CSC 440 Stage D Deliverable

- Timespan: 10/25/2017 - 11/1/2017
- Submitted: 11/1/2017
- GitHub Repo

Team Name: Redbeard

Team Members: Nick Tope, Eric Spahr, Katie Bolen

Application of Theories

(Note: While only a few of the principles below applied to the current stage, we thought it beneficial to write more principles in advance that we believe will apply to later stages)

1. KISS - Keep it simple stupid.
    - I used this rule when trying to explain Flask and Python code to my team members to show them what I learned from the class that was given on Friday night a couple of weeks ago. This helped make things easier to understand for my team members since they didn't attend. [Eric 11/1/17]
2. DRY - Don't repeat yourself. [Eric]
3. Always design for your audience. [Eric]
4. Stockholm Syndrome - Confront the brutal truth of the situation, yet at the same time, never give up hope. [Eric]
5. Adding human resources to a late software project will make it later. Create a procedure for delays and make all team members aware of this procedure; what is expected of them, how they should respond, and any potential consequences that may result from failure to follow. - Fred Brooks, Brooks' Law [Katie]
6. Focus project on a concept at which you can be the best. Formulate a list of potential concepts and attempt to explain why the team believes they can be the best. - Good to Great, Hedgehog Concept [Katie]
7. Team members should be in constant communication. In the earliest stages of the project, everyone should exchange contact information, and agree upon a regular meeting schedule. Some means of emergency contact should also be agreed upon and checked regularly. - Fred Brooks, manage complexity by simplifying communication [Katie]
8. Establish a design concept before beginning coding. Begin with a list of product requirements, and meet as a group to discuss any additional ideas. - David Parnas, Rational Design Process [Katie]
9. Create documentation for all stages and update frequently. A standardized form will ensure that all necessary information is provided. - David Parnas, Rational Design Process [Katie]
10. Design tests early in the planning process to help focus the architectural structure of software. - Advice from professor [Katie]
11. Utilize the principle of encapsulation to streamline debugging/testing, and to maintain a high level of adaptability of product. - David Parnas, Rational Design Process [Katie]
12. Assess team members' individual strengths to ensure that they are in assigned the most appropriate role within the team. - Good to Great, First Who…Then What [Katie]

13. Require the use of comments in coding to clarify program logic, thus simplifying communication between team members. Have team members meet up in pairs/small groups and explain their newly added code using only comments to verify its effectiveness. - Advice from professor [Katie]
14. UIP Principle - Understand, Iterate, Practice. [Nick]
15. Peer reviews catch 60 percent of the defects. - Barry Boehm and Victor R
    - This principle will apply to every stage. Much of the mistake that are made can be caught through peer review. While we haven't written code for this stage, we did have the team peer review this report to look for mistakes. [Nick 11/1/17]
16. Disciplined personal practices can reduce defect introduction rates by up to 75 percent. - Barry Boehm and Victor R
    - I normally wait till the last minute to complete projects and assignments. However, after watching studying videos from earlier in the semester, I've introduced changes to my studying to reduce the risk of lateness and increase efficiency. The most influencial change has been establishing a designated study area. [Nick 11/1/17]
17. If you don't know history, you're doomed to repeat it.
    - This principle applies to past projects we've done. In the past, my team has had problems communicating, ending in delayed deliverables. Learning from that, we've turned to trello, Github, and text to facilitate communication between our team members and deliver deliverables on time. [Nick 11/1/17]

Teamwork

- [11/1/2017, 40 minutes] All team members met up afterclass to share their software rules and complete the agile retrospective.

Schedule

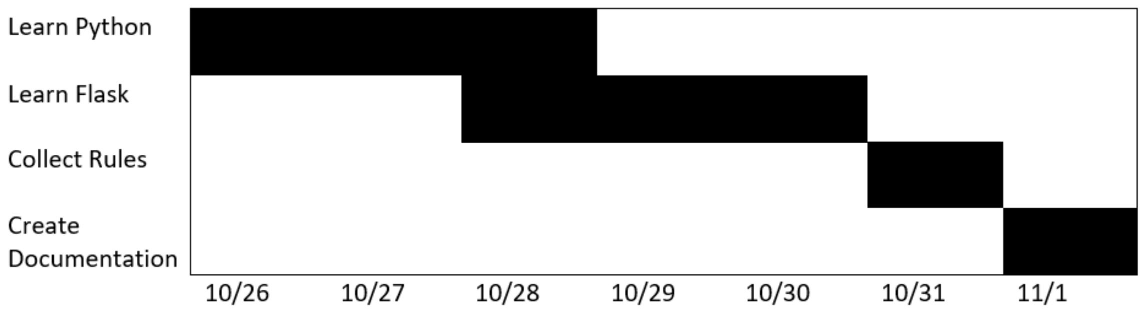
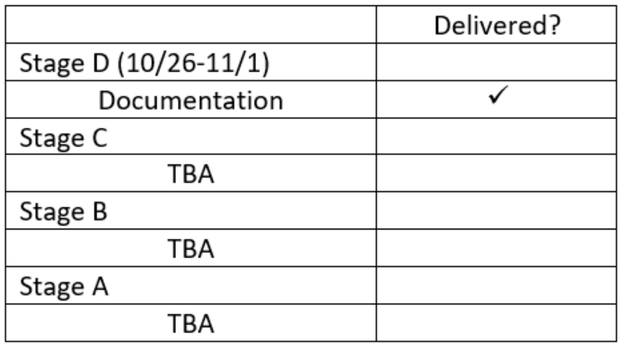


Figure 1: Gantt Chart



	Delivered?
Stage D (10/26-11/1)	
Documentation	✓
Stage C	
TBA	
Stage B	
TBA	
Stage A	
TBA	

Figure 2: Check List

Agile Retrospective

Team Leader: Nick Tope

Team Members: Katelin Bolen, Eric Spahr

Facilitator: Dr. Samuel Cho

Meeting: * Date: November 1, 2017 * In attendance: Nick Tope, Katelin Bolen, Eric Spahr * Goals: Compile all documents, complete retrospective * Total meeting length: 40 minutes

Documents: * Stage D Application of Theories: Software Engineering Rules [ver1] * Stage D Schedule * Stage D Agile Retrospective

Things That Went Well

- Time management allowed for deadline to be met.
- Trello communication went smoothly.
- GitHub repository created without issue.

Things That Could Be Improved Upon

- Team could devote more time to Python lessons.

Lessons Learned

- Allow time for unscheduled issues in Stage Schedule.

Final Thoughts

- Before the next stage, we will meet a second time and update our documents.
- Stage D established a good foundation for future stages of our project.