Introduction:

In this project, I need to emphazise i used a redactor to help with the report if you find the report very structured-format it is because i am using Grammarly to help me redact this report. to develop a comprehensive banking system comprising both a console-based application and a Model-View-Controller (MVC) application using C#. The project entails creating a system where users, both bank employees and customers, can interact with accounts, perform transactions, and manage banking operations. The project is divided into two parts: UML design and coding implementation. The UML design involves creating class and actor diagrams, while the coding phase involves actual implementation based on the design.

UML Design (Part 1) = >

For the UML design phase, I utilized LucidChart.com to create both class and actor diagrams. The class diagram illustrates the structure of the banking system, depicting classes such as Bank, Customer, Account, Transaction, etc. Each class represents a distinct entity within the system, and their relationships and attributes were carefully mapped out to ensure a clear understanding of the system's architecture.

Additionally, the actor diagram delineates the various actors involved in the system, including Bank Employees and Customers. It outlines the interactions between these actors and the system components, providing a visual representation of how users will interact with the application.

Moreover, in the short report accompanying the UML diagrams, I discussed key concepts such as abstract classes, concrete classes, inheritance, polymorphism, and access modifiers (public, private). I outlined how these concepts are utilized within the project, emphasising their role in achieving modularity, extensibility, and code reusability.

Coding Implementation (Part 2):

The coding phase constitutes the bulk of the project, where I translated the UML design into functional code. The project was divided into separate code files, each containing my name and student number for identification purposes. Adhering to the requirements, I ensured proper commenting throughout the codebase, providing clarity on the functionality of each component.
The code demonstrates the appropriate use of various object-oriented programming principles, including abstract classes, interfaces, inheritance, and polymorphism. Abstract classes were employed to define common behavior and attributes shared

among related classes, promoting code reuse and abstraction. Interfaces were utilized to enforce a contract for implementing specific functionalities across different classes.

Inheritance facilitated the creation of hierarchical relationships between classes, enabling code reuse and promoting a more organized class structure. Polymorphism was leveraged to enable objects of different classes to be treated interchangeably, enhancing flexibility and scalability within the application.
Access modifiers (public, private, protected) were used to control the visibility and accessibility of class members, encapsulating data and behavior within classes and preventing unauthorized access.
Arrays and collections were employed to manage and manipulate sets of data efficiently, facilitating tasks such as storing customer information, transaction records, and account details.
The code was appropriately separated into files to promote modularity and maintainability, ensuring that each component has a clear and distinct purpose within the application architecture. A testing module was implemented to validate the functionality of functions and objects, ensuring that the application behaves as expected under various scenarios.

Challenges Encountered:

Throughout the development process, several challenges were encountered, primarily related to the integration of different components and ensuring consistency between the console-based and MVC applications, unfortunately i could not successfully finish the Web App even though i found a lot of information about MVC Apps, the main problem was i following some tutorials of how to establish a mvc app and i got lost many times, my first try i could do a simple web app, but when i started the app i realised i was not separating my code into files and that made a lot of conflicts in the project.. Additionally, implementing file operations for data storage in the console application and we could create some files for the customers and their accounts.
Now i am more excited to learn more about the web app because i feel i spent many hours on the console app and i could not even reuse the code for the mvc app

In conclusion, this project served as a valuable learning experience, providing insights into software development methodologies, object-oriented design principles, and practical application development using C#. It underscored the importance of proper planning, design, and implementation in delivering a functional and efficient software solution. Moving forward, I am confident that the knowledge and skills acquired through this project will serve as a solid foundation for future endeavours in software development.