SOFTWARE ENGINEERING

CSYE 7230

Project Name: MedSign

Semester: Fall 2014

Instructor: Prof. Mieczyslaw .M. Kokar

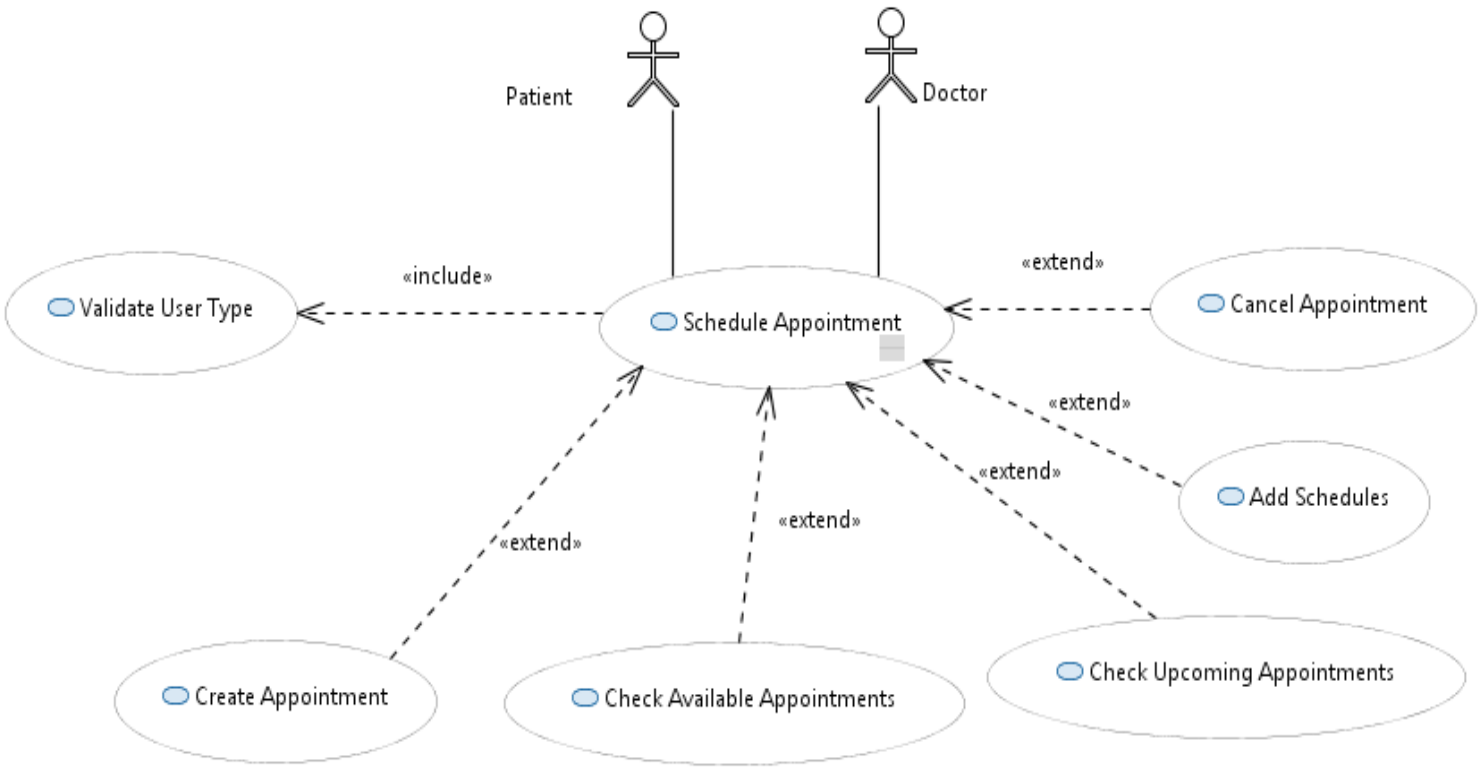
Author: Anantha Shankar Arun Kumar

NUID: 001726278

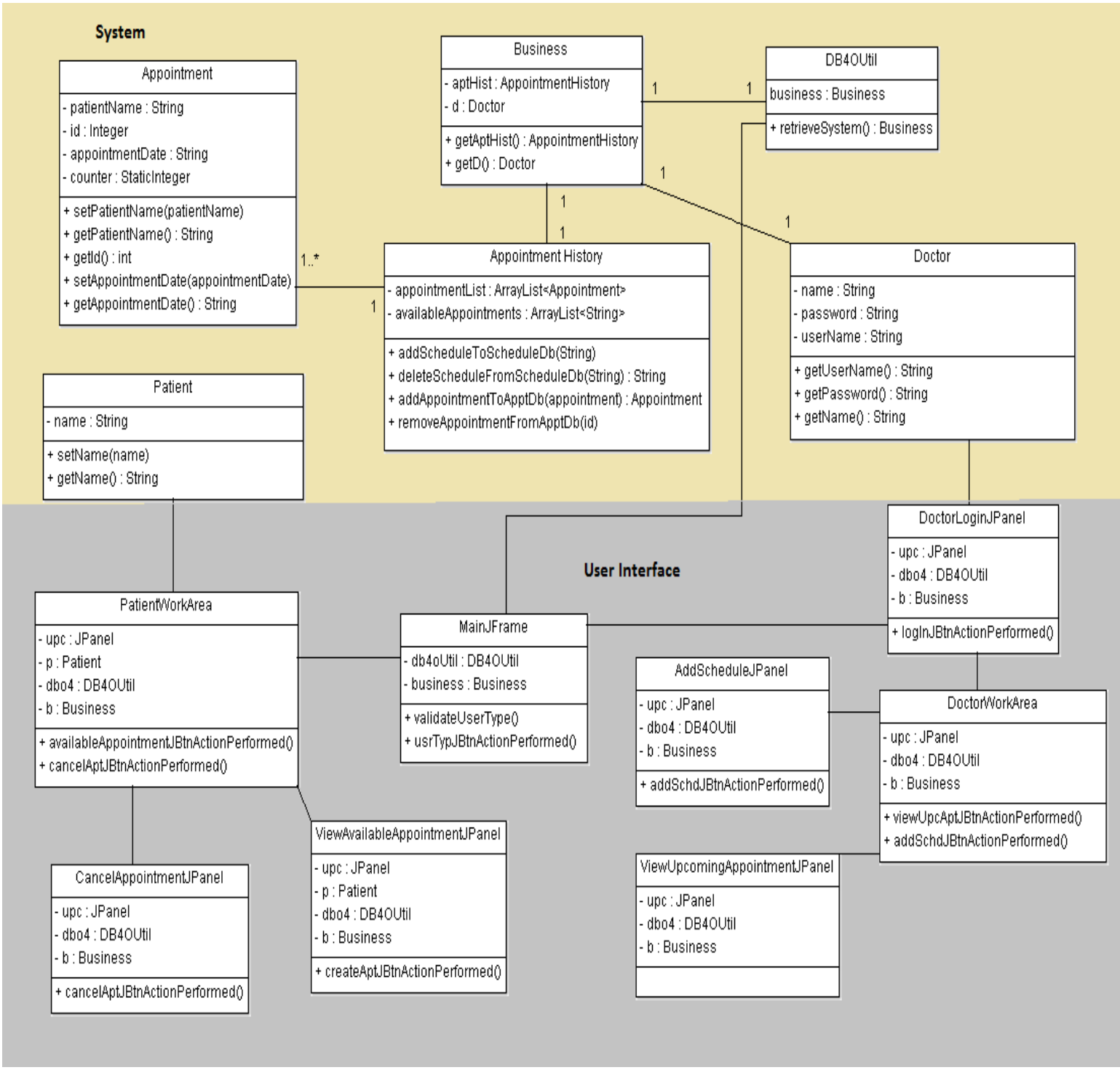
Submission Date: 11/30/2014

Table of Contents

Use Case Diagram

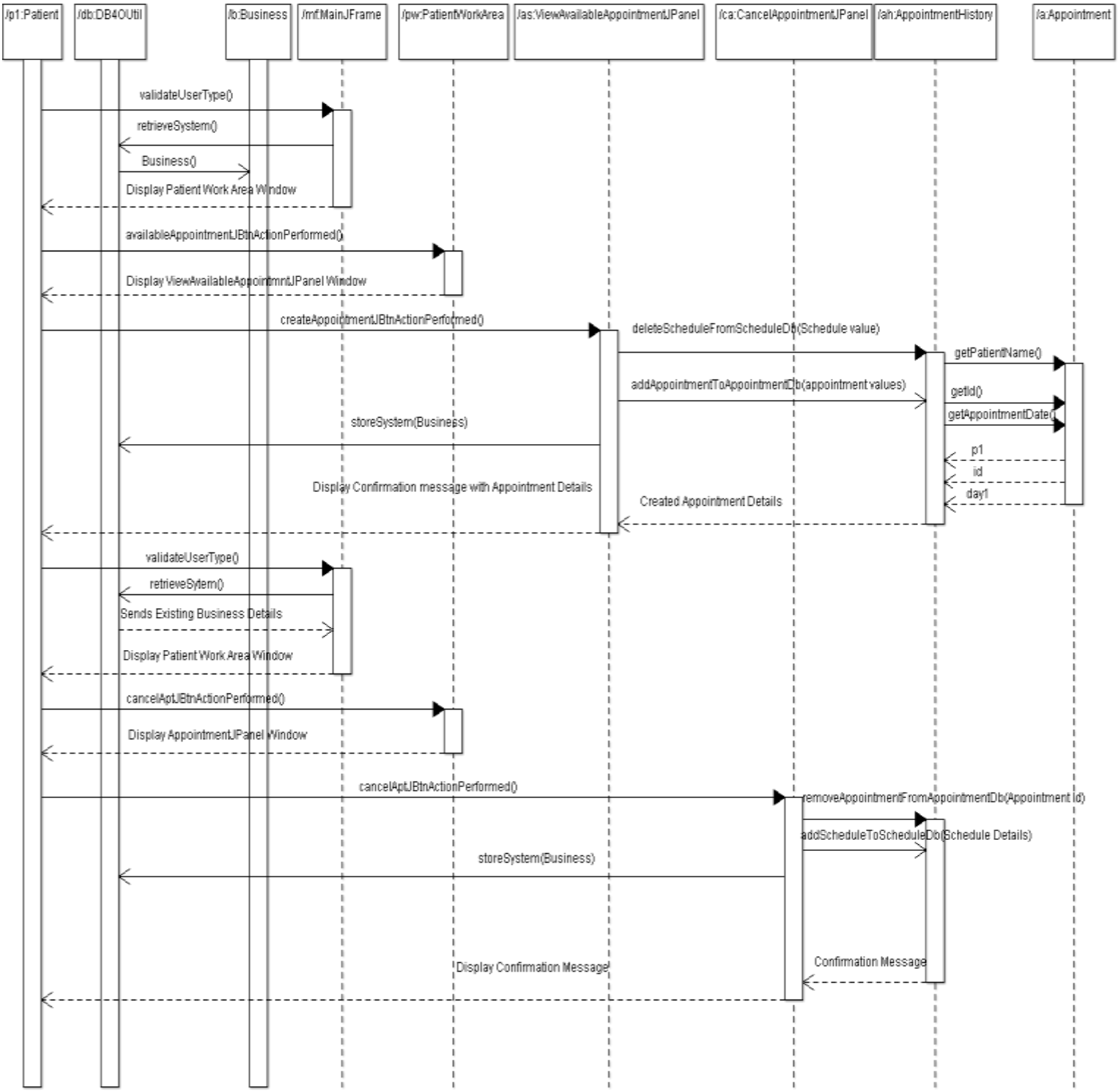


Class Diagram with Java and System Classes

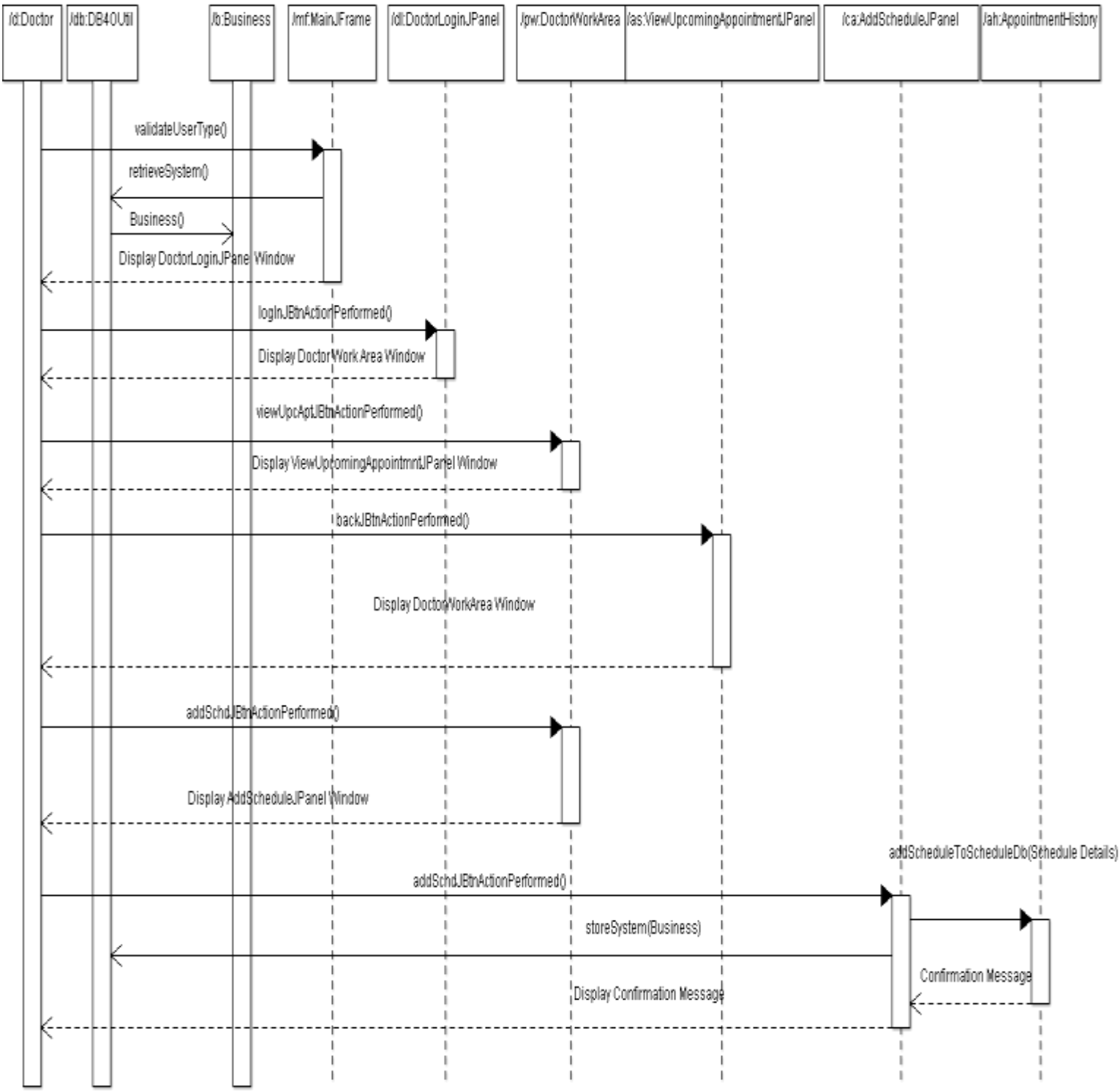


Sequence Diagrams

Patient Transaction Sequence Diagram



Doctor Transaction Sequence Diagram



Mapping Description

- When the application is run for the first tie an instance of **Doctor** class is created and in further use same doctor object is used as there will be only one doctor.
- **Patient** object is created on every appointment creation.
- The Association between **DB4OUtil** class and **Business** class is established by creating a new instance of **Business** class in method **retriveSystem()** in **DB4OUtil** class in case the application is being used for the first time. If the application has been used and if it consists of appointments or schedules then the existing data will be retrieved from the DB using **retriveSystem()** in **DB4OUtil** class.
- The association of Business class with Doctor and **AppointmentHistory** class is established by creating the instance of **Doctor** and **AppointmentHistory** class in **Business** class. So, whenever the **Business** class is initiated a new instance of **Doctor** and **AppointmentHistory** class is created, which is only once during the first use of the application and later existing instances will be used.
- The association between **AppointmentHistory** class and Appointment class is established by creating an attribute of appointment in **AppointmentHistory** class.
- The association between **DB4OUtil** and **MainJFrame** class is established calling the **getInstance()** and **retieveSystem()** methods of **DB4OUtil** class to retrieve the existing or new instance of **DB4OUtil** class.
- The association between JPanel classes like **PatientWorkArea, DoctorWorkArea, CancelAppointmentJPanel, ViewAvailableAppointmentJPanel, AddScheduleJPanel, ViewUpcomingJPanel** have been established by passing the required parameter to process in the next JPanel class. Eg. From PatientWorkArea class Patient, DB4OUtil and Business instance parameters are passed to viewavailableAppointmentJPanel class which are required for the methods in the class to process.

