**Final project writeup**

The following are the implemented patterns.

* Iterator design pattern - The iterator pattern is a behavioral design pattern that provides a way to access the elements of an aggregate object without exposing its underlying representation.
* Command pattern - The command pattern is a behavioral design pattern in which an object is used to encapsulate all information needed to perform an action or trigger an event at a later time. This information includes the method name, the object that owns the method and values for the method parameters.
* Template method - Template Method is a behavioral design pattern that defines the skeleton of an algorithm in the superclass but lets subclasses override specific steps of the algorithm without changing its structure.
* Observer pattern - Observer is a behavioral design pattern that lets you define a subscription mechanism to notify multiple objects about any events that happen to the object they’re observing.

The following are some of the challenges faced while developing the system and lessons learnt from them.

* Java - The project required to work with only java programming language to implement the design patterns, this was a challenge given the lack of extended experience with the language, this however, provided an opportunity to learn and understand the core principles involved in object oriented programming with Java.
* System requirements and design flow - Identifying a clear design flow was a challenge at first. This challenge was however solved by coming up with proper flow of the system which made it easy to choose and implement the appropriate design patterns that were needed.