**C868 – Software Capstone Project Summary**

**Task 2 – Section C**

|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | http://www.idevnews.com/views/images/uploads/general/wgu_logo.png  Appointment/Customer Data Custom Software |
| **Student Name:** | Micah Senn |

Table of Contents

[Application Design and Testing 3](#_Toc128827987)

[Design Document 3](#_Toc128827988)

[UML Class Design 3](#_Toc128827989)

[UI Design 4](#_Toc128827991)

[Unit Test Plan 5](#_Toc128827992)

[Purpose 5](#_Toc128827993)

[Overview 5](#_Toc128827994)

[Test Plan 6](#_Toc128827995)

[Items 6](#_Toc128827996)

[Features 6](#_Toc128827997)

[Deliverables 6](#_Toc128827998)

[Tasks 6](#_Toc128827999)

[Needs 7](#_Toc128828000)

[Pass/Fail Criteria 7](#_Toc128828001)

[Specifications 7](#_Toc128828002)

[Procedures 8](#_Toc128828003)

[Results 9](#_Toc128828004)

[C4. Source Code 9](#_Toc128828005)

[User Guide 9](#_Toc128828006)

[Installation 9](#_Toc128828007)

[*Login and Accounts* 11](#_Toc128828008)

[*Customers* 12](#_Toc128828009)

[*Add a Customer Record* 12](#_Toc128828010)

[*Modify an Existing Customer Record* 13](#_Toc128828011)

[*Deleting a Customer Record* 14](#_Toc128828012)

[*Searching for Customers* 14](#_Toc128828013)

[*Appointments* 15](#_Toc128828014)

[*Add an Appointment Record* 15](#_Toc128828015)

[*Modify an Appointment Record* 16](#_Toc128828016)

[*Deleting an Appointment Record* 17](#_Toc128828017)

[*Searching for Appointments* 17](#_Toc128828018)

[*View Calendar and Reports* 17](#_Toc128828019)

[*View Appointment Calendar* 17](#_Toc128828020)

[*View Job Type Report* 18](#_Toc128828021)

[*View Scheduling Report* 18](#_Toc128828022)

[*View Customer Report* 18](#_Toc128828023)

# Application Design and Testing

# Design Document

## UML Class Design

The diagram below shows the UML class design for the delivered application. The diagram demonstrates the four main classes used in the application and the relationships between them. As you can see the user class has a one-to-many relationship with the appointment class, the appointment class has a many-to-one relationship with the customer class, and the customer class has a one-to-one relationship with the address class. You will also see the methods of each class listed in the lower portion of each class model.

## 

## UI Design

I have included below a low-fidelity wireframe of the entirety of the application. It shows the general design of each window and the relationships between the pages. I have also included a high-fidelity prototype design of the main window of the delivered application.

Diagram

Description automatically generated with medium confidence

Figure 1: Low Fidelity Application Design

Graphical user interface

Description automatically generated with medium confidence

Figure 2: High Fidelity Prototype of the main window of the application

# Unit Test Plan

### Purpose

One of the primary reasons that Ace Automotive wanted this application developed is due to the common issue of being double booked. Due to this, we have developed several unit tests to ensure that the “overlap” method is working as intended using the NUnit testing platform.

### Overview

The overlap method is used before adding any appointment to our database. The overlap method validates that there are no appointments currently in the database that would collide with the appointment being added for the user assigned to the appointment. The overlap method will return true if there is an overlapping appointment present in the database and false if there is not.

## Test Plan

### Items

Before we can run the test we need to create two mock appointments both with the same user selected. The first appointment will be the one we are trying to insert, and the second will be an appointment that we add to the database to compare against. For our first test, we will create two appointments that overlap. For our second test, we will create two appointments that do not overlap. For our final test, we will create two appointments that do not overlap but are within one minute of each other to show that the overlap method is working as designed.

### Features

For my unit testing, I used the assert feature of NUnit testing to assert that the overlap method returned true if there were overlapping appointments and it returned false if there were no overlapping appointments.

### Deliverables

Using NUnit for Visual Studio I can run each test and it will return either a pass or fail result through the test explorer.

### Tasks

To execute unit testing through NUnit I had to do the following:

1. Download the necessary NuGet packages.
2. Add an NUnit test project to the solution.
3. Write the unit testing code.
4. Run the tests.
5. Ensure the test output all returned as pass.

### Needs

The following NuGet packages had to be installed on Visual Studio to run the unit tests:

1. Microsoft.NET.Test.Sdk
2. NUnit
3. NUnit.Console
4. NUnit.ConsoleRunner
5. NUnit3TestAdapter
6. MySql.Data.EntityFrameworkCore

The unit tests also required the application’s database to be running for the duration of the unit tests.

