RMS Design Document

Prepared by: Abdul Halim

Date of Submission: Jun 29, 2013

Contents

[Overview 1](#_Toc138971574)

[Prerequisite: 2](#_Toc138971575)

[Swagger screen shot 2](#_Toc138971576)

[Connect API with Razor web App: 2](#_Toc138971577)

[API description: 2](#_Toc138971578)

[Deployment Diagram: 4](#_Toc138971579)

[API design: 4](#_Toc138971580)

[CQRS and mediator pattern: 4](#_Toc138971581)

[Why not minimal API chosen: 5](#_Toc138971582)

[Database design 5](#_Toc138971583)

[UI Design: Web UI is designed with Razor page. 7](#_Toc138971584)

[UI screen shot: Below are some screen shot for UI design for CRUD operation with API. 7](#_Toc138971585)

[Assumptions: 10](#_Toc138971586)

[Future enhancement: 10](#_Toc138971587)

# Overview

This document describes the design the Referral management system based on the requirement provided in the assessment. It is also describe the steps How to run the code that has been written for the assessment test.

# Prerequisite:

Tools required to run the API

1. Lasted release of Visual studio 2022
2. Programing language C#
3. SQL Server to create and connect database
4. Management studio to run attached SQL script

Download and run the Api code:

1. Download the code from github link:
2. Run the SQL script provided to create RMS database (file Name: SQLServer\_Database\_Scripts\_final.sql)
3. Open the RMSTestApi.sln file with Visual Studio 2022
4. Replace the connection string values from the below DBcontext.

RMSTestApi.Data->RmsDbContext.cs

protected override void OnConfiguring(DbContextOptionsBuilder options)

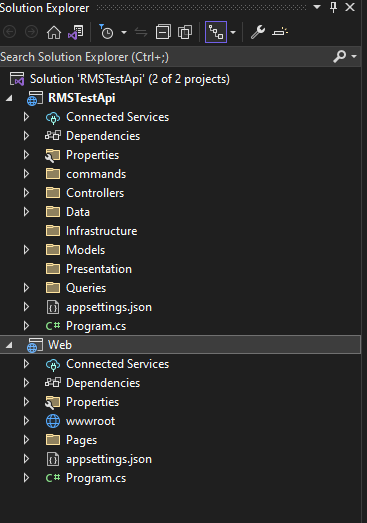
{

string ? conn= @"Server=H505495\SQLEXPRESS;Database=RMS;user id=sa;password=sa@1234;TrustServerCertificate=True";

options.UseSqlServer(conn);

