Thrive: Freelancing Application

Welcome to Thrive, a freelancing platform that reimagines client-freelancer collaboration.Thrive is a comprehensive platform designed to connect freelancers with clients for project bidding, management, and communication. The system facilitates seamless project submissions, real-time chat, and efficient freelancer-client interactions. Our platform prioritizes simplicity and transparency, enabling clients to explore detailed freelancer profiles and review past work, ensuring informed decision-making.

Objectives:

- To provide a user-friendly interface for freelancers and clients.
- To streamline project bidding and management.
- To enable real-time communication between users.

Key Features

1. **User Authentication:**
    - Secure login and registration for clients and freelancers.
    - Role-based access control.

2. **Dashboard:**
    - Personalized dashboards with an overview of projects, bids, and messages.

3. **Project Management:**
    - Project creation, bidding, and selection.
    - Seamless project submission and feedback.

4. **Real-time Chat:**
    - Instant messaging between clients and freelancers.
    - Real-time notifications for project updates and messages.

5. **Profile Management:**
    - Editable profiles showcasing skills and past work.

6. Analytics:

- Charts and graphs for project and bidding statistics.
- Admin dashboard for monitoring platform activity.

Technologies Used

Frontend:

- **React.js:**

  - A JavaScript library for building dynamic user interfaces.

  - Utilizes a component-based architecture for reusable UI components and efficient updates.

  - Ensures a smooth and responsive user experience

- **Bootstrap:**

  - A popular CSS framework for responsive design and styling.

  - Provides pre-styled components and utilities to create a consistent look and feel.

  - Ensures the platform is mobile-friendly and accessible across different devices.

Backend:

- **Node.js:**

  - A JavaScript runtime for server-side development.

  - Features a non-blocking, event-driven architecture ideal for scalable

applications.

- Allows for building fast and efficient server-side applications.

- **Express.js:**

  - A web application framework for Node.js.

  - Simplifies setting up middleware, handling routing, and implementing server-side logic.

  - Provides robust features for web and mobile applications.

- **Mongoose:**

  - An ODM (Object Data Modeling) library for MongoDB and Node.js.

  - Facilitates schema-based modeling and validation of data.

  - Manages data interactions with MongoDB efficiently.

Database:

- **MongoDB:**

  - Stores data in a flexible, JSON-like format.

  - Handles large datasets efficiently, ensuring scalability and high performance.

  - Facilitates quick access and storage of user and project data.

Real-time Communication:

- **Socket.io:**

  - Enables real-time, bidirectional communication between web clients and

servers.

- Facilitates instant messaging and notifications within the platform.

- Ensures a seamless communication experience for users.

Authentication:

- **JWT (JSON Web Tokens):**

  - Provides secure user authentication.

  - Tokens are stateless and can be easily verified.

  - Enhances security by ensuring that user data is protected during communication.

API and State Management:

- **Axios**:

  - Promise-based HTTP client for making API requests.

  - Simplifies communication with the backend server.

  - Handles asynchronous operations for login, registration, and data fetching.

- **React Context API:**

  - Provides a way to share values between components without passing props.

  - Manages global state such as user authentication and real-time socket connection.

  - Simplifies state management and makes it easier to pass data through the component tree.

Server-Side Handling:

- **Body-Parser:**

  - Middleware for parsing incoming request bodies in various formats (JSON, URL-encoded).

  - Ensures smooth handling of request data in Express applications.

- **Cors:**

  - Middleware for enabling Cross-Origin Resource Sharing.

  - Allows secure cross-origin requests by configuring allowed origins and methods.

- **Bcrypt:**

  - A library for hashing passwords.

  - Provides secure encryption and validation of user passwords to protect sensitive data.

- **HTTP Server:**

  - Utilized with Node.js to create and manage HTTP servers.

  - Handles incoming requests and integrates with various middleware and routes.

