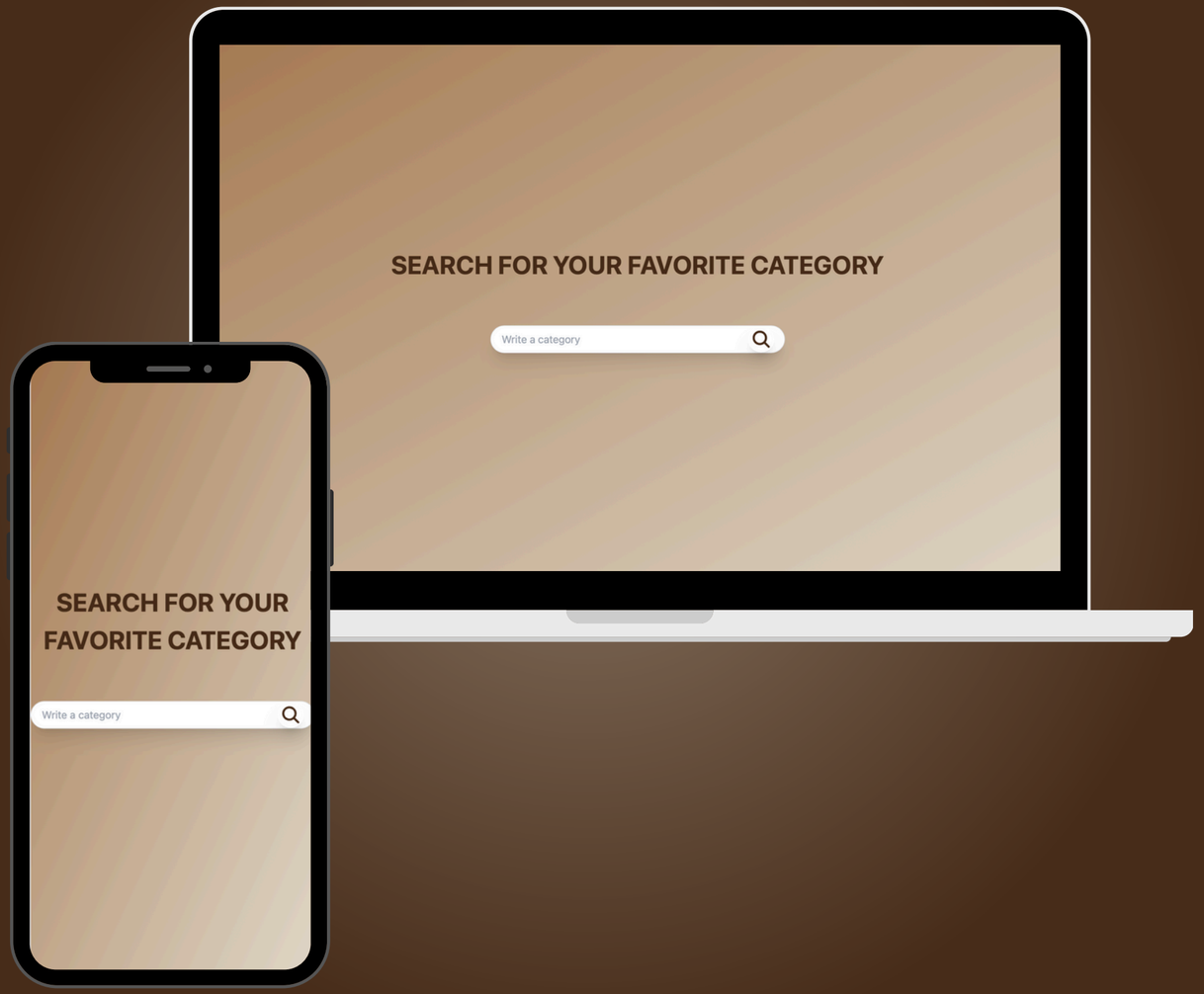
JAVASCRIPT

MIRIAM TRUIOLO

FRONT END DEVELOPMENT



BOOK APP

GitHub

Project

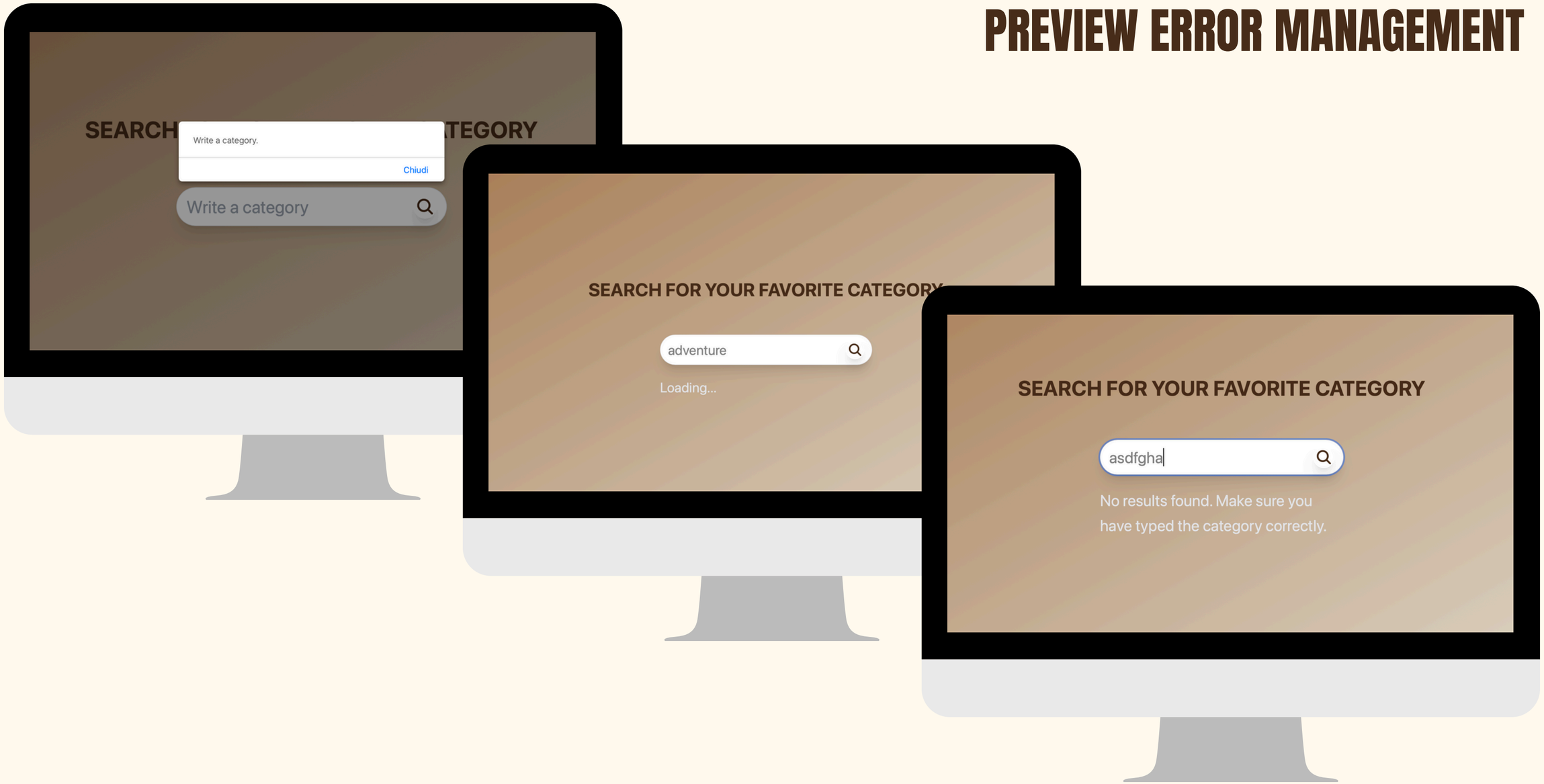
RAGIONAMENTO

Struttura del Progetto: ho organizzato il progetto in cartelle per HTML, JavaScript, CSS e immagini, facilitando così la manutenzione e un'eventuale collaborazione in team. Questa struttura rende il codice più leggibile e accessibile sia per me che per eventuali collaboratori. A tal proposito ho lavorato utilizzando GitHub e la creazione di branch con pull request e marge del progetto.

Tecnologie Utilizzate: per la logica dell'app, ho scelto JavaScript e Webpack per gestire le dipendenze e ottimizzare il caricamento delle risorse. Ho utilizzato Axios per semplificare le richieste API e Lodash per manipolare facilmente i dati.

La scelta dello stile: ho adottato Tailwind CSS per applicare stili in modo rapido e coerente, riducendo il tempo di sviluppo e migliorando l'estetica dell'interfaccia utente.

PREVIEW ERROR MANAGEMENT



RAGIONAMENTO

Gestione degli Errori: ammetto che gestire gli errori e le problematiche sorte durante lo sviluppo non è stato semplice, tuttavia ho voluto implementare un sistema di gestione degli errori al fine di guidare l'utente in caso di input non valido, migliorando così l'usabilità dell'app. Per fare ciò ho utilizzato la funzione get() della libreria Lodash.

