Assignment 1

Prasad Hadbe

Prateek Majumder

Shikha Thakur

The system intends to use AWS RDS to maintain and analyze a large-scale database of over 100,000 users, with an emphasis on improving customer experience and operational efficiency. The project collects full user, order, and feedback data using a sophisticated relational database structure, allowing for precise analytics on user behavior, order patterns, and service performance.

Assumptions:

1. The users will be modelled after real life demographics of people in Mumbai and Navi Mumbai.
2. All types of restaurants are present.
3. The order values will follow normal distribution of data.
4. Users have unique identifiers, such as email addresses or usernames.
5. Each order corresponds to a certain user and restaurant.
6. Users may write reviews and give ratings for restaurants and food.

Data Modelling:

1. Users Table

- **user_id**: Primary key, auto-incremented integer.
- **name**: User's full name, string, not null.
- **email**: User's email address, string, unique, not null.
- **password**: User's password, string, not null.
- **phone**: User's phone number, string, not null.
- **created_at**: Timestamp of when the user was created, defaults to the current timestamp.
- **updated_at**: Timestamp of when the user was last updated, defaults to the current timestamp and updates automatically.

2. Restaurants Table

- **restaurant_id**: Primary key, auto-incremented integer.
- **name**: Restaurant's name, string, not null.
- **address**: Restaurant's address, text, not null.
- **phone**: Restaurant's phone number, string, unique, not null.
- **created_at**: Timestamp of when the restaurant was created, defaults to the current timestamp.
- **updated_at**: Timestamp of when the restaurant was last updated, defaults to the current timestamp and updates automatically.

3. Orders Table

- **order_id**: Primary key, auto-incremented integer.
- **user_id**: Foreign key to Users table, integer, not null.
- **restaurant_id**: Foreign key to Restaurants table, integer, not null.
- **order_total**: Total amount of the order, decimal, not null.
- **delivery_status**: Status of the delivery (Pending, In Progress, Completed, Cancelled), string, defaults to 'Pending'.
- **order_date**: Timestamp of when the order was placed, defaults to the current timestamp.
- **delivery_date**: Timestamp of when the order was delivered.
- **user_feedback**: User's feedback on the order, text.
- **user_id** and **restaurant_id** have indexes for quicker lookup.

4. Drivers Table

- **driver_id**: Primary key, auto-incremented integer.
- **name**: Driver's full name, string, not null.
- **phone**: Driver's phone number, string, unique, not null.
- **location**: Driver's current location, string, not null.
- **email**: Driver's email address, string, unique, not null.
- **created_at**: Timestamp of when the driver was created, defaults to the current timestamp.
- **updated_at**: Timestamp of when the driver was last updated, defaults to the current timestamp and updates automatically.

5. Payment Table

- **payment_id**: Primary key, auto-incremented integer.

- **order_id**: Foreign key to Orders table, integer, not null.
- **payment_method**: Method of payment (Credit Card, Debit Card, Cash, Online), string, not null.
- **amount**: Payment amount, decimal, not null.
- **status**: Payment status (Pending, Completed, Failed), string, defaults to 'Pending'.
- **payment_date**: Timestamp of when the payment was made, defaults to the current timestamp.
- **order_id** has an index for quicker lookup.

6. Rating Table

- **rating_id**: Primary key, auto-incremented integer.
- **user_id**: Foreign key to Users table, integer, not null.
- **restaurant_id**: Foreign key to Restaurants table, integer, not null.
- **rating**: Rating value, integer, between 1 and 5, not null.
- **review**: User's review, text.
- **rating_date**: Timestamp of when the rating was given, defaults to the current timestamp.
- **user_id** and **restaurant_id** have indexes for quicker lookup.

7. Address Table

- **address_id**: Primary key, auto-incremented integer.
- **user_id**: Foreign key to Users table, integer, not null.
- **state**: State of the address, string, not null.
- **city**: City of the address, string, not null.
- **street**: Street of the address, string, not null.
- **pincode**: Pincode of the address, string, not null.
- **created_at**: Timestamp of when the address was created, defaults to the current timestamp.
- **updated_at**: Timestamp of when the address was last updated, defaults to the current timestamp and updates automatically.
- **user_id** has an index for quicker lookup.

