Benjamin Carson (bkc52), Nicholas Rutledge (njr48), Timothy Eng (te76)

CS 3110

M. Clarkson

MS0: Proposal

Regular status meeting:

Outside of our twice a week discussion, which is where we will bring up any issues that we are having with our TAs, we plan to meet **(time and place)**. In addition, we have a couple other communication/project management systems set up – besides the obligatory Cornell GitHub repo, we also have a group chat for quick issues that don’t need us to be in person to discuss, and an Azure DevOps board, which is an Agile/Scrum project management tool that (te76) has used at work and found helpful for user story tracking, epic/issue/task management, division of labor, sprint tracking, etc.

System proposal:

A proposal for your system. Summarize the system you intend to build. Tell us what will be the most important functionality of your system. This summary should be about one page long.

Provide:

A very short statement of the core vision or key idea for your system.

A short (no more than six items) bulleted list of the key features of the system.

A narrative description of the system you intend to build. Go into enough detail that, if your charter were given to another team, and that team were never allowed to talk to you, they would still understand more or less the functionality that you have in mind.

A roadmap for your system. There are going to be three “sprints” (implementation phases) after this charter: MS1 (alpha), MS2 (beta), MS3 (release) Each will last about two weeks. Make a list of what features you plan to implement in each sprint. Then, for the features you plan to implement in MS1, sort them into three categories— Satisfactory, Good, and Excellent—like how the programming assignments in this class have been. That gives you a plan for what to do if MS1 is going worse or better than you expected.

A preliminary design sketch for your system. Spend some time thinking through what you are going to need to build. Of course, your plans will evolve throughout the project. But it’s good to have talked as a team up front about what seems necessary. In about one page, answer the following questions:

What are the important modules that will be implemented? What is the purpose of each module?

What data will your system maintain? What formats will be used for storage or communication? What data structures do you expect to use as part of your implementation?

What third-party libraries (if any) will you use?

How will you test your system throughout development? What kinds of unit tests will you write? How will you, as a team, commit to following your testing plan and holding each other accountable for writing correct code?