```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

namespace WebApplication2.DAL
{
   public class myDAL
   {
      private static readonly string connString =
         ConfigurationManager.ConnectionStrings["sqlCon1"].ConnectionString;


      public bool ValidateUser(string email, string password, string role)
      {
         using (SqlConnection con = new SqlConnection(connString))
         {
            string query = "SELECT COUNT(*) FROM Users WHERE Email=@Email AND
Password=@Password AND Role=@Role";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", email);
            cmd.Parameters.AddWithValue("@Password", password);
            cmd.Parameters.AddWithValue("@Role", role);

            con.Open();
            int count = (int)cmd.ExecuteScalar();
            return count > 0;
         }
      }

      public bool RegisterUser(string fname, string lname, string email, string password, string
role)
      {
         using (SqlConnection con = new SqlConnection(connString))
         {
            string checkQuery = "SELECT COUNT(*) FROM Users WHERE Email=@Email";
            SqlCommand checkCmd = new SqlCommand(checkQuery, con);
            checkCmd.Parameters.AddWithValue("@Email", email);

            con.Open();
            int exists = (int)checkCmd.ExecuteScalar();

            if (exists > 0)
            {
                return false;
            }

            string query = "INSERT INTO Users (FirstName, LastName, Email, Password, Role) "
+
                        "VALUES (@FName, @LName, @Email, @Password, @Role)";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@FName", fname);
            cmd.Parameters.AddWithValue("@LName", lname);
            cmd.Parameters.AddWithValue("@Email", email);
            cmd.Parameters.AddWithValue("@Password", password);
            cmd.Parameters.AddWithValue("@Role", role);

            int rows = cmd.ExecuteNonQuery();
            return rows > 0;
        }
    }


    public int GetUserId(string email, string role)
    {
        using (SqlConnection con = new SqlConnection(connString))
        {
            string query = "SELECT Id FROM Users WHERE Email=@Email AND Role=@Role";
            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", email);
            cmd.Parameters.AddWithValue("@Role", role);

            con.Open();
            object result = cmd.ExecuteScalar();
            return result != null ? Convert.ToInt32(result) : 0;
        }
    }

    public int GetInspectorIdByEmail(string email)
    {
        using (SqlConnection con = new SqlConnection(connString))
        using (SqlCommand cmd = new SqlCommand("SELECT Inspector_ID FROM
INSPECTOR WHERE Email=@Email", con))
        {
            cmd.Parameters.AddWithValue("@Email", email);
            con.Open();
            object result = cmd.ExecuteScalar();
            return result != null ? Convert.ToInt32(result) : 0;
        }
    }


    public bool RegisterInspector(string fname, string lname, string email)
    {
        using (SqlConnection con = new SqlConnection(connString))
        using (SqlCommand cmd = new SqlCommand(
            "INSERT INTO INSPECTOR (First_Name, Last_Name, Email) VALUES (@FName,
@LName, @Email)", con))
        {
            cmd.Parameters.AddWithValue("@FName", fname);
            cmd.Parameters.AddWithValue("@LName", lname);
            cmd.Parameters.AddWithValue("@Email", email);

            con.Open();
            int rows = cmd.ExecuteNonQuery();
            return rows > 0;
        }
    }


    public bool SaveRestaurant(string name, string owner, string phone,
                    string email, string address, string city, string zip,
                    string license, string openingDate, string status,
                    string seatingCapacity, string deliveryAvailable,
                    string createdByEmail)
    {
        using (SqlConnection con = new SqlConnection(connString))
        using (SqlCommand cmd = new SqlCommand("sp_SaveRestaurantt", con)) // ✅
matches your procedure name
        {
            cmd.CommandType = CommandType.StoredProcedure;

            DateTime parsedDate;
            if (!DateTime.TryParseExact(openingDate, "yyyy-MM-dd", null,
                System.Globalization.DateTimeStyles.None, out parsedDate))
            {
                throw new Exception("Invalid date format. Please use YYYY-MM-DD.");
            }
```

