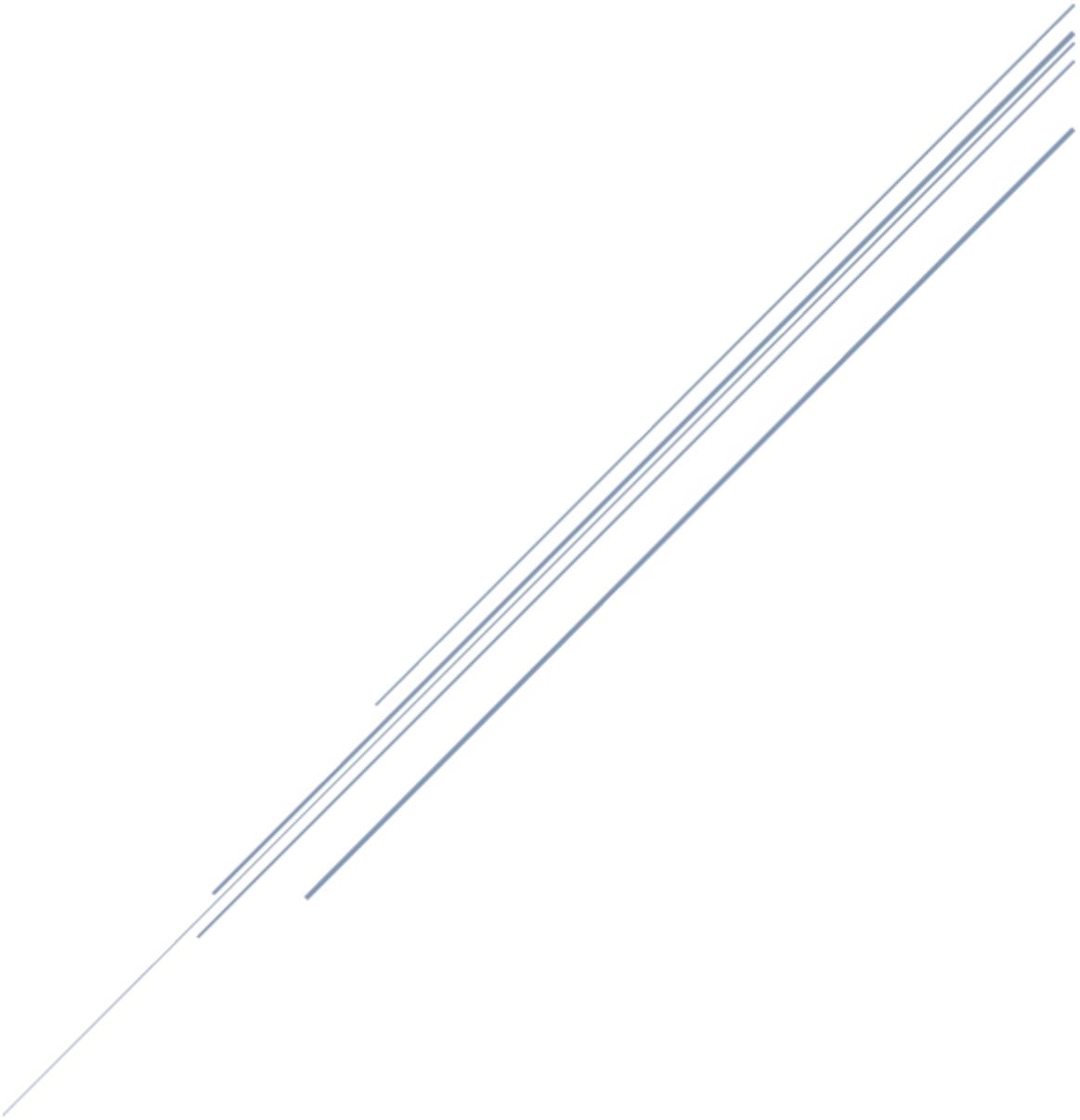
DOCUMENTATION

CK Chan 30924427

Mahmoud Elsafi 22153787

Jadon Hladyshevsky 48301162

Jin Hee Yun 19023407



University of British Columbia
CPEN 333

Executive Summary

Introduction

Amazoom has been looking for a way to automate its warehouses to cut costs. However, Amazoom has many warehouses that come in various dimensions with its products randomly distributed across all warehouses. A flexible solution is needed to accommodate for this variance in warehouse size as well as product placement.

Proposed Solution

In order to properly visualize this problem without investing a lot of money at the start, we will be creating a computer simulation of our solution to observe the effectiveness of our solution.

