NEW HORIZON
COLLEGE OF ENGINEERING
Autonomous College, Affiliated to VTU | Approved by AICTE New Delhi & UGC
Accredited by NAAC with 'A' Grade & Accredited by NBA

Golden Jubilee
YEARS OF EXCELLENCE IN EDUCATION

A MINI PROJECT

REPORT

for

Mini Project in Web Frame Works or Operating System (20CSE68)

WebMed

Submitted by

RAHUL MUSALIYATH DINESH
USN: 1NH18CS738,
Semester-Section: 6-D

In partial fulfillment for the award of
the degree of

Bachelor of Engineering

in

COMPUTER SCIENCE AND ENGINEERING

NEW HORIZON COLLEGE OF ENGINEERING

Autonomous College, Affiliated to VTU | Approved by AICTE New Delhi & UGC
Accredited by NAAC with 'A' Grade & Accredited by NBA

Certificate

This is to certify that the mini project work titled

WebMed

Submitted in partial fulfillment of the degree of
Bachelor of Engineering in
Computer Science and Engineering by

RAHUL MUSALIYATH DINESH
USN: 1NH18CS738

DURING
EVEN SEMESTER 2020-2021
for
COURSE CODE: 20CSE68

Signature of Reviewer Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner	*Signature with date*
1. _____	_____
2. _____	_____

738

ORIGINALITY REPORT

1%	%	1%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	"Advanced Topics in Intelligent Information and Database Systems", Springer Science and Business Media LLC, 2017 Publication	<1%
2	Robert E. Beasley. "Essential ASP.NET Web Forms Development", Springer Science and Business Media LLC, 2020 Publication	<1%
3	William Panek. "MCSA Windows Server 2016 Complete Study Guide", Wiley, 2018 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

ABSTRACT

Amidst the ongoing pandemic, it has been widely reported across the world that the pricing for various tests have gone up. Lab tests are generally expensive, especially if performed in the private sector, but there has been an exponential increase in the same since the onset of the pandemic.

While the upper and middle class strata of our society may have the financial capabilities to access these life-saving tests, the same is not the case with the lower strata of our society, the downtrodden, who have suffered the most since the global epidemic began.

The existing public health infrastructure funded the by the government, which the poor were highly dependent on, has not been able to bear the brunt of the pandemic's magnitude. This has made these vital tests even more inaccessible to many as the existing infrastructure is focused on mitigating the severity of the pandemic.

The advancements made in the field of Artificial Intelligence and Machine Learning, have enabled many of these life-saving tests to be performed at a much lower cost and greater accuracy. Prediction models built using AI and ML algorithms can be leveraged to make these tests much more accessible to a larger section of the society.

The onset of Web 2.0 has enabled the development of various websites that are user-friendly and dynamic and that can be accessed at any time, from any device and any place.

This aim of this mini-project is to leverage the power of AI and ML to develop prediction models for detecting various diseases through the development of a website built using Web 2.0 technologies like HTML, CSS and JavaScript.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal, New Horizon College of Engineering, for his constant support and encouragement.

I am grateful to **Dr. Amarjeet Singh,** Dean - Academics, for his unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank **Dr. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I also express my gratitude to **Ms. Uma N.**, Senior Assistant Professor, Department of Computer Science and Engineering, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

RAHUL MUSALIYATH DINESH

USN: 1NH18CS738

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

The price of several tests in India for detecting various diseases like lung cancer, brain tumors, skin diseases, pneumonia, diabetic retinopathy and breast cancer is very high. The cost for diagnosing such diseases is therefore expensive, which makes it nearly impossible for a large percent of our population to get diagnosed timely.

With the pandemic showing no signs of slowing down, the existing government funded public health infrastructure is under severe stress and therefore the poor which were completely reliant on it before the pandemic, are now finding it difficult to get lab tests and timely diagnosis as the prices for the same in the private sector is unaffordable.

1.2 OBJECTIVES

Once developed, the dynamic website will provide services to detect various diseases like

- Lung Cancer
- Brain Tumours
- Skin Disease
- Pneumonia
- Diabetic Retinopathy
- Breast Cancer

using prediction models that will be built using Python's various Machine Learning algorithms. The website shall be user friendly, interactive and informative. It will be built using various web technologies like HTML, CSS and JavaScript. It will leverage the power of AI and ML to detect the presence of the above-mentioned diseases.

1.3 METHODOLOGY TO BE FOLLOWED

The mini-project aims to build a website that would be dynamic, interactive and user-friendly. For this purpose, various Web 2.0 technologies like HTML 5.0, CSS and JavaScript would be used. Additionally, the Python Language will be used to develop prediction models that would be used to detect the previously mentioned diseases. These models will be built using Python's various AI and ML libraries.

1.4 EXPECTED OUTCOMES

The mini-project is expected to develop a dynamic website that shall be responsive and interactive. The website shall contain navigation links to separate pages that will cater to each disease. The user will be able to choose the diagnosis they require, following which they will be required to upload a scan image. The prediction model will then be used to predict an appropriate result.

CHAPTER 2

FUNDAMENTALS OF WEB FRAMEWORK TECHNOLOGIES

2.1 INTRODUCTION TO THE INTERNET

The Department of Defense of the United States, in the 1960s wanted to develop a new large-scale network of computers. The main intention of this network of computers were program sharing, communications and remote access to computers for researchers.

The Advanced Research Projects Agency or commonly abbreviated as ARPA of the USG's DoD funded the making of the first such network. It connected many ARPA funded universities and several research labs. Since this network was sanctioned by ARPA, hence it was called the ARPAnet. The initial use of the ARPAnet was simple text-based communications via the electronic mail/e-mail.

Other networks like the BITNET (Because Its Time Network) and CSNET (Computer Science Network) were created in the 1970s and 1980s.

After a few years, the NSFnet was created, which was a new national network funded by the NSF or National Science Foundation, initially for scientific research purpose. The NSFnet had replaced ARPAnet by the 1990s for most nonmilitary uses like in universities and research labs. Approximately 1 million computers across the world were connected together by the NSFnet. A small part of the NSFnet returned to being a just a network for research purposes in 1995. The rest became what we today refer to as the Internet.

"The Internet is a huge collection of computers connected in a communication network."

These computers can be of any variable size, manufacturer and configuration . In fact, some of the devices like plotters and printers which are also connected to the internet are not computers at all. The innovation that allows all these physically different devices to interact and communicate with each other is a single, low-level protocol called the TCP/IP or the Transmission Control Protocol / Internet Protocol. It is used to directly to allow a program

on one computer or device to communicate with a program on another computer or device via the Internet.

Normally the individual computers in an organization are connected to each other in a local network rather than connecting every computer or device on the Internet directly to every other computer or device on the Internet. One node (i.e., a computer/device) on this local network is physically connected to the Internet. Hence, the Internet is in fact a network of networks, instead of a network of computers.

2.2 WORLD WIDE WEB

"*World Wide Web, which is also known as a Web, is a collection of websites or web pages stored in web servers and connected to local computers through the internet.*"

These websites contain digital images, text pages, videos, audios etc. These sites can be accessed by users from any part of the world, from any country over the internet using their devices like computers, cell phones, laptops etc

The internet along with the World Wide Web, enables the retrieval and display of media and text to your device. The initial aim of the WWW, was to enable scientists around the world to use the Internet to exchange documents that describe their work. It was designed and developed to allow a user anywhere on the Internet, from any place to retrieve and search for documents from databases on computer connected to the Internet.

2.3 WEB BROWSERS

When computers communicate over some network, in many cases one performs as a client while the other as plays the role of a server. The communication is initiated by the client, which is generally a request for information stored, that is typically stored on a server. The server then sends the requested information back to the client.

Browsers request the documents that are provided by the servers. Browsers are programs running on client devices. They are termed as browsers as they allow the user or users to browse the resources which are available on the servers.

A browser is in fact a client on the Web as it initiates the communication with a server. The server waits for a request from the client before doing anything. In the elementary case, a browser (i.e., the client) requests a static document from a server. The requested document is located among its servable documents and sends it to the browser, which then displays it for the user.

The most common protocol used by the Web is the HTTP. HTTP provides a standard form of communication between Web servers and browsers. Examples: Google Chrome, Microsoft Edge, Opera, Mozilla Firefox etc.

2.4 OPERATION OF WWW

1. User enters the URL (say, http://www.newhorizonindia.edu) of the web page in the address bar of web browser.
2. Then client (browser) requests the Domain Name Server for the IP address corresponding to www.newhorizonindia.edu.
3. After receiving IP address, browser sends the request for web page to the web server using HTTP protocol.
4. Then web server then searches/locates the requested web page. If found, it returns it back to the web browser and closes the HTTP connection.
5. After the web browser receives the web page, it interprets it and display the contents of web page in web browser's window.

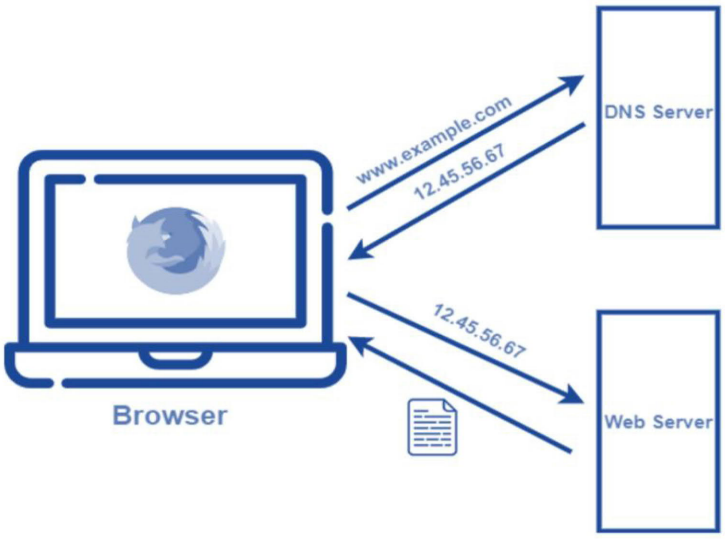


Figure 2.1: Operation of the World Wide Web

2.5 WEB 2.0

Web 2.0 can be generally described as a term for such internet applications which allow sharing and collaboration opportunities to users and help them express themselves online. It's an enhanced version of the first worldwide web, characterized by the change from static to responsive and dynamic or user-generated content.

The idea behind Web 2.0 refers to rich web applications, social web and web-oriented architecture. It refers to modifications in the way web pages are used and designed by the users, without any change in the technical aspects. One of the main attributes of Web 2.0 is that users are encouraged to contribute content, instead of just reading what is already there (i.e., the static content).

