PROJECT

ON

Bank Account System

Overview

The Bank Account System is a Python-based program designed to simulate basic banking functionalities. It allows users to create accounts, deposit and withdraw money, and check account balances. The system ensures that users cannot overdraw from their accounts and only permits valid inputs for transactions.

Features

- **Create New Accounts**: Users can create new bank accounts by providing details such as the account holder's name, account number, and initial deposit.
- **Deposit Money**: Users can deposit funds into an existing account.
- **Withdraw Money**: Users can withdraw funds from their account, with checks in place to prevent overdrawing.
- **Check Balance**: Users can check their account balance.
- **Menu Navigation**: A simple menu-based interface that allows users to choose between different banking operations.

Technologies Used

- Language: Python
- Concepts:
    - Classes and Objects
    - Functions
    - Conditionals (if-else)
    - Loops

Project Structure

1. BankAccount Class

This class represents an individual bank account and includes methods for depositing money, withdrawing money, and checking the balance.

- **Attributes**:
    - name: The account holder's name.
    - account_number: A unique identifier for the account.
    - balance: The current balance in the account.
- **Methods**:
    - __init__(self, name, account_number, balance=0): Initializes the account with the holder's name, account number, and an optional initial balance (default is 0).
    - deposit(self, amount): Adds a specified amount to the balance, if the amount is positive.
    - withdraw(self, amount): Deducts a specified amount from the balance if there are sufficient funds and the amount is valid.
    - check_balance(self): Displays the current balance of the account.

Code

```
class BankAccount:
    def __init__(self, name, account_number, balance=0):
        self.name = name
        self.account_number = account_number
        self.balance = balance
    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            print(f"{amount} deposited successfully. New balance is:
{self.balance}")
        else:
            print("Invalid deposit amount!")
    def withdraw(self, amount):
        if amount > self.balance:
            print("Insufficient funds! Withdrawal failed.")
        elif amount <= 0:
            print("Invalid withdrawal amount!")
        else:
            self.balance -= amount
            print(f"{amount} withdrawn successfully. New balance is:
{self.balance}")
    def check_balance(self):
        print(f"Account Balance for {self.name}: {self.balance}")
def create_account(accounts):
    name = input("Enter account holder's name: ")
    account_number = input("Enter a new account number: ")
```

```
    initial_balance = float(input("Enter initial deposit amount: "))
    account = BankAccount(name, account_number, initial_balance)
    accounts[account_number] = account
    print(f"Account created for {name} with balance {initial_balance}\n")
def bank_system():
    accounts = {}

    while True:
        print("\n--- Bank System Menu ---")
        print("1. Create New Account")
        print("2. Deposit Money")
        print("3. Withdraw Money")
        print("4. Check Balance")
        print("5. Exit")

        choice = input("Enter your choice: ")

        if choice == '1':
            create_account(accounts)

        elif choice == '2':
            account_number = input("Enter account number: ")
            if account_number in accounts:
                amount = float(input("Enter amount to deposit: "))
                accounts[account_number].deposit(amount)
            else:
                print("Account not found!")
```

```
        elif choice == '3':
            account_number = input("Enter account number: ")
            if account_number in accounts:
                amount = float(input("Enter amount to withdraw: "))
                accounts[account_number].withdraw(amount)
            else:
                print("Account not found!")

        elif choice == '4':
            account_number = input("Enter account number: ")
            if account_number in accounts:
                accounts[account_number].check_balance()
            else:
                print("Account not found!")

        elif choice == '5':
            print("Exiting the system.")
            break

        else:
            print("Invalid choice, please try again.")
bank_system()
```

output:-

```
--- Bank System Menu ---
1. Create New Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Enter your choice:  1
Enter account holder's name:  p.v.v.d.srcharan
Enter a new account number:  4113228556
Enter initial deposit amount:  200000
Account created for p.v.v.d.srcharan with balance 200000.0


--- Bank System Menu ---
1. Create New Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Enter your choice:  2
Enter account number:  4113228446
Account not found!
```

```
--- Bank System Menu ---
1. Create New Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Enter your choice:  3
Enter account number:  4113228336
Account not found!

--- Bank System Menu ---
1. Create New Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Enter your choice:  4
Enter account number:  41132222885
Account not found!

--- Bank System Menu ---
1. Create New Account
2. Deposit Money
3. Withdraw Money
4. Check Balance
5. Exit
Enter your choice:  5
```

Conclusion

This Bank Account System provides a simple and efficient way to simulate banking operations like creating new accounts, depositing, withdrawing, and checking balances. The project demonstrates important programming concepts such as object-oriented programming, conditionals, loops, and function handling in Python. It can be extended to include additional features like account history, transfers, and interest calculations.