Web Technology Mini Project

Book Directory

Team Members:

22ALR075 – Ramprasath K T
22ALR079 – Rishikeshan K
22ALR094 – Sivaraj S

Introduction:

The "Book Directory", is a comprehensive and user-friendly web application designed for book enthusiasts of all ages and interests. In today's digital age, finding and exploring books has never been easier, yet the sheer volume of available titles can make the process overwhelming. "Book Directory" aims to simplify this task by providing a streamlined and efficient platform where users can effortlessly search for books based on various parameters such as title, author, and more.

"Book Directory" leverages the powerful Google Books API to deliver real-time data and up-to-date information on millions of books from around the world. Whether you are looking for a classic novel, the latest bestseller, a scientific tome, or a children's book, our application ensures you have all the information you need at your fingertips. The dynamic search capabilities allow for quick and precise queries, making it easy to find exactly what you are looking for in a matter of seconds.

The project is designed with the user in mind, featuring an intuitive interface that is both aesthetically pleasing and highly functional. The clean layout and straightforward navigation ensure that users of all technical skill levels can enjoy a seamless browsing experience. Additionally, the responsive design guarantees that "Book Directory" is accessible from any device, whether you are using a desktop, tablet, or smartphone.

At the heart of "Book Directory" is the commitment to enhancing the reading experience by connecting users with the vast universe of books available online. By providing detailed information about each book, including the author, publisher, and direct links to Google Books for further reading or purchasing, our application serves as a valuable resource for both casual readers and avid bibliophiles alike.

Key Features:

- Dynamic Search Functionality:
  Users can search for books using keywords related to the title, author, or other relevant parameters. The search results are fetched dynamically from the Google Books API, ensuring up-to-date information.

- Comprehensive Book Information:
  Each search result includes detailed information about the book, such as the author, publisher, and a link to read more on Google Books. This ensures that users have access to all necessary information to make informed choices.

- User-Friendly Interface:

The application features a clean and intuitive interface, making it easy for users to navigate and find the books they are looking for. The design is responsive and accessible, ensuring a seamless experience across different devices.

- Book Cover Display:

The application displays the cover images of the books, providing a visually appealing and engaging user experience.

- Read More Links:

Each book entry includes a "Read more" link that redirects users to the Google Books page for additional information and purchase options, enhancing the usability and functionality of the application.

- Responsive Design:

The application is designed to be responsive, ensuring a seamless user experience across various devices and screen sizes.

REQUIREMENTS:

Front-End Programming:

• CSS (Cascading Style Sheets): CSS is used to define the visual appearance and layout of web pages. It allows developers to apply styles, such as colors, fonts, spacing, and positioning, to HTML elements, creating a visually appealing and consistent user interface.

• JavaScript: JavaScript is a powerful programming language that adds interactivity and dynamic behavior to web pages. It enables developers to manipulate and modify HTML and CSS, handle user interactions, perform calculations, validate forms, and make asynchronous requests to servers.

• React: React is a popular JavaScript library for building user interfaces. It provides a component-based approach to web development, allowing developers to create reusable UI components that update efficiently based on changes in data. React uses a virtual DOM
(Document Object Model) to optimize performance and facilitate the building of complex web applications.

Back-End Programming:

• APIs (Application Programming Interfaces): Backend developers build APIs that allow communication between the frontend and backend components of a web application. APIs define the protocols and rules for how different software components can interact and
exchange data. Commonly used API standards include RESTful APIs and GraphQL.

