Done — I created a minimal StoryChain project, marked where to put API keys / wallet addresses, and packaged everything into a zip you can download and push to GitHub.

* The zip file: Download StoryChain MVP ZIP

Quick pointers (must-read):

* **Where to put secrets**
  + Backend OpenAI key: backend/.env → set OPENAI\_API\_KEY=sk-... (do **not** commit this file).
  + Receiver crypto address: set RECEIVER\_ADDRESS in backend/.env and replace the placeholder in frontend/app.js (or modify frontend to fetch it from the backend).
* **API key safety**
  + NEVER put OpenAI API keys in the frontend. The backend server.js reads process.env.OPENAI\_API\_KEY and uses it server-side only.
* **Local run**
  + cd backend && npm install
  + Copy backend/.env.example -> backend/.env and edit.
  + node server.js
  + Open frontend/index.html in the browser (or serve it via simple static server).
* **Deploying**
  + **Backend (Railway / Render / Heroku):** push the repo, set env vars in the dashboard (OPENAI\_API\_KEY, RECEIVER\_ADDRESS, etc.). Railway will detect Node and install dependencies.
  + **Frontend (Vercel / Netlify):** deploy the frontend/ folder as a static site. If you host frontend separately, set API\_BASE in frontend/index.html/app.js to your backend URL.
* **Monetization via crypto wallets**
  + The included Donate button uses MetaMask and client-side eth\_sendTransaction to send ETH directly to your wallet address. This is the simplest way to accept crypto without integrating custodial payment processors.
  + For selling digital goods inside mobile apps (Google Play), check Play Billing policies — they may require using in-app purchases for paid digital content.

If you want, I can:

* Convert this into a ready GitHub repo with a proper README and .github/workflows CI,
* Add deployment instructions for Vercel (frontend) + Railway (backend) with exact environment variable screenshots,
* Implement server-side endpoint to expose RECEIVER\_ADDRESS safely (so you don't hardcode it in frontend),
* Add moderation and rate-limiting, or
* Build the Stripe (web) or on-chain payment flows (server-side invoice generation or verifying txs).

Which next step would you like me to do?

Next steps I can do (pick any):

* Initialize a GitHub repo for you (I can't push to your GitHub — but I can provide the exact git commands and a GitHub Actions secret setup guide).
* Add automated deployment workflow (GitHub → Railway / Vercel) using deployment tokens (I'll show how to set secrets).
* Add moderation (OpenAI moderation endpoint) and rate-limiting to the backend.
* Replace MetaMask donate flow with a server-side verification of on-chain donations (e.g., watch for incoming transactions and grant credits).