White Box and Black Box testing

Black Box Testing

Testing Class: **AddScheduleJPanel** class

The above class allows the doctor to add schedules by entering date and selecting the time slots from UI window. It has to be made sure that the date entered is valid and for same date and time multiple slots should not be created.

Below are the test cases or equivalence classes for which input types needed to be created and tested with expected result to make sure that the unwanted appointments are not created:

- Schedule with date value 30 – Schedule should not get created and should throw error.
- Schedule with date value 31 – Schedule should not get created and should throw error.
- Schedule with wrong date Value for the Febraury > 29 – Schedule should not get created and should throw error.
- Schedule with wrong month values: 0, 13, -1 – Schedule should not get created and should throw error.
- Leap years to validate Febraury month date value -
    - Divisible by /4 – Schedule should not get created and should throw error.
    - Divisible by /100 – Schedule should not get created and should throw error.
    - Divisible by /400 – Schedule should not get created and should throw error.
- Schedule with year value lesser than 2014 – Schedule should not get created and should throw error.
- Schedule with date value lesser than the day the schedule is being added (schedule should be added to current date or upcoming date) – Schedule should not get created and should throw error.
- Schedule with date and time values same as the existing available schedule – Schedule should not get created and should throw error.
- Schedule with all the proper values – Schedule should not get created and should throw error.

BlackBox Test Result Table

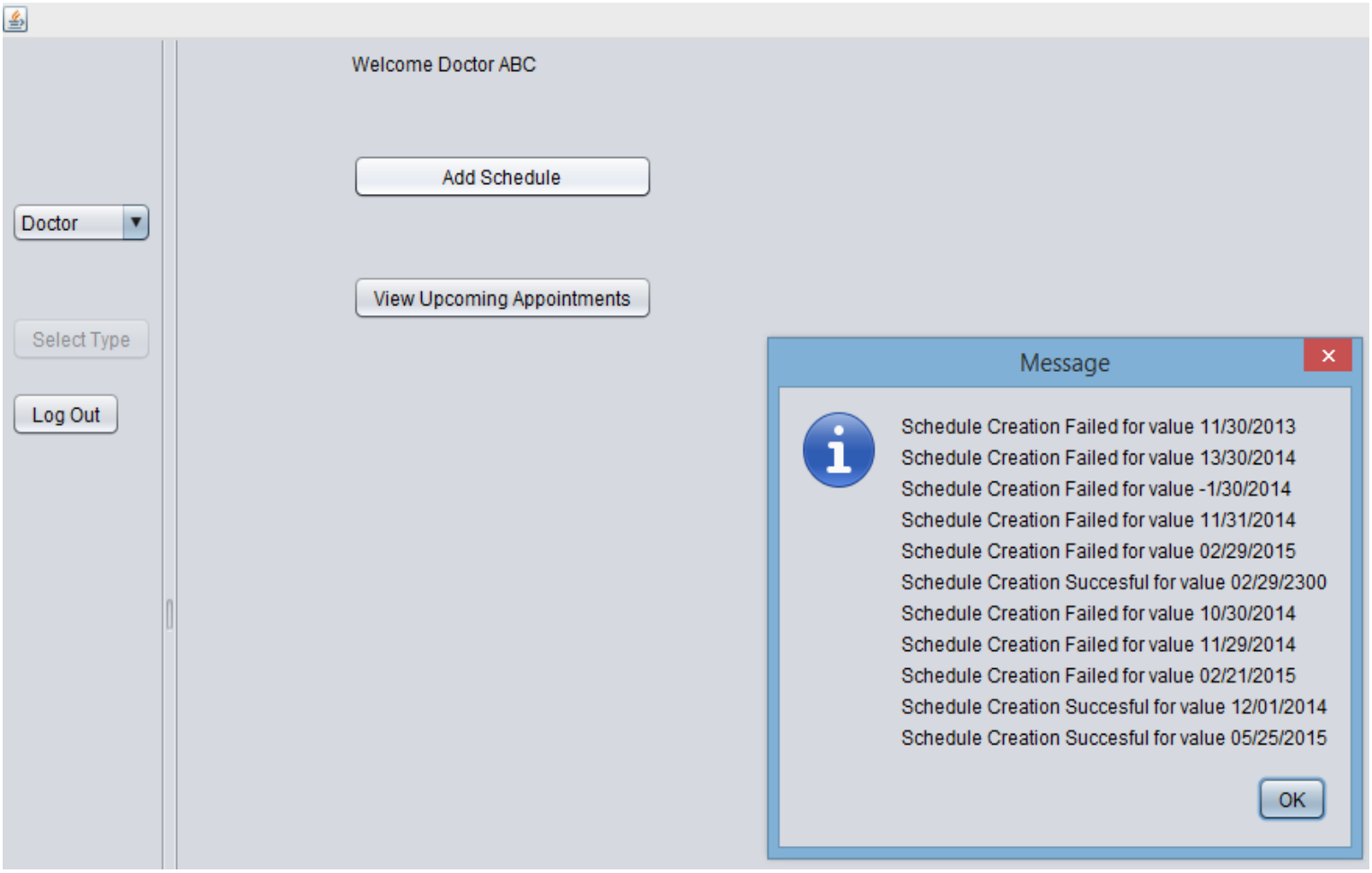
Test Case	Input Value	Expected Result	Actual Result
Value for testing old date year - current date = 11/30/2014	11/30/2013 6:30 PM	Schedule should not get created	Pass
Value for wrong month value	13/30/2014 6:30 PM	Schedule should not get created	Pass
Value for wrong month value and number exception	-1/30/2014 6:30 PM	Schedule should not get created	Pass
Value for wrong date	11/31/2014 6:30 PM	Schedule should not get created	Pass
Value for leap date error	02/29/2015 6:30 PM	Schedule should not get created	Pass
Value for leap date error	02/29/2300 6:30 PM	Schedule should not get created	Fail
Value for testing old date month - current date = 11/30/2014	10/30/2014 6:30 PM	Schedule should not get created	Pass
Value for testing old date day - current date = 11/30/2014	11/29/2014 6:30 PM	Schedule should not get created	Pass
Value for testing schedule with existing schedule value 02/21/2015 6:30 PM	02/21/2015 6:30 PM	Schedule should not get created	Pass
Value for testing existing appointment value - 12/01/2014 6:30 PM	12/01/2014 6:30 PM	Schedule should not get created	Fail
value with valid Value for which appointment should get created	05/25/2015 6:30 PM	Schedule should get created	Pass

Two of the above cases failed

1. Leap year that is not divisible by 400 but divisible 100.
2. Schedule got created to the date and time for there is already an existing appointment.

The above failures were noted and were modified to resove the same in amin source code.

Below is the screen shot of the test run result



White Box Testing

Testing Class: **AppointmentHistory** class

The above system class is the class that performs operation of adding and removal of appointments and schedules from DB and it has to be made sure that the operation of the methods need to be proper.

To achieve this loop testing is the appropriate way to verify the operation of the methods in the class.

Below are the things that will be included in the driver to be created to test the class:

Methods in the class are

- addScheduleToScheduleDb()
- removeScheduleFromScheduleDb()
- addAppointmentToAppointmentDb()
- removeAppointmentFromAppointmentDb()

To test the above methods the driver should have input ranges in terms of existing schedule or appointments, past schedule or appointments and future schedule or appointments.

While running loop tests the methods has to executed exactly once, when they're individually called to process. But when the methods are dependent they also have to run when dependent methods have been called.

Eg: when method removeScheduleFromScheduleDb() is called individually it has to run once individually. When addAppointmentToAppointmentDb() is called it is dependent on method removeAppointmentFromAppointmentDb() has to run first and then addAppointmentToAppointmentDb() has to run.

