BookVerse – Day 2 Documentation Report

This report documents the implementation of Day 2 for the React-based project 'BookVerse', focusing on Component Composition, Styling, and Class Components. It introduces AuthorInfo, demonstrates Refs (Uncontrolled Components), PropTypes validation, and lifecycle methods with Bootstrap styling.

1. Project Overview

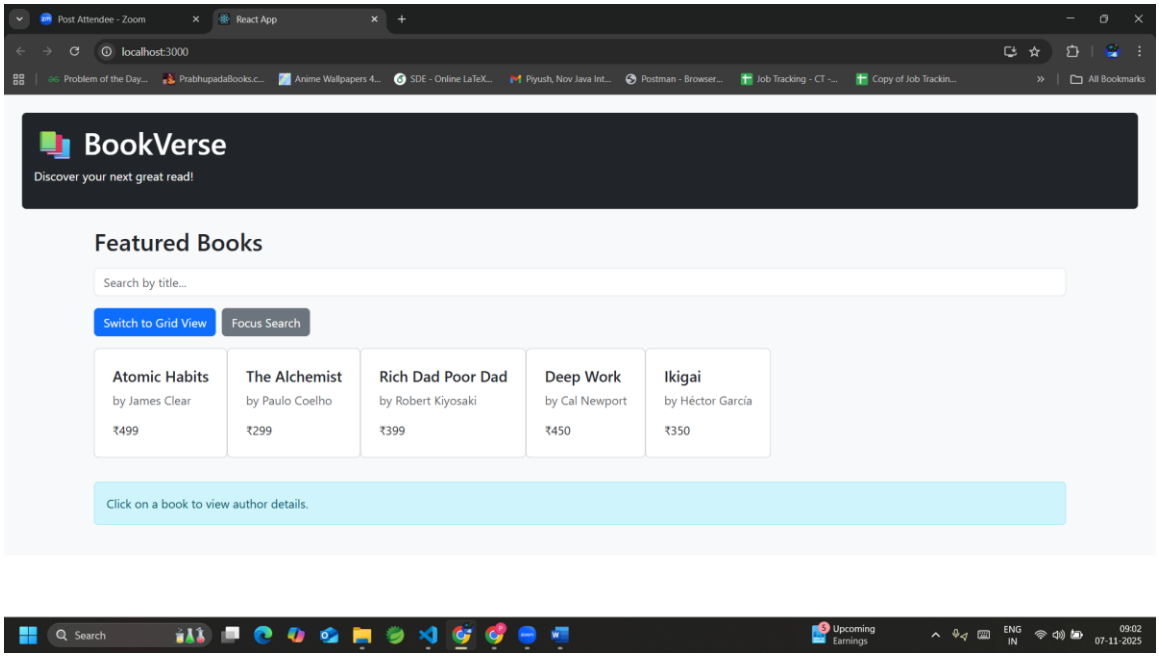
Title: BookVerse – Author Details and Styling the Components

Objective: Enable users to click on a book to view author details while applying consistent styling using Bootstrap and CSS.

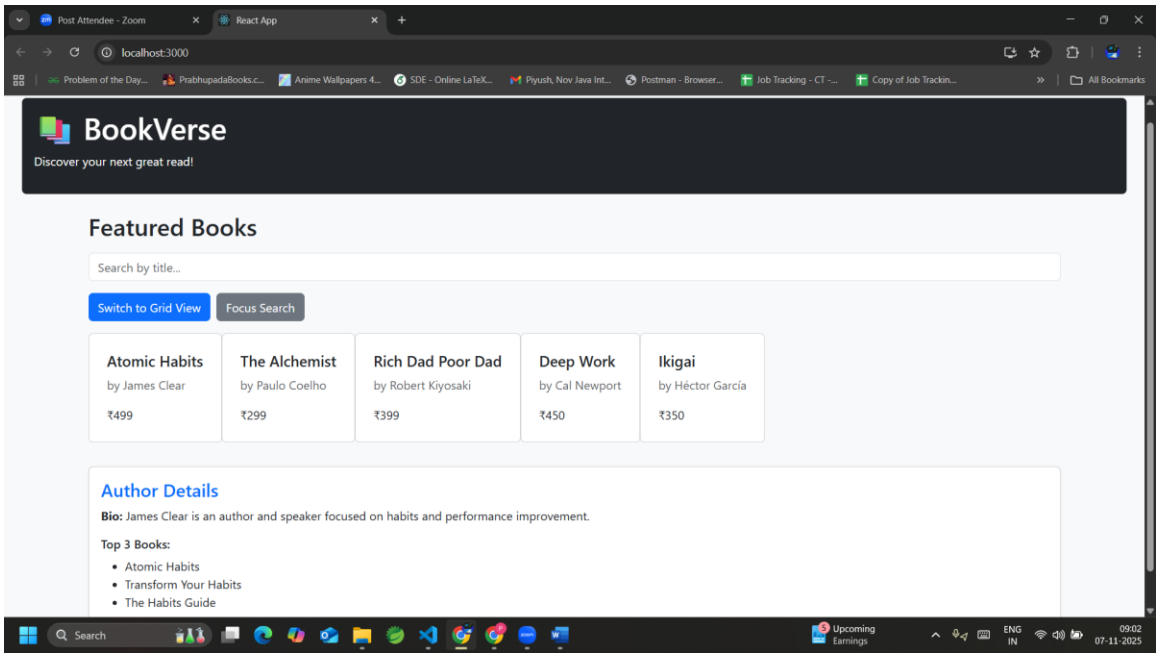
Key Features:
- Display author bio and top books using a class component.
- Validate props using PropTypes.
- Focus search input via Refs.
- Demonstrate lifecycle methods.
- Apply Bootstrap-based styling for layout consistency.