Amazoom warehouses are visualized as a grid and can be created with flexible dimensions to closely mimic the real-life sizes of the warehouses. Each warehouse has its inventory so an admin chooses where to place the item and specifies the item information. The users only see the inventory in all warehouses, where items with the same name are grouped together and then can add the items to cart. Each warehouse also has a loading bay where trucks can receive items from robots.

Our simulated website will be able to keep track of the location as well as the number of every product in our warehouses. It will also inform our managers when a certain product is low in stock, so that they can place orders to replenish the stock. Finally, the managers can monitor the status of all current orders to see if any problems arise.

Robots in the warehouses will keep track of the location of other robots and know their place. In order for only one robot to be located in one position, each warehouse creates a thread-safe array map. Also, robots are programmed to be only moving clockwise to avoid collision. A semaphore is implemented with the robots so that only a certain number of robots can be located at a time. Additionally, task chaining is used for robots so that each item to pick up is chained after the antecedent is successfully completed.

Multithreading is used in the whole system to create a more efficient simulation. Specifically, we use multithreading for the robots retrieving the items to minimize the time it takes for all the items to be collected from the warehouses. Moreover, in order to test our model view controller page functionality, we tested by using writing to the console to ensure all the possibilities we consider meet the expected results. In addition, tests were done by creating randomized databases and having items/warehouses automatically be added/removed and observing if the central computer robot and truck transportation systems still accurately functioned regardless of specific parameters.

Final Thoughts

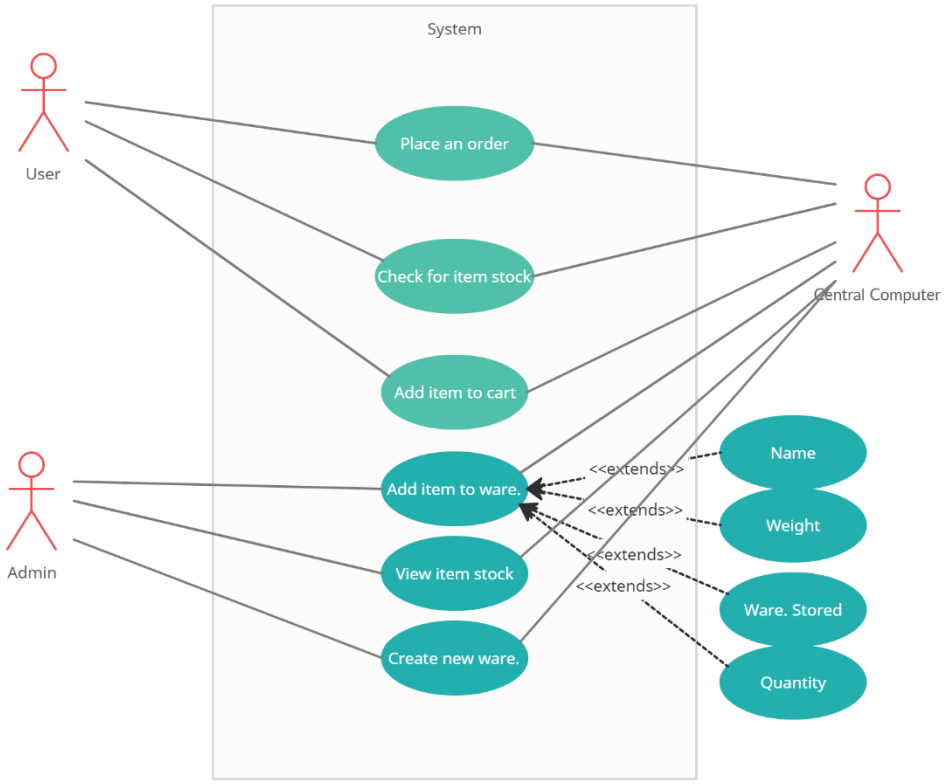
We believe that our solution will help Amazoom move into the next era of automation. By creating this flexible simulation, we can easily improve and change parts of it, creating the most efficient solution without sacrificing many resources.

