RAJALAKSHMI
ENGINEERING COLLEGE
An AUTONOMOUS Institution
Affiliated to ANNA UNIVERSITY, Chennai

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

IN

ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM CHENNAI-602105

2024 - 2025

CS23333 OBJECT ORIENTED PROGRAMMING USING JAVA LAB

A PROTFOLIO WEB APPLICATION

A MINI PROJECT REPORT

Submitted by

SURWEESH SP 231501165

VIGNESH BALA 231501180

SUJITH P 231501163

BONAFIDE CERTIFICATE

Certified that this project report “Personal Portfolio Website” is the Bonafide work of “SURWEESH SP (231501165), VIGESH BALA (231501180), SUJITH (231501163)” who carried out the project work under my supervision.

Submitted for the Practical Examination held on ______________

SIGNATURE

Mr.Devandra Rao,

Assistant Professor, AIML,

Rajalakshmi Engineering College, (Autonomous)

Thandalam, Chennai – 602 105

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ABSTRACT

Personal Portfolio Website Development

This project involves the creation of a personal portfolio website, designed to serve as an online platform for showcasing professional skills, past work, and personal achievements. The portfolio website will act as a dynamic, user-friendly space where potential employers, clients, or collaborators can get a comprehensive view of the creator's professional background, projects, and areas of expertise. It will integrate modern web development techniques, ensuring that the website is both aesthetically appealing and functionally robust, with a focus on responsive design, smooth navigation, and high user experience.

The website will consist of several sections, including an introductory home page, a detailed "About" section with personal and professional background, a comprehensive "Projects" section showcasing work samples, and a contact form for easy communication. The core design will leverage HTML5, CSS3, and JavaScript for the frontend, while additional tools like React (for dynamic content), Bootstrap (for responsive layouts), and optional backend technologies (Node.js with Express) will be employed for more complex interactions or content management.

Key features will include a visually striking home page with an interactive hero section, a professional layout with a consistent design language, and a well-organized project gallery, all while ensuring mobile and tablet optimization. The contact section will offer potential clients and employers an intuitive way to reach out, further enhancing the website's utility.

One of the primary goals of the project is to demonstrate proficiency in modern web technologies and user-centric design, while also establishing an easily navigable and aesthetically professional online presence. The project will also incorporate SEO best practices to improve visibility on search engines, ensuring that the site ranks well and is discoverable by a wider audience.

The finished website will be deployed on cloud platforms such as GitHub Pages, Netlify, or Vercel, ensuring seamless access and quick load times. It will also be continuously updated to reflect new projects and career milestones. In the future, additional features such as a blog system or advanced user interactions could be incorporated, allowing the portfolio to grow with the creator’s professional journey.

This project ultimately aims to provide a powerful tool for personal branding and professional outreach, combining technical expertise with creative presentation in a polished and effective digital showcase.

TABLE OF CONTENTS

1. Introduction

In this section, you will introduce the project, explaining the purpose of the personal portfolio website and its importance. Discuss the goals behind building such a website, such as showcasing skills, projects, and enhancing personal branding. Mention the target audience (potential clients, employers, collaborators) and why an online portfolio is essential in today's digital world. You may also briefly introduce the technologies used for building the website.

1.1 INTRODUCTION

The concept of a personal portfolio website has gained widespread importance in today's digitally-driven world, where an individual's online presence often serves as the first point of interaction with potential employers, clients, or collaborators. Unlike a traditional resume, which offers a static and limited view of one’s skills and experience, a portfolio website provides a dynamic platform for individuals to demonstrate their expertise through practical examples, engaging multimedia content, and detailed explanations of past projects and achievements. A portfolio website acts as a digital footprint, reflecting not only the skills and qualifications of the individual but also their personality, creativity, and professional style.

In fields such as web development, design, photography, and writing, having an online portfolio has become an indispensable tool for standing out in a competitive job market. With employers and clients frequently turning to the internet to find talent, a personal portfolio website offers a convenient and effective way for professionals to showcase their best work. The growing trend of remote work, freelancing, and digital entrepreneurship further emphasizes the need for individuals to establish a robust online presence that speaks to their skills and capabilities.

One of the primary advantages of a personal portfolio website is the flexibility it offers in terms of presentation. Unlike a printed resume, which is confined to a single page or a limited number of sections, a portfolio website can accommodate an unlimited amount of content. This includes detailed case studies, project galleries, blog posts, testimonials from clients or colleagues, and even a personal blog. A portfolio website allows individuals to curate their work in a way that best represents their unique abilities, interests, and achievements, providing visitors with a comprehensive and immersive experience.

Beyond showcasing work, a portfolio website serves as an invaluable networking tool. By integrating social media links, professional networks like LinkedIn, and contact forms, a personal portfolio website can foster meaningful connections with potential collaborators, employers, and clients. The contact form, in particular, allows website visitors to reach out directly, facilitating communication and inquiries about job opportunities, freelance work, or collaborations. This direct line of communication can significantly enhance an individual’s visibility and make them more approachable to those seeking their skills.

As the website will be built using modern web technologies, it will not only serve as a platform for displaying content but also as a demonstration of my proficiency in web development. From ensuring a responsive design that adapts to various screen sizes to implementing interactive features using JavaScript and integrating a back-end system for managing content, the website will be a true representation of my skills and expertise in web development.

1.2 Objectives

The development of a personal portfolio website serves multiple objectives, each aimed at creating an effective and professional online presence. The primary goals of this project can be outlined as follows:

***1. Showcase of Professional Work**:*

The primary objective of the website is to present a curated collection of my best work in a way that is accessible, visually appealing, and easy to navigate. This will include past web development projects, designs, writing samples, and other relevant work. By displaying these examples in a well-organized portfolio section, the website will demonstrate my proficiency in various areas and highlight the breadth and depth of my skill set. Each project will be accompanied by detailed descriptions and links to the live projects or their code repositories (e.g., GitHub).

***2. Demonstrate Technical Skills**:*

The portfolio website is not only a showcase for past work but also a demonstration of my technical abilities. The site will be built using a combination of modern web development technologies, including HTML5, CSS3, JavaScript, and potentially back-end technologies such as Node.js or Java. Through the use of interactive features, animations, and responsive design, the website will reflect my competence in both front-end and back-end development. Additionally, I will ensure that the website is optimized for performance, security, and accessibility, further showcasing my attention to detail.

*3. **Personal Branding**:*

A personal portfolio website serves as a tool for personal branding, allowing me to establish a unique online identity that aligns with my professional values and goals. The design, color scheme, and overall aesthetic of the website will reflect my personal style, while the content will convey my professional background, values, and aspirations. Through consistent and thoughtful design, I aim to create a site that is not only functional but also memorable, making a lasting impression on visitors and helping me stand out in a competitive market.

*4. **Provide Easy Contact Mechanism**:*

An essential objective of the website is to make it easy for potential clients, employers, or collaborators to contact me. The contact section will feature a simple yet effective contact form that allows users to send inquiries or requests directly. This will be accompanied by links to my social media profiles, such as LinkedIn and GitHub, enabling visitors to connect with me through various channels. The goal is to provide an accessible and straightforward way for people to reach out, whether for job opportunities, project collaborations, or freelance work.

*5. **Responsive and Mobile-Friendly Design**:*

A key objective of the project is to ensure that the portfolio website is fully responsive, meaning that it will provide an optimal browsing experience across a variety of devices, including desktops, tablets, and smartphones. Given the growing use of mobile devices to browse the web, it is essential that the website adapts seamlessly to different screen sizes. To achieve this, I will implement responsive design techniques using CSS media queries and flexible layout systems like Flexbox and Grid.

1.3 MODULES

- Sign Up
- Login
- Portfolio
- Updating Page

2. Software Description

Visual Studio Code (VS Code)

VSC is an open-source code editor developed by Microsoft that has gained widespread popularity among web developers due to its versatility, speed, and extensive features. It supports a wide range of programming languages, including HTML, CSS, JavaScript, Python, and many others. One of its standout features is its robust extension ecosystem, where developers can install plugins to add support for various languages, frameworks, and tools. Extensions like Prettier, ESLint, and GitLens help improve the development workflow by enabling auto-formatting, error checking, and version control integration.

VS Code offers an intuitive user interface that is highly customizable, allowing users to adjust themes, layouts, and keyboard shortcuts to suit their personal preferences. It also provides built-in Git integration, allowing developers to easily manage version control directly within the editor. The integrated terminal allows for quick command execution without leaving the editor, which enhances productivity.

Its support for real-time collaboration, through features like Live Share, enables teams to work together on the same codebase simultaneously, regardless of their physical location. With its lightweight nature, powerful features, and continuous updates, Visual Studio Code remains one of the most preferred development tools for web development.

LANGUAGES

The development of a portfolio website typically involves a combination of programming languages that work together to provide both structure, style, interactivity, and back-end functionality. The following languages are commonly used in the development process:

2.2.1 HTML (Hypertext Markup Language)

HTML is the foundational language for creating web pages and web applications. It provides the basic structure of a webpage, using a system of tags to define elements like headings, paragraphs, images, links, forms, and more. HTML allows developers to organize content into a meaningful hierarchy, making it easier for users to navigate and interact with a site. For example, HTML tags like <header>, <footer>, <section>, and <article> help structure content in a logical manner, improving accessibility and search engine optimization (SEO).

HTML5, the latest version of HTML, introduces semantic elements that provide better clarity and meaning to the structure of web pages. Elements like <nav>, <article>, and <footer> help define different parts of a web page in a more descriptive manner, which is crucial for SEO and accessibility purposes. HTML also allows developers to embed multimedia content, such as images, videos, and audio, directly into web pages, making it a versatile language for building content-driven websites.

The HTML code works alongside CSS (Cascading Style Sheets) and JavaScript to ensure that web pages not only have structure but also appear visually appealing and interact in a dynamic way. HTML is often the first step in the web development process, laying the foundation for the rest of the website’s design and functionality.

CSS (Cascading Style Sheets)

CSS is a stylesheet language used to control the presentation and layout of HTML elements on a webpage. It defines how HTML elements should appear visually, including their color, size, position, font styles, and spacing. CSS provides a powerful way to enhance the look and feel of a website by controlling elements such as the page layout, typography, and responsiveness.

One of the most valuable features of CSS is its ability to create responsive web designs. By using CSS media queries, developers can apply different styles based on the user’s screen size, ensuring that websites look great on any device, from desktop computers to smartphones and tablets. CSS frameworks like **Bootstrap** or **Tailwind CSS** are commonly used to streamline the design process by providing pre-built, customizable design components that follow responsive principles.

CSS also supports animations and transitions, allowing developers to add dynamic visual effects to websites. This can improve user engagement by providing smooth transitions when interacting with elements like buttons, images, or menus. In combination with HTML, CSS is essential for turning a basic webpage into an aesthetically pleasing, user-friendly site.

JavaScript

JavaScript is a versatile and powerful programming language that enables developers to add interactivity and dynamic functionality to a website. It allows for the manipulation of HTML elements and the creation of complex behaviors on a webpage, such as form validation, user input handling, dynamic content updates, and interactive UI elements like sliders, dropdowns, and modals. JavaScript is commonly used to create Single Page Applications (SPAs), where users can interact with a site without needing to refresh the page.

JavaScript runs in the browser, which means it can directly interact with HTML and CSS to update the user interface in real-time. This enables developers to build responsive and interactive websites that can respond instantly to user actions. For example, JavaScript is used to validate form data before submission, display interactive maps, handle user clicks, and load new content without reloading the entire page.

