

A Project Report

on

**Detecting COVID-19 with Chest X-Ray with Geo
visualization of the Dataset**

by

Rohit Naikade (TECOC339)

Mehul Lokhande (TECOC331)

Vijay Dabhade (TECOC308)

Govind Madankar (TECOC332)

Under the guidance of

Prof. Rajesh Lomte

**DEPARTMENT OF COMPUTER ENGINEERING,
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
SECTOR26, NIGDI, PRADHIKARAN**

SAVITRIBAI PHULE PUNE UNIVERSITY

Academic Year: 2020-21

SEMESTER I

**DEPARTMENT OF COMPUTER ENGINEERING,
PIMPRI CHINCHWAD COLLEGE OF ENGINEERING
SECTOR26, NIGDI, PRADHIKARAN**

Date: 09/12/2020

CERTIFICATE

This is to certify that,

Rohit Naikade (TECOC339)

Mehul Lokhande (TECOC331)

Vijay Dabhade (TECOC308)

Govind Madankar (TECOC332)

of class T.E Computer Engineering; have successfully completed their project work on “Detecting COVID-19 with Chest X-Ray with Geo visualization of the Dataset” at PIMPRI CHINCHWAD COLLEGE OF ENGINEERING in the partial fulfillment of the Graduate Degree course in T.E at the Department of **Computer Engineering**, in the academic Year 2020-2021 Semester – I as prescribed Savitribai Phule Pune University.

Prof. Rajesh Lomte

Acknowledgements

We feel immense pleasure in presenting this project report on “Detecting COVID-19 with Chest X-Ray with Geo visualization of the Dataset”. We wish to express true sense of gratitude towards **Prof. Rajesh Lomte**, our project guide who at very discrete step in study of this project, contributed his valuable guidance and helped to solve every problem that arose.

We take this opportunity to thank our Principal **Dr. N.B. Chopade** and **Prof. Dr. K. Rajeswari** (HOD Comp Dept.) for opening the doors of the department towards the realization of our project, for their guidance and encouragement.

Most importantly, we would also like to express our sincere gratitude towards all the staff members of Computer Department. We also express our thanks to all our friends for their support and suggestions shown during the completion of our project. We take this opportunity to express our thanks to all who rendered their valuable help, along with all those unseen people across the internet for maintaining those valuable resources for the successful completion of our project. I owe our success to all of them.

Rohit Naikade (TECOC339)

Mehul Lokhande (TECOC331)

Vijay Dabhade (TECOC308)

Govind Madankar (TECOC332)

Contents

| Sr. No. | Topic | Page No. |
|------------------|--|-----------------|
| | Acknowledgement | I |
| | Contents | II |
| | List of Tables | III |
| | Abstract | IV |
| Chapter-1 | Introduction | 1 |
| | 1.1 Motivation | 1 |
| | 1.2 Problem Statement | 2 |
| | 1.3 Project Objectives | 2 |
| Chapter-2 | Literature Survey/ Requirement Analysis | 3 |
| | 2.1 Introduction | 3 |
| | 2.2 Existing methodologies/System | 4 |
| | 2.3 Proposed methodologies/System | 5 |
| | 2.4 Project Plan | 6 |
| Chapter-3 | Project Design | 8 |
| | 3.1 Hardware and Software Requirements in detail | 8 |
| | 3.2 Modules in Project | 9 |
| | 3.3 Database Design | 11 |
| | 3.4 Source Code | 14 |
| Chapter-4 | Results | 30 |
| | 4.1 GUI Forms | 30 |
| | 4.2 Sample Database Tables | 37 |
| | 4.3 Output Graphs/Tables | 39 |
| Chapter-5 | Conclusion | 44 |
| | References | 45 |

List of Images

| Image No. | Title | Page No. |
|-----------|---|----------|
| 2.1 | Architecture Diagram | 7-8 |
| 3.1 | Test and Train Dataset | 11-12 |
| 3.2 | API response in postman | 13 |
| 4.1 | Folder Structure | 30 |
| 4.2 | Home Tab | 31 |
| 4.3 | Features Tab | 32 |
| 4.4 | X-ray Prediction Tab | 33 |
| 4.5 | Dataset visualization Tab | 34 |
| 4.6 | Contact Us Tab | 35 |
| 4.7 | About Us Tab | 36 |
| 4.8 | Dataset Folder Structure | 37 |
| 4.9 | Dataset Classes | 38-39 |
| 4.10 | Base Map | 39 |
| 4.11 | Choloropleth Map | 40 |
| 4.12 | Heat Map | 41 |
| 4.13 | Classification Model Prediction Results | 42-43 |

Abstract

Our project is aimed at providing solution to detect covid-19 cases by detecting patient's chest x-rays using pre-trained Resnet-18 machine learning model. This solution will increase speed of detection of covid-19 cases with accuracy. Nowadays, our doctors, nurses are facing problems to detect covid-19 cases rapidly. This project will help to those covid-19 warriors to easily classify and identify the current condition of patient. Our model will work in chest x-ray and classify that image in three classes as normal, viral pneumonia and covid-19. This model will detect condition of patient within a seconds.it will bring speed and accuracy in covid-19 rapid tests.

In second part, we are visualizing live covid-19 related data over the globe using choropleth and heat maps. This will visualize current active cases, death cases, recovered cases in countries over the globe. We have live web dashboard with choropleth maps with live covid-19 related data. We are fetching live covid-19 data from <https://covid19api.com/> api which returns data in json format. After performing data cleaning, It is visualized using choropleth maps. This live data will be updated daily as countries declares their daily covid-19 records.

Keywords:Resnet-18 model, choropleth maps, speed and accuracy, geovisualization of live covid-19 related data, JSON data, data cleaning, pre-trained machine learning model.

Introduction

Introduction

This chapter is organized as follows.

Section 1.1 describes the Motivation for the project.

Section 1.2 explains Problem Statement

Section 1.3 lists the project objectives.

1.1 Motivation

- Watching the huge impact of covid-19 on global level, it is time to design and develop some smart and effective solutions.
- Medical staff and doctors have played a major role in tackling the pandemic.
- In order to assist these Covid warriors, we have designed a machine learning model that helps classifying and identifying the respective condition of the patient.
- This will provide consulting advice to doctors and help in speedy treatment.
- For assisting administrative field, we are developing a dashboard which provides geo visualisation of the cases around different regions of the world.

1.2 Problem Statement

1. Create an image classification model that can predict Chest X-Ray scans that belong to one of the three classes with a reasonably high accuracy.
2. Visualize the number of cases of covid-19 using different maps.
3. Integrate these parts into a Dashboard designed in Flask.

1.3 Project Objectives

- Create an image classification model that can classify the Chest X-Ray scans into one of the 3 categories – Normal , Covid-19 or Viral Pneumonia.
- Classifying, will help predict the accurate disease or result.
- Generate different visual maps like – Heat Map , Choropleth Map etc. with Circular Markers for better visuals.
- Convert Jupyter notebook to dashboard.

Literature Survey/ Requirement Analysis

Introduction

This chapter is organized as follows.

Section 2.1 Introduction

Section 2.2 Existing methodologies/System

Section 2.3 Proposed methodologies/System

Section 2.4 Project Plan

2.1 Introduction:

The COVID-19 pandemic has led to a dramatic loss of human life worldwide and presents an unprecedented challenge to public health, food systems and the world of work. The economic and social disruption caused by the pandemic is devastating. The pandemic has been affecting the entire food system and has laid bare its fragility. Millions of enterprises face an existential threat.

Thus, Countries dealing with existing humanitarian crises or emergencies are particularly exposed to the effects of COVID-19. Responding swiftly to the pandemic, while ensuring that humanitarian and recovery assistance reaches those most in need, is critical.

2.2 Existing methodologies/System

- Chest X-Ray (CXR) is one of the important, non-invasive clinical adjuncts that play an essential role in the preliminary investigation of different pulmonary abnormalities.
- It can act as an alternative screening modality for the detection of nCOVID-19 or to validate the related diagnosis, where the CXR images are interpreted by expert radiologists to look for infectious lesions associated with nCOVID-19.
- The earlier studies reveal that the infected patients exhibit distinct visual characteristics in CXR images. These characteristics typically include multi-focal, bilateral ground-glass opacities and patchy reticular (or reticulonodular) opacities in non-ICU patients, while dense pulmonary consolidations in ICU patients .
- **However, the manual interpretation of these subtle visual characteristics on CXR images is challenging and require domain expert.**
- **Moreover, the exponential increase in the number of infected patients makes it difficult for the radiologist to complete the diagnosis in time, leading to high morbidity and mortality.**

2.3 Proposed methodologies/System

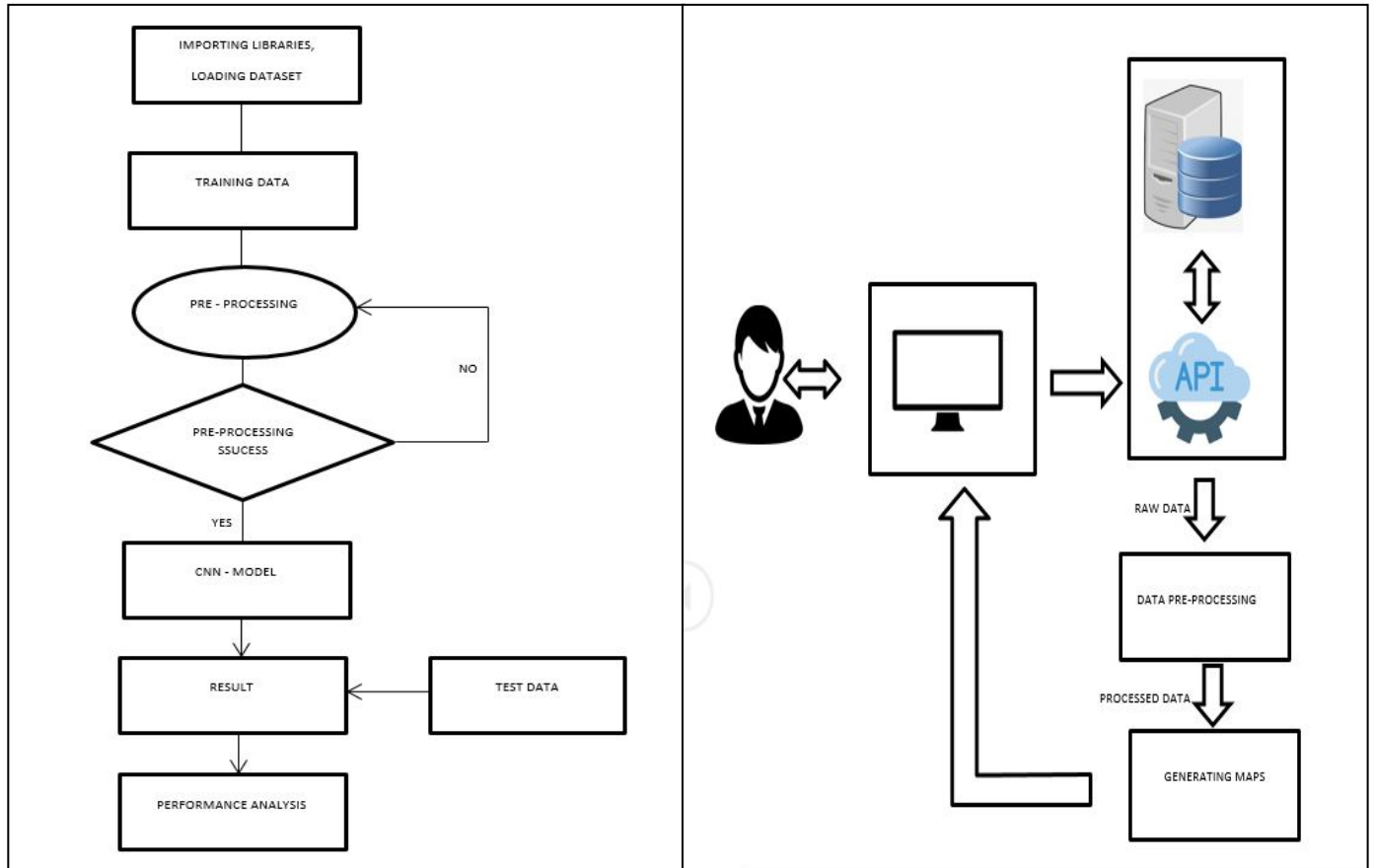
- To fight against nCOVID-19 epidemic, the recent machine learning (ML) techniques can be embedded to develop an automatic computer-aided diagnosis (CAD) system.
- In this direction, many clinical and radiological studies have been reported, describing various radio-imaging findings and epidemiology of nCOVID-19.
- Further, many deep-learning models like deep convolutional network, recursive network, transfer learning models, etc. have been implemented to automatically analyze the radiological disease characteristics used a convolutional neural network (CNN) to assign a class label to different superpixels extracted from the lungs parenchyma and localize tuberculosis-infected regions in CXR images with average dice index of 0.67.
- Another work used two novel models, the first model is based on backpropagation neural network that uses weakly labeled CXR images and generates visual attention feedback for accurate localization of pulmonary lesions; the second model used reinforcement learning-based recurrent attention model, which learns the sequence of images to find the nodules.