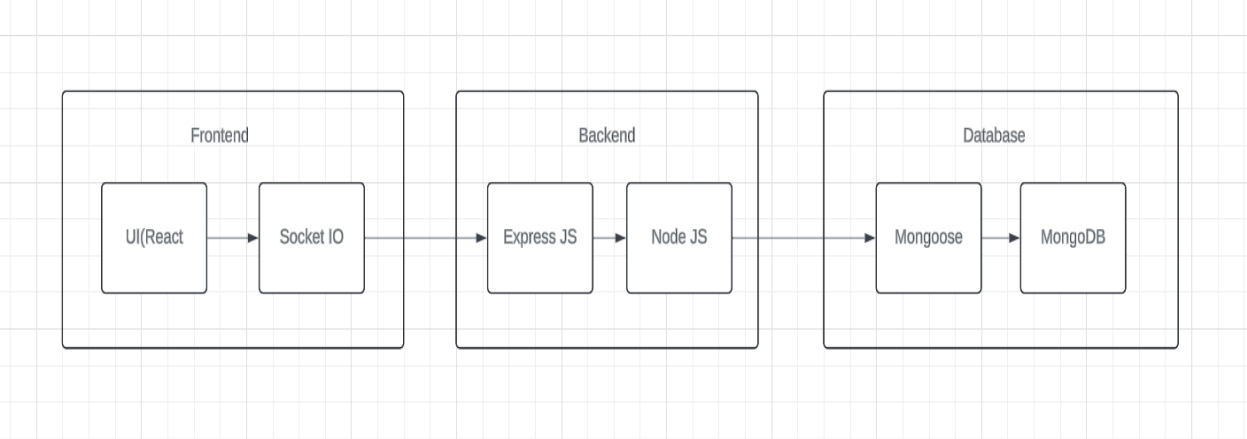
- **SocketHandler:**

  - Custom module for managing socket events and interactions

  - Encapsulates socket event handling logic to keep code organized and modular.

Architecture

Diagram:



Description:

- **Frontend:** The React.js frontend communicates with the backend API for data retrieval and submission.
- **Backend:** The Node.js server handles API requests, processes data, and communicates with the MongoDB database.
- **Database:** MongoDB stores user, project, and bid information.
- **Real-time Communication:** Socket.io manages real-time messaging between users.

Usage

User Registration:

- Freelancers and clients can register using the sign-up form.
- Upon registration, users can log in using their credentials.

Dashboard Navigation:

- The dashboard displays an overview of projects, bids.
- Users can navigate to different sections using the navbar.

Project Management:

- Clients can create new projects and view bids from freelancers.
- Freelancers can view project details and submit bids.

Real-time Chat:

- Users can initiate a chat by clicking on a project or freelancer/client profile.
- The chat feature appears at the bottom right of the page.
- They can chat only when the client approves their proposal

Admin

Roles and Responsibilities:

- **Manage Users:** Admins have the ability to add, and remove users (freelancers and clients).
- **Monitor Projects:** Oversee all active and completed projects.
- **Handle Bidding:** Review and approve/disapprove project bids.
- **Generate Reports:** Create and view various reports on system usage and project status.
- **System Maintenance:** Ensure the platform runs smoothly, perform updates.

Available Pages:

- **Dashboard:** Overview of system status, active users, and projects.
- **User Management:** Page to manage freelancers and clients.
- **Project Management:** Overview of all projects.
- **Bidding Management:** Review and manage project bids.
- **Applications** : Review all projects

Features:

- **User Role Management:** Assign roles and permissions to users.
- **Analytics:** View detailed analytics on system usage and project performance.

Freelancer

Roles and Responsibilities:

- **Profile Management:** Update personal and professional information.
- **Browse Projects:** Search for projects to bid on.
- **Submit Bids:** Apply for projects by submitting bids.
- **Work on Projects:** Manage tasks, submit work, and communicate with clients.

Available Pages:

- **Dashboard:** Overview of active bids, ongoing projects, and earnings. Update personal information and skills.
- **Project :** Browse available projects to bid on.
- **Bid project:** Track the status of submitted bids.
- **My Projects:** Manage tasks and communicate with clients on active projects.
- **Applications** : Review all projects

Features:

- **Bid Submission:** Apply for projects with detailed proposals.
- **Task Management:** Organize and track project tasks.
- **Messaging System:** Communicate with clients within the platform.
- **File Uploads:** Upload and manage project files.

Client

Roles and Responsibilities:

- **Project Creation:** Create new projects with detailed descriptions and requirements.
- **Review Bids:** Review and select bids from freelancers.
- **Manage Projects:** Oversee project progress and communicate with freelancers.
- **Approve Work:** Review and approve completed work.

Available Pages:

- **Dashboard:** Overview of active projects and pending approvals.
- **New Project:** Form to create new projects.
- **Applications** : Review all projects

Features:

- **Project Posting:** Create and manage project listings.
- **Bid Review:** Evaluate bids based on freelancer profiles and proposals.
- **Communication Tools:** Built-in messaging system to communicate with freelancers.
- **Approval System:** Approve or request revisions on submitted work.