C868 – Software Capstone Project Summary

Task 2 – Section C

Capstone Proposal Project Name: Appointment Scheduling Application

Student Name: Brian Mason

Table of Contents

Contents

Task 2 Part C – C868 Software Development Capstone

Application Design and Testing

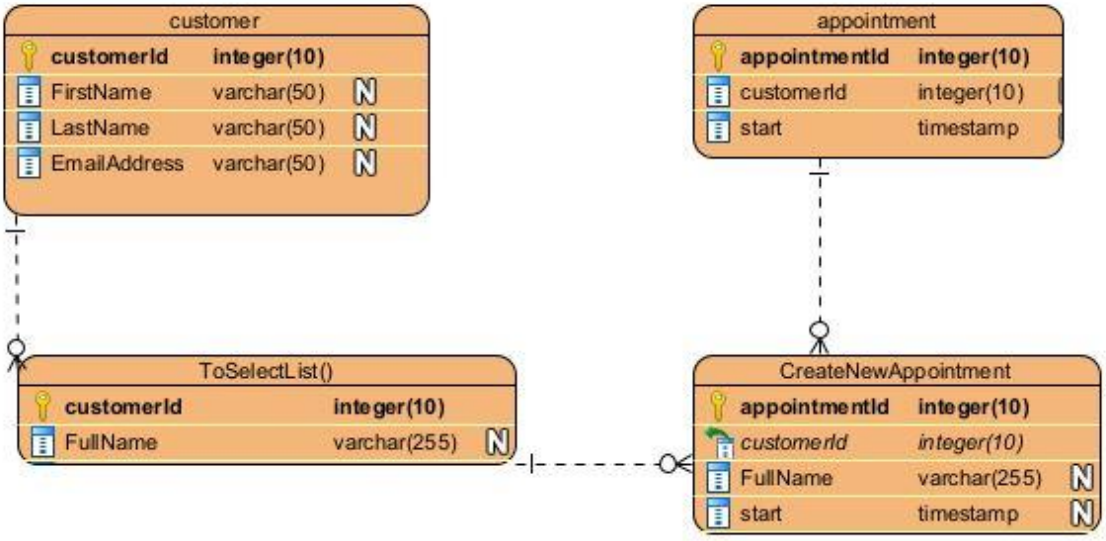
Design Document

Class Design

Below is an ERD diagram of the classes used to create customer objects and appointment objects, as well as the web views associated with creating, viewing, and updating those customers and appointments.

Creating a customer object requires that a user enter valid information for a first name, a last name, and an email address (with confirmation). Razor syntax applies a regular expression (regex) to ensure the email field meets the standard requirements. The customerID field is automatically created and incremented when the HttpPost function sends the other fields to the database.

Creating a new appointment object or editing an existing one only calls for referencing the customerID field of the associated customer object, so a helper function ToSelectList() located in the MainController aggregates the first and last name fields into a new full name field, directly tied to the customerID in a SelectList object. The SelectList object is used to view and manipulate the customerID field of the appointment object, while being user friendly and accessible by using the associated customer object's full name.



UI Design

In the images below, Figure 1 demonstrates the look of the Login page presented to the user on the application home page. The CSS was modified from an open-source template, with attribution to the site at https://w3layouts.com. Figure 2 shows the design of the 'Create New Customer' page, which is also representative of the 'Create New Appointment' page. Figure 3 shows the 'View All Appointments' page, which is also representative of the 'View All Customers' and 'Display Report' pages.

