
Embrace Opportunity

Software Engineering Day 1

Introduction

Today's Overview

- Introduction
- Course Overview
- Capstone Project Overview
- HTML & CSS
- JavaScript
- Exercise
- Homework

- Cheat Sheet:
  - https://byrondev121.github.io/ix-vue-press/
  - https://byrondev121.github.io/ix-vue-press/day-1

Introduction to the Team

Head Teacher:

- Byron de Villiers
  - Senior Software Engineer
  - byron.devilliers@ixperience.co

Teaching Assistants:

- Sheahan Hearn
  - Software Engineer
  - sheahan.hearn@ixperience.co

Course Overview

Full Stack Development

Tech Stack We'll Be Using

FRONTEND

BACKEND

Day 1 - Monday

- Introduction
- Course Expectations & Overview
- Capstone Project Overview
- Recap of HTML, CSS & JavaScript
- Connecting & Using Bootstrap
- Exercises:
  - Bootstrap grid system.
- Homework:
  - Use Bootstrap to update the styling of your personal portfolio web page from the pre-work exercise.

Day 2 - Tuesday

- JavaScript Deep Dive
- Navigating the Files & Folders
- Version Control Best Practices with GitHub & Git
- Exercises:
  - Write a function that prints out the first 10 digits of the Fibonacci sequence.
- Homework:
  - JavaScript:
    - Calculating the area of a triangle.
  - GitHub and Git
    - Create a GitHub repo for the iX course.
    - Create 2 new directories in the workspace for your personal portfolio web page from the day 1 and the JavaScript function from the homework.

Day 3 - Wednesday

- Introduction to ReactJS
- JSX
- Components
- Exercises:
  - Building a Simple React Application.
- Homework:
  - Generate React application:
    - Create one custom component that displays students name and surname.
    - Push homework application to GitHub.

Day 4 - Thursday

- React State Management
- React Props
- Exercises:
  - Building the BlogItemText Component.
- Homework:
  - Building and updating the Categories page component and subcomponent using the techniques learnt today.

Day 5 - Friday

- React Lifecycle
- React Hooks
- Routing with React Router
- Exercise:
  - Creating the routes for the front end pages.
- Homework:
  - Start building the front end application for the Capstone project
  - Create Capstone repository.
  - Push changes to GitHub repository.

Day 6 - Monday

- Asynchronous JavaScript
- Javascript Event Loop
- Callback functions, Promises Async/Await
- Introduction to Backends, APIs and REST
- Homework
    - Update front end application to fetch data from teacher server
    - Push changes to GitHub

Day 7 - Tuesday

- Introduction to Node.js
- Setup for Express
- RESTful APIs with Express
- API Routes
- Backend Service
- Homework
  - Create Express backend server
  - Create a route and service to fetch static data from the backend server
  - Use postman to test the backend server
  - Push changes to GitHub & submit link to homework repo
- Pre-Work
  - Install Mongo DB server and compass on local machine

Day 8 - Wednesday

- Introduction to MongoDB
- Connecting MongoDB with Node.js
- Database Models
- CRUD Operations with Express
- Homework
  - Connect backend Express server with MongoDB
  - Create a data model for blog posts and add data to local MongoDB server
  - Update blog app front end to fetch blog posts in the DB from the Express backend server

Day 9 - Thursday

- Database Design
- UML Diagrams
- Introduction to Admin Panel
- CRUD Operations
- Homework
    - Create a design diagram of the blog posts
    - Create a React app for an admin panel and build CRUD operations.

Day 10 - Friday

- Recap of the last two weeks
- Succeeding in your internships
- Q&A
- Homework
    - Work on Capstone project

Day 11 - Tuesday

- JSON Web Token (JWT) Authentication
- Introduction to Middleware
- Authorisation in Node.js
- Homework
    - Add JWT authentication to the admin panel of the Capstone project for blog posts

Day 12 - Thursday

- Introduction to Redux for State Management in React
- Exercise:
  - Implement Redux
  - Identify which state slice should be added to global state in your project
  - Update the code to handle that slice of state in Redux
- Homework:
  - Continue converting React state to Redux global state in blog posts web app

Day 13 - Tuesday

- Introduction to Mutli-Part Form Data Upload
- Introduction to Multer for Node.js
- Homework
  - Add upload functionality to the admin panel blog posts web app
  - Consume the images from the front end

Day 14 - Thursday

- Data Structures and Algorithms
  - Arrays
  - Hash maps
  - Linked lists
  - Stacks
  - Queues
  - Trees
- Exercise:
  - Palindrome / Anagram
- Homework:
  - Solve TwoSum problem
  - Bonus: Solve longest substring without repeating characters.
  - Create new directory for solution and push changes to GitHub.

Day 15 - Tuesday

- Finalising Capstone Project
- Preparing for Presentations
- Homework:
  - Complete Capstone project
  - Prepare final presentation

Day 16 - Thursday

- Successfully present Capstone project
  - Highlight key features
  - Technologies used
  - Challenges overcome
  - Lessons learned

Day 17 - Tuesday

- Advanced Software Development Practices
  - Pull Requests Using Github
  - Code Reviews
  - Testing with Jest
- Homework
  - Add unit tests to Capstone project
- Pre-work
  - Read Chapter 2, 4 and 6 of “Cracking The Coding Interview”

Day 18 - Thursday

- Successfully present internship project
  - Highlight key features
  - Technologies used
  - Challenges overcome
  - Lessons learned

Capstone Project Overview

Blog App

Project Requirements: Blog App

- Example: https://ix-blog-app-2d5c689132cd.herokuapp.com/
- Create a **Blog Post Application**, that incorporates these aspects:
- Front end web app - Utilising ReactJS:
  - Allowing users to be registered and login.
  - Allow users to browse, read and create blog posts and categories.
- Backend and API – Utilising NodeJS & Express & MongoDB:
  - Handle authentication and CRUD operations for the blogs, categories and authentication.

Project Requirements: Blog App - Frontend

- This blog app will consist of these pages:
  - Landing page:
    - This view displays the title card, recent blog posts section and categories section.
    - The two sections will interact with the Blogs page and the Blog page.
  - Blogs page:
    - This view will allow the user to view the title card of all blogs posted.
    - Logged in users will be able to create, edit and delete blogs.
      - Each with their own modal popup.
    - Selecting a blog will open the Blog page on the selected post.
  - Blog page:
    - This view will allow the user to read the full blog post.
    - Accessed from both the Blogs page, Profile page and Landing page.

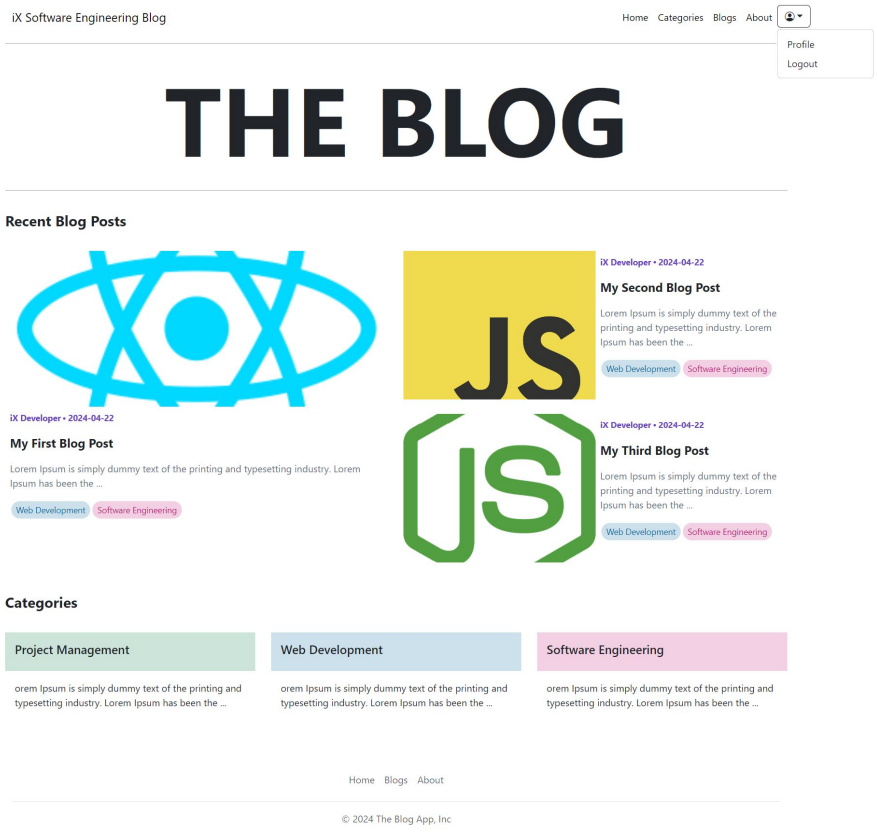
Project Requirements: Blog App - Frontend

- Categories page:
  - This view will allow user to view all categories of blogs available.
  - Logged in users will be able to create, edit and delete categories.
    - Each with their own modal popup.
  - Selecting category will filter the posts on the Blogs page.
- Profile page:
  - This view will allow users to view details of the user, and the blog posts created by that user.
  - Logged in user will be able to edit their profile details.