Coding:

```
//index.js
import React from "react";
import ReactDOM from "react-dom";
import App from "./app";

ReactDOM.render(
  <React.StrictMode>
    <App></App>
  </React.StrictMode>,
  document.getElementById("root")
);

//app.js
import React from "react";
import Books from "./books";
import Header from "./header";
import ScrollToTop from "./scrolltotop";
import "./sass/style.css"

const App = () => {
  return (
    <div>
      <Header></Header>
      <Books></Books>
      <ScrollToTop></ScrollToTop>
    </div>
  );
};

export default App;

//book.js
import React from "react";
import { BiLinkExternal } from "react-icons/bi";
import defaultBook from "./img/defaultBook.png";
import "../src/sass/style.css";
import { motion } from "framer-motion";

const Book = ({ id, volumeInfo }) => {
  const imageVariants = {
    hover: {
      scale: 1.7,
      boxShadow: "0px 0px 8px #000",
      transition: {
        duration: 0.5,
        type: "spring",
        delay: 0.15,
      },
    },
  };
  let { title, authors, publisher, previewLink, imageLinks } = volumeInfo;

  //setting up default values for volume info data
  title = title || "Title is not available";
  authors = authors || "Author(s) name not available";
  publisher = publisher || "Publisher company not available";
  previewLink = previewLink || "https://books.google.co.in/";

  return (
    <section key={id} className="loading-card">
      <div>
        <div>
          <motion.img
            src={imageLinks ? imageLinks.thumbnail : defaultBook}
            width="100px"
            alt="Book-cover"
            variants={imageVariants}
            whileHover="hover"
          />
        </div>
        <div>
          {title && (
            <div>
              <h3 className="inline">{title}</h3>
            </div>
          )}
        </div>

        <div>
          {authors && (
            <h4 style={{ paddingBottom: "1rem", color: "black" }}>
              {" "}
              Author:{" "}
              <span
                style={{
                  fontWeight: "bold",
                  color: "#3B3B3B",
                }}
              >
                {" "}
                {authors}{" "}
              </span>
            </h4>
          )}
        </div>

        <div>
          {publisher && (
            <h5 style={{ paddingBottom: "1rem", color: "black" }}>
              {" "}
              Published by:{" "}
              <span
                style={{
                  fontWeight: "bold",
                  color: "#3B3B3B",
                }}
              >
                {" "}
                {publisher}{" "}
              </span>
            </h5>
          )}
        </div>

        <div>
          {previewLink && (
            <h5
              style={{
                fontWeight: "bold",
                color: "black",
                paddingBottom: "1rem",
              }}
            >
              Read more :{" "}
              <a href={previewLink} target="_blank" rel="noreferrer">
                {" "}
                Google Books <BiLinkExternal></BiLinkExternal>{" "}
              </a>
            </h5>
          )}
        </div>

        {/* <div
              style={{
                display: "flex",
                justifyContent: "center",
                alignItems: "center",
                paddingTop: "1rem",
              }}
            > {language && }
              <p>
                {" "}
                <span style={{ fontWeight: "bold", color: "black" }}>
                  {" "}
                  Language :{" "}
                </span>{" "}
                {language}{" "}
              </p>
              <p>
                {" "}
                <span
```

```
              style={{
                fontWeight: "bold",
                color: "black",
                marginLeft: "1rem",
              }}
            >
              {" "}
              Average Rating :{" "}
            </span>{" "}
            {averageRating}
          </p>
        </div> */}
    </div>
  </section>
 );
};

export default Book;

//books.js
import React, { useState, useEffect } from "react";
import axios from "axios";
import logo from "./img/Book.svg";
import Searchform from "./searchform";
import Book from "./book";
import LoadingCard from "./loadingCard";

const BookDetails = () => {
  const [details, setDetails] = useState([]);

  const [term, setTerm] = useState("Ruskin Bond");

  const [isLoading, setIsLoading] = useState(true);

  useEffect(() => {
    const fetchDetails = async () => {
      setIsLoading(true);
      const resources = await axios.get(
        `https://www.googleapis.com/books/v1/volumes?q=${term}&maxResults=11`
      );
      setDetails(resources.data.items);
      setIsLoading(false);
    };
    fetchDetails();
  }, [term]);

  const loadMore = async () => {
    const resources = await axios.get(
      `https://www.googleapis.com/books/v1/volumes?q=${term}&maxResults=8&startIndex=$
{details.length}`
    );
    setDetails((oldDetails) => [...oldDetails, ...resources.data.items]);
```

```
};

return (
  <>
    <h2
      style={{
        textTransform: "capitalize",
        color: "#DB4437",
        fontSize: 40,
        marginTop: -40,
      }}
    >
      {term}
    </h2>
    <Searchform searchText={(text) => setTerm(text)}></Searchform>
    {isLoading ? (
      <section className="container" style={{ padding: "2rem 0rem" }}>
        <LoadingCard />
        <LoadingCard />
        <LoadingCard />
        <LoadingCard />
      </section>
    ) : !details ? (
      <h1
        className="loading-name"
        style={{
          background: "white",
          borderRadius: "1rem",
          color: "#DB4437",
          padding: "1rem",
          position: "absolute",
          top: "50%",
          left: "50%",
          fontSize: 33,
          transform: "translate(-50%,-50%)",
          textTransform: "capitalize",
        }}
      >
        😞 Couldn't find books about {term}
      </h1>
    ) : (
      <section>
        <section className="container" style={{ padding: "2rem 0rem" }}>
          {details.map((book, index) => (
            <Book {...book} key={index} />
          ))}
          <div className="custom-card">
            <h3 style={{ fontSize: "1.32rem", color: "white" }}>
              Didn't find the book you love?
            </h3>
            <br />
```

