Project 7 Final Report

Project: Undercooked

Team Members: John Lee, Charles Zhang, Dominic Lee

System Statement:

In our final project, all features stated in project 5 were implemented, including the tiles for grilling, cutting, and assembling. We merged the serving window with the assembly tile together so that the assembler fulfills the functionality for both. We also introduced some new tiles, such as the floor and countertop tiles, where countertops can have things placed on them, but cannot be walked on, and floors can be walked on, but cannot have items put on them.

There were a lot of technicalities that we ran into, and we therefore added a bunch more fields, such as the player being able to face multiple directions.

From project 6, there was a lot more implementation of GUI elements that we didn't have much exposure to previously, and therefore things like keyboard inputs, drawing screens, and timing were all implemented, which wasn't included in our original plan. The scoring system that compares orders to served food and creates a score was added also.

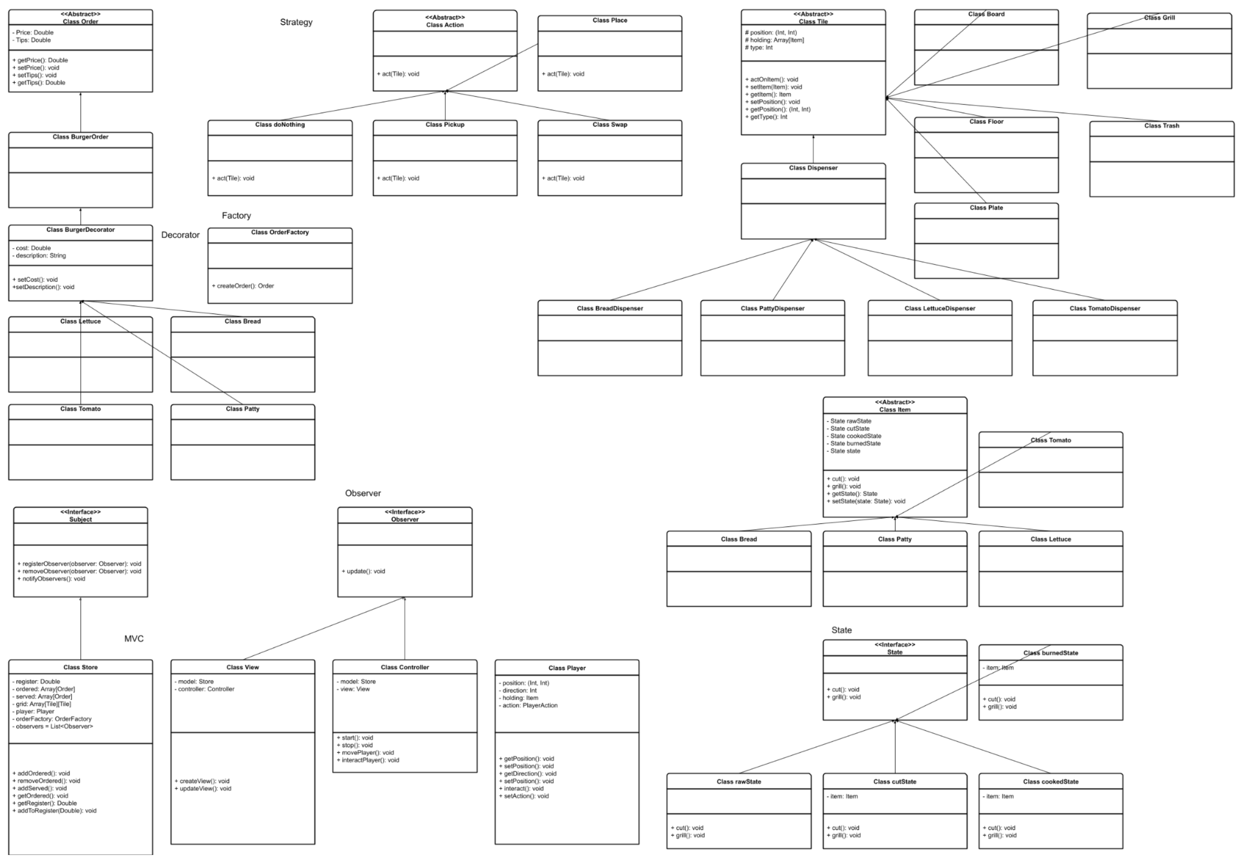
Class Diagram and Comparison Statement:

Looking back at our Project 5 Diagram, it was a lot larger than what our final project came out to be. I think this is because some of the patterns implemented didn't quite fit their purpose/were not necessary, or they were better implemented elsewhere. Two good examples of this is the strategy pattern, and the item states pattern.

