**Personal Blog on IBM Cloud Static Web Apps**

**Phase4: Development part-2**

**Team Members:**

**Sahaya Amir Vishwa M**

**Santhiyagu Jeffrin C**

**Praveen S**

**Eugene Ebenezer A**

**Stalin J**

**Sooriya R C**

**Planning:**

* This is the initial stage where you define the purpose and structure of your website.
* Decide on the website's theme, goals, and target audience.
* Consider the number of pages you want, such as a homepage, blog posts, an about page, and a contact page.
* Planning helps you create a clear vision for your website.

**HTML Structure:**

* HTML (HyperText Markup Language) is the fundamental language for creating the structure and content of your web pages.
* In this step, you create the basic structure of your website by using HTML tags. This includes defining the overall layout of your site, creating a header, navigation menu, main content area, and a footer.
* You link to your external CSS file to apply styling to your HTML structure.

**CSS Styling:**

* Cascading Style Sheets (CSS) is used to control the presentation and layout of your web pages.
* In this step, you create a CSS file (usually named "style.css") where you define the visual appearance of your website.
* This includes specifying fonts, colors, backgrounds, margins, and padding.
* By linking your HTML file to the CSS file, you can control the design of your site consistently across all pages.

**Content:**

* Content is the heart of your travel blog website.
* You need to create engaging and informative content, which typically includes travel stories and photos from your adventures.
* The content should be well-organized and divided into blog posts.
* You can create your content in various formats, such as text, images, videos, and more.

**Blog Posts:**

* + Each blog post should have its own HTML page.
* The HTML structure for a blog post typically includes a title, date, images, and the body of your travel story.
* You can create separate HTML files for each blog post to keep your website organized.

**Navigation:**

* You should include a navigation menu in the header of your website to help users easily navigate between different pages.
* The navigation menu links to the various sections of your website, such as the homepage, blog posts, about page, and contact page.
* Use relative URLs to ensure that the links work correctly.

**Testing:**

* After coding your website, it's crucial to test it in different web browsers and on various devices (desktop, tablet, and mobile) to ensure that it appears and functions correctly.
* Check for any layout issues, broken links, or other problems that may affect the user experience.

**Deployment:**

* Once your website is complete and tested, you can make it accessible to the public by deploying it.
* This involves choosing a web hosting service, purchasing a domain name (if you haven't already), and uploading your website files to the server.
* Popular hosting options include platforms like WordPress, Wix, or custom hosting using services like Netlify or GitHub Pages.
* This step ensures that your website is live and can be accessed by visitors.

**Planning:**

**html**

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Your Travel Blog</title>

<link rel="stylesheet" href="style.css">

</head>

<body>

<header>

<h1>Welcome to Your Travel Blog</h1>

<p>Your gateway to wanderlust and adventure</p>

</header>

<nav>

<ul>

<li><a href="index.html">Home</a></li>

<li><a href="blog.html">Blog</a></li>

<li><a href="about.html">About</a></li>

<li><a href="contact.html">Contact</a></li>

</ul>

</nav>

<main>

<article>

<h2>Featured Post Title</h2>

<p>Published on: Date</p>

<img src="featured-image.jpg" alt="Featured Image Description">

<p>A captivating introduction to the featured blog post. Tease the reader with a glimpse of your travel adventure.</p>

<a href="featured-post.html">Read More</a>

</article>

<section>

<h2>Recent Blog Posts</h2>

<div class="blog-post">

<img src="post1-image.jpg" alt="Post 1 Image Description">

<h3>Post 1 Title</h3>

<p>Published on: Date</p>

<p>A brief description of the content in this post. Invite readers to explore more.</p>

<a href="post1.html">Read More</a>

</div>

<div class="blog-post">

<img src="post2-image.jpg" alt="Post 2 Image Description">

<h3>Post 2 Title</h3>

<p>Published on: Date</p>

<p>Another enticing description of the content in this post. Encourage readers to dive in.</p>

