Summer Project On

COVID - 19 TRACKER

By

Sakshi Naik (2021510036)
Hema Manoj (2021510021)

Under the guidance of
Internal Supervisor

Prof. Harshil Kanakia

Department of Master Of Computer Application
Sardar Patel Institute of Technology
Autonomous Institute Affiliated to Mumbai University
2021-22

CERTIFICATE OF APPROVAL

This is to certify that the following students

Sakshi Naik (2021510036)
Hema Manoj (2021510021)

Have satisfactorily carried out work on the project entitled

"COVID 19 TRACKER"

Towards the fulfilment of project, as laid down
by
Sardar Patel Institute of Technology
during year
2021-22.

Project Guide:
Prof. Harshil Kanakia

PROJECT APPROVAL CERTIFICATE

This is to certify that the following students

Sakshi Naik (2021510036)
Hema Manoj(2021510021)

Have successfully completed the Project report on

“COVID-19 TRACKER”,

which is found to be satisfactory and is approved

at

SARDAR PATEL INSTITUTE OF TECHNOLOGY,
ANDHERI (W), MUMBAI

INTERNAL EXAMINER EXTERNAL EXAMINER

HEAD OF DEPARTMENT PRINCIPAL

Contents

Sakshi Naik (2021510036)
Hema Manoj (2021510021)

Abstract

Covid-19 has put the world to a standstill. Doctors, healthcare workers and-personnel of many other essential services are fighting at the frontline to tackle this global pandemic. Although we are not fighting the battle at the frontline, as students of statistics this is our humble attempt at partaking in the struggle. We have created a website to track COVID-19 where we have displayed the data from the world as a whole and also country wise. The data is categorised into three components: confirmed cases, deaths and recovered. The values are given for both daily and cumulative type. We have tried our best to keep the display simple yet visually appealing. We have used line charts and pie charts and also an exquisite race chart for display. All the above-mentioned charts are interactive and are customized to give the user a clear idea of the intended meaning of the values as all the categories are separated by different colours, this not only made the graphs more appealing to the eyes but also helped in distinguishing different aspects. There are two sections dedicated to graphs, one for India and the other for the entire world, graphs for India are under Graphs under India Tracker and those for the world are under Graphs under Home

Objectives

The React based "COVID-19 TRACKER" react-app is used -

- To provide the total corona virus cases worldwide segregated into total recovered and total deaths,
- To provide a graph of total cases monthly.
- To provide a world map signifying regions of active cases.The map pulls in dynamic information and shows the concentration of cases
- To provide live cases country wise, sorted in descending order.

1 Introduction

1.1 Problem Definition

The COVID-19 TRACKER projects provides an interface to view relavent covid-19 numbers across the globe. The data is distributed according to the respective countries in descending order and also visually depicts the concentration of cases with the help of a map.

1.2 Objectives and Scope

1.2.1 Objectives

The React based "COVID-19 TRACKER" react-app is used -

- To provide the total corona virus cases worldwide segregated into total recovered and total deaths,
- To provide a graph of total cases monthly.
- To provide a world map signifying regions of active cases.The map pulls in dynamic information and shows the concentration of cases
- To provide live cases country wise, sorted in descending order.

1.2.2 Scope

You can select either number of new cases each day, number of new recovered cases or number of new deaths.

We have a map which you can drag around and click on a specific country and it will tell you the cases, the recovered and the deaths.

The circle will represents how much cases each region takes up, the bigger the circle the more cases.

1.3 Existing System

Currently no such kind of application exists for the COVID-19 Tracker worldwide specifically. There are app such as Arogya setu which is specific to india Arogya Setu App - Aarogya Setu is an Indian COVID-19 ”contact tracing, syndromic mapping and self-assessment” digital service, primarily a mobile app, developed by the National Informatics Centre under the Ministry of Electronics and Information Technology (MeitY).The stated purpose of this app is to spread awareness of COVID-19 and to connect essential COVID-19-related health services to the people of India.

1.4 Proposed System

We have created a website to track COVID-19 where we have displayed the data from the world as a whole and also country wise. The data is categorised into three components: confirmed cases, deaths and recovered. The values are given for both daily and cumulative type. We have tried our best to keep the display simple yet visually appealing.

We have used line charts and pie charts and also an exquisite race chart for display. All the above-mentioned charts are interactive and are customized to give the user a clear idea of the intended meaning of the values as all the categories are separated by different colours, this not only made the graphs more appealing to the eyes but also helped in distinguishing different aspects.

The data is fetched from the API named – disease.sh The API, disease.sh-OPEN DISEASE DATA is an external API service and what we do is we call that service and we pull in all the live stats.

The COVID-19 Tracker app is divided into sections and each section is described as follows:-

1) Covid-19 numbers total numbers tracking section. This section here has namely three sections as follows:-
1.Coronavirus Cases – This displays the daily coronavirus cases and below shows the total cases worldwide
2.Recovered – This part displays the amount of recovered people daily.
3.Deaths – This part displays the daily deaths that occur.
2) MAP – The map is basically a visual representation of the covid 19 numbers.

