Coursework 1:

Hrishikesh Srinivasan

S1913948

And

Jatin Soni

S1851999

3.1 Identify stakeholders

Stakeholders identified are:

1. Scottish government – The government is responsible for the functioning of this system and the funding as they will have to subside free meals. The system would affect the Scottish government as it would have to make its order data accessible to allow the government to coordinate as COVID restrictions and policies may change.
2. Catering companies – As the companies will get more orders, they will improve the quality of the food. The system would affect catering companies as it would need to be in regular contact with the catering companies system to receive updates regarding the delivery and any delivery status changes.
3. Supermarkets – As they will tend to get more orders, they will have to increase and update their stocks quite often. They may have to act quicker if any issues were to arise.
4. Public Health Scotland – As they should be able to act quicker and solve issues more quickly due to more individuals being involved with the system. The system also makes use of their CHI number to keep records of individuals who are shielding and retrieves further information regarding their location.
5. Users of the system(Shielding individuals) – As the system allows those who are shielding to order food boxes with catering companies or supermarkets once a week with these orders being free of charge making it easier and safer for the users as they can place their orders from their home locations without the needing of having to go out physically.
6. Other organisations – As the system is providing confidential information like its ordering data so that these organisations can produce statistics and reports.
7. Local councils – The system would affect local councils as local councils would be able to make use of the system to gather statistical information like the number of shielding individuals in their region and gather other details that may be of their use to produce reports.
8. External delivery service provider used by catering companies – The system would affect the external delivery service provider as not all catering companies will be able to offer in-house delivery to shielding individuals and so they will have to make use of an

external delivery service provider which would mean that the system of the delivery service provider would have to communicate with the food box delivery system to provide updates regarding the delivery of food boxes.

9. Carers – As there will be certain groups of individuals like the elderly that would be shielding and would be looked after by carers. The system would affect the carers as they would have to order food boxes through catering companies or via supermarkets on behalf of these individuals making it safer for them as they are able to place food orders remotely without the need of having to go out.

3.2 Describe system state

1. Regarding the users, the system needs to keep track of:

1.1 The user's CHI number.

1.2 full name and DOB.

1.3 Hold their address.

1.4 Hold their phone number.

1.5 Any dietary requirements.

1.6 Any medical conditions.

2. Regarding the food box orders, the system needs to keep information of:

2.1 Whether the order has been requested from a catering company.

2.1.1 Requirements of any dietary restrictions e.g., vegan, vegetarian.

2.1.2 Any amendment option selected to the food box's contents.

2.1.2.1 Listing of the box's contents which contain the name and quantity of each product.

2.1.2.2 The updated quantity (only reduced) of each product.

2.1.3 The selected delivery date and time.

2.1.4 The delivery status of the order.

2.2 Whether the order has been requested to be placed from a supermarket.

2.2.1 The user's postcode.

2.2.2 Confirmation of ordering from the supermarket.

2.2.3 Status of order changes.

3. Regarding catering companies, the system needs to keep hold of:

3.1 Database using their business names.

3.2 List of default food boxes one for each type of dietary restriction.

3.3 Changes to the delivery status.

4. Regarding supermarkets, the system needs to keep information of:

4.1 Their business name.

4.2 Their phone number.

4.3 Their website URL and address.

4.4 Any food box orders.

5. Regarding the delivery service provider, the system needs to keep information of:

5.1 Updates on the delivery status.

6. Regarding the status of the boxes and orders, the system should keep track of:

6.1 The description and quantities of products of each box per order as modified by the user.

6.2 Changes in delivery as provided by the catering company system. Possible various states could be order received, order being processed, order amended, order cancelled, order dispatched, and order delivered.

3.3 Describe use cases

Use case name: Place order.

Primary actor: Users like shielding individuals who are using the system.

Supporting actors: Catering companies, Supermarkets.

Summary: Shielding individuals or carers places order to request for a food box delivery.

Precondition: User has already registered on the system and are identified as being shielding individuals.

Trigger: User pressing the order button after populating the mandatory fields such as type of box, ordering from either a supermarket or catering company.

Guarantee: If the order process is a success then the system will ensure that the order is placed at the respective supermarket or food catering company and notifies the user that the order has been placed and provides a unique reference number.

Main Success Scenario:

1. User is validated to be registered with the system.

2. User decides on either ordering a preassembled food box from the catering company or ordering a box from the supermarket.

3. User can choose to amend food box order contents and its quantities (only reduction/removal allowed) when ordering from catering company.

