# 🔐 BlockReceipt.ai – Unified Login & Wallet UX Flow

## 📣 Objective

The current login and signup experience is fragmented between Web3 wallets and email/password. This document outlines a new unified, intuitive user flow with robust backend logic for wallet generation, encryption with TACo, and seamless authentication.

## 🎯 Finalized User Flow

1. ✅ Onboarding starts with a single \*\*Login/Signup button\*\* in the top-right corner.  
2. ✅ User can choose:  
 - Sign in with Web3 wallet (connect MetaMask)  
 - Sign up or log in with email/password  
3. ✅ If signing up via email, user is asked: \_“Would you like us to create a free Ethereum wallet for you?”\_  
 - If yes: a wallet is generated, and private key is shown with a warning to save it  
 - If no: user can link their own wallet later via `/link-wallet`  
4. ✅ Wallet is encrypted using TACo with user’s chosen credentials  
5. ✅ All NFTs from receipts are routed to the generated wallet  
6. ✅ When the user logs in again, the system automatically fetches and loads their wallet

## 🛠 Backend Implementation

Update `authService.ts` to integrate wallet generation at signup:

async function registerUser({ email, password, wantsWallet, tacoPublicKey }) {  
 const hashedPassword = await bcrypt.hash(password, 10);  
 const user = await db.users.insert({ email, password: hashedPassword });  
  
 let walletAddress = null;  
  
 if (wantsWallet && tacoPublicKey) {  
 const wallet = Wallet.createRandom();  
 const encrypted = await tacoService.encryptPrivateKeyWithTACo(tacoPublicKey, wallet.privateKey);  
  
 await db.userWallets.insert({  
 userId: user.id,  
 address: wallet.address,  
 capsule: encrypted.capsule,  
 ciphertext: encrypted.ciphertext,  
 policyPublicKey: encrypted.policyPublicKey  
 });  
  
 walletAddress = wallet.address;  
 }  
  
 return { user, walletAddress };  
}

## 🧠 Key Detail: Link Wallet to Session Automatically

After successful login, lookup the user’s wallet and preload it to the session:

async function attachWalletToSession(userId) {  
 const wallet = await db.userWallets.findOne({ userId });  
 return wallet?.address || null;  
}

## 🧾 Web3 Login Support (Optional)

For MetaMask logins, support Ethereum-based authentication:

POST /api/auth/web3-login  
Body:  
{  
 "walletAddress": "0x...",  
 "signature": "signed login message",  
 "nonce": "retrieved login nonce"  
}

Verify the signature using ethers.js:

function verifySignature(walletAddress, signature, nonce) {  
 const msg = `Log into BlockReceipt: ${nonce}`;  
 const recovered = ethers.utils.verifyMessage(msg, signature);  
 return recovered.toLowerCase() === walletAddress.toLowerCase();  
}

## 💻 Frontend Login Page (React / Tailwind)

Create a single modal or page with toggle tabs for Email / Wallet login:

<LoginModal>  
 <Tabs>  
 <Tab label="Email Login">  
 <form onSubmit={handleEmailLogin}>  
 <input name="email" />  
 <input type="password" name="password" />  
 {isSignup && (  
 <div>  
 <label>  
 <input type="checkbox" onChange={toggleWalletOptIn} />  
 Create a wallet for me  
 </label>  
 {wantsWallet && (  
 <input name="tacoPublicKey" placeholder="Your TACo public key" />  
 )}  
 </div>  
 )}  
 <button>Submit</button>  
 </form>  
 </Tab>  
  
 <Tab label="Wallet Login">  
 <Web3ConnectButton onConnect={handleWalletLogin} />  
 </Tab>  
 </Tabs>  
</LoginModal>

## 🎉 Success State After Signup

Immediately show wallet address + backup reminder:

<div className="p-4 bg-green-100 rounded">  
 <h3>🎉 Welcome to BlockReceipt!</h3>  
 <p>Your Ethereum wallet address:</p>  
 <code>{walletAddress}</code>  
 <p>⚠️ Please copy your private key and store it securely. This wallet will be used to collect your NFT receipts.</p>  
</div>

## 📂 Files to Create or Update

- `server/services/authService.ts`  
- `server/routes/auth.ts`  
- `server/routes/wallet.ts`  
- `client/components/LoginModal.jsx`  
- `client/hooks/useWalletLogin.js`

## ✅ Final Instruction

Please:  
1. Replace all fragmented login components with a single modal or page  
2. Ensure wallet creation is user-controlled, encrypted with TACo, and visibly confirmed  
3. Automatically use the stored wallet in all receipt and NFT workflows  
4. Remove any duplicated login forms or disconnected flows