Freelancing Application MERN

INTRODUCTION

FreelanceHub is a freelancing platform that connects clients with skilled freelancers. It offers an intuitive interface for project posting, bidding, and streamlined collaboration. With a dedicated admin team ensuring security and smooth communication, FreelanceHub aims to be the go-to platform for both clients and freelancers.

Description

Welcome to FreelanceHub, a modern freelancing platform that transforms the way clients connect with skilled freelancers. Our intuitive interface provides clients with the opportunity to post diverse projects, ranging from creative endeavors to technical tasks, while freelancers can seamlessly bid on these projects based on their expertise and capabilities.

At FreelanceHub, we prioritize efficiency and transparency in the freelancing process. Clients can review freelancer profiles, assess past work, and select the perfect candidate for their project. Once a freelancer is chosen, the client can easily communicate and collaborate with them within the platform, streamlining the entire workflow.

Our dedicated admin team ensures the integrity and security of every transaction. With stringent oversight, we guarantee the reliability and quality of the freelancers on our platform. The admin's role is not only to maintain the platform's integrity but also to facilitate smooth communication between clients and freelancers, ensuring a positive and productive working relationship.

Freelancers on FreelanceHub benefit from a straightforward project submission process. After completing the assigned project, freelancers can submit their work directly through the platform, offering clients a hassle-free experience. Clients have the opportunity to review the work and provide feedback, fostering a collaborative environment that values excellence.

Stay informed about the latest projects and industry trends with real-time updates and notifications. FreelanceHub aims to be the go-to platform for clients seeking reliable freelancers and freelancers looking for exciting opportunities to showcase their skills.

Join FreelanceHub today and experience a new era of freelancing where your projects are efficiently managed, your skills are recognized, and collaborations flourish in a secure and dynamic environment.

Scenario based case-study

Sarah, a recent graduate with a degree in graphic design, is eager to showcase her skills and build a strong freelance portfolio. She stumbles upon FreelanceHub while searching for online freelancing opportunities.

Finding the Perfect Project: Impressed by the user-friendly interface, Sarah browses through various project categories. She discovers a project posted by a local bakery, "Sugar Rush," seeking a logo redesign. The project description details the bakery's brand identity and target audience, giving Sarah a clear understanding of the client's needs on FreelanceHub.

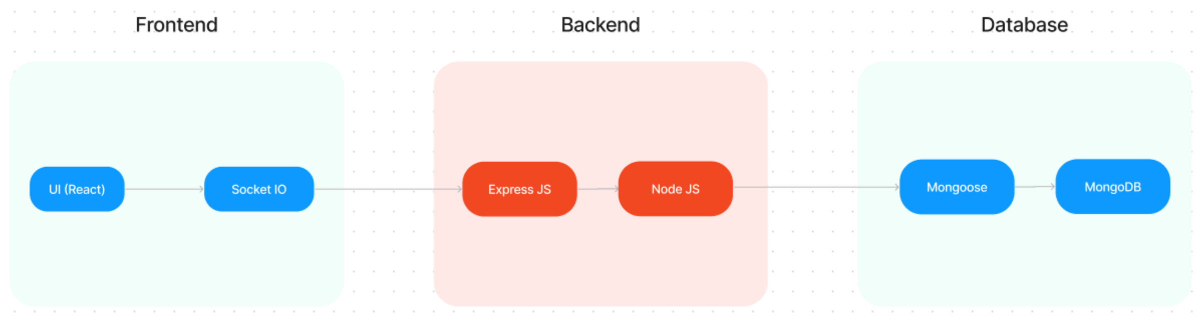
Bidding with Confidence: Confident in her creative abilities, Sarah explores the project requirements in detail. FreelanceHub enables her to analyze the bakery's existing branding materials, refining her design concept. She crafts a persuasive proposal that emphasizes her relevant experience and includes curated samples from her portfolio, all securely stored within the platform.

Communication & Collaboration: "Sugar Rush" chooses Sarah's proposal thanks to her strong portfolio and appealing pricing. FreelanceHub ensures smooth communication between Sarah and the bakery, giving them the ability to discuss project details and fine-tune the design direction through its built-in chat feature.

Delivery & Feedback: Once the design is finalized, Sarah submits her logo through the FreelanceHub platform. "Sugar Rush" can easily review the submission, share feedback, and request any minor revisions if necessary. FreelanceHub supports a collaborative environment where both client and freelancer work together to achieve the best possible result.

Building a Thriving Career: After successfully completing the project and receiving a glowing review from "Sugar Rush," Sarah's profile on FreelanceHub gains valuable traction. This positive experience motivates her to actively seek new opportunities on the platform. With an expanding portfolio and strong client testimonials, Sarah is well on her way to building a successful freelance career through FreelanceHub.

TECHNICAL ARCHITECTURE



The technical architecture of FreelanceHub follows a client–server model, with the frontend acting as the client and the backend functioning as the server. The frontend delivers the user interface and presentation layer, using the Axios library to simplify communication with the backend through RESTful APIs.

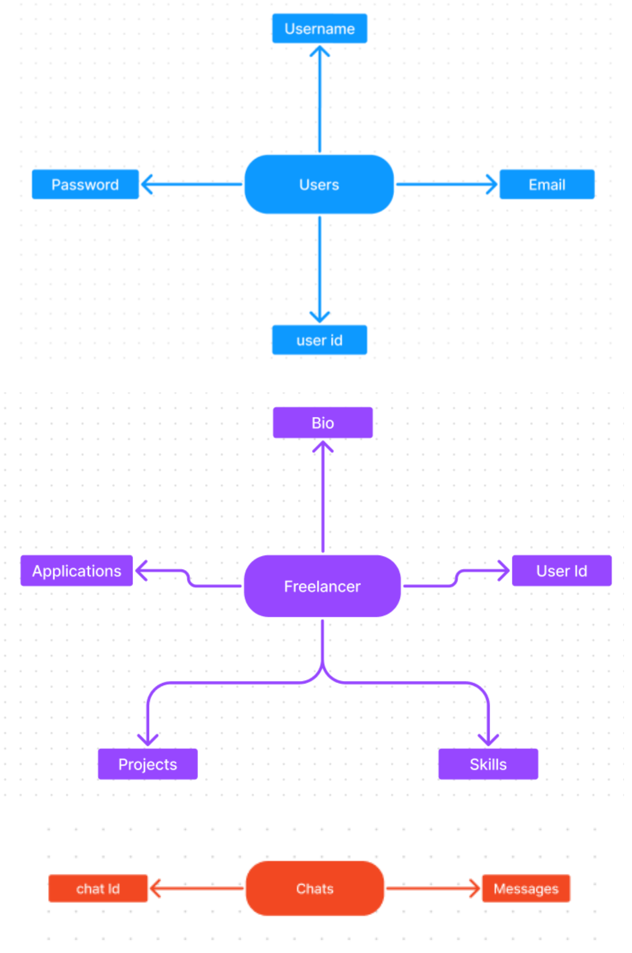
To ensure a polished and engaging user experience, the frontend is built with React and styled using Bootstrap, along with custom CSS for consistent theming across pages. This combination provides an interactive, responsive, and professional interface for users.

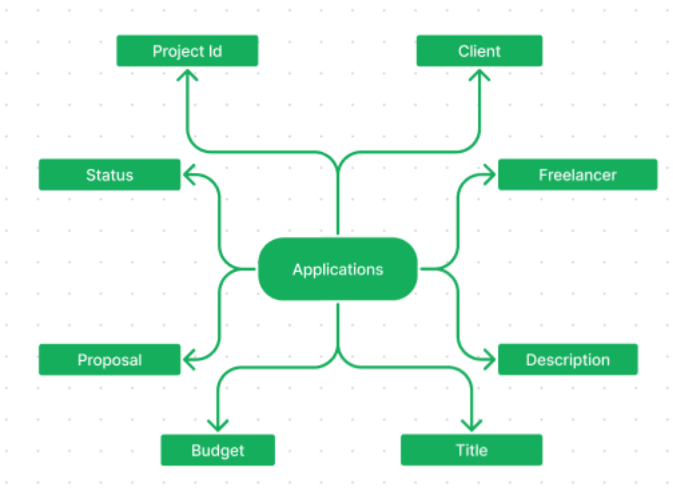
On the backend, FreelanceHub uses the Express.js framework to manage server-side logic, route handling, and communication. Express.js offers a solid and maintainable foundation for efficiently processing client requests and serving responses.

