|  |  |
| --- | --- |
|  | **Cognizant Academy**  **Covid Vaccination Center App**  **ASP.Net MVC, Entity Framework, SQL Server - Integrated Capability Test**  **Version 0.1** |
|  |
|  |

Table of Contents

[1.0 Introduction 3](#_Toc62106646)

[1.0 Purpose of this document 3](#_Toc62106647)

[2.0 Definitions & Acronyms 3](#_Toc62106648)

[3.0 Project Overview 3](#_Toc62106649)

[4.0 Scope 4](#_Toc62106650)

[5.0 Hardware and Software Requirment 4](#_Toc62106651)

[2.0 System diagram 5](#_Toc62106652)

[3.0 Architecture 5](#_Toc62106653)

[4.0 Solution Creation 6](#_Toc62106655)

[5.0 Add vaccination center details 7](#_Toc62106656)

[1.0 Requirement flow 7](#_Toc62106657)

[2.0 Technical guidelines 9](#_Toc62106658)

[1. Implementing POCO/Entity class 9](#_Toc62106659)

[2. Implementing DataContext 10](#_Toc62106660)

[3. Implementing Repository class 11](#_Toc62106661)

[4. Implementing Controllers and Views 11](#_Toc62106662)

[6.0 Search vaccination centers 12](#_Toc62106663)

[1.0 Requirement flow 12](#_Toc62106664)

[2.0 Technical guidelines 14](#_Toc62106665)

[1. Implementing ViewModel class 14](#_Toc62106666)

[2. Implementing Repository class 15](#_Toc62106667)

[3. Implementing Controllers and Views 15](#_Toc62106668)

[7.0 List vaccination centers 16](#_Toc62106669)

[1.0 Requirement flow 16](#_Toc62106670)

[2.0 Technical guidelines 17](#_Toc62106671)

[1. Implementing Repository class 17](#_Toc62106672)

[2. Implementing Controllers and Views 17](#_Toc62106673)

[8.0 Evaluation Areas 18](#_Toc62106674)

# Introduction

## Purpose of this document

The purpose of this document is outline the specification for a MVC application which uses Entity Framework Code-First approach to connect a database and allows its users to keep a record of vaccination centers across the cities.

## Definitions & Acronyms

|  |  |
| --- | --- |
| Definition / Acronym | Description |
| C# | C# (prounced C sharp) is an object-oriented server-side programming language for developing .NET application |
| SQL Server | SQL Server is a powerful relational database for storing data |
| Asp.NET MVC | Light weight framework for developing server-side application which provides separation of concerns for developing applications using asp.net |
| Entity framework | Provides an ORM to map a relational model with the object oriented model. |

## Project Overview

Cowin Authority want to develop a database of vaccination centers for various vaccine categories in the country. As the part of development team for the application you are assigned the task of recording details of vaccination centers into the database. Apart from recording the vaccination center details you are also expected to develop a display and search feature in the application which will allows it’s users to see the details of vaccination centers for a given city and vaccine category type.

Use Case Diagram

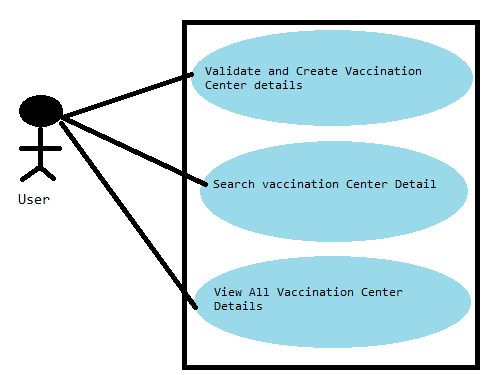


Figure 1: Use Case Diagram

## Scope

1. Creating an asp.net mvc application to store and retrieve data from sql server database using entity framework code first approach.