4. User selects a preferable date and time for delivery.

5. User presses the place order button.

6. Once an order is placed successfully, user receives confirmation of the order.

Extensions

2a. User decides to order from a supermarket.

.1 System provides possible supermarkets based on the user's postcode, delivery coverage and delivery time.

.2 System provides links to the dedicated food box web pages of the supermarket

6a. Once an order has been placed from the supermarket, the supermarket notifies the system of the success.

Stakeholders & Interests:

Shielding individuals (users) – These are interested in placing an order for a food delivery box.

Catering companies – These are interested in the number of orders received to accommodate for stock and staff.

Supermarkets – These would also be interested in the number of orders that they have received to allow them to accommodate for stock and staff.

External delivery service provider – These would be interested in accommodating for staff based on the quantity of orders received.

Use case name: Cancel order.

Primary actor: Shielding individuals who are using the system.

Supporting actors: Food box delivery system, supermarkets, catering companies.

Summary: User decides to cancel their request for a food box delivery.

Precondition: User must be registered to the system. User must already have placed an order. Order must not have been dispatched for delivery.

Trigger: User pressing the cancel order button.

Guarantee: If the cancel process is successful then the order placed will not be delivered to the user's location. User receives a conformation that their order has been cancelled.

Main Success Scenario:

1. User is logged in on the system.

2. User enters order number/details to view their placed order.

3. User selects the order that they wish to cancel if it has not already been dispatched.

4. User presses the cancel button to cancel the order that they have placed from the catering company.

5. User receives some sort of notification as verification that their order has been cancelled.

Extensions

4a. User may not be able to cancel their order due to system malfunction.

5a. User may not receive a notification to confirm their order.

Stakeholders & Interests:

Shielding individuals (users) – These are interested in cancelling their order for a food delivery box.

Catering companies – These are interested in knowing cancelled orders to reuse their stock and staff for the next order.

Supermarkets – These would also be interested in knowing orders that may have been cancelled so that they are able to reuse their stock and staff for the next order.

External delivery service provider – These would be interested in knowing cancelled orders so they can use staff and stock for other orders.

Use case name: Edit food box.

Primary actor: Shielding individuals who are using the system.

Supporting actors: Food box delivery system, supermarkets, catering companies.

Summary: User has decided to edit the order they have placed.

Precondition: User must be logged in the system. User must have already placed an order. Order must have not been dispatched.

Trigger: User pressing the edit order button to make changes to the order that they have made.

Guarantee: If editing the order is successful then the order will be updated, and a conformation will be given to the user.

Main Success Scenario:

1. User is logged in on the system.

2. User enters order number/details to view their placed order.

3. User selects the order that they wish to edit if it has not already been dispatched.

4. User presses the edit button to edit the order that they have placed from the catering company.

5. User receives some sort of notification as verification that their order has been changed.

Extensions

4a. User may not be able to edit their order due to system malfunction.

5a. User may not receive a notification to confirm changes made to their order.

Stakeholders & Interests:

Shielding individuals (users) – These are interested in editing their order for a food delivery box.

Catering companies – These are interested in knowing the changes to content and quantity of the orders to reuse their stock.

Supermarkets – These would also be interested in knowing changes made to the content and quantity of the orders that may have been cancelled so that they are able to reuse their stock.

Use case name: Registering to use the system.

Primary actor: Shielding individuals who want to use the delivery system.

Supporting actors: Public Health Scotland's electronic system.

Summary: User decides to register for food box delivery system.

Precondition: Users must have a valid CHI number. Users are recognised as shielding.

Trigger: User decides to register on the system.

Guarantee: Users receive some sort of confirmation that they have been registered on the system.

Main Success Scenario:

1. User has valid CHI number.

2. User is recognised as being an shielding individual.

3. User enters their required details on the registration forum/screen.

4. User presses the register button to register on the system.

5. User receives some sort of notification as verification that have been registered on the system.

Extensions

1a. User may not have valid CHI number, or their CHI number is not being recognised.

2a. User is not recognised as a shielding individual.

3a. User may have entered their details incorrectly.

4a. User may not have received confirmation of their registration.

Stakeholders & Interests:

Shielding individuals (users) – They are interested in registering successfully to make use of the service.

Public Health Scotland electronic system – They are interested in keeping users records up to date.

Scottish Government – They would be interested in providing estimates to catering companies/supermarkets so that they can plan accordingly.

Use case name: Order confirmation.

Primary actor: Food delivery system and shielding individuals who want to use the delivery system.

