1. **Team Name:** 1337\_hAXX
2. **Team Leader for this deliverable:** Alex Jacobs
3. **Team Members:** Logan Brincks, Alex Jacobs, Erik Rasmussen, John Polus, Steven Karrmann
4. **Meetings:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Time-date** | **Attendees** | **Agenda** | **Action Items (who will do what)** |
| 3/30 | Logan, Alex, John, Steven, Erik | Begin OO Design | Logan – Functional Requirements  ***Missing the time log!***  Steven – Non-Functional Requirements  Erik – Class Diagram  Alex – Functional Requirements  John – Class Diagram ***Missing the time log!*** |
| 4/6 | Logan, Alex, John, Steven, Erik | Finish OO Design | Logan – Group Report  Steven – Project Schedule  Erik – Project Schedule  Alex- Final Report, Requirement Cleanup  John- Final Class Diagram |

1. **Weekly Time Logs:**

|  |  |  |
| --- | --- | --- |
| **Person** | **Total Time in minutes** | **Tasks** |
| Logan | ***???*** | Functional Requirements, Group Report |
| Steven | ***???*** | Non-Functional Requirements, Project Schedule |
| Alex | 164 | Functional Requirements, Group Report, Requirement Cleanup, Report Summaries |
| Erik | ***???*** | Initial Class Diagram, Project Schedule |
| John | 113 | Initial Class Diagram, Final Class Diagram |
| **Total Time:** | 0 ***???*** | ***It’s not clear how you distribute the workload for this deliverable, and how much time each of you spent to work on this deliverable!*** |

1. **Issues:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Issue Number** | **Discovery Date** | **Resolution Date ( Est. – Act. )** | **Responsible Person** | **Description ( Prob / Resolution )** |
| 1 | 3/31 | 4/6 | Logan | Functional requirements mentioned the “user” instead of defining in terms of the “system”. |

1. **Files and their locations:**

|  |  |  |
| --- | --- | --- |
| **Filename** | **Location** | **Contents** |
| d9\_groupReport\_OO | \2\_ReengineeringProject\OO Design\ | Group report for OO design phase. |
| d9\_OODesign | \2\_ReengineeringProject\OO Design\ | Main deliverable. OO Design report. |
| d9\_InitialOODesign | \2\_ReengineeringProject\OO Design\ | Top-level, 1st-pass class diagram. |
| d9\_FinalClassDiagram | \2\_ReengineeringProject\OO Design\Nibbles | Final detailed class diagram. |
| d9\_FinalDiagramImage | \2\_ReengineeringProject\OO Design\ | Final detailed class diagram image. |
| d9\_ProjectPlan | \2\_ReengineeringProject\OO Design\ | Project plan with critical path. |

1. **Plans for Coming Week:**

For the upcoming week we will begin on the next phase of the reengineering of our project. We will start discussing and designing test cases to verify future code functionality and, more importantly, verify the requirements we defined in this project phase. Our plans will follow the schedule specified in our MS Project plan. ***Good.***

1. **Comments:**

**Engineer 1:** Logan Brincks

In working on this deliverable, it was different to start a project from existing legacy code. In working on defining the system requirements, we already knew what the system was supposed to perform. This perspective changed the approach that was necessary in creating these requirements we could take some previous implementation that existed in the previous example of the system but we also learned to be able to adapt and change the system to the new code language. ***Good.***

**Engineer 2:** Alex Jacobs

Due to this project phase being focused on requirement specification and OO design, it was important to ensure the analysis done in the previous project phase was done correctly. With each of these phases building on the previous, it would be very easy for a mistake made in one phase to propagate into the next and throw the entire project off. In regards to this phase specifically, I think it is extremely important we have a well-constructed class diagram and OO design. If we do not, it will make the testing phase far more difficult. ***Good point!***

**Engineer 3:** John Polus

For this deliverable, we started doing more detailed design and really looking at how we were going to implement the program. I realized that it is really nice to have a class diagram that lays out the connections between various classes. It helps you to organize your thoughts and also to quickly notice if you have made a bad design decision. Rhapsody isn’t as scary as I thought it would be either, it seemed pretty user friendly and I would definitely use it again in the future. ***Great!***

**Engineer 4:** Steven Karrmann

I spent most of my time working on the UML class diagram. Trying to look ahead and design a program without having code to look at and compare to is difficult for me, so I was glad to have the extra practice in that regard. Having the class diagram while coding is very helpful though, especially when we have multiple people coding the same project. The schedule will be helpful when dealing with having multiple people working on the project, as it will help keep us all on track, and make sure that everyone stays on the same page. ***Great!***