While creating the driver the inputs to these methods should be defined in different ranges as mentioned above i.e.,

- For addScheduleToScheduleDb() method if existing schedule value is given then the schedule should get added again to db which will create two copies of same schedule and instead should throw error. Same applies to addAppointmentToAppointmentDb() as well, is appointment has been created for a date and time, another appointment should not be created for same date and time.
- The addScheduleToScheduleDb() method should also throw error when an schedule is being added for date and time for which appointment already exists.
- Schedule or appointment must allowed to create only for current date or future date and should not be allowed for previous dates.
- While creating an appointment or schedule if any null reference is there in it then the methods should to create it and should throw error.
- For proper input provided the method should be able to add or remove appointments or schedule from DB successfully

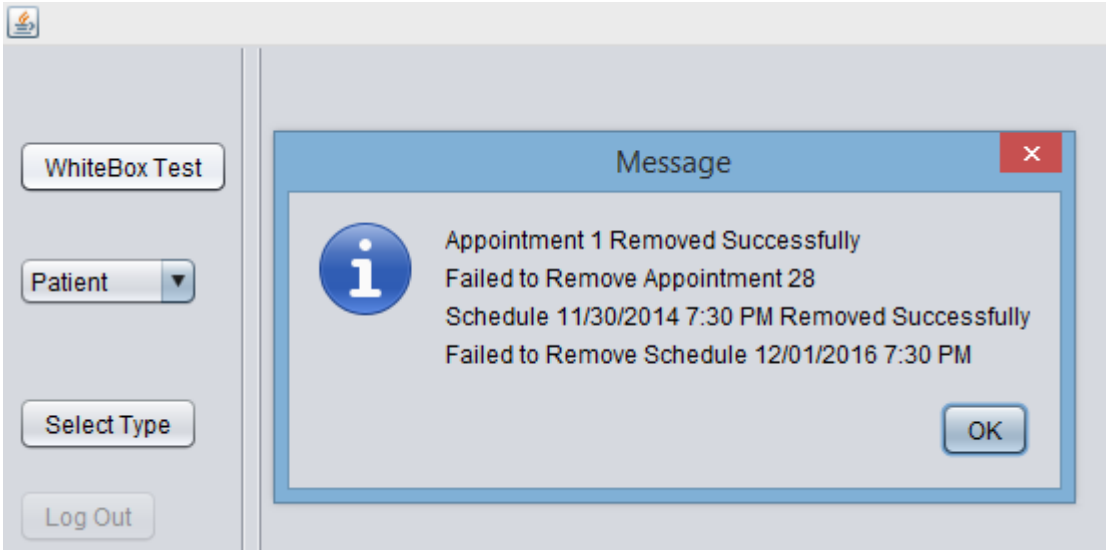
Considering all the above conditions the driver will be designed to provide various range of required inputs dynamically and will be coded to throw error on exceptions.

WhiteBox Test Result Table:

Test Case	Input Value	Expected Result	Actual Result
Remove existing appointment	2	Appointment should not get deleted	Pass
Remove non existing appointment	28	Throw error	Pass
Remove existing schedule	11/30/2014 6:30 PM	Schedule should not get deleted	Pass
Remove non-existing schedule	12/01/2016 7:30 PM	Throw error	Pass

The remaining test cases defined above (appointment and schedule creation) were already tested as a part of black box testing and the results would be same.

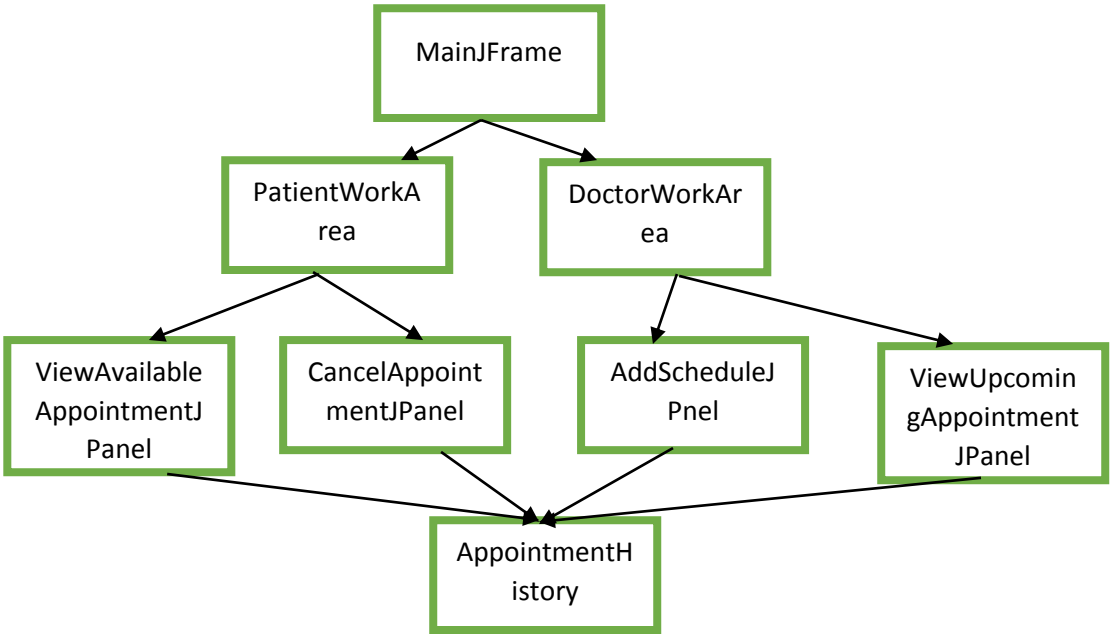
Below is the screen shot of the test run result

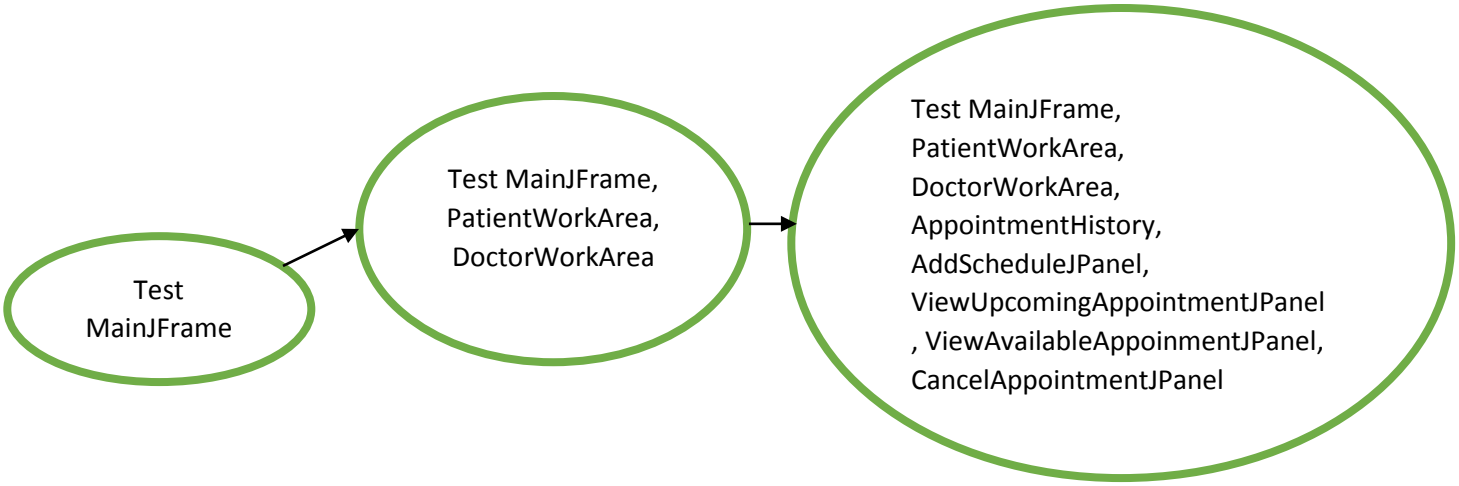


Integration Testing

Component Classes: Business, DB4OUtil, MainJFrame, PatientWorkArea, DoctorWorkArea, AppointmentHistory, Appointment, AddScheduleJPanel, ViewUpcomingAppointmentJPanel, ViewAvailableAppoinmentJPanel, CancelAppointmentJPanel

Order Integration Strategy Top Down





- Stubs created are used to provide values of user type in the MainJFrame window. On selected value to the MainJframe should navigate the user to either PatientWorkArea or DoctorWorkArea. The navigation o window should only happen at the moment of value setting.
- To test further the navigation of PatientWorkArea and DoctorWorkArea the stubs are created and to select the options available in the window.
- In the classes AddScheduleJPanel, ViewUpcomingAppointmentJPanel, ViewAvailableAppoinmentJPanel, CancelAppointmentJPanel necessary dummy values are provided using stubs and corresponding appointments and schedules are verified using AppointmentHistory class by creating stubs to return values from DB.