```
                cmd.Parameters.AddWithValue("@Name", name);
                cmd.Parameters.AddWithValue("@OwnerName", owner);
                cmd.Parameters.AddWithValue("@Phone", phone);
                cmd.Parameters.AddWithValue("@Email", email);
                cmd.Parameters.AddWithValue("@Address", address);
                cmd.Parameters.AddWithValue("@City", city);
                cmd.Parameters.AddWithValue("@ZipCode", zip);
                cmd.Parameters.AddWithValue("@LicenseNumber", license);
                cmd.Parameters.AddWithValue("@OpeningDate", parsedDate);
                cmd.Parameters.AddWithValue("@Status", status);
                cmd.Parameters.AddWithValue("@SeatingCapacity",
string.IsNullOrEmpty(seatingCapacity) ? (object)DBNull.Value :
Convert.ToInt32(seatingCapacity));
                cmd.Parameters.AddWithValue("@DeliveryAvailable",
string.IsNullOrEmpty(deliveryAvailable) ? (object)DBNull.Value : deliveryAvailable);
                cmd.Parameters.AddWithValue("@ManagerEmail", createdByEmail);

                try
                {
                    con.Open();
                    int rows = cmd.ExecuteNonQuery();
                    return rows > 0;
                }
                catch (Exception ex)
                {

                    throw new Exception("SaveRestaurant failed: " + ex.Message);
                }
            }
        }




        public DataTable GetRestaurants(string createdByEmail)
        {
            using (SqlConnection con = new SqlConnection(connString))
            {
                string query = @"SELECT r.Restaurant_ID, r.Name, r.Owner_Name,
r.Phone_Number, r.Email,
                                r.Address, r.City, r.Zip_Code, r.License_Number, r.Opening_Date,
r.Status,
                                d.Seating_Capacity, t.Delivery_Available
```

```
                        FROM RESTAURANT r
                        LEFT JOIN DINE_IN_RESTAURANT d ON r.Restaurant_ID =
d.Restaurant_ID
                        LEFT JOIN TAKEAWAY_OUTLET t ON r.Restaurant_ID = t.Restaurant_ID
                        WHERE r.Created_By_Email = @Email";

            SqlCommand cmd = new SqlCommand(query, con);
            cmd.Parameters.AddWithValue("@Email", createdByEmail);

            SqlDataAdapter da = new SqlDataAdapter(cmd);
            DataTable dt = new DataTable();
            da.Fill(dt);
            return dt;
        }
    }

    public bool UpdateRestaurant(int id, string name, string owner, string phone,
            string email, string address, string city, string zip,
            string license, string openingDate, string status,
            string seatingCapacity, string deliveryAvailable,
            string createdByEmail)
    {
        using (SqlConnection con = new SqlConnection(connString))
        {
            con.Open();
            SqlTransaction tran = con.BeginTransaction();

            try
            {
                DateTime parsedDate;
                if (!DateTime.TryParseExact(openingDate, "yyyy-MM-dd", null,
                    System.Globalization.DateTimeStyles.None, out parsedDate))
                {
                    throw new Exception("Invalid date format. Please use YYYY-MM-DD.");
                }

                string query = @"UPDATE RESTAURANT SET
                 Name=@Name, Owner_Name=@Owner, Phone_Number=@Phone,
Email=@Email, Address=@Address, City=@City, Zip_Code=@Zip,
                 License_Number=@License, Opening_Date=@OpeningDate, Status=@Status
                 WHERE Restaurant_ID=@Id AND Created_By_Email=@CreatedByEmail";

                SqlCommand cmd = new SqlCommand(query, con, tran);
                cmd.Parameters.AddWithValue("@Id", id);
                cmd.Parameters.AddWithValue("@Name", name);
                cmd.Parameters.AddWithValue("@Owner", owner);
                cmd.Parameters.AddWithValue("@Phone", phone);
                cmd.Parameters.AddWithValue("@Email", email);
                cmd.Parameters.AddWithValue("@Address", address);
                cmd.Parameters.AddWithValue("@City", city);
                cmd.Parameters.AddWithValue("@Zip", zip);
                cmd.Parameters.AddWithValue("@License", license);
                cmd.Parameters.AddWithValue("@OpeningDate", parsedDate);
                cmd.Parameters.AddWithValue("@Status", status);
                cmd.Parameters.AddWithValue("@CreatedByEmail", createdByEmail);

