Task 2 Development Documentation

CARL BAINES

Contents

Version Changelog

V1.0

Version 1.0

Version 1.0 is most basic version of my application. It includes basic functionality but optional features of the application such as user accessibility options and security features such as the hashing of passwords and user access levels have not been implemented yet.

Bugs

- The sign-up page initially rejects user-inputted usernames and passwords that don't meet the requirements for database storage. However, after the error message box the code displays the message box that says an account has been created successfully, redirects the user to the home page and creates a record with the invalid details inside of the userDetails database.
- The input validation on the Forgot Password page allows the user to change the password of an account even if they enter an invalid username.
- The change account details page allows the user to create a new username and change their password even if they enter an invalid existing username.
- The code breaks when null inputs are entered on the Zoo and Hotel booking pages.

V1.01

Version 1.01

- Added a password char feature so that the password inputs are hidden by a '*' character as they are inputted by the user. Added across all pages that require password inputs.
- Added BCrypt hashing when storing passwords in the userDetails database. The sign in page now converts the user-inputted password for sign in into a hash which is then compared to the hash key stored in the database, this is so that validation does not completely break as it would be comparing unhashed passwords to hashed passwords.
- The upper bounds of input validation checks for user-inputted passwords are increased to account for the fact that the passwords are now stored as hashes.
- Fixed an issue on the sign-up page where it would allow the user to sign-up and access the home page with a username and password of an inappropriate length.
- Added a font size changer feature which changes the font size of common controls across all forms.

Bugs

- Adding the font size changer feature broke the entire code as the accessibilityHelper instance of the AccessibilityHelper kept returning null. All the code had to be commented out in the next version.

V1.02

Version 1.02

- Commented out the font size changer code across all forms so that the application could be functional again.
- Fixed the issue where the forgot password page was not restricting users from resetting their password if they had entered an invalid current username.
- Fixed the issue where the change account details page was not restricting users from resetting their password if they had entered an invalid current username.

V1.03

Version 1.03

- Re-added the font size changer accessibility feature and fixed the errors when implementing it before. It is now functions as intended.
- Changed error message boxes to have cross icons and error message box headings.
- Removed the continue without sign-in button on the sign in page and the continue without account creation button on the sign-up page. This is so that user access level control does not need to be implemented into the code as the user will now have to create an account to gain access to the application.

V1.04

Version 1.04

- Implements a functioning high contrast mode by clicking the checkbox in the settings page.
- Implemented input validation on the sign-up page to prevent identical registered usernames from being added to records in the userDetails database.
- Implemented input validation to prevent identical registered usernames from being added to records in the userDetails database, via the change account details page.
- Added a consistent use of comments throughout the source code for each form.

Classes

ACCESSIBILITY CLASS

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
```

```
namespace Task2_Code_LL_000013680_Baines_C
{
   public class Accessibility
   {
      public class AccessibilityHelper
      {
         public float? fontSize { get; set; }

         public User currentUser { get; set; } = null;

         public AccessibilityHelper() { }
         public AccessibilityHelper(float fontSize)
         {
            this.fontSize = fontSize;
         }

         public void UpdateFontSize(Control.ControlCollection controls)
         {
            if (fontSize == null) return;

            foreach (Control control in controls)
            {
               control.Font = new System.Drawing.Font(control.Font.Name, (float)fontSize);
            }
         }


      }



   }
}
```

DESCRIPTION OF CODE

This is the code for the accessibility class file used in my application. It returns the value of fontSize and assigns a value to the variable. The fontSize value is taken as a parameter into the AccessibilityHelper class and it gets the current instance of the parameter. The class file also contains an UpdateFontSize method which takes the collection of controls in C# as a parameter. It checks to see if the fontSize entered, from the settings page, is null using selection. If the fontSize is not null, the method loops through each common control in the forms and sets the font of each control across all the forms in the application to the fontSize value.

Modules

BCRYPT.NET

Bcrypt.Net is a module that allows you to perform hashing on user inputs. I will use it in my application to hash passwords in the userDetails database and verify inputs by converting them into hashes.

EXAMPLE OF USE

```
string hashedRegisteredPassword = BCrypt.Net.BCrypt.HashPassword(txtCreatePassword.Text);
```

Here, Bcrypt is used to hash user-inputted passwords for database storage.

SQL Stored Procedures

ADDHOTELRECORD

```
CREATE PROCEDURE [dbo].[AddHotelRecord]

(

@checkInDate nvarchar(50),
@numOfResidents int,
@RoomType nvarchar(25),
@checkOutDate nvarchar(50)

)

as

begin

Insert into HotelBookingDetails values (@checkInDate, @numOfResidents, @RoomType, @checkOutDate)

End
```

DESCRIPTION

This stored procedure adds a new hotel booking record to the HotelBookingDetails database. It takes various parameters such as a checkInDate, number of residents etc, which are inserted as values into the database.

ADDRECORD

```
CREATE PROCEDURE [dbo].[AddRecord]

(
```

```
@userName nvarchar(25),
@password nvarchar(50),
@highContrast bit

)

as

begin

Insert into userDetails values (@userName, @password, @highContrast)

End
```

DESCRIPTION

This stored procedure adds a new record to the userDetails database. It takes a username, password and high contrast parameter and inserts them into the database as values.

ADDZOORECORD

```
CREATE PROCEDURE [dbo].[AddZooRecord]

(

        @visitDate nvarchar(50),
        @numOfChildTickets int,
        @numOfAdultTickets int

)

as

begin

Insert into ZooBookingDetails values (@visitDate, @numOfChildTickets, @numOfAdultTickets)

End
```

DESCRIPTION

This stored procedure adds a new zoo booking record to the ZooBookingDetails database. It takes various parameters such as a visitDate, number of child tickets etc, which are inserted as values into the database.

GETHOTELBOOKINGDETAILS

```
CREATE PROCEDURE [dbo].[GetHotelBookingDetails]

as
```

```
begin

select * from HotelBookingDetails

End
```

DESCRIPTION

This stored procedure fetches all values from the HotelBookingDetails database.

GETUSERDETAILS

```
CREATE PROCEDURE [dbo].[GetUserDetails]

as

begin

select * from userDetails

End
```

DESCRIPTION

This stored procedure fetches all values from the userDetails database.

GETZOOBOOKINGDETAILS

```
CREATE PROCEDURE [dbo].[GetZooBookingDetails]

as

begin

select * from ZooBookingDetails

End
```

DESCRIPTION

This stored procedure fetches all values from the ZooBookingDetails database.

UPDATEPASSWORD

```
CREATE PROCEDURE [dbo].[UpdatePassword]

(

@password nvarchar(50)

)
```

```
as

begin

update userDetails
set password = @password


end
```

DESCRIPTION

This stored procedure updates the password in the userDetails database.

UPDATERECORD

```
CREATE PROCEDURE [dbo].[UpdateRecord]

(

@StdId int,
@userName nvarchar(25),
@password nvarchar(50),
@highContrast bit

)
as

begin

update userDetails
set userName=@userName,
password=@password,
highContrast=@highContrast
where Id=@StdId


end
```

DESCRIPTION

This stored procedure updates an entire record that is stored in the userDetails database. It takes all values that are stored in the database as parameters and updates the record where the user's ID is equal to the current account ID.

UPDATEUSERNAME

```
CREATE PROCEDURE [dbo].[UpdateUsername]

(

@userName nvarchar(25)
```

```
)
as

begin

update userDetails
set userName=@userName


end
```

DESCRIPTION

This stored procedure updates the username stored in the userDetails database.

Solution Code

SIGNINPAGE.CS

```
using System;
using System.Data;
using System.Data.Common;
using System.Data.SqlClient;
using System.Windows.Forms;
using static System.Windows.Forms.VisualStyles.VisualStyleElement;

namespace Task2_Code_LL_000013680_Baines_C
{
   public partial class SignInPage : Form
   {
      public SignInPage()
      {
         InitializeComponent();
      }

      private void textBox1_TextChanged(object sender, EventArgs e)
      {

      }

      private void textBox2_TextChanged(object sender, EventArgs e)
      {

      }

      private void textBox6_TextChanged(object sender, EventArgs e)
      {

      }
```

```
private void Form1_Load(object sender, EventArgs e)
{

}

private void textBox9_TextChanged(object sender, EventArgs e)
{

}

private void textBox11_TextChanged(object sender, EventArgs e)
{

}

private void txtGoToSignUp_MouseClick(object sender, MouseEventArgs e)
{

   //Hides the form, loads and displays the form associated with the link that was clicked on.
   Hide();
   SignUpPage signUp = new SignUpPage();
   signUp.ShowDialog();
}

private void textBox4_TextChanged(object sender, EventArgs e)
{

}

private void btnNoSignIn_Click(object sender, EventArgs e)
{
   Hide();
   Home home = new Home();
   home.ShowDialog();

}

private void forgotPassword_Click(object sender, EventArgs e)
{
   Hide();
   ForgotPasswordPage forgotPassword = new ForgotPasswordPage();
   forgotPassword.ShowDialog();
}

private void txtSignUp_TextChanged(object sender, EventArgs e)
{

}

private void txtSignUp_MouseClick(object sender, MouseEventArgs e)
{
   navbarPanel.Left = txtSignUp.Left;
   Hide();
   SignUpPage signUp = new SignUpPage();
   signUp.ShowDialog();
```

```
    }

    private void textBox3_TextChanged(object sender, EventArgs e)
    {

    }

    private void textBox5_TextChanged(object sender, EventArgs e)
    {

    }

    private void textBox7_TextChanged(object sender, EventArgs e)
    {

    }

    private void textBox8_TextChanged(object sender, EventArgs e)
    {

    }

    private void settingsIcon_Click(object sender, EventArgs e)
    {
        Hide();
        SettingsPage settingsPage = new SettingsPage();
        settingsPage.ShowDialog();
    }

    private void btnSignIn_Click(object sender, EventArgs e)
    {
        //Gets the user inputted username and password from the form.
        string username = txtEnterUsername.Text;
        string password = txtEnterPassword.Text;

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        //Takes the connection string as a parameter for the SqlConnection class.
        SqlConnection sqlConnection = new SqlConnection(connectionString);



        //Create command that selects all usernames and passwords that are equal to user input.
        SqlCommand cmd = new SqlCommand("select * from userDetails where username like @userName and
password = @password;");
```

```
    cmd.Connection = sqlConnection;
    cmd.Parameters.AddWithValue("@userName", username);
    cmd.Parameters.AddWithValue("@password", password);

    sqlConnection.Open();


