applications when high stress, load, concurrency, volume is applied.

- Different performance testing use cases are
- validation of application behaviour under high load
- validation of application behaviour under peak usage
- validation under high constant load and identification of various bottlenecks in the application

❖ Analysis Tool:

In the ever growing competitive industry of online food ordering delivery, continuous improvement to achieve higher customer satisfaction is the main agenda for all organizations involved. Today, when customers look to order food online through phone apps, their decision making is eased by the multitude of food service options available to them, catered to their taste and personality.

Much of this ease can be attributed to the data analytics performed by the online food delivery applications to gather necessary insights that can bridge the gap between their current offerings and customer demands.



Online Food Delivery Dashboard

Revenue	Total Orders	Total Restaurants	Avg. Customer Review	Median Time to Prepare
9787.73	149	20	3.94	00:08:14

Restaurant Complaints

Broken egg	Delayed delivery
Burger King	Low quality food
Burger King	Unfresh food
Chicken bees	Conflicts with deliver person
Chicken bees	Delayed delivery
Chipotle	Delayed delivery
Chipotle	Unfresh food
Cindy's icecream	Missing food
Cindy's icecream	Unfresh food
Cindy's icecream	Unpleasant experience with deliver pe
Coffee lover	Missing utensils
Great harvest	Missing condiments
Greentree	Delayed delivery
Happy China	Bad food quality
Happy China	Food damage or quality issue
Happy China	I had an issue with my delivery partne
Little Italy	Delayed delivery
Little Italy	Missing utensils

❖ Code review technique for Application:

a. Understand the Requirements: Before diving into the code, make sure you have a clear understanding of the feature or functionality being implemented. Refer to the user stories, design documents, and any other relevant resources.

b. Code Review Checklist: Develop a checklist of items to review based on best practices, coding standards, and potential issues specific to the Zomato app. This might include aspects like security, performance, scalability, maintainability, and adherence to architectural patterns.

c. Review Process: Establish a process for the code review. This could involve using a code review tool like GitHub or GitLab, where reviewers can leave comments directly on the code. Determine who will be involved in the review process and set expectations for timelines and feedback.

d. Start with High-level Overview: Begin by reviewing the overall architecture and design of the code. Ensure that it aligns with the broader architecture of the Zomato app and follows established patterns and practices.

e. Functional Review: Test the functionality of the code to ensure that it meets the requirements specified in the user stories. Look for edge cases, error handling, and any potential bugs.

f. Code Quality: Evaluate the quality of the code in terms of readability, maintainability, and adherence to coding standards. Pay attention to variable names, comments, code structure, and formatting.

g. Performance and Scalability: Assess the performance implications of the code, especially if it involves operations that could impact the app's responsiveness or scalability. Look for potential bottlenecks or inefficient algorithms.

h. Security Review: Check for security vulnerabilities such as input validation issues, authentication weaknesses, or data exposure risks. Ensure that sensitive data is handled securely and that the code is resistant to common security threats.

i. Testing and Test Coverage: Verify that the code is adequately tested and that the test coverage is sufficient. Look for unit tests, integration tests, and any other relevant test cases.

j. Documentation: Ensure that the code is well-documented, both inline and in any accompanying documentation. This includes documenting the purpose of classes and methods, as well as any assumptions or constraints.

k. Feedback and Collaboration: Provide constructive feedback to the developer based on your review. Focus on specific issues and suggest improvements or alternative approaches where necessary. Encourage collaboration and open communication throughout the review process.

l. Follow-up: Follow up on any issues raised during the review to ensure that they are addressed appropriately. Monitor the progress of the code changes and provide additional guidance or support as needed.

❖ Test Cases for Integration Testing

Integration testing for the delivery functionality in the Zomato app involves testing how different modules or components work together to ensure smooth delivery operations. Here are some test cases you might consider:

1.User Authentication Integration:

- Test that the delivery functionality integrates seamlessly with the user authentication system.
- Verify that delivery personnel are authenticated properly before accessing delivery-related features.

2.Order Management Integration:

- Test the integration between the delivery module and the order management system.
- Verify that delivery personnel can view assigned orders, update order status (e.g., picked up, en route, delivered), and handle order exceptions (e.g., cancellations, address changes).

3.Location Services Integration:

- Test integration with location services to ensure accurate tracking of delivery personnel.
- Verify that delivery personnel can access accurate customer addresses, navigate to delivery locations, and provide real-time location updates.

4.Payment Integration:

- Test the integration between delivery and payment systems.
- Verify that delivery personnel can handle cash payments, process refunds if necessary, and update order status based on payment completion.