JavaScript also enables asynchronous operations, which means web pages can fetch data from servers in the background using technologies like **AJAX** (Asynchronous JavaScript and XML) and **Fetch API**. This is crucial for creating fast and efficient websites, as it minimizes page load times and enhances user experience.

Popular JavaScript frameworks and libraries, such as **React**, **Vue.js**, and **Angular**, make it easier to develop complex, component-based user interfaces with reusable code. By using JavaScript in combination with HTML and CSS, developers can create rich, interactive websites that provide seamless user experiences.

MySQL

MySQL is an open-source relational database management system (RDBMS) that is widely used for managing and storing data in web applications. It uses Structured Query Language (SQL) to interact with databases, allowing developers to retrieve, update, insert, and delete data efficiently. MySQL is an essential component in full-stack web development, particularly for applications that require data persistence, such as portfolio websites that may include a contact form, blog, or user login system.

In the context of a portfolio website, MySQL can be used to store various types of dynamic content, such as user comments, blog posts, or portfolio projects. The relational model of MySQL ensures that data is organized into tables with defined relationships, making it easy to maintain and retrieve information. For example, a portfolio website could have tables for "Projects," "Users," and "Contact Submissions," with foreign key relationships ensuring data integrity.

MySQL is highly scalable and performs well even with large datasets, making it a great choice for websites that anticipate growth. It is often used in combination with server-side programming languages like PHP, Python, or Node.js to create dynamic websites that interact with the database in real-time. For instance, when a user submits a contact form on the portfolio website, the data can be stored in the MySQL database for future reference.

3. REQUIREMENTS AND ANALYSIS\

3.1 Requirement Specification

The requirement specification is a critical part of the software development process, as it provides a clear and detailed description of the functional and non-functional requirements for the project. In the context of building a personal portfolio website, the requirements will focus on ensuring the website's performance, user experience, functionality, and scalability.

Functional Requirements:

1. **User Interface (UI):** The website should have a clean, modern, and responsive design. It must provide key sections like the "Home" page, "About Me," "Portfolio," "Blog" (optional), and "Contact" pages. Each section should be accessible from the main navigation menu.
2. **Portfolio Showcase:** A dynamic portfolio page where visitors can view past projects. Each project entry should include an image, a brief description, and a link to the live project or its GitHub repository.
3. **Contact Form:** A functional contact form where visitors can submit inquiries or feedback. The form should include fields for the visitor's name, email, and message, with validation to ensure proper data entry.
4. **SEO Optimization:** The website must be designed with SEO best practices in mind, ensuring that all pages are search engine-friendly, with appropriate meta tags, headings, and alt attributes for images.
5. **Accessibility:** The website should be accessible to all users, including those with disabilities, adhering to WCAG (Web Content Accessibility Guidelines) 2.1 standards for content readability, keyboard navigation, and screen reader compatibility.
6. **Performance Optimization:** The website must load quickly, with optimized images, minimized scripts, and caching strategies to improve performance.

Non-Functional Requirements:

1. **Scalability:** The website should be scalable to accommodate future updates, such as adding new projects or a blog section.
2. **Responsiveness:** The website should work seamlessly on all devices, including desktops, tablets, and smartphones.
3. **Security:** The website must ensure user data is protected, especially for any form submissions.

3.2 Hardware and Software Requirements

Hardware Requirements:

1. **Client Device:** The website should be accessible on devices with standard web browsers such as Chrome, Firefox, Safari, and Edge. This includes personal computers, laptops, tablets, and smartphones with internet access.
2. **Server:** The website can be hosted on a standard web server, such as Apache or Nginx, with sufficient bandwidth to handle expected traffic. For dynamic features like the contact form, a basic server with PHP or Node.js support will be required.

Software Requirements:

1. **Development Tools:**
   - **Visual Studio Code:** An integrated development environment (IDE) for writing and editing HTML, CSS, JavaScript, and server-side scripts.
   - **Browser:** A modern web browser (e.g., Google Chrome) for testing and debugging.
   - **Database Management:** MySQL (or any relational database management system) to store dynamic content like form submissions or blog posts.
2. **Languages and Frameworks:**
   - **HTML5, CSS3, JavaScript:** Core front-end languages for creating the structure, style, and interactivity of the website.
   - **Node.js or PHP (for back-end):** If required, for handling server-side processes such as contact form submissions.
   - **MySQL:** For storing and managing dynamic data, if the portfolio includes features like a blog or user comments.
3. **Web Hosting Service:** A reliable hosting provider to store the website files and ensure high availability. Options include shared hosting, VPS hosting, or cloud platforms like AWS, Heroku, or Netlify.

4. PROGRAM CODE

HTML SIGN UP

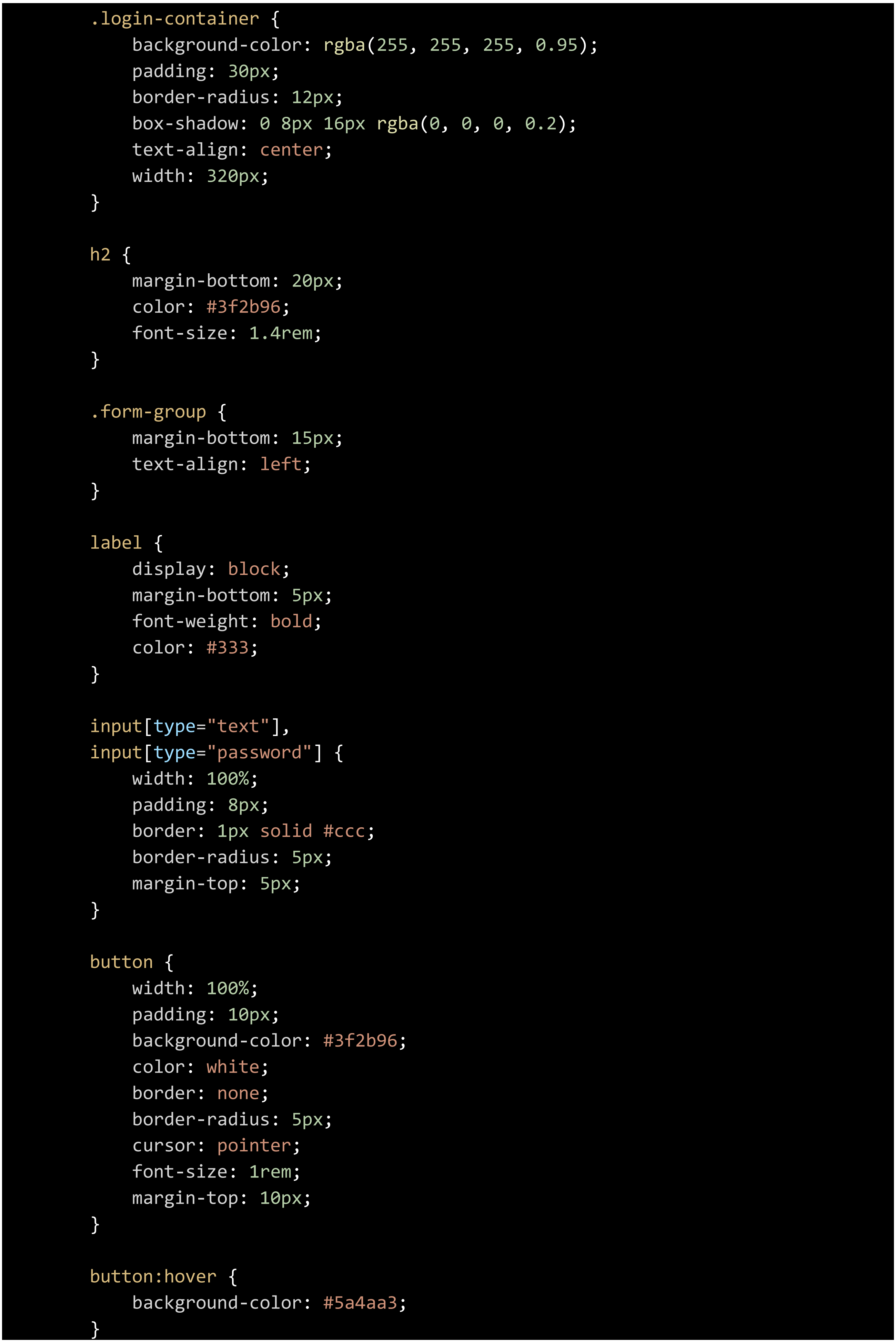
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Portfolio - Register</title>
    <style>
        /* Set background with a calming gradient */
        body {
            background: linear-gradient( #020202);
            margin: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            font-family: 'Arial', sans-serif;
        }

        .register-container {
            background-color: rgba(255, 255, 255, 0.95);
            padding: 20px;
            border-radius: 12px;
            box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);
            text-align: center;
            width: 320px;
            position: relative;
        }

        .profile-pic-container {
            margin-bottom: 15px;
        }

        .profile-pic {
            width: 100px;
            height: 100px;
            border-radius: 50%;
            border: 2px solid #3f2b96;
            object-fit: cover;
            background-color: #f0f0f0;
            display: block;
            margin: 0 auto;
        }

        .upload-btn {
```

```css
        display: inline-block;
        margin-top: 8px;
        background-color: #3f2b96;
        color: white;
        padding: 5px 10px;
        font-size: 12px;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        text-align: center;
    }

    .upload-btn:hover {
        background-color: #5a4aa3;
    }

    input[type="file"] {
        display: none;
    }

    h2 {
        color: #3f2b96;
        font-size: 1.4rem;
        margin-bottom: 15px;
    }

    .form-group {
        margin-bottom: 15px;
        text-align: left;
    }

    label {
        display: block;
        margin-bottom: 5px;
        font-weight: bold;
        color: #333;
    }

    input[type="text"],
    input[type="email"],
    input[type="password"] {
        width: 100%;
        padding: 8px;
        border: 1px solid #ccc;
        border-radius: 4px;
        box-sizing: border-box;
    }