    DataSet dataset = new DataSet();
    SqlDataAdapter dataadapter = new SqlDataAdapter(cmd);
    dataadapter.Fill(dataset);

    sqlConnection.Close();


    /*cmd.ExecuteNonQuery();

    sqlConnection.Close(); */

    bool loginSuccessful = ((dataset.Tables.Count > 0) && (dataset.Tables[0].Rows.Count > 0));

    //Checks to see if the username and/or password fields are empty.
    if (username == string.Empty || password == string.Empty)
    {
        MessageBox.Show("Invalid username or password");
    }

    if (loginSuccessful)
    {
        MessageBox.Show("You have successfully signed in.");
        Hide();
        Home home = new Home();
        home.ShowDialog();
    }

    else
    {
        MessageBox.Show("Invalid username or password");
    }


}

private void txtGoToSignUp_MouseHover(object sender, EventArgs e)
{
    txtGoToSignUp.Text = txtGoToSignUp.Text.ToUpper();
}

private void txtGoToSignUp_MouseLeave(object sender, EventArgs e)
{
    txtGoToSignUp.Text = "Create an account";
}

private void forgotPassword_MouseHover(object sender, EventArgs e)
{
    txtforgotPassword.Text = txtforgotPassword.Text.ToUpper();
}
```

```
    private void txtforgotPassword_MouseLeave(object sender, EventArgs e)
    {
        txtforgotPassword.Text = "Forgot Password -->";
    }

    private void txtSignIn_MouseHover(object sender, EventArgs e)
    {
        txtSignIn.Text = txtSignIn.Text.ToUpper();
    }

    private void txtSignIn_MouseLeave(object sender, EventArgs e)
    {
        txtSignIn.Text = "Sign In";
    }

    private void txtSignUp_MouseHover(object sender, EventArgs e)
    {
        txtSignUp.Text = txtSignUp.Text.ToUpper();
    }

    private void txtSignUp_MouseLeave(object sender, EventArgs e)
    {
        txtSignUp.Text = "Sign Up";
    }
  }
}
```

DESCRIPTION OF CODE

This is my code for the sign-in page of my application. My code ensures there is a functioning navbar on the page as it redirects the user to the specific page of the application that the text and icons represent. My code also ensures the page is navigable by using button click events to allow the user to continue to the home page without sign in, sign up page, forgot password or settings page. The code also implements animations such as a moving navbar panel, text turning to uppercase on hover etc. The main part of my code gets the user-inputted username and password from the sign-in form and checks the inputs against those that are stored in the user Details database. If the username and password entered by the user is found in the database, it redirects the user to the home page. The code features input validation to check if the user has made their inputs.

CHANGES DURING DEVELOPMENT

Version 1.01

In Version 1.01, hashing was implemented to password creation. Hashing was added to the sign-in page by hashing the user-inputted password and comparing the hash key with the one stored in the userDetails database. This was added so that the validation does not completely break the functionality of the sign in page as it would be comparing unhashed passwords to hashed

passwords, meaning that it would never allow the user to sign in even if they had entered a valid username and password.

```
string hashedPassword = BCrypt.Net.BCrypt.HashPassword(txtEnterPassword.Text);
```

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the sign-in page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public SignInPage(AccessibilityHelper accessibilityHelper)
    {
       InitializeComponent();
       this.accessibilityHelper = accessibilityHelper;
       accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Also in Version 1.03, the continue without sign in button was removed. This was so that user access level control did not need to be implemented into the code. This made it so that the user now must sign in to gain access to the application.

Version 1.04

SIGNUPPAGE.CS

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class SignUpPage : Form
  {
    public SignUpPage()
    {
       InitializeComponent();
    }

    private void txtSignIn_TextChanged(object sender, EventArgs e)
    {
```

```
}

private void txtSignIn_MouseClick(object sender, MouseEventArgs e)
{
   //Hides the form, loads and displays the form associated with the link that was clicked on.
   navbarPanel.Left = txtSignIn.Left;
   Hide();
   SignInPage signIn = new SignInPage();
   signIn.ShowDialog();
}

private void txtBacktoSignIn_TextChanged(object sender, EventArgs e)
{

}

private void txtBacktoSignIn_MouseClick(object sender, MouseEventArgs e)
{
   //Moves the navbar underline panel under the sign up navbar link text when it has been clicked on.
   //Hides the sign in form and opens the sign up page.
   navbarPanel.Left = txtSignIn.Left;
   Hide();
   SignInPage signIn = new SignInPage();
   signIn.ShowDialog();
}

private void settingsIcon_Click(object sender, EventArgs e)
{
   //Opens the settings page when the settings icon is clicked on.
   Hide();
   SettingsPage settingsPage = new SettingsPage();
   settingsPage.ShowDialog();
}

private void SignUpPage_Load(object sender, EventArgs e)
{

}

private void txtBacktoSignIn_MouseHover(object sender, EventArgs e)
{
   //Capitalises the link text upon mouse hover.
   txtBacktoSignIn.Text = txtBacktoSignIn.Text.ToUpper();

}

private void txtBacktoSignIn_MouseLeave(object sender, EventArgs e)
{
   //Reverts the text back to its original state once the mouse cursor leaves the link text.
   txtBacktoSignIn.Text = "Back to Sign In";
}

private void btnSignUp_Click(object sender, EventArgs e)
{
```

```
        //Takes the user-inputted username and password from the form.
        string registeredUsername = txtCreateUsername.Text;
        string registeredPassword = txtCreatePassword.Text;

        //Tells the code where the database is.

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("AddRecord", sqlConnection);

        sqlConnection.Open();



        /*cmd.ExecuteNonQuery();

        sqlConnection.Close();

        */


        //Checks to see if the user-inputted username or password are empty. Returns error message if so.
        if (registeredUsername == string.Empty || registeredPassword == string.Empty )
        {
            MessageBox.Show("Please enter a valid username and password");
        }

        //Length checks for username and password requirements within database.
        //Also to ensure inputted usernames and passwords are not too small and are not too large.
        if (registeredUsername.Length < 5 || registeredPassword.Length < 5)
        {
            MessageBox.Show("Please enter a username and password that are at least five letters long");
        }

        if (registeredUsername.Length > 25)
        {
            MessageBox.Show("Please enter a smaller username");
        }

        if (registeredPassword.Length > 50)
        {
```

```
            MessageBox.Show("Please enter a smaller password");
        }

        else
        {
            //Run stored procedure and save the user-inputted username and password to the userDetails
database.

            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.AddWithValue("@userName", registeredUsername);

            cmd.Parameters.AddWithValue ("@password", registeredPassword);

            cmd.Parameters.AddWithValue("@highContrast", 0);

            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully created an account. Please sign in with your new account.");

            sqlConnection.Close();


            //Hide the sign up page and redirect the user to the sign in page to sign in with their new account.
            Hide();
            SignInPage signInPage = new SignInPage();
            signInPage.ShowDialog();

        }

        sqlConnection.Close();

    }

    private void txtSignIn_MouseHover(object sender, EventArgs e)
    {
        txtSignIn.Text = txtSignIn.Text.ToUpper();
    }

    private void txtSignIn_MouseLeave(object sender, EventArgs e)
    {
        txtSignIn.Text = "Sign In";
    }

    private void txtSignUp_MouseHover(object sender, EventArgs e)
    {
        txtSignUp.Text = txtSignUp.Text.ToUpper();
    }

    private void txtSignUp_MouseLeave(object sender, EventArgs e)
    {
        txtSignUp.Text = "Sign Up";
    }
```

```
    private void btnContinueWithoutAccount_Click(object sender, EventArgs e)
    {
        Hide();
        Home home = new Home();
        home.ShowDialog();
    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the sign-up page of my application. It allows the user to sign up for an account by entering a username and password. The user-inputted username and password is length checked by the code to ensure that it is not too short or too long for storage requirements in the userDetails database. The code also performs presence checks to ensure that the user has made an input to register both a username and password. The main part of the code executes a stored procedure which adds the registered username and password (if valid) to the userDetails database and it navigates the user to the sign in page to log in with their new account.

CHANGES DURING DEVELOPMENT

Version 1.01

In Version 1.01, I added a hashing method using the BCRYPT.net module, for the registered passwords for storage in the userDetails database. This is done for security purposes.