1.5 System Requirements

- Hardware Requirements on Server Side

Table 1.5.1: Hardware Requirements on Server Side

Processor	Dual Core Processor or Above
RAM	Minimum 4 GB RAM
Storage	Minimum 10 GB Hard Disk Space for smooth run

- Hardware Requirements on Client Side

Table 1.5.2: Hardware Requirements on Client Side

Processor	Dual Core Processor or Above
RAM	Minimum 2 GB RAM
Storage	Minimum 250 MB Storage Space

- Software Requirements on Server Side

Table 1.5.3: Software Requirements on Server Side

Operating System	OS Independent
Database	Firestore

- Software Requirements on Client Side

Table 1.5.3: Software Requirements on Client Side

Operating System	Android/IOS Smartphone
Server	Not Required

2 Software Requirement Specification (SRS) and Design

2.1 Purpose

The purpose of is to create a website to track COVID-19 where we have displayed the data from the world as a whole and also country wise. The data is categorised into three components: confirmed cases, deaths and recovered. The values are given for both daily and cumulative type. We have tried our best to keep the display simple yet visually appealing. We have used line charts and pie charts and also an exquisite race chart for display. All the above-mentioned charts are interactive and are customized to give the user a clear idea of the intended meaning of the values as all the categories are separated by different colours, this not only made the graphs more appealing to the eyes but also helped in distinguishing different aspects. There are two sections dedicated to graphs, one for India and the other for the entire world, graphs for India are under Graphs under India Tracker and those for the world are under Graphs under Home.

2.2 Definations, Acronyms, Abbreviations

ERD – Entity Relationship Diagram DB - Database IEEE-Institute of Electrical and Electronics Engineers

2.3 Document Overview

This document contains the functional and non-functional requirements of the system

2.4 References

IEEE standard -830 -1998, Pankaj Jalote Software Engineering.

2.5 Intended Audience

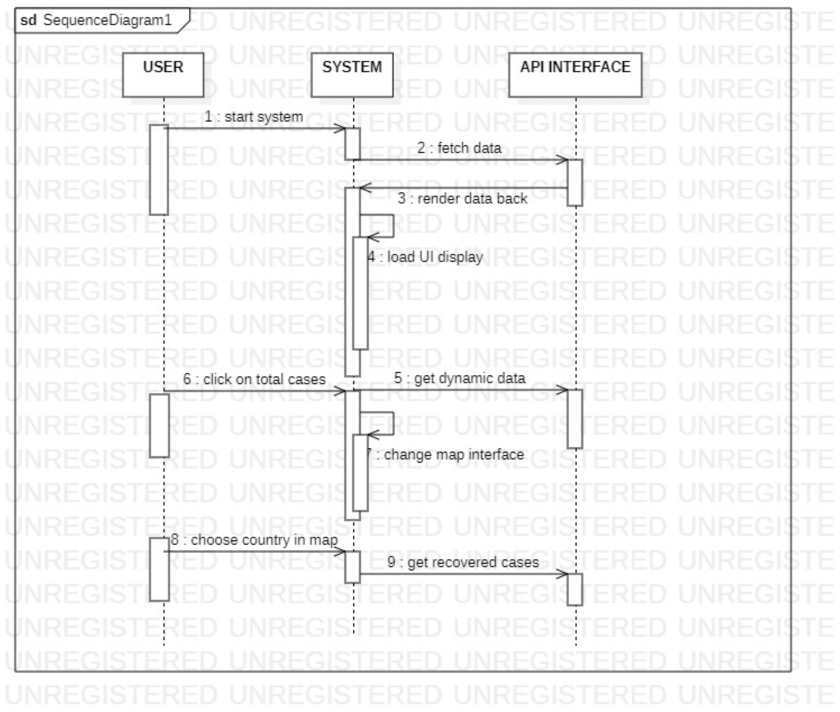
This document will be used for design purpose by the developer and design team. It will be the basis for validating the final delivered system.

2.6 Srs Team Members

The document is written by Hema Manoj (2021510021) and Sakshi Naik (2021510036).

