Full Stack Development with MERN
Project Documentation

1.INTRODUCTION

- **Project Title**: FREELANCING APPLICATION MERN
- **Team Members** (TEAM ID: NM2024TMID00550)
    - **Rohith CS – Project Manager (Team Lead)**
    - **Praveen P – Frontend Developer**
    - **Sathya S – Backend Developer**
    - **Santhosh C – Database Manager**

2.PROJECT OVERVIEW

- **Purpose: RPS Works** is a freelancing platform that connects clients who need services with freelancers who offer those services. The purpose of RPS Works is to bridge the gap between clients needing specific services and freelancers offering their skills. By providing a streamlined process for project posting, bidding, and communication, RPS Works facilitates efficient interactions between clients and freelancers, making it easier to initiate, manage, and complete projects.
- **Features:**
    - **User Registration & Authentication**: Secure account creation and login functionality with different roles for clients, freelancers, and admin.
    - **Freelancer Dashboard**: A personalized space where freelancers can update their skills, view available projects, submit bids, and track ongoing projects.
    - **Client Dashboard**: Allows clients to create and post new projects, review freelancer bids, and manage the status of their projects.
    - **Real-time Chat**: Allows seamless communication between freelancers and clients on project requirements and progress.
    - **Project Posting and Bidding**: Clients post job listings with project details, required skills, and budget. Freelancers can place bids by submitting proposals and estimated days.
    - **Admin Panel**: Provides the admin with a centralized view and control over all users, projects, and applications, allowing monitoring and management of the entire system.

3.ARCHITECTURE

- **Client:**
  - **React.js**: The frontend was built with React, chosen for its component-based architecture, allowing for a scalable and maintainable UI.
  - **State Management:** Uses React's context or a global state management solution like Redux to manage user session data and project state.
  - **Routing:** The application uses React Router for navigation between different views, such as the client dashboard, freelancer dashboard, and bid pages.
  - Main components and pages include:
    - **Components:** Login, Register, Navbar, etc are essential components for user authentication and navigation.
    - **Dashboard Components:** Separate dashboards **(Client.js** and **Freelancer.js)** that provide tailored views and functionality based on user roles.
    - **Context:** Centralized context management in **GeneralContext.js** for handling global state.
    - **Project Components**: Dedicated components for project interactions, like NewProject.js for creating projects, and ProjectApplications.js for viewing and managing applications.

- **Server:**
  - Implemented with **Node.js** and **Express.js**, located in the server folder. API endpoints handle user registration, authentication, project management, and bid processing.
  - Key files include:
    - **index.js**: The main entry point for the server.
    - **Schema.js**: Contains database schema definitions for users, projects, and bids.
    - **SocketHandler.js**: Manages real-time socket connections for chat functionality.

- **Database:**
  - **MongoDB Database Name:** freelancing
  - Collections:
    - ***applications*****:** Stores application details for freelancers and clients.
    - ***chats*****:** Manages communication between freelancers and clients.
    - ***freelancers*****:** Contains information about freelancer profiles and skills.
    - ***projects:*** Stores project data, including client details, budget, and skills required.
    - ***Users:*** Manages user profiles, login credentials, and roles (client, freelancer, admin).

4.SETUP INSTRUCTIONS

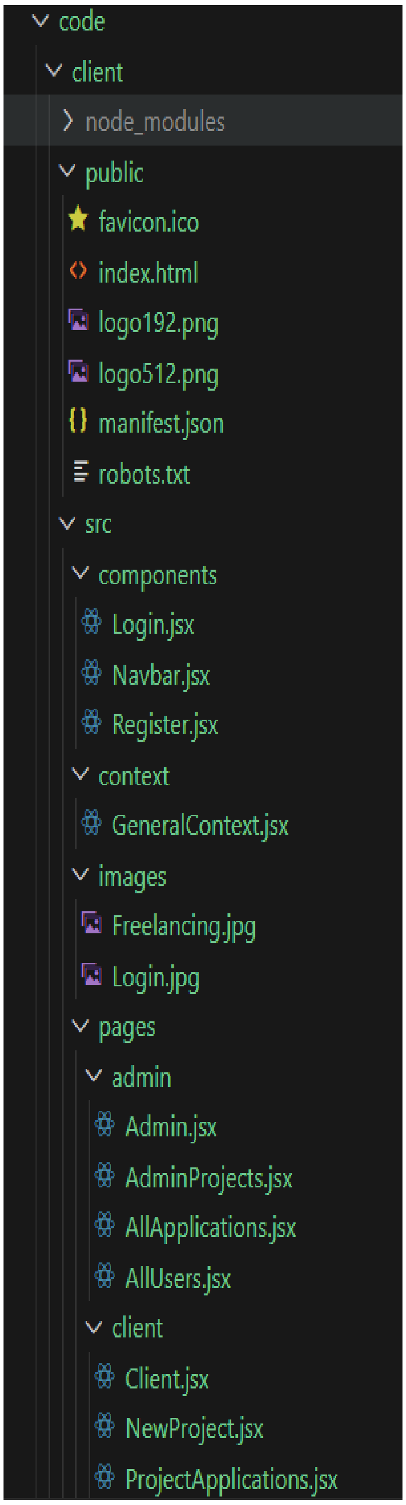
- **Prerequisites:**
  - **Node.js**: Version 14 or higher, required to run the backend and install packages.
  - **MongoDB**: A local instance or cloud-based MongoDB (such as MongoDB Atlas) for storing project data.
  - **npm**: Node package manager to handle dependencies for both frontend and backend.

- **Installation:**
  1. Clone the repository:
     - Clone the project repository to your local machine:
       - `git clone <repository-url>`
  2. Navigate to the project directory.
  3. Install project dependencies:
     - The project contains two main directories: **client** for the frontend and **server** for the backend. You will need to install dependencies for each separately.
     - First, install dependencies for the **frontend:**
       - `cd client`
       - `npm install`
     - Then, navigate to the **server** directory and install backend dependencies**:**
       - `cd server`
       - `npm install`
  4. Set up environment variables for database configuration.
  5. Run the application:
     - Start the backend server:
       - `cd server`
       - `node index.js`
     - In a new terminal, start the frontend:
       - `cd client`
       - `npm start`
     - The server will be accessible on http://localhost:6001 and the client on http://localhost:3000.

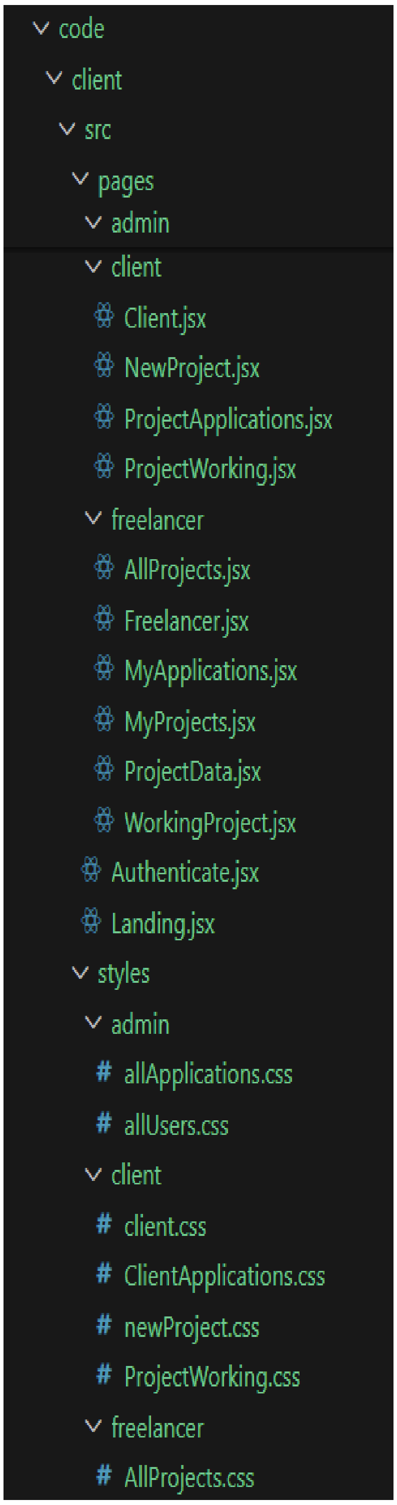
5.FOLDER STRUCTURE

- **Client:**
  - **node_modules:** Contains all the npm packages required for the frontend project.

