# CS 3365 Software Engineering I Project: Vault

**Project Goal:**

The goal of our project is to create a vault application with 3 use cases. Specifically, the use cases will be defined by who is using the vault and what action they are planning to take while using it. Based on the user type (user, employee, administrator) they will gain access to different actions they can take while using the vault. Achieving this feature will require us to have different login information stored within a data structure, and a verification method that allows us to differentiate between user types. A graphical user interface (using JavaFX) will also be implemented for ease of access to the user(s).

Preconditions: The user wants to access the vault, the program can run on their machine

Post Conditions: The vault is left in the state that the user desires, whether that is open or closed, locked or unlocked, or adding and changing users' passwords.

Sequence:

1) The user wants to access the vault
2) The user starts the vault application
3) If the user is an employee or administrator, they press login
4) The vault application prompts the user to log in
5) The user enters their login information, including username and password
6) The vault application verifies their information and decides what type of login information it is
7) If the user has entered invalid login information, the vault application displays an error message notifying the user that their information is incorrect and could not be found.
8) If the user's login information is correct, the vault will allow the user to perform certain actions based on the type of their login information.
9) If not logged in, the user can only open and close the vault if open.
10) If the login information matches that of an employee, the employee can open and close the vault, as well as lock and unlock it.
11) If the login information matches that of an administrator, the administrator can open, close, lock, unlock, and change the password of the accounts for the vault system or add new accounts into the system.

**Potential Users:**

Realistically, the vault would be accessed by those who want to secure a resource or withdraw some a resource they have already secured within the vault. For example, a bank vault stores money, and there are phone apps that allow certain data on devices to be locked up, like messages, passwords, or photos. More potential users would include "administrators" which are users who have permission to add and modify vault users, and "employees" which are users who also have access to locking and unlocking the vault.

**Use Cases**:

Users: The user class allows regular users of the vault to access the vault. They can open and close the vault if it is in the unlocked state, which is set by either the employee or administrator.

Employees: When the employee wishes to access the vault, they do so by logging in with their employee login credentials, which allows them to change the state of the vault from locked to unlocked, or from unlocked to locked. They are also able to open and close the vault like users.

Administrator: When the administrator wishes to access the vault, they can do so with their administrator login credentials, which allows them to change the state of the vault's lock, as well as add and modify employees to the vault.

**Main.java:**

This class contains the main class that launches the program. It includes the code for the main GUI, which displays the buttons "Login", "Open Vault", "Unlock Vault", and "Modify/Add User".

**Employee.java:**

Employee.java is a class that contains a method *verifyPassword* that returns true if the entered password matches that of the specified password, and false if it does not. It also contains a method *changePassword* that takes in a String, and sets the old password equal to the new one to change it.

**Administrator.java:**

This class extends Employee.java and creates a classification for the administrator type. When the user enters the login information classified as type "administrator", they will gain more actions to perform within the vault. This class is used for polymorphism.

**Vault.java:**

This class contains the *accounts* HashMap that stores the account information. It also includes the constructor, which specifies the employee and administrator account information and stores it in the *accounts* HashMap. When the user has entered their login information, the method *verified* checks to see if the username entered is in the accounts HashMap and then if the employee associated with that username has the same password as inputted. This method will return true if the password matches the Employee associated with the username. Vault.java contains several more methods: *toggleLock* will change the state of the vault from "locked" to "not locked"; *isUnlocked* will return true if the vault is "not locked"; *toggleOpen* will change the state of the vault from "open" to "not open"; and *isOpen* will return true if the vault is open. The *setUser* method checks to see if the entered username is in the *accounts* HashMap and sets the current user to the associated Employee. The *changePassword* checks to see if the entered username is in the *accounts* HashMap, and if it exists, it calls the *changePassword* method found in Employee.java.

**AccountButton.java:**

AccountButton.java contains the constructor method that creates a popup dialogue box which both prompts the user to enter their login information and verifies the User. It then stores the result of this check in a boolean called *loggedIn.* If the boolean *loggedIn* is true, it sets the vault user to the current user whose login information has been entered. If boolean *loggedIn* returns false, a popup window displays that notifies the user that their credentials could not be verified and they need to try again. Once the user has logged in, the text on the top button in the GUI that says "Login" is changed to "Logout". This button can be clicked once to log out of the vault application, and user information will have to be re-entered upon clicking the "login" button again.