    button {
        width: 100%;
        padding: 10px;
```

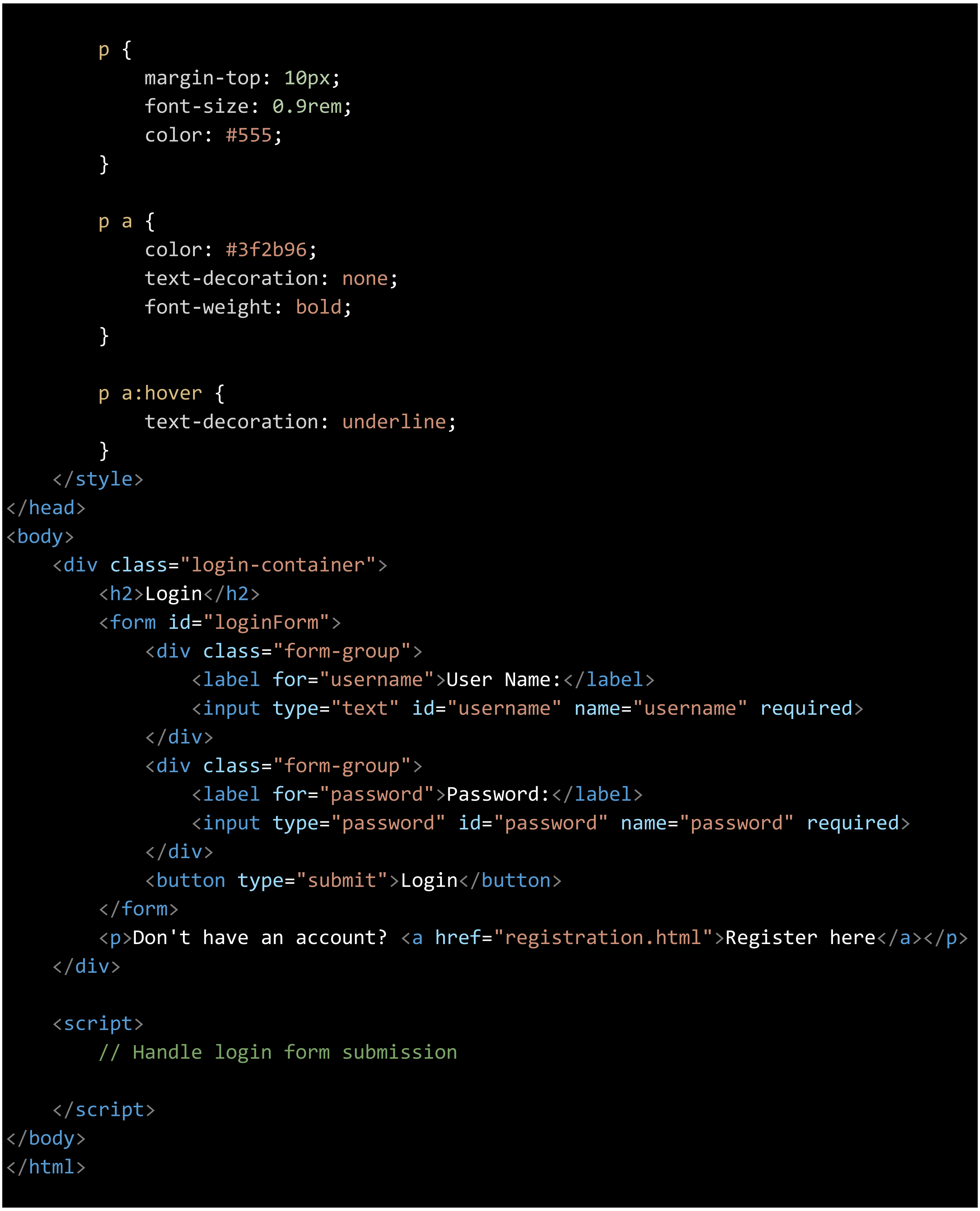
```
            background-color: #3f2b96;
            color: white;
            border: none;
            border-radius: 4px;
            font-size: 1rem;
            cursor: pointer;
            margin-top: 10px;
        }

        button:hover {
            background-color: #5a4aa3;
        }

        p {
            margin-top: 10px;
            font-size: 0.9rem;
            color: #555;
        }

        p a {
            color: #3f2b96;
            text-decoration: none;
            font-weight: bold;
        }

        p a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
    <div class="register-container">
        <!-- Profile Picture Upload Section -->
        <div class="profile-pic-container">
            <img src="default-profile.png" id="profilePic" class="profile-pic">
            <label for="profileUpload" class="upload-btn">Upload Picture</label>
            <input type="file" id="profileUpload" accept="image/*">
        </div>

        <h2>Register for Portfolio</h2>
        <form id="registerForm">
            <div class="form-group">
                <label for="email">Email:</label>
                <input type="email" id="email" name="email" required>
            </div>
            <div class="form-group">
                <label for="username">Username:</label>
                <input type="text" id="username" name="username" required>
            </div>
            <div class="form-group">
                <label for="password">Password:</label>
                <input type="password" id="password" name="password" required>
            </div>
            <button type="submit">Register</button>
        </form>
        <p>Already have an account? <a href="signin.html">Login here</a></p>
    </div>

    <script>
        const profilePic = document.getElementById('profilePic');
        const profileUpload = document.getElementById('profileUpload');

        // Event listener to display the image preview when a user selects an image
        profileUpload.addEventListener('change', (event) => {
            const file = event.target.files[0];
            if (file) {
                const reader = new FileReader();
                reader.onload = function(e) {
                    profilePic.src = e.target.result; // Set the image source to the selected file
                };
                reader.readAsDataURL(file);
            }
        });
    </script>
</body>
</html>
```

SIGN IN

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Stock Market and Prediction - Login</title>
    <style>
        /* Set a calming gradient background */
        body {
            background: linear-gradient(#030303);
            margin: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            font-family: 'Arial', sans-serif;
        }
```



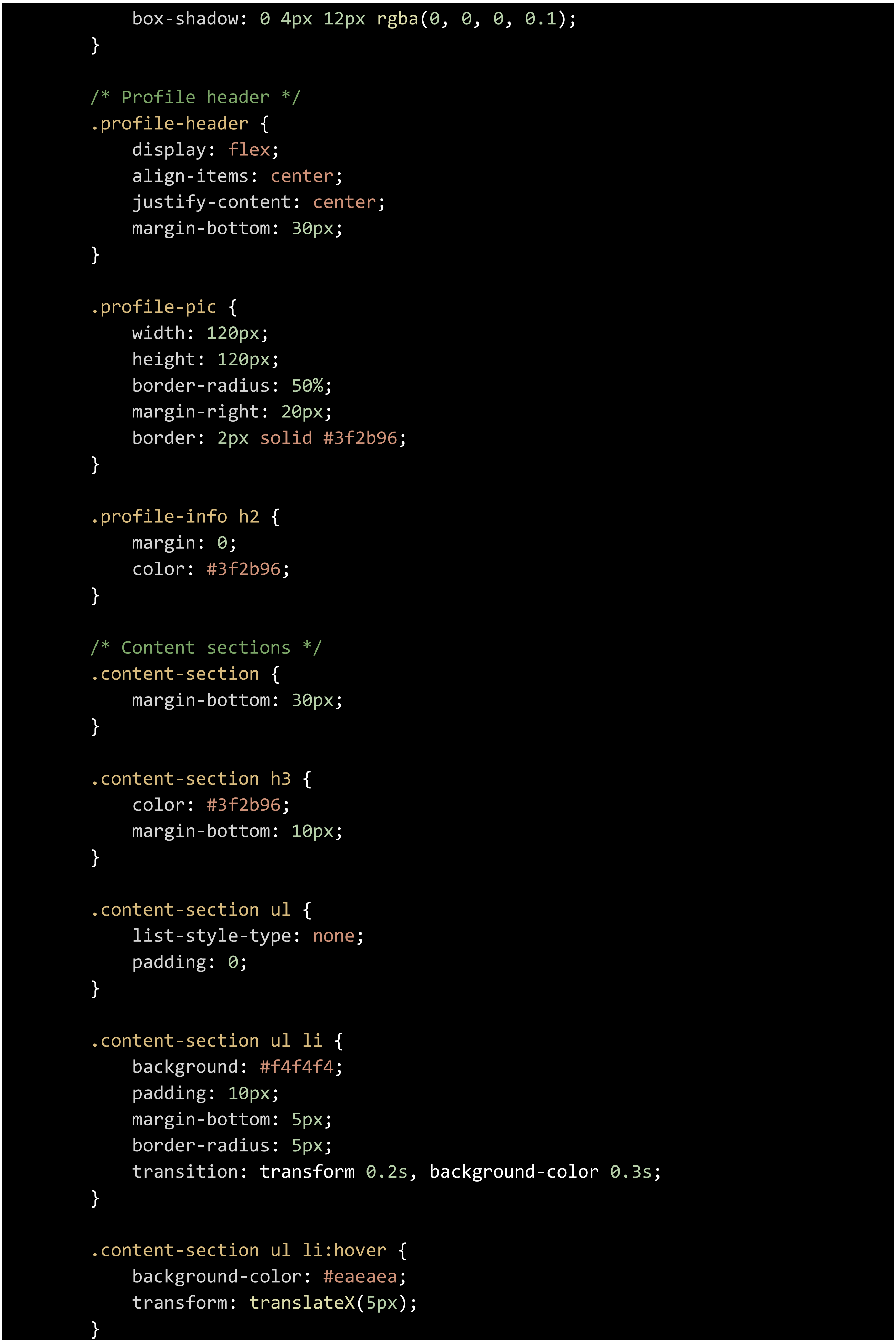
```
        .login-container {
            background-color: rgba(255, 255, 255, 0.95);
            padding: 30px;
            border-radius: 12px;
            box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);
            text-align: center;
            width: 320px;
        }

        h2 {
            margin-bottom: 20px;
            color: #3f2b96;
            font-size: 1.4rem;
        }

        .form-group {
            margin-bottom: 15px;
            text-align: left;
        }

        label {
            display: block;
            margin-bottom: 5px;
            font-weight: bold;
            color: #333;
        }

        input[type="text"],
        input[type="password"] {
            width: 100%;
            padding: 8px;
            border: 1px solid #ccc;
            border-radius: 5px;
            margin-top: 5px;
        }

        button {
            width: 100%;
            padding: 10px;
            background-color: #3f2b96;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 1rem;
            margin-top: 10px;
        }

        button:hover {
            background-color: #5a4aa3;
        }
```



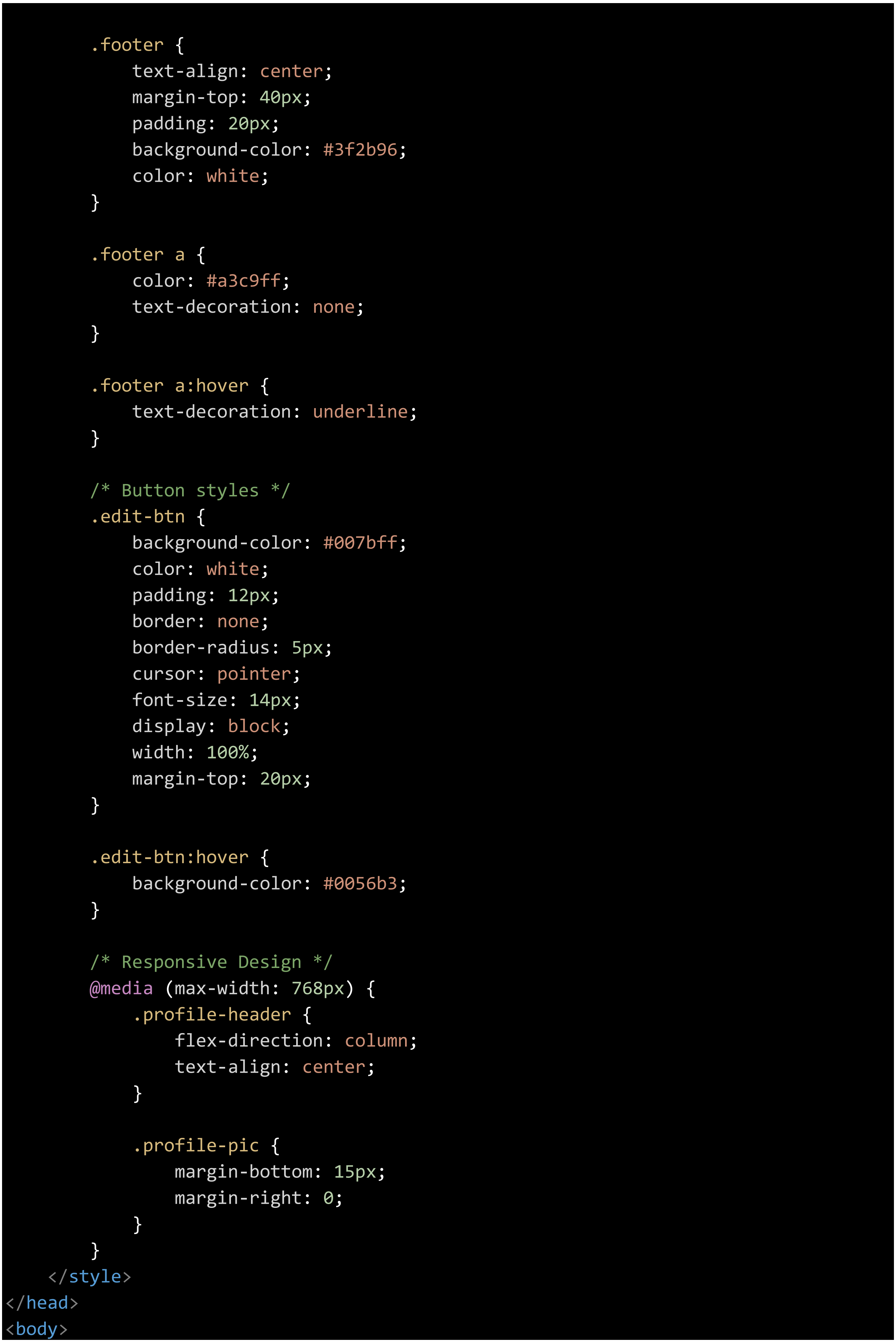
```
        p {
            margin-top: 10px;
            font-size: 0.9rem;
            color: #555;
        }

        p a {
            color: #3f2b96;
            text-decoration: none;
            font-weight: bold;
        }

        p a:hover {
            text-decoration: underline;
        }
    </style>
</head>
<body>
    <div class="login-container">
        <h2>Login</h2>
        <form id="loginForm">
            <div class="form-group">
                <label for="username">User Name:</label>
                <input type="text" id="username" name="username" required>
            </div>
            <div class="form-group">
                <label for="password">Password:</label>
                <input type="password" id="password" name="password" required>
            </div>
            <button type="submit">Login</button>
        </form>
        <p>Don't have an account? <a href="registration.html">Register here</a></p>
    </div>

    <script>
        // Handle login form submission

    </script>
</body>
</html>
```

