**NATURAL DISASTERS INTENSITY ANALYSIS AND CLASSIFICATION USING AI**

**FINAL DELIVERABLES**

**HTML,CSS CODE:**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>NATURAL DISASTERS</title>

<link rel="stylesheet" href="home.css" type="text/css">

<link rel="stylesheet" href="footer.css" type="text/css">

<link rel="stylesheet" href="style.css" type="text/css">

<link href='https://unpkg.com/boxicons@2.0.9/css/boxicons.min.css' rel='stylesheet'>

<style>

body {

margin: 2%;

justify-content: center;

}

.topnav {

overflow: hidden;

background-color:coral;

}

.topnav a {

float: right;

color: black;

text-align: center;

padding: 14px 16px;

text-decoration: none;

font-size: 17px;

}

.topnav a:hover {

color: white;

}

/\* Create a right-aligned (split) link inside the navigation bar \*/

.topnav a.split {

float: left;

text-decoration: none;

color: white;

font-weight: bolder;

font-size: 30px;

}

.flex{

max-width: 1170px;

margin: auto;

border-radius: 10px;

position: relative;

margin-left: 200px;

}

.flex-col1{

width: 400px;

height: 400px;

border-radius: 8px;

margin: 20px;

}

.flex-col2{

width: 400px;

height: 400px;

border-radius: 8px;

margin: 20px;

}

.flex-row{

display: flex;

flex-wrap: wrap;

}

.flex-row a img{

border-radius: 10px;

box-shadow: 7px 7px 7px 7px lightslategray;

}

.weather{

height: 400px;

}

@import url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;500;600;700&display=swap');

::selection{

color: #fff;

background: #43AFFC;

}

.content input{

width: 320px;

}

.weather{

display: flex;

align-items: center;

justify-content: center;

}

.wrapper{

width: 400px;

height: 300px;

background: #fff;

border-radius: 7px;

box-shadow: 7px 7px 20px rgba(0, 0, 0, 0.05);

box-shadow: 7px 7px 7px 7px skyblue;

}

.wrapper header{

display: flex;

font-size: 21px;

font-weight: 500;

color: #43AFFC;

padding: 16px 15px;

align-items: center;

border-bottom: 1px solid #ccc;

}

header i{

font-size: 0em;

cursor: pointer;

margin-right: 8px;

}

.wrapper.active header i{

margin-left: 5px;

font-size: 30px;

}

.wrapper .input-part{

margin: 20px 25px 30px;

}

.wrapper.active .input-part{

display: none;

}

.input-part .info-txt{

display: none;

font-size: 17px;

text-align: center;

padding: 12px 10px;

border-radius: 7px;

margin-bottom: 15px;

}

.input-part .info-txt.error{

color: #721c24;

display: block;

background: #f8d7da;

border: 1px solid #f5c6cb;

}

.input-part .info-txt.pending{

color: #0c5460;

display: block;

background: #d1ecf1;

border: 1px solid #bee5eb;

}

.input-part :where(input, button){

width: 100%;

height: 55px;

border: none;

outline: none;

font-size: 18px;

border-radius: 7px;

}

.input-part input{

text-align: center;

padding: 0 15px;

border: 1px solid #ccc;

}

.input-part input:is(:focus, :valid){

border: 2px solid #43AFFC;

}

.input-part input::placeholder{

color: #bfbfbf;

}

.input-part .separator{

height: 1px;

width: 100%;

margin: 25px 0;

background: #ccc;

position: relative;

display: flex;

align-items: center;

justify-content: center;

}

.separator::before{

content: "or";

color: #b3b3b3;

font-size: 19px;

padding: 0 15px;

background: #fff;

}

.input-part button{

color: #fff;

cursor: pointer;

background: #43AFFC;

transition: 0.3s ease;

}

.input-part button:hover{

background: #1d9ffc;

}

.wrapper .weather-part{

display: none;

margin: 30px 0 0;

align-items: center;

justify-content: center;

flex-direction: column;

}

.wrapper.active .weather-part{

display: flex;

}

.weather-part img{

max-width: 125px;

