Coursework 2:

Hrishikesh Srinivasan

S1913948

&

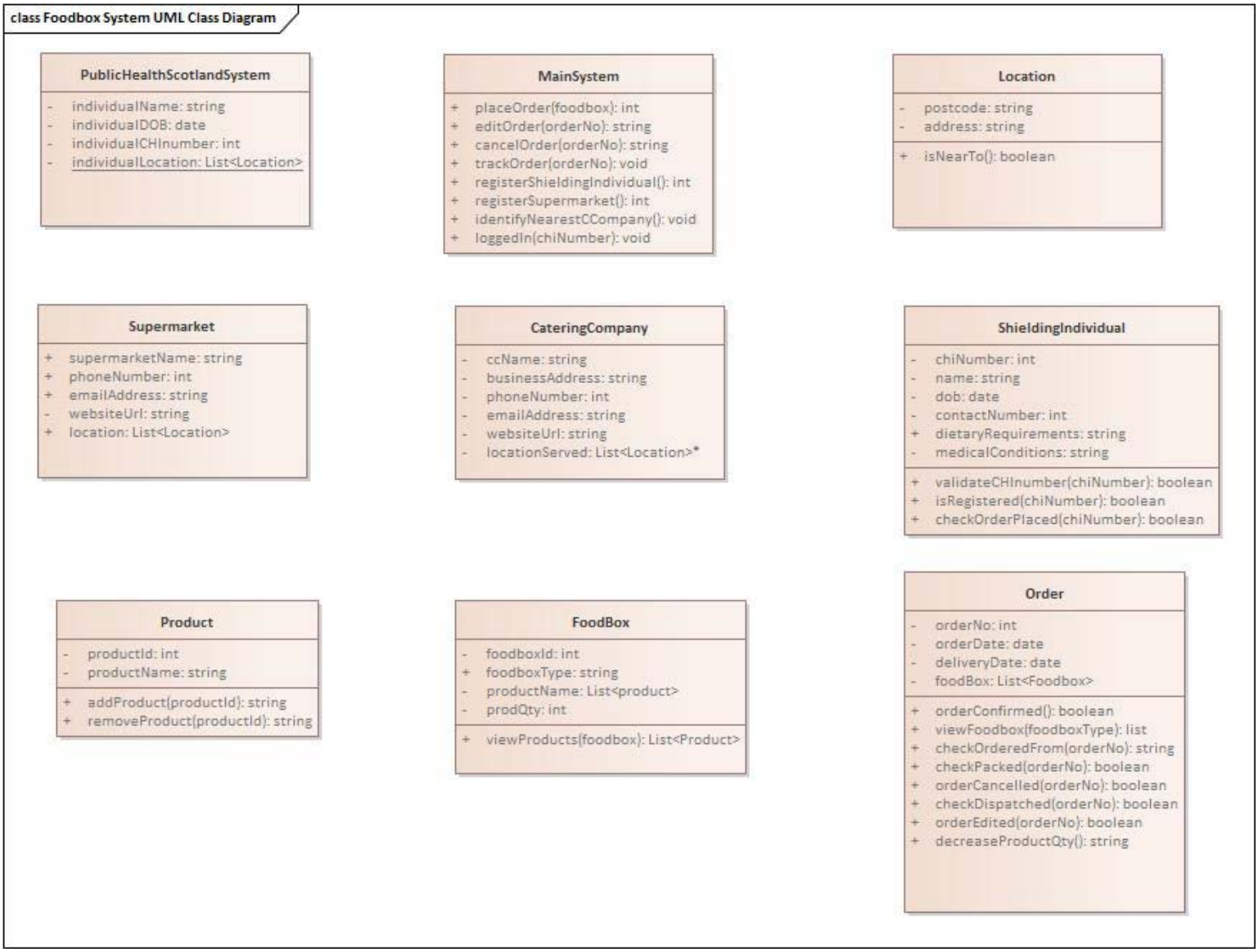
Jatin Soni

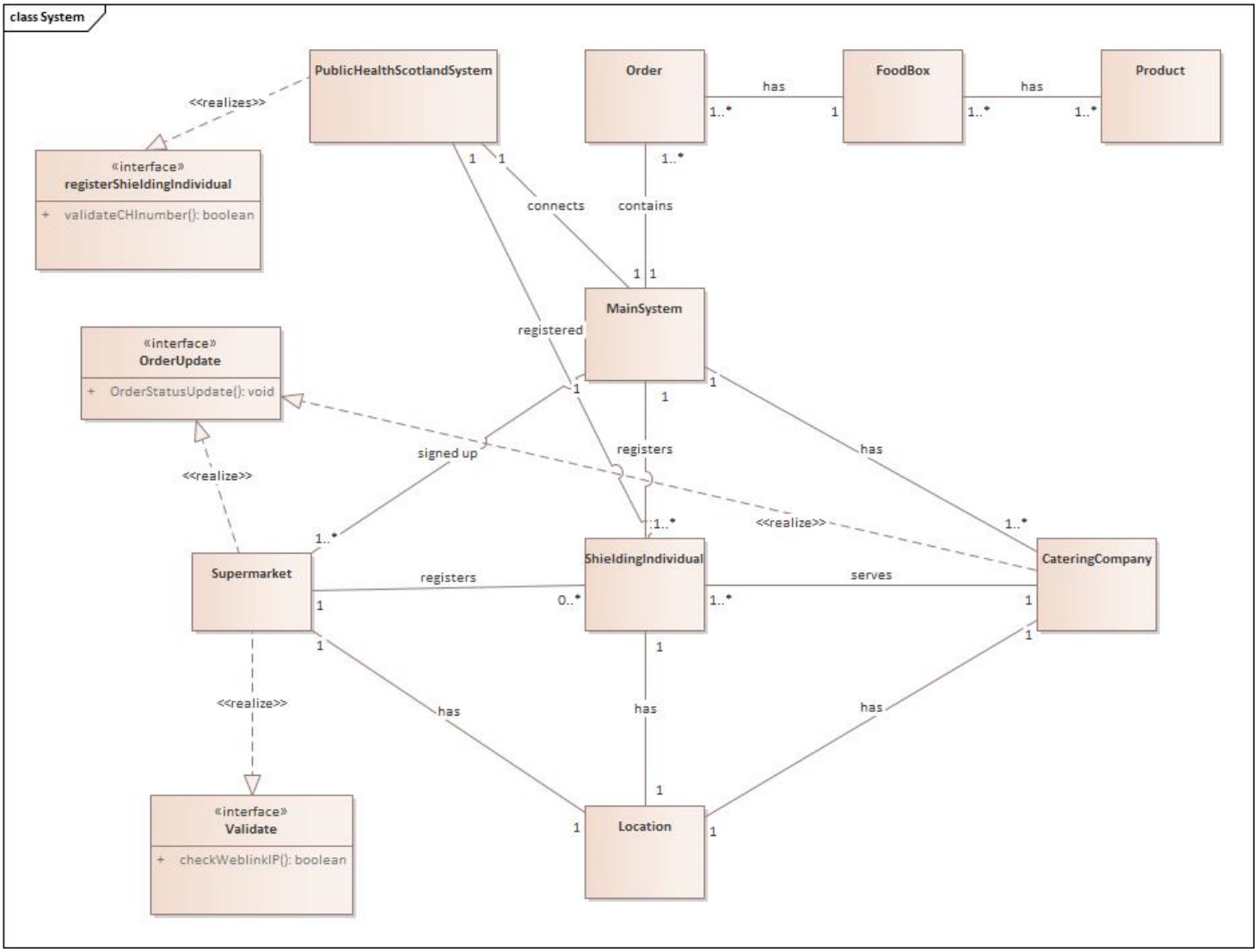
S1851999

3.1.1 The system architecture

The type of architecture that would be considered is distributed architecture - service orientated architecture. This is because service orientated architecture allows for easier scalability as increases in demand and tends to be resilient to failure. Services are stateless which means that they can be distributed, replicated, and migrated between servers. As mentioned in the brief that the customer is foreseeing plans to expand the system allowing for deliveries of other goods and services such as first aid kits, medication and provision of emergency services which would be achievable using this architecture due to services being stateless and with many servers being able to provide services meaning a unit of functionality. As we know that the system will be used over the web and will be interacting with other systems like the supermarket system, the use of a service orientated architecture would mean that it be easier to scale if demand continues to increase and resilient to failure meaning if one of the servers were to fail then it would affect rest of the other servers.