5.Notification Integration:

- Test integration with notification services to ensure timely communication with customers and delivery personnel.
- Verify that customers receive notifications for order confirmation, order status updates, and delivery tracking.
- Verify that delivery personnel receive notifications for new orders, order updates, and customer messages.

6.Inventory Management Integration:

- Test integration with inventory management systems (if applicable).
- Verify that delivery personnel can view item availability, handle out-of-stock items, and update inventory status during delivery.

7.Customer Support Integration:

- Test integration with customer support systems.
- Verify that delivery personnel can escalate delivery-related issues to customer support, communicate with customers regarding delivery concerns, and access relevant customer information.

8.Performance and Load Testing:

- Test the performance and scalability of the delivery functionality under various load conditions.
- Verify that the system can handle multiple concurrent delivery requests, update order status in real-time, and provide accurate delivery estimates.

9.Error Handling and Recovery:

- Test how the system handles errors and failures gracefully.
- Verify that appropriate error messages are displayed to delivery personnel and customers in case of delivery failures, network issues, or system errors.
- Test recovery mechanisms for failed deliveries, such as reassigning orders, rescheduling deliveries, or providing compensation to customers.

10.Integration with Third-party Services:

- Test integration with third-party services, such as mapping APIs for navigation, SMS gateways for notifications, or payment gateways for transactions.
- Verify that the Zomato app interacts correctly with these external services and handles any errors or inconsistencies.

These test cases cover various aspects of integration testing for the delivery functionality in the Zomato app, ensuring that all components work seamlessly together to provide a smooth and reliable delivery experience for both customers and delivery personnel.

❖ Product Metrics-Size and complexity

1.Lines of Code (LoC):

- Measure the total number of lines of code across the Zomato application.
- Break down LoC by programming language (e.g., Java, Kotlin for Android; Swift for iOS; JavaScript, Python for backend).

2.Function Points:

- Calculate function points to quantify the functional size of the application.
- Assess the number of inputs, outputs, inquiries, internal files, and external interfaces to determine function points.

3. Cyclomatic Complexity:

- Measure the complexity of the codebase using cyclomatic complexity metrics.
- Identify complex functions or modules that may require refactoring or additional testing.

4.Component Dependencies:

- Analyze the dependencies between different components/modules within the Zomato application.
- Measure the coupling between components to assess the potential impact of changes or updates.

5.API Endpoints:

- Count the total number of API endpoints exposed by the Zomato backend.
- Track the usage and performance of each endpoint to identify bottlenecks or areas for optimization.

6.Database Schema Complexity:

- Assess the complexity of the database schema used by the Zomato application.
- Measure the number of tables, indexes, relationships, and constraints within the database.

7.Third-party Integrations:

- Identify the number of third-party services or APIs integrated into the Zomato application.
- Track usage metrics and dependencies on external services to ensure reliability and performance.

8.User Interface Elements:

- Count the number of user interface elements (e.g., screens, buttons, inputs) across the Zomato app.
- Assess the complexity of each UI element in terms of interactions and visual design.

9.Feature Count:

- Enumerate the total number of features or functionalities offered by the Zomato application.
- Track the adoption and usage of each feature to prioritize development efforts.

10.Code Churn:

- Measure the rate of change in the codebase over time.
- Analyze code churn metrics to identify areas of frequent updates or instability**.**

❖ Founction Count:

To estimate the number of functions in the Zomato application, we'll consider various components including frontend, backend, and any other auxiliary services. The Zomato application likely contains a wide range of functions to handle user interactions, data processing, business logic, and more. Here's a breakdown of potential functions in each component:

1. Frontend (Web/Mobile App):

- **Authentication functions:** Login, Logout, Sign Up, Forgot Password, Social Login
- **Search functions:** Search Restaurants, Search by Cuisine, Search by Location
- **Restaurant functions**: View Restaurant Details, View Menu, Add to Favorites, Write Reviews, Rate Restaurants
- **Order functions:** Place Order, Track Order, View Order History, Cancel Order
- **User Profile functions:** View Profile, Edit Profile, Change Password, Delete Account

- **Location functions:** Detect User Location, Set Delivery Address, View Nearby Restaurants
- **Notification functions:** Receive Order Confirmation, Delivery Updates, Promotional Notifications
- **Payment functions:** Choose Payment Method, Make Payment, View Payment History

