EMR System

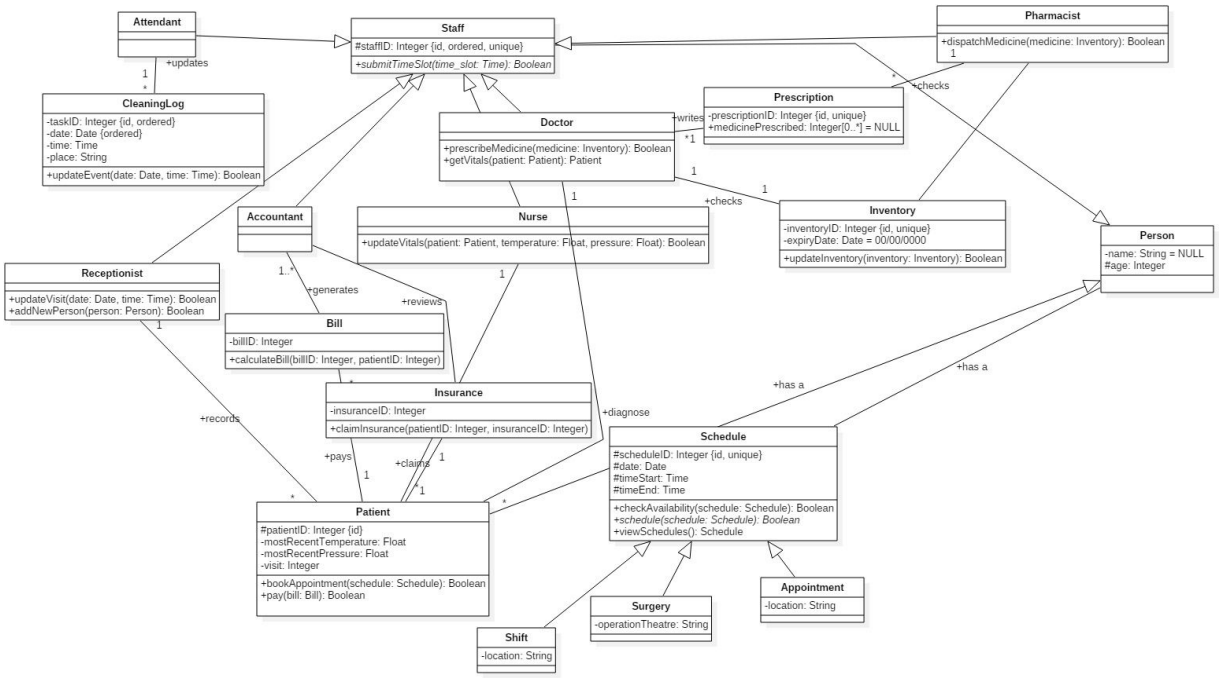
Project Part 3: Refactoring

Selva Priya Selvarathinavel, Selva Priyanka Selvarathinavel, Poorwa Hirve

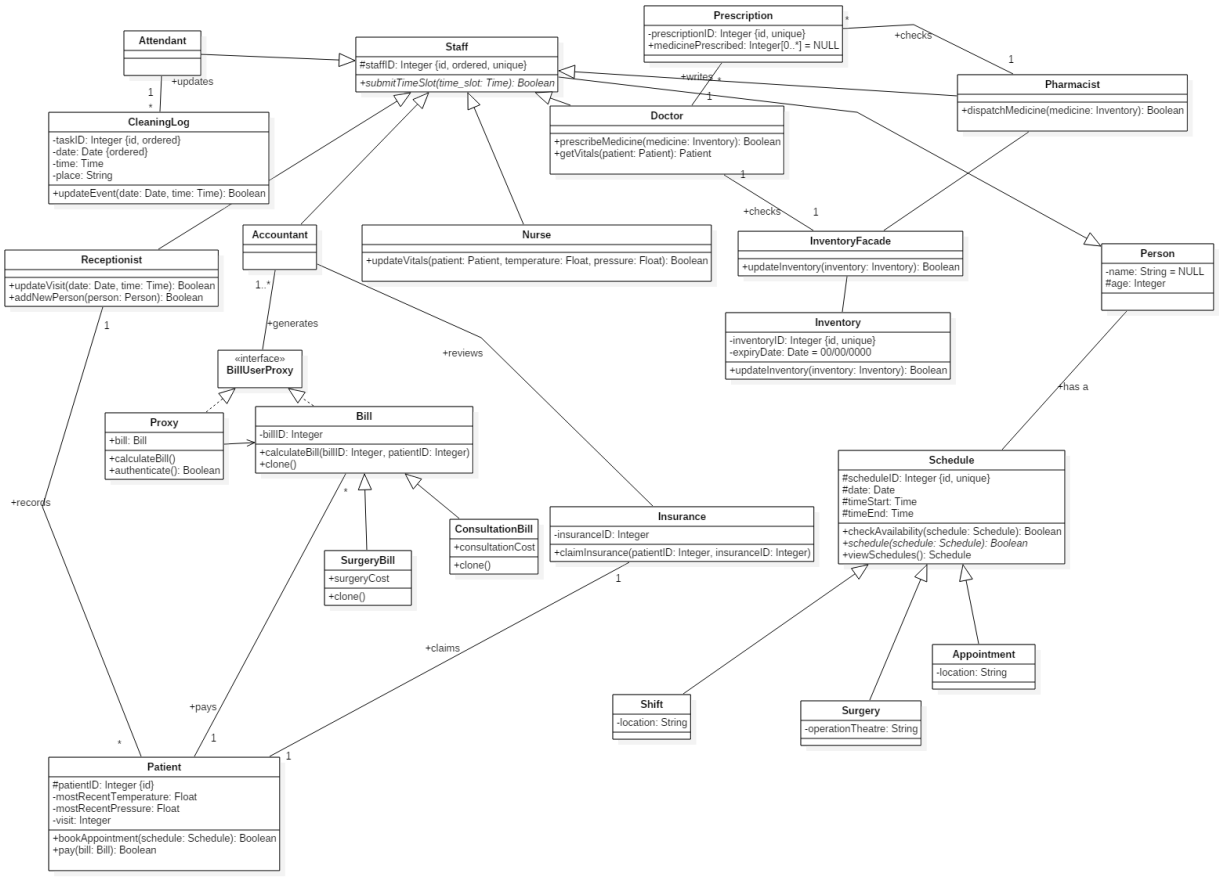
We have shown our old class diagram (corrected), followed by the refactored class diagram, and how and why we have chosen the design patterns used.