Inputs and Results

- For error values of usertype the MainJFrame should not navigate to next window.
- While creating appointment if no proper name is provided then the appointment should not be created and should throw warning.
- If no appointment is selected and if called create appointment method then error has to be thrown and appointment should be created.
- For proper inputs appointment should be successfully created and respective schedule should be deleted from schedule list.
- In DoctorLoginWindow for wrong credentials, DoctorWorkArea window should not be navigated and should throw error.
- If null value, invalid date value or invalid data type value be passed while creating schedule, the schedule should get created and an error should be thrown.
- For the same date and time multiple schedule should be created and should throw error message for the same.
- Given the proper values as inputs to create schedule, the schedule should get created and be added in schedule DB.

Integration Test Result Table

Test Case	**Expected Result**	**Actual Result**
Navigation of MainJFrame	Should navigate with valid inputs	Pass
Navigation between MainJFrame, PatientWorkArea, DoctorWorkArea	Should navigate with valid inputs	Pass
Test MainJFrame, PatientWorkArea, DoctorWorkArea, AppointmentHistory, AddScheduleJPanel, ViewUpcomingAppointmentJPanel, ViewAvailableAppoinmentJPanel, CancelAppointmentJPanel	Should navigate with valid inputs and process methods for valid inputs and throw error for invalid inputs	Pass

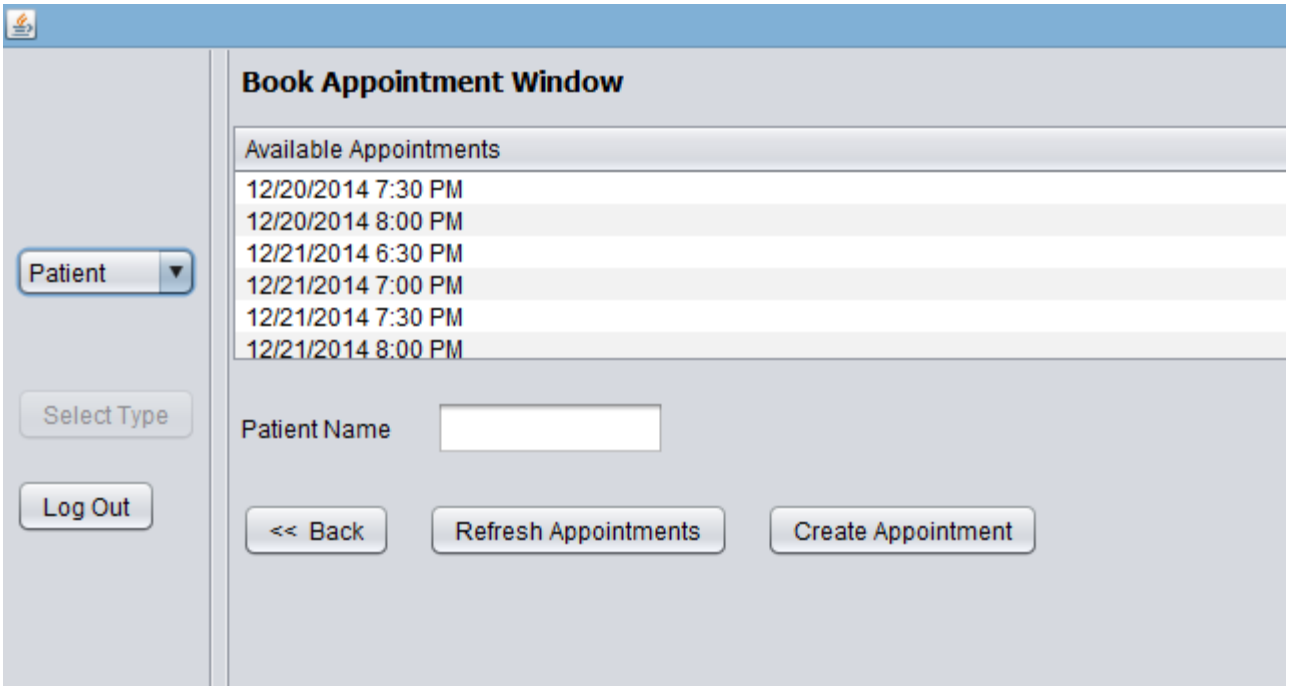
Steps To Run the Application

- Open the folder named “Med_Sign” from the zip attachment. This folder will be having the predefined appointments and schedules (this file be DataBase with extension .db4o), executable .jar file and a library folder.
    - Please note that all the above mentioned files are necessary for the application to run. These files must in the folder in which they were before zipped. (Med_Sign)
- Run the .jar file named Med_Sign.jar file to run the application.
- Once the application opens you will prompted to select the type of user.
- If you select patient as user, you’ll be able to view existing slots to book appointments and to cancel the existing booked appointments if you know the appointment id.
- If you select the doctor as user you’ll be navigated to doctor login window.
- The default credential created
    - UserName – “**Doctor**”
    - Password – “**Doctor**”
    - Doctor Name – “ABC”
- Once the entered credentials are verified, you’ll be navigated to doctor work area where you can add new schedules and view upcoming appointments.
- Few of the schedules and appointments has already been created for your reference.

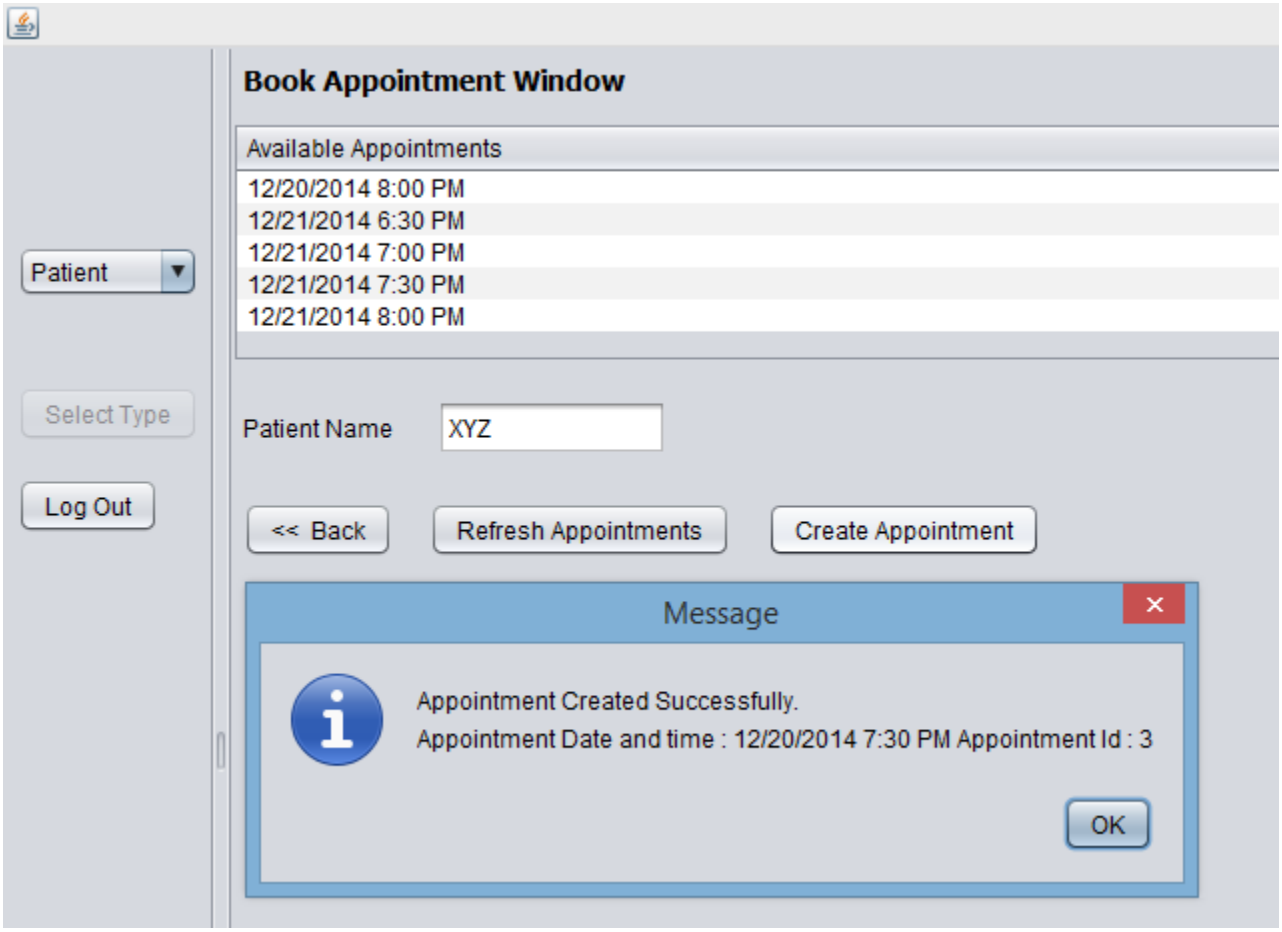
Using application as Patient

- When clicked on user type patient you’ll be get two buttons “View Available Appointments”, “Cancel Appointment”

- When clicked on “View Available Appointments” you’ll get to see the slots available to book.