### Pass/Fail Criteria

The first test is a test of two overlapping appointments and for it to pass the overlap method must return as two. The second test is a test of two non-overlapping appointments and for it to pass it must return false. The final test is also a test of two non-overlapping appointments and for it to pass it must also return false.

## Specifications

I have included a screenshot of the test code below. It shows all three tests, but the entirety of the unit test project can be found in the solution provided with the source code.

Text

Description automatically generated

## Procedures

The first test is a test of overlapping appointments and should return true. To achieve this, I add two appointments using the same user that are overlapping. Then, I add the second appointment to the database and use the overlap method on the first appointment. Since these appointments are overlapping the bool overlap is set to true. Then, I delete the test appointment just added to the database to ensure we aren’t filling the database with test data. Finally, I use the assert function of NUnit testing to ensure that overlap returns as true.

The second test is a test of two non-overlapping appointments and should return false. To achieve this, I add two appointments using the same user that do not overlap. Then, I add the second appointment to the database and use the overlap method on the first appointment. Since these appointments are not overlapping the bool overlap is false. Then, I delete the test appointment just added to the database to ensure we aren’t filling the database with test data. Finally, I use the assert function of NUnit testing to ensure that overlap returns as false.

The third test is just the same as the second test the only difference is we are testing with two appointments that are only a minute apart.

## Results

The following test results were returned through the use of the test explorer in Visual Studio. As you can see it lists the outcomes as “3 passed” with three tests in the group asserting that all unit tests have passed.

A screenshot of a computer

Description automatically generated with medium confidence

# C4. Source Code

The source code and executable file have been submitted in a separate folder.

# User Guide

## Installation

Before our application will be able to be installed and run we must install MySQL Workbench 8.0. To install MySQL Workbench 8.0 please follow the installation guidelines found at <https://dev.mysql.com/doc/workbench/en/wb-installing.html>. When prompted for a password during installation enter “Passw0rd!” with no quotations. Once you have completed the installation launch MySQL Workbench and follow the following steps. (Skip the first three steps if using WGU’s Virtual Lab Environment)

1. Click the + button next to MySQL Connections

Graphical user interface, text, application, chat or text message

Description automatically generated

1. Enter the following information:

Connection Name: CapstoneDatabase

Hostname: localhost

Port: 3306

Username: root

Password: click “Store in Vault…” then enter “Passw0rd!” with no quotations and click okay.

Default Schema: Leave this blank.

1. Click “OK”
2. Double-click the box stating “SQLInstall”
3. Click File and from the drop-down menu click “Open SQL Script”.
4. Navigate to the installation folder and locate the file titled

“SQLInstall.sql” and click “Open”.

1. Click the lightning bolt pictured to execute the MySQL script.

Graphical user interface, text, application, email

Description automatically generated

Your MySQL server should now be set up and ready to use. If you have any issues please contact customer support at (911)-867-5309 or email us at msenn3@wgu.edu.

## *Login and Accounts*

1. *There are two accounts set up for use by Ace Automotive. User 1 has a username of “admin” and a password of “password”, user 2 has a username of “test” and a password of “test”.*
2. *To log in simply enter your username in the “Enter Username” field and your designated password in the “Enter Password” field and click the “Login” button.*
3. *If you have any appointments scheduled within the next 15 minutes you will be notified upon logging in to the application.*
4. *There is also a login.txt file that logs all login attempts and the time that the login was attempted located in the main folder of the program.*

## *Customers*

### *Add a Customer Record*

1. *Once logged in you will see two main sections of the program. On the bottom half, you have your customer information section.*

*Graphical user interface, table

Description automatically generated*

1. *To add a customer, click on the “Add” button in the customer information section.*
2. *You will now see the “Add Customer” form with fields “Customer Name” which is the customer’s name, “Address” which is the customer’s address, “City” which is the customer’s city, “State” which is the customer’s state, “Zipcode” which is the customer’s zip code, “Phone Number” which is the customer’s phone number including the area code, and “Active” which denotes whether or not the customer is an active customer.*
3. *All fields must be filled out to complete the addition of the customer record.*
4. *Once all fields have been filled out click “Save” to add the customer record to the database or “Exit” to close the “Add Customer” form.*

### *Modify an Existing Customer Record*

1. *To modify a customer record, select the customer record you wish to modify in the grid view then click on the modify button in the customer information section of the program. The selected record will be highlighted in blue.Graphical user interface, table

   Description automatically generated*
2. *You will now see the “Modify Customer” form with fields “Customer ID” which is the unique ID associated with that customer and cannot be changed, “Customer Name” which is the customer’s name, “Address” which is the customer’s address, “City” which is the customer’s city, “State” which is the customer’s state, “Zipcode” which is the customer’s zip code, “Phone Number” which is the customer’s phone number including the area code, and “Active” which denotes whether or not the customer is an active customer.*
3. *All fields must be filled out to complete the modification of the customer record.*
4. *Once all fields have been filled out click “Save” to modify the customer record or “Exit” to close the “Modify Customer” form.*