```
string hashedRegisteredPassword = BCrypt.Net.BCrypt.HashPassword(txtCreatePassword.Text);
```

	Id	userName	password	highContrast
▷	7	admin123	$2a$10$WZzLw...	False

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the sign-up page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public SignUpPage(AccessibilityHelper accessibilityHelper)
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Also, in Version 1.03, the continue without account creation was removed so that user access levels were not required to be implemented. This made it so that the user must create an account in order to sign in and gain access to the application.

HOME.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
   public partial class Home : Form
   {
      public Home()
      {
         InitializeComponent();
      }

      private void label3_Click(object sender, EventArgs e)
      {

      }

      private void Home_Load(object sender, EventArgs e)
      {

      }

      private void label9_Click(object sender, EventArgs e)
      {

      }

      private void settingsIcon_Click(object sender, EventArgs e)
      {
         Hide();
         SettingsPage settingsPage = new SettingsPage();
         settingsPage.ShowDialog();
      }

      private void infoIcon_Click(object sender, EventArgs e)
      {
         Hide();
         ZooInfo zooInfo = new ZooInfo();
         zooInfo.ShowDialog();
      }

      private void calendarIcon_Click(object sender, EventArgs e)
      {
```

```
    //Check to see if user is signed in before allowing the user to go to the booking page.
    //

    Hide();
    BookingTypePage bookingTypePage = new BookingTypePage();
    bookingTypePage.ShowDialog();
}

private void bookIcon_Click(object sender, EventArgs e)
{
    Hide();
    AnimalFactsPage animalFactsPage = new AnimalFactsPage();
    animalFactsPage.ShowDialog();
}

private void lblBookZoo1_MouseHover(object sender, EventArgs e)
{
    lblBookZoo1.Text = lblBookZoo1.Text.ToUpper();
}

private void btnReadMore_Click(object sender, EventArgs e)
{
    Hide();
    ZooInfo zooInfo = new ZooInfo();
    zooInfo.ShowDialog();
}

private void btnFunFacts_Click(object sender, EventArgs e)
{
    Hide();
    AnimalFactsPage animalFactsPage = new AnimalFactsPage();
    animalFactsPage.ShowDialog();
}

private void label2_MouseClick(object sender, MouseEventArgs e)
{
    Hide();
    ZooInfo zooInfo = new ZooInfo();
    zooInfo.ShowDialog();
}

private void label2_MouseHover(object sender, EventArgs e)
{
    label2.Text = label2.Text.ToUpper();
}

private void label2_MouseLeave(object sender, EventArgs e)
{
    label2.Text = "ZooLand";
}

private void lblBookZoo1_MouseLeave(object sender, EventArgs e)
{
    lblBookZoo1.Text = "Book";
```

```
}

private void lblBookZoo2_MouseClick(object sender, MouseEventArgs e)
{

}

private void lblBookZoo2_MouseHover(object sender, EventArgs e)
{
   lblBookZoo2.Text = lblBookZoo2.Text.ToUpper();
}

private void lblBookZoo2_MouseLeave(object sender, EventArgs e)
{
   lblBookZoo2.Text = "the Zoo";
}

private void lblBookHotel1_MouseHover(object sender, EventArgs e)
{
   lblBookHotel1.Text = lblBookHotel1.Text.ToUpper();
}

private void lblBookHotel1_MouseLeave(object sender, EventArgs e)
{
   lblBookHotel1.Text = "Book a";
}

private void lblBookZoo1_Click(object sender, EventArgs e)
{

   //Check to see if user is signed in before allowing the user to go to the booking page.
   //

   Hide();
   txtZooBookingPage zooBookingPage = new txtZooBookingPage();
   zooBookingPage.ShowDialog();

}

private void lblBookZoo2_Click(object sender, EventArgs e)
{

   //Check to see if user is signed in before allowing the user to go to the booking page.
   //

   Hide();
   txtZooBookingPage zooBookingPage = new txtZooBookingPage();
   zooBookingPage.ShowDialog();
}

private void lblBookHotel1_Click(object sender, EventArgs e)
{

   //Check to see if user is signed in before allowing the user to go to the booking page.
```

```
        //

        Hide();
        HotelBookingPage hotelBookingPage = new HotelBookingPage();
        hotelBookingPage.ShowDialog();
    }

    private void lblBookHotel2_Click(object sender, EventArgs e)
    {

        //Check to see if user is signed in before allowing the user to go to the booking page.
        //

        Hide();
        HotelBookingPage hotelBookingPage = new HotelBookingPage();
        hotelBookingPage.ShowDialog();
    }

    private void lblBookHotel2_MouseHover(object sender, EventArgs e)
    {
        lblBookHotel2.Text = lblBookHotel2.Text.ToUpper();
    }

    private void lblBookHotel2_MouseLeave(object sender, EventArgs e)
    {
        lblBookHotel2.Text = "Hotel Stay";
    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the home page of the application. It acts as the main landing page of my app, and it is filled with many internal links which allows the user to navigate to other pages on the application. The navbar on the home page makes use of mouse position events as on mouse hover, the text becomes capitalized. I used this simply as a quality-of-life feature.

CHANGES DURING DEVELOPMENT

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the sign-in page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public Home()
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the home page.

```
private void Home_Load(object sender, EventArgs e)
    {


        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
          if (highContrastReturned[0].ToString() == "0")
          {
            return;
          }
          else
          {
            //enable high contrast
            BackgroundImage = null;
            this.BackColor = Color.FromArgb(255,234,0);
          }
        }

        sqlConnection.Close();
    }
```

ANIMALFACTSPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class AnimalFactsPage : Form
  {
    public AnimalFactsPage()
    {
      InitializeComponent();
    }

    private void btnReadMoreLions_Click(object sender, EventArgs e)
    {
      Hide();
      LionFactsPage lionFactsPage = new LionFactsPage();
      lionFactsPage.ShowDialog();
    }

    private void btnReadMoreBears_Click(object sender, EventArgs e)
    {
      Hide();
      BearFactsPage bearsFactsPage = new BearFactsPage();
      bearsFactsPage.ShowDialog();
    }

    private void btnReadMoreGiraffes_Click(object sender, EventArgs e)
    {
      Hide();
      txtGiraffeFactsPage giraffeFactsPage = new txtGiraffeFactsPage();
      giraffeFactsPage.ShowDialog();
    }

    private void btnReadMoreMonkeys_Click(object sender, EventArgs e)
    {
      Hide();
      MonkeyFactsPage monkeyFactsPage = new MonkeyFactsPage();
      monkeyFactsPage.ShowDialog();
    }

    private void txtAboutUs_MouseHover(object sender, EventArgs e)
    {
      txtAboutUs.Text = txtAboutUs.Text.ToUpper();

    }
```

```
private void txtAboutUs_MouseLeave(object sender, EventArgs e)
{
   txtAboutUs.Text = "About Us";
}

private void txtAboutUs_MouseClick(object sender, MouseEventArgs e)
{
   navbarPanel.Left = txtAboutUs.Left;
   Hide();
   ZooInfo zooInfo = new ZooInfo();
   zooInfo.ShowDialog();
}

private void txtBookingTypePage_MouseHover(object sender, EventArgs e)
{
   txtBookingTypePage.Text = txtBookingTypePage.Text.ToUpper();
}

private void txtBookingTypePage_MouseLeave(object sender, EventArgs e)
{
   txtBookingTypePage.Text = "Booking";
}

private void txtBookingTypePage_Click(object sender, EventArgs e)
{
   navbarPanel.Left = txtBookingTypePage.Left;
   Hide();
   BookingTypePage bookingTypePage = new BookingTypePage();
   bookingTypePage.ShowDialog();
}

private void txtAnimalFacts_MouseHover(object sender, EventArgs e)
{
   txtAnimalFacts.Text = txtAnimalFacts.Text.ToUpper();
}

private void txtAnimalFacts_MouseLeave(object sender, EventArgs e)
{
   txtAnimalFacts.Text = "Animal Facts";
}

private void rzaNavLogo_Click(object sender, EventArgs e)
{
   Hide();
   Home home = new Home();
   home.ShowDialog();
}

private void settingsIcon_Click(object sender, EventArgs e)
{
   Hide();
   SettingsPage settingsPage = new SettingsPage();
   settingsPage.ShowDialog();
}
```

```
    private void AnimalFactsPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is my code for the animal facts page of my application. It displays widgets on the form which allows the user to navigate to facts pages about specific animals. It also contains the same navbar which is used across all my application pages with text capitalization on mouse hover.

CHANGES DURING DEVELOPMENT

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the animal facts page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public AnimalFactsPage(AccessibilityHelper accessibilityHelper)
    {
      InitializeComponent();
      this.accessibilityHelper = accessibilityHelper;
      accessibilityHelper.UpdateFontSize(this.Controls);
    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the home page.

```
private void AnimalFactsPage_Load(object sender, EventArgs e)
    {
      //DEFAULT PATH
      string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
      //

      //BACKUP PATH
      //
      //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";
```

```
        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
          if (highContrastReturned[0].ToString() == "0")
          {
            return;
          }
          else
          {
            //enable high contrast
            BackgroundImage = null;
            this.BackColor = Color.FromArgb(255,234,0);
          }
        }

        sqlConnection.Close();
      }

    }
```

ZOOINFO.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using Task2_Code_LL_000013680_Baines_C.Properties;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class ZooInfo : Form
  {
    public ZooInfo()
    {
      InitializeComponent();
    }

    private void rzaNavLogo_Click(object sender, EventArgs e)
    {
```

```
    Hide();
    Home home = new Home();
    home.ShowDialog();
}

private void settingsIcon_Click(object sender, EventArgs e)
{
    Hide();
    SettingsPage settingsPage = new SettingsPage();
    settingsPage.ShowDialog();
}

private void txtBookingTypePage_TextChanged(object sender, EventArgs e)
{

}

private void txtBookingTypePage_MouseClick(object sender, MouseEventArgs e)
{
    navbarPanel.Left = txtBookingTypePage.Left;
    Hide();
    BookingTypePage bookingTypePage = new BookingTypePage();
    bookingTypePage.ShowDialog();
}

private void txtAboutUs_MouseHover(object sender, EventArgs e)
{
    txtAboutUs.Text = txtAboutUs.Text.ToUpper();
}

private void txtAboutUs_MouseLeave(object sender, EventArgs e)
{
    txtAboutUs.Text = "About Us";
}

private void txtBookingTypePage_MouseHover(object sender, EventArgs e)
{
    txtBookingTypePage.Text = txtBookingTypePage.Text.ToUpper();
}

private void txtBookingTypePage_MouseLeave(object sender, EventArgs e)
{
    txtBookingTypePage.Text = "Booking";
}

private void textBox8_MouseHover(object sender, EventArgs e)
{
    textBox8.Text = textBox8.Text.ToUpper();
}

private void textBox8_MouseLeave(object sender, EventArgs e)
{
    textBox8.Text = "Hotel";
}
```

```
private void textBox2_MouseHover(object sender, EventArgs e)
{
   textBox2.Text = textBox2.Text.ToUpper();
}

private void textBox2_MouseLeave(object sender, EventArgs e)
{
   textBox2.Text = "Toilets";
}

private void textBox3_MouseHover(object sender, EventArgs e)
{
   textBox3.Text = textBox3.Text.ToUpper();
}

private void textBox3_MouseLeave(object sender, EventArgs e)
{
   textBox3.Text = "Toilets";
}

private void textBox6_MouseHover(object sender, EventArgs e)
{
   textBox6.Text = textBox6.Text.ToUpper();
}

private void textBox6_MouseLeave(object sender, EventArgs e)
{
   textBox6.Text = "Monkeys";
}

private void textBox4_MouseHover(object sender, EventArgs e)
{
   textBox4.Text = textBox4.Text.ToUpper();
}

private void textBox4_MouseLeave(object sender, EventArgs e)
{
   textBox4.Text = "Bears";
}

private void textBox5_MouseHover(object sender, EventArgs e)
{
   textBox5.Text = textBox5.Text.ToUpper();
}

private void textBox5_MouseLeave(object sender, EventArgs e)
{
   textBox5.Text = "Lions";
}

private void textBox7_MouseHover(object sender, EventArgs e)
{
   textBox7.Text = textBox7.Text.ToUpper();
```

```
    }

    private void textBox7_MouseLeave(object sender, EventArgs e)
    {
      textBox7.Text = "Giraffes";
    }

    private void textBox9_MouseHover(object sender, EventArgs e)
    {
      textBox9.Text = textBox9.Text.ToUpper();
    }

    private void textBox9_MouseLeave(object sender, EventArgs e)
    {
      textBox9.Text = "Event Center";
    }

    private void textBox6_Click(object sender, EventArgs e)
    {
      Hide();
      MonkeyFactsPage monkeyFactsPage = new MonkeyFactsPage();
      monkeyFactsPage.ShowDialog();
    }

    private void textBox4_Click(object sender, EventArgs e)
    {
      Hide();
      BearFactsPage bearFactsPage = new BearFactsPage();
      bearFactsPage.ShowDialog();
    }

    private void textBox5_Click(object sender, EventArgs e)
    {
      Hide();
      LionFactsPage lionFactsPage = new LionFactsPage();
      lionFactsPage.ShowDialog();
    }

    private void textBox7_Click(object sender, EventArgs e)
    {
      Hide();
      txtGiraffeFactsPage giraffeFactsPage = new txtGiraffeFactsPage();
      giraffeFactsPage.ShowDialog();
    }

    private void ZooInfo_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the zoo information page of my application. It displays the opening and closing times of the zoo on the form and a map which can be clicked on to provide facts about the host of animals that RZA has. It also has a link to the hotel booking page on the page for user convenience.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the zoo information page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public ZooInfo()
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);
    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the zoo information page.

