# Chapter 4

# Software Development

During implementation we face some difficulties but were later resolved.

* Data base Migration
* Connection of controller with view

## Coding Standards

* **Camel Case**: For arguments and local variable.
* **Pascal Case**: For class names and methods.

# 4.1.1 Indentation

Tab spaces (8 spaces) are used as unit of indentation. The indentation is followed throughout the project consistently

**4.1.2 Declaration**

* Instance variables are placed in the sequence: First public instance variables, protected,
* Package level with no access modifier and then private.
* Next the class constructors are declared.
* Html and stylesheet are used.

## Development Environment

There are different types of languages used in the project and the type of Environment used is given below.

* **Programming Language:**
  + **C#** will be used at backend for web application.
  + **HTML5, CSS3** will be used at frontend for web application.
* **Development Environment:**
* **Visual Studio:** It will be used for the development, debugging and simulation of the webserver.

## Software Description

Main modules of our project are

* User Registration.
* User Login.
* Reservation
* Menu.

**Snippet 1**

[HttpPost]

[AllowAnonymous]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Login(LoginViewModel model, string returnUrl = null)

{

ViewData["ReturnUrl"] = returnUrl;

if (ModelState.IsValid)

{

// This doesn't count login failures towards account lockout

// To enable password failures to trigger account lockout, set lockoutOnFailure: true

var result = await \_signInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, lockoutOnFailure: false);

if (result.Succeeded)

{

\_logger.LogInformation("User logged in.");

return RedirectToLocal(returnUrl);

}

if (result.RequiresTwoFactor)

{

return RedirectToAction(nameof(LoginWith2fa), new { returnUrl, model.RememberMe });

}

if (result.IsLockedOut)

{

\_logger.LogWarning("User account locked out.");

return RedirectToAction(nameof(Lockout));

}

else

{

ModelState.AddModelError(string.Empty, "Invalid login attempt.");

return View(model);

}

}

// If we got this far, something failed, redisplay form

return View(model);

}

**Description**:

The above code describe how the login process will work.If its login successfully then what it shows and if it not works then what happens. Users are login into the system in this module. User enters the username and password if the username and password matches with the saved data in database it will navigate to the next page. If the required filed is incorrect it will display Email is required, and password is required

**Snippet :2**

public class Reservation

{

[Key]

public int id { get; set; }

public string FirstName { get; set; }

public string LastName { get; set; }

public String Address { get; set; }

public String Country { get; set; }

public String City { get; set; }

public int PhoneCode { get; set; }

public int PhoneNo { get; set; }

public String State { get; set; }

public int ZipCode { get; set; }

public int NumberofGuests { get; set; }

public DateTime ReservationDate { get; set; }

public string Title { get; set; }

public Boolean Status { get; set; }

public ApplicationUser user { get; set; }

}

}

**Description**:

Reservation class what attributes are used to make reservation. And all method of reservation is called in this class.

Snippet:3

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("id,FirstName,LastName,Address,Country,City,PhoneCode,PhoneNo,State,ZipCode,NumberofGuests,ReservationDate,Title,Status")] Reservation reservation)

{

if (ModelState.IsValid)

{

\_context.Add(reservation);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(reservation);

}

Description:

When we make any reservation

# Chapter 5

# Software Testing

Software Testing is the most crucial part of Software Development Process. It is the investigation or evaluation of a software component, improving them, and finding bugs and defects. Testing is usually done by executing a system in such a way that it identifies any gaps, errors, or missing requirements in contrary to the actual requirements.

## 5.1. Testing Methodology.

It is essential to have a testing plan in place to ensure that the product delivered is stable, and is delivered on a predictable timeline.

* + Testing is done
  + Project is stable.
  + If a person gives a wrong value’s system should display error message.
  + If a person gives a Right value’s system should display right result.

## 5.2. Testing Environment

*Describe and discuss the reason to use the selected testing environment.*

## 5.3. Test Cases

* **Test Case 1**

**Test Case: Registration**

Table 1: Registration/Password Validation

|  |  |
| --- | --- |
| Date: 26 May 2018 |  |
| *System: wedding vibes* |  |
| *Objective:* Registration/Password Validation | *Test ID:*1 |
| *Version:*1 | *Test Type:* Black Box testing |
| *Input:*  Password: 12345 | |
| *Expected Result:* Return Error of minimum 8 characters required | |
| *Actual Result:* Error of minimum 8 characters required | |

* **Test Case 2**

**Test Case: Registration**

***Table 2: Registration/Email Validation***

|  |  |
| --- | --- |
| Date: 28 May 2018 |  |
| *System:* wedding vibes |  |
| *Objective:* Registration/ Email Validation | *Test ID:*2 |
| *Version:*1 | *Test Type:* Black Box testing |
| *Input:*  Email Id: ali | |
| *Expected Result:* return Error of please enter valid email address | |
| *Actual Result:* Error of invalid email address | |

### 

* **Test Case 3**

**Test Case: Registration**

**Table 3: Registration/Required Field Validation**

|  |  |
| --- | --- |
| Date: 28 May 2018 |  |
| *System:* wedding vibes |  |
| *Objective:* Registration/ Required field Validation | *Test ID:*3 |
| *Version:*1 | *Test Type:* Black Box testing |
| *Input: null* | |
| *Expected Result:* return name, password, confirm password | |
| *Actual Result:* Error of name, password, confirm password, email | |

* **Test Case 4**

**Test Case: Reservation**

**Table 4: Reservation Required Field Validation**

|  |  |
| --- | --- |
| Date: 28 May 2018 |  |
| *System:* wedding vibes |  |
| *Objective:* Reservation Required field Validation | *Test ID:*4 |
| *Version:*1 | *Test Type:* Black Box testing |
| *Input:* | |
| *Expected Result:* return name, password, confirm password | |
| *Actual Result:* Error of name, password, confirm password, email | |