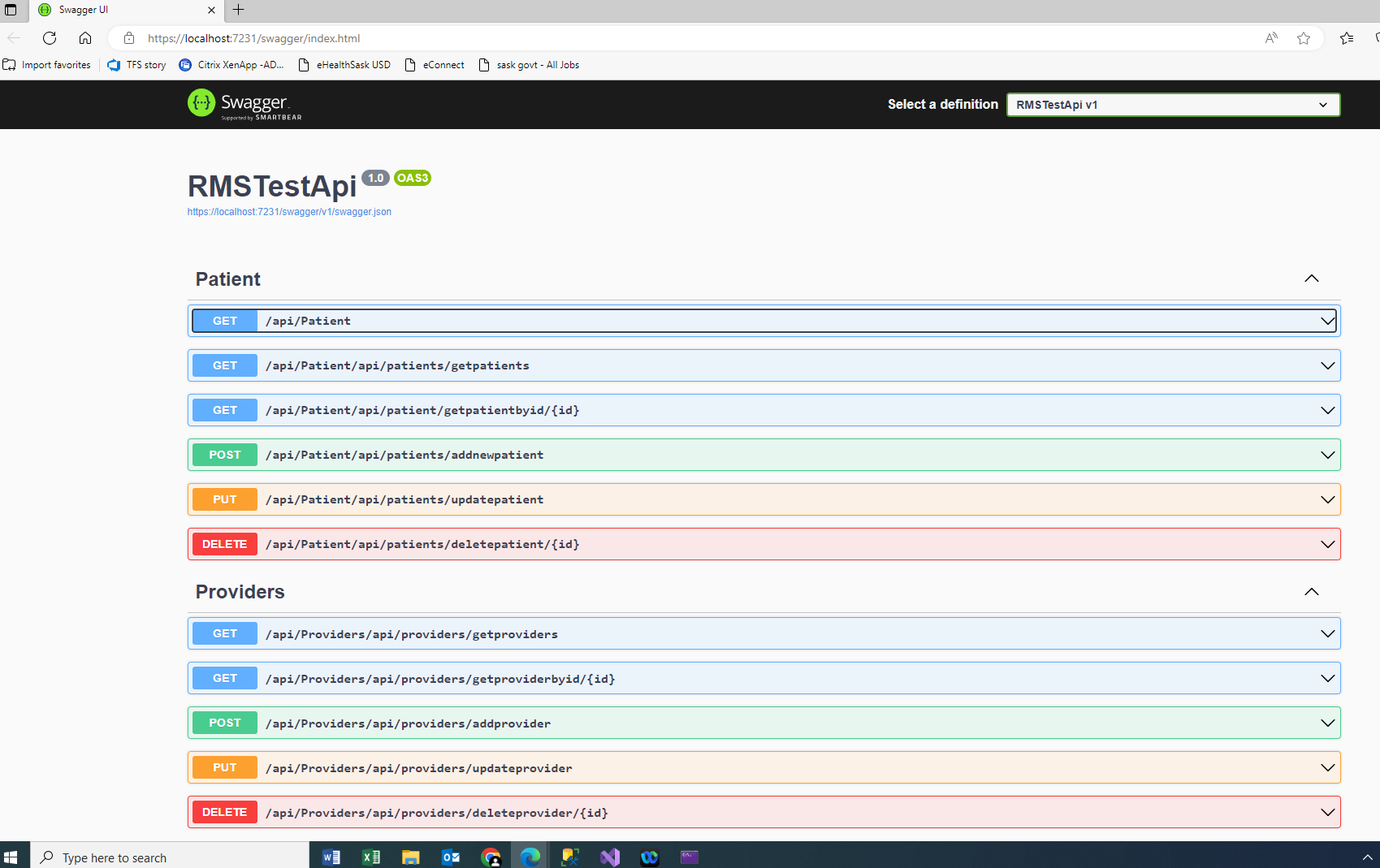
}

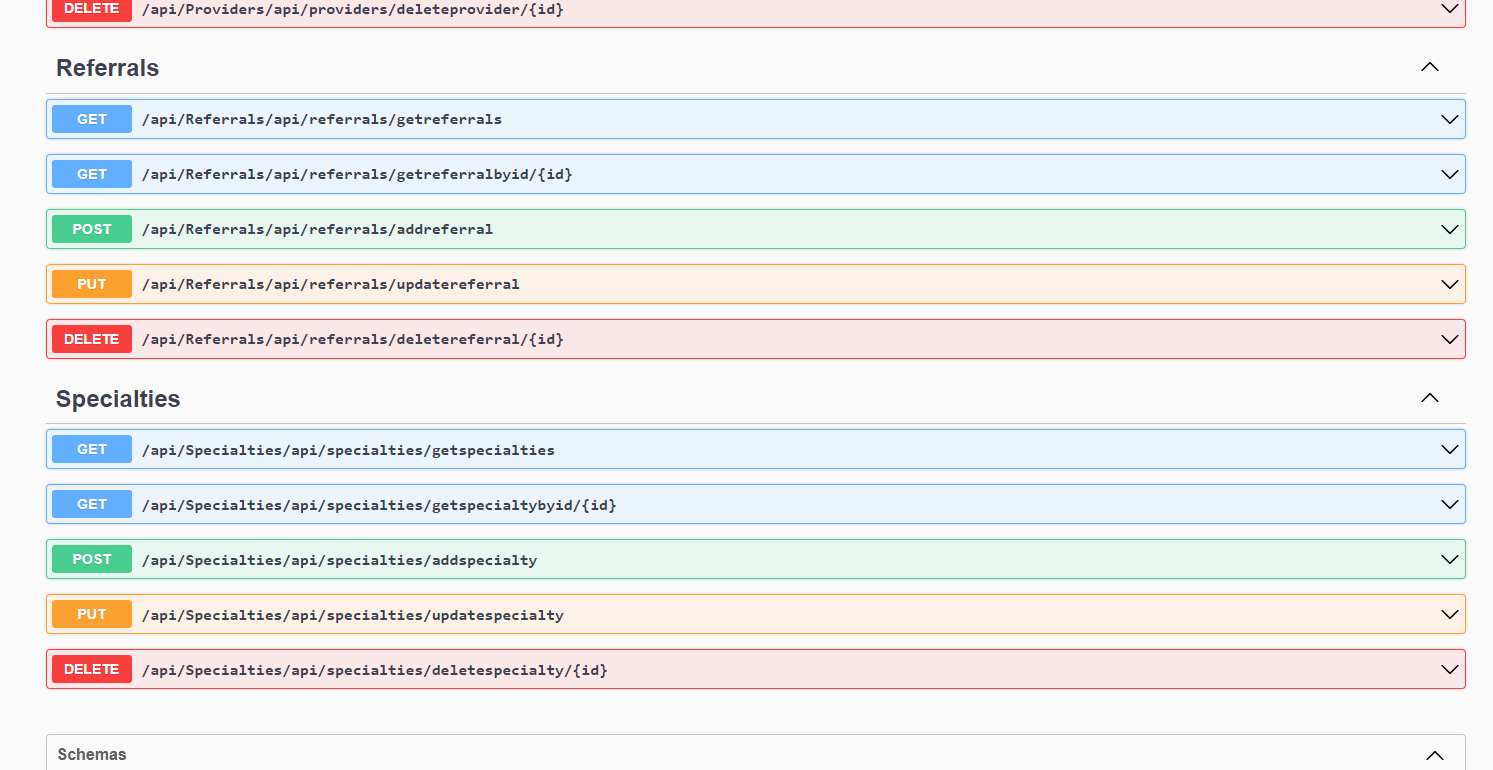
1. To rest he API with swagger or postman , Make the project RMSTestApi as start up project. Then Run the API code from Visual studio(ctrl+f5). It is swagger enabled so API should be shown in the swagger window.
2. To run Web UI. Select the **Web** Project as start up project and run the application with (Ctrl+F5). It should show the Website pages to CRUD operation from UI.



# API description:

Below are the screen shot in swagger RMS API endpoints for CRUD operation?





# Deployment Diagram:



# API design:

## CQRS and mediator pattern:

API is designed with Command Query Responsibility Segregation (CQRS) and Mediator pattern. CQRS pattern separate the load of reading and writing operation. So the operations can be scaled independently. Mediator pattern control the communication between objects. For the large-scale complex application, it is easier to maintain and scaled. It also increase code readability. Although it was intended for writing all APIs in this patter but due to the time constraints, it was not applied for all of the APIs. I kept this option for future refinement.