## Hardware and Software Requirment

1. Hardware Requirement:
   1. Developer PC with 8GB RAM
2. Software Requirement
   1. .Net Framework 4.5
   2. Visual Studio Professional Edition 2015
   3. SQL Server enterprise edition 2014
   4. Internet explorer/Google Chrome

# System diagram

Figure 2: System Diagram

# Architecture

Database

Entity Framework

CovidVaccinationCenterApp

Figure 3: Architecture Diagram

# Solution Creation

1. Create a new project in visual studio using the mvc template.
2. Give the project name as “**CovidVaccinationCenter**”
3. Change the authentication mode to “**No Authentication**”
4. Ensure to uncheck the following checkboxes
   1. Configure for Https
   2. Docker support
   3. Create unit test for project
5. Delete the default “**HomController**” and it’s associated views folder.
6. Delete the default “**Error.cshtml**” view from **Views/Shared** folder.
7. Go to **NuGet Package Manager** and install “**EntityFramework**” package.
8. Open Web.config file and add a connection string in **<ConnectionStrings>** with the name as “**SqlCon**”.
9. Keep the values for DataSource, Database and security as per your system configuration.
10. Go to solution explorer, expand “**App\_Start**” folder and modify the “**RouteConfig.cs**” to start the application with “**VaccinationCenter**” controller and “**Add**” action which will be created subsequently during the application development.

**NOTE:** Solution is already created for you on the platform. Do not make any changes to connection string and custom errors section of Web.config present on the platform.

# Add vaccination center details

## Requirement flow

**Steps Explanation**

1. User launches the application and add vaccination center page is displayed to the user as follows

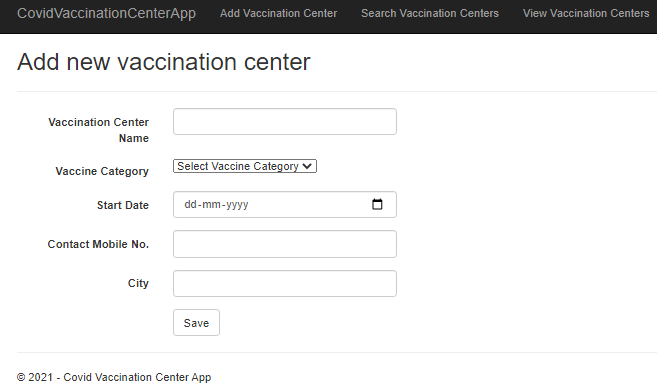


Figure 4 : Add new vaccination center page

1. User will fill up all required details and click the save button which will save the vaccination center details into the database and displays a message to the user as follows

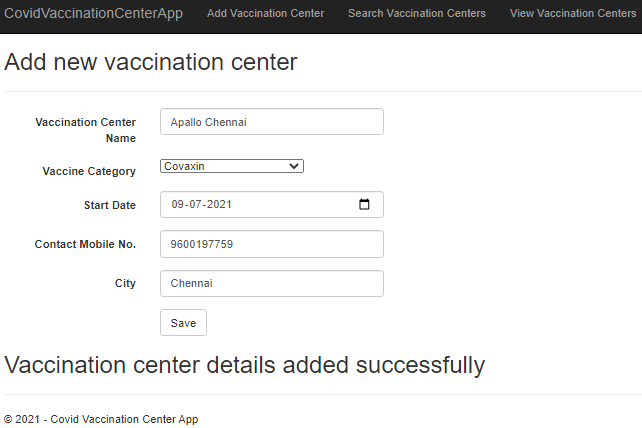


Figure 5 : Add new vaccination center successful

1. If there are any validation failures application will display appropriate validation message as follows

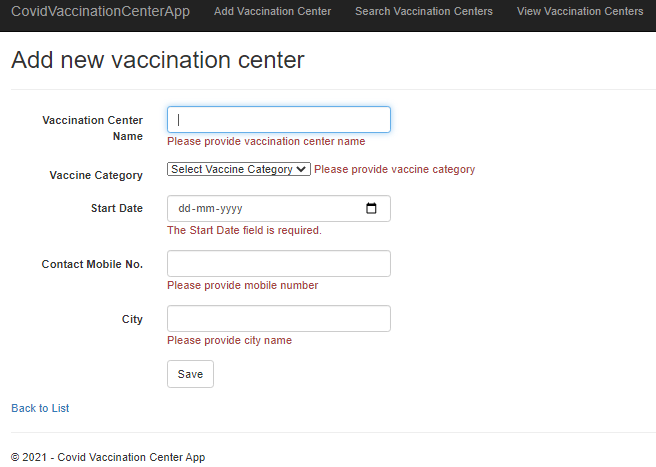


Figure 6 : Add new vaccination center validation failure

1. If the vaccination center details are not saved in the database then application will also display an appropriate error message as follows (In case of repeated details entered or start date is earlier than current date)

