# **Your First Challenge: Build a Mini-Agent**

### **Welcome**

First of all, thanks for being here. We know this is not your usual coding test — and that’s the point. We don’t want “just another developer.” We’re looking for builders who can hustle, think creatively, and ship fast.

This challenge is designed to be tough on purpose. Not because we want to stress you out, but because we’re building [DataJar](https://datajar.co/) with a very specific kind of caliber: people who love creating solutions, move quickly, and know how to turn ambiguity into working code. If that’s you — you’ll actually have fun with this.

### **How to Succeed (Hustler’s Guide)**

This is not a “do the assignment and get a gold star” type of task. We want to see how you think and hustle. The stronger you make it, the more you stand out.

Try to win this Hackathon by Building:

* Better product = better score. Add small features that make sense.
* More ideas = better. Show us your vision beyond the MVP.
* Always explain why. Don’t just build — justify.
* Break things into tiers. Frontend, backend, DB, APIs — we want to see structure.
* Hosting is a bonus. If we can click a link and play with it, you win.
* APIs are better than hardcoding. Show us you think like a system builder.
* Speed matters. Ship something working — don’t spend 96 hours polishing pixels.
* Hustler energy. If you surprise us with something creative and scrappy, that’s a huge plus.

### **The Mission**

You have 96 hours to design and ship a **mini agent** that:

* Does **coding, testing, or simple data analysis** (pick one), or come up with any other
* Exposes a **chat interface** for users to interact with it.
* Uses **Python + LLMs** for the agent logic.
* Has a minimal **frontend in Next.js** that connects to it. Use Vibe Coding Tools
* Stores basic interactions (prompts/responses) in **Postgres**.

Use any open-source tools, vibe coding assistants, or frameworks that help you move fast. The goal isn’t to over-engineer — it’s to prove you can make something useful and scrappy that works.

### **Deliverables**

* A short Document (max 2 page) describing (Feel Free to use Chatgpt to create it, yet make sure you make it with your own tone):
  + The idea you chose and why.
  + Tools you used (and why).
  + Time spent and what you’d do with more time.
* A GitHub repo with:  
  + frontend/ → [Next.js](http://next.js)
  + backend/ → Python (FastAPI/Flask)
  + db/ → Postgres schema/migrations
* A short demo video (2–3 mins) showing how to use it.

### **What We’ll Evaluate**

* Creativity → Did you just build a boring Q&A bot, or something with personality?
* Time management → Did you scope it well for 48 hours, or try to boil the ocean?
* Stack familiarity → Do you know your way around Next.js, Python, Postgres?
* Tool usage → How did you use AI/vibe coding tools to speed yourself up?
* LLM integration → Even if simple, does the chat agent loop actually work?

If you’re the kind of person who hacks on weekends for fun, breaks things to learn faster, and believes done is better than perfect — this challenge is for you.

### **Some Agent Ideas for inspirations**

1. **Log File Debugger**
   1. Upload a log file (e.g., server logs) → agent highlights errors, warnings, and potential fixes.  
      Bonus: agent suggests regex filters to extract important parts.
   2. Why it’s good: Simple data parsing + LLM reasoning + Postgres storage.
2. **Markdown-to-Slides Agent**
   1. Paste a markdown doc → agent generates a minimal slide deck (exportable to HTML).
   2. Bonus: agent suggests a design theme.  
      Why it’s good: Fun, creative, touches Python libraries + LLM transformation.
3. **Test Data Generator**
   1. Give the agent a schema → it generates fake test data (JSON or CSV).
   2. Bonus: allows inserting it into Postgres.
   3. Why it’s good: Practical dev tool, tests creativity with data handling.
4. **Simple Deployment Helper**
   1. Paste a short project description → agent outputs a Dockerfile + Railway/Netlify config.
   2. Bonus: agent explains what each line does.
   3. Why it’s good: Shows stack + infra awareness, but easy to scope.
5. **Documentation Helper (RAG Agent)**
   1. Pick any **open-source package** (e.g., Pandas, FastAPI, Supabase client, etc.).
   2. Ingest part of its documentation into Postgres (or even just a local store).
   3. The agent answers user questions about the library using **retrieval-augmented generation (RAG)**.
   4. Bonus: allow follow-up questions in chat (multi-turn).