**APPENDIX:1 Assignment SUBMISSION COVER Sheet**

**Student Id: cse24-111**

**Student name: Garebantsi thuso**

**STUDENT EMAIL:** [**CSE24-111@THUTO.BAC.AC.BW**](mailto:CSE24-111@THUTO.BAC.AC.BW)

**COHORT : CSE 1 GROUP 4**

**ASSIGNMENT TITLE: object oriented analysis and design with java**

**DATE OF SUBMISSION: 19 09 2025**

**PROGRAMME OF STUDY: BSC COMPUTER SYSTEMS ENGINEERING**

**YEAR OF STUDY: YEAR 2**

**INTELLECTUAL PROPERTY STSTEMENT**

**BY CHECKING THE BOX BELOW, I CERTIFY THAT ASSSIGNMENT IS MY OWNWORK AND IS FREE FROM PLAGIARISM. I UNDERSTAND THAT THE ASSIGNMENT MAY BE CHECKED FOR PLAGIARISM BY ELECTRONIC OR OTHER MEANS AND MAY BE TRANSFERRED AND STORED IN A DATABASE FOR THE PURPOSES OF DATA-MATCHING TO HELP DETECT PLAGIARISM. THE ASSIGNMENT HAS NOT PREVIOUSLY BEEN SUBMITTED FOR ASSESSMENT IN ANY OTHER UNIT OR ANY OTHER INSTITUTION. I HAVE READ AND UNDERSTOOD THE BOTSWANA ACCOUNTANCY COLLEGE PLAGIARISM GUIDELINES POLICY.**

