WAPH-Web Application Programming and Hacking

Instructor: Dr. Phu Phung

Student

Name: Tejaswee Rayana

Email: rayanate@mail.uc.edu

Short-bio: Tejaswee Rayana has a great interest in coding. She wants to become a full-stack developer.

Figure 1: Teju's Headshot

Repository Information

Repository's URL: https://github.com/rayanate/rayanate.github.io

This is a private repository for Tejaswee rayana to store all the code from the course. The organization of this repository is as follows.

Overview and Requirements

In this project, I have developed a professional website using HTML, CSS, Javascript and Bootstrap. The application has been deployed on GitHub website. Below is the URL for the portfolio website.

Website's URL: https://github.com/rayanate/rayanate.github.io

General requirements:

I have successfully crafted a professional and visually appealing resume webpage using HTML5, CSS, JavaScript, and Bootstrap. The page incorporates

essential details such as my name, contact information, a professional headshot, and comprehensive sections highlighting my educational background and work experiences. The use of Bootstrap ensures a responsive and well-structured design, creating a seamless and polished presentation of my resume online.

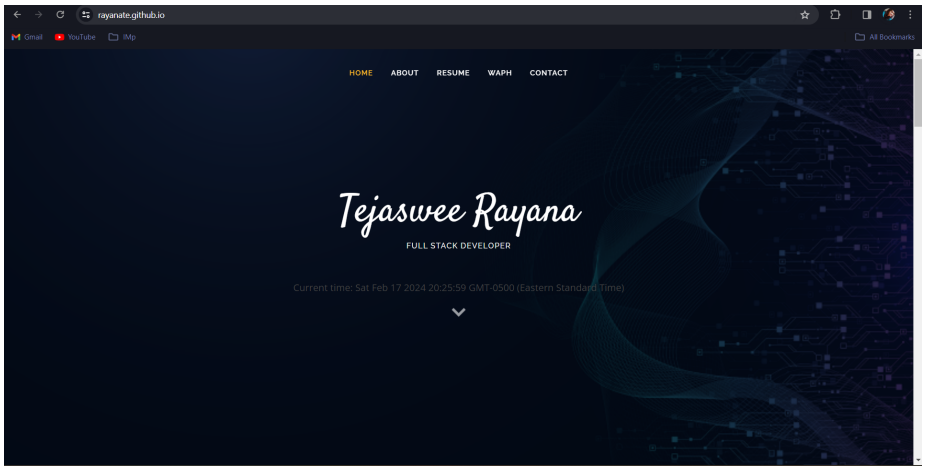

Figure 2: Home Page of Portfolio Website - Main Secton

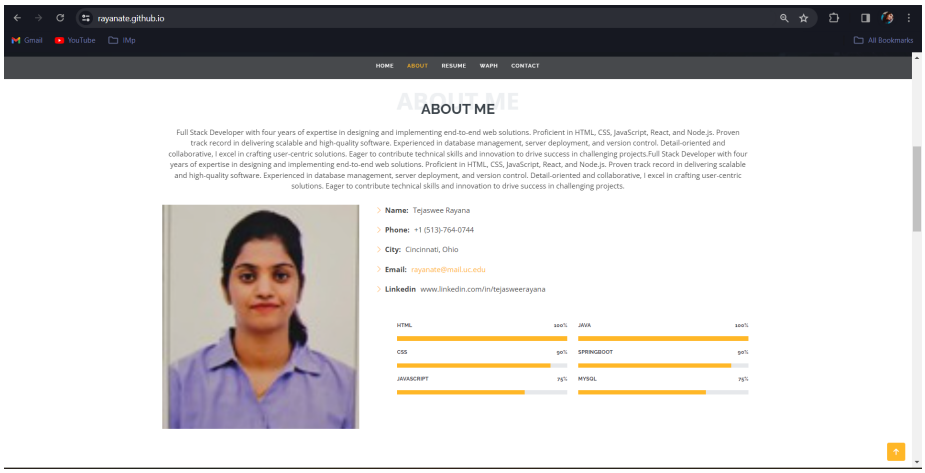

Figure 3: Home Page of Portfolio Website - About Section

Non-technical requirements

I have added bootstrap cdn in the HTML page and used bootstrap classes to style the elements on the page quickly. This has helped me achieve a clean and professional design without having to write extensive CSS code. Below are the

