WAPH-Web Application Programming and Hacking

Professor: Dr. Phu Phung

Student

Name: Yeshwanth Gudise

Email: gudiseyh@mail.uc.edu

Short-bio: Yeshwanth Gudise has a better understanding and passion towards programming. He wish to be as a software Engineer.

Repository Information

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

Yeshwanth Gudise can place all of the code in this private repository. This repository is structured as follows.

Overview and Requirements

In this project, I've created a professional website using HTML, CSS, Javascript, and Bootstrap. The application has been deployed on the GitHub website. Here is the URL for the portfolio website.

Website's URL: https://gudiseyh.github.io/

General requirements:

I have skillfully designed a professional and aesthetically pleasing resume webpage by utilizing HTML5, CSS, JavaScript, and Bootstrap. The webpage includes crucial information such as my name, contact details, a professional headshot, and comprehensive sections that showcase my educational history and work experiences. The incorporation of Bootstrap ensures that the design is responsive and well-organized, providing a smooth and refined online presentation of my resume.

Non-technical requirements

I've incorporated Bootstrap CDN into my HTML page and leveraged Bootstrap classes to swiftly style the elements on the page. This approach has enabled me to attain a polished and professional design without the need for extensive CSS coding. Below, you'll find the CDN links for Bootstrap and the icons utilized in the project.