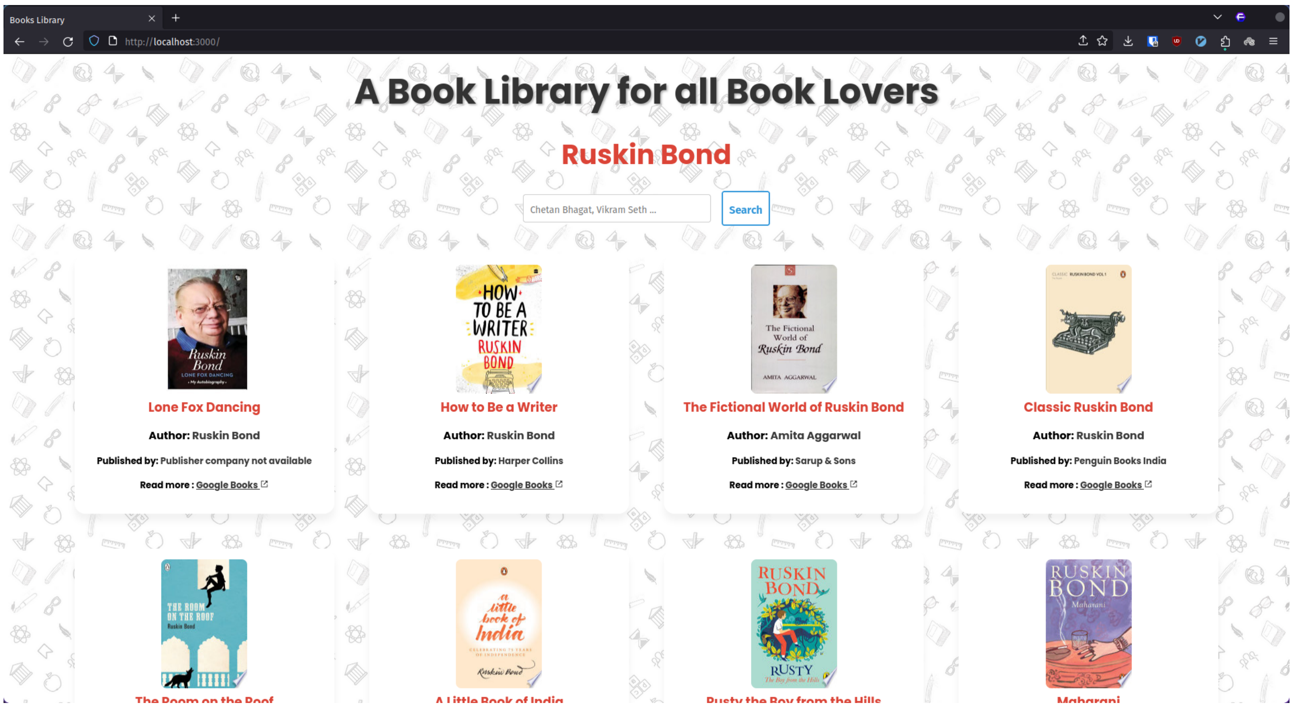
```
              <img
                style={{ width: "100%" }}
                src={logo}
                alt="A man reading a book"
                srcSet=""
              />

              <h3 style={{ fontSize: "1.21rem", color: "white" }}>
                Search for your favourite{" "}
                <span style={{ fontWeight: "bold", color: "black" }}>
                  Genre{" "}
                </span>
                or{" "}
                <span style={{ fontWeight: "bold", color: "black" }}>
                  Author{" "}
                </span>
                in the search box!!
              </h3>
            </div>
          </section>
          <div className="load-more">
            <button onClick={() => loadMore()}>Load More!</button>
          </div>

        </section>
      )}
    </>
  );
};

export default BookDetails;
```