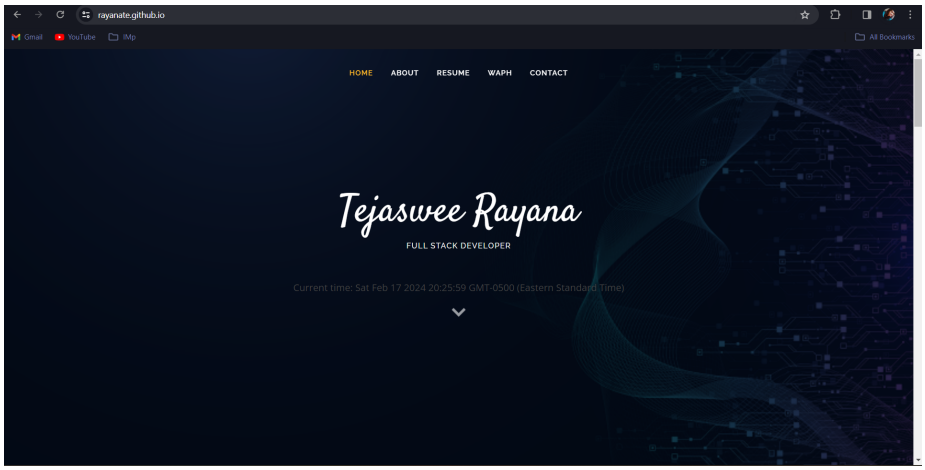

Figure 4: Home Page of Portfolio Website - Resume Section

CDN links for bootstrap and the icons used in the project.

```
<link href="assets/vendor/bootstrap/css/bootstrap.min.css" rel="stylesheet">
<link href="assets/vendor/bootstrap-icons/bootstrap-icons.css" rel="stylesheet">
<link href="assets/vendor/boxicons/css/boxicons.min.css" rel="stylesheet">
<link href="assets/vendor/glightbox/css/glightbox.min.css" rel="stylesheet">
<link href="assets/vendor/swiper/swiper-bundle.min.css" rel="stylesheet">
```

Technical Requirements

Basic Javascript code

I have used JQuery CDN in the HTML page to access the elements and performed the API calls for retrieving data from the server. This allows for dynamic updating of content without having to reload the entire page.

Below is the JQuery CDN used in the project

```
<script src="https://code.jquery.com/jquery-3.7.1.min.js"
    integrity="sha256-/JqT3SQfawRcv/BIHPThkBvs0OEvtFFmqPF/lYI/Cxo=" crossorigin="anonymous">
```

I have added React CDN in the HTML page to include react library in the website.

```
<script crossorigin src="https://unpkg.com/react@17/umd/react.production.min.js"></script>
<script crossorigin src="https://unpkg.com/react-dom@17/umd/react-dom.production.min.js"></s
```

```
<div id="root"></div>

      <script>
        const App = () => {
```

```
        return React.createElement('h1', null, 'Tejaswee Rayana');
      };

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

I have included the functionalities from Lab2 i.e. analog clock, digital clock, show/hide email, Joke API 1. Digital Clock