3.1.2 UML class model

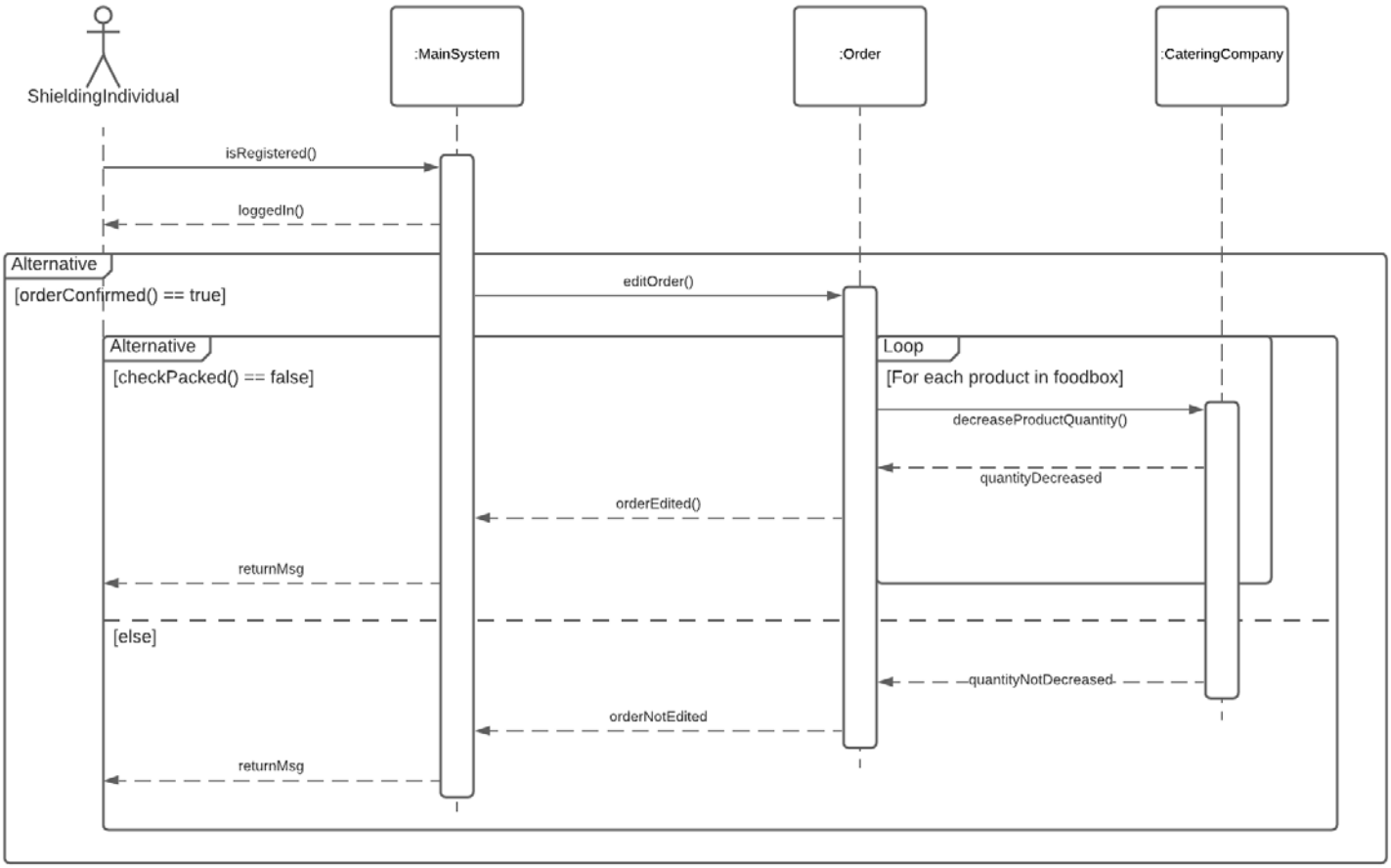



class System
PublicHealthScotlandSystem
Order
FoodBox
Product
has
has
1..*
1
1..*
1..*
<<realizes>>
«interface»
registerShieldingIndividual
+ validateCHInumber(): boolean
1
1
connects
contains
1..*
1 1
MainSystem
registered
«interface»
OrderUpdate
+ OrderStatusUpdate(): void
1
1
1
signed up
registers
has
<<realize>>
<<realize>>
1..*
1..*
1..*
Supermarket
registers
ShieldingIndividual
serves
CateringCompany
1
0..*
1..*
1
1
1
1
<<realize>>
has
has
has
«interface»
Validate
+ checkWeblinkIP(): boolean
1
1
Location
1

3.1.3 High-level description of the UML class model

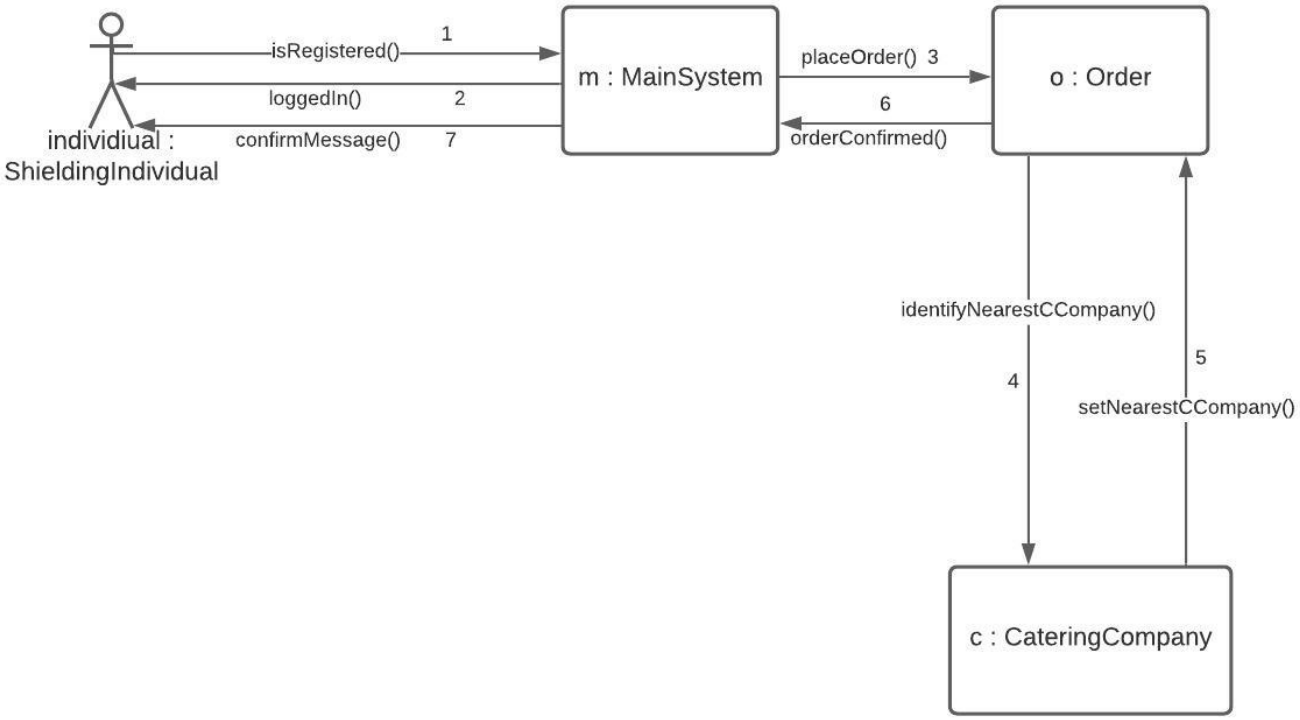
Modelling the UML class diagram required additional attributes, operations, and associations to other class objects in the diagram. In deciding on the classes to be used in the design, we have strived to keep to the object-oriented design principles of abstraction, encapsulation, inheritance, and polymorphism. One such design decision was the inclusion of a MainSystem class. The purpose of this class is to abstract away and encapsulate the details of performing most of the operations within the system such as searching and filtering. The MainSystem class would again act to abstract the details of doing such a way and encapsulating all methods for doing so. As seen in the UML class diagram, all the methods in the MainSystem class are static. A design we considered was to make use of controller classes instead of having something like a main system class. However, this would not be as effective as its overarching complexity would result in low cohesion and high coupling which contradicts our design philosophy. We also implemented a couple of interfaces in our model such as validate interface which is involved in checking if the web link provided is valid or not. Another interface included is the identifyNearestCateringCompany interface which deals with identifying the nearest catering company to the shielding individuals A particular ambiguity that was discussed in the requirements document was the lack of procedural details when the shielding individual chooses to order from the supermarket as they are not able to enquire to the system about their order and obtain status of their order. This impacted on our design as it meant that we tried to include an interface that would be handling the updates that would have been sent from the catering company or the supermarket.

