CSC 490
1st Sprint Retrospective

Suggested Sprint Start: Friday, February 2, 2018 (to give you time to get everything ready)
Suggested Spring Stop: Friday, February 16, 2018
Retrospective Due: Tuesday, February 20, 2018 at 11:55pm

Summary

Convert your requirements into user stories and place them in your backlog. You don't have to convert all of them, but I would recommend enough for the next 2 sprints (approximately). Take the most important of these (the ones you anticipate to be added in your first sprint) and play the Planning Poker game to determine the amount of story points for each. After this you should decide on a reasonable number of points you want to tackle for your sprint. If all pointed stories don't make it into the sprint, that's ok, as sometimes during Planning Poker we realize we need to break down stories into more stories or that some stories or more difficult than we initially imagined. Everyone should then choose the stories they want to be responsible for. Try to make this even as I want each team member to have similar amounts of points, however no one should be forced to work on a story if they don't want to. This means ultimately it becomes a negotiation where we choose and trade off stories so we each have equal (or close to equal) amounts of work. Now start your sprint.

If during your sprint, you finish all your stories, you are to ask your teammates if they need help accomplishing their stories. Do not just sit around and do nothing thinking you are finished. If there is time left in the sprint, I want you to continue to do work. If the entire team finishes the sprint's stories or if you are finished with yours and all your teammates don't need help, you are to grab from the stories in the backlog that have story points associated with them. This may mean that your anticipated velocity for the sprint is lower than your actual velocity, and that's ok (the alternative being that you do nothing for the remainder of the sprint, and that is definitely not ok). If no stories are left that have points, then grab the most important ones from the backlog and estimate a point value for them (you can do it yourself or do Planning Poker if your team is available, but if they aren't, instead of waiting just point it yourself with your best estimate and get to work on it).

During your sprint I want everyone participating in daily scrums. Yes I know this isn't your full time job. I know you have other assignments/classes/work etc. However daily scrums should really only last like 10 mins and can be done via FaceTime, Skype, Duo, or whatever you want if physical presence is not possible. Obviously meeting in person is the best option, but I'll accept video streaming software in lieu of this, however I would recommend against a simple voice only call. A lot of communication is non-verbal and I feel a lot is lost when we communicate with voice only means. I do not want to hear you only texted each other or emailed as your daily scrum. This is definitely not acceptable.

At the end of the sprint, I want to see your burndown charts. If you are using Trello, this website may help: http://info.trello.com/power-ups/burndown

Every team and every member of the team should already be using Git and GitHub. You should have been using them for your documents and models, however if you used something like Dropbox or Drive, that's ok but now it's time for code. Dropbox and Drive are not good choices for writing code collaboratively (this is not normally part of their intended usage). I expect every team to have a GitHub project created and EVERY MEMBER OF THE TEAM must be added as a contributor to that project, with full read and write access. I'm saying this now so there is no confusion, when I check your GitHub project at the end of your sprint, if you are not on the project as a contributor, you (individually) will be deducted serious points for the project. Also, ADD YOUR PERSONAL NAME TO YOUR GITHUB ACCOUNT. I don't care if your GitHub name is something weird or random, but if I click on your profile, I expect your real name to be there. I'm not emailing everyone to figure out who is who. Finally, use Git frequently to commit your changes and push them frequently to GitHub. Version control is utterly pointless if you wait until the end of the week to commit and push everything. You must be doing this every day (maybe even several times a day) if you have changes that are stable and that add to the project in a meaningful way.

Submission
Zip the following two documents up and submit them on Moodle.

In a file called **"info.pdf"** put the following information:

- Team name.
- URL to your GitHub project.
- URL to your JIRA or Trello project
    - I probably already have this from most of you but this will help me to not have to look for it in my email
- List all dependencies used.
- Any special instructions necessary for me to run your code?

In a file called **"retrospective.pdf"** answer the following questions completely:

- Restate your team name at the top of the page.
- Did going through all the requirement engineering and UML modeling before your first sprint help in implementation? Did you feel as though implementation was easier having gone through all that work before writing any code? Do you feel as though your product is currently (and will be when you are done) more stable and more in-line with what your client wants vs. if you just talked to them a little bit, got an idea, and went straight to coding? For your next big project, would you prefer to do it again like this or would you rather jump straight into programming the system?
- How many of your requirements did you convert into user stories? Was it enough for your sprint or did you have to convert some during the sprint? Was there a method or procedure you used to convert them to user stories? Was everyone on the team involved in the process of converting each requirement or were members given specific requirements to convert?
- Did playing "Planning Poker" help out with story point estimation? Were there stories where the team couldn't compromise on a number? Were there stories where the team was initially very different in their estimates? Which stories did this happen to and how did you resolve it?
- What was your means of communication for your daily scrums? Did you meet in person or use some other form of communication? What time of day did you meet and was it the same time every day or did it fluctuate? How long did your scrums last (on average)?
- What was your velocity for your sprint (both estimated and actually accomplished? Did you finish everything for your sprint? Did you have to grab more stories (i.e. did any of your team members finish all their stories before the sprint was done)? If so, who finished? Who didn't finish all of their stories? Is your product potentially shippable? Did you get all the core/most important stories accomplished?
    - Tell me exactly how many story points each member accomplished during the sprint (what they set out to accomplish and what they actually accomplished). Only count the stories as accomplished if they were done and marked as done in JIRA (or Trello)
        - List name followed by given and accomplished story points for each person
- How many meaningful commits did each team member make? List name followed by total commits for the sprint. By "meaningful" I mean don't include any test commits used to get things setup correctly or to test GitHub collaboration. Remember that a single push can contain multiple commits. If you had to create a new repo for whatever reason, I want both GitHub links (unless you deleted the first one) and I want the meaningful commits for both repos (list them separately). This information should be

available in GitHub itself but I want this listed in case I miss something when examining your GitHub repo myself.

- For your next sprint, would you increase or decrease the amount of planned story points? What is your reasoning behind your answer? What all are you planning on doing differently for the next sprint?

- When answering these questions, it is helpful to use the charts/reports that JIRA provides. Go to JIRA and view all the different reports to see how your project is going. Attach a screenshot of your Burndown chart to this report. While you are looking at the Burndown chart, notice the table of information below it. All these reports, charts, and corresponding table data are used for process improvement.
  - This is for Trello users too

- Format for "retrospective.pdf":
  - 2-5 pages (no header page nor references, but I do want your team name at the top of the first page then a blank line and then begin answering the questions. The Burndown screenshot does <u>not</u> count towards this page limit range)
  - 12pt font
  - Any standard, legible font type is fine
  - Single or 1.5 spaced (no double spaced)
  - Margins can be between 0.5" and 1.0"
  - PDF format (I don't care how you generate this PDF, I just want the final file to be a PDF)

Rubric
TBD

Total: 50pts (which is 25% of overall course grade)