**C868 – Software Capstone Project Summary**

**Task 2 – Section C**

|  |  |
| --- | --- |
| **Capstone Proposal Project Name:** | http://www.idevnews.com/views/images/uploads/general/wgu_logo.png  Scheduling Application |
| **Student Name:** | Isam Elder |

Contents

[Task 2 Part C – C868 Software Development Capstone 3](#_Toc132125029)

[Application Design and Testing 3](#_Toc132125030)

[Design Document 3](#_Toc132125031)

[Class Design 3](#_Toc132125032)

[UI Design 4](#_Toc132125033)

[Unit Test Plan 5](#_Toc132125034)

[Introduction 5](#_Toc132125035)

[Purpose 5](#_Toc132125036)

[Overview 6](#_Toc132125037)

[Test Plan 6](#_Toc132125038)

[Items 6](#_Toc132125039)

[Features 6](#_Toc132125040)

[Deliverables 6](#_Toc132125041)

[Tasks 6](#_Toc132125042)

[Needs 7](#_Toc132125043)

[Pass/Fail Criteria 7](#_Toc132125044)

[Specifications 7](#_Toc132125045)

[Procedures 7](#_Toc132125046)

[Results 8](#_Toc132125047)

[C4. Source Code 8](#_Toc132125048)

[C5. Link to Live Version 8](#_Toc132125049)

[User Guide 10](#_Toc132125050)

[Introduction 10](#_Toc132125051)

[Installation and Using the Application 10](#_Toc132125052)

[*Login and Exit* 10](#_Toc132125053)

[*Customer* 11](#_Toc132125054)

[*Appointment* 16](#_Toc132125055)

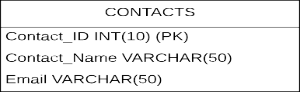
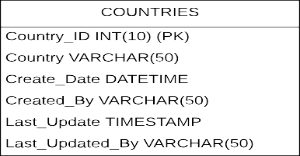
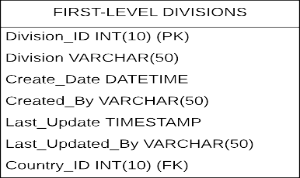
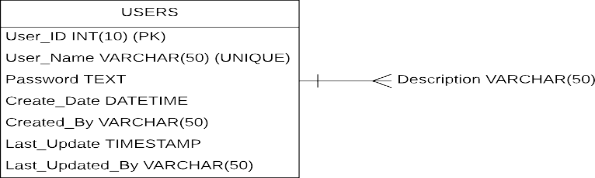
*Report*  [17](#_Toc132125056)

Task 2 Part C – C868 Software Development Capstone

# Application Design and Testing

# Design Document

## Class Design



The class setup for the customer application is shown in the illustration below. Because the Java framework employed in this program supports the Schema Console architecture, I have opted to demonstrate the design classes because they immediately correspond to the source code that comprise the software's foundation.

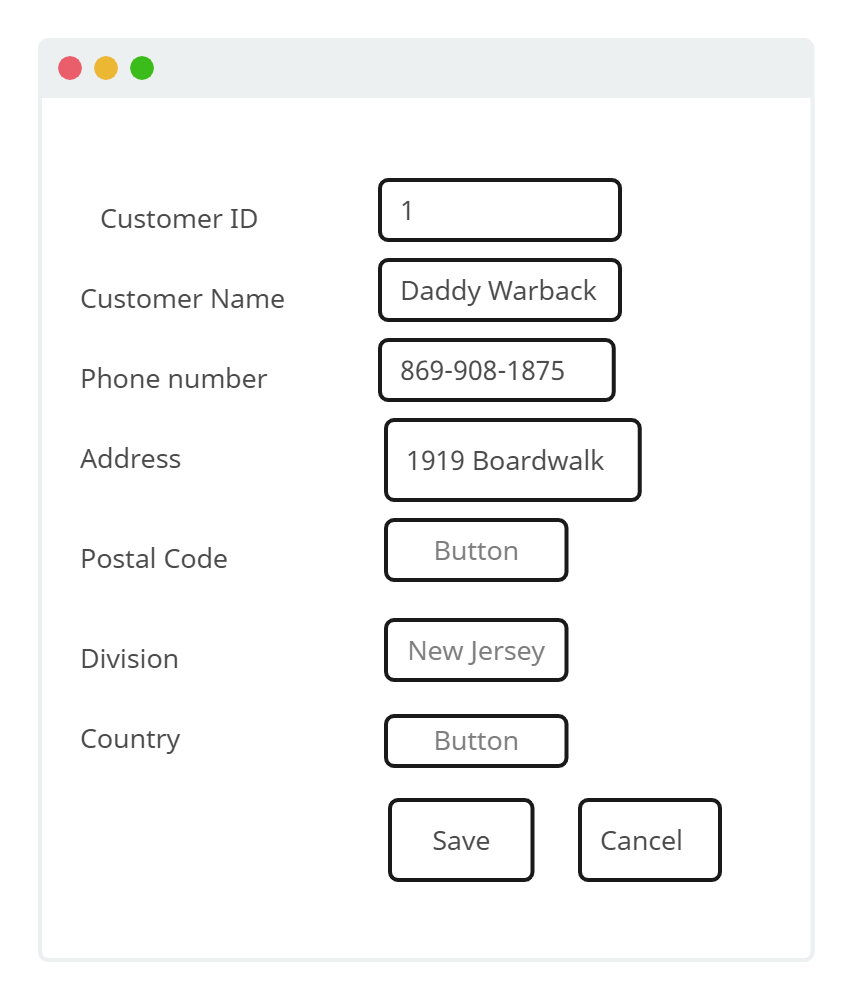
Report, Customer, and Appointments are the three basic parameter classes. Customer and Appointments have a one-to-many connections, as do Report and Appointments. Appointments, Customer (an appointment must correspond to one and only one customer), Appointments, and Report, on the other hand, must have a several association (Bennett & McRobb, 2010). Customer and address have a symmetric one-to-one association. The Address class in Java contains two parameters that demonstrate its versatile character: addressable id (which holds the id of the customer) and the type of address (for the name of the class, customer). A Customer class, which is used to generate administrator profiles, cannot be seen.

Underlining indicates class functions. One instance is the Customer class function age min (integer), which produces an object of clients with requirements provided by the method's parameter. A minus symbol denotes private techniques (-). For example, the Appointment's unique analytical procedure ensures there is no overlap for appointment booked() that almost no additional consultations is incompatibility with the prevailing appointment.

## UI Design

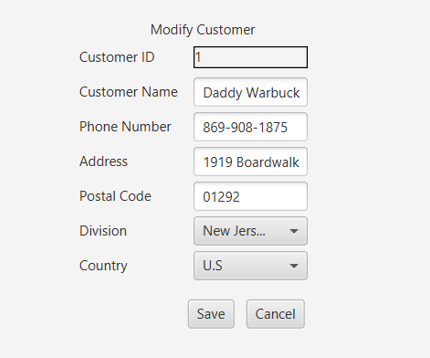
**Low-fidelity version:**

The low-fidelity version of the detail page for booking a customer appointment is a rough sketch of the page's layout. It shows the basic structure and content of the page without much design or detail. In this version, you can expect to see simple boxes and labels for customer ID, name, phone number, address, postal code, division, and country. The layout will likely be basic and unpolished, with minimal color and design elements (Bennett & McRobb, 2010). The focus will be on displaying the necessary information and functionality in the simplest way possible.



**High-fidelity version:**

The high-fidelity version of the detail page for booking a customer appointment is a fully-designed prototype that showcases a more polished version of the page. It includes detailed design elements such as typography, color schemes, images, and graphics (Bennett & McRobb, 2010). In this version, you can expect to see a more visually appealing layout that enhances the user experience. You can expect to see features such as drop-down menus or auto-complete text fields for quicker data input, along with well-organized information sections that make the data easy to understand and interact with. Overall, while the low-fidelity version prioritizes function over design, the high-fidelity version focuses on both function and form to create a delightful user experience.



# Unit Test Plan

## Introduction

### Purpose

To check the Appointment class validations, I crafted unit tests to ensure their accuracy using a JavaScript testing library. As the appointment class has significant importance in the app, I sought to confirm that no flawed appointment is saved to the database. The possibility of an invalid appointment can arise in multiple ways; however, my current focus is to guarantee non-duplication of appointments booked by a customer. While JavaScript' validations mechanically execute when creating or updating records, these unit tests ensure that the proper validation technique is implemented to discontinue coinciding appointments with the same customer from being persisted.