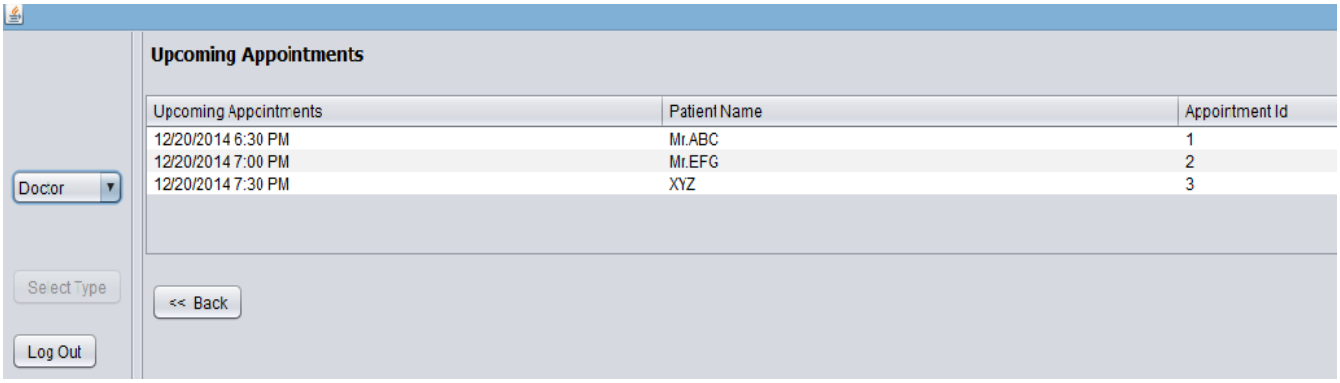


- Entering a valid name and selecting an available slot you can book a new appointment and appointment details will be displayed.

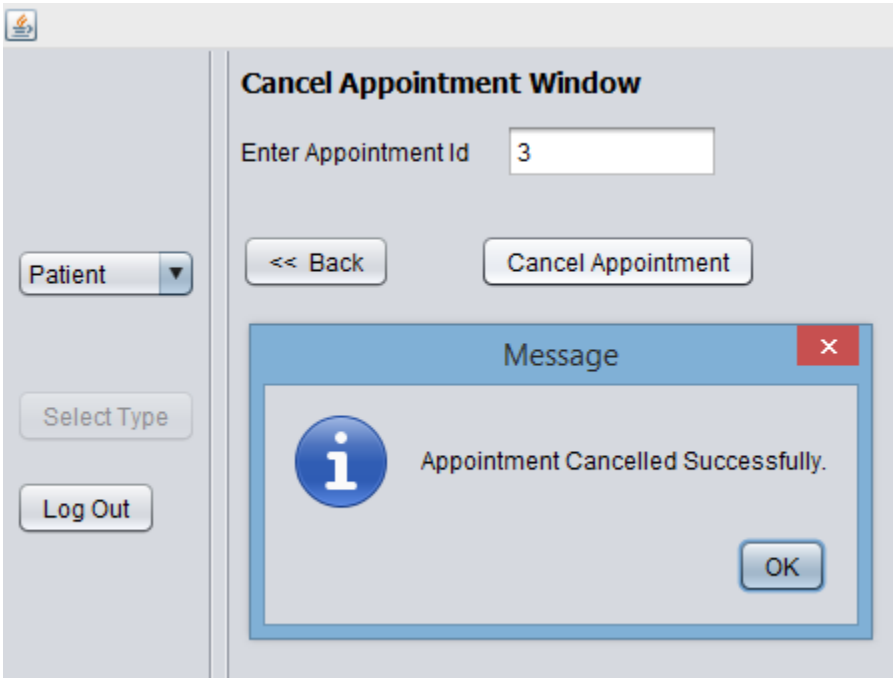


- On clicking “Back” button you’ll go back to “PatientWorkArea” window again. On clicking cancel appointment and entering booked appointment id you’ll be able to cancel booked appointment.

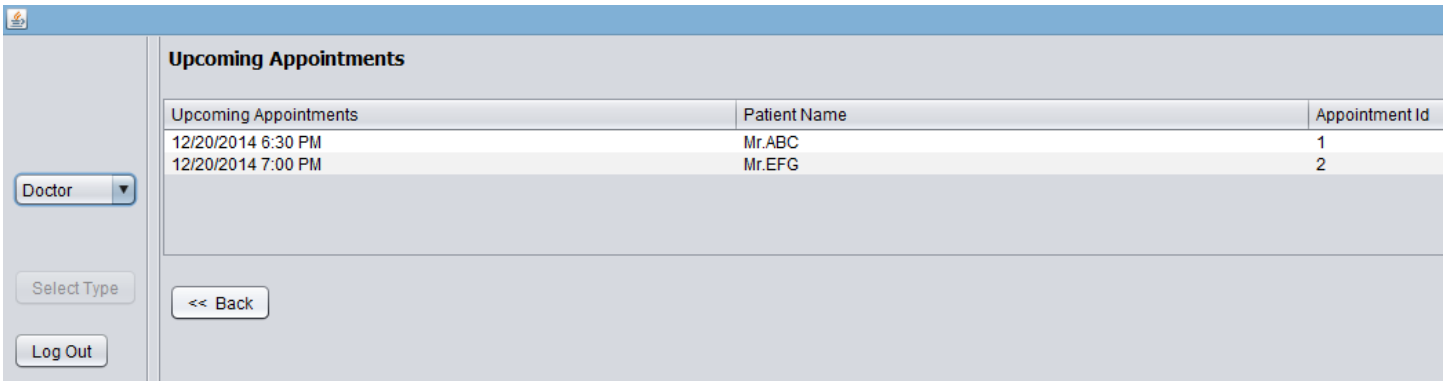
Upcoming appointments that the doctor sees before the appointment is cancelled (Appointment Id - 3)



Patient cancelling the appointment



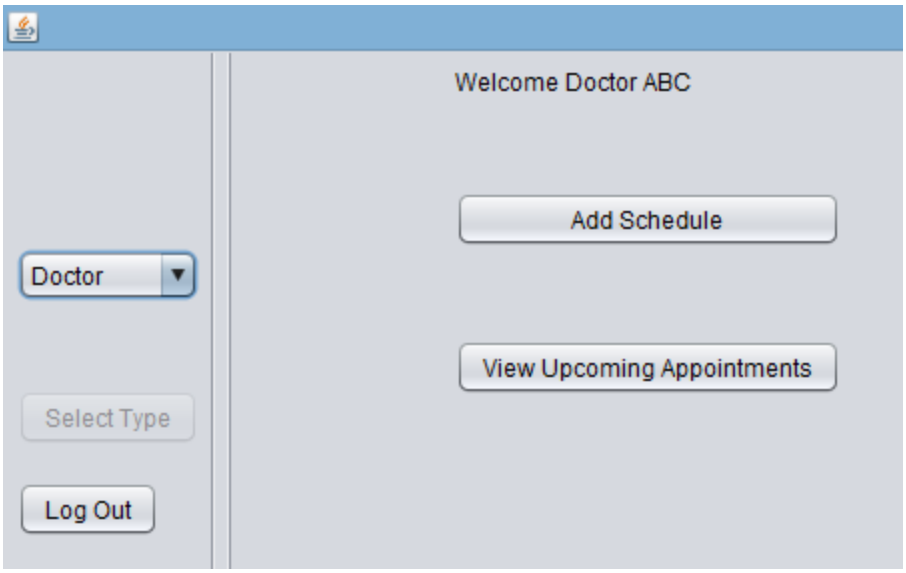
Upcoming appointments doctor sees after the appointment is cancelled (Appointment Id - 3)