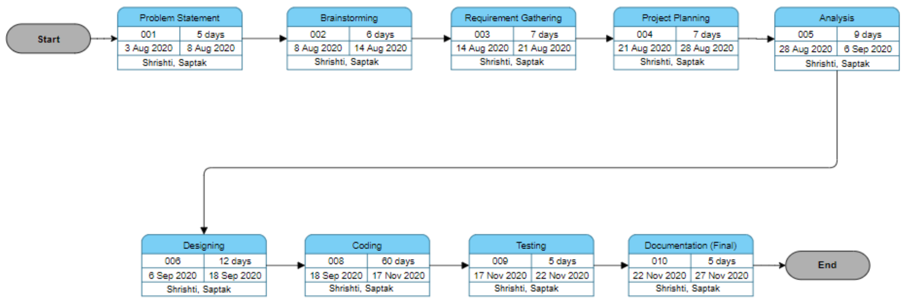
2.7 Modules

2.7.1 Sequence Diagram



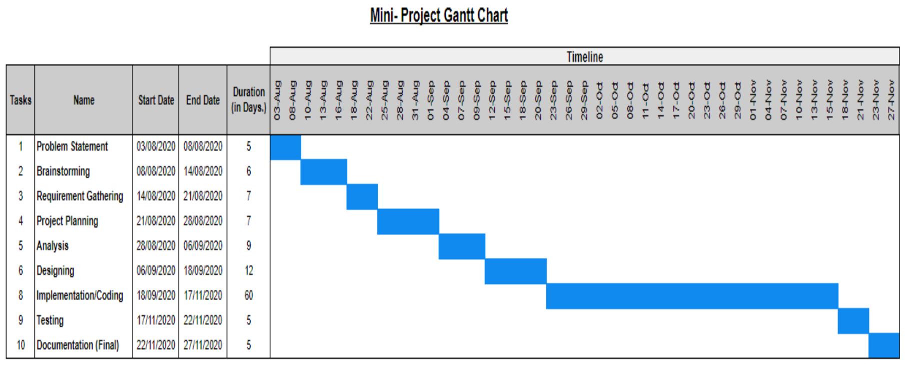
2.7.1: Sequence Diagram

2.7.2 PERT Chart



2.7.2: PERT Chart

2.7.3 Gantt Chart

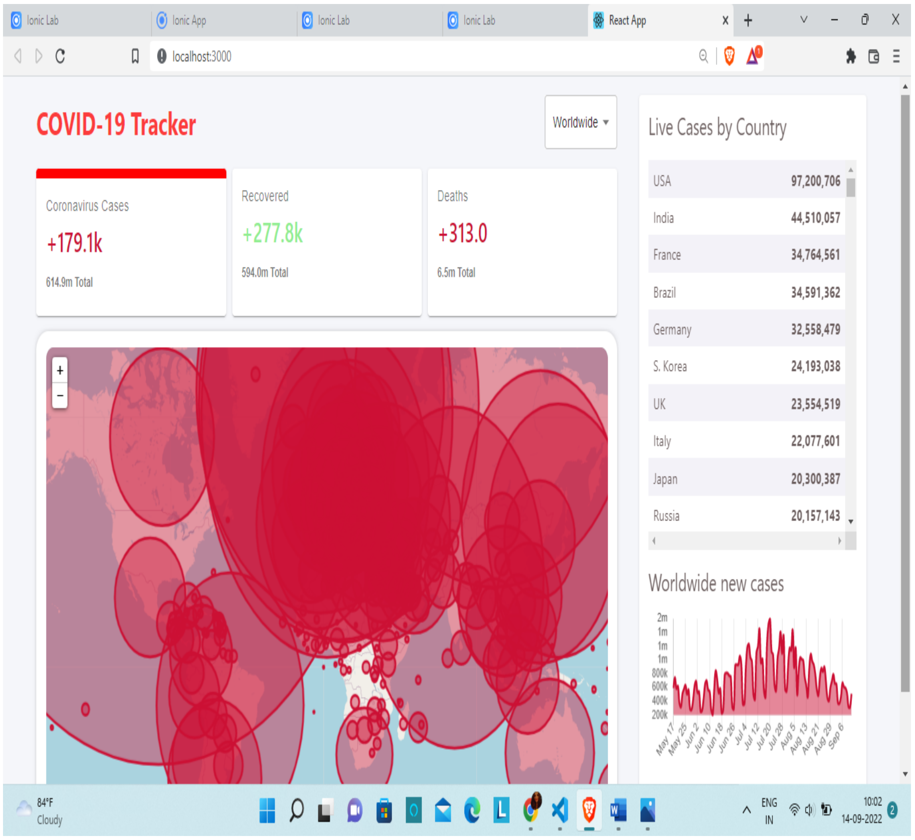

Mini- Project Gantt Chart

Tasks	Name	Start Date	End Date	Duration (in Days.)
1	Problem Statement	03/08/2020	08/08/2020	5
2	Brainstorming	08/08/2020	14/08/2020	6
3	Requirement Gathering	14/08/2020	21/08/2020	7
4	Project Planning	21/08/2020	28/08/2020	7
5	Analysis	28/08/2020	06/09/2020	9
6	Designing	06/09/2020	18/09/2020	12
8	Implementation/Coding	18/09/2020	17/11/2020	60
9	Testing	17/11/2020	22/11/2020	5
10	Documentation (Final)	22/11/2020	27/11/2020	5

2.7.3: Gantt Chart

3 Project Implementation and Testing

3.1 COVID -19 TRACKER



COVID-19 Tracker

Coronavirus Cases
+179.1k
614.9m Total

Recovered
+277.8k
594.0m Total

Deaths
+313.0
6.5m Total

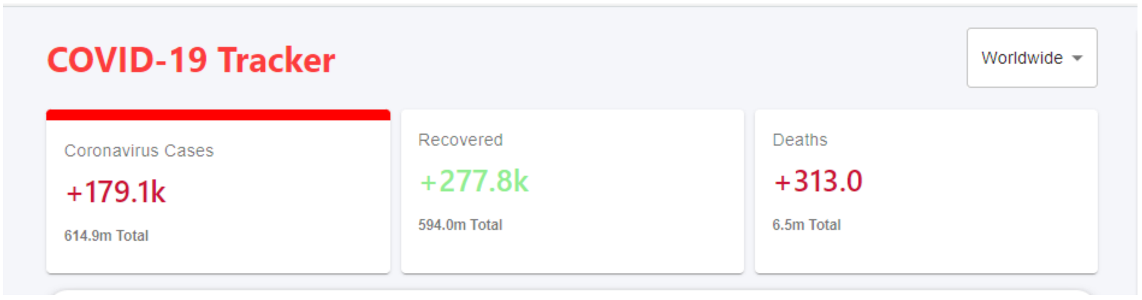
Live Cases by Country

USA	97,200,706
India	44,510,057
France	34,764,561
Brazil	34,591,362
Germany	32,558,479
S. Korea	24,193,038
UK	23,554,519
Italy	22,077,601
Japan	20,300,387
Russia	20,157,143