}

.weather-part .temp{

display: flex;

font-weight: 500;

font-size: 72px;

}

.weather-part .temp .numb{

font-weight: 600;

}

.weather-part .temp .deg{

font-size: 40px;

display: block;

margin: 10px 5px 0 0;

}

.weather-part .weather{

font-size: 21px;

text-align: center;

margin: -5px 20px 15px;

}

.weather-part .location{

display: flex;

font-size: 19px;

padding: 0 20px;

text-align: center;

margin-bottom: 30px;

align-items: flex-start;

}

.location i{

font-size: 22px;

margin: 4px 5px 0 0;

}

.weather-part .bottom-details{

display: flex;

width: 100%;

justify-content: space-between;

border-top: 1px solid #ccc;

}

.bottom-details .column{

display: flex;

width: 100%;

padding: 15px 0;

align-items: center;

justify-content: center;

}

.column i{

color: #5DBBFF;

font-size: 40px;

}

.column.humidity{

border-left: 1px solid #ccc;

}

.column .details{

margin-left: 3px;

}

.details .temp, .humidity span{

font-size: 18px;

font-weight: 500;

margin-top: -3px;

}

.details .temp .deg{

margin: 0;

font-size: 17px;

padding: 0 2px 0 1px;

}

.column .details p{

font-size: 14px;

margin-top: -6px;

}

.humidity i{

font-size: 37px;

}

.footer{

background-color: coral;

padding: 70px 0;

text-align: center;

}

.foo-container{

max-width: 1170px;

background-color: coral;

margin: auto;

}

ul{

list-style:none;

}

.footer-col{

width: 25%;

padding: 0 15px;

}

.footer-col h3{

font-size: 19px;

color: black;

text-transform: capitalize;

margin-bottom: 30px;

font-weight: 500;

position: relative;

}

.footer-col ul li:not(:last-child){

margin-bottom: 10px;

}

.footer-col ul li a{

font-size: 16px;

text-transform: capitalize;

color: #ffffff;

text-decoration: none;

font-weight: 300;

color:black;

display: block;

transition: all 0.3s ease;

}

.footer-col ul li a:hover{

color: #ffffff;

padding-left: 8px;

}

.footer-col .social-links a{

display: inline-block;

height: 40px;

width: 40px;

background-color: rgba(255, 255, 255, 0.2);

margin: 0 10px 10px 0;

text-align: center;

line-height: 40px;

border-radius: 50%;

color: black;

transition: all 0.5s ease;

}

@media(max-width:767px){

.footer-col{

width: 25%;

margin-bottom: 30px;

}

}

.footer-row{

display: flex;

flex-wrap: wrap;

}

</style>

</head>

<body>

<div class="topnav">

<a href="upload.html">Open Webcam</a>

<a href="introduction.html">Introduction</a>

<a href="home.html">Home</a>

<a href="#about" class="split">AI Based Natural disasters intensity analysis</a>

</div>

<br><br>

<div class="wrapper">

<header><i class='bx bx-left-arrow-alt'></i>Weather App</header>

<section class="input-part">

<p class="info-txt"></p>

<div class="content">

<input type="text" spellcheck="false" placeholder="Enter city name" required>

<div class="separator"></div>

<button>Get Device Location</button>

</div>

</section>

<section class="weather-part">

<img src="" alt="Weather Icon">

<div class="temp">

<span class="numb">\_</span>

<span class="deg">°</span>C

</div>

<div class="weather">\_ \_</div>

<div class="location">

<i class='bx bx-map'></i>

<span>\_, \_</span>

</div>

<div class="bottom-details">

<div class="column feels">

<i class='bx bxs-thermometer'></i>

<div class="details">

<div class="temp">

<span class="numb-2">\_</span>

<span class="deg">°</span>C

</div>

<p>Feels like</p>

</div>

</div>

<div class="column humidity">

<i class='bx bxs-droplet-half'></i>

<div class="details">

<span>\_</span>

<p>Humidity</p>

</div>

</div>

</div>

</section>

</div>

<script src="script.js"></script>

</div>

<div class="flex">

<div class="flex-row">

<div class="flex-col1">

<a href="wildfire.html"><img src="https://media.istockphoto.com/photos/forest-fire-wildfire-at-night-time-on-the-mountain-with-big-smoke-picture-id1266552048?b=1&k=20&m=1266552048&s=170667a&w=0&h=88DkS6W7nDa9rvrWo\_qe6GhgCF6HGgih7-B8WxHIodc=" width="400px" height="400px"></a>