Group 1, Grouped object **SIGNATURE g. tHUSO**

**DATE: 19 09 2025**

Part A: System Documentation

**1. Requirements Elicitation**

* 1. **Functional Requirements**

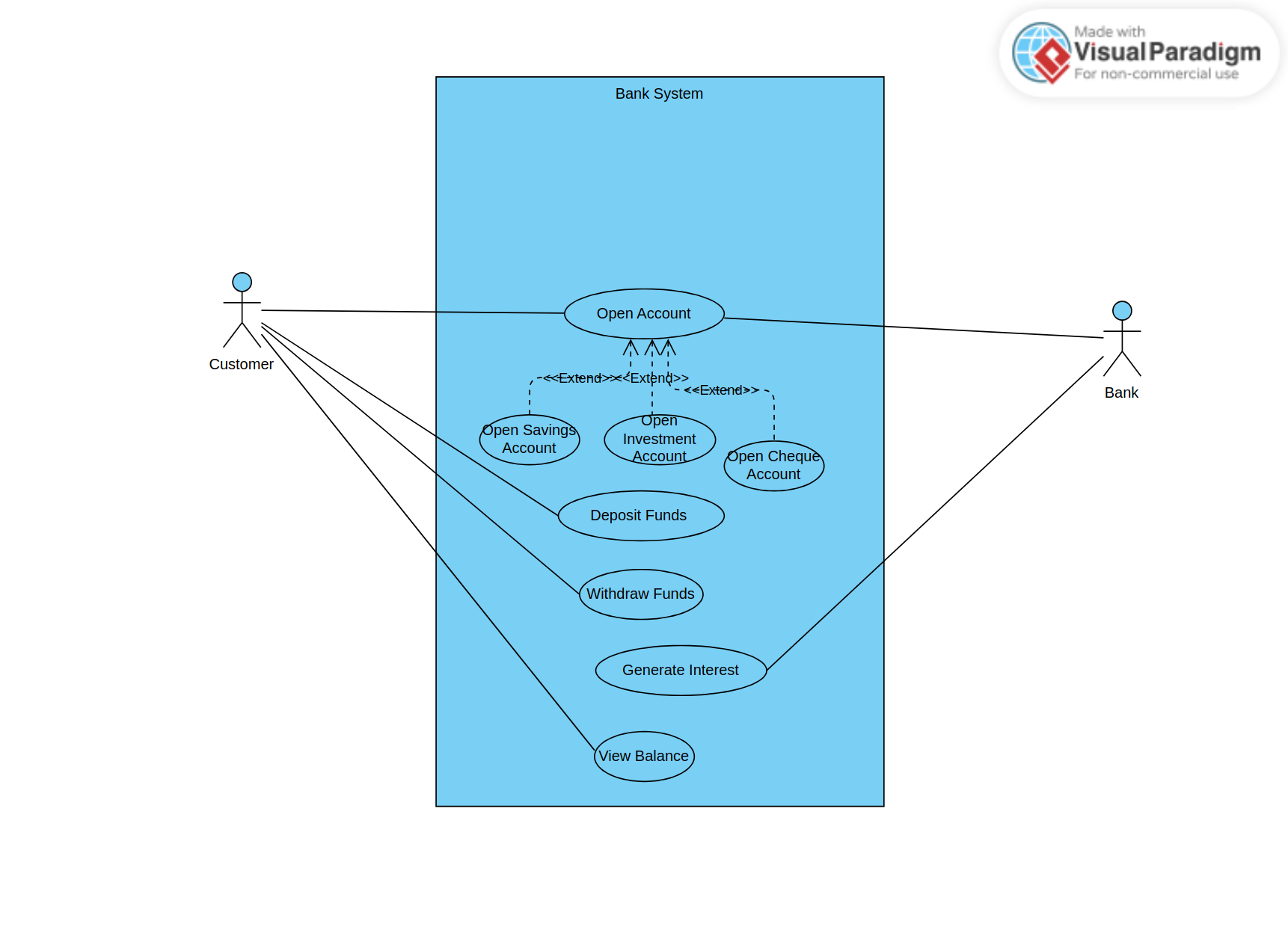
|  |  |
| --- | --- |
|  | The system must allow customers to open accounts (Savings, Investment, Cheque). |
|  | Each account must be linked to a customer. |
|  | The system must allow a customer to hold multiple accounts. |
|  | Savings accounts must allow deposits but **not** withdrawals. |
|  | Investment accounts must allow deposits and withdrawals. |
|  | Investment accounts must require a minimum opening deposit of BWP500.00. |
|  | Cheque accounts must allow deposits and withdrawals. |
|  | Cheque accounts must require employment details for account creation. |
|  | The system must calculate and apply monthly interest: |
|  | - 5% for Investment accounts |
|  | - 0.05% for Savings accounts |
|  | The system must allow deposits into any account held by the customer. |
|  | The system must store and retrieve customer and account data securely. |
|  | The system must validate account type before performing transactions. |
|  | The system must support login and authentication for customers. |
|  | The system must support object-oriented design principles. |
|  | - Inheritance, Polymorphism, Abstraction, Interfaces, Overriding, Overloading |

* 1. **Non-Functional Requirements**:

