Frontend Development with React.js
Project Documentation Format

1. Introduction

- Project Title: Insight Stream: Navigate the News Landscape
- Team Members: Mastani I – Project Lead & API Integration; Selvi G – Developer & Error Handling; Gopika R – Documentation Supporting & Voice Over; Deepika G – Responsive Design; Nasrin Fathima M – Documentation

2. Project Overview

- Purpose: The project aims to provide users with an efficient way to navigate the news landscape by organizing and filtering articles based on categories and content-based features.
- Features: Filter news articles by categories. Organize content using content-based features. Responsive design for seamless use across devices. Simple, user-friendly interface.

3. Architecture

- Component Structure: The frontend was developed using HTML. The initial code was taken from SmartInt, integrated into Visual Studio, and enhanced with API support.
- State Management: Basic error handling and API response management using Rapid API.
- Routing: Not applicable (single-page application).

4. Setup Instructions

- Prerequisites: Node.js, Visual Studio Code, Rapid API account
- Installation: 1. Clone the repository. 2. Open the project in Visual Studio Code. 3. Configure the Rapid API key for news data fetching. 4. Run the project using the development server.

5. Folder Structure

- Client: index.html (main entry point), assets/ (images, CSS files), scripts/ (JavaScript for API integration and UI updates).
- Utilities: Helper functions for error handling and API calls.

6. Running the Application

- Frontend: Run 'npm start' inside the client directory after setup.

7. Component Documentation

- Key Components: API Integration (fetches news articles from Rapid API), News Filtering (categorizes and displays articles dynamically).
- Reusable Components: API call utility functions, Error handling functions.

8. State Management

- Global State: Managed via API calls and fetched data.
- Local State: Maintains selected categories and displayed articles.

9. User Interface

- UI Features: Responsive design ensures proper display on mobile and desktop. Categorized news display. Minimalist and clean design.

10. Styling

- CSS Frameworks/Libraries: Custom CSS styling was applied.
- Theming: Lightweight, responsive design with readability focus.

11. Testing

- Testing Strategy: Manual testing of API integration, error handling, and responsive design.
- Code Coverage: Verified API responses, content filtering, and responsiveness across devices.

12. Screenshots or Demo

- Provide screenshots or a demo link showcasing the news filtering and responsive UI.

13. Known Issues

- API response delays may occur depending on internet connection.
- Some categories may show limited news due to API data availability.

14. Future Enhancements

- Add personalized feeds with user authentication.
- Include location-based and trending news filters.
- Enhance UI with animations and advanced theming.
- Integrate voice-based news search and reading.