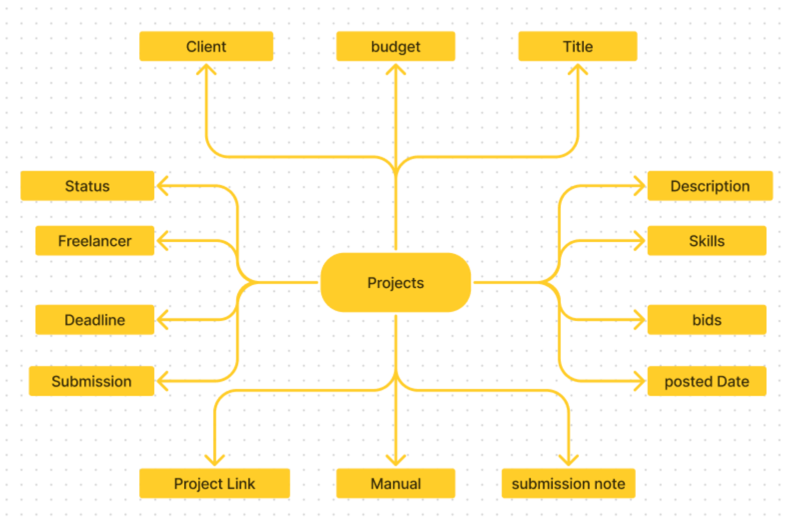
For persistent data storage, FreelanceHub relies on MongoDB. This NoSQL database ensures flexibility and scalability for storing user profiles, projects, bids, applications, and chat messages. The design supports secure, efficient, and real-time access to all critical data required to manage freelancing workflows.

Together, the integrated frontend and backend, powered by React, Express.js, and MongoDB, form a comprehensive technical architecture for FreelanceHub. This architecture enables smooth real-time interactions, reliable data exchange, and seamless collaboration between clients, freelancers, and administrators.

ER DIAGRAM

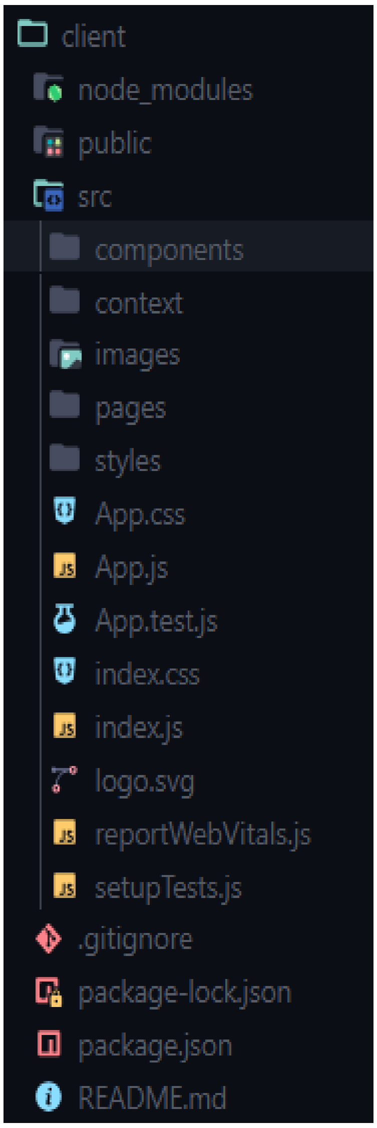




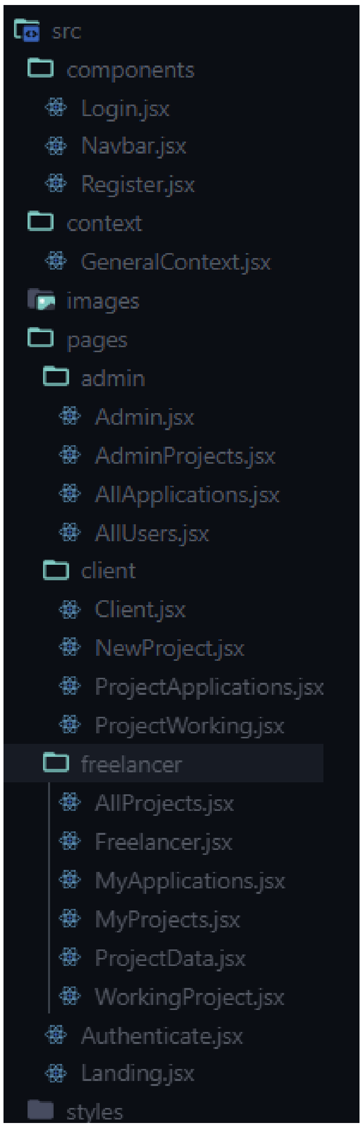


FreelanceHub connects clients with skilled freelancers through a user-friendly platform. Clients can post projects with details and browse freelancer profiles to find the ideal match. Freelancers can submit proposals, collaborate with clients via secure chat, and submit work for review and payment. An admin team oversees quality and communication, making FreelanceHub a trusted platform for both clients and freelancers.

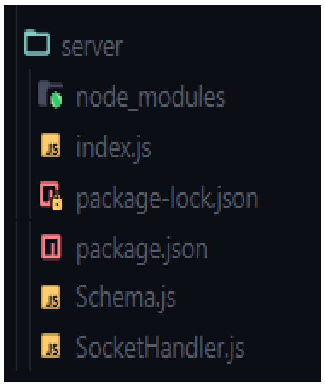
PROJECT STRUCTURE



```
client
  node_modules
  public
  src
    components
    context
    images
    pages
    styles
    App.css
    App.js
    App.test.js
    index.css
    index.js
    logo.svg
    reportWebVitals.js
    setupTests.js
  .gitignore
  package-lock.json
  package.json
  README.md
```



```
src
  components
    Login.jsx
    Navbar.jsx
    Register.jsx
  context
    GeneralContext.jsx
  images
  pages
    admin
      Admin.jsx
      AdminProjects.jsx
      AllApplications.jsx
      AllUsers.jsx
    client
      Client.jsx
      NewProject.jsx
      ProjectApplications.jsx
      ProjectWorking.jsx
    freelancer
      AllProjects.jsx
      Freelancer.jsx
      MyApplications.jsx
      MyProjects.jsx
      ProjectData.jsx
      WorkingProject.jsx
    Authenticate.jsx
    Landing.jsx
  styles
```



```
server
  node_modules
  index.js
  package-lock.json
  package.json
  Schema.js
  SocketHandler.js
```

FreelanceHub leverages React.js for its user interface. The client-side code consists of reusable components for dashboards, project listings, user authentication, and integrated chat features, assembled into pages for browsing projects or managing profiles. Shared data such as user roles and authentication states is managed with React Context. On the server side, Node.js with Express.js handles API requests for user management, project posting, bidding, and messaging. Mongoose models ensure structured and reliable interaction with the MongoDB database. This design offers a clear, maintainable architecture for FreelanceHub's smooth and scalable operation..

PRE-REQUISTIC:

Here are the key prerequisites for developing a full-stack application using Express Js, MongoDB, React.js:

✓ Node.js and npm:
Node.js is a robust JavaScript runtime that powers server-side code, enabling scalable and efficient network applications for *FreelanceHub*. It forms the backbone for building the backend with Express.js.
Make sure Node.js and npm (Node Package Manager) are installed on your development machine to manage dependencies and run server-side JavaScript seamlessly.
Download: https://nodejs.org/en/download/
Installation instructions: https://nodejs.org/en/download/package-manager/

✓ Express.js:
Express.js is a lightweight and flexible web framework for Node.js that powers the backend of FreelanceHub. It streamlines creating APIs and server logic with robust routing, middleware support, and a modular design.
Install Express.js to handle server-side routing, API endpoints, and middleware integration for seamless client-server communication.
Installation: Open your command prompt or terminal and run the following command:
npm install express

✓ MongoDB:
MongoDB is a powerful, scalable NoSQL database used by FreelanceHub to store data in a flexible JSON-like structure. It delivers high performance, horizontal scalability, and integrates smoothly with Node.js, making it ideal for managing diverse project and user data.
Set up a MongoDB database to manage your application's data effectively.
Download: https://www.mongodb.com/try/download/community
Installation instructions: https://docs.mongodb.com/manual/installation/

✓React.js:

React.js is a popular JavaScript library for building user interfaces. It enables developers to create interactive and reusable UI components, making it easier to build dynamic and responsive web applications.
Install React.js, a JavaScript library for building user interfaces.
Follow the installation guide: https://reactjs.org/docs/create-a-new-react-app.html

✓HTML, CSS, and JavaScript: Basic knowledge of HTML for creating the structure of your app, CSS for styling, and JavaScript for client-side interactivity is essential.

