# Training & Freelance Platform

## Freelance Marketplace MVP: Backlog, Estimates & Sprint Plan

### 1. Project Overview

* **Sprint Length:** 2 weeks
* **Team Velocity:** 40 story points/sprint (4 interns at 50% effective time)
* **Team Roles & Owners:**
  + Heran (Frontend)
  + Helina (Backend)
  + Saron (DevOps/Backend)
  + Mahlet (Frontend)

### 2. Definition of Done & Acceptance Criteria

* **Definition of Done** (all must be met):
  1. Code committed to correct branch, peer-reviewed, passes lint/tests
  2. Manual testing completes, unit tests added & passing
  3. Acceptance criteria fully met, no high-severity bugs
  4. UI matches designs, responsive & accessible
  5. Documentation & environment setup updated
  6. Deployed to staging/demo environment
* **Acceptance Criteria Format:** Using Given/When/Then blocks per scenario.

### 3. Tech Stack & Environments

* **Frontend:** Next.js (React 18+), Tailwind CSS, shadcn/ui, Zustand, TanStack Query
* **Backend:** NestJS (Node.js), PostgreSQL, real-time chat (Socket.IO)
* **CI/CD & Testing:** GitHub Actions + Docker; Jest (backend), React Testing Library + Vitest (frontend)

4. Epics, User Stories

| **Epic** | **User Story** | **Acceptance Criteria (G/W/T)** | **Subtasks** | **Est. (pts)** |
| --- | --- | --- | --- | --- |
| **Account & Onboarding** | As a client, I can register/sign-in so that I can submit job requests to the company. | **G:** I land on the signup page**W:** I register with email & password and verify my account**T:** I’m redirected to a “Submit Job Request” form and see my name in the header. | 1. Signup/sign-in form2. Email verification flow3. Auth guard on “Submit Job” page4. Header user badge | 5 |
| **Profile Management** | As a **developer**, I can build/edit my internal profile (skills, experience) so the owner knows who’s best for each request. | **G:** I open “My Profile” in nav**W:** I update skills, experience, availability**T:** Those details show in my profile and in the admin’s “Dev Pool” view. | 1. Profile edit UI2. Skills & availability fields3. Profile API endpoints4. Admin “Dev Pool” stub view | 5 |
| **Job Submission** | As a **client**, I can submit a job request (title, description, required skills, deadline) so the owner can review it. | **G:** I click “New Job Request”**W:** I fill fields and submit**T:** My request appears in the owner’s “Incoming Jobs” list with all details and status “Pending Assignment.” | 1. “New Job” form2. Front-end validation3. Create-job API4. Owner “Incoming Jobs” component | 8 |
| **Dev Assignment** | As the **owner/admin**, I can assign an internal developer to a job so they know to work on it. | **G:** I view “Incoming Jobs” list**W:** I click “Assign” on a job, pick a dev from dropdown**T:** Job status updates to “Assigned,” and that dev sees it in their “My Assignments” list. | 1. “Assign” action in job list2. Fetch dev pool API3. Assignment API4. Update job and dev dashboards | 8 |
| **Voluntary Applications** | As a **developer**, I can see unassigned jobs and apply if I want so the owner has backup candidates. | **G:** I open “Available Jobs” list**W:** I click “Apply” on a job**T:** My name appears in that job’s “Applicants” list; owner can review and reassign if desired. | 1. “Available Jobs” component2. Apply-to-job API3. Applicants list UI4. Owner review interface stub | 8 |
| **Messaging Core** | As any **participant** (client, dev, owner), I can send/receive messages in a job thread so we can clarify requirements. | **G:** I open a job’s chat pane**W:** I type & send a message**T:** It appears immediately, persists on reload, and unread counts update correctly for each role’s inbox. | 1. Chat widget per job2. WebSocket + fallback3. Persist & fetch messages API4. Unread badge logic | 13 |
| **Modular Architecture** | As a **dev**, I need feature-flags and clean module boundaries so we can add LMS, analytics, payment gateways, etc., later. | **G:** I clone the repo and boot the app**W:** I toggle any feature-flag in config**T:** Only that module’s routes and UI exist; others are hidden or stubbed without errors. | 1. Integrate feature-flag library2. Env-driven flag config3. Scaffold modules (jobs, chat, profiles, etc.)4. Wrap modules in flags | 3 |

### 5. Additional Notes & Constraints

* **Subscriptions:** Integration with Lovable and Cursor for analytics & user insights.
* **Compliance:** GDPR-ready, data stored in PostgreSQL regionally.
* **Performance:** Page-load ≤2s on 3G; chat latency <200ms.