Figure 1: Yeshwanth's Headshot

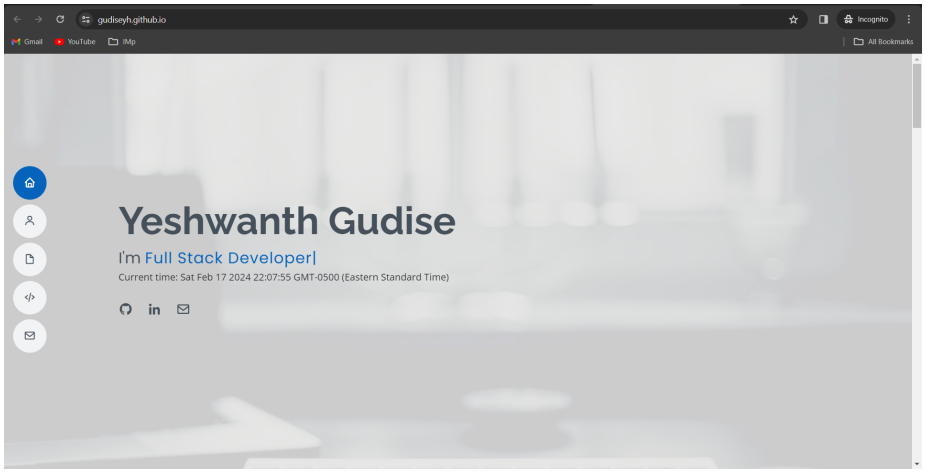

Figure 2: Main Page of Website - Main Secton

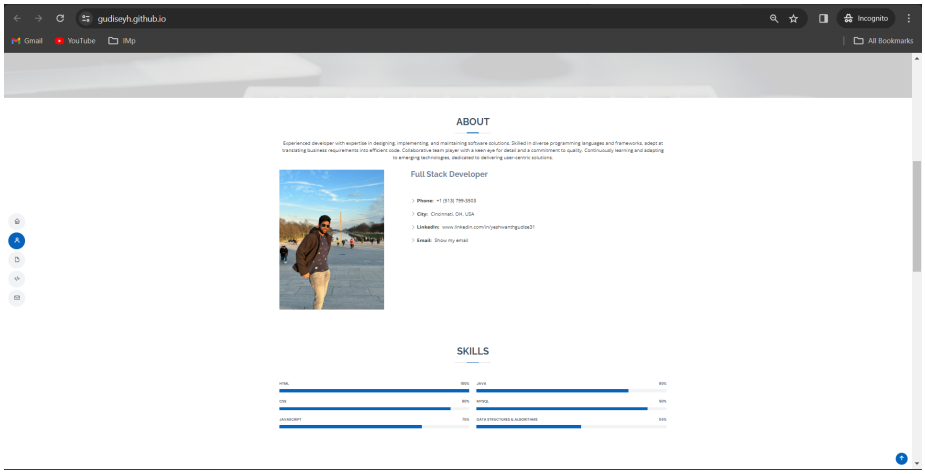

Figure 3: Portfolio Website - About Section

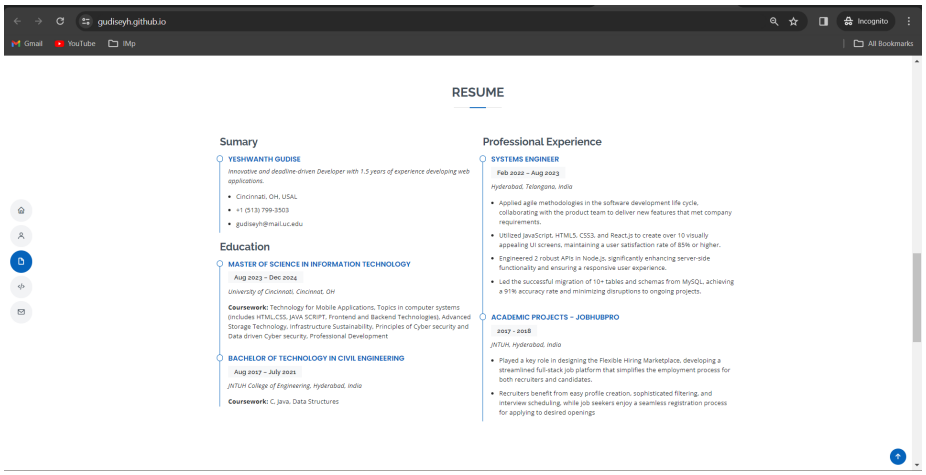

Figure 4: Portfolio Website - Resume Section

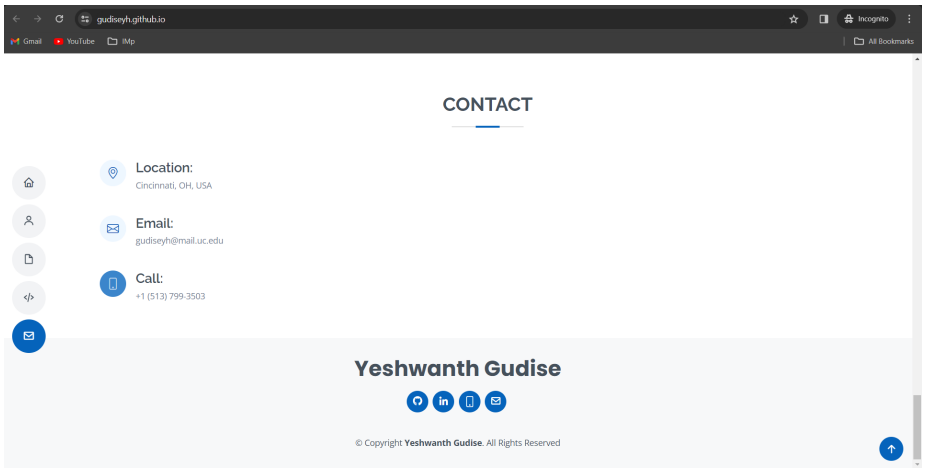

Figure 5: Portfolio Website - Contact Section

```
<link
  href="https://fonts.googleapis.com/css?family=Open+Sans:300,300i,400,400i,600,600i,700,7
  rel="stylesheet">

<link href="assets/vendor/aos/aos.css" rel="stylesheet">
<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've integrated the JQuery CDN into my HTML page to access elements and execute API calls for fetching data from the server. This enables dynamic content updates without the necessity of reloading 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
```

```
<script>
            const App = () => {
              return React.createElement('p', null, 'Experienced developer with expertise in
            };

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

I've incorporated the functionalities from Lab2 into the project, including features like an analog clock, digital clock, the ability to show/hide email, and integration with the Joke API.

1. Digital Clock

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

