.NET Core Azure

Sprint-1

Hospital Management System

Document Revision History

Date	Revision No.	Author	Summary of Changes

Table of Contents

INTRODUCTION

This document outlines a project for the .NET Line of Technology (LOT). The project is to develop Hospital Management System. This document contains the requirements, work flow of the system and gives guidelines on how to build the functionality gradually in each of the course modules of the .NET LOT.

SETUP CHECKLIST

Minimum System Requirements

- Intel Pentium 4 and Windows 2010
- Memory 4 GB
- Internet Explorer 11.0 or higher / Chrome
- SQL Server 2014 or 2016 client and access to SQL Server 2014 or 2016 server
- Visual Studio 2019
- Visual Studio Code
- Git

INSTRUCTIONS

- The code modules in the mini project should follow all the coding standards.

PROBLEM STATEMENT

OBJECTIVE

Need to develop Hospital Management System

Abstract of the project:

Hospital Management System (HMS) is an application, which helps to manage Patient (Both inpatient and Outpatient) and their Treatment details. Patient details like Patient ID, Name, Age, Phone No etc. And Treatment details like – Doctor Name, Room No, Date of Admission, and discharge details etc.

HMS aims to manage Patient Lab reports like adding report information – Test type, Test Date, Price & Result. Along with add, system should display all Report summaries (View) and must support search facility. The Application should also generate Bills for the patients like Bill id, Room amount, Total days, Theatre amount (if applicable), medicine bill, Total Amount, etc...

1. Adding Patient Information
2. Modify Patient Info
3. Add Patient Treatment Information
4. View Patient History
5. Search Patient Info
6. Generate Lab Report
7. Generate Medical Bills

Phase 1: Create Client Application in Angular and Services using ASP.NET Core Web API.

MODULE LIST and MODULE DETAILS

CREATE PATIENT DATA

Following info need to capture

- Patient ID (Must be Unique)
- Name
- Gender
- Age
- Address
- Phone No
- Weight

CREATE PATIENT APPOINTMENT DATA (INPATIENT / OUTPATIENT)

Following info need to capture

- Appointment ID (Must be Unique)
- Patient ID
- Doctor Name (Based on Disease)
- Room No
- Date of Visit (For Inpatient)
- Admission Date (For Outpatient)
- Discharge Date (For Outpatient)
- Remarks

SEARCH PATIENT

Admin / Hospital Staff should be able to search a Patient details by Patient ID, Date of Visit, Doctor Name, Date of Admission or Discharge. They may use any one, two or all the applicable parameters to search a patient.

MODIFY PATIENT

Search (Patient Id) Patient and modify the information. System should show existing data/info of the patient and should support modify the eligible details.

FILM SUMMARY (VIEW)

System should show (display) patient / patient list in a tabular format (one row for each patient, and columns for patient info). It is not required to show all the patient data. Only important info like – Name, Date of Visit, Doctor name, etc… can be displayed in table.

LAB REPORT

Patients can take various medical test as per doctor's prescription. After completion, they will get lab report which contains information like – Report ID, Name, Test Date, Doctor, Test Type & Remarks.

BILL GENERATION

After completion of treatment, system should generate Bill which contains information like – Bill ID, Room Amount (Inpatient), Operation Theatre amount (Inpatient), Medical Bill & Lab Bill.

Constrains

- Proper validation is required (especially Patient Id, Age, Amount, etc…) System must check uniqueness on film Patient ID, Report ID, Bill ID.
- System must show appropriate massage on all activity (whether activity is successful or failure)
- User must have proper menu to select the activity (create, modify, search, view, remove) that user want to perform.

In this course you need to develop the user interface using Angular. The screens should include the fields as per the functionality mentioned above. Include Validations wherever required.