By rich user experience we mean dynamic, interactive content (for example, a user can click on an image of a planet to get more information about that planet — i.e., size, conditions, distance from the sun, and more).

User participation refers to the flow of information between the owner of a certain website and the users (for example, the online encyclopaedia, Wikipedia allows any user to create new pages and edit existing pages to ensure up to date information)

Advantages of Web 2.0:

- It is available at any place and at any time
- The variety of media is huge
- It is user friendly and easy to use
- Knowledge building happens both ways, i.e., between the users and site owners
- Enables the creation of dynamic learning communities
- Every user is the editor and the author, edits can be made and tracked.
- Modifications are seen immediately, thus giving researches more sources.

Five major features of Web 2.0

- Information is sorted freely, thus enabling users to classify and retrieve the information collectively.
- The content is dynamic and responsive to the users' inputs.
- There is a constant flow of information between the site users and owners via means of evaluations, online commenting and feedback.
- Enables the development of APIs that can be used by site creators or by software applications
- The scope or audience of the internet expands as more people are benefitted

2.6 HTML

Hyper Text Markup Language or abbreviated as HTML is a formatting / markup language used to describe the look and contents of a website or webpage. It is used to organize text, audio, graphics, video etc. on a web page. The document type declaration: <!DOCTYPE html>must be used at the beginning of every HTML document.

The <!DOCTYPE> declaration describes the document type. It also helps browsers to display web pages accurately. The HTML document begins with <html> start tag and ends with </html> ending tag. The <body> and </body> tags enclose the visible part of the HTML document. The web page's title is stated between the <title> and </title> tags which is itself enclosed in the <head> and </head> tags.

2.7 HTML TAGS

Tag is a keyword that instructs the web browser on how to show the text, graphics, audio or video on a web page.

Key Points:

- Tags are referred with a pair of angle brackets (<, >).
- They begin with a less-than (<) character and end with a greater-than (>) character.

- The name of the tag is given between the angle brackets.
- Generally, most of the tags appear as a pair: the starting tag and the closing tag
- The start tag is just the name of the tag, enclosed between the angle bracket. The closing tag is similar except that it has a forward slash (/) before the tag name.
- Empty tags are those tags that don't have/require the closing tag.
- Tags are case insensitive.
- The starting and closing tag name must be the identical. For example, <h1> hello is invalid as both tag names are not the same.
- If the tag name is not enclosed between the angle characters, the browser will simply recognize them as plain text.
- Attributes can also be mentioned in the tags to provide additional information to the browser.

Table 2.1: Commonly used HTML Tags

Tags	Use / Description
** **	HTML headings are defined with the <h1> to <h6> tags. <h1> defines the most important heading. <h6> defines the least important heading.
<pre> </pre>	Used to define preformatted text. The browsers render the enclosed text with white spaces and line breaks.
** **	Sentences and paragraphs are enclosed in the tag
**<a> **	Syntax: link text Example: Click here to go to Google! • a stands for anchor and href is an attribute By default, the linked page will be displayed in the current browser window. To change this, you must specify another target for the link. **Target attribute:** specifies where to open the linked document.

	The target attribute can have one of the **following values**: • _self - Default. Opens the document in the same window/tab as it was clicked • _blank - Opens the document in a new window or tab • _parent - Opens the document in the parent frame • _top - Opens the document in the full body of the window Ex: Click here to go to Google! **Absolute vs Relative URL:** A local link (a link to a page within the same website) is specified with a relative URL (without the "https://www" part) Ex: CSS Tutorial
<img> </img>	Attributes: src (source), alt (alternative text), width and height Example: <img src="mypic.png" alt="Rahul M D" height="50" width = "100">
** **	Specifies the text as bold. Eg. this is bold text, bold text, without any extra importance i.e., without any semantic emphasis
** **	It is a phrase tag. It specifies an important text. Eg. this is strong text, strong importance i.e., with semantic emphasis. The content inside is typically displayed in bold
** **	The content of italic tag is displayed in italic. Eg. Italic text
****	Specifies the subscripted text. Eg. X_1
****	Defines the superscripted text. Eg. X^2
<ins> </ins>	Used to specify a block of inserted text. GFG is a <del> mathematical </del> <ins> computer </ins> science portal **Output**: GFG is a mathematical computer science portal
<del> </del> or <s> </s>	Strikethrough

<mark> **</mark>**	Highlight: My name is Rahul Example: Hello my name is <mark> Rahul </mark>
<u> </u>	Underline
** **	Inserts a newline / single line break
<hr>	Inserts a horizontal line. Used to separate two elements or content

2.8 XHTML

E**X**tensible **H**yper**T**ext **M**arkup **L**anguage commonly abbreviated as XHTML is the next advancement in the development of the internet. In the XHTML family, the XHTML 1.0 is the first document type. XHTML and HTML 4.0 have very few differences and are almost similar. XHTML is a stricter and cleaner form of HTML 4.0.

A basic XHTML document consists of the following main parts:

- The *DOCTYPE* (DTD - Document Type Definition)
- *html* element - which is document's root
- *xmlns* (XML namespace) attribute for the *html* element
- *head* element with a child *title* element to represent the website's title
- *body* element where the main content that is displayed to the users is written

2.9 CSS

Cascading Style Sheets or abbreviated as CSS is used in almost all webpages and user interfaces written using HTML. It is basically a style sheet language that provides styling and formatting to documents written in a markup language like HTML or XML.

CSS is essential as it allows web developers, designers, bloggers, content creators etc.to make websites look unique and attractive. CSS provides the freedom to play with a page layout, adjust colors and fonts (size, font family, text decoration), add effects to images, etc. While in the yesteryears, it was feasible to add style using HTML or an earlier version of CSS, the latest version of CSS, i.e., the CSS3 has really broadened up the scope of styling and allows for more creativity.

The term cascading in CSS implies that a style used on a parent element will also apply to all child elements within the parent. Hence, if you change the color of the body text to "orange", then the same colour will also be applied on all the elements like paragraphs, headings, etc. within the body.

CSS can be applied to HTML documents in three ways:

- Inline - using the style attribute within the HTML elements
- Internal - using a <style> element in the <head> section of the document
- External - using a <link> element to link to an external CSS file

The most common way to add CSS is the third way, i.e., to keep the styles in an external CSS files and link it to the HTML document.

2.10 JAVASCRIPT

JavaScript is a scripting language that is used to make and regulate dynamic content on a website, i.e., anything that refreshes, moves, or changes on the screen without requiring the user to manually reload a web page. JavaScript provides features like:

- animated graphics based on user inputs
- autocomplete text suggestions
- interactive and responsive forms
- photo slideshows that change automatically

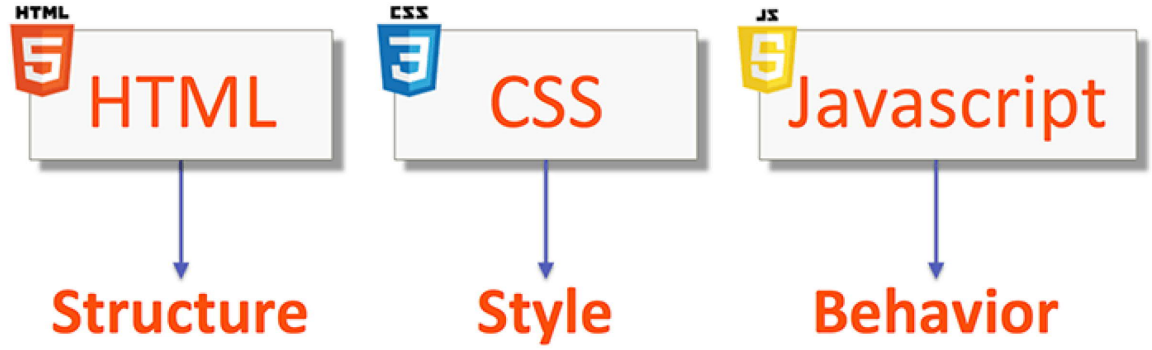
JavaScript is mainly used for applications like:

- **Increasing or adding interactivity and responsiveness to websites**—JavaScript enables static websites to have dynamic and interactive content that are responsive to the users' actions.

- **Designing mobile applications**—JavaScript is also used for developing various apps that we use on our mobiles and tablets.

- **Developing browser based games**—JavaScript is also used for making fun and interactive games that can be played directly on the browser.

- **Development of back-end applications**—Though JavaScript is used mainly for front-end development, it is also used occasionally for developing back-end infrastructure as well.



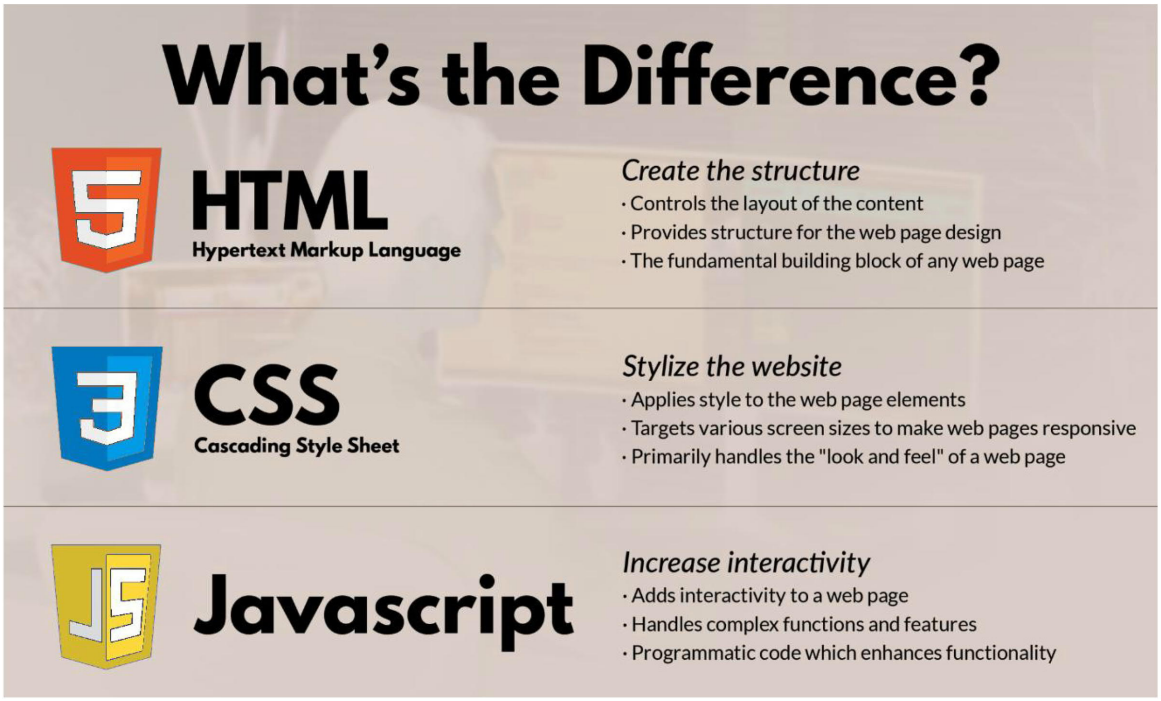


Figure 2.2: Basic components of a Web Page

CHAPTER 3

REQUIREMENT SPECIFICATION

3.1 HARDWARE REQUIREMENTS

The following are the hardware specifications required for the implementation of this miniproject:

- Processor : Intel 386 or higher
- RAM : 4 MB or higher
- Hard Disk : 25 MB or higher
- Input device : Standard Keyboard and Mouse
- Output device : VGA and High-Resolution Monitor or higher

3.2 SOFTWARE REQUIREMENTS

The following are the software specifications required for the implementation of this miniproject:

- Operating system : Microsoft Windows 7, Microsoft Windows Vista or later
- Python version : 3.8
- IDE : Microsoft Visual Studio Code
- Browser: Google Chrome
- HTML version: HTML 5.0
- Web Technologies: HTML 5.0, CSS, JavaScript, JQuery, PHP
- Web Service: XAMPP Apache
- Web Framework: Flask

CHAPTER 4

DESIGN

4.1 PREDICTION MODEL

For implementing the machine learning part of the mini-project to predict lung cancer and pneumonia, convolutional neural network (CNN) algorithms were used to construct a model that is developed from a data set of test and train images.