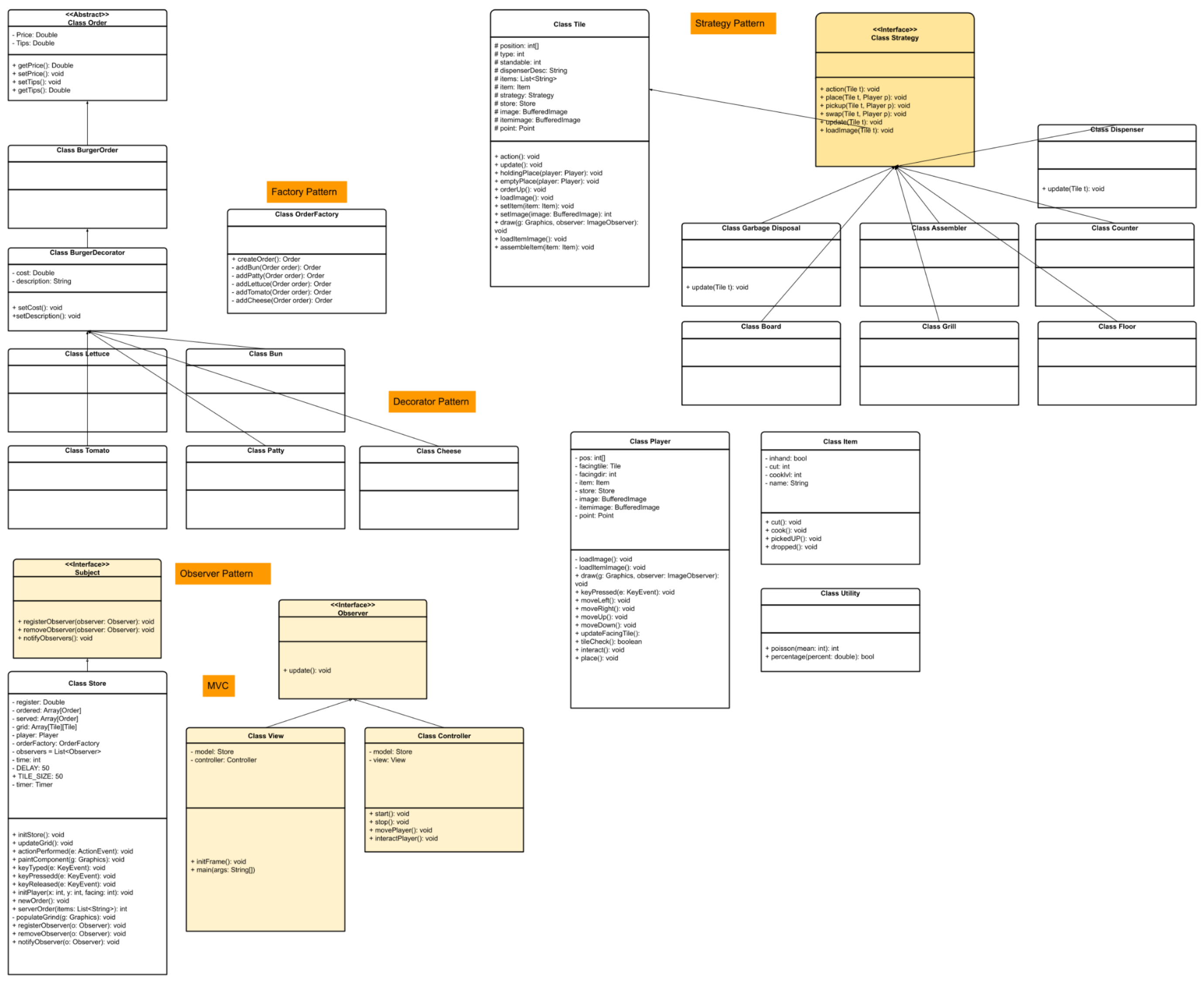
The strategy pattern turned out to be better utilized when placed in the tiles class. Since tiles don't differ much besides in their forms of operation, strategy was a perfect fit here, meaning we didn't have 6 subclasses of tile, but instead 6 different functions for strategy.

The state pattern we also completely dropped, as items were not complex enough to warrant using them, and we just stuck with checking the fields of items when it came time to determine how to work with them if their statuses changed.

Project 5 Class Diagram for comparison:



Final Project Diagram:



Third Party Resources:

The majority of code we wrote was on our own. We only really referenced and used snippets when working on the GUI, whose sources have been listed below. Images and Texture links are also added.

The GUI is created and managed through the Swing package. The rendering of the tiles and icons inside the window is handled thorough Swing. We also take advantage of the awt package, using its KeyListener and ActionListener interfaces to handle user inputs.

Code:

Example Timer format:

https://docs.oracle.com/javase/7/docs/api/javax/swing/Timer.html

For learning Swing and implementing it:
https://learncodebygaming.com/blog/how-to-make-a-video-game-in-java-2d-basics

Working with GUI's
https://stackoverflow.com/questions/36894487/java-gui-how-to-move-a-ball-using-wasd-keys

Image Textures:
Links have been shrunk to reduce clutter.
Images were taken for cut and whole. Other variations are color filters edited on.

Item Images:
Whole Patty
Cut Patty
Whole Bun
Cut Bun
Whole Tomato
Cut Tomato
Whole Lettuce
Cut Lettuce
Whole Cheese
Cut Cheese

Floor Tiles:
Dispenser Crate
Cutting Board
Garbage Disposal
Assembler Plate
Floor
Grill
CounterTop

Player Icons:
cheficon(up/down/left/right)

OOAD Process Statement:

1. Connecting code was very clean, since interfaces and objects as well as what they do was communicated well beforehand
2. Communication and expression of ideas was easy. Being able to explain complicated design ideas with simple pattern terms made working together far smoother.

3. Revisions and Deviations from UML Diagram were hard to follow, since adding major changes such as a GUI changed the way a lot of things worked, it took time to reorient ourselves.