3.1.4 UML sequence diagram

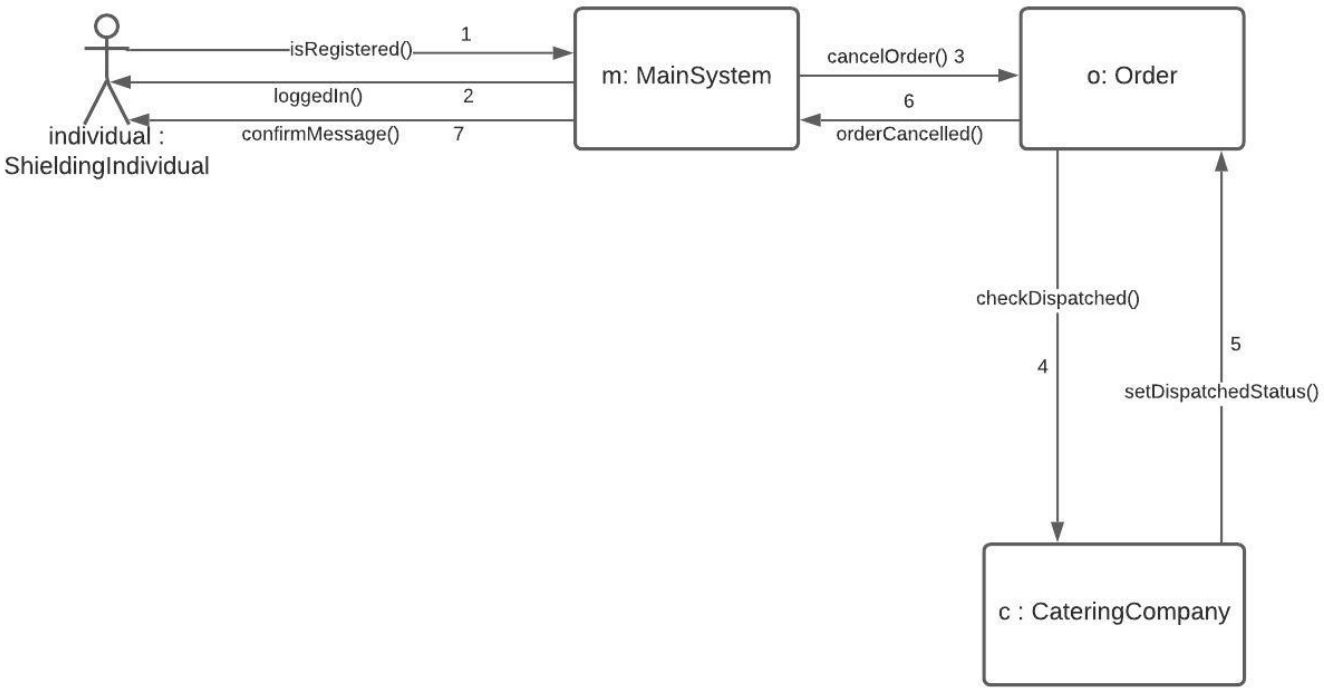


3.1.5 UML communication diagrams

Place Order case:



Cancel Order case:



3.1.6 Conformance to the requirements

We found that there were not really any major issues or apparent divergences with the requirements documents.

3.2 The software development

1) Plan driven software development process is handling design more like in this coursework. This is due to the heavy use of modelling and notation such as UML class, communication, and sequence diagrams in this coursework which are one of the characteristics of plan driven software development process. Another characteristic of plan driven development process is that design as requirements and activities were seen as being separate stages from each other in the software development which is the case for this coursework as we did each stage in turn, for example we did the requirements stage earlier and now we are focussing on the design stage and doing this in turn rather than in parallel to one another. Also, as the outputs from design mainly, UML diagrams mentioned earlier will be then used to plan the implementation.

2) Agile software development process handles design in the sense that it focuses on design and implementation meaning producing 'working software' after each iteration. In this development process, no formal, detailed design documents are produced because they are seen as a waste of time. Instead, informal documentation is generated automatically by the programming environment. This is different to the first in terms of outcomes as outputs of design may not be in terms of documentation but rather may be reflected later during engineering when implementing the code. Also, the use of UML diagrams like communications, sequence and class diagrams are seen as informal and considered important for facilitating team communication compared to plan driven where they are considered being important for documentation.

3.3 Software Engineering – Self Assessment

1) Overall, we think that the type of software architecture we identified was suitable based on the given system descriptions and the customer's plans for the future. The strength of our solution is that we believe that we have provided with enough reasoning as to why we considered this type of architecture to be most suitable. A difficulty that we encountered when writing up our solutions was that we initially were not able to decide confidently as to which software architecture would be most suited. To address this difficulty, we had to read through the lecture slides and their corresponding reading along with online resources to get a better understanding and use the system description to decide on the type of software architecture that would be most appropriate. To improve on our solution, we could have tried to add more detail to the points we gave, and use more information provided in the system description to back up the point we are making.

