# Project Deadline-6

Chahat,Gagan,Nikita | 2022138,2022183,2022323

---

## Transactions:

### Conflicting:

- **Concurrently Update User Contact Details and Place an Order:**
  We are using FOR UPDATE locking to ensure that the order is placed with the most current user information after any updates have been made.

  **Transaction A: Update User Contact Details**

  **MySql Code:**

```
START TRANSACTION;
-- Update contact details with a lock to ensure visibility of changes
UPDATE USERS SET ContactDetails = '+911234567890' WHERE UserID = 1;
COMMIT;
```

  **Transaction B: Place an Order Using User's Updated Contact Details**

  **MySql Code:**

```
START TRANSACTION;
-- Selects and locks the user's details to ensure they are not changed
mid-transaction
SELECT ContactDetails INTO @contact FROM USERS WHERE UserID = 1 FOR UPDATE;
-- Inserts the order using the locked contact details
INSERT INTO ORDERS (UserID, PickupLocation, DropOffLocation, DeliveryTime,
ContactDetails)
VALUES (1, '50 MG Road, Mumbai', '75 Link Road, Mumbai', '2023-12-01 15:00:00',
@contact);
COMMIT;
```

This setup ensures that the order is always placed with the most recent contact details. The FOR UPDATE clause locks the relevant row in the USERS table, preventing other transactions from reading or modifying it until the transaction is completed.

- **Concurrent Order Cancellation and Update:** For this scenario, ensuring that no updates are applied to an order that is about to be canceled is crucial. Using locking or checking the existence of the record before performing operations can manage this.

  **Transaction A: Cancel Order**

  **MySql Code:**

  ```
  START TRANSACTION;
  -- Deletes the order and holds an exclusive lock until the transaction is complete
  DELETE FROM ORDERS WHERE OrderID = 101;
  COMMIT;
  ```

  **Transaction B: Update Delivery Time of the Same Order**

  **MySql Code:**

  ```
  START TRANSACTION;
  -- Attempts to update the delivery time only if the order still exists
  SELECT OrderID FROM ORDERS WHERE OrderID = 101 FOR UPDATE;
  IF FOUND_ROWS() > 0 THEN
     UPDATE ORDERS SET DeliveryTime = '2023-12-02 17:00:00' WHERE OrderID = 101;
  ELSE
     SELECT 'Order no longer exists' AS message;
  END IF;
  COMMIT;
  ```

In this modification, Transaction B uses FOR UPDATE to lock the order record if it exists, preventing Transaction A from deleting it midway. If the record is already deleted by Transaction A, Transaction B will find no rows to lock and will thus not attempt an update.

## Non-Conflicting:

- **User Registration:** A new user registers on the platform, Inserts a new record into the 'USERS' table.
  **MySql Code:**
  START TRANSACTION;
  INSERT INTO USERS (Username, Password, Name, ContactDetails, UserType)
  VALUES ('rajeshkumar', 'securePass123', 'Rajesh Kumar', '+919876543210', 'Customer');
  COMMIT;

- **Place an Order:** Inserts a new record into the 'ORDERS' table linked to a user.
  **MySql Code:**
  *START TRANSACTION;*
  *INSERT INTO ORDERS (UserID, PickupLocation, DropOffLocation, DeliveryTime)*
  *VALUES (1, '10 MG Road, Bengaluru', '25 CP, New Delhi', '2023-12-25 18:30:00');*
  *COMMIT;*

- **Add a Promotion:** Inserts a new promotion into the 'PROMOTIONS' table.
  **MySql Code:**
  *START TRANSACTION;*
  *INSERT INTO PROMOTIONS (Description, DiscountAmount, StartDate, EndDate)*
  *VALUES ('Diwali Dhamaka Sale', 15.00, '2023-11-01', '2023-11-07');*
  *COMMIT;*

- **Submit a Support Ticket:** Inserts a new record into the 'CUSTOMER_SUPPORT' table for a user.
  **MySql Code:**
  *START TRANSACTION;*
  *INSERT INTO CUSTOMER_SUPPORT (UserID, IssueDescription, Status)*
  *VALUES (1, 'Non-delivery of products ordered during Diwali Sale', 'Open');*
  *COMMIT;*

---

# A user guide:

## Features and their working:

1. **User Registration:** Users can register by providing a username, password, full name, contact details, and user type (Customer/Admin).
2. **User Login:** Registered users can log in with their username and password. The system checks credentials and allows access if it is valid.
3. **View Order History:** Users can view their order history, including order ID, pickup location, drop-off location, and delivery time.
4. **View Promotions:** Users can view current promotions, including descriptions, discount amounts, and validity periods.
5. **Contact Customer Support:** Users can contact customer support by describing their issue, which creates a support ticket.
6. **Update Contact Details:** Users can update their contact details.
7. **Place Order:** Users can place an order by providing a pickup location, drop-off location, and optional instructions.
8. **Cancel Order:** Users can cancel orders which are still not processed
9. **Track Order:** Users can track an order by providing the order ID, which displays the order status and current location.
10. **Rate Order:** Users can rate an order from 1 to 5 and provide optional feedback.
11. **Admin Login**: Administrators can log in with their credentials to access admin features.
12. **Manage Users (Admin):** Administrators can view all users, block/unblock users, and delete users.
13. **Manage Orders (Admin):** Administrators can view undergoing orders that are still not processed and delete them.
14. **View and Manage Promotions (Admin):** Administrators can view existing and pre-existing promotions and add new ones.
15. **View and Manage Support Tickets (Admin):** Administrators can view support tickets and change their status.
16. **Admin Reports:** Generate reports on users and orders.

---