### Overview

As the application revolves around appointment tracking, it was vital to meticulously consider the logic behind their creation and preservation (Dennis et al., 2018). Validations are a set of methods that ensure if a record adheres to the pre-defined validation regulations before saving it. Once they are triggered right before data saving, any violation would produce certain error characteristics causing records to fail persisting to the database. The objective of this unit test is to confirm whether the implemented validation regulation can effectively prevent the inadvertent double booking of appointments by a doctor. Similar validation tests for deterring double booking of patients are also in place (Dennis et al., 2018). This unit test employs a specific JavaScript testing framework to verify the appointment validation rule accurately via terminal messages indicating test success or failure.

## Test Plan

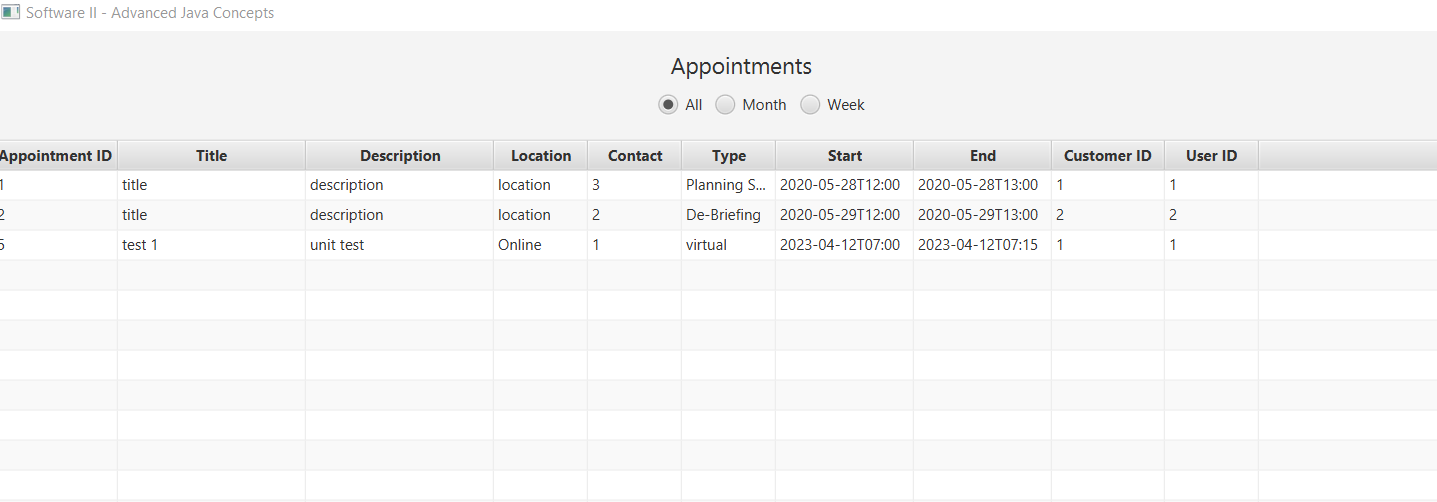
### Items

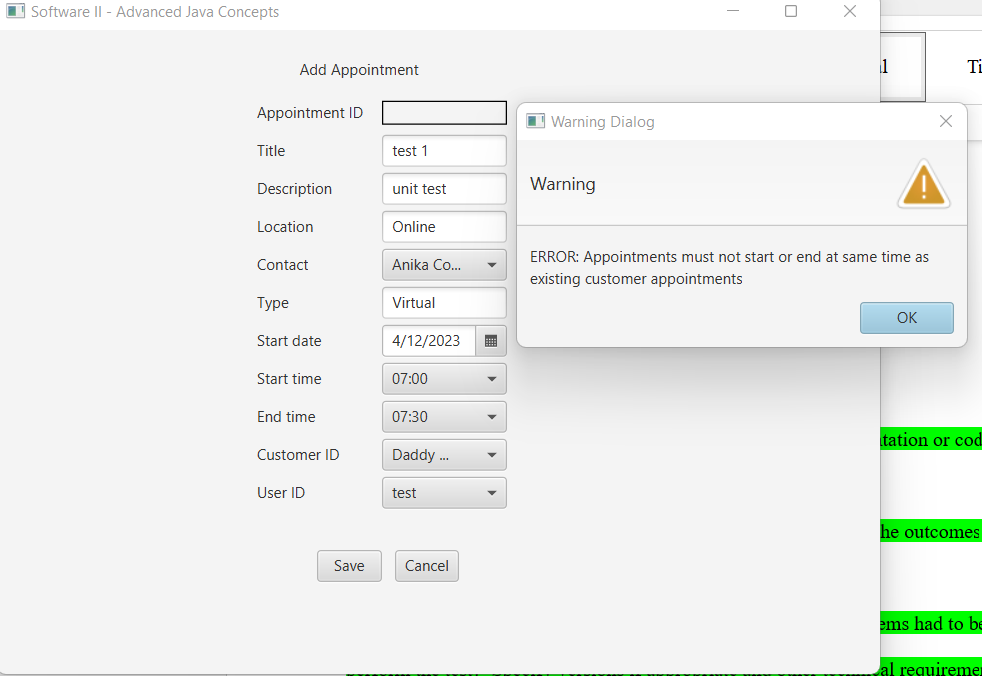
What is required to complete the test(s)?

The requirement to complete this test is that no appointments can be added that has the same time as an existing customer appointment.

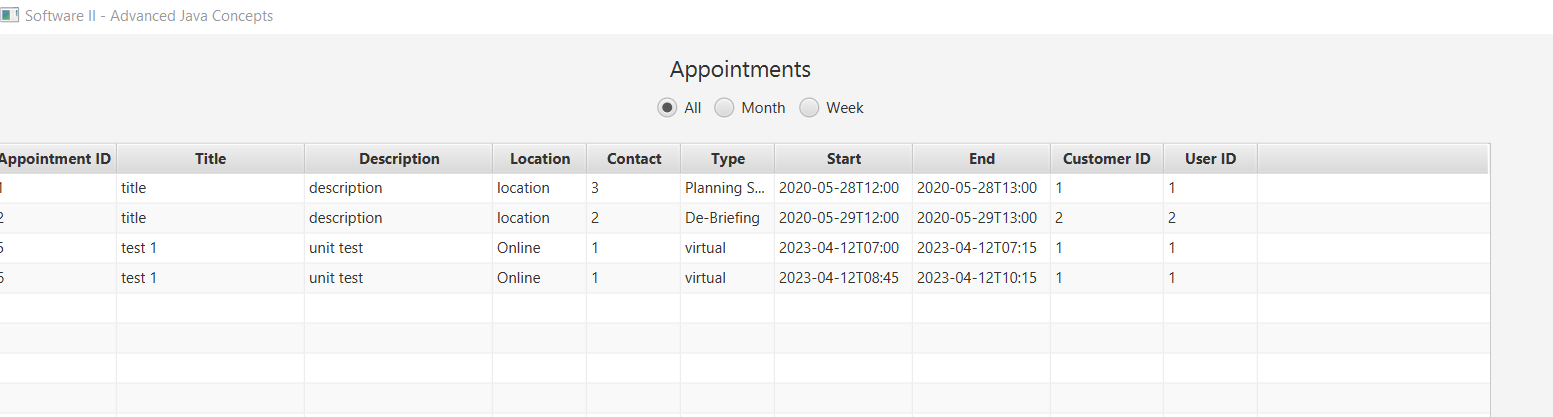
### Features

The first step in this test is to create an appointment and take note of the start an end time. The Image below represents addition of an appointment with title test 1.

To perform a unit test, we try to add a new appointment with the same start time. It throws the error in the picture below stating; Appointments must not start or end at same time as existing customer appointments.

