**INTI International College Penang School of Computing**

**3+0 Bachelor of Science (Hons) in Computer Science, in collaboration with Coventry University, UK 3+0 Bachelor of Science (Hons) in Computing, in collaboration with Coventry University, UK**

# Coursework cover sheet

**Section A - To be completed by the student.**

|  |  |
| --- | --- |
| Full Name: STEVEN TEO BOON JIE | |
| CU Student ID Number: P23014826 | |
| Semester: 2 | |
| Session:  **April 2023** | |
| Lecturer:  **Puteri Nursyawati Azzuri (puteri.azzuri@newinti.edu.my)** | |
| Module Code and Title:  **4067CEM Software Design** | |
| Assignment No. / Title:  **Continuous Assessment** | % of Module Mark:  **50** |
| Hand out Date:  **12 May 2023** | Due Date:  **Task 1: 02 June 2023, by 11.59pm.**  **Task 2: 07 July 2023, by 11.59pm**  **Task 3: 23 June 2023, by 11.59pm.**  **Task 4: 23 June 2023, by 11.59pm.**  **Task 5: 23 June 2023, by 11.59pm.** |
| Penalties: No late work will be accepted. If you are unable to submit coursework on time due  to extenuating circumstances, you may be eligible for an extension. Please consult the lecturer. | |
| Declaration: I/we the undersigned confirm that I/we have read and agree to abide by the University regulations on plagiarism and cheating and Faculty coursework policies and procedures. I/we confirm that this piece of work is my/our own. I/we consent to the appropriate storage of our work for plagiarism checking.  Signature(s): *STEVENTEO* | |

# Section B - To be completed by the module leader

|  |  |  |
| --- | --- | --- |
| Intended learning outcomes assessed by this work:   1. Understand and apply appropriate concepts, tools, and techniques to each stage of the software development. 2. Understand and apply design patterns to software components in developing new software. 3. Demonstrate an understanding of project planning and working to agreed deadlines, along with professional, interpersonal skills and effective communication required for software production.   5. Demonstrate an awareness of, and ability to apply, social, professional, legal, and ethical standards as documented in relevant laws and professional codes of conduct such as that of  the Malaysian National Computer Confederation. | | |
| Marking scheme | Max | Mark |
| 1. User Story Mapping | 20 |  |
| 2. Setting up a GitHub |  |
| Repository | 10 |
| 3. Creating a Class diagram and |  |
| design pattern selection | 30 |
| 4. Creating a Prototype User |  |
| Interface and Usability Testing | 20 |
| 5. Discuss the ethical issue |  |
| related to the software | 20 |
| Total | 100 |  |

**The 4067CEM assessment should be completed as a full individual work over the course of the module. The assessment output are only judged at the end of the module and not by the expectations during that week. The assessment should be undertaken individually. All submissions will be checked against each other and the internet for possible plagiarism.**

Activities – These activities consist of **50%** of your coursework marks. It will be run throughout the semester and there will be a final submission at the end of the semester. These activities consist of activities that will be done in a software design phase.

# System

Student Business System for College.

# Task 1 – User Story Mapping (20 marks)

The first thing that you need to do is ask the user what they wished for in a system. The user here can be your friends as the system is related to them. Get at least 10 real users to get their feedback. Document their feedback. Use software like Trello to complete this activity.

Output – All the user stories, and backlog with goals, activities, and tasks. In Word format, uploaded it to GitHub.

Due – Week 9 of the semester. 02 June 2023, by 11.59 pm.

# Task 2 – Setting up a GitHub Repository (10 marks)

This is where the output of the tasks will be stored, Make sure you register an account, create a repository and your files are uploaded here and it is in an organized manner and can be easily found.

Output – GitHub Repository with Task 1, Task 3, Task 4 and Task 5 documents. Take note the date of the files will be shown so you must follow the due date of each task.

Due – It will be accessed on Week 14 of the semester. 07 July 2023, by 11.59 pm

# Task 3 – Creating a Class diagram and design pattern selection (30 marks)

Create a simple Class diagram which should consists of the Classes that might be used to represent the system and the association between them. You don’t have to declare the attributes and operations for this activity. You do have to explain the class responsibility of each class declared. You can use software like StarUML to complete this activity.

