Project Documentation

# 1. Introduction

• Project Title: Insight stream

• Version: 0.1.0

• Team ID:

• Team Leader: Nithishram.R

• Team Members:

– Chandru

– Mohan

– Praveen kumar

# 2. Project Overview

• Description: This is a React front-end application created with Create React App.

• Purpose: A modern React-based frontend starter application.

• Features:

– React 19 for building UI components

– Routing with react-router-dom

– HTTP client with axios

– UI with Bootstrap and react-bootstrap

– Animations with framer-motion

– Icons with react-icons

– Basic performance metrics with web-vitals

– Testing utilities with React Testing Library and jest-dom

# 3. Architecture

• Frontend: React.js (Create React App) with react-bootstrap and framer-motion

• Backend: Not included (this repository is frontend-only). You can integrate Node.js + Express.js for APIs.

• Database: Not included. Recommended: MongoDB or PostgreSQL depending on requirements.

# 4. Setup Instructions

• Prerequisites:

– Node.js (recommended >= 14.x)

– npm (comes with Node)

– Git

– Visual Studio Code (recommended)

• Installation Steps:

1. Clone the repository

git clone <repo\_url>

2. Navigate into the project folder

cd myapp

3. Install dependencies

npm install

4. Start the development server

npm start

# 5. Folder Structure

myapp/

|-- node\_modules/

|-- public/

|-- src/

| |-- components/

| |-- pages/

| |-- App.js

|-- package.json

|-- README.md

# 6. Running the Application

• Development:

npm start

• Build for production:

npm run build

• Test:

npm test

• Access: Visit http://localhost:3000 after starting the dev server.

# 7. API Documentation

This project does not include a backend API by default. If you integrate an API, configure axios base URL in src/api or src/utils (example):

axios.defaults.baseURL = 'http://localhost:5000/api'

Example endpoints (if you create a backend):

– POST /api/user/register

– POST /api/user/login

– GET /api/projects

– POST /api/projects/create

# 8. Authentication

• Authentication is not implemented in this repo. Recommended approach:

– Implement JWT-based authentication on the server and store access tokens in HttpOnly cookies or secure storage.

– Protect routes on the frontend with higher-order components or route guards.

# 9. User Interface

• Typical pages to include:

– Landing Page

– Dashboard (user/freelancer)

– Project Details Page

– Admin Panel (if required)

• Navigation: React Router for route handling.

# 10. Testing

• Test framework: React Testing Library + Jest (included in dependencies).

• Run tests: npm test

• Tools for manual testing: Postman (for APIs), Chrome Dev Tools

# 11. Screenshots or Demo

Add screenshots here (e.g., screenshots/ directory) or include a link to a hosted demo.

# 12. Known Issues

• Repo currently contains frontend only; no backend or database integration.

• Add any runtime or build issues you encounter here.

# 13. Future Enhancements

• Add backend (Node.js + Express) with proper API endpoints.

• Integrate MongoDB or PostgreSQL for persistence.

• Add authentication and authorization (JWT + refresh tokens).

• Add CI/CD pipeline and unit/integration test coverage.

# Appendix A — Dependencies

• @testing-library/dom: ^10.4.1

• @testing-library/jest-dom: ^6.7.0

• @testing-library/react: ^16.3.0

• @testing-library/user-event: ^13.5.0

• axios: ^1.11.0

• bootstrap: ^5.3.7

• framer-motion: ^12.23.12

• react: ^19.1.1

• react-bootstrap: ^2.10.10

• react-dom: ^19.1.1

• react-icons: ^5.5.0

• react-router-dom: ^7.8.1

• react-scripts: 5.0.1

• web-vitals: ^2.1.4

# Appendix B — npm Scripts

start: react-scripts start

build: react-scripts build

test: react-scripts test

eject: react-scripts eject