🧠 PHASE 1: Planning & Scoping (Day 1)

1. Define Scope (MVP Version)

- Freelancer does task
- Screenpipe captures activity
- Groq (or placeholder AI) checks for task completion
- Stellar sends micro-payout

2. Use Case

- Example: A designer is paid for every Figma wireframe completed.
- Start with a single task type for simplicity.

🛠️ PHASE 2: System Setup

3. Set Up Repos

- Create a GitHub repo (frontend + backend folders).
- Choose tech stack: e.g.,
  - **Frontend**: React + Tailwind
  - **Backend**: Node.js/Express
  - **DB**: Firebase or MongoDB (to store user sessions, verification status, etc.)

4. Integrate Screenpipe

- If not Screenpipe directly, simulate screen recording:
  - Use browser-based recording API (like MediaRecorder API)
  - Save video snippets (or logs/screenshots) periodically

⚙️ PHASE 3: Groq AI / Verification

5. Set Up Groq Integration (or Simulated AI)

- If Groq isn't publicly available or accessible, simulate it with:
    - Python backend with OpenAI or local model (e.g., detect keywords or UI changes in screenshots)
    - For now, define **"task completion" rule** (e.g., tab open for 10 mins + certain button click)

💸 PHASE 4: Stellar Payment Setup

6. Integrate Stellar

- Set up a testnet Stellar wallet
- Use [Stellar SDK for JS]()
- Logic: Once verification = true → trigger Stellar micro-transaction

🖥️ PHASE 5: Frontend Dashboard

7. Create Dashboards

- **Freelancer Dashboard:**
  - Start/Stop Task
  - Task Log
  - Payment Received
- **Client/Admin Dashboard:**
  - Task Monitor
  - Verification Logs
  - Payout History

🧪 PHASE 6: Test the Flow

8. End-to-End Test

- Simulate a task session
- Mock AI verification (simulate success/fail)
- Confirm Stellar payout happens (testnet)
- Log everything clearly for the demo