Java Database Scheduler

March 9, 2020

Juan Garcia

# Introduction

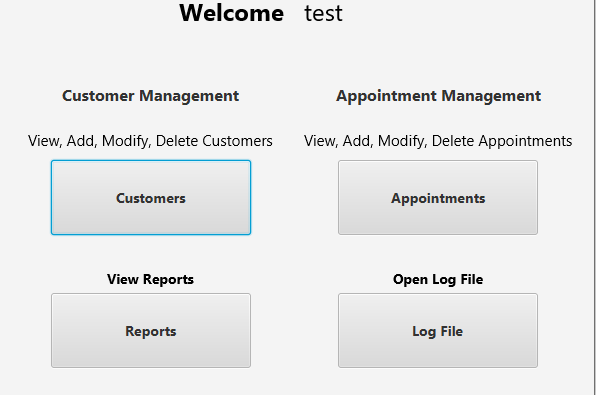
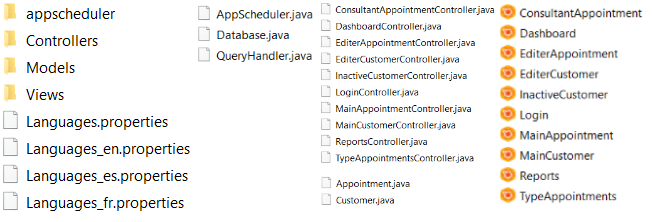
Throughout your career in software design and development, you will be asked to create applications with various features and criteria based on a variety of business requirements. For this assessment, you will create your own Java application with requirements that mirror those you will encounter in a real-world job assignment.

# Scenario

You are working for a software company that has been contracted to develop a scheduling desktop user interface application. The contract is with a global consulting organization that conducts business in multiple languages and has main offices in Phoenix, Arizona; New York, New York; and London, England. The consulting organization has provided a MySQL database that your application must pull data from. The database is used for other systems and therefore its structure cannot be modified.

The organization outlined specific business requirements that must be included as part of the application. From these requirements, a system analyst at your company created solution statements for you to implement in developing the application. These statements are listed in the requirements section.

## Dashboard Screen

**A.   Create a log-in form that can determine the user’s location and translate log-in and error control messages (e.g., “The username and password did not match.”) into two languages.**

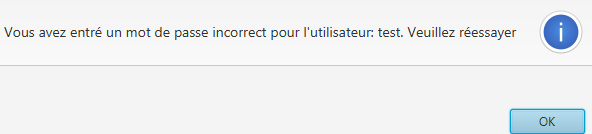
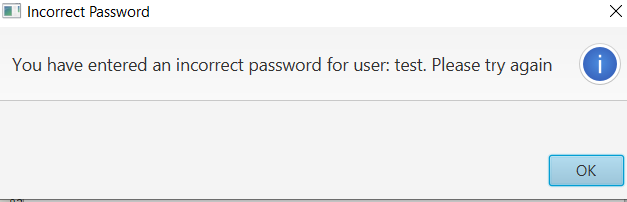
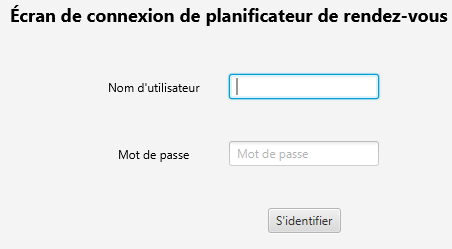
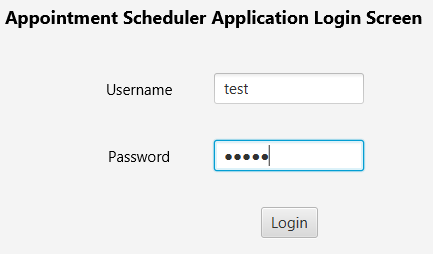
Controllers.LoginController.java ( Alert within group at line 44)

appscheduler.QueryHandler.java (line 36)

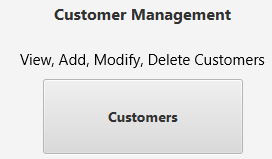
Languages.properties

Languages\_en.properties (English)