```
function displayTime() {
    document.getElementById('digit-clock').innerHTML = "Current time: " + new Date();
    }
    setInterval(displayTime, 500);
```

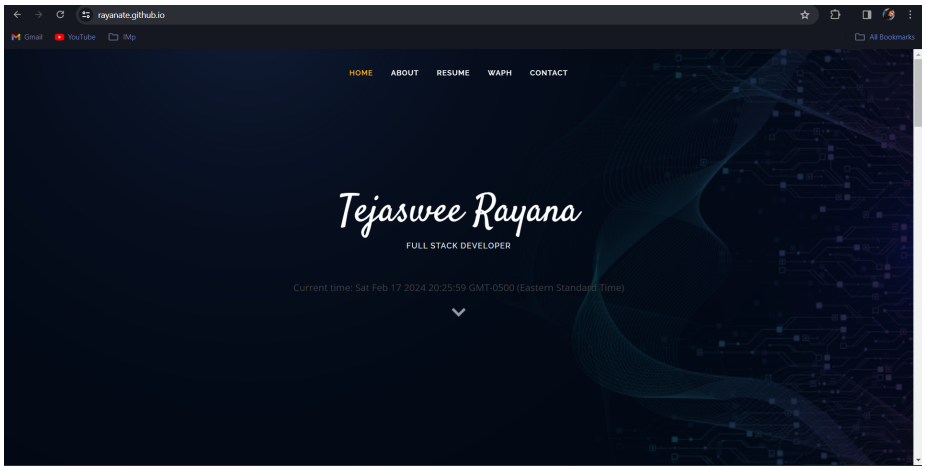

Figure 5: Digital Clock

2. Analog Clock

```
<div class="col">
  <h3>Analog Clock</h3>
  <canvas id="analog-clock" width="150" height="150" style="background-color:#999"></canvas>
  <script src="https://waph-uc.github.io/clock.js"></script>
  <script>
    function displayTime() {
        document.getElementById('digit-clock').innerHTML = "Current time: " + new Date();
          }
          setInterval(displayTime, 500);
          var canvas = document.getElementById("analog-clock");
          var ctx = canvas.getContext("2d");
```

```
      var radius = canvas.height / 2;
          ctx.translate(radius, radius);
          radius = radius * 0.90;
          setInterval(drawClock, 1000);
      function drawClock() {
            drawFace(ctx, radius);
              drawNumbers(ctx, radius);
              drawTime(ctx, radius);
              }
  </script>
```

3. Show/Hide Email

```
var shown = false;
function showhideEmail() {
  if (shown) {
      document.getElementById('email').innerHTML = "Show my email";
      shown = false;
  }
  else {
      var myemail = "<a href='mailto:rayanate" + "@" +
          "mail.uc.edu'> rayanate" + "@" + "mail.uc.edu</a>";
      document.getElementById('email').innerHTML = myemail;
      shown = true;
  }
}
```

4. Joke API

```
function fetchJoke() {
    $.get('https://v2.jokeapi.dev/joke/Any', function (data) {
        $('#joke-container').html(`
            <p>${data.setup || ''}</p>
            <p>${data.delivery || data.joke || ''}</p>
        `);
    }).fail(function (error) {
        console.error('Error fetching joke:', error);
    });
}

fetchJoke();