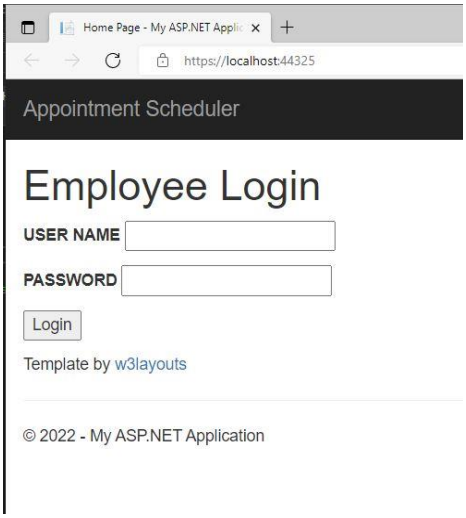


Figure 1: High Fidelity Login Screen

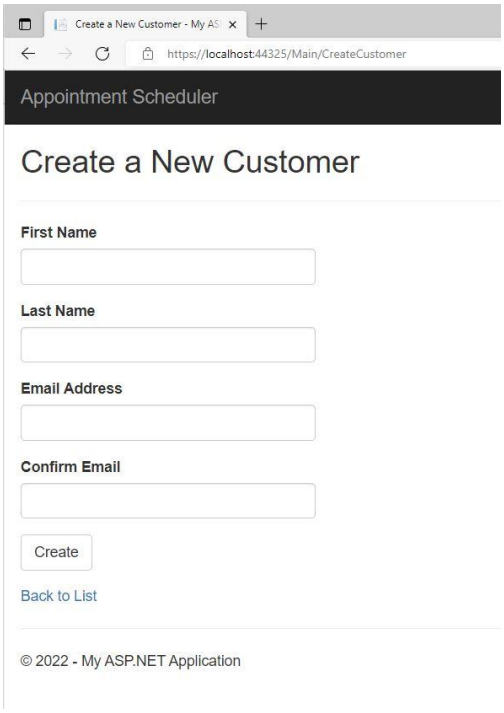


Figure 2: High Fidelity 'Create New Customer' Screen

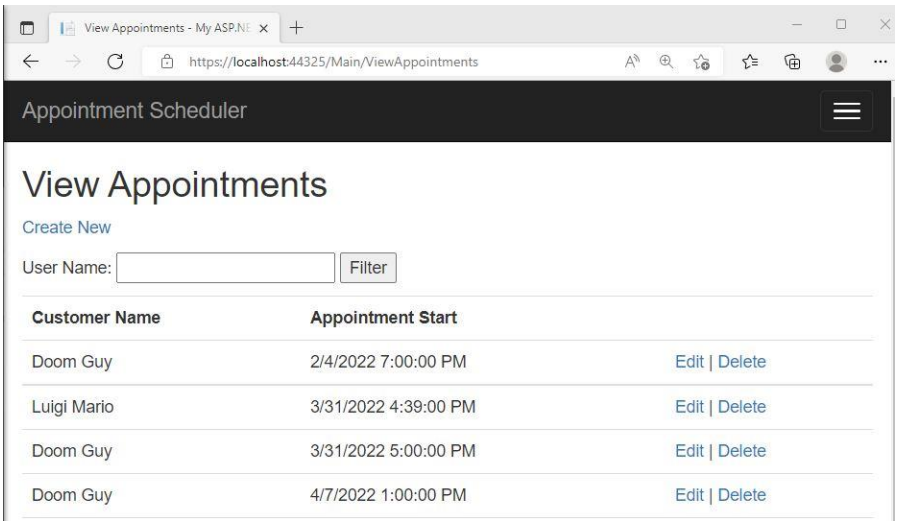


Figure 3: High Fidelity 'View All Appointments' Screen

Unit Test Plan

Introduction

Purpose

In order to prevent appointments from being accidentally scheduled outside of business hours, validation is required both when creating and editing appointments within this application.

Overview

In the two pages which use DateTime pickers to select the start time for an appointment (‘CreateAppointment’ and ‘EditAppointment’), a JavaScript function named validateDateTime() is called when the Submit button is clicked. This function is used to verify that the selected DateTime is not prior to the current moment, and that it falls between Monday and Friday and the hours of 8am to 5pm.

Test Plan

Items

A JavaScript DateTime picker plugin (https://xdsoft.net/jqplugins/datetimepicker/) is used on these pages, so that a user can select a date and time from a calendar, as opposed to typing it in manually.

Features

When the user clicks the Submit button on the web form, the function validateDateTime() is called. The function gathers the values for the current date and time from the user’s system, as well as the selected date and time from the picker. It then verifies that the selected date and time falls within the allowed ranges.