Home PAGE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Portfolio</title>
    <style>
        /* Basic styles */
        body {
            font-family: 'Arial', sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 0;
        }

        /* Navigation bar */
        .navbar {
            display: flex;
            justify-content: space-around;
            align-items: center;
            background-color: #3f2b96;
            color: white;
            padding: 15px;
            position: sticky;
            top: 0;
            z-index: 1000;
        }

        .navbar a {
            color: white;
            text-decoration: none;
            font-weight: bold;
            font-size: 16px;
            padding: 8px 16px;
        }

        .navbar a:hover {
            background-color: #5643c2;
            border-radius: 5px;
        }

        /* Main container */
        .portfolio-container {
            background-color: #fff;
            width: 100%;
            max-width: 900px;
            margin: 20px auto;
            padding: 30px;
            border-radius: 10px;
```



```
            box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
        }

        /* Profile header */
        .profile-header {
            display: flex;
            align-items: center;
            justify-content: center;
            margin-bottom: 30px;
        }

        .profile-pic {
            width: 120px;
            height: 120px;
            border-radius: 50%;
            margin-right: 20px;
            border: 2px solid #3f2b96;
        }

        .profile-info h2 {
            margin: 0;
            color: #3f2b96;
        }

        /* Content sections */
        .content-section {
            margin-bottom: 30px;
        }

        .content-section h3 {
            color: #3f2b96;
            margin-bottom: 10px;
        }

        .content-section ul {
            list-style-type: none;
            padding: 0;
        }

        .content-section ul li {
            background: #f4f4f4;
            padding: 10px;
            margin-bottom: 5px;
            border-radius: 5px;
            transition: transform 0.2s, background-color 0.3s;
        }

        .content-section ul li:hover {
            background-color: #eaeaea;
            transform: translateX(5px);
        }
```



```
        .footer {
            text-align: center;
            margin-top: 40px;
            padding: 20px;
            background-color: #3f2b96;
            color: white;
        }

        .footer a {
            color: #a3c9ff;
            text-decoration: none;
        }

        .footer a:hover {
            text-decoration: underline;
        }

        /* Button styles */
        .edit-btn {
            background-color: #007bff;
            color: white;
            padding: 12px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 14px;
            display: block;
            width: 100%;
            margin-top: 20px;
        }

        .edit-btn:hover {
            background-color: #0056b3;
        }

        /* Responsive Design */
        @media (max-width: 768px) {
            .profile-header {
                flex-direction: column;
                text-align: center;
            }

            .profile-pic {
                margin-bottom: 15px;
                margin-right: 0;
            }
        }
    </style>
</head>
<body>
```

```html
    <!-- Navigation Bar -->
    <div class="navbar">
        <a href="#profile">Profile</a>
        <a href="#about">About</a>
        <a href="#skills">Skills</a>
        <a href="#projects">Projects</a>
        <a href="#experiences">Experiences</a>
        <a href="#contact">Contact</a>
    </div>

    <!-- Portfolio Content -->
    <div class="portfolio-container">
        <div class="profile-header" id="profile">
            <img id="profilePic" src="default-profile.png" class="profile-pic">
            <div class="profile-info">
                <h2 id="profileName">SURWEESH SP</h2>
                <p id="profileBio">A passionate developer with experience in
building dynamic websites and applications.</p>
            </div>
        </div>

        <!-- About Section -->
        <div class="content-section" id="about">
            <h3>About Me</h3>
            <p id="aboutMe">
                Hello! I'm Surweesh SP, a dedicated developer with a knack for
solving problems and creating engaging web applications.
                My journey in technology started with a deep curiosity for how
things work, which soon turned into a passion for building innovative solutions.
                Outside work, I enjoy exploring new places, reading, and honing my
skills in emerging technologies.
            </p>
        </div>

        <!-- Skills Section -->
        <div class="content-section" id="skills">
            <h3>Skills</h3>
            <ul id="skillsList">
                <li>JavaScript</li>
                <li>HTML & CSS</li>
                <li>Node.js</li>
            </ul>
        </div>

        <!-- Projects Section -->
        <div class="content-section" id="projects">
            <h3>Projects</h3>
            <ul id="projectsList">
                <li>Portfolio Website</li>
                <li>Task Management App</li>
            </ul>
        </div>

        <!-- Experiences Section -->
        <div class="content-section" id="experiences">
            <h3>Experiences</h3>
            <ul id="experiencesList">
                <li>Web Developer at ABC Corp</li>
                <li>Freelance Developer</li>
            </ul>
        </div>

        <!-- Documents Section -->
        <div class="content-section" id="documents">
            <h3>Documents</h3>
            <ul id="documentsList">
                <!-- Dynamically loaded documents -->
            </ul>
        </div>

        <!-- Links Section -->
        <div class="content-section" id="links">
            <h3>Links</h3>
            <ul id="linksList">
                <!-- Dynamically loaded links -->
            </ul>
        </div>

        <!-- Contact Section -->
        <div class="content-section" id="contact">
            <h3>Contact Information</h3>
            <p id="contactEmail">Email: surweeshsp@example.com</p>
            <p id="contactPhone">Phone: +123 456 7890</p>
        </div>

        <button class="edit-btn" onclick="window.location.href =
'update.html'">Edit Portfolio</button>
    </div>

    <!-- Footer -->
    <div class="footer">
        <p>&copy; 2024 Portfolio Website | <a href="#">Privacy Policy</a></p>
    </div>

    <script>
        // Function to dynamically load portfolio data from localStorage
        window.onload = function () {
            // Load Profile Picture
            if (localStorage.getItem('profilePic')) {
                document.getElementById('profilePic').src =
localStorage.getItem('profilePic');
            }
            // Load Name
            if (localStorage.getItem('name')) {
                document.getElementById('profileName').textContent =
localStorage.getItem('name');
            }

            // Load Bio
            if (localStorage.getItem('bio')) {
                document.getElementById('profileBio').textContent =
localStorage.getItem('bio');
            }

            // Load About Me
            if (localStorage.getItem('aboutMe')) {
                document.getElementById('aboutMe').textContent =
localStorage.getItem('aboutMe');
            }

            // Load Skills
            if (localStorage.getItem('skills')) {
                const skills = JSON.parse(localStorage.getItem('skills'));
                document.getElementById('skillsList').innerHTML = skills.map(skill
=> `<li>${skill}</li>`).join('');
            }

            // Load Projects
            if (localStorage.getItem('projects')) {
                const projects = JSON.parse(localStorage.getItem('projects'));
                document.getElementById('projectsList').innerHTML =
projects.map(project => `<li>${project}</li>`).join('');
            }

            // Load Experiences
            if (localStorage.getItem('experiences')) {
                const experiences =
JSON.parse(localStorage.getItem('experiences'));
                document.getElementById('experiencesList').innerHTML =
experiences.map(exp => `<li>${exp}</li>`).join('');
            }

            // Load Documents
            if (localStorage.getItem('documents')) {
                const documents = JSON.parse(localStorage.getItem('documents'));
                document.getElementById('documentsList').innerHTML = documents
                    .map(doc => `<li><a href="${doc}"
target="_blank">${getFileName(doc)}</a></li>`)
                    .join('');
            }

            // Load Links
            if (localStorage.getItem('links')) {
                const links = JSON.parse(localStorage.getItem('links'));
                document.getElementById('linksList').innerHTML = links
                    .map(link => `<li><a href="${link}"
target="_blank">${link}</a></li>`)
                    .join('');
            }

            // Contact Email
            if (localStorage.getItem('email')) {
                document.getElementById('contactEmail').textContent = `Email:
${localStorage.getItem('email')}`;
            }

            // Contact Phone
            if (localStorage.getItem('phone')) {
                document.getElementById('contactPhone').textContent = `Phone:
${localStorage.getItem('phone')}`;
            }
        };

