REPORT

SINGLETON DESIGN PATTERN

In the class UserPayment, the design pattern Singleton has been use. This is because Singleton design pattern enables a single object to be created at runTime. Creating multiple user objects for each user trial will be useless, and will waste a lot of resources. Rather ,we have ensured to create a single object and modify its attributes each time they change which saves in time, code length and memory.

STATE DESIGN PATTERN

In the package vendingMachine as a whole, the state design pattern has been use which is ideal for such an environment. The vendingMachine as seen in the code simply passes on to 4 methods which represent the different states of the machine (Start, DispenseChange, DispenseProduct, CancelTransaction). Depending on the user ,the vendingMachine oscillates between these 4 states. It should be noted that this pattern is similar to the Strategy design pattern.

MEMENTO DESIGN PATTERN

In the package State as a whole, as well as in the Client class, the Memento state design pattern has been used. This design pattern is quite helpful to archive states so as to reuse them and switch between them. This prevents other states to be created for nothing, and helps us to save a lot of resources. By using a takeCare class ,we store the different states into a list and once switching between such states, we simply retrieve appriopriate state from the list and set that state to the actual state(active state).

The project in general has other small interfaces found in package to enable it respect SOLID PRINCIPLES and the classes have been structured in an architectural manner for that same purpose