Project Proposal:
RIGELNI A Freelance Platform.

1. Group 02

- **BOUSSEKINE Mohamed Ismail** - A1
- **FERKIOUI Akram** - A1
- **AMEDJKOUH Darine** - A1
- **HAMMOUTI Walid** - A2
- **BENTALEB Lisa** - A3

2. Overview

RIGELNI is a modern freelance platform that connects skilled professionals with clients looking for services. The platform provides a secure and intuitive environment where users can both **buy and sell** services, making it a flexible solution for freelancers and businesses alike. With robust features such as **secure payments, real-time chat, and an easy-to-use dashboard**, RIGELNI aims to streamline freelance transactions and project management.

Platform Availability:

- **Website** – Desktop-friendly platform for easy access.
- **Mobile App** – Optimized experience for users on the go.

3. Objectives

Our key objectives include:

1. **Dual Role Functionality –** Allow users to switch between being a **buyer (client)** and a **seller (freelancer)** within the same account.
2. **Multi-Platform Access –** Provide both a **website and a mobile app** for user convenience.
3. **Secure and Efficient Payments –** Implement a safe transaction system with **stripe.**

4. **User-Centric Experience** – Deliver a **well-designed and seamless interface** for an engaging user experience.
5. **Integrated Communication System** – Enable smooth interactions between buyers and sellers through **real-time chat**.
6. **Robust Project Tracking** – Provide dashboards for monitoring services, transactions, and platform activity.

4. Features

User Roles & Permissions

- **Buyers (Clients):**
  - Manage projects and payments.
  - Provide feedback and ratings..
- **Sellers (Freelancers):**
  - List services and set pricing.
  - Communicate with clients and deliver work.
  - Receive payments securely.
- **Admin Panel (Team Access Only):**
  - Monitor user activities and transactions.
  - Handle disputes and customer support.
  - Manage platform analytics and security.

Core Functionalities

✔ **User Registration & Profile Management** – Sign-up, login, and profile customization.
✔ **Dual Role Mode** – Users can switch between being a buyer and a seller.
✔ **Service Categories** – Various categories to help users find the right services.
✔ **Order Management** – Track job requests, ongoing work, and completed orders.
✔ **Payment System** – Payments processed through stripe.
✔ **Real-Time Chat System** – Direct messaging for buyer-seller communication.
✔ **Review & Rating System** – Buyers can leave feedback and rating for completed projects.
✔ **Admin Dashboard** – Monitor platform activity and manage disputes.

5. Team Members & Responsibilities

- **Ismail (Project Manager & Backend Developer)**
  - Manages project execution and team coordination.
  - Develops and manages the backend logic.
  - Ensures **database** security and **API** integrations and its documentation.
- **Akram (Styling & Documentation)**
  - Designs the **visual styles** and **layout** of the website.
  - Prepares project documentation, including reports, meeting notes, and specifications.
  - Ensures design consistency across the website.
- **Walid (Mobile App Developer)**
  - Develops **the mobile application** for the platform.
  - Implements UI components and API integrations for mobile users.
  - Ensures a seamless user experience on mobile devices.
- **Darine (Frontend Developer)**
  - Builds the **frontend** structure of the website.
  - Implements interactive features and logic.
  - Optimizes website performance for a better user experience.
- **Lisa (UI/UX Designer & Styling)**
  - Designs the user interface using **Figma**.
  - Focuses on **User Interface, User experience** and Navigation flow.
  - Works with Akram to ensure styling consistency.

6. Software & Tools Used

- **Project Management :**
  - **Jira** – Task and project management tool for agile software development.
- **Development:**
  - **VS Code** – Lightweight and powerful code editor with extensive extensions.
  - **Postman** – API development, testing, and debugging tool.
  - **Apidog** – API design, testing, and collaboration platform.
  - **Excalidraw** – Open-source tool for creating hand-drawn-style diagrams and wireframes.