setInterval(fetchJoke, 60000);
```

5. Public API I have integrated Weather API to display weather and NASA API to display the image of the day.

    a. Weather API - I have included graphic i.e. image of the cloud.

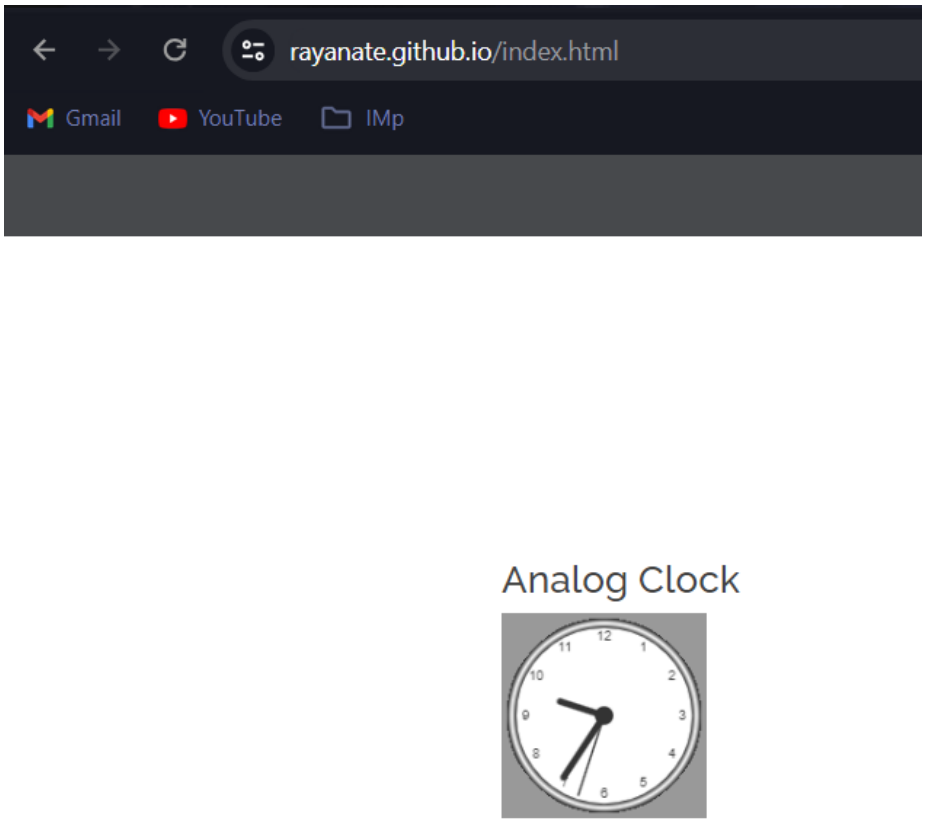

Figure 6: Analog Clock

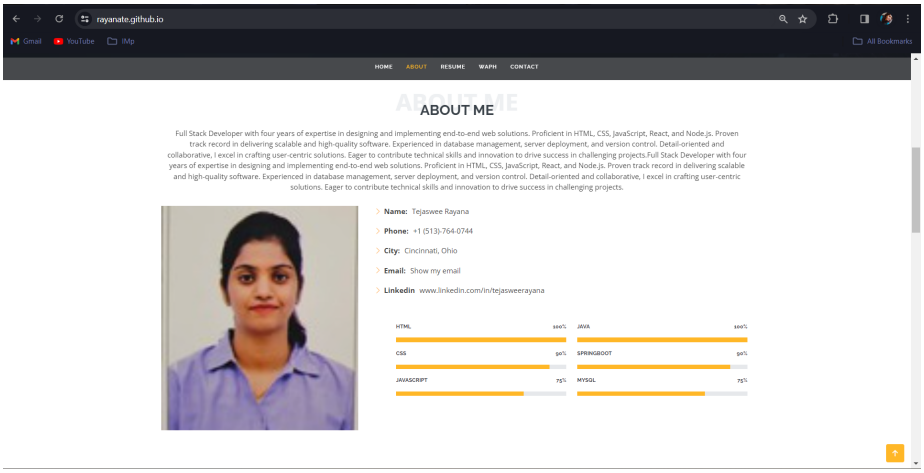

Figure 7: Show/Hide Email - Hide Email

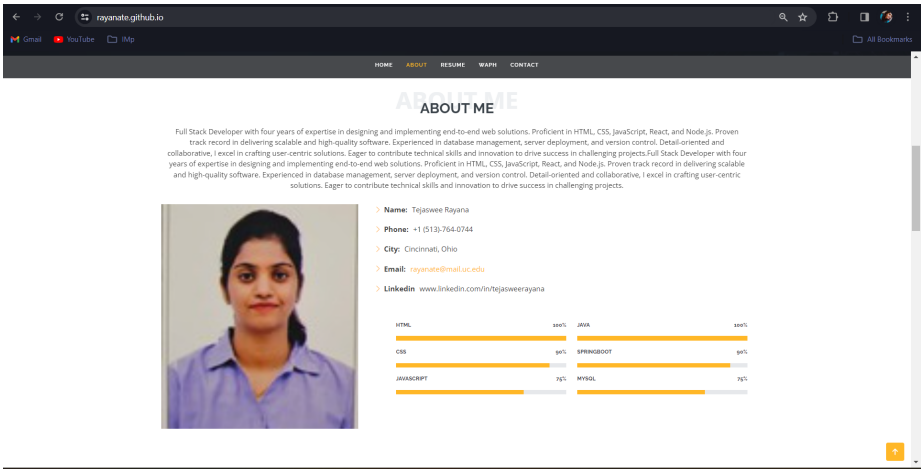

Figure 8: Show/Hide Email - Email Shown

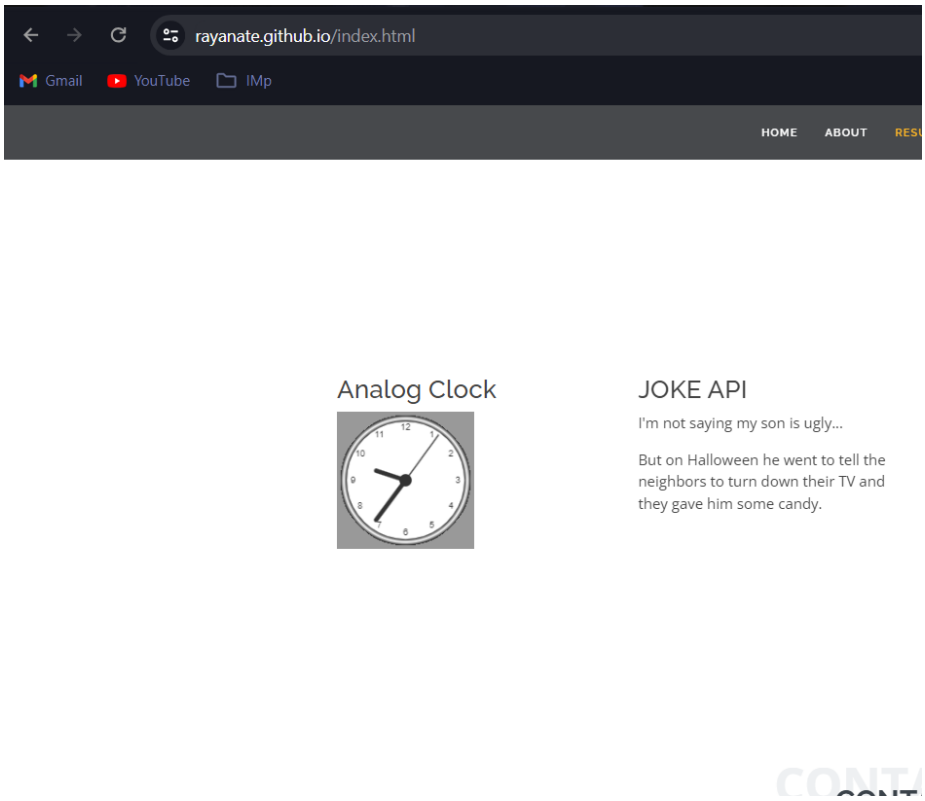

Figure 9: Displaying Joke for every one minute

