Business Startup Game Coding Summary

Group 1 - Alfonso Arias, Vivek Ganesan, Juan Zambrano

Our semester's coding project is a simulation game of a Business Startup. This game will allow typical gamers and business professionals to understand how to run an effective startup company. The system must resemble the basic functionality of running a traditional business. The main components being tested are the basic components to running any business: profit, revenue, expenses, employees, reserves, assets, liabilities, equity, etc.

The game is built on a JavaFX GUI, allowing the user to navigate analytic menus via button functionality and display windows, such as text fields and PNG maps. The maps are acquired through the Static Maps library offered in the Google Maps API. The main system architecture used in this application was Model-View-Controller. This program is purely an event driven program. When a button is clicked, information is updated which then sends a signal to different objects to change their respective states. Some of the design patterns used in the program were Observer, Facade, and Decorator. These design patterns are primarily used when using any GUI event driven program.

The following are the four main components in running the simulation, accessed via buttons:

- **Track:** All the basic functions of a business:
    - Hire Employees,
    - Purchase Inventory,
    - Check Legal Status,
    - Track Revenue/Profit/Expenses, where the essentials of the business are provided.
- **Reserves:** The reserves feature will show the cash on hand. It will also allow the user to purchase a building.
- **Borrow:** The borrow feature will allow a user to borrow and pay back loans for a debt that is to no more than $200,000.
- **Map:** use the map to pick a location, use that location as a main marker; the map used in this program was primarily static. The maps included are from Chicago, LA, and NYC.

Some items to be tested to ensure the code is running without error and avoiding the possibility of interfering with the goals of the software are the implementations of "hiring employees", "making sales", "purchasing inventory", and "borrowing/paying back loans". An example of a test that ensure the functionalities of the aforementioned implementations include:

ID#1 - Make Sales

Description: System is able to generate sales
Items covered by this test: Revenue, Expenses, Inventory, Profit
Requirements addressed by this test: Not Applicable
Environmental needs: No external systems needed

Intercase Dependencies: purchaseInventory, hireEmployees
Test Procedures: The system must verify that at least 35 units 1 employee is hired in order to generate sales for a specific month.
Input Specification:. The inventory must be at least 35; the number of salespeople must be at least 1,
Output Specifications: The number of inventory must be reduced by the quantity sold, the amount in profit must decrease, the amount in revenue must increase,
Pass/Fail Criteria: Inventory will equal the original inventory minus the quantity sold; profit will increase by the inventory sold multiplied by the price; revenue will increase by the same amount; cash on hand will increase by revenue

Inspection:

With inspecting our code we agreed to meet in person; with the author presents, they will explain the functionality of method/ code they have written and the remaining partners will view the code with the aid of the checklist that was mutually agreed upon beforehand. After reviewing the method(s), the partners would discuss the results to the author whether or not it was approved by the checklist. After the inspection the partners may meet with the author to improve the code. An example of an inspection bellow:

Code: hireEmployee(String employee, HashMap<String, Integer> costs)
Inspectors: Alfonso and Juan
Date: 7/30/2019
Time: 3:00 pm
Results: Overall the inspection of Vivek's code for the method of hiring an Employee for the business has passed. There was an appropriate number of comments in his code that was easy to read and follow through. Each condition for accepting an employee for a certain position had the correct response when the business could not afford nor if the position has already been filled.

The basic functions of the business were the main accomplishments of the system. In this system, the user could perform basic tasks such as purchase a building, hire employees, purchase inventory, make sales, and report revenue/profit. Since these are the pillars to any startup, it was essential that these tasks were included in the system.

In the future, the goal is to integrate a map that provides real time data on the most effective locations to do business. Since the location of the business was intended to be the main aspect of the game, getting a real time Google Maps API was a constant focus of our time. However, due to costs and other constraints, the system was limited to static maps of just 3 cities. Another task that could not be accomplished was allowing the system to be specific to a certain type of business. This could not be done primarily because of the need to develop specific algorithms for specific businesses. Since that was the main part of the system intent, that is a critical component that should be the center of focus in future iterations.