Supporting actors: Supermarkets, catering companies and external delivery service provider.

So, the main scenario would be that users will have placed their orders either from supermarkets or the catering companies and then they will receive a message/notification to advice that the order that they have placed has been confirmed. From their onwards users will be able to view updates about the status of their orders. Alternative scenario would that the user has not received a notification advising that their order has been confirmed.

Use case name: Order delivery.

Primary actor: Food delivery system, supermarkets, catering companies and external delivery service provider.

Supporting actors: shielding individuals who want to use the delivery system.

So, the main scenario would be that the food box has been delivered and updates regarding the status of the delivery has been recorded in the food delivery box system. An alternative scenario would be that the food box has not been delivered to the user. Another alternative scenario would be that the food delivery system has not been able to record the different status of delivery.

Use case name: Specific dietary food box not available.

Primary actor: shielding individuals who want to use the delivery system and food delivery system.

Supporting actors: Supermarkets, catering companies.

The main scenario would be that as the quantity of food boxes available for each specific dietary is finite and may run out quickly based on demands. In such a scenario, the user won't be able to select that particular food box based on their dietary requirements. An alternative

scenario would be that the user may be forced to change their preference and select a different dietary food box.

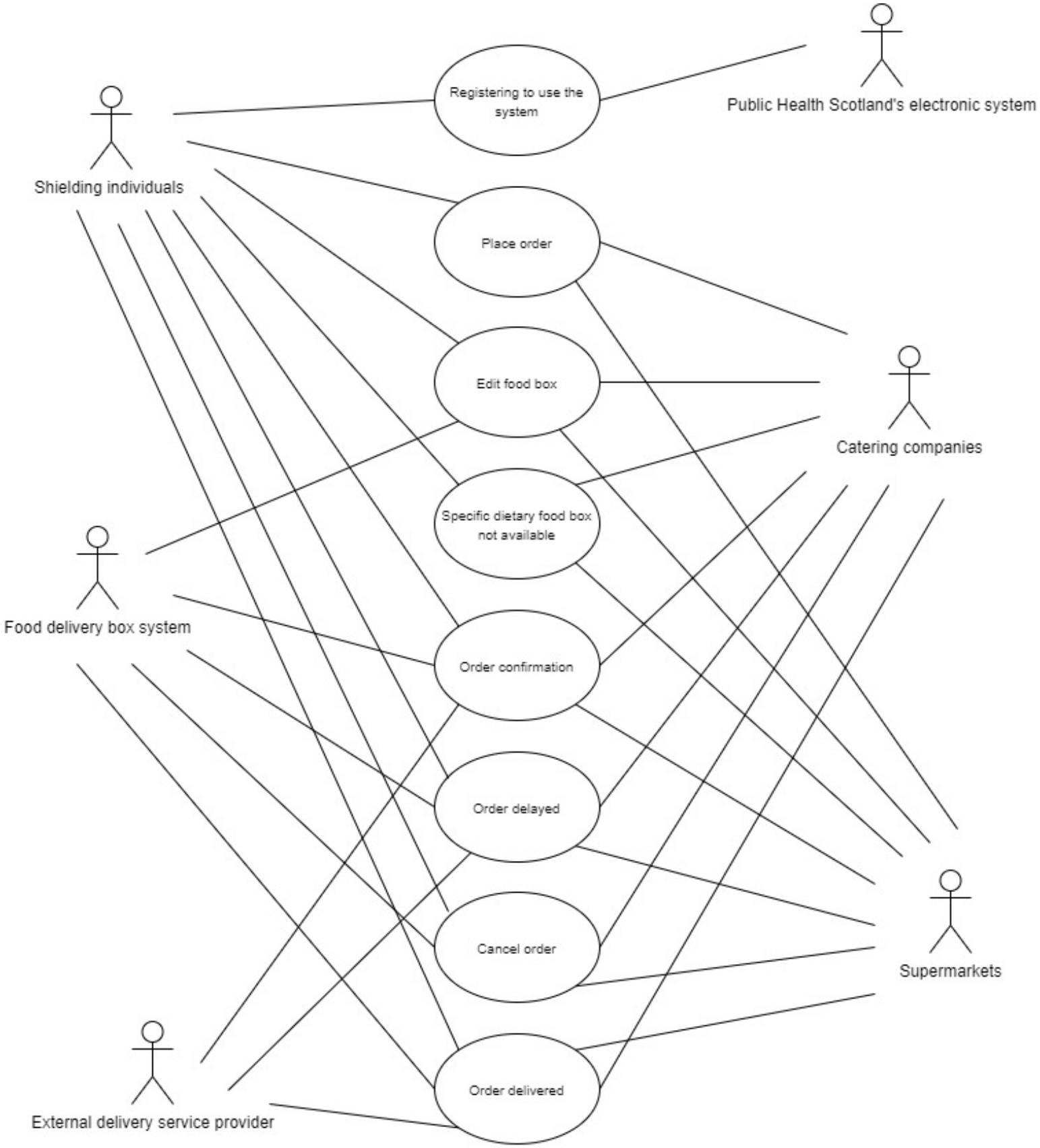
Use case name: Order delayed.