For this, Python's Keras and Tensorflow packages were required which are commonly used for machine learning and deep learning projects. The models developed using this process were used for the prediction of lung cancer and pneumonia in this mini-project.

The initial step is to collect and organize the data set which is used to develop the prediction model. In both diseases like pneumonia, lung cancer, breast cancer and brain tumour we have two types of classifications: normal and abnormal. In other diseases like diabetic retinopathy and skin diseases, we have many classifications like type of skin disease or stage of the disease as in diabetic retinopathy.

These classifications are identified. Then we normalize the dataset and define the CNN by constructing the various layers using a series of built-in functions which are provided by the Keras and Tensorflow libraries. The resulting model at the end of the process is saved as a (.h5) file and is used in the mini-project.

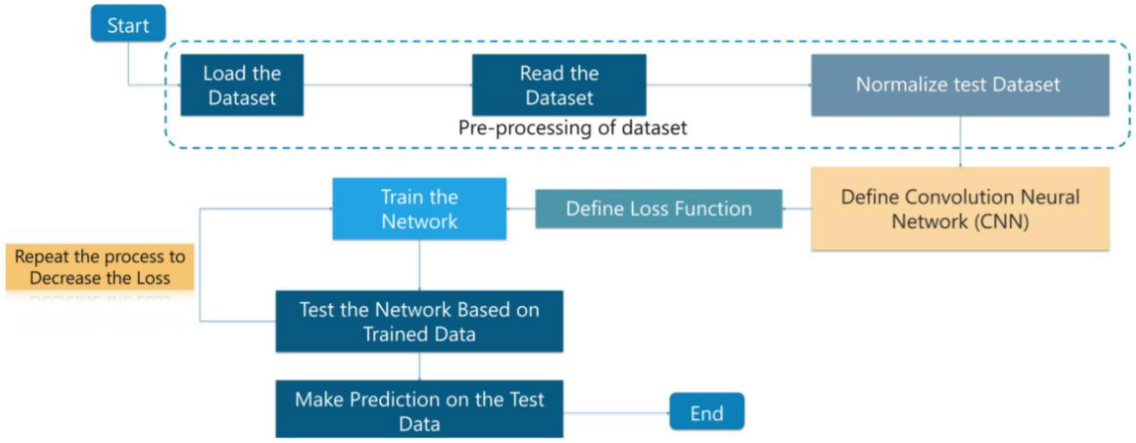


Figure 4.1: Process of making a Prediction Model

4.2 WEBSITE STRUCTURE

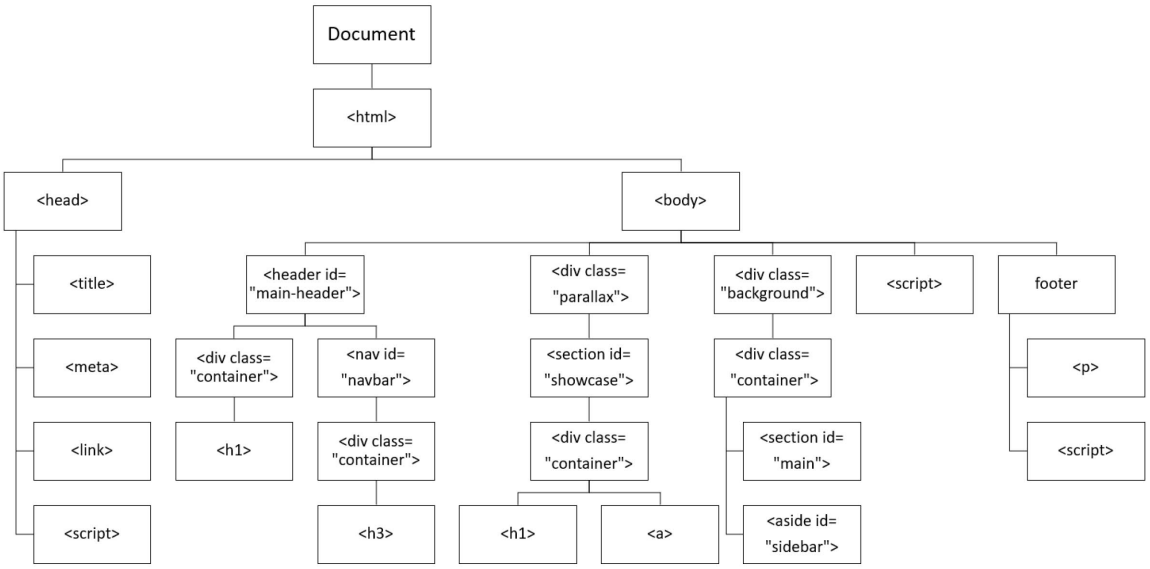


Figure 4.2: Document Object Model

When a web page is loaded, the browser creates a Document Object Model of the page. The HTML DOM model is constructed as a tree of Objects.

The Document Object Model (DOM) is a cross-platform and language-independent interface that treats an XML or HTML document as a tree structure wherein each node is an object representing a part of the document. The DOM represents a document with a logical tree.

Figure 3.2 shows the Document Object Model tree structure of one of the webpages which were designed during the development of this miniproject.

Each branch of the tree ends in a node, and each node contains objects. DOM methods allow programmatic access to the tree; with them one can change the structure, style or content of a document. Nodes can have event handlers attached to them. Once an event is triggered, the event handlers get executed.

The DOM is a W3C (World Wide Web Consortium) standard. The DOM defines a standard for accessing documents:

"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."

With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

The HTML DOM is a standard object model and programming interface for HTML. It defines:

- The HTML elements as objects
- The properties of all HTML elements
- The methods to access all HTML elements
- The events for all HTML elements

In other words: The HTML DOM is a standard for how to get, change, add, or delete HTML elements.

CHAPTER 5

IMPLEMENTATION

5.1 WEBPAGES

The webpage which is used for the front end of the miniproject was built entirely based on web technologies like HTML, CSS, JavaScript, PHP and JQuery.

The front end consists of 3 main pages:

- The Home page
- The Prediction Models Page
- The Contact Page

Additionally, each Prediction Model has its own page, namely:

- Lung Cancer Prediction Model page
- Skin Diseases Prediction Model page
- Brain Tumour Prediction Model page
- Pneumonia Prediction Model page
- Diabetic Retinopathy Prediction Model page
- Breast Cancer Prediction Model page

Home Page

The home page is the gateway to the entire website. The home page broadly consists of the following components:

- Navigation Bar
    - The navigation bar is common to all pages in the website.
    - It contains links to the main pages of the website mentioned previously

- Sample code:

```
<nav>
  <div>
    <a href="/">WebMed</a>
  </div>
  <div>
    <ul>
      <li><a href="/">Home</a></li>
      <li><a href="/models">Prediction Models</a></li>
      <li><a href="/contact">Contact</a></li>
    </ul>
  </div>
</nav>
```

- Catchy phrase display area
  - A catchy phrase, “Leveraging the power of AI and Deep Learning to increase early diagnosis of diseases!” is displayed here with a suitable background picture.
- Prediction Models Slider
  - A slider which can accommodate 3 Cards at a time (for normal window size).
  - Each card displays the information about each prediction model and links to their respective pages.
  - The slider moves automatically after a small time period but can also be manually moved using a left and right arrow present on each side of the slider.

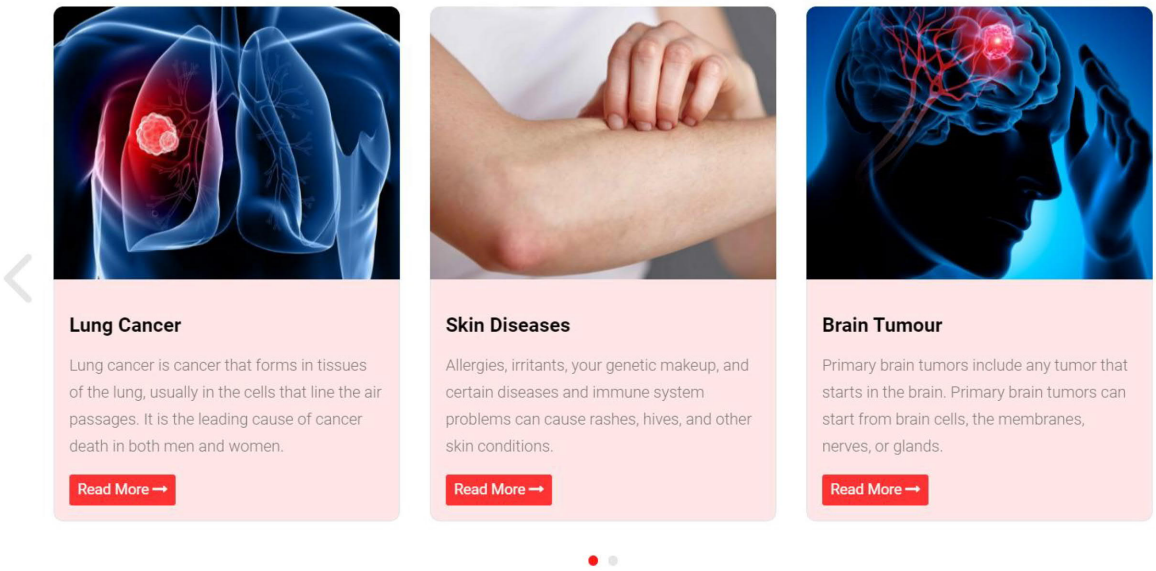


Figure 5.1: Prediction Model Slider

- About Us, Man behind it, Vision
    - About Us: It has a brief description about the website and its purpose
    - Man behind it: It contains a brief paragraph about the person who developed the website (i.e., the author of this report)
    - Vision: It has a brief description about the goals and objectives of the website.
- Motto display bar
    - A full-length bar that displays the motto, "WebMed: Making healthcare more accessible!"
- Footer
    - Contains the imaginary copyright notice of the WebMed brand, name of the designer and developer of the website along with relevant details.

