# Freelancer market place for college students

**✅ 1. Clear Strategy for Division**

* **Frontend Developer (Person A):** Focuses on UI/UX, React components, page navigation, API calls.
* **Backend Developer (Person B):** Focuses on REST API, database models, business logic, authentication, and hosting backend.

📌 **Rule of thumb:** If it's **seen on screen**, it's frontend. If it's **stored or processed**, it's backend.

| **Feature** | **Frontend (Person A)** | **Backend (Person B)** |
| --- | --- | --- |
| **Authentication** | Forms for login/register, error handling, routing to dashboards | JWT Auth API, password hashing, role-based access |
| **User Dashboard** | Show user info, active jobs, navigation UI | Fetch user data from MongoDB via /api/user/:id |
| **Post Project** | Form to enter title, description, budget, skills | Save project via POST /api/projects, validate input |
| **View Projects** | Grid/list of projects, filters by skill/budget | API: GET /api/projects, optional filters |
| **Project Details** | Show project info, button to apply | GET project by ID /api/projects/:id |
| **Submit Proposal** | Form (cover letter, price, days), submit button | POST /api/proposals, link to user/project |
| **View Proposals (Client)** | List proposals for a project | GET /api/projects/:id/proposals |
| **Hire Freelancer** | Click to hire → show modal → confirm | PUT /api/projects/:id/hire |
| **Chat System** | Chat UI, message input | Socket.IO, message storage API |
| **Review & Rating** | Review form after completion | POST /api/review, link to project & freelancer |
| **Notifications** | Show new proposals, hire confirmation | Optional: Send from backend via API or socket |
| **Profile Page** | View/edit profile, list of past work | GET & PUT /api/user/:id, file upload via Multer |

GET /api/projects # List projects

POST /api/projects # Create project

GET /api/projects/:id # View one project

POST /api/proposals # Submit a proposal

GET /api/user/:id # Get user profile

POST /api/auth/register # Register user

POST /api/auth/login # Login

PUT /api/projects/:id/hire # Hire freelancer

## 📐 4. ****Component Mapping: Who Builds What****

### 🧑‍🎨 Person A – Frontend (React + Tailwind)

* /login → LoginPage.jsx
* /register → RegisterPage.jsx
* /dashboard → Dashboard.jsx
* /project/:id → ProjectDetails.jsx
* /post-project → PostProjectForm.jsx
* /chat → ChatBox.jsx
* /profile/:id → ProfilePage.jsx
* /proposals → ProposalsList.jsx
* Setup React Router, State Management (Context API or Redux)
* Use axios to talk to backend APIs

### 🛠️ Person B – Backend (Node.js + Express + MongoDB)

* models/User.js, Project.js, Proposal.js
* routes/auth.js, routes/project.js, routes/proposal.js
* controllers/ for each route logic
* middleware/auth.js for protecting routes
* utils/socket.js for chat
* Connect MongoDB using mongoose.connect
* Setup nodemon, .env, CORS

| **Task** | **Who Leads** | **Collaborates** |
| --- | --- | --- |
| Plan API endpoints | Backend dev | Frontend dev (needs to know endpoints) |
| Define database schema | Backend dev | Frontend dev (needs to know expected JSON) |
| Authentication flow | Backend dev | Frontend dev (design forms, call login API) |
| Project form validation | Frontend dev | Backend dev (duplicate server-side) |
| Hosting / Deployment | Deploy separately | Deploy Netlify (frontend) and Render (backend) |
| GitHub Repo Structure | Shared | Define /client and /server subfolders |

## 🔄 6. ****Example Workflow for One Feature: “Post a Project”****

### Frontend

* Create React form: title, description, budget
* On submit → axios.post("/api/projects", data)
* If success → show toast & redirect

### Backend

* Route: POST /api/projects
* Controller: Validate data, save to DB
* Return status 201 Created + project JSON

## 7. ****Tips to Work Together****

* Use **GitHub** with branches:
  + frontend/feature-name
  + backend/feature-name
* Sync daily on:
  + New APIs added
  + Schema changes
  + Testing frontend-backend connection
* Use tools:
  + [Postman](https://www.postman.com/) (backend test)
  + [Swagger](https://swagger.io/) (document APIs)
  + [Figma](https://www.figma.com/) (optional UI planning)