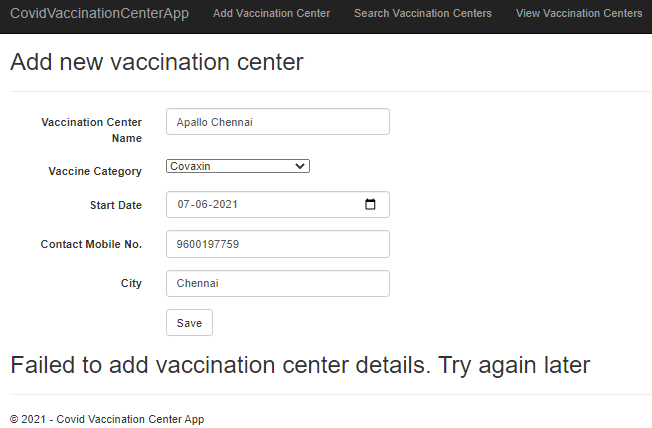


Figure 7 : Add new vaccination center failed

## Technical guidelines

### Implementing POCO/Entity class

1. Create a new class in the “Models” folder with the name as “**VaccinationCenters**” with the following specification.

Table 1 : VaccinationCenters class properties

|  |  |  |
| --- | --- | --- |
| Property Name | Type | Modifier |
| Id | int | public |
| VaccinationCenterName | string | public |
| VaccineCategory | string | public |
| StartDate | DateTime | public |
| ContactMobileNo | string | public |
| City | string | public |

1. VaccinationCenters entity class will be used as POCO class for generating the database table and also for creating strongly-typed views for the actions method.
2. Modify the VaccinationCenters entity using DataAnnotations to so that entity framework can generate a table with following specification.

Table 2 : VaccinationCenters class database table specification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Property Name | Mapped Database Column Name | Column Data Type | Is Nullable | Contraint |
| Id | Id | Int | No | Primary key, identity |
| VaccinationCenterName | VaccinationCenterName | nvarchar(25) | No |  |
| VaccineCategory | VaccineCategory | nvarchar(3) | No |  |
| StartDate | StartDate | datetime | No |  |
| ContactMobileNo | ContactMobileNo | nvarchar(10) | No |  |
| City | City | nvarchar(25) | No |  |

1. Modify the “VaccinationCenters” class with appropriate DataAnnotations to match the following validation rules

Table 3 : VaccinationCenters class validation specifications

|  |  |  |  |
| --- | --- | --- | --- |
| Property | Display Attribute Value | Validation | ErrorMessage |
| VaccinationCenterName | Vaccination Center Name | Must not be left blank | Please provide vaccination center name |
|  | Should allow maximum 25 characters only | Vaccination center name cannot exceed 25 characters |
| VaccineCategory | Vaccine Category | Must not be left blank | Please provide vaccine category |
| String length should be 30 characters max |  |
| StartDate | Start Date | Should support entering date only |  |
| ContactMobileNo | Contact Mobile No. | Must not be left blank | Please provide mobile number |
| Should accept exactly 10 digits | Please enter 10 digit mobile number |
| City | City | Must not be left blank | Please provide city name |
| Should allow maximum 25 characters only | City name must not exceed 25 characters |

### Implementing DataContext

1. Go to **Models** folder and create a new datacontext class named “**VaccinationCentersContext**” which inherits from the “**DbContext**” class.
2. Modify the **VaccinationCentersContext to** add following members

Table 4 : VaccinationCentersContext Constructors

|  |  |  |
| --- | --- | --- |
| Constructor Type | Input Parameters | Modifier |
| Default | - | public |

Table 5 : VaccinationCentersContext Properties

|  |  |  |
| --- | --- | --- |
| Property Name | Type | Modifier |
| Centers | DbSet<VaccinationCenters> | public |

1. The default constructor must call the base class “**DbContext**” constructor by passing the required connection string name.

### Implementing Repository class

1. Create a new class named “**VaccinationCentersRepository**” in the models folder with the following specification

Table 4 : VaccinationCentersRepository class fields

|  |  |  |
| --- | --- | --- |
| Field Name | Type | Modifier |
| context | VaccinationCentersContext | public |

Table 5 : VaccinationCentersRepository class constructors

|  |  |  |
| --- | --- | --- |
| Constructor Type | Input Parameters | Modifier |
| Default | - | public |

Table 6 : VaccinationCentersRepository class methods

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Input Parameters | Return Type | Modifier |
| AddVaccinationCenter () | VaccinationCenters model | bool | public |

1. Use the constructor to create a new instance of the VaccinationCentersContext class.
2. Implement the AddVaccinationCenter() to save vaccination center details into the database.
3. Adding duplicate vaccination center is not supported, vaccination center detail can be verified using vaccination center name, city and mobile no together, before saving it in the database.
4. The start date should be greater than or equal to current date; otherwise the data will not be allowed to be saved in the database.
5. Based on whether vaccination center details are saved or not method should return true/false.

