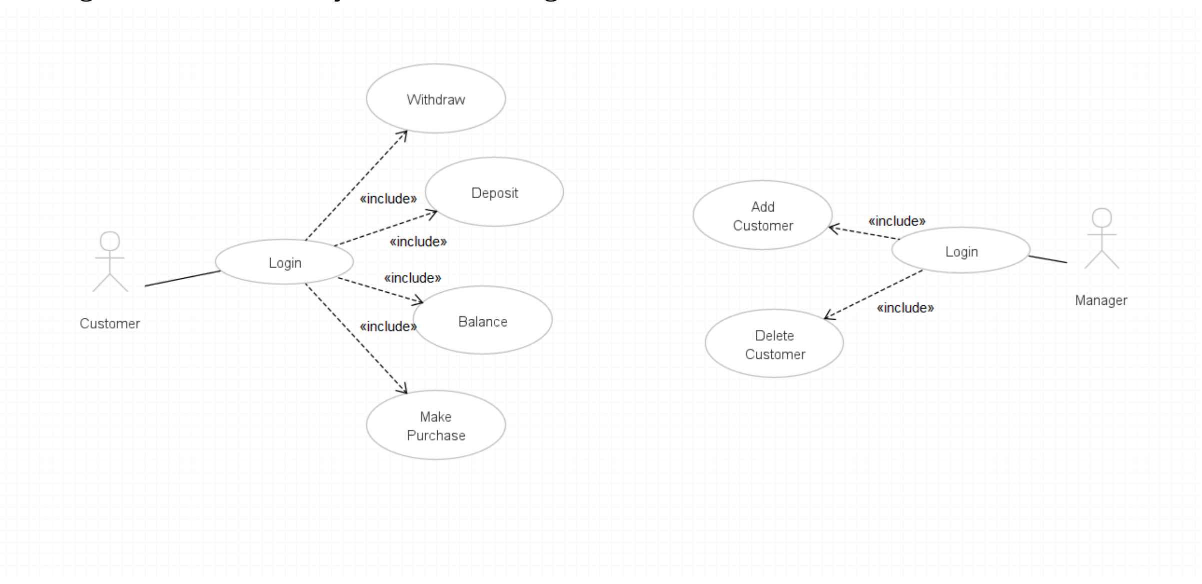
Name: Ndu Hachikaru

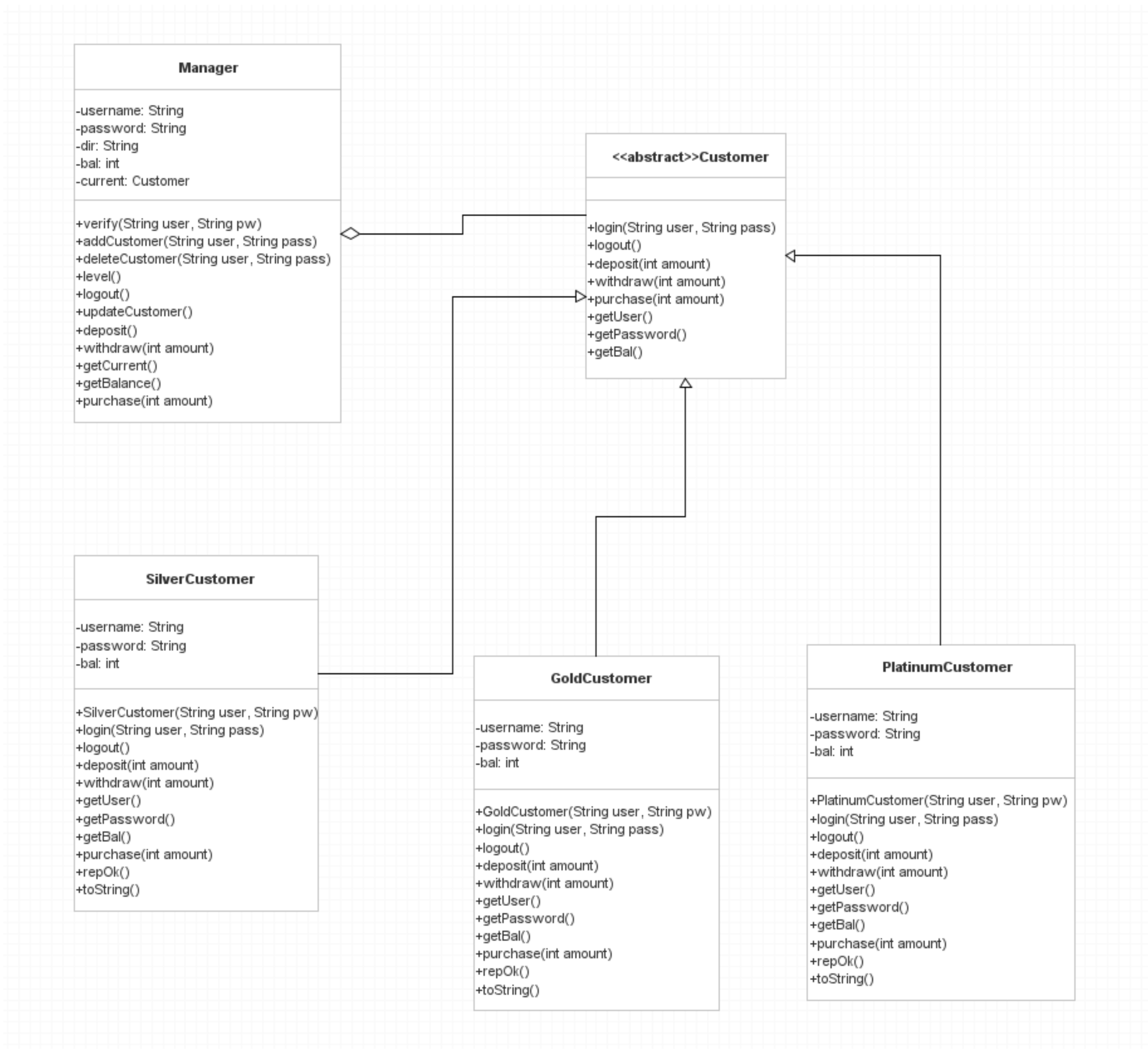
Student Number: 501211070

Section number: 06

The Use Case Diagram shows what users can do in the system. There are two types of users: Customers and Managers. Both can log in to the system, which is needed before they can do anything else. After logging in, Customers can do things like withdraw money, deposit money, check their balance, and make online purchases. Each of these actions also includes logging out when they're done. Managers, once logged in, can add new customers or remove existing ones, and then they also need to log out.



The Class Diagram shows the different parts of the system, like the classes and how they're connected. It has a main class called 'Customer' which includes basic info and actions for all customers. There are also three other classes called “SilverCustomer”, “GoldCustomer”, and “PlatinumCustomer” representing different customer levels. Another class, 'Manager', handles special tasks like adding or removing customers and checking login details. These classes work together to make the bank app work, and the 'Manager' does special stuff to manage the system.



For point number 2, I use the “Manager” class to write an Overview clause, an abstraction function, and a rep invariant. This class is responsible for managing customer accounts and overseeing the system's administrative functions. It is mutable as it allows for adding and deleting customers.

The State design pattern is reflected in the 'Customer' class and its subclasses “SilverCustomer”, “GoldCustomer”, and “PlatinumCustomer”. The different states represented by these subclasses dictate the behaviour of the customer operations such as withdrawal limits and fees for online purchases. Each subclass can override methods to reflect the behaviour appropriate for the customer's level or state.