```
private void ZooInfo_Load(object sender, EventArgs e)
    {
        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
```

```
        if (highContrastReturned.Read())
        {
            if (highContrastReturned[0].ToString() == "0")
            {
                return;
            }
            else
            {
                //enable high contrast
                BackgroundImage = null;
                this.BackColor = Color.FromArgb(255,234,0);
            }
        }

        sqlConnection.Close();
    }
}
```

BEARFACTSPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
    public partial class BearFactsPage : Form
    {
        public BearFactsPage()
        {
            InitializeComponent();
        }

        private void rzaNavLogo_Click(object sender, EventArgs e)
        {
            Hide();
            Home home = new Home();
            home.ShowDialog();
        }

        private void settingsIcon_Click(object sender, EventArgs e)
        {
            Hide();
            SettingsPage settingsPage = new SettingsPage();
            settingsPage.ShowDialog();
        }
```

```
    private void backToFactsPage_Click(object sender, EventArgs e)
    {
      Hide();
      AnimalFactsPage animalFactsPage = new AnimalFactsPage();
      animalFactsPage.ShowDialog();
    }

    private void txtBearFactsPage_MouseHover(object sender, EventArgs e)
    {
      txtBearFactsPage.Text = txtBearFactsPage.Text.ToUpper();
    }

    private void txtBearFactsPage_MouseLeave(object sender, EventArgs e)
    {
      txtBearFactsPage.Text = "Bear Facts";
    }

    private void txtMoreFacts_MouseHover(object sender, EventArgs e)
    {
      txtMoreFacts.Text = txtMoreFacts.Text.ToUpper();
    }

    private void txtMoreFacts_MouseLeave(object sender, EventArgs e)
    {
      txtMoreFacts.Text = "More Facts";
    }

    private void txtMoreFacts_Click(object sender, EventArgs e)
    {
      navbarPanel.Left = txtMoreFacts.Left;
      Hide();
      AnimalFactsPage animalFactsPage = new AnimalFactsPage();
      animalFactsPage.ShowDialog();
    }

    private void BearFactsPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the bear facts page of my application. It displays text about bears on the form and ensures that the page is navigable as there is internal links on the navbar that redirects the user to the specific page that the navbar links correspond to.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the bear facts page.

```
public BearFactsPage(AccessibilityHelper accessibilityHelper)
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the bear facts page.

```
private void BearFactsPage_Load(object sender, EventArgs e)
    {
        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
            if (highContrastReturned[0].ToString() == "0")
            {
                return;
            }
            else
            {
                //enable high contrast
                BackgroundImage = null;
                this.BackColor = Color.FromArgb(255,234,0);
            }
        }

        sqlConnection.Close();
```

```
    }
  }
```

GIRAFFEFACTS.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class txtGiraffeFactsPage : Form
  {
    public txtGiraffeFactsPage()
    {
      InitializeComponent();
    }

    private void rzaNavLogo_Click(object sender, EventArgs e)
    {
      Hide();
      Home home = new Home();
      home.ShowDialog();
    }

    private void backToFactsPage_Click(object sender, EventArgs e)
    {
      Hide();
      AnimalFactsPage animalFactsPage = new AnimalFactsPage();
      animalFactsPage.ShowDialog();
    }

    private void settingsIcon_Click(object sender, EventArgs e)
    {
      Hide();
      SettingsPage settingsPage = new SettingsPage();
      settingsPage.ShowDialog();
    }

    private void txtGiraffeFacts_MouseHover(object sender, EventArgs e)
    {
      txtGiraffeFacts.Text = txtGiraffeFacts.Text.ToUpper();
    }

    private void txtGiraffeFacts_MouseLeave(object sender, EventArgs e)
    {
```

```
    txtGiraffeFacts.Text = "Giraffe Facts";
}

private void txtMoreFacts_MouseHover(object sender, EventArgs e)
{
    txtMoreFacts.Text = txtMoreFacts.Text.ToUpper();
}

private void txtMoreFacts_MouseLeave(object sender, EventArgs e)
{
    txtMoreFacts.Text = "More Facts";
}

private void txtMoreFacts_Click(object sender, EventArgs e)
{
    navbarPanel.Left = txtMoreFacts.Left;
    Hide();
    AnimalFactsPage animalFactsPage = new AnimalFactsPage();
    animalFactsPage.ShowDialog();
}

private void txtGiraffeFactsPage_Load(object sender, EventArgs e)
{
```

DESCRIPTION OF CODE

This is the code for the giraffe facts page of my application. It displays text about giraffes on the form and ensures that the page is navigable as there is internal links on the navbar that redirects the user to the specific page that the navbar links correspond to.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the giraffe facts page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public txtGiraffeFactsPage(AccessibilityHelper accessibilityHelper)
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the giraffe facts page.

```
//DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
          if (highContrastReturned[0].ToString() == "0")
          {
            return;
          }
          else
          {
            //enable high contrast
            BackgroundImage = null;
            this.BackColor = Color.FromArgb(255,234,0);
          }
        }

        sqlConnection.Close();
    }

  }
```

LIONFACTSPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
```

```
{
    public partial class LionFactsPage : Form
    {
        public LionFactsPage()
        {
            InitializeComponent();
        }

        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void rzaNavLogo_Click(object sender, EventArgs e)
        {
            Hide();
            Home home = new Home();
            home.ShowDialog();
        }

        private void backToFactsPage_Click(object sender, EventArgs e)
        {
            Hide();
            AnimalFactsPage animalFactsPage = new AnimalFactsPage();
            animalFactsPage.ShowDialog();
        }

        private void settingsIcon_Click(object sender, EventArgs e)
        {
            Hide();
            SettingsPage settingsPage = new SettingsPage();
            settingsPage.ShowDialog();
        }

        private void txtLionFacts_MouseHover(object sender, EventArgs e)
        {
            txtLionFacts.Text = txtLionFacts.Text.ToUpper();
        }

        private void txtLionFacts_MouseLeave(object sender, EventArgs e)
        {
            txtLionFacts.Text = "Lion Facts";
        }

        private void txtMoreFacts_MouseHover(object sender, EventArgs e)
        {
            txtMoreFacts.Text = txtMoreFacts.Text.ToUpper();
        }

        private void txtMoreFacts_MouseLeave(object sender, EventArgs e)
        {
            txtMoreFacts.Text = "More Facts";
        }
```

```
    private void txtMoreFacts_Click(object sender, EventArgs e)
    {
        navbarPanel.Left = txtMoreFacts.Left;
        Hide();
        AnimalFactsPage animalFactsPage = new AnimalFactsPage();
        animalFactsPage.ShowDialog();
    }

    private void LionFactsPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the lion facts page of my application. It displays text about lions on the form and ensures that the page is navigable as there is internal links on the navbar that redirects the user to the specific page that the navbar links correspond to.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the lion facts page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public LionFactsPage()
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the lion facts page.

```
private void LionFactsPage_Load(object sender, EventArgs e)
    {
        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
```

```
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
          if (highContrastReturned[0].ToString() == "0")
          {
            return;
          }
          else
          {
            //enable high contrast
            BackgroundImage = null;
            this.BackColor = Color.FromArgb(255,234,0);
          }
        }

        sqlConnection.Close();
      }

    }
```

MONKEYFACTSPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class MonkeyFactsPage : Form
  {
    public MonkeyFactsPage()
    {
      InitializeComponent();
```

```
}

private void MonkeyFactsPage_Load(object sender, EventArgs e)
{

}

private void rzaNavLogo_Click(object sender, EventArgs e)
{
   Hide();
   Home home = new Home();
   home.ShowDialog();
}

private void settingsIcon_Click(object sender, EventArgs e)
{
   Hide();
   SettingsPage settingsPage = new SettingsPage();
   settingsPage.ShowDialog();
}

private void txtMonkeyFacts_MouseHover(object sender, EventArgs e)
{
   txtMonkeyFacts.Text = txtMonkeyFacts.Text.ToUpper();
}

private void txtMonkeyFacts_MouseLeave(object sender, EventArgs e)
{
   txtMonkeyFacts.Text = "Monkey Facts";
}

private void txtMoreFacts_MouseHover(object sender, EventArgs e)
{
   txtMoreFacts.Text = txtMoreFacts.Text.ToUpper();
}

private void txtMoreFacts_MouseLeave(object sender, EventArgs e)
{
   txtMoreFacts.Text = "More Facts";
}

private void txtMoreFacts_Click(object sender, EventArgs e)
{
   navbarPanel.Left = txtMoreFacts.Left;
   Hide();
   AnimalFactsPage animalFactsPage = new AnimalFactsPage();
   animalFactsPage.ShowDialog();
}

private void backToFactsPage_Click(object sender, EventArgs e)
{
   Hide();
   AnimalFactsPage animalFactsPage = new AnimalFactsPage();
   animalFactsPage.ShowDialog();
```

```
    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the monkey facts page of my application. It displays text about monkeys on the form and ensures that the page is navigable as there is internal links on the navbar that redirects the user to the specific page that the navbar links correspond to.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the monkey facts page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public MonkeyFactsPage(AccessibilityHelper accessibilityHelper)
    {
      InitializeComponent();
      this.accessibilityHelper = accessibilityHelper;
      accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the monkey facts page.

