Clinic Appointments System

Dongfan Zhang

Bbackgrounds: a system to manage appointments for small clinics

System can eliminate the manual appointment taking. Increase efficiency.

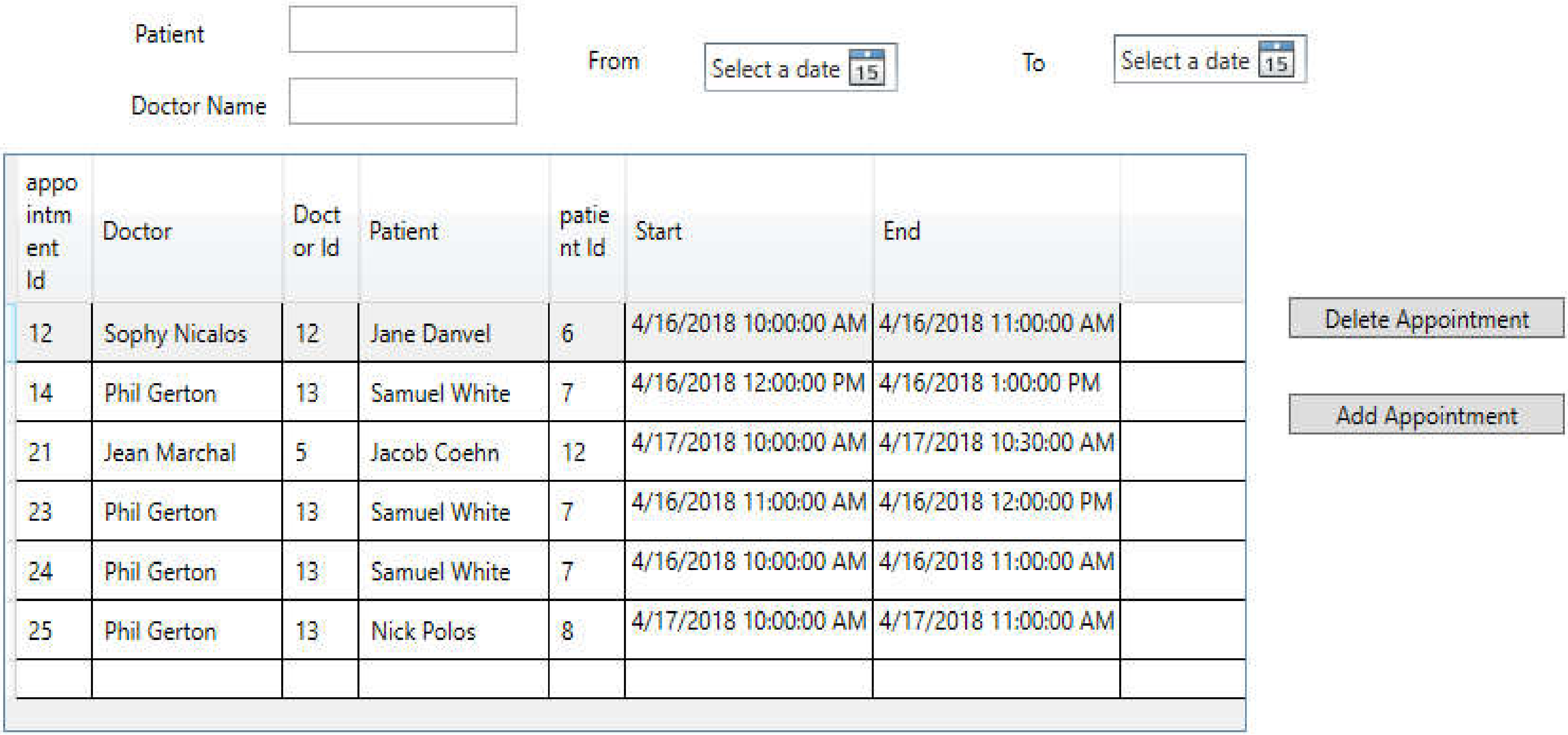
- allow patients to take appointments, view appointments and cancel appointments, update his own contact data
- allow doctors to manage their schedule, view patients’ profile, view existing appointments, update his own contact data
- allow administrator to manage doctors, patients, appointments, etc.

Solution overview – three user interface

When user successfully logged in, based on their user profile, system will populate the correct user interface for them and each will have different functionalities implemented

