**C Sc 335 Analysis and Design Artifacts for Final Project**

*This must be in a private Github repo in a directory named documents*

*with your project manager added as a collaborator*

**1. Team Name:**  Team Rocket

**2. Team Members**: Daniel Phillips, Reaper Romero, Alex Yee, Benjamin Shields

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**3. Candidate Objects or Class Hierarchies**

List the most important objects, or the name of an inheritance hierarchy, and the main responsibility.

|  |  |
| --- | --- |
| **Candidate Object** | **Single Responsibility in 1 or 2 sentences** |
| Board | Responsible for all of the territory objects, |
| Player | Represents the User and the AI, takes I/O from the GUI and sends it to the Game Class, Controls the attacking and defending ABSTRACT CLASS, |
| Card | Represents the card type, and the country that will receive the units |
| Card Type | An ENUM that represents the card type(Soldier Cannon or Horse) |
| Territory | A country represented in the game, holds the |
| Game | Sets up the board, Handles the Deck |
| Deck | Handles a set of Risk cards, records the number of sets of cards turned in. |
| Dice | Takes in the number of dice to roll, rolls them, returns the values of the dice |
| PlayerType(Easy AI, Medium AI, Hard Ai, user) | Represents all of the types of players that could be in this game, the user is a real person, the Easy Medium and Hard Ai. all classes extend Player |
|  |  |

*These Class and Sequence Diagrams may be written by hand and scanned or drawn with a UML editor such as Violet* [*http://sourceforge.net/projects/violet/files/violetumleditor/*](http://sourceforge.net/projects/violet/files/violetumleditor/) *and / or the sequence diagram editor or* [*https://www.websequencediagrams.com/#*](https://www.websequencediagrams.com/)

4. Class Diagram: Your team UML Class Diagram must show at least all of your candidate objects from above. Show any relationships between them the classes such as inheritance or interface implementation. Draw general associations such as dependency or aggregation. Label some to help explain things. Add any multiplicity adornments that seem appropriate. Use notes to explain things if you feel it will help. Each UML class must show the class name. For full credit, each class must have an average of at least one attribute per class. There must be an average of at least 2.0 methods per class, which may be implicit (no need to repeat methods) if the class implements a Java interface with methods shown there.

**5. Sequence Diagram:** Your team UML Sequence Diagram should show the most important scenario you can think of. Your sequence diagram should show most of your objects from above and how they communicate with each other.