```
async function getWeatherData(cityName) {
    const API_KEY = '82bf333e96f9446884281940ce9c06b1';
    const API_URL = `https://api.weatherbit.io/v2.0/current?city=${cityName}&key=${

    try {
        const weatherResponse = await fetch(API_URL);
        const responseData = await weatherResponse.json();
        document.getElementById('city').innerText = responseData.data[0].city_name,
        document.getElementById('weather-icon').src = `https://www.weatherbit.io/st
        document.getElementById('temperature').innerText = `Temperature: ${response
        document.getElementById('description').innerText = `Description: ${response
    } catch (error) {
        console.error('Error fetching weather data:', error);
    }
}

const city = 'Cincinnati';
getWeatherData(city);
```

Figure 10: Dsplaying Weather

b. NASA Astronomy Picture of the Day (APOD)

```
    const apiKey = 'rD28vkNDNAtUjRvS4jcgLJnBII2IbEtZpFY7PtMO';
fetch(`https://api.nasa.gov/planetary/apod?api_key=${apiKey}`)
  .then(response => response.json())
  .then(data => {
      const apodImage = document.getElementById('apodImage');
      apodImage.src = data.url;
      apodImage.alt = data.title;
  })
  .catch(error => {
```

```
        console.error('Error fetching APOD image:', error);
    });
```

Figure 11: Displaying NASA's Astronomy Picture of the Day

6. Javascript Cookies

I have used the localstorage to store the session of the user. Below is the code snippet of the implementation.

