## **Milestone 1:**

* Performed some team building exercises in order to get everyone acquainted with the rest of the team
* Took a tally of everyone's skills and risks
* Designed a decision matrix in such a way that the team can build on already common common and familiar strength and avoid risks -later to be mitigated if possible-
* The team will be work on designing mini-milestones that fulfill the goals of deliverable milestone
  + This will benefit the team by having increased focus on important and high risk components
* Assigned team members individual task in order to prepare for ID-1
  + All team members are to research about testing standards - specifically Web application testing
  + Mahmoud is start looking into test plans and best practices
  + Jordan is to look into Selenium Webdriver and its applications with respect to the project
  + Bengin is gather and monitor risks involved with each mini-milestone and find means of responding to such risks
  + Angela is to look into into Robot framework and how it can be used to automate testing
* Jordan was Assigned to be the backup team lead in case Mahmoud goes MIA
* We are a still in deliberation to assign backup members for the rest of the team members

#### **Plans for ID-2:**

* finalize methodologies and technologies for the testing team
* construct a testing plan
* write up of basic smoke tests
* write up of basic feature tests

## **Milestone 2:**

* The team is to build on already gathered information form milestone 1
* Due to change of technologies used by the dev team the testing will have to wait until such technologies are finalized
* The testing team will be using the following technologies in order to test various aspects of the product:
  + QUnit for JavaScript unit testing
  + Selenium Webdriver for automated testing of the web app
  + Selenium grid for parallel testing
  + JMeter or TestNG to used along Selenium grid in order to perform stress and load tests
  + Robot framework for automated and keyword-driven testing
* Levels of testing are:
  + Alpha testing (AKA Unit testing)
    - Utilize QUnit framework to facilitate easier unit testing
    - TDD is strongly recommended
  + System and Feature tests
    - Includes Smoke test, Feature tests
    - System integration tests will be coupled with the two test previously mentioned
  + Regression testing
    - used to verify if defects have been fixed
  + Performance and load tests
  + User acceptance tests
    - Will use the tractability matrix against requirements mentioned in SRD
* The risks analysed from ID-1 will be listed in the master testing plan along with the response and trigger actions
* An idea was proposed to create a separate repo for test scripts which we can update and then test engine (test harness) can bring and execute the latest copy of each script
* Another idea was to create a database will use PHP and PostgreSQL to store and create test scenarios, test cases and test scripts

#### **Plans for ID-3:**

* Task and labor division will take effect once implementation team finalizes their plans
* write up of smoke tests
* write up of core feature tests
* write and script systematically elicited use cases
* prepare test suits for performance and load test - to be deployed in ID-4 -

## **Milestone 3:**

* We have created automation tests using Selenium
  + Tests authentication and the creation of new objects in the canvas
  + Integrated with Travis-CI
* Creation of the testing matrix
  + Information given in the test scenario is incorporated into the testing matrix
* Populated the testing scenario
* Added test cases in the testing scenario
* Added some of the implicit requirements to the testing scenario
* Minor updates to the master testing plan

#### **Plans for ID-4**

* Unit tests
  + Usage of QUnit for javascript unit tests
* Coverage testing/code coverage
  + Research into istanbul, blanket.js, and other coverage testing tools
* Mocking
  + Use a mocking plugin like jsMockito (will be evaluated which mocking tool is the best choice)
* Continue recording completed tests in matrix
* Continue adding test cases to the document
* Additional Selenium tests
* Path coverage diagrams

## **Milestone 4:**

* Unit tests
  + Tested methods in js using QUnit
  + Coverage overall is about 60%
  + Mocking with spys, stubs and mocks using sinon are used in some unit tests
* Coverage
  + Coverage is done using blanket.js
* Syntax/sanity tests
  + Tests all user created code to see if it has proper syntax and runnable code
  + Ran via JSHint via Grunt
* Selenium tests
  + Selenium is more adaptable and flexible to future builds
    - Checks other ways to approach test. Example is if Google login has changed it will try a different method to test.
    - Doesn't use css paths but xpaths to find a general approach to finding the object
  + Selenium now tests if a file was created by the system.
* Automation tests
  + Currently runs syntax tests, unit tests, and selenium tests
  + If unit tests or selenium tests fail the app will be marked with a fail
  + If syntax tests fail the app will be marked with a warning
  + Automated syntax tests are ran via JSHint on Grunt and automated unit tests are ran on Qunit via Grunt.
* Bug Reports
  + All bugs will be reported on GitHub's issue tracker.
  + If they are fixed they will be marked as resolved
* Path Coverage Diagram
  + Diagram using a flowchart to demonstrate the general flow of the program
  + Helps give a general sense on what to tests manually when a new build is pushed
* Added additional test cases to the document and updated testing matrix
  + It appears that in general system tests, feature tests, integration tests, and user acceptance tests are very close to each other in terms of approach
    - Test types may change in the future or test scenarios might change type.

#### **Plans for ID-5**

* Evaluation of further test cases and see what's testable and what isn't
  + If it isn't testable there may need to be changes in the code itself
* More unit tests
* Stress testing
  + Test to see if it can handle a lot of users or high connection
* Possibility of BDD
  + Use of Jasmine however it seems that BDD seems a lot like TDD
* More bug reports