Output – A class diagram containing classes and associations. In Word format, uploaded it to GitHub.

Consider the problem and select a suitable design pattern that can be implemented on the problem. Give justification on why the design pattern was chosen. Draw the UML diagram representing your class diagram as a design pattern UML. Include all the abstract class/interface, concrete class, and inheritance (if any) used to represent the problem.

Output – UML diagram representing the design pattern. In Word format, uploaded it to GitHub. Due – Week 12 of the semester. 23 June 2023, by 11.59 pm.

# Task 4 – Creating a Prototype User Interface and Usability Testing (20 marks)

Create a Prototype User Interface (hand drawn/digital) of TWO (2) important functions of the proposed system. Come up with usability testing questions. You don’t have to carry out the test, just prepare the questions. You should indicate what you are testing for in the Usability Testing.

Output – A Prototype and Usability Testing Questions. In Word format, uploaded it to GitHub. Due – Week 12 of the semester. 23 June 2023, by 11.59 pm.

# Task 5 – Discuss the ethical issue related to the software (20 marks)

Discuss and do a critical analysis of your software in these areas, privacy concerns, intellectual property rights, and effects on society.

Output – A report in Word format, uploaded to GitHub.

Due – Week 12 of the semester. 23 June 2023, by 11.59 pm.

# Submission

All tasks needed to be documented in Word format and submitted for SafeAssign checking (Links will be provided before the due date).

Upload the document and the SafeAssign report to your GitHub repository by each task due date. Due – It will be accessed on Week 14 of the semester. 07 July 2023, by 11.59 pm

# Marking Rubric for Continuous Assessment

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Marks Below 40%** | **Marks in the range 40 – 49%** | **Marks in the range 50 – 59%** | **Marks in the range 60 – 69%** | **Marks 70% and above** |
| **User Story** | User Story Mapping | User Story Mapping | User Story Mapping | User Story Mapping | User Story Mapping done and does capture most important activities of the system. The breakdown of the user story mapping is excellent and uses software that can assist that process (For example Trello compared to Ms.  Word). |
| **Mapping** | not done or User | done at a minimum | done and does | done and does |
| **(20 marks)** | Story copied/does  not match the exact | level and does not  capture the | capture several  important activities of | capture several  important activities of |
|  | system. | important activities of | the system. The | the system. The |
|  |  | the system. | breakdown of the | breakdown of the user |
|  |  |  | user story mapping | story mapping is good |
|  |  |  | can be improved. | and uses software that |
|  |  |  |  | can assist that |
|  |  |  |  | process (For example |
|  |  |  |  | Trello compared to |
|  |  |  |  | Ms. Word). |
| **Setting up a** | GitHub repository | GitHub repository | GitHub repository | GitHub repository exist | GitHub repository |
| **GitHub** | does not exist or | exist and some of | exist and most of the | and all of the required | exist and all of the |
| **Repository** | cannot be accessed | the required files are | required files are | files are available at | required files are |
| **(10 marks)** | or the required files  are not available at | not available at the  time of access. | available at the time  of access. However | the time of access.  However the dates for | available at the time  of access. The dates |
|  | the time of access. |  | the dates does not | some files does not | on the files follows |
|  |  |  | follow the required | follow the required | the required |
|  |  |  | deadline. | deadline. | deadline. |
| **Creating a** | The Class diagram | The Class diagram | The Class diagram | The Class diagram | The Class diagram |
| **Class diagram** | does not represent | and design pattern | and design pattern | and design pattern | and design pattern |
| **and design pattern selection (30 marks)** | the required solution (contains generic or non- related classes  such as admin), the design pattern | represent the required solution but in a very general and incomplete way.  Required classes in | represent the required solution in a partial way. A few  required classes in the design are not | represent the required solution in a satisfactory way. Most  required classes are declared. | represent the required solution in an excellent way. All  required classes are declared. |
|  | suggested is not | the design are not | declared. |  |  |
|  | suitable for the given | declared. |  |  |  |
|  | problem. |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Creating a Prototype User Interface and Usability Testing**  **(20 marks)** | No prototype were available or the measurement for the usability testing is not clear. | The prototype cover minimalist and trivial design (such as login) and the measurements for the usability testing are not clear. | The prototype cover adequate design and several measurements for the usability testing are not clear. | The prototype cover good design and most measurements for the usability testing are clear. | The prototype cover excellent design and all measurements for the usability testing are clear. |
| **Discuss the ethical issue related to the software**  **(20 marks)** | There is no discussion on the ethical issue or only the theories are pasted back for this component. | There is an attempt to discuss on the ethical issue but no critical  analysis was done | There is an attempt to discuss on the ethical issue with some critical analysis was done | There is an attempt to discuss on the ethical issue with good critical analysis. | There is an attempt to discuss on the ethical issue with excellent critical analysis. |