</div>

<div class="flex-col2">

<a href="earthquake.html"><img src="https://media.istockphoto.com/photos/nepal-which-was-damaged-after-the-major-earthquake-on-25-april-2015-picture-id1282961362?b=1&k=20&m=1282961362&s=170667a&w=0&h=YpkYD68dQb4wJLOWm9rMpflnJqsYr6kINHsV8fTAK7o=" width="400px" height="400px"></a>

</div>

</div>

</div><br>

<div class="flex">

<div class="flex-row">

<div class="flex-col1">

<a href="flood.html"><img src="https://ak.picdn.net/shutterstock/videos/1055450345/thumb/1.jpg?ip=x480" width="400px" height="400px"></a>

</div>

<div class="flex-col2">

<a href="cyclone.html"><img src="https://c1.wallpaperflare.com/preview/68/855/818/winter-storm-hurricane-cyclone-typhoon.jpg" width="400px" height="400px"></a>

</div>

</div>

</div><br><br>

<footer class="footer">

<div class="foo-container">

<div class="footer-row">

<div class="footer-col">

<h3>company</h3>

<ul>

<li><a href="#">about us</a> </li>

<li><a href="#">our services</a> </li>

<li><a href="#">privacy policy</a> </li>

</ul>

</div>

<div class="footer-col">

<h3>follow us</h3>

<div class="social-links">

<a href="#"><i class="fa-brands fa-instagram"></i></a>

<a href="#"><i class="fa-brands fa-google"></i></a>

<a href="#"><i class="fa-brands fa-facebook-f"></i></a>

</div>

</div>

</div>

</div>

</footer>

<script>

const wrapper = document.querySelector(".wrapper"),

inputPart = document.querySelector(".input-part"),

infoTxt = inputPart.querySelector(".info-txt"),

inputField = inputPart.querySelector("input"),

locationBtn = inputPart.querySelector("button"),

weatherPart = wrapper.querySelector(".weather-part"),

wIcon = weatherPart.querySelector("img"),

arrowBack = wrapper.querySelector("header i");

let api;

inputField.addEventListener("keyup", e =>{

if(e.key == "Enter" && inputField.value != ""){

requestApi(inputField.value);

}

});

locationBtn.addEventListener("click", () =>{

if(navigator.geolocation){

navigator.geolocation.getCurrentPosition(onSuccess, onError);

}else{

alert("Your browser not support geolocation api");

}

});

function requestApi(city){

api = `https://api.openweathermap.org/data/2.5/weather?q=${city}&units=metric&appid=b7de0b69ae55dfbc30709d0f335bf65f`;

fetchData();

}

function onSuccess(position){

const {latitude, longitude} = position.coords;

api = `https://api.openweathermap.org/data/2.5/weather?lat=${latitude}&lon=${longitude}&units=metric&appid=b7de0b69ae55dfbc30709d0f335bf65f`;

fetchData();

}

function onError(error){

infoTxt.innerText = error.message;

infoTxt.classList.add("error");

}

function fetchData(){

infoTxt.innerText = "Getting weather details...";

infoTxt.classList.add("pending");

fetch(api).then(res => res.json()).then(result => weatherDetails(result)).catch(() =>{

infoTxt.innerText = "Something went wrong";

infoTxt.classList.replace("pending", "error");

});

}

function weatherDetails(info){

if(info.cod == "404"){

infoTxt.classList.replace("pending", "error");

infoTxt.innerText = `${inputField.value} isn't a valid city name`;

}else{

const city = info.name;

const country = info.sys.country;

const {description, id} = info.weather[0];

const {temp, feels\_like, humidity} = info.main;

if(id == 800){

wIcon.src = "icons/clear.svg";