As per feedback for part 2, we have corrected our old class diagram by putting in data types of the attributes and function parameters of the class. We have also mentioned return types of the operators.

Our old class diagram:



Refactored class diagram:



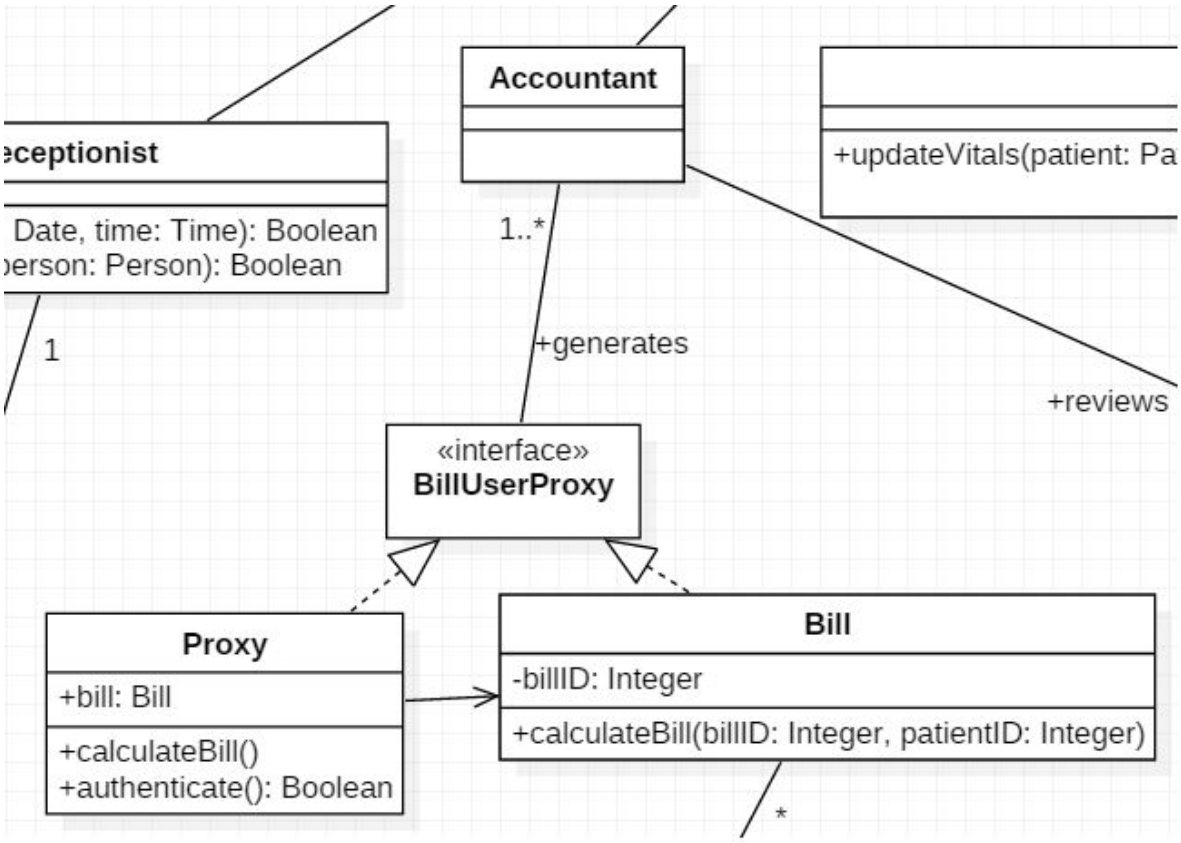
Design Patterns used:

1. Proxy

To authenticate the accountants' access to the bill.

Only some accountants can access the bill of the patient. To protect this access, we used a "Protection Proxy."

Absence of the Proxy would allow all clerks who update bills to view the details of the accounts of the patient / hospital, which would be a violation of the patient's rights and violation of hospital policies. The presence of Proxy ensures that the only the authorized accountants can access the invoices.

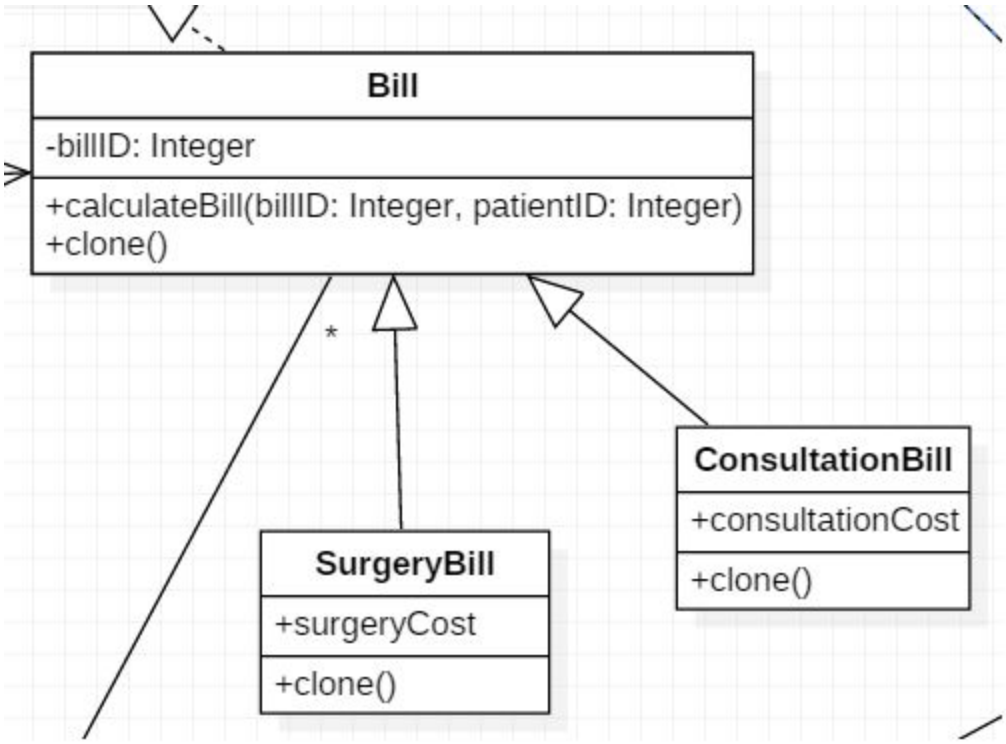


2. Prototype

Used for types of bills like Surgery and Consultation to generate common features.

Bill generation for surgery and consultation have similar features like cost and doctor. We can use these prototypes to instantiate the Bill objects faster.

If there was no Prototype, each bill would have to be defined separately with redundant attributes. Prototype ensures that these redundant attributes which are common to the different types of bills can be generalized.



3. Facade

Inventory changes are common which should not affect the functionalities of the rest of the classes. For example, the client code should have the same behavior even if inventory system changes. Absence of Facade would cause changes in the inventory to be reflected on the Doctor and Pharmacist where it should not. Hence Facade is necessary to keep the functionalities of inventory independent from the other classes that use the inventory.