- Patient
- Doctor
- Admin

Admin- Manage appointments



Patient

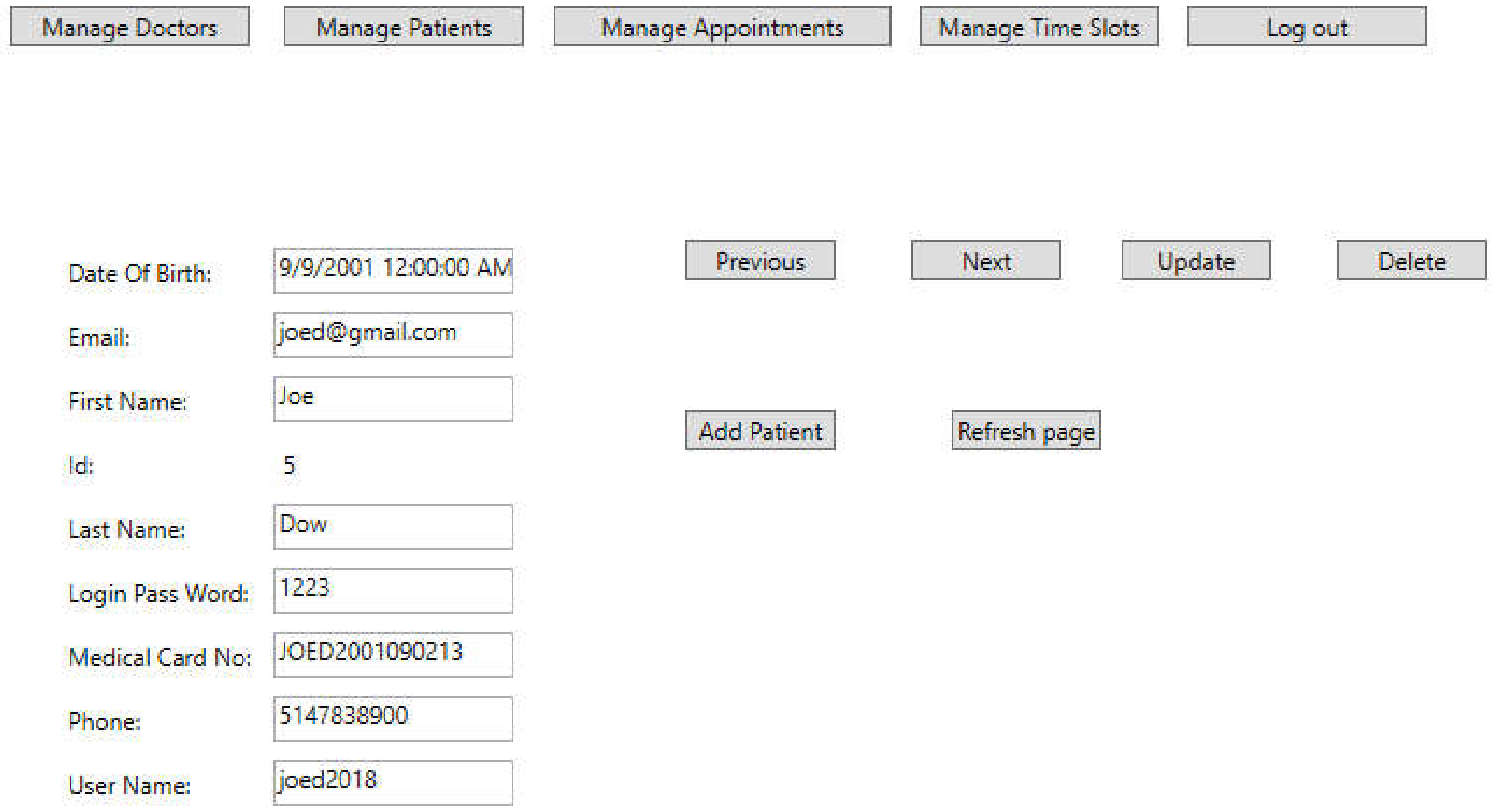
Doctor Name

From Select a date 15

To Select a date 15

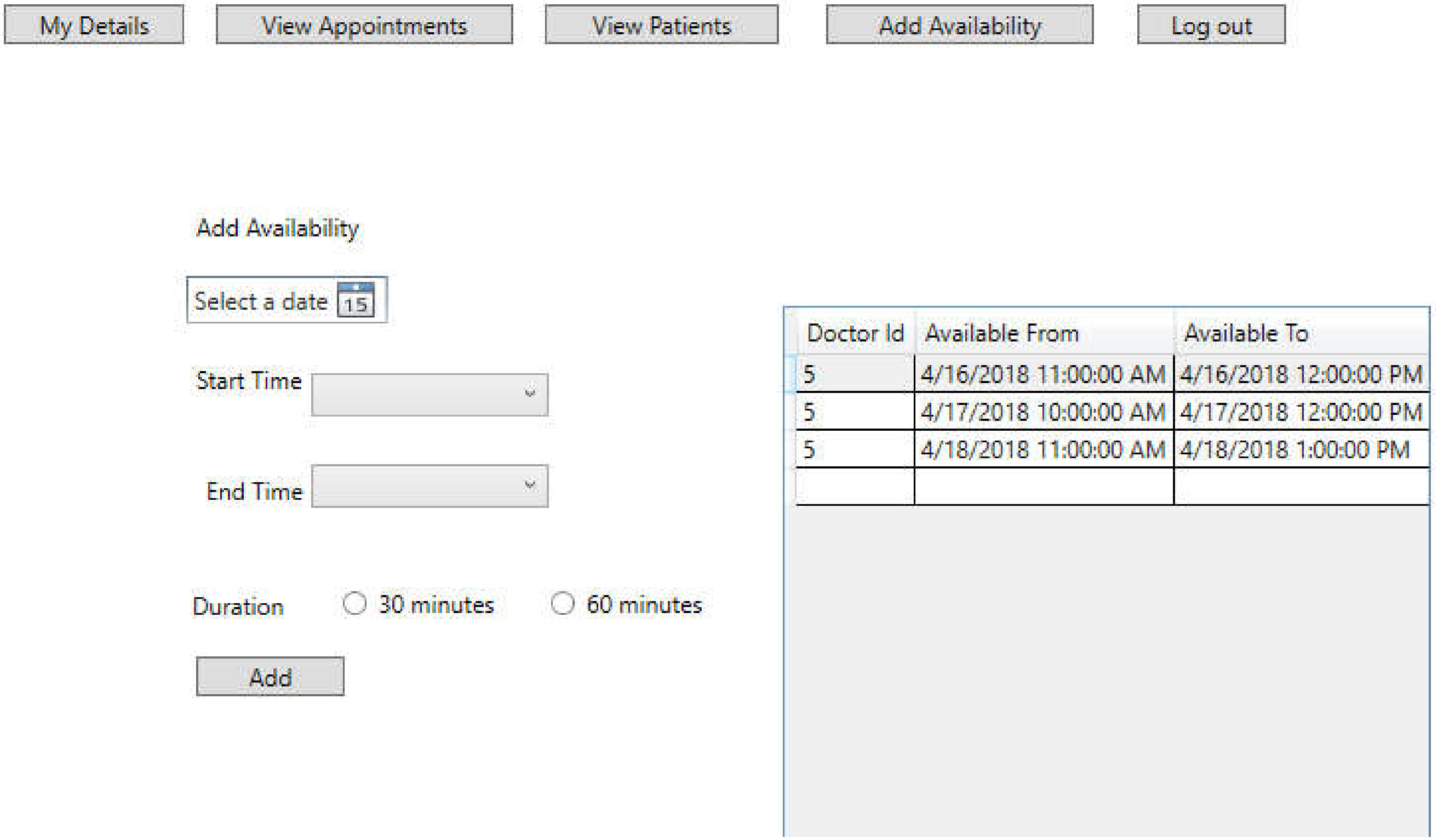
appointment Id	Doctor	Doctor Id	Patient	patient Id	Start	End	
12	Sophy Nicalos	12	Jane Danvel	6	4/16/2018 10:00:00 AM	4/16/2018 11:00:00 AM	
14	Phil Gerton	13	Samuel White	7	4/16/2018 12:00:00 PM	4/16/2018 1:00:00 PM	
21	Jean Marchal	5	Jacob Coehn	12	4/17/2018 10:00:00 AM	4/17/2018 10:30:00 AM	
23	Phil Gerton	13	Samuel White	7	4/16/2018 11:00:00 AM	4/16/2018 12:00:00 PM	
24	Phil Gerton	13	Samuel White	7	4/16/2018 10:00:00 AM	4/16/2018 11:00:00 AM	
25	Phil Gerton	13	Nick Polos	8	4/17/2018 10:00:00 AM	4/17/2018 11:00:00 AM	

Admin – Manage Patients



Date Of Birth:	9/9/2001 12:00:00 AM
Email:	joed@gmail.com
First Name:	Joe
Id:	5
Last Name:	Dow
Login Pass Word:	1223
Medical Card No:	JOED2001090213
Phone:	5147838900
User Name:	joed2018