2. Output Screenshots (Placeholders)

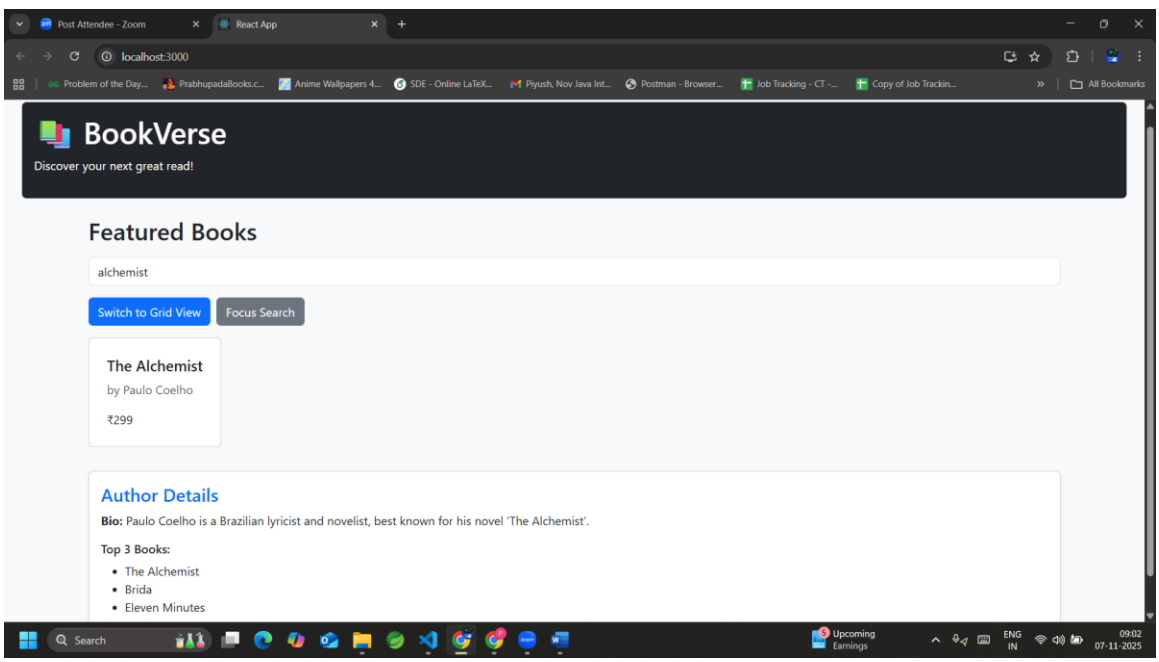
[Screenshot 1: Home Page – Grid View Placeholder]



[Screenshot 2: Author Info Section Placeholder]



[Screenshot 3: Focus Search Button Functionality Placeholder]



3. Components and Code Snippets

App.jsx

Entry component that initializes BookList and sets up the styled header section.

```
// Entry point integrating styled components with Bootstrap

import React from "react";
```

```
import BookList from "./components/BookList";
import "bootstrap/dist/css/bootstrap.min.css";

const App = () => {
  return (
    <div className="App bg-light p-4">
      <header className="bg-dark text-white p-3 rounded">
        <h1>📚 BookVerse</h1>
        <p>Discover your next great read!</p>
      </header>

      <BookList />
    </div>
  );
};

export default App;
```

Explanation: The App component integrates Bootstrap and renders BookList as the main section.