Tech Stack

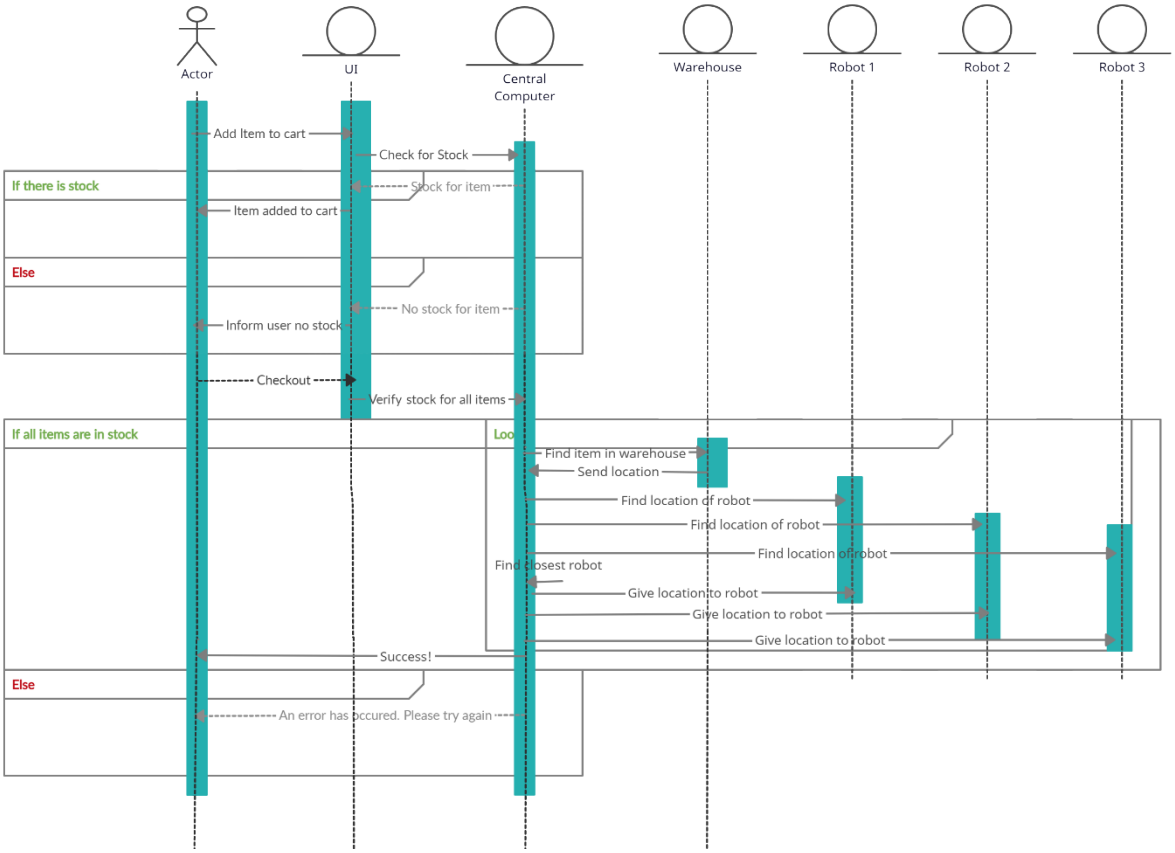
For the front end, we opted to use ASP.NET with MVC model. This is because we saw that the front end coding is separate in many ways from the back end coding, such as the robot traversal. This was beneficial for us as there was a lot of back end code that we had, and proceeding with an MVC model allowed members to work simultaneously on the front and back end and then integrate them together nearing the conclusion of the project.

For the back end, we used C# because it was first of all required, but it was also the language in which we were most comfortable with as most of this course material was taught using C#. A lot of the functions that we learned and put into use in this project were in C#.