```
private void MonkeyFactsPage_Load(object sender, EventArgs e)
    {
      //DEFAULT PATH
      string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
      //

      //BACKUP PATH
      //
      //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

      SqlConnection sqlConnection = new SqlConnection(connectionString);

      SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
      sqlCommand.CommandType = CommandType.StoredProcedure;

      sqlCommand.Parameters.AddWithValue("@highContrast", 0);

      sqlConnection.Open();
```

```
            SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
            if (highContrastReturned.Read())
            {
                if (highContrastReturned[0].ToString() == "0")
                {
                    return;
                }
                else
                {
                    //enable high contrast
                    BackgroundImage = null;
                    this.BackColor = Color.FromArgb(255,234,0);
                }
            }

            sqlConnection.Close();
        }
```

BOOKINGTYPEPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
    public partial class BookingTypePage : Form
    {
        public BookingTypePage()
        {
            InitializeComponent();
        }

        private void settingsIcon_Click(object sender, EventArgs e)
        {
            Hide();
            SettingsPage settingsPage = new SettingsPage();
            settingsPage.ShowDialog();
        }

        private void rzaNavLogo_Click(object sender, EventArgs e)
        {
            Hide();
            Home home = new Home();
            home.ShowDialog();
        }
```

```
private void txtAboutUs_MouseHover(object sender, EventArgs e)
{
   txtAboutUs.Text = txtAboutUs.Text.ToUpper();
}

private void BookingTypePage_MouseLeave(object sender, EventArgs e)
{
   txtAboutUs.Text = "About Us";
}

private void txtAboutUs_TextChanged(object sender, EventArgs e)
{

}

private void txtAboutUs_MouseClick(object sender, MouseEventArgs e)
{
   navbarPanel.Left = txtAboutUs.Left;
   Hide();
   ZooInfo zooInfo = new ZooInfo();
   zooInfo.ShowDialog();
}

private void txtAnimalFactsPage_MouseClick(object sender, MouseEventArgs e)
{
   navbarPanel.Left = txtAnimalFactsPage.Left;
   Hide();
   AnimalFactsPage animalFactsPage = new AnimalFactsPage();
   animalFactsPage.ShowDialog();
}

private void txtAboutUs_MouseLeave(object sender, EventArgs e)
{

}

private void btnZooBooking_Click(object sender, EventArgs e)
{
   Hide();
   txtZooBookingPage zooBookingPage = new txtZooBookingPage();
   zooBookingPage.ShowDialog();
}

private void txtAnimalFactsPage_TextChanged(object sender, EventArgs e)
{

}

private void btnHotelBooking_Click(object sender, EventArgs e)
{
   Hide();
   HotelBookingPage hotelBookingPage = new HotelBookingPage();
   hotelBookingPage.ShowDialog();
}
```

```
    private void backtoHomeIcon_Click(object sender, EventArgs e)
    {
        Hide();
        Home home = new Home();
        home.ShowDialog();
    }

    private void txtAnimalFactsPage_MouseHover(object sender, EventArgs e)
    {
        txtAnimalFactsPage.Text = txtAnimalFactsPage.Text.ToUpper();
    }

    private void txtAnimalFactsPage_MouseLeave(object sender, EventArgs e)
    {
        txtAnimalFactsPage.Text = "Animal Facts";
    }

    private void textBox1_MouseHover(object sender, EventArgs e)
    {
        txtBookingSelect.Text = txtBookingSelect.Text.ToUpper();
    }

    private void txtBookingSelect_MouseLeave(object sender, EventArgs e)
    {
        txtBookingSelect.Text = "Booking Select";
    }

    private void btnViewHotelRooms_Click(object sender, EventArgs e)
    {
        Hide();
        HotelRooms hotelRooms = new HotelRooms();
        hotelRooms.ShowDialog();
    }

    private void BookingTypePage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the booking selection page of my application. It simply allows the user to select whether they want to make a booking for the hotel or the zoo and redirects them to the appropriate page. As well as this, it allows the user to navigate to the hotel rooms page to view the rooms that RZA offer for their on-site hotel.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the booking selection page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public BookingTypePage(AccessibilityHelper accessibilityHelper)
    {
      InitializeComponent();
      this.accessibilityHelper = accessibilityHelper;

    }
```

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the monkey facts page.

```
private void BookingTypePage_Load(object sender, EventArgs e)
    {
      //DEFAULT PATH
      string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
      //

      //BACKUP PATH
      //
      //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

      SqlConnection sqlConnection = new SqlConnection(connectionString);

      SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
      sqlCommand.CommandType = CommandType.StoredProcedure;

      sqlCommand.Parameters.AddWithValue("@highContrast", 0);

      sqlConnection.Open();

      SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
      if (highContrastReturned.Read())
      {
        if (highContrastReturned[0].ToString() == "0")
        {
          return;
        }
        else
        {
          //enable high contrast
```

```
                BackgroundImage = null;
                this.BackColor = Color.FromArgb(255,234,0);
            }
        }

        sqlConnection.Close();
    }

}
```

HOTELROOMS.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
    public partial class HotelRooms : Form
    {
        public HotelRooms()
        {
            InitializeComponent();
        }

        private void rzaNavLogo_Click(object sender, EventArgs e)
        {
            Hide();
            Home home = new Home();
            home.ShowDialog();
        }

        private void settingsIcon_Click(object sender, EventArgs e)
        {
            Hide();
            SettingsPage settingsPage = new SettingsPage();
            settingsPage.ShowDialog();
        }

        private void lblGoToZooMap_Click(object sender, EventArgs e)
        {
            Hide();
            ZooInfo zooInfo = new ZooInfo();
            zooInfo.ShowDialog();
        }

        private void lblGoToZooMap_MouseHover(object sender, EventArgs e)
        {
```

```
    lblGoToZooMap.Text = lblGoToZooMap.Text.ToUpper();
}

private void lblGoToZooMap_MouseLeave(object sender, EventArgs e)
{
    lblGoToZooMap.Text = "ZooLand";
}

private void txtHotelRooms_MouseHover(object sender, EventArgs e)
{
    txtHotelRooms.Text = txtHotelRooms.Text.ToUpper();
}

private void txtHotelRooms_MouseLeave(object sender, EventArgs e)
{
    txtHotelRooms.Text = "Hotel Rooms";
}

private void txtAboutUsLink_MouseHover(object sender, EventArgs e)
{
    txtAboutUsLink.Text = txtAboutUsLink.Text.ToUpper();
}

private void txtAboutUsLink_MouseLeave(object sender, EventArgs e)
{
    txtAboutUsLink.Text = "About Us";
}

private void txtAboutUsLink_Click(object sender, EventArgs e)
{
    navbarPanel.Left = txtAboutUsLink.Left;
    Hide();
    ZooInfo zooInfo = new ZooInfo();
    zooInfo.ShowDialog();
}

private void btnBookNow_Click(object sender, EventArgs e)
{
    Hide();
    HotelBookingPage bookingPage = new HotelBookingPage();
    bookingPage.ShowDialog();
}

private void btnBookNow2_Click(object sender, EventArgs e)
{
    Hide();
    HotelBookingPage bookingPage = new HotelBookingPage();
    bookingPage.ShowDialog();
}

private void btnBookNow2_Click_1(object sender, EventArgs e)
{

}
```

```
    private void btnBookNow3_Click(object sender, EventArgs e)
    {
        Hide();
        HotelBookingPage bookingPage = new HotelBookingPage();
        bookingPage.ShowDialog();
    }

    private void HotelRooms_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the hotel rooms page of my application. It displays the room types that the RZA on-site hotel offers and has buttons to redirect the user to the hotel booking page.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the hotel rooms page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public HotelRooms()
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);
    }
```

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the hotel rooms page.

```
private void HotelRooms_Load(object sender, EventArgs e)
    {
        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
```

```
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand sqlCommand = new SqlCommand("GetHighContrast", sqlConnection);
        sqlCommand.CommandType = CommandType.StoredProcedure;

        sqlCommand.Parameters.AddWithValue("@highContrast", 0);

        sqlConnection.Open();

        SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
        if (highContrastReturned.Read())
        {
          if (highContrastReturned[0].ToString() == "0")
          {
            return;
          }
          else
          {
            //enable high contrast
            BackgroundImage = null;
            this.BackColor = Color.FromArgb(255,234,0);
          }
        }

        sqlConnection.Close();
      }

    }
```

ZOOBOOKINGPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class txtZooBookingPage : Form
  {
    public txtZooBookingPage()
    {
      InitializeComponent();
```

```
}

private void BackToBookingSelect_Click(object sender, EventArgs e)
{
   Hide();
   BookingTypePage bookingTypePage = new BookingTypePage();
   bookingTypePage.ShowDialog();
}

private void rzaNavLogo_Click(object sender, EventArgs e)
{
   Hide();
   Home home = new Home();
   home.ShowDialog();
}

private void settingsIcon_Click(object sender, EventArgs e)
{
   Hide();
   SettingsPage settingsPage = new SettingsPage();
   settingsPage.ShowDialog();
}

private void txtHotelBooking_MouseHover(object sender, EventArgs e)
{
   txtZooBooking.Text = txtZooBooking.Text.ToUpper();
}

private void txtHotelBooking_MouseLeave(object sender, EventArgs e)
{
   txtZooBooking.Text = "Zoo Booking";
}

private void txtHotelBooking_Click(object sender, EventArgs e)
{

}

private void txtAboutUsLink_MouseHover(object sender, EventArgs e)
{
   txtAboutUsLink.Text = txtAboutUsLink.Text.ToUpper();
}

private void txtAboutUsLink_MouseLeave(object sender, EventArgs e)
{
   txtAboutUsLink.Text = "About Us";
}

private void txtAboutUsLink_Click(object sender, EventArgs e)
{
   navbarPanel.Left = txtAboutUsLink.Left;
   Hide();
   ZooInfo zooInfo = new ZooInfo();
   zooInfo.ShowDialog();
```

```
    }

    private void visitDate_DateChanged(object sender, DateRangeEventArgs e)
    {
        lblVisitDate.Text = "Date of Visit: " + visitDate.SelectionEnd.ToString("dddd, d MMMM, yyyy");
    }

    private void btnConfirmBooking_Click(object sender, EventArgs e)
    {
        //Save zoo booking details to ZooBookingDetails database
        //Take visitDate, numOfChildTickets, numOfAdultTickets

        string dateOfVisit = visitDate.SelectionEnd.ToString("dddd, d MMMM, yyyy");
        int? numofChildTickets = Convert.ToInt32(txtNumOfChildTickets.Text);
        int? numofAdultTickets = Convert.ToInt32(txtNumOfAdultTickets.Text);

        //Tells the code where the database is.

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("AddZooRecord", sqlConnection);

        sqlConnection.Open();

        //Input validation to check if all user inputs have been entered.

        if (dateOfVisit == string.Empty || numofChildTickets == null || numofAdultTickets == null)
        {
            MessageBox.Show("Please provide all booking details");
        }

        if (numofChildTickets < 0)
        {
            MessageBox.Show("Please enter a valid number of child tickets");
        }

        if(numofAdultTickets < 0)
        {
            MessageBox.Show("Please enter a valid number of adult tickets");
        }
```

```
        else
        {
            //Run stored procedure and save zoo booking details to database.
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.AddWithValue("@visitDate", dateOfVisit);
            cmd.Parameters.AddWithValue("@numOfChildTickets", numofChildTickets);
            cmd.Parameters.AddWithValue("@numOfAdultTickets", numofAdultTickets);

            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully reserved ticket(s) for Riget Zoo Adventures. You can collect
your ticket(s) on-visit.");
            sqlConnection.Close();

            Hide();
            OrderSummaryPage orderSummaryPage = new OrderSummaryPage();
            orderSummaryPage.ShowDialog();

        }

        sqlConnection.Close();
    }

    private void txtZooBookingPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the zoo booking page of my application. It allows the user to book a ticket to the zoo for collection. It asks the user to confirm booking details such as a visit date and the number of child, adult tickets. These booking details are then added as a record in the zooBookingDetails database. There is input validation throughout the code that ensures that all booking details must be valid before a booking can be confirmed and written to the database. The page is navigable as it features the same navbar with internal links. If a successful booking is made, the code redirects the user to the order summary page for user convenience.

CHANGES DURING DEVELOPMENT

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the zoo booking page.