## Why not minimal API chosen:

New and simplified way of writing API is introduced in .NET 6. Since it is introduced recently the technology is not yet matured with all features and support for example filter option is not introduced in .NET 6. So although it is easier way to write API but I did not chose this option. As the technology become mature this could be the way of writing API in future but not yet in my opinion.

Why ASP.NET core 6 is chosen: it is the latest long term support version but ASP. NET 7 is a short term support version. So to get long-term support from Microsoft and longer stability of the application I chose .NET 6.

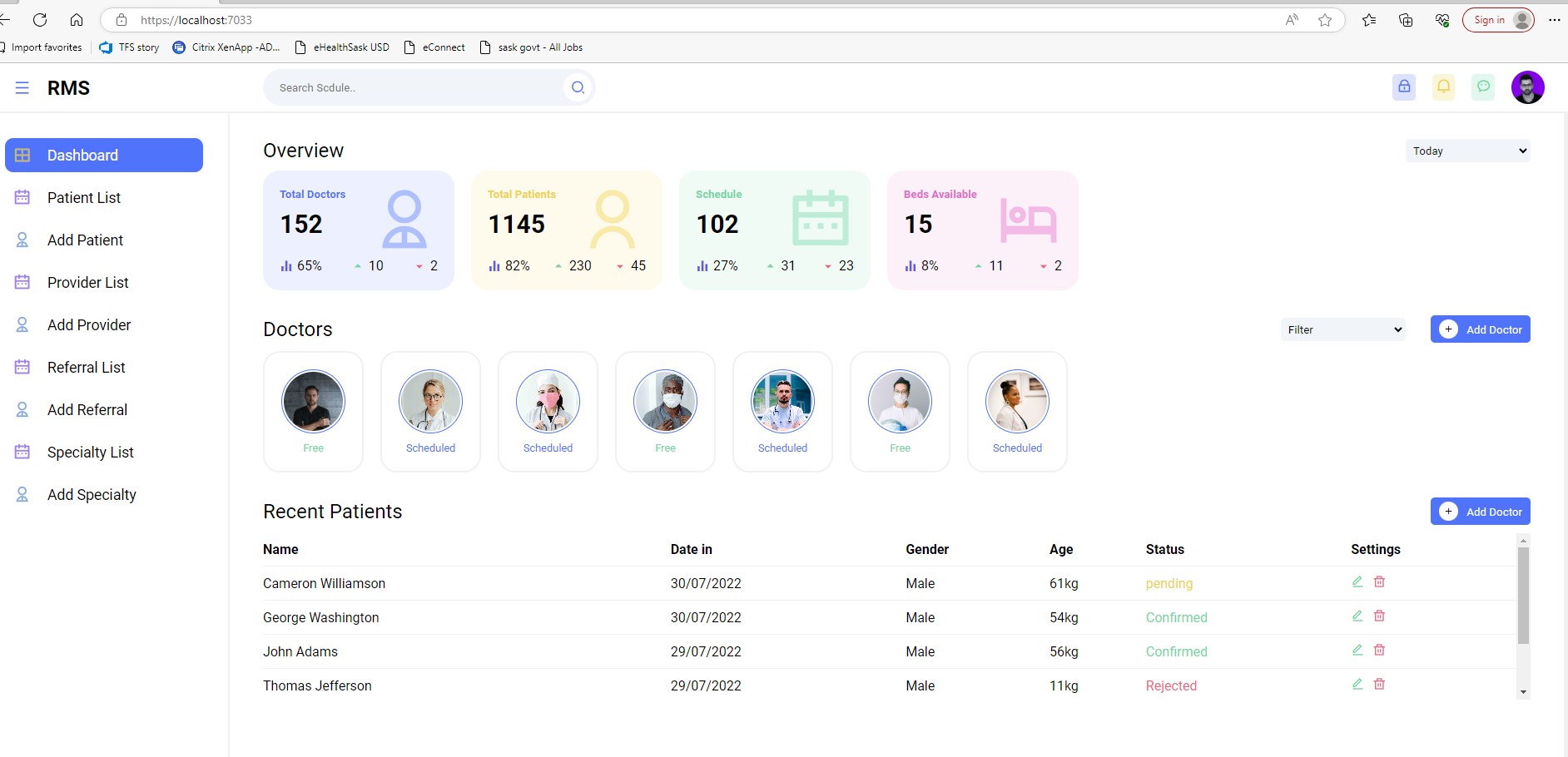
# Database design

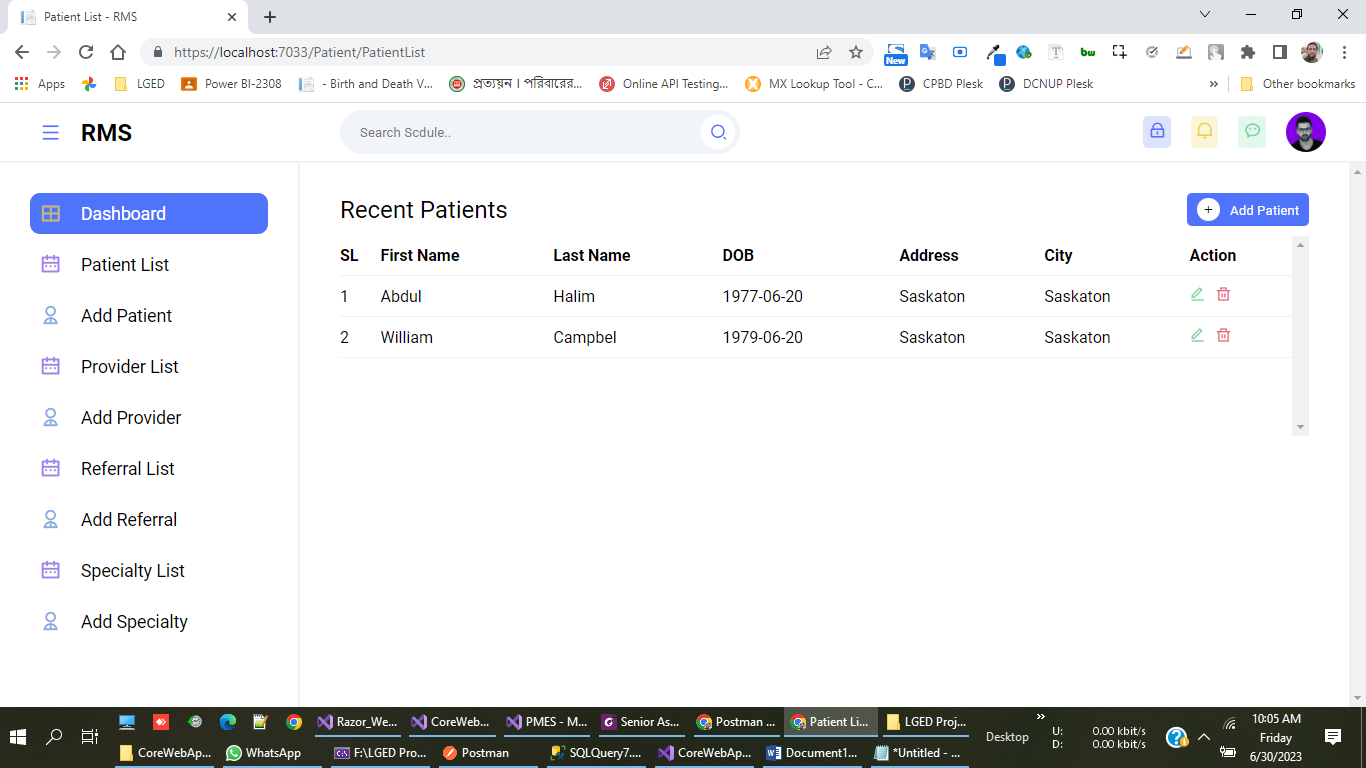
Below is the Visio diagram that shows the tables and their relationship