Deliverables

The test produces an alert popup on the web page if the selected time is determined to not be valid, along with an appropriate error message to tell the user what went wrong. Once the user interacts with the alert, the picker data is cleared to allow them to select again.

Tasks

No additional steps must be taken to run this test, as it was left in place in the application. It was decided that this unit test also served as the most efficient way to validate the user's input and ensure that clean values were sent to the database.

Needs

Requirements for this test involve two applications:

1. Visual Studio 2019 is used to run the application itself
2. MySQL Workbench is used to host the database

Pass/Fail Criteria

The result is considered a failure when a selected date time is rejected, resulting in the error message being presented to the user as an alert message within the web browser. It is considered a success when the date time is accepted, and the appointment is created or updated within the database.

Specifications

Below is a screenshot of the validateDateTime() function. This JavaScript function is located within the '@section Scripts' section of the CreateAppointment.cshtml and EditAppointment.cshtml pages.

```
function validateDateTime() {

    let testStartTime = new Date($('#Start').val()).getTime();
    let testStartDate = new Date($('#Start').val());
    let testStartHour = new Date($('#Start').val()).getHours();
    let testStartDay = new Date($('#Start').val()).getDay();
    let currentDate = new Date();
    let currentTime = new Date().getTime();

    if (testStartDate < currentDate) {
        alert("Appointment start date and time cannot be before the current date and time.");
        $("#Start").datetimepicker('reset');
        return false;
    }
    if (testStartDate == currentDate) {
        if (testStartTime < currentTime) {
            alert("Appointment start time cannot be before the current time.");
            $("#Start").datetimepicker('reset');
            return false;
        }
    }
    if (!((0 < testStartDay) && (testStartDay < 6))) {
        alert("Selected date must be between Monday through Friday.");
        $("#Start").datetimepicker('reset');
        return false;
    }
    if (((testStartHour < 8) || (testStartHour > 17))) {
        alert("Selected time must be between 8am and 5pm. Please select a time from the picker.");
        $("#Start").datetimepicker('reset');
        return false;
    } else {
        return true;
    }
}
```

