**Anthem Dashboard Development**

**CMSC 451 Senior Project CM327**

Back ground and introduction:

Anthem as a health insurance company, dealt with a large amount of insurance claims from Doctors whom patients finished visiting. They currently had a system that sorted these claims in an autonomous manor such that claims could be processed by their claim specialist much faster. Their claim processing system had 22 different procedures to handle the claims. Yet all the procedures needed a constant monitor in case of unexpected delays or false termination. Anthem wanted a software solution to monitor all the claim procedures, alert them and most ideally predict when those procedure were more likely to have error occur.

Development goals:

Develop a prototype web-base API that can receive procedure duration time data and flag unexpected long or short run time. This API should be able to apply to all 22 procedures. Also, by analyzing current data, the API should extrapolate future threshold of error in manor of what time in a month, or in which quarter of a year.

Milestones:

1. Access Sample Data- Gain access to sample data that is central to the project from one or more procedures. – Top Priority (Achieved)
2. Creating a UI and feed it with sample data such that the client can see what direction we are heading to, and we can gain valuable feedback from them – second top priority (Achieved)
3. Setup our working environment on Amazon Web Service (AWS) – need to be done before or a few week after algorithm starts to implement (Achieved)
4. Work on the API - Develop an API that correctly interpret the start and end time stamp from one procedure’s the sample data-third Top priority
   1. Implement Algorithm – Actually implement the algorithm of tracking the error by analyzing data. – Second to Accomplish
   2. Test the work – After the core function is implemented test it on the data – wrap up step
5. Make it smart - Develop prediction algorithm on that API so that it will give the system administrator a heads up when there’s a high possibility of more error may occur. – post accomplish add on
   1. Get the quarter - Implement prediction on which quarter of a year that more error occurs – accomplish fourth
   2. Test the work – After the core function is implemented test it on the data – wrap up step
6. Test it out - Test the API on the rest of the procedures and modify so that it can be apply to them – hope for the best
   1. Test the cores – Choose the more important procedures to test the API on and gather information about error – accomplish first
   2. Fix the bug – If there’s any incompatibility when the API is applied on the other procedure, find out why and try to fix it. – accomplish second
   3. Loop again – after the previous patch, test it on the target again and repeat the test-fix step – keep looping

Methodology:

The team will be using Agile-Scrum method on a two-week iteration. Jessy Gordon will be scrum master, David Coruso will be product donor.