```
    }
    setInterval(displayTime, 500);
```

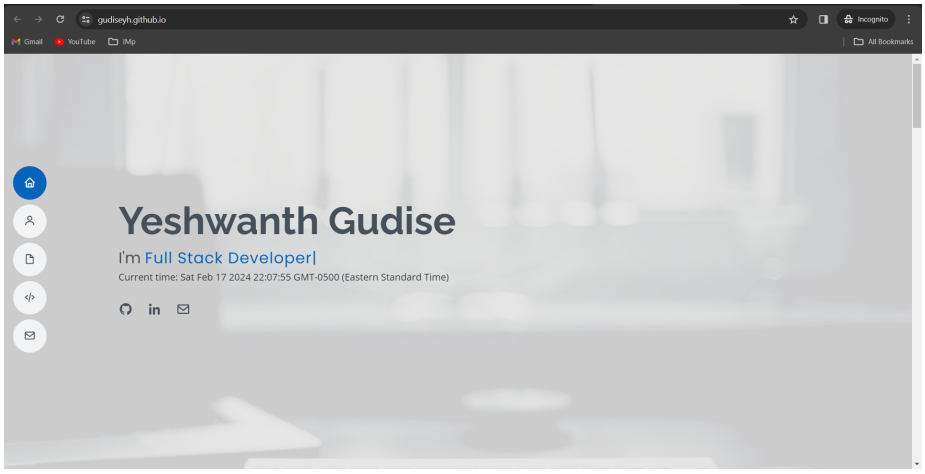

Figure 6: Digital Clock

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;
```

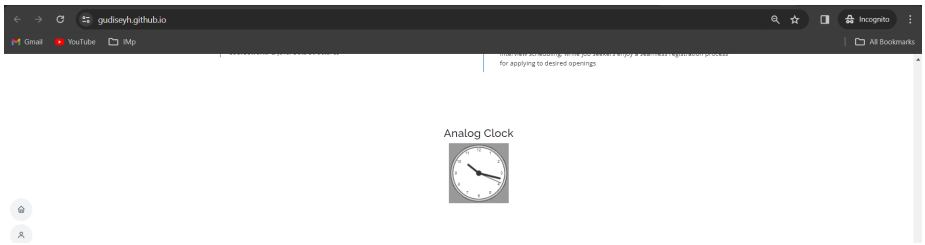

Figure 7: Analog Clock

```
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;
  }
}
```

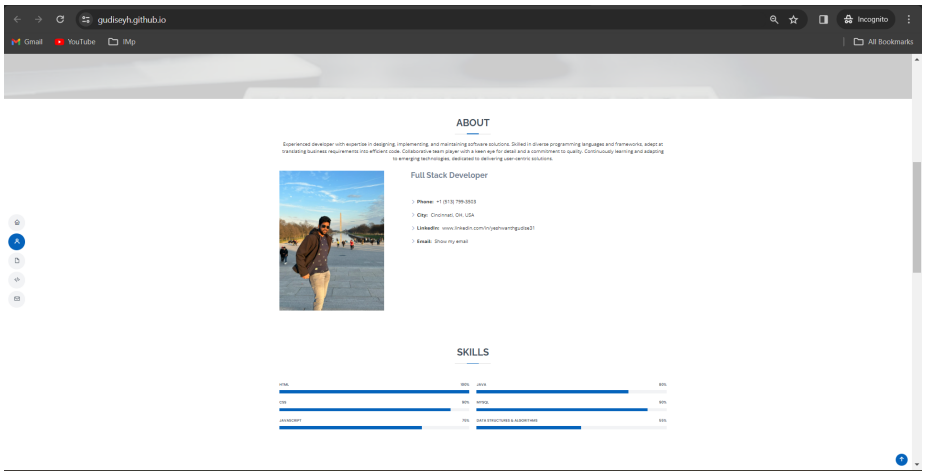

Figure 8: Show/Hide Email - Hide Email

4. Joke API

```
function fetchJoke() {
      $.get('https://v2.jokeapi.dev/joke/Any', function (data) {
        $('#joke-section').html(`
```