        // Helper function to extract the file name from a document URL
        function getFileName(url) {
            return url.split('/').pop(); // Gets the last part of the URL
        }
    </script>
</body>
</html>
```

UPDATE

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit Portfolio</title>
    <style>
        body {
            font-family: 'Arial', sans-serif;
            background-color: #f4f4f4;
            margin: 0;
            padding: 20px;
        }

        .form-container {
            background-color: #fff;
            max-width: 800px;
            padding: 30px;
            border-radius: 10px;
            box-shadow: 0 4px 12px rgba(0, 0, 0, 0.1);
            margin: 0 auto;
        }

        .form-group {
            margin-bottom: 20px;
        }

        .form-group label {
            font-size: 16px;
            margin-bottom: 5px;
        }

        .form-group input, .form-group textarea {
            width: 100%;
            padding: 10px;
            font-size: 14px;
            border: 1px solid #ccc;
            border-radius: 5px;
        }

        .form-group img {
            display: block;
            margin-top: 10px;
            max-width: 100px;
            border-radius: 50%;
        }

        .form-group button {
            padding: 10px;
            background-color: #007bff;
            color: white;
            border: none;
            border-radius: 5px;
            cursor: pointer;
            font-size: 14px;
        }

        .form-group button:hover {
            background-color: #0056b3;
        }

        .form-group textarea {
            resize: vertical;
        }
    </style>
</head>
<body>
    <div class="form-container">
```

```
        <h2>Edit Your Portfolio</h2>
        <form id="portfolioForm">
            <div class="form-group">
                <label for="profileName">Name</label>
                <input type="text" id="profileName" name="profileName"
placeholder="Enter your full name">
            </div>

            <div class="form-group">
                <label for="profileBio">Bio</label>
                <textarea id="profileBio" name="profileBio" placeholder="Write a
short bio about yourself"></textarea>
            </div>

            <div class="form-group">
                <label for="profilePic">Profile Picture</label>
                <input type="file" id="profilePic" name="profilePic"
accept="image/*" onchange="handleFileSelect(event)">
                <img id="profilePicPreview" src="" alt="Profile Preview"
style="display: none;">
            </div>

            <div class="form-group">
                <label for="skills">Skills (comma-separated)</label>
                <input type="text" id="skills" name="skills" placeholder="e.g.,
JavaScript, HTML, CSS">
            </div>

            <div class="form-group">
                <label for="projects">Projects (comma-separated)</label>
                <input type="text" id="projects" name="projects" placeholder="e.g.,
Portfolio Website, Task Manager">
            </div>

            <div class="form-group">
                <label for="experiences">Experiences (comma-separated)</label>
                <input type="text" id="experiences" name="experiences"
placeholder="e.g., Web Developer, Freelancer">
            </div>

            <div class="form-group">
                <label for="interests">Personal Interests</label>
                <input type="text" id="interests" name="interests"
placeholder="e.g., Coding, Reading, Traveling">
            </div>

            <div class="form-group">
                <label for="email">Email</label>
                <input type="email" id="email" name="email" placeholder="Enter your
email address">
            </div>
            <div class="form-group">
                <label for="phone">Phone</label>
                <input type="text" id="phone" name="phone" placeholder="Enter your 
phone number">
            </div>

            <div class="form-group">
                <button type="button" onclick="savePortfolio()">Save 
Portfolio</button>
            </div>
            <div class="form-group">
                <label for="documents">Documents (PDF, Word, etc.)</label>
                <input type="file" id="documents" name="documents" 
accept=".pdf,.doc,.docx,.txt" multiple onchange="handleDocumentsSelect(event)">
                <div id="documentsPreview" class="file-preview"></div>
            </div>

            <div class="form-group">
                <label for="links">Links (Comma separated)</label>
                <input type="text" id="links" name="links" placeholder="e.g., 
https://example.com, https://anotherlink.com">
            </div>
        </form>
    </div>

    <script>
        // Load existing data into form fields when the page loads
        window.onload = function() {
            // Existing fields are loaded here ...

            // Load documents
            if (localStorage.getItem('documents')) {
                const documents = JSON.parse(localStorage.getItem('documents'));
                const documentPreview = 
document.getElementById('documentsPreview');
                documentPreview.innerHTML = documents
                    .map(doc => `<p><a href="${doc}" 
target="_blank">${getFileName(doc)}</a></p>`)
                    .join('');
            }

            // Load links
            if (localStorage.getItem('links')) {
                document.getElementById('links').value = 
JSON.parse(localStorage.getItem('links')).join(', ');
            }
        };

        // Handle document file upload
        function handleDocumentsSelect(event) {
            const files = event.target.files;
            if (files.length > 0) {
                const fileUrls = Array.from(files).map(file => {
                    const fileUrl = URL.createObjectURL(file);
                    return fileUrl;
                });
                localStorage.setItem('documents', JSON.stringify(fileUrls));

                const documentPreview = 
document.getElementById('documentsPreview');
                documentPreview.innerHTML = fileUrls
                    .map(doc => `<p><a href="${doc}" 
target="_blank">${getFileName(doc)}</a></p>`)
                    .join('');
            }
        }

        // Helper function to extract the file name from a URL
        function getFileName(url) {
            return url.split('/').pop();
        }

        // Save the updated portfolio data to localStorage
        function savePortfolio() {
            // Get the values from the form
            const name = document.getElementById('profileName').value;
            const bio = document.getElementById('profileBio').value;
            const skills = 
document.getElementById('skills').value.split(',').map(item => item.trim());
            const projects = 
document.getElementById('projects').value.split(',').map(item => item.trim());
            const experiences = 
document.getElementById('experiences').value.split(',').map(item => item.trim());
            const interests = document.getElementById('interests').value;
            const email = document.getElementById('email').value;
            const phone = document.getElementById('phone').value;
            const links = 
document.getElementById('links').value.split(',').map(item => item.trim());

            // Save to localStorage
            localStorage.setItem('name', name);
            localStorage.setItem('bio', bio);
            localStorage.setItem('skills', JSON.stringify(skills));
            localStorage.setItem('projects', JSON.stringify(projects));
            localStorage.setItem('experiences', JSON.stringify(experiences));
            localStorage.setItem('interests', interests);
            localStorage.setItem('email', email);
            localStorage.setItem('phone', phone);
            localStorage.setItem('links', JSON.stringify(links));

            alert("Portfolio updated successfully!");
            // Redirect to the main portfolio page
            window.location.href = 'home.html'; // Update to your actual portfolio 
page URL
        }
    </script>
</body>
</html>
```

BACKEND JAVA

POM.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" 
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.4.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.portfollioweb</groupId>
    <artifactId>porfolioweb</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>porfolioweb</name>
    <description>Demo project for Spring Boot</description>
    <url/>
    <licenses>
        <license/>
    </licenses>
    <developers>
        <developer/>
    </developers>
    <scm>
        <connection/>
        <developerConnection/>
        <tag/>
        <url/>
    </scm>
    <properties>
        <java.version>17</java.version>
    </properties>
    <dependencies>
```

```xml
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>


    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>3.4.0</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.portfollioweb</groupId>
    <artifactId>porfolioweb</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>porfolioweb</name>
    <description>Demo project for Spring Boot</description>
    <url/>
    <licenses>
        <license/>
    </licenses>
    <developers>
        <developer/>
    </developers>
    <scm>
        <connection/>
        <developerConnection/>
        <tag/>
        <url/>
    </scm>
    <properties>
        <java.version>17</java.version>
    </properties>
    <dependencies>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>


    <dependency>
        <groupId>com.mysql</groupId>
        <artifactId>mysql-connector-j</artifactId>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.projectlombok</groupId>
        <artifactId>lombok</artifactId>
        <optional>true</optional>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
        </plugins>
    </build>

</project>
```

USERCONTROLLER.JAVA

```
package com.portfollioweb.porfolioweb.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.portfollioweb.porfolioweb.model.User;
import com.portfollioweb.porfolioweb.service.UserService;

@RestController
@RequestMapping("/api/users")
@CrossOrigin(origins = "*")
public class UserController
{

    @Autowired
    private UserService userService;

    @PostMapping("/register")
    public ResponseEntity<String> registerUser(@RequestBody User user) {
        try {
            userService.saveUser(user);
            return ResponseEntity.ok("User registered successfully!");
        } catch (Exception e) {
            return ResponseEntity.status(500).body("Error: " + e.getMessage());
        }
    }
}
```

USER REPO.JAVA

```
package com.portfollioweb.porfolioweb.repository;

import org.springframework.data.jpa.repository.JpaRepository;

import com.portfollioweb.porfolioweb.model.User;

public interface UserRepository extends JpaRepository<User, Long> {
}
```

APPLICATION PROPERTIES.JAVA

```
spring.application.name=portfolioweb

spring.datasource.url=jdbc:mysql://localhost:3306/portfolio?useSSL=false&allowPubli
cKeyRetrieval=true&serverTimezone=UTC
spring.datasource.username=root
spring.datasource.password=vignesh17
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

spring.jpa.database-platform=org.hibernate.dialect.MySQLDialect
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
```

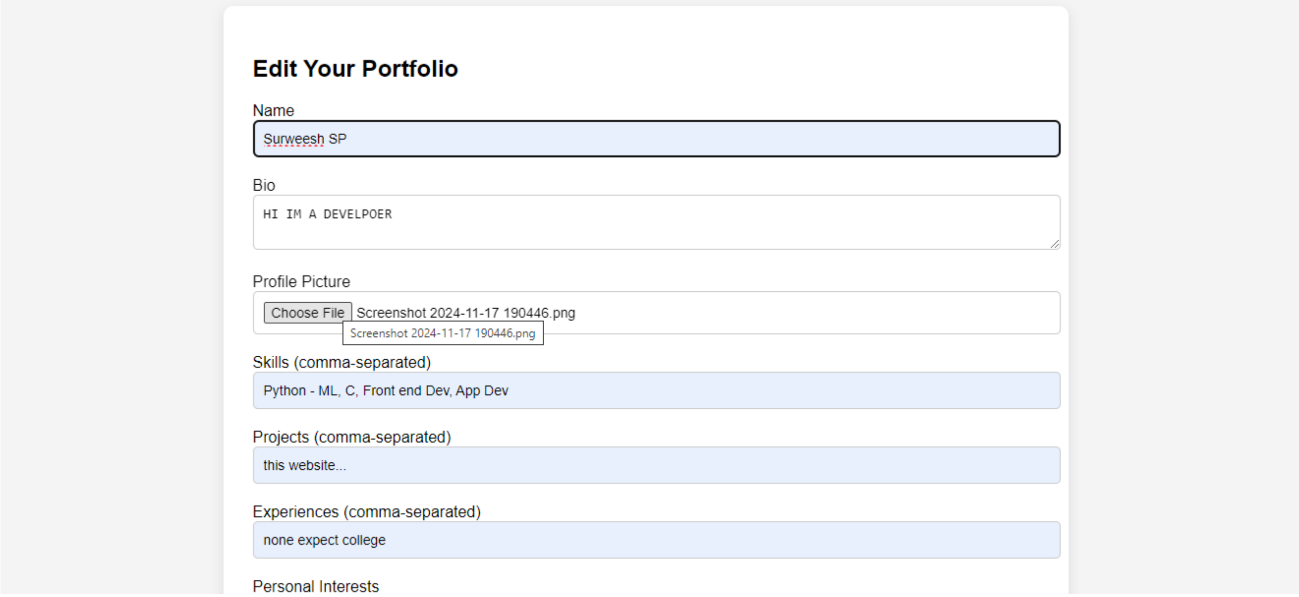
USER.JAVA

```
package com.portfollioweb.porfolioweb.model;

import jakarta.persistence.Entity;
import jakarta.persistence.GeneratedValue;
import jakarta.persistence.GenerationType;
import jakarta.persistence.Id;
import jakarta.persistence.Table;

@Entity
@Table(name = "users")
public class User {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    private String email;
    private String username;
    private String password;


    public Long getId() {
        return id;
    }

    public void setId(Long id) {
        this.id = id;
    }

    public String getEmail() {
        return email;
    }

    public void setEmail(String email) {
        this.email = email;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

USERSERVICE.JAVA

```
package com.portfollioweb.porfolioweb.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.portfollioweb.porfolioweb.model.User;
import com.portfollioweb.porfolioweb.repository.UserRepository;

@Service
public class UserService {

    @Autowired
    private UserRepository userRepository;

    public User saveUser(User user) {
        return userRepository.save(user);
    }
}
```