BookList.jsx

Functional component combining BookCard, AuthorInfo, and SearchBox with Refs and layout toggling.

```
// Combines BookCard, AuthorInfo, and demonstrates Refs (Uncontrolled
Components)

import React, { useState, useRef } from "react";
import BookCard from "./BookCard";
import SearchBox from "./SearchBox";
import AuthorInfo from "./AuthorInfo";
import booksData from "../data/booksData";
import "bootstrap/dist/css/bootstrap.min.css";

const BookList = () => {
    const [layout, setLayout] = useState("grid");
    const [searchTerm, setSearchTerm] = useState("");
    const [selectedAuthor, setSelectedAuthor] = useState(null);

    // Ref for focusing on the search input
    const searchRef = useRef(null);

    const toggleLayout = () => {
```

```
        setLayout(layout === "grid" ? "list" : "grid");
    };

    const filteredBooks = booksData.filter((book) =>
        book.title.toLowerCase().includes(searchTerm.toLowerCase())
    );

    const focusSearch = () => {
        if (searchRef.current) {
            searchRef.current.focus();
        }
    };

    return (
        <div className="container">
            <h2 className="mt-4">Featured Books</h2>

            <SearchBox
                searchTerm={searchTerm}
                setSearchTerm={setSearchTerm}
                inputRef={searchRef}
            />

            <div className="d-flex gap-2 mb-3">
                <button className="btn btn-primary"
onClick={toggleLayout}>
                    Switch to {layout === "grid" ? "List" : "Grid"} View
                </button>
                <button className="btn btn-secondary"
onClick={focusSearch}>
                    Focus Search
                </button>
            </div>

            <div
                className={`book-list d-${layout === "grid" ? "grid" :
"flex"} flex-wrap`}
            >
                {filteredBooks.map((book) => (
                    <BookCard
                        key={book.id}
                        title={book.title}
                        author={book.author}
                        price={book.price}
                        layout={layout}
```

```
                        onSelectAuthor={setSelectedAuthor}
                    />
                ))}
            </div>

            {/* Author Info Section */}
            <AuthorInfo author={selectedAuthor} />
        </div>
    );
};

export default BookList;
```

Explanation: Manages book list rendering, search, and layout toggle; integrates AuthorInfo and Refs.

BookCard.jsx

Displays individual book information and handles click events to show author details.

```
// Functional component with PropTypes and click event for author details

import React from "react";
import PropTypes from "prop-types";

const BookCard = ({ title, author, price, layout, onSelectAuthor }) => {
    return (
        <div
            className={`book-card card mb-3 ${layout === "grid" ? "p-3" :
"p-2"}`}
            onClick={() => onSelectAuthor(author)}
            style={{ cursor: "pointer" }}
        >
            <div className="card-body">
                <h5 className="card-title">{title}</h5>
                <p className="card-text text-muted">by {author}</p>
                <p className="card-text">₹{price}</p>
            </div>
        </div>
    );
};

// PropTypes validation
BookCard.propTypes = {
    title: PropTypes.string.isRequired,
```

```
    author: PropTypes.string.isRequired,
    price: PropTypes.number.isRequired,
    layout: PropTypes.string.isRequired,
    onSelectAuthor: PropTypes.func.isRequired,
};

export default BookCard;
```

Explanation: Implements PropTypes validation and triggers author selection via onClick event.

AuthorInfo.jsx

Class component showing author bio and top books with lifecycle methods.

```
// AuthorInfo.jsx
// Class component to display author details and lifecycle usage

import React, { Component } from "react";
import authorsData from "../data/authorsData";
import "bootstrap/dist/css/bootstrap.min.css";

class AuthorInfo extends Component {
    constructor(props) {
        super(props);
        this.state = { author: null };
    }

    componentDidMount() {
        // Log when author details load
        console.log("AuthorInfo component mounted, loading author
data...");
        this.loadAuthorInfo(this.props.author);
    }

    componentDidUpdate(prevProps) {
        // Load author details when a different book is clicked
        if (prevProps.author !== this.props.author) {
            this.loadAuthorInfo(this.props.author);
        }
    }

    loadAuthorInfo = (authorName) => {
        const info = authorsData[authorName] || null;
        this.setState({ author: info });
```

```
    };

    render() {
        const { author } = this.state;

        if (!author) {
            return (
                <div className="alert alert-info mt-3">Click on a book to
view author details.</div>
            );
        }

        return (
            <div className="card mt-3 shadow-sm">
                <div className="card-body">
                    <h4 className="card-title text-primary">Author
Details</h4>
                    <p><strong>Bio:</strong> {author.bio}</p>
                    <h6>Top 3 Books:</h6>
                    <ul>
                        {author.topBooks.map((book, idx) => (
                            <li key={idx}>{book}</li>
                        ))}
                    </ul>
                </div>
            </div>
        );
    }
}

export default AuthorInfo;
```

Explanation: Uses lifecycle methods to dynamically load and update author details when a book is clicked.

4. Summary

Day 2 implementation enhances BookVerse with interactive author details, lifecycle methods, and reusable components. It demonstrates class components, PropTypes, Refs, and Bootstrap styling for improved UI/UX.