8. Menu Table

- **menu_id**: Primary key, auto-incremented integer.
- **restaurant_id**: Foreign key to Restaurants table, integer, not null.
- **item_name**: Name of the menu item, string, not null.

- **price**: Price of the menu item, decimal, not null.
- **created_at**: Timestamp of when the menu item was created, defaults to the current timestamp.
- **updated_at**: Timestamp of when the menu item was last updated, defaults to the current timestamp and updates automatically.
- **restaurant_id** has an index for quicker lookup.

9. DeliveryExperience Table

- **delivery_experience_id**: Primary key, auto-incremented integer.
- **order_id**: Foreign key to Orders table, integer, not null.
- **driver_id**: Foreign key to Drivers table, integer, not null.
- **delivery_time**: Delivery time in minutes, integer, not null.
- **delivery_rating**: Rating of the delivery experience, integer, between 1 and 5, not null.
- **delivery_comments**: Comments on the delivery, text.
- **order_id** and **driver_id** have indexes for quicker lookup.

10. UserInteraction Table

- **interaction_id**: Primary key, auto-incremented integer.
- **user_id**: Foreign key to Users table, integer, not null.
- **interaction_type**: Type of interaction (Support, Complaint, Inquiry, Feedback), string, not null.
- **interaction_detail**: Details of the interaction, text.
- **interaction_date**: Timestamp of when the interaction occurred, defaults to the current timestamp.
- **user_id** has an index for quicker lookup.

Relationships

- **Users and Orders**: A user can place multiple orders.
- **Restaurants and Orders**: An order is associated with one restaurant.
- **Orders and Payment**: Each order has a corresponding payment entry.
- **Users and Rating**: Users can rate multiple restaurants.
- **Restaurants and Rating**: A restaurant can have multiple ratings.
- **Users and Address**: A user can have multiple addresses.
- **Restaurants and Menu**: A restaurant can have multiple menu items.
- **Orders and DeliveryExperience**: Each order has a delivery experience entry.

- **Drivers and DeliveryExperience**: Each delivery experience is associated with a driver.
- **Users and UserInteraction**: A user can have multiple interactions logged.

Schema Design:

Users Table

- **user_id**: INT (Primary Key, Auto Increment)
- **name**: VARCHAR(255) (Not Null)
- **email**: VARCHAR(255) (Unique, Not Null)
- **password**: VARCHAR(255) (Not Null)
- **phone**: VARCHAR(20) (Not Null)
- **created_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **updated_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP, Update on modification)

Restaurants Table

- **restaurant_id**: INT (Primary Key, Auto Increment)
- **name**: VARCHAR(255) (Not Null)
- **address**: TEXT (Not Null)
- **phone**: VARCHAR(20) (Unique, Not Null)
- **created_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **updated_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP, Update on modification)

Orders Table

- **order_id**: INT (Primary Key, Auto Increment)
- **user_id**: INT (Foreign Key to Users, Not Null, Indexed)
- **restaurant_id**: INT (Foreign Key to Restaurants, Not Null, Indexed)
- **order_total**: DECIMAL(10, 2) (Not Null)
- **delivery_status**: ENUM('Pending', 'In Progress', 'Completed', 'Cancelled') (Default: 'Pending', Not Null)
- **order_date**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **delivery_date**: TIMESTAMP
- **user_feedback**: TEXT

Drivers Table

- **driver_id**: INT (Primary Key, Auto Increment)
- **name**: VARCHAR(255) (Not Null)
- **phone**: VARCHAR(20) (Unique, Not Null)
- **location**: VARCHAR(255) (Not Null)
- **email**: VARCHAR(255) (Unique, Not Null)
- **created_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **updated_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP, Update on modification)

Payment Table

- **payment_id**: INT (Primary Key, Auto Increment)
- **order_id**: INT (Foreign Key to Orders, Not Null, Indexed)
- **payment_method**: ENUM('Credit Card', 'Debit Card', 'Cash', 'Online') (Not Null)
- **amount**: DECIMAL(10, 2) (Not Null)
- **status**: ENUM('Pending', 'Completed', 'Failed') (Default: 'Pending', Not Null)
- **payment_date**: TIMESTAMP (Default: CURRENT_TIMESTAMP)