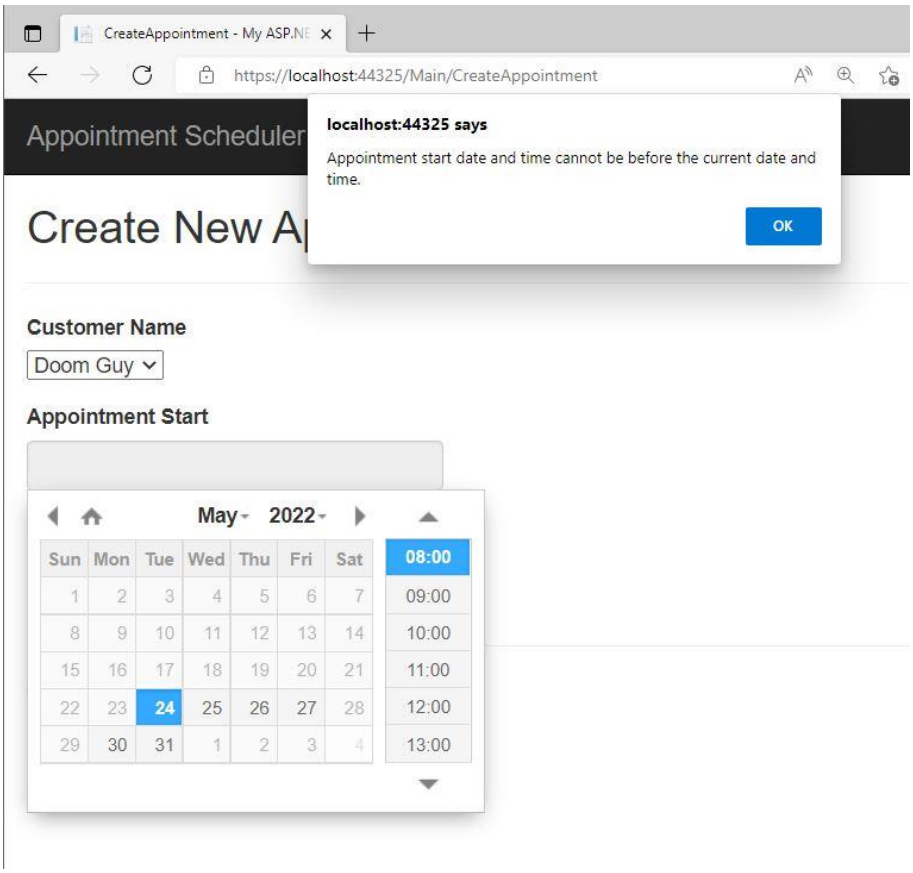
Procedures

The procedures I used for performing this test are as follows:

1. Navigated to the Create New Appointment page.
2. Ensured that a valid user was selected from the Customer Name dropdown list.
3. Selected a date and time prior to the current date and time.
4. Clicked on the Create/Submit button in order to run the validation script.
5. Verified that the appropriate error message was generated by the web page, and that it did not allow the invalid data to be submitted.

Results

The resulting error message is sent to the user as an alert, requiring them to acknowledge the error and correct the mistake in order to proceed.



C4. Source Code

The attached zip "Mason_capstone.zip" contains the source code for the application.

C5. Link to Live Version

The attached zip "Mason_capstone.zip" contains the source code for the application. Once extracted, navigate to /SchedulerWebApp/SchedulerWebApp/ and launch 'SchedulerWebApp.sln' into Visual Studio 2019, then press Ctrl+F5 to 'Start without

debugging', which will launch the website into your default web browser. The login page will require a username of 'test' as well as a password of 'test' to access the remainder of the site.

(Note: prior to running the site and logging in successfully, the database scripts must be run to create and populate the database. Instructions for this process are in the following section of this document.)

User Guide

Introduction

The sections provided below in the User Guide will instruct the user on how to install, log into, and use all of the functions of the application. This tutorial is best run using Windows 7 (or later) or Windows Server 2008 R2 (or later).

(Note: the default username and password for this site are both 'test', and the user will be required to login using those credentials before accessing the core of the site.)

Installation and Launching the Application

Prerequisites:

- Visual Studio 2019
- MySQL Workbench 8.0.26 or newer
- .NET Framework version 4.8

(Note: WGU course instructions have us hosting this application on a Virtual Machine hosted by the school. The link from the course itself is for the Java track and does not have the required software installed. I am using the link from C969 which is compatible with C# applications, at https://lrps.wgu.edu/provision/270736325)

Installation:

1. Update .NET on the VM to version 4.8 via the URL https://dotnet.microsoft.com/en-us/download/dotnet-framework/thank-you/net48-web-installer
2. Open Visual Studio 2019 and click ‘Continue without code’ at the bottom of ‘Get Started’ on the right side of the window.
3. At the top of the window, click on Tools -> Get Tools and Features. Select ‘ASP.NET and web development’ near the top left, as well as ‘.NET Framework 4.8 development tools’ along the right side. Download and install both, then restart Visual Studio.
4. Unzip the provided source code folder, “Mason_capstone.zip”, and load /SchedulerWebApp/SchedulerWebApp/SchedulerWebApp.sln into Visual Studio
5. From the Solution Explorer window, right-click on SchedulerWebApp, then click ‘Set as Startup Project’. Right click on SchedulerWebApp again, then click on Properties at the bottom of the window. In the new Properties window, select ‘Web’ from the left sidebar, then the radio button for ‘Start URL’ and enter https://localhost:44325/
6. Update the NuGet package ‘Microsoft.CodeDom.Providers.DotNetCompilerPlatform’ to version 3.6.0
7. Open the MySQL workbench, and double-click on the existing database connection, likely entitled “JavaConnection”. Copy and paste the script data from CreateDBScript.sql into the workbench and execute it to create the required database tables. Copy and paste the script data from PopulateDBScript.sql once this is complete, to populate the database with startup values.