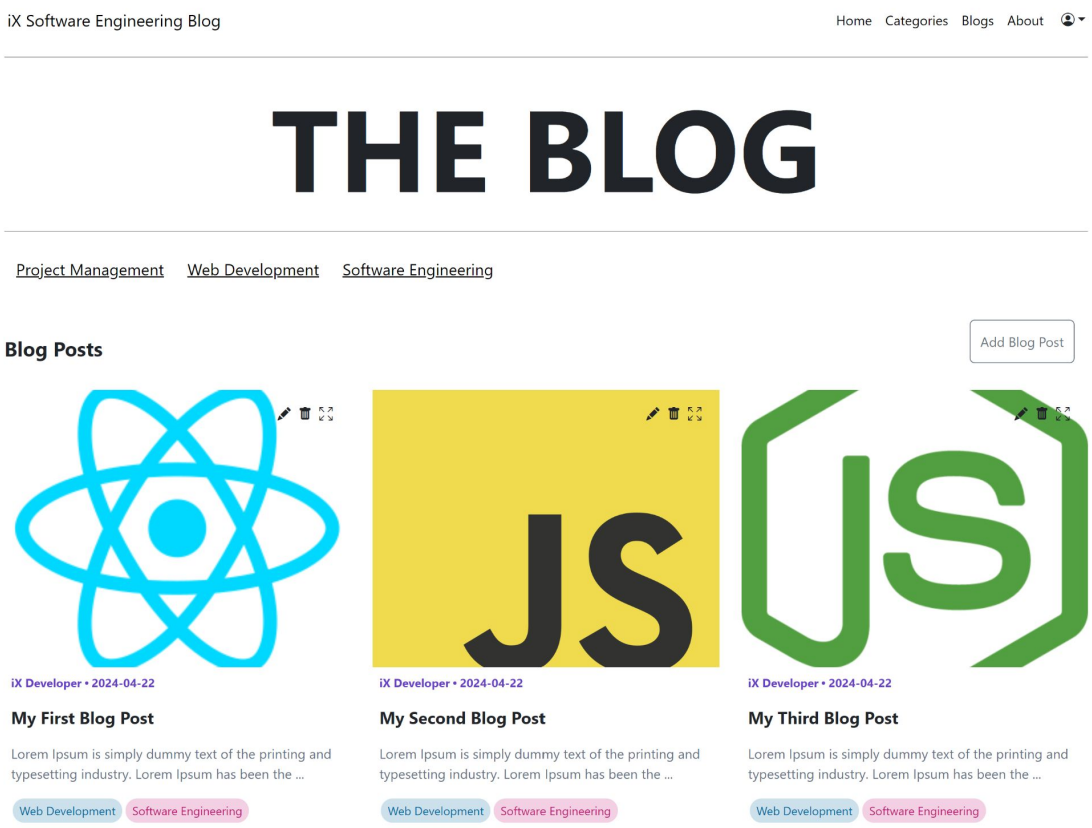
Project Requirements: Blog App - Frontend

- Login page:
  - This view will allow users to login to their profile.
  - Users can navigate to the register page from here if they do not already have a profile.
- Register page:
  - This view will allow users to create their profile.
  - Users can navigate to the login page from here if they already have a profile.

Expected Frontend - Landing Page

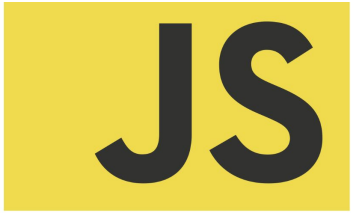


Expected Frontend - Blogs Page



Expected Frontend - Categories Page

Expected Frontend - Blog View



My Second Blog Post

2024-04-22 by iX Developer

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Web Development | Software Engineering

Introduction

I'm so excited to share my first blog post with the world. I've been working on this for a while and I'm happy to finally share it with you. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Body

This is the body of my blog post. I hope you enjoy reading it as much as I enjoyed writing it. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

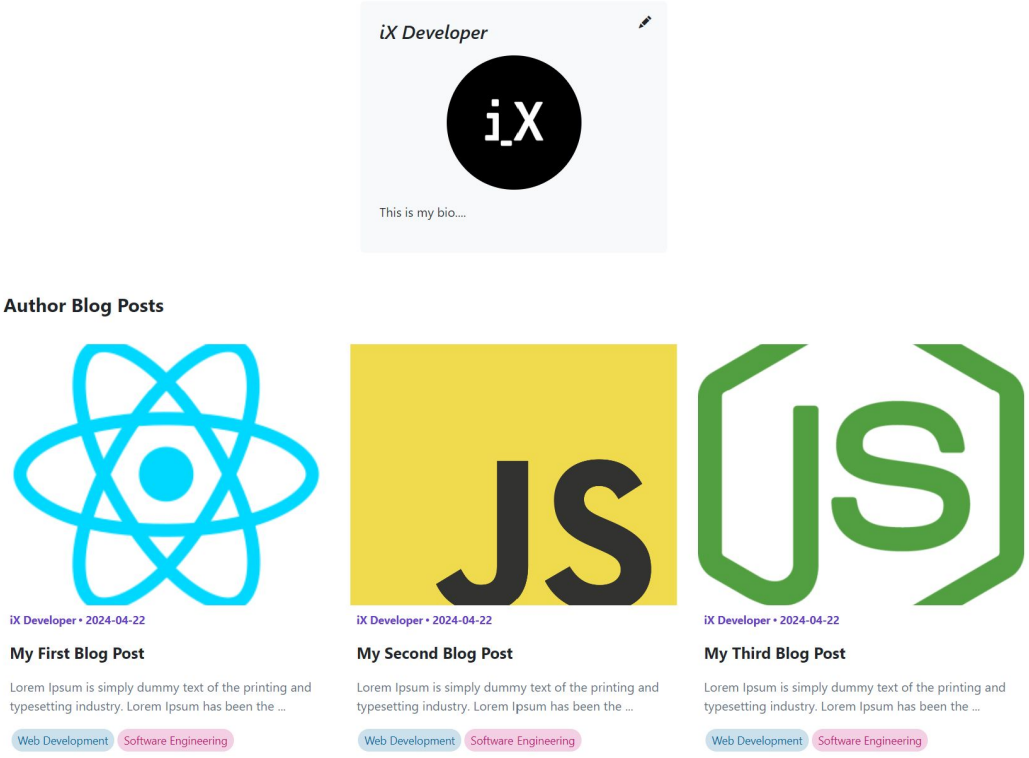
Conclusion

I hope you enjoyed reading my first blog post. I'm looking forward to sharing more with you in the future. Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

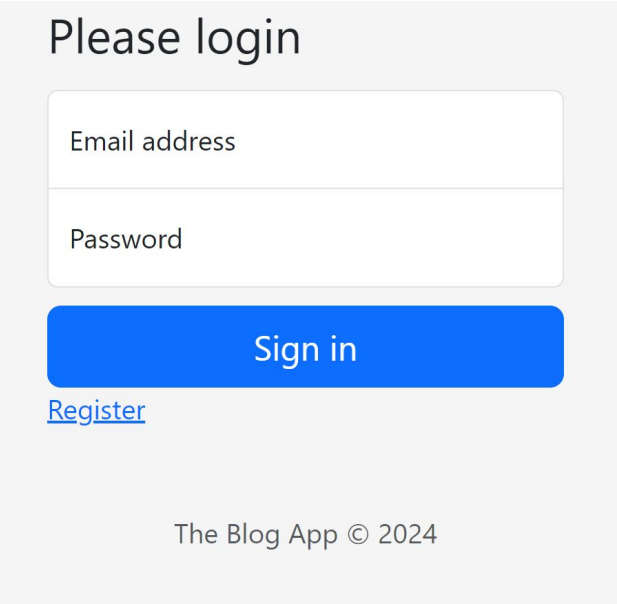
About the author

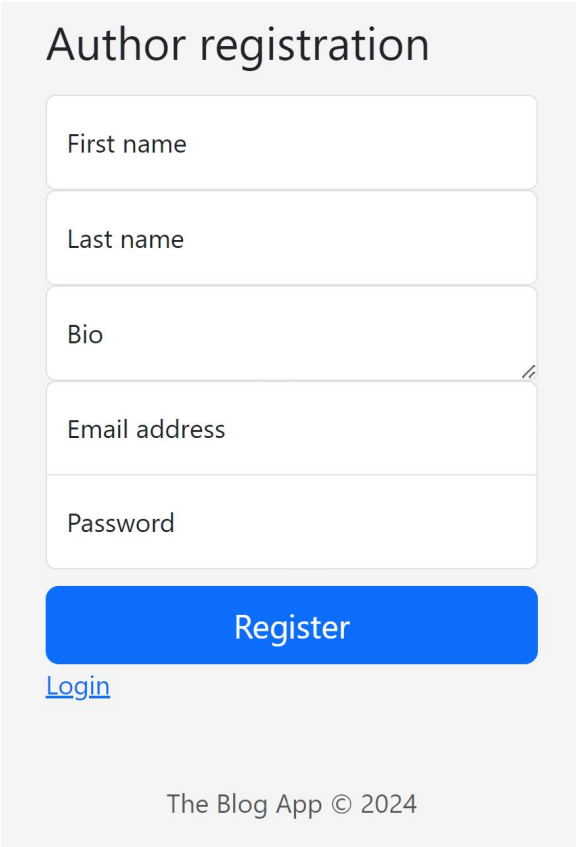
This is my bio....