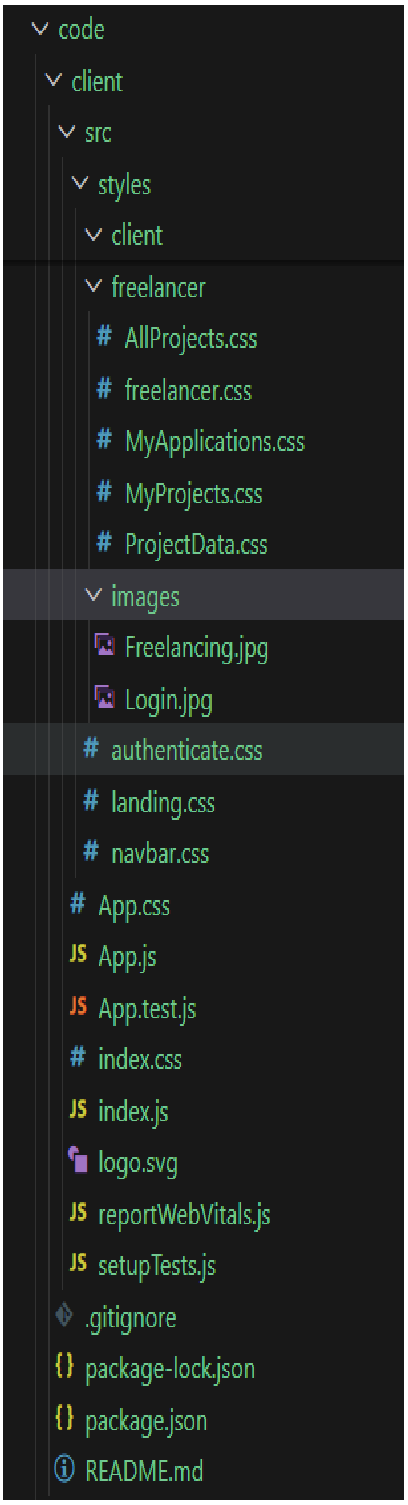
- **src/components**: Contains reusable UI components like Login, Navbar, and Register.
- **src/context**: Includes GeneralContext.js for shared state management.
- **src/pages**: Main application pages such as AdminProjects, Freelancer, Client, and Landing.
- **styles**: Contains CSS and styling files that ensure consistent design.



```
v code
  v client
    > node_modules
    v public
      favicon.ico
      index.html
      logo192.png
      logo512.png
      manifest.json
      robots.txt
    v src
      v components
        Login.jsx
        Navbar.jsx
        Register.jsx
      v context
        GeneralContext.jsx
      v images
        Freelancing.jpg
        Login.jpg
      v pages
        v admin
          Admin.jsx
          AdminProjects.jsx
          AllApplications.jsx
          AllUsers.jsx
        v client
          Client.jsx
          NewProject.jsx
          ProjectApplications.jsx
```

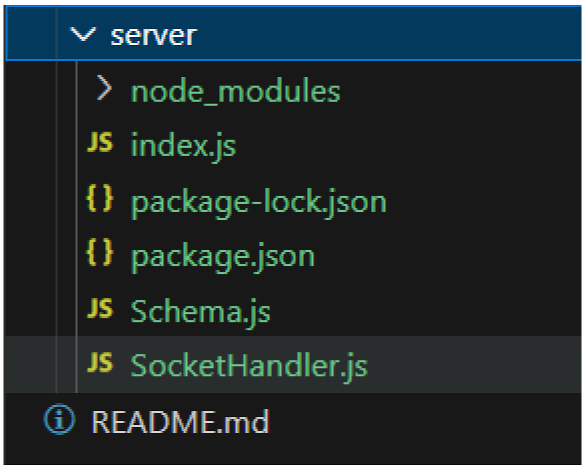


```
v code
  v client
    v src
      v pages
        v admin
        v client
          Client.jsx
          NewProject.jsx
          ProjectApplications.jsx
          ProjectWorking.jsx
        v freelancer
          AllProjects.jsx
          Freelancer.jsx
          MyApplications.jsx
          MyProjects.jsx
          ProjectData.jsx
          WorkingProject.jsx
        Authenticate.jsx
        Landing.jsx
      v styles
        v admin
          allApplications.css
          allUsers.css
        v client
          client.css
          ClientApplications.css
          newProject.css
          ProjectWorking.css
        v freelancer
          AllProjects.css
```



```
v code
  v client
    v src
      v styles
        v client
        v freelancer
          AllProjects.css
          freelancer.css
          MyApplications.css
          MyProjects.css
          ProjectData.css
        v images
          Freelancing.jpg
          Login.jpg
        authenticate.css
        landing.css
        navbar.css
      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
```

- **Server:**
  - **index.js:** Server entry point that initializes Express and configures middleware.
  - **Schema.js**: Defines all schemas and models, using MongoDB to handle data structure for users, projects, and bids.
  - **SocketHandler.js**: Manages real-time messaging, using sockets for live client-freelancer communication.



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

6.RUNNING THE APPLICATION

- **Client:**
  - Run `npm start` within the client directory to launch the React app.
  - The application will launch on a local development server, typically at `http://localhost:3000`.

- **Server:**
  - Run `node index.js` within the server directory to start the Express server.
  - The backend runs on a different port http://localhost:6001.

7.API DOCUMENTATION

- **Authentication Endpoints:**
  - `POST /register`: Registers a new user with role-specific functionality (client or freelancer).
  - `POST /login`: Authenticates a user and returns a JSON Web Token (JWT) for session management.

- **Project Management Endpoints:**
    - `POST /projects`: Allows clients to create and post new projects.
    - `GET /projects`: Lists all available projects for freelancers to browse and select.
    - `POST /projects/:id/bid`: Freelancers submit a bid, including their proposal and estimated timeline.

- **Chat Endpoints:**
    - `POST /chat`**:** Starts a chat session between freelancer and client, using socket events to enable real-time communication.

- **Admin Endpoints:**
    - `GET /admin/projects:` Admins view all project postings, with options to manage or remove them if needed.
    - `GET /admin/users:` Admins view all users and manage access, allowing for moderation of platform activities.

8.AUTHENTICATION

- **JWT-Based Authentication:**
    - RPS Works uses JWTs for secure user authentication. Tokens are generated upon login and are stored in the client's local storage, enabling secure access to API endpoints. Each request to protected endpoints includes the token, which is verified on the server.
    - Tokens are stored client-side and included with API requests for protected endpoints. The backend validates tokens to ensure secure access.

