🏦Python Bank Management System

Beginner → Intermediate Level Project

🚃Introduction

This project is a **console-based Bank Management System** built using **Python and Object-Oriented Programming (OOP)**. It simulates real-world banking operations such as account creation, login, deposits, withdrawals, balance checking, and transaction history management.

The goal of this project is to help learners understand:

- How OOP works in real applications
- How to store data permanently using JSON
- How to design menu-driven programs
- How to structure a Python project professionally

Concepts Used

- Python Basics
- Classes & Objects
- File Handling (JSON)
- Datetime module
- Random number generation
- Menu-driven logic

Project Structure

```
bank-system-project/
│
├── bank_system.py
├── data/
│   └── accounts.json
└── README.md
```

💈 Complete Source Code (bank_system.py)

```
import json
import os
from datetime import datetime
import random

# ----------------------------
# Helper: Load & Save Data
# ----------------------------
DATA_FILE = "data/accounts.json"

def load_data():
    if not os.path.exists("data"):
        os.makedirs("data")
    if not os.path.isfile(DATA_FILE):
        with open(DATA_FILE, "w") as f:
            json.dump({}, f)
    with open(DATA_FILE, "r") as f:
        return json.load(f)

def save_data(data):
    with open(DATA_FILE, "w") as f:
        json.dump(data, f, indent=4)

# ----------------------------
# Bank Account Class
# ----------------------------
class BankAccount:
    def __init__(self, name, pin, account_number=None, balance=0, history=None):
        self.name = name
        self.pin = pin
        self.balance = balance
        self.account_number = account_number or self.generate_acc_no()
        self.history = history or []

    @staticmethod
    def generate_acc_no():
        return str(random.randint(10000000, 99999999))

    def add_history(self, action, amount=0):
        self.history.append({
            "action": action,
            "amount": amount,
            "time": datetime.now().strftime("%Y-%m-%d %H:%M:%S")
        })
```

```
    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            self.add_history("Deposit", amount)
            print(f"{amount} deposited successfully!")
        else:
            print("Amount must be greater than zero.")

    def withdraw(self, amount):
        if amount <= 0:
            print("Amount must be greater than zero.")
        elif amount > self.balance:
            print("Insufficient balance.")
        else:
            self.balance -= amount
            self.add_history("Withdraw", amount)
            print(f"{amount} withdrawn successfully!")

    def check_balance(self):
        print(f"\nCurrent Balance: {self.balance}\n")
        self.add_history("Checked Balance")

    def show_history(self):
        print("\nTransaction History:")
        if not self.history:
            print("No transactions yet.")
        else:
            for h in self.history:
                print(f"{h['time']} — {h['action']} — {h['amount']}")
        print()

# ----------------------------
# Banking System Controller
# ----------------------------
class BankSystem:
    def __init__(self):
        self.data = load_data()

    def save_account(self, acc):
        self.data[acc.account_number] = {
            "name": acc.name,
            "pin": acc.pin,
            "balance": acc.balance,
            "history": acc.history
        }
        save_data(self.data)
```

```
    def create_account(self):
        print("\nCreate New Account")
        name = input("Enter your name: ")
        pin = input("Set a 4-digit PIN: ")

        acc = BankAccount(name, pin)
        self.save_account(acc)

        print(f"\nAccount created successfully!")
        print(f"Your Account Number: {acc.account_number}")

    def login(self):
        print("\nLogin to Your Account")
        acc_no = input("Enter Account Number: ")
        pin = input("Enter PIN: ")

        if acc_no in self.data and self.data[acc_no]["pin"] == pin:
            acc_info = self.data[acc_no]
            print("\nLogin Successful!")
            return BankAccount(
                acc_info["name"],
                acc_info["pin"],
                acc_no,
                acc_info["balance"],
                acc_info["history"]
            )
        else:
            print("Invalid account number or PIN.")
            return None

# ------------------------------
# Main Application Menu
# ------------------------------
def main():
    bank = BankSystem()

    while True:
        print("\nBANK SYSTEM MENU")
        print("1. Create Account")
        print("2. Login")
        print("3. Exit")

        choice = input("Enter choice: ")

        if choice == "1":
            bank.create_account()

        elif choice == "2":
```

```
            acc = bank.login()
            if acc:
                while True:
                    print("\nACCOUNT MENU")
                    print("1. Deposit")
                    print("2. Withdraw")
                    print("3. Check Balance")
                    print("4. Transaction History")
                    print("5. Logout")

                    c = input("Enter choice: ")

                    if c == "1":
                        amt = float(input("Amount to deposit: "))
                        acc.deposit(amt)
                        bank.save_account(acc)

                    elif c == "2":
                        amt = float(input("Amount to withdraw: "))
                        acc.withdraw(amt)
                        bank.save_account(acc)

                    elif c == "3":
                        acc.check_balance()
                        bank.save_account(acc)

                    elif c == "4":
                        acc.show_history()

                    elif c == "5":
                        print("\nLogged out successfully!")
                        break

                    else:
                        print("Invalid choice.")

        elif choice == "3":
            print("\nThank you for using Bank System!")
            break

        else:
            print("Invalid choice.")

if __name__ == "__main__":
    main()
```

README.md (Final Version)

```
# 🏦 Python Bank System

A beginner to intermediate level **Bank Management System** built using Python
and Object-Oriented Programming.

##    Features
- Account creation
- Secure login using Account Number & PIN
- Deposit and Withdraw money
- Balance inquiry
- Transaction history tracking
- Persistent data storage using JSON

##    Technologies Used
- Python
- JSON
- OOP Concepts
- Datetime Module

##    Project Structure
```

bank-system-project/ ├── bank_system.py ├── data/ │ └── accounts.json └── README.md

```
## ▶️ How to Run
```bash
python bank_system.py
```

Follow @mrthanks111
```