Team Members: Aaron Geesink, Garrett Geesink, Ali Parva, Faris Hijazi

The rules of Slam Jam are as follows:

1. Be a team player.
2. Team members will keep other members up-to-date on discord as to what they are working on and meet up in voice chat for help
3. Get your code done: don't be the guy that comes to class with his part not done
4. Be active on discord! If there is discussion there, make sure to pay attention to it.
5. Do NOT merge any changes to the project to the master branch in github before it is reviewed Aaron and the rest of the team. Upload it as a separate branch for it to be reviewed, and Aaron will merge it into master
6. Don't merge or upload on GitHub without knowing how to use Git. Ask the other team members if you need help with using git.
7. Include a comment with your commit explaining what you added/changed, don't commit without comments
8. If you are having trouble implementing code, ask another group member for advice and assistance. Speak up about coding issues before it becomes an issue.

The Coding Standards of Steve are as follows:

1. Tools
    a. The project will be programmed using C++ in QT version 5.13.1
    b. The UI will be designed as a QT widget application
    c. The UML Diagrams will be made in Microsoft Visio
    d. The State Machine Diagrams will be made using Microsoft Visio
    e. The Use Case Diagrams will be made using Microsoft Visio
    f. The class and function documentation will be made using Doxygen
    g. User stories will be posted on GitHub, sectionalized by sprint
    h. All other scrum artifacts will be made using google docs, then turned into a PDF
    i. The project will be uploaded and saved in a GitHub repository
2. Procedure
    a. Before programming a class, the UML diagram for that class must be completed, reviewed, and accepted by all team members.
    b. ALWAYS test the program to confirm that it works before pushing or merging any changes
3. Coding Style
    a. Don't try to do everything in one huge function. Break a problem into many smaller functions that call each other.
    b. Don't have an entire function for each small feature that is only used once that could easily be combined into one slightly larger function to increase efficiency.
    c. Variable and function names should be easy to understand by all developers. Give variables descriptive names that express what they do.
4. Commenting

a. At least one comment must exist per chunk of code describing what said chunk of code does
b. At least one comment for every function