**Unit 4 – Blockchain and Decentralized Applications**

**Lab 3 Manual - Assignment**

**Assignment - Building a Mini-Banking Contract**

**Objective**

Create a contract that simulates a basic bank account. It should include functions to:

* Deposit and withdraw funds.
* Check the balance.
* Calculate simple interest.

**BankAccount.sol**

// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract BankAccount {

uint private balance;

// Deposit function: adds an amount to the balance

function deposit(uint amount) public {

balance += amount;

}

// Withdraw function: deducts an amount from the balance

function withdraw(uint amount) public {

require(amount <= balance, "Insufficient balance");

balance -= amount;

}

// View function to check balance

function getBalance() public view returns (uint) {

return balance;

}

// Pure function to calculate simple interest

function calculateInterest(uint principal, uint rate, uint time) public pure returns (uint) {

return (principal \* rate \* time) / 100;

}

}

**Explanation:**

* **deposit**: Adds funds to the account balance.
* **withdraw**: Allows withdrawal if balance is sufficient.
* **getBalance**: A view function that retrieves the current balance.
* **calculateInterest**: A pure function that calculates interest, simulating a real-world calculation.

**Testing the Contracts with Truffle**

To test your contracts, create a test file in your directory.

**Example: BankAccount.test.js**

const BankAccount = artifacts.require("BankAccount");

contract("BankAccount", (accounts) => {

let bank;

before(async () => {

bank = await BankAccount.deployed();

});

it("should deposit funds", async () => {

await bank.deposit(1000);

const balance = await bank.getBalance();

assert.equal(balance.toNumber(), 1000, "Deposit was not successful");

});

it("should withdraw funds", async () => {

await bank.withdraw(500);

const balance = await bank.getBalance();

assert.equal(balance.toNumber(), 500, "Withdrawal was not successful");

});

it("should calculate interest correctly", async () => {

const interest = await bank.calculateInterest(1000, 5, 1);

assert.equal(interest.toNumber(), 50, "Interest calculation is incorrect");

});

});

**Additional Exercises**

1. Modify the BankAccount contract to limit withdrawals to a certain daily amount.