Expected Frontend - Profile Page

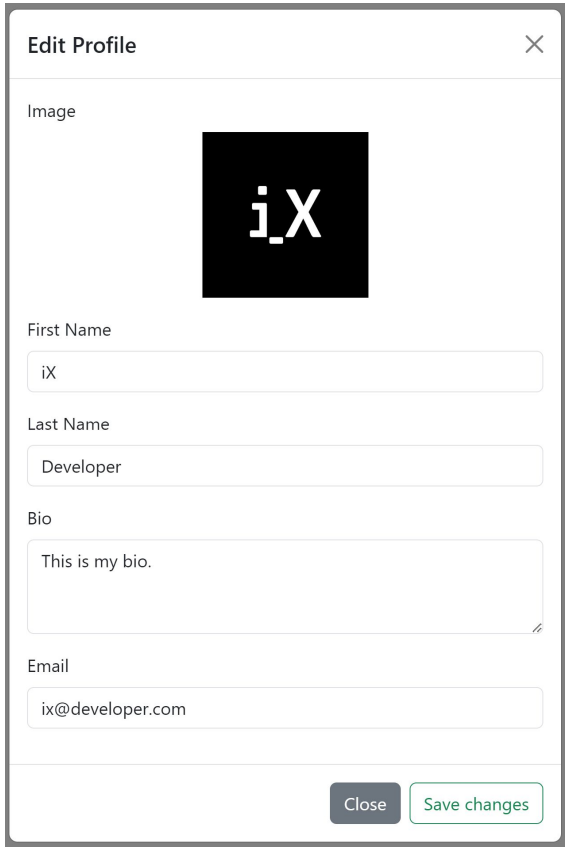


Expected Frontend - Login / Registration

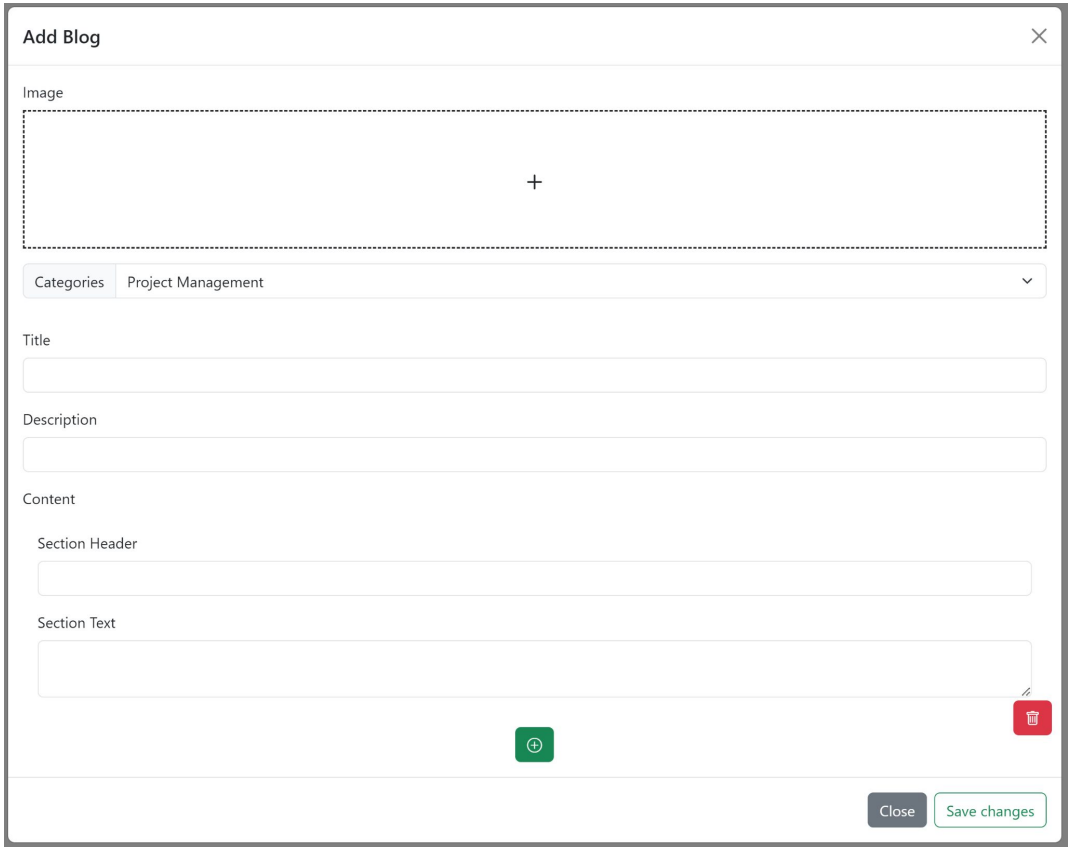




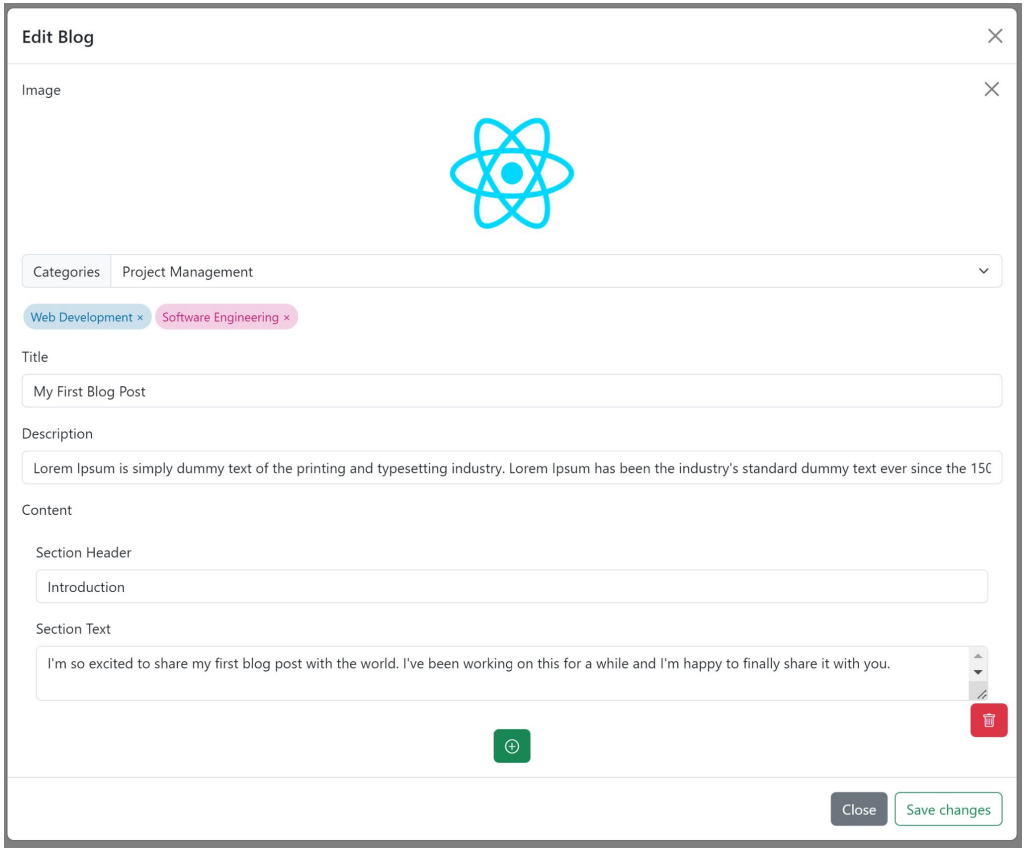
Expected Frontend - Profile - Update



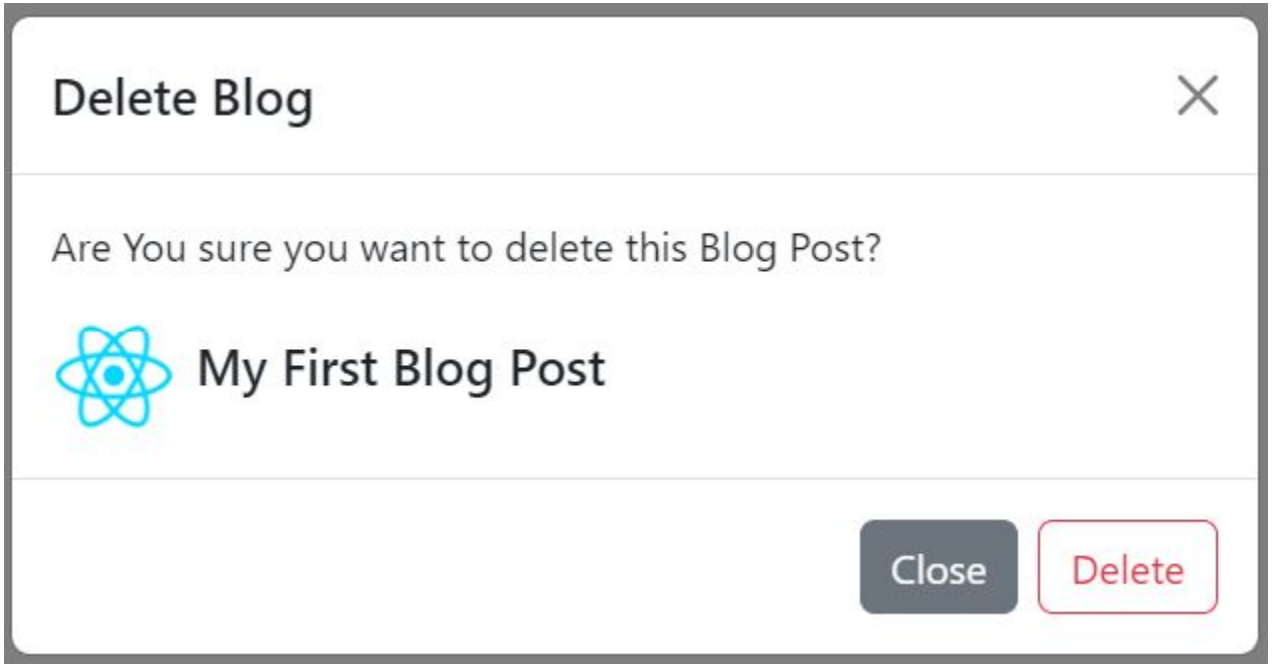
Expected Frontend - Blog - Create



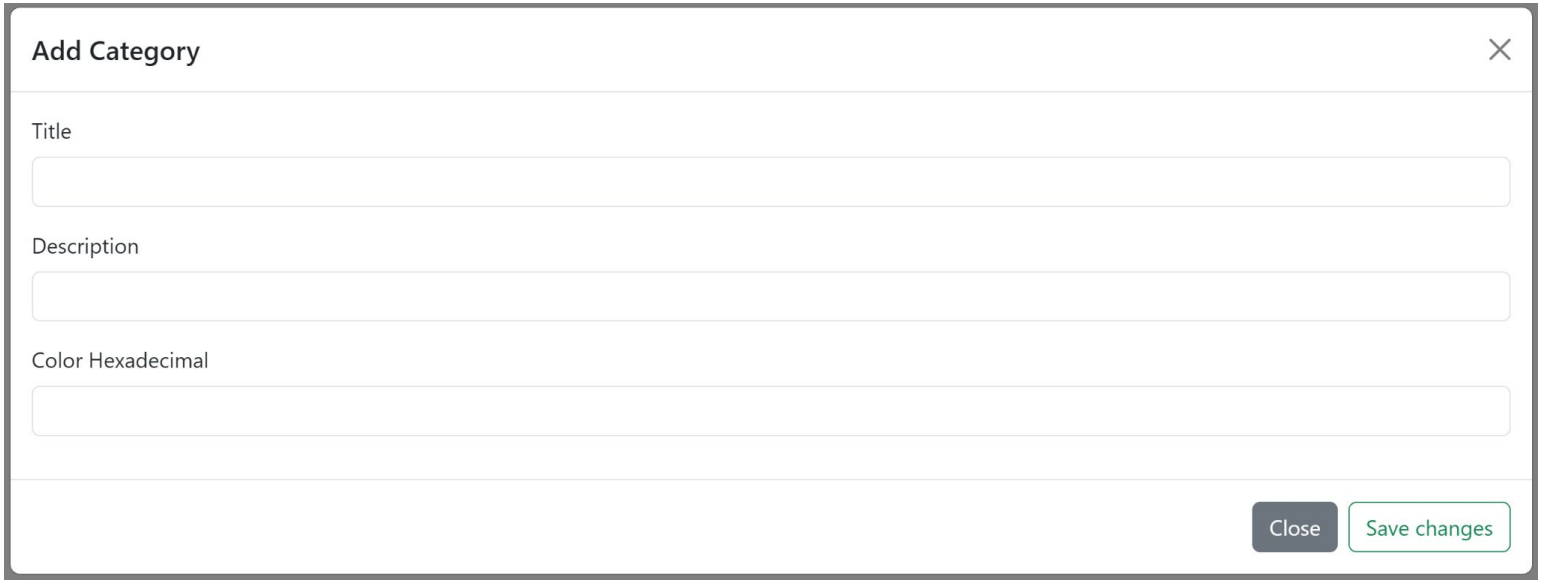
Expected Frontend - Blog - Update



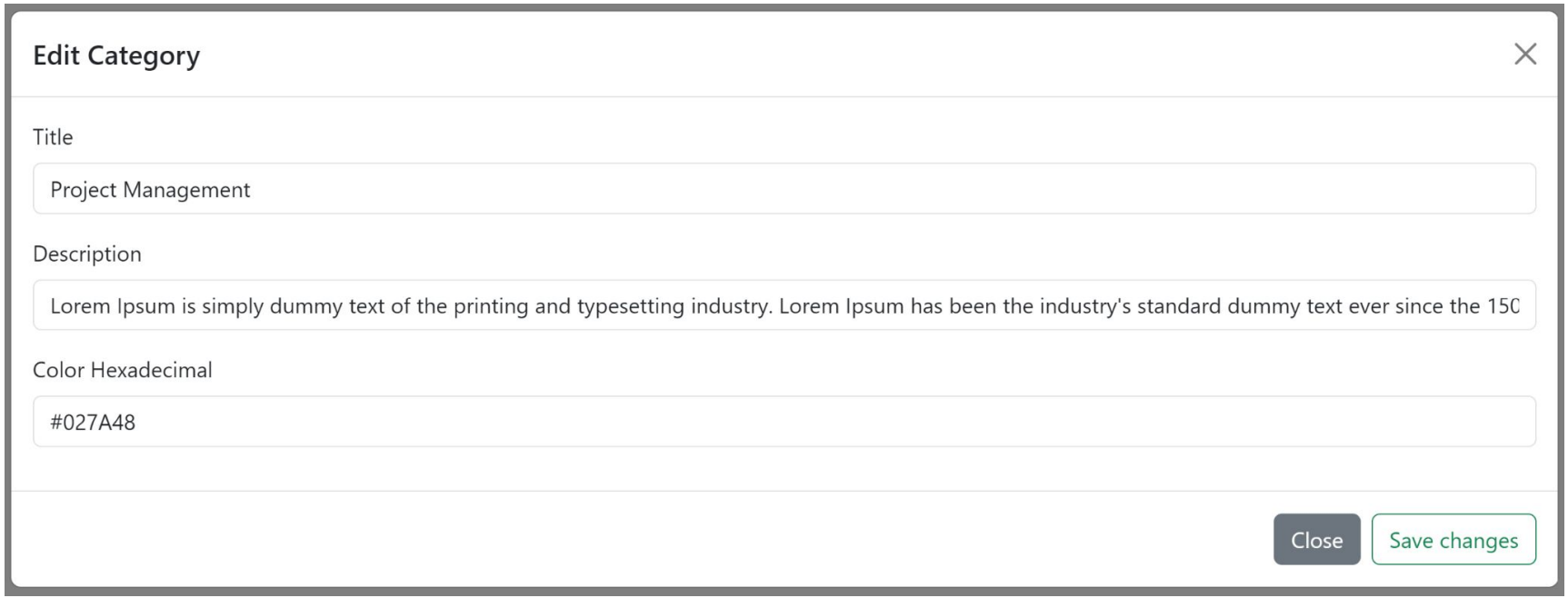
Expected Frontend - Blog - Delete



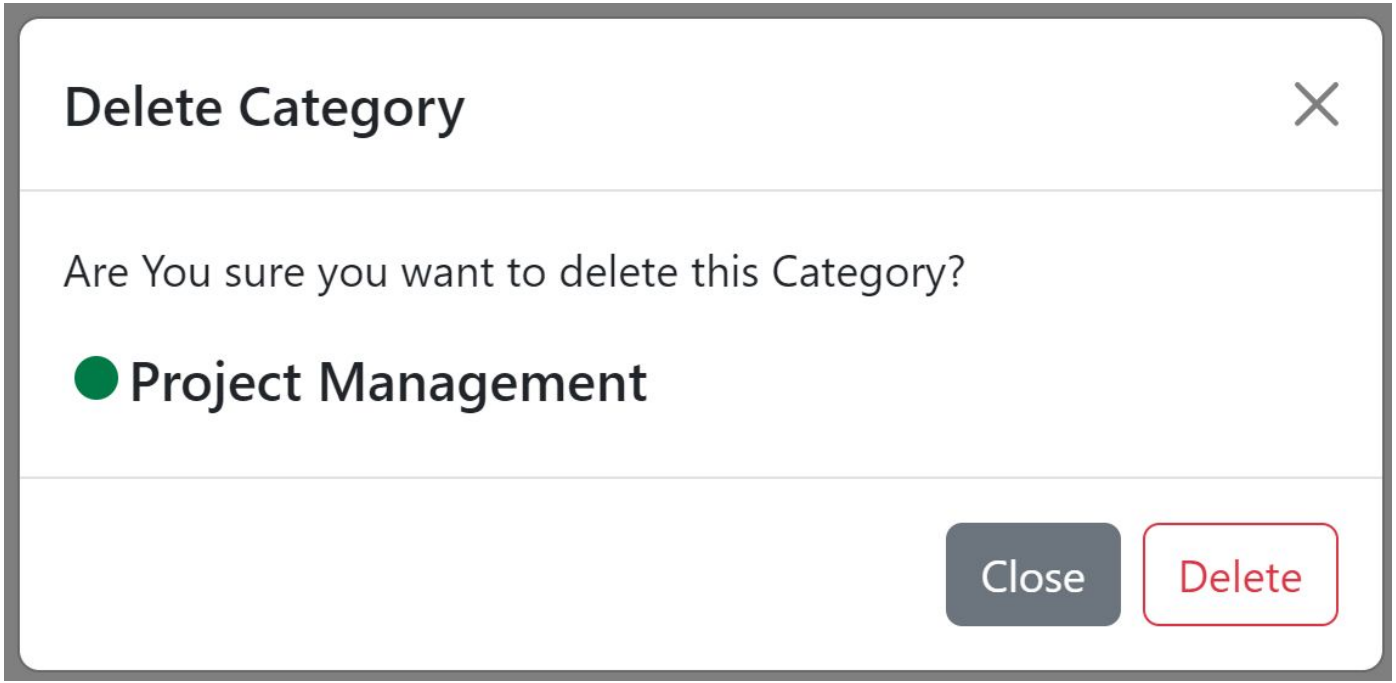
Expected Frontend - Category - Create



Expected Frontend - Category - Update



Expected Frontend - Category - Delete



Project Requirements: Blog App - Backend

- Node.js, Express, and MongoDB - Backend & API
- API with routes:
  - Handling authentication for users:
    - Registration
    - Login
  - CRUD for blogs, categories and authentication:
    - Create
    - Read
    - Update
    - Delete

Project Requirements: Blog App - API Routes

- Routes:
  - Authentication:
    - Create user - *login*
    - Register new user - *register*
    - Get user by ID - *getUser*
    - Update user - *updateUser*
  - Categories:
    - Create category - *createCategory*
    - Get all categories - *getCategories*
    - Get category by ID - *getCategory*
    - Update category - *updateCategory*
    - Delete category - *deleteCategory*

Project Requirements: Blog App - API Routes