✓Database Connectivity: Use a MongoDB driver or an Object-Document Mapping (ODM) library like Mongoose to connect your Express Js server with the MongoDB database and perform CRUD (Create, Read, Update, Delete) operations

✓**Front-end Framework**: Utilize React Js to build the user-facing part of the application, including entering booking room, status of the booking, and user interfaces for the admin dashboard. For making better UI we have also used some libraries like material UI and bootstrap.

✓**Version Control**: Use Git for version control, enabling collaboration and tracking changes throughout the development process. Platforms like GitHub or Bitbucket can host your repository.
Git: Download and installation instructions can be found at: https://git-scm.com/downloads

✓**Development Environment**: Choose a code editor or Integrated Development Environment (IDE) that suits your preferences, such as Visual Studio Code, Sublime Text, or WebStorm.

• Visual Studio Code: Download from https://code.visualstudio.com/download

To run the existing Freelancer App project downloaded from Drive:
Use the code:
Drive link:
https://drive.google.com/drive/folders/1dnRpwEHi05fugNUDGTM7h7UaWqbBcmhg?usp=sharing

Install Dependencies:

• Navigate into the cloned repository directory:

```
cd freelancer-app-MERN
```

• Install the required dependencies by running the following commands:

```
cd client
npm install
../cd server
npm install
```

Start the Development Server:

• To start the development server, execute the following command:

```
npm start
```

• The SB Works app will be accessible at http://localhost:3000

You have successfully installed and set up the SB Works application on your local machine. You can now proceed with further customization, development, and testing as needed.

Application flow:

Freelancer Responsibilities:

• Project Submission: Freelancers must submit completed, high-quality work for assigned projects directly through the FreelanceHub platform.
• Adherence to Requirements: Ensure all submitted work fully meets client expectations, industry standards, and any specific instructions provided within the platform.
• Clear Communication: Maintain active communication with clients—responding quickly to messages, asking questions for clarification, and sharing regular updates on project progress.
• Time Management: Plan and prioritize tasks effectively to meet agreed-upon deadlines and deliver work on time.
• Professional Conduct: Maintain professionalism by demonstrating respect, courtesy, and collaboration when interacting with clients and other freelancers.
• Quality Assurance: Consistently deliver precise, polished, and error-free work to uphold high standards and ensure client satisfaction.

Client Responsibilities:

• Clear Project Description: Provide detailed and precise project briefs, clearly outlining deliverables, expectations, and any specific requirements to ensure freelancers understand the scope of work.
• Timely Communication: Respond quickly to freelancer questions, share necessary details, and offer feedback promptly to keep the project on track.
• Payment Obligations: Honor agreed payment terms by completing payments promptly and fairly once the project is satisfactorily delivered.
• Feedback and Evaluation: Offer thoughtful, constructive feedback and rate freelancer performance to help them improve and maintain a high-quality community on FreelanceHub.

Admin Responsibilities:

• Data Oversight: Admins are responsible for monitoring and safeguarding the integrity and security of all user data and platform transactions.
• Policy Enforcement: Ensure platform guidelines and ethical standards are upheld, maintaining a professional and trustworthy environment for all users.
• Conflict Resolution: Address disputes or issues between clients and freelancers promptly and fairly, ensuring a balanced and respectful community.
• User Support and Communication: Offer assistance and clear guidance to users, helping them navigate the platform effectively.
• Platform Maintenance and Improvement: Oversee the continuous maintenance, updates, and enhancements of FreelanceHub to ensure a smooth, secure, and modern user experience.

Project Flow:

Use the code in**:**
https://drive.google.com/drive/folders/1dnRpwEHi05fugNUDGTM7h7UaWqbBcmhg?usp=sharing

Demo video:
https://drive.google.com/file/d/1a78sICiBmVPkMvgdhC9jmOXolitCBjsn/view?usp=sharing

Milestone 1: Project setup and configuration.

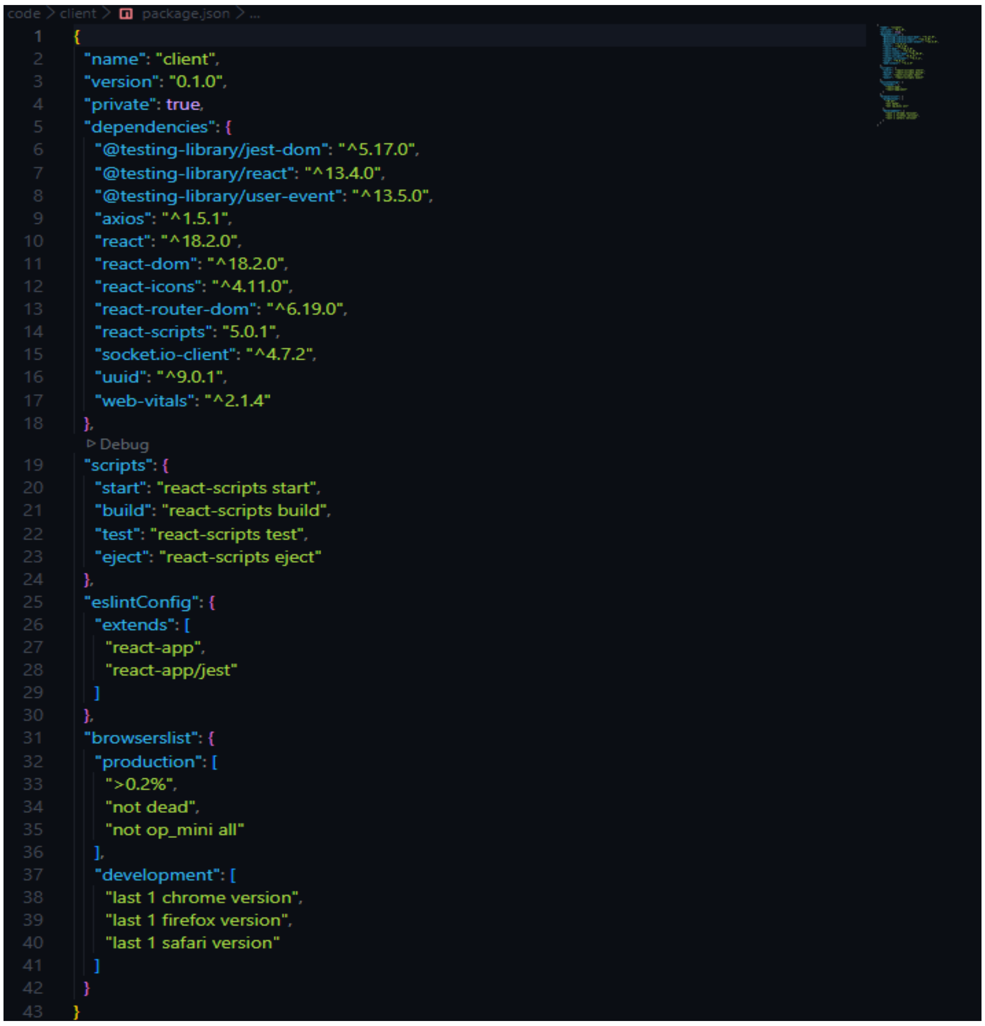
- ✓ **Folder setup:**
  First, create dedicated folders for the frontend and backend to organize and develop the respective parts of the FreelanceHub platform.
  - **Client folder:** Contains all React.js frontend code and dependencies.
  - **Server folder:** Holds Express.js backend code, API routes, and database models.
- ✓ **Installation of required tools:**
  1. Open the frontend folder to install necessary tools

     For frontend, we use:
     - React
     - Bootstrap
     - Material UI
     - Axios
     - react-bootstrap