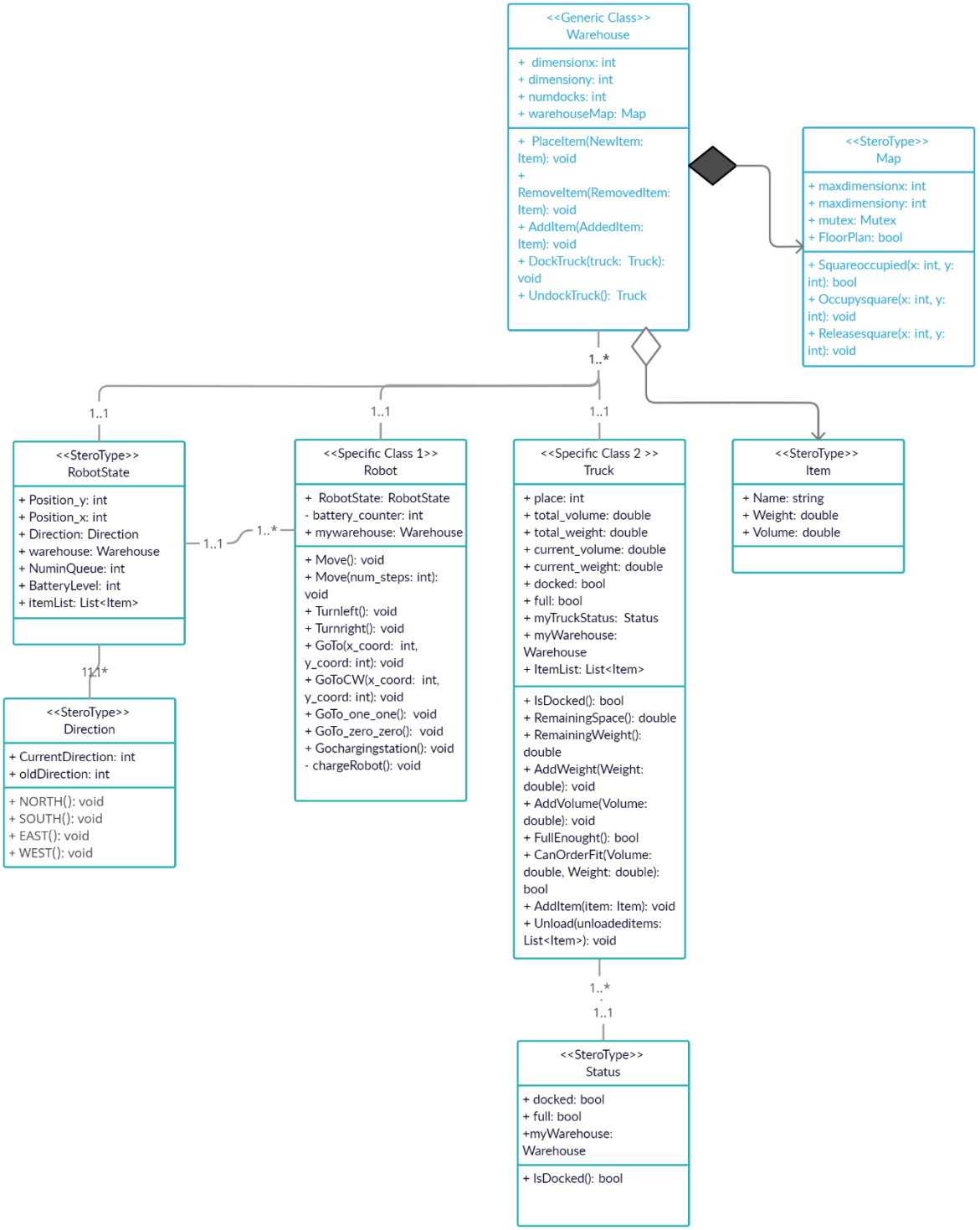
Use Case Diagram



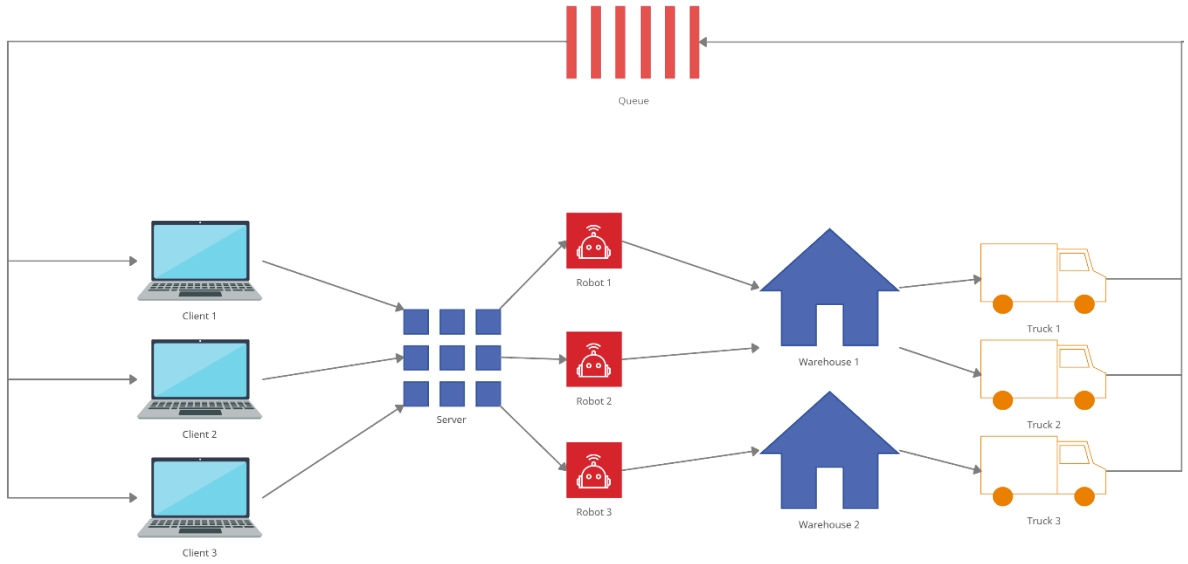
Sequence Diagram



Class Diagram



Object Interaction Diagram



Further Justification

Additional details on specific choices we made when implementing this system are found in the expanded pdf version of our presentation slides also in our github repository. Our readme included with the github repo should include an administrator username and password to log in with in order to access all the admin features of our system, such as warehouse and item management.