- Blog Posts:
  - Create blogs - *createBlog*
  - Get all blog posts - *getBlogs*
  - Get blog by ID - *getBlog*
  - Get blogs by author ID - *getBlogsByAuthorId*
  - Get blogs by categories ID - *getBlogsByCategoriesId*
  - Update blog - *updateBlog*
  - Delete blog - *deleteBlog*

HTML

Standard Markup Language

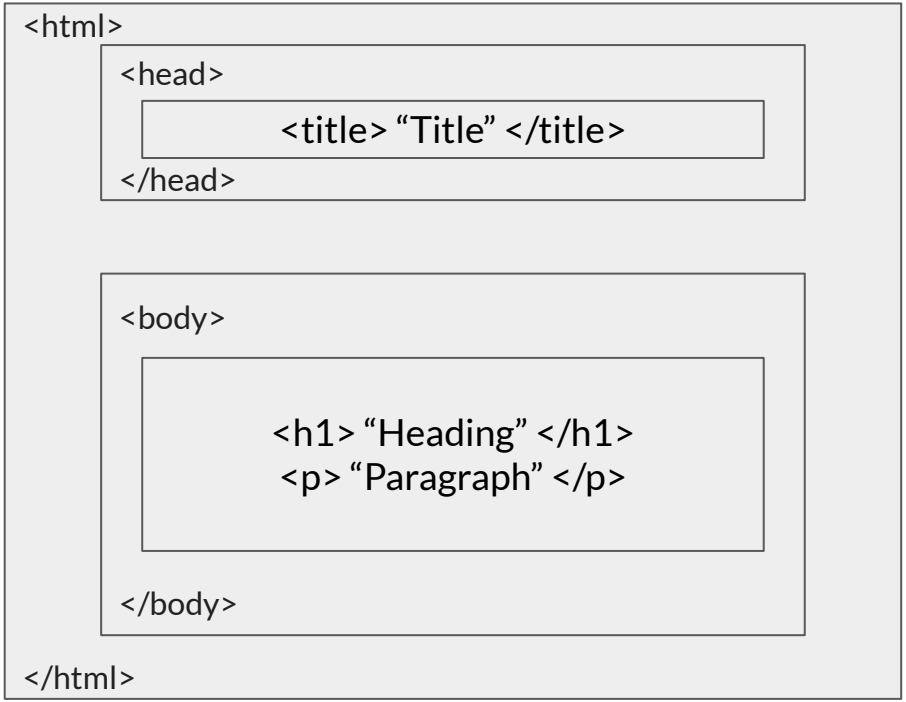
HTML - The Building Block of the Web

- What is HTML?
  - HyperText Markup Language
  - Structures web content using elements and tags.

- Why Learn HTML?
  - Foundation for creating web pages.
  - Essential for building more complex and interactive applications using frameworks like React.

HTML - Basic Structure

- <html>
  - Defines the start of an HTML document.
- <head>
  - Contains meta-information like the title and links to stylesheets.
- <body>
  - Includes all the content displayed on the web page.



HTML - Basic Structure

```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>My Title</title>
    <meta name="description" content="" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="stylesheet" href="./path-to-css-file" />
  </head>
  <body>
    <!-- Body Content -->
    <h1>Day 1 - HTML and CSS recap</h1>
    <!-- JS imports -->
    <!-- <script src="./path-to-js-file"></script> -->
  </body>
</html>
```

HTML - Tags

- **Tags are** used to mark up **HTML elements**.
