## Lab Manual: Transactions and Stored Procedures in ADO.NET (Blazor Server)

### Introduction

This lab focuses on working with database transactions and stored procedures using ADO.NET in a Blazor Server application. You will learn how to ensure data integrity with transactions and improve performance and security using stored procedures.

### Objectives

By the end of this lab, students will be able to:

- Understand the role and importance of transactions in database operations.
- Implement transactions in ADO.NET to manage multiple operations as a single unit.
- Create and use stored procedures in SQL Server.
- Call stored procedures from a Blazor Server app using ADO.NET.
- Pass and retrieve parameters using stored procedures.

### Walkthrough

#### Understanding Transactions in ADO.NET

A transaction is used to group multiple SQL operations into a single unit. Either all operations succeed, or none do (commit or rollback).

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    // Open the database connection
    connection.Open();

    // Begin a SQL transaction
    SqlTransaction transaction = connection.BeginTransaction();

    // Create a SQL command and assign the transaction to it
    SqlCommand command = connection.CreateCommand();
    command.Transaction = transaction;
```

```
    try
    {
        // Deduct 100 from AccountID = 1
        command.CommandText = "UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = 1";
        command.ExecuteNonQuery();

        // Add 100 to AccountID = 2
        command.CommandText = "UPDATE Accounts SET Balance = Balance + 100 WHERE AccountID = 2";
        command.ExecuteNonQuery();

        // Commit the transaction if both commands succeed
        transaction.Commit();
    }
    catch
    {
        // Roll back the transaction if any command fails
        transaction.Rollback();
    }
}
```

**Creating and Using Stored Procedures**

**Step 1: Create Stored Procedure in SSMS**

```
CREATE PROCEDURE GetEmployeeById
    @EmployeeID INT
AS
BEGIN
    SELECT * FROM Employees WHERE Id = @EmployeeID
END
```

**Step 2: Use it in Blazor via ADO.NET**

```
using (SqlConnection connection = new SqlConnection(connectionString))
{
    connection.Open();
    SqlCommand command = new SqlCommand("GetEmployeeById", connection);
    command.CommandType = CommandType.StoredProcedure;
    command.Parameters.AddWithValue("@EmployeeID", 1);

    SqlDataReader reader = command.ExecuteReader();
    while (reader.Read())
    {
        Console.WriteLine(reader["Name"]);
    }
    reader.Close();
}
```

**Lab Tasks:**

**Task1: Implement a fund transfer module using ADO.NET transactions as follows:**

SQL Task:

1. Create tables: Accounts and Transactions.

```
CREATE TABLE Accounts (
    AccountId INT PRIMARY KEY IDENTITY,
    AccountHolder NVARCHAR(100),
    Balance DECIMAL(10, 2)
);

CREATE TABLE Transactions (
    TransactionId INT PRIMARY KEY IDENTITY,
    FromAccountId INT,
    ToAccountId INT,
    Amount DECIMAL(10, 2),
    TransferDate DATETIME DEFAULT GETDATE()
);
```

2. Insert at least 5 sample accounts.
3. Create a stored procedure TransferFunds using transactions:
    - Deduct from one account.

```
-- Debit from sender
UPDATE Accounts
SET Balance = Balance - @Amount
WHERE AccountId = @FromAccountId;
```

    - Add to another.

```
-- Credit to receiver
UPDATE Accounts
SET Balance = Balance + @Amount
WHERE AccountId = @ToAccountId;
```

    - Log the transfer in the Transactions table.

```
-- Log the transaction
INSERT INTO Transactions (FromAccountId, ToAccountId, Amount)
VALUES (@FromAccountId, @ToAccountId, @Amount);
```

    - Rollback if any step fails.

Blazor Task:

4. Create a form to:
    - Input sender ID, receiver ID, and amount.
    - On submission, call the stored procedure using ADO.NET.
    - Show success or error message based on transaction result.

**Task 2: Create a Mini Employee Management System using the following:**

1: SQL Setup

- Create tables: Departments and Employees.
- Insert at least 10 records in each.

2: Create Stored Procedures

- Write stored procedures for Insert, Select, Update, and Delete for Employees (using JOIN with Departments).

3: Blazor Integration

- Create a Blazor Server app that:
    - Displays all employees (name, salary, department).
    - Provides forms for adding, updating, and deleting employees.
    - Uses only stored procedures and ADO.NET.