**Report on Static Web Page with Splash Screen and Dynamic Data Fetching**

**1. Introduction**

The project titled **"Static Web Page with Splash Screen and Dynamic Data Fetching"** is a web-based application that demonstrates the integration of dynamic content fetching and static web design. It uses technologies like **HTML**, **CSS**, **JavaScript**, and the **Bootstrap framework** to create a responsive and interactive web page. The primary goal of this project is to display a splash screen upon the first page load, which includes a "Fetch Data" button. Once clicked, the page fetches data from an external API and transitions into the main content of the website.

**2. Objective**

The primary objective of this project is to:

1. Create an interactive splash screen that appears when the user first accesses the website.
2. Implement a "Fetch Data" button on the splash screen that, when clicked, fetches data from an external API and displays it.
3. Transition to the main page after the data has been fetched.
4. Ensure the website is **responsive**, adapting seamlessly across devices (mobile, tablet, and desktop).

**3. Methodology**

To achieve the goals of this project, the following methodology was employed:

1. **HTML**:
   * The structure of the web page was created using HTML, where the splash screen is designed initially, followed by the main content.
   * Dynamic content (data fetched from the API) is inserted into the HTML structure using JavaScript.
2. **CSS**:
   * Custom styling was added using internal CSS to style the splash screen, navigation bar, and other sections.
   * Bootstrap classes were used for responsive layout management, ensuring that the web page looks good across various devices.
3. **JavaScript**:
   * JavaScript was used to handle the click event for the "Fetch Data" button.
   * The **XMLHttpRequest** object was used to send a GET request to the API and retrieve data.
   * The splash screen is hidden, and the main content is displayed after the data is successfully fetched.
   * Error handling was implemented to display error messages if the data fetch fails.
4. **Bootstrap**:
   * Bootstrap's grid system and pre-designed components (e.g., navigation bar, jumbotron) were used to make the page responsive and visually appealing.

**4. Code and Implementation Details**

The project consists of a single index.html file that combines **HTML**, **CSS**, and **JavaScript**. The splash screen initially covers the whole page and displays a button. When the button is clicked, JavaScript fetches data from the API, and the splash screen disappears to show the main page with the fetched data.

Here is the **code** used for the implementation:

**HTML, CSS, and JavaScript Combined in index.html**

html

Copy code

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

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

<title>Static Web Page with Splash Screen</title>

<!-- Bootstrap CSS -->

<link href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" rel="stylesheet">

<style>

body {

font-family: Arial, sans-serif;

}

#splashScreen {

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background-color: #f8f9fa;

}

#mainWebsite {

display: none;

}

</style>

</head>

<body>

<div id="splashScreen">

<div class="text-center">

<h1>Welcome to My Website</h1>

<button id="fetchDataBtn" class="btn btn-primary mt-3">Fetch Data</button>

<div id="fetchDataContainer" class="mt-3"></div>

</div>

</div>

<div id="mainWebsite">

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<a class="navbar-brand" href="#">Navbar</a>

<button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarNav" aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">

<span class="navbar-toggler-icon"></span>

</button>

<div class="collapse navbar-collapse" id="navbarNav">

<ul class="navbar-nav">

<li class="nav-item active"><a class="nav-link" href="#">Home</a></li>

<li class="nav-item"><a class="nav-link" href="#">Features</a></li>

<li class="nav-item"><a class="nav-link" href="#">Pricing</a></li>

</ul>

</div>

</nav>

<header class="jumbotron text-center">

<h1 class="display-4">Welcome to My Static Web Page</h1>

<p class="lead">This is a simple, static web page using HTML, CSS, and Bootstrap.</p>

</header>

<div class="container">

<div class="row">

<div class="col-md-8">

<h2>Main Content</h2>

<p>This is the main content area.</p>

</div>

<div class="col-md-4">

<h2>Sidebar</h2>

<p>This is the sidebar content area.</p>

</div>

</div>

</div>

<footer class="text-center mt-4">

<p>&copy; 2024 Your Name</p>

</footer>

</div>

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"></script>

<script src="https://cdn.jsdelivr.net/npm/@popperjs/core@2.5.4/dist/umd/popper.min.js"></script>

<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js"></script>

<script>

document.getElementById('fetchDataBtn').addEventListener('click', function () {

fetchData();

});

function fetchData() {

const xhr = new XMLHttpRequest();

xhr.open('GET', 'https://jsonplaceholder.typicode.com/posts/1', true);

xhr.onload = function () {

if (this.status === 200) {

const post = JSON.parse(this.responseText);

document.getElementById('fetchDataContainer').innerHTML = `

<div class="alert alert-success">

<strong>Data Fetched:</strong> ${post.title}

</div>

`;

setTimeout(function () {

document.getElementById('splashScreen').style.display = 'none';

document.getElementById('mainWebsite').style.display = 'block';

}, 2000);

} else {

document.getElementById('fetchDataContainer').innerHTML = `

<div class="alert alert-danger">Error fetching data. Please try again.</div>

`;

}

};

xhr.onerror = function () {

document.getElementById('fetchDataContainer').innerHTML = `

<div class="alert alert-danger">Network Error. Please check your connection.</div>

`;

};

xhr.send();

}

</script>

</body>

</html>

**5. Results and Observations**

1. **Splash Screen**:
   * The splash screen displays initially, covering the full viewport with a message and a button.
   * Once the button is clicked, the web page fetches data from the API.
2. **API Data Fetching**:
   * The data is successfully fetched from the API (https://jsonplaceholder.typicode.com/posts/1).
   * The title of the fetched data is displayed on the splash screen for a short duration.
3. **Main Content Display**:
   * After the data is fetched, the splash screen disappears, and the main website content appears, showing the navigation bar, main content area, and footer.
4. **Error Handling**:
   * If there is an error fetching the data (e.g., network issues), an error message is displayed.

**6. Conclusion**

This project successfully implements a static web page that integrates **dynamic data fetching** and **interactive features**. By utilizing **HTML**, **CSS**, **JavaScript**, and **Bootstrap**, the web page provides an engaging user experience through the splash screen and smooth data loading. The project demonstrates the capability to combine static web design with dynamic content fetching, making it ideal for showcasing basic web development skills.

The splash screen functionality provides an intuitive way to fetch data and transition to the main page, while error handling ensures a reliable user experience. The project can be further extended to include more dynamic features, advanced styling, or integration with real-time data sources.