CSC 530 – Software Engineering Methodology

Project 02: Web-Based Application

Objective: Design, develop, and test a web-based application using Agile methods.

Client: Sung Kim

Deadline: December 6, 2023

Project Description:

In this project, you must design, develop, and test a web-based application. Your application should be a multi-user platform with multiple levels of access (e.g., basic user, admin, manager, etc) who can query and add data to a back-end database.

You can choose what type of application you want to build, provided that there are:

1. A web-based interface that users can use to add, edit, request, etc. data stored in a database.
2. A database that will serve the requested data.
3. Multiple categories of users who will interact with the system with varying levels of permissions.

Some potential examples include:

- Issue tracking systems are a collaborative system for maintaining a list of outstanding items. Organizational IT support is a common example, where a request leads to a generation of a "ticket" which is added to the issue tracking system until it is updated/resolved.
- Client Relationship Management (CRM) systems used by businesses to add, edit, and maintain information on current and potential clients.
- A basic online store (minus the payment processing aspect).
- But be creative if you have a good idea.

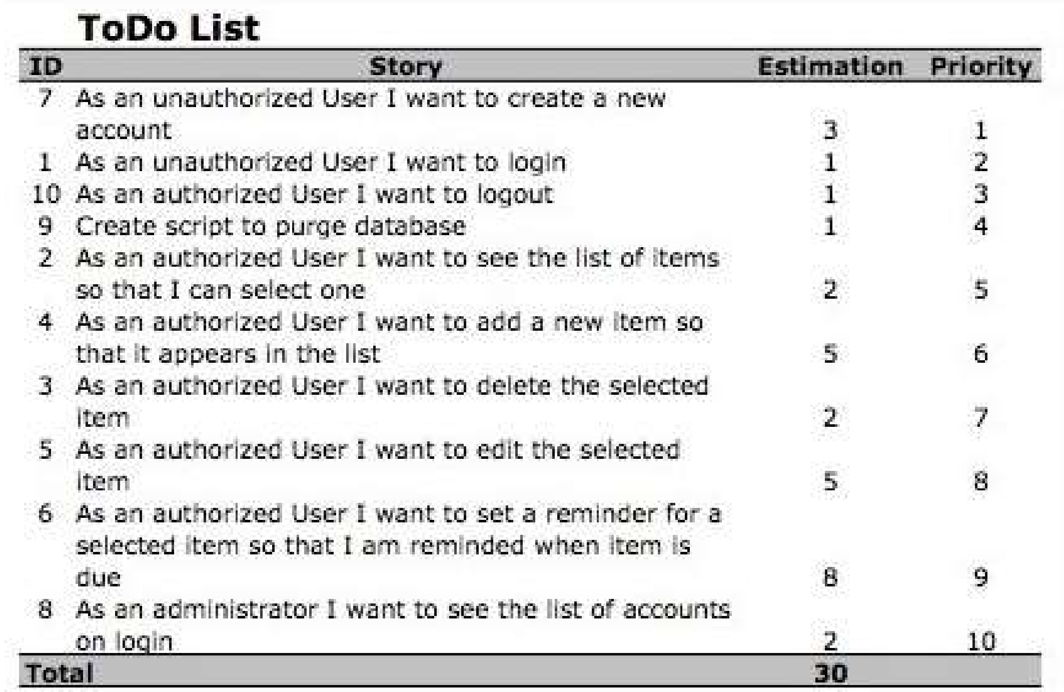
You will submit the following deliverables:

1. Working software (including its full revision history)
2. Any documentation created. At a minimum, this should include the **product backlog** and **testing plans/results**.
3. A recorded oral presentation and demonstration of the developed work, to be uploaded to YouTube, no later than **Wednesday, December 6, 2023.**

Ground Rules:

- The database schema must feature numerous different fields that can be used by users to submit queries.
- Each person must demonstrably execute an equitable proportion of overall work.
- You MUST use **GitHub** (https://github.com/) or a similar revision control system to track any code you develop.
- You MUST actively maintain a product backlog (more on this below).
- You SHOULD use an Agile development workflow to organize your work (e.g., Jira, IceScrum, Smartsheet, etc.)
- You MUST organize your work in **sprints** spanning specified timeboxed periods. At the end of each sprint, you should have completed all of the tasks assigned to the sprint and deliver the work product completed to date. You may decide on the timing of the sprints and the tasks to be assigned to each sprint. Each class, I will meet with each class in a quasi-scrum to discuss the progress.

The Backlog (*see* https://www.scrum-institute.org/The_Scrum_Product_Backlog.php)



ToDo List

ID	Story	Estimation	Priority
7	As an unauthorized User I want to create a new account	3	1
1	As an unauthorized User I want to login	1	2
10	As an authorized User I want to logout	1	3
9	Create script to purge database	1	4
2	As an authorized User I want to see the list of items so that I can select one	2	5
4	As an authorized User I want to add a new item so that it appears in the list	5	6
3	As an authorized User I want to delete the selected item	2	7
5	As an authorized User I want to edit the selected item	5	8
6	As an authorized User I want to set a reminder for a selected item so that I am reminded when item is due	8	9
8	As an administrator I want to see the list of accounts on login	2	10
Total		30	

Example Scrum Product Backlog

Purpose
In its simplest form, the backlog is a single repository containing all tasks that need to be completed before a project reaches its final objective. In software development, these tasks may include writing programs that implement new features, but anything that becomes a task should be recorded. This includes bug fixes, providing documentation, recording instruction videos, performing testing, and much more.

Creation
Tasks start when *user stories* are envisioned. The user story describes what features a software system should provide as seen from an end-user perspective. One stories are accepted, they can be refined into tasks and the work effort can be estimated.

Increments
At the beginning of each sprint, the team sits down and decides which user stories will be accepted into the current sprint. Once the stories have been accepted, a work estimate will be associated with them to facilitate planning. Each developer has a rough idea of how fast they work (their *velocity*, in Agile terms) and, consequently, how much work they can take on.

Since there are tasks associated with each story, we now know what to do. In many agile methods, tasks are not assigned. Rather, individual team member claim a task to work on and see it through completion.

Burndown

By tracking task completion, an estimate emerges that outlines how much work is left. This is called the work burndown.