  2. Open the backend folder to install necessary tools

     For backend, we use:
     - Express Js
     - Node JS
     - MongoDB
     - Mongoose
     - Cors
     - Bcrypt

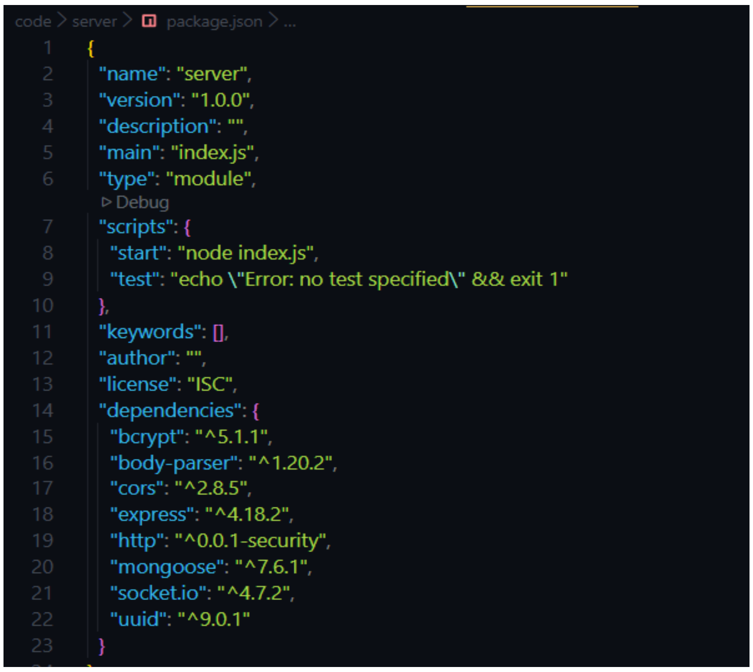
After the installation of all the libraries, the package.json files for the frontend looks like the one mentioned below.



code > client > package.json > ...

```
{
  "name": "client",
  "version": "0.1.0",
  "private": true,
  "dependencies": {
    "@testing-library/jest-dom": "^5.17.0",
    "@testing-library/react": "^13.4.0",
    "@testing-library/user-event": "^13.5.0",
    "axios": "^1.5.1",
    "react": "^18.2.0",
    "react-dom": "^18.2.0",
    "react-icons": "^4.11.0",
    "react-router-dom": "^6.19.0",
    "react-scripts": "5.0.1",
    "socket.io-client": "^4.7.2",
    "uuid": "^9.0.1",
    "web-vitals": "^2.1.4"
  },
  ▷ Debug
  "scripts": {
    "start": "react-scripts start",
    "build": "react-scripts build",
    "test": "react-scripts test",
    "eject": "react-scripts eject"
  },
  "eslintConfig": {
    "extends": [
      "react-app",
      "react-app/jest"
    ]
  },
  "browserslist": {
    "production": [
      ">0.2%",
      "not dead",
      "not op_mini all"
    ],
    "development": [
      "last 1 chrome version",
      "last 1 firefox version",
      "last 1 safari version"
    ]
  }
}
```

After the installation of all the libraries, the package.json files for the backend looks like the one mentioned below.



code > server > package.json > ...

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "type": "module",
  ▷ Debug
  "scripts": {
    "start": "node index.js",
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "bcrypt": "^5.1.1",
    "body-parser": "^1.20.2",
    "cors": "^2.8.5",
    "express": "^4.18.2",
    "http": "^0.0.1-security",
    "mongoose": "^7.6.1",
    "socket.io": "^4.7.2",
    "uuid": "^9.0.1"
  }
```

Milestone 2: Backend Development

1. **Project Setup:**

- Create the server-side project folder and initialize it using npm init to manage dependencies.
- Install essential backend packages like Express.js for routing, Mongoose for database modeling, and middleware such as body-parser and cors for handling requests.

2. **Database Configuration:**

- Set up a MongoDB database, either locally or via a cloud service like MongoDB Atlas.
- Design collections to store and manage key platform data:
- **Users** – user profiles, account types (client, freelancer, admin).
- **Projects** – project details, budgets, required skills, status.
- **Applications** – freelancer proposals, bid amounts, timelines.
- **Chats** – messaging history and communication between clients and freelancers.
- **Freelancer Profiles** – extended details such as skills, experience, and ratings.

3. **Express.js Server:**

- Create an Express.js server to handle HTTP requests and API endpoints.
- Configure body-parser to parse request bodies and cors for cross-origin requests.

4. **API Routes:**

- Define separate route files for user management, project listing, application handling, chat functionality, and freelancer profiles.
- Implement route handlers using Express.js to interact with the database:
- User routes: registration, login, profile management.
- Project routes: project creation, listing, details retrieval.
- Application routes: submit proposals, view applications.
- Chat routes: send and receive messages within projects.
- Freelancer routes: view and update profiles, showcase skills.

5. **Data Models:**

- Define Mongoose schemas for each data entity:
- User schema
- Project schema
- Application schema
- Chat schema
- Freelancer schema (extends User schema with skills, experience)
- Create Mongoose models to interact with the MongoDB database.
- Implement CRUD operations for each model to manage data.

6. User Authentication:

- Implement user authentication using JWT or session-based methods.
- Create routes and middleware for user registration, login, and logout.
- Use authentication middleware to protect routes requiring user authorization (e.g., applying for projects).

7. Project Management:

- Allow clients to post projects with details and budget.
- Enable freelancers to browse projects, search by skills, and submit proposals.
- Implement a system for clients to review applications and choose freelancers.

8. Secure Communication & Collaboration:

- Integrate a secure chat system within projects for communication between clients and freelancers.
- Allow file attachments and feedback exchange to facilitate collaboration.

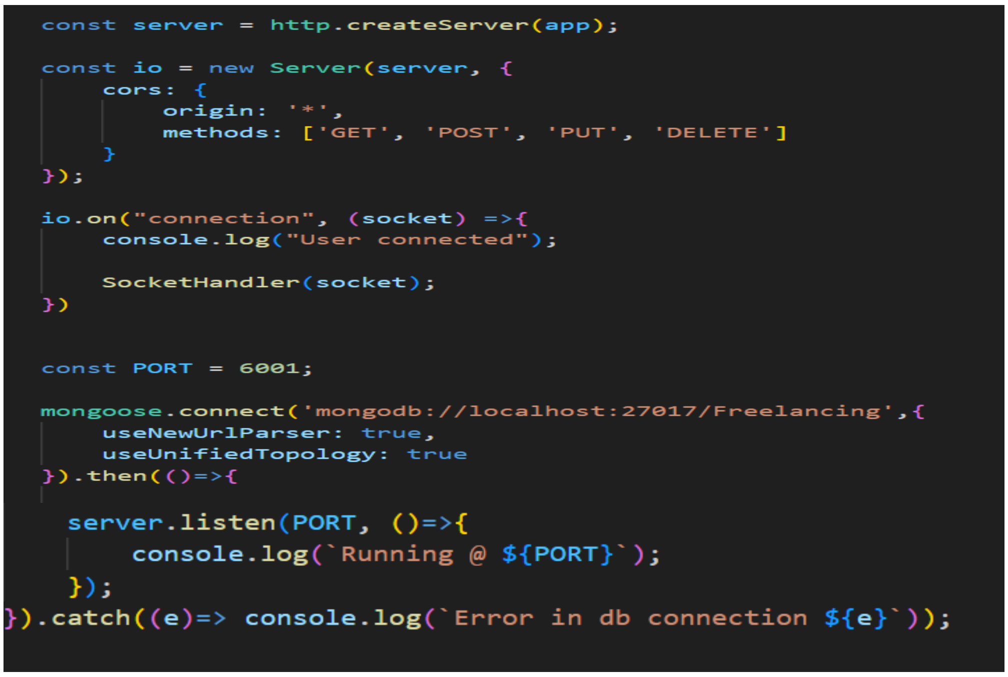
9. Admin Panel (Optional):

- Implement an admin panel with functionalities like:
- Managing users
- Monitoring project updates and applications
- Accessing transaction history

Reference video:
https://drive.google.com/file/d/1a78sICiBmVPkMvgdhC9jmOXolitCBjsn/view?usp=sharing

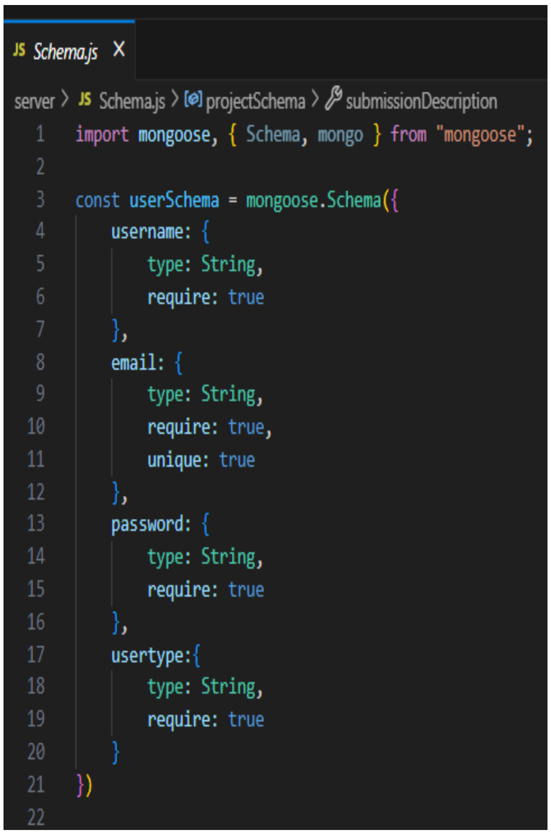
Milestone 3: Database development

- Set up a MongoDB database either locally or using a cloud-based MongoDB service like MongoDB Atlas.
- Create a database and define the necessary collections for users, freelancer, projects, chats, and applications.
- Connect the database to the server with the code provided below.



