2026

Lab 5 Report

SUBMITTED BY:

DANIYA AAMIR

Problem 1: New Customer Registration (Database Identity & Insertion)

This page inserts new customer records into the CUSTOMER_t database table. The table was updated at the database level to utilize an Auto-Incrementing Identity column for Customer_Id, preventing race conditions and eliminating the need for manual ID generation. Parameterized queries were utilized to prevent SQL injection.

register.aspx.vb

```
Imports System.Data.SqlClient


Public Class register

    Inherits System.Web.UI.Page


    Protected Sub btnRegister_Click(sender As Object, e As EventArgs) Handles btnRegister.Click

        'connection string

        Dim connString As String =
ConfigurationManager.ConnectionStrings("PVFCConnString").ConnectionString


        'Define the parameterized INSERT command; Customer_ID has been set to auto-increment
in the DB via SSMS

        Dim insertSQL As String = "INSERT INTO CUSTOMER_T (Customer_Name,
Customer_Address, Customer_City, Customer_State, Postal_Code) VALUES (@name, @address,
@city, @state, @zip)"


        Dim con As New SqlConnection(connString)

        Dim cmd As New SqlCommand(insertSQL, con)


        'Bind parameters
```

```
        cmd.Parameters.AddWithValue("@name", txtName.Text)
        cmd.Parameters.AddWithValue("@address", txtAddress.Text)
        cmd.Parameters.AddWithValue("@city", txtCity.Text)
        cmd.Parameters.AddWithValue("@state", txtState.Text)
        cmd.Parameters.AddWithValue("@zip", txtPostalCode.Text)

        'Execute and display eror or success signals
        Try
            con.Open()
            Dim added As Integer = cmd.ExecuteNonQuery()

            ' SUCCESS SIGNAL: Turn label green and display confirmation
            lblMessage.ForeColor = System.Drawing.Color.Green
            lblMessage.Text = "Success! " & added.ToString() & " account created for " &
txtName.Text & "."

        Catch Err As Exception
            ' ERROR SIGNAL: Turn label red and display the exception message
            lblMessage.ForeColor = System.Drawing.Color.Red
            lblMessage.Text = "Registration Error: " & Err.Message
        Finally
            con.Close()
        End Try
    End Sub
```

End Class

Problem 2: Customer Profile Update

This page provides a two-step interface for existing customers to update their information. It utilizes a Two-Factor Search (Full Name and Postal Code) to accurately identify the customer record without requiring them to memorize their Primary Key.

updateCustomer.aspx

```
<label>Enter Your Full Name:</label>

<asp:TextBox ID="txtSearchName" runat="server" required="true" placeholder="e.g., Contemporary Casuals"></asp:TextBox>


<label>Enter Your Postal Code:</label>

<div style="display: flex; gap: 10px; margin-bottom: 20px;">

   <asp:TextBox ID="txtSearchZip" runat="server" required="true" placeholder="e.g., 32601-2871"></asp:TextBox>

   <asp:Button ID="btnFetch" runat="server" Text="Find My Profile" />

</div>


<asp:Panel ID="pnlEditProfile" runat="server" Visible="false">

   <asp:HiddenField ID="hfCustomerId" runat="server" />

   <label>Full Name</label>

   <asp:TextBox ID="txtName" runat="server" required="true"></asp:TextBox>

   <asp:Button ID="btnUpdate" runat="server" Text="Save Changes" />

</asp:Panel>
```