Primary actor: Food delivery system, supermarkets, catering companies and external delivery service provider.

Supporting actors: Shielding individuals who want to use the delivery system and food delivery system.

The main scenario would be that the placed order has been delayed due to many reasons such as adverse weather conditions, workers being contracted with COVID-19. The catering companies/supermarkets/external delivery service providers should notify the system of the delay in delivery. An alternative scenario would be that the order may have to be switched to a different provider to fulfil and deliver that order.

3.4 Draw use case diagram



3.5 Describe non-functional requirements

1. Security:

1.1 An example would be that the system should suggest its users to change their password on a regular basis to prevent their password from being stolen. This could be measured by the users given a date that they must have changed their password by. The security could be measured as the system could give force users to change their passwords after a set period e.g., 30 days.

1.2 Another example would be that the software should also be free from malware and anything that could cause potential harm to the computer or the device they are using. This could be assessed by ensuring that tests/checks are occurring on the server where the application is hosted which can be tracked/measured using log files.

2. Performance:

2.1 An example would be that the system should be able to allow for orders to be placed concurrently by many users at the same time. There should be no lag in the software. The performance could be measured by the system recording a timestamp of successful placing of orders.

2.2 Another example would if there are backups being run in the background, it should not hamper with the performance of the application. This can be measured through the load time of the various webpages.

3. Privacy:

3.1 An example would be that the computing professionals should decide on if a limited number of operators can access its customer details on the system or all of them.

3.2 Another example would be that the data on the systems data should not be shared with any other external sources and

should only be used according to the Scottish government requirements. Privacy could be measured based on number of incidents of data leaks.

4. Usability:

4.1 An example would be that the system should be easy to use for all its users (user-friendly).

4.2 Another example would be to include help guides with FAQs in assisting users. Usability can be measured based on how often FAQs and user guides have been visited by users.

5. Availability:

5.1 An example would be that the system could be available for use for 24 hours a day or between a specific time each day. This could be measured by its uptime through system performance logs.

5.1 Another example would be to ensure that there is minimum down time for the application i.e., only downtime for essential maintenance as it is a critical application for shielding individuals. This could be assessed and measured by the use of log files.

6. Accessibility:

6.1 An example would be that the system should be accessible to users with specific needs i.e., providing these users with options such as screen magnifier and screen reader. This could be measured by how often these tools have been accessed by users.

6.2 Another example would be to ensure that the system is also accessible to colour blind users by making sure the colour schemes and fonts being used should be readable by users

impacted by colour blindness. This could be assessed during the testing phase.

7. Platform Compatibility:

7.1 Platform compatibility is the feature that the system should be able to function on other platforms such as mobile, tablet etc. An example would be that the system should be available on IOS, Android and PC. This can be measured by number of downloads from Appstore/google play store or by the number of visits to the site.

7.2 Another example would be that the system should be compatible with other systems like the supermarket system, catering companies' system. This could be assessed based on if regular updates are being received on the system from these other systems.

8. Data retention:

8.1 An example would be that the system may be programmed to archive a user's details if they have not placed an order in a very long time. This could be measured by adherence to the GDPR policy.

8.2 Another example would be that the system could instead of retaining data at an individual level, data could be aggregated or rolled up so that long term reporting can be achieved without users being identified individually. This could be assessed by ensuring that users name, email address etc are removed.

3.6 Identify ambiguities, subtleties, incompleteness

One ambiguity is that when the user is ordering from the supermarket, the system only suggests the user possible supermarkets to order from based on their postcode and so the system is not used in the ordering process. This means that the user is not able to enquire the system about the status of their delivery from the supermarket and these details are only shared between the supermarket's system and the software system so an error message would be received to the user if they tried to enquire about an order number which is not from a catering company.

To clarify this ambiguity, there would have to be some sort of method such as allowing the user to enquire about these details through the system in which the user is able to view the status of their delivery from the supermarket.