```
public AccessibilityHelper accessibilityHelper { get; set; }

    public txtZooBookingPage()
```

```
    {
      InitializeComponent();
      this.accessibilityHelper = accessibilityHelper;
      accessibilityHelper.UpdateFontSize(this.Controls);
    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the zoo booking page.

```
private void txtZooBookingPage_Load(object sender, EventArgs e)
        {
            //DEFAULT PATH
            string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDri
ve - Middlesbrough College\\Documents\\Carl Baines\\Task 2\\zoo application
database.mdf\";Integrated Security=True;Connect Timeout=30; Integrated
Security = True";
            //

            //BACKUP PATH
            //
            //string connectionString = "Data Source =
(LocalDB)\\MSSQLLocalDB; AttachDbFilename = \"C:\\Users\\EXAM1238\\OneDrive -
Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout =
30;";

            SqlConnection sqlConnection = new
SqlConnection(connectionString);

            SqlCommand sqlCommand = new SqlCommand("GetHighContrast",
sqlConnection);
            sqlCommand.CommandType = CommandType.StoredProcedure;

            sqlCommand.Parameters.AddWithValue("@highContrast", 0);

            sqlConnection.Open();

            SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
            if (highContrastReturned.Read())
            {
                if (highContrastReturned[0].ToString() == "0")
                {
                    return;
                }
                else
                {
                    //enable high contrast
                    BackgroundImage = null;
                    this.BackColor = Color.FromArgb(255,234,0);
                }
            }
```

```
            sqlConnection.Close();
        }
    }
```

HOTELBOOKINGPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Runtime.InteropServices.ComTypes;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
  public partial class HotelBookingPage : Form
  {
    public HotelBookingPage()
    {
      InitializeComponent();
    }

    private void BackToBookingSelect_Click(object sender, EventArgs e)
    {
      Hide();
      BookingTypePage bookingTypePage = new BookingTypePage();
      bookingTypePage.ShowDialog();
    }

    private void rzaNavLogo_Click(object sender, EventArgs e)
    {
      Hide();
      Home home = new Home();
      home.ShowDialog();
    }

    private void settingsIcon_Click(object sender, EventArgs e)
    {
      Hide();
      SettingsPage settingsPage = new SettingsPage();
      settingsPage.ShowDialog();
    }

    private void txtAboutUsLink_Click(object sender, EventArgs e)
```

```
    {
        navbarPanel.Left = txtAboutUsLink.Left;
        Hide();
        ZooInfo zooInfo = new ZooInfo();
        zooInfo.ShowDialog();
    }

    private void txtHotelBooking_MouseHover(object sender, EventArgs e)
    {
        txtHotelBooking.Text = txtHotelBooking.Text.ToUpper();
    }

    private void txtHotelBooking_MouseLeave(object sender, EventArgs e)
    {
        txtHotelBooking.Text = "Hotel Booking";
    }

    private void txtAboutUsLink_MouseHover(object sender, EventArgs e)
    {
        txtAboutUsLink.Text = txtAboutUsLink.Text.ToUpper();
    }

    private void txtAboutUsLink_MouseLeave(object sender, EventArgs e)
    {
        txtAboutUsLink.Text = "About Us";
    }

    private void btnConfirmBooking_Click(object sender, EventArgs e)
    {
        //Save booking details to HotelBookingDetails database.
        //Take Check-In Date, Check-Out Date, Number of Residents, Room Type from form

        string checkInDate = lengthOfStay.SelectionStart.ToString("dddd, d MMMM, yyyy");
        string checkOutDate = lengthOfStay.SelectionEnd.ToString("dddd, d MMMM, yyyy");

        //int? to check if it has a null value when input validating.
        int? numofResidents = Convert.ToInt32(txtNumOfResidents.Text);

        string roomType = selectRoomType.Text;

        //Tells the code where the database is.

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";
```

```
        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("AddHotelRecord", sqlConnection);

        sqlConnection.Open();




        //Input validation to check if all inputs have been entered/selected

        if (checkInDate == string.Empty || checkOutDate == string.Empty || numofResidents == null ||
roomType == string.Empty)
        {
            MessageBox.Show("Please provide all booking details");
        }

        if (numofResidents < 0)
        {
            MessageBox.Show("Please enter a valid number of residents");
        }

        else
        {
            //Run stored procedure and save hotel booking details to database.
            cmd.CommandType = CommandType.StoredProcedure;

            cmd.Parameters.AddWithValue("@checkInDate", checkInDate);
            cmd.Parameters.AddWithValue("@numOfResidents", numofResidents);
            cmd.Parameters.AddWithValue("@RoomType", roomType);
            cmd.Parameters.AddWithValue("@checkOutDate", checkOutDate);

            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully reserved a booking to RZA's On-Site Hotel!");

            sqlConnection.Close();

            Hide();
            OrderSummaryPage orderSummaryPage = new OrderSummaryPage();
            orderSummaryPage.ShowDialog();
        }

        sqlConnection.Close();


    }

    private void lengthOfStay_DateChanged(object sender, DateRangeEventArgs e)
    {

        //Changes the startDate label text to the check-in date.
```

```
        startDate.Text = "Check-In Date: " + lengthOfStay.SelectionStart.ToString("dddd, d MMMM, yyyy");
        //Changes the endDate label text to the check-out date.
        endDate.Text = "Check-Out Date: " + lengthOfStay.SelectionEnd.ToString("dddd, d MMMM, yyyy");
    }

    private void HotelBookingPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the hotel booking page of my application. It allows the user to make a reservation for RZA's on-site hotel. It requires the user to input a check-in date, check-out date, the number of residents and a room type. The booking details, if valid, are then added as a record to the HotelBookingDetails database. Input validation is implemented throughout the code to ensure that all booking details are entered and that they are valid before they are written to the database. The page is navigable as it features the application's navbar and it also navigates the user to the order summary page, if a successful booking is made, for the purpose of user convenience.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the hotel booking page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public HotelBookingPage(AccessibilityHelper accessibilityHelper)
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the hotel booking page.

```
//DEFAULT PATH
            string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDri
ve - Middlesbrough College\\Documents\\Carl Baines\\Task 2\\zoo application
database.mdf\";Integrated Security=True;Connect Timeout=30; Integrated
Security = True";
```

```
            //

            //BACKUP PATH
            //
            //string connectionString = "Data Source =
(LocalDB)\\MSSQLLocalDB; AttachDbFilename = \"C:\\Users\\EXAM1238\\OneDrive -
Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout =
30;";

            SqlConnection sqlConnection = new
SqlConnection(connectionString);

            SqlCommand sqlCommand = new SqlCommand("GetHighContrast",
sqlConnection);
            sqlCommand.CommandType = CommandType.StoredProcedure;

            sqlCommand.Parameters.AddWithValue("@highContrast", 0);

            sqlConnection.Open();

            SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
            if (highContrastReturned.Read())
            {
                if (highContrastReturned[0].ToString() == "0")
                {
                    return;
                }
                else
                {
                    //enable high contrast
                    BackgroundImage = null;
                    this.BackColor = Color.FromArgb(255,234,0);
                }
            }

            sqlConnection.Close();
        }

    }
```

ORDERSUMMARYPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Runtime.CompilerServices;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Task2_Code_LL_000013680_Baines_C
{
```

```
public partial class OrderSummaryPage : Form
{
    public OrderSummaryPage()
    {
        InitializeComponent();
    }

    private void rzaNavLogo_Click(object sender, EventArgs e)
    {
        Hide();
        Home home = new Home();
        home.ShowDialog();
    }

    private void txtOrderSummary_TextChanged(object sender, EventArgs e)
    {

    }

    private void txtOrderSummary_MouseHover(object sender, EventArgs e)
    {
        txtOrderSummary.Text = txtOrderSummary.Text.ToUpper();
    }

    private void txtOrderSummary_MouseLeave(object sender, EventArgs e)
    {
        txtOrderSummary.Text = "Order Summary";
    }

    private void txtAboutUs_MouseHover(object sender, EventArgs e)
    {
        txtAboutUs.Text = txtAboutUs.Text.ToUpper();
    }

    private void txtAboutUs_MouseLeave(object sender, EventArgs e)
    {
        txtAboutUs.Text = "About Us";
    }

    private void txtAboutUs_Click(object sender, EventArgs e)
    {
        navbarPanel.Left = txtAboutUs.Left;
        Hide();
        ZooInfo zooInfo = new ZooInfo();
        zooInfo.ShowDialog();
    }

    private void backToHome_Click(object sender, EventArgs e)
    {
        Hide();
        Home home = new Home();
        home.ShowDialog();
    }
```

```
    private void textBox6_TextChanged(object sender, EventArgs e)
    {

    }

    private void btnZooOrderSummary_Click(object sender, EventArgs e)
    {
        //Get zoo booking details from database using stored procedure and display on form's controls.
        //Tells the code where the database is.