Worldwide new cases

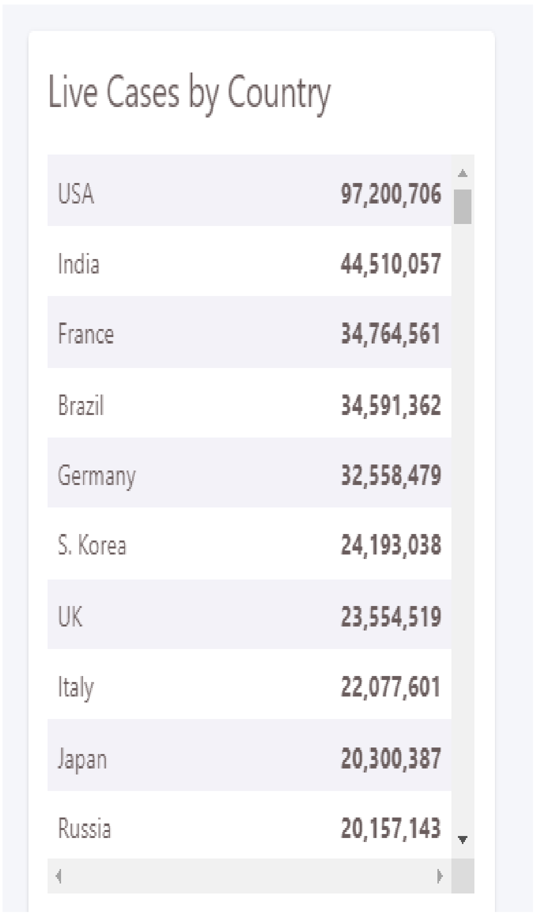
3.1.1: Main Page

3.2 COVID 19 TABS



3.2.1: Home View

3.3 LIVE CASES BY COUNTRY

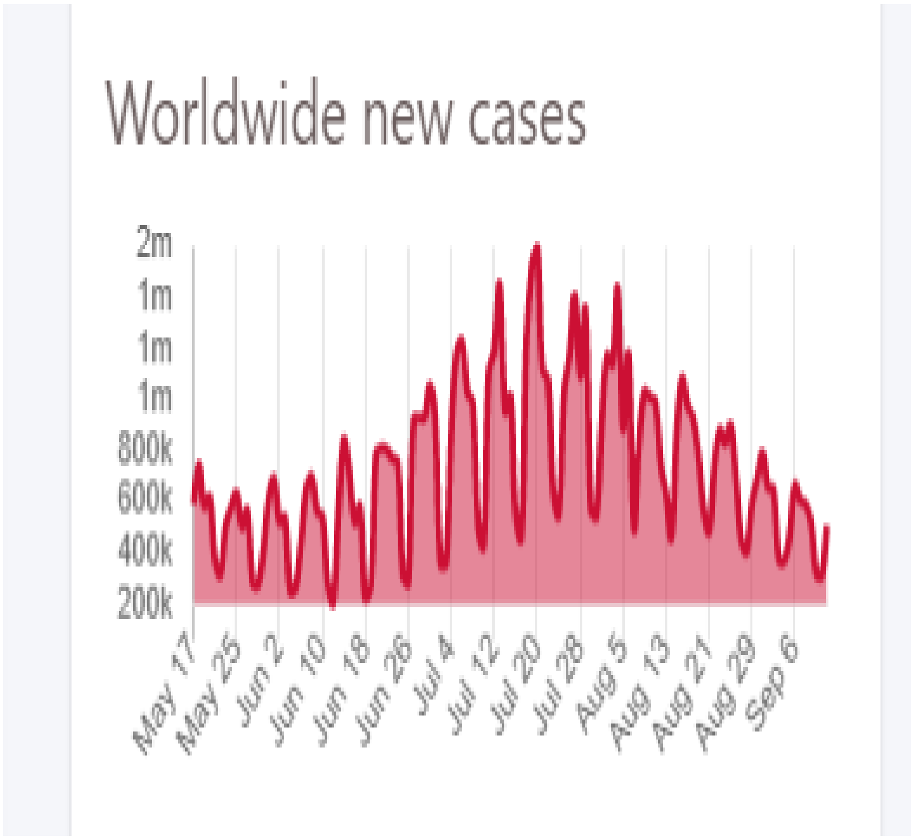


Live Cases by Country

USA	97,200,706
India	44,510,057
France	34,764,561
Brazil	34,591,362
Germany	32,558,479
S. Korea	24,193,038
UK	23,554,519
Italy	22,077,601
Japan	20,300,387
Russia	20,157,143