- Basic syntax:
  - Tag/element name surrounded by angle brackets
  - Normally in pairs (start tag and end tag)
  - End tag usually the same but with a forward slash
  - i.e. <tagname> content</tagname>
  - e.g. <h1>Heading</h1>

HTML - Block vs. Inline Tags

- Block elements:
  - Start on a new line
  - Take the full width available
  - E.g. , <h1>-<h6>, <p>, <form>

- Inline elements:
  - Do not start a new line
  - Take only necessary width
  - E.g. , <img>, <a>, ,

HTML - Attributes

- What are attributes:
  - HTML tag attributes provide additional information about an HTML element. They are used to define the properties of the element and to provide extra details that affect how the element behaves or appears. Attributes are included in the opening tag of an element and usually come in name/value pairs like name="value".
- Attributes are placed inside the opening tag:
  - i.e. <tag name="value">Content</tag>
  - e.g. <h1 class=”title” >Title<h1>
- Global Attributes:
  - *‘class’*
  - *‘dir’*
  - *‘id’*
  - *‘style’*

HTML - Attributes

- Element Specific Attributes:
  - Anchor Tag:
    - <a href=”http.com”>
  - Image Tag:
    - <img src=”source”>

HTML - Common Elements - Header Tags

- Header
  - <h1>,<h2>,<h3>,<h4>,<h5>,<h6>

```html
<!-- Headings -->
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

HTML - Common Elements - Paragraph Tags

