

GUIDE

COVID19 Updates Dashboard & API and User Manual

Testing (Version 1.1)

Author:

Ravi Koppula

Index

Contents

1.0 Section A – API Call from GUI.....	3
<i>1.1 Introduction</i>	<i>3</i>
Step 1: Launch API solution to view the dashboard.	3
Step 2: View Updates and filter updates	3
2.0 Section B – API Call from the Swager UI or Postman	5
<i>2.1 Introduction</i>	<i>5</i>
Test Case 1: View data based on country and state.....	6
Test Case 2: View data based on country	7
Test case 3: View data based on state.....	7
Test case 4: View data based on date	8
3.0 Section C – COVID19Updates API – Architecture.....	9
<i>Step1: Solution pattern</i>	<i>10</i>
4.0 Section D: NUnitTest Project:	11
<i>4.1 NUnit Test folder structure:.....</i>	<i>11</i>
<i>4.2 Testing method calls</i>	<i>11</i>

1.0 Section A – API Call from GUI

1.1 Introduction

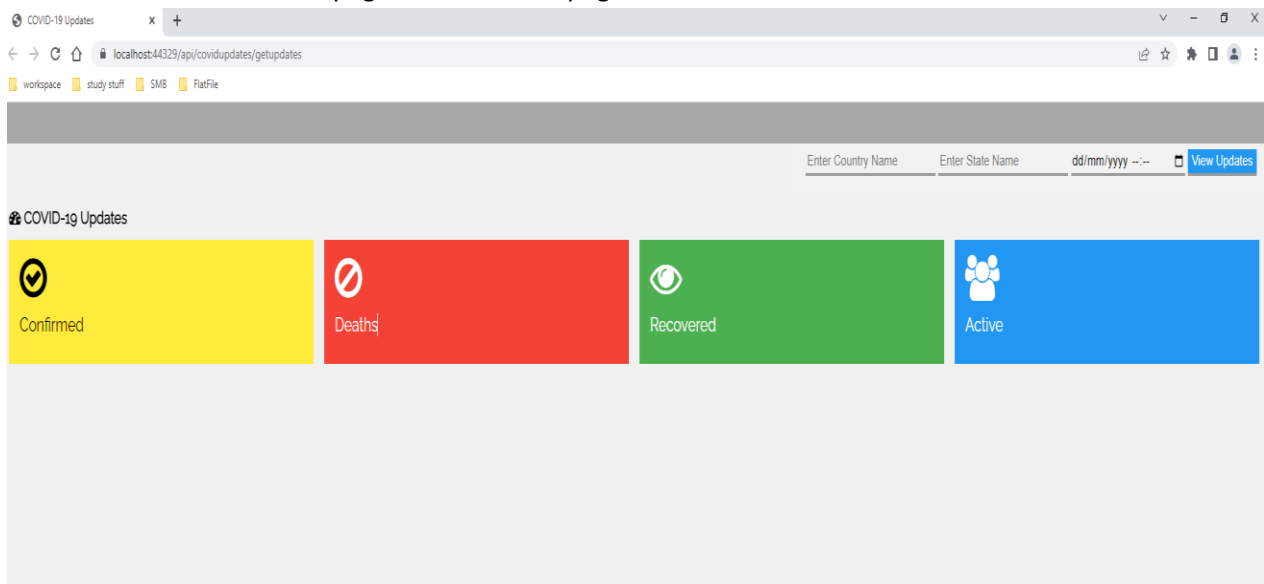
The purpose of this document is to provide necessary guidance on Covid-19 updates via API call and GUI intro.

Covid-19 updates dashboard contains 3 input params

1. Country: Textbox
2. State: Textbox and;
3. Last updated date: Date picker.

Step 1: Launch API solution to view the dashboard.

