**Frontend Development with React.js**

**Project Documentation format**

1. **Introduction**
   * Project Title: FD-react News App
   * Team Members:  
     Team Leader- Vivekanandhan  
     Team Member 2 – Sachin G
   * Team Member 3- Sachin S
   * Team member 4- Sakthyan K
2. **Project Overview**
   * Purpose: A React.js-based frontend for a news platform with category-based navigation and newsletter features.
   * Features:  
     Responsive news homepage with hero section  
     Navigation bar with category routing  
     Category-wise news listing  
     Individual news detail pages  
     Newsletter subscription form  
     Reusable components for articles, stories, and layout
3. **Architecture**
   * Component Structure:  
     - App.js: Main entry point, sets up routing and global context.  
     - NavbarComponent: Provides navigation between pages.  
     - Hero, TopStories, HomeArticles, NewsLetter: Core homepage sections.  
     - Pages (Home, CategoryPage, NewsPage): Page-level components.  
     - Footer: Reusable site footer component.
   * State Management: Uses React Context API via GeneralContext.jsx for global state handling.
   * Routing: Uses react-router-dom for client-side routing between Home, Category, and News pages.
4. **Setup Instructions**
   * Prerequisites:  
     Node.js (>=14.x)  
     npm or yarn
   * Installation:  
     Clone the repository.  
     Navigate to the `code` folder.  
     Run `npm install` to install dependencies.  
     Configure environment variables if required.  
     Run `npm start` to start the development server.
5. **Folder Structure**
   * Client:  
     - public/: Static assets and index.html  
     - src/: Application source code  
      - components/: Reusable UI components  
      - pages/: Page-level components  
      - styles/: CSS files for styling components  
      - context/: Global context management  
      - App.js, index.js: Entry points
   * **Utilities**: Explain any helper functions, utility classes, or custom hooks used in the project.
6. **Running the Application**
   * Provide commands to start the frontend server locally.
     + **Frontend**: npm start in the client directory.
7. **Component Documentation**
   * **Key Components**: Document major components, their purpose, and any props they receive.
   * **Reusable Components**: Detail any reusable components and their configurations.
8. **State Management**
   * **Global State**: Describe global state management and how state flows across the application.
   * **Local State**: Explain the handling of local states within components.
9. **User Interface**
   * Provide screenshots or GIFs showcasing different UI features, such as pages, forms, or interactions.
10. **Styling**

* CSS Frameworks/Libraries: CSS modules located in `src/styles`. No external CSS framework detected.
* **Theming**: Explain if theming or custom design systems are implemented.

1. **Testing**

* Testing Strategy: Uses Jest and React Testing Library (setupTests.js) for unit and integration testing.
* Code Coverage: Default coverage reporting via Jest.

1. **Screenshots or Demo**

* Provide screenshots or a link to a demo showcasing the application’s features and design.

1. Known Issues:  
   No critical issues reported. Placeholder for bug tracking.

* Document any known bugs or issues that users or developers should be aware of.

1. Future Enhancements:  
   Add dark mode support.  
   Improve performance with lazy loading.  
   Enhance category filtering and search functionality.  
   Integration with live news APIs.

* Outline potential future features or improvements, such as new components, animations, or enhanced styling.