```html
<!-- Paragraphs -->
    <p>
        Lorem ipsum dolor sit amet consectetur adipisicing elit. <strong>Provident, mollitia fac
        pariatur nihil vel,
        provident necessitatibus aspernatur dicta dolorem unde enim dolor porro voluptatum sapie
        Quibusdam blanditiis doloremque error minus soluta placeat ipsa quam, <em>tempora nihil
            tempore</em>
        accusamus magnam explicabo dolore obcaecati nisi asperiores numquam saepe, hic ea offici
        dignissimos? Animi, magni totam recusandae autem, iste culpa quaerat quae pariatur iure
        sapiente optio quisquam praesentium numquam voluptas, qui aliquid quia omnis id explicab
        odio
        quod! Maxime totam, nostrum officia optio esse ipsa facere, facilis quod temporibus, non
        necessitatibus.
        Veritatis numquam aliquid corporis. Quis consequuntur, nulla, ratione libero et similiqu
        ducimus minima maiores quod voluptatibus ab error. Beatae adipisci repudiandae iste cupi
        perferendis
        expedita harum et sequi? Recusandae, asperiores non facilis dolore autem sapiente corrup
        cumque
    </p>
```

HTML - Key Text Elements

Element	Meaning	Purpose
<b>	Bold	Highlight important information
	Strong	Similarly to bold, to highlight key text
<i>	Italic	To denote text
	Emphasised Text	Usually used as image captions
<mark>	Marked Text	Highlight the background of the text
<small>	Small Text	To shrink the text
<strike>	Striked Out Text	To place a horizontal line across the text
<u>	Underlined Text	Used for links or text highlights
<ins>	Inserted Text	Display with an underline to show an inserted text
<sub>	Subscript Text	Typographical stylistic choice
<sup>	Superscript Text	Another typographical style

HTML - Common Elements - Anchor

- Anchor
  - <a>
  - Used for navigation

```html
<a href="https://www.google.com" target="_blank">Go to google</a>

<a href="https://www.github.com" target="_blank">Go to github</a>
```

HTML - Common Elements - Lists

- Lists:
  - Ordered [Numbered]:
    -
  - Unordered [Bullets]:
    -
  - List Item
    -

```html
<!-- Lists -->

<ul>
    <li>List item 1</li>
    <li>List item 2</li>
    <li>List item 3</li>
    <li>List item 4</li>
    <li>List item 5</li>
</ul>


<ol>
    <li>List item 1</li>
    <li>List item 2</li>
    <li>List item 3</li>
    <li>List item 4</li>
    <li>List item 5</li>
</ol>
```

HTML - Common Elements - Form

- Form:
  - <form>
  - Elements Inside:
    - Input: <input>
      - Input can be: text / checkbox / radio / drop down

HTML - Common Element - Form

```html
<!-- Forms -->

    <form submit="submitFunction()">

        <div>
            <label>First name</label>
            <input type="text" name="firstName">
        </div>
        <br>
        <div>
            <label>Email address</label>
            <input type="email" name="email">
        </div>
        <br>
        <div>
            <label>Gender</label>
            <select>
                <option>Male</option>
                <option>Female</option>
                <option>Other</option>
            </select>
        </div>
        <br>
        <div>
            <label>Message</label>
            <textarea></textarea>
        </div>
        <br>
        <div>
            <input type="submit" value="Save">
        </div>

    </form>
```

HTML - Common Elements - Button

- Button:
  - <button>

```html
<!-- Buttons -->
<button click="submitFunction()">Save</button>
```

HTML - Common Elements - Image

- Image - Inserting Images
  - <img>

```html
<img alt="Test image" width="200px" height="auto"
       src="https://github.githubassets.com/assets/GitHub-Mark-ea2971cee799.png">
```

HTML - Container Elements

- Div:
  -
  - Used for sections
- Span:
  -
  - Used for inline texts
- Semantic tags
  - <header>
  - <nav>
  - <article>
  - <figure>
  - <footer>

HTML - Semantic Tags

```html
<header class="container">
        <h1>Blog Posts</h1>
    </header>
    <section class="container">
        <article>
            <h3>Blog post 1</h3>
            <small>Published by Byron de Villiers on 7 July 2022</small>
            <p>
                Lorem Ipsum Paragraph.
            </p>
            <a href="#">Read more</a>
        </article>
    </section>
    <aside>
        <h3>Categories</h3>
        <ul>
            <li><a href="https://www.google.com">Google</a></li>
            <li><a href="https://www.github.com">Github</a></li>
        </ul>
    </aside>
    <footer class="footer">
        <p>&copy 2022 Blog posts</p>
    </footer>
```

HTML - Useful Documentation

- Elements
- Attributes
- Global Attributes

CSS

Style Sheet

CSS - What is CSS and Why CSS?

- CSS
  - Cascading Style Sheet

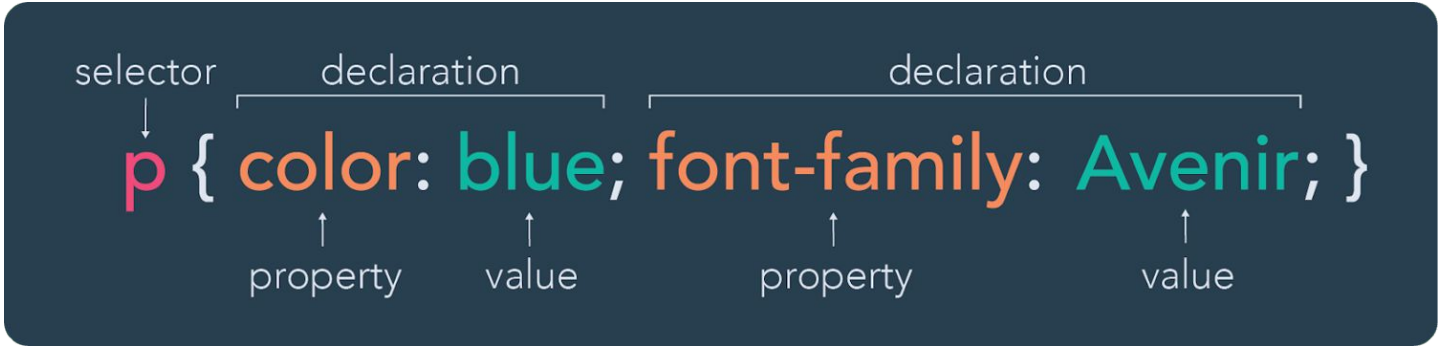
- Why do we use CSS?
  - **CSS** is a stylesheet language used to describe the presentation of am HTML document.
  - CSS defines how elements should be rendered on screen.. It is one of the cornerstone technologies of the Web, alongside HTML and JavaScript.

CSS - Methods

- Three typical implementations of CSS:
  - Inline:
    - Not a recommended method
    - Includes CSS directly in our HTML elements
  - Internal:
    - Using style element in the head section of the HTML file
    - <style> tag
  - External:
    - Recommended method
    - External stylesheet - Creating a style sheet “style.css” and link it in the HTML file
      - “<link rel=”stylesheet” href=”style.css

CSS - Syntax

- Main components:
  - Selector
  - Declaration
    - CSS property
    - Value



CSS - Selectors

- To style an HTML element you must first select the element, using 3 different methods of selection:
  - Element:
    - Selecting by using the HTML element name
    - h1 {font-size: 20px}
  - Class:
    - Selecting by using the class attribute
    - “class=’container’”
    - .container {margin: 10px}
  - ID:
    - Selecting by using the ID attribute
    - “id=’paragraph’”
    - #paragraph {color: green}

CSS - Selectors

```css
/* Target class name */
.class-name {
  css-style: css-value
}

/* Target tag */
tag-name {
  css-style: css-value
}
```

CSS - Fonts

- Documentation
- Properties:
  - font-family
  - font-size
  - font-stretch
  - font-style
  - font-variant
  - font-weight
  - line-height

Example View	font-family
font-family	Arial, sans-serif
font-family	Comic Sans MS, cursive
font-family	Times New Roman, serif
font-family	Courier New, monosapce
font-family	Georgia, serif
font-family	Impact, sans-serif

CSS - Fonts

```css
body {
  background-color: rgb(245, 245, 245);

  font-family: Arial, Helvetica, sans-serif;
  font-size: 16px;
  font-weight: normal;
  line-height: 1.6em;
}
```

CSS - Color

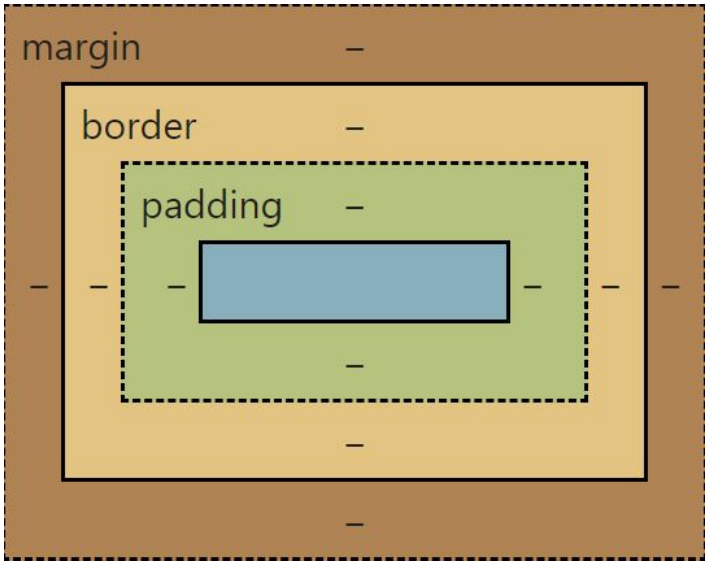
- Multiple ways to style in CSS:
  - Color names:
    - E.g. - color:blue
  - #HEX code:
    - #RRGGBB
    - E.g. - #262E69
  - RGB value:
    - rgb(red, green, blue)
    - E.g. - rgb(0, 0, 255)

CSS - Color