<u>//searchform.js</u>

```
import React, { useState } from "react";

const Searchform = ({ searchText }) => {
  const [text, setText] = useState("");
  const [showValidTextModal, setShowValidTextModal] = useState(false);

  const handleSubmit = (e) => {
    e.preventDefault();
    if (text === "" || !text.trim()) {
      setShowValidTextModal(true);
      return;
    }
    searchText(text);
  };

  const onChangevalue = (e) => {
    e.preventDefault();
    setText(e.target.value);
    searchText(e.target.value);
    if (e.target.value === "") {
      setText("Ruskin Bond");
      searchText("Ruskin Bond");
    }
  };

  return (
    <div>
      <br />
      <form
        onSubmit={handleSubmit}
        style={{
          display: "flex",
          justifyContent: "center",
          alignItems: "center",
        }}
      >
        <input
          type="text"
          placeholder="Chetan Bhagat, Vikram Seth ..."
          onChange={onChangevalue}
        />
        <button
          style={{
            marginLeft: "1rem",
            transition: "all 0.3s ease 0s",
            padding: "0.6rem",
            borderRadius: "0.2rem",
            cursor: "pointer",
          }}
          type="submit"
        >
          Search
        </button>
      </form>
      <div
        id="popup1"
        className={showValidTextModal ? "overlay modal-active" : "overlay"}
      >
        <div className="popup">
          <div className="close" onClick={() => setShowValidTextModal(false)}>
            &times;
          </div>
          <h3 className="content">Please Enter the valid text</h3>
        </div>
      </div>
    </div>
  );
};

export default Searchform;
```

Screenshots:

Home Page:


Books Library

http://localhost:3000/

A Book Library for all Book Lovers

Ruskin Bond

Chetan Bhagat, Vikram Seth ...