5. RESULT



Edit Your Portfolio

Name
Surweesh SP

Bio
HI IM A DEVELPOER

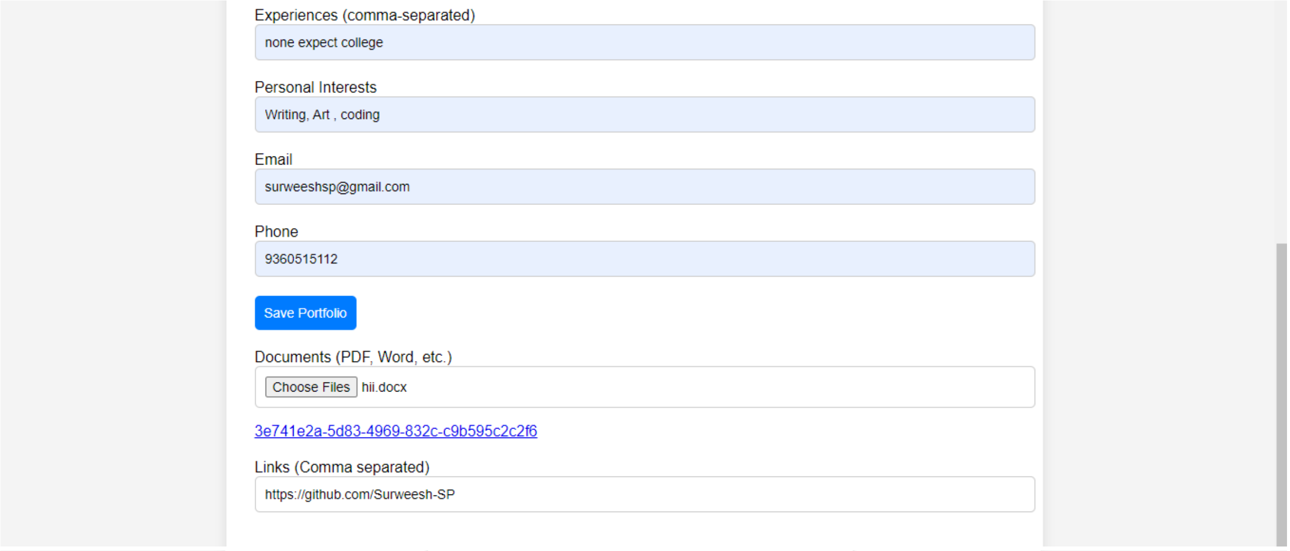
Profile Picture
Choose File Screenshot 2024-11-17 190446.png
Screenshot 2024-11-17 190446.png

Skills (comma-separated)
Python - ML, C, Front end Dev, App Dev

Projects (comma-separated)
this website...

Experiences (comma-separated)
none expect college

Personal Interests



Experiences (comma-separated)
none expect college

Personal Interests
Writing, Art , coding

Email
surweeshsp@gmail.com

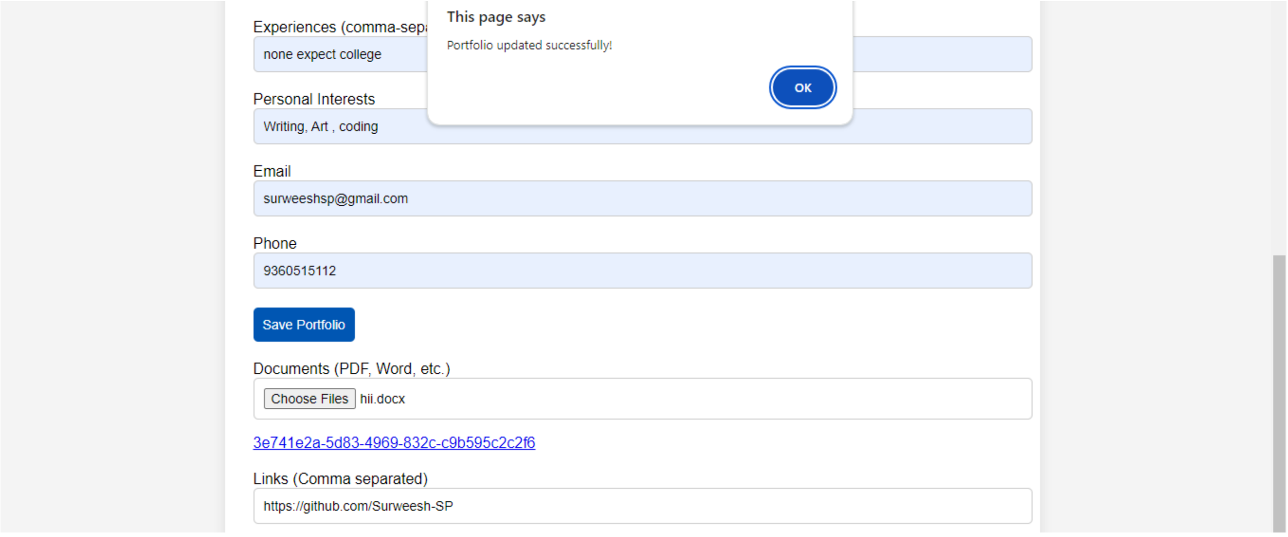
Phone
9360515112

Save Portfolio

Documents (PDF, Word, etc.)
Choose Files hii.docx

3e741e2a-5d83-4969-832c-c9b595c2c2f6

Links (Comma separated)
https://github.com/Surweesh-SP



Experiences (comma-sep
none expect college

This page says
Portfolio updated successfully!
OK

Personal Interests
Writing, Art , coding

Email
surweeshsp@gmail.com

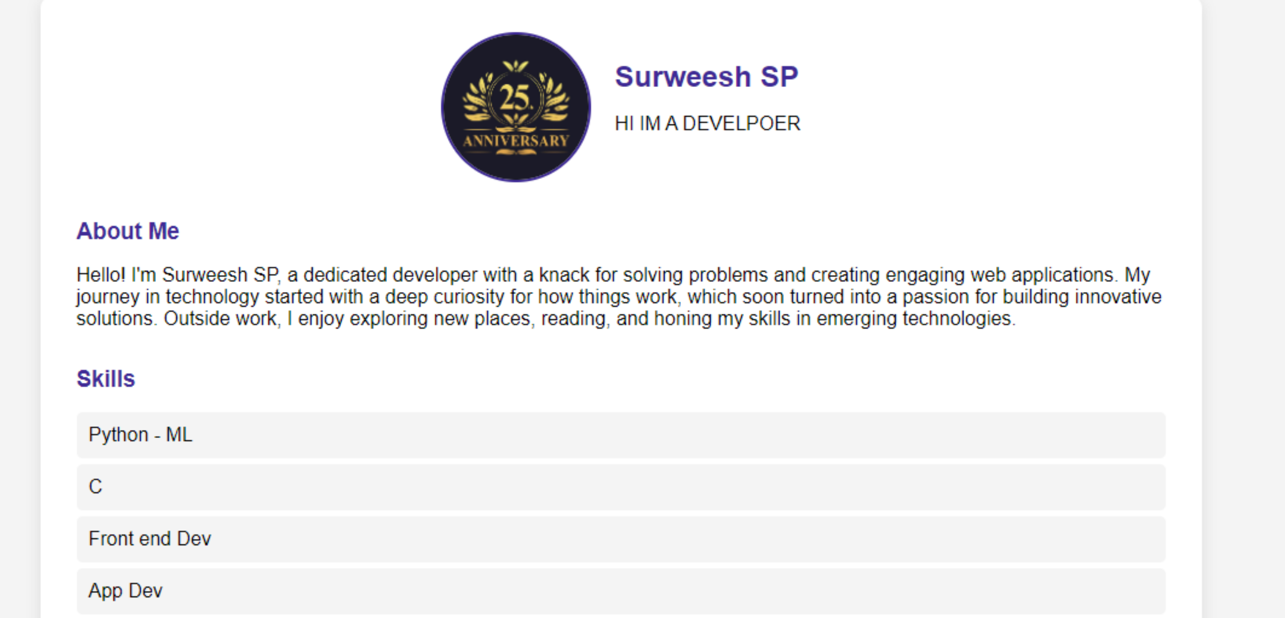
Phone
9360515112

Save Portfolio

Documents (PDF, Word, etc.)
Choose Files hii.docx

3e741e2a-5d83-4969-832c-c9b595c2c2f6

Links (Comma separated)
https://github.com/Surweesh-SP



Surweesh SP

HI IM A DEVELPOER

About Me

Hello! I'm Surweesh SP, a dedicated developer with a knack for solving problems and creating engaging web applications. My journey in technology started with a deep curiosity for how things work, which soon turned into a passion for building innovative solutions. Outside work, I enjoy exploring new places, reading, and honing my skills in emerging technologies.

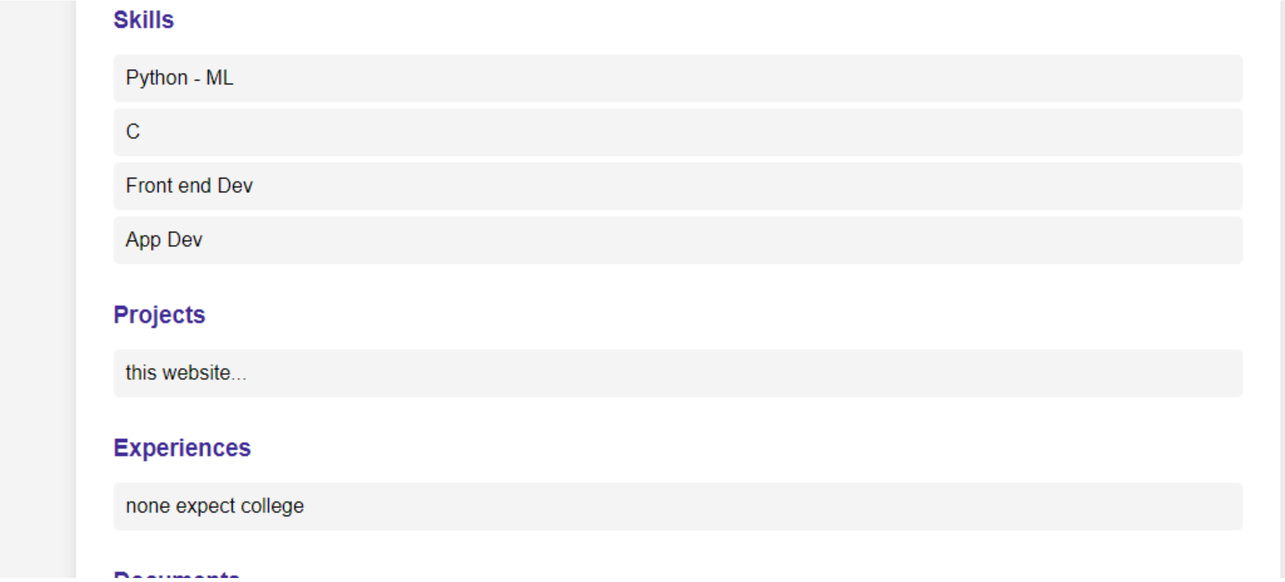
Skills

Python - ML

C

Front end Dev

App Dev



Skills

Python - ML

C

Front end Dev

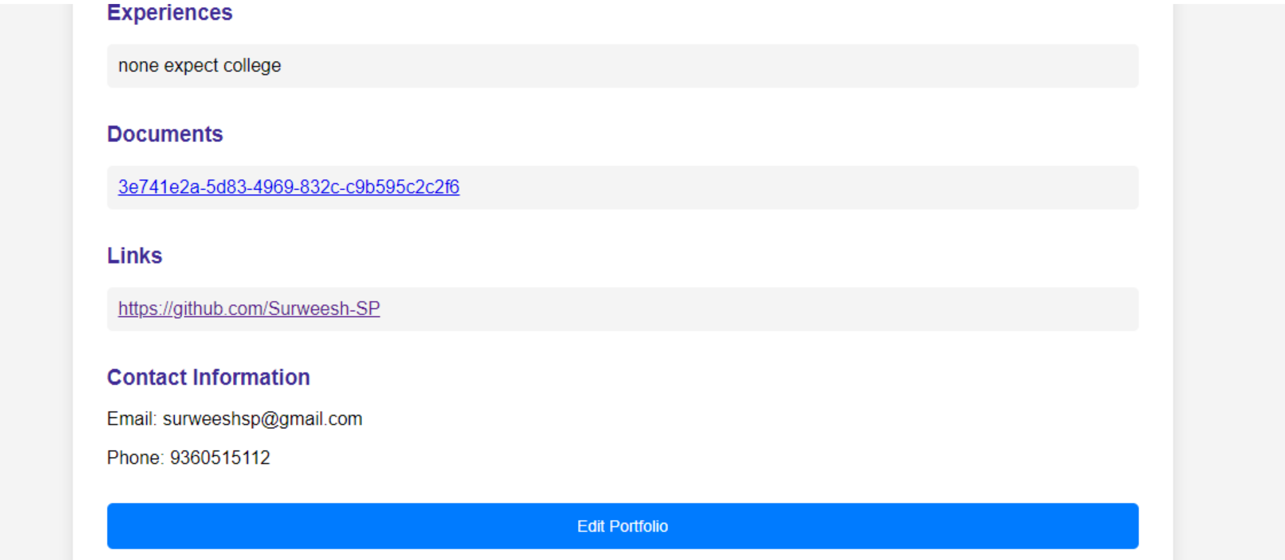
App Dev

Projects

this website...