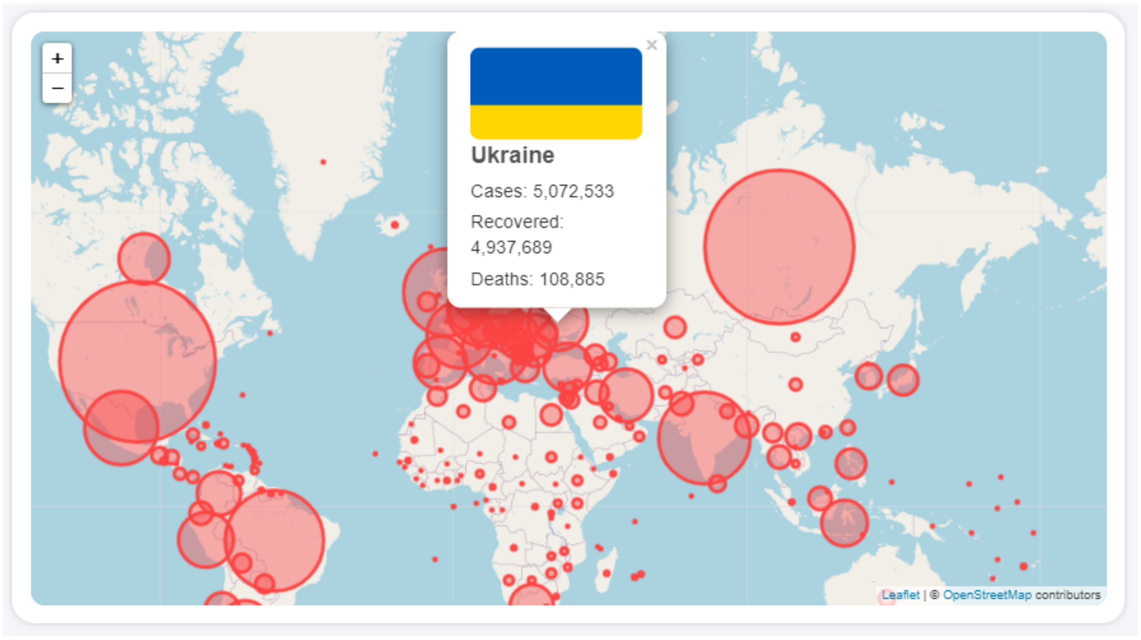
3.3.1: cases to country list

3.4 GRAPH OF CASES ACCORDING TO MONTH



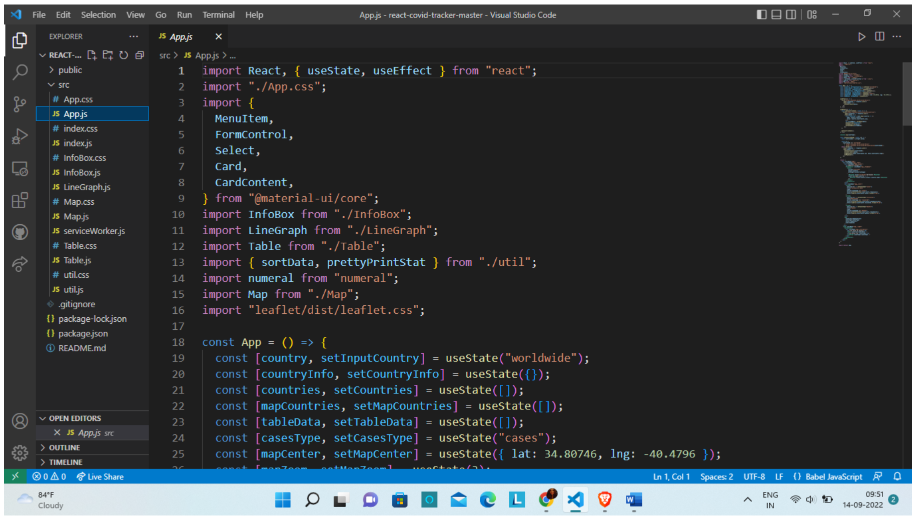
3.4.1: Graph

3.5 MAP INTERFACE



3.5.1: map interface

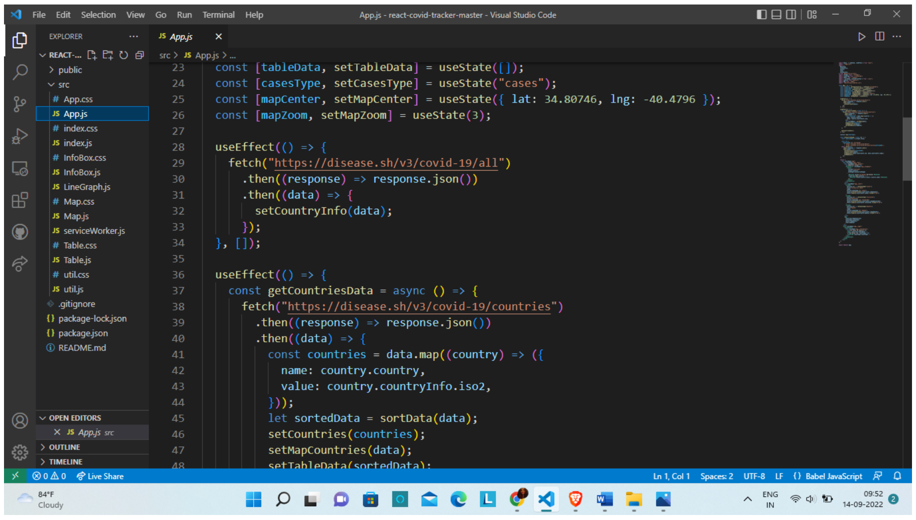
3.6 Code 1



```
import React, { useState, useEffect } from "react";
import "./App.css";
import {
  MenuItem,
  FormControl,
  Select,
  Card,
  CardContent,
} from "@material-ui/core";
import InfoBox from "./InfoBox";
import LineGraph from "./LineGraph";
import Table from "./Table";
import { sortData, prettyPrintStat } from "./util";
import numeral from "numeral";
import Map from "./Map";
import "leaflet/dist/leaflet.css";

const App = () => {
  const [country, setInputCountry] = useState("worldwide");
  const [countryInfo, setCountryInfo] = useState({});
  const [countries, setCountries] = useState([]);
  const [mapCountries, setMapCountries] = useState([]);
  const [tableData, setTableData] = useState([]);
  const [casesType, setCasesType] = useState("cases");
  const [mapCenter, setMapCenter] = useState({ lat: 34.80746, lng: -40.4796 });
```