```
const server = http.createServer(app);

const io = new Server(server, {
    cors: {
        origin: '*',
        methods: ['GET', 'POST', 'PUT', 'DELETE']
    }
});

io.on("connection", (socket) =>{
    console.log("User connected");

    SocketHandler(socket);
})


const PORT = 6001;

mongoose.connect('mongodb://localhost:27017/Freelancing',{
    useNewUrlParser: true,
    useUnifiedTopology: true
}).then(()=>{

  server.listen(PORT, ()=>{
      console.log(`Running @ ${PORT}`);
  });
}).catch((e)=> console.log(`Error in db connection ${e}`));
```

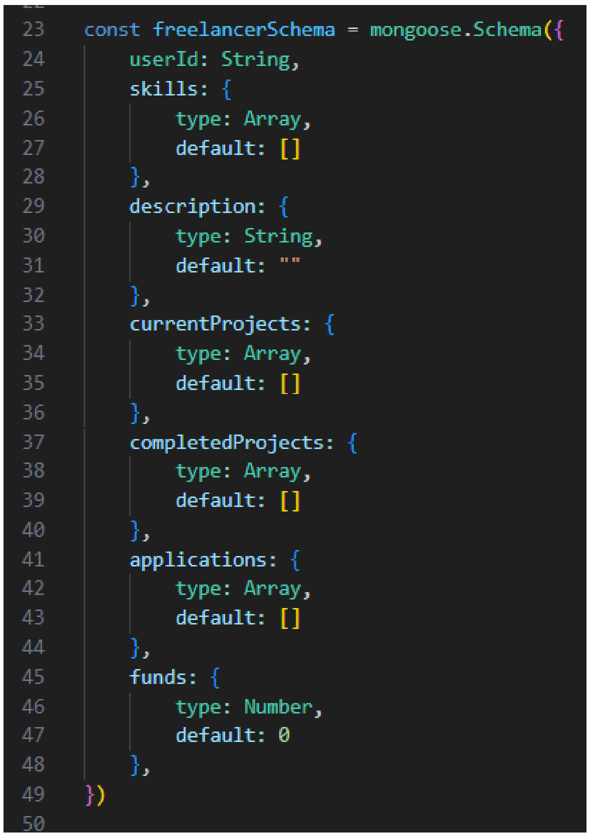
The Schemas for the database are given below



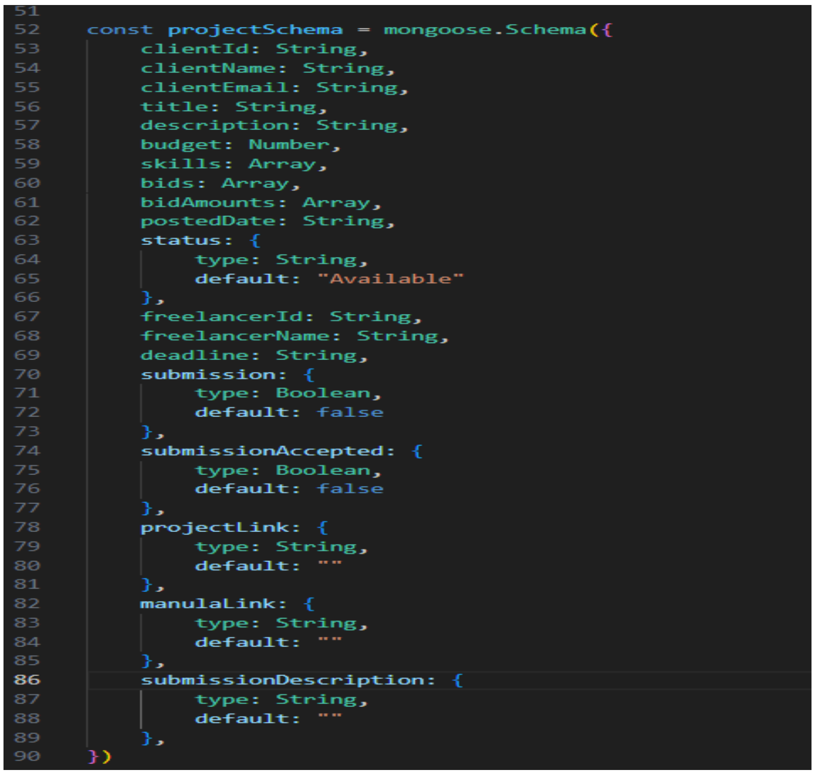
JS Schema.js X

server > JS Schema.js > [@] projectSchema > submissionDescription

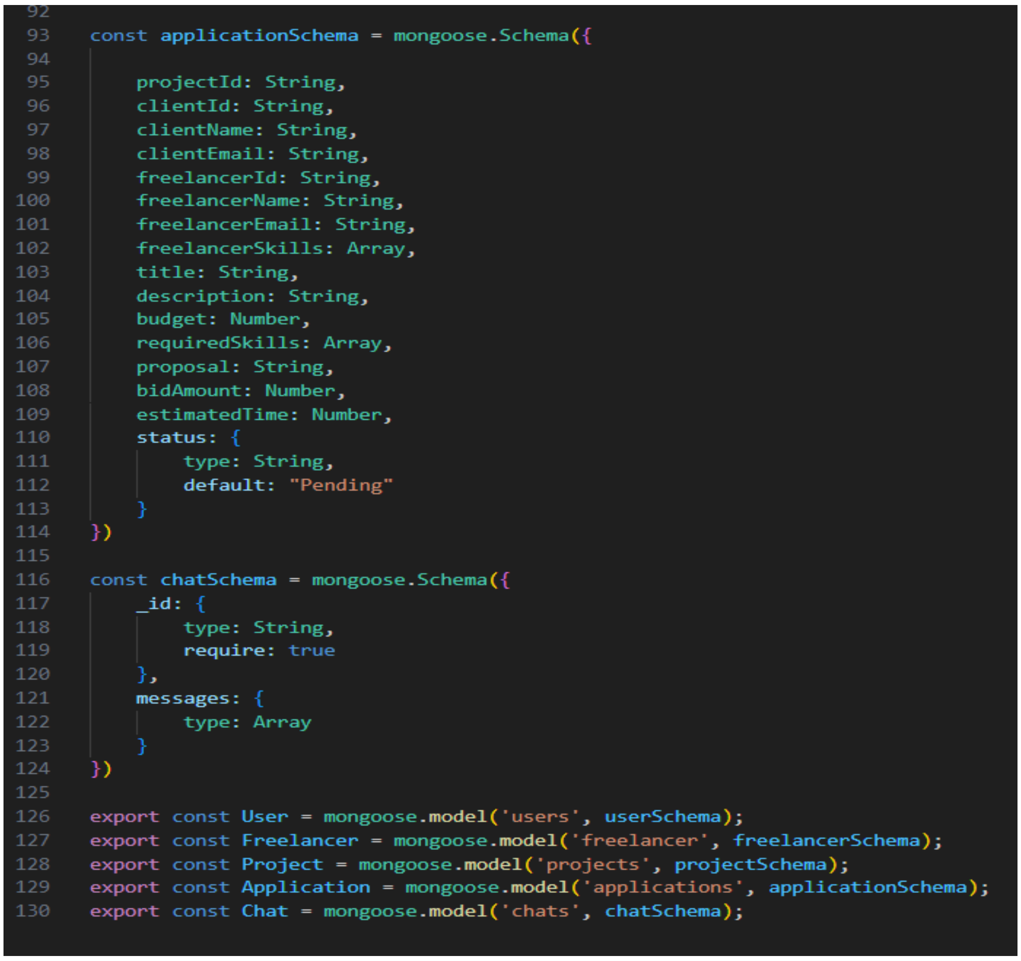
```
import mongoose, { Schema, mongo } from "mongoose";

