```
pragma solidity >= 0.7.0;
// Write a smart contract on a test network, for Bank account of a customer for
 // following operations: Deposit money | Withdraw Money | Show balance
contract Bank{
   mapping(address => uint) public user_account;
   mapping(address => bool) public user_exist;

   function create_account() public payable returns(string memory){
      require(user_exist[msg.sender] == false, "Account Already created!");
      user_account[msg.sender] = msg.value;
      user_exist[msg.sender] = true;
      return "Account created";
   }

   function deposit(uint amount) public payable returns(string memory){
      require(user_exist[msg.sender] == true, "Account not created!");
      require(amount > 0, "Amount should be greater than 0");
      user_account[msg.sender] += amount;
      return "Amount deposisted sucessfully";
   }

   function withdraw(uint amount) public payable returns(string memory){
      require(user_exist[msg.sender] == true, "Account not created!");
      require(amount > 0, "Amount should be greater than 0");
      require(user_account[msg.sender] >= amount, "Amount is greater than money deposisted");
      user_account[msg.sender] -= amount;
      return "Amount withdrawn sucessfully";
   }

   function account_balance() public view returns(uint){
      return user_account[msg.sender];
```

```
    }

    function account_exists() public view returns(bool){
        return user_exist[msg.sender];
    }

}
```