3.7 Code 2

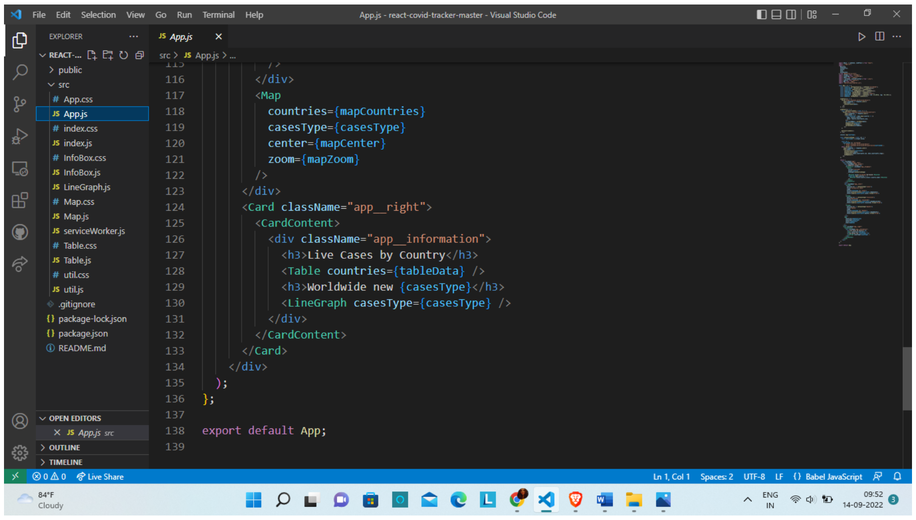


```
  const [tableData, setTableData] = useState([]);
  const [casesType, setCasesType] = useState("cases");
  const [mapCenter, setMapCenter] = useState({ lat: 34.80746, lng: -40.4796 });
  const [mapZoom, setMapZoom] = useState(3);

  useEffect(() => {
    fetch("https://disease.sh/v3/covid-19/all")
      .then((response) => response.json())
      .then((data) => {
        setCountryInfo(data);
      });
  }, []);

  useEffect(() => {
    const getCountriesData = async () => {
      fetch("https://disease.sh/v3/covid-19/countries")
        .then((response) => response.json())
        .then((data) => {
          const countries = data.map((country) => ({
            name: country.country,
            value: country.countryInfo.iso2,
          }));
          let sortedData = sortData(data);
          setCountries(countries);
          setMapCountries(data);
          setTableData(sortedData);
```

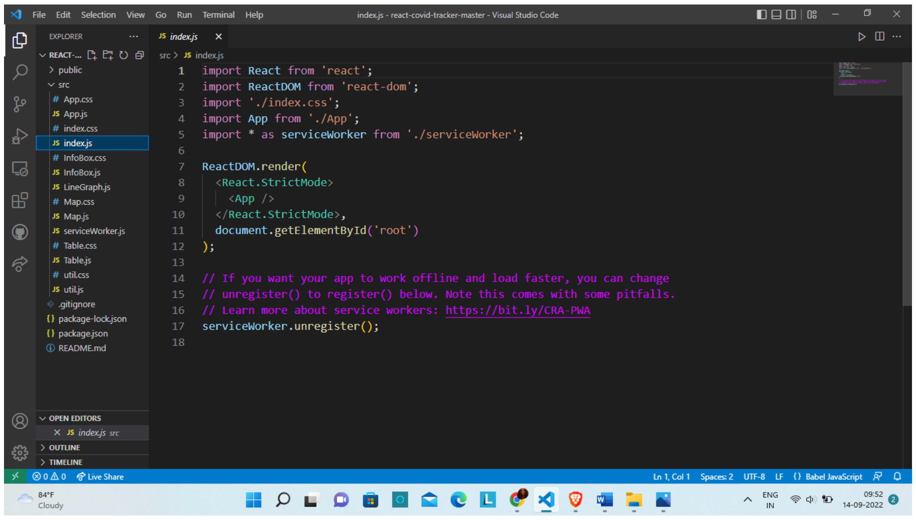
3.8 Code 3



```
        </div>
        <Map
          countries={mapCountries}
          casesType={casesType}
          center={mapCenter}
          zoom={mapZoom}
        />
      </div>
      <Card className="app__right">
        <CardContent>
          <div className="app__information">
            <h3>Live Cases by Country</h3>
            <Table countries={tableData} />
            <h3>Worldwide new {casesType}</h3>
            <LineGraph casesType={casesType} />
          </div>
        </CardContent>
      </Card>
    </div>
  );
};

export default App;
```