updateCustomer.aspx.vb

```
Imports System.Data.SqlClient

Imports System.Configuration


Public Class updateCustomer

    Inherits System.Web.UI.Page


    Dim connString As String =
ConfigurationManager.ConnectionStrings("PVFCConnString").ConnectionString


    'Fetching customer data by name and postal code

    Protected Sub btnFetch_Click(sender As Object, e As EventArgs) Handles btnFetch.Click

        'searching using both criteria to ensure uniqueness

        Dim fetchSQL As String = "SELECT Customer_Id, Customer_Name, Customer_Address,
Customer_City, Customer_State, Postal_Code FROM CUSTOMER_t WHERE Customer_Name =
@name AND Postal_Code = @zip"


            Using con As New SqlConnection(connString)

                Using cmd As New SqlCommand(fetchSQL, con)

                'binding both search parameters

                cmd.Parameters.AddWithValue("@name", txtSearchName.Text)

                    cmd.Parameters.AddWithValue("@zip", txtSearchZip.Text)


                    Try

                        con.Open()
```

```
                Dim reader As SqlDataReader = cmd.ExecuteReader()

                If reader.Read() Then
                'storing the Primary Key securely in the hidden field
                hfCustomerId.Value = reader("Customer_Id").ToString()

                'populating the editable text boxes
                txtName.Text = reader("Customer_Name").ToString()
                   txtAddress.Text = reader("Customer_Address").ToString()
                   txtCity.Text = reader("Customer_City").ToString()
                   txtState.Text = reader("Customer_State").ToString()
                   txtPostalCode.Text = reader("Postal_Code").ToString()

                'unhiding the editing panel
                pnlEditProfile.Visible = True

                   lblMessage.ForeColor = System.Drawing.Color.Blue
                   lblMessage.Text = "Profile found! Make your changes below."
                Else
                'Keep panel hidden if no exact match is found
                pnlEditProfile.Visible = False
                   lblMessage.ForeColor = System.Drawing.Color.Red
                   lblMessage.Text = "We couldn't find a record matching that exact Name and
Postal Code."
```

```
                End If
                reader.Close()
            Catch Err As Exception
                lblMessage.ForeColor = System.Drawing.Color.Red
                lblMessage.Text = "Error fetching profile: " & Err.Message
            End Try
        End Using
    End Using
End Sub

'Saving updated customer data
Protected Sub btnUpdate_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click
    'pdating based on the Hidden ID
    Dim updateSQL As String = "UPDATE CUSTOMER_t SET Customer_Name = @name,
Customer_Address = @address, Customer_City = @city, Customer_State = @state, Postal_Code
= @zip WHERE Customer_Id = @id"

        Using con As New SqlConnection(connString)
            Using cmd As New SqlCommand(updateSQL, con)
                cmd.Parameters.AddWithValue("@name", txtName.Text)
                cmd.Parameters.AddWithValue("@address", txtAddress.Text)
                cmd.Parameters.AddWithValue("@city", txtCity.Text)
                cmd.Parameters.AddWithValue("@state", txtState.Text)
                cmd.Parameters.AddWithValue("@zip", txtPostalCode.Text)
                cmd.Parameters.AddWithValue("@id", hfCustomerId.Value)
```

```
            Try
                con.Open()
                Dim rowsAffected As Integer = cmd.ExecuteNonQuery()

                If rowsAffected > 0 Then
                    lblMessage.ForeColor = System.Drawing.Color.Green
                    lblMessage.Text = "Success! Your profile has been updated."

                'hiding the panel again and updating the search boxes to reflect the new data
                pnlEditProfile.Visible = False
                    txtSearchName.Text = txtName.Text
                    txtSearchZip.Text = txtPostalCode.Text
                Else
                    lblMessage.ForeColor = System.Drawing.Color.Red
                    lblMessage.Text = "Update failed. Something went wrong."
                End If
            Catch Err As Exception
                lblMessage.ForeColor = System.Drawing.Color.Red
                lblMessage.Text = "Error updating profile: " & Err.Message
            End Try
        End Using
    End Using
End Sub
```

End Class

Problem 3: Product Catalog Update

Provides administrators with an interface to interact with and update inventory line items. The previously hardcoded table was upgraded to a dynamic ASP.NET GridView using TemplateFields, automatically binding to the Product_t table and securely updating rows via the RowCommand event.

catalog.aspx

```
<asp:GridView ID="gvCatalog" runat="server" AutoGenerateColumns="False" Width="100%"
  OnRowCommand="gvCatalog_RowCommand" DataKeyNames="Product_Id">
  <Columns>
    <asp:BoundField DataField="Product_Id" HeaderText="ID" ReadOnly="True" />
    <asp:TemplateField HeaderText="Description">
      <ItemTemplate>
        <asp:TextBox ID="txtDesc" runat="server" Text='<%# Bind("Product_Description")
%>'></asp:TextBox>
      </ItemTemplate>
    </asp:TemplateField>
    <asp:TemplateField HeaderText="Action">
      <ItemTemplate>
        <asp:Button ID="btnUpdate" runat="server" Text="Update"
          CommandName="UpdateRow" CommandArgument='<%# Container.DataItemIndex
%>' />
      </ItemTemplate>
    </asp:TemplateField>
```

```
    </Columns>
</asp:GridView>
```