### Implementing Controllers and Views

1. Add a new controller named “VaccinationCenterController” in the controllers folder with the following specification.

Table 7 : VaccinationCenterController Fields

|  |  |  |
| --- | --- | --- |
| Field Name | Field Type | Modifier |
| repository | VaccinationCentersRepository | private |

Table 8 : VaccinationCenterController Constructors

|  |  |  |
| --- | --- | --- |
| Constructor Type | Input Parameters | Modifier |
| Default | - | public |

Table 9 : VaccinationCenterController Add Actions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ActionName | Input Parameters | Http Request Type | Modifier | Return Type |
| Add() | - | Get | public | ActionResult |
| Add() | VaccinationCenters model | Post | public | ActionResult |

1. Implement the constructor to instantiate the repository instance.
2. Implement the Add() action for the http get request to perform the following operation.
   1. Create a string list named VaccineCategories with values "Covaxin", "Covishield", "Spuntik V" .
   2. Store the list data in ViewBag named “VaccineCategories”.
   3. Return a strongly typed view named as Add.cshtml which is binded with “VacciantionCenters” entity class. Scaffold the view using Create template.
3. Go to Add.cshtml view and make the following changes.
4. Display the dropdown list for “VaccineCategory” property of VacciantionCenters model and it’s source should be taken from the ViewBag’s property VaccineCategories created earlier.
5. Set an ID attribute on the submit button with the value “btnSubmit”
6. Change the sumbit button text to “Save”
7. Go to layout view and add a hyperlink to “Add” action with id=“ lnkAdd” attribute
8. Implement the Add() action for http post request to carryout the following operations
9. Validate the model and return the view if model is invalid
10. If model is valid save the model in the database using the repository object’s AddVaccinationCenter () method.
11. When the vaccination center details are saved successfully in the database, the put an acknowledgement in ViewBag by creating a “Message” property with value as “Vaccination center details added successfully”
12. In case of failure put the message in ViewBag’s Message property as “Failed to add vaccination center details. Try again later”
13. Also ensure that ViewBag is also passing in the VaccineCategories types to the view for displaying the drop down list.
14. Modify the Add.cshtml view to display the value of ViewBag’s Message property inside an <h2> element. Assign the ID=”Message” attribute to <h2>.

# Search vaccination centers

## Requirement flow

**Steps Explanation**

1. User navigates to search page by clicking the “Search Vaccination Centers” link in the menu and search page will be displayed to the user as follows.

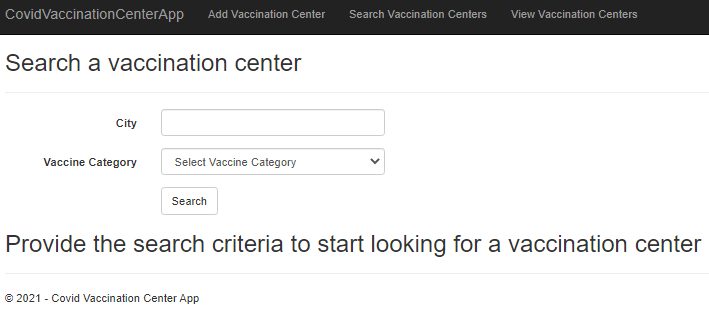


Figure 8: Search vaccination center page

1. If user clicks without providing the search value, then the appropriate validation error messages will display as follows

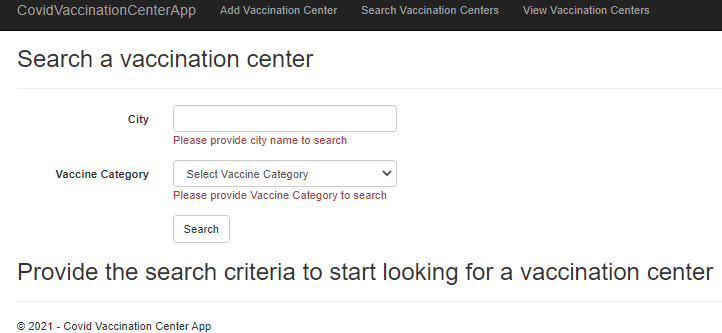


Figure 9: Search vaccination center page validation error

1. User will fill up the search form and click the search button which will display the vaccination center details as follows

