Design Rationale

Group Name: AndrewAndBassel

There are several design patterns and refactoring methodologies that we adhered to while extending our software. The main design pattern being Model–view–controller (usually known as **MVC**). The main reason behind choosing that design pattern was the fact that MVC supports rapid and parallel development in a sense that since MVC basically divides the system into three interconnected elements, one programmer can focus on the view while the other can work on the controller to create the business logic. Since the assignment was a pair programming assignment, this model aided the process and helped us become more effective. In addition, modification in the MVC model does not affect the entire model. Adding a new type of view, for example, is very easy in the MVC pattern since the Model does not depend on the views party. Hence,changes in the Model won't have any impact on the entire architecture. Moreover, the Adapter design pattern was also adhered to mainly because it is the easiest way to make classes work together that couldn't otherwise because of incompatible interfaces. An Adapter will basically wrap around a class to provide it what an interface most users would expect. The Adapter pattern was our way around incompatible interfaces. In addition, we have followed the Observer pattern to ensure that there is a clear separation between the subject and the observer objects. Main reason the observer pattern was followed was that it allowed us to define boundaries between various modules, which improves the reusability and maintainability of our software