Search

Lone Fox Dancing

Author: Ruskin Bond

Published by: Publisher company not available

Read more : Google Books

How to Be a Writer

Author: Ruskin Bond

Published by: Harper Collins

Read more : Google Books

The Fictional World of Ruskin Bond

Author: Amita Aggarwal

Published by: Sarup & Sons

Read more : Google Books

Classic Ruskin Bond

Author: Ruskin Bond

Published by: Penguin Books India

Read more : Google Books

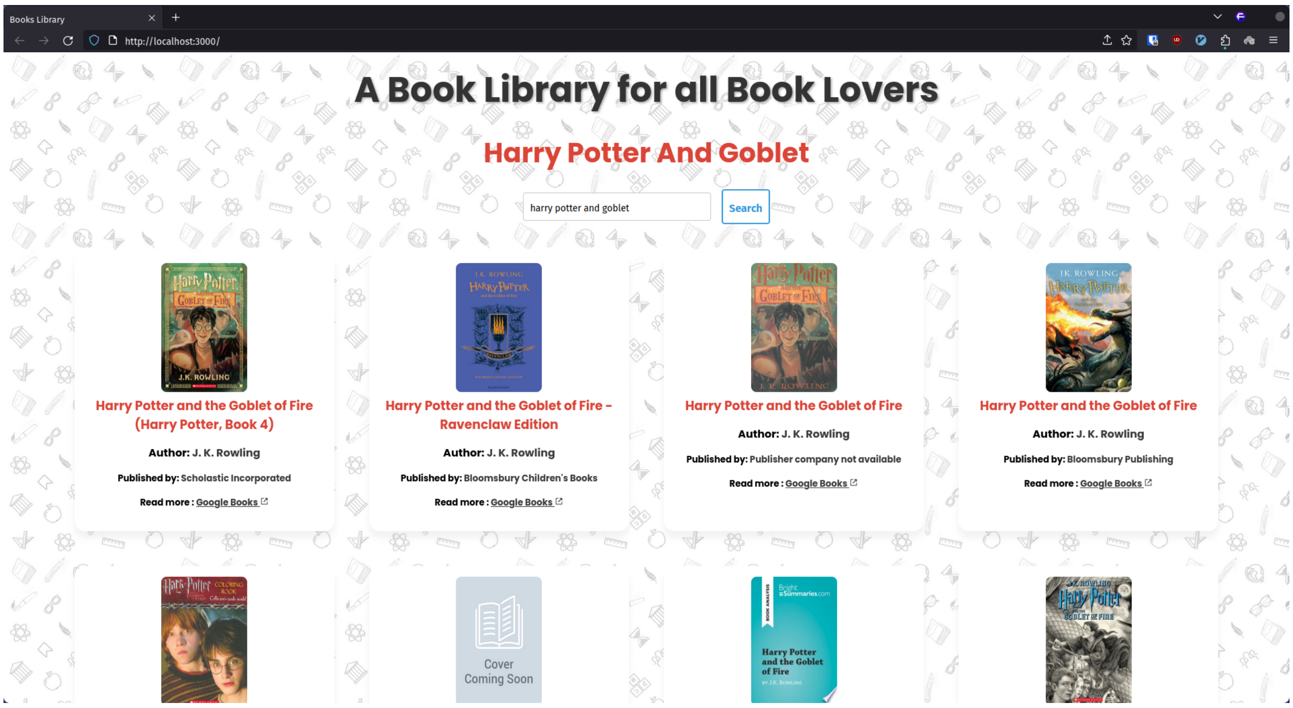
The Room on the Roof

A Little Book of India

Rusty the Boy from the Hills

Maharani

Searching for ‘Harry Potter and Goblet’ (it auto searches for the full book name):


Books Library

http://localhost:3000/

A Book Library for all Book Lovers

Harry Potter And Goblet

harry potter and goblet

Search

Harry Potter and the Goblet of Fire (Harry Potter, Book 4)

Author: J. K. Rowling

Published by: Scholastic Incorporated

Read more : Google Books

Harry Potter and the Goblet of Fire - Ravenclaw Edition

Author: J. K. Rowling

Published by: Bloomsbury Children's Books

Read more : Google Books

Harry Potter and the Goblet of Fire

Author: J. K. Rowling

Published by: Publisher company not available

Read more : Google Books

Harry Potter and the Goblet of Fire

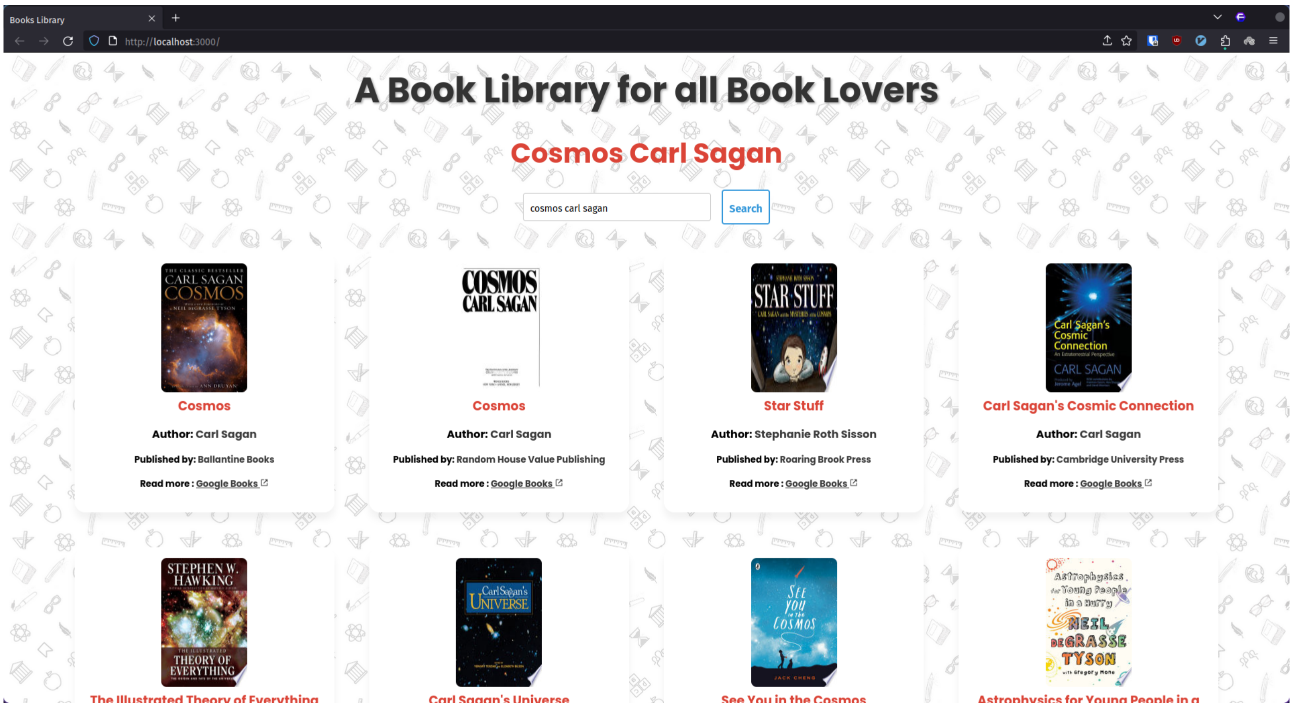
Author: J. K. Rowling

Published by: Bloomsbury Publishing

Read more : Google Books

Cover Coming Soon

Searching for a book with author name: ‘Cosmos Carl Sagan’



