const Web3 = require('web3');

const solc = require('solc');

// Set up the Web3 provider

const provider = new Web3.providers.HttpProvider('https:

const web3 = new Web3(provider);

const contractSourceCode = fs.readFileSync('//[mainnet.infura.io/v3/YOUR\_PROJECT\_ID'](http://mainnet.infura.io/v3/YOUR_PROJECT_ID'));

const web3 = new Web3(provider);

// Compile the smart contract

const contractSourceCode = fs.readFileSync('TransactionManager.sol', 'utf8');

const compiledContract = solc.compile(contractSourceCode);

const contractABI = JSON.parse(compiledContract.contracts[':TransactionManager'].interface);

const contractBytecode = compiledContract.contracts[':TransactionManager'].bytecode;

const deployContract = async () => {

  const accounts = await web3.eth.getAccounts();

  const contract = new web3.eth.Contract(contractABI);

  const deployedContract = await contract.deploy({

    data: contractBytecode,

  })

    .send({

      from: accounts[0],

      gas: '// Deploy the smart contract

const deployContract = async () => {

  const accounts = await web3.eth.getAccounts();

  const contract = new web3.eth.Contract(contractABI);

  const deployedContract = await contract.deploy({

    data: contractBytecode,

  })

    .send({

      from: accounts[0],

      gas: '2000000',

    });

  console.log(`Contract deployed at address: ${deployedContract.options.address}`);

  return deployedContract;

};

const interactWithContract = async (contract) => {

  const accounts = await web3.eth.getAccounts();

  await contract.methods.deposit().send({

    from: accounts[0],

    value: web3.utils.toWei('// Interact with the smart contract

const interactWithContract = async (contract) => {

  const accounts = await web3.eth.getAccounts();

  // Deposit funds

  await contract.methods.deposit().send({

    from: accounts[0],

    value: web3.utils.toWei('1', 'ether'),

  });

  const balance = await contract.methods.getBalance().call({

    from: accounts[0],

  });

  console.log(`User balance: ${balance}`);

  await contract.methods.transfer(accounts[1], web3.utils.toWei('// Get user balance

  const balance = await contract.methods.getBalance().call({

    from: accounts[0],

  });

  console.log(`User balance: ${balance}`);

  // Transfer funds

  await contract.methods.transfer(accounts[1], web3.utils.toWei('0.5', 'ether')).send({

    from: accounts[0],

  });

  // Get user transaction history

  const transactionHistory = await contract.methods.getTransactionHistory().call({

    from: accounts[0],

  });

  console.log(`User transaction history: ${transactionHistory}`);

};

// Deploy and interact with the contract

deployContract().then((contract) => {

  interactWithContract(contract);

});