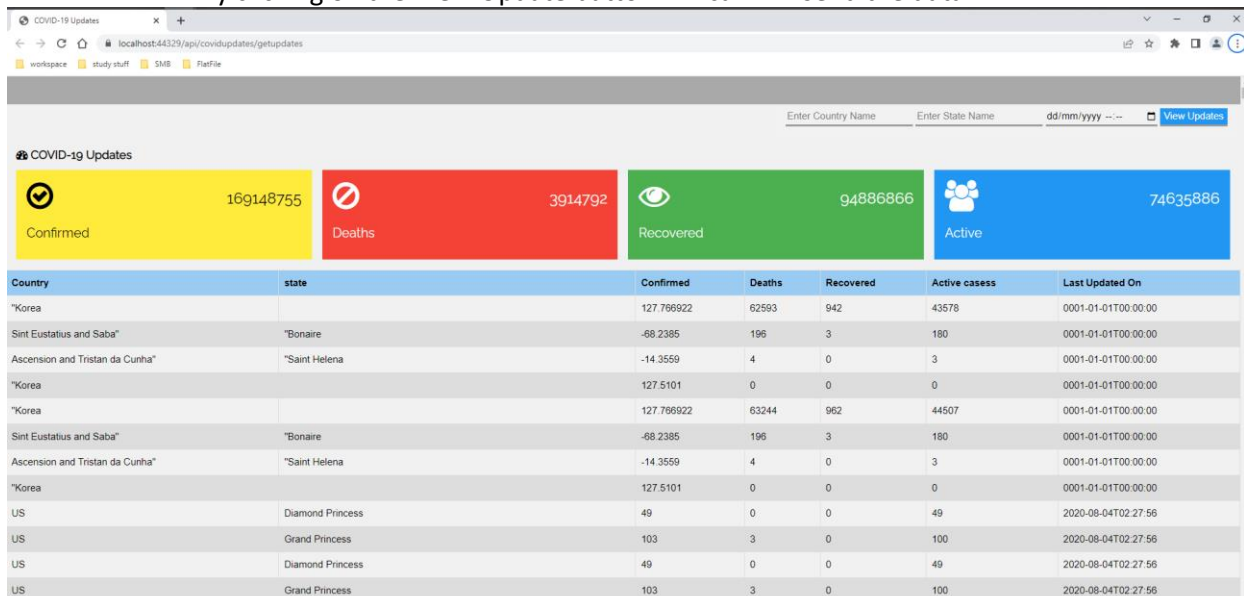
The moment we launch the COVID19Updates API, system will redirect to the dashboard page as the default page.



Step 2: View Updates and filter updates

As shown in the below picture, user must either enter the country, state or select date from the date picture.

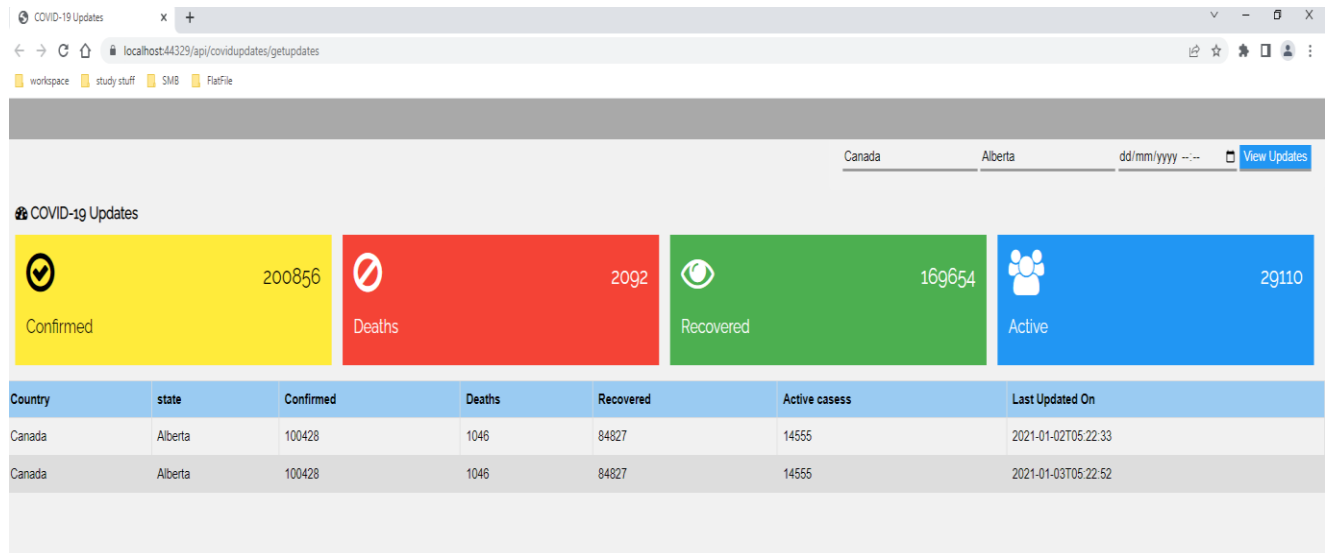
By clicking on the View Update button API call will send the data.