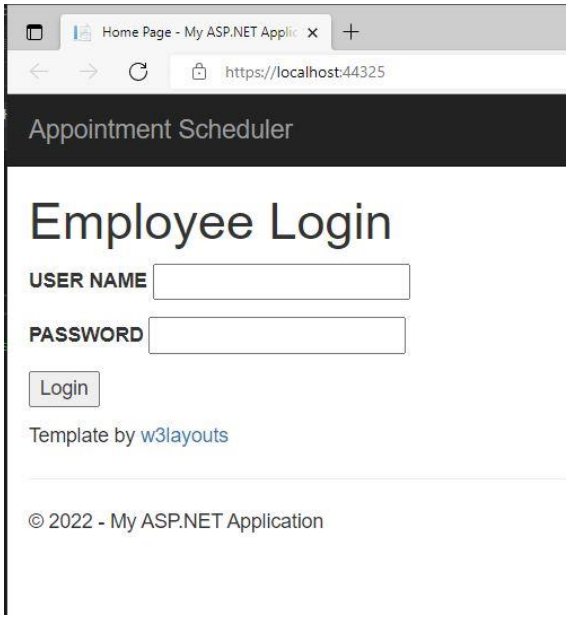
8. Back in Visual Studio, you should now be able to launch the website using CTRL-F5, allowing it to launch in the default web browser.

   (Note: Accept and install any SSL certificates if prompted)

Using the Application:

Login Steps

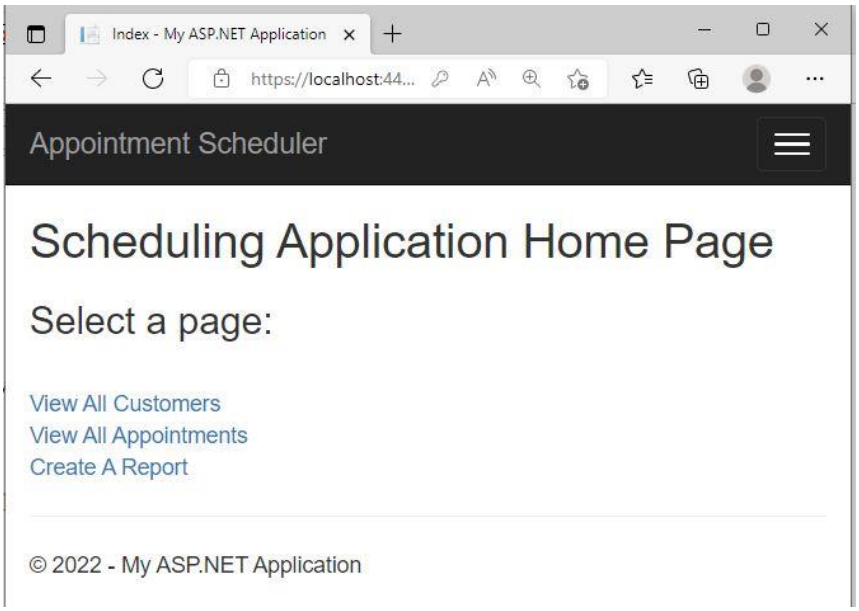
1. Launch the website from Visual Studio as described above.
2. Enter your credentials in the User Name and Password fields, and click Login



3. For this version of the application, user accounts must be created by an administrator with access to the database. Password resets must similarly go through a local administrator.

