**Frontend Development with React.js**

**Project Documentation format**

1. **Introduction**
   * **Project Title**:News app

* **TEAM MEMBERS AND THEIR ROLES**:

|  |  |  |
| --- | --- | --- |
| TEAM MEMBERS | E-MAIL ID | ROLE |
| BHAVANI V | [bhavavenkat2901@gmail.com](mailto:bhavavenkat2901@gmail.com) | Team Leader |
| ABITHA N | [abitha160205@gmail.com](https://d.docs.live.net/5FA994F69B16D876/Documents/abitha160205@gmail.com) | UI/UX designer |
| GIRIJA K | [girijakannadass@gmail.com](mailto:girijakannadass@gmail.com) | Frontend developer |
| ABINAYA M | [abimunusamy15@gmail.com](https://d.docs.live.net/5FA994F69B16D876/Documents/abimunusamy15@gmail.com) | Frontend developer |

1. **project Overview**
   * **Purpose**: The news app is a frontend application that provides users with the latest news from various sources. It fetches news articles through an API and displays them in an organized manner. Users can browse articles by categories such as Business, General, Politics, Health etc.
   * **Features**: Fetch news from an external API. Categorized news sections. Search functionalities. Responsive design for mobile and desktop.
2. **Architecture**
   * **Component Structure**:
   1. **Header**: Contains the app logo, navigation menu, and search bar.
   2. **NewsList**: Displays a list of news articles fetched from the API.
   3. **NewsCard**: A reusable component that displays an individual news article.
   4. **CategorySelector:** Allows users to filter news by category.
   5. **Footer**: Contains links and copyright information.
   * **State Management**:
   * React’s built-in useState and useEffect hooks are used for state management.Context API is used to manage global state (e.g., theme selection).
   * **Routing**: React Router is used for navigation between different pages (e.g., Home, Categories, About).
3. **Setup Instructions**
   * **Prerequisites**: Node.js installed and Visual studio code installed.
   * **Installation**:
   * **Clone the repository:**
   * cd news-app
   * **Install dependencies:**
   * npm install
   * **Create a .env file and add your API key:**
   * ini
   * REACT\_APP\_NEWS\_API\_KEY=your\_api\_key\_here
   * **Start the development server:**
   * npm start
4. **Folder Structure**

/news-app

│── /src

│ ├── /components

│ │ ├── Header.js

│ │ ├── NewsList.js

│ │ ├── NewsCard.js

│ │ ├── CategorySelector.js

│ │ ├── Footer.js

│ ├── /pages

│ │ ├── Home.js

│ │ ├── Category.js

│ │ ├── About.js

│ ├── /utils

│ │ ├── api.js (handles API calls)

│ ├── App.js

│ ├── index.js

│── package.json

│── .env

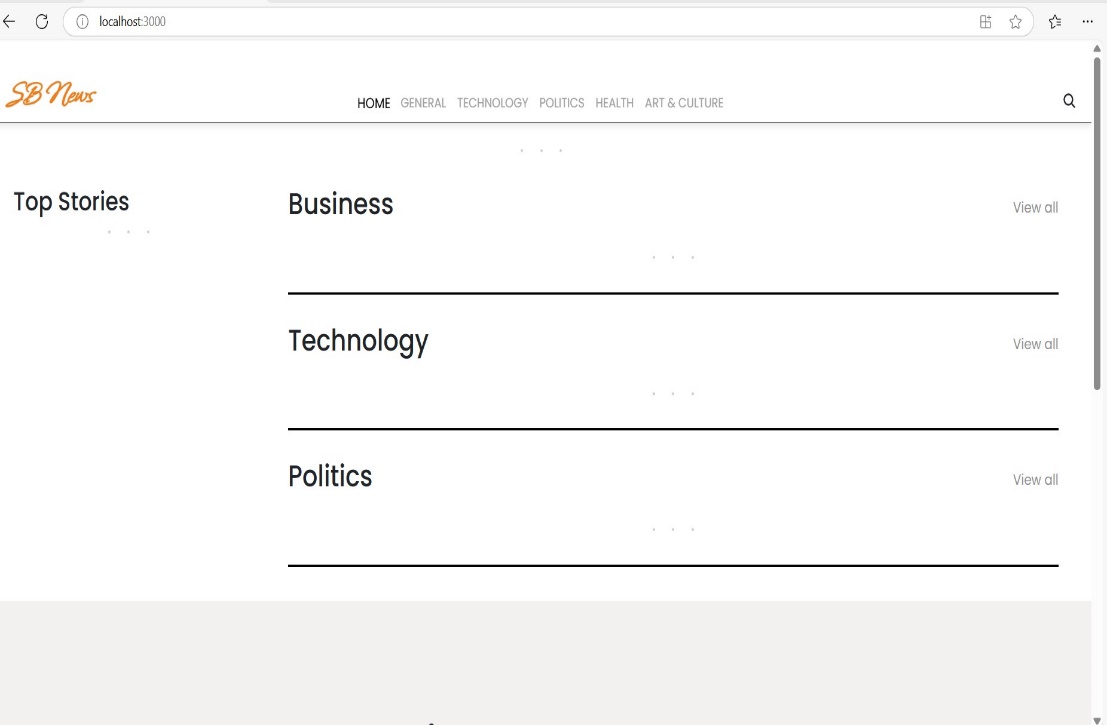
│── README.md

1. **Running the Application**
   * Provide commands to start the frontend server locally.
     + **Frontend**: npm start.
2. **Component Documentation**
   * **Key Component:**
   1. **News card:** Displays the title, image and summary of an article.
   2. **Category Selector:** Allow users to filter news by category.

