Software Engineer Intern - Build Clarity

Hey there!

We're excited to see what you can build! This assessment is designed to be fun and give you a chance to show off your skills. We're not expecting perfection. We want to see how you think, code, and solve problems.

The Challenge: Build Clarity

Clarity is a personal expense tracking app that helps people understand their finances better. Think of it as a smart way to track where your money goes.

Your job? Build a working version of Clarity that you'd actually want to use yourself.

What Clarity Should Do (Minimum Requirements)

The Basics:

- Add expenses and income (amount, category, date, description)
- Edit and delete transactions
- View all your transactions in a list
- Filter by category or date range
- Simple dashboard showing your spending overview
- Works on mobile and desktop

Tech Stack:

- Frontend: React with TypeScript
- Backend: Node.js or Python
- Database: PostgreSQL, MongoDB, MySQL, up to you

Must Include:

- Clean, readable code
- A README explaining how to run your app
- At least basic user authentication (sign up/login)

Ways to Stand Out (Totally Optional!)

These are just ideas, not requirements. We want to show you different ways you can impress us if you have the time and interest. Pick one or two things that excite you, or come up with your own ideas. You absolutely do not need to do all (or even any) of these.

AI Features (if you have API keys):

- Auto-categorize expenses using AI ("$45 at Starbucks" becomes Food & Dining)

Other Cool Feature ideas:

- Deploy it live (Vercel, Netlify, Railway, Render)
- Implement a category-wise expense and income list view with drag-and-drop functionality to allow users to sort and reorder items
- Ensure that when filters are applied, they are preserved across page refreshes and page navigation within the same session
- Breakdown form into multiple steps (asking income/expense amount, category, description in three steps/phases)
- Beautiful charts and graphs
- Dark mode

AI Usage Policy

AI tools (ChatGPT, Claude, Copilot, etc.) are fine to use as learning aids and for help with specific problems. However, we're evaluating YOUR skills, not the AI's.

Important: We may ask you to explain any part of your code in detail. If you used AI to generate sections, make sure you understand how they work and why you chose that approach.

Submitting AI-generated code you don't understand will be immediately obvious to us and will disqualify your application.

What We're Actually Looking For

Don't stress about doing everything perfectly. We want to see:

- **You can build stuff that works.** The core features should function properly.
- **Your code makes sense.** We should be able to read and understand it.
- **You care about details.** Little touches matter (error handling, loading states, helpful messages).
- **You can learn and adapt.** Using new tools or APIs? That's awesome!
- **You're thoughtful.** Why did you choose that approach? Tell us in your README.
- **You have fun with it.** Build something you're proud of!

Project Ideas to Get You Started

Not sure where to begin? Here's a simple roadmap:

1. **Day 1**: Set up your project, build basic CRUD (add/view/edit/delete transactions)
2. **Day 2**: Add authentication, categories, filters, basic dashboard
3. **Day 3**: Polish the UI, maybe add one standout feature, write your README, final testing

(This is just a suggestion. Work at your own pace!)

How to Submit

When you're ready, please submit this form: https://forms.gle/wZ2NvheeGX5oEuzi8.

1. **GitHub repo link** (make it public so we can check it out)
2. **A short description** (2-3 paragraphs):
    - What you built
    - What you're most proud of
    - What you'd add if you had more time
3. **Any API keys or test accounts** we need (put them in a .env.example file and mention them)
4. **Live demo URL** (optional, but a great way to stand out!)

Timeline: Try submitting it by **12 Feb, 2026**.

A Few Tips

- **Start simple, then add features.** Get the basics working first.
- **Git commits matter.** We'll look at your commit history, so commit often with clear messages.
- **Make it yours.** Have a cool idea? Go for it!
- **Don't overthink it.** Show us your current skills and potential.

We can't wait to see what you build! Good luck, and most importantly, have fun with it!