```css
.paragraph-1 {
  color: white;

  /* HTML 5 color */
  background-color: coral;

  /* RGB color */
  background-color: rgb(255, 127, 80);

  /* Hexadecimal color */
  background-color: #ff7f50;
}
```

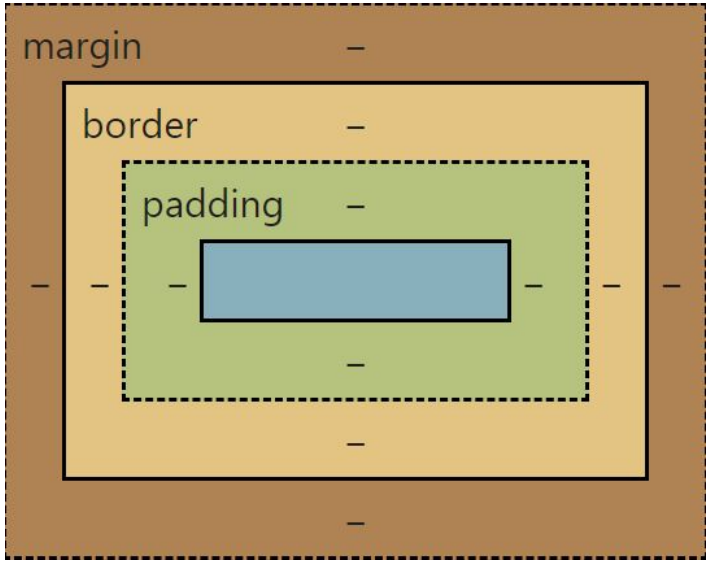
CSS - Layout - Model

- CSS forms around the box model
- Model:
  - Blue - Content
    - Text and image
  - Green - Padding
  - Yellow - Border
  - Orange - Margin



CSS - Layout - Attributes

- CSS forms around the box model
- Attributes:
  - Content
    - width & height
    - Usually percentages of parent container
  - Padding:
    - padding: top, right, bottom, left
    - padding-left: 10px
  - Border:
    - border: size, type, color
  - Margin:
    - Same as padding, with 1 to 4 values
    - margin: 10px 20px 30px 40px



CSS - Layout - Attributes - Margin

```css
.container {
  margin: 16px;

  margin-top: 16px;
  margin-right: 16px;
  margin-bottom: 16px;
  margin-left: 16px;

  /* Margin top/bottom (32px) and right/left (16px) */
  margin: 32px 16px;
}
```

CSS - Layout - Attributes - Padding

```css
.container {
  padding: 16px;

  padding-top: 16px;
  padding-right: 16px;
  padding-bottom: 16px;
  padding-left: 16px;

  /* padding top/bottom (32px) and right/left (16px) */
  padding: 32px 16px;
}
```

CSS - Layout - Attributes - Border

```css
.categories li {
  border-bottom: 1px solid rgb(0, 255, 34);
}

.products {
  border: 1px dashed rgb(0, 151, 161);
  border-radius: 5px;
}
```

CSS - Flexbox

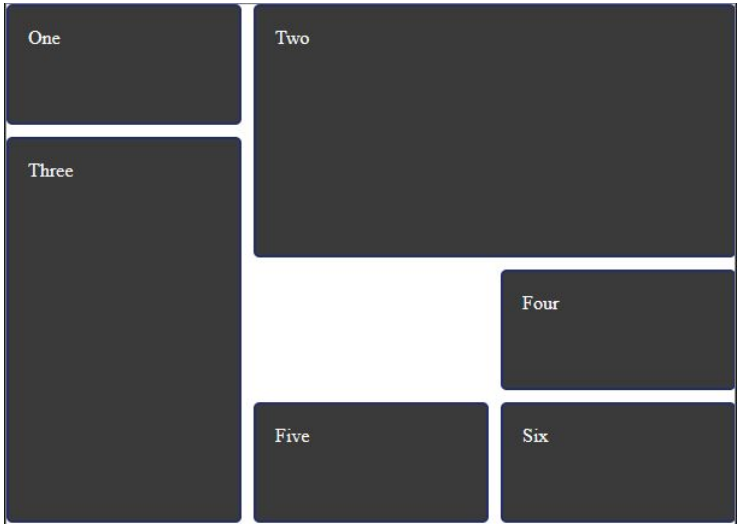
- CSS flexbox layout:
  - Main axis & Cross axis:
    - row
    - row-reverse
    - column
    - Column-reverse
- Syntax:
  - display: flex
  - flex-grow;
  - flex-shrink
  - flex-basis
- Documentation

CSS - Flexbox

```css
.flex-box {
  display: flex;
  flex-wrap: wrap;
}

.flex-item {
  width: 30%;
  margin: 16px auto;
}
```

CSS - Grid

- CSS grid layout:
  - Diving a page into major regions
  - Defining the relationship in terms of:
    - Size
    - Position
    - Layer
  - Align elements:
    - Column
    - Row



- Documentation

CSS - Grid

```css
.grid {
  display: grid;
  grid-template-columns: 30% 30% 30%;
  column-gap: 5%;
}

.grid-item {
  margin: 16px 0px;
}
```

CSS - Additional Information

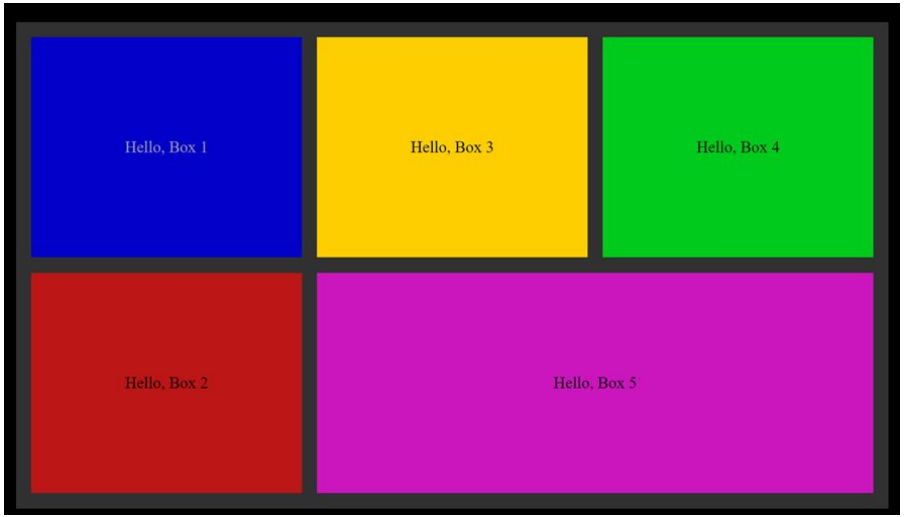
- Specificity:
  - IF two or more rules point to the same element, the selector with the highest specificity will take preference.
    - id > class > element
- Pseudo-classes:
  - Keyword added to a selector that specifies a special state of the selected element.
    - hover
    - visited
    - active
    - Checked
  - Documentation

HTML & CSS

Example

Exercise - Grid Exercise

- Firstly we'll create the below grid using CSS & HTML
- Then we'll recreate using Bootstrap grid



HTML & CSS - Example

```html
<!--Color grid-->

<div class="color-grid">

    <div class="color-grid-item box-1">
        <div class="box-text">
            Hello box-1
        </div>
    </div>

    <div class="color-grid-item box-2">
        <div class="box-text">
            Hello box-2
        </div>
    </div>

    <div class="color-grid-item box-3">
        <div class="box-text">
            Hello box-3
        </div>
    </div>

</div>
```

HTML & CSS - Example

```css
.color-grid {
  display: grid;
  grid-template-columns: 34% 33% 33%;
  border: 5px rgb(80, 80, 80) solid;
}

.color-grid-item {
  height: 250px;
  display: flex;
  align-items: center;
  border: 5px rgb(80, 80, 80) solid;
}