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("select * from ZooBookingDetails;");

        cmd.Connection = sqlConnection;

        sqlConnection.Open();

        SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(cmd);

        DataTable dt = new DataTable();

        sqlDataAdapter.Fill(dt);

        if (dt.Rows.Count > 0)
        {
            MessageBox.Show("Zoo Order Summary successfully found.");

            //Make label controls visible on form
            lblVisitDate.Visible = true;
            lblNumOfChildTickets.Visible = true;
            lblNumOfAdultTickets.Visible=true;

            txtDateOfVisit.Text = dt.Rows[0]["visitDate"].ToString();
            txtNumOfChildTickets.Text = dt.Rows[0]["numOfAdultTickets"].ToString();
            txtNumOfAdultTickets.Text = dt.Rows[0]["numOfChildTickets"].ToString();
        }

        else
        {
            MessageBox.Show("Zoo Booking Details were not found");
```

```
        }

        sqlConnection.Close();



    }

    private void btnHotelOrderSummary_Click(object sender, EventArgs e)
    {
        //Get hotel booking details from database using stored procedure and display on form's controls.
        //Tells the code where the database is.

        //DEFAULT PATH
        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from
backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("select * from HotelBookingDetails;");

        sqlConnection.Open();

        cmd.Connection = sqlConnection;

        SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(cmd);

        DataTable dt = new DataTable();

        sqlDataAdapter.Fill(dt);

        if (dt.Rows.Count > 0)
        {
            //Make label controls visible on form
            MessageBox.Show("Hotel Order Summary successfully found.");

            lblCheckInDate.Visible = true;
            lblCheckOutDate.Visible = true;
            lblNumOfResidents.Visible = true;
            lblRoomType.Visible = true;

            txtCheckInDate.Text = dt.Rows[0]["checkInDate"].ToString();
            txtCheckOutDate.Text = dt.Rows[0]["checkOutDate"].ToString();
            txtNumOfResidents.Text = dt.Rows[0]["numOfResidents"].ToString();
            txtRoomType.Text = dt.Rows[0]["RoomType"].ToString();
```

```
        }

        else
        {
            MessageBox.Show("Hotel Booking Details were not found");
        }

        sqlConnection.Close();

    }

    private void OrderSummaryPage_Load(object sender, EventArgs e)
    {

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the order summary page of my application. It fetches the hotel booking details from the HotelBookingDetails database and the zoo booking details from the ZooBookingDetails database to display an order summary.

CHANGES DURING DEVELOPMENT

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the order summary page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public OrderSummaryPage(AccessibilityHelper accessibilityHelper)
    {
        InitializeComponent();
        this.accessibilityHelper = accessibilityHelper;
        accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

FORGOTPASSWORDPAGE.CS

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Data.SqlClient;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using static Task2_Code_LL_000013680_Baines_C.Accessibility;
```

```
namespace Task2_Code_LL_000013680_Baines_C
{
   public partial class ForgotPasswordPage : Form
   {
      //public AccessibilityHelper accessibilityHelper { get; set; }
      public ForgotPasswordPage()
      {
         InitializeComponent();
         //this.accessibilityHelper = accessibilityHelper;
         //accessibilityHelper.UpdateFontSize(this.Controls);
      }

      private void settingsIcon_Click(object sender, EventArgs e)
      {
         //Hides the form, loads and displays the form associated with the link that was clicked on.
         Hide();
         SettingsPage settingsPage = new SettingsPage();
         settingsPage.ShowDialog();
      }

      private void txtSignIn_TextChanged(object sender, EventArgs e)
      {

      }

      private void txtSignIn_MouseClick(object sender, MouseEventArgs e)
      {
         navbarPanel.Left = txtSignIn.Left;
         Hide();
         SignInPage signInPage = new SignInPage();
         signInPage.ShowDialog();

      }

      private void txtSignUp_TextChanged(object sender, EventArgs e)
      {

      }

      private void txtSignUp_MouseClick(object sender, MouseEventArgs e)
      {
         navbarPanel.Left = txtSignIn.Left;
         Hide();
         SignUpPage signUpPage = new SignUpPage();
         signUpPage.ShowDialog();
      }

      private void ForgotPasswordPage_Load(object sender, EventArgs e)
      {

      }

      private void rzaNavLogo_Click(object sender, EventArgs e)
```

```
    {

    }

    private void button1_Click(object sender, EventArgs e)
    {
        //Check to see if the user has entered their current username which is stored in the userDetails database using an SQL query.

        string currentUsername = txtEnterCurrentUsername.Text;

        //DEFAULT PATH
        string connectionString = "Data Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename = \"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";
        SqlConnection sqlConnection = new SqlConnection(connectionString);

        //SQL Query which selects all records from the userDetails database where the user-inputted username is equal to the username stored in the database.
        SqlCommand cmd = new SqlCommand("select * from userDetails where username like @userName;");

        cmd.Connection = sqlConnection;

        cmd.Parameters.AddWithValue("@userName", txtEnterCurrentUsername.Text);

        sqlConnection.Open();

        DataSet dataset = new DataSet();
        SqlDataAdapter dataadapter = new SqlDataAdapter(cmd);
        dataadapter.Fill(dataset);

        sqlConnection.Close();

        bool currentUsernameFound = (dataset.Tables.Count > 0);



        //Checks to see if the user has made an input.
        if (txtEnterCurrentUsername.Text == string.Empty)
        {
            MessageBox.Show("Please enter your current username.");
        }

        //If current username is found it makes the controls visible in the form which allows the user to enter a new password.
        if (currentUsernameFound)
```

```
        {
            MessageBox.Show("Please create a new password");
            lblResetPassword.Visible = true;
            txtCreatePassword.Visible = true;
            btnResetPassword.Visible = true;

        }

        else
        {
            MessageBox.Show("Your current username has not been found. Please try entering it again.");
        }

        sqlConnection.Close();
    }

    private void btnResetPassword_Click(object sender, EventArgs e)
    {
        //string newPassword = txtCreatePassword.Text;

        string hashedPassword = BCrypt.Net.BCrypt.HashPassword(txtCreatePassword.Text);

        //Tells the code where the database is.

        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH

        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2 from backup\\zoo application database.mdf\";Integrated
Security=True;Connect Timeout=30";

        SqlConnection sqlConnection = new SqlConnection(connectionString);

        SqlCommand cmd = new SqlCommand("update userDetails set password=@password");

        cmd.Connection = sqlConnection;

        sqlConnection.Open();

        //Check to see if a new password has been entered

        if (hashedPassword == string.Empty)
        {
            MessageBox.Show("Please enter a new password to reset your old one.");
        }

        //Length check to check if the new password entered meets the requirements for the database.
```

```
        //Also to ensure usernames are not too small or too large.
        if (hashedPassword.Length < 5 || hashedPassword.Length > 100)
        {
            MessageBox.Show("Your password is either too small or too large. It should be between 5 characters
to 50 characters long.");
        }

        else
        {

            cmd.Parameters.AddWithValue("@password", hashedPassword);

            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully reset your password. Please sign back in.");

            sqlConnection.Close();

            Hide();
            SignInPage signInPage = new SignInPage();
            signInPage.ShowDialog();
        }

        sqlConnection.Close();



    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the forgot password page on my application. It allows the user to reset their password if they enter their existing username. It checks to see if the existing username is stored in the userDetails database by executing an SQL query which selects all records from the userDetails database where the user-inputted username is equal to the username stored in the database. The user can then reset their password which is hashed and updated in their record, stored in the userDetails database. The same length checks and prescence checks that were in the sign-up page are used to determine whether the user's new password meets the requirements for storage in the database.

CHANGES DURING DEVELOPMENT

Version 1.01

Hashing was implemented to the forgot password page in version 1.01 for security purposes so that the reset passwords would also be stored as hashes in the userDetails database.

Version 1.02

In Version 1.02, the bug was fixed on the forgot password page where the page was not restricting users from resetting their password if they had entered an invalid current username.

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the forgot password page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public ForgotPasswordPage(AccessibilityHelper accessibilityHelper)
    {
       InitializeComponent();
       this.accessibilityHelper = accessibilityHelper;
       accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the forgot password page.

```
private void ForgotPasswordPage_Load(object sender, EventArgs e)
        {
            //DEFAULT PATH
            string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDri
ve - Middlesbrough College\\Documents\\Carl Baines\\Task 2\\zoo application
database.mdf\";Integrated Security=True;Connect Timeout=30; Integrated
Security = True";
            //

            //BACKUP PATH
            //
            //string connectionString = "Data Source =
(LocalDB)\\MSSQLLocalDB; AttachDbFilename = \"C:\\Users\\EXAM1238\\OneDrive -
Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout =
30;";

            SqlConnection sqlConnection = new
SqlConnection(connectionString);

            SqlCommand sqlCommand = new SqlCommand("GetHighContrast",
sqlConnection);
            sqlCommand.CommandType = CommandType.StoredProcedure;

            sqlCommand.Parameters.AddWithValue("@highContrast", 0);

            sqlConnection.Open();

            SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
```

```
            if (highContrastReturned.Read())
            {
                if (highContrastReturned[0].ToString() == "0")
                {
                    return;
                }
                else
                {
                    //enable high contrast
                    BackgroundImage = null;
                    this.BackColor = Color.FromArgb(255,234,0);
                }
            }

            sqlConnection.Close();
```

CHANGEACCOUNTDETAILS.CS

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Data.SqlClient;

using System.Drawing;

using System.Linq;

using System.Runtime.Remoting.Contexts;

using System.Security.Cryptography.X509Certificates;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.ProgressBar;

using static System.Windows.Forms.VisualStyles.VisualStyleElement.StartPanel;

using static Task2_Code_LL_000013680_Baines_C.Accessibility;


namespace Task2_Code_LL_000013680_Baines_C

{
```

```
public partial class ChangeAccountDetails : Form
{
    //public AccessibilityHelper accessibilityHelper { get; set; }
    public ChangeAccountDetails()
    {
        InitializeComponent();
        //this.accessibilityHelper = accessibilityHelper;
        //accessibilityHelper.UpdateFontSize(this.Controls);
    }

    private void btnSubmit_Click(object sender, EventArgs e)
    {
        //Check to see if user-inputted username is in the database using SQL query.

        string oldUsername = txtEnterOldUsername.Text;

        //Tells the code where the database is.

