**Assignment 5 - COMP 3220**

1. Discuss the benefits of studying design patterns.

- Provide you with a way to solve issues related to software development using a proven solution

- Reusing design patterns help to prevent subtle issues that can cause major problems, and it also improves code readability for programmers who are familiar with the patterns

- Provide general solutions, documented in a format that doesn't require specifics tied to a particular problem

- Allow developers to communicate using well understood names for software interactions

- Make communication between designers more efficient

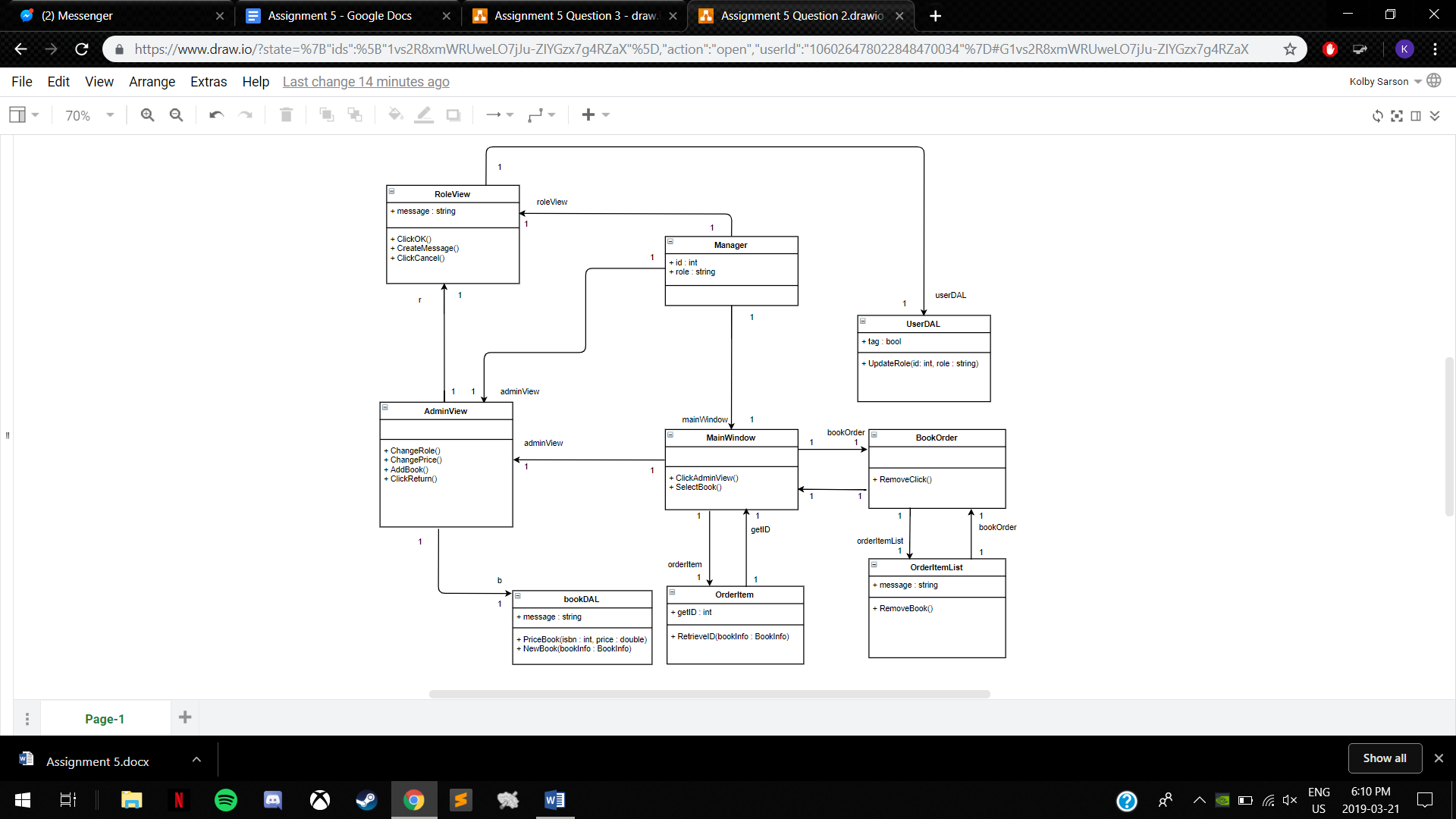
- Speed up the development process by providing tested, proven development paradigms

- Common design patterns can be improved over time

4. There are different types of users for the online bookstore system, including guest users, club members, salespersons, and managers. Discuss how the maintainability of the system can be improved by using a design pattern to deal with the variety of users.

The maintainability of the system could benefit from the design patterns of low coupling and high cohesion when dealing with a variety of users. Either of these design patterns will improve the maintainability of the system. Low coupling means that the individual classes or objects are not dependent on one another and high cohesion means that these classes and objects are grouped by how related they are to one another. In combination, we group related classes and objects while ensuring that they have as little dependency on one another as possible. This can improve the maintainability of the system by making it easier to perform error corrections, system performance improvements, and system changes.

2. In Assignment 4 (A4), a class diagram was derived according to the provided sequence diagram. In A3, a sequence diagram was constructed for the “Remove Book” use case. Use your “Remove Book” sequence diagram to update the class diagram that you had finished in A4.



3. Illustrated in the video clip of A1 includes a feature of the online bookstore system, i.e., the “Modify Amount” use case that allows a user to modify the amount of a selected book as long as the modified amount does not exceed the number of the book on stock. Go through the Unified Process to update your class diagram for this new use case. In the process, the OOA artifacts need to include a **fully dressed use case** and a **system sequence diagram,** and the OOD artifacts include a **sequence diagram** before updating the existing **class diagram.**

|  |  |
| --- | --- |
| **Use Case** | **Modify Amount** |
| Brief Description | This use case allows a Customer to modify the amount of a book from their current booklist |
| Actors | Customer |
| Preconditions | The bookList must contain at least one book. |
| Main Flow A1 | Customer selects a book from the bookList. |
| A2 | Customer clicks ‘Modify Amount’. |
| A3 | Customer inputs quantity into text field. |
| A4 | Customer clicks ‘OK’. |
| A5 | The system updates book quantity in the bookList. |
| Alternative Flows A6 | The system outputs an error message if quantity inputted exceeds Books in Stock. |
| A7 | The system outputs an error message if quantity inputted is less than 0. |
| Post-Conditions | The booklist is updated. |