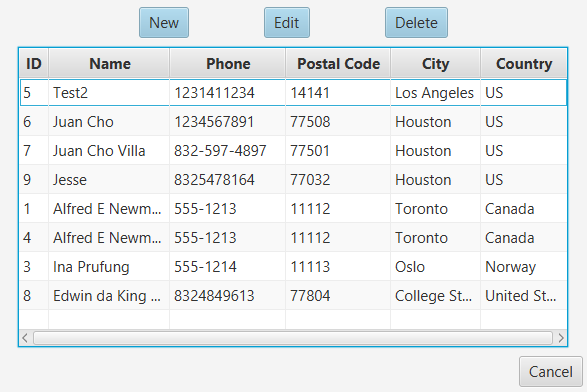
Languages.\_fr.properties (French)



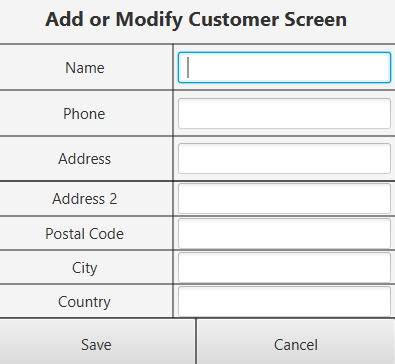
**B. Provide the ability to add, update, and delete customer records in the database, including name, address, and phone number.**



**Customer Management**



## ADD CUSTOMER

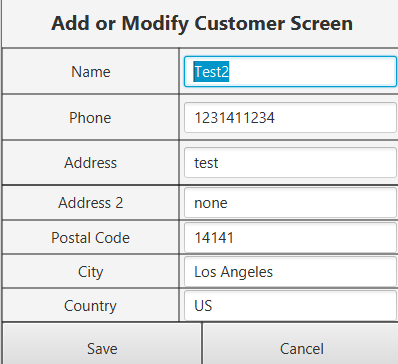
Add Customer option in Customer Management section: **New**

Controllers.EditerCustomerController.java

Models.Customer.java

Appscheduler.QueryHandler.java ( line **306**)

## Modify Customer

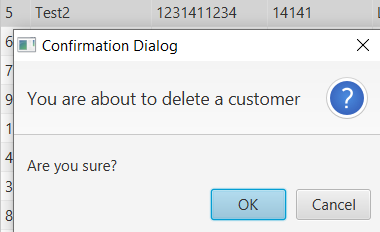
Edit Customer option in Customer Management section: **Edit**

Controllers.EditerCustomerController.java ( line 85)

Models.Customer.Java

Appscheduler.QueryHandler.java (Lines 126, 143, 189 )

## Delete Customer

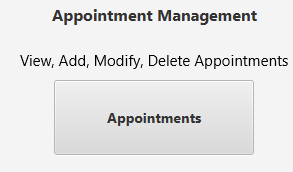
Delete Customer option in Customer Management section: **Delete**

Controllers.EditerCustomerController.java

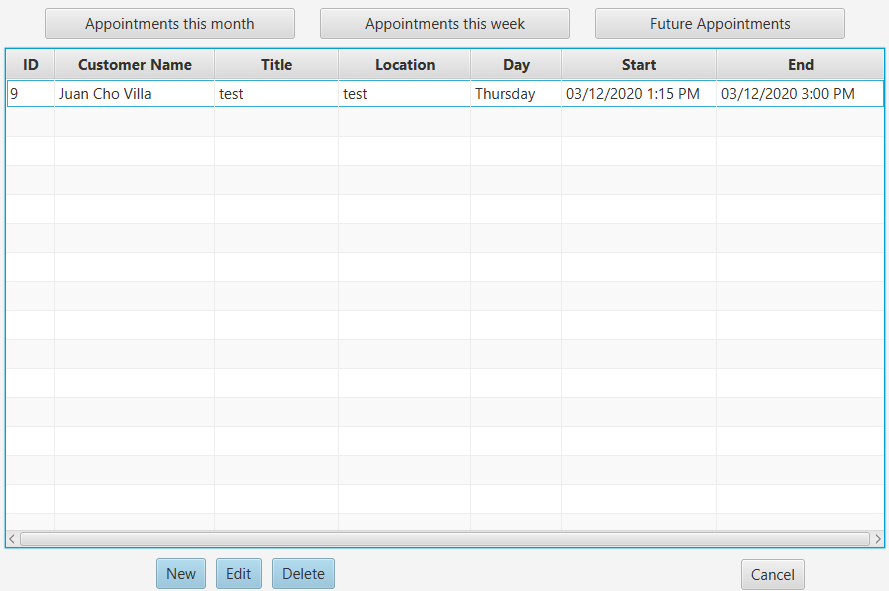
Models.Customer.Java

Appscheduler.QueryHandler.java ( line 111 )