                int rows = cmd.ExecuteNonQuery();


                if (rows > 0)
                {
                    if (!string.IsNullOrEmpty(seatingCapacity))
                    {
                        string dineQuery = @"IF EXISTS (SELECT 1 FROM DINE_IN_RESTAURANT
WHERE Restaurant_ID=@Id)
                            UPDATE DINE_IN_RESTAURANT SET Seating_Capacity=@Capacity
WHERE Restaurant_ID=@Id
                            ELSE
                            INSERT INTO DINE_IN_RESTAURANT (Restaurant_ID,
Seating_Capacity) VALUES (@Id, @Capacity)";
                        SqlCommand dineCmd = new SqlCommand(dineQuery, con, tran);
                        dineCmd.Parameters.AddWithValue("@Id", id);
                        dineCmd.Parameters.AddWithValue("@Capacity",
Convert.ToInt32(seatingCapacity));
                        dineCmd.ExecuteNonQuery();
                    }

                    if (!string.IsNullOrEmpty(deliveryAvailable))
                    {
                        string takeQuery = @"IF EXISTS (SELECT 1 FROM TAKEAWAY_OUTLET
WHERE Restaurant_ID=@Id)
                            UPDATE TAKEAWAY_OUTLET SET Delivery_Available=@Delivery
WHERE Restaurant_ID=@Id
                            ELSE
                            INSERT INTO TAKEAWAY_OUTLET (Restaurant_ID, Delivery_Available)
VALUES (@Id, @Delivery)";
```

```
                SqlCommand takeCmd = new SqlCommand(takeQuery, con, tran);
                takeCmd.Parameters.AddWithValue("@Id", id);
                takeCmd.Parameters.AddWithValue("@Delivery", deliveryAvailable);
                takeCmd.ExecuteNonQuery();
            }
        }

        tran.Commit();
        return rows > 0;
    }
    catch (Exception ex)
    {
        tran.Rollback();
        throw new Exception("UpdateRestaurant Error: " + ex.Message);
    }
}



}
public bool DeleteRestaurant(int id, string createdByEmail)
{
    using (SqlConnection con = new SqlConnection(connString))
    {
        con.Open();
        SqlTransaction tran = con.BeginTransaction();

        try
        {

            string archiveQuery = @"INSERT INTO RESTAURANT_ARCHIVE
        SELECT r.Restaurant_ID, r.Name, r.Owner_Name, r.Phone_Number, r.Email,
               r.Address, r.City, r.Zip_Code, r.License_Number, r.Opening_Date,
               r.Status, d.Seating_Capacity, t.Delivery_Available, GETDATE()
        FROM RESTAURANT r
        LEFT JOIN DINE_IN_RESTAURANT d ON r.Restaurant_ID = d.Restaurant_ID
        LEFT JOIN TAKEAWAY_OUTLET t ON r.Restaurant_ID = t.Restaurant_ID
        WHERE r.Restaurant_ID = @Id AND r.Created_By_Email = @CreatedByEmail";

            using (SqlCommand archiveCmd = new SqlCommand(archiveQuery, con, tran))
            {
                archiveCmd.Parameters.AddWithValue("@Id", id);
                archiveCmd.Parameters.AddWithValue("@CreatedByEmail", createdByEmail);
```

```
                    archiveCmd.ExecuteNonQuery();
                }


                string delDine = "DELETE FROM DINE_IN_RESTAURANT WHERE
Restaurant_ID=@Id";
                using (SqlCommand cmdDine = new SqlCommand(delDine, con, tran))
                {
                    cmdDine.Parameters.AddWithValue("@Id", id);
                    cmdDine.ExecuteNonQuery();
                }

                string delTake = "DELETE FROM TAKEAWAY_OUTLET WHERE
Restaurant_ID=@Id";
                using (SqlCommand cmdTake = new SqlCommand(delTake, con, tran))
                {
                    cmdTake.Parameters.AddWithValue("@Id", id);
                    cmdTake.ExecuteNonQuery();
                }


                string delMain = "DELETE FROM RESTAURANT WHERE Restaurant_ID=@Id
AND Created_By_Email=@CreatedByEmail";
                using (SqlCommand cmdMain = new SqlCommand(delMain, con, tran))
                {
                    cmdMain.Parameters.AddWithValue("@Id", id);
                    cmdMain.Parameters.AddWithValue("@CreatedByEmail", createdByEmail);
                    int rows = cmdMain.ExecuteNonQuery();

                    tran.Commit();
                    return rows > 0;
                }
            }
            catch (Exception ex)
            {
                tran.Rollback();
                throw new Exception("DeleteRestaurant Error: " + ex.Message);
            }
        }
    }

    public bool AssignInspection(int restaurantId, int inspectorId, DateTime date, TimeSpan?