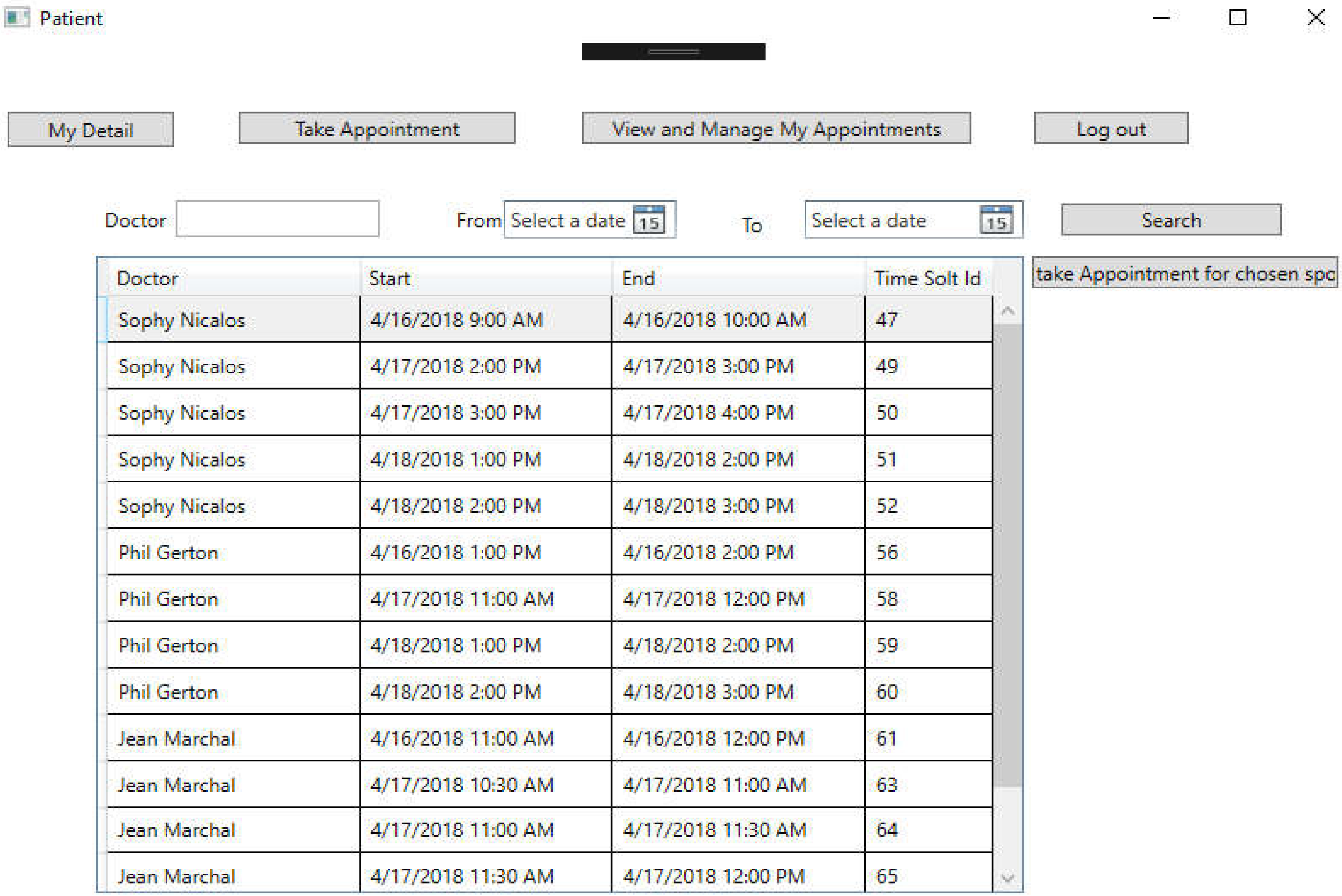
doctor – Add availability



Duration ○ 30 minutes ○ 60 minutes

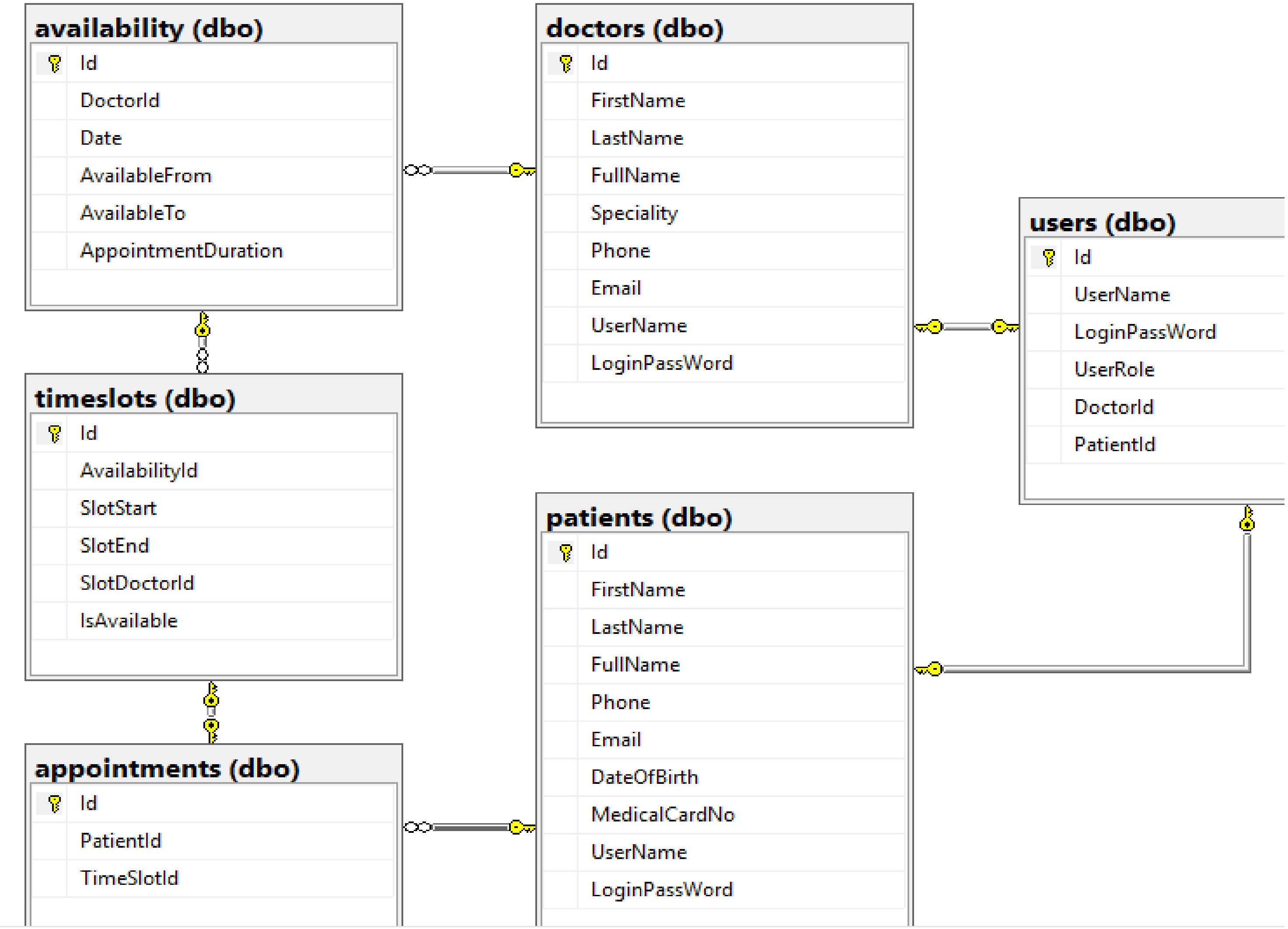
Doctor Id	Available From	Available To
5	4/16/2018 11:00:00 AM	4/16/2018 12:00:00 PM
5	4/17/2018 10:00:00 AM	4/17/2018 12:00:00 PM
5	4/18/2018 11:00:00 AM	4/18/2018 1:00:00 PM

