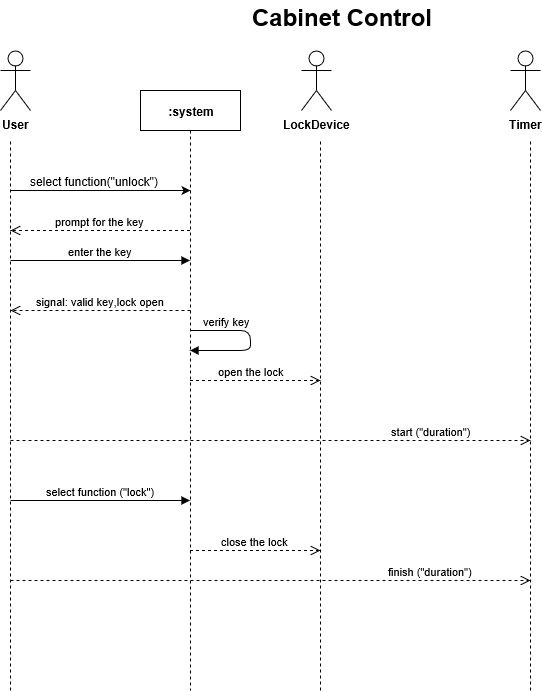
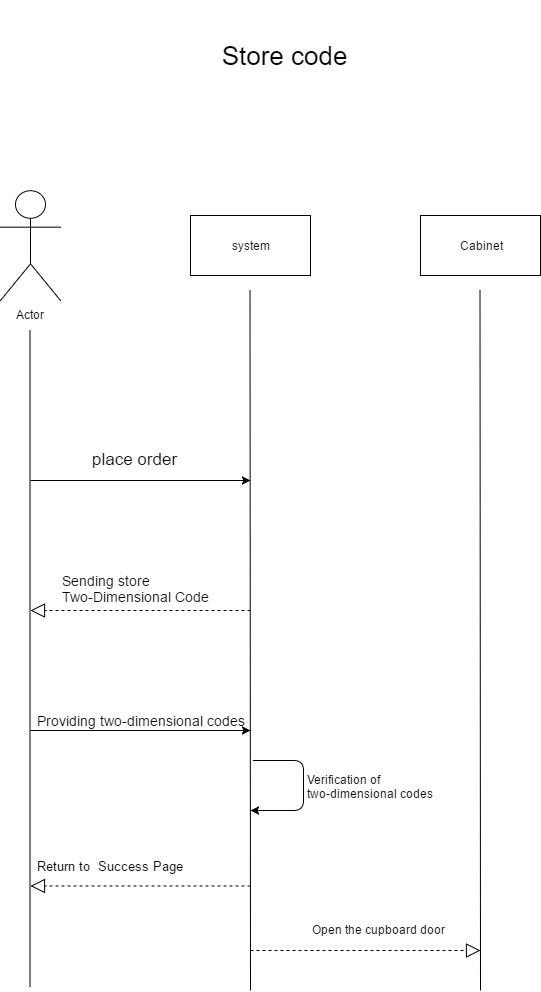
**Design pattern**