Rating Table

- **rating_id**: INT (Primary Key, Auto Increment)
- **user_id**: INT (Foreign Key to Users, Not Null, Indexed)
- **restaurant_id**: INT (Foreign Key to Restaurants, Not Null, Indexed)
- **rating**: INT (Check: rating >= 1 AND rating <= 5, Not Null)
- **review**: TEXT
- **rating_date**: TIMESTAMP (Default: CURRENT_TIMESTAMP)

Address Table

- **address_id**: INT (Primary Key, Auto Increment)
- **user_id**: INT (Foreign Key to Users, Not Null, Indexed)
- **state**: VARCHAR(255) (Not Null)
- **city**: VARCHAR(255) (Not Null)
- **street**: VARCHAR(255) (Not Null)
- **pincode**: VARCHAR(10) (Not Null)
- **created_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **updated_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP, Update on modification)

Menu Table

- **menu_id**: INT (Primary Key, Auto Increment)
- **restaurant_id**: INT (Foreign Key to Restaurants, Not Null, Indexed)
- **item_name**: VARCHAR(255) (Not Null)
- **price**: DECIMAL(10, 2) (Not Null)
- **created_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP)
- **updated_at**: TIMESTAMP (Default: CURRENT_TIMESTAMP, Update on modification)

DeliveryExperience Table

- **delivery_experience_id**: INT (Primary Key, Auto Increment)
- **order_id**: INT (Foreign Key to Orders, Not Null, Indexed)
- **driver_id**: INT (Foreign Key to Drivers, Not Null, Indexed)
- **delivery_time**: INT (Not Null, in minutes)
- **delivery_rating**: INT (Check: delivery_rating >= 1 AND delivery_rating <= 5, Not Null)
- **delivery_comments**: TEXT

UserInteraction Table

- **interaction_id**: INT (Primary Key, Auto Increment)
- **user_id**: INT (Foreign Key to Users, Not Null, Indexed)
- **interaction_type**: ENUM('Support', 'Complaint', 'Inquiry', 'Feedback') (Not Null)
- **interaction_detail**: TEXT
- **interaction_date**: TIMESTAMP (Default: CURRENT_TIMESTAMP)

Relationships

1. **Users to Orders**: One-to-Many (A user can place multiple orders).
2. **Restaurants to Orders**: One-to-Many (An order is associated with one restaurant).
3. **Orders to Payment**: One-to-One (Each order has a corresponding payment entry).
4. **Users to Rating**: One-to-Many (Users can rate multiple restaurants).
5. **Restaurants to Rating**: One-to-Many (A restaurant can have multiple ratings).
6. **Users to Address**: One-to-Many (A user can have multiple addresses).
7. **Restaurants to Menu**: One-to-Many (A restaurant can have multiple menu items).
8. **Orders to DeliveryExperience**: One-to-One (Each order has a delivery experience entry).
9. **Drivers to DeliveryExperience**: One-to-Many (Each delivery experience is associated with a driver).

10. **Users to UserInteraction**: One-to-Many (A user can have multiple interactions logged).

Analytics Capabilities:

With an assumed demographics of 1,00,000 people around Mumbai, a lot of user analytics can be performed. They are:

1. User Growth and Engagement
2. Order Trends and Patterns
3. Mode of payment analysis
4. Delivery Time and Rating analysis
5. Average Order value
6. Customer Feedback and Ratings
7. Driver Efficiency and performance
8. Review Sentiment Analysis
9. Restaurant Menu Price Analysis
10. User support and complaint analysis

Cloud Infrastructure:

AWS RDS (Amazon Relational Database Service) is ideal for this sort of database. AWS RDS offers a scalable, simple-to-install, and highly available relational database solution. It is a managed SQL database; hence it is best suited for this solution.