2) We think that the UML class diagram notation we have designed is correct in the sense that it considers the use of the 5 given cases along with the system description and any future updates being made to it. The strength of our solution and the thing that we are proud of is that we believe that we have used all the given cases based on the system description for out class diagram notation. A difficulty that we encountered when designing our diagram was trying to come up with the possible operations that we would need to have in our diagram and which we could consider being important. To address this difficulty, we had to read through the system description again along with the section where it mentions the customers future plans and we also had to go back on to the previous coursework to have a look at the given use cases in detail.

3) We feel that the static design represented in the class model was of good object orientated design and good quality. The strength of our solution and the thing that we are proud of is that we believe that we have considered things like design principals and patterns which assist in

ensuring a good object orientated design. A difficulty that we encountered initially when designing our class model was that our model did not use good object orientated design. To address this difficulty, we had to read through the lecture slides and their corresponding to obtain a better understanding of the design patterns and principals which we would then use in our implementation of the model.

4) We would say that the explanations and justifications that we gave in the high-level description of the class models were satisfactory and sufficient. The strength of our solution and the thing that we are proud of is that we believe that we have provided with enough justification and detail in terms of as to why we made these design choices in the high-level description of the class model. A difficulty that we encountered when writing up our explanation to this section was that we were at first unsure of how exactly the incompleteness and other ambiguities that we mentioned in the first coursework would have an impact on our design. To address this difficulty, we had to discuss among ourselves and come up reasons as to why these assumptions that we mentioned in the previous coursework have impacted on our design.