**C.   Provide the ability to add, update, and delete appointments, capturing the type of appointment and a link to the specific customer record in the database.**

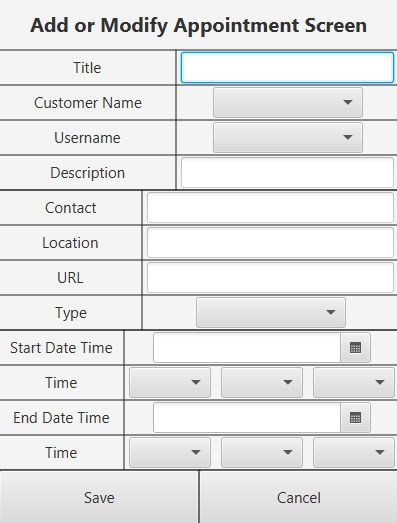


## Appointments Screen





## Add Appointment

New appointment option in Appointment Management section: **NEW**

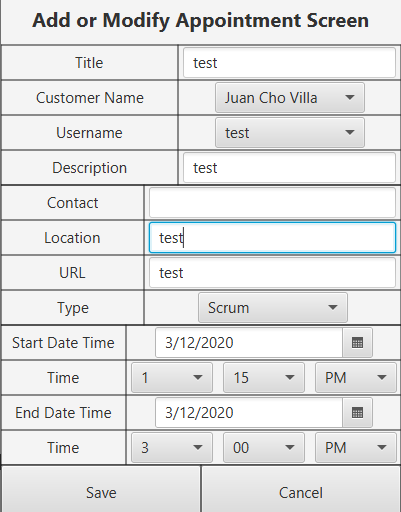
Controllers.EditerAppointmentController.Java

Appscheduler.QueryHandler.java (Line 539)

Models.Appointment.java

Models.Customer.java

## Modify Appointment

Edit appointment option in Appointment Management section: **Edit** Controllers.EditerAppointmentController.java

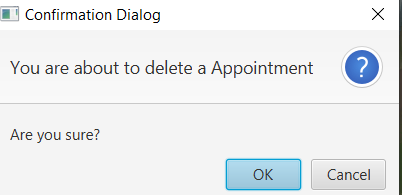
Appscheduler.QueryHandler.java (Line 561)

Models.Appointment.java

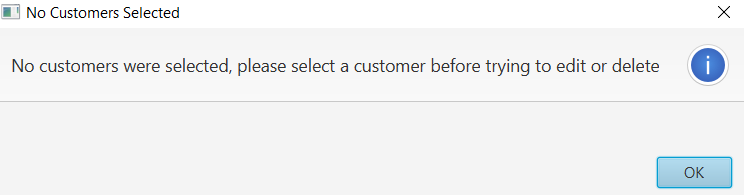
Models.Customer.java

Observable List initialized and getAppointmentTable() method from QueryHandler called to retrieve table. Tableview is then populated with function popApptTableView in MainAppointmentController.java starting at Line 163. Customer Linked by Id and Name.

## Delete Appointment

Delete appointment option in Appointment Management section: **Delete** Appscheduler.QueryHandler.java (Line 594)

Models.Appointment.java

Models.Customer.java

D.   Provide the ability to view the calendar



## by month

Client Dashboard option in the Appointments section “**Appointments this month**” (Displays all appointments in the next week for all consultants )

Appscheduler.QueryHandler.java (Line 609)

Controllers.MainAppointmentController.java

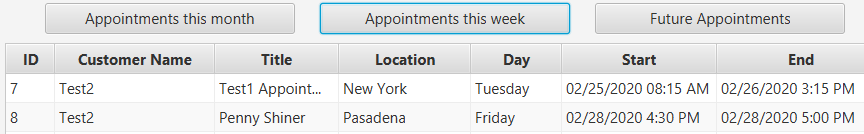


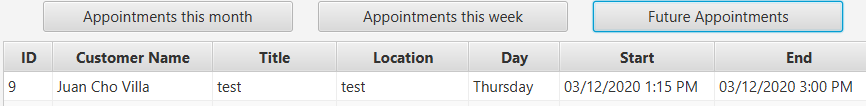
## and by week.