Experiences

none expect college



Experiences

none expect college

Documents

3e741e2a-5d83-4969-832c-c9b595c2c2f6

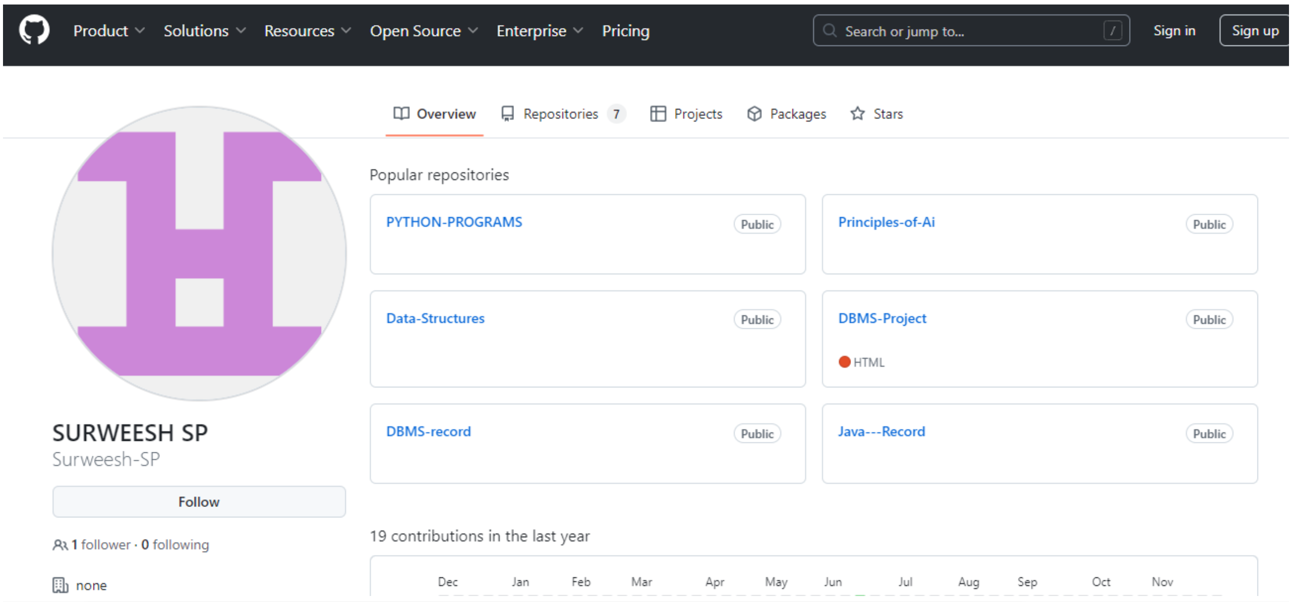
Links

https://github.com/Surweesh-SP

Contact Information

Email: surweeshsp@gmail.com

Phone: 9360515112



SURWEESH SP
Surweesh-SP

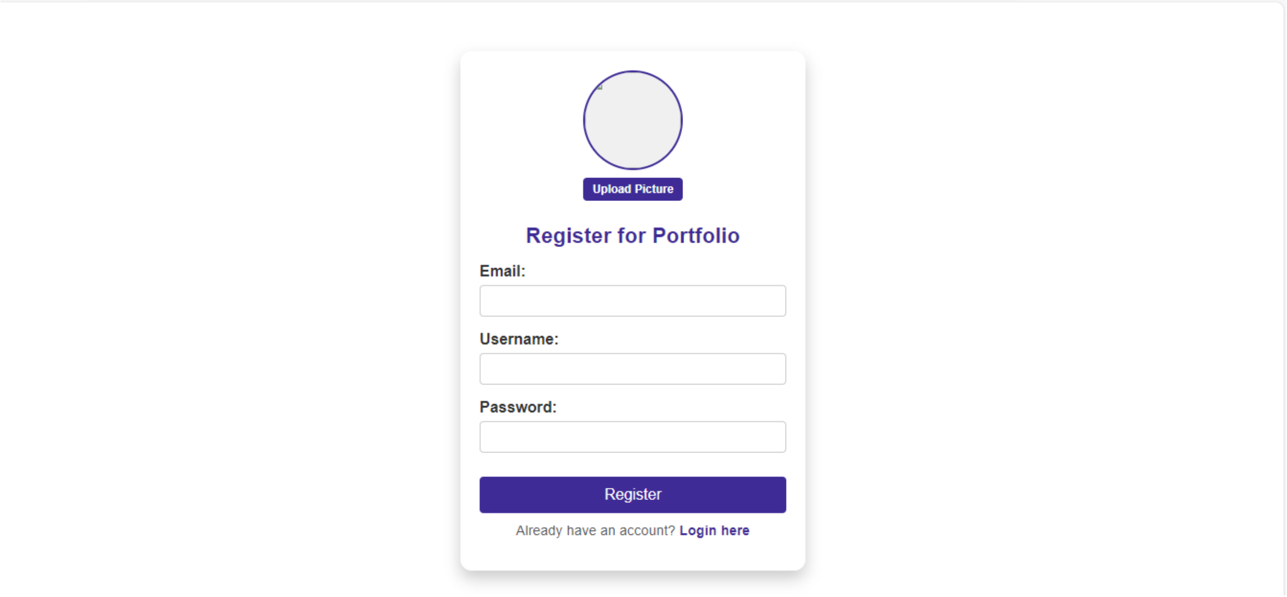
1 follower · 0 following

none

Popular repositories

PYTHON-PROGRAMS Public	Principles-of-Ai Public
Data-Structures Public	DBMS-Project Public ● HTML
DBMS-record Public	Java---Record Public

19 contributions in the last year



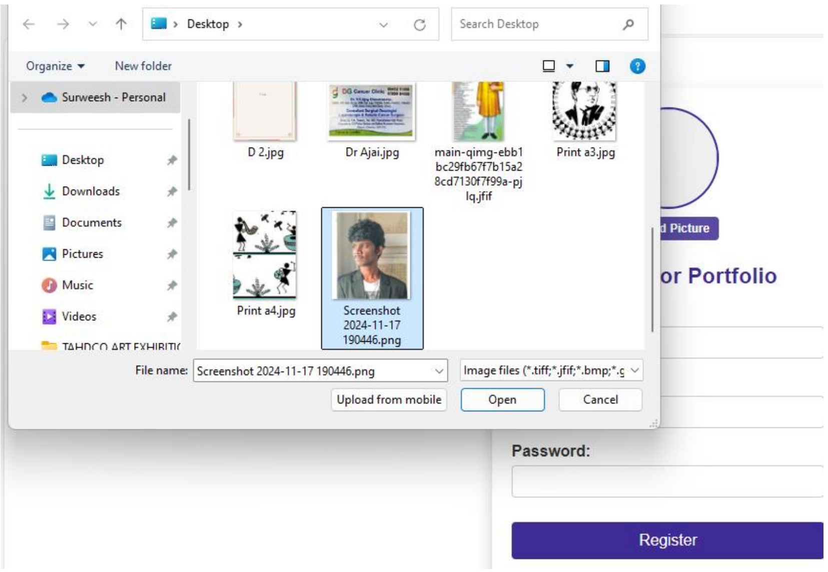
Register for Portfolio

Email:

Username:

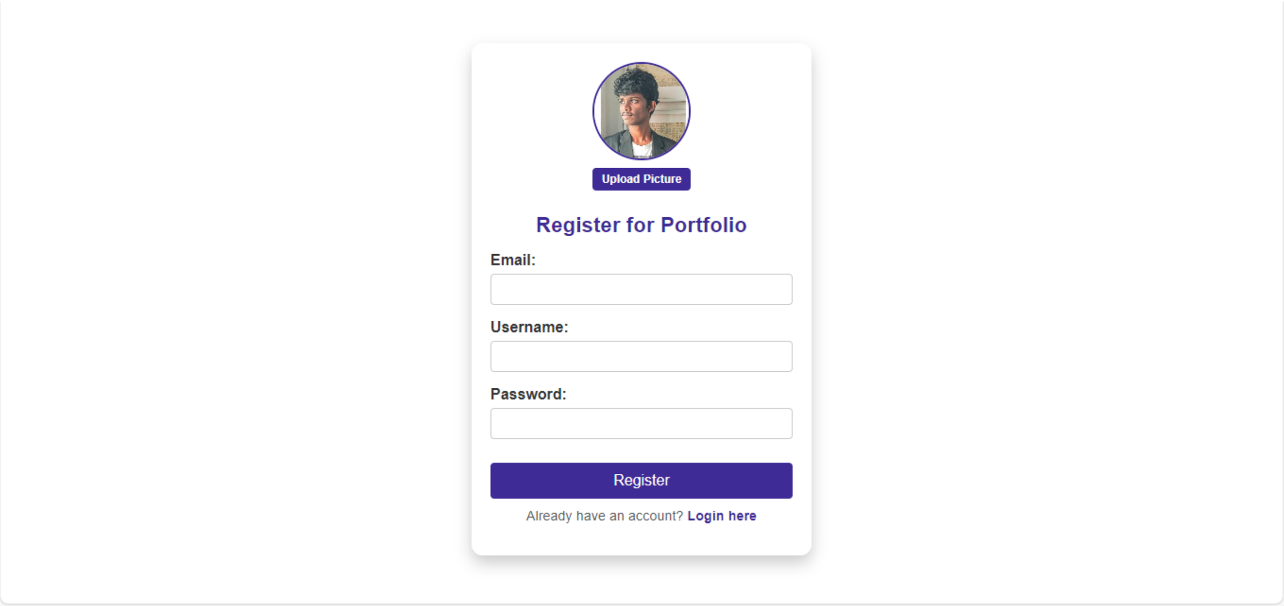
Password:

Already have an account? Login here



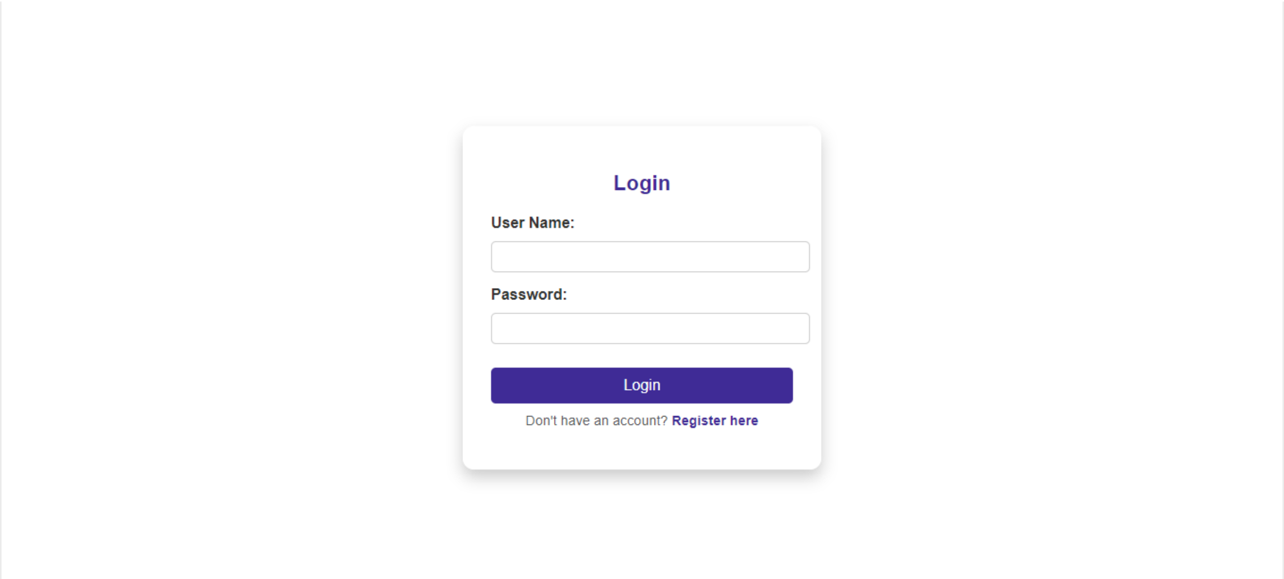
File name: Screenshot 2024-11-17 190446.png

Image files (*.tiff;*.jfif;*.bmp;*.g



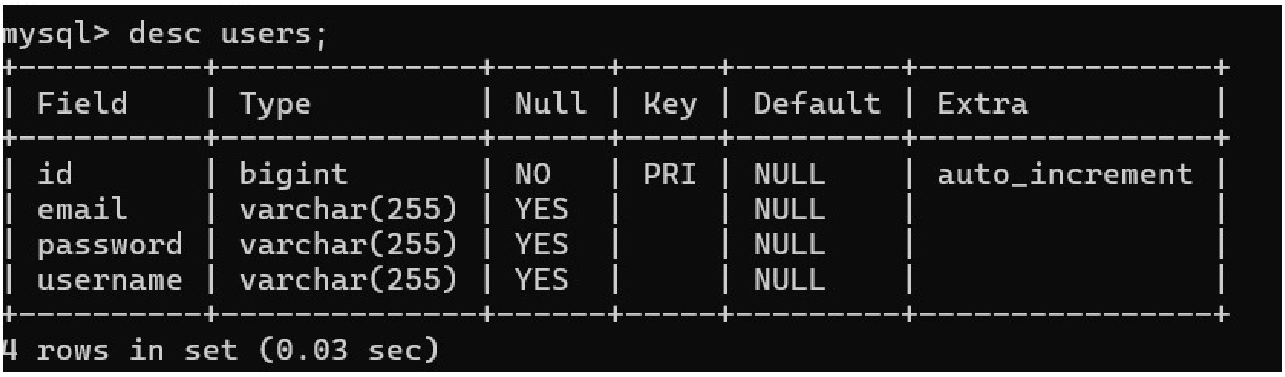
Register for Portfolio

Already have an account? **Login here**



Login

Don't have an account? **Register here**



```
mysql> desc users;
+----------+--------------+------+-----+---------+----------------+
| Field    | Type         | Null | Key | Default | Extra          |
+----------+--------------+------+-----+---------+----------------+
| id       | bigint       | NO   | PRI | NULL    | auto_increment |
| email    | varchar(255) | YES  |     | NULL    |                |
| password | varchar(255) | YES  |     | NULL    |                |
| username | varchar(255) | YES  |     | NULL    |                |
+----------+--------------+------+-----+---------+----------------+
4 rows in set (0.03 sec)
```

6. Testing Details

Testing is a critical phase in the development of any web application, including a personal portfolio website. It ensures that all features work as expected and provides a seamless user experience. The goal of testing is to identify any bugs or issues, validate the functionality of the website, and ensure it meets the defined requirements. Below, we will discuss the different types of testing involved, along with the tools and processes used to ensure the website's functionality, usability, and performance.

1. Types of Testing:

1.1 Functional Testing

Functional testing focuses on verifying whether the website's features are working as intended. For a portfolio website, this involves testing the basic functionalities, such as:

- **Navigation**: Verifying that all links in the navigation menu (Home, About, Portfolio, Contact, etc.) are correctly routed to their respective pages.
- **Forms**: Ensuring that the contact form works by submitting test data (e.g., name, email, and message) and verifying that the form data is sent correctly to the server or email.
- **Portfolio Content**: Ensuring that all portfolio items are displayed correctly, with their descriptions, images, and links functioning properly.
- **Responsive Design**: Testing if the website adjusts appropriately to different screen sizes (desktop, tablet, and mobile). This ensures a seamless user experience across all devices.

Functional testing will be carried out manually and with the help of automated testing tools where applicable. Tools like **Selenium** can be used to automate repetitive browser testing.

1.2 Usability Testing

Usability testing ensures that the website is easy to use and provides a good user experience. Key usability aspects include:

- **Intuitive Navigation**: Verifying that users can easily find the information they are looking for, such as the portfolio or contact details, without confusion.
- **User Interaction**: Testing interactive elements like buttons, hover effects, and pop-ups to ensure they work as expected.
- **Forms and Inputs**: Ensuring that form validation messages (e.g., for email fields) are clear and informative, guiding users to correct any errors.

Usability testing is typically done with real users or stakeholders, who provide feedback based on their experience navigating the website. **UserTesting** and similar platforms can be used to gather user feedback efficiently.

1.3 Performance Testing

Performance testing evaluates how well the website performs under various conditions, such as heavy traffic, slow internet speeds, and large datasets. Key aspects of performance testing include:

- **Page Load Speed**: Measuring how quickly the website loads. A delay in page loading can negatively impact user experience and SEO rankings. Tools like **Google PageSpeed Insights** and **GTmetrix** can be used to analyze load times and suggest performance improvements (e.g., image optimization, script minification).
- **Stress Testing**: Simulating heavy traffic to see how the website performs under high user load. This is particularly important if the website is expected to handle a large number of visitors.
- **Caching**: Ensuring that caching mechanisms are working to store and serve commonly accessed content, improving load speeds for returning visitors.

By optimizing performance, the website will provide users with a faster, more responsive experience.

1.5 Security Testing

Security testing is crucial to protect the website from vulnerabilities that could compromise user data or functionality. Key security tests include:

- **Form Validation**: Ensuring that all form fields, especially the contact form, are protected against malicious input such as SQL injection or cross-site scripting (XSS). Input fields should be sanitized and validated properly.
- **SSL/TLS Encryption**: Verifying that the website uses HTTPS (secure HTTP) and that data transmitted through the site is encrypted.
- **Authentication**: If the website requires any user login functionality (e.g., admin access), security tests should ensure that authentication mechanisms are secure and that unauthorized access is prevented.

Security testing can be done using security scanning tools such as **OWASP ZAP** or **Burp Suite** to identify common vulnerabilities.

1.6 Accessibility Testing

Accessibility testing ensures that the website is usable by people with disabilities, adhering to accessibility guidelines such as WCAG (Web Content Accessibility Guidelines). Key accessibility tests include:

- **Keyboard Navigation**: Ensuring that all interactive elements, such as buttons, links, and forms, can be navigated and interacted with using only the keyboard.
- **Screen Reader Compatibility**: Verifying that the website is compatible with screen readers, allowing visually impaired users to navigate the site.
- **Color Contrast**: Ensuring that text and background colors provide sufficient contrast for readability, especially for users with visual impairments.

Accessibility can be tested using tools like **WAVE** or **axe**, which automatically identify accessibility issues.

7. Conclusion

The development of a personal portfolio website is an essential step for showcasing an individual’s skills, experience, and projects. Throughout the process, we have discussed various technologies, methodologies, and best practices to create a dynamic, functional, and visually appealing website. This project aims to establish a digital presence, allowing potential employers, clients, and collaborators to easily access and evaluate one's work.

The website's design and structure are developed using core web technologies such as HTML, CSS, and JavaScript. These provide the foundation for creating a clean, user-friendly, and interactive interface. In addition, we used MySQL for managing any dynamic content, such as contact form submissions and portfolio project details. Visual Studio Code was chosen as the primary code editor, providing an efficient development environment with support for various extensions to streamline the coding process.

A robust testing strategy, including functional, usability, performance, security, and accessibility testing, was employed to ensure the website operates smoothly across different browsers, devices, and user environments. It is important to ensure that the website is responsive, performs well, and adheres to security and accessibility standards.

By using best practices such as responsive design, SEO optimization, and adherence to web standards, the portfolio website can provide a seamless user experience across different platforms, ensuring accessibility for all users. The use of automated testing tools, cross-browser checks, and performance optimizations guarantees the site is stable, fast, and secure.

The final portfolio website will be a representation of one’s skills, ready to impress potential employers, clients, or collaborators, and will continue to evolve as more projects, experiences, and feedback are added over time. Future iterations of the website may include additional features such as a blog, interactive elements, or a live project showcase to further enhance its functionality and appeal.

8. References

1. **HTML Living Standard** – WHATWG. (2023). *Hypertext Markup Language (HTML)*. Retrieved from: https://html.spec.whatwg.org/
2. **CSS3 Specification** – W3C. (2023). *Cascading Style Sheets Level 3 (CSS3)*. Retrieved from: https://www.w3.org/TR/css3-roadmap/
3. **JavaScript (ECMAScript) Specification** – ECMA International. (2023). *ECMAScript® 2023 Language Specification*. Retrieved from: https://www.ecma-international.org/publications-and-standards/
4. **MySQL Documentation** – Oracle. (2023). *MySQL 8.0 Reference Manual*. Retrieved from: https://dev.mysql.com/doc/
5. **Google Developers**. (2023). *PageSpeed Insights* – How to measure and improve your website's performance. Retrieved from: https://developers.google.com/speed/pagespeed/insights
6. **OWASP ZAP** – Open Web Application Security Project. (2023). *Zed Attack Proxy (ZAP) Documentation*. Retrieved from: https://www.zaproxy.org/
7. **W3C Web Content Accessibility Guidelines (WCAG)**. (2023). *Web Content Accessibility Guidelines (WCAG) Overview*. Retrieved from: https://www.w3.org/WAI/WCAG21/quickref/
8. **Visual Studio Code** – Microsoft. (2023). *Visual Studio Code Documentation*. Retrieved from: https://code.visualstudio.com/docs