```
console.clear();

// Function to fetch users using Promises

const fetchUsers = () => {

    // Return a new Promise. This allows us to handle success (resolve) and failure (reject) manually

    return new Promise((resolve, reject) => {

        // Use fetch API to get data from a fake REST API

        fetch("https://jsonplaceholder.typicode.com/users")

            // Handle the response object

            .then(response => {

                // If response is not OK (e.g., 404, 500 errors), reject the Promise

                if (!response.ok) {

                    reject("Network response was not ok");
                }

                // Otherwise, parse the response body as JSON and response.json() also returns a Promise

                return response.json();
            })

            // When JSON is successfully parsed, resolve the Promise with the data

            .then(data => resolve(data))

            // If any error occurs during fetch or parsing, reject the Promise

            .catch(error => reject(error));
```

```
    });
};


// Function to display the users on the webpage

const displayUsers = (users) => {

    // Create an empty string to store HTML content

    let output = "";

    // Loop through each user in the data array

    users.forEach(user => {

        // Append user details inside <ul> (unordered list) Each user will appear inside a separate box (styled with CSS)

        output += `

        <ul>

            <li>ID: ${user.id}</li>

            <li>Name: ${user.name}</li>

            <li>Email: ${user.email}</li>

            <li>City: ${user.address.city}</li>

        </ul>

        `;
    });

    // Insert the generated HTML into the div with id="users-container". This replaces the content of that div with our users list

```

```
    document.getElementById("users-container").innerHTML = output;
};


// Call fetchUsers function

fetchUsers()

    // If the Promise is resolved successfully, display the users

    .then(users => displayUsers(users))

    // If the Promise is rejected (error occurred), log the error

    .catch(error => console.log(error))

    // finally() runs no matter what → whether resolved or rejected. This is useful for cleanup tasks or status messages

    .finally(() => console.log("The operation is completed."));
```