Guidelines for Kick A$$ projects instead of Kicked A$$ projects

1. **Define requirements/features/user stories**
   - Nail this down in the beginning because everything else is based on this step
   - If we miss a few minor requirements that's OK because we are "Agile" and changes afterwards will not derail the project
   - Put these in your team project Trello board

2. **Mock up the Web pages you need based on the requirements defined in step 1**
   - Use the mock ups to role play user flow through your application to find any gaps in the original requirements and update the requirements accordingly
   - Don't worry about beautifying Web pages/screens. It's OK for things to look crude at this point.

3. **Work on the application design as a group and document it (picture of white board or paper sketch or Class diagrams, etc…)**
   - The design should be a conceptual representation of how the application will implement all the items in the requirements
   - You should be using the MVC design pattern which already defines some of the design. You only need to identify which artifacts (classes and methods) will be responsible for implementing each requirement in **Step 1**.
   - Designs patterns such as MVC, DAO and OOD should help you with much of the heavy lifting in this step (don't recreate the wheel).

4. **Now you can start generating code. Do not do any coding in steps 1 – 3 ((Recommended)**
   - Stub out the application as a group to generate a code base project that all group members will clone to their local repos.
   - Generate all the classes, method signatures, controller mappings, jsp files, xml, pom.xml, gitignore, etc... but do not implement any of these artifacts except for the pom.xml and gitignore. Everything else are only stubs.
   - List these artifacts as task on your team Trello board.
   - You can push this code base to the team project github repo if there are no red error messages in the Eclipse/Intellij project.
   - All team members can clone the project now.

5. Assign building/coding of the artifacts identified in **Step 4** to the team members on your Trello board. Give each task a deadline. We can use your Trello board to track your progress and identify issues before they become show stoppers.

6. Throughout this process communicate with your team any requirements or design changes that need to be made for the project before implementing them to prevent breaking someone else's code or design.

7. The scrum master should monitor the state of the app by testing it after each team member commit to ensure it is working and as designed. Then instruct each team member to pull the latest working code. Just as a reporter does not write todays headlines based on old facts you should not write code based on an out of date local repo (git pull first). The scrum master can

manage this to prevent constant github repo updates (git pull each morning before starting to work on your code).

8. Finally, **Remember how to Receive and give Feedback.**