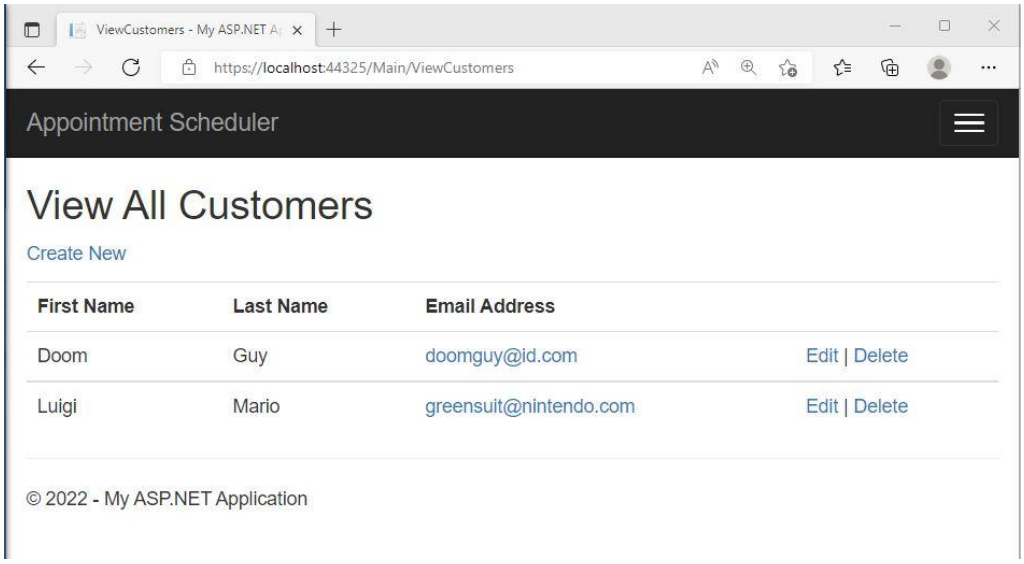
Navigating the Site

The site is broken down into three major components, each of which can be accessed by links on the home page or from the same links on the navigation bar. The primary link will remain static, but the navigation bar will resize itself into a hamburger menu in the top right if the window is shrunk to a narrower size.



View All Customers

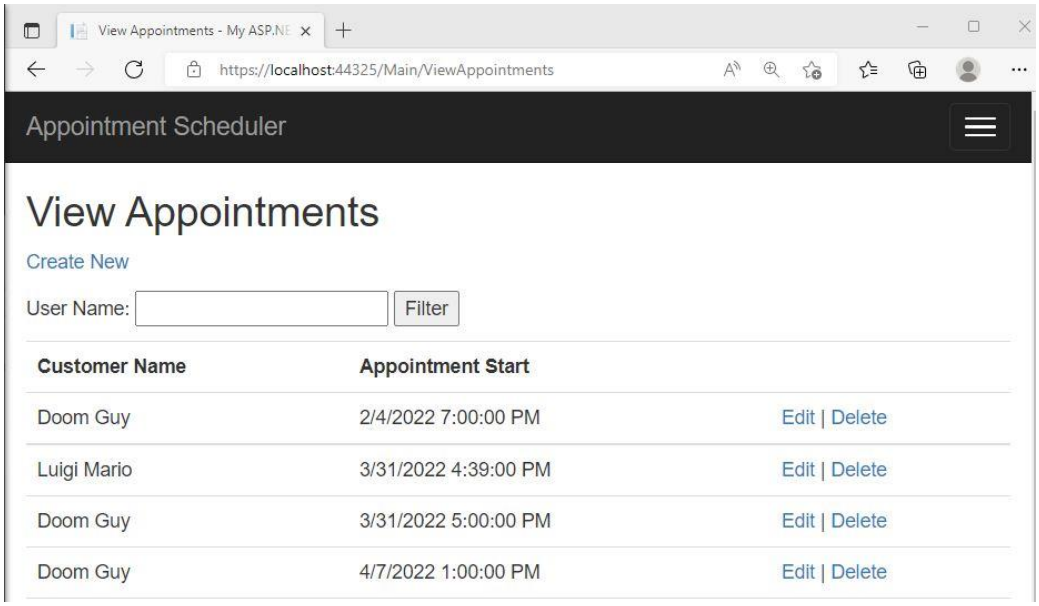
1. The View All Customers page will provide a complete listing of existing customers. At the top left of the page is a link to Create a New Customer, and at the right side of each existing customer is a link to edit or delete that customer.



View All Appointments

1. The View All Appointments page will similarly provide a list of all existing appointments. The same Create New, Edit, and Delete links exist. Additionally,

there is a search filter which allows the user to refine the view to specific customer names.



Reports

1. The Create a Report page will allow a user to show all existing appointments between a date range that they specify. Both a start and end date for the report must be selected from the Date Pickers on the page.