Client Dashboard option in the Appointments section “**Appointments this week**” (Displays all appointments in the next week for all consultants )

Appscheduler.QueryHandler.java (Line 701)

Controllers.MainAppointmentController.java





**E.    Provide the ability to automatically adjust appointment times based on user time zones and daylight saving time.**

Models.Appointment.Java

Controllers.MainAppointmentController.java

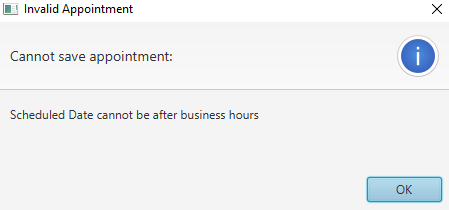
Appscheduler.QueryHandler.java ( Line 359 )

Controllers.ReportsController.java

When adding a new appointment, appointment times appear the local date and time of the current user and added to the database in UTC time.

**F.   Write exception controls to prevent each of the following. You may use the same mechanism of exception control more than once, but you must incorporate at least  two different mechanisms of exception control.**

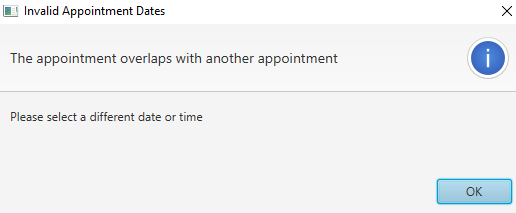
## Scheduling An Appointment Outside Business Hours (Throws)

Although the Hour and Period ComboBoxes appear to allow the user the ability to schedule appointments outside of business hours, since all hours are listed, when the user attempts to save the appointment, they are alerted that they are unable to save appointment due to scheduling outside of business hours.   
Further, the date picker prevents the ability to make appointments prior to the current date or on weekends.

Controller.EditerAppointmentController.java ( starting at Line 225)

Models.Appointment.java (Line 293)

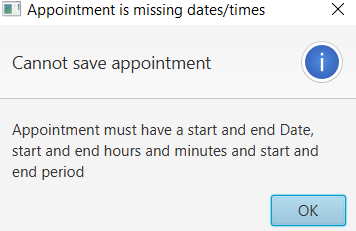
## scheduling overlapping appointments (Throw)

Appscheduler.QueryHandler.java (Line 427)

Controllers.EditerAppointmentController ( Line 168 )

This uses a throw method of exception.

## entering nonexistent or invalid customer data (Throws)

Models.Customer.Java (line 191)

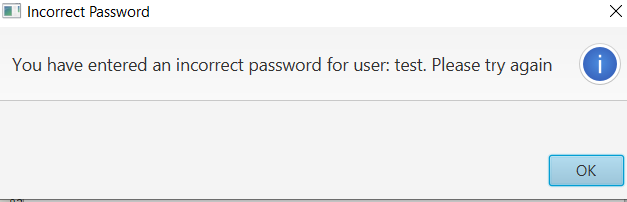
Controllers.EditerCustomerController.java

To validate entries before executing the add customers query, I check if all required text fields have a length greater than 0 (or not blank. See line 191)

To validate entries before running the modify/update customers query, I use a Boolean method to check whether a field was left blank, or if there were any changes whatsoever. (Starting line 81 in Editer Customer Controller.)

Both Adding and Updating customer functions utilize isCustomerValid from Customer.java.

## entering an incorrect username and password (Try/Catch)

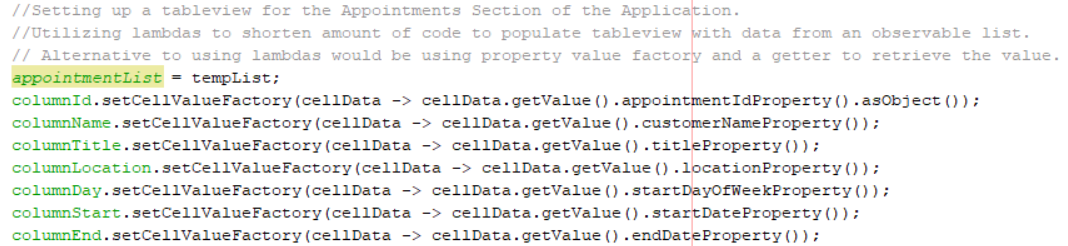
Appscheduler.QueryHandler.java starting line 36. ( Uses a try/catch method of exception )