2.4 Project Plan

The project will be executed in the following steps :

- Importing Libraries
- Creating Custom Dataset
- Image Transformations
- Prepare DataLoader
- Data Visualization
- Creating the Model
- Training the Model
- Generating a pickle file
- Preparing a Dashboard in Flask
- Integrating the pickle file, Python Maps live dashboard with Flask Website.
- Analyzing the results

Architecture Diagram



Project Design

Introduction

This chapter is organized as follows.

Section 3.1 Hardware and Software Requirements

Section 3.2 Modules In project

Section 3.3 Database Design

Section 3.4 Source Code

3.1 Hardware and Software Requirements

- **Hardware Requirements :**
 - Any OS which can support Jupyter Notebook .

- **Software Requirements :**
 - Python – Flask for Server side routing
 - Anaconda or Jupyter Notebook / Google Colab Account.
 - Python (any version above 3).
 - Browser.
 - Postman
 - IDE – Visual Studio Code

3.2 Modules In project

1. Image Classification Model

- Create custom Dataset and DataLoader in PyTorch :
 - This involves creating a custom Dataset and preparing a DataLoader in Pytorch.
- Train a ResNet-18 model in PyTorch to perform Image Classification :
 - ResNet-18 is a convolutional neural network that is 18 layers deep.
 - It is a Pre-trained Model for PyTorch.
 - Use a ResNet-18 model and train it on a COVID-19 Radiography dataset.

2. Geo visualization of the Dataset

- Access data via an API by sending a GET request to the API.
- Processing the received Data into a suitable JSON format.
- Generate a Choropleth map , heat map etc. with Circular Markers :
 - A Choropleth map has shaded or patterned regions corresponding to a statistical variable that represents an aggregate summary of a geographic characteristic within each area.
 - A Heat map is a data visualization technique that shows magnitude of a phenomenon as two dimensional colors.
- Convert Jupyter notebook to dashboard : This step generates a “.html” file , which provides a suitable dashboard.

3. Web Page Rendering

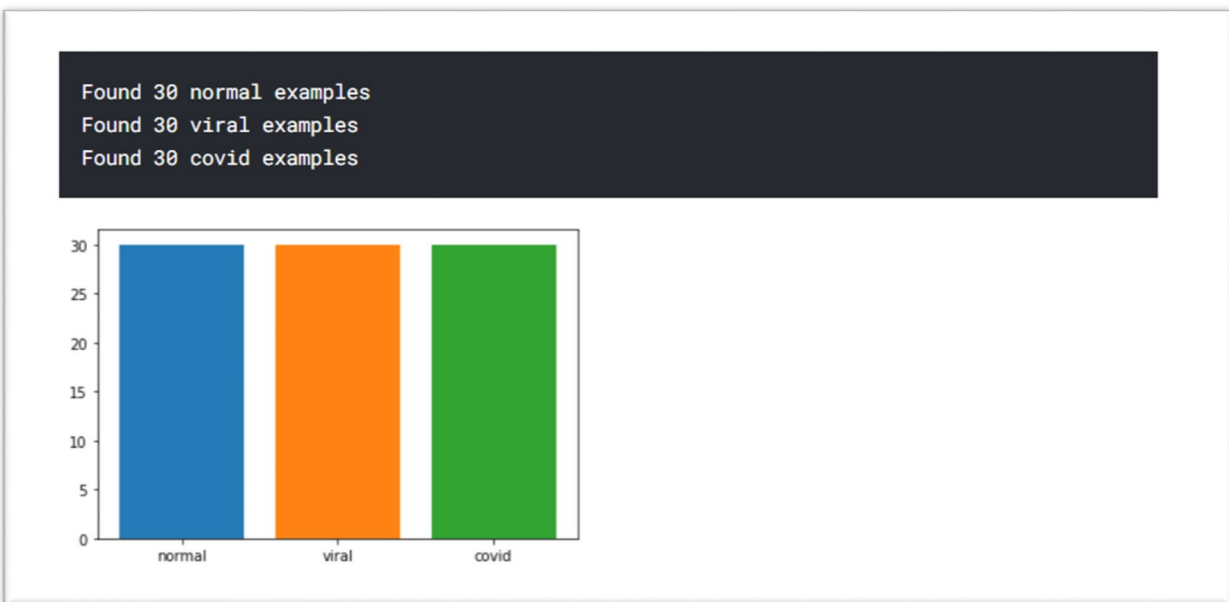
- Flask Frame work is used to render the Web Pages and its routes.
- `render_template()` is used to render the .html file
- POST/GET request are used to render and use the uploaded file.

2.3 Database Design:

2.3.1 Classification Model :

We have created a database of chest X-ray images for COVID-19 positive cases along with Normal and Viral Pneumonia images. In our dataset, there are 219 COVID-19 positive images, 1341 normal images and 1345 viral pneumonia images. We will continue to update this database as soon as we have new x-ray images for COVID-19 pneumonia patients. Researchers can use this database to produce useful and impactful scholarly work on COVID-19, which can help in tackling this pandemic.

- **Test Dataset :**
 - 30 (Normal Class) + 30 (Covid Class) + 30(Viral pneumonia) Images.



- **Training Dataset:**
 - Classes with remaining dataset.

```
Found 1311 normal examples  
Found 1315 viral examples  
Found 189 covid examples
```

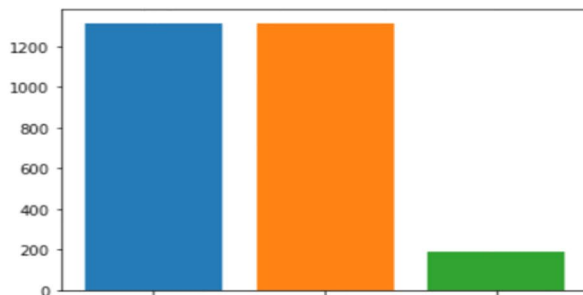


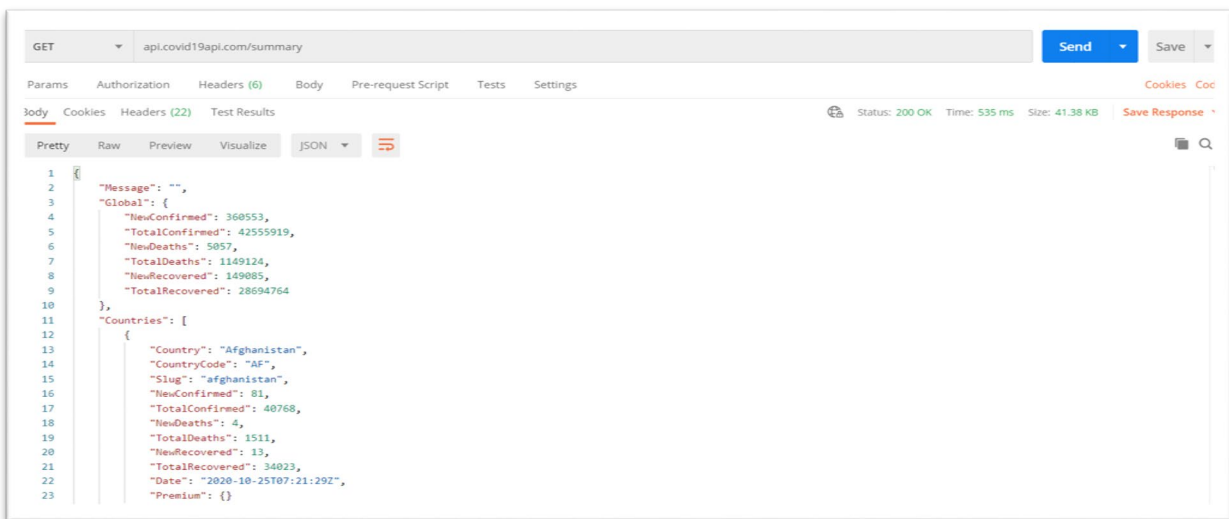
Image Format

- All the images are in Portable Network Graphics (PNG) file format and resolution is 1024*1024 pixels, which can be easily converted to 224*224 or 227*227 pixels typically required by the popular Convolutional Neural Networks (CNNs).

2.3.2 Visualisation of Covid Data:

For covid-19 geodata visualization, we are fetching live covid-19 related data from <https://covid19api.com/> api which responds with data in JSON format.

This is the snapshot of API response in postman.



3.4 Source Code:

Full Project Source Code : <https://github.com/Baltic-Knights/SDL-Final-Submission>

1) app.py:

```
from flask import *
import torch
import torchvision
import numpy as np
from distutils.dir_util import copy_tree

from PIL import Image
from matplotlib import pyplot as plt
import os
import shutil
import random

app = Flask(__name__)

@app.route('/')
def hello():
    return render_template('index.html')

@app.route('/covid_map')
def covid():
    return render_template('covid_map.html')

@app.route('/predict', methods=['POST', 'GET'])
def predict():
```

```

if request.method == 'POST':
    #f = request.files.get('file')
    f = request.files['file']
    fname=f.filename
    fname = fname.split(".")
    name=fname[0]

    img_transform = torchvision.transforms.Compose([
        torchvision.transforms.Resize(size = (224,224)),
        torchvision.transforms.RandomHorizontalFlip(),
        torchvision.transforms.ToTensor(),
        torchvision.transforms.Normalize(mean = [0.485, 0.456, 0.406],
                                         std = [0.229, 0.224, 0.225])
    ])

    img = torch.utils.data.DataLoader(img_transform, batch_size = 6,
                                       shuffle = True)

    model = pickle.load(open(r'model.pkl', 'rb'))

    result = model(img)

    return render_template('index.html', pred1="Condition of given X-
ray Scan : {}".format(result))
    #return render_template('index.html', pred1="Success")

if __name__ == "__main__":
    app.run(debug = True)