<a href="post2.html">Read More</a>

</div>

</section>

</main>

<footer>

<p>&copy; 2023 Your Travel Blog</p>

</footer>

</body>

</html>

**Post1.html**

<!DOCTYPE html>

<html>

<head>

<title>Post 1</title>

</head>

<body>

<header>

<h1>Post Title</h1>

<p>Posted on October 18, 2023</p>

</header>

<article>

<p>This is the content of Post 1. Replace this text with your actual post content.</p>

</article>

<footer>

<p>Tags: <a href="#">Tag 1</a>, <a href="#">Tag 2</a></p>

</footer>

</body>

</html>

**Post2.html**

<!DOCTYPE html>

<html>

<head>

<title>Post 1</title>

</head>

<body>

<header>

<h1>Post Title</h1>

<p>Posted on October 18, 2023</p>

</header>

<article>

<p>This is the content of Post 1. Replace this text with your actual post content.</p>

</article>

<footer>

<p>Tags: <a href="#">Tag 1</a>, <a href="#">Tag 2</a></p>

</footer>

</body>

</html>

**1.Homepage:**

* + - A brief introduction to your travel blog and its theme.
    - A featured blog post with a catchy title and a teaser.
    - A selection of captivating travel images in a slideshow or grid.
    - Links to your most recent blog posts.

**html**

<article>

<h2>Welcome to Your Travel Blog</h2>

<p>Your journey to exploration and adventure begins here. We invite you to explore captivating destinations, immerse yourself in thrilling travel stories, and discover valuable tips to plan your own remarkable journeys.</p>

<a href="blog.html">Explore Our Blog</a>

</article>

**2.Blog Posts**

* Each blog post should have its own HTML page, and the content for each post should include:
* A descriptive and engaging title.
* The date of the travel experience.
* High-quality photos or images relevant to the travel story.
* The travel story itself, including details about the destination, your experiences, recommendations, and any interesting anecdotes.
* Consider using subheadings, bullet points, and paragraphs to break up the text and make it more readable.
* A call to action at the end, encouraging readers to leave comments, share the post on social media, or explore other posts on your blog.

**html**

<article>

<h2>Post Title</h2>

<p>Published on: Date</p>

<img src="image.jpg" alt="Description of the image">

<p>Your travel story goes here. Share your experiences, recommendations, and captivating tales from your journey. Make it engaging and informative, transporting readers to the destination.</p>

</article>

**3.About Page:**

* Information about you as the author of the travel blog, including your name, background, and interests.
* Details about your travel experiences, what inspires you to travel, and any unique travel-related achievements or affiliations.
* A personal photo of yourself.
* Contact information or links to your social media profiles.

**Html**

<article>

<h2>About Us</h2>

<p>We are passionate travelers, adventurers, and storytellers. Our love for exploration has taken us to breathtaking destinations around the world. We're here to share our experiences and inspire your wanderlust.</p>

<p>Meet the team:</p>

<ul>

<li>Author Name 1 - Bio and background.</li>

<li>Author Name 2 - Bio and background.</li>

</ul>

</article>

**4.Contact Page:**

**Html**

* A contact form for readers to get in touch with you, including fields for their name, email address, subject, and message.
* An email address or other contact information if you prefer direct email communication.
* A brief message welcoming feedback, questions, and collaboration inquiries.
* Information about your response time and a commitment to maintaining user privacy.

<article>

<h2>Contact Us</h2>

<p>We'd love to hear from you. Whether you have questions, feedback, or collaboration inquiries, don't hesitate to reach out. Your thoughts and ideas are important to us.</p>

