const { TonClient, abi } = require("@tonclient/core");

const { libNode } = require("@tonclient/lib-node");

TonClient.useBinaryLibrary(libNode);

const bip39 = require("bip39");

const hdkey = require("ethereumjs-wallet/hdkey");

const { toHex } = require("web3-utils");

const setupWallet = async () => {

const client = new TonClient({

network: {

endpoints: ["https://toncenter.com/api/v2/jsonRPC"]

}

});

const walletAddress = "EQA\_gft901TRFYjWatkOSpFM0bB0EJuqGst9Akz5iYSdJYbj"; // Your wallet address

const seedPhrase = "kingdom hungry number apple plug borrow flame dose broken reject roof worry gallery gaze cost mind similar stool retire nephew unable prize involve slim"; // 24-word seed phrase

// Derive keys from seed phrase

const seed = bip39.mnemonicToSeedSync(seedPhrase);

const hdWallet = hdkey.fromMasterSeed(seed);

const wallet = hdWallet.derivePath(`m/44'/60'/0'/0/0`).getWallet();

const publicKey = toHex(wallet.getPublicKey());

const secretKey = toHex(wallet.getPrivateKey());

const callSet = {

function\_name: "setWalletType",

input: {

new\_wallet\_type: "wallet\_v3R2"

}

};

const signer = {

type: "Keys",

keys: {

public: publicKey,

secret: secretKey

}

};

try {

const { message } = await client.abi.encode\_message({

address: walletAddress,

call\_set: callSet,

signer: signer,

abi: {

type: "Contract",

value: {

"ABI version": 2,

header: ["time", "expire"],

functions: [

{

name: "setWalletType",

inputs: [

{ name: "new\_wallet\_type", type: "string" }

],

outputs: []

}

],

data: [],

events: []

}

}

});

await client.processing.send\_message({

message,

send\_events: false

});

console.log("Wallet type successfully updated to v3R2");

} catch (error) {

console.error("Error updating wallet type:", error);

}

};

setupWallet();