2. Backend (Server-side):

- **User Management functions:** Create User, Authenticate User, Update User Profile, Delete User
- **Restaurant Management functions:** Add Restaurant, Update Menu, Manage Reviews, Handle Orders
- **Order Management functions:** Process Orders, Assign Delivery Personnel, Calculate Delivery Time
- **Notification functions:** Send Push Notifications, Email Notifications, SMS Notifications
- **Payment Processing functions:** Handle Payment Gateway Integration, Process Payments
- **Location Services functions:** Geocoding, Reverse Geocoding, Distance Calculation
- **Recommendation functions:** Generate Restaurant Recommendations, Personalized Suggestions
- **Analytics functions:** Track User Behavior, Generate Reports, Monitor Performance Metrics

3. Auxiliary Services:

- **Content Management functions:** Manage Static Content, Images, and Media
- **Security functions:** Encrypt Data, Validate Inputs, Prevent Cross-Site Scripting (XSS), Cross-Site Request Forgery (CSRF)
- **Logging functions**: Log Events, Errors, and Debug Information
- **Caching functions:** Cache Frequently Accessed Data, Improve Performance

These are just examples of potential functions in the Zomato application. The actual number of functions may vary depending on the application's specific features, complexity, and architecture. Additionally, functions may be further broken down into smaller units or combined into larger modules based on design decisions and development practices.

❖Mention availability of application required and why ?

1. **Customer Satisfaction:** Availability ensures that users can access the Zomato app whenever they need to order food or check restaurant listings. A reliable and accessible application contributes to a positive user experience, leading to increased customer satisfaction and loyalty.

2. **Business Continuity:** For Zomato, whose primary business model revolves around facilitating food delivery services, application availability is essential for business continuity. Any downtime or unavailability of the app can directly impact revenue generation and market reputation.

3. **Competitive Advantage:** In the highly competitive online food delivery market, maintaining high availability gives Zomato a competitive edge over rivals. Users are more likely to choose a platform that is consistently accessible and responsive, enhancing Zomato's market position.

4. **Brand Reputation:** Application availability reflects on Zomato's brand reputation and credibility. A reliable app builds trust among users and reinforces Zomato's image as a dependable service provider in the food delivery industry.

5. **Operational Efficiency:** With high availability, Zomato can efficiently process orders, manage restaurant listings, and handle customer inquiries. This contributes to operational efficiency and ensures smooth business operations.

6. **Customer Retention:** Users are less likely to switch to competing platforms if they can rely on Zomato's app for their food delivery needs. High availability reduces the risk of user churn and helps in retaining customers over the long term.

7. **Regulatory Compliance:** In some regions, regulatory authorities may require online service providers like Zomato to maintain a certain level of availability to ensure consumer rights and protection. Compliance with such regulations is essential for legal and ethical reasons.

❖ Portability

- **Data Portability:**

  - Some users might be interested in porting their data from Zomato to another platform or vice versa. Currently, Zomato doesn't offer a built-in feature for data portability, but you can manually export some data like your reviews, favorites, or photos, though it might not be as straightforward.

- **Restaurant Listings:**
  - If you're referring to the portability of restaurant listings or information on Zomato, it's not entirely clear what you mean.
  - Zomato does provide a platform where restaurants can list their information, menus, and services, but this data isn't typically portable in the traditional sense. However, users can search for restaurants in different locations, so in that sense, the restaurant listings are portable across different areas.

  - Zomato is available as a mobile application on both Android and iOS platforms, making it portable across various smartphones and tablets.

  - Users can also access Zomato through a web browser on their desktop or laptop computers, providing flexibility in how they interact with the platform.

- **Cross-Platform Compatibility:**

  - The app might be available on multiple platforms like iOS, Android, and web browsers, allowing users to access it from various devices such as smartphones, tablets, and computers.

- **User Data Portability:**

  - Users may be able to carry their account information, order history, preferences, and saved addresses across different devices. This ensures a seamless experience regardless of the device they're using.

- **Payment Portability:**
  - Zomato may offer various payment methods and allow users to link multiple payment sources (credit/debit cards, digital wallets, etc.) to their account. This means users can pay for orders using their preferred payment method regardless of the device they're using.
- **Menu Porbablity:**
  - Users might be able to browse menus and place orders from any device with the Zomato app installed or through their web browser. This allows users to access the same set of restaurants and menu items regardless of the device they're using.

These are just a few examples of how portability might manifest in the Zomato app. The specific features and functionalities may vary based on updates and changes made by Zomato over time. If you're looking for more specific information about portability features in the Zomato app, I'd recommend checking the latest version of the app or their official website for the most accurate details.