Another ambiguity is that when catering companies are using an external delivery service provider their updates are being received through to our system but there are no details with regards to how both the systems are communicating with each other or how compatible both systems are.

To clarify this ambiguity, you would approach system architects or system designers who would advice of a possible web service/API service to make use of.

Another ambiguity is that the system makes use of a database to store the names of catering companies but there is no mention of what type of database models/schema being used.

To clarify this ambiguity, data architects would have to be approached and discussed.

Another ambiguity is that there is no mention of whether catering companies can provide all dietary type of food boxes as there maybe certain catering companies which can only provide specific dietary food boxes.

To resolve this ambiguity, catering companies would have to be contacted and discussed with. Also, the database table for catering companies need to be amended.

3.7 The software development

1. Software project-based engineering would be the better choice as the project/plan to develop a food delivery system was initiated by the Scottish government (external customer) as they wanted to support those citizens during shielding. Another reason is that project-based engineering would be the better choice because as the system would be developed based on the requirements as advised by the Scottish government. In addition, as the external customer (Scottish government) would be mainly responsible for paying for any amendments that would have be made to the system indicating the use of project-based engineering.

Product-based engineering would not be a better choice as the project of developing the delivery system was not initiated by the developer nor was it developed as a generic solution for all users, as the delivery system was only meant for those individuals that were shielding. Another reason as to why product-based engineering would not be considered is due to the fact that the life of the product would not be decided by the developers, but the Scottish government. Furthermore, the Scottish government would be mainly responsible for deciding on any changes that are required rather than the developers deciding on changes.

2. Plan driven process has been used so far in this assignment as the delivery system was specified in detail before the beginning of the implementation phase. In addition to this, as UML modelling was being used for documenting design and requirements which is one of the main characteristics of a plan driven approach. Furthermore, as critical information such as the involvement of stakeholders has been pre-collected and documented thoroughly indicating the use of a plan driven approach.

3. Agile driven process would not be a better choice because for this project, as requirements have already been pre-decided and there are no further planned changes to be made to the

requirements. Another reason is, as this system would be considered of being of high criticality since individuals are shielding and would need supplies of food frequently rather than being of low criticality, which is suggesting that agile driven process would be less suitable for this case.

5.1 Software Engineering Self-Assessment

1. Overall, we think that we were able identify accurately the critical stakeholders based on the given system descriptions and others that we considered being relevant to the study. The strength of our solutions and the thing that we are proud of is that we believe that we have provided with sufficient detail as to how these stakeholders are being affected by the system. A possible weakness in our solution would be that we could have identified more stakeholders that would have been relevant to the case study. Another would be that we could have gone into more detail for some of the stakeholders that we discussed about. A difficulty that we encountered when writing up our solutions was that we initially identified a lot of stakeholders, but they were not relevant to our case study. 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. To improve on our solution, we could have tried to identify more stakeholders that we knew would be relatable to the description given.

2. We believe that we were able to correctly identify the appropriate system states mentioned in the system description. The main strengths of our solution to this part were that we believe that we answered each of the sub-questions that we were proposed and the fact that we had use the correct formatting which was to make use of an enumerated list and this is one of the main things that we are proud of. The main difficulty that we encountered was in being able to figure out how to make use of the enumerated correctly by that we mean as to where we can introduce new levels to our list. To address this difficulty, we had to read through the system description very carefully and then try to figure out how we can number those points based on the details given in the case study. We feel that there is not really a lot we can improve on, rather a few minor things that we consider improving on are that we could have elaborated some of the points a bit more that we mentioned in our answers.

3. We feel that we were able to come with appropriate use cases that we thought were of high level. The strengths of our solution are that we believe that when writing these solutions, we have only considered the main interactions between actors external to the system and the system itself rather than the details of user interface interactions. Another strength of our solution is that we have based our solution around what the question demands of and tried to answer all those points. The main weakness of our solution that we think is that we were not able to identify more than 8 use cases which would be a reasonable amount of use cases. Thus, the thing to improve would be to consider coming up with more use cases ideally 9 or more. We could improve on this by studying the system description in depth and try to identify more high-level use cases which would be of use. The main difficulty we faced when coming up with use cases is that we thought of use cases which were not of a high level, so we had to discard them. To resolve this, we had to read through the system description and try to figure out what other high level use cases there could be.