```
function setCookies() {
        var currentDate = new Date();
        var lastVisit = localStorage.getItem("lastVisit");

        if (!lastVisit) {
          localStorage.setItem("lastVisit", currentDate.toISOString());
          alert("Welcome to my homepage!");
        } else {
          var lastVisitDate = new Date(lastVisit);
          alert("Welcome back! Your last visit was on " + lastVisitDate);
        }
      }
      setCookies();
```

A pandoc file has been generated from Readme.md with the name rayanate-waph-project1.pdf

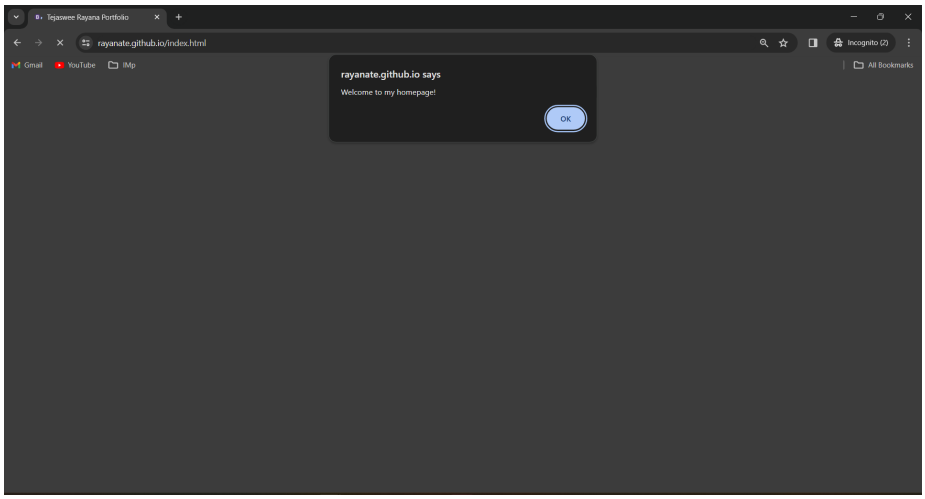

Figure 12: Alert Message shown on First Visit

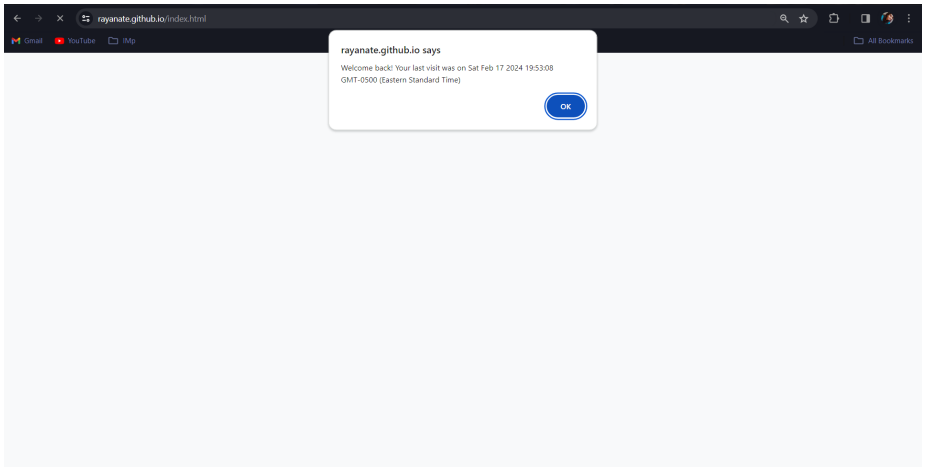

Figure 13: Alert Message shown on revisit