# COVID Assessment Project Documentation

## 1. Project Setup

The project starts by creating a Django project named 'covid\_assessment' using the command:  
```bash  
django-admin startproject covid\_assessment  
```  
This sets up a standard Django project structure, including key files like `settings.py`, `urls.py`, and `manage.py`.

## 2. Create the Charts App

Next, a new app named `charts` is created using:  
```bash  
python manage.py startapp charts  
```  
This creates a new app with necessary files such as `views.py`, `models.py`, and `admin.py`. The app is added to the project’s `INSTALLED\_APPS` in `settings.py`:  
```python  
INSTALLED\_APPS = [  
 ...  
 'charts',  
]  
```

## 3. Create a JSON Data File in Static Folder

A JSON file named `covid.json` is created inside the `static/charts/` directory to store the data. The data is structured as follows:  
```json  
[  
 {

"Date": "2020/06/08",

"Total Confirmed Cases": "50 879",

"Total Deaths": "1 080",

"Total Recovered": "26 099",

"Active Cases": "23 700",

"Daily Confirmed Cases": "2 594",

"Daily deaths": 82

},  
]  
```

## 4. Create a View to Load the Data

In `charts/views.py`, a view is created to read the JSON data from the static folder and pass it to the template:  
```python  
from django.shortcuts import render  
from django.templatetags.static import static  
import json  
  
def chart\_view(request):

with open('charts/static/charts/covid.json', 'r') as file:

data = json.load(file)

return render(request, 'charts/chart.html', {'covid\_data': json.dumps(data)})  
```

**static directory is created to store CSS files:**

/\* charts/static/charts/style.css \*/

.chart-container {

width: 80%;

margin: auto;

}

## 5. Define URLs

A `urls.py` file is created inside the `charts/` directory to define the route for the chart:  
```python  
from django.urls import path  
from .views import chart\_view  
  
urlpatterns = [  
 path('chart/', chart\_view, name='chart')  
]  
```  
The main `urls.py` file is updated to include the `charts` app:  
```python  
from django.contrib import admin  
from django.urls import path, include  
  
urlpatterns = [  
 path('admin/', admin.site.urls),  
 path('', include('charts.urls')),  
]  
```

## 6. Create a Static Directory

A `static` directory is created to store CSS and JSON files:  
```bash  
mkdir -p charts/static/charts  
```  
A CSS file named `style.css` is created:  
```css  
.chart-container {  
 width: 80%;  
 margin: auto;  
}  
```

## 7. Create the Template to Render the Chart

The `chart.html` file is created inside `charts/templates/charts/` and updated to use Highcharts to view the graphs

How it works:

* Highcharts is imported using a CDN link.
* The Highcharts.chart function creates a line chart.
* The categories and data fields are populated with data from the view using Django’s {{ ... }} syntax.
* Highcharts automatically processes and displays the chart.

## 8. Why I used Highcharts Chart.js

* Highcharts offers more customization options and better performance with large datasets.
* Highcharts supports real-time updates, hover states, and interactive charts out of the box.
* Highcharts works better with large datasets and has better cross-browser compatibility.
* Highcharts supports more chart types (e.g., stock charts, heatmaps) than Chart.js.
* Built-in accessibility features make Highcharts easier to use for diverse audiences.

## 9. Why No Virtual Environment Was Used

A virtual environment was not used because:  
The project only relies on Django and Highcharts (loaded via a CDN).  
 No additional dependencies are required, simplifying the setup process.  
 Highcharts is a client-side library, so no backend installation is needed.

## 10. How to run it

1. cd .\covid\_assessment\
2. python manage.py runserver