```

2) index.html

```
<!DOCTYPE html>
<html lang="en">

  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">

    <!-- =====Fav Icon===== -->
    <link rel = "icon" type = "image/png" href = "assets/img/rover1.jp
g">

    <!-- =====linking external css===== -->
    <link rel="stylesheet" href="{{ url_for('static',filename="css/sty
les.css") }}">

    <!-- =====BOX ICONS===== -->
    <link href='https://cdn.jsdelivr.net/npm/boxicons@2.0.5/css/boxico
ns.min.css' rel='stylesheet'>

    <!-- Bootstrap Js -->
    <script src="https://code.jquery.com/jquery-
3.4.1.slim.min.js"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd
/popper.min.js"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js
/bootstrap.min.js"></script>

    <title>X-ray Predictor</title>
  </head>

  <body>

    <!--===== HEADER =====>
    <header class="l-header">
```

```

        <nav class="nav bd-grid">
            <div>
                <a href="#" class="nav__logo">X-ray Predictor</a>
            </div>

            <div class="nav__menu" id="nav-menu">
                <ul class="nav__list">
                    <li class="nav__item"><a href="#features" class="nav__link active">Features</a></li>
                    <li class="nav__item"><a href="#skills" class="nav__link">X-ray Predictor</a></li>
                    <li class="nav__item"><a href="#project" class="nav__link">Covid Live Updates</a></li>
                    <li class="nav__item"><a href="#contact" class="nav__link">Contact Us</a></li>
                    <li class="nav__item"><a href="#hobby" class="nav__link">About Us</a></li>
                </ul>
            </div>

            <div class="nav__toggle" id="nav-toggle">
                <i class='bx bx-menu'></i>
            </div>
        </nav>
    </header>

    <main class="l-main">

        <!--===== HOME ===== banner.png ---->
        <section class="home bd-grid" id="home">
            <div class="home__data">
                <h1 class="home__title">X-
ray <br> Predictor, <br> Covid Live<br> Updates</h1>

                <a href="#contact" class="button">Contact</a>
            </div>

            <div class="home__img">

```

```

        </div>
    </section>

    <!--===== ABOUT =====>
    <section class="features section " id="features">
        <h2 class="section-title">Features</h2>
        <br><br>

        <!--===== Education =====>

        <div class="about__container bd-grid">
            <div class="about__img">
                

                <br><br>
            </div>

            <div>
                <h1 class="about__subtitle">Live Updates</h1><br>
                <p class="about__text">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
sp; Get Live Updates related to Total Corona cases <br> &nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
</p>

            </div>
        </div>

        <br><br><br>

        <!--===== Web development =====>
        <div class="about__container bd-grid">

            <div>
                <h1 class="about__subtitle">Accurate Results</h1><br>
                <p class="about__text">&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
sp;Predict results with high accuracy</p>

```


[illegible]

```

        <div class="skills__container bd-grid" style="text-align: center;">
            <div>
                <h2>Predict X-ray Scans and determine the defect.</h2>

                <div style="text-align: center;">
                    <!-- Section 5 -->
                    <section div class="" id="section-5" >
                        <div class="container">
                            <div class="row">
                                <div class="">
                                    <form class="md-form" action="/predict" method="POST" enctype="multipart/form-data">
                                        <div class="file-field medium">
                                            <div >
                                                <br><br>
                                                <h2><span>Choose file to predict Condition<i class="fas fa-cloud-upload-alt ml-3" aria-hidden="true"></i></span></h2><input class="button" name="file" type="file">
                                            </div>
                                            <br><br><br>
                                            <button class="button">
                                                PREDICT
                                            </button>
                                        </div>
                                    </form>
                                </div>
                            </div>
                        </div>
                    </div>
                </div>
            </div>
            <div class="about__img">
                
                <br><br>
            </div>
            <div class="container">
                <h2>Result</h2>

```

```

        <div id="pred"><h2>{{pred1}}</h2></div>
    </div>

</div>

<!--==== Projects =====>
<section class="project section" id="project">
    <h2 class="section-title">Covid Live Updates</h2>
    <br><br>

    <div class="project__container bd-grid">
        <div class="contact__img">
            
        </div>

        <div style="text-align: center;">

            <h2>Get Live updates and visualize using different map
s.</h2>

            <br><br>
            <a href="{{ url_for('covid') }}" class="button">Visual
ize</a>

        </div>
    </div>

</section>

<!--==== CONTACT =====>
<section class="contact section" id="contact">
    <h2 class="section-title">Contact Us</h2>
    <br><br>
    <div class="contact__container bd-grid">
        <div class="contact__img">
            
        </div>

```

```

        <div >
            <form action="" class="contact__form">
                <input type="text" placeholder="Name" class="contact__input">
                <input type="mail" placeholder="Email" class="contact__input">
                <textarea name="" id="" cols="0" rows="10" class="contact__input"></textarea>
                <input type="button" value="Submit" class="contact__button button">
            </form>
        </div>
    </div>

    <!--===== Hobbies =====>
    <section class="hobby section" id="hobby">
        <h2 class="section-title">About Us</h2>
        <div class="project__container bd-grid">

            <div class="card" >
                <br><br><br>
                
                <div class="container"><br>
                    <h3 class="about__subtitle">Rohit Naikade</h3>
                    <p>Web Developer</p>
                </div>
                <br>
                <input onclick="window.location.href='https://github.com/MEHUL';" type="button" value="Contact" class="project__button button" ><br><br>
            </div>

            <div class="card"><br>
                
                <div class="container">
                    <h3 class="about__subtitle">Mehul Lokhande</h3>
                    <p>Web Developer</p>
                </div>
            </div>
        </div>
    </section>

```

```

        <br>
        <input onclick="window.location.href='https://github.com/MEHUL';" type="button" value="Contact" class="project__button button" ><br><br>
    </div>

    <div class="card">
        
        <div class="container">
            <h3 class="about__subtitle">Vijayraj Dabhade</h3>
            <p>Web Developer</p>
        </div>
        <br>
        <input onclick="window.location.href='https://github.com/MEHUL';" type="button" value="Contact" class="project__button button" ><br><br>
    </div>

    <div class="card">
        
        <div class="container">
            <h3 class="about__subtitle">Govind Madankar</h3>
            <p>Web Developer</p>
        </div>
        <br>
        <input onclick="window.location.href='https://github.com/MEHUL';" type="button" value="Contact" class="project__button button" ><br><br>
    </div>
</div>
</section>

</section>
<br><br><br><br><br>
</main>

```

```

<!--===== FOOTER =====>
<footer class="footer">

    <div class="about__container bd-grid">

        <div>
            <p class="footer__title">Team Baltic Knight</p>
            <p class="about__subtitle">Pune | India</p>
            <br>
            <div class="footer__social">
                <a href="mailto:mehullokhande18@gmail.com" class="footer__icon"><i class='bx bxl-facebook' ></i></a>
                <a href="https://github.com/MEHUL" class="footer__icon"><i class='bx bxl-instagram' ></i></a>
                <a href="https://www.linkedin.com/in/mehul-lokhande-94167a190/" class="footer__icon"><i class='bx bxl-twitter' ></i></a>
                <a href="https://www.codechef.com/users/mehul25" class="footer__icon"><i class='bx bxl-twitter' ></i></a>
            </div>
        </div>
    </div>

</div>

</footer>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
<!--===== Scroll to top =====>

```

```

<script type="text/javascript">
    var scrolltotop={setting:{startline:100,scrollto:0,scrollduration:
1e3,fadeduration:[500,100]},controlHTML:'',controlattrs:{offsetx:5,off
sety:5},anchorkeyword:"#top",state:{isvisible:!1,shouldvisible:!1},scrollu
p:function(){this.cssfixedsupport||this.$control.css({opacity:0});var t=is
NaN(this.setting.scrollto)?this.setting.scrollto:parseInt(this.setting.scr
ollto);t="string"==typeof t&&1==jQuery("#"+t).length?jQuery("#"+t).offset(
).top:0,this.$body.animate({scrollTop:t},this.setting.scrollduration)},kee
pfixed:function(){var t=jQuery(window),o=t.scrollLeft()+t.width()-
this.$control.width()-
this.controlattrs.offsetx,s=t.scrollTop()+t.height()-
this.$control.height()-
this.controlattrs.offsety;this.$control.css({left:o+"px",top:s+"px"}),tog
glecontrol:function(){var t=jQuery(window).scrollTop();this.cssfixedsuppor
t||this.Keepfixed(),this.state.shouldvisible=t>=this.setting.startline?!0:
!1,this.state.shouldvisible&&!this.state.isvisible?(this.$control.stop().a
nimate({opacity:1},this.setting.fadeduration[0]),this.state.isvisible=!0):
0==this.state.shouldvisible&&this.state.isvisible&&(this.$control.stop().a
nimate({opacity:0},this.setting.fadeduration[1]),this.state.isvisible=!1)}
,init:function(){jQuery(document).ready(function(t){var o=scrolltotop,s=do
cument.all;o.cssfixedsupport=!s||s&&"CSS1Compat"==document.compatMode&&win
dow.XMLHttpRequest,o.$body=t(window.opera?"CSS1Compat"==document.compatMod
e?"html":"body":"html,body"),o.$control=t('<div id="topcontrol">'+o.contro
lHTML+"</div>").css({position:o.cssfixedsupport?"fixed":"absolute",bottom:
o.controlattrs.offsety,right:o.controlattrs.offsetx,opacity:0,cursor:"poin
ter"}).attr({title:"Scroll to Top"}).click(function(){return o.scrollup(),
!1}).appendTo("body"),document.all&&!window.XMLHttpRequest&&"!"=o.$control
.text())&&o.$control.css({width:o.$control.width()}),o.togglecontrol(),t('a
[href="'+o.anchorkeyword+'"]').click(function(){return o.scrollup(),!1}),t
(window).bind("scroll resize",function(t){o.togglecontrol()}))});scrollto
top.init();
</script>

<!--===== SCROLL REVEAL =====>
<script src="https://unpkg.com/scrollreveal"></script>

<!--===== MAIN JS =====>
<script src="{ url_for('static',filename='js/main.js')}"></scrip
t>
</body>

```

```
</html>
```

3) main.js

```
/*===== MENU SHOW =====*/
const showMenu = (toggleId, navId) =>{
  const toggle = document.getElementById(toggleId),
  nav = document.getElementById(navId)

  if(toggle && nav){
    toggle.addEventListener('click', ()=>{
      nav.classList.toggle('show')
    })
  }
}
showMenu('nav-toggle', 'nav-menu')

/*===== ACTIVE AND REMOVE MENU =====*/
const navLink = document.querySelectorAll('.nav__link');

function linkAction(){
  /*Active link*/
  navLink.forEach(n => n.classList.remove('active'));
  this.classList.add('active');

  /*Remove menu mobile*/
  const navMenu = document.getElementById('nav-menu')
  navMenu.classList.remove('show')
}
navLink.forEach(n => n.addEventListener('click', linkAction));
```



```

/*===== SCROLL REVEAL ANIMATION =====*/
const sr = ScrollReveal({
  origin: 'top',
  distance: '80px',
  duration: 2000,
  reset: true
});

/*SCROLL HOME*/
sr.reveal('.home__title',{});
sr.reveal('.button',{delay: 100});
sr.reveal('.home__img',{delay: 100});
sr.reveal('.home__social-icon',{ interval: 100});

/*SCROLL ABOUT*/
sr.reveal('.about__img',{});
sr.reveal('.about__subtitle',{delay: 100});
sr.reveal('.about__text',{delay: 100});

/*SCROLL SKILLS*/
sr.reveal('.skills__subtitle',{});
sr.reveal('.skills__text',{});
sr.reveal('.skills__data',{interval: 100});
sr.reveal('.skills__img',{delay: 100});

/*SCROLL WORK*/
sr.reveal('.work__img',{interval: 100});

/*SCROLL CONTACT*/

```

4) Jupyter Notebooks:

A) Classification Model :

Kaggle Kernel Notebook : <https://www.kaggle.com/mehul25/sdl-project>

B) Visualisation Dash Board (Google Colab Notebook) :

https://colab.research.google.com/drive/15r1lfJLk5yasB2xz42So7Y_gNkq9puAM

Result

Intoduction

This chapter is organized as follows.








Section 4.1 GUI Screenshots

Section 4.2 Database Screenshots

Section 4.3 Graphs, Charts etc.

