**Group Name:** The Red Dragons

**Group Members:** Logan Zipkes, Will He, Sofia Shubert, Mai-Anh Ha, Nina Ignatchenko, Matt McClelland

\* Link for the website: <http://www.cs.utexas.edu/~maianhha/Page%201.html>

\* Username for Admin: Stephany Coffman-Wolph; password: UT\_Coffman-Wolph

\* Username for student: John Smith; UT\_Smith

● Description of Project

Our project is a Canvas-like website that is designed for a homeschooled student. For example, our project accounts for one administrator and one student. From this, the student is in his/her own group, the “red dragons guild”. We decided to reuse the code from the Project #0 website and incorporate the trivia game into the website as well. The assignments are divided into four different categories: Homework, Quizzes, Projects, and Exams. The student has a range of 2-4 existing documents that are already linked, and he/she have a button linked to a page where new potential uploads from the admin will be located. We have four pre-made quests for the students: Math, Geography, US History, and Biology. The quests use counters to keep track of the score and displays the total at the end, as well as a summary of success. The level of difficulty of each question is also on display. There is a separate link for the list of corresponding point values for the student to understand their success level.

We were unable to figure out how to implement the unit tests through a JavaScript testing framework, so we made “homemade tests” by keeping track of the functions of the code in the console. We checked this by printing out the login information (whether it was correct or not), keeping track of the running quest scored (whether or not they were being added correctly), and displaying (in the console) which questions were answered correctly after submission. We were also unable to allow the admin to create their own quest, but we did allow them to download a template of a new quest, customize it, and upload it to the student page. The new uploads from the admin are stored in corresponding assignment directories, so there is little confusion for the student user.

● Features Implemented

* Login: There is a public view and two types of users eligible for login. (admin and student)
* View Assignments: The student is able to view existing assignments and save them
* Test: The student is able to take the quests to test their skill level.
* Upload: The admin is able to upload new assignments and quests.
* Download: The admin is able to download a quest template to customize.

● Work Plan (We used out user stories as our plan of work)

**User Stories**

1. [✓](https://emojipedia.org/heavy-check-mark/) Create website for the “Quest/Canvas”.
2. [✓](https://emojipedia.org/heavy-check-mark/) Create a login for users and administrators. (public vs private view)
3. [✓](https://emojipedia.org/heavy-check-mark/) Create (or upload) homework assignments, projects, quizzes, and test content into the system
   1. Difficulty: 2
   2. Time Estimation: 30 minutes
   3. Actual Time: 50 minutes
4. [✓](https://emojipedia.org/heavy-check-mark/) Create a series of “Quests” to build certain skills
   1. Difficulty: 4
   2. Time Estimation: 1 hour
   3. Actual Time: 3 hours
5. [✓](https://emojipedia.org/heavy-check-mark/) Create (or display) a skill list related to a class/assignment/project/quizzes/ tests
   1. Difficulty: 2
   2. Time Estimation: 30 minutes
   3. Actual Time: 20 minutes
6. [✓](https://emojipedia.org/heavy-check-mark/)Assign a skill, difficulty level, etc. to each problem
   1. Difficulty: 1
   2. Time Estimation: 10 minutes
   3. Actual Time: 10 minutes
7. [✓](https://emojipedia.org/heavy-check-mark/) Create a level list with corresponding point values
   1. Difficulty: 4
   2. Time Estimation: 45 minutes
   3. Actual Time: 10 minutes
8. [✓](https://emojipedia.org/heavy-check-mark/) Each question / item / Quest has a skill(s) designation
   1. Difficulty: 2
   2. Time Estimation: 15 minutes
   3. Actual Time: 10 minutes
9. [✓](https://emojipedia.org/heavy-check-mark/) Quests can contain multiple combinations of skills or skill levels
   1. Difficulty: 3
   2. Time Estimation: 30 minutes
   3. Actual Time: 45 minutes
10. [✓](https://emojipedia.org/heavy-check-mark/) Keep track of individual student progress
    1. Difficulty: 2
    2. Time Estimation: 15 minutes
    3. Actual Time: 25 minutes

1. [✓](https://emojipedia.org/heavy-check-mark/)Allows students to be members of various defined groups (guilds)
   1. Difficulty: 2
   2. Time Estimation: 20 minutes
   3. Actual Time: 5 minutes
2. [✓](https://emojipedia.org/heavy-check-mark/)Generate reports in various ways (i.e., individual quests, individual skills, individual students, specific groups, etc.)
   1. Difficulty: 4
   2. Time Estimation: 1 hour
   3. Actual Time: 50 minutes
3. [✓](https://emojipedia.org/heavy-check-mark/) Calculate the current level for each student
   1. Difficulty: 3
   2. Time Estimation: 30 minutes
   3. Actual Time: 15 minutes

● Paired Programming Chart



● Summary and Refactoring Log

Overall, everything seems to be working as expected. We found out that trying to link the Processing code from our trivia GUI was too difficult and time consuming, so we decided to make the trivia game in JavaScript and handle the visuals with CSS. We linked some pre-existing homework assignments, quizzes, projects, and exams onto the student page, and gave them a button to check for new uploads. We have two premade quests for the student to take, and a button for the student to check for new quest uploads. The admin page has the upload buttons and functions, specifically categorized into homework, quizzes, projects, exams, and quests, in order to better organize it. The admin has an option to download a quest template document to customize and upload to the student. We also gave the admin instructions to upload only PDF documents and restricted the file size.

With regards to refactoring, here is a list of the major refactorings we performed:

* We decided to use JavaScript and CSS to make the quests, instead of trying to link our Processing code from our previous project submission.
* We decided to make our website designed for only one homeschooled student, in order to maintain the KISS principle, instead of keeping track of 30+ student pages.
* We decided to “unit test” our code in the console instead of using a testing framework.
* We decided to give the admin an option to download and customize a template for their student, instead of allowing them to create their own quest game.
* We decided to create separate directories in the server for Homework, Quizzes, Projects, Exams, and Quests, in order to better organize the assignments and make it easier to look at.