We can also view the dashboard with and without filter conditions
For an instance:

- Users are able view the number of death cases, confirmed cases, recovered and active cases based on the country or state or by date.
- The screen below shows users selected Canada is the country and the chosen state is Alberta to view the infected cases.

Below is the reference of the search:



- Yellow represents: Confirmed cases
- Red represents: Death cases
- Green represents: Recovered and
- Blue represents: Active cases

All these dashboards' values will change dynamically based on the user call.

2.0 Section B – API Call from the Swager UI or Postman

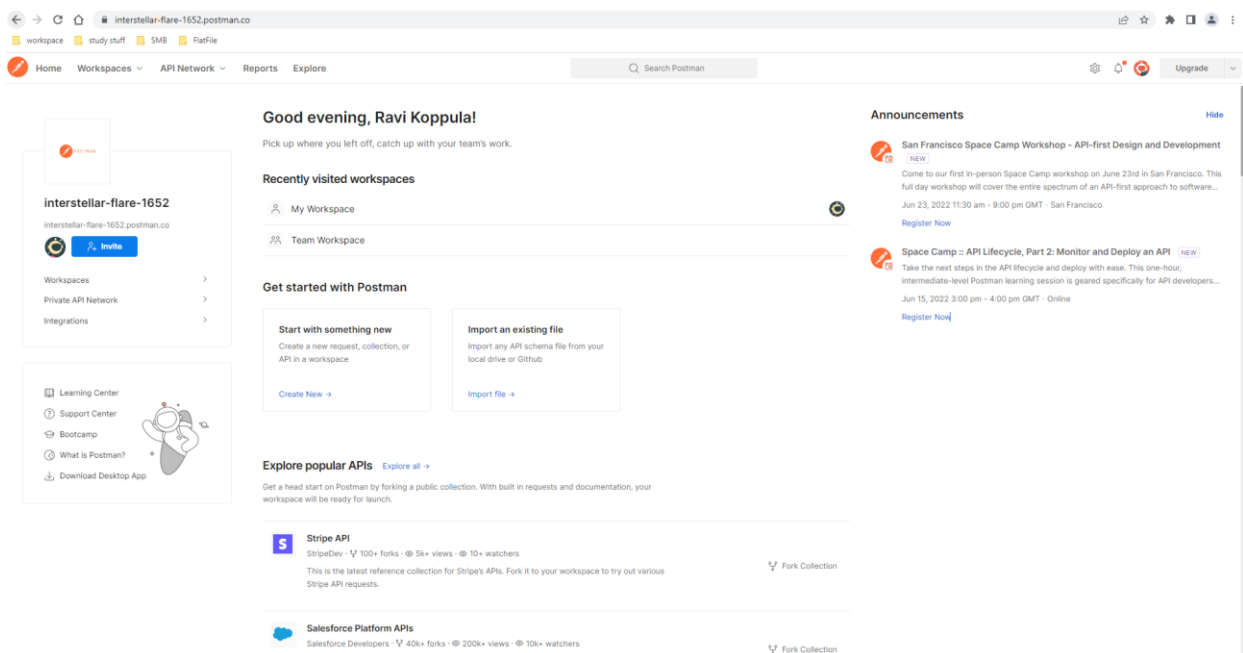
2.1 Introduction

The purpose of this section B is to demonstrate how testing can be done using third party tools such as POSTMAN and SWAGGER UI.