9.USER INTERFACE

- **Key Screens:**
    - **Landing Page**: Provides option to sign in. Users can select their role during registration.
    - **Freelancer Dashboard**: Displays available projects, allows profile updates, and tracks project bids.
    - **Client Dashboard:** Provides tools for clients to create new projects, review freelancer bids, and manage project statuses.
    - **Chat Interface**: Real-time messaging for clients and freelancers to discuss project requirements within each project.

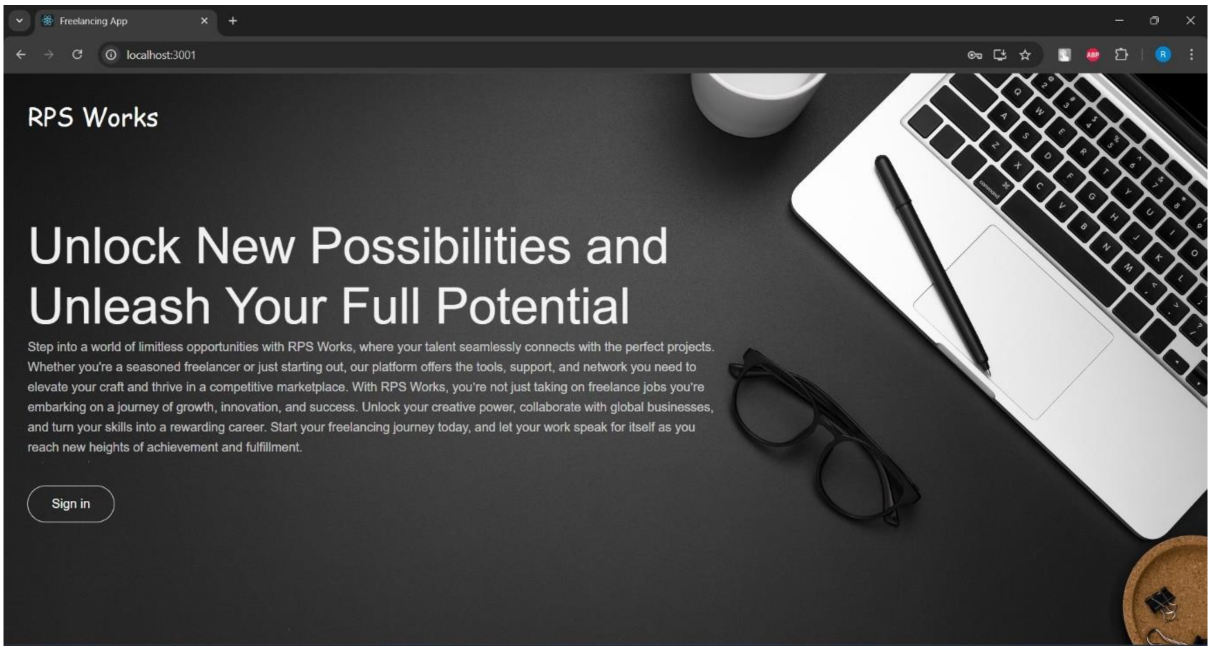
- **Admin Panel**: Offers an overview of platform activities, such as monitoring users, projects, and bids for moderation purposes.

10.TESTING

- **Testing Strategy:**
    - **Unit Testing:**
        - Test individual components, such as login and registration forms, to ensure correct functionality.
    - **Verified major workflows:**
        - Verify the functionality of complete workflows, such as project posting and bid approval, to verify interaction between frontend and backend.
- **Database Validation:**
    - Tools used:
        - **Postman**: API testing for endpoints to ensure request-response integrity and correctness.

11.SCREENSHOTS OR DEMO

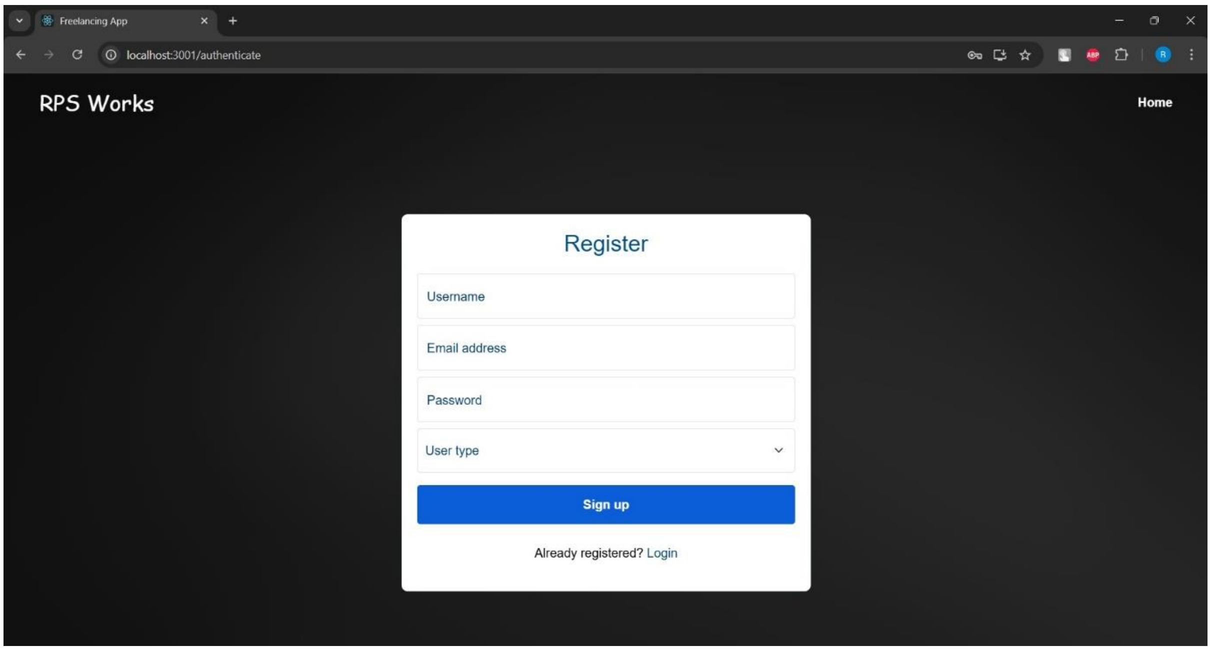
- **Home page**



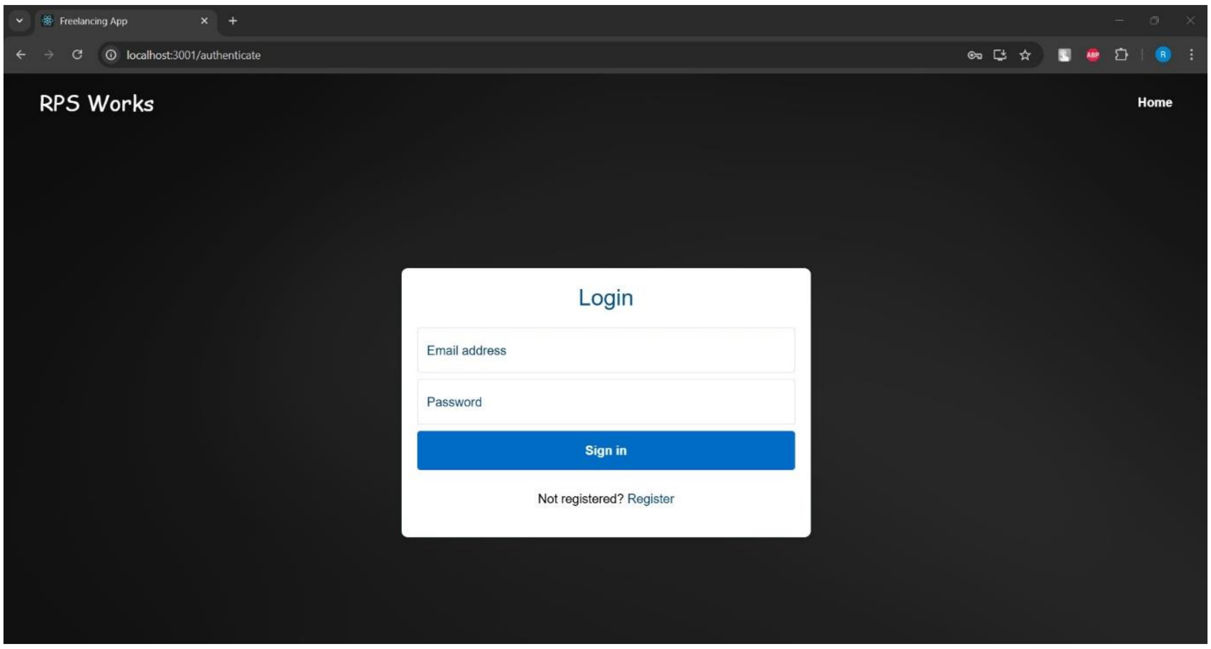
Unlock New Possibilities and Unleash Your Full Potential

