GitHub link: <https://github.com/saurabhjain071993/TEAM-02-CMPE-202>

Waffle.io link: <https://waffle.io/saurabhjain071993/TEAM-02-CMPE-202>

Week 4

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Team Member** | **SJSU ID** | **XP Values** | **Design Pattern Used** | **GitHub ID** |
| Ankil Shah | 010817703 | Respect | Decorator | Ankil0007 |
| Bhargav Jain | 010806510 | Communication | Decorator | jainbhargav |
| Bhavin Agrawal | 010827206 | Courage | Observer | Bhavin7181 |
| Saurabh Jain | 010809344 | Simplicity | Prototype | saurabhjain071993 |
| Vansh Shah | 010823761 | Feedback | Factory Method | vansh007 |

**XP Values**

1. **Simplicity**

As we completed the Activity, Use Case and State Diagram we designed the Class Diagram this week and I ensured that the designs were simple. Initially while designing the game we did decide about different patterns but while implementing I came across more simple designing patterns. After designing our class diagram we started with the implementations of the designing patterns in our code. I made the Prototype Designing pattern. With the right designing of the class diagram the implementation was very easy. I redesigned the designing patterns and ensured that the team mates individual designing patterns were simple. I have ensured that the game is in the deliverable state with basic functionalities working which were initially proposed. We have other features to add which will enhance the levels and complexity in the game. We will do that if time permits. In the last week I observed that proper planning helped us for simple designing and keep our complexity low.

I am writing the test cases for different scenarios and making sure that the game is running for different situation. We are ready with the final code to present.

1. **Communication**
2. **Courage**

In this week, my team was working on the implementation part. We as a team worked on each module together and finished the code. Our next task was to implement the patterns which we decided for each team member. Owing to facing difficulties initially, we discussed patterns concept again. Apart from this, we also discussed where and how we can apply these patterns and what will be the challenges.

As we had to make changes into one shared file, we encountered lots of conflicts. Upon discussion, we decided to sit together and implement the patterns individually. Some of my team members were not comfortable on patterns but I made sure nobody gets stuck at any point by encouraging everyone to be truthful about their current progress. I implemented Observer pattern which involved highest complexity compared to all other patterns as I had to integrate other patterns into Observer pattern. I started implementation after my team members completed their code and could complete successfully with the inputs from them. I made sure no other modules are affected with my pattern's implementation.

By maintaining the true definition of Courage value of XP, we could successfully reach our milestone to create the Chicken Runner game.

1. **Respect**

Our main task in this week is to implement design patterns to different modules of the game. With this we were required to implement remaining operations of the game like character selections. We collaboratively require to decide which characters should be included and which should be not. We have finalized three characters which are Chicken which is selected by default, Chicken Brown and Duck. One of our team member designed and implemented character selection functionality in the game. After implementing all the functionalities, it was required to design class diagram for each and every pattern.

My part was to design class diagram for Score Module in which I am implementing decorator pattern. I have designed classes and implemented my pattern to the score module. Main challenge for me was to integrate my pattern to one of our team member’s pattern Observer. It was not difficult to implement pattern individually but for us both pattern should go hand on hand and it was quite difficult to maintain behaviour of both patterns. We have implemented both the patterns successfully without disturbing behaviour of each pattern. It was required to change some implementation part of both the patterns to work with each other and we have achieved it successfully. My XP value is respect and in this way we have respected each other's implementation.

1. **Feedback**