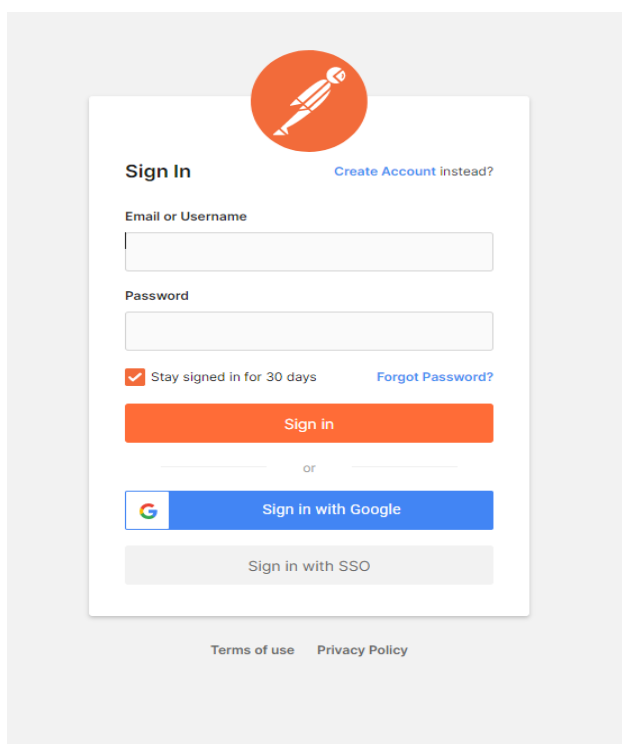
Step 1: Login to Postman Web

1. Please click the link below to open Postman Web. Alternatively, Postman can be downloaded directly on Windows.

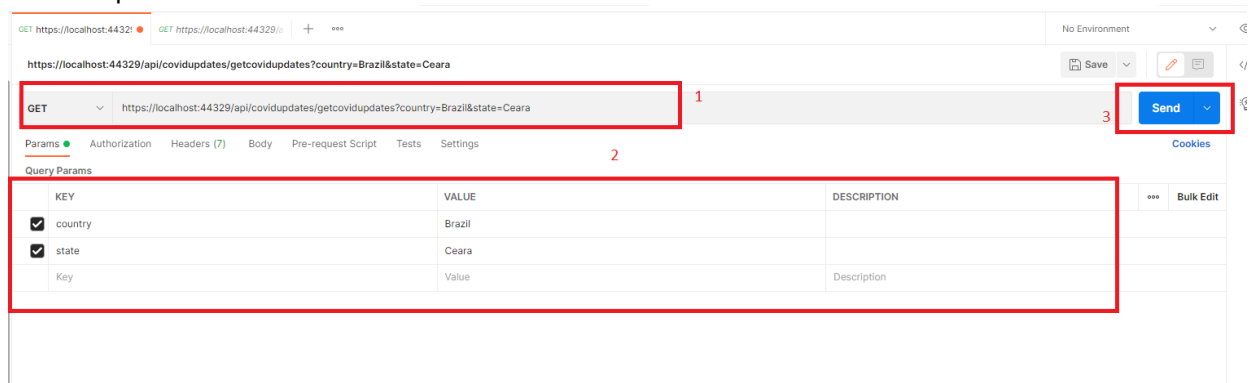
Post man web - <https://interstellar-flare-1652.postman.co/>



2. For first-time users, please sign-in to record your activities under history session.



- Please note that keep the API solution file in run mode or host it in local IIS
Screncap for reference:



Request Type is GET

API URL: <https://localhost:44329/api/covidupdates/getcovidupdates>

API Parms:

KEY	VALUE
country	Brazil
state	Ceara
Key	Value

- Click the Send button to see the response.

Alternatively, this link can be used for quick access:

Test Case 1: View data based on country and state

<https://localhost:44329/api/covidupdates/getcovidupdates?country=Brazil&state=Ceara>

API Response:

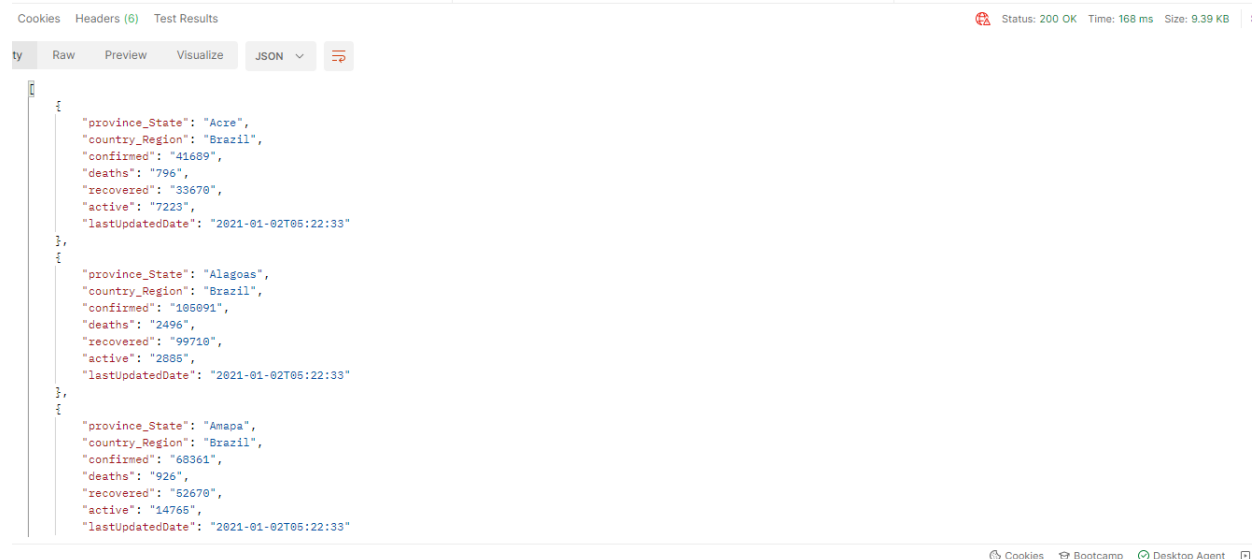
```
[
  {
    "province_State": "Ceara",
    "country_Region": "Brazil",
    "confirmed": "336504",
    "deaths": "10004",
    "recovered": "269846",
    "active": "56153",
    "lastUpdatedDate": "2021-01-02T05:22:33"
  },
  {
    "province_State": "Ceara",
    "country_Region": "Brazil",
    "confirmed": "336574",
    "deaths": "10015",
    "recovered": "270203",
    "active": "56356",
    "lastUpdatedDate": "2021-01-03T05:22:52"
  }
]
```

Test Case 2: View data based on country

<https://localhost:44329/api/covidupdates/getcovidupdates?country=Brazil>

API Response:

API response is lengthy, therefore screencap are attached to show the reference of code.



```
[{"province_State": "Acre", "country_Region": "Brazil", "confirmed": "41689", "deaths": "796", "recovered": "33670", "active": "7223", "lastUpdatedDate": "2021-01-02T05:22:33"}, {"province_State": "Alagoas", "country_Region": "Brazil", "confirmed": "105091", "deaths": "2496", "recovered": "99710", "active": "2885", "lastUpdatedDate": "2021-01-02T05:22:33"}, {"province_State": "Amapa", "country_Region": "Brazil", "confirmed": "68361", "deaths": "926", "recovered": "52670", "active": "14765", "lastUpdatedDate": "2021-01-02T05:22:33"}]
```

Test case 3: View data based on state

<https://localhost:44329/api/covidupdates/getcovidupdates?state=Ceara>

API Response:

```
[{"province_State": "Ceara", "country_Region": "Brazil", "confirmed": "336504", "deaths": "10004", "recovered": "269846", "active": "56153", "lastUpdatedDate": "2021-01-02T05:22:33"}, {"province_State": "Ceara", "country_Region": "Brazil", "confirmed": "336574", "deaths": "10015", "recovered": "270203", "active": "56356", "lastUpdatedDate": "2021-01-03T05:22:52"}]
```

Test case 4: View data based on date

API URL: <https://localhost:44329/api/covidupdates/getcovidupdates?lastUpdatedDate=2021-01-02>