3.9 Code 4

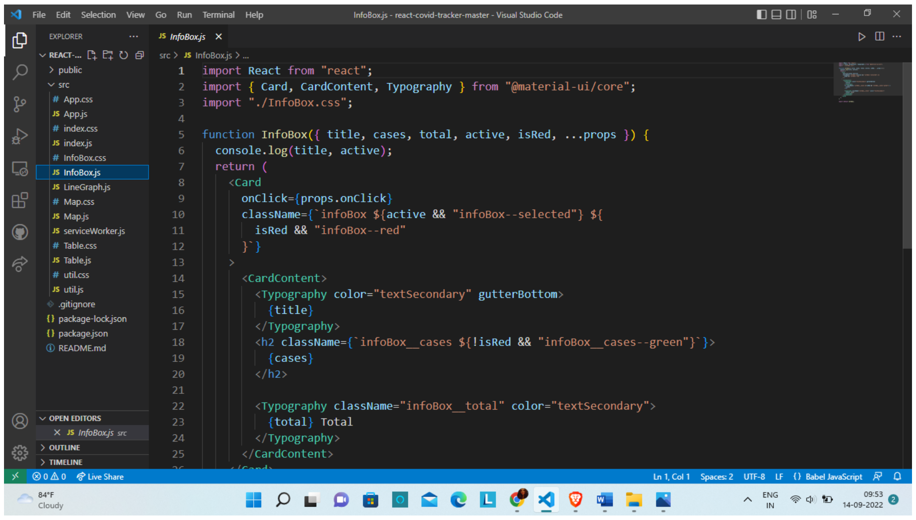


```
import React from 'react';
import ReactDOM from 'react-dom';
import './index.css';
import App from './App';
import * as serviceWorker from './serviceWorker';

ReactDOM.render(
  <React.StrictMode>
    <App />
  </React.StrictMode>,
  document.getElementById('root')
);

// If you want your app to work offline and load faster, you can change
// unregister() to register() below. Note this comes with some pitfalls.
// Learn more about service workers: https://bit.ly/CRA-PWA
serviceWorker.unregister();
```

3.10 Code 5

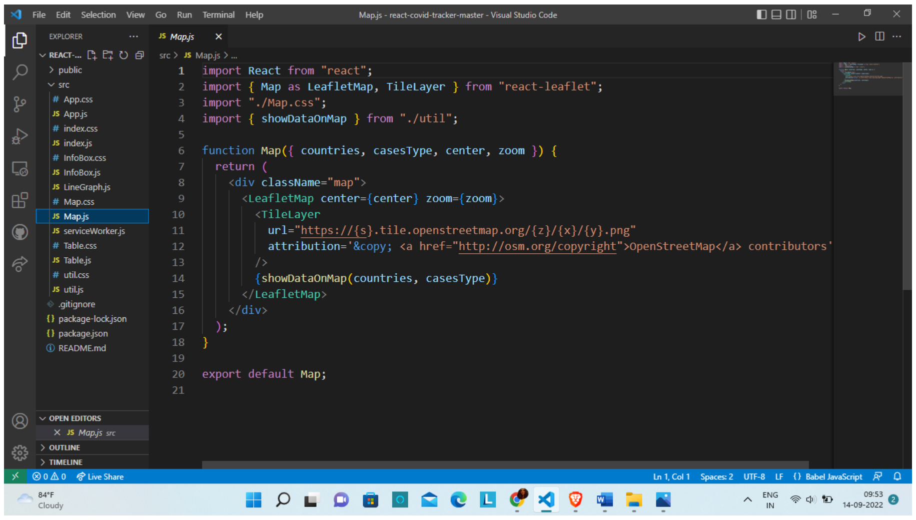


```
import React from "react";
import { Card, CardContent, Typography } from "@material-ui/core";
import "./InfoBox.css";

function InfoBox({ title, cases, total, active, isRed, ...props }) {
  console.log(title, active);
  return (
    <Card
      onClick={props.onClick}
      className={`infoBox ${active && "infoBox--selected"} ${
        isRed && "infoBox--red"
      }`}
    >
      <CardContent>
        <Typography color="textSecondary" gutterBottom>
          {title}
        </Typography>
        <h2 className={`infoBox__cases ${!isRed && "infoBox__cases--green"}`}>
          {cases}
        </h2>

        <Typography className="infoBox__total" color="textSecondary">
          {total} Total
        </Typography>
      </CardContent>
```

3.11 Code 6

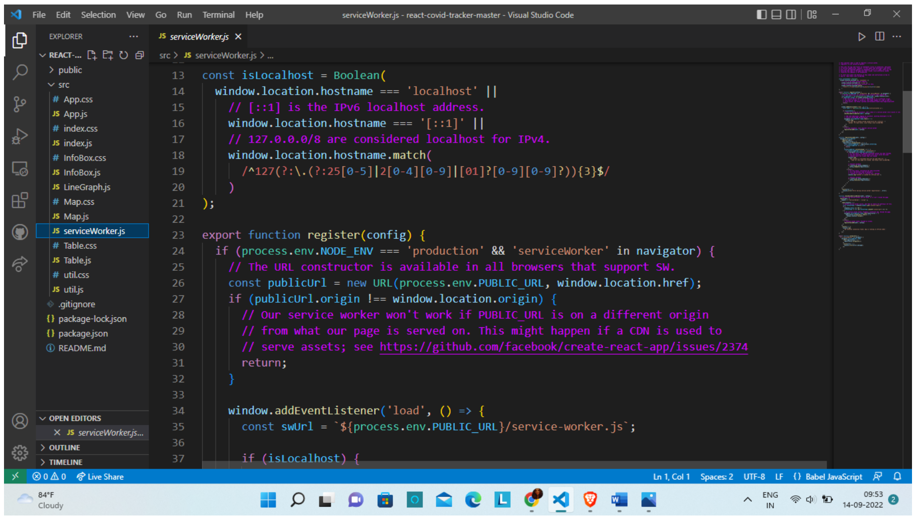


```
import React from "react";
import { Map as LeafletMap, TileLayer } from "react-leaflet";
import "./Map.css";
import { showDataOnMap } from "./util";

function Map({ countries, casesType, center, zoom }) {
  return (
    <div className="map">
      <LeafletMap center={center} zoom={zoom}>
        <TileLayer
          url="https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png"
          attribution='&copy; <a href="http://osm.org/copyright">OpenStreetMap</a> contributors'
        />
        {showDataOnMap(countries, casesType)}
      </LeafletMap>
    </div>
  );
}

export default Map;
```