Patient



Doctor	Start	End	Time Solt Id
Sophy Nicalos	4/16/2018 9:00 AM	4/16/2018 10:00 AM	47
Sophy Nicalos	4/17/2018 2:00 PM	4/17/2018 3:00 PM	49
Sophy Nicalos	4/17/2018 3:00 PM	4/17/2018 4:00 PM	50
Sophy Nicalos	4/18/2018 1:00 PM	4/18/2018 2:00 PM	51
Sophy Nicalos	4/18/2018 2:00 PM	4/18/2018 3:00 PM	52
Phil Gerton	4/16/2018 1:00 PM	4/16/2018 2:00 PM	56
Phil Gerton	4/17/2018 11:00 AM	4/17/2018 12:00 PM	58
Phil Gerton	4/18/2018 1:00 PM	4/18/2018 2:00 PM	59
Phil Gerton	4/18/2018 2:00 PM	4/18/2018 3:00 PM	60
Jean Marchal	4/16/2018 11:00 AM	4/16/2018 12:00 PM	61
Jean Marchal	4/17/2018 10:30 AM	4/17/2018 11:00 AM	63
Jean Marchal	4/17/2018 11:00 AM	4/17/2018 11:30 AM	64
Jean Marchal	4/17/2018 11:30 AM	4/17/2018 12:00 PM	65

Schema



availability (dbo)

- Id
- DoctorId
- Date
- AvailableFrom
- AvailableTo
- AppointmentDuration

doctors (dbo)

- Id
- FirstName
- LastName
- FullName
- Speciality
- Phone
- Email
- UserName
- LoginPassWord

users (dbo)

- Id
- UserName
- LoginPassWord
- UserRole
- DoctorId
- PatientId

timeslots (dbo)