<form>

<label for="name">Name:</label>

<input type="text" id="name" name="name" required>

<label for="email">Email:</label>

<input type="email" id="email" name="email" required>

<label for="subject">Subject:</label>

<input type="text" id="subject" name="subject" required>

<label for="message">Message:</label>

<textarea id="message" name="message" rows="4" required></textarea>

<button type="submit">Send Message</button>

</form>

</article>

**1.Setting Up Jekyll:**

You'll need to have Ruby installed on your system. You can install Jekyll via RubyGems using the following command:

bash

gem install jekyll bundler

**2. Create a Jekyll Site:**

bash

jekyll new my-travel-blog

cd my-travel-blog

**3. Content Organization:**

Jekyll expects your content to be organized in Markdown files in the \_posts directory. Here's a sample structure:

my-travel-blog/

├── \_posts/

| ├── 2023-10-29-welcome-to-my-blog.md

| ├── 2023-10-30-post-1.md

| ├── 2023-11-01-post-2.md

├── \_layouts/

| ├── default.html

├── \_config.yml

**4. Customize Templates:**

Modify the default layout in \_layouts/default.html and create other templates as needed. Jekyll uses Liquid templating syntax for templates.

**5. Build Your Site:**

bash

bundle exec jekyll build

**6. Deployment:**

Your Jekyll-generated site will be in the \_site directory. You can deploy the contents of this directory to your IBM Cloud Static Web App repository.

**### Using Hugo:**

**1. Setting Up Hugo:**

Download and install Hugo from the official website (https://gohugo.io/).

**2. Create a Hugo Site:**

bash

hugo new site my-travel-blog

cd my-travel-blog

**3. Content Organization:**

Hugo expects your content to be in Markdown files within the content directory. Here's a sample structure:

my-travel-blog/

├── content/

| ├── \_index.md

| ├── post-1.md

| ├── post-2.md

├── layouts/

| ├── \_default/

| | ├── single.html

├── config.toml

**4. Customize Templates:**

Modify the templates in the layouts directory. Hugo uses Go templating syntax for templates.

**5. Build Your Site:**

bash

hugo

**6. Deployment:**

Your Hugo-generated site will be in the public directory. You can deploy the contents of this directory to your IBM Cloud Static Web App repository.

Creating content for a static web app involves creating HTML, CSS, and potentially JavaScript files to structure and style your website. Below, I'll provide a simplified example of content for a static web app, including an index.html file, a style.css file, and a simple JavaScript file.

**1. index.html (Sample Web Page):**

html

<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>My Travel Blog</title>

<link rel="stylesheet" type="text/css" href="style.css">

</head>

<body>

<header>

<h1>Welcome to My Travel Blog</h1>

</header>

<nav>

<ul>

<li><a href="/">Home</a></li>

<li><a href="/about.html">About</a></li>

<li><a href="/blog.html">Blog</a></li>

<li><a href="/contact.html">Contact</a></li>

</ul>

</nav>

<main>

<section>

<h2>Latest Blog Posts</h2>

<ul>

<li>

<a href="/blog-post-1.html">Exploring Paris</a>

</li>

<li>

<a href="/blog-post-2.html">Hiking in the Amazon</a>

</li>

</ul>

</section>

</main>

<footer>

<p>&copy; 2023 My Travel Blog</p>

</footer>

</body>

</html>

**2. style.css (Sample CSS):**

css

body {

font-family: Arial, sans-serif;

margin: 0;

padding: 0;

}

header {

background-color: #333;

color: #fff;

text-align: center;

padding: 20px;

}

nav ul {

list-style: none;

padding: 0;

}

nav li {

display: inline;

margin-right: 20px;

}

main {

margin: 20px;

}

section {

padding: 20px;

}

footer {

background-color: #333;

color: #fff;

text-align: center;

padding: 10px;

}

**App.js**

// Wait for the DOM to be fully loaded before executing the JavaScript code

document.addEventListener("DOMContentLoaded", function () {

// Get all the like buttons on the page

const likeButtons = document.querySelectorAll(".like-button");

// Add a click event listener to each like button

likeButtons.forEach(function (button) {

button.addEventListener("click", function () {

// Find the parent post container

const postContainer = button.closest(".post-container");

if (postContainer) {

// Find the like count element within the post container

const likeCount = postContainer.querySelector(".like-count");

if (likeCount) {

// Get the current like count value

let currentLikes = parseInt(likeCount.textContent);

// Increment the like count

currentLikes++;

// Update the like count on the page

likeCount.textContent = currentLikes;

}

}

});

});

});