API Response:

```
Raw Preview Visualize JSON ⌵ ⌵
```

```
{
  "province_State": "",
  "country_Region": "Afghanistan",
  "confirmed": "52513",
  "deaths": "2201",
  "recovered": "41727",
  "active": "8585",
  "lastUpdatedDate": "2021-01-02T05:22:33"
},
{
  "province_State": "",
  "country_Region": "Albania",
  "confirmed": "58316",
  "deaths": "1181",
  "recovered": "33634",
  "active": "23501",
  "lastUpdatedDate": "2021-01-02T05:22:33"
},
{
  "province_State": "",
  "country_Region": "Algeria",
  "confirmed": "99897",
  "deaths": "2762",
  "recovered": "67395",
  "active": "29740",
  "lastUpdatedDate": "2021-01-02T05:22:33"
}
```

Test 4: View data without any filter:

API Request: <https://localhost:44329/api/covidupdates/getcovidupdates>

API Response:

```
GET https://localhost:44329/api/covidupdates/getcovidupdates
```

Params Authorization Headers (7) Body Pre-request Script Tests Settings

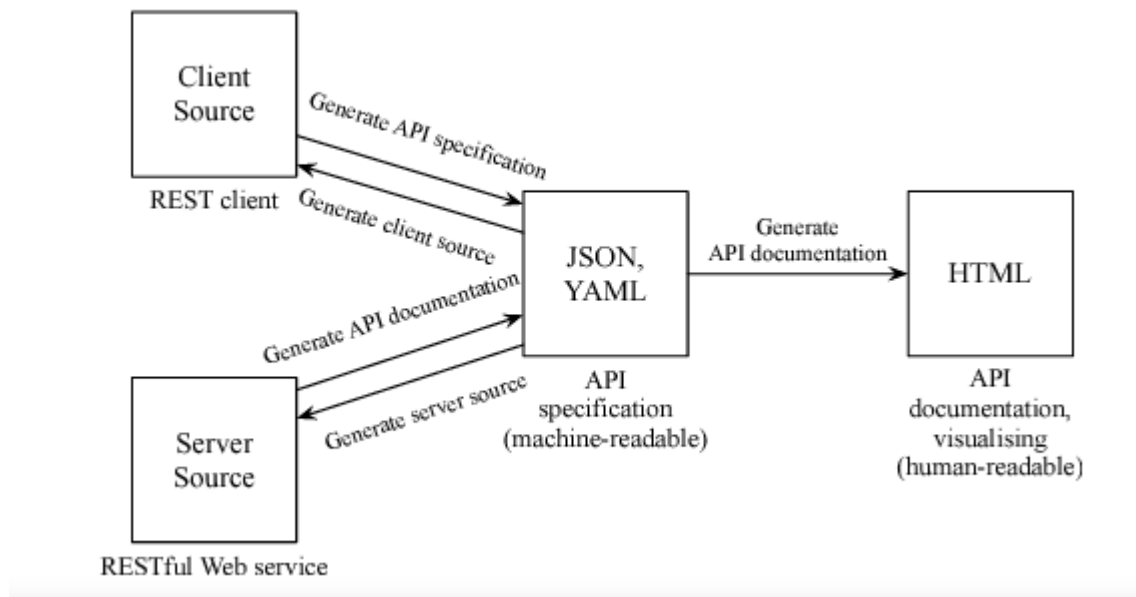
ody Cookies Headers (6) Test Results Status: 200 OK Time: 183 ms Size: 1.21 MB

```
Pretty Raw Preview Visualize JSON ⌵ ⌵
```

```
4   "country_Region": "\"Korea",
5   "confirmed": "127.766922",
6   "deaths": "62593",
7   "recovered": "942",
8   "active": "43578",
9   "lastUpdatedDate": "0001-01-01T00:00:00"
10 },
11 {
12   "province_State": "\"Bonaire",
13   "country_Region": " Sint Eustatius and Saba\"",
14   "confirmed": "-68.2385",
15   "deaths": "196",
16   "recovered": "3",
17   "active": "180",
18   "lastUpdatedDate": "0001-01-01T00:00:00"
19 },
20 {
21   "province_State": "\"Saint Helena",
22   "country_Region": " Ascension and Tristan da Cunha\"",
23   "confirmed": "-14.3559",
24   "deaths": "4",
25   "recovered": "0",
26   "active": "3",
27   "lastUpdatedDate": "0001-01-01T00:00:00"
28 },
29 {
30   "province_State": "",
31   "country_Region": "\"Korea",
32   "confirmed": "127.5181",
33   "deaths": "0",
34   "recovered": "0",
35   "active": "0",
36   "lastUpdatedDate": "0001-01-01T00:00:00"
}
```


3.0 Section C – COVID19Updates API – Architecture

Model View Controller pattern (MVC) For each request is used to design the Covid-19 API.