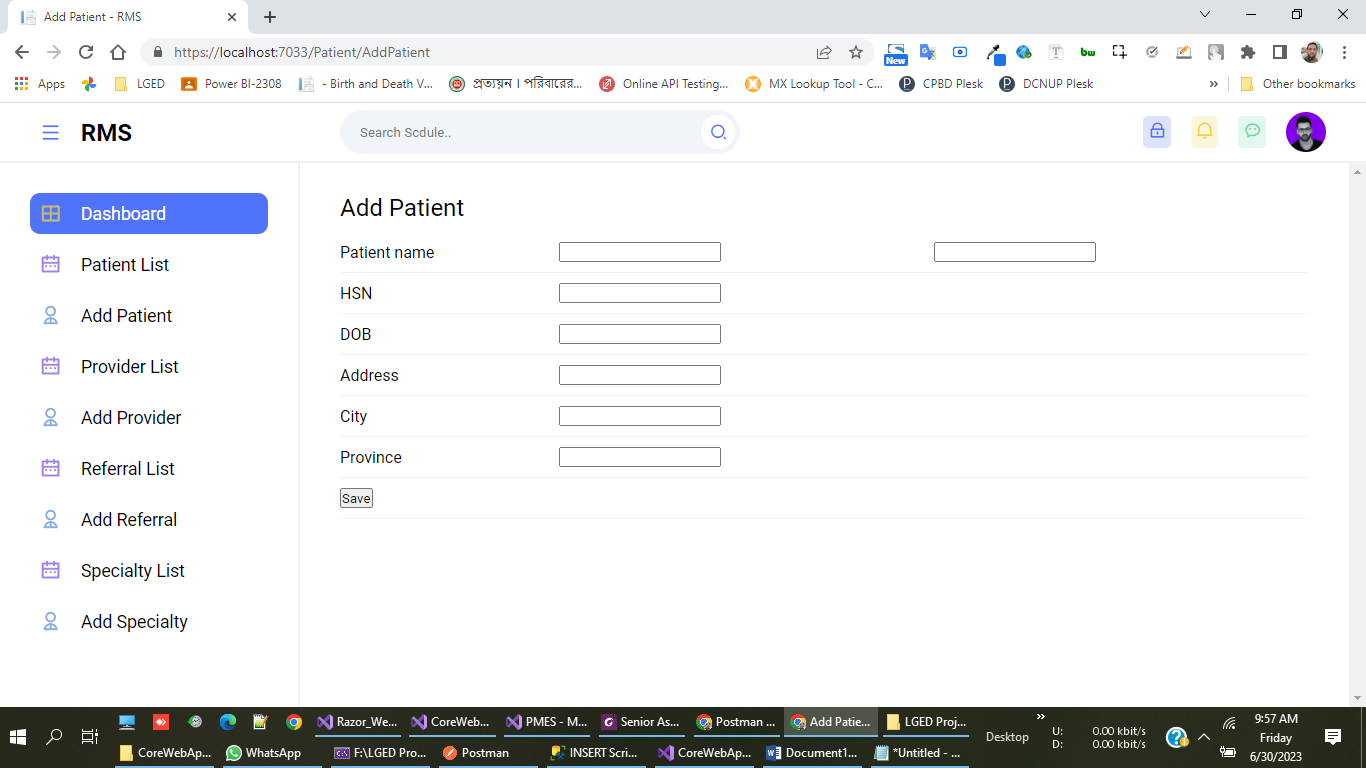


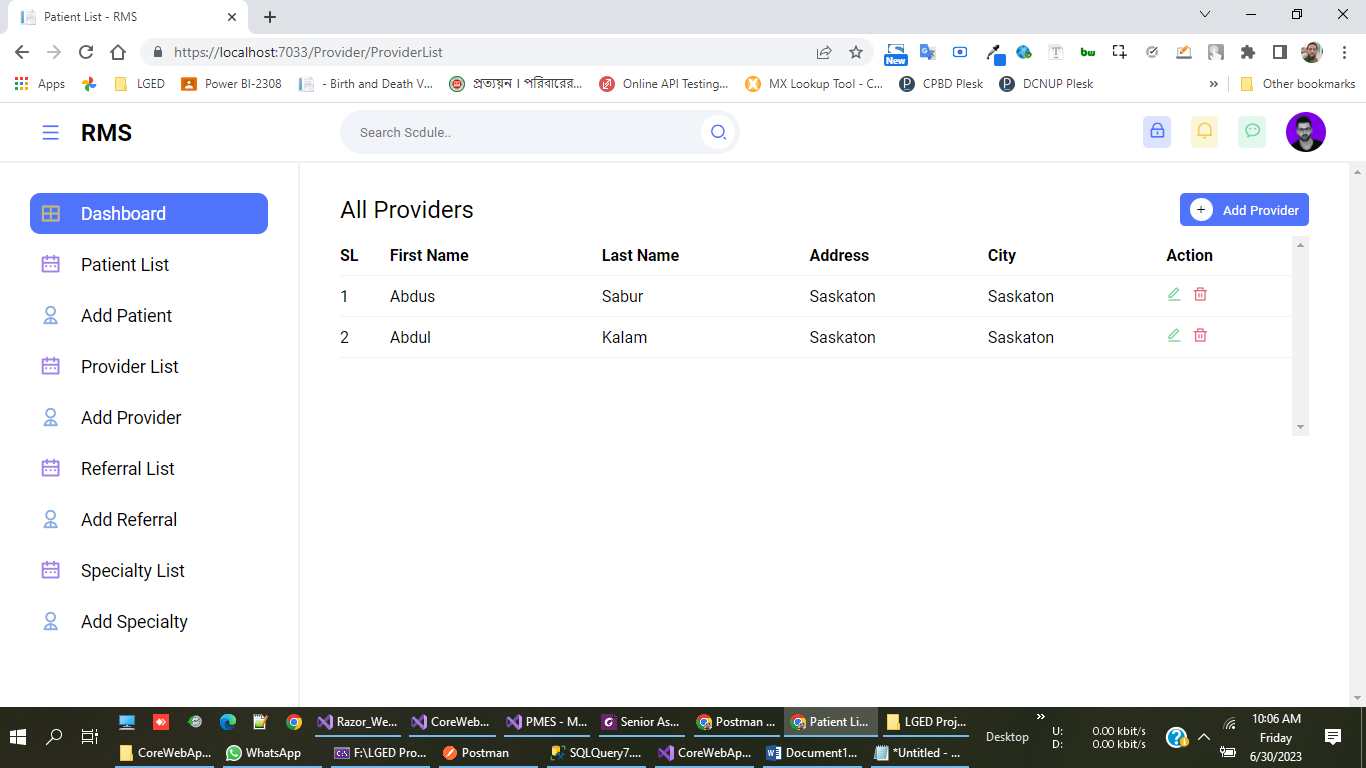
# UI Design: Web UI is designed with Razor page.