Integrazione delle API: ho collegato l'app alle API di Open Library per recuperare informazioni sui libri, permettendo agli utenti di cercare per categoria e visualizzare titolo e autori e, successivamente al click, anche le descrizioni.

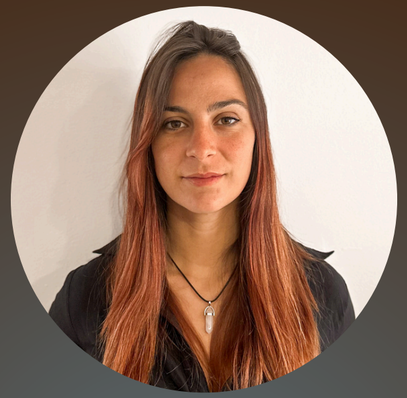
Pertanto ho cercato di mantenere il codice semplice e intuitivo, facilitando la lettura e la manutenzione. Credo che questo approccio migliori la qualità complessiva del progetto e la sua scalabilità nel tempo.

CODICE JS

Nell’utilizzo di Javascript ho sfruttato la funzione **fetchBooksByCategory** per comunicare con l’API e recuperare i libri di una specifica categoria usando Axios. Mi sono avvalsa della funzione **displayBooks** per creare elementi HTML per ciascun libro, mostrando titolo e autore.

In seguito ho inserito l’**evento click** nella funzione **fetchBookDescription**, in questo modo quando l'utente clicca su un libro, questa funzione recupera la descrizione del libro dall'API tramite un **alert**.

L’**Event Listener** del Bottone di Ricerca mi è servito per avviare la ricerca. Basandosi sulla categoria inserita, mostra infatti un messaggio di caricamento e **gestisce eventuali errori** o assenza di risultati.



GRAZIE PER L'ATTENZIONE

MIRIAM TRUIOLO

Front End Developer
Email: miriamtruiolo@outlook.it