}else if(id >= 200 && id <= 232){

wIcon.src = "icons/storm.svg";

}else if(id >= 600 && id <= 622){

wIcon.src = "icons/snow.svg";

}else if(id >= 701 && id <= 781){

wIcon.src = "icons/haze.svg";

}else if(id >= 801 && id <= 804){

wIcon.src = "icons/cloud.svg";

}else if((id >= 500 && id <= 531) || (id >= 300 && id <= 321)){

wIcon.src = "icons/rain.svg";

}

weatherPart.querySelector(".temp .numb").innerText = Math.floor(temp);

weatherPart.querySelector(".weather").innerText = description;

weatherPart.querySelector(".location span").innerText = `${city}, ${country}`;

weatherPart.querySelector(".temp .numb-2").innerText = Math.floor(feels\_like);

weatherPart.querySelector(".humidity span").innerText = `${humidity}%`;

infoTxt.classList.remove("pending", "error");

infoTxt.innerText = "";

inputField.value = "";

wrapper.classList.add("active");

}

}

arrowBack.addEventListener("click", ()=>{

wrapper.classList.remove("active");

});

</script>

</body>

</html>

**PYTHON CODE:**

**import warnings**

**warnings.filterwarnings("ignore")**

**import numpy as np**

**import matplotlib.pyplot as plt**

**import pandas as pd**

**from tensorflow.keras.models import Sequential,Model**

**from tensorflow.keras.layers import Dense,Activation,Dropout,Conv2D,Flatten,MaxPool2D,Reshape,GlobalAveragePooling2D,InputLayer**

**from tensorflow.keras.applications.resnet50 import preprocess\_input**

**from tensorflow.keras.preprocessing import image**

**from tensorflow.keras.preprocessing.image import ImageDataGenerator,load\_img,img\_to\_array**

**from tensorflow.keras.callbacks import EarlyStopping,ReduceLROnPlateau**

**IMAGE\_SIZE=[229,229]**

**train\_path='../input/natural-disaster-intensity/dataset/train\_set'**

**test\_path='../input/natural-disaster-intensity/dataset/test\_set'**

**train\_data\_gen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,zoom\_range=0.2,horizontal\_flip=True,validation\_split=0.30)**

**test\_data\_gen=ImageDataGenerator(rescale=1./255,validation\_split=0.30)**

**training\_set=train\_data\_gen.flow\_from\_directory(train\_path,target\_size=(229,229),batch\_size=100,class\_mode='categorical',shuffle=True,color\_mode='rgb',subset='training')**

**testing\_set=test\_data\_gen.flow\_from\_directory(test\_path,target\_size=(229,229),batch\_size=100,class\_mode='categorical',shuffle=True,color\_mode='rgb',subset='validation')**

**model=Sequential()**

**model.add(Conv2D(32,(3,3),input\_shape=(229,229,3),activation='relu'))**

**model.add(MaxPool2D(pool\_size=(2,2)))**

**model.add(Conv2D(32,(3,3),activation='relu'))**

**model.add(MaxPool2D(pool\_size=(2,2)))**

**model.add(Flatten())**

**model.add(Dense(units=128,activation='relu'))**

**model.add(Dense(units=4,activation='softmax'))**

**model.summary()**

**model.compile(optimizer='adam',loss='categorical\_crossentropy',metrics=['accuracy'])**

**model.fit\_generator(**

**generator=training\_set,steps\_per\_epoch=len(training\_set),**

**epochs=20,validation\_data=testing\_set,validation\_steps=len(testing\_set))**

**model.save("nature.h5")**

**from tensorflow.keras.models import load\_model**

**from keras.preprocessing import image**

**model = load\_model("nature.h5")**

**a = ['Cyclone','Earthquake','Flood','Wildfire']**

**img=image.load\_img('../input/natural-disaster-intensity/dataset/test\_set/Cyclone/876.jpg',**

**target\_size=(229,229))**

**x=image.img\_to\_array(img)**

**x=np.expand\_dims(x,axis=0)**

**pred=np.argmax(model.predict(x))**

**a[pred]**

**testing\_set.class\_indices**