**G.  Write two or more lambda expressions to make your program more efficient, justifying the use of each lambda expression with an in-line comment.**

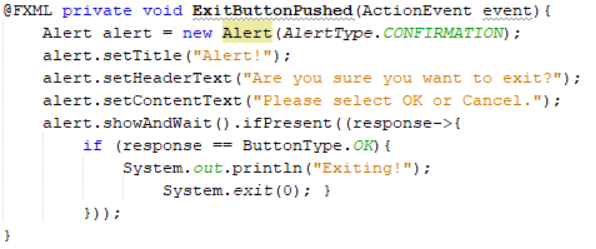
Controllers.MainAppointmentController.java

Controller.LoginController.java

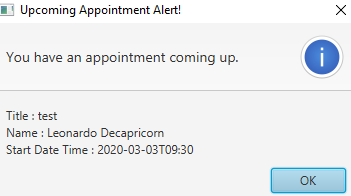
Example 1 at Line 169 in Main Appointment Controller: setCellValueFactory



Example 2 at Line 51 in Login Controller: Listener



**H.   Write code to provide an alert if there is an appointment within 15 minutes of the user’s log-in.**

Controller.LoginController.java ( Line 87 )

Appscheduler.QueryHandler.java ( Line 57 )

Models.Appointment.java

To test alerts as a consultant, make sure to have an upcoming appointment within the next 15 minutes with user Test.

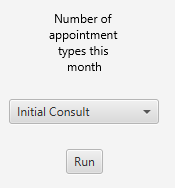
**I.   Provide the ability to generate each  of the following reports:**

**•   number of appointment types by month**

**•   the schedule for each consultant**

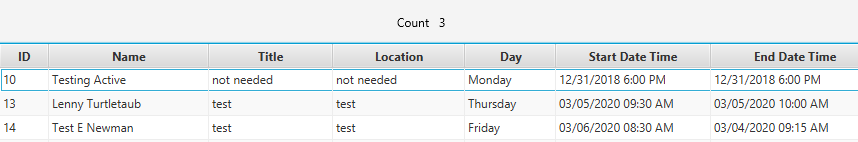
**•   one additional report of your choice**

## Number of Appointment Types By Month

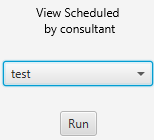
Client Dashboard Option in the Reports Section: **Number Of Appointment Types This Month**

Appscheduler.QueryHandler.java (Line 942)

Controllers.ReportsController.java

Use the Appointment Types ComboBox to select an appointment type. 

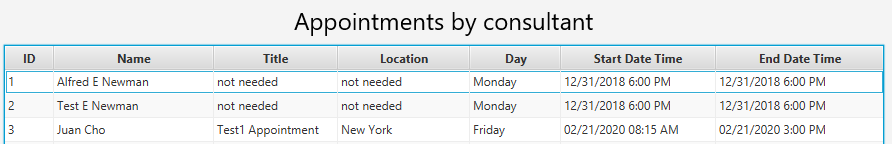
## Appointment By Consultant

Client Dashboard Option in the Reports Section: **View Scheduled By Consultant**

Appscheduler.QueryHandler.java (Line 828)

Controllers.ReportsController.java

Use the Consultant ComboBox to select a consultant (user) to filter results by consultant.



## View Inactive Customers

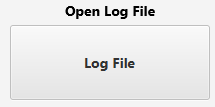
Client Dashboard Option in the Reports Section: **View Inactive Customers**

Appscheduler.QueryHandler.java (Line 791)

Controllers.ReportsController.java

Use the Run Button underneath View Inactive Customers to view a list of all inactive customers.

**J.   Provide the ability to track user activity by recording timestamps for user log-ins in a .txt file. Each new record should be appended to the log file, if the file already exists.**



Client Dashboard option in the Log File section: **Log File**

Controllers.DashboardController.java ( Line 50 )

Controllers.LoginController.java ( Line 47 ).