4.1 GUI Screenshots

1) Folder Structure

| | | | |
|--|------------------|---------------|------|
|  Dataset | 03-12-2020 14:22 | File folder | |
|  env | 02-12-2020 13:00 | File folder | |
|  static | 03-12-2020 13:48 | File folder | |
|  templates | 03-12-2020 12:55 | File folder | |
|  app | 03-12-2020 16:49 | Python File | 1 KB |
|  model.pickle | 02-12-2020 10:41 | PICKLE File | 1 KB |
|  Virtual env | 02-12-2020 19:38 | Text Document | 1 KB |

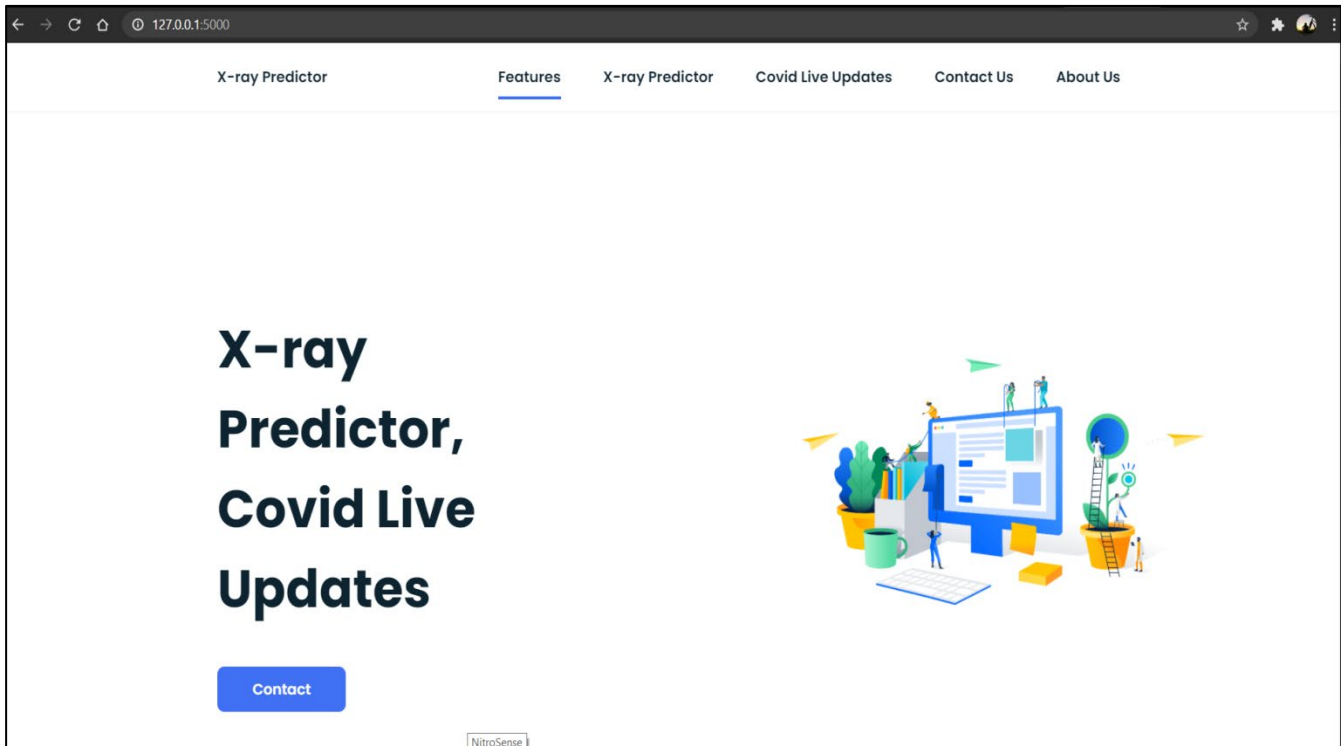
2) Running the Webserver

```
Microsoft Windows [Version 10.0.19041.630]
(c) 2020 Microsoft Corporation. All rights reserved.

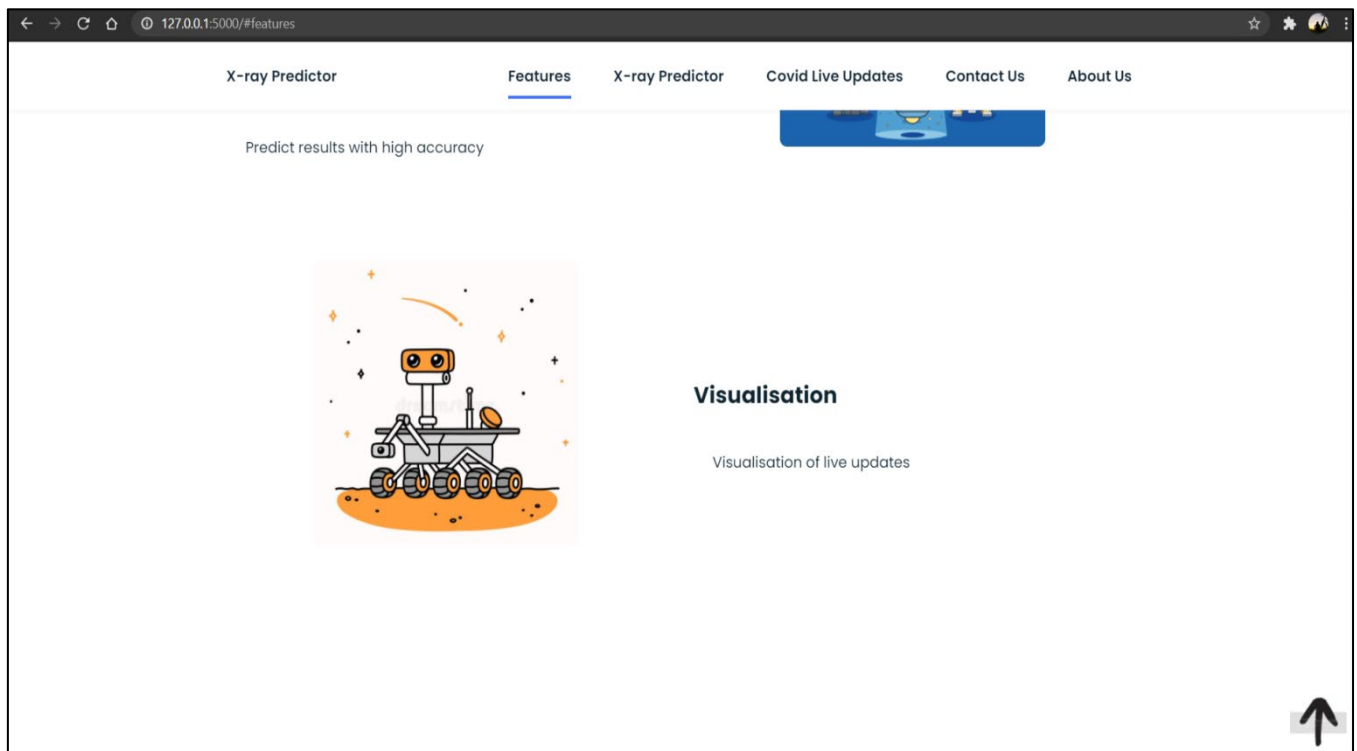
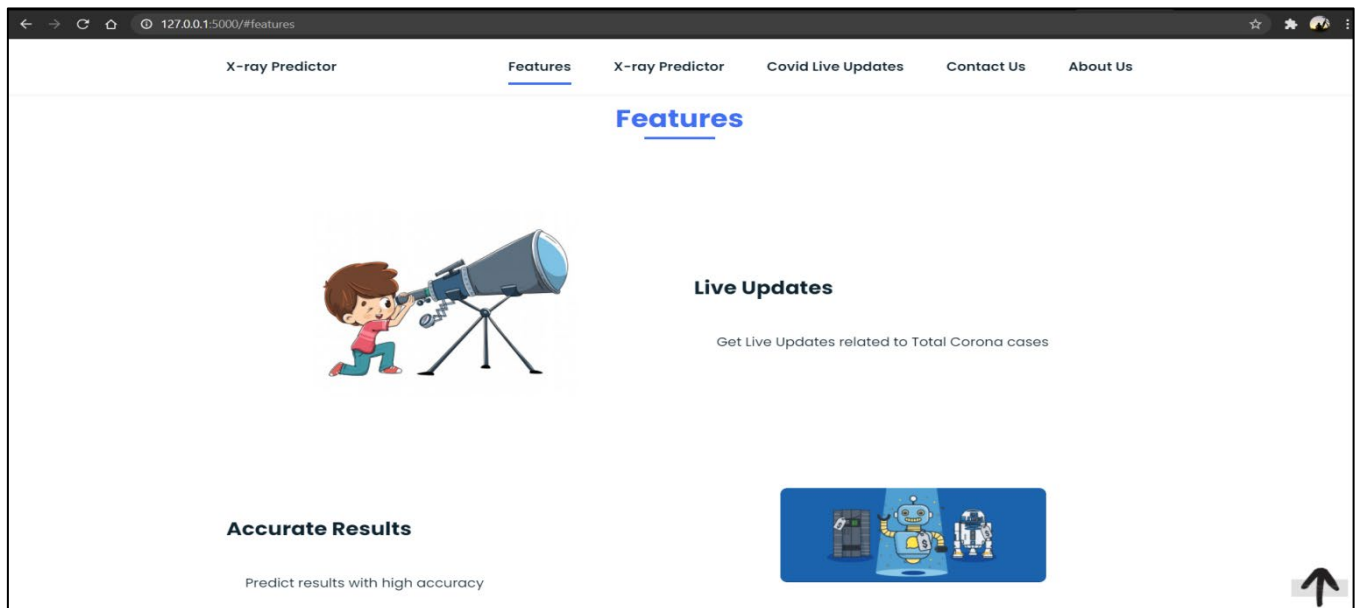
C:\Users\91906\Desktop\Covid>env\Scripts\activate

(env) C:\Users\91906\Desktop\Covid>python app.py
* Serving Flask app "app" (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
* Debugger is active!
* Debugger PIN: 110-001-131
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET / HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[36mGET /static/css/styles.css HTTP/1.1←[0m" 304 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/rover1.jpg HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/tele.jpg HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/about.png HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/cha.png HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/web.png HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/download.jfif HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/banner.png HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/js/main.js HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:21] "←[37mGET /static/img/engi.jfif HTTP/1.1←[0m" 200 -
127.0.0.1 - - [10/Dec/2020 12:41:23] "←[33mGET /assets/img/rover1.jpg HTTP/1.1←[0m" 404 -
```