- The HTTP request will hit to the controller and controller will hit to the services from the services.
- Business Logic (BL) is implemented from service level which are loosely coupled with the interfaces.
- The controller are the mediators for the request and response which takes the HTTP response and service the response.

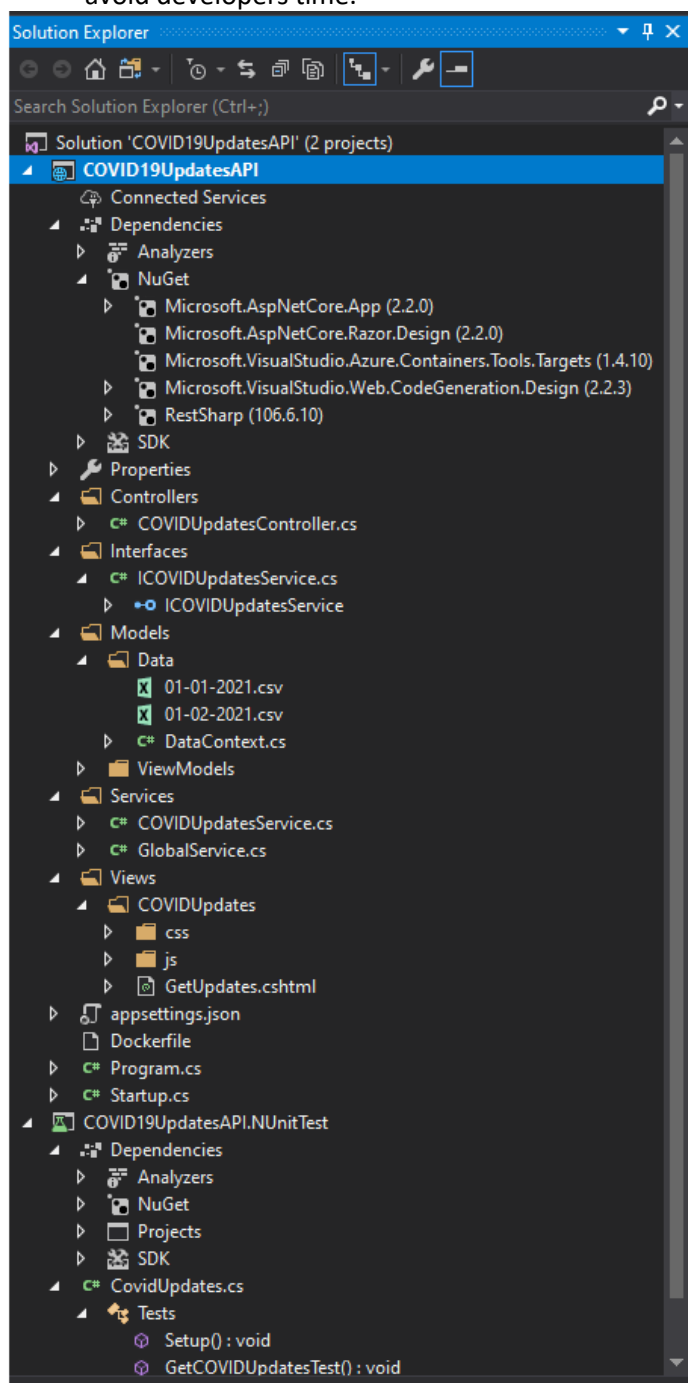
A simple UI is created for COVID-19 updates dashboard along with the filters.

*Note that the services will read the data from flat files which is stored local folder path.

In the future, data will be accessed from the database or directly from Github based on authentication or authorization.

