Adrian Pulido

Zachary Cain

CIS 280

Professor Huitsing

3/9/21

Executive Summary

For Feasibility Analysis, there are four forms of feasibility analysis: Operational feasibility- The app will be quite simple regarding accessibility. It will have a basic design with distinct categories to ensure everyone will be able to easily learn how to use it. Technical feasibility- Tech resources will not be a problem since we will be sure to have a highly skilled tech team working to design and develop the app. Once implemented, we can use iPads or tablets for employees to update the information when necessary. Economic feasibility- $60,000-$80,000 is an exceptionally low cost to potentially end a drastic problem during a global pandemic. Once the app proves effective and the smaller, local hospital we will expand it to larger hospitals, and it will all go towards to fight to stop the spread of Covid and lower cases. Schedule feasibility- Time to develop a simple but effective app that everyone will be able to learn that can also do everything we plan will be about 2 months. That is not long enough to cancel our plans and we will be able to continue tracking PPE usage until the app is developed.

For Project Plan, we used a sprint planning template. This project plan is also known as the sprint plan as it contains our user stories from the project. Sprint planning is to ensure success through a shared and detailed understanding of the work ahead. Sprint planning helps teams control projects and better manage product backlog to deploy small parts of projects quicker and more frequently to enhance customer satisfaction. Sprints also encourage teams to keep the big picture in mind but to focus on the necessary steps—rather than leaps—required to ensure there are no gaps or weaknesses in the product, such as key UX considerations.

For Requirements Modeling, we did a Systems Requirements Checklist for each of the four primary requirements. Those requirements are Input Requirements (what do you plan to enter/input/come into your system)-1) Healthcare employees must swipe their ID cards. 2) Online data collection terminals will record labor costs and calculate production efficiency. 3) The department head must enter overtime hours on a separate screen in a healthcare provider's office. Output Requirements (what do you want to output/produce/results from your system)-1) The website must report online volume statistics every four hours and hourly during peak periods in a provider's office. 2) The inventory system must produce a daily report showing the description of a patients' health information. 3) In EHR, functional requirements must be in place for patients to access their records. Process Requirements (what specific task(s), work, or user story(ies), does your system accomplish)- 1) Storing clinical information and data in an electronic format that can be retrieved and viewed efficiently via health information and data. 2) The ability to manage test results will be accomplished by result management. 3) Electronic processing of orders and prescriptions will be accomplished by order management. Control Requirements (what types of security or control should your system provide)- 1) Security Management would provide policies and procedure for protecting assets. 2) Operational Security would identify critical information if friendly actions can be observed by enemy intelligence. 3)

Physical Security Controls would provide deterrent, detective, and preventive measures that are meant to mitigate physical security issues for EHR.

For Data Flow Diagrams, we did two types of DFDs: Context Diagram and Diagram 0. Context Diagrams are used to establish the context and boundaries of the system to be modelled: which things are inside and outside of the system being modelled, and what is the relationship of the system with these external entities. They are drawn to define and clarify the boundaries of the software system. It identifies the flows of information between the system and external entities. The entire software system is shown as a single process. Diagram 0s are another type of Context Diagrams. They represent a basic overview of the entire system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities. The main difference is that the context diagram provides different views of information system. Diagram 0 is used to provide insight view of an info system represents internal process, entities, data flow, and data storage. Data stores are not used in a context diagram.

For UML/Object Models, an Object Diagram can be referred to as a screenshot of the instances in a system and the relationship that exists between them. Since object diagrams depict behavior when objects have been instantiated, we are able to study the behavior of the system at a particular instant. Object diagrams are vital to portray and understand functional requirements of a system. An object diagram is like a class diagram except it shows the instances of classes in the system. We depict actual classifiers and their relationships making the use of class diagrams. On the other hand, an Object Diagram represents specific instances of classes and relationships between them at a point of time. An Object Diagram is a structural diagram which uses notation like that of class diagrams. We can design object diagrams by instantiating classifiers. In UML, a classifier refers to a group of elements that have some common features like methods, attributes, and operations. A classifier can be thought of as an abstract meta-class which draws a boundary for a group of instances having common static and dynamic features. For example, we refer a class, an object, a component, or a deployment node as classifiers in UML since they define a common set of properties. UML has been used as a general-purpose modeling language in the field of software engineering. However, it has now found its way into the documentation of several business processes or workflows. For example, activity diagrams, a type of UML diagram, can be used as a replacement for flowcharts.

Works Cited

https://www.lucidchart.com/blog/sprint-planning

https://www.lucidchart.com/pages/data-flow-diagram

https://www.geeksforgeeks.org/unified-modeling-language-uml-object-diagrams/