### *Deleting a Customer Record*

1. *To delete a customer, select the customer record you wish to delete from the grid view then click on the “Delete” button located in the customer information section of the application. The selected record will be highlighted in blue.*

*Table

Description automatically generated*

1. *A warning will pop up to ask, “Are you sure you want to delete this item?”.*
2. *To continue with the deletion of the selected customer record click “OK” but use caution as this cannot be undone. To cancel the deletion of the selected customer record simply click “Cancel”.*

### *Searching for Customers*

1. *To search for customers in the database navigate to the customer information section of the application and click the field that states, “Search Customers”.*
2. *The first way to search is by the customer’s name which returns all results from the “Customer Name” column matching your search.*
3. *The second way to search is by the customer’s phone number which returns all results from the “Phone Number” column matching your search. Be aware that the search must match the layout of the customer record such as “(904) 898-2231” including all special characters.*

## *Appointments*

### *Add an Appointment Record*

1. *Once logged in you will see two main sections of the program. On the upper half, you have your appointment information section.*

*Table

Description automatically generated*

1. *To add an appointment record, click the “Add” button in the appointment information section.*
2. *You will now see the “Add Appointment” form with fields “Customer ID” which is the customer’s unique ID, “User ID” which is the user ID associated with the appointment, “Vehicle” which is the customer’s vehicle that the appointment is for, “Job Type” which is the type of job the appointment is for, “Appointment” which is the customer’s time of appointment, and “Due By” which is the time the customer expects their vehicle to be finished.*
3. *All fields must be filled out to complete the addition of the appointment record.*
4. *The “Customer ID” field must match a customer ID record in the customer information section.*
5. *The “User ID” field must match an active user ID noted previously in the “Login and Accounts” section of the user guide.*
6. *The Appointment must be made during business hours (9 AM-5 PM) and cannot overlap with existing appointments.*
7. *Once all fields have been filled out click “Save” to add the appointment record to the database or “Exit” to close the “Add Appointment” form.*

### *Modify an Appointment Record*

1. *To modify an appointment record, click the “Modify” button in the appointment information section.*
2. *You will now see the “Modify Appointment” form with fields “Appointment ID” which is the appointment’s unique ID and cannot be changed, “Customer ID” which is the customer’s unique ID, “User ID” which is the user ID associated with the appointment, “Vehicle” which is the customer’s vehicle that the appointment is for, “Job Type” which is the type of job the appointment is for, “Appointment” which is the customer’s time of appointment, and “Due By” which is the time the customer expects their vehicle to be finished.*
3. *All fields must be filled out to complete the modification of the appointment record.*
4. *The “Customer ID” field must match a customer ID record in the customer information section.*
5. *The “User ID” field must match an active user ID noted previously in the “Login and Accounts” section of the user guide.*
6. *The Appointment must be made during business hours (9 AM-5 PM) and cannot overlap with existing appointments.*
7. *Once all fields have been filled out click “Save” to modify the appointment record or “Exit” to close the “Modify Appointment” form.*

### *Deleting an Appointment Record*

1. *To delete an appointment, select the appointment record you wish to delete from the grid view then click on the “Delete” button located in the appointment information section of the application. The selected record will be highlighted in blue.*
2. *A warning will pop up to ask, “Are you sure you want to delete this item?”.*
3. *To continue with the deletion of the selected appointment record click “OK” but use caution as this cannot be undone. To cancel the deletion of the selected appointment record simply click “Cancel”.*

### *Searching for Appointments*

1. *To search for appointments in the database navigate to the appointment information section of the application and click the field that states, “Search Appointments”.*
2. *The first way to search is by the customer’s name which returns all results from the “Customer Name” column matching your search.*
3. *The second way to search is by the vehicle which returns all results from the “Vehicle” column matching your search.*
4. *The third and final way to search is by the job type which returns all results from the “Type” column matching your search.*

## *View Calendar and Reports*

### *View Appointment Calendar*

1. *To access the appointment calendar, click on the “View Calendar” button on the right-hand side of the main page.*
2. *By default, all appointments are shown but you also have the option to click the buttons for “Weekly Appointments” and “Monthly Appointments” to show the corresponding calendars.*

### *View Job Type Report*

1. *To access the job type report, click on the “Job Type Report” button on the right-hand side of the main page.*
2. *This report will list the types of jobs scheduled for the selected month and how many appointments with that job type are in the selected month.*

### *View Scheduling Report*

1. *To access the scheduling report, click on the “Scheduling Report” button on the right-hand side of the main page.*
2. *This report will list all appointments by the user selected in the drop-down list.*

### *View Customer Report*

1. *To access the customer report, click on the “Customer Report” button on the right-hand side of the main page.*
2. *This report will list all appointments by the customer selected in the drop-down list.*