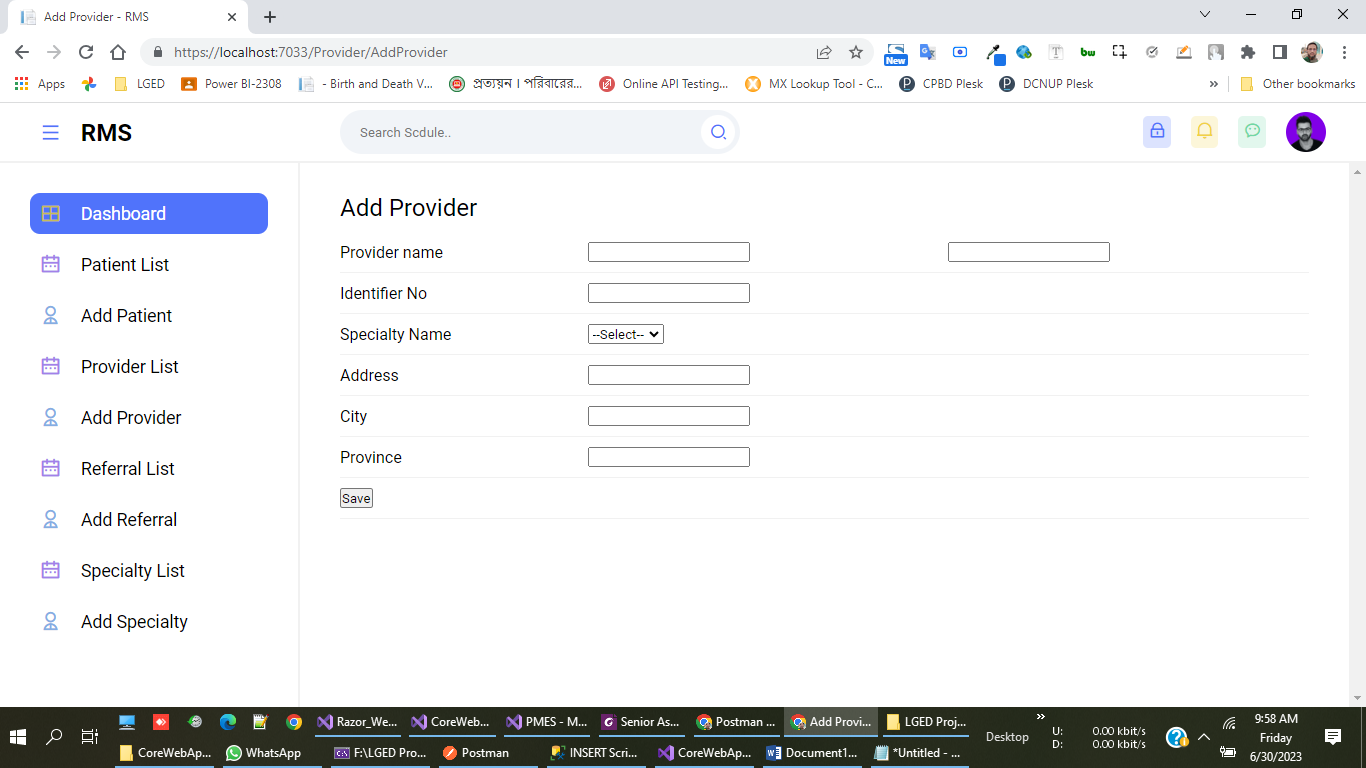
## UI screen shot: Below are some screen shot for UI design for CRUD operation with API.