Create the following tables

Table Name: Patient		
Field Name	**Constraints**	**Data Type**
Patient id	Primary Key	Text
Name		Text
Age		Number
Weight		Number
Gender		Text
Address		Text
Phone no		Number
Disease		Text
Doctor id	Foreign Key	Text

Table Name: Doctor		
Field Name	**Constraints**	**Data Type**
Doctor id	Primary Key	Text
Doctor name		Text
Dept		Text

Table Name: Lab		
Field Name	**Constraints**	**Data Type**
Lab Id	Primary Key	Text
Pid	Foreign Key	Text
Doctor id		Text
Test Date		Date
Test Type		Text
Patient Type		Text

Table Name: Inpatient		
Field Name	**Constraints**	**Data Type**
Pid	Primary Key	Text
Room No	Foreign Key	Text
Doctor id	Foreign Key	Text
Admission Date		Date
Discharge Date		Date
Lab No	Foreign Key	Text
Amount Per Day		Number

Table Name: Outpatient		
Field Name	**Constraints**	**Data Type**
Pid	Primary Key	Text
Treatment Date		Date
Doctor id	Foreign Key	Text
Lab No	Foreign Key	Text

Table Name: Room Data		
Field Name	**Constraints**	**Data Type**
Room No	Primary Key	Text
Treatment Date		Date
Doctor id	Foreign Key	Text
Lab No	Foreign Key	Text

Table Name: Bill Data		
Field Name	**Constraints**	**Data Type**
Bill No	Primary Key	Text
Pid	Foreign Key	Text
Patient Type		Text
Doctor id	Foreign Key	Text
Doctor Fees		Number
Room Charge		Number
Operation Charges		Number
Medicine Fees		Number
Total Days		Number
Lab Fees		Number
Total Amount		Number

Note : You may add/normalize/denormalize the tables if your application so demands it.

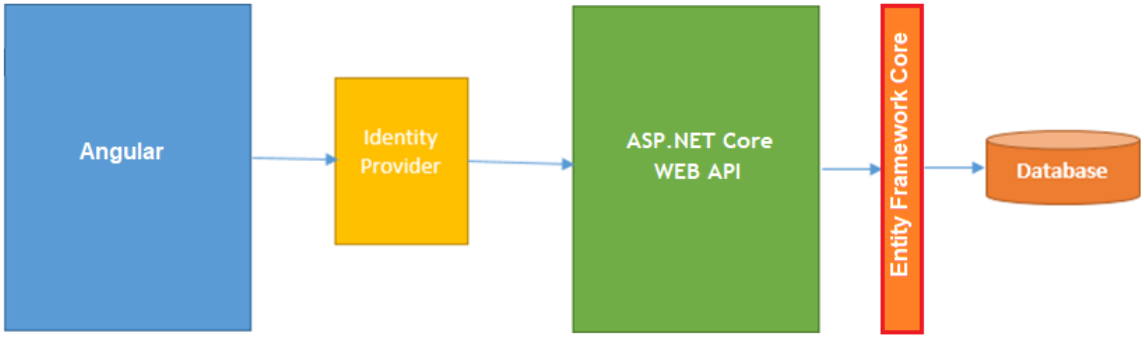
PROJECT STRUCTURE

Description: Create below sample reference project structure, which will help to reuse most of module for Web application. You are allowed to bring your own project structure and project to achieve the requirement.

In this sprint you have to use

- Angular UI as the presentation layer
- ASP.NET Core WEB API to create services
- Entity Framework Core to interact with the database.

The project should have minimum target framework – .Net Core 3.1 for Core WEB API Services and Angular 10 for UI Design



Design guidelines

- All the exceptions/errors to be captured and user friendly message to be displayed on the Common Error page.

IMPLEMENTATION

SUMMARY OF THE FUNCTIONALITY TO BE BUILT:

The participants need to develop the Hospital Management System by building the functionality incrementally in each of the course modules of .NET LOT.

Sr. No	Course	Duration (in PDs)	Functionality to be built
1	Angular ASP.NET Core Web API Entity Framework Core SQL Server	5	Developing Presentation components (Angular), Business components (ASP.NET Core WEB API) and Data access components (Entity Framework Core)