- Id
- AvailabilityId
- SlotStart
- SlotEnd
- SlotDoctorId
- IsAvailable

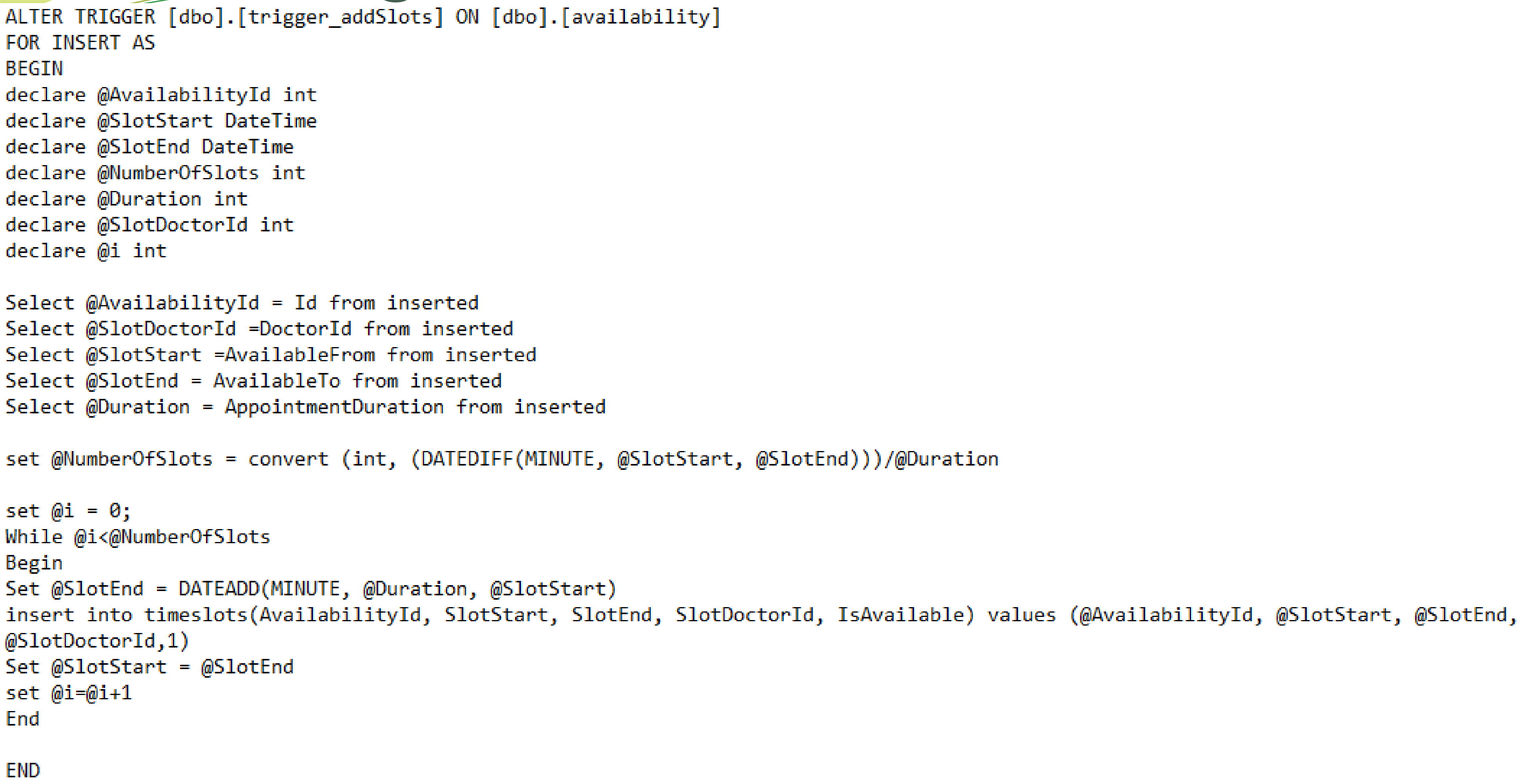
patients (dbo)

- Id
- FirstName
- LastName
- FullName
- Phone
- Email
- DateOfBirth
- MedicalCardNo
- UserName
- LoginPassWord

appointments (dbo)

- Id
- PatientId
- TimeSlotId

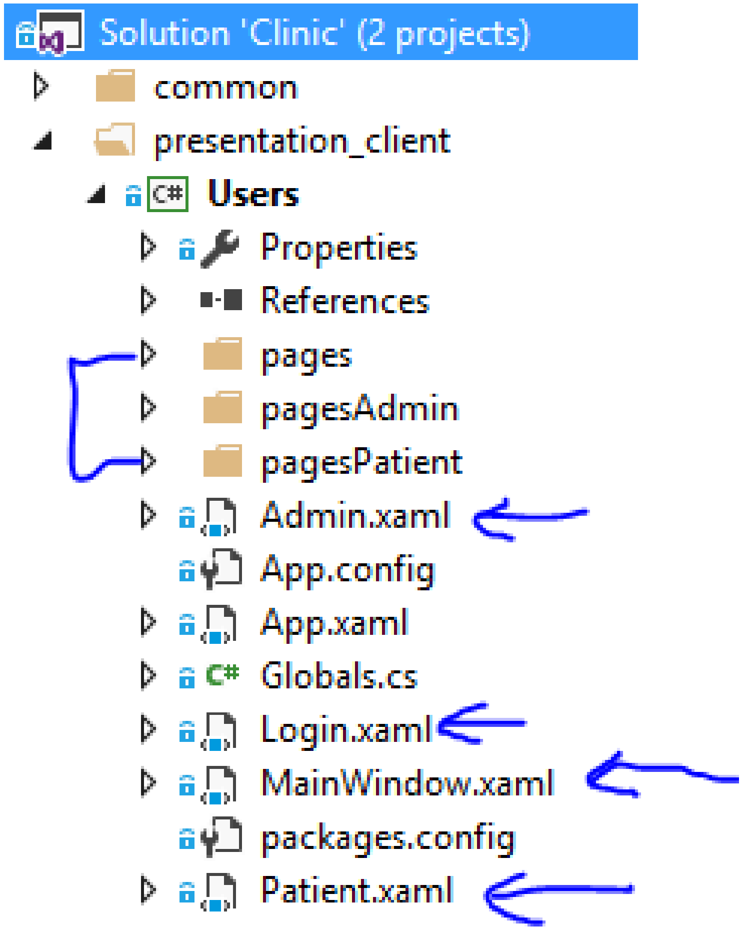
Challenge: time slots



```
ALTER TRIGGER [dbo].[trigger_addSlots] ON [dbo].[availability]
FOR INSERT AS
BEGIN
declare @AvailabilityId int
declare @SlotStart DateTime
declare @SlotEnd DateTime
declare @NumberOfSlots int
declare @Duration int
declare @SlotDoctorId int
declare @i int

Select @AvailabilityId = Id from inserted
Select @SlotDoctorId =DoctorId from inserted
Select @SlotStart =AvailableFrom from inserted
Select @SlotEnd = AvailableTo from inserted
Select @Duration = AppointmentDuration from inserted

set @NumberOfSlots = convert (int, (DATEDIFF(MINUTE, @SlotStart, @SlotEnd)))/@Duration

set @i = 0;
While @i<@NumberOfSlots
Begin
Set @SlotEnd = DATEADD(MINUTE, @Duration, @SlotStart)
insert into timeslots(AvailabilityId, SlotStart, SlotEnd, SlotDoctorId, IsAvailable) values (@AvailabilityId, @SlotStart, @SlotEnd,
@SlotDoctorId,1)
Set @SlotStart = @SlotEnd
set @i=@i+1
End

END
```