Step into a world of limitless opportunities with RPS Works, where your talent seamlessly connects with the perfect projects. Whether you're a seasoned freelancer or just starting out, our platform offers the tools, support, and network you need to elevate your craft and thrive in a competitive marketplace. With RPS Works, you're not just taking on freelance jobs you're embarking on a journey of growth, innovation, and success. Unlock your creative power, collaborate with global businesses, and turn your skills into a rewarding career. Start your freelancing journey today, and let your work speak for itself as you reach new heights of achievement and fulfillment.

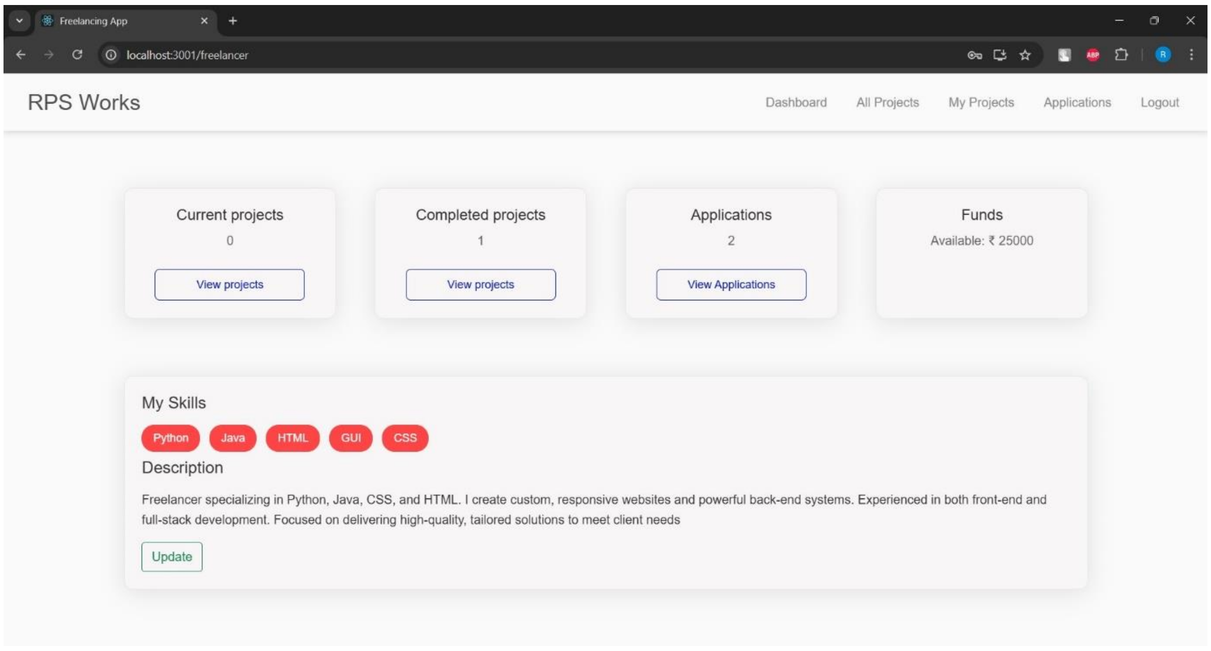
- **Register page**



- **Login page**



- **Freelancer Dashboard**


Freelancing App

localhost:3001/freelancer

RPS Works

Dashboard All Projects My Projects Applications Logout

Current projects	Completed projects	Applications	Funds
0	1	2	Available: ₹ 25000
View projects	View projects	View Applications	

My Skills

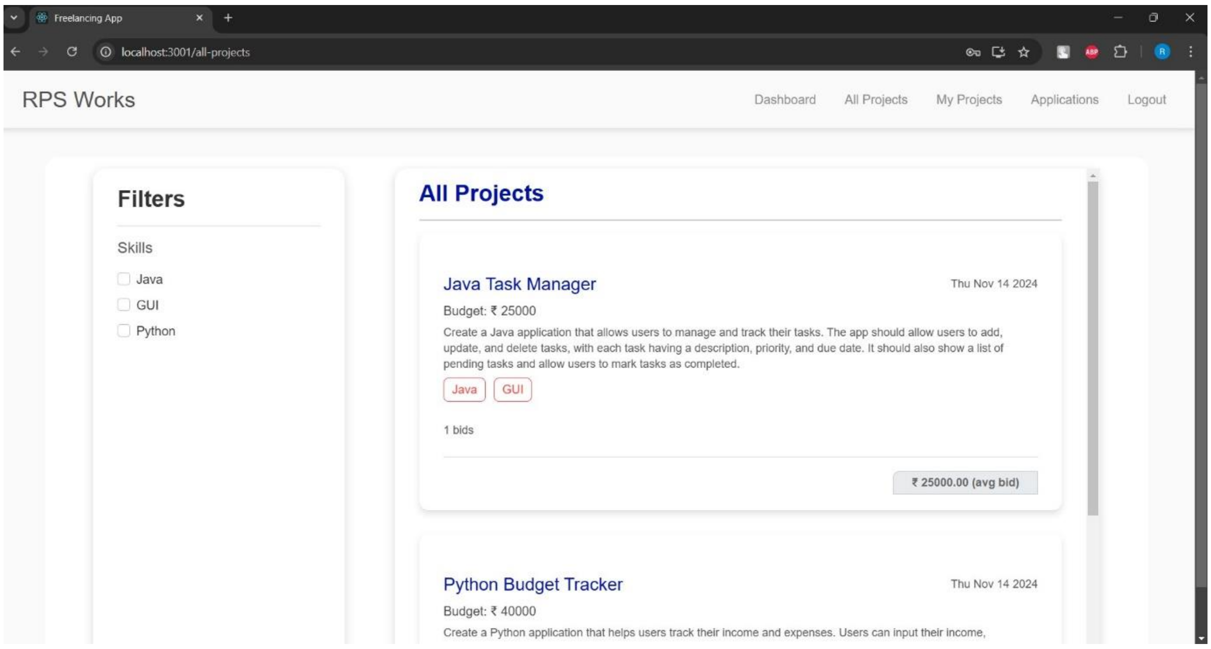
Python Java HTML GUI CSS

Description

Freelancer specializing in Python, Java, CSS, and HTML. I create custom, responsive websites and powerful back-end systems. Experienced in both front-end and full-stack development. Focused on delivering high-quality, tailored solutions to meet client needs

