Final Project Meeting

During this meeting, you should do the following:

1. (15 min) A run-through of your program and how it functions. Be sure to demonstrate all the features of your program (please mention those things that are not fully working as well).
2. (30 min) A walk through of the final code. The code should be relatively clean and well-refactored (show us how the code looked before and after the refactorings). You should try to present the code at a high level, explaining its meaning (e.g., "This class implements the controller for handling user authentication"), as opposed to low level explanations (e.g., "This method has a switch statement that selects what to return based on the input y"). For each piece of code you should show the tests which exercise it, and explain them. We won't have time to go through all the code but each member of the team should be prepared to talk in depth about a particular component.
3. (15 min) A discussion of the process. You should first restate the process that you planned to follow at the beginning of the semester. Then, give your insights on what worked well and what did not, and how you adjusted along the way. This should include a short overview of your wiki page and git/svn account history, which will serve as evidence that your team actually followed the described process.
4. Each member of your team should participate in the meeting. In other words, everyone must present something.

Deliverables

(by the deadline above)

1. In your code repository, create a final-release tag/release which marks all your code, tests, and documentation. See the grading guidelines below.
2. Final Documentation
3. Peer evaluation (can be submitted up to May 6 Friday 11:59pm after the final project meeting)

Wiki

We are going to use the contents of your team's wiki and/or the other tools your team used (e.g., Pivotal Tracker, Redmine, trello, etc.) to evaluate whether you consistently followed the software development process that was agreed upon.

We are looking for the following:

- Planning game (division of user stories into iterations).
- User stories (description of user stories similar to the suggested format or something similar).
- Minutes of each iteration meeting as recorded by the scribe.
- Other evidence to convince us that you followed the process: meeting schedule, chat logs, etc.

The wiki has a history system so we can check what/when changes were made. We would be suspicious if multiple large changes were made during the last few days before submission.

Peer Evaluation

Peer evaluation is mandatory.

Remember that peer evaluations are confidential. Your teammates will not know what you write about them. So please be honest when you do the evaluations.

Additionally, you should write a personal evaluation of your own contributions to the project.

You need to checkout final-peer-evaluation.txt from SVN, complete the form, and commit your changes:
Grading Guidelines

The final project is worth **60 points**:  

Software Product (40 points)

- Polished software (10 points). All features are well implemented and account for corner cases.
- Code documentation (10 points). Have comments for all public methods and public classes following the proper commenting conventions (i.e., Javadoc, Doxygen, Pydoc, etc.). Do not commit automatically generated HTML documentation to your repository. Instead, document in your project documentation the command which generates the code documentation.
- Testing (10 points). Your code should be very well tested. The majority of your code should be covered by automated tests. Tests should also be documented. It should be easy for a new developer to understand what a test does by just looking at its code and documentation. **Manual tests are only acceptable if you discussed beforehand with your team's TA.**
- Code formatting (3 points). Code is properly and consistently formatted, either manually or by using the code formatter.
- Dead code (2 points). Your code base should not contain any dead code, e.g., unused code snippets, unused import statements, etc.
- Code tag/release (1 point). The final version of your code is tagged (the tag may only contain commits dated before the deadline; anything not included in the tag will not be considered).
- Installation and distribution (3 points). Your project should be very easy to build and run by following the documentation. Whenever possible your application should be easily installed by an end user (e.g., through an app store, as an .exe, etc.).
- README (1 point). The root of your repository should contain a short README file (or README.txt, or README.md, ...) that describes shortly what your program does, lists contributors (team members), and links to other relevant documents for using your project (INSTALL, Documentation, etc)

Documentation (20 points)

- Must address the goals/issues from the Final Documentation page. It should be committed to your code repository and tagged under final-release.

Misc (no point allocated but possible point deduction depending on severity)

- If appropriate, include licensing information.
- There will be a very large penalty for a broken final submission.
- Wiki should be up-to-date; minutes of iteration meetings should be on there.
- Any remaining issues with the team/TA needs to be fully resolved.