3.12 Code 7


serviceWorker.js - react-covid-tracker-master - Visual Studio Code
EXPLORER
serviceWorker.js
src > serviceWorker.js > ...
Ln 1, Col 1 Spaces: 2 UTF-8 LF Babel JavaScript

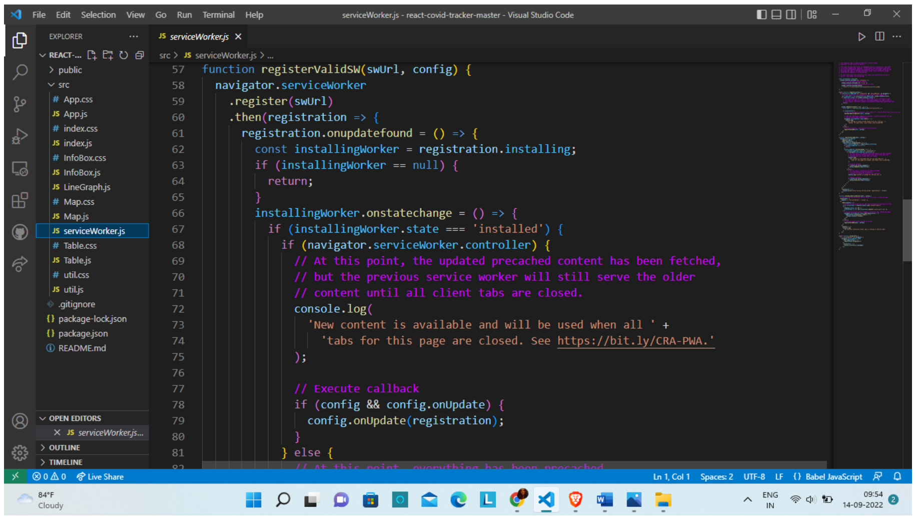
```javascript
const isLocalhost = Boolean(
  window.location.hostname === 'localhost' ||
    // [::1] is the IPv6 localhost address.
    window.location.hostname === '[::1]' ||
    // 127.0.0.0/8 are considered localhost for IPv4.
    window.location.hostname.match(
      /^127(?:\.(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)){3}$/
    )
);

export function register(config) {
  if (process.env.NODE_ENV === 'production' && 'serviceWorker' in navigator) {
    // The URL constructor is available in all browsers that support SW.
    const publicUrl = new URL(process.env.PUBLIC_URL, window.location.href);
    if (publicUrl.origin !== window.location.origin) {
      // Our service worker won't work if PUBLIC_URL is on a different origin
      // from what our page is served on. This might happen if a CDN is used to
      // serve assets; see https://github.com/facebook/create-react-app/issues/2374
      return;
    }

    window.addEventListener('load', () => {
      const swUrl = `${process.env.PUBLIC_URL}/service-worker.js`;

      if (isLocalhost) {
```

3.13 Code 8


serviceWorker.js - react-covid-tracker-master - Visual Studio Code
EXPLORER
serviceWorker.js
src > serviceWorker.js > ...
Ln 1, Col 1 Spaces: 2 UTF-8 LF Babel JavaScript

```javascript
function registerValidSW(swUrl, config) {
  navigator.serviceWorker
    .register(swUrl)
    .then(registration => {
      registration.onupdatefound = () => {
        const installingWorker = registration.installing;
        if (installingWorker == null) {
          return;
        }
        installingWorker.onstatechange = () => {
          if (installingWorker.state === 'installed') {
            if (navigator.serviceWorker.controller) {
              // At this point, the updated precached content has been fetched,
              // but the previous service worker will still serve the older
              // content until all client tabs are closed.
              console.log(
                'New content is available and will be used when all ' +
                  'tabs for this page are closed. See https://bit.ly/CRA-PWA.'
              );

              // Execute callback
              if (config && config.onUpdate) {
                config.onUpdate(registration);
              }
            } else {
              // At this point, everything has been precached
```

4 Test Cases

Table 6.1: Test Case - Login and Register

Test Case ID	Test Case Name	Test Data	Expected Output	Actual Output	Result
1	System fetches total cases, recovered cases and deaths.	Correct data for the three shows up	Correct data shows	Valid data	pass
2	User clicks a country in MAP	The country's data shows up	Right country is displayed	Valid country	Pass
3	User clicks on recovered cases	Map of recovered cases is displayed	Recovered cases map displayed	Valid map	Pass
4	Fetching data	Data of few countries is not fetched properly	To be fetched correct data	No Data	Fail

5 Limitations

- It needs internet to be accessed.
- It does not have a warning message for when the cases exceed a limit.
- It fetches from an API.
- It requires the API to be updated

6 Future Enhancements

- The website should display warning sign for areas where the covid-19 numbers are too high
- The tracker should track state wise too.
- Tracking can be more precisely done on the map.

7 Bibliography

7.1 Web References

[1.] `https://v4.mui.com/`
[2.] `https://www.youtube.com`
[3.] `https://stackoverflow.com/`
[4.] `https://www.draw.io/`
[5.] `https://react-leaflet.js.org/`
[6.] `https://www.geeksforgeeks.org/`
[7.] `https://www.npmjs.com/package/react-numeral`