Update

- **Freelancer's All Projects page**


Freelancing App

localhost:3001/all-projects

RPS Works

Dashboard All Projects My Projects Applications Logout

Filters

Skills

- Java
- GUI
- Python

All Projects

Java Task Manager

Thu Nov 14 2024

Budget: ₹ 25000

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

Java GUI

1 bids

₹ 25000.00 (avg bid)

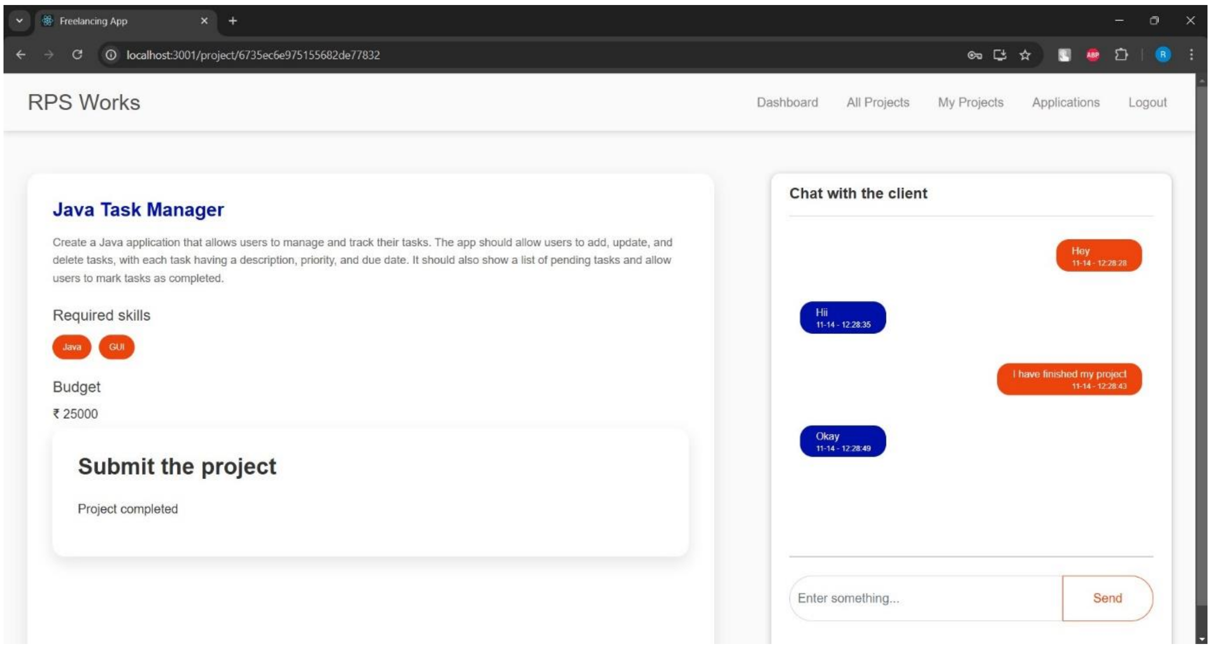
Python Budget Tracker

Thu Nov 14 2024

Budget: ₹ 40000

Create a Python application that helps users track their income and expenses. Users can input their income,

- **Projects page**



Freelancing App

localhost:3001/project/6735ec6e975155682de77832

RPS Works

Dashboard All Projects My Projects Applications Logout

Java Task Manager

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

Required skills

Java GUI

Budget

₹ 25000

Submit the project

Project completed

Chat with the client

Hey
11-14 - 12:28:28

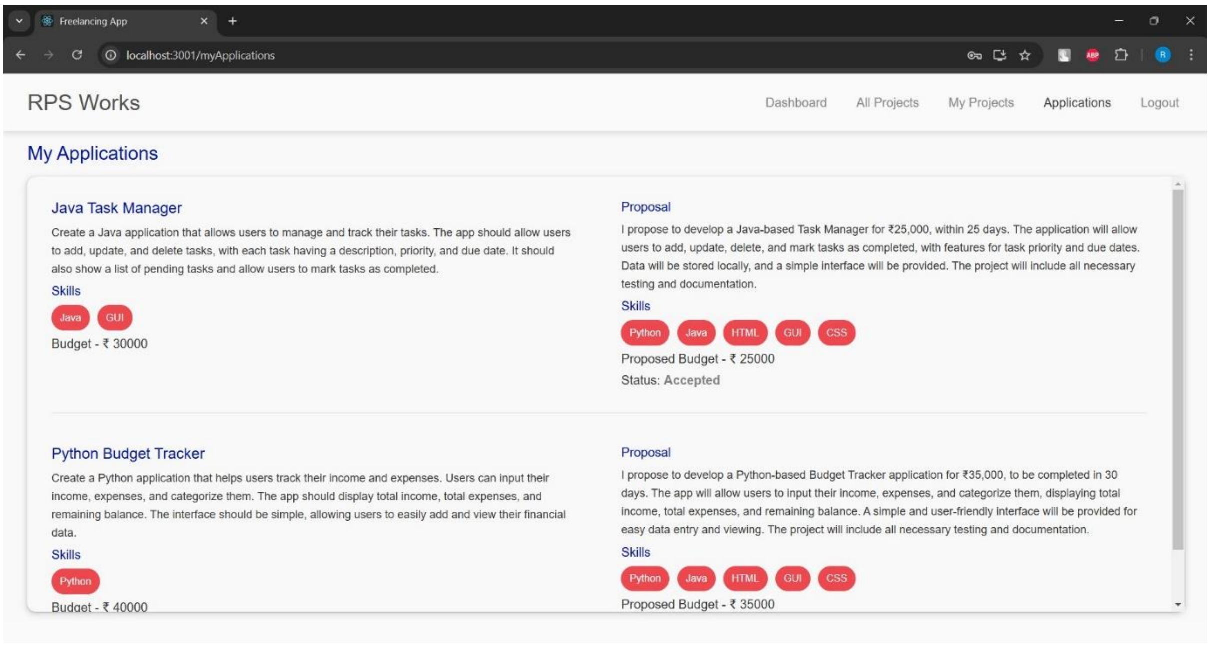
Hii
11-14 - 12:28:35

I have finished my project
11-14 - 12:28:43

Okay
11-14 - 12:28:49

Enter something... Send

- **Freelancer's Applications page**



Freelancing App

localhost:3001/myApplications

RPS Works

Dashboard All Projects My Projects Applications Logout

My Applications

Java Task Manager

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

Skills

Java GUI

Budget - ₹ 30000

Proposal

I propose to develop a Java-based Task Manager for ₹25,000, within 25 days. The application will allow users to add, update, delete, and mark tasks as completed, with features for task priority and due dates. Data will be stored locally, and a simple interface will be provided. The project will include all necessary testing and documentation.

Skills

Python Java HTML GUI CSS

Proposed Budget - ₹ 25000

Status: Accepted

Python Budget Tracker

Create a Python application that helps users track their income and expenses. Users can input their income, expenses, and categorize them. The app should display total income, total expenses, and remaining balance. The interface should be simple, allowing users to easily add and view their financial data.