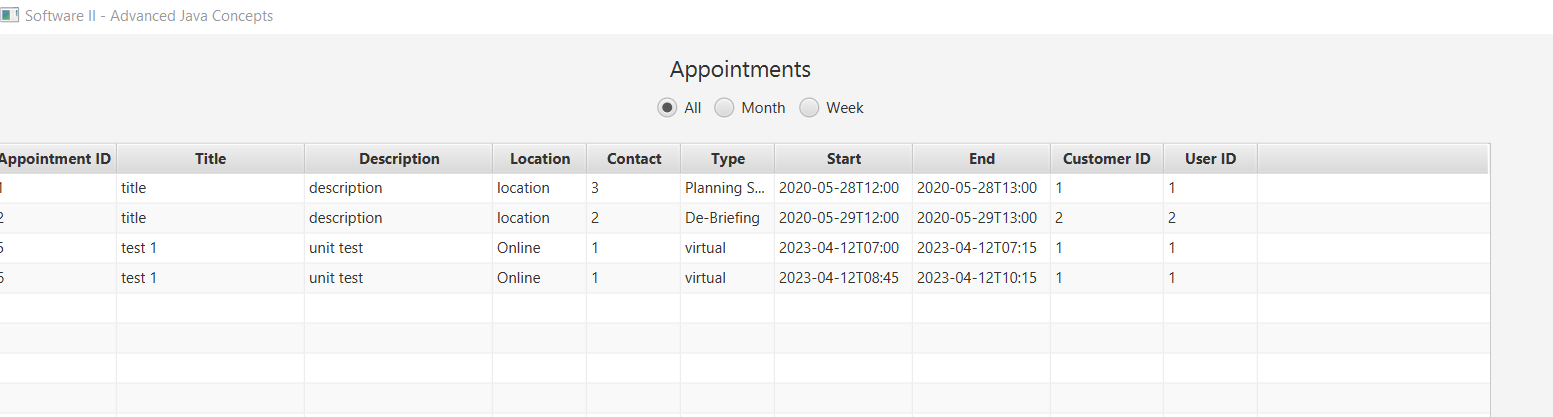


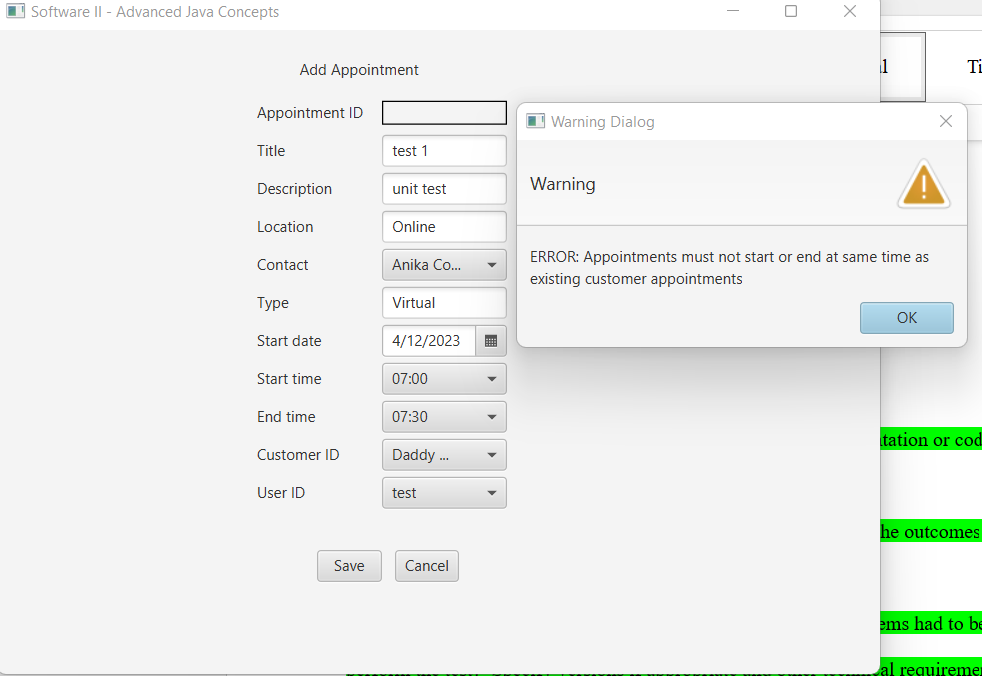
The third unit test is to make an appointment with the same information but different start and end time as the first appointment. No error is produced and the new appointment is successfully added to the table as show below.



### Deliverables

The test produces two options; an error or a save successful in the database. Below are the two deliverables’ screenshots.





### Tasks

Three tasks are required to complete these unit test.

1. Add an appointment taking note of the start and end time.
2. Add a second appointment with same details and also same start and end time as the first appointment.
3. Add a third appointment with same details as the first appointment but different start and end time.

### Needs

The following was needed in order to perform the unit test;

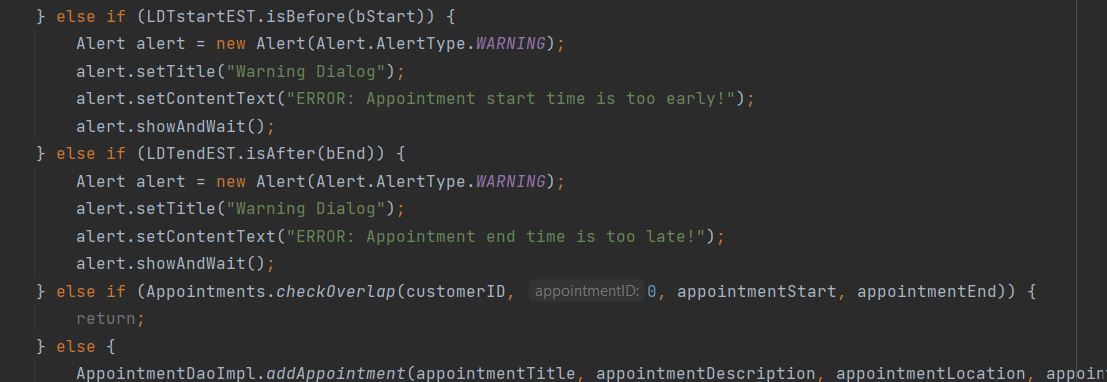
1. JetBrains IntelliJ IDEA Community Edition
2. Java SDK 17.0.1
3. JavaFX SDK 17.0.6sn

### Pass/Fail Criteria

The pass criteria of this test is that if an appointment is made with overlapping start and end time with another appointment then an error is produced and the appointment is not accepted. The fail criteria of this test is that an appointment is made with overlapping start and end time with another appointment and that it is successfully saved.

## Specifications

Provide sample code that represents what testing code was used. Screenshots are acceptable.