Step1: Solution pattern

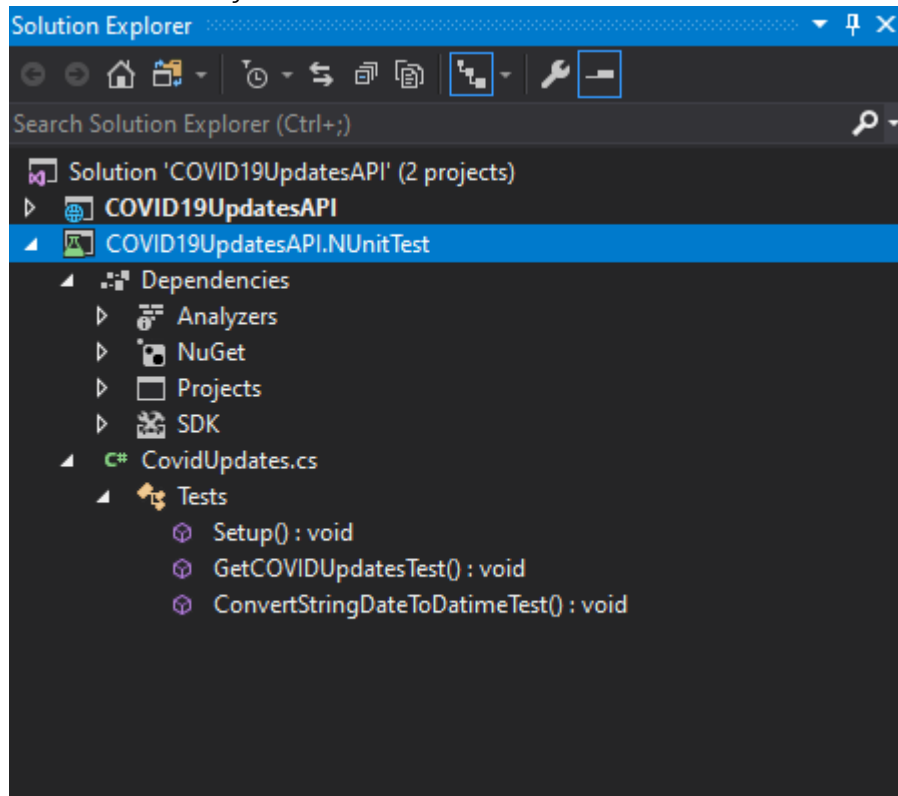
1. Inside the COVID19UpdatesAPI Solution controllers can be found under the controller folder (which is used for request & user response)
2. Interfaces folder contains all the interfaces which is implemented for the future.
3. Under the Models folder - View model folder and Data folder are stored
 - a. View Models is used to bind the output parameter.
 - b. Data Folder will be used to store the files (Ref: Git repo).
4. Services folder will store the logic and process HTTP response (User response).
5. View folder to design HTML pages to interact with the user.
6. Appsettings.json is used for data connection or any security tokens in the future Git Access.
7. In Startup.cs file contains all the dependency injection (DI) which is used in the project along with the default project path.
8. Dockerfile is created for the future to deploy across platforms to reduce TechOps works and to avoid developers time.



4.0 Section D: NUnitTest Project:

Single file is used here under COVIDUpdates.cs which was used to individual functionality with varies scenarios.

4.1 NUnit Test folder structure:



Please note that the testing was conducted, and the result passed for all.

4.2 Testing method calls

```
using COVID19UpdatesAPI.Services;
using NUnit.Framework;

namespace CovidUpdates
{
    0 references
    public class Tests
    {
        [Setup]
        0 references | 0 exceptions
        public void Setup()
        {
        }

        [Test]
        0 references | 0 exceptions
        public void GetCOVIDUpdatesTest()
        {
            string fpath = "C:\\Users\\qxz1xpw\\source\\repos\\COVID19UpdatesAPI\\COVID19UpdatesAPI\\Models\\Data\\01-01-2021.csv";
            var status = COVIDUpdatesService.ConvertCsvFileToJsonObject(fpath);

            Assert.IsNotNull(status);
        }

        [Test]
        0 references | 0 exceptions
        public void ConvertStringDateToDatetimeTest()
        {
            string lastUpdatedDate = "2021-03-01 00:00:00";

            var date = COVIDUpdatesService.ConvertStringDateToDatetime(lastUpdatedDate);

            Assert.Pass();
        }
    }
}
```