// SPDX-License-Identifier: MIT

pragma solidity ^0.8.0;

contract BankAccount {

 // State variable to store the balance of the customer

 mapping(address => uint) private balances;

 // Event to emit when a deposit is made

 event Deposited(address indexed account, uint amount);

 // Event to emit when a withdrawal is made

 event Withdrawn(address indexed account, uint amount);

 // Function to deposit money into the customer's account

 function deposit() public payable {

 require(msg.value > 0, "Deposit amount must be greater than 0");

 // Add the deposited amount to the customer's balance

 balances[msg.sender] += msg.value;

 // Emit the deposit event

 emit Deposited(msg.sender, msg.value);

 }

 // Function to withdraw money from the customer's account

 function withdraw(uint \_amount) public {

 require(\_amount <= balances[msg.sender], "Insufficient balance");

 // Deduct the amount from the customer's balance

 balances[msg.sender] -= \_amount;

 // Transfer the amount to the customer's address

 payable(msg.sender).transfer(\_amount);

 // Emit the withdrawal event

 emit Withdrawn(msg.sender, \_amount);

 }

 // Function to check the current balance of the customer

 function getBalance() public view returns (uint) {

 return balances[msg.sender];

 }

}