# Sprint-02 Grading Rubric

## Grading Style

The second sprint report will be defined by your own group. The goals per person will be laid out based on your User Story and you will break the work into parts and based on your goals, you will essentially grade yourself.

## Points

The assignment will be 100 points cumulative.

* 20 points for the UI/UX portion
* 20 points for the Security Assumptions
* 20 points for IT and Infrastructure Report
* 20 points for Developer and Coding assumptions
* 10 points for the cumulative report and listing of goals for the next sprint
* 10 points for the Presentation style
  + The grading for the presentation will be a standard presentation
  + Was there a clear opening?
  + Was there a clear explanation of the entire report content (not read of the page)?
  + Was there a clear conclusion?

Each section will be graded on a Likert Scale:

* Meets expectations
* Somewhat meets expectations
* Somewhat meets expectations, with a few missing items
* Doesn’t meet expectations
* Non-effort
* 20, 16, 15, 12, and 5 points respectively
* 10, 8, 7, 6, and 5 points respectively

## Content

The deliverable will be a single file named **sprint-02.md** located in a folder named **sprint-02** under the **reports** folder in your team GitHub repo. The main goal here will be that each member will determine a minimum of 5 goals for themselves for the sprint. They will pick at least 1 from my list below, and the other 4 will be related to their job. These minimum 5 goals will be your own grading rubric for the sprint. Your report will detail the completion of these items. Those items include:

### Project Base Requirements

Each Project will have additional specific details to be delivered once the exact project subject is determined. But in this case we are going to disallow non-frameworked PHP and Google Firebase based applications. Note some of the requirements below span across multiple categories.

### Coding and Programming

* Language and framework of your choice (ITM 311, ITM 312, ITMD 361 & 362, ITMD 411) Suggestions:
  + Android
  + PHP – Cake/Symphony/Zend
  + Java – Spring/JSP/Hibernate
  + JavaScript – NodeJS/Angular/React/Vue.js
  + Ruby – Ruby on Rails
  + C# - ASP.NET
* Application Database Reads and Writes should go to different databases (see the replication section in IT Operations)

### Infrastructure and IT

* Operating System Platform of Choice (ITM 301, ITMO 356)
* Use of Data Store (ITMD 321, ITMD 411, ITMO 356)
  + Database or similar storage technology
* Datastore makes use of master/slave replication (~ITMD 321, ITMD 411)
  + Master for database writes
  + Slave for database reads
* Creation of Data Schema
* Creation of infrastructure diagram tool and work flow (Visio or comparable) (ITM 301, ITMO 356)
* Team must generate at least 15 real “test” users and proper data to test functionality of a system
  + No system is ever used “blank” always fill it up with real data.

### Security

* Data encrypted at rest (ITMS 448)
* Use of https (ITMS 448, ITMO 356)
  + Self-signed certs
* Login authentication mechanism. Google authentication for login, such as:
  + [SQRL](https://www.grc.com/sqrl/