Structure of projects

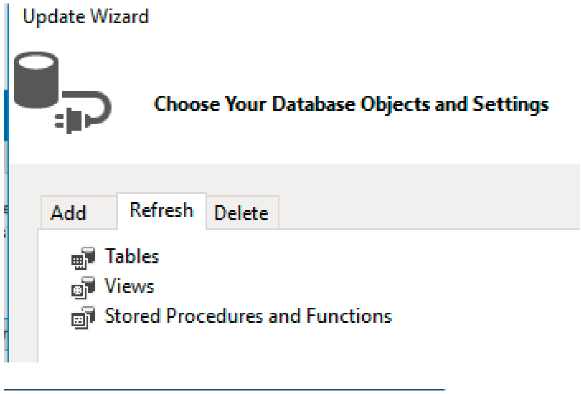
- Common- entity framework with all the classes reflect to tables of database
- User – presentation – login is the entry point of the program then direct to different windows as doctor, patient, or admin.
- In each of those window, has pages to navigate through different functions.
- Has one Globals class to keep the information which user is logged in



```
Solution 'Clinic' (2 projects)
  common
  presentation_client
    Users
      Properties
      References
      pages
      pagesAdmin
      pagesPatient
      Admin.xaml
      App.config
      App.xaml
      Globals.cs
      Login.xaml
      MainWindow.xaml
      packages.config
      Patient.xaml
```

challenge – using entity framework for this project

- I adopted database first approach.
- While binding tables / views to UI controls is not a problem, but while the project is going, I need to add and modify views in the database, then have to find way to update model in project as well.
- I tried to update through refresh, and always have a bunch of errors. Then I figured out whenever update a table, or a view, first delete that table or view. Then add through update wizard Add, that will not give error for mapping.

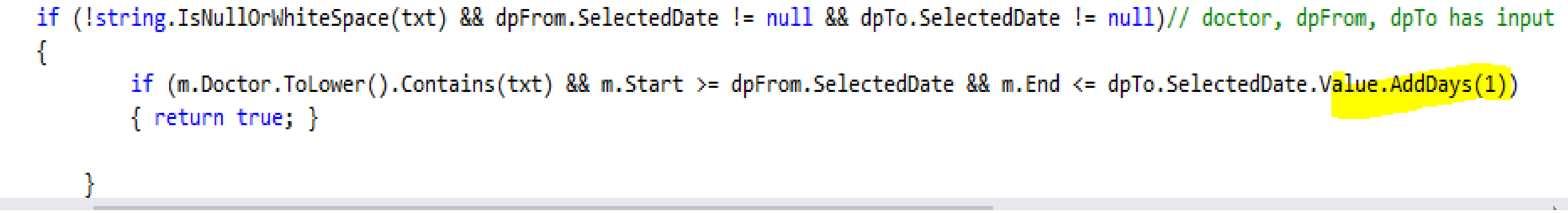

Update Wizard
Choose Your Database Objects and Settings
Add
Refresh
Delete
Tables
Views
Stored Procedures and Functions