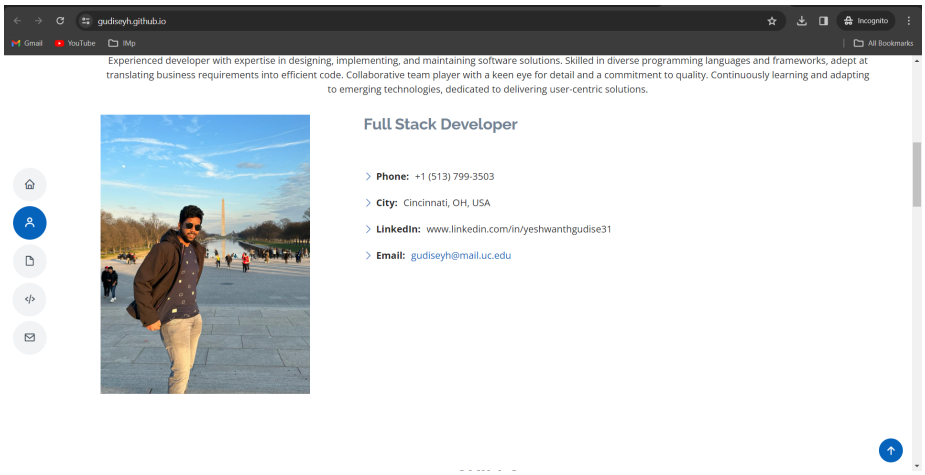

Figure 9: Show/Hide Email - Email Shown

```
        <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've integrated the Weather API to showcase real-time weather information and the random Quote Generator to present quotes on the webpage.

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

"'JS

async function getWeatherData(city) { const apiKey = 'd719fc7c306d442ba48f6e1b722bcc2c'; const apiUrl = `https://api.weatherbit.io/v2.0/current?city=${city}&key=${apiKey}`; try { const response = await fetch(apiUrl); const data = await response.json();

document.getElementById('city').innerText = data.data[0].city_name; document.getElementById('weather-icon').src = `https://www.weatherbit.io/static/img/icons/${da document.getElementById('temperature').innerText = `Temperature: ${data.data[0].temp}°C`; document.getElementById('description').innerText = `Description: ${data.data[0].weather.description}`; } catch (er-

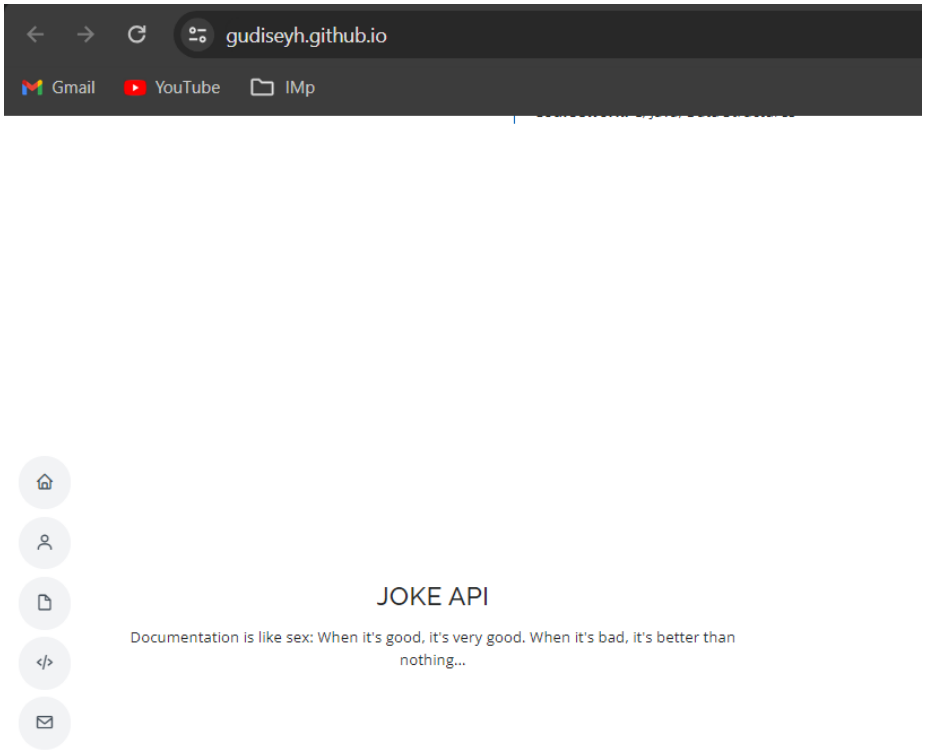

Figure 10: Displaying Joke for every one minute

ror) { console.error(‘Error fetching weather data:’, error); } }