Prediction Models Page

The Prediction Models page contains the individual boxes or containers for each disease prediction model. It is arranged in the form of a grid and each box contains a link to their respective webpage.

Contact Page

- The contact page contains 3 text-fields for name, email and comment. It allows users of the website to give suggestions, ask queries directly to the designer-developer of the website.
- This page runs on the XAMPP Apache server, unlike the other pages which run on the Flask server.
- The form is collected using jQuery and PHP.
- The jQuery file provides validation to the deat
- The PHP file sends the message containing the name, email and comment to the designer-developer's email so that further action can be taken by the designer-developer.

5.2 CONTACT FORM

The contact form present in the Contact page is used for collecting feedbacks and suggestions from the website users. The user's name, email and suggestions/comments are sent to the designer-developer's email for further action.

This functionality is implemented using jQuery and PHP. When the submit button is clicked, the jQuery file provides validation and sends the data, if no errors are found to the PHP file who sends the content as an email to the designer-developer's email.

The following is the code for the jQuery file:

```
jQuery(document).ready(function($) {

  //if submit button is clicked
  $('#submit').click(function() {

    //Get the data from all the fields
    var name = $('input[name=name]');
    var email = $('input[name=email]');
    var regx = /^([a-z0-9_\-\.])+\@([a-z0-9_\-\.])+\.([a-z]{2,4})$/i;
```

```
var comment = $('textarea[name=comment]');
var returnError = false;
//Simple validation to make sure user entered something
//Add your own error checking here with JS,
//but also do some error checking with PHP.
//If error found, add hightlight class to the text field

if (name.val() == '') {
    name.addClass('error');
    returnError = true;
} else name.removeClass('error');

if (email.val() == '') {
    email.addClass('error');
    returnError = true;
} else email.removeClass('error');

if (!regx.test(email.val())) {
    email.addClass('error');
    returnError = true;
} else email.removeClass('error');


if (comment.val() == '') {
    comment.addClass('error');
    returnError = true;
} else comment.removeClass('error');

// Highlight all error fields, then quit.
if (returnError == true) {
    return false;
```

```
}

//organize the data
var data = 'name=' + name.val() +
    '&email=' + email.val() +
    '&comment=' + encodeURIComponent(comment.val());

//disabled all the text fields
$('.contact-text').attr('disabled', 'true');

//show the loading sign
$('.loading').show();

//start the ajax
$.ajax({
    //this is the php file that processes the data and sends email
    url: "contact.php",

    //GET method is used
    type: "GET",

    //pass the data
    data: data,

    //Do not cache the page
    cache: false,

    //success
    success: function(html) {
        //if contact.php returned 1/true (send mail success)
        console.log(html);
```

```
            if (html == 1) {

                //show the success message
                $('.done').fadeIn('slow');

                $(".form").find('input[type=text], textarea').val("");

                //if contact.php returned 0/false (send mail failed)
            } else alert('Sorry, unexpected error. Please try again later.');
        }
    });

    //cancel the submit button default behaviours
    return false;
});

$('.close').click(function() {
    location.reload();
});
});
```

The PHP file which is used to send the email is coded as follows:

```
<?php

//Retrieve form data.
//GET - user submitted data using AJAX
$name = ($_GET['name']) ? $_GET['name'] : $_POST['name'];
$email = ($_GET['email']) ?$_GET['email'] : $_POST['email'];
$comment = ($_GET['comment']) ?$_GET['comment'] : $_POST['comment'];

$errors = false;
```

```
//if the errors array is empty, send the mail
if (!$errors) {
        //recipient - replace your email here
        $to = 'rahul.1nh18cs738.cse@gmail.com';
        //sender - from the form
        $from = 'WebMed Contact Form' . ' <' . $email . '>';

        //subject and the html message
        $subject = 'New Message from ' . $name;

        $message = 'Name: ' . $name . '<br/><br/>
                Email: ' . $email . '<br/><br/>
                Message: ' . nl2br($comment) . '<br/>';

        //send the mail
        $result = sendmail($to, $subject, $message, $from);

        echo $result;
}

//Simple mail function with HTML header
function sendmail($to, $subject, $message, $from) {
        $headers = "MIME-Version: 1.0" . "\r\n";
        $headers .= "Content-type:text/html;charset=iso-8859-1" . "\r\n";
        $headers .= 'From: ' . $from . "\r\n";

        $result = mail($to,$subject,$message,$headers);
        if ($result) return 1;
        else return 0;
}
?>
```

5.3 RUNNING THE FLASK APP

All the webpages in this miniproject run on the Flask server except for the Contact page (which runs on XAMPP Apache). The python file 'app.py', which starts the Flask server is executed.

The file also loads the various prediction models and is used to display the appropriate message after the user uploads the image to be checked. The code of the 'app.py' file is as shown: (shows the code to be executed once the user uploads the image only for Lung Cancer prediction model)

```
#Import all necessary libraries


app = Flask(__name__)
model_lc = load_model("lungcancer.h5")
model_pn = load_model("pneumonia.h5")
model_dr = load_model("diabeticretinopathy.h5")
model_bt = load_model("braintumour.h5")
model_bc = load_model("breastcancer.h5")
model_sd = load_model("skindiseases.h5")

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

@app.route('/models')
def model():
    return render_template(r'models.html')

@app.route('/contact')
def contact():
    return redirect("https://localhost/WebMed/contact.html", code=302)
```

```
@app.route('/models/lungcancer')
def lc():
    return render_template(r'lungcancer.html')

@app.route('/models/pneumonia')
def pn():
    return render_template('pneumonia.html')

@app.route('/models/diabeticretinopathy')
def dr():
    return render_template('diabeticretinopathy.html')

@app.route('/models/braintumour')
def bt():
    return render_template(r'braintumour.html')

@app.route('/models/breastcancer')
def bc():
    return render_template('breastcancer.html')

@app.route('/models/skindiseases')
def sd():
    return render_template('skindiseases.html')

@app.route('/predict_lc', methods = ['GET','POST'])
def upload_lc():
    if request.method == 'POST':
        f = request.files['image']
        print("current path")
        basepath = os.path.dirname(__file__)
```

```
        print("current path", basepath)
        filepath = os.path.join(basepath,'uploads',f.filename)
        print("upload folder is ", filepath)
        f.save(filepath)

        img = image.load_img(filepath,target_size = (64,64), color_mode="grayscale")
        x = image.img_to_array(img)
        x = np.expand_dims(x,axis =0)

        with graph.as_default():
            preds = model_lc.predict_classes(x)

            print("prediction",preds)

        index = ['Cancer','NonCancer']
        text = str(index[preds[0]])

        if (text == "Cancer"):
            text = "Cancer is seen.”
        else:
            text = "Cancer not seen. Stay safe and healthy."
        print(text)
    return text

if __name__ == '__main__':
    app.run(debug = True, threaded = False)
```

