National University of Computer and Emerging Sciences, Lahore Campus

Course:	**Advanced Programming**	**Course Code:**	**CS 2000**
Program:	**BS (CS)**	**Semester:**	**Fall 2024**
Due Date	**October 25,2024**	**Total Marks:**	**100**
Section:	**7A , 7B**	**Page(s):**	**2**
Type:	**Assignment 2**		

Problem Statement

Develop a comprehensive book management application called "Book Store" using Next.js. The aim of this assignment is to implement various routing strategies and pre-rendering techniques using data stored in a static JSON file. The application should allow users to browse, search, and view detailed information about books, categories, and authors while leveraging Next.js features for optimized performance.

Requirements:

Data Handling

- Use a JSON file books.json to store information about books, genres, and authors.
  - Each book should include properties like id, title, author, description, price, genre, and rating.
  - Each genre should have an id and name.
  - Each author should include id, name, and biography.

Home Page (index.js)

- Create a homepage that displays a list of featured books with a "View Genres" button.
- Use programmatic navigation (e.g., router.push()) to navigate to the genres page when the button is clicked.

Dynamic Routes

- Implement dynamic routes for individual book details (/books/[id]), showing the book's details based on the id parameter from the JSON file.
- Use getStaticPaths() to generate paths for these dynamic routes and getStaticProps() to pre-render book pages.

Nested Routes

- Include nested routes within the book details page for author information (e.g., /books/[id]/author).

Catch-All Routes

- Create a catch-all route for information pages (e.g., /info/[...slug]) to handle multiple levels of routing for sections like FAQs or support. Examples:
- /info
- /info/faqs
- /info/support

Custom 404 Page

- Implement a custom 404.js page to display a user-friendly error message when a page is not found.

Static Generation (SSG)

- Use getStaticProps() to generate the book list page statically.
- Include the revalidate key in getStaticProps() to implement Incremental Static Regeneration (ISR) for periodic updates of static pages.
- Use the notFound key to handle cases where book data is unavailable.

Static Generation with Dynamic Routes

- Use getStaticPaths() and getStaticProps() to statically generate pages for individual books.
- Implement the fallback key to handle dynamic book pages that are not pre-generated.

Server-Side Rendering (SSR)

- Create a page for book genres using getServerSideProps() to fetch data at request time.

Client-Side Rendering (CSR)

- Use the useSWR() hook to fetch data dynamically on the client side, particularly for the list of authors.

Books Page

- Display a list of all books, with filtering options by genre.
- Each book card should have a "View Details" button that links to the dynamic book page.

Book Details Page

- Show full book details, including the title, description, author, price, and rating.
- Provide a nested route to display detailed information about the book's author.

Genres Page

- Show a list of book genres, each linking to a filtered list of books in that genre.

Authors Page

- Display a list of authors using CSR with the useSWR() hook for dynamic data fetching.

Additional Features

- Implement a dark mode toggle for the application, allowing users to switch between light and dark themes.
- Store user search history using local storage and display recent searches on the search page.