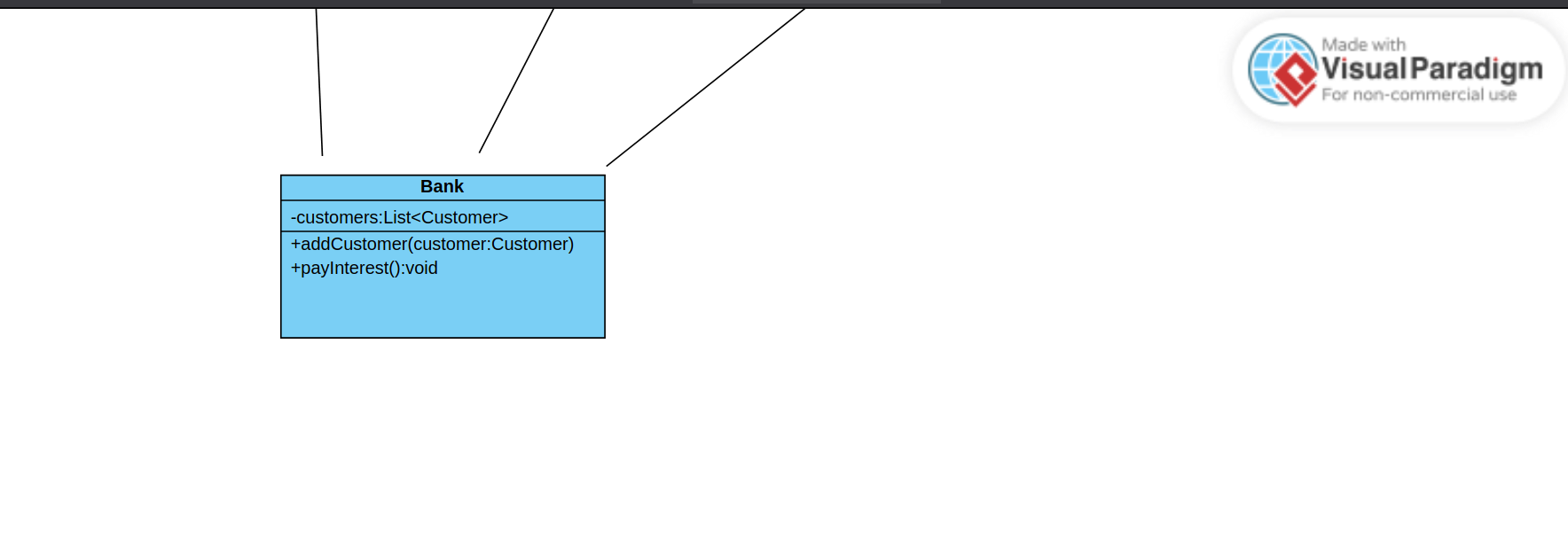
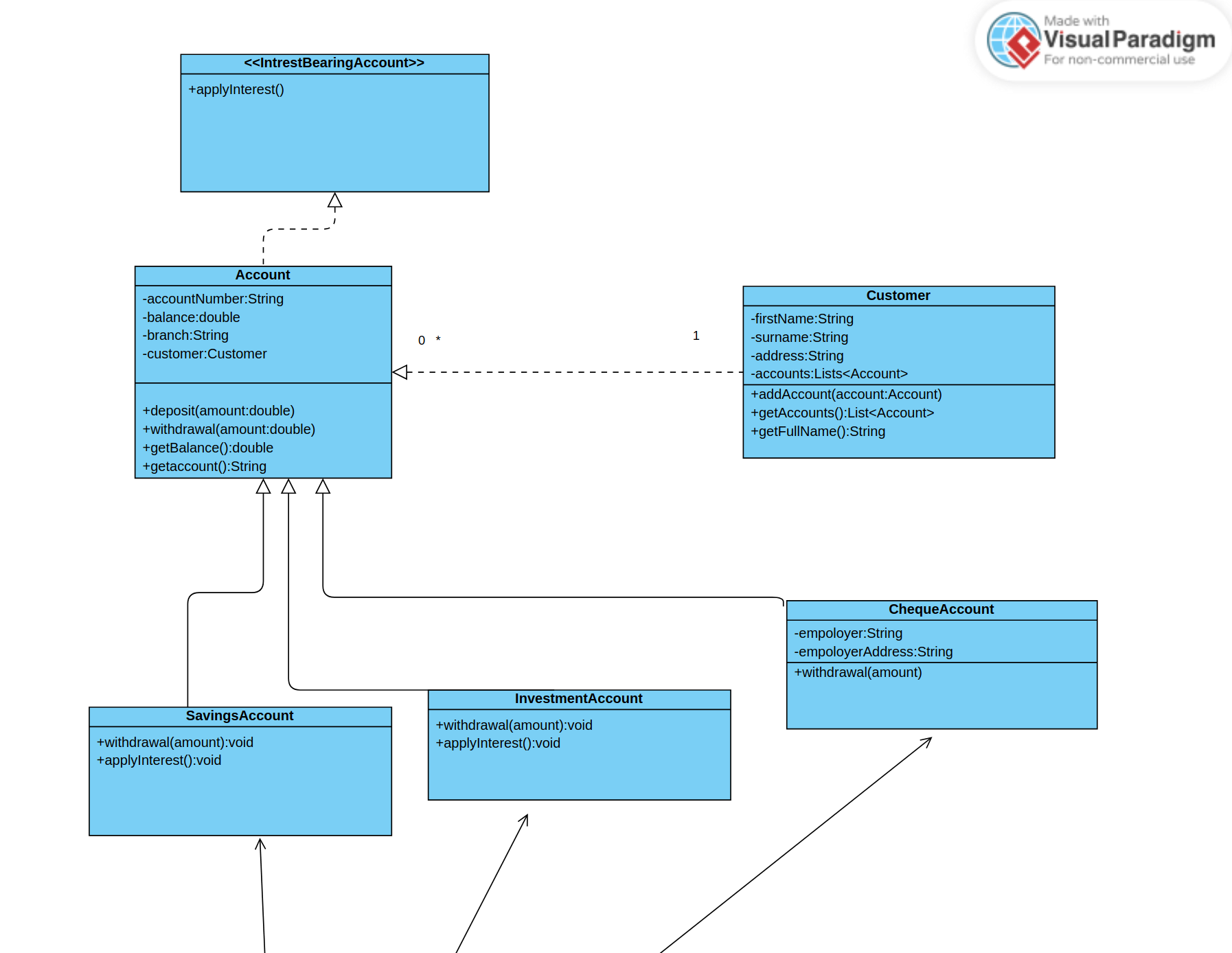
|  |  |
| --- | --- |
| Performance | The system should process transactions (deposit, interest) within 2 seconds. |
| Scalability | The system should support thousands of customers and accounts. |
| Security | Customer data and account information must be encrypted and securely stored. |
| Usability | The interface should be intuitive and easy to navigate for all users. |
| Maintainability | The system should be modular and follow OOP principles for easy updates. |
| Reliability | The system should ensure accurate interest calculation and transaction logging. |
| Availability | The system should be available 99.9% of the time. |
| Compatibility | The system should run on both desktop and mobile platforms. |
| Auditability | All transactions should be logged for future auditing and reporting. |