const userSchema = mongoose.Schema({
    username: {
        type: String,
        require: true
    },
    email: {
        type: String,
        require: true,
        unique: true
    },
    password: {
        type: String,
        require: true
    },
    usertype:{
        type: String,
        require: true
    }
})

```



```
const freelancerSchema = mongoose.Schema({
    userId: String,
    skills: {
        type: Array,
        default: []
    },
    description: {
        type: String,
        default: ""
    },
    currentProjects: {
        type: Array,
        default: []
    },
    completedProjects: {
        type: Array,
        default: []
    },
    applications: {
        type: Array,
        default: []
    },
    funds: {
        type: Number,
        default: 0
    },
})

```



```
const projectSchema = mongoose.Schema({
    clientId: String,
    clientName: String,
    clientEmail: String,
    title: String,
    description: String,
    budget: Number,
    skills: Array,
    bids: Array,
    bidAmounts: Array,
    postedDate: String,
    status: {
        type: String,
        default: "Available"
    },
    freelancerId: String,
    freelancerName: String,
    deadline: String,
    submission: {
        type: Boolean,
        default: false
    },
    submissionAccepted: {
        type: Boolean,
        default: false
    },
    projectLink: {
        type: String,
        default: ""
    },
    manulaLink: {
        type: String,
        default: ""
    },
    submissionDescription: {
        type: String,
        default: ""
    },
})
```



```
const applicationSchema = mongoose.Schema({

    projectId: String,
    clientId: String,
    clientName: String,
    clientEmail: String,
    freelancerId: String,
    freelancerName: String,
    freelancerEmail: String,
    freelancerSkills: Array,
    title: String,
    description: String,
    budget: Number,
    requiredSkills: Array,
    proposal: String,
    bidAmount: Number,
    estimatedTime: Number,
    status: {
        type: String,
        default: "Pending"
    }
})

const chatSchema = mongoose.Schema({
    _id: {
        type: String,
        require: true
    },
    messages: {
        type: Array
    }
})

