BACK END CODE

```
-- ============================================
-- STEP 0: Disable Safe Update Mode
-- ============================================
SET SQL_SAFE_UPDATES = 0;

-- ============================================
-- STEP 1: Drop and create the database
-- ============================================
DROP DATABASE IF EXISTS ComplaintManagement;
CREATE DATABASE ComplaintManagement;
USE ComplaintManagement;

-- ============================================
-- STEP 2: Create Tables
-- ============================================
CREATE TABLE User (
    user_id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    email VARCHAR(50) UNIQUE,
    password VARCHAR(50),
    user_type VARCHAR(20)
);

CREATE TABLE Admin (
    admin_id INT PRIMARY KEY,
    name VARCHAR(50)
);
```

```
CREATE TABLE Complaint (
    complaint_id INT AUTO_INCREMENT PRIMARY KEY,
    user_id INT,
    submit_date DATE,
    description TEXT,
    type VARCHAR(20),
    status VARCHAR(20),
    FOREIGN KEY (user_id) REFERENCES User(user_id) ON DELETE CASCADE
);

CREATE TABLE Response (
    response_id INT AUTO_INCREMENT PRIMARY KEY,
    complaint_id INT,
    admin_id INT,
    response_date DATE,
    response_text TEXT,
    FOREIGN KEY (complaint_id) REFERENCES Complaint(complaint_id) ON DELETE CASCADE,
    FOREIGN KEY (admin_id) REFERENCES Admin(admin_id)
);

CREATE TABLE Complaint_Log (
    log_id INT AUTO_INCREMENT PRIMARY KEY,
    complaint_id INT,
    deleted_on DATETIME
);

-- ==========================================
-- STEP 3: Insert Sample Data
-- ==========================================
INSERT INTO User (name, email, password, user_type) VALUES
('Dhanush', 'dhanush@gmail.com', '1234', 'Student'),
```

```sql
('Rahul', 'rahul@gmail.com', 'abcd', 'Student');

INSERT INTO Admin (admin_id, name) VALUES
(100, 'Akshay'),
(101, 'Rohit'),
(102, 'Priya');

INSERT INTO Complaint (user_id, description, type) VALUES
(1, 'Wi-Fi not working', 'Issue'),
(2, 'Mess food issue', 'Issue'),
(1, 'AC not functioning', 'Issue'),
(2, 'Hostel water problem', 'Issue');

INSERT INTO Response (complaint_id, admin_id, response_date, response_text) VALUES
(2, 100, '2025-10-18', 'Mess issue fixed.'),
(3, 100, '2025-10-19', 'AC has been repaired.');

-- ==========================================
-- STEP 4: Create Triggers
-- ==========================================
DELIMITER //

CREATE TRIGGER set_default_status
BEFORE INSERT ON Complaint
FOR EACH ROW
BEGIN
   SET NEW.status = 'Pending';
   SET NEW.submit_date = CURDATE();
END;
//
```

```
CREATE TRIGGER update_status_on_response
AFTER INSERT ON Response
FOR EACH ROW
BEGIN
   UPDATE Complaint
   SET status = 'Resolved'
   WHERE complaint_id = NEW.complaint_id;
END;
//

CREATE TRIGGER log_complaint_deletion
AFTER DELETE ON Complaint
FOR EACH ROW
BEGIN
   INSERT INTO Complaint_Log (complaint_id, deleted_on)
   VALUES (OLD.complaint_id, NOW());
END;
//

DELIMITER ;

-- ===========================================
-- STEP 5: Create Stored Procedures
-- ===========================================
DELIMITER //

-- Add a complaint
CREATE PROCEDURE AddComplaint(IN p_user_id INT, IN p_description TEXT)
BEGIN
   INSERT INTO Complaint(user_id, description) VALUES(p_user_id, p_description);
   SELECT * FROM Complaint
```

```
    WHERE user_id = p_user_id
    ORDER BY submit_date DESC