CHAPTER 6

RESULTS

6.1 HOME PAGE

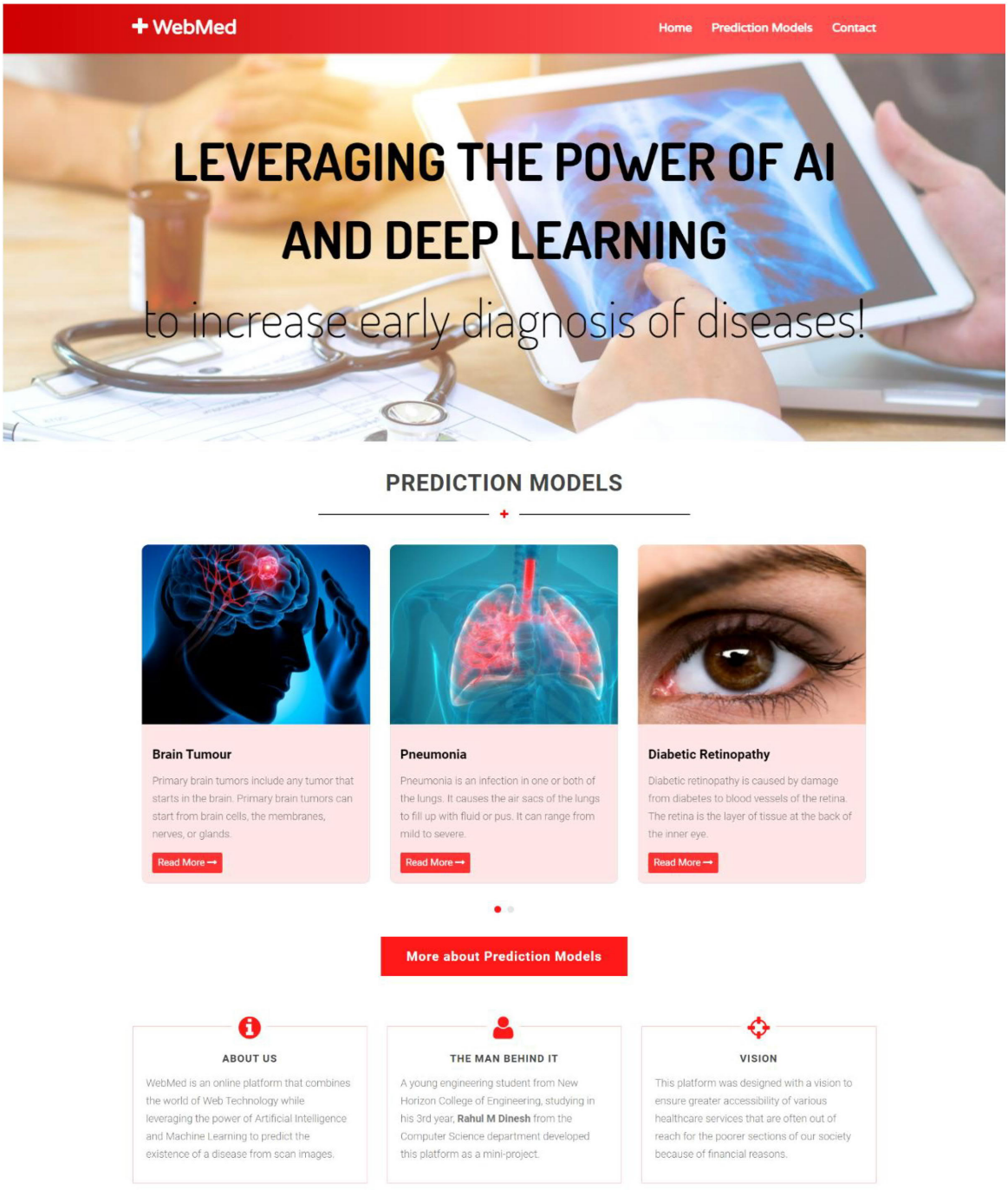


Figure 6.1: Home Page

6.2 PREDICTION MODELS PAGE

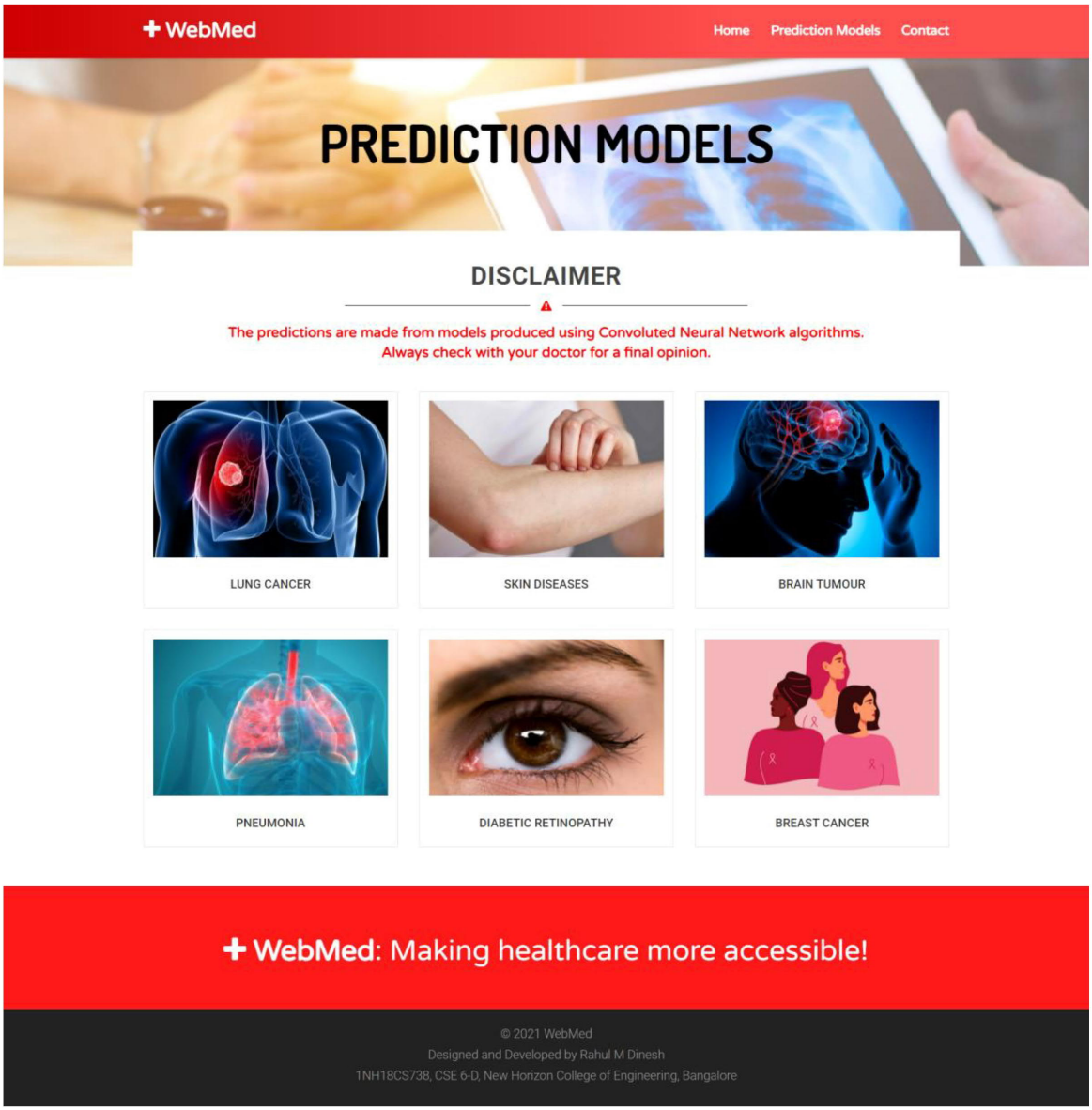


Figure 6.2: Prediction Models Page

6.3 LUNG CANCER PREDICTION

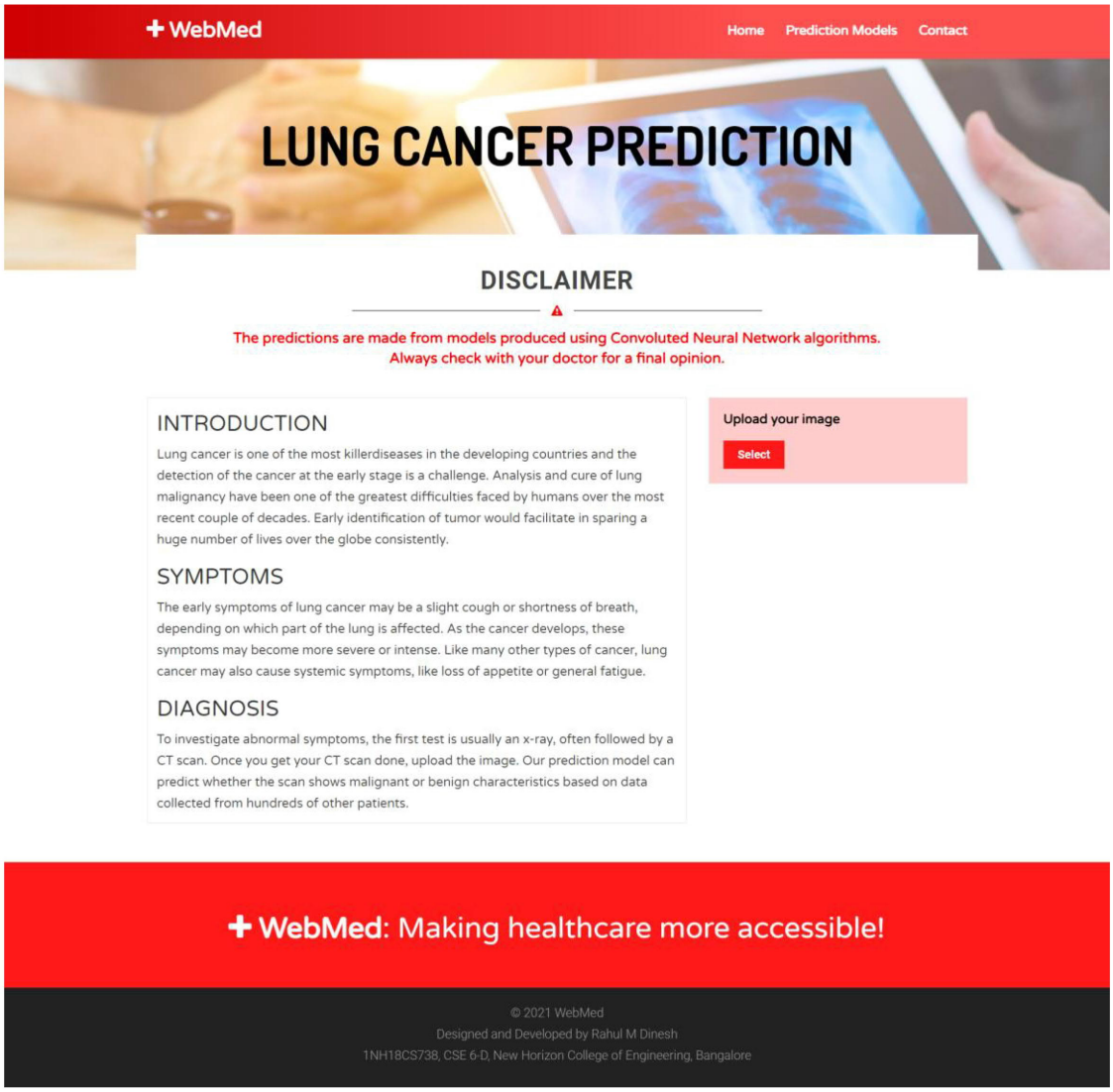


Figure 6.3: Lung Cancer Prediction Model Page

Once User uploads an image

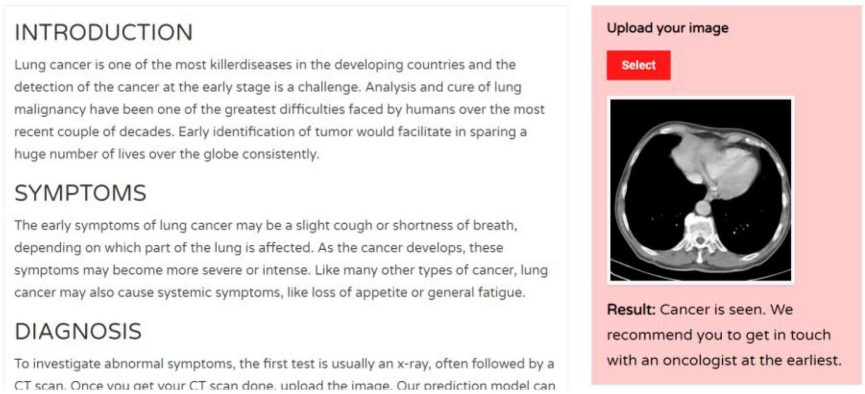


Figure 6.4: Lung Cancer Prediction Result

6.4 SKIN DISEASE PREDICTION

Figure 6.5: Skin Disease Prediction Model Page

Once User uploads an image,

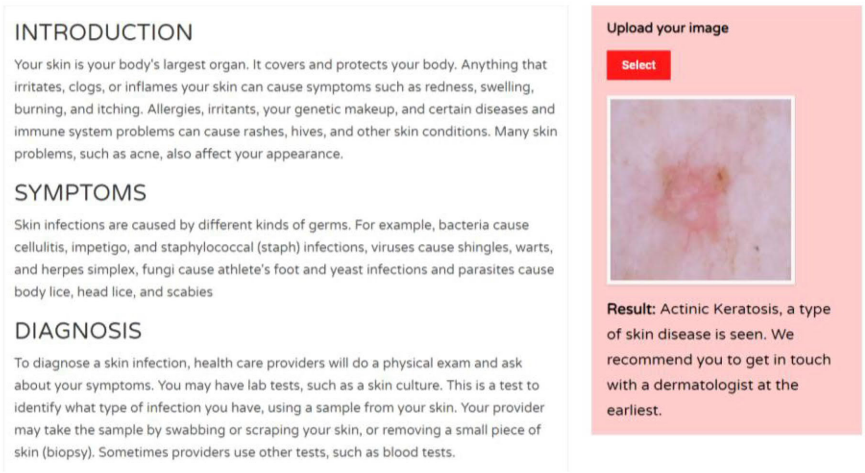


Figure 6.6: Skin Disease Prediction Result

6.5 BRAIN TUMOUR PREDICTION

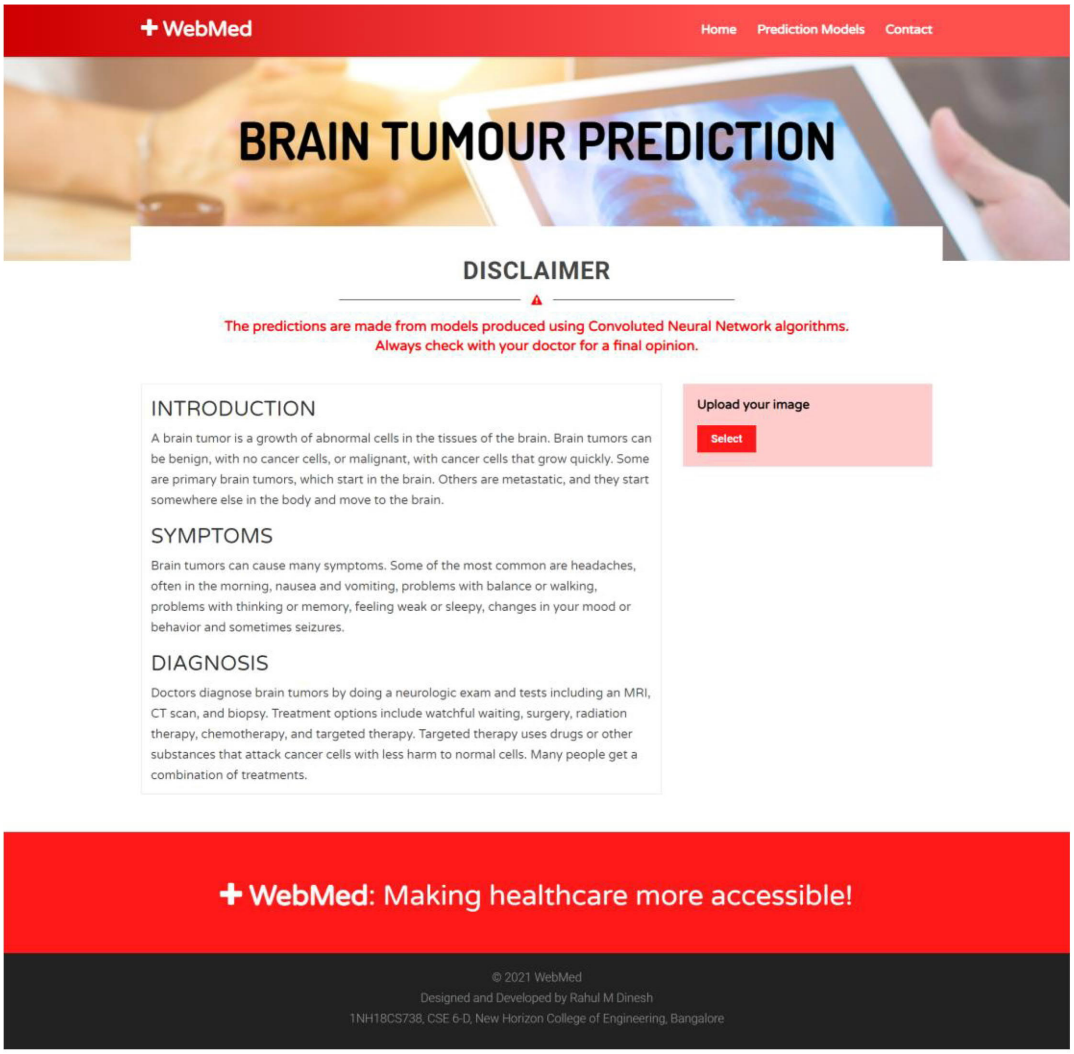


Figure 6.7: Brain Tumour Prediction Model Page

Once User uploads an image,

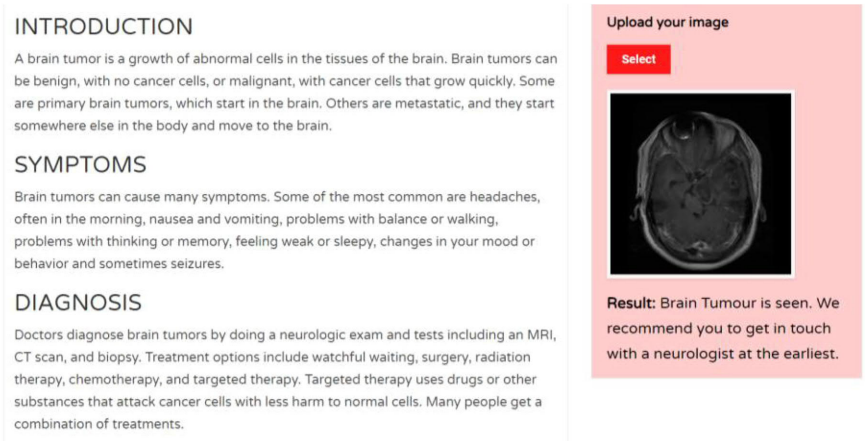


Figure 6.8: Brain Tumour Prediction Result

6.6 PNEUMONIA PREDICTION

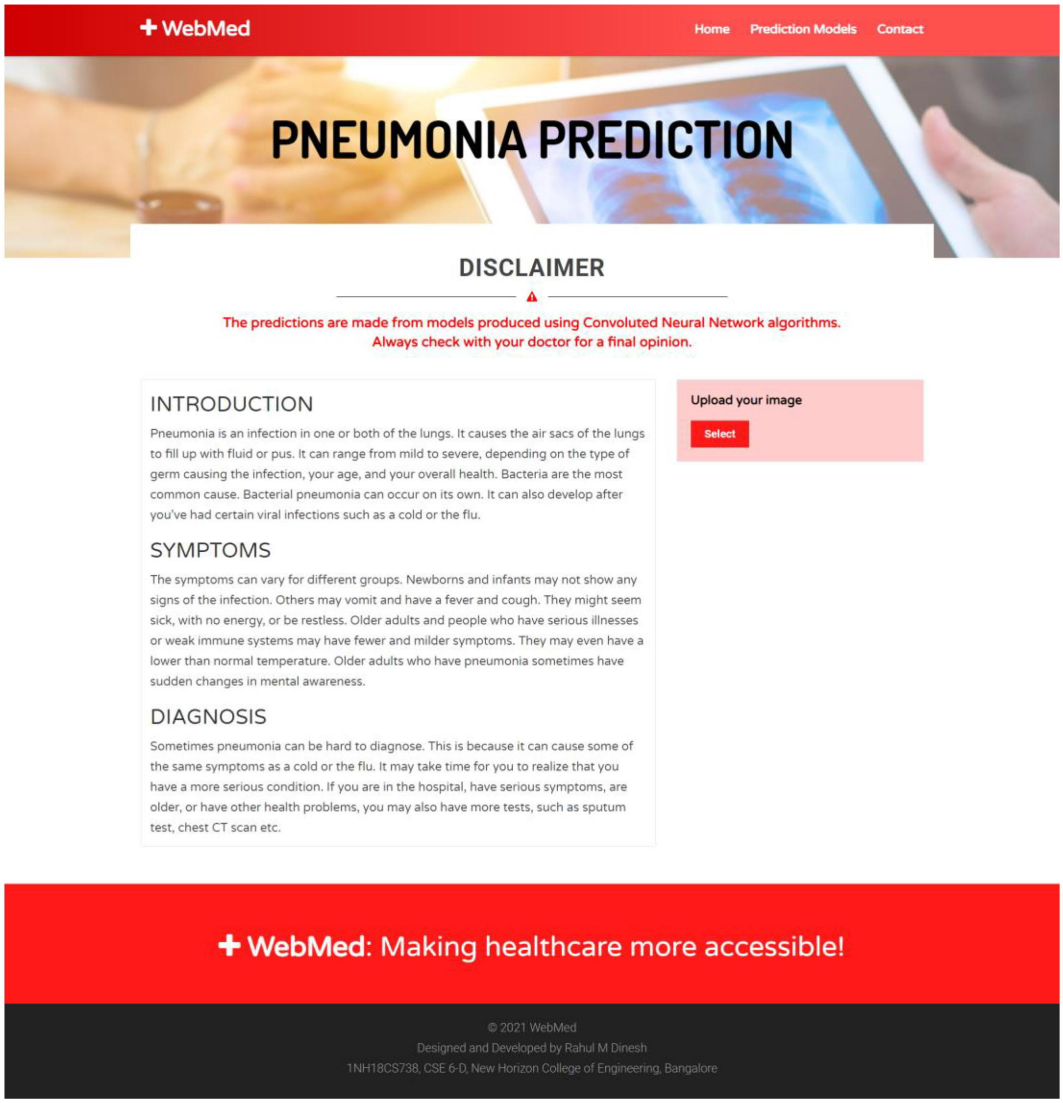


Figure 6.9: Pneumonia Prediction Model Page

Once User uploads an image,

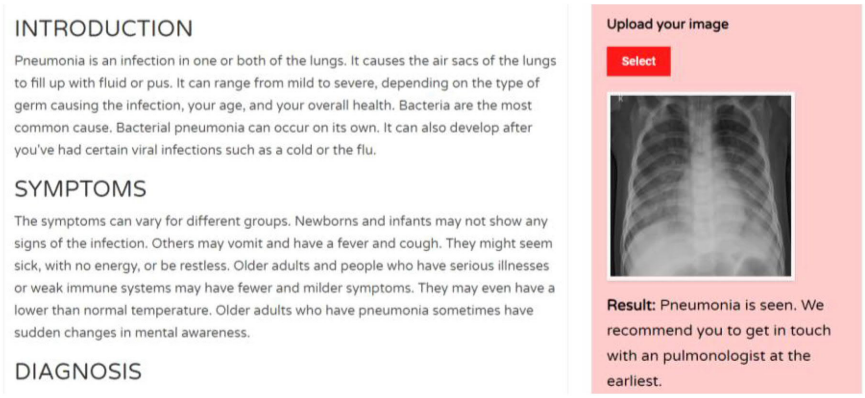


Figure 6.10: Pneumonia Prediction Result

