Project Documentation

---

1. Introduction

- **Project Title:** SB Works — Freelancing Platform
- **Team Members:**
  - RACHAMADUGU SIVASANKAR
  - Rottela Vishnuvardhan
  - Gurram Sreekanth
  - R SANTHOSH KUMAR REDDY
  - Chilakala Vinay Kumar

---

2. Project Overview

- **Purpose:**
  SB Works is designed to connect clients with skilled freelancers, offering a secure and transparent environment for project postings, bidding, communication, and transaction management — all powered by the MERN stack.
- **Features:**
  - Client project posting and management
  - Freelancer project bidding and submission
  - Real-time secure messaging system
  - Admin-controlled platform monitoring
  - JWT-based authentication and authorization
  - Responsive user interface with Material UI and Bootstrap
  - MongoDB-based dynamic NoSQL database

---

3. Architecture

- **Frontend:**
  Developed using React.js, structured around reusable components for dashboards, projects, authentication, and messaging. Axios is used for API calls. Styling handled via Material UI and Bootstrap.
- **Backend:**
  Built with Node.js and Express.js. Includes route controllers for users, projects, applications, and chat functionalities. Authentication is handled using JWT. Middleware ensures secure API access.

- **Database:**
  MongoDB handles data storage for:
  - Users
  - Projects
  - Applications
  - Chat Messages
  - Freelancer profiles
    Structured via Mongoose schemas for consistency.

---

4. Setup Instructions

- **Prerequisites:**
  - Node.js & npm
  - MongoDB
  - Git
  - Visual Studio Code
- **Installation:**

1. Clone project: git clone [Drive Link]
2. Install dependencies:

- cd client
- npm install
- cd ../server
- npm install

3. Configure environment variables in .env (DB URI, JWT_SECRET)

4. Start the servers:

  - **Frontend:** npm start in client/
  - **Backend:** npm start in server/

---

5. Folder Structure

- **Client (React):**
  - /src/components/ — UI components
  - /src/pages/ — Page-level components
  - /src/context/ — Global state management
- **Server (Node.js/Express):**

- /models/ — Mongoose schemas
- /routes/ — API endpoints
- /controllers/ — Business logic handlers
- /middleware/ — JWT and request validation
- /config/ — Database connection

---

6. Running the Application

- **Frontend:** npm start inside client/
- **Backend:** npm start inside server/

---

7. API Documentation

Endpoint	Method	Description
/api/register	POST	Register a new user
/api/login	POST	User login
/api/projects	GET	Retrieve project listings
/api/projects	POST	Post new project (client only)
/api/apply	POST	Submit application for project
/api/chat	POST	Send chat message
/api/chat/:id	GET	Retrieve chat history

Authentication required on protected routes via JWT token header

---

8. Authentication

Implemented using **JWT tokens**:

- User logs in and receives a token.
- Token is stored in local storage on the client side.
- Token is attached in headers for secure API calls.
- Middleware verifies token on protected routes.

---

9. User Interface

Responsive and modern using **React.js**, **Material UI**, and **Bootstrap**.

Screens include:

- Landing Page
- Login/Register
- Client Dashboard
- Freelancer Dashboard
- Project List and Application Form
- Admin Dashboard
- Chat Interface

---

10. Testing

- **Manual testing:** Through API testing tools like **Postman**
- **Frontend testing:** Form validation, button functionalities, routing checks
- **Backend testing:** CRUD operations on MongoDB via API calls

---

11. Screenshots or Demo

- **App Demo Video:** Watch Here

---

12. Known Issues

- No integrated payment gateway yet
- No mobile app version
- Limited user analytics dashboard

---

13. Future Enhancements

- Integrating payment gateway (Stripe/Razorpay)
- Mobile app development using React Native
- AI-powered freelancer recommendations
- Video call integration for project meetings
- Enhanced analytics for admin