# 🔐 BlockReceipt.ai – Secure Wallet Generation + TACo Encryption

## 📣 Overview

To improve UX and enable full NFT custody, we are implementing automatic hot wallet generation when a user creates an account. The wallet’s private key is encrypted using Threshold Network’s Proxy Re-Encryption (TACo) so that only the user can unlock it. NFTs will be sent to this address, which becomes their personal on-chain identity inside BlockReceipt.

## 🛠 Feature Plan

1. When a user signs up, generate a new Ethereum wallet  
2. Encrypt the private key using the user’s public key and TACo  
3. Store the encrypted key and `capsule` in your metadata store  
4. Use this address for NFT delivery on receipt uploads

## 🔐 Wallet Generation Code (Node.js)

Add to your Auth or AccountService logic:

import { Wallet } from 'ethers';  
import { encryptPrivateKeyWithTACo } from '../services/tacoService';  
  
async function createUserWallet(userId, userPublicKey) {  
 const wallet = Wallet.createRandom();  
 const privateKey = wallet.privateKey;  
 const address = wallet.address;  
  
 const encrypted = await encryptPrivateKeyWithTACo(userPublicKey, privateKey);  
  
 await db.userWallets.insert({  
 userId,  
 address,  
 capsule: encrypted.capsule,  
 ciphertext: encrypted.ciphertext,  
 policyPublicKey: encrypted.policyPublicKey  
 });  
  
 return { address };  
}

## 🔐 TACo Proxy Re-Encryption (Simplified)

Assuming you have access to a functioning Threshold client and node service:

async function encryptPrivateKeyWithTACo(publicKey, data) {  
 const { capsule, ciphertext, policyPublicKey } = await thresholdClient.encrypt({  
 recipientPublicKey: publicKey,  
 data: data  
 });  
  
 return { capsule, ciphertext, policyPublicKey };  
}

## 📥 How NFT Flow Changes

1. Upload receipt as usual  
2. Instead of asking for connected wallet, the system uses the generated address tied to the user account  
3. The NFT is purchased and delivered to this hot wallet  
4. To export or view their key, the user must request a decryption using their login credentials or passphrase

## 🧱 Future Decryption Route (Optional)

POST /api/wallet/decrypt  
  
Body:  
{  
 "userId": "abc123",  
 "recipientPrivateKey": "...",  
 "tokenId": 1234  
}  
  
Result:  
{  
 "privateKey": "0xabc..."  
}

## 📦 Suggested Files to Modify or Add

- `server/services/accountService.ts`  
- `server/services/tacoService.ts`  
- `db/userWallets` table or collection  
- `routes/wallet.ts` (optional for decryption)

## ✅ Summary Instruction to Replit

Please implement:  
1. Wallet generation at signup using `ethers.Wallet`  
2. Private key encryption using TACo via user's public key  
3. Store capsule + ciphertext in secure storage  
4. Update NFT bot logic to always send to this wallet  
5. Add a future route to request decryption