# InsightStream - React Application Documentation

## 1. Introduction

**Project Title:** InsightStream – A Dynamic Web Platform

### Team Leader:

Parvesh U

### Team Members:

* Chandru M
* Divakar S
* MadhanRaj E
* Shyam M

## 2. Project Overview

### Purpose

InsightStream is a React-based web application built using Vite. It provides an interactive UI for users to browse, interact, and manage real-time streaming content seamlessly.

### Features

- Smooth content streaming experience.  
- Dynamic UI built using React and Tailwind CSS.  
- User-friendly navigation and interactive design.  
- JSON-based content database (powered by json-server).  
- Dark and light mode support.

## 3. Architecture

### Component Structure

- App Component: The root component managing global state and layout.  
- Navbar Component: Handles navigation between sections.  
- Player Component: Manages streaming content playback.  
- ContentList Component: Displays the list of available content.  
- Favorites Component: Allows users to save favorite content.

### State Management

Uses React's useState and useEffect for local state management.

### Routing

- / - Home page  
- /content - Content listing  
- /favorites - Saved items  
- /settings - User settings

## 4. Setup Instructions

### Prerequisites

- Node.js (v16+)  
- npm or yarn

### Installation

Clone the repository:  
```  
git clone https://github.com/your-repo/insightstream.git  
```

Navigate to the project directory:  
```  
cd insightstream  
```

Install dependencies:  
```  
npm install  
```

## 

## 5. Folder Structure

insightstream/  
│── public/  
│── src/  
│ ├── components/  
│ ├── pages/  
│ ├── assets/  
│ ├── store/  
│ ├── utils/  
│ ├── App.jsx  
│ ├── main.jsx  
│── .env  
│── package.json  
│── vite.config.js

## 6. Running the Application

To start the development server:  
```  
npm run dev  
```

## 7. Component Documentation

### Key Components

- ContentList: Displays a list of available content.  
- Player: Controls content playback.  
- SearchBar: Allows users to search for content.

### Reusable Components

- Button: Customizable button component.  
- Card: Styled container for displaying data.

## 8. State Management

### Global State

- Uses React Context API for theme switching and user preferences.  
- JSON-based database (json-server) for storing content and user data.

### Local State

- useState for UI-based interactions (e.g., play/pause, search, filters).

## 9. User Interface

- Responsive design supporting mobile and desktop views.  
- Built with Tailwind CSS for styling.

## 10. Styling

### CSS Frameworks/Libraries

- Tailwind CSS for component styling.  
- Styled Components for dynamic theming.

### Theming

Supports light and dark modes using context-based theming.

## 11. Testing

### Testing Strategy

- Unit Testing: Jest & React Testing Library for component tests.  
- Integration Testing: Ensuring API data rendering correctly.  
- End-to-End Testing: Cypress for user interaction testing.

### Code Coverage

Jest is configured for test coverage reports.

## 12. Screenshots or Demo

[Insert screenshots or a link to the live demo]

## 13. Known Issues

- API rate limits might affect real-time updates.  
- Mobile responsiveness improvements in progress.

## 14. Future Enhancements

- Integration with third-party streaming APIs.  
- AI-based content recommendations.  
- Offline mode for accessing saved content.