time, string type)
```

```
        {
            using (SqlConnection con = new SqlConnection(connString))
            {

                string checkQuery = "SELECT COUNT(*) FROM INSPECTION WHERE
Restaurant_ID=@RestaurantID AND Inspection_Date=@Date";
                SqlCommand checkCmd = new SqlCommand(checkQuery, con);
                checkCmd.Parameters.AddWithValue("@RestaurantID", restaurantId);
                checkCmd.Parameters.AddWithValue("@Date", date);

                con.Open();
                int exists = (int)checkCmd.ExecuteScalar();
                if (exists > 0)
                {
                    return false;
                }


                string insertQuery = @"INSERT INTO INSPECTION
            (Restaurant_ID, Inspector_ID, Inspection_Date, Inspection_Time, Inspection_Type,
Outcome, Follow_Up_Required)
            VALUES (@RestaurantID, @InspectorID, @Date, @Time, @Type, 'Conditional', 'No')";

                SqlCommand cmd = new SqlCommand(insertQuery, con);
                cmd.Parameters.AddWithValue("@RestaurantID", restaurantId);
                cmd.Parameters.AddWithValue("@InspectorID", inspectorId);
                cmd.Parameters.AddWithValue("@Date", date);
                cmd.Parameters.AddWithValue("@Time", (object)time ?? DBNull.Value);
                cmd.Parameters.AddWithValue("@Type", type);

                int rows = cmd.ExecuteNonQuery();
                return rows > 0;
            }
        }

        public DataTable GetInspectionDetails(int restaurantId)
        {
            using (SqlConnection con = new SqlConnection(connString))
            {
                string query = @"
            SELECT i.Inspection_ID,
                   i.Inspection_Date,
                   i.Inspection_Time,
                   i.Inspection_Type,
                   i.Outcome,
                   i.Follow_Up_Required,
                   v.Violation_Code,
                   v.Description,
                   v.Severity,
                   v.Fine_Amount,
                   v.Corrective_Action,
                   v.Resolved_Date
            FROM INSPECTION i
            LEFT JOIN VIOLATION v ON i.Inspection_ID = v.Inspection_ID
            WHERE i.Restaurant_ID = @RestaurantID
            ORDER BY i.Inspection_Date DESC;";
                SqlCommand cmd = new SqlCommand(query, con);
                cmd.Parameters.AddWithValue("@RestaurantID", restaurantId);

                SqlDataAdapter da = new SqlDataAdapter(cmd);
                DataTable dt = new DataTable();
                da.Fill(dt);
                return dt;
            }
        }


        public bool AddViolation(int inspectionId, int categoryId, string code, string description,
                                 string severity, decimal fine, string correctiveAction, DateTime?
resolvedDate)
        {
            using (SqlConnection con = new SqlConnection(connString))
            {
                string query = @"INSERT INTO VIOLATION
                        (Inspection_ID, Category_ID, Violation_Code, Description, Severity,
Fine_Amount, Corrective_Action, Resolved_Date)
                        VALUES (@InspectionID, @CategoryID, @Code, @Description, @Severity,
@Fine, @Action, @ResolvedDate)";

                SqlCommand cmd = new SqlCommand(query, con);
                cmd.Parameters.AddWithValue("@InspectionID", inspectionId);
                cmd.Parameters.AddWithValue("@CategoryID", categoryId);
                cmd.Parameters.AddWithValue("@Code", code);
                cmd.Parameters.AddWithValue("@Description", description);
                cmd.Parameters.AddWithValue("@Severity", severity);
                cmd.Parameters.AddWithValue("@Fine", fine);
                cmd.Parameters.AddWithValue("@Action", correctiveAction);
                cmd.Parameters.AddWithValue("@ResolvedDate", (object)resolvedDate ??
DBNull.Value);

                con.Open();
                int rows = cmd.ExecuteNonQuery();
                return rows > 0;
            }
        }

    }
}
```