# 🔍 BlockReceipt.ai v25 – Codebase Audit & Next Steps

## 1. Audit of Current v25 Codebase

- The repository structure appears corrupted or incomplete. Instead of `/contracts`, `/server`, `/client`, we have a series of hex-named directories and an `uploads/` folder of images only.  
- Critical code files (smart contract `.sol`, backend routes, frontend components, services) are missing from the extracted zip.  
- Tests, configurations, and environment scaffolding are absent or not recognized.  
- This prevents any meaningful review or further development.

## 2. Immediate Action Required

Please repackage your code into a standard project layout so we can proceed. The expected structure is:

/  
├── contracts/  
│ └── BlockReceiptCollection.sol  
├── scripts/  
│ ├── deploy.js  
│ └── verify.js  
├── server/  
│ ├── routes/  
│ │ ├── uploadAndMint.ts  
│ │ └── nfts.ts  
│ └── services/  
│ ├── ocrService.ts  
│ ├── tpreService.ts  
│ ├── ipfsService.ts  
│ └── metadataService.ts  
├── client/  
│ ├── components/  
│ │ ├── ReceiptUpload.jsx  
│ │ ├── ReceiptGallery.jsx  
│ │ └── NFTSelection.jsx  
│ └── hooks/  
│ └── useWalletConnect.ts  
├── data/  
│ └── nft\_pool.json  
├── tests/  
│ └── mintReceipt.test.ts  
├── package.json  
└── hardhat.config.js

## 3. Next Steps & Sprint Plan

Once the codebase is restored to the above layout, implement the following sprint:

### 3.1 Consolidation & Cleanup

- Remove any duplicate or legacy code folders.  
- Ensure single `uploadAndMint` endpoint and one Uploader/Gallery component.

### 3.2 Functional Integration

- Confirm OCR extraction via `ocrService`.  
- Wire TPRE encryption in `tpreService`.  
- Implement robust IPFS pinning in `ipfsService`.  
- Update ERC-1155 contract to accept `metadataUri`.  
- Ensure `uploadAndMint` calls each service in sequence with error handling.

### 3.3 Deployment & Testing

- Deploy contract on Polygon testnet with Hardhat.  
- Seed `data/nft\_pool.json` and confirm pool selection.  
- Test full `/upload-and-mint` flow from frontend.  
- Validate on-chain events and metadata URI via Polygonscan.

### 3.4 UX & Polish

- Ensure wallet connect is unified and enforced.  
- Add stage-by-stage toasts: Parsing, Encrypting, Pinning, Minting, Success.  
- Auto-refresh gallery after mint.  
- Final QA and recording demo.

## 4. Conclusion

Please re-upload a complete code snapshot adhering to the standard project structure above. Once in place, we can execute the Next Steps sprint to bring BlockReceipt.ai to a live, functional MVP.