**Task 3: Creating a Class diagram and design pattern selection**

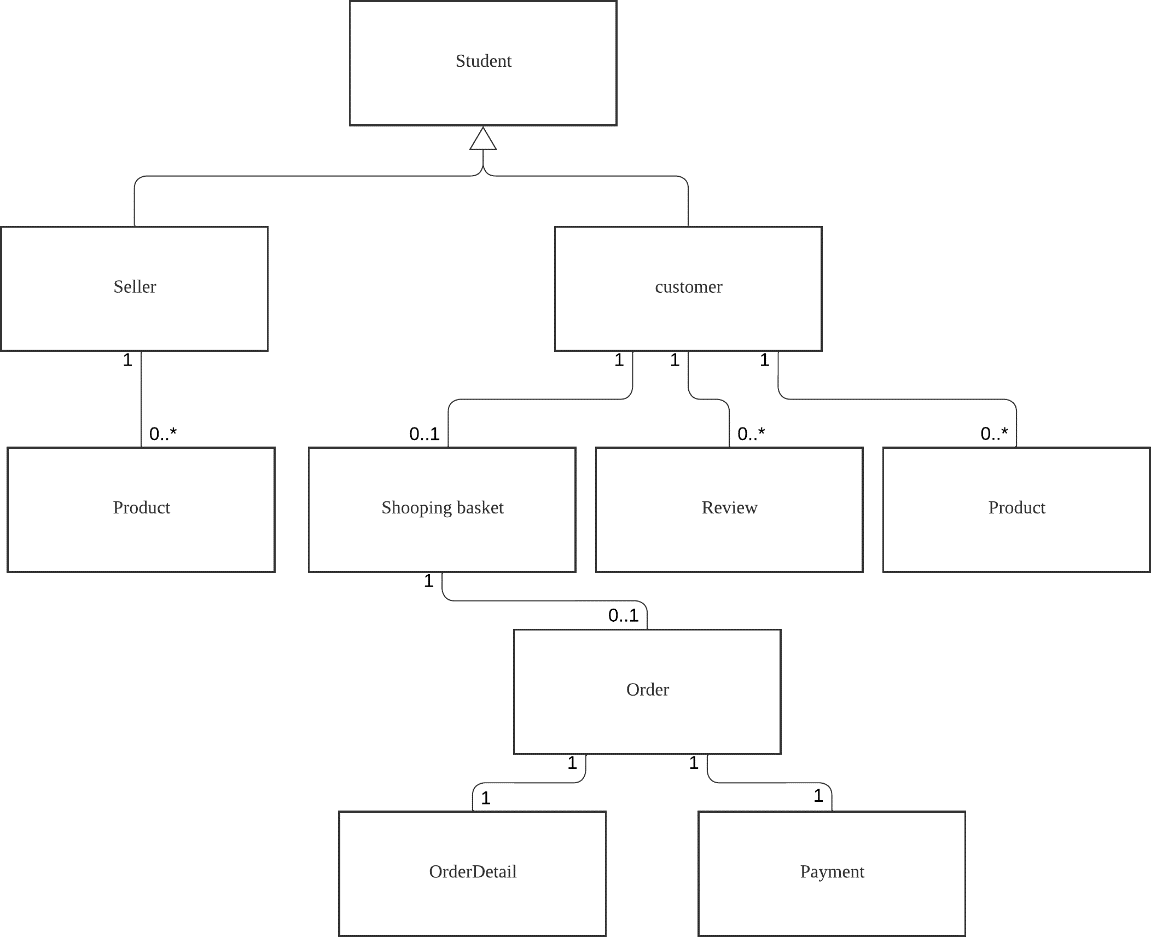


Figure 1: Simple Class Diagram

Base on the figure 1 shown above, the system consists of several classes that interact with each other to facilitate the selling and purchasing process. The Student class represents a student in the system. It has the responsibility of providing information related to the student, such as attributes for personal details and operations for performing student-specific actions. In the system, the student can choose to be the seller or the customer or both. The Seller class represents a seller in the system. It is responsible for providing information specific to a seller, such as attributes for seller details and operations for performing seller-specific actions. The Seller class establishes a one-to-zero-or-many relationship with the Product class, allowing a seller to have zero or multiple products for sale while each product is associated with only one seller. This relationship allows the seller to list and manage their own products.

The Customer class has several relationships. The Customer class represents a customer in the system. It is responsible for providing information related to the customer, such as attributes for customer details and operations for performing customer-specific actions. The Customer class has an updated one-to-zero-or-many relationship with the Review class, enabling a customer to write zero or more reviews to provide feedback on products or services. The Customer class also has an updated one-to-zero-or-many relationship with the Product class, allowing a customer to purchase zero or more products.

The ShoppingBasket class represents a customer's shopping basket and is responsible for managing the items added for purchase. It is associated with the Customer class, allowing customers to have zero or more shopping baskets. Moreover, there have a one-to-zero-or-one relationship in between the ShoppingBasket class and Order class to show that the shopping basket can be converted into an order or the customer can choose to stay at the basket without convert it into an order.

The Order class has an updated one-to-one relationship with the Payment class, representing the payment made for the order. It also has an updated one-to-one relationship with the OrderDetail class, which contains the specific details of the order.

These classes and their relationships form the foundation of the system, allowing sellers to list their products, customers to browse, purchase, and review products, and manage their shopping baskets. Orders are created from shopping baskets, and the associated payments and order details are tracked for efficient processing and fulfillment.