1. **Structural UML Modelling**

**2.1. System Use Case Diagram**

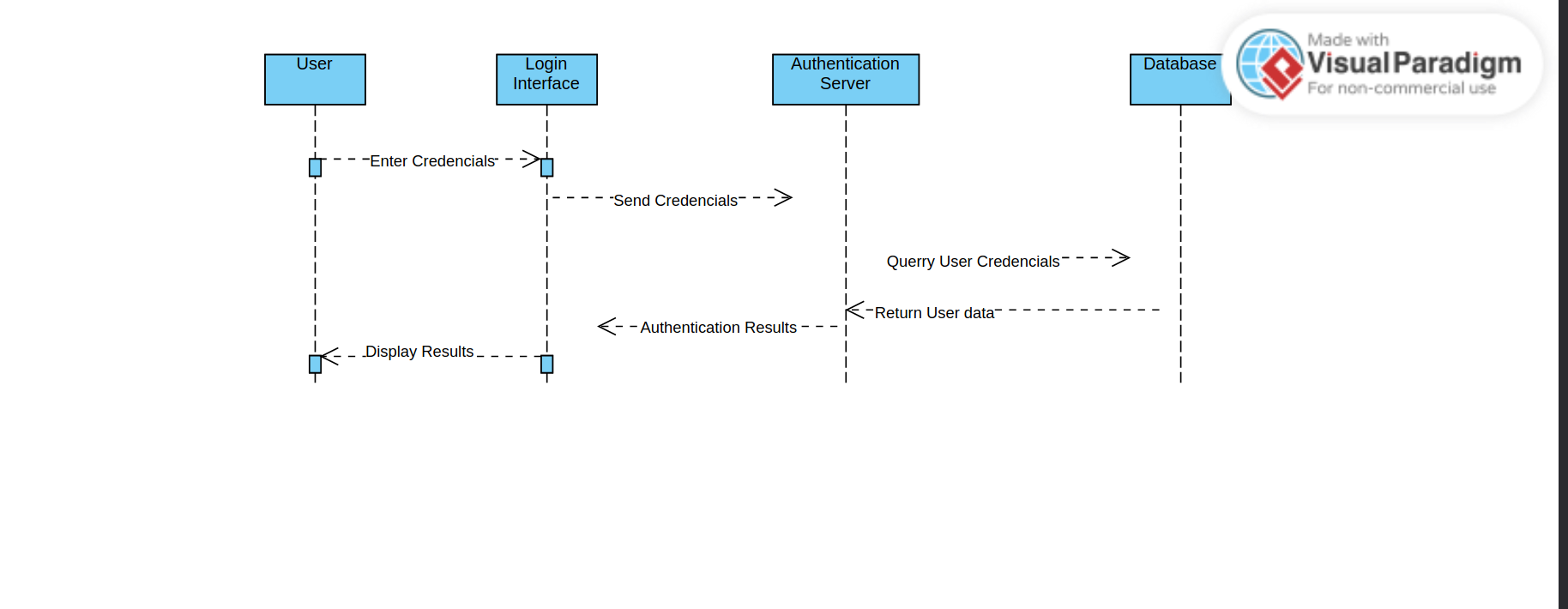


**2.2. Class diagram**

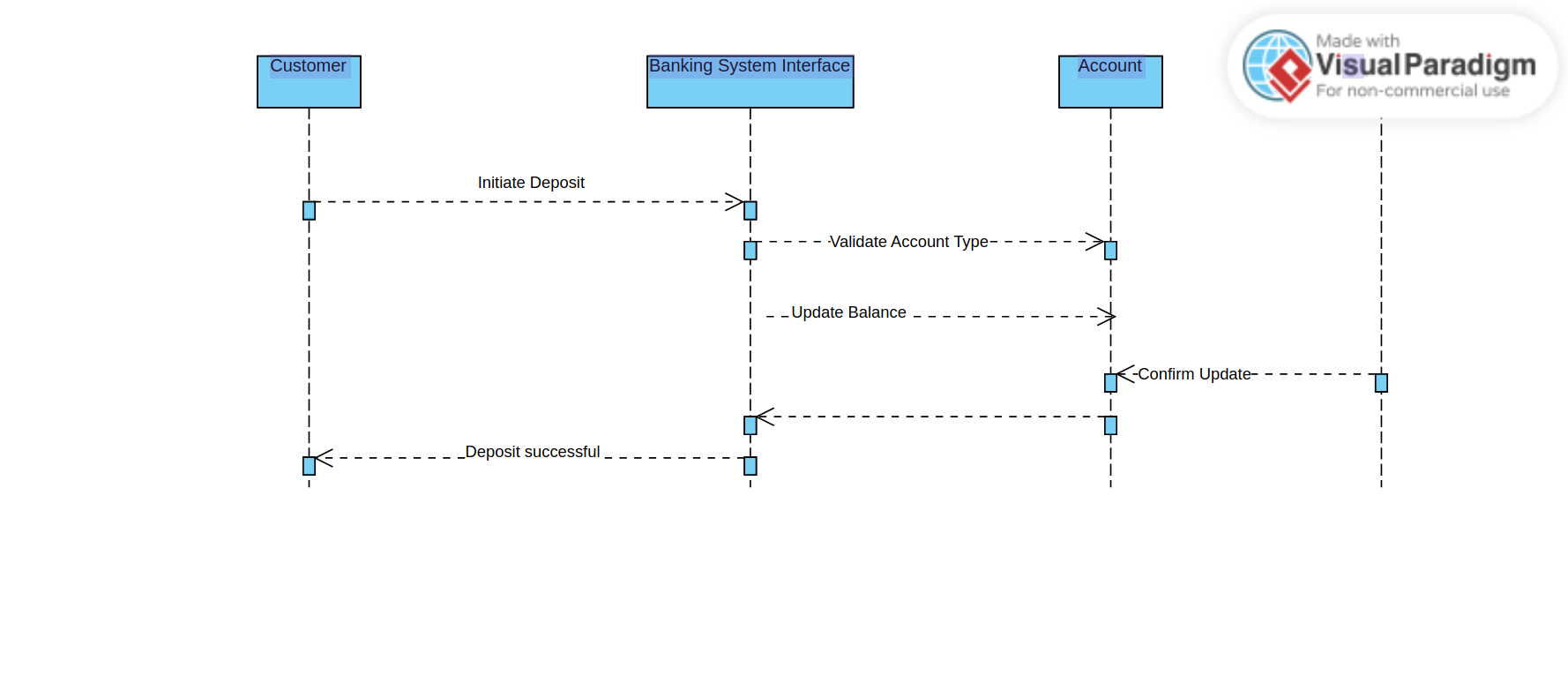