3) Home Tab

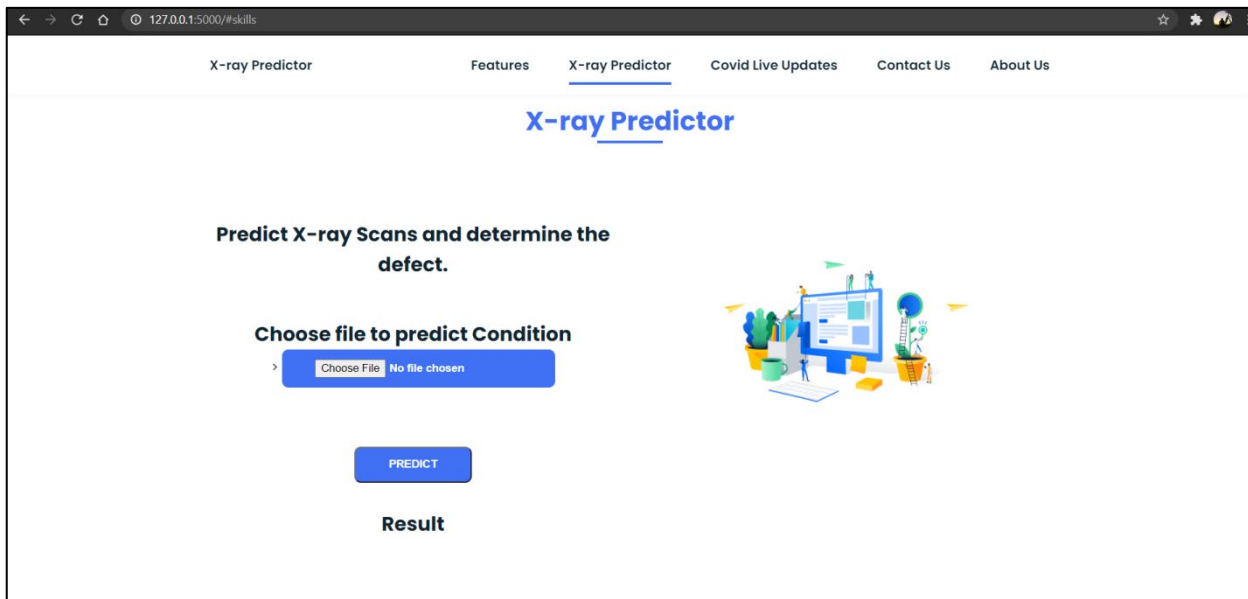


4) Features Tab :

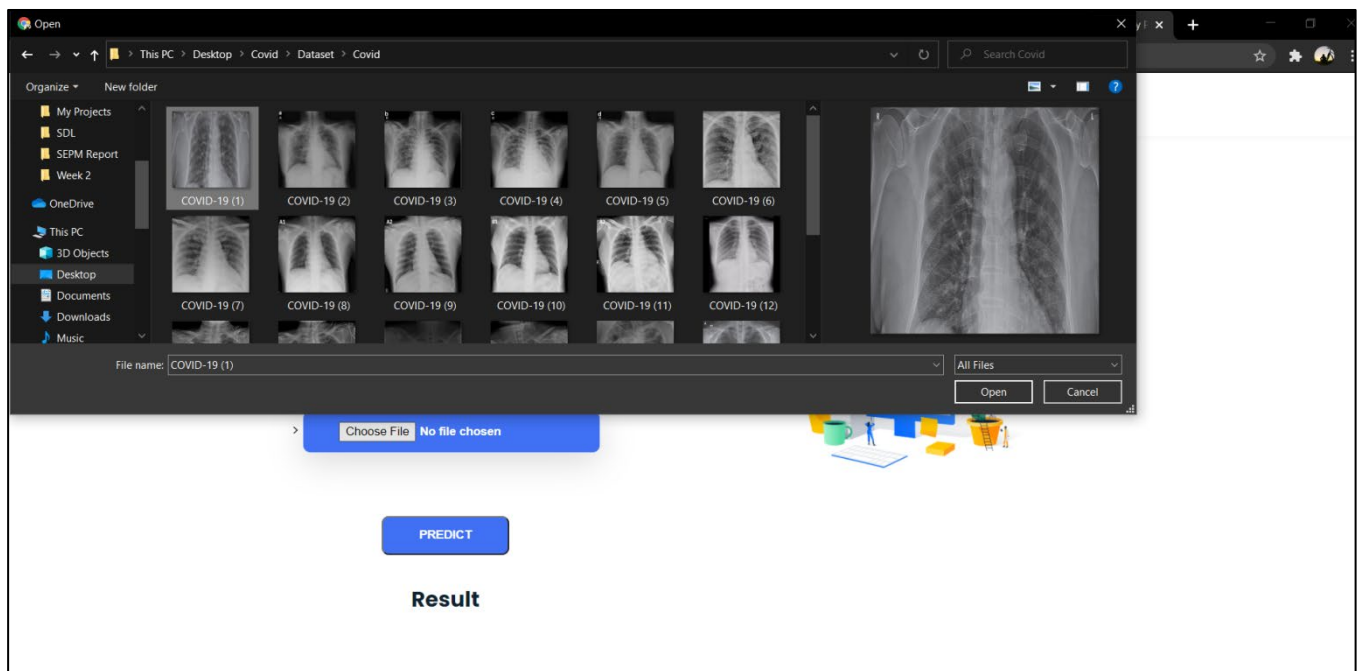


5) X-ray Predication Tab

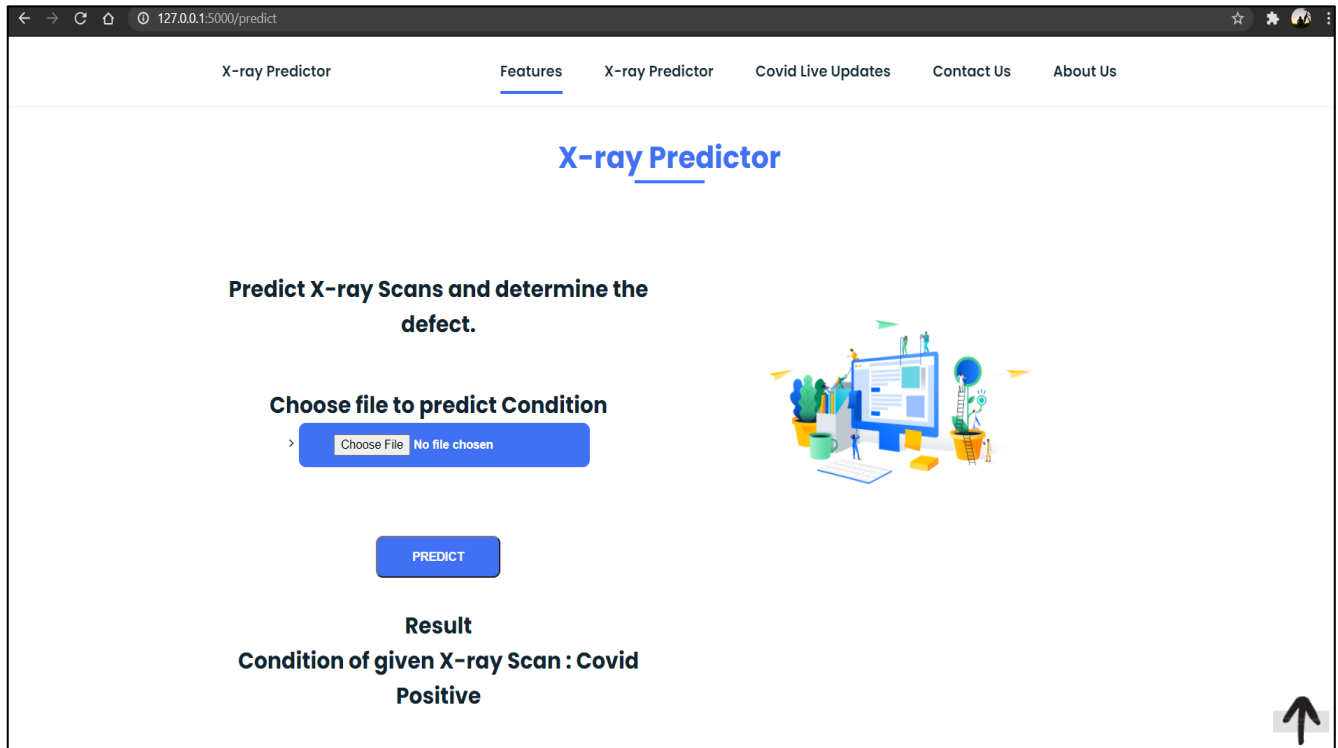
STEP 1 : Click on Choose File and browse window opens.



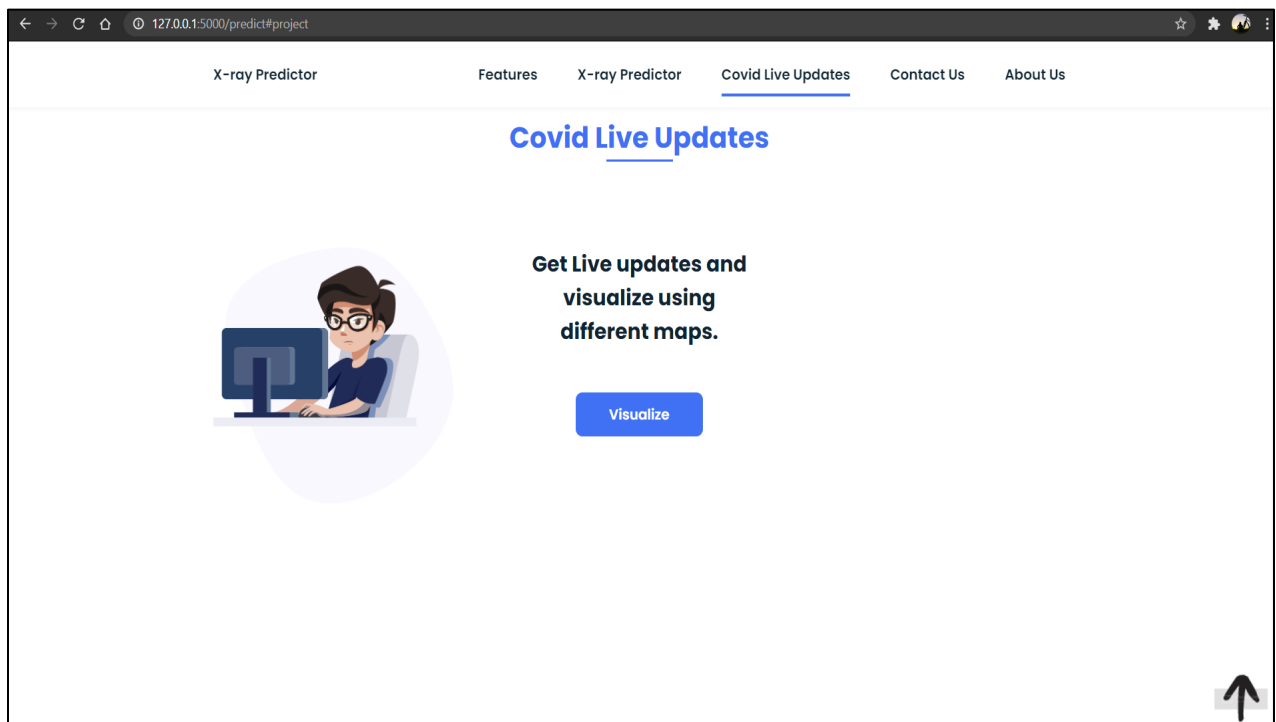
STEP 2: Choose the desired test image.

