**ZOMATO SYSTEM DESIGN**

* 1. **Introduction:**

The Zomato application has revolutionized the way users discover, order, and enjoy food from a wide range of restaurants. With its intuitive interface and extensive database of eateries, Zomato has become a go-to platform for millions of users worldwide. This document aims to provide a detailed low-level system design for the Zomato application, focusing specifically on the food and delivery modules.

Throughout this document, we will explore the architecture, database design, component interactions, and implementation details of the food and delivery modules. This information will serve as a valuable resource for guiding the development, maintenance, and enhancement of these critical components within the Zomato ecosystem.

* 1. **Purpose:**
* Offer developers, architects, and other stakeholders a comprehensive understanding of the system's design and structure.
* Provide guidance for the development, maintenance, and enhancement of the food and delivery modules within the Zomato application.
* Ensure alignment with the overall objectives and requirements of the Zomato application by detailing how the food and delivery modules contribute to its functionality.
* Serve as a reference document for future development efforts, allowing for efficient troubleshooting, optimization, and scalability planning.
* Facilitate effective collaboration and communication among project team members by offering a common reference point for discussing design decisions and implementation strategies related to the food and delivery modules.
  1. **Scope:**

This document focuses on providing a detailed low-level system design for the food and delivery modules of the Zomato application. Specifically, it covers the following aspects:

Architecture and Component Design:

Detailed explanation of the architecture, components, and interactions within the food and delivery modules. This includes the design of microservices, APIs, and integration points for handling user interactions and processing orders.

Database Design:

Schema design for the databases used to store data related to users, restaurants, food items, carts, orders, payments, and notifications within the food and delivery modules.

User Authentication and Authorization:

Design considerations for user authentication mechanisms, access control policies, and security features related to user accounts and authentication processes.

Food Module Functionality:

Design details for functionalities such as restaurant search, listing food items, managing carts, placing orders, and viewing order history within the Zomato application.

Delivery Module Functionality:

Design considerations for managing delivery metadata, tracking order status, optimizing delivery routes, and coordinating interactions between users, restaurants, and delivery partners.

Payment Integration:

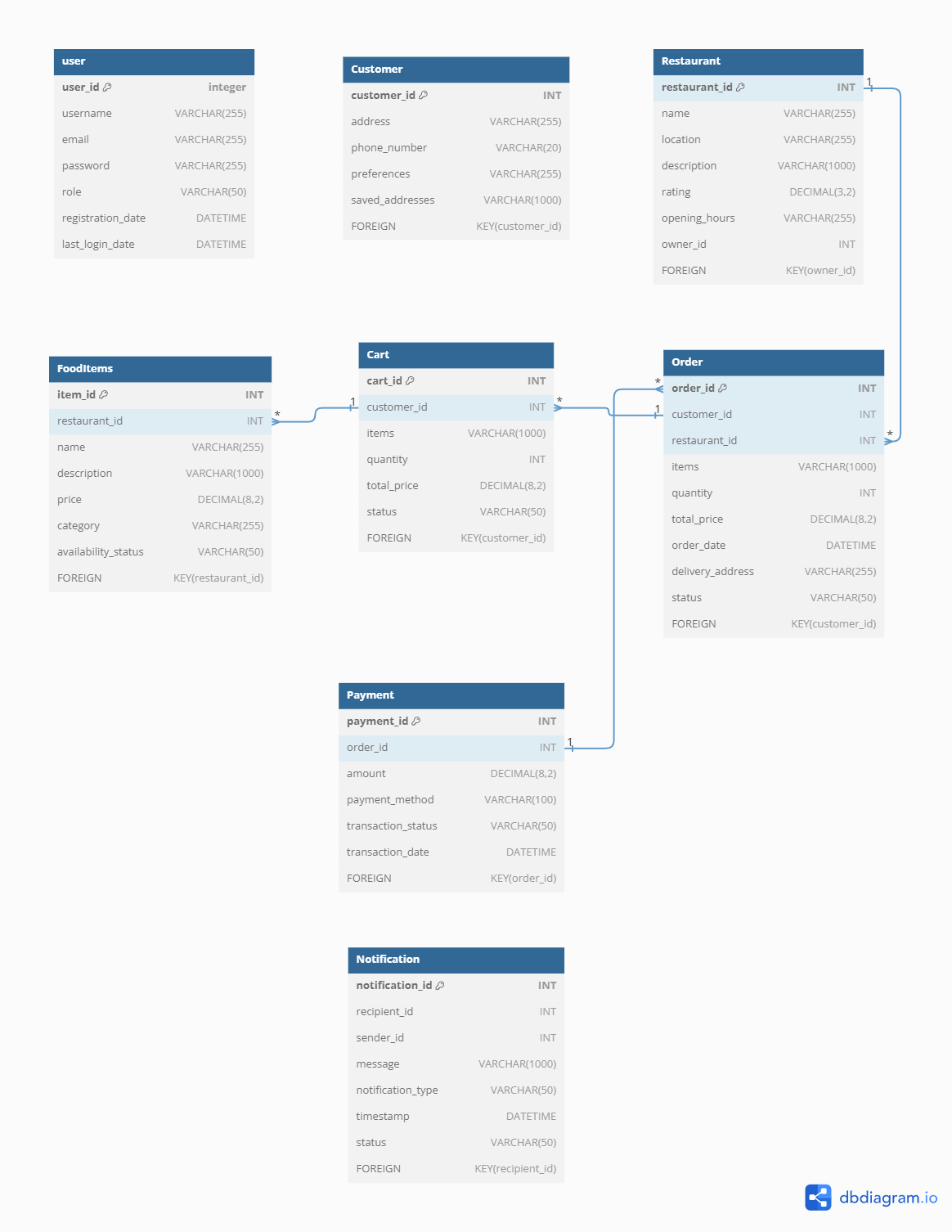
Design of payment processing functionalities, including integration with payment gateways, handling payment authorization, capture, and settlement, and ensuring compliance with security and regulatory standards.

Notification System:

Design of a notification system for sending updates to users regarding order status, payment confirmations, promotions, and other relevant information within the food and delivery modules.

**2.1 Database Design:**

For the food and delivery modules of the Zomato application, the database design plays a crucial role in efficiently storing and managing data related to users, restaurants, food items, orders, payments, and notifications.



Here's an overview of the database schema for each entity:

**User:**

* Attributes:

user\_id (primary key), username, email, password, role (customer/restaurant owner), registration\_date, last\_login\_date, etc.

**Customer:**

* Inherits from User entity.
* Additional attributes:

address, phone\_number, preferences, saved\_addresses, etc.

**Restaurant:**

* Attributes:

restaurant\_id (primary key), name, location, description, rating, opening\_hours, owner\_id (foreign key to User), etc.

**Food Items:**

* Attributes:

item\_id (primary key), restaurant\_id (foreign key to Restaurant), name, description, price, category, availability\_status, etc.

**Cart:**

* Attributes:

cart\_id (primary key), customer\_id (foreign key to Customer), items (list of item\_ids), quantity, total\_price, status (active/completed), etc.

**Order:**

* Attributes:

order\_id (primary key), customer\_id (foreign key to Customer), restaurant\_id (foreign key to Restaurant), items (list of item\_ids), quantity, total\_price, order\_date, delivery\_address, status (pending/confirmed/delivered), etc.

**Payment:**

* Attributes:

payment\_id (primary key), order\_id (foreign key to Order), amount, payment\_method, transaction\_status, transaction\_date, etc.

**Notification:**

* Attributes:

notification\_id (primary key), recipient\_id (foreign key to User), sender\_id (foreign key to User), message, notification\_type, timestamp, status (read/unread), etc.

**3.1 System Design:**



A black and white background with text

Description automatically generated

A black background with white squares

Description automatically generated