PHASE- 4

**PERSONAL BLOG ON IBM CLOUD STATIC WEB APPS**

**DESIGN INNOVATION**

converting the HTML content into a template that can be easily updated using Strapi

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<title>TravelLog</title>

<link rel="icon" type="image/png" href="Images/alps\_favicon.png">

<meta name="description" content="Connect with people over travelling">

<meta name="viewport" content="width=device-width, initial-scale=1">

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

<!-- google font api -->

<link rel="preconnect" href="https://fonts.gstatic.com">

<link href="https://fonts.googleapis.com/css2?family=Merriweather+Sans:wght@300&family=Montserrat&display=swap" rel="stylesheet">

<link href="https://fonts.googleapis.com/css2?family=Raleway:wght@500&display=swap" rel="stylesheet">

<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.0.13/css/all.css" integrity="sha384-DNOHZ68U8hZfKXOrtjWvjxusGo9WQnrNx2sqG0tfsghAvtVlRW3tvkXWZh58N9jp" crossorigin="anonymous">

<script src="JS/change-bg.js" async></script>

</head>

<body>

<!-- short content -->

<div id="content">

<header>

<h2><a href="#">TRAVELLOG</a></h2>

<nav>

<ul>

<% for (const link of navigationLinks) { %>

<li><a href="<%= link.url %>"><%= link.title %></a></li>

<% } %>

</ul>

</nav>

</header>

<div class="Tagline">

<h1><%= tagline.title %></h1>

<h3><%= tagline.description %></h3>

<br> <br>

<div class="buttons-flex">

<a href="<%= buttons.loginUrl %>" class="button"><%= buttons.loginText %></a>

<a href="<%= buttons.signupUrl %>" class="button"><%= buttons.signupText %></a>

</div>

</div>

</div>

<!-- footer section -->

<footer class="footer-distributed">

<div class="footer-top">

<p class="footer-links">

<% for (const footerLink of footerLinks) { %>

<a href="<%= footerLink.url %>"><%= footerLink.title %></a>

<% } %>

</p>

</div>

<div class="footer-icons">

<% for (const icon of socialIcons) { %>

<a href="<%= icon.url %>"><i class="fab <%= icon.iconClass %>"></i></a>

<% } %>

</div>

<hr>

<p class="footer-company-name"><strong><%= footerCompanyName %> &copy; <%= currentYear %></strong></p>

</footer>

</body>

</html>

**Integrate Strapi with Hugo:**

1. Initially we are setting up a Strapi project and defining our content such as blog posts, authors, or categories, using the Strapi admin panel.
2. Using the Strapi API to retrieve data from your Strapi project. we are using JavaScript to make API calls from our Hugo site.
3. Then we are creating Hugo templates and partials to render the data fetched from the Strapi API.
4. Using JavaScript and Hugo shortcode to consume the Strapi API and render dynamic content on our Hugo site.

**Integrating Strapi with Hugo:**

1. Set up your Strapi project and define content types:
   * Install Strapi globally using npm: **npm install strapi@beta -g**.
   * Create a new Strapi project: **strapi new myProject --quickstart**.
   * Define your content types in the Strapi admin panel.
2. Fetch data from Strapi in Hugo:
   * Use JavaScript to fetch data from your Strapi project's API.
   * Create a script in your Hugo project to handle data fetching and rendering.
3. Render dynamic content in Hugo:
   * Create Hugo templates and partials to render the fetched data.
   * Use Hugo shortcodes or templates to display dynamic content on your Hugo site.

**Javascript code**

// File: ./api/destination/models/Destination.settings.json

{

"collectionName": "destinations",

"info": {

"name": "Destination",

"description": ""

},

"options": {

"increments": true

},

"attributes": {

"title": {

"type": "string"

},

"description": {

"type": "text"

},

"image": {

"model": "file"

},

"country": {

"type": "string"

},

"region": {

"type": "string"

}

}

}

// File: ./api/blog-post/models/BlogPost.settings.json

{

"collectionName": "blogposts",

"info": {

"name": "BlogPost",

"description": ""

},

"options": {

"increments": true

},

"attributes": {

"title": {

"type": "string"

},

"content": {

"type": "richtext"

},

"author": {

"type": "string"

},

"date": {

"type": "date"

},

"destination": {

"model": "destination"

}

}

}

// File: ./api/author/models/Author.settings.json

{

"collectionName": "authors",

"info": {

"name": "Author",

"description": ""

},

"options": {

"increments": true

},

"attributes": {

"name": {

"type": "string"

},

"bio": {

"type": "text"

},

"profilePicture": {

"model": "file"

}

}

}

**Destination Model**

File: **./api/destination/models/Destination.settings.json**

{

"collectionName": "destinations",

"info": {

"name": "Destination",

"description": "A travel destination for the blog"

},

"options": {

"increments": true

},

"attributes": {

"title": {

"type": "string"

},

"description": {

"type": "text"

},

"image": {

"model": "file"

},

"country": {

"type": "string"

}

}

}

**BlogPost Model:**

File: **./api/blogpost/models/BlogPost.settings.json**

{

"collectionName": "blogposts",

"info": {

"name": "BlogPost",

"description": "A blog post about a travel experience"

},

"options": {

"increments": true

},

"attributes": {

"title": {

"type": "string"

},

"content": {

"type": "richtext"

},

"author": {

"model": "author"

},

"destination": {

"model": "destination"

}

}

}

**Author Model:**

{

"collectionName": "authors",

"info": {

"name": "Author",

"description": "An author contributing to the blog"

},

"options": {

"increments": true

},

"attributes": {

"name": {

"type": "string"

},

"bio": {

"type": "text"

},

"profilePicture": {

"model": "file"

}

}

}

**CONCLUSION**

Building a travel blog using Strapi enables me to create dynamic content, manage diverse data types, and establish relationships between various entities like destinations, blog posts, and authors. By defining custom models and configurations, I can organize and structure my content effectively, providing a seamless experience for both content creators and end-users. Strapi's flexibility and ease of use make it a powerful tool for developing and managing dynamic websites and applications.