4. Overall we thought we did a pretty good job in using the use case template to describe the use cases. The main strengths of our solution would be that our solution makes use of the points mentioned in the use case template to describe the use cases. The main difficulty that we faced when writing up a solution for this part was on determining whether what exactly our extensions would be for some of the use cases. To address this, we had to go through the system description and try to brainstorm what might happen in alternate scenarios. Again, the main weakness of our solution and the thing to improve on would be that we only came with 8 use cases, but we should have tried to come up with at least 9 or more.

5. We think that our constructed UML diagram is complete and is accurate in terms of including all the use cases and their associated actors. The main strength of our solution is that we have made sure that we include all the associated actors with the use cases and to make sure that we have ordered the use cases. The main difficulty we faced was that when drawing the diagram, it would often be the case that there would be lines which would overlap with each other

which would then make the diagram messy and less readable. To address this, we had to come up with ways in which there would be less overlap of lines between the actors such as shifting some actors on one side and some of the other depending on which use cases they connect to. The main thing that we are proud of is the fact that we think we have managed to reduce the overlap of lines in our diagram and made it more readable.

6. We feel that we have managed to describe the non-functional requirements very well as the main strengths of our solution would be that we have included more than the specified number of non-functional requirements along with ensuring that we have included at least some of the ones mentioned in the question along with trying to explain ways in which these could be measured and assessed which would be one of things that we are proud of. Some of the weaknesses in our solution would be the fact we think that we could have expanded on some points more which we could have done by studying about these requirements and then coming up with more ways in which they would be relevant with the system description. The main weakness and the thing that we are less pleased about is that in our solution we only came up with 2 levels of the enumerated list which we could have dealt with if we had gone through the lecture slides and associative reading to find out more points.

7. For this, we think that we have gone into quite detail while trying to answer this question. The main strengths of our solution are that we think we have managed to answer all the sub-questions that were given to us. The main difficulty that we had while answering this question is that we were not certain if we were able to assume things when trying to solve these ambiguities as some of the details were omitted in the system description. Thus, the main weakness in our solution is that as the solution is based on our assumptions it maybe the case the solution maybe incorrect.

8. We strongly believe that our solution is accurate in justifying our choice of selection. The main strengths of our solution are that we have included enough detail by examining lecture materials and their associated reading in our answers to help justifying the

reasons as to why we support those claims. Some aspects to improve on would be possibly to try to come up more reasons which would help reinforce our claims which could be achieved by going through the system description in detail to try to come up with more reasons to support our answer. Furthermore, we would also consider the point mentioned above as being a weakness for our solution as in some parts of our answer we feel that we could have added more to it to better justify our reasoning. However, the aspect that we are pleased about in our solution is the fact that we believe that we have answered all the sub-questions that were given to us.

9. 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 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 and to improve upon 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.

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 use case diagram we have considered what we wrote for the previous question and have made sure to include all the possible use cases and actors and not to miss out on any details. Another example would be that when we talk about what type of software engineering processes and software development processes, we should use for this work we have made sure that our chosen software development process is not contradicting with its software engineering process as plan driven is used in project based

engineering and agile processes are generally used in product-based engineering.

11. We believe 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 and other platforms like Facebook Messenger 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.

5.2 Professional Issues Self-Assessment

1. We think that our answers were successfully in answering the questions proposed to us. This it the case because for our solution we looked to answer the question to us to the best of our ability.

2. We believe that our solutions are quite clear. We have tried to answer the questions being proposed using the ACM code of principals to ensure that our explanations are being concise.

3. We feel that we have managed to keep an appropriate structure when writing our blogs. Throughout our blogs, we think that there is some structure which helps us in formulating our arguments.

4. We think that are for our blogs have not made use of o a great use of other sources in aiding us to answer the questions. That being said, we feel that we could have tried to make use of other resources which would have helped us in improving the arguments that we mention.

5. From the arguments presented in our blog, we believe that we have provided valid arguments that help us in answering the question being given to the best of our ability.

6. In our solution we believe that we have not presented enough counter arguments or alternative views. We could have mentioned some more counter arguments to provide the reader with other viewpoints.

7. Based on our blog, we would say that we have been fairly successful in being able to apply our knowledge and understanding when writing up our blogs. This is the case as we have used the ACM principals to demonstrate our understanding of them.

8. We believe that we have tried to write our blogs based on the specification in the brief. We feel that we have tried to structure our blog based on the format being mentioned in the brief.

9. Overall, we think that we worked hard on the pi blogs, we tried to write the blogs by using the structure that was given to us. We also found out more information from the case study or online to gain a

better understanding of what we were to write and how we could go about writing it.