catalog.aspx.vb

```
Imports System.Data.SqlClient
Imports System.Configuration

Public Class catalog
    Inherits System.Web.UI.Page

    Dim connString As String =
ConfigurationManager.ConnectionStrings("PVFCConnString").ConnectionString

    'Load the entire catalog into the GridView on page load
    Protected Sub Page_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles
Me.Load
        If Not IsPostBack Then
            BindCatalog()
        End If
    End Sub

    ' Helper method to fetch and bind products
    Private Sub BindCatalog()
        Dim selectSQL As String = "SELECT Product_Id, Product_Description, Product_Finish,
Standard_Price FROM Product_t"
```

```
        Using con As New SqlConnection(connString)
            Using cmd As New SqlCommand(selectSQL, con)
                Try
                    con.Open()
                    Dim reader As SqlDataReader = cmd.ExecuteReader()
                    gvCatalog.DataSource = reader
                gvCatalog.DataBind() ' Automatically create rows
                reader.Close()
                Catch Err As Exception
                    lblMessage.ForeColor = System.Drawing.Color.Red
                    lblMessage.Text = "Error loading catalog: " & Err.Message
                End Try
            End Using
        End Using
    End Sub


    'Handle the "Update" button click inside the GridView
    Protected Sub gvCatalog_RowCommand(sender As Object, e As
GridViewCommandEventArgs)
        If e.CommandName = "UpdateRow" Then
            ' Identify which row was clicked based on the CommandArgument
            Dim rowIndex As Integer = Convert.ToInt32(e.CommandArgument)
            Dim row As GridViewRow = gvCatalog.Rows(rowIndex)
```

```
        ' Extract the Primary Key (Product_Id) securely from the DataKeys property
        Dim productId As Integer = Convert.ToInt32(gvCatalog.DataKeys(rowIndex).Value)

    'find the textboxes inside that specific row
    Dim txtDesc As TextBox = CType(row.FindControl("txtDesc"), TextBox)
        Dim txtFinish As TextBox = CType(row.FindControl("txtFinish"), TextBox)
        Dim txtPrice As TextBox = CType(row.FindControl("txtPrice"), TextBox)

        Dim updateSQL As String = "UPDATE Product_t SET Product_Description = @desc,
Product_Finish = @finish, Standard_Price = @price WHERE Product_Id = @id"

        Using con As New SqlConnection(connString)
            Using cmd As New SqlCommand(updateSQL, con)
                cmd.Parameters.AddWithValue("@desc", txtDesc.Text)
                cmd.Parameters.AddWithValue("@finish", txtFinish.Text)
                cmd.Parameters.AddWithValue("@price", Convert.ToDecimal(txtPrice.Text))
                cmd.Parameters.AddWithValue("@id", productId)

                Try
                    con.Open()
                    Dim rowsAffected As Integer = cmd.ExecuteNonQuery()

                    If rowsAffected > 0 Then
                        ' SUCCESS SIGNAL
```

```
                    lblMessage.ForeColor = System.Drawing.Color.Green
                    lblMessage.Text = "Success! Product ID " & productId.ToString() & " has been
updated."

                ' Re-bind the grid to ensure fresh data
                BindCatalog()
                Else
                  ' ERROR SIGNAL
                  lblMessage.ForeColor = System.Drawing.Color.Red
                  lblMessage.Text = "Update failed. Product ID not found."
                End If
              Catch Err As Exception
                lblMessage.ForeColor = System.Drawing.Color.Red
                lblMessage.Text = "Error updating product: " & Err.Message
              End Try
            End Using
          End Using
        End If
      End Sub
  End Class
```

Problem 4: Update Algorithms

Product Selection & Order Placement Algorithm:

1. **Catalog Initialization:** Render the product search interface, allowing the customer to browse the inventory and apply filters based on category (e.g., Natural Ash, Walnut, Cherry) or specific text queries.

2. **Product Selection:** Capture the customer's specific product choice. Retrieve the corresponding Product_Id, Product_Description, and Standard_Price from the Product_t database table.

3. **Cart Staging:** Store the selected item details in a temporary session state or cart payload and transition the user to the secure payment interface.

4. **Order Summary Generation:** Dynamically render the Order Summary on the checkout page, calculating and displaying the final total price (e.g., "Computer Desk (Cherry) - 25,000 PKR").

5. **Data Collection:** Prompt the user to select a preferred payment method (e.g., Visa / Master Card, Cash on Delivery) and input their secure billing credentials (Card Number, Expiry, CVV).

6. **Validation & Processing:** Upon submission, validate the frontend input fields to ensure required data formats are met.

7. **Database Commit:** Execute a parameterized INSERT statement to permanently record the transaction details into the ORDER_t (and corresponding ORDER_LINE_t) tables, associating the order with the active Customer_Id.

8. **Confirmation:** Clear the temporary cart session and return a final order success confirmation to the user.

Problem 5: System Test Cases

Test Case	Description	Input Data	Expected Output
1	Register New Customer	Name: "Daniya Aamir", Zip: "12345-6678"	Success message displayed; new row added to CUSTOMER_t with auto-

			generated ID.
2	Customer Fetch (Valid)	Name: "Contemporary Casuals", Zip: "32601-2871"	Profile found message; Edit panel becomes visible with populated data.
3	Customer Fetch (Invalid)	Name: "Fake Name", Zip: "00000"	"No record matching" error message; Edit panel remains hidden.
4	Update Customer	Name: "Daniya Aamir", Zip: "12345-6678" --- Changed **Customer_Address** From “1234 some street” To “4567 some street”	Fetch Successful --- Change successful; Reflected in the DB.
5	Catalog Price Update	Change End Table price to 180 and click Update	Success message; GridView reloads showing 199.99 for the End Table.