About the job

Title: AI First Engineer (Founder's Office)
Location: Remote / Bangalore
CTC: 7-12 LPA (based on experience)
Experience: 1-2 Years

The Mission

This is a Founder's Office role for a high-agency engineer who wants to own, build, and ship core, AI-driven products. You will be deployed on our most critical 0-to-1 challenges, working directly with the founders to build the backbone of our product.

Your Role

You are an engineer who lives to build and thinks like a Product Manager. You will own the full lifecycle of new features, from backend services to AI integration.

- Backend & Data Ownership: Design and evolve scalable backend services, data models, and execute database migrations with minimal downtime.
- Reliable Infrastructure: Build robust APIs, handle external integrations (with reliability, retries, error handling), and deliver scalable infrastructure (caching, rate limiting, high availability).
- Core Logic: Implement custom business logic, workflows, and background jobs for automation and document processing.
- AI Integration: Collaborate with product teams to integrate LLMs (Gemini, GPT) via SDKs/APIs to ship new AI-driven features.
- AI-Native Development: You must be a "builder." We expect you to build RAG pipelines, fine-tune models, or build AI agents to solve real business problems.

Who You Are

- A Self-Starter: You find the biggest problem and start solving it without a task list.
- Proven (1-2 Years Exp): You have 1-2 years of experience in a startup environment, but more importantly, you have proof of work (GitHub/Portfolio).
- High-Velocity: You have a track record of shipping significant features in short timeframes.
- A Product Thinker: You understand the why behind the what and can own a feature from spec to launch.
- Tech Stack: Experience with AWS/GCP, integrating LLM APIs.
- Bonus: Experience with Flutter or MERN stack is a major plus.

Non-Negotiable Requirement

- You must have experience building with AI-native tools. Proficiency in coding using AI-assisted environments like Cursor or directly with Claude is a mandatory requirement.

To Apply

Send us your resume and a one-paragraph summary of the most impressive thing you've built or achieved in the last year. This summary is mandatory.

CoinedOne Founder’s Office: AI Engineering Challenge

"The Anti-Calculator"

Role: AI First Engineer (Founder’s Office)
Time Limit: 24 Hours

---

1. The Context

The UAE mortgage market is unique and intimidating.

- **The User:** Typically an Expat who is new to the country. They are afraid of hidden fees, fluctuating interest rates, and being "locked in".
- **The Current Solution:** "Calculators" that ask for Price, Down Payment, and Rate to spit out a monthly payment (EMI).
- **The Failure:** This is mathematically correct but emotionally useless. It doesn't tell the user *if* they can afford it, what the *hidden costs* are, or if they should just keep renting.

Your Mission: Build an AI Agent that acts like a "Smart Friend," not a calculator. It needs to guide the user through the financial maze using natural conversation.

2. Domain Cheat Sheet (The "Logic")

Don't waste time researching UAE Mortgage laws. Use these primitives to build your logic.

A. The Hard Constraints (Rules for your Code) If your AI hallucinates these numbers, it fails. These must be handled via **Function Calling/Tools**:

1. **Maximum LTV (Loan to Value):** Expats can only borrow up to **80%** of the property value (meaning a 20% down payment is mandatory).
2. **Upfront Costs (The Hidden Killer):** Buying a house costs ~**7%** on top of the price (4% Transfer Fee + 2% Agency Fee + Misc). *Your AI should warn users about this.*
3. **Interest Rates:** Assume a standard market rate of **4.5%** for your calculations.
4. **Max Tenure:** 25 Years.

B. The Math (The "Tool" You Must Build) Your AI must accurately calculate the EMI (Equated Monthly Installment).

- *Inputs:* Loan Amount (Price minus Down Payment), Interest Rate (Annual), Tenure (Years).
- *The Test:* If a user says "I want to buy a 2M AED apartment," your bot shouldn't just guess. It should calculate: 2M * 0.8 = 1.6M Loan. Then run the EMI math on 1.6M.

C. The "Buy vs Rent" Logic (Heuristics) If you choose the "Buy vs Rent" scenario, use this logic to generate the advice:

- *If User plans to stay < 3 years:* **Advise Renting** (Transaction fees kill the profit).
- *If User plans to stay > 5 years:* **Advise Buying** (Equity buildup beats rent).
- *The Calculation:* Compare (Monthly Rent) vs (Monthly Mortgage Interest + Maintenance Fees).

3. The Build

You must build a functional **Conversational Mortgage Assistant**.

Core Scenario (Pick One):

1. **Buy vs. Rent:** Help a user decide if they should stop renting and buy a home based on their finances and market trends.
2. **The Refinance Check:** Help a user decide if switching their current mortgage to a new rate is actually worth the switching costs.

Minimum Viable Requirements (Must Have):

- **Conversational Interface:** A chat UI (Web or Mobile) where the user speaks naturally.
- **Intent Recognition:** The AI must understand vague inputs (e.g., "I make 20k a month and want to buy in Marina").
- **Data Collection:** It must unobtrusively gather the required variables (Income, Down Payment, Tenure) without feeling like a robotic survey.
- **The "Math" Integration:** The AI must provide a calculated numerical recommendation.
- **Lead Capture:** The conversation must end with a compelling reason for the user to provide their contact details.

4. The Engineering Constraints (Crucial)

This is where we filter "Script Kiddies" from "AI Engineers."

Constraint A: The Hallucination Trap
LLMs are notoriously bad at arithmetic. If your bot uses a raw LLM to calculate the mortgage payment, it will be wrong, and you will fail this assignment.

- **Requirement:** You must demonstrate **Tool Use / Function Calling**. The LLM should handle the empathy and intent, but hand off the math to a deterministic function (code) to get the exact numbers.

Constraint B: Latency vs. Quality
The user shouldn't wait 10 seconds for a reply.

- **Requirement:** Optimize for speed. Streaming responses are highly preferred.

Constraint C: AI-Native Workflow
We expect you to use modern tools to move fast.

- **Requirement:** We encourage the use of Cursor, V0, Lovable, Bolt, or similar AI-assisted dev tools.

5. Evaluation Metrics

We are grading you on four specific axes (MECE):

1. Architecture & Reliability (40%)

- Did you solve the hallucination problem?
- How do you manage conversation state/history?
- How do you handle edge cases (e.g., user says "I have zero income")?

2. Product Sense (30%)

- Does the bot feel "human"? (Empathy, checking for understanding).
- Did you effectively guide the user to the conversion point?
- Is the UI/UX usable?

3. Velocity & Tooling (20%)

- How much did you achieve in 24 hours?
- Did you leverage AI coding tools effectively?

4. Code Quality (10%)

- Is the code modular? Can we easily plug in a different LLM model later?

5. Submission

Please reply back to mithun@coined.one and **CC to** nithesh@coined.one with:

1. **Live Link:** A URL where we can test the bot immediately.
2. **Repo Link:** Public GitHub repository.
3. **The "Black Box" Walkthrough (Loom or README):**
    - **Architecture:** Which LLM? What framework (LangChain, Vercel AI SDK, etc.)? Why?
    - **The Math:** Show us the specific code block handling the Tool Call for calculation.
    - **The Speed Run:** Briefly mention which AI tools (Cursor, Claude, etc.) you used to accelerate your build.

Good luck! Start building.