# 🛠 BlockReceipt.ai – Full Code Correction & Feature Upgrade for Replit

## 🚨 Critical Fixes and Corrections

- The current `uploadReceipt.ts` route does not enforce wallet connection.  
- The `extractReceiptData` OCR function fails to categorize line items.  
- The NFT bot exists but is not triggered after upload.  
- TACo encryption is coded but unused for receipt metadata.  
- No NFT gallery exists for users to view issued tokens.

## ✅ Fixed Upload + OCR + NFT Trigger (uploadReceipt.ts)

// Require connected wallet in request  
const userWallet = req.body.wallet;  
if (!userWallet) {  
 return res.status(400).json({ success: false, message: "No wallet address provided" });  
}  
  
// After file upload  
const parsed = await extractReceiptData(req.file.path);  
if (!parsed || !parsed.items || parsed.items.length === 0) {  
 return res.status(400).json({ success: false, message: "Could not parse line items from receipt" });  
}  
  
// Encrypt the parsed line items using TACo  
const { encryptLineItems } = require('../utils/encryptLineItems');  
const { ciphertext, capsule, policyPublicKey } = await encryptLineItems(userPublicKey, parsed);  
  
// Auto trigger NFT purchase bot  
const { purchaseAndTransferNFT } = require('../services/nftPurchaseBot');  
const nftResult = await purchaseAndTransferNFT(userWallet);  
  
// Store encrypted metadata with tokenId for unlock later  
saveEncryptedReceipt({  
 tokenId: nftResult.tokenId,  
 wallet: userWallet,  
 capsule,  
 ciphertext,  
 policyPublicKey  
});

## 🔠 Updated `extractReceiptData()`

Ensure this function returns an `items[]` array with labels + prices. Use keyword mapping for categories.

{  
 merchant: "CVS",  
 total: 12.20,  
 date: "2025-05-14",  
 items: [  
 { name: "Twix Bar", price: 1.49, category: "snack" },  
 { name: "Pantene Shampoo", price: 6.99, category: "grooming" }  
 ]  
}

## 🔐 TACo Integration Snippet

File: `server/utils/encryptLineItems.js`

const { encrypt } = require('@nucypher/taco');  
  
async function encryptLineItems(userPublicKey, receipt) {  
 const data = JSON.stringify({ merchant: receipt.merchant, items: receipt.items });  
 const { ciphertext, capsule, policyPublicKey } = await encrypt({ data, recipientPublicKey: userPublicKey });  
 return { ciphertext, capsule, policyPublicKey };  
}

## 🖼 NFT Gallery Route

File: `server/routes/gallery.ts`

router.get('/gallery/:wallet', async (req, res) => {  
 const wallet = req.params.wallet.toLowerCase();  
 const nfts = await db.getNFTsForWallet(wallet);  
 const withLockStates = nfts.map(nft => ({  
 ...nft,  
 metadataLocked: true // frontend decides unlock display  
 }));  
 res.json({ success: true, nfts: withLockStates });  
});

## 📦 Files to Review or Add

- `server/utils/encryptLineItems.js`  
- `server/routes/gallery.ts`  
- Update `uploadReceipt.ts` with:  
 - wallet validation  
 - improved OCR output  
 - NFT bot + TACo encryption integration  
 - metadata logging

## ✅ What Replit Should Do

1. Fix OCR so it parses and categorizes receipt items.  
2. Wire up NFT bot to automatically issue a reward after receipt parsing.  
3. Encrypt itemized metadata using TACo.  
4. Display NFTs on a gallery page with locked/unlocked UI states.  
5. Refactor UI + README to reflect this new end-to-end privacy-preserving NFT reward engine.