    LIMIT 5;
END;
//

-- Add a response
CREATE PROCEDURE AddResponse(IN p_complaint_id INT, IN p_admin_id INT, IN p_response_text
TEXT)
BEGIN
    INSERT INTO Response(complaint_id, admin_id, response_date, response_text)
    VALUES(p_complaint_id, p_admin_id, CURDATE(), p_response_text);
    SELECT * FROM Complaint WHERE complaint_id = p_complaint_id;
END;
//

-- Delete a complaint
CREATE PROCEDURE DeleteComplaint(IN p_complaint_id INT)
BEGIN
    DELETE FROM Complaint WHERE complaint_id = p_complaint_id;
    SELECT * FROM Complaint_Log WHERE complaint_id = p_complaint_id;
END;
//

-- Get pending complaints
CREATE PROCEDURE GetPendingComplaints()
BEGIN
    SELECT c.complaint_id, u.name, c.description
    FROM Complaint c
    JOIN User u ON c.user_id = u.user_id
    WHERE c.status = 'Pending';
```

```
END;
//

-- Get resolved complaints
CREATE PROCEDURE GetResolvedComplaints()
BEGIN
  SELECT c.complaint_id, u.name, c.description
  FROM Complaint c
  JOIN User u ON c.user_id = u.user_id
  WHERE c.status = 'Resolved';
END;
//

-- Get complaints with responses
CREATE PROCEDURE GetComplaintsWithResponses()
BEGIN
  SELECT u.name, c.description, r.response_text, r.response_date, a.name AS admin_name
  FROM Complaint c
  JOIN Response r ON c.complaint_id = r.complaint_id
  JOIN User u ON c.user_id = u.user_id
  JOIN Admin a ON r.admin_id = a.admin_id;
END;
//

-- Search complaints by status
CREATE PROCEDURE SearchComplaintsByStatus(IN p_status VARCHAR(20))
BEGIN
  SELECT c.complaint_id, u.name, c.description, c.status, c.submit_date
  FROM Complaint c
  JOIN User u ON c.user_id = u.user_id
  WHERE c.status = p_status
```

```sql
    ORDER BY c.submit_date DESC;
END;
//

-- ==========================================
-- New Procedures (User-requested)
-- ==========================================

-- 1 Get complaints by user
CREATE PROCEDURE GetComplaintsByUser(IN p_user_id INT)
BEGIN
    SELECT complaint_id, description, type, status, submit_date
    FROM Complaint
    WHERE user_id = p_user_id
    ORDER BY submit_date DESC;
END;
//

-- 3 Get complaints by date range
CREATE PROCEDURE GetComplaintsByDate(IN start_date DATE, IN end_date DATE)
BEGIN
    SELECT c.complaint_id, u.name, c.description, c.status, c.submit_date
    FROM Complaint c
    JOIN User u ON c.user_id = u.user_id
    WHERE c.submit_date BETWEEN start_date AND end_date
    ORDER BY c.submit_date DESC;
END;
//

-- 4 Count complaints by status
CREATE PROCEDURE CountComplaintsByStatus()
```

```sql
BEGIN
    SELECT status, COUNT(*) AS total
    FROM Complaint
    GROUP BY status;
END;
//

DELIMITER ;

-- ==========================================
-- STEP 6: Testing Queries / Output
-- ==========================================
SHOW TABLES;

SELECT * FROM User;
SELECT * FROM Admin;
SELECT * FROM Complaint;
SELECT * FROM Response;

-- Complaints with user name and status
SELECT c.complaint_id, u.name, c.description, c.status
FROM Complaint c JOIN User u ON c.user_id = u.user_id;

-- Example calls
-- CALL GetComplaintsByUser(1);
-- CALL GetComplaintsByDate('2025-10-15', '2025-10-20');
-- CALL CountComplaintsByStatus();
-- CALL SearchComplaintsByStatus('Pending');

-- ==========================================
--  END OF COMPLETE SCRIPT
-- ============================================

select * from Complaint;
```