

FETCH API

File name: Index.html

Code:

```
<!DOCTYPE html>

<html lang="en">
  <head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Blog & Todo Fetch App</title>
    <link rel="stylesheet" href="style.css">
  </head>
  <body>
    <header>
      <h1>Blog Posts & Todo List</h1>
    </header>

    <section id="posts">
      <h2>Blog Posts</h2>
      <div id="postsContainer"></div>
    </section>

    <section id="todos">
      <h2>Todo List</h2>
      <div id="todosContainer"></div>
    </section>

    <script src="script.js"></script>
  </body>
</html>
```

File name: style.css

Code:

```
body {  
    font-family: 'Segoe UI', sans-serif;  
    background-color: #f4f4f9;  
    margin: 0;  
    padding: 0;  
}  
  
header {  
    background-color: #4CAF50;  
    color: white;  
    text-align: center;  
    padding: 15px 0;  
}  
  
section {  
    margin: 20px;  
    background: #fff;  
    padding: 20px;  
    border-radius: 8px;  
    box-shadow: 0 2px 5px rgba(0,0,0,0.1);  
}  
  
h2 {  
    color: #333;  
}  
  
ul {  
    list-style-type: none;  
    padding: 0;  
}  
  
li {  
    margin: 10px 0;  
}
```

```
background: #e9f5e9;  
padding: 10px;  
border-radius: 6px;  
}  
.error {  
color: red;  
font-weight: bold;  
}
```

File Name: script.js

Code:

```
// Using Module Pattern for clean and maintainable code  
const App = (function() {  
    // API Endpoints  
    const postsAPI = "https://jsonplaceholder.typicode.com/posts";  
    const todosAPI = "https://jsonplaceholder.typicode.com/todos";  
  
    // Private function to fetch data from given API  
    async function fetchData(url) {  
        try {  
            const response = await fetch(url);  
            if (!response.ok) {  
                throw new Error(`Server Error: ${response.status}`);  
            }  
            const data = await response.json();  
            return data;  
        } catch (error) {  
            displayError(error.message);  
            return [];  
        }  
    }  
});
```

```
// Display blog posts

function displayPosts(posts) {
  const container = document.getElementById("postsContainer");
  container.innerHTML = ""; // Clear any existing content

  if (posts.length === 0) {
    container.innerHTML = `<p class="error">No posts available.</p>`;
    return;
  }

  container.innerHTML = "<h2>Latest Blog Posts</h2>";

  // Display first 5 posts
  posts.slice(0, 5).forEach(post => {
    const postCard = document.createElement("div");
    postCard.classList.add("card");
    postCard.innerHTML = `
      <h3>${post.title}</h3>
      <p>${post.body}</p>
    `;
    container.appendChild(postCard);
  });
}

// Display todo list

function displayTodos(todos) {
  const container = document.getElementById("todosContainer");
  container.innerHTML = ""; // Clear any existing content

  if (todos.length === 0) {
    container.innerHTML = `<p class="error">No todos found.</p>`;
  }
}
```

```

    return;
}

container.innerHTML = "<h2>✓ To-Do List</h2>";
]

const list = document.createElement("ul");

todos.slice(0, 10).forEach(todo => {

  const item = document.createElement("li");

  item.textContent = `${todo.title} — ${todo.completed ? "✓ Completed" : "✗ Pending"}`;

  list.appendChild(item);
});

container.appendChild(list);

}

// Display error message in both sections

function displayError(message) {

  const postsContainer = document.getElementById("postsContainer");

  const todosContainer = document.getElementById("todosContainer");

  postsContainer.innerHTML = `<p class="error">⚠ Error: ${message}</p>`;

  todosContainer.innerHTML = `<p class="error">⚠ Error: ${message}</p>`;

}

// Public method to initialize the app

async function init() {

  const posts = await fetchData(postsAPI);

  displayPosts(posts);

  const todos = await fetchData(todosAPI);

  displayTodos(todos);

}

// Return public methods (Module Pattern)

return {

  init: init
}

```

```

};

})();

//  Initialize the application when DOM is loaded

document.addEventListener("DOMContentLoaded", App.init);

```

The screenshot shows a web browser window with two tabs open. The active tab is titled "Blog Posts & Todo List". Below the title bar, there are two sections: "Blog Posts" and "Todo List".

Blog Posts

Latest Blog Posts

- sunt aut facere repellat provident occaecati excepturi optio reprehenderit
- quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto
- qui est esse**
- est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis possimus qui neque nisi nulla
- ea molestias quasi exercitationem repellat qui ipsa sit aut**
- et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem doloribus vel accusantium quis paratur molestiae porro eius odio et labore et velit aut
- eum et est occaecati**
- ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque ipsum iure quis sunt voluptatem rerum illo velit
- nesciunt quas odio**
- repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque

Todo List

To-Do List

- delectus aut autem — Pending
- quis ut nam facilis et officia qui — Pending
- fugiat veniam minus — Pending
- et porro tempora — Completed
- laboriosam mollitia et enim quasi adipisci quia provident illum — Pending
- qui ullam ratione quibusdam voluptatem quia omnis — Pending
- illo expedita consequatur quia in — Pending
- quo adipisci enim quam ut ab — Completed
- molestiae perspicacis ipsa — Pending
- illo est ratione doloremque quia maiores aut — Completed