A. **Database Selection:** We compared AWS, GCP, and Azure based on the following criteria, and decided to go with AWS as the cloud infrastructure provider

to store user data and enable analytics to improve business KPIs for the food delivery startup:

1. **Cost**: The following lists the costs for each cloud database provider:
   PostgreSQL on **AWS** RDS: Monthly instance cost: around $80
   GCP: PostgreSQL Cloud SQL: Monthly instance cost: around $89.37
   PostgreSQL **Azure** Database Instance Cost: about $90.72/month
   AWS is the most economical choice for this configuration, according to the cost comparison.

1. **Automated Scaling**: Amazon In addition to enabling infrastructure scalability based on user demand, auto scaling aids in preserving application availability. We can scale our resources if demand increases.
1. **Serverless Architecture**: We can create a highly scalable and reasonably priced backend for the food delivery application by utilizing AWS Lambda and other serverless services.
2. **Availability Zones**: High availability and disaster recovery options are provided by AWS's several availability zones and regions, guaranteeing the dependability and resilience of your services.
3. **Huge Range of Services**: AWS offers the broadest and most comprehensive range of services, ranging from sophisticated artificial intelligence and machine learning to computing and storage.
4. **Extensive data analytics**: Platforms like as AWS Redshift, Kinesis, and QuickSight provide strong data analytics functionalities that enable the extraction of insights from massive data sets to facilitate well-informed business choices. It is going to improve the database's analytical power.
5. **Innovation Pace**: With its frequent releases of new services and capabilities, AWS usually sets the bar for cloud innovation development.
6. Cost Management: AWS offers a range of cost management tools, including AWS Budgets, AWS Cost Explorer, and Trusted Advisor, to assist you in tracking and maximizing your expenses.
7. Broad Ecosystem: It's simpler to add and integrate extra tools and services that can improve your food delivery platform with AWS's extensive ecosystem of third-party connectors.

B. Fixed and Variable cost of the Database infrastructure for 1 year:

Fixed Cost:

1. **Database service subscription:** The estimate is based on the chosen AWS S3 subscription plan.
2. **Storage costs:** Fixed monthly cost for storing data based on the volume of data expected to be stored.
3. **Backup Storage:** Fixed monthly cost for backup storage on the volume of the data to be stored.
4. **Data Transfer:** The Data Transfer rates for the stored data (e.g., GET, POST, PUT, PATCH, DELETE).

Variable Cost:

1. **Additional Storage Costs:**
    a. **Extra Data Storage:** Anticipating a surge in data due to increased user activity, orders, and interactions during the festive season. Let's assume an extra storage for two months.

Cost Comparison Table:

	AWS	GCP	Azure
Instance (12 Months)	$960	$1072.44	$1088.64
Regular Storage (12 Months)	$690	$1020	$720
Backup Storage (12 Months)	$138	$0 (Included in Storage cost)	$720
Data Transfer (12 Months)	$108	$144	$104.40
Addition Storage (2 Months)	$115	$170	$120
TOTAL(12 Months)	$2011	$2406.44	$2753.04

C. Business Metrics to Track User Experience

Running a successful food delivery service necessitates close monitoring of key business performance measures to maintain operational efficiency, customer happiness, and overall business development. From the data that we are gathering from day-to-day operations, we can analyze many essential business metrics.

1. **Order Preparation and Delivery Time**

We are storing the delivery time, and based on that, we can analyze how fast the deliveries are being performed. We can calculate the average delivery time for our business. The lower the delivery time, the better.

2. **Average Order Value**

We can calculate the average order value over the year, if it's increasing, we can assume that the food delivery startup is growing.

3. **Customer Lifetime Value (CLV):**

Estimated revenue from a user over their lifetime as a customer.

4. **Customer Satisfaction**

Satisfied consumers are more likely to submit favorable internet reviews and refer your company to their friends, family, and coworkers, resulting in increased sales. We can calculate customer satisfaction via reviews text analytics and rating scores. Other metrics can be if the average order value is increasing for a customer ID.

5. **Increase in User Base**

An increase in the number of users on the database will indicate that more people are usng the food delivery startup.

6. **Support and Resolution**

The reduction in the number of support and complaint user interactions will imply that the business is running smoothly.

7. **Repeat Purchase Rate:**

Percentage of users who place more than one order. A gradual increase in Repeat purchase rate will be a positive business indicator.

8. **Average Order Rating:**

Average of all ratings provided by users.