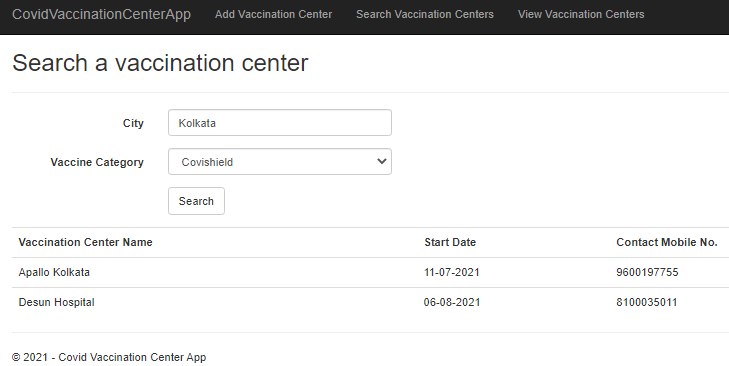


Figure 10 : Search vaccination center page lists vaccination centers details

1. If there are not vaccination center present matching the search criteria then the application will display the appropriate message to the user as follows.

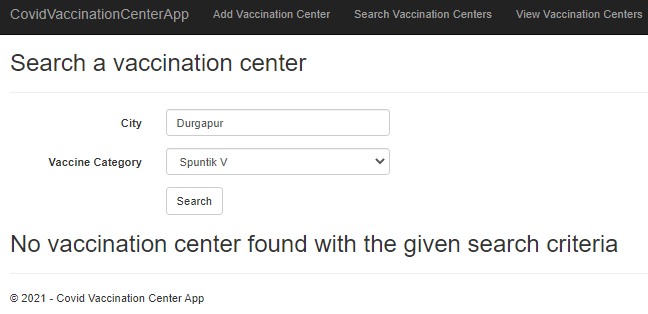


Figure 11 : Search vaccination center page displays no vaccination center found message

## Technical guidelines

### Implementing ViewModel class

1. Create a new class in the models folder with the name as “**SearchVaccinationCentersViewModel**” which will be used an model for binding with search views. Use the following specification for the class.

Table 10 : SearchVaccinationCentersViewModel class Properties

|  |  |  |
| --- | --- | --- |
| Property Name | Type | Modifier |
| VaccineCategory | string | public |
| City | string | public |
| Centers | List<VaccinationCenters> | public |

1. Modify the SearchVaccinationCentersViewModel class to implement the following validation rules.

Table 11 : SearchVaccinationCentersViewModel validation rules

|  |  |  |
| --- | --- | --- |
| Property | Validation | ErrorMessage |
| VaccineCategory | Must not be left blank | Please provide Vaccine Category to search |
| City | Must not be left blank | Please provide city name to search |
| Should allow maximum 25 characters only | City name must not exceed 25 characters |

### Implementing Repository class

1. Go to VaccinationCentersRepository class in the Models folder and modify the class with the following method specifications.

Table 12 : VaccinationCentersRepository Search specificattion

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Input Parameters | Return Type | Modifier |
| Search() | string city, string category | List<VaccinationCenters> | public |

1. Implement the Search() method to search for vaccination centers in a given city for the specified vaccine category passed to the method and return all vaccination centers which matches the criteria.
2. Seach operation should ignore case of city name and vaccine category.

### Implementing Controllers and Views

1. Go to VaccinationCenterController and add the following specified actions to implement the search feature in the application.