**Reusable Components**:

* 1. **Button Component:** Used for all buttons in the app.
  2. **Loading Spinner**: Displays when news data is being fetched.

1. **State Management**
   * **Global State**: The News App uses React Context API to manage the global state, such as the selected news category and theme (light/dark mode).
   * API data (news articles) is fetched once and stored in the global state to minimize API calls and improve performance.
   * **Local State**: The useState hook is used to manage local states like search queries, loading states, and toggling UI elements.
   * Example: The search bar maintains its own state before filtering news results dynamically.
2. **User Interface**
   * The UI is designed with a clean and modern approach, ensuring readability and easy navigation.
   * **Features:**
   * **Homepage:** Displays top headlines with categories.
   * **Category Pages:** News articles filtered by categories.
   * **Search Feature**: Allows users to search for specific news topics.
   * **Dark Mode Toggle**: Users can switch between light and dark themes.



1. **Styling**:

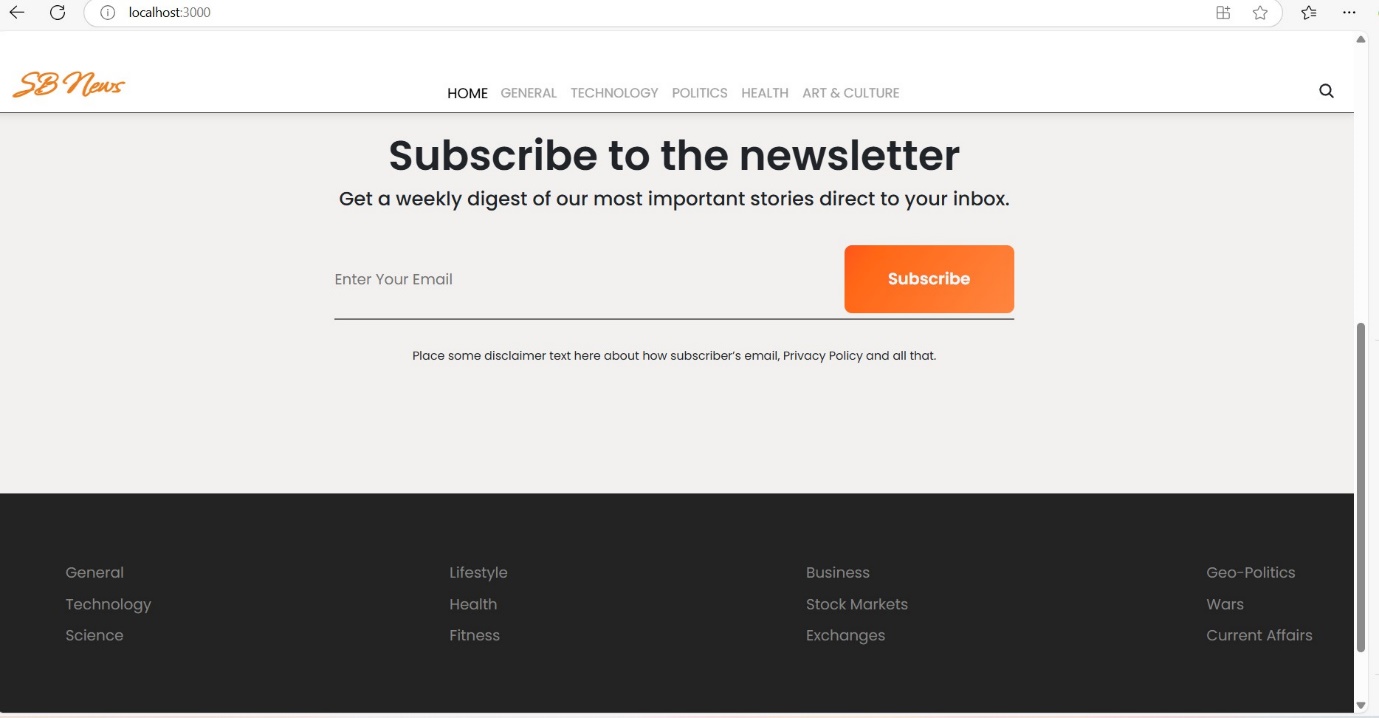
**CSS Frameworks/Libraries:** The app uses Tailwind CSS for styling, ensuring responsiveness and a modern look.Styled-components were used for theme-based styling.

**Theming:**The app supports light and dark themes using Context API.Users can toggle themes, and the preference is stored in localStorage.

1. **Testing**

* **Testing Strategy**: Unit tests for individual components using Jest and React Testing Library.Integration tests ensure proper API data fetching and state updates.End-to-end tests using Cypress to verify UI behavior.
* **Code Coverage**: Jest’s built-in coverage tools are used to measure test coverage.
* **Focus areas:**
* **API calls:** Ensuring data is fetched and displayed correctly.
* **UI Components:** Checking rendering and state updates.

1. **Screenshots or Demo**



**Demo video link:** <https://drive.google.com/file/d/10x0NIBKhJLF0D3xgeJ0r6fuNRayYad9o/view?usp=sharing>

1. **Known Issues**:

* Sometimes, API requests exceed the rate limit, leading to failed fetch requests.
* Articles with missing images appear with a default placeholder image.
* The search functionality does not currently support fuzzy matching.

1. **Future Enhancements:**

* Add offline support using Service Workers for better performance.
* Implement user authentication to allow users to save favorite articles.
* Introduce a "Trending Now" section based on article popularity.
* Enhance search functionality with auto-suggestions.