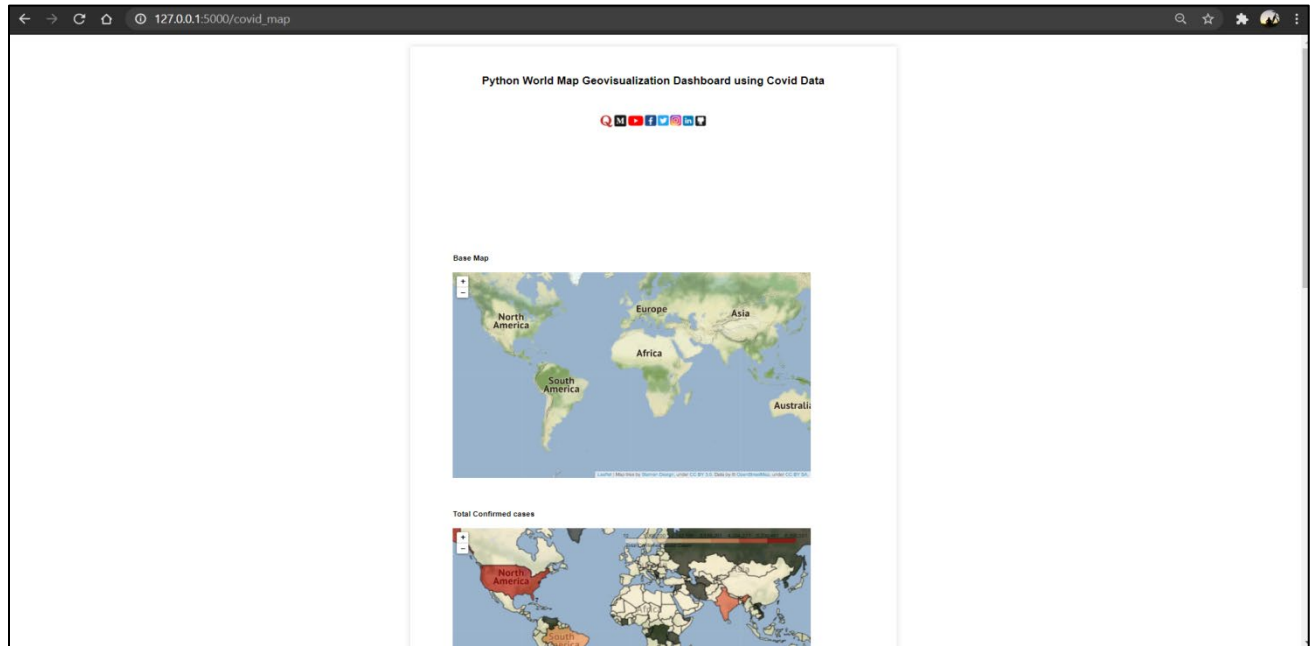


STEP 3: Click on Predict and visualize the result.



6) Dataset Visualisation Dashboard

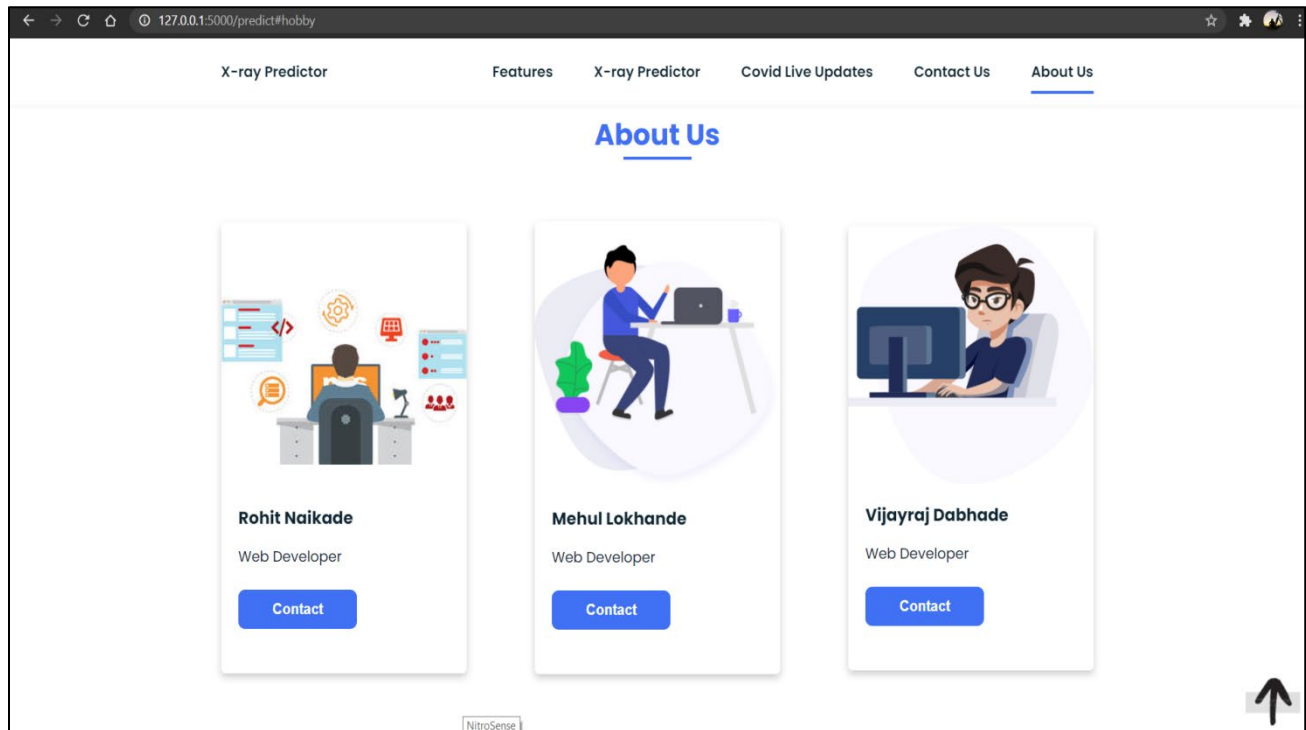




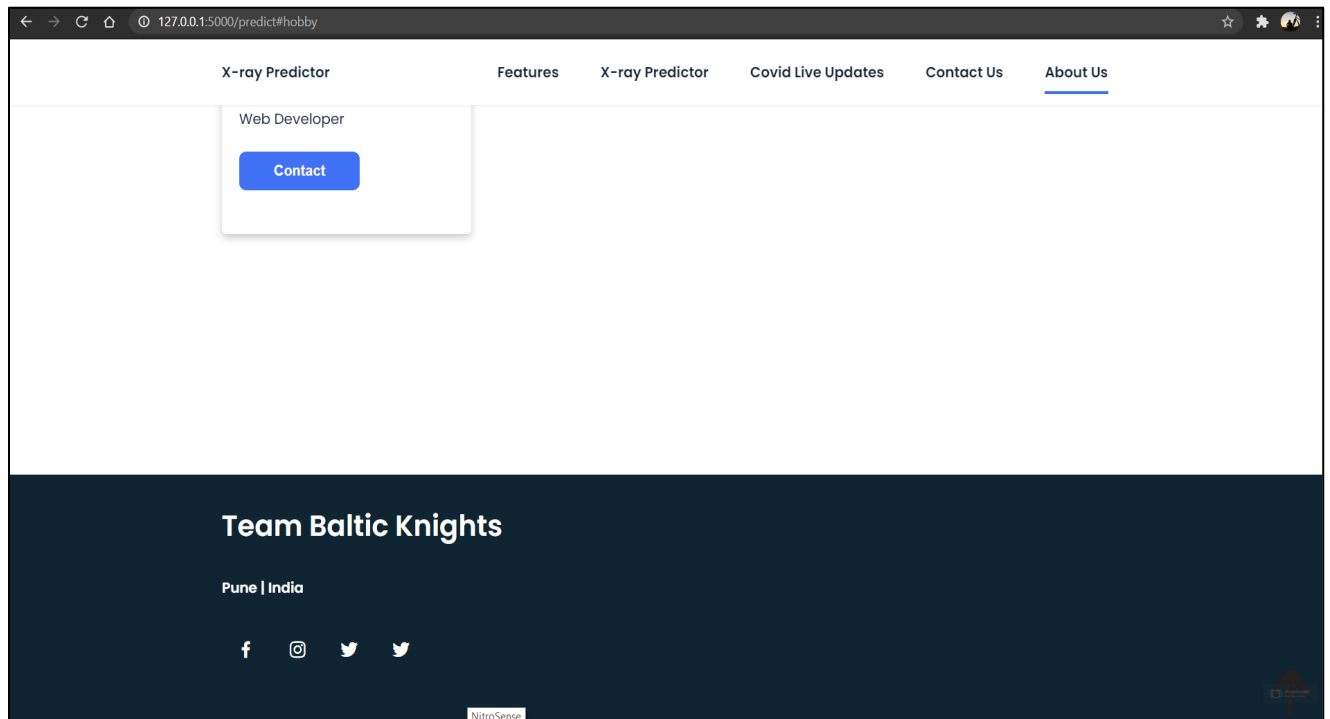
7) Contact Us Tab

The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/predict#contact". The page has a navigation bar with the following links: "X-ray Predictor", "Features", "X-ray Predictor", "Covid Live Updates", "Contact Us" (which is underlined), and "About Us". The main content area is titled "Contact Us" in blue text. On the left side, there is an illustration of a person with glasses sitting at a desk with a computer monitor. On the right side, there is a contact form with three input fields: "Name", "Email", and a larger text area. Below the input fields is a blue "Submit" button. In the bottom right corner, there is a small icon of an upward-pointing arrow.

8) About Us Tab



9) Footer Section



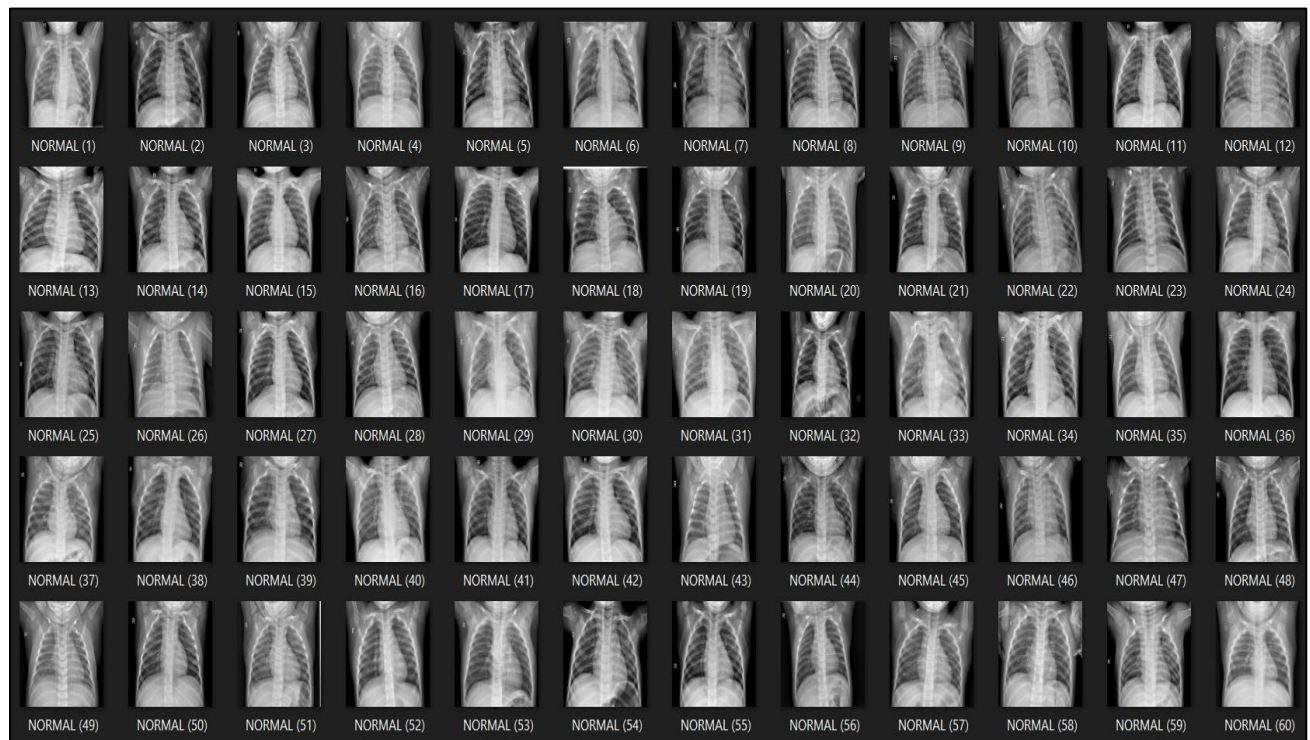
4.2 Database Screenshots

Dataset Folder Structure:

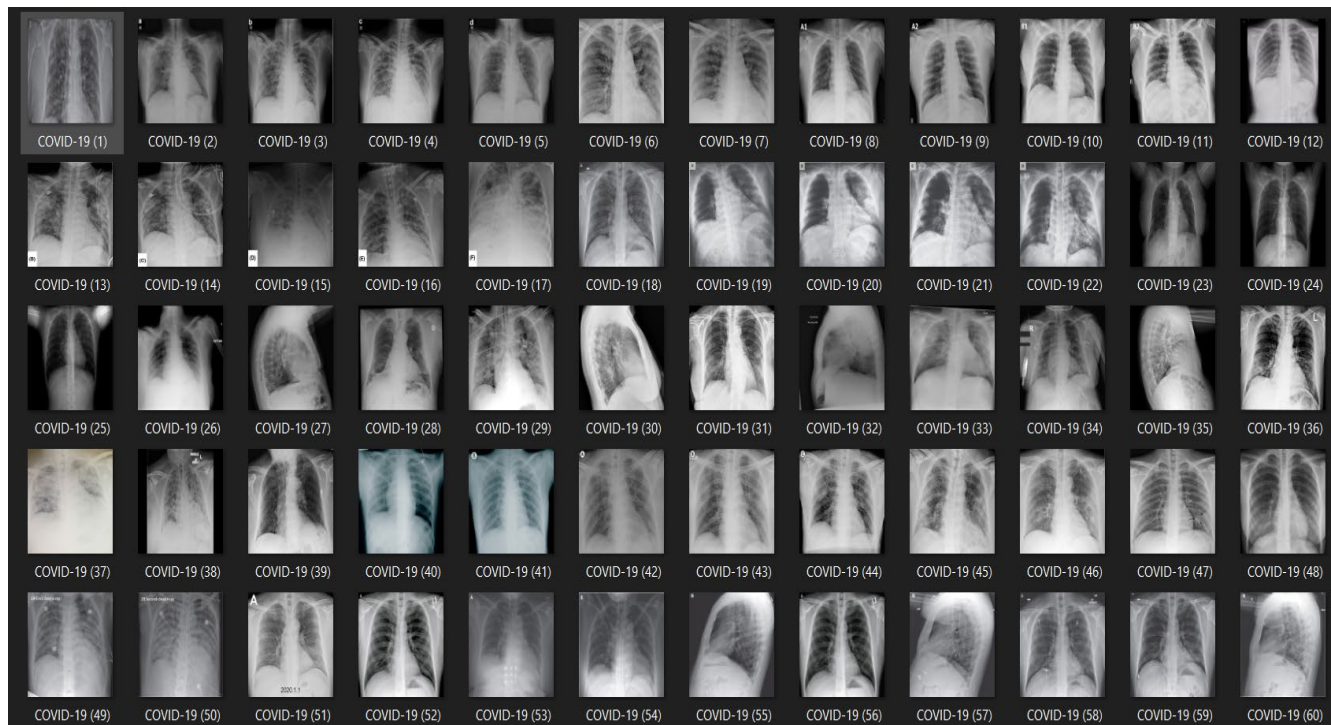
| | | | |
|--------------------------|------------------|----------------------|-------|
| COVID-19 | 19-11-2020 17:05 | File folder | |
| NORMAL | 19-11-2020 17:06 | File folder | |
| Viral Pneumonia | 19-11-2020 17:07 | File folder | |
| COVID-19.metadata | 28-03-2020 23:36 | Microsoft Excel W... | 20 KB |
| NORMAL.metadata | 28-03-2020 23:36 | Microsoft Excel W... | 42 KB |
| README.md | 28-03-2020 23:37 | Text Document | 2 KB |
| Viral Pneumonia.matadata | 28-03-2020 23:37 | Microsoft Excel W... | 39 KB |



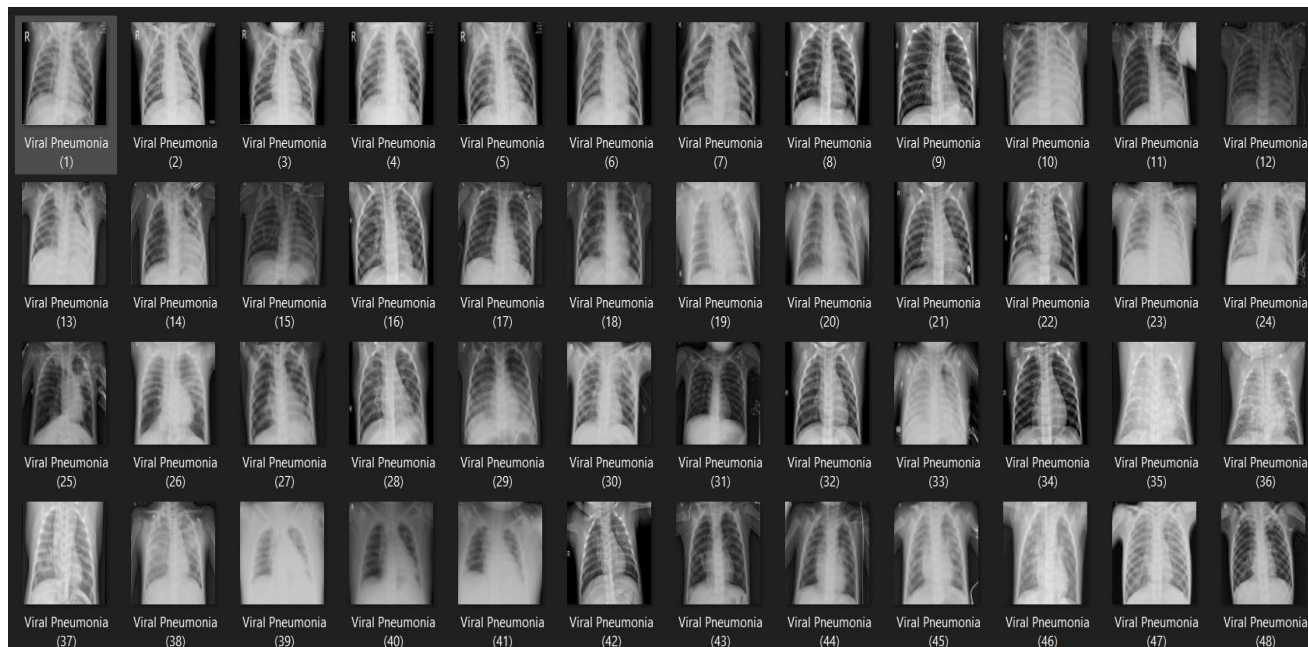
Normal Class Images



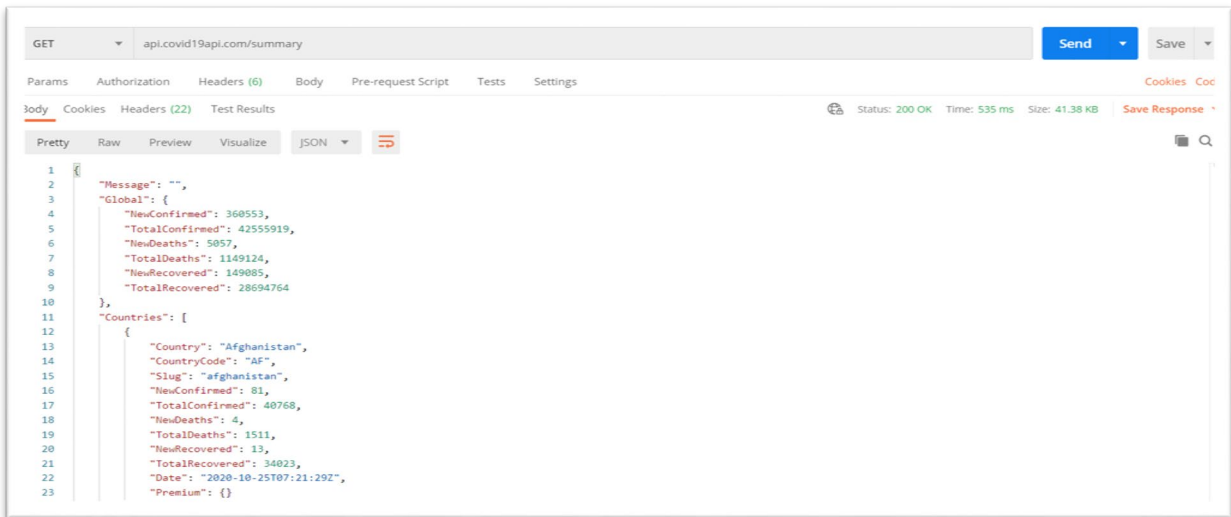
Covid Class Images



Viral pneumonia Class Images



API JSON Response



The screenshot shows a REST client interface with a GET request to `api.covid19api.com/summary`. The response is a JSON object with the following structure:

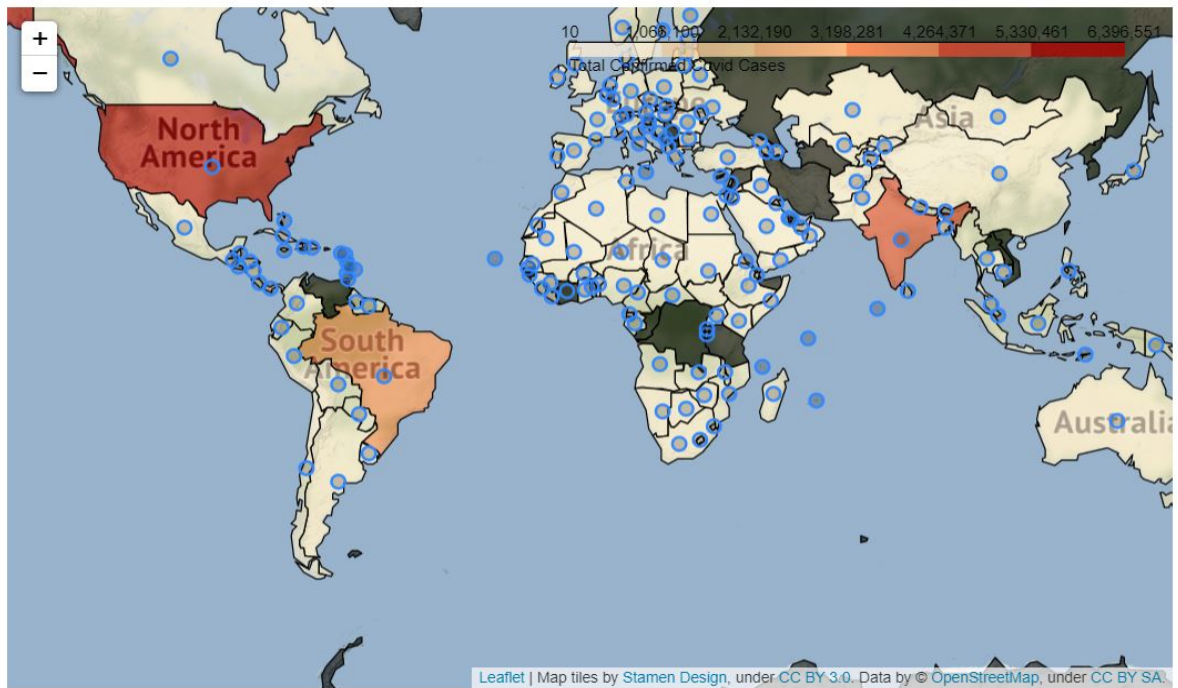
```
1 {
2   "Message": "",
3   "Global": {
4     "NewConfirmed": 360553,
5     "TotalConfirmed": 42555919,
6     "NewDeaths": 5057,
7     "TotalDeaths": 1149124,
8     "NewRecovered": 149085,
9     "TotalRecovered": 28694764
10  },
11  "Countries": [
12    {
13      "Country": "Afghanistan",
14      "CountryCode": "AF",
15      "Slug": "afghanistan",
16      "NewConfirmed": 81,
17      "TotalConfirmed": 40768,
18      "NewDeaths": 4,
19      "TotalDeaths": 1511,
20      "NewRecovered": 13,
21      "TotalRecovered": 34023,
22      "Date": "2020-10-25T07:21:29Z",
23      "Premium": {}
24    }
25  ]
26 }
```