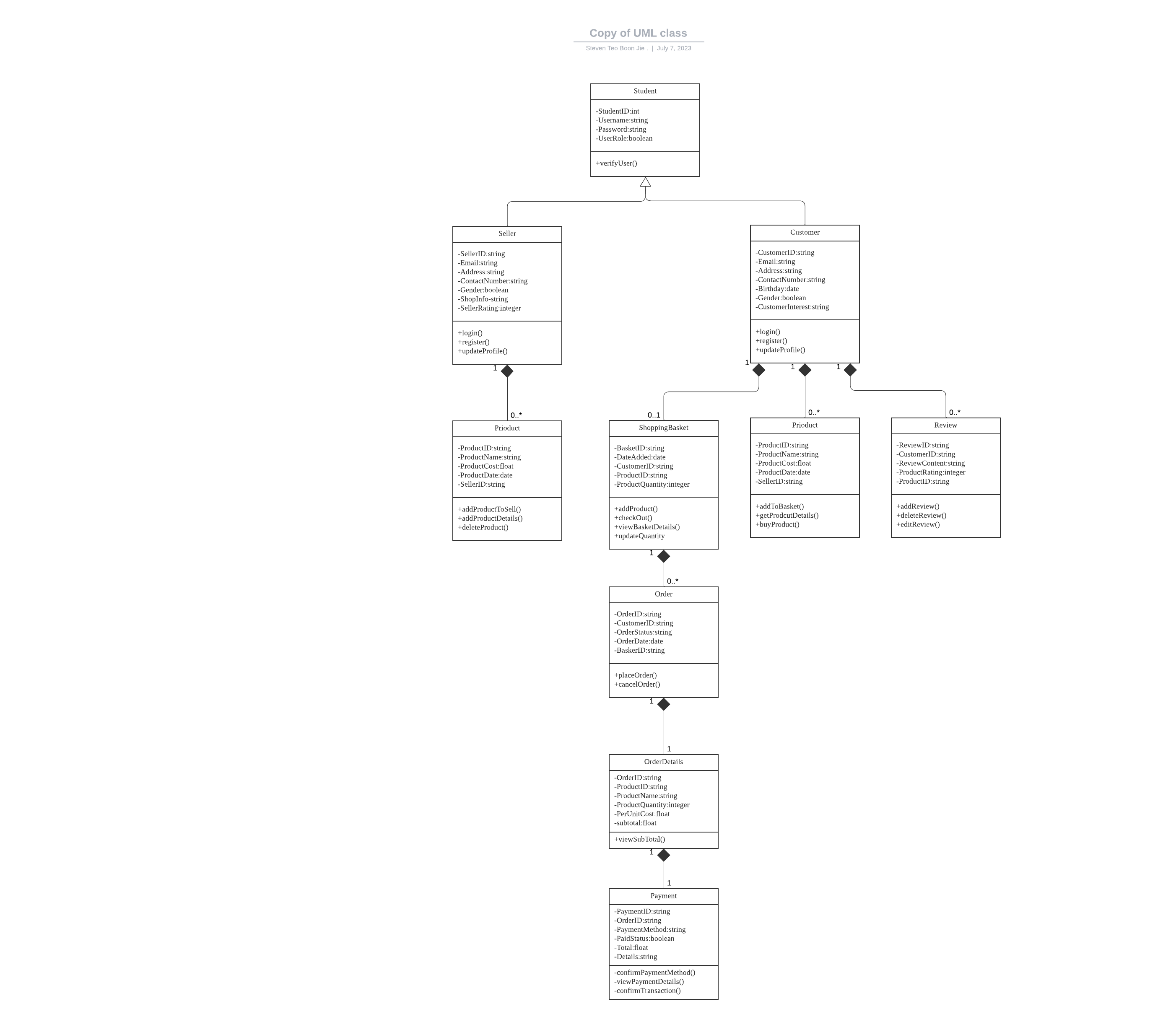


Figure 2: Design Pattern UML Diagram

The Composite pattern is considered well-suited for systems with hierarchical structures, where objects can be organized in a tree-like manner. In the system, multiple areas have been identified that exhibit hierarchical relationships, such as order details, product, shopping basket and review hierarchies. With the Composite pattern, these hierarchical structures can be effectively represented and managed.

Objects, both individual and composite, can be treated uniformly through the application of the Composite pattern. Various entities exist in the system, including products, reviews, order details, and shopping basket, which can function as standalone objects or components within composite objects. By applying the Composite pattern, common operations and behaviors can be defined that are applicable to both individual objects and composite objects. This promotes code reusability, simplifies client code, and reduces complexity.

Flexibility and scalability are provided by the Composite pattern when dealing with complex structures. As the Student Business System evolves over time, the possibility of adding new entities or introducing nested relationships between components arises. The Composite pattern allows for the seamless integration of new components into the existing structure and supports dynamic changes without impacting the overall system. This flexibility ensures easier adaptation to future requirements and maintenance of the system.

Client code is simplified through the application of the Composite pattern, as the composite structure can be interacted with without explicitly distinguishing between individual objects and composite objects. This simplification improves code readability and maintainability. Clients can perform operations on the composite object, and the pattern ensures consistent application of these operations to all components within the structure.

Recursive operations within the composite structure are supported by the Composite pattern. Scenarios requiring traversing nested review hierarchies or performing calculations on order details necessitate recursive operations. The Composite pattern provides an efficient mechanism for performing these recursive operations, ensuring consistency and accuracy throughout the structure.

By incorporating the Composite pattern into the Student Business System, the benefits of effectively representing hierarchical structures, treating objects uniformly, providing flexibility and scalability, simplifying client code, and supporting recursive operations can be realized. These advantages contribute to the improved flexibility, maintainability, and extensibility of the system over time.