        //DEFAULT PATH
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";
        //

        //BACKUP PATH
        //
        //string connectionString = "Data Source = (LocalDB)\\MSSQLLocalDB; AttachDbFilename =
\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout = 30;";
```

```
        SqlConnection sqlConnection = new SqlConnection(connectionString);









        //SQL Query which selects all records from the userDetails database where the user-inputted username is
equal to the username stored in the database.
        SqlCommand cmd = new SqlCommand("select * from userDetails where username like @userName;");


        cmd.Connection = sqlConnection;


        cmd.Parameters.AddWithValue("@userName", oldUsername);


        sqlConnection.Open();


        DataSet dataset = new DataSet();
        SqlDataAdapter dataadapter = new SqlDataAdapter(cmd);
        dataadapter.Fill(dataset);


        sqlConnection.Close();


        bool oldUsernameFound = (dataset.Tables.Count > 0);


        //Checks to see if the user has made an input.
        if (txtEnterOldUsername.Text == string.Empty)
        {
            MessageBox.Show("Old username not found. Please try again.");
        }
```

```
        //If old username is found it makes the controls visible in the form which allows the user to enter a new
username.
        if (oldUsernameFound)
        {
            MessageBox.Show("You can now create a new username. You can also change your password if wanted.");
            lblEnterNewUsername.Visible = true;
            txtEnterNewUsername.Visible = true;
            btnSubmitNewUsername.Visible = true;

            lblCreateNewPassword.Visible = true;
            txtCreateNewPassword.Visible = true;
            btnSubmitNewPassword.Visible = true;
        }

        else
        {
            MessageBox.Show("Old username not found. Please try again.");
        }
    }

    private void btnSubmitOldPassword_Click(object sender, EventArgs e)
    {

    }

    private void txtAccountDetails_MouseHover(object sender, EventArgs e)
    {
        txtAccountDetails.Text = txtAccountDetails.Text.ToUpper();
    }
```

```
private void txtAccountDetails_MouseLeave(object sender, EventArgs e)
{
    txtAccountDetails.Text = "Account Details";
}

private void txtSignIn_MouseHover(object sender, EventArgs e)
{
    txtSignIn.Text = txtSignIn.Text.ToUpper();
}

private void txtSignIn_MouseLeave(object sender, EventArgs e)
{
    txtSignIn.Text = "Sign In";
}

private void txtSignIn_MouseClick(object sender, MouseEventArgs e)
{
    navbarPanel.Left = txtSignIn.Left;
    Hide();
    SignInPage signInPage = new SignInPage();
    signInPage.ShowDialog();
}

private void txtSignUp_MouseHover(object sender, EventArgs e)
{
    txtSignUp.Text = txtSignUp.Text.ToUpper();
}
```

```
private void txtSignUp_MouseLeave(object sender, EventArgs e)
{
    txtSignUp.Text = "Sign Up";
}

private void txtSignUp_MouseClick(object sender, MouseEventArgs e)
{
    navbarPanel.Left = txtSignUp.Left;
    Hide();
    SignUpPage signUpPage = new SignUpPage();
    signUpPage.ShowDialog();
}

private void ChangeAccountDetails_Load(object sender, EventArgs e)
{

}

public void btnSubmitNewUsername_Click(object sender, EventArgs e)
{
    //Update record in database with the new username.


    string newUsername = txtEnterNewUsername.Text;

    //Tells the code where the database is.

    //DEFAULT PATH
```

```
        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";

        //



        //BACKUP PATH



        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2 from backup\\zoo application database.mdf\";Integrated
Security=True;Connect Timeout=30";



        SqlConnection sqlConnection = new SqlConnection(connectionString);



        SqlCommand cmd = new SqlCommand("update userDetails set userName=@userName");



        cmd.Connection = sqlConnection;



        //SqlCommand cmd = new SqlCommand("UpdateUsername", sqlConnection);



        //Run stored procedure and update the record in the database with the new username.

        //cmd.CommandType = CommandType.StoredProcedure;



        sqlConnection.Open();



        //Checks to see if a new username has been entered.



        if (txtEnterNewUsername.Text == string.Empty)

        {

            MessageBox.Show("Please enter a new username");
```

```
        }

        //Length check to check if the new username entered meets the requirements for the database.
        //Also to ensure usernames are not too small or too large.
        if (txtEnterNewUsername.Text.Length < 5 || txtEnterNewUsername.Text.Length > 25)
        {
            MessageBox.Show("Your username is either too small or too large. It should be between 5 characters to 25 characters long.");
        }

        else
        {




            cmd.Parameters.AddWithValue("@userName", newUsername);

            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully changed your username.");

            sqlConnection.Close();
        }

        sqlConnection.Close();
    }

    private void btnSubmitNewPassword_Click(object sender, EventArgs e)
```

```
    {
        //Create new password and update record in database


        //string newPassword = txtCreateNewPassword.Text;


        string newHashedPassword = BCrypt.Net.BCrypt.HashPassword(txtCreateNewPassword.Text);


        //Tells the code where the database is.


        //DEFAULT PATH

        string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2\\zoo application database.mdf\";Integrated Security=True;Connect
Timeout=30; Integrated Security = True";

        //


        //BACKUP PATH

        //

        //string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDrive - Middlesbrough
College\\Documents\\Carl Baines\\Task 2 from backup\\zoo application database.mdf\";Integrated
Security=True;Connect Timeout=30";




        SqlConnection sqlConnection = new SqlConnection(connectionString);


        SqlCommand cmd = new SqlCommand("UpdatePassword", sqlConnection);


        sqlConnection.Open();


        //Checks to see if a new password has been entered.
```

```
        if (newHashedPassword == string.Empty)
        {
            MessageBox.Show("Please enter a new password");
        }

        //Length check to check if the new password entered meets the requirements for the database.
        //Also to ensure passwords are not too small or too large.
        if (newHashedPassword.Length < 5 || newHashedPassword.Length > 100)
        {
            MessageBox.Show("Your password is either too small or too large. It should be between 5 characters to 50 characters long.");
        }

        else
        {
            //Run stored procedure and update the record in the database with the new username.
            cmd.CommandType = CommandType.StoredProcedure;

            //FIND A WAY TO GET CURRENT ID OF USER WITHOUT THEM HAVING TO INPUT IT (THEY WOULDN'T KNOW IT)
            //REST OF CODE WORKS :)



            //cmd.Parameters.AddWithValue("@StdID", WE NEED A PARAMETER EMERGENCY);




            cmd.Parameters.AddWithValue("@password", newHashedPassword);
```

```
            cmd.ExecuteNonQuery();

            MessageBox.Show("You have successfully changed your password. You will be redirected back to the sign in page.");

            sqlConnection.Close();

            Hide();
            SignInPage signInPage = new SignInPage();
            signInPage.ShowDialog();
        }

        sqlConnection.Close();

    }
  }
}
```

DESCRIPTION OF CODE

This is the code for the change account details page of my application. It works by allowing the user to change their username by entering their old one. If they enter their old username, they can create a new username as well as change their password for their account.

The code checks to see if the user-inputted username is the current username of their account by running an SQL query which selects all records from the userDetails database where the user-inputted username is equal to the username stored in the database. If the current username of the account is found it makes the necessary controls visible in the form which allow the user to enter a new username. The new username input is length and presence checked before being updated in the user's record, stored in the userDetails database, through an SQL command.

The change account details page also allows the user to change their password. The new password is length and presence checked to see if it meets the requirements for database storage before being hashed and updating within the user's record, inside the userDetails database.

CHANGES DURING DEVELOPMENT

Version 1.01

Hashing was implemented to the change account details page in version 1.01 for security purposes so that the reset passwords would also be stored as hashes in the userDetails database.

Version 1.02

In Version 1.02, an issue was fixed where the page was not restricting users from resetting their password if they had entered an invalid current username.

Version 1.03

In Version 1.03, a font size changer accessibility feature was added allowing the user to change the font size of the common controls within all the application forms including the forgot password page.

```
public AccessibilityHelper accessibilityHelper { get; set; }
    public ChangeAccountDetails()
    {
       InitializeComponent();
       this.accessibilityHelper = accessibilityHelper;
       accessibilityHelper.UpdateFontSize(this.Controls);

    }
```

Version 1.04

In Version 1.04, a functioning high contrast mode was implementing allowing the user to change the appearance of the application to high contrast by clicking the checkbox in the settings page. The setting applies to all pages across the application including the change account details page.

```
private void ChangeAccountDetails_Load(object sender, EventArgs e)
        {
            //DEFAULT PATH
            string connectionString = "Data
Source=(LocalDB)\\MSSQLLocalDB;AttachDbFilename=\"C:\\Users\\EXAM1238\\OneDri
ve - Middlesbrough College\\Documents\\Carl Baines\\Task 2\\zoo application
database.mdf\";Integrated Security=True;Connect Timeout=30; Integrated
Security = True";
            //

            //BACKUP PATH
            //
            //string connectionString = "Data Source =
(LocalDB)\\MSSQLLocalDB; AttachDbFilename = \"C:\\Users\\EXAM1238\\OneDrive -
Middlesbrough College\\Documents\\Carl Baines\\Task 2 from backup\\zoo
application database.mdf\"; Integrated Security = True; Connect Timeout =
30;";
```

```
            SqlConnection sqlConnection = new
SqlConnection(connectionString);

            SqlCommand sqlCommand = new SqlCommand("GetHighContrast",
sqlConnection);
            sqlCommand.CommandType = CommandType.StoredProcedure;

            sqlCommand.Parameters.AddWithValue("@highContrast", 0);

            sqlConnection.Open();

            SqlDataReader highContrastReturned = sqlCommand.ExecuteReader();
            if (highContrastReturned.Read())
            {
                if (highContrastReturned[0].ToString() == "0")
                {
                    return;
                }
                else
                {
                    //enable high contrast
                    BackgroundImage = null;
                    this.BackColor = Color.FromArgb(255,234,0);
                }
            }

            sqlConnection.Close();
        }
```

Also, in Version 1.04, input validation was implemented to prevent identical registered usernames from being added to records in the userDetails database, via the change account details page.

```
if (reader.HasRows)
            {
                //Username is already taken in database
                MessageBox.Show("Username is already taken. Please enter a
different one.", "Error",
    MessageBoxButtons.OK, MessageBoxIcon.Error);
                reader.Close();
                sqlConnection.Close();
            }

            else
            {
                reader.Close();

                //Changes command to update username in userDetails record.
                cmd.CommandText = ("update userDetails set
userName=@userName");

//Rest of code
```