.box-text {
  margin: auto;
}

.box-1 {
  background-color: blue;
  color: white;
}

.box-2 {
  background-color: rgb(4, 197, 4);
  color: black;
}

.box-3 {
  background-color: rgb(207, 99, 211);
  color: black;
  grid-column: 2 / span 2;
}
```

JavaScript

Connecting HTML & CSS

JavaScript - HTML - CSS

- Incorporating HTML, CSS and JavaScript into a single interactive, aesthetic web page.
- Three technologies, and their purpose:
  - HTML:
    - Foundation
  - CSS:
    - Styling
  - JavaScript:
    - Interactivity

JavaScript - HTML - CSS

- Incorporating all languages:
  - Inside HTML
    - Use the <style> tag for CSS
    - Use the <script> tag for JavaScript
- For cleaner code you generate separate files and import them:
  - CSS:
    - Create “style.css”
    - Import “<link rel=”stylesheet” href=”style.css”>
  - JavaScript:
    - Create “index.js”
    - Import “<script src=”index.js”></script>”

Bootstrap

CSS Framework

Bootstrap

- CSS framework
- Why use Bootstrap:
  - Popular HTML, CSS and JavaScript Library
  - HTML & CSS based design templates
  - Responsive designs

- Official Documentation

Bootstrap - Installation

- Installing via package manager [npm]
  - Run:

```sh
npm install bootstrap
```

Bootstrap - Installation

- OR via CDN

```html
<link
  href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
  rel="stylesheet"
  integrity="sha384-QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwIH"
  crossorigin="anonymous"
/>
```

```html
<script
  src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
  integrity="sha384-YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
  crossorigin="anonymous"
></script>
```

Bootstrap - Buttons

- Buttons can be created in two ways in Bootstrap:
  - Tags:
    - <a>
    - <button>
- Colours:
  - btn-default
  - btn-primary
  - btn-success
  - btn-info
  - btn-warning
  - btn-danger
  - btn-link

Bootstrap - Buttons

- Sizes:
  - btn-lg
  - btn-sm
  - btn-xs
  - btn-block

Bootstrap - Buttons

```html
<button type="button" class="btn">Basic</button>
<button type="button" class="btn btn-default">Default</button>
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-link">Link</button>
```

Bootstrap - Tables

- Bootstrap has multiple types of tables:
  - Basic: .table
  - Striped: .table-striped
  - Hiverable rows: .table-hover
  - Hover state with <tbody>
  - Active: .table-active
  - Highlights a row or cell
  - Bordered: .table.bordered
  - Borderless: .table-borderless
  - Small: .table-sm
  - Cuts cell padding
- Syntax:
  - <table class=”table table-bordered”><table>

Bootstrap - Tables

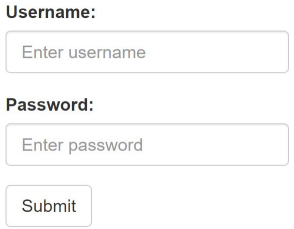
```html
<table class="table">
    <thead>
        <tr>
            <th>Heading 1</th>
            <th>Heading 2</th>
            <th>Heading 3</th>
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>Column 1</td>
            <td>Column 2</td>
            <td>Column 3</td>
        </tr>
        <tr>
            <td>Column 1</td>
            <td>Column 2</td>
            <td>Column 3</td>
        </tr>
    </tbody>
</table>
```

Bootstrap - Forms

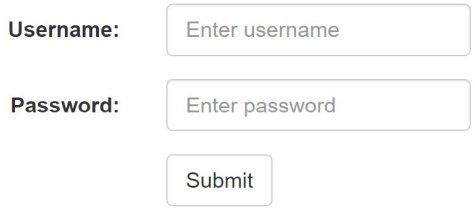
- Form layouts
  - Vertical (Default)
  - Horizontal
  - Inline
- Initialising:
  - <form> tag
  - Wrap all the labels and form controls inside the tag <div class=”form-group”
  - Inputs:
    - <label for=”label”>”Lable Name></label>
    - <input type=”label” class=”form-control” id=”label”>

Bootstrap - Forms

- Vertical
  - <form action=””></form>



- Horizontal
  - <form action=”” class=”form-horizontal></form>
  - Additional rules:
  - .form-horizontal to <form> element
  - .control-label to <label> elements



Bootstrap - Forms

```html
<form action="/action_page.php">
  <div class="form-group">
    <label for="email">Username:</label>
    <input type="email" class="form-control" id="email">
  </div>
  <div class="form-group">
    <label for="pwd">Password:</label>
    <input type="password" class="form-control" id="pwd">
  </div>
  <button type="submit" class="btn btn-default">Submit</button>
</form>
```

Bootstrap - Forms

- Inline
  - <form class="form-inline" action="/action_page.php">
  - Additional rule:
    - .form-inline to <form> element

Username: Enter username **Password:** Enter password Submit

Bootstrap - Dropdowns

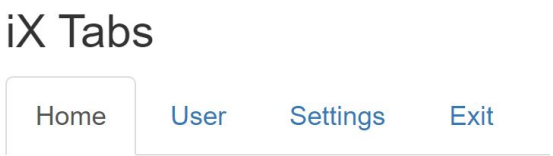
- Dropdowns
  -
  - Properties:
    - Divider:
      - .divider
    - Header:
      - .dropdown-header
    - Disable and Active items:
      - .disabled
    - Alignment:
      - .dropdown-menu-right

Bootstrap - Dropdowns

```html
<div class="dropdown">
  <button class="btn btn-primary dropdown-toggle" type="button" data-toggle="dropdown">
    Dropdown Example
  <span class="caret"></span></button>
  <ul class="dropdown-menu">
    <li class="dropdown-header">Dropdown header 1</li>
    <li><a href="#">Item 1</a></li>
    <li class="dropdown-header">Dropdown header 2</li>
    <li><a href="#">Item 2</a></li>
  </ul>
</div>
```

Bootstrap - Tabs

- Tabs
  - Syntax:
    - <ul class=”nav nav-tabs”>
    - Items: ”Tab Name”
    - Mark current page: <li class=”active”>



Bootstrap - Tabs

```html
<ul class="nav nav-tabs">
    <li class="active"><a href="#">Home</a></li>
    <li><a href="#">User</a></li>
    <li><a href="#">Settings</a></li>
    <li><a href="#">Exit</a></li>
</ul>
```

Bootstrap - Grid Example

```html
<div class="container mt-5">
    <h3>
        Bootstrap Tutorial
    </h3>
    <div class="mt-3">
        <div class="container">
            <div class="row">
                <div class="col-4 color-grid-item box-1">
                    <div class="box-text">
                        Hello box-1
                    </div>
                </div>
                <div class="col-4 color-grid-item box-2">
                    <div class="box-text">
                        Hello box-2
                    </div>
                </div>
                <div class="col-4 color-grid-item box-3">
                    <div class="box-text">
                        Hello box-3
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
```

Bootstrap - Grid Example

```css
.color-grid-item {
  height: 250px;
  display: flex;
  align-items: center;
  border: 5px rgb(80, 80, 80) solid;
}

.box-text {
  margin: auto;
}

.box-1 {
  background-color: blue;
  color: white;
}

.box-2 {
  background-color: yellow;
  color: black;
}

.box-3 {
  background-color: rgb(4, 197, 4);
  color: black;
}
```

Bootstrap - Icons

- Install Bootstrap icons:

```sh
npm i bootstrap-icons
```

- Syntax:

```html
<i class="bi bi-airplane"></i>
```

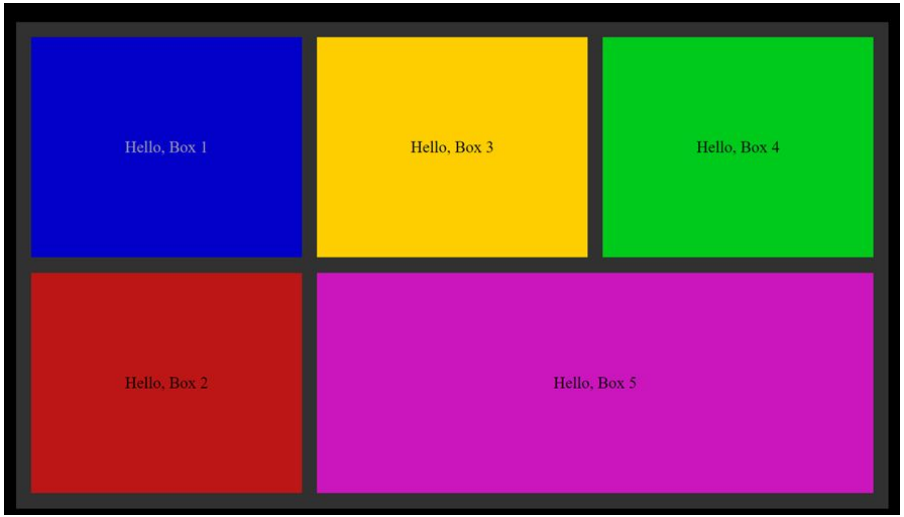
- Documentation

Exercise

HTML, CSS & Bootstrap

Exercise - Grid Exercise

- Firstly we'll create the below grid using CSS & HTML
- Then we'll recreate using Bootstrap grid



Homework

Apply what we have learned

Homework

- Use Bootstrap to update the styling of your personal portfolio web page from the pre-work exercise.
- Recap Day 1 slides & review day 2 slides.
- Install node JS

Next Class

We will dive into Javascript and Version Control