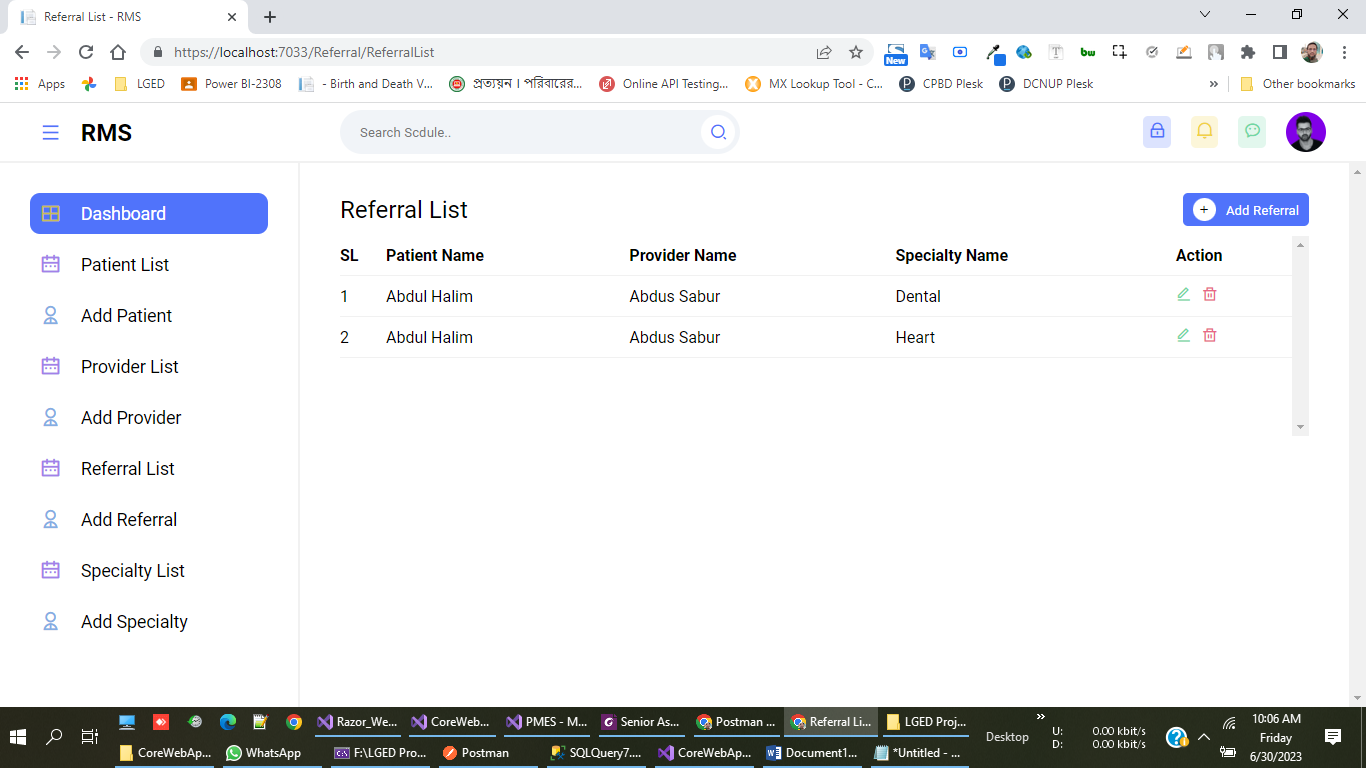


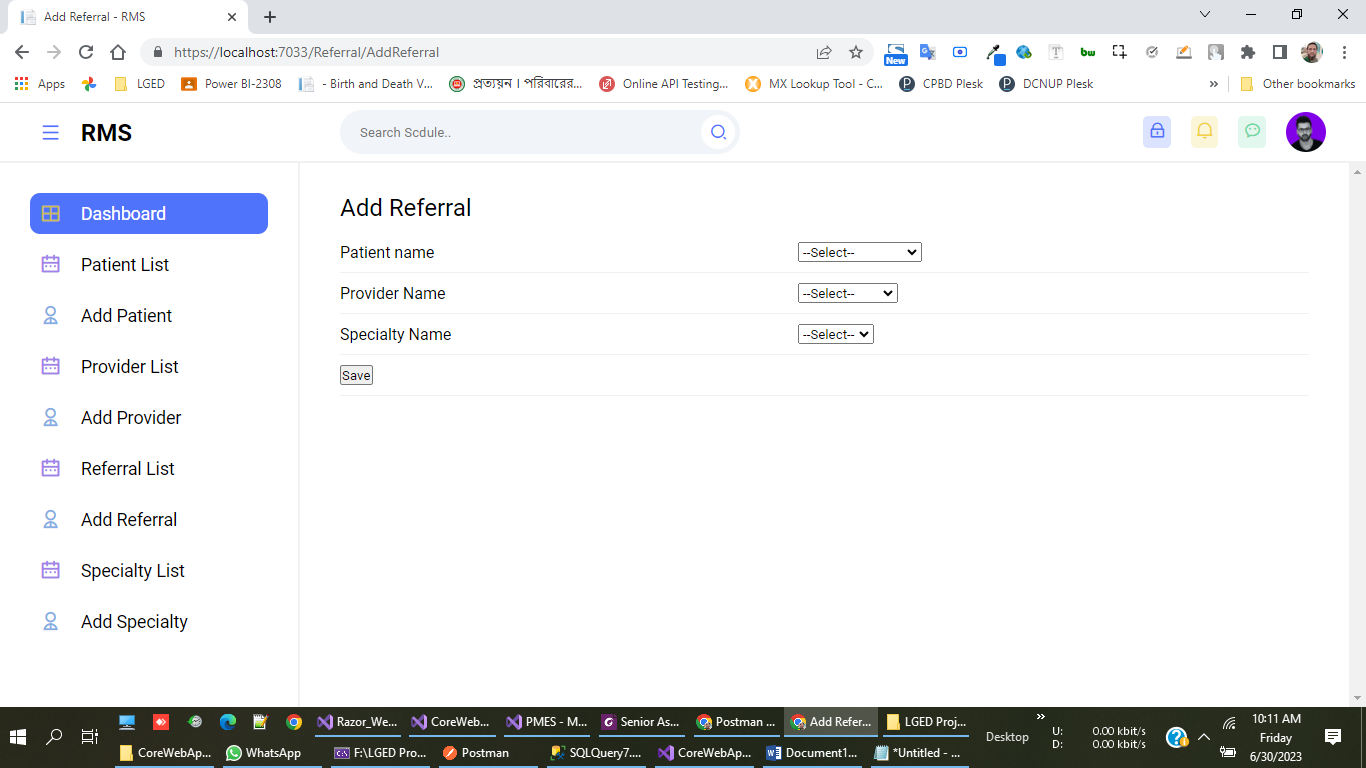


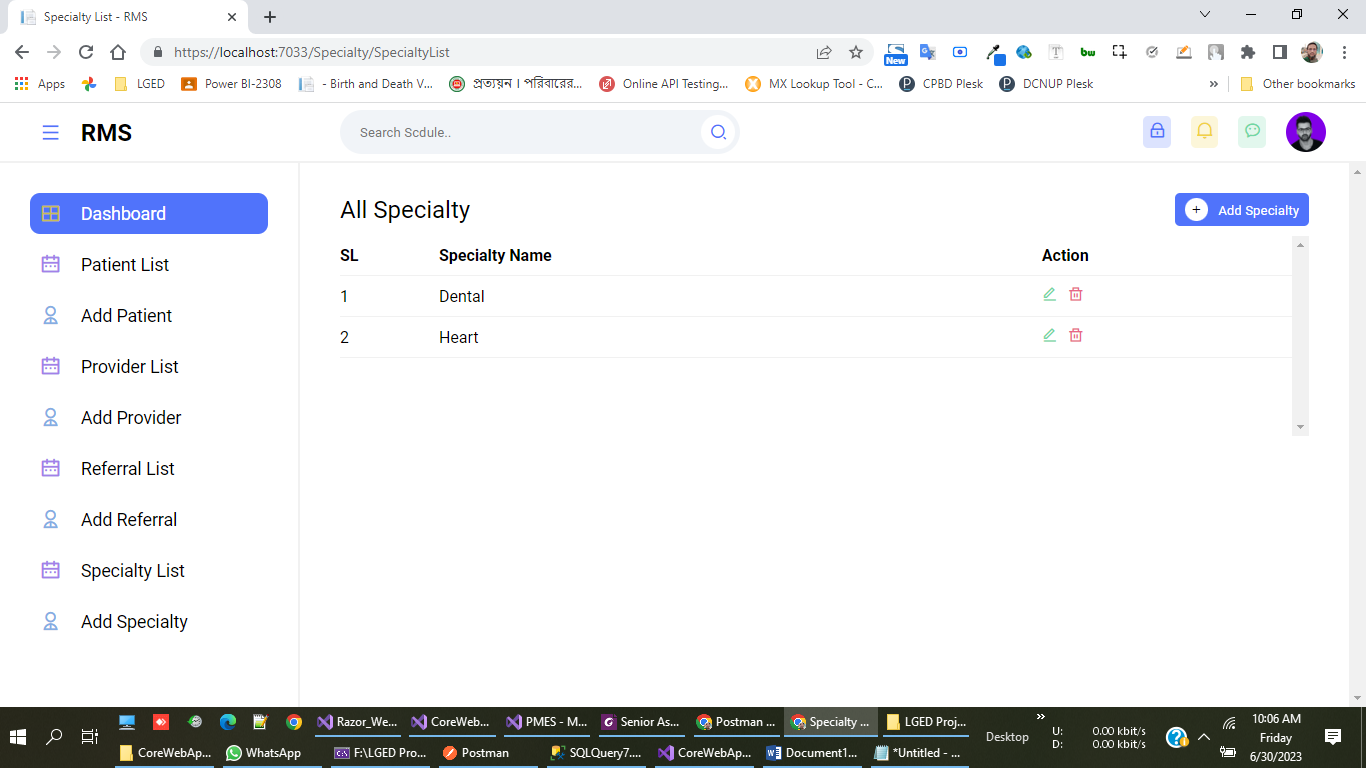


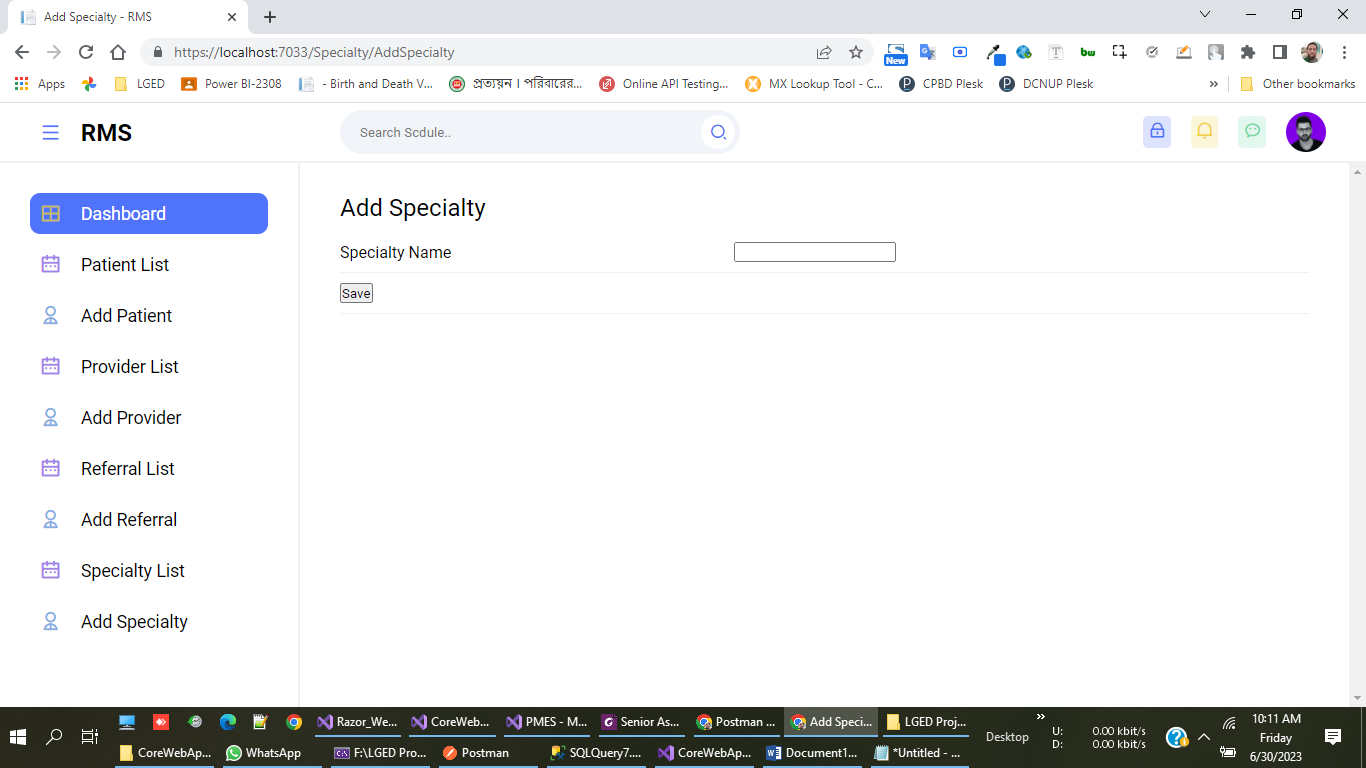












# Assumptions:

1. Patient HSN and Health Coverage is already verified from the PHRS system
2. Provider identifier ID is valid and verified from the PRS system
3. Provider latest availability date is pulled and verified from provider schedule

# Future enhancement:

1. Add the error logging features with SeriLog
2. Complete CQRS and mediator pattern for all the APIs
3. Refine the API design architecture with Clean Architecture. Clean architecture is being very popular for Rest API design as it easier for maintain and testing and separate the API layer so that each is loosely couple with one another.