Skills

Python

Budget - ₹ 40000

Proposal

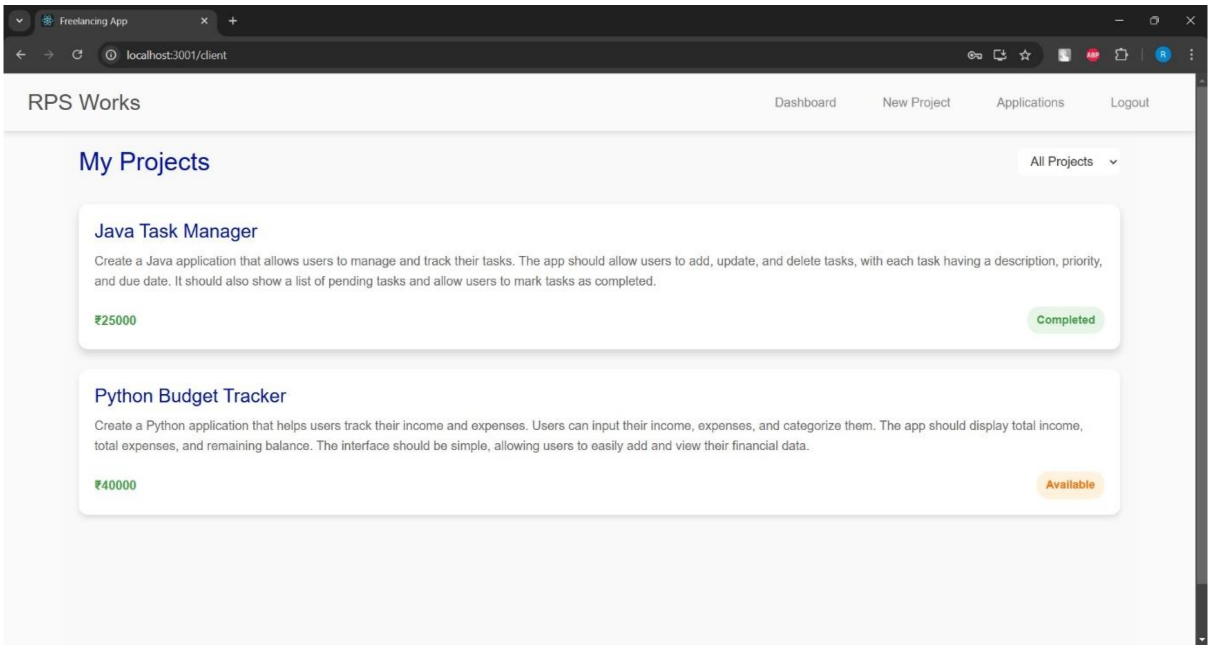
I propose to develop a Python-based Budget Tracker application for ₹35,000, to be completed in 30 days. The app will allow users to input their income, expenses, and categorize them, displaying total income, total expenses, and remaining balance. A simple and user-friendly interface will be provided for easy data entry and viewing. The project will include all necessary testing and documentation.

Skills

Python Java HTML GUI CSS

Proposed Budget - ₹ 35000

- **Client Dashboard**



Freelancing App

localhost:3001/client

RPS Works

Dashboard New Project Applications Logout

My Projects

All Projects

Java Task Manager

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

₹25000

Completed

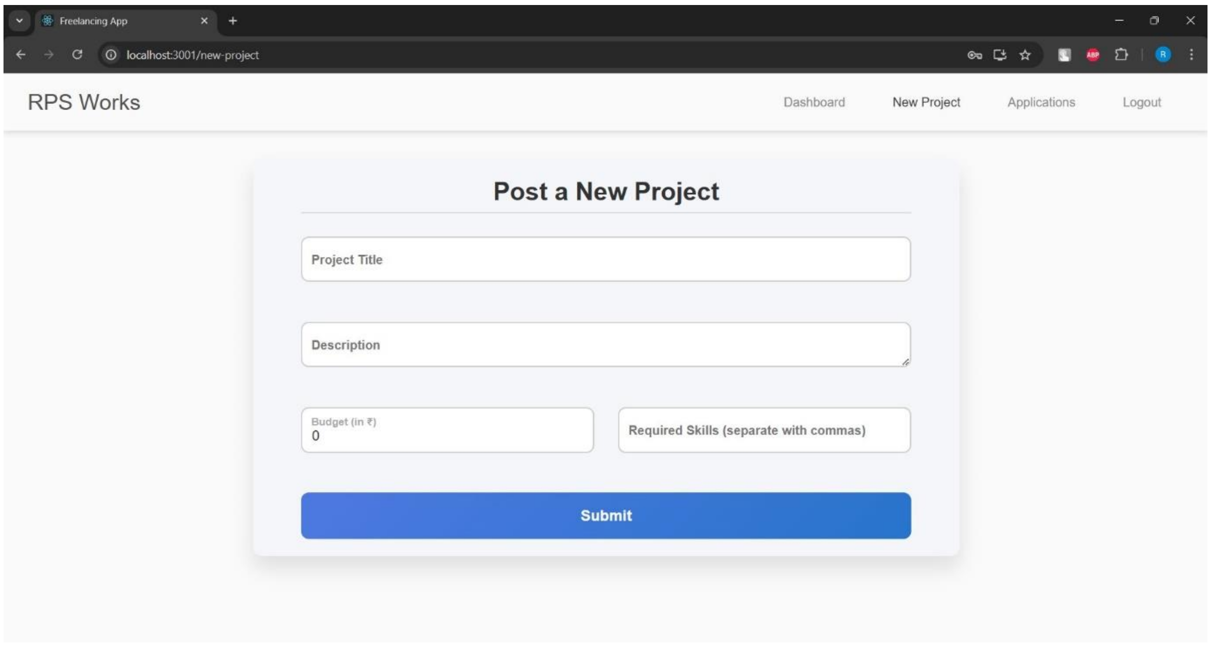
Python Budget Tracker

Create a Python application that helps users track their income and expenses. Users can input their income, expenses, and categorize them. The app should display total income, total expenses, and remaining balance. The interface should be simple, allowing users to easily add and view their financial data.

₹40000

Available

- **New project page**



Freelancing App

localhost:3001/new-project

RPS Works

Dashboard New Project Applications Logout

Post a New Project

Project Title

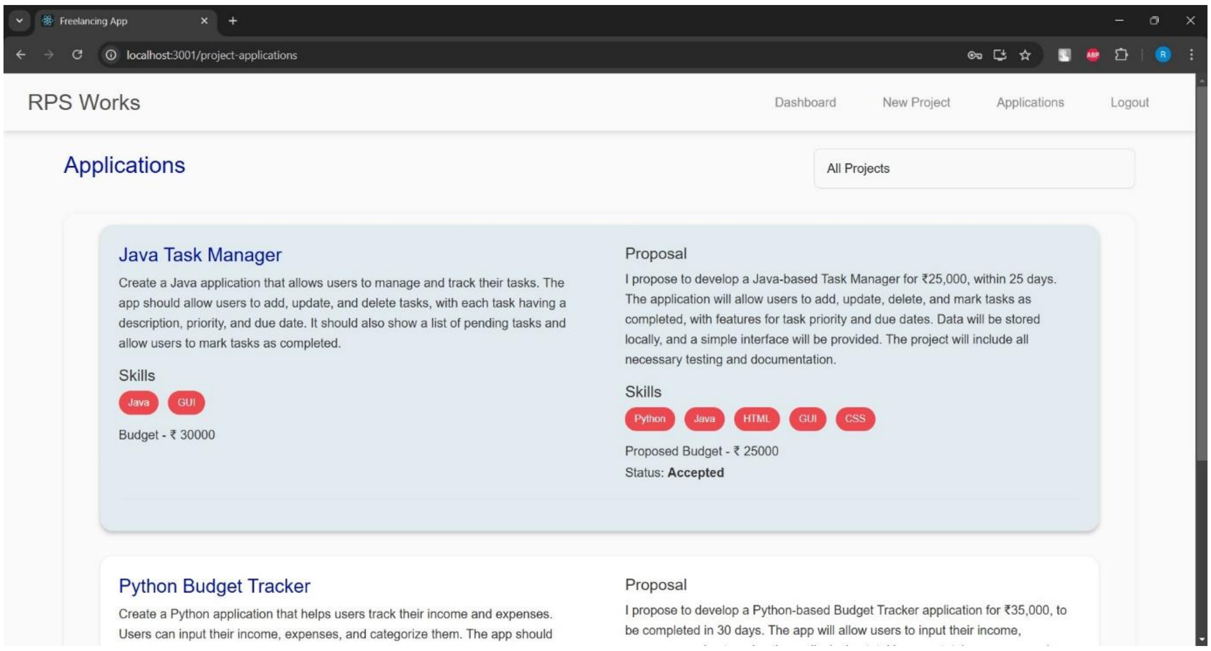
Description

Budget (in ₹)
0

Required Skills (separate with commas)