4.3 Output Graphs/Tables:

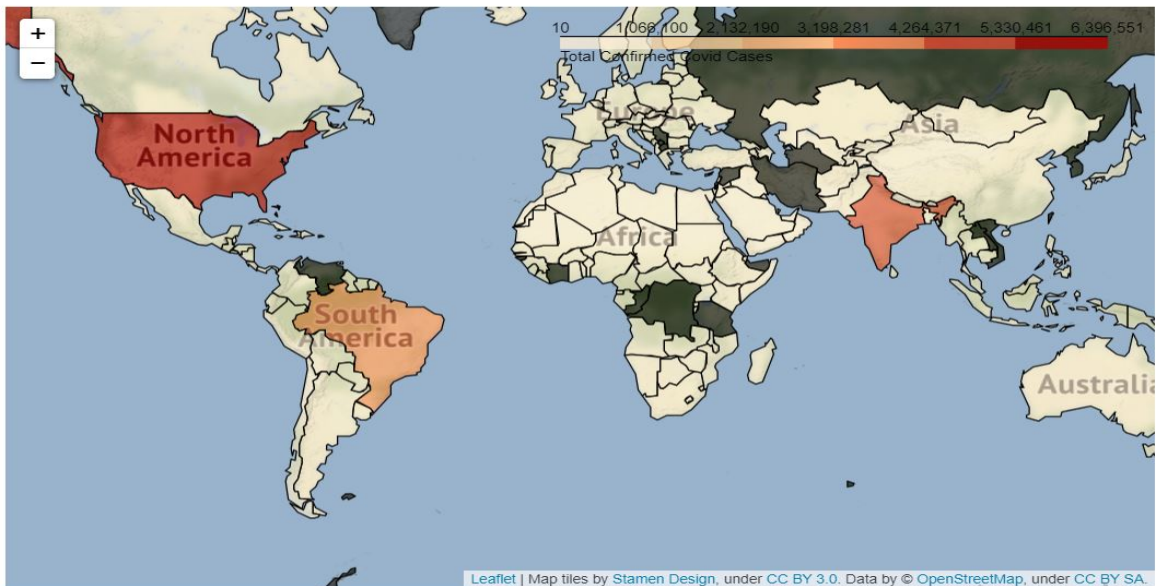
Base Map



Total Confirmed cases with circle markers



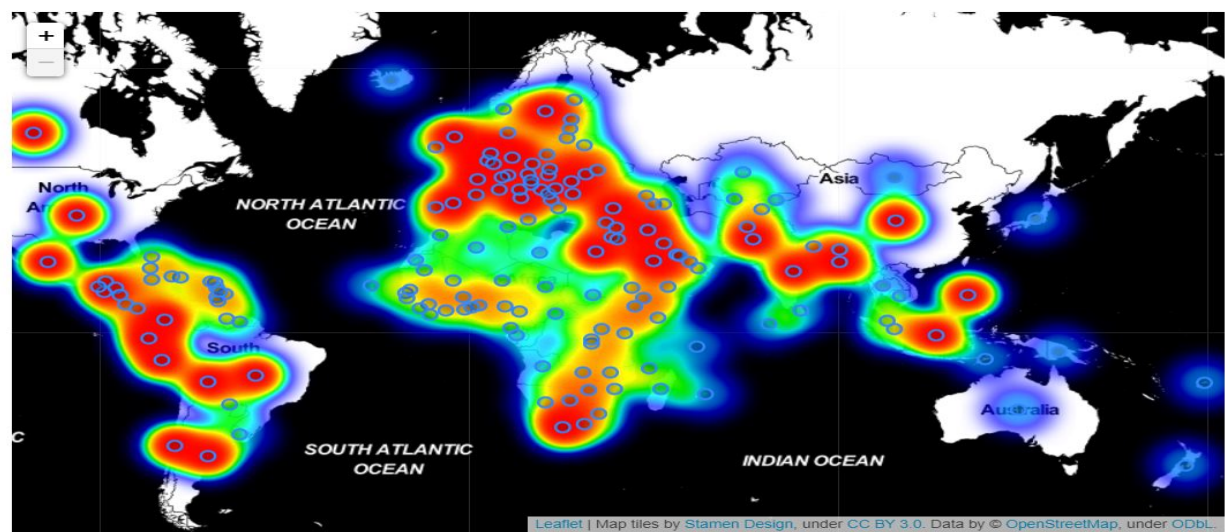
Total Confirmed cases



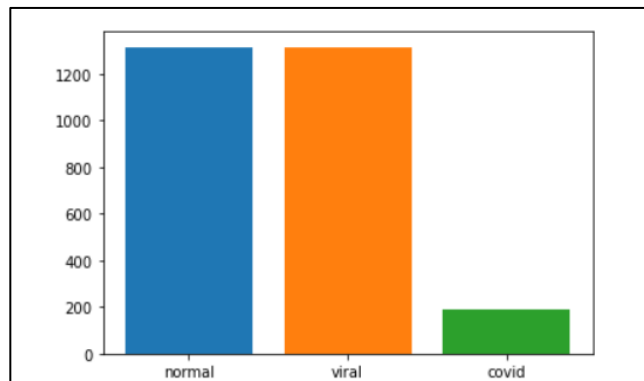
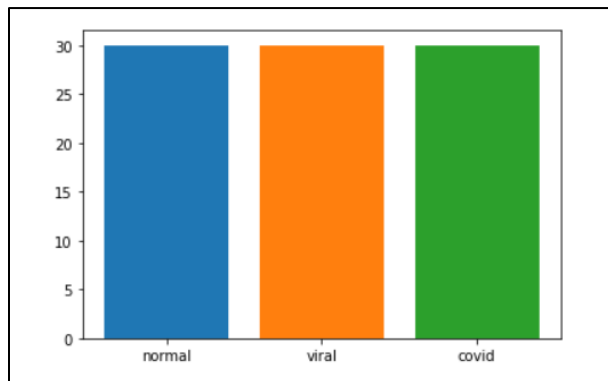
Base Map for Heat Map



Heat Map for Total confirmed cases



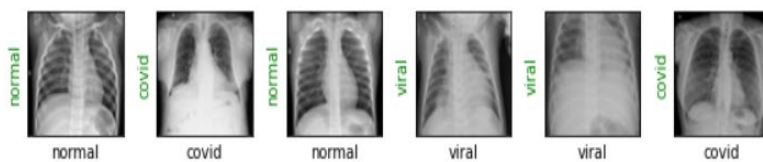
Matplotlib plots showing the training and test dataset



Classification Model Prediction Results

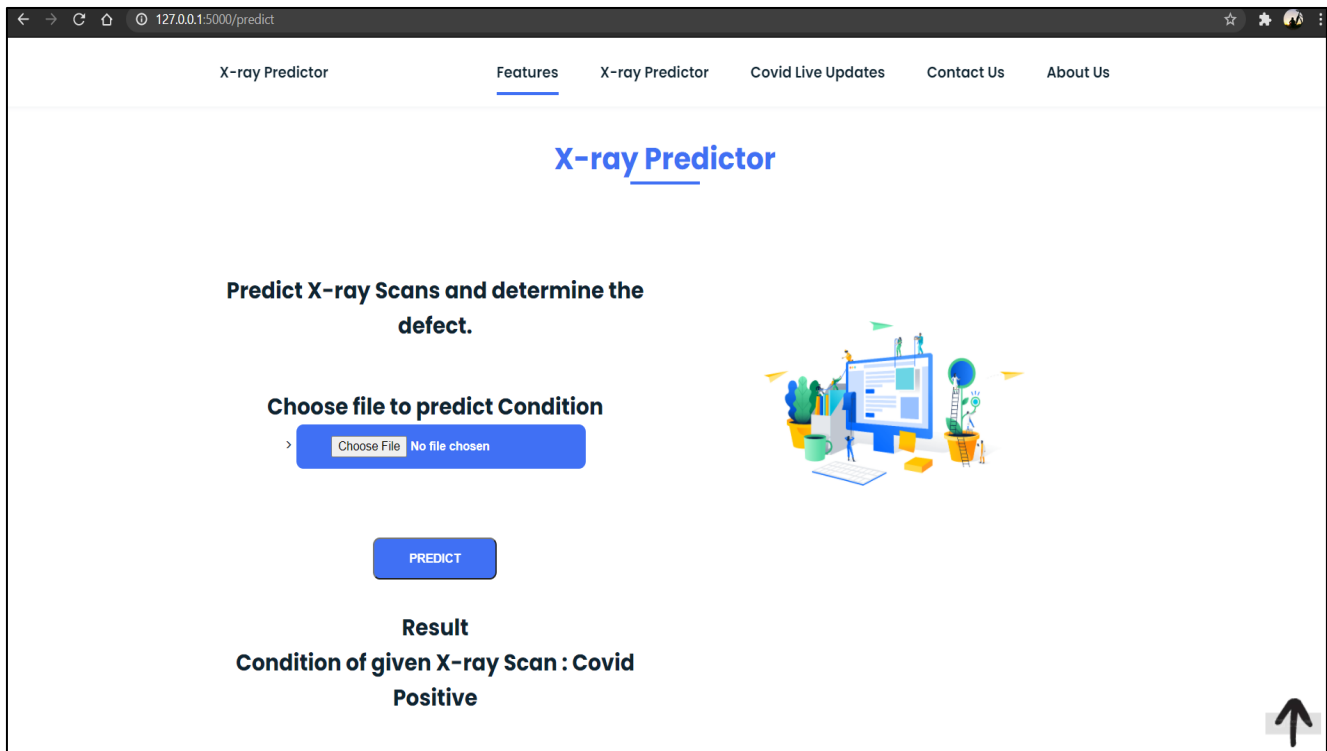
Final Result

```
In [16]: show_preds()
```



```
In [ ]:
```

Dashboard Output



The screenshot shows a web browser window with the address bar displaying "127.0.0.1:5000/predict". The page has a navigation bar with links: "X-ray Predictor", "Features", "X-ray Predictor", "Covid Live Updates", "Contact Us", and "About Us". The main heading is "X-ray Predictor". Below it, the text reads "Predict X-ray Scans and determine the defect." followed by "Choose file to predict Condition". There is a file upload button with a dropdown menu showing "Choose File" and "No file chosen". A "PREDICT" button is located below the file upload section. The result section displays "Result" and "Condition of given X-ray Scan : Covid Positive". An illustration of a computer monitor with various icons is positioned to the right of the file upload section. A small upward arrow icon is in the bottom right corner of the page.

X-ray Predictor

Features X-ray Predictor Covid Live Updates Contact Us About Us

X-ray Predictor

Predict X-ray Scans and determine the defect.

Choose file to predict Condition

> Choose File No file chosen

PREDICT

Result

Condition of given X-ray Scan : Covid Positive

Conclusion

- Thus this dashboard will help in easy geo visualisation of the cases around different regions of the world.
- The machine learning model helps classifying and identifying the respective condition of the patient.
- This will provide consulting advice to doctors and help in speedy treatment.

References

- Kaggle Dataset : <https://www.kaggle.com/tawsifurrahman/covid19-radiography-database>
- API : <https://covid19api.com/>
- Deep Residual Learning for Image Recognition by Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun, Microsoft Research
- **Resnets** by Pablo Ruiz – Harvard University – August 2018
- **PyTorch Documentation**
- **CNN – Documentation**
- Coronavirus disease (COVID-19) detection in Chest X-Ray images using majority voting based classifier ensemble by Tej Bahadur Chandra, Kesari Verma, Bikesh Kumar Singh, Deepak Jain, and Satyabhuwan Singh Netam