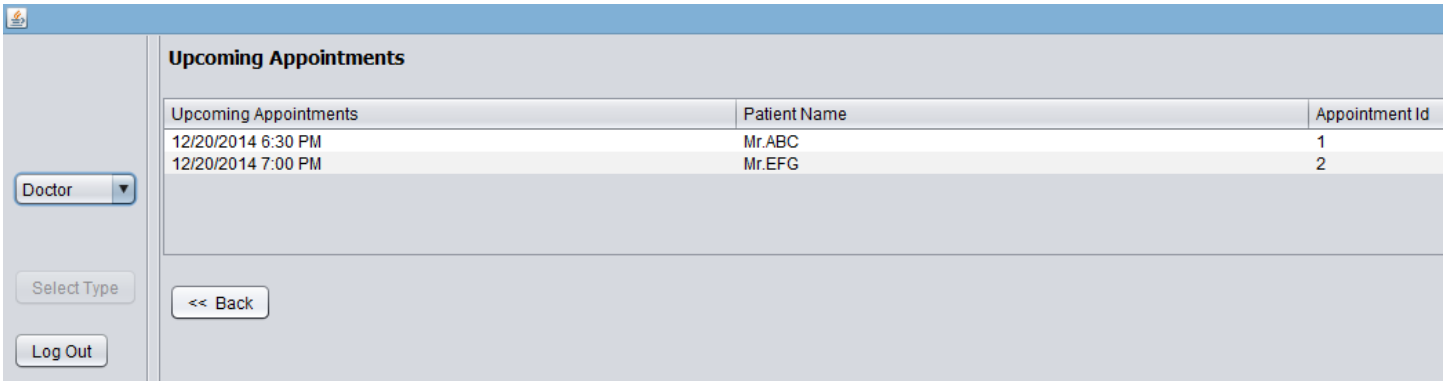
Using application as Doctor

- When clicked on user type patient you'll prompted to enter login credentials of doctor which is "Doctor", "Doctor"

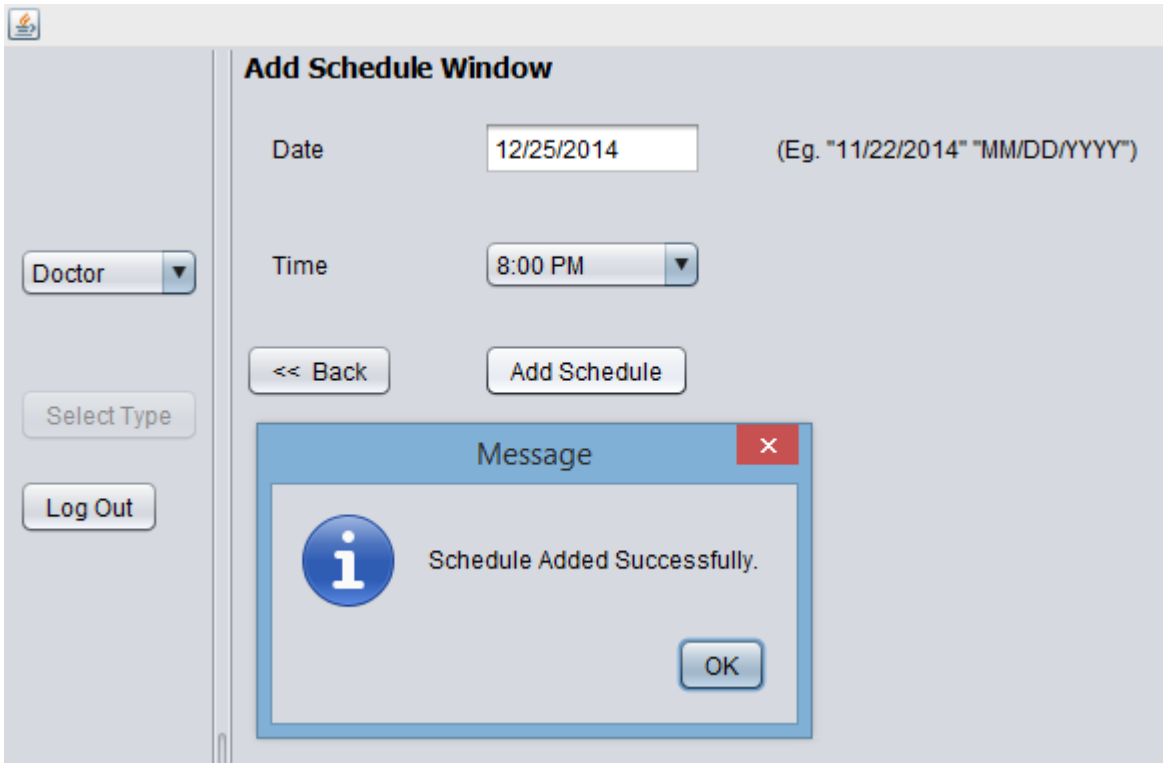
- On clicking "Log In" button you'll navigated to Doctor Work Area window where you'll get buttons "Add Schedule" and "Check Upcoming Appointments"



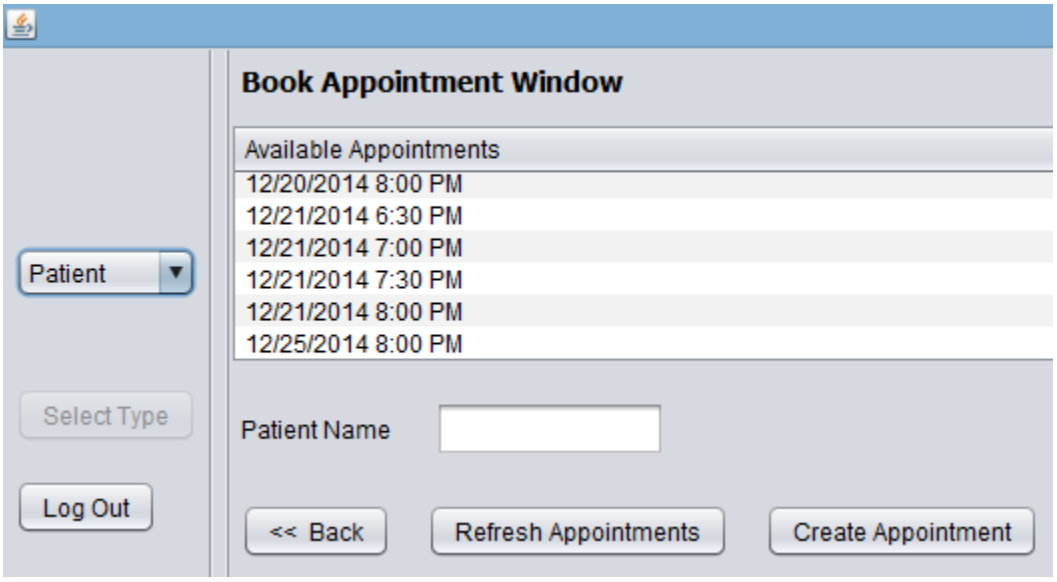
- On clicking “View Upcoming Appointments” you’ll get to see upcoming appointments