6.7 DIABETIC RETINOPATHY PREDICTION

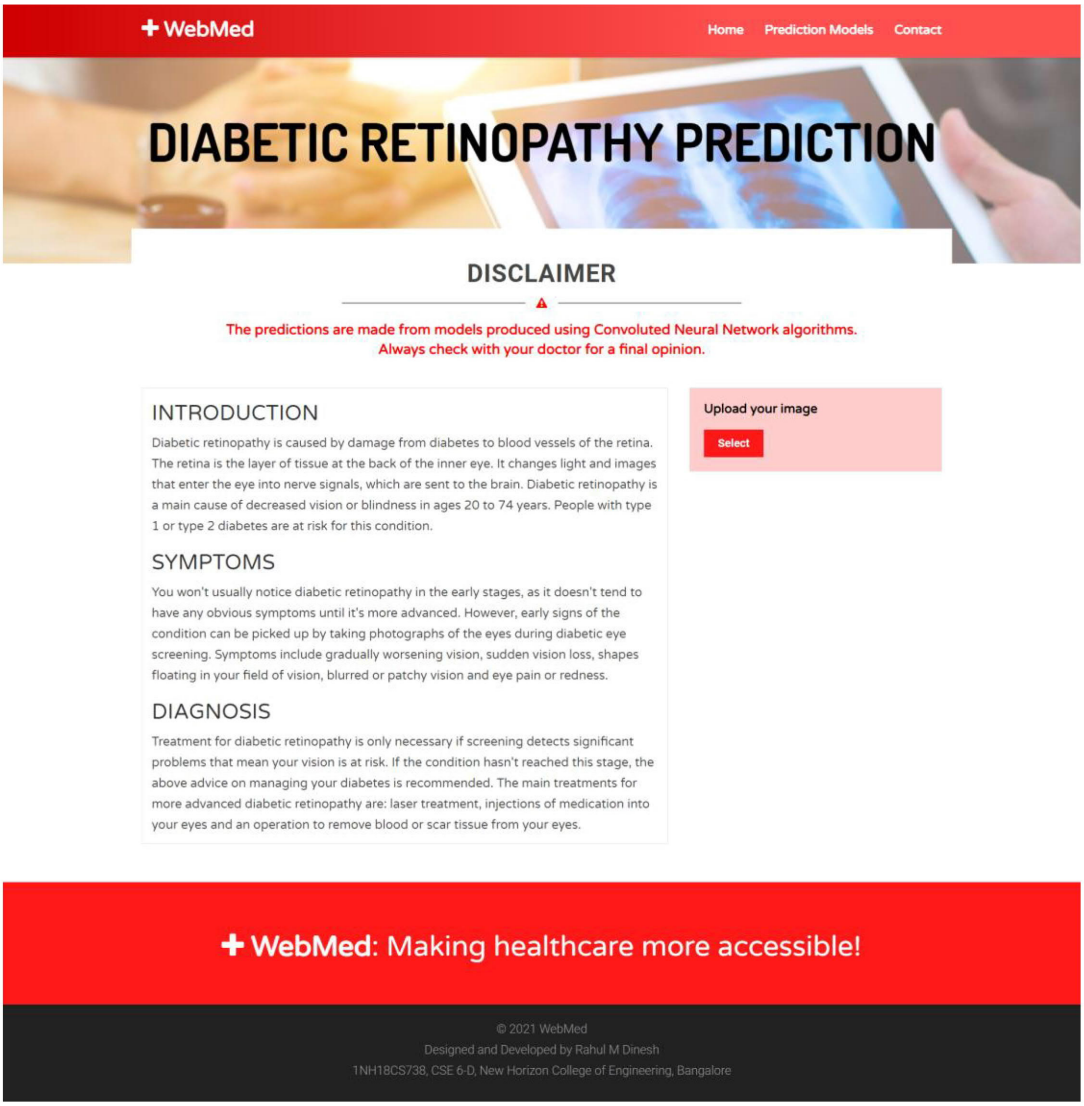


Figure 6.11: Diabetic Retinopathy Prediction Model Page

Once User uploads an image,

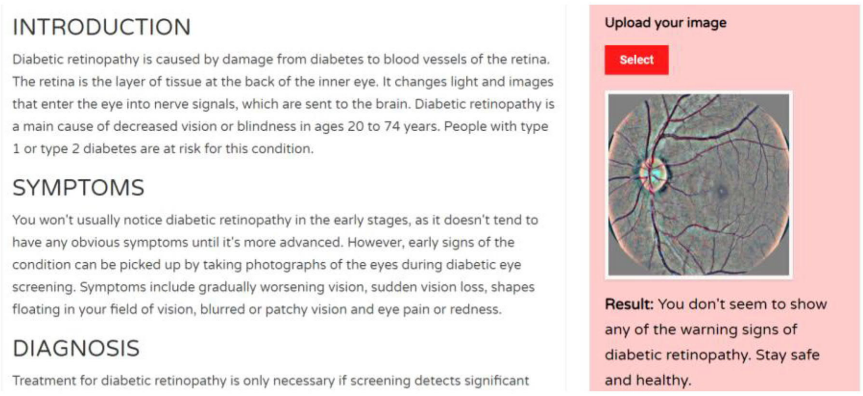


Figure 6.12: Diabetic Retinopathy Prediction Result

6.8 BREAST CANCER PREDICTION

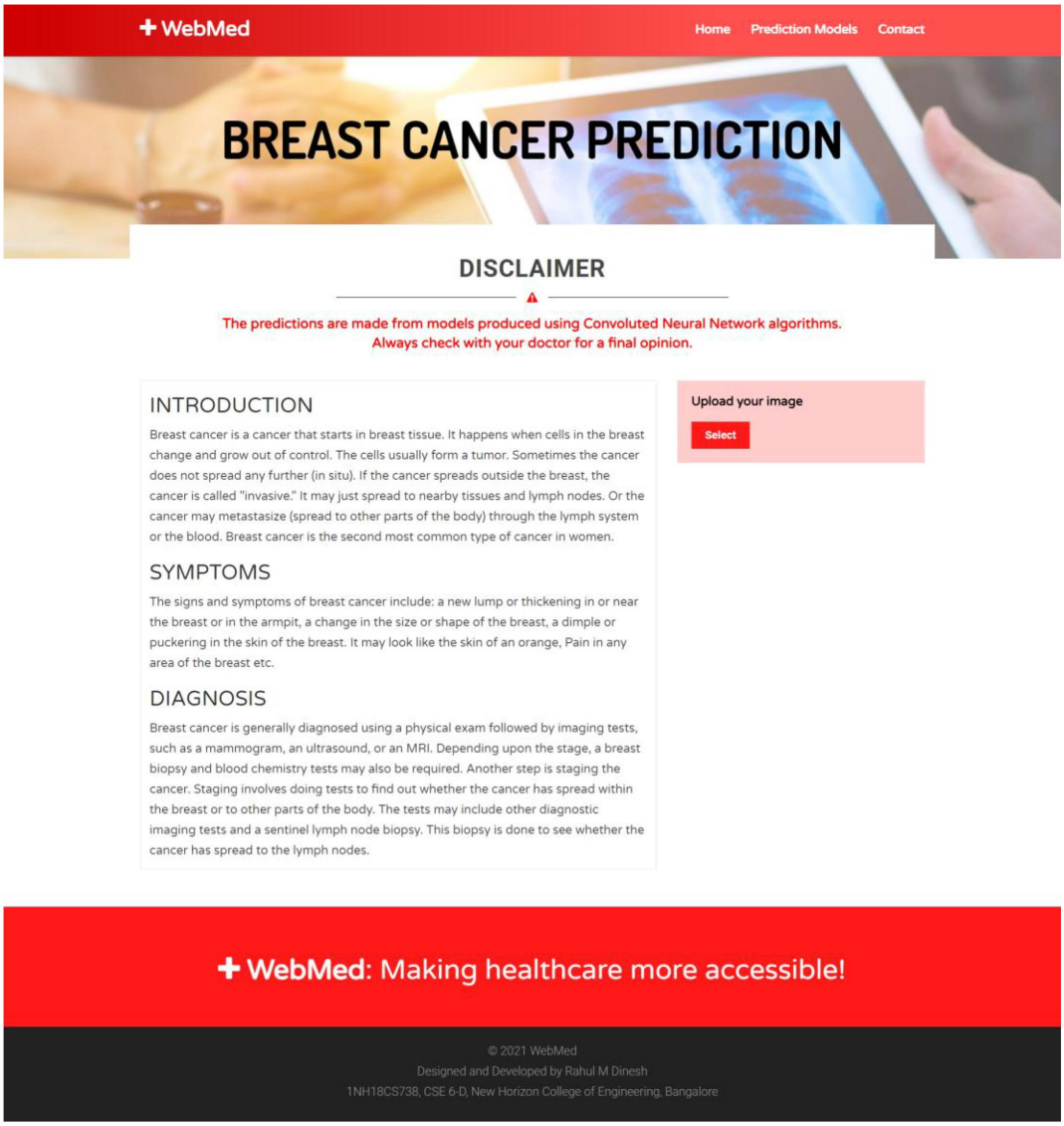


Figure 6.13: Breast Cancer Prediction Model Page

Once User uploads an image,

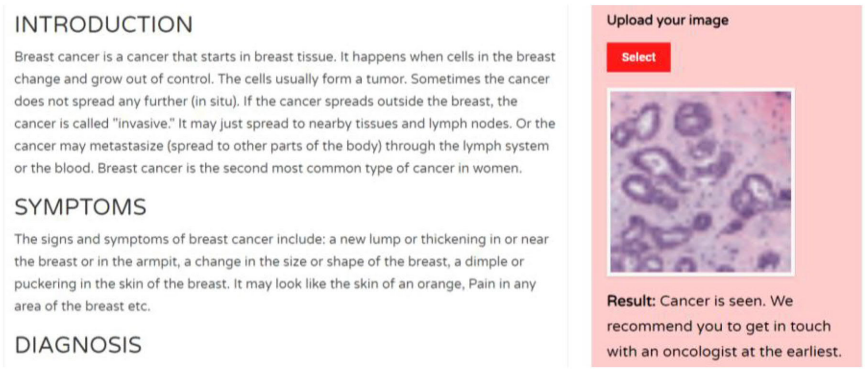


Figure 6.14: Breast Cancer Prediction Result

6.9 CONTACT PAGE

Filled contact form, before submitting

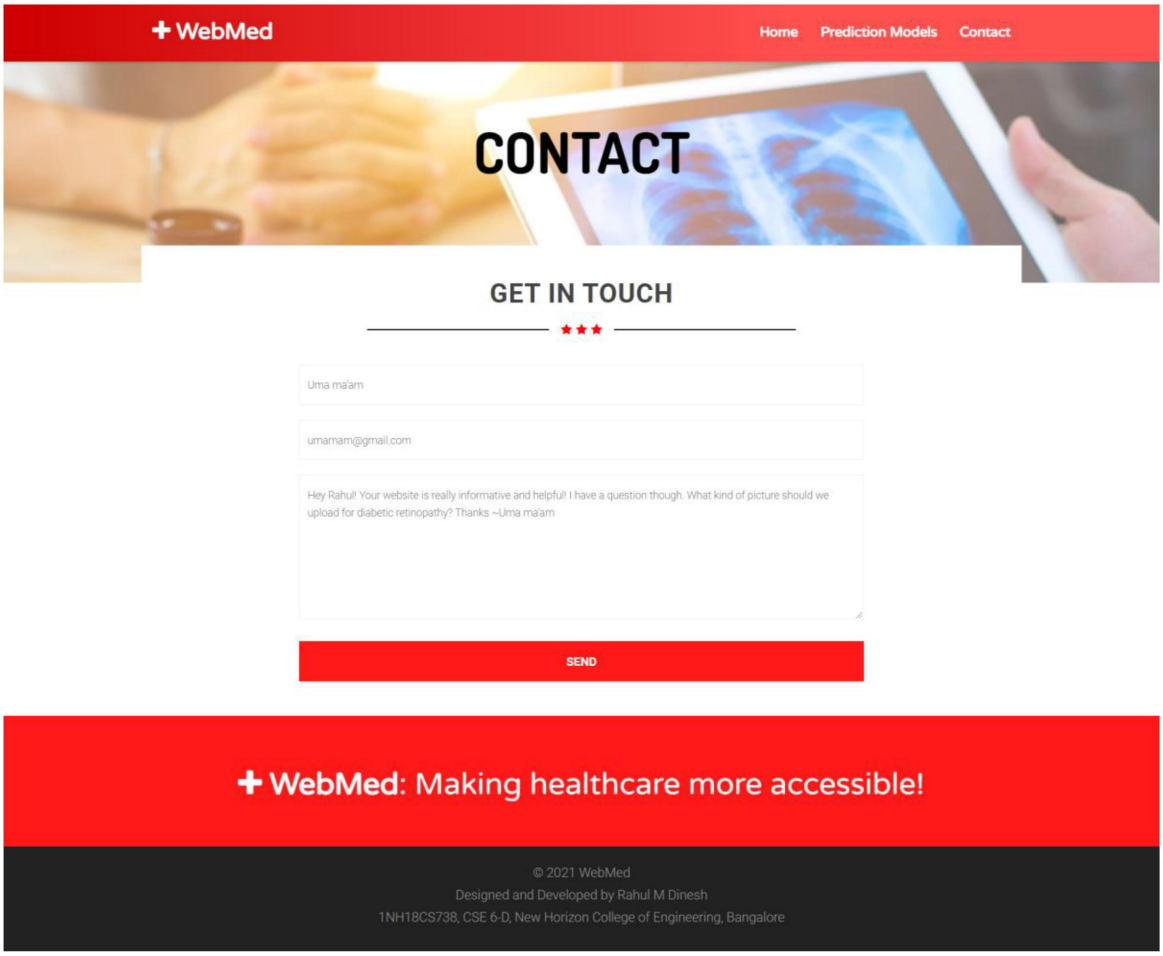


Figure 6.15: Contact form before submitting

After submitting,

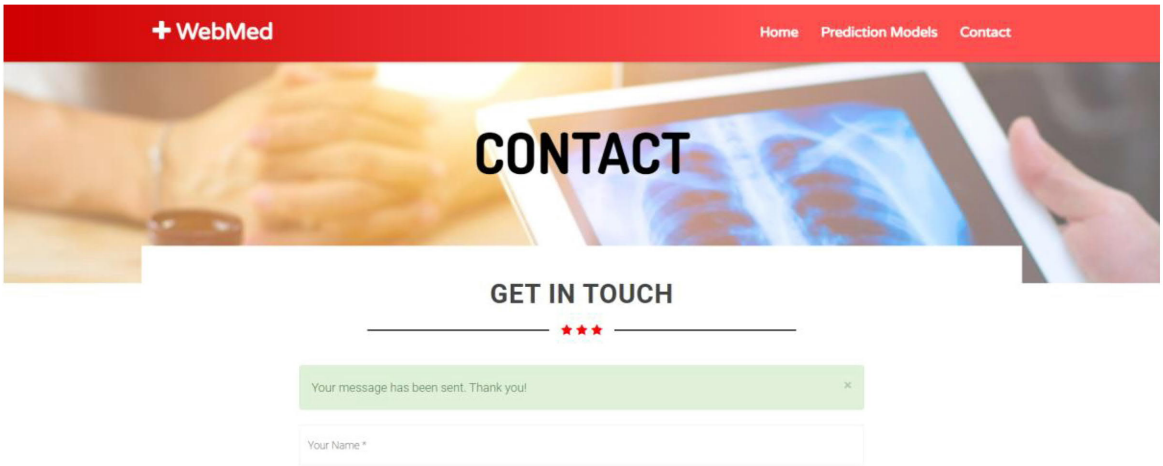


Figure 6.16: Contact form after submitting

Email sent to the designer-developer of the website,

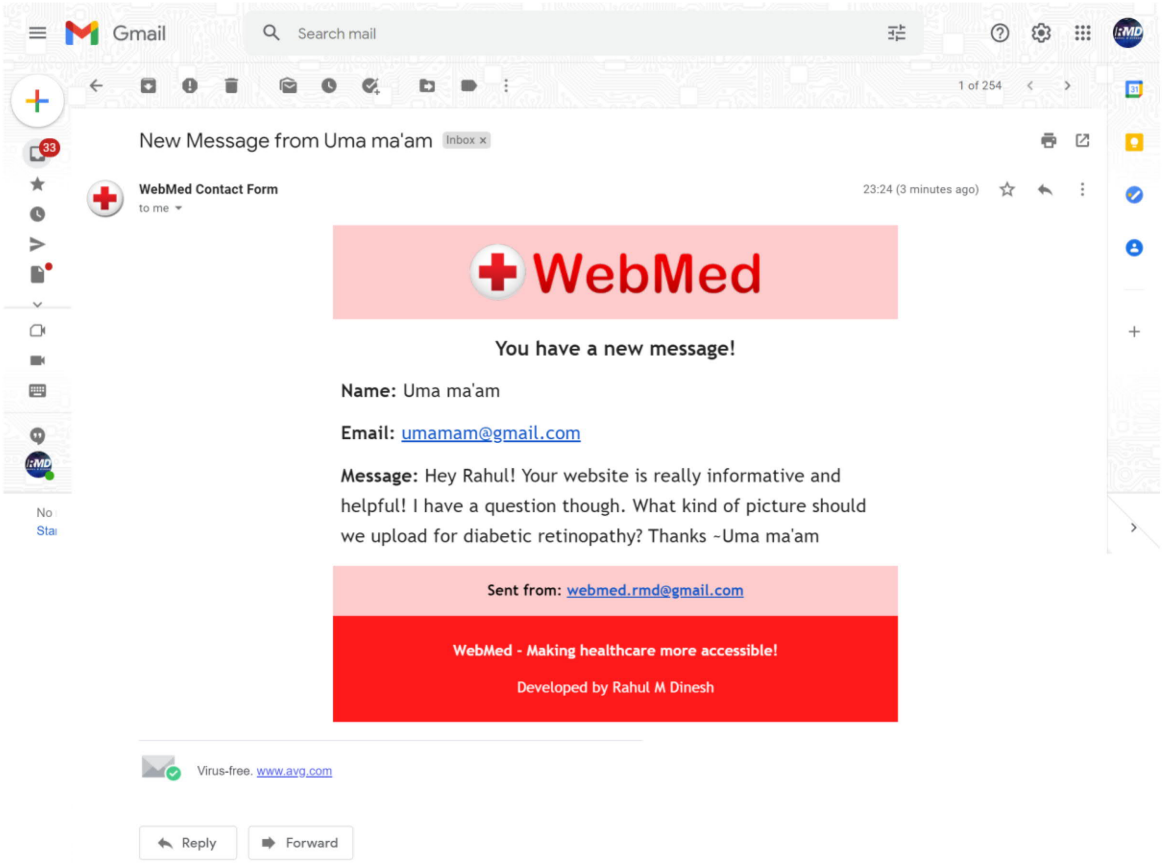


Figure 6.17: Email sent to the designer-developer from the Contact form

CHAPTER 7

CONCLUSION

The mini project has successfully accomplished the goals it had set out in the objectives and design sections of this report.