Submit

- **Client’s Applications page**



Applications

Java Task Manager

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

Skills

Java GUI

Budget - ₹ 30000

Proposal

I propose to develop a Java-based Task Manager for ₹25,000, within 25 days. The application will allow users to add, update, delete, and mark tasks as completed, with features for task priority and due dates. Data will be stored locally, and a simple interface will be provided. The project will include all necessary testing and documentation.

Skills

Python Java HTML GUI CSS

Proposed Budget - ₹ 25000

Status: **Accepted**

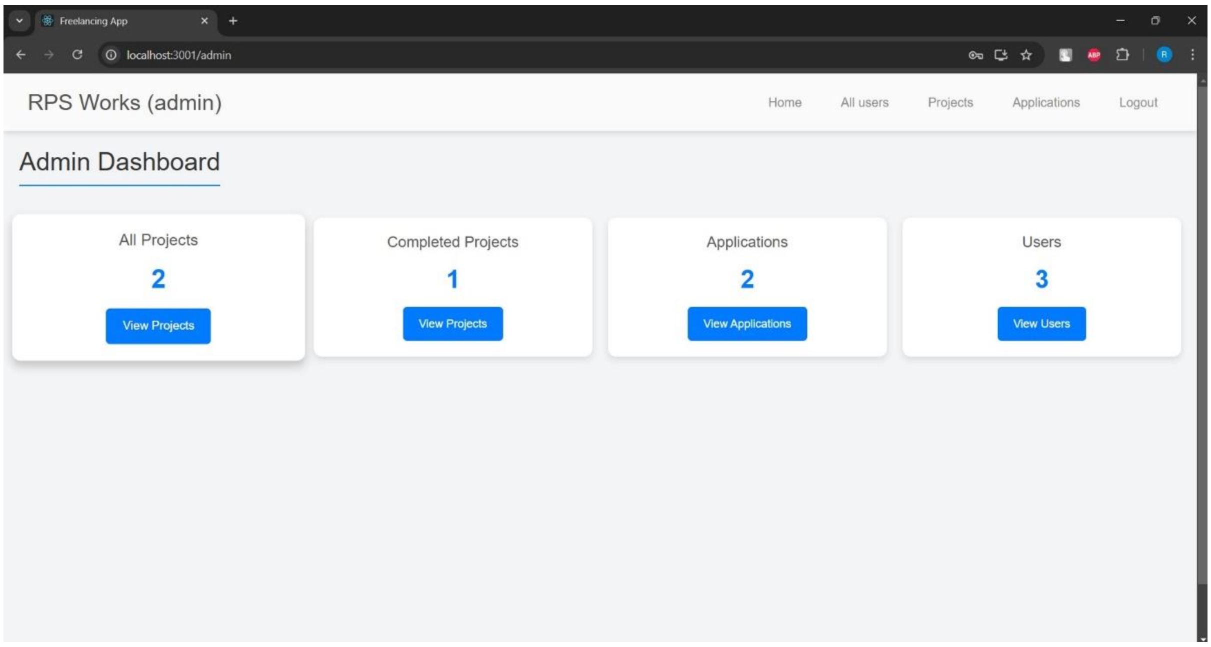
Python Budget Tracker

Create a Python application that helps users track their income and expenses. Users can input their income, expenses, and categorize them. The app should

Proposal

I propose to develop a Python-based Budget Tracker application for ₹35,000, to be completed in 30 days. The app will allow users to input their income,