What we learned

- Working with dates in C# and WPF

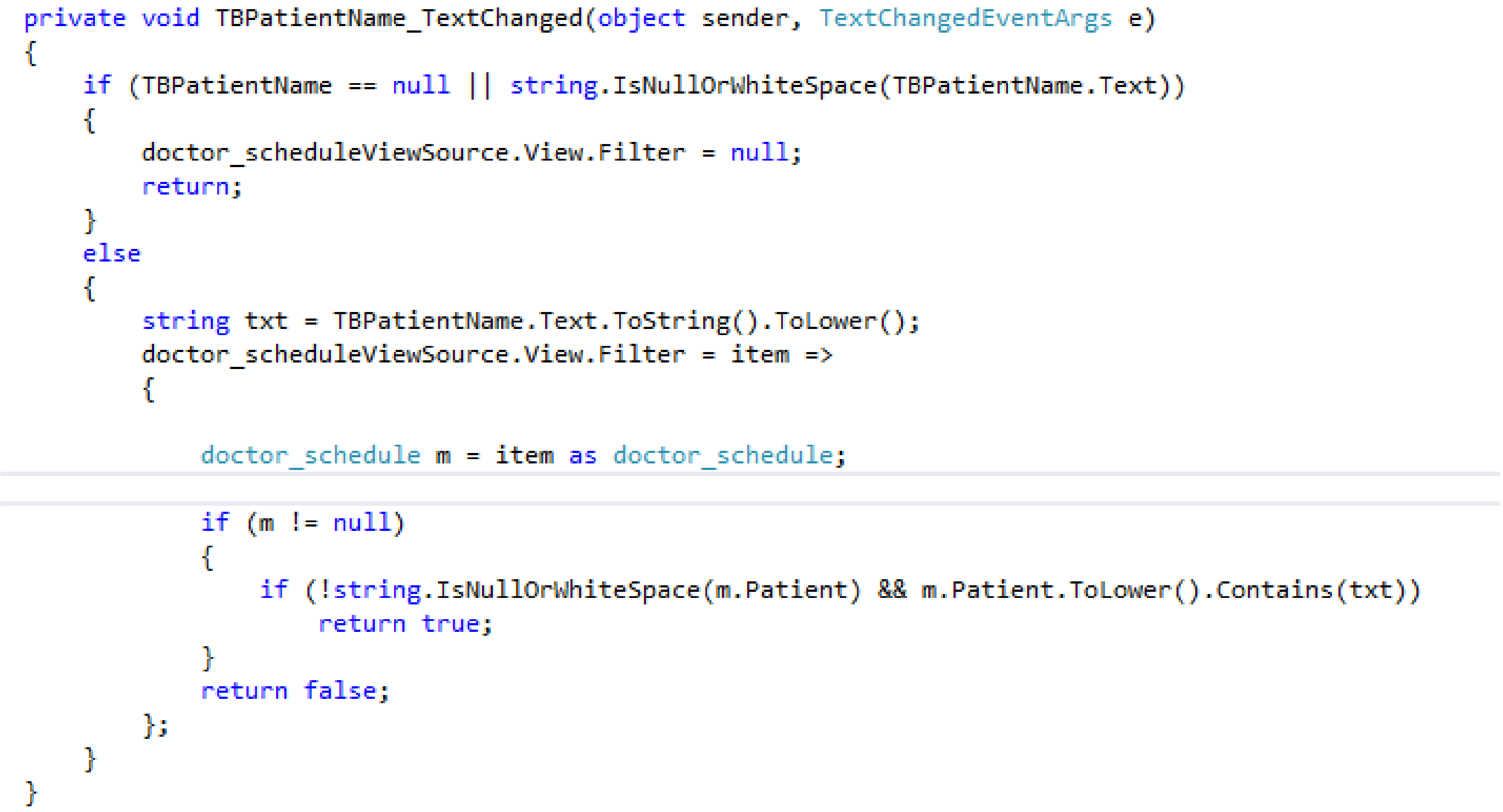
The DateTime is the the data type to be used in C#, and by default it is set to the a date with a time as 12:00 am.

When I tried to set up the search by date From and To, I realize there is something wrong, it does not work properly, that is because the date To was set with a time 12:00 am. So I need to add one day for the search to work.



```
if (!string.IsNullOrWhiteSpace(txt) && dpFrom.SelectedDate != null && dpTo.SelectedDate != null)// doctor, dpFrom, dpTo has input
{
        if (m.Doctor.ToLower().Contains(txt) && m.Start >= dpFrom.SelectedDate && m.End <= dpTo.SelectedDate.Value.AddDays(1))
        { return true; }

   }
```

Dynamically filter the data



```
private void TBPatientName_TextChanged(object sender, TextChangedEventArgs e)
{
    if (TBPatientName == null || string.IsNullOrWhiteSpace(TBPatientName.Text))
    {
        doctor_scheduleViewSource.View.Filter = null;
        return;
    }
    else
    {
        string txt = TBPatientName.Text.ToString().ToLower();
        doctor_scheduleViewSource.View.Filter = item =>
        {

            doctor_schedule m = item as doctor_schedule;

            if (m != null)
            {
                if (!string.IsNullOrWhiteSpace(m.Patient) && m.Patient.ToLower().Contains(txt))
                    return true;
            }
            return false;
        };
    }
}
```

Future improvement

- Add user registration
- Add some data entry validation
- Add report function to print out the schedule
- Add email reminder 48 hours before the appointments to patients

Summary

- The functions planned for different users are all implemented and working.
- Did not use the scheduler control at this point because the third party tool DevExpress who provides the scheduler requires a implementation of MVVM (Model–View–ViewModel), I am not sure how to use a structure of MVVM.
- Overall I am satisfied with the result achieved so far.