The website developed in this mini project has been able to provide a simple and easy to use graphical user interface that is informative yet dynamic and responsive by combining the power of various web technologies and Python's AI and ML libraries.

The website has successfully incorporated all the features which were to be implemented, as mentioned in the beginning of the report along with some additional features, thus giving the user a visually appealing, error-free and hassle-free experience.

Three main pages were developed, i.e., the Home page, which was the gateway to the entire website. It contained navigation links to all the prediction model pages developed in this miniproject as well as information regarding the WebMed brand (about us, the man behind it and vision). The Prediction Models page cumulates all the prediction models in one webpage thus making it easier for the users to navigate to the required page. The Contact page provides the users with the ability to send suggestions and comments directly to the website's designer-developer. The content of this message was sent directly to the designer-developer's Email ID.

Various disease prediction models were also developed which provide a high degree of accuracy using Python's various Machine Learning libraries. Prediction Models were devised for a wide range of diseases like Lung Cancer, Pneumonia, Brain Tumour, Breast Cancer, Diabetic Retinopathy and Skin Diseases. These models were then used to make the appropriate prediction based on the user's uploaded image.

In the future, I aim to extend the scope of this mini project by having a database that can collect users' information and help provide them the ability to make appointments with doctors and hospitals. I also plan to add additional functionalities that would enhance the user experience.

REFERENCES

[1] https://www.programiz.com/

[2] https://www.geeksforgeeks.org/

[3] https://www.javatpoint.com/

[4] https://anzeljg.github.io/

[5] https://www.tutorialspoint.com/

[6] https://docs.python.org/

[7] https://realpython.com/

[8] https://www.edureka.co/

[9] https://pythonprogramming.net/

[10] https://www.tutorialsteacher.com/