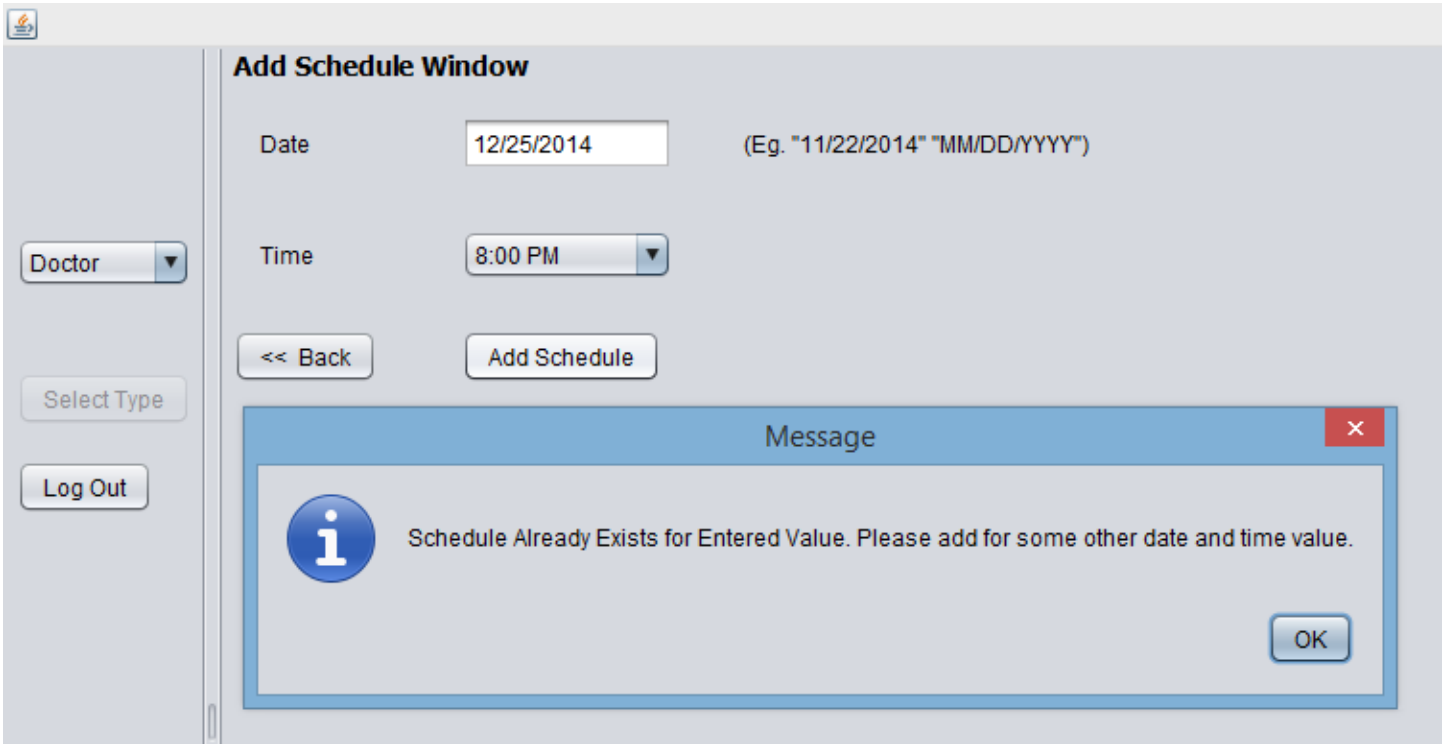
- On clicking back and clicking on “Add Schedule”, you’ll be navigated to Add Schedule window. Where by entering proper date with proper format and selecting a time from the Time options you can create a new schedule which will be visible to patients in Available Appointments slots. The time slot has been standardized to four slots i.e., 6:30 PM, 7:00 PM, 7:30 PM and 8:00 PM



Appointments patient can see after doctor adding schedule (12/25/2014 8:00 PM)

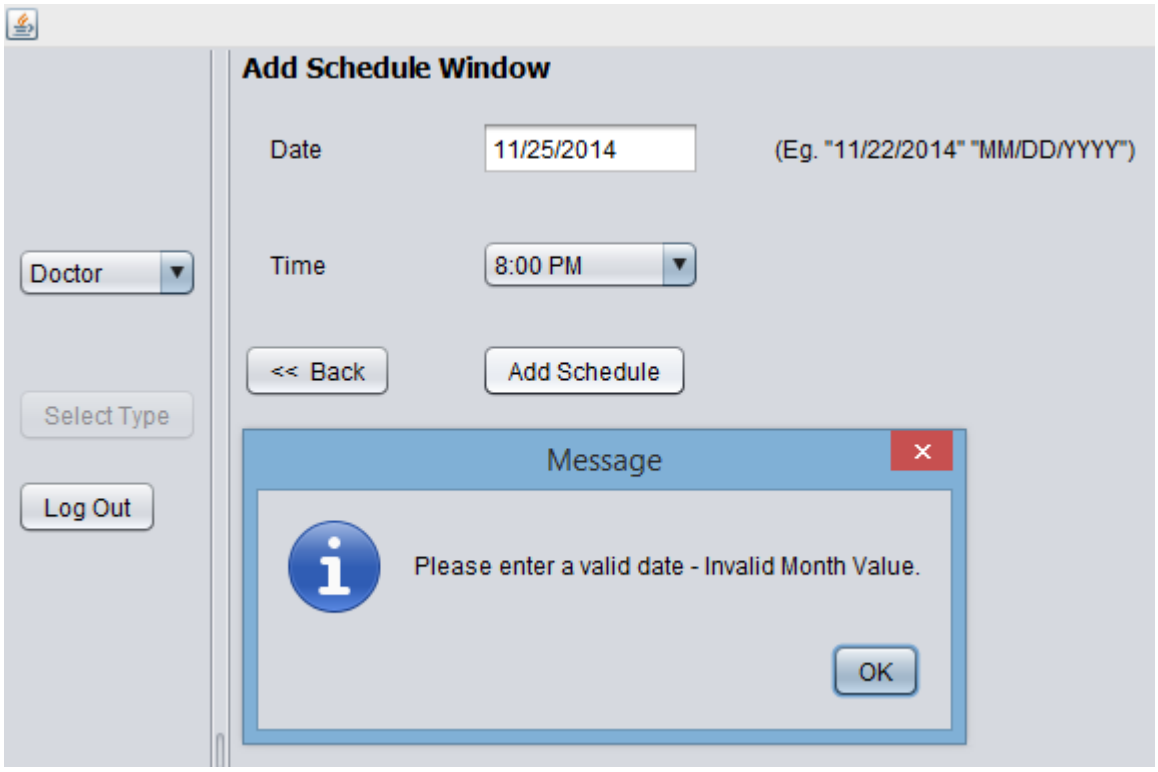


- If you try to add a schedule existing date and time you'll not be allowed to create schedule

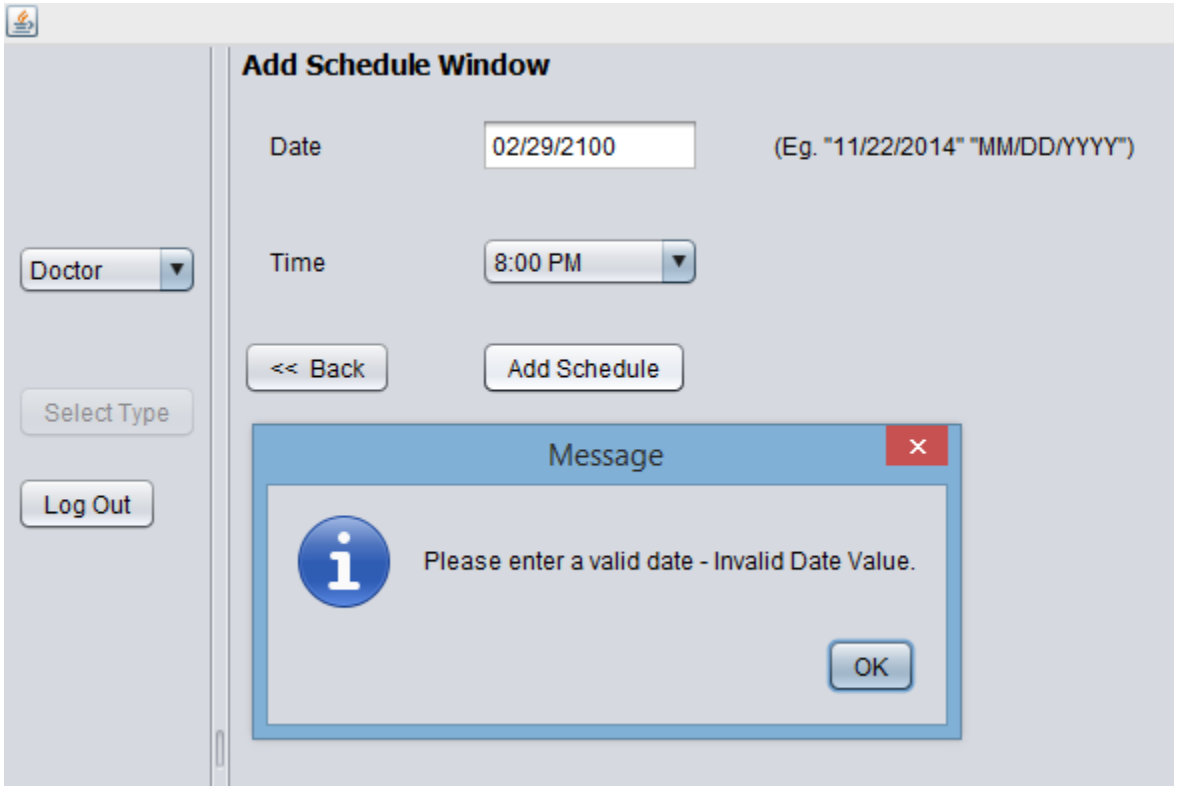


- Also if you try to add schedule with invalid date, invalid leap year date or old date, you'll not be allowed to create schedule

Old date value



Invalid leap year



Concurrency Handling

Two handle concurrency you need to run two different sessions. If the two patients selects same available appointment and tries to book it then whoever clicks on the “Create Appointment” button will be able to book it and other patient will be prompted to book other appointment by clicking “Refresh” button to update available appointment table.