Books Library

http://localhost:3000/

A Book Library for all Book Lovers

Cosmos Carl Sagan

cosmos carl sagan

Search

Cosmos

Author: Carl Sagan

Published by: Ballantine Books

Read more : Google Books

Cosmos

Author: Carl Sagan

Published by: Random House Value Publishing

Read more : Google Books

Star Stuff

Author: Stephanie Roth Sisson

Published by: Roaring Brook Press

Read more : Google Books

Carl Sagan's Cosmic Connection

Author: Carl Sagan

Published by: Cambridge University Press

Read more : Google Books

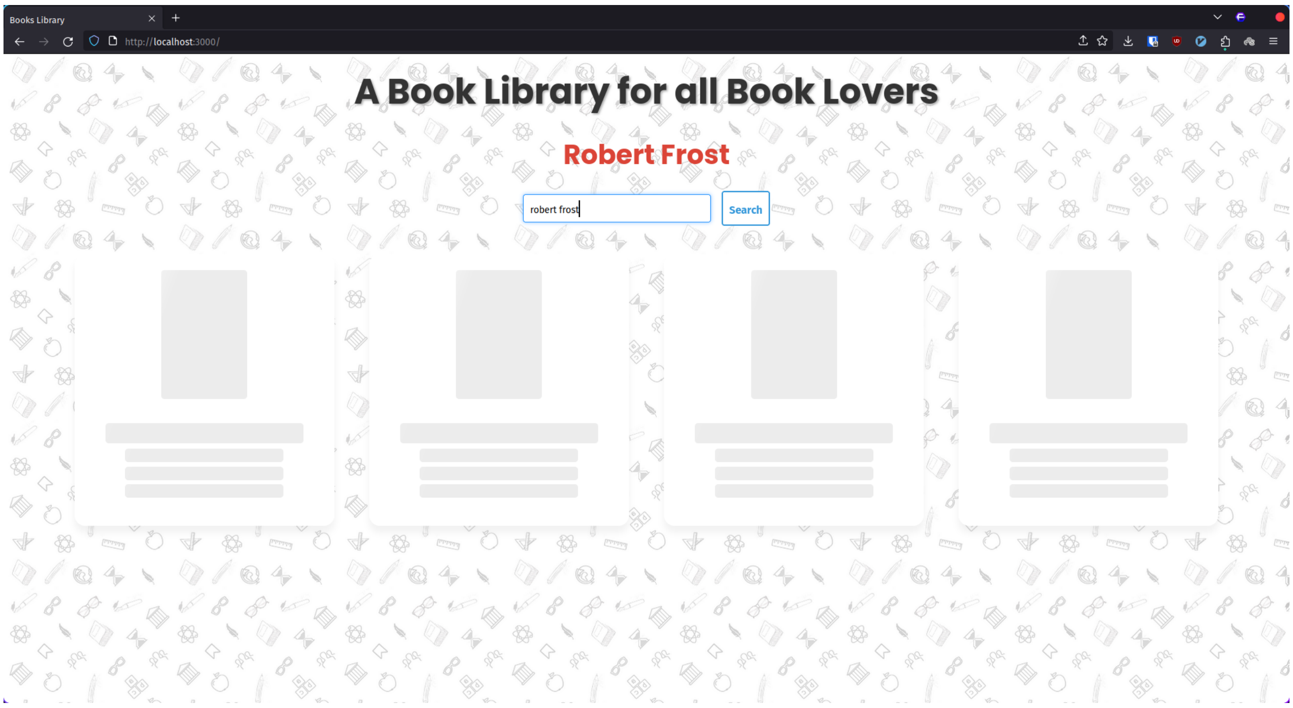
The Illustrated Theory of Everything

Carl Sagan's Universe

See You in the Cosmos

Astrophysics for Young People in a

Dynamic Loading Page:



Books Library

http://localhost:3000/

A Book Library for all Book Lovers

Robert Frost

robert frost

Search