```
const city = 'New York';
getWeatherData(city);

```

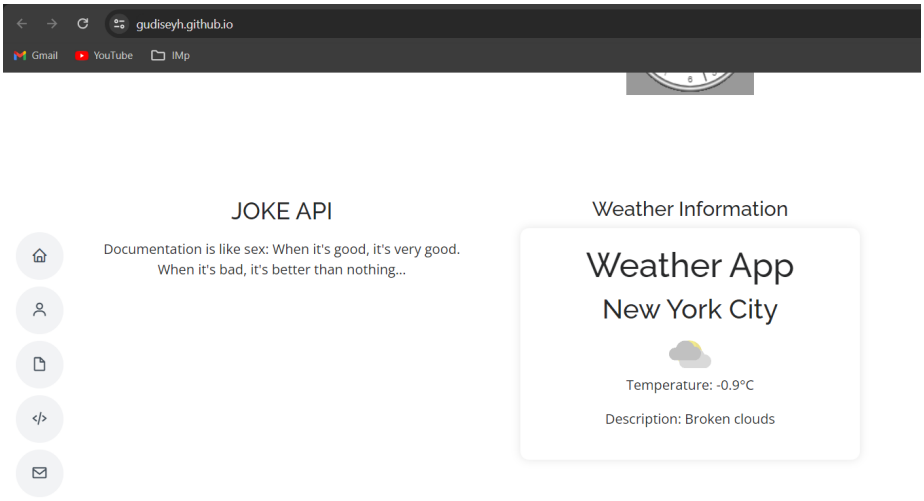

Figure 11: Dsplaying Weather

b. Random Quote Generator

```JS
       function getRandomQuote() {
  const apiUrl = 'https://api.quotable.io/random';
  $.get(apiUrl, function (data) {
    const quoteContainer = $('#quote');
    quoteContainer.html(`
                <p>${data.content}</p>
                <p>- ${data.author}</p>
            `);
  })
    .fail(function (error) {
      console.error('Error:', error);
    });
}
```

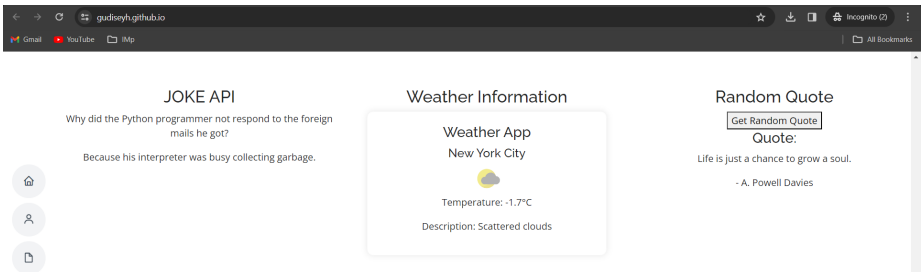

Figure 12: Random Number Facts

4. Javascript Cookies

I've implemented user session storage using localStorage. Here is the code snippet for the implementation.

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

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

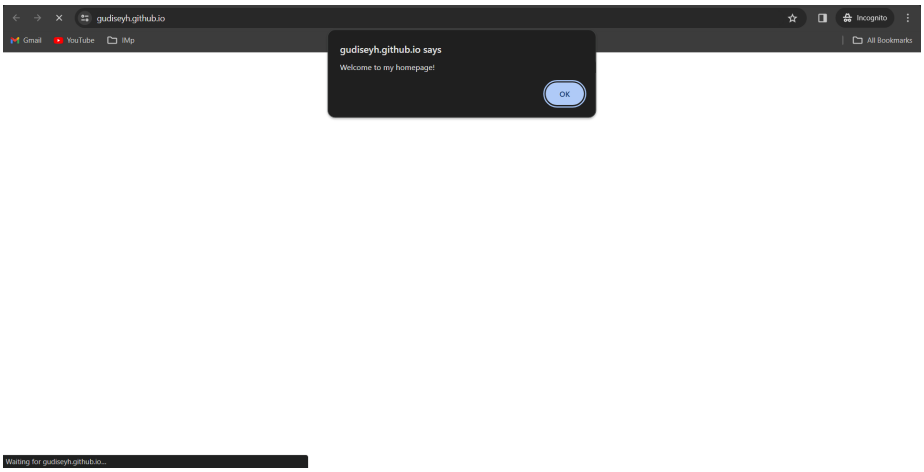

Figure 13: Alert Message shown on First Visit

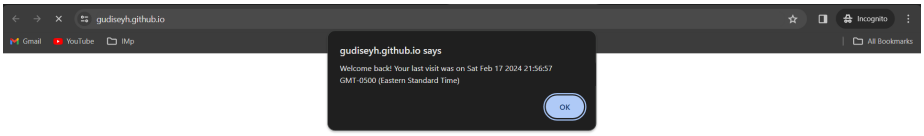

Figure 14: Alert Message shown on revisit