- **Admin’s Home page**



Admin Dashboard

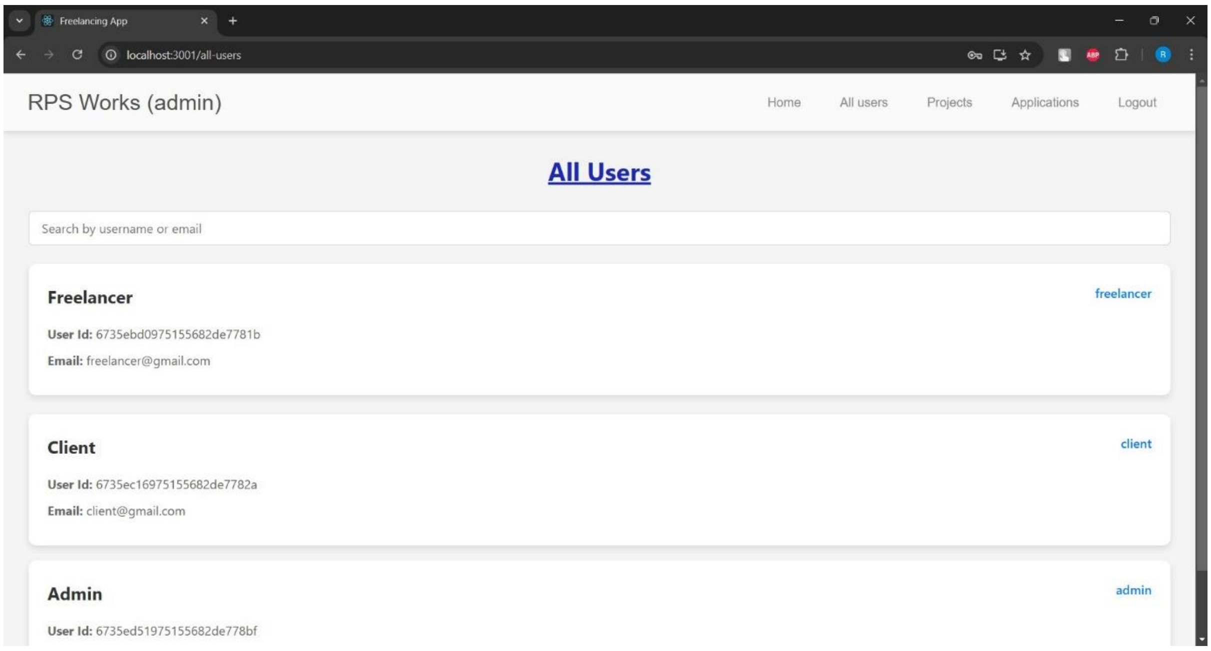
All Projects 2 View Projects

Completed Projects 1 View Projects

Applications 2 View Applications

Users 3 View Users

- **Admin’s All users page**



All Users

Search by username or email

Freelancer freelancer

User Id: 6735ebd0975155682de7781b

Email: freelancer@gmail.com

Client client

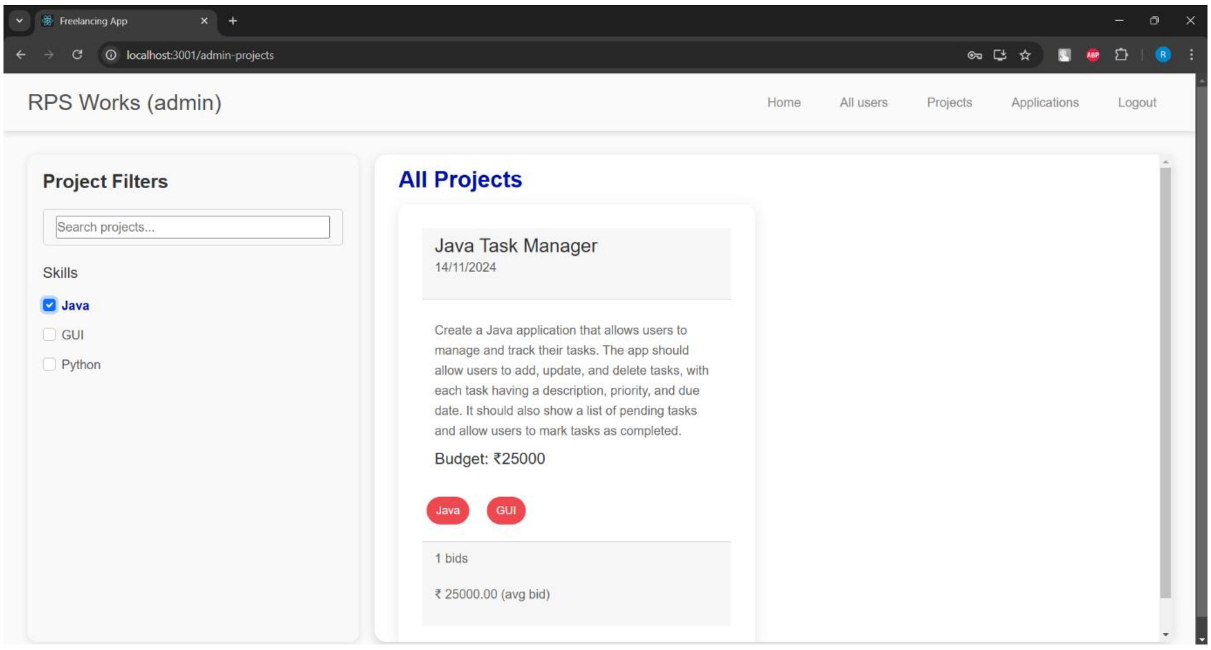
User Id: 6735ec16975155682de7782a

Email: client@gmail.com

Admin admin

User Id: 6735ed51975155682de778bf

- **Admin’s All Projects page**



Project Filters

Search projects...

Skills

[✓ Java]

GUI

Python

All Projects

Java Task Manager

14/11/2024

Create a Java application that allows users to manage and track their tasks. The app should allow users to add, update, and delete tasks, with each task having a description, priority, and due date. It should also show a list of pending tasks and allow users to mark tasks as completed.

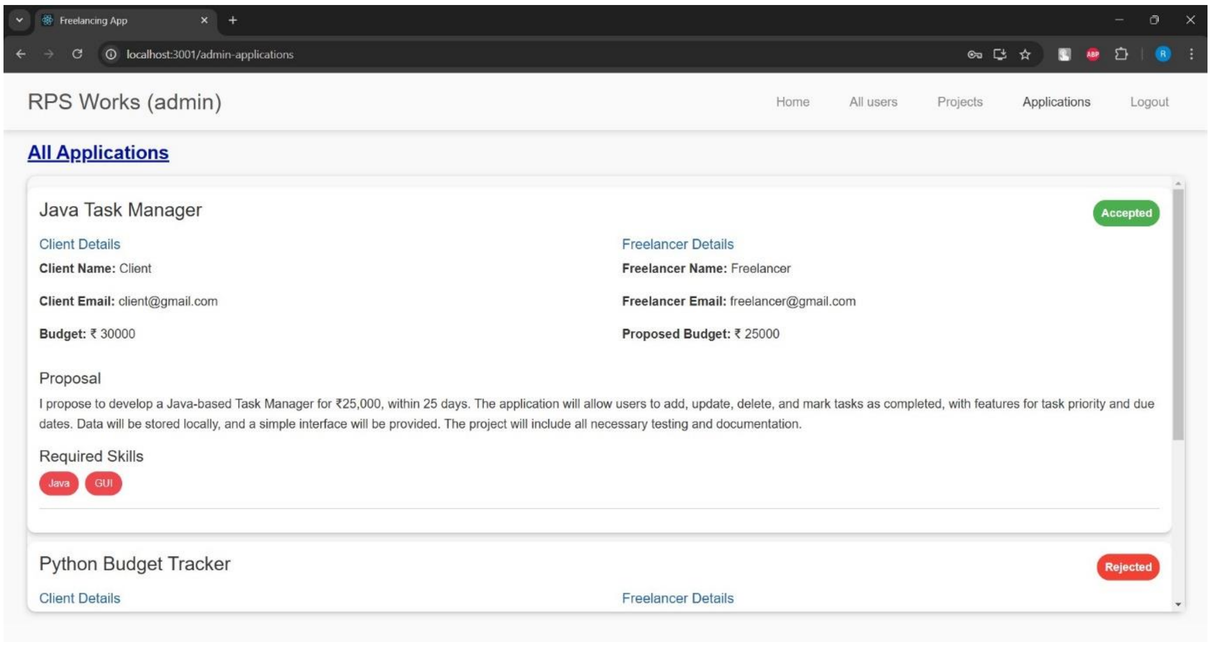
Budget: ₹25000

Java GUI

1 bids

₹ 25000.00 (avg bid)

- **Admin's All Applications page**



All Applications

Java Task Manager

Accepted

Client Details

Client Name: Client

Client Email: client@gmail.com

Budget: ₹ 30000

Freelancer Details

Freelancer Name: Freelancer

Freelancer Email: freelancer@gmail.com

Proposed Budget: ₹ 25000

Proposal

I propose to develop a Java-based Task Manager for ₹25,000, within 25 days. The application will allow users to add, update, delete, and mark tasks as completed, with features for task priority and due dates. Data will be stored locally, and a simple interface will be provided. The project will include all necessary testing and documentation.

Required Skills

Java GUI

Python Budget Tracker

Rejected

Client Details

Freelancer Details

The demo of the app is available at:

https://drive.google.com/file/d/1kxbnWgPMxBk69sH4b41MZr0WC5yWXOFF/view?usp=sharing

12.KNOWN ISSUES

- **Real-time Chat Lag**: Occasionally, there may be a slight delay in chat messages due to WebSocket response time under high loads.
- **Rendering Issue on Large Data Sets**: Pages with extensive data (such as numerous bids or projects) can cause slow rendering.

13.FUTURE ENHANCEMENTS

- **Notifications:** Implement notification systems to alert users of new bids, project approvals, and messages.
- **Payment Integration**: Add a secure payment system to facilitate direct payments between clients and freelancers.
- **Enhanced Project Filters**: Allow freelancers to filter available projects by criteria such as budget, timeline to find suitable opportunities faster.

14.CONCLUSION

RPS Works is a robust freelancing platform built with the MERN stack, offering features like user registration, project posting, bidding, real-time chat, and admin oversight. The project effectively demonstrates how MongoDB, Express, React, and Node.js can be integrated to create a seamless, user-friendly experience for clients and freelancers. Our team collaborated to address challenges in real-time communication, role-specific functionality, and security, creating a functional prototype with room for future enhancements like payment integration. Overall, RPS Works exemplifies a full-stack application that supports efficient project management in the gig economy.