**3. Behavioural UML Modelling)**

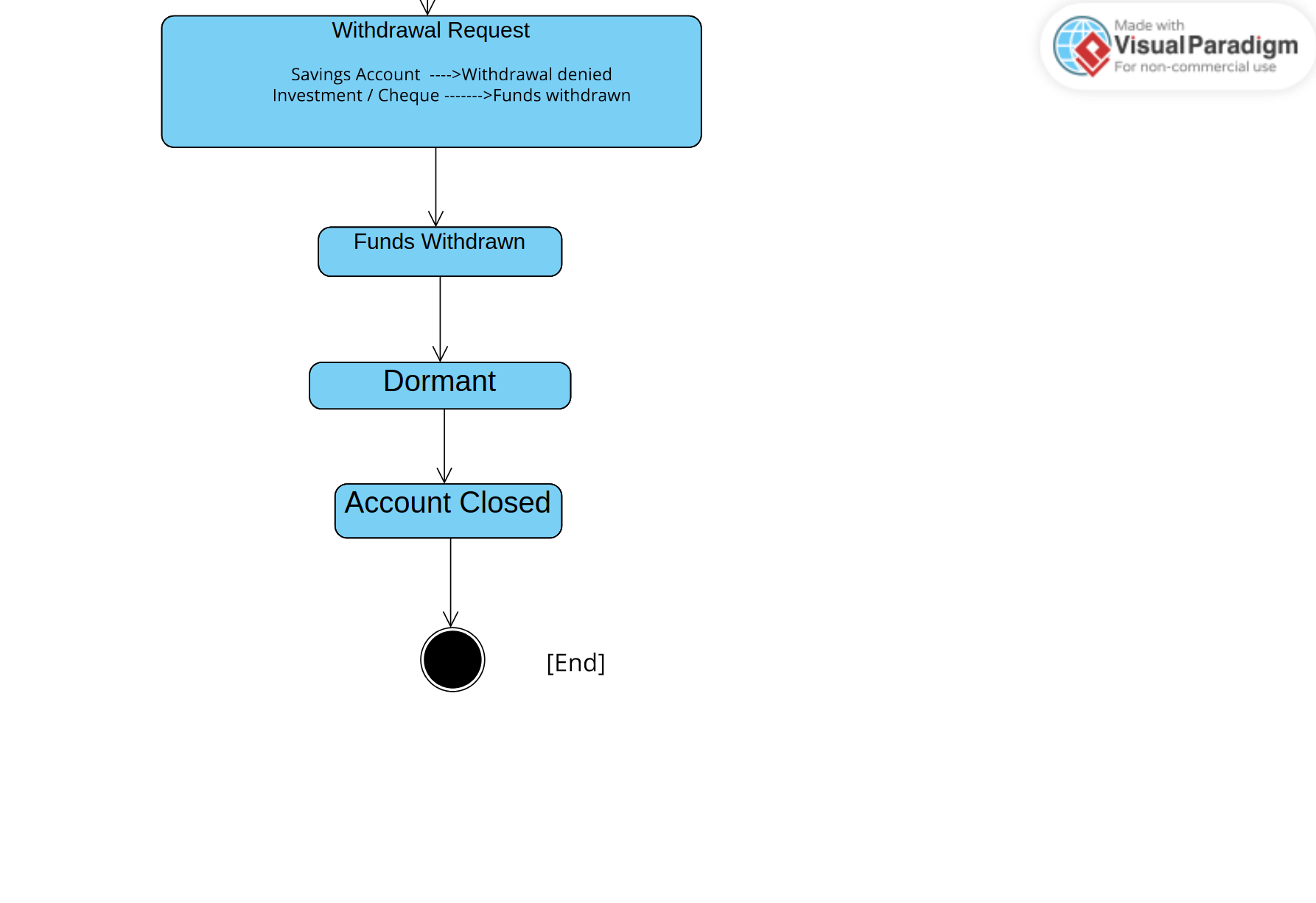
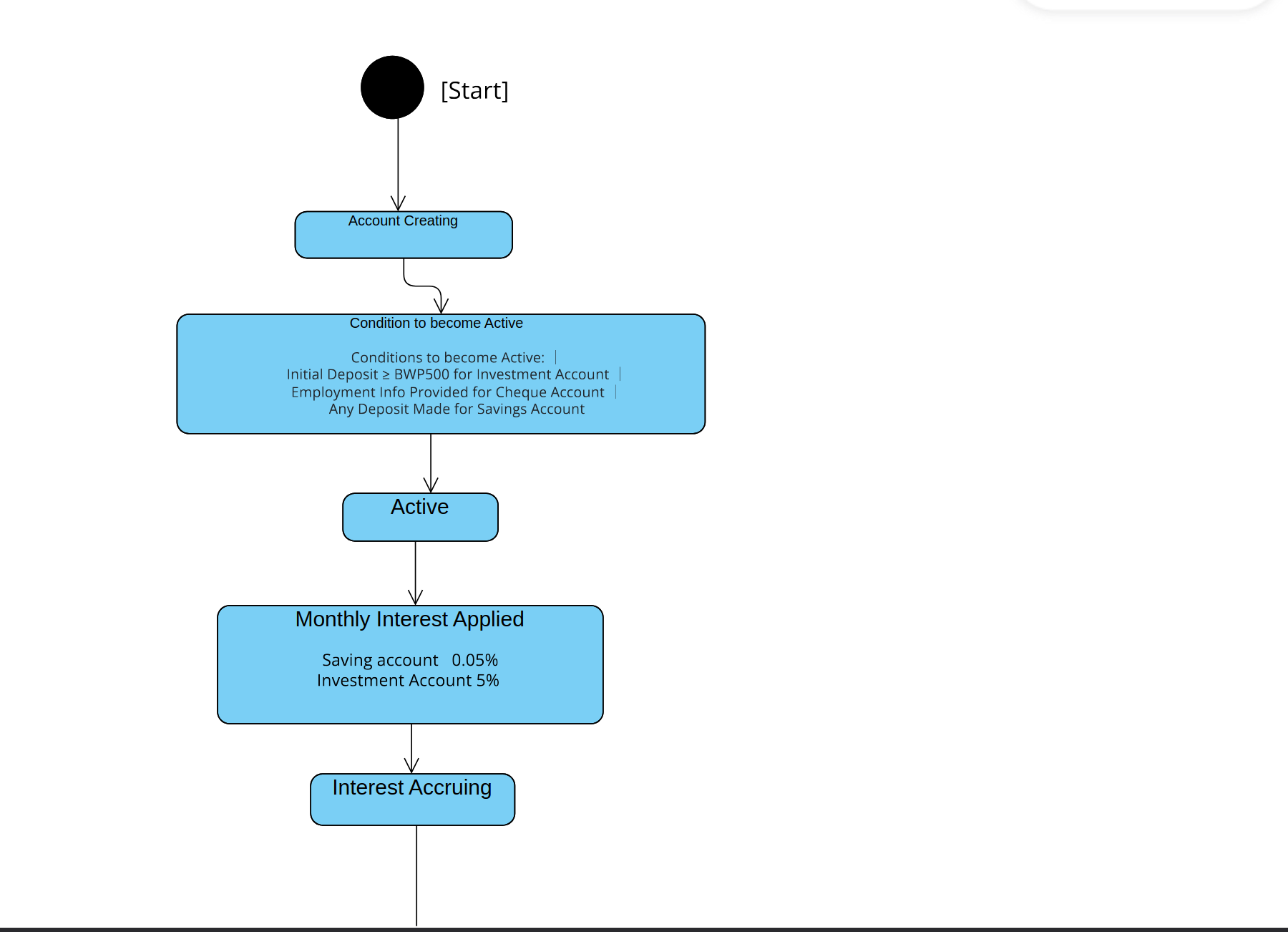
**3.1.**  Login **Sequence Diagrams:**



Deposit **Sequence Diagrams:**

There is DATABASE as the last lifeline where there is Visual Paradgm logo.

**3.2. State Diagram:**



**Meeting record appendix**