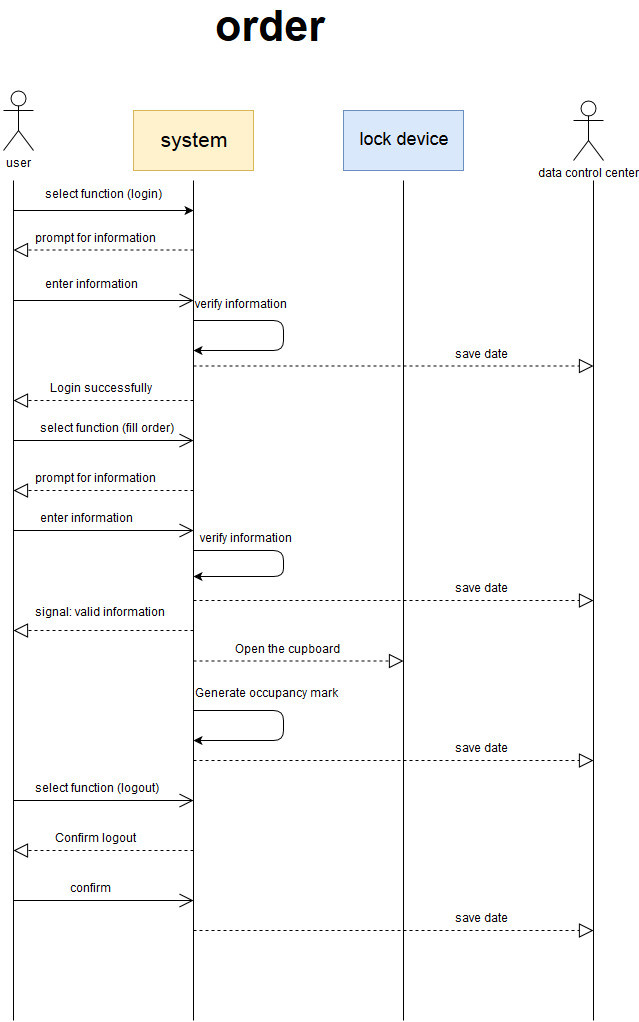
Cabinet control

The user chooses to unlock, and the system prompts the user to enter the validation code. LockDevice opens the cabinet after the user enters the validation code system to verify that it is correct. Then the user closes the cabinet.



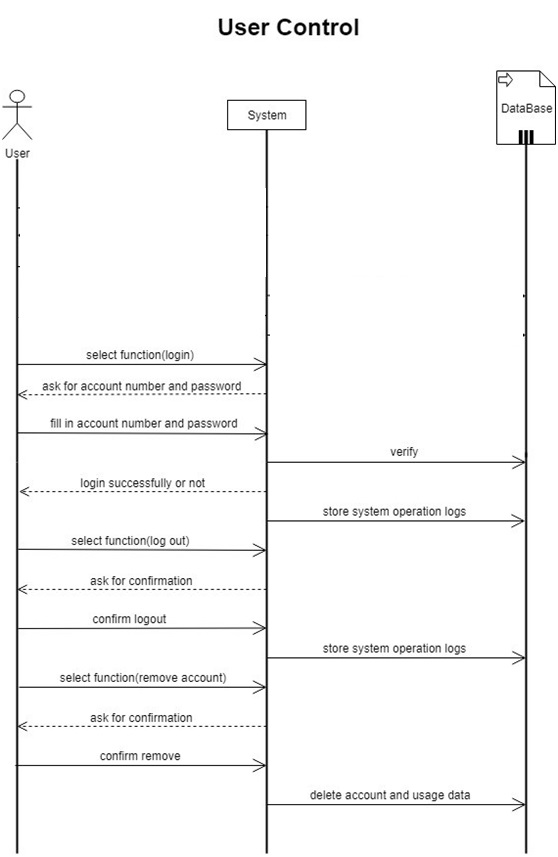
Store code

After users place orders, the system will send authentication codes to users. The user enters the authentication code and the system returns to the success page after verification. Cabinet opens the cabinet door.



Order

Log in to see if there are any orders. Some orders go to the order page. No orders directly show no orders. Users with orders enter validation codes and open the cabinet door after system validation is correct. Users can log out.



User control

Users choose to login, and the system displays the login page. The user enters the phone number and verification code, and the system verifies whether the login is successful or not. Database storage system operation log. Users can opt out and the system requires confirmation. User can quit after confirmation.

**We use the singleton pattern .**

Our entire system only needs to have a global object, which is conducive to our coordination of the overall behavior of the system. Singleton pattern can be controlled by instances. The singleton pattern prevents other objects from instantiating copies of their own singleton objects, thus ensuring that all objects have access to unique instances and flexibility. Because classes control the instantiation process, classes can flexibly change the instantiation process.

But singleton pattern also has its drawbacks.

1. expenses

Although the number is small, it will still require some overhead if an instance of a class is checked every time an object requests a reference. This problem can be solved by using static initialization.

2. Possible development confusion

When using singleton objects (especially those defined in class libraries), developers must remember that they cannot instantiate objects using new keywords. Because the source code of the library may not be accessible, application developers may find themselves unable to instantiate such a class directly by accident.

3. Object Survival Period

The problem of deleting a single object cannot be solved. In languages that provide memory management, such as. NET Framework-based languages, only singleton classes can cause instances to be unassigned because they contain private references to that instance. In some languages (such as C++), other classes can delete object instances, but this can lead to suspended references in singleton classes.