Table 13 : VaccinationCenterController Search Actions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ActionName | Input Parameters | Http Request Type | Modifier | Return Type |
| Search() | - | Get | public | ActionResult |
| Search() | SearchVaccinationCentersViewModel model | Post | public | ActionResult |

1. Implement Search() action which executes for http get request to perform the following operations
   1. Return a Search.cshtml view which is scaffolded using the create template and SearchVaccinationCentersViewModel class.
   2. Method must return an empty model to the view
   3. Method should also create a string list with name VaccineCategories having values :- "Covaxin", "Covishield", "Spuntik V"and store it in the view bag’s “VaccineCategories” property.
2. Implement Search() action method for an http post request to perform the following activities.
3. Validate the model and return the view in-case of validation failure
4. If validation passes search the vaccination centers using the repository object’s “Search()” and return the view with list of vaccination centers found.
5. Generate the dropdown items for VaccineCategories as mentioned earlier for get action.
6. Search.cshtml razor view must have the following specification
7. Change the default submit button text to “Search”
8. Provide an ID=”btnSubmit” attribute to the default submit button in the form.
9. Specify a dropdown list inplace of a textbox for “VaccineCategory” property and provide the data source from the ViewBag.
10. Initially when the view is displayed to the user on http get action display the message - “Provide the search criteria to start looking for a vaccination center “ in<h2> tag with ID=”Message” attribute.
11. On subsequent execution of view for post method when there are no vaccination centers found for a given city and vaccine category type display the message – “No vaccination center found with the given search criteria” inside <h2> element with ID=”Message” attribute.
12. If there any vaccination center found then display the Vaccination Center Name, Start Date and Contact Mobile Noinside a html <table> element with ID attribute set to “tblVaccinationCenters”.
13. Go to layout view and add a hyper link in the menu for the Search() action with the ID attribute set to “lnkSearch”.

# List vaccination centers

## Requirement flow

**Steps Explanation**

1. User navigates to search page by clicking the “View Vaccination Centers” link in the menu and view page will be displayed to the user as follows.

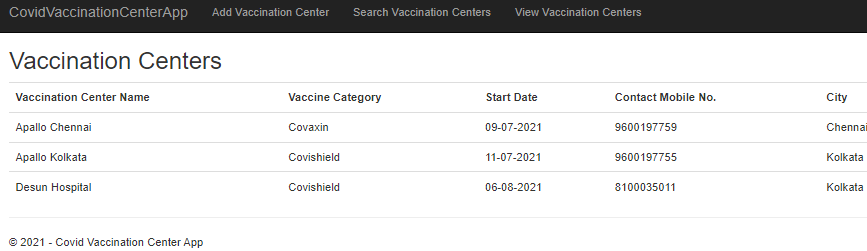


Figure 12: Index page

1. If there are not vaccination center present then the application will display the appropriate message to the user as follows.

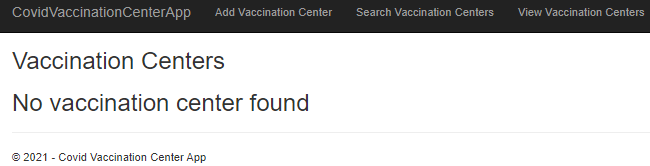


Figure 13 :Index displays no vaccination center found message

## Technical guidelines

### Implementing Repository class

1. Go to VaccinationCentersRepository class in the Models folder and modify the class with the following method specifications.

Table 12 : VaccinationCentersRepository Search specificattion

|  |  |  |  |
| --- | --- | --- | --- |
| Method Name | Input Parameters | Return Type | Modifier |
| ListVaccinationCenters () | **-** | List<VaccinationCenters> | public |

1. Implement the method and return all vaccination centers from database.

### Implementing Controllers and Views

1. Go to VaccinationCenterController and add the following specified actions to implement the index feature in the application.

Table 13 : VaccinationCenterController Index Actions

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ActionName | Input Parameters | Http Request Type | Modifier | Return Type |
| Index() | - | Get | public | ActionResult |

1. Implement Index() action which executes for http get request to perform the following operations
   1. Return an Index.cshtml view which is scaffolded using the list template and Model VaccinationCenters class
2. Index.cshtml razor view must have the following specification
3. When there are no vaccination center found, display the message – “No vaccination center found” inside <h2> element with ID=”Message” attribute.
4. If there any vaccination center found then display the Vaccination Center Name, Vaccine Category, Start Date,Contact Mobile No and City inside a html <table> element with ID attribute set to “tblVaccinationCenters”.
5. Go to layout view and add a hyper link in the menu for the Index() action with the ID attribute set to “lnkIndex”.

# Evaluation Areas

|  |  |
| --- | --- |
| 01 | Launch of the application from Add vaccination center page |
| 02 | Navigation to search vaccination center page |
| 03 | Validation of input on add vaccination center and search page |
| 04 | Saving of vaccination center details into the database |
| 05 | Searching and displaying the vaccination center details |
| 06 | Display all vaccination centers from database |