export const User = mongoose.model('users', userSchema);
export const Freelancer = mongoose.model('freelancer', freelancerSchema);
export const Project = mongoose.model('projects', projectSchema);
export const Application = mongoose.model('applications', applicationSchema);
export const Chat = mongoose.model('chats', chatSchema);
```

Milestone 4: Frontend development

1. Setting the Stage:

The FreelanceHub frontend is built with React.js for a seamless user experience. To get started, we'll:

- Create the initial React project structure.
- Install key libraries to boost functionality and design.
- Organize component and page files for clear development flow.
- This strong foundation ensures an efficient workflow as we develop the FreelanceHub interface.

2. Crafting the User Experience:

Next, we’ll focus on building FreelanceHub’s user interface (UI). This involves:

- Designing reusable UI components such as forms, cards, and navigation links.
- Creating consistent layouts and styling for a professional, engaging appearance.
- Implementing clear navigation for easy movement between features.
- These steps ensure a seamless and user-friendly experience for both clients and freelancers.

3. Bridging the Gap:

The final stage connects the visual interface with the backend data. We'll:

- Integrate the frontend with FreelanceHub’s API endpoints.
- Implement data binding to ensure dynamic updates between user actions and displayed information.

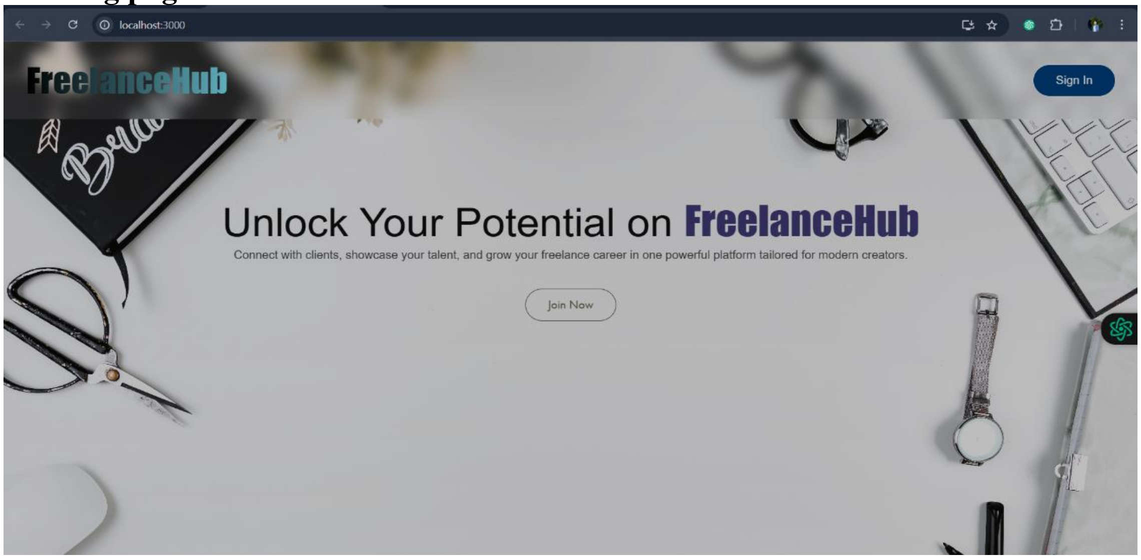
This completes the frontend development, bringing the FreelanceHub platform to life for users.

Reference video:
https://drive.google.com/file/d/1a78sICiBmVPkMvgdhC9jmOXolitCBjsn/view?usp=sharing

Milestone 5: Project Implementation

On completing the development part, we then run the application one last time to verify all the functionalities and look for any bugs in it. The user interface of the application looks a bit like the images provided below.

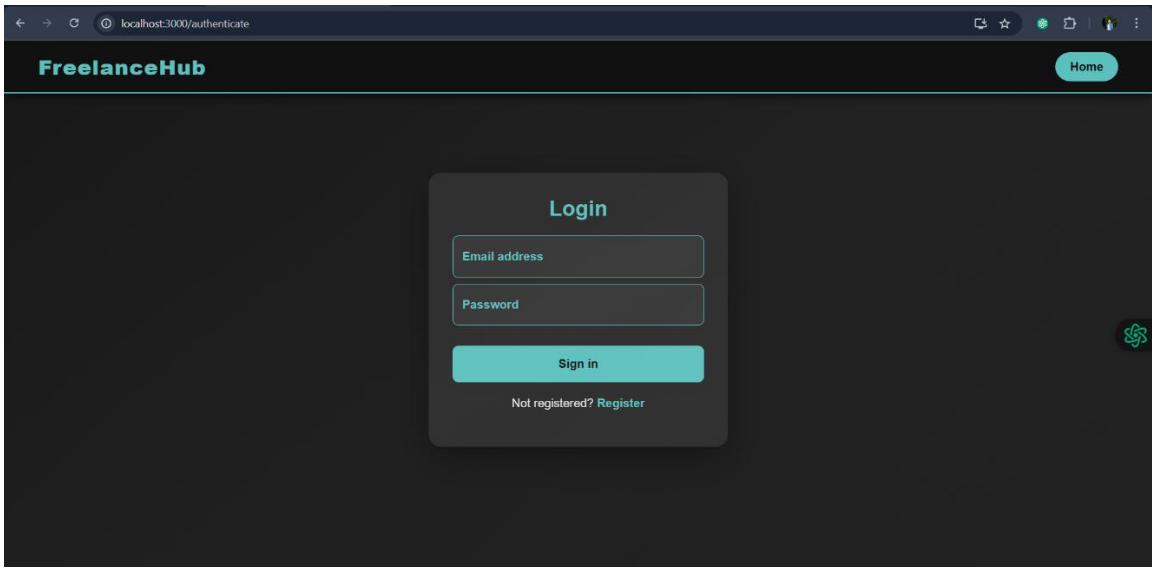
Landing page:



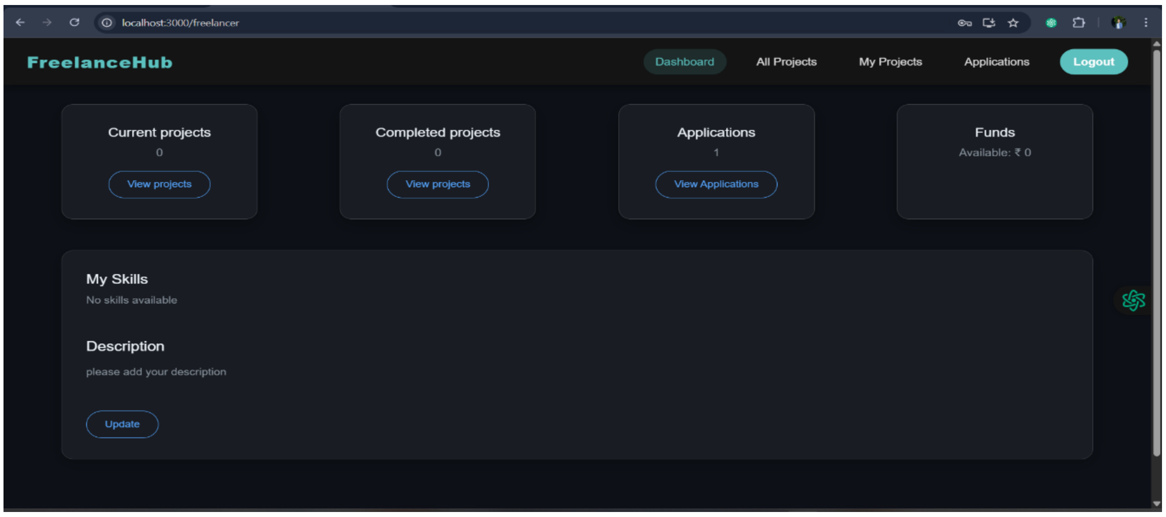
Unlock Your Potential on FreelanceHub

Connect with clients, showcase your talent, and grow your freelance career in one powerful platform tailored for modern creators.

Authentication:



Freelancer dashboard:



Current projects
0

Completed projects
0

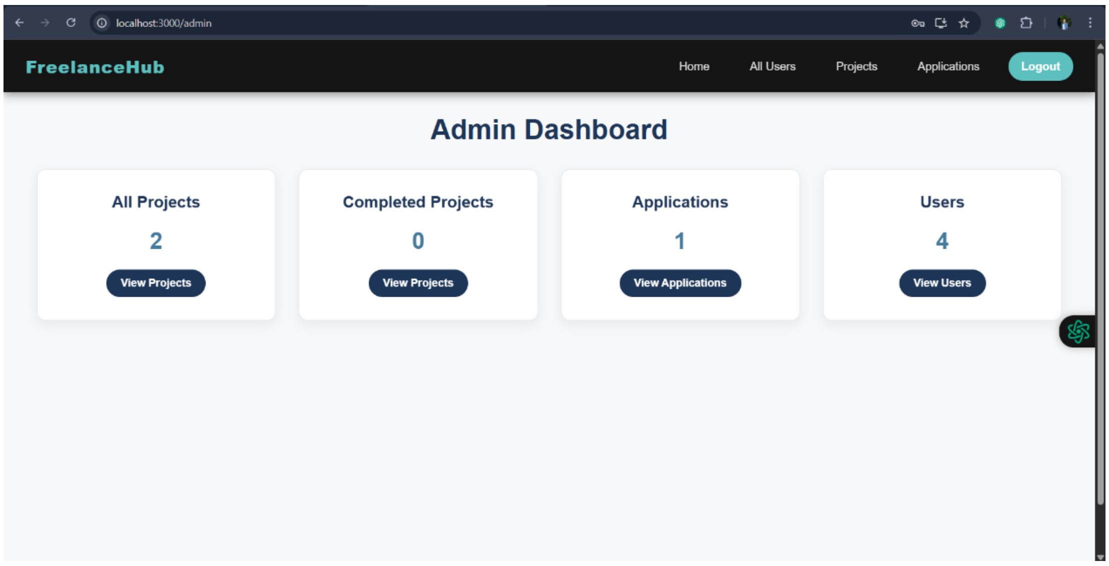
Applications
1

Funds
Available: ₹ 0

My Skills
No skills available

Description
please add your description

Admin dashboard:



Admin Dashboard

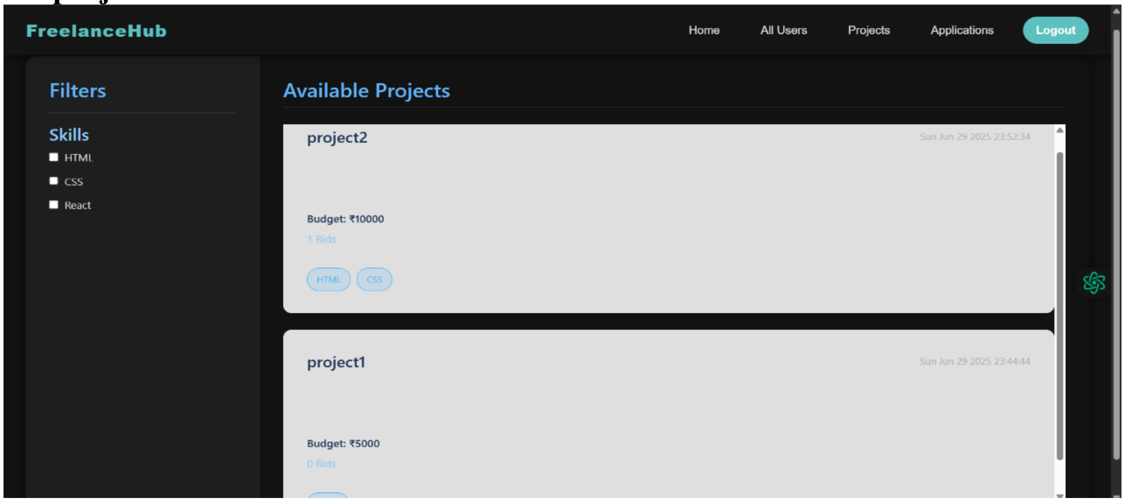
All Projects
2

Completed Projects
0

Applications
1

Users
4

All projects:



Available Projects

project2 Sun Jun 29 2025 23:52:34

Budget: ₹10000

1 Bids

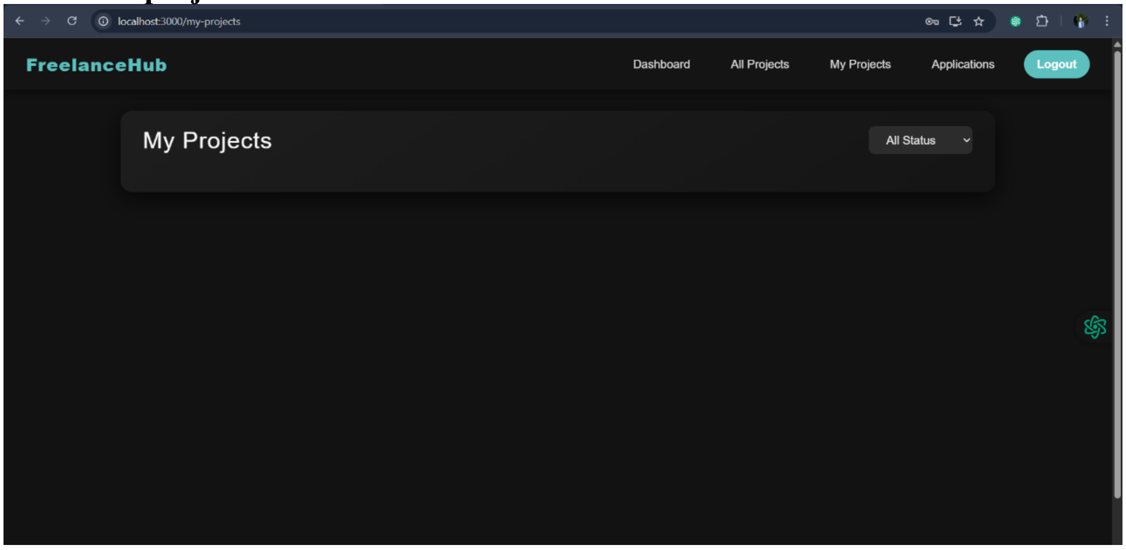
HTML CSS

project1 Sun Jun 29 2025 23:44:44

Budget: ₹5000

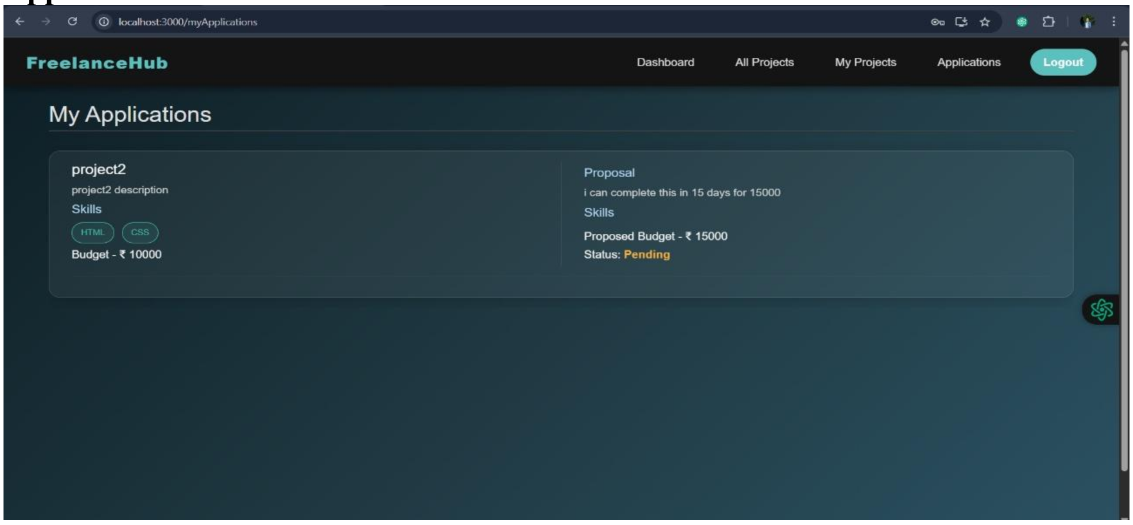
0 Bids

Freelance projects:



My Projects

Applications:

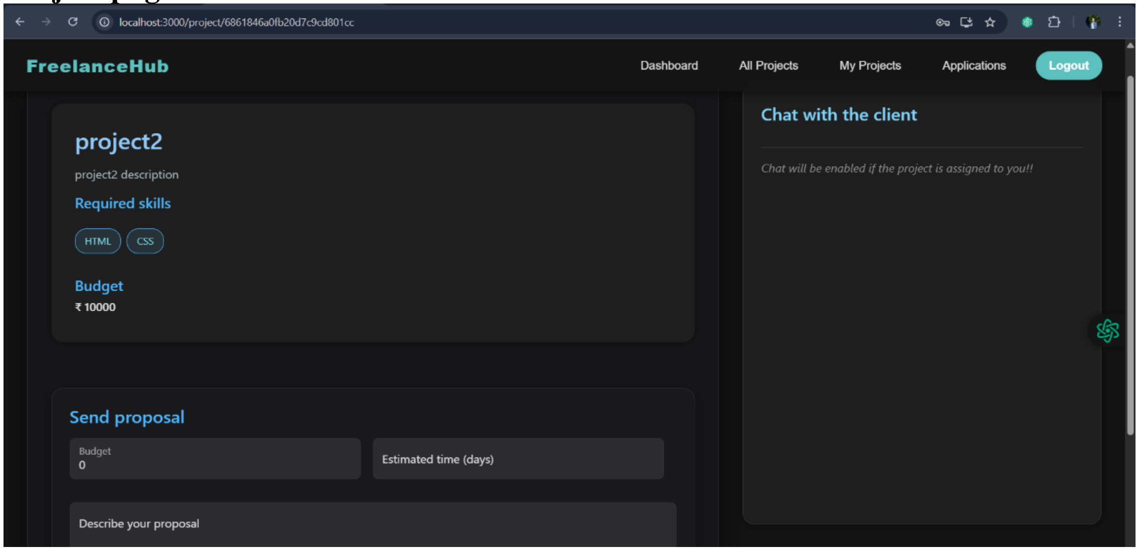

localhost:3000/myApplications

My Applications

project2
project2 description
Skills
HTML CSS
Budget - ₹ 10000

Proposal
i can complete this in 15 days for 15000
Skills
Proposed Budget - ₹ 15000
Status: Pending

Project page:

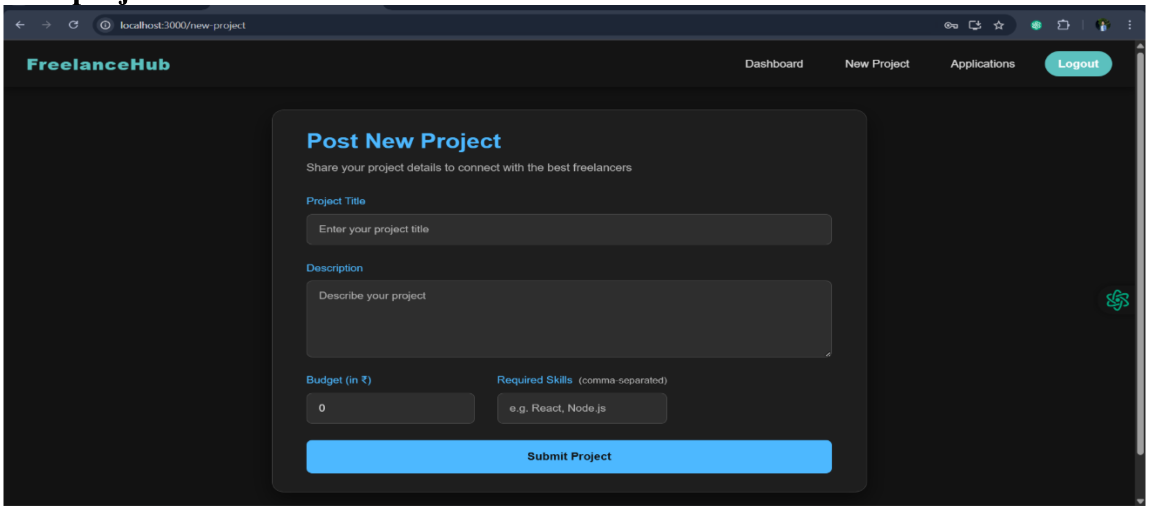

localhost:3000/project/6861846a0fb20d7c9cd801cc

project2
project2 description
Required skills
HTML CSS
Budget
₹ 10000

Chat with the client
Chat will be enabled if the project is assigned to you!!

Send proposal
Budget: 0
Estimated time (days)
Describe your proposal

New project:


localhost:3000/new-project

Post New Project
Share your project details to connect with the best freelancers

Project Title: Enter your project title
Description: Describe your project
Budget (in ₹): 0
Required Skills (comma-separated): e.g. React, Node.js

For any further doubts or help, please consider the code in the drive link given below,

https://drive.google.com/drive/folders/1dnRpwEHi05fugNUDGTM7h7UaWqbBcmhg?usp=sharing

The demo of the app is available at:

https://drive.google.com/file/d/1a78sICiBmVPkMvgdhC9jmOXolitCBjsn/view?usp=sharing

** Happy Coding **