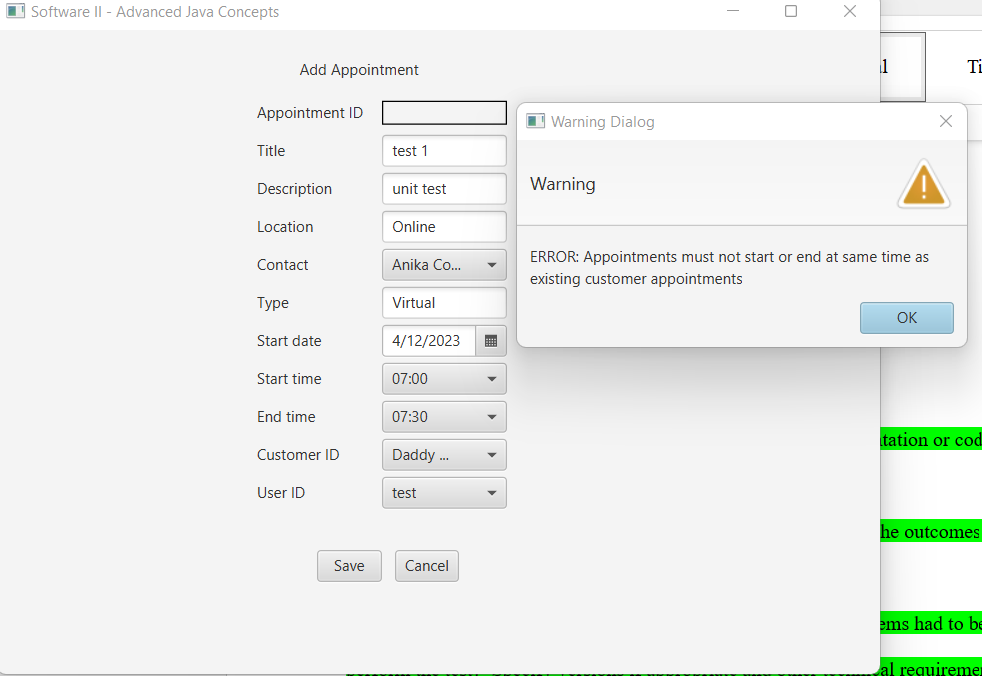


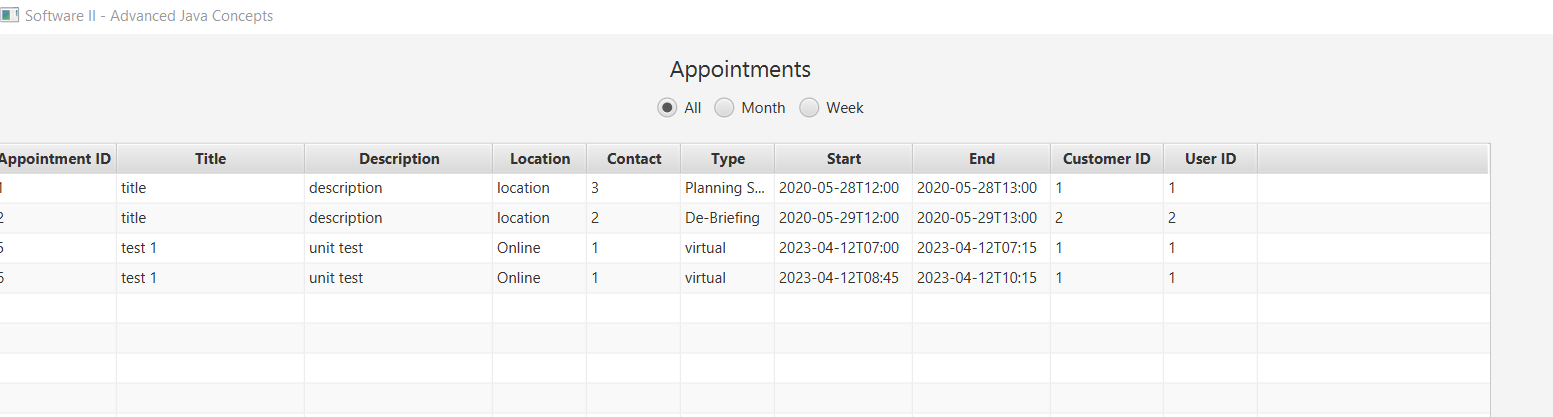
## Procedures

1. The first step in this test is to create an appointment and take note of the start an end time.
2. The second step is to add a new appointment with the same start time and end time as the first appointment.
3. The third unit test is to make an appointment with the same information but different start and end time as the first appointment.

## Results

Below are the screenshots of the results of the unit test.





# C4. Source Code

# The source code of this application is found on this zip file. Once you open the file in IntelliJ Idea you will find all the source code organized in an MVC architecture.

# C5. Link to Live Version

This application is a standalone application and does not have a link to a live website. The details on how to run the application is provided in the user guide section.

# User Guide

*Note: This may be included as a separate document if you desire.*

## Introduction

The following guide offer the user useful information on how to install the application, log into, sign up, and other critical function of the application.

## Installation and Using the Application

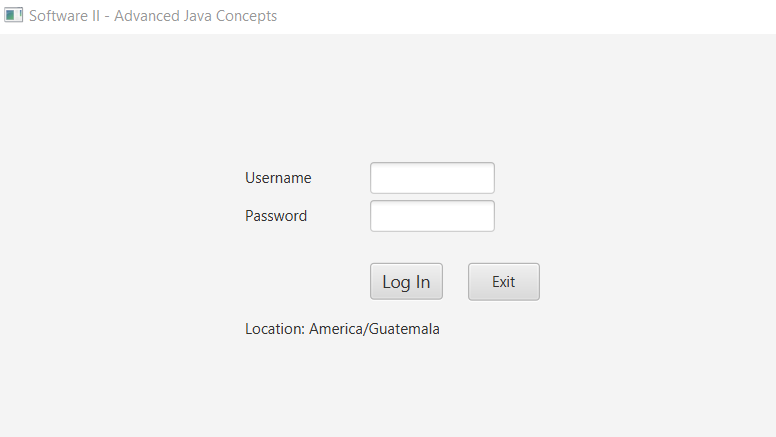
**Prerequisites**

1. JetBrains IntelliJ IDEA Community Edition
2. Java SDK 17.0.1
3. JavaFX SDK 17.0.6

**Installation**

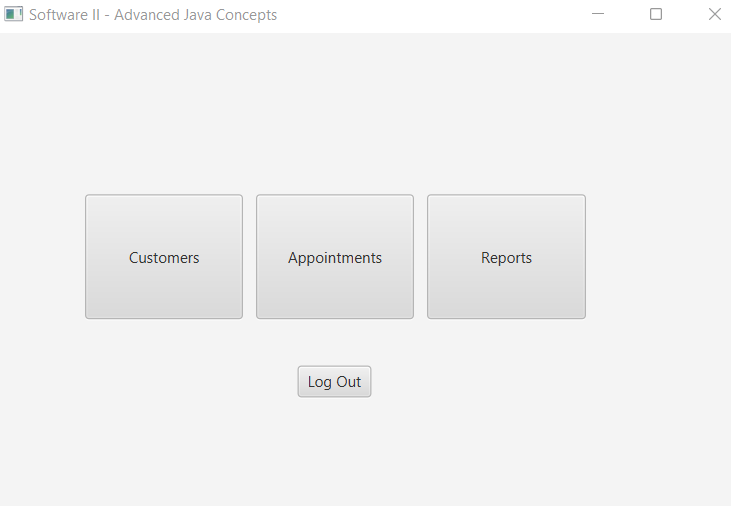
1. Open the folder/zip in IntelliJ and update the paths to your JavaFX SDK, JDK and JDBC
2. Launch the application.
3. Login with one of the following: username “test” and password “test” or username “admin” and password “admin”.

## *Login and Exit*



*Here is how to log in:*

1. *First enter the username and password in the provided fields.*
2. *On the landing page, click on the “Log In” button.*
3. *You will be logged in and taken to the main menu, which contains the customer, appointments, and report buttons.*

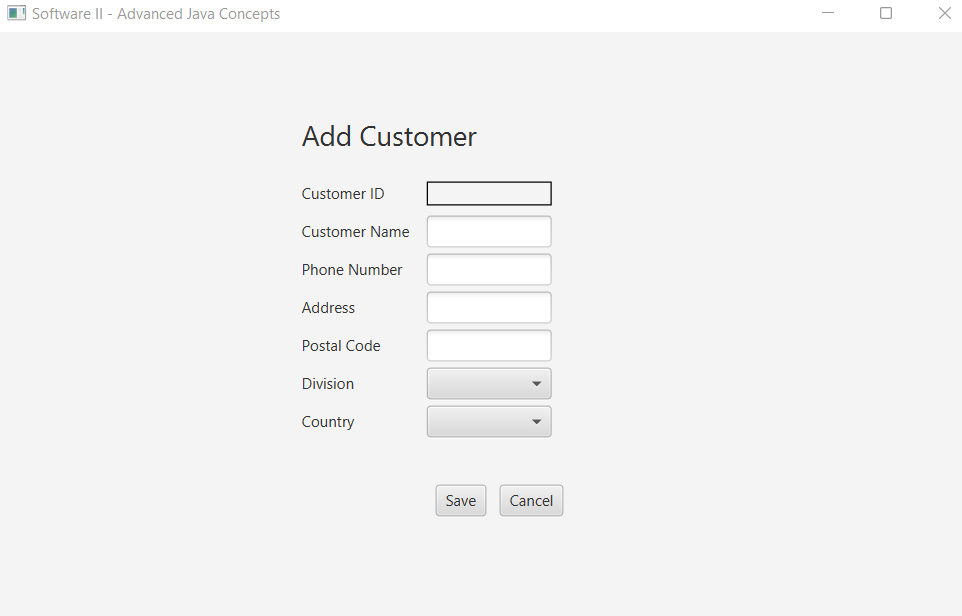


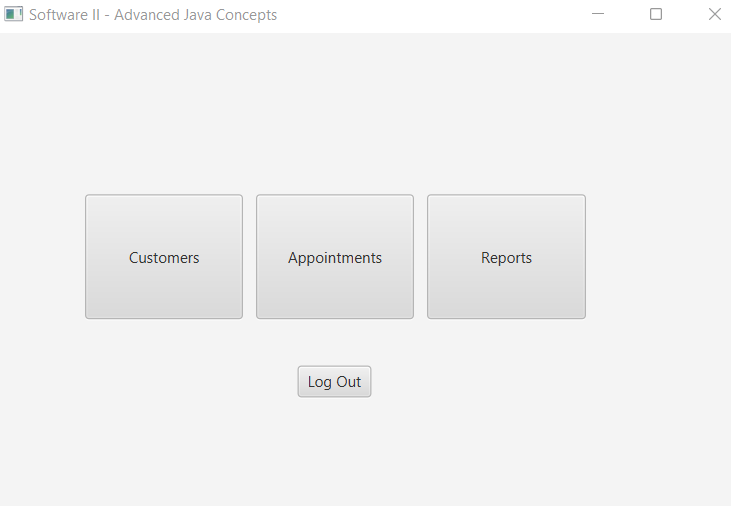
## *Customer*

1. You can use your login credentials to access the application and enter your before you book an appointment.
2. Enter your full name as it appears on their ID or government-issued document.