5) Overall, we think that our UML sequence diagram was fairly accurate in terms of displaying the sequence of actions that would occur in the edit food box scenario. The strength of our solution and the thing that we are proud of is that we feel that the notation we use is correct and that our diagram has sufficient detail through looking at the system description from the previous coursework and clearly shows sequence of the edit food box use case. A difficulty that we encountered when coming up with the design of our sequence diagram is that we were unsure of as to whether we should try to include the preconditions of the edit food box use case. To address this, we had to go back to our solutions for the previous coursework and have a look at the edit food box use case scenario, look back at the system description in detail and what exactly did the question wanted us to design.

6) We think that the interactions that we have represented in our sequence diagram made use of high-quality object orientated design. The strength of our solution is that we made correct use of notation to

represent the sequence diagram along with showing the interactions between each of the classes. A difficulty that we encountered when designing our sequence diagram was that we were not too sure as to whether we should consider including the supermarket class in our diagram or not. To resolve this difficulty, we had to discuss with each other and decide on whether it would be valid to include this class in our diagram based on the reviewing the system description and our use case scenario from the previous coursework.

7) We think that our UML communication diagrams are fairly accurate in terms of displaying communication that would occur between classes for the place order and cancel order scenario. The strength of our solution and the thing that we are proud of is that we feel that the notation we use is correct and that our diagram has sufficient detail through looking at the system description from the previous coursework and shows communication of the place order and cancel order use case. A difficulty that we encountered when coming up with the design of our communication diagrams is that we were unsure of as to what classes would we want to most include such as the supermarket class, and which ones should we leave out. To fix this, we reviewed the use case scenarios for each of the given scenario from the previous coursework to get an idea of which classes would be most important to use and base our diagram from that.

8) We feel that the interactions that we have represented in our communication diagrams made use of high-quality object orientated design. The strength of our solution is that we made correct use of notation to represent each of the communication diagrams along with showing the interactions between each of the classes. A difficulty that we encountered when designing our sequence diagram was that we were not too sure as to whether we should consider including the supermarket class for one of our diagrams. To resolve this difficulty, we reviewed the system description and our use case scenario from the previous coursework to gain a better understanding of what classes to include for our diagrams.

9) Overall, we feel that our solutions between cw1 and cw2 were fairly consistent. One of the main strengths of our solution is that we have ensured that we examine in detail the systems description along with our own solutions from the previous coursework which allowed us to ensure that our solution for each section of this coursework is consistent and there are no contradictions.

10) We feel that all our solutions are consistent with each other among each of the questions. This is the case because for each of the solutions for our questions we have considered to ensure that the answers to the previous questions are accounted for while we are answering the next questions to make sure that there is no contradiction in our answers, or we have not missed on any details. Some examples of these in our coursework are that while drawing the communications and sequence diagram we have considered what we wrote for the previous questions and have made sure to include all possible classes and methods from the class model and ensured not to miss out on any details.

11) Overall, we believe that our solution for this was very well formulated and provided justification to the question being proposed. The main strengths of our solution would be that we have provided with valid reasons that help to justify the points that we are trying to get across. We felt that that the main weakness in our solution was the fact that we provided only a certain number of reasons to support the points that we are trying to make which could have been improved and dealt with by reading about the software development processes from other places rather than what was just in the lectures and required reading. Thus, a thing to improve upon would be to try and include more points in order to justify the reasons we were giving.

12) We feel that we contributed fairly to these tasks and were honest in terms of how we approached each section of the work. We came up with plans on how we could best approach the given tasks and collaborated frequently with each other on Teams to discuss and work on the tasks.

We feel that we were fair and honest in the self-assessment that we gave of our work and were honest in considering those parts of our answers which we could have improved on or were not particularly pleased about.