- **Design:**
  - **Figma** – UI/UX design and prototyping tool for collaborative interface design.
- **Version Control:**
  - **GitHub & Git** – Version control system and cloud-based repository hosting for collaboration.
- **Mobile App:**
  - **Flutter** – For cross-platform mobile development to enhance compatibility and user experience.
- **Frontend Development:**
  - **TypeScript** – A strongly typed superset of JavaScript for safer and scalable development.
  - **React.js with Next.js** – React for UI development, with Next.js for server-side rendering (SSR), static site generation (SSG), and API routes.
  - **Zustand & Redux Toolkit** – State management libraries for handling complex application state.
  - **Tailwind CSS** – Utility-first CSS framework for building responsive and modern UIs.
  - **shadcn/ui** – A component library built on Radix UI and Tailwind CSS for elegant and accessible UI components.
- **Backend Development:**
  - **Node.js (With TypeScript)** – JavaScript runtime for scalable backend applications, with TypeScript for type safety.
  - **Express.js** – Minimal and flexible web framework for building APIs and web applications.
  - **PostgreSQL** – Relational database management system known for performance and reliability.
  - **Supabase** – Open-source Firebase alternative providing authentication, database, and storage services.
  - **Stripe** – Online payment processing platform for handling transactions, subscriptions, and financial services.
  - **Jest** – JavaScript testing framework for writing unit and integration tests in Node.js.
- **Hosting And Deployment:**
  - **Vercel** – Cloud platform for frontend frameworks, enabling fast deployment and global scalability.

- **Namecheap** – Domain registration and hosting service provider.

7. Project Timeline (15-02-2025 to 15-04-2025)

Phase 1: Planning & Design (Week 1-2)

✔ Define requirements & project scope.
✔ UI/UX design & wireframing in Figma.
✔ Backend architecture & database setup.

Phase 2: Development (Week 3-6)

✔ Implement authentication & user management.
✔ Develop frontend & backend functionalities.
✔ Integrate messaging & payment systems.

Phase 3: Testing & Deployment (Week 7-8)

✔ Debug & test platform functionalities.
✔ Optimize performance & security.
✔ Deploy project on **Vercel** & finalize documentation.

8. Challenges & Solutions

1. Scalability Issues

- **Problem:** Increased users may lead to slow performance.
- **Solution:** Optimize database queries and use **Supabase** for efficient data management.

2. Payment Security & Fraud Prevention

- **Problem:** Risk of fraudulent transactions.

- **Solution:** Implement **escrow payments** and identity verification measures.

3. Ensuring a Smooth User Experience

- **Problem:** Users may find it difficult to navigate the platform.
- **Solution:** Conduct **user testing** and improve UI/UX based on feedback.

4. Cross-Platform Compatibility (Web & Mobile)

- **Problem:** Different devices may display inconsistent UI elements.
- **Solution:** Use **responsive design** principles and test extensively on multiple devices.

5. Real-Time Chat Optimization

- **Problem:** Message delays or failures could affect communication.
- **Solution:** Use **WebSockets** for seamless real-time messaging.

6. API Hosting Issues on Vercel

- **Problem:** Vercel's serverless functions have limitations like request timeouts, cold starts, and lack of persistent connections.
- **Solution:** Use **Railway, Fly.io, or Render** to host your API instead. If using Vercel, try **Edge Functions** for faster responses.

7. Infinite Loop in Fetching Logic

- **Problem:** Your frontend keeps calling the API again and again, making the app slow.
- **Easy Solution:**
    - **Use `useEffect` correctly** – add the right dependencies.
    - **Use a state check** before updating to avoid unnecessary re-fetches.
    - **Use React Query** or a similar library to handle API calls automatically.

8. Single Point of Failure in API

- **Problem:** If one part of your API crashes, everything stops working.
- **Easy Solution:**
    - **Use multiple servers** (e.g., deploy on **Railway + Render**).
    - **Add a backup database** so if one fails, another takes over.
    - **Use a load balancer** (like **Cloudflare or Nginx**) to distribute traffic.

9. Conclusion

RIGELNI aims to be a **leading freelance platform**, offering a **secure, user-friendly, and feature-rich platform** for both buyers and sellers. With a **well-defined project structure, skilled team, and clear objectives**, we are confident in delivering a **high-quality web and mobile solution** within the given timeframe.