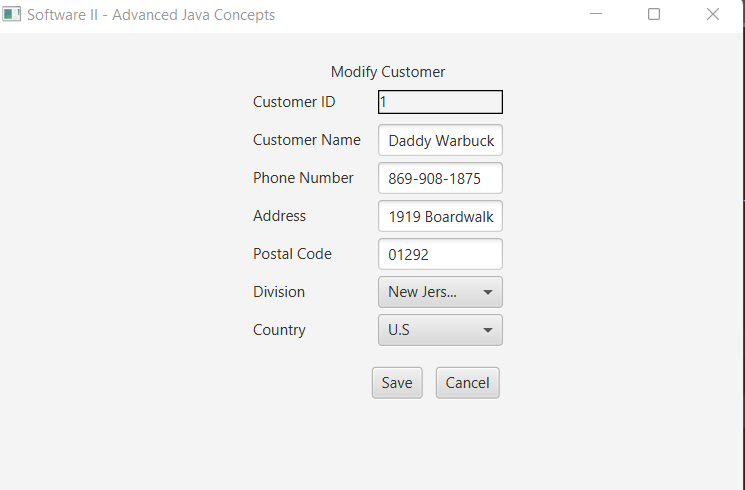
Enter the phone number:

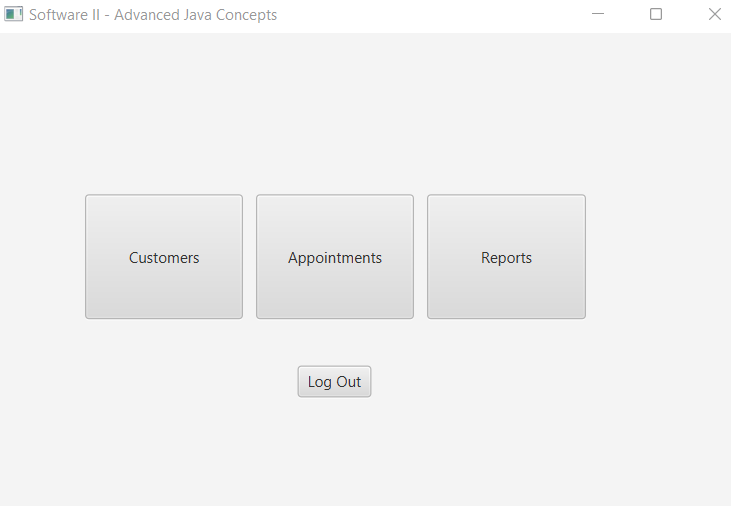
1. Enter the phone number of the primary contact that you will be reached on, making sure to include the country code if necessary.
2. Enter your full postal address, including the street address, city, state, and zip code.
3. Choose the appropriate division or department using the drop down option.
4. Choose the appropriate country area or region for your customer.
5. Once you have filled out all of the required fields, you can save and cancel

**

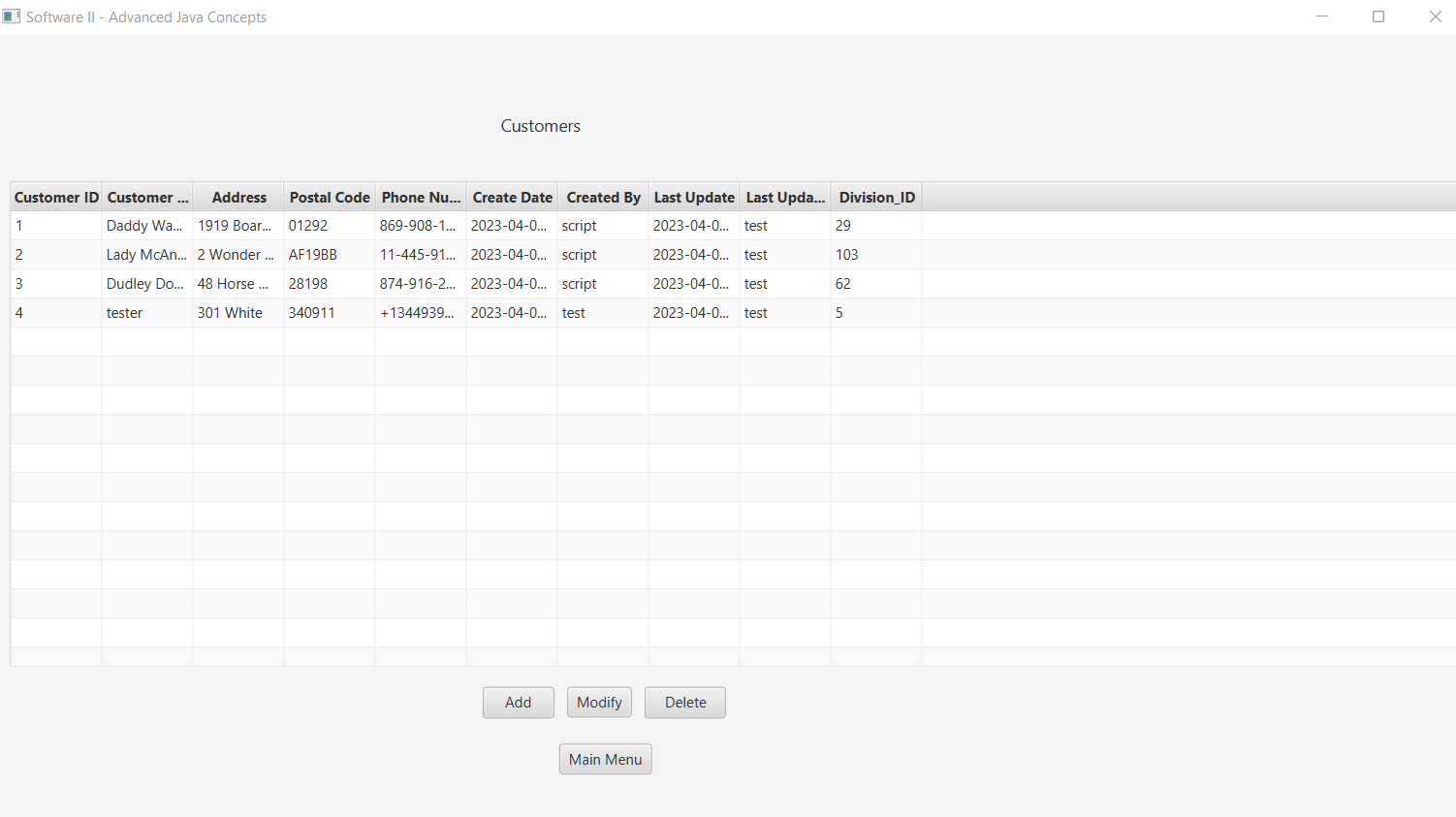
User can also modify customer details by clicking on the main menu, and selecting customers. 

You will be directed to modify customer details

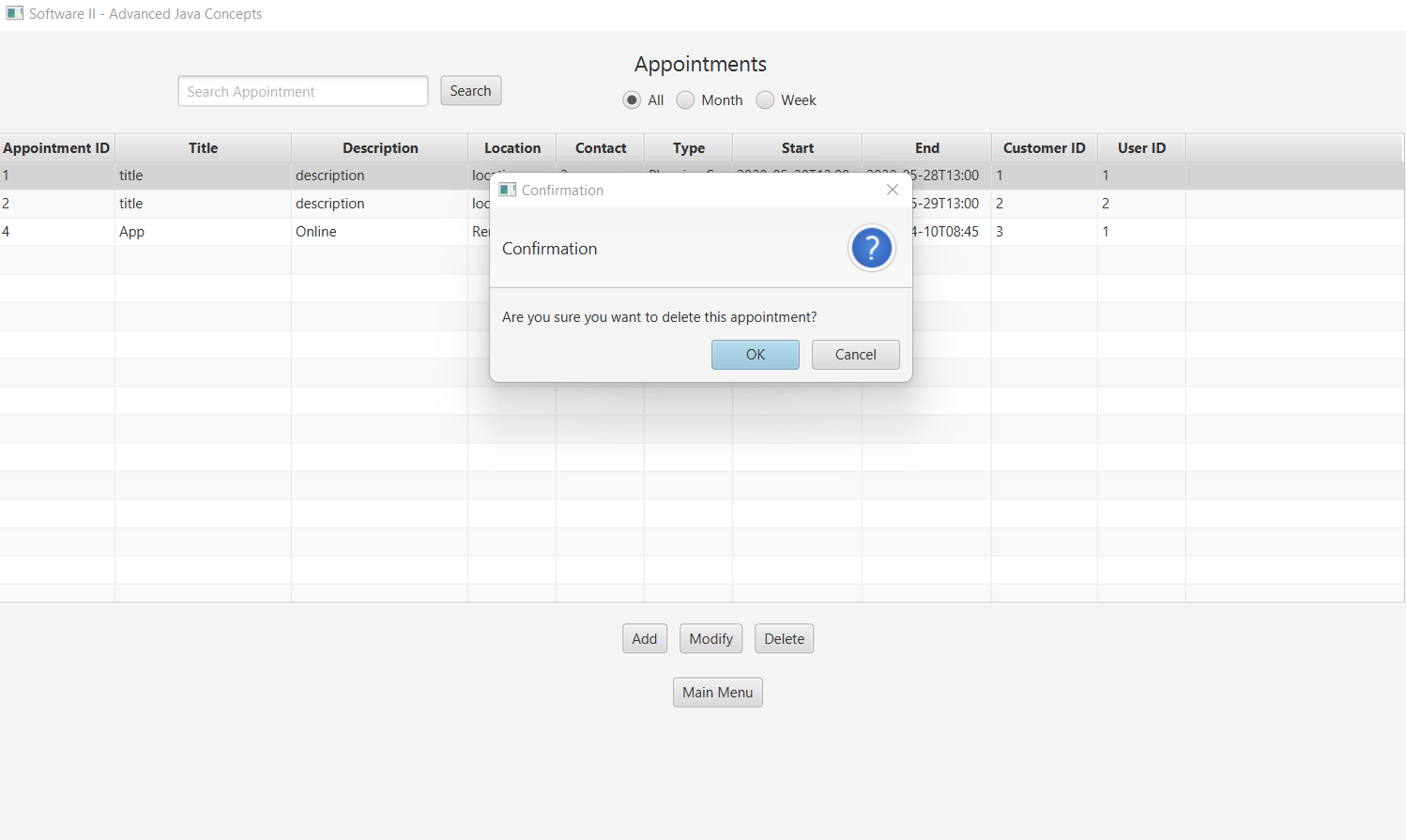


You can also check customers’ lists by going back to main menu can clicking customers 

You will be directed to a list of customers as shown below.



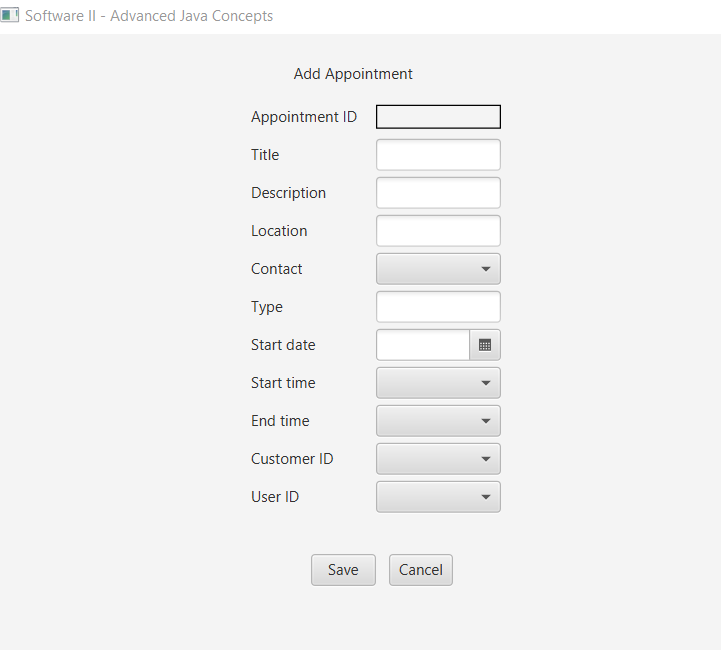
In the list you can also delete customers by marking then deleting down as shown in the arrow below



## *Appointment*

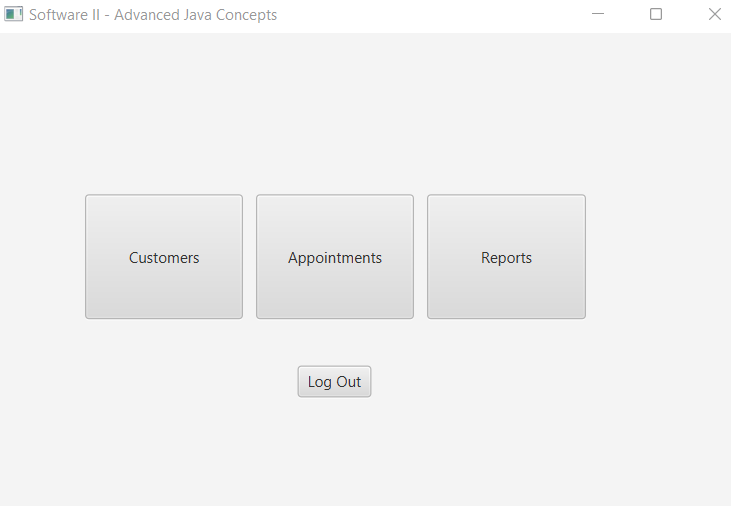
Here is a general outline for a users to adding appointments:

1. Launch the application and log in to your account.
2. Begin by adding appointment ID appointments.
3. Enter a new title for the appointment.
4. Add a description for the appointment that includes any relevant information about the purpose.
5. Enter a new location for the appointment either remote or online.
6. Enter the contact information of the person scheduling the appointment,
7. Modify the start date, start time, and end time of the appointment, as needed.
8. Enter the Customer ID number for the person who is attending the appointment, if applicable.
9. Enter the User ID number for the person who is making the modification, if needed.
10. Review the updated information and confirm the changes.
11. Save the changes to the system.

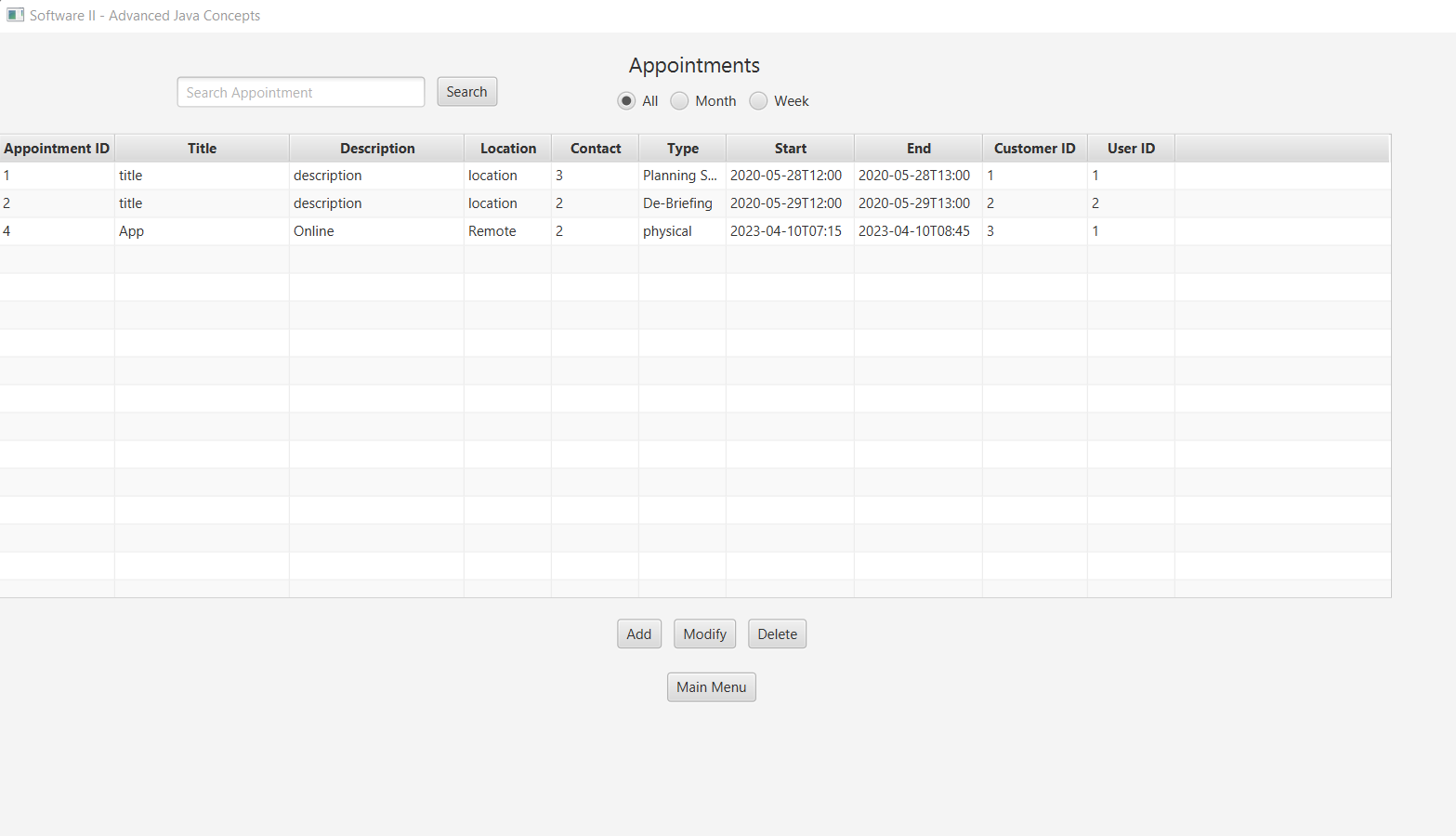


***Checking appointment***

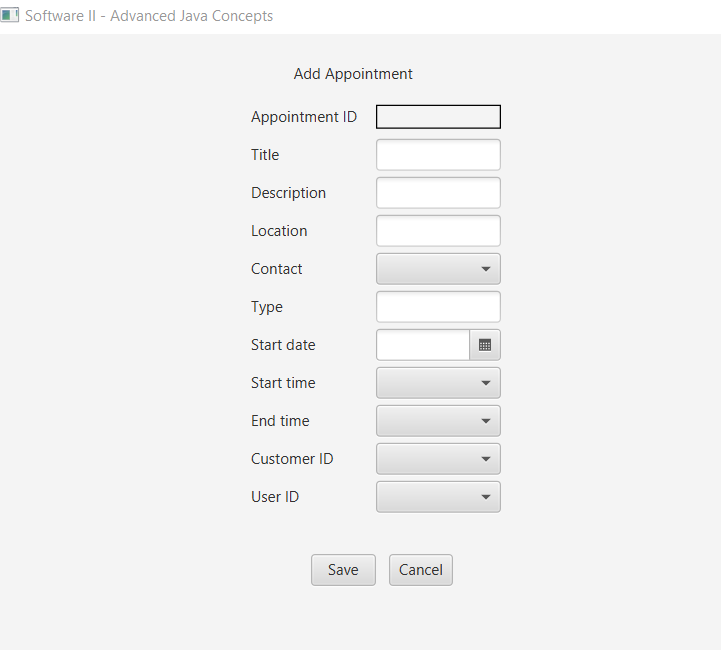
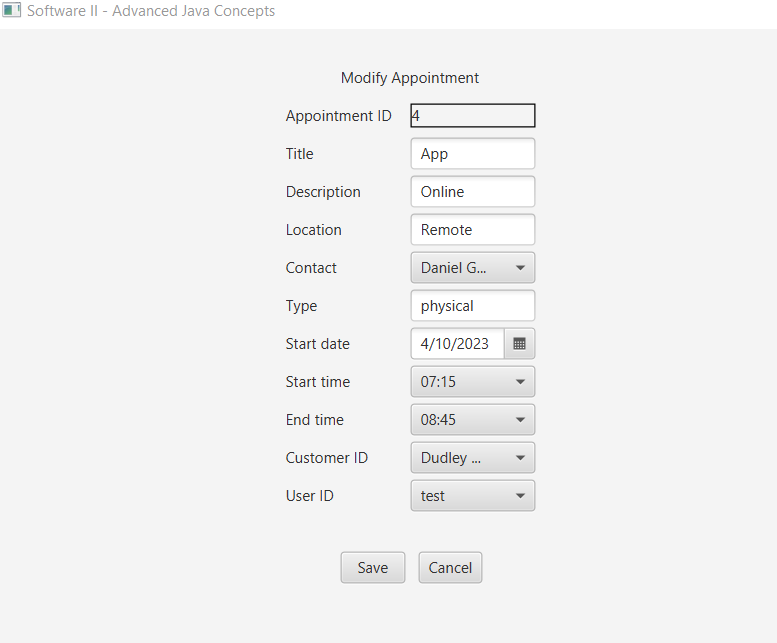
You can also check the list of appointment by clicking in the main menu the appointment which



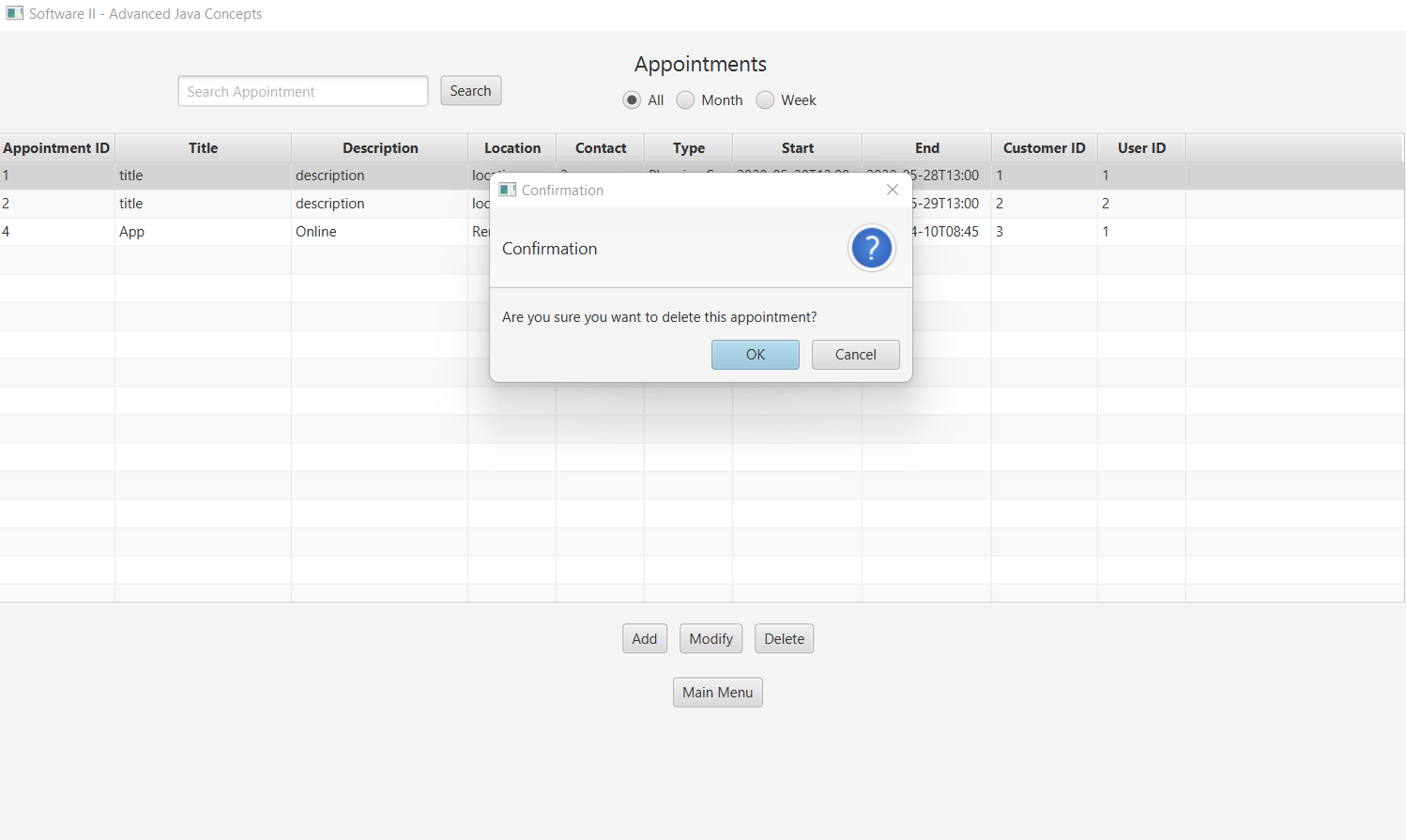
Then it will direct you to a list of appointment shown below.



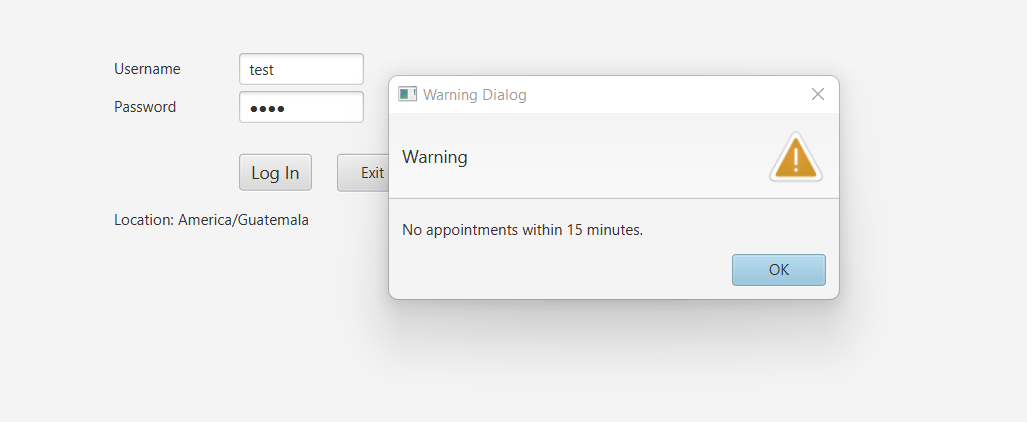
You can click on the appointment and modify the one you want as shown below.



You can also delete an appointment as shown below

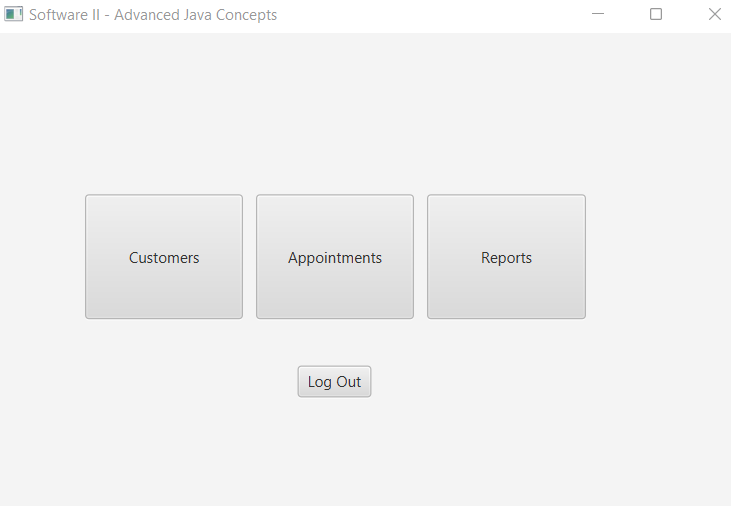


If you have no appointment, the application will return with a message shown below

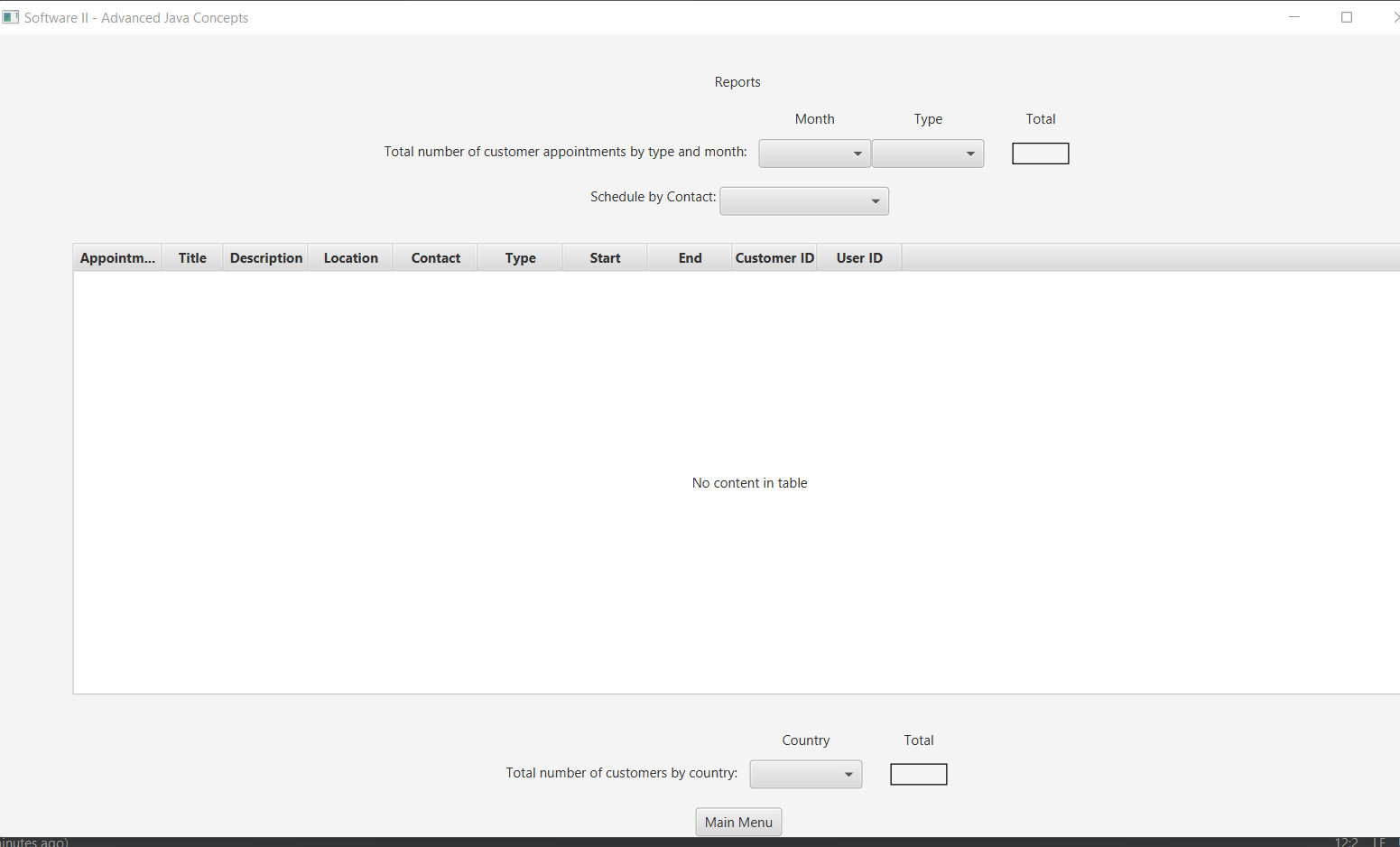


**Report**

You can generate the report for the customers and appointment booked after every session by clicking report in the main menu



You will be directed to reporst as shown below, where you view monthly report of customers including their description.



References

Bennett, S., & McRobb, S. (2010). Object-Oriented Systems Analysis and Design Using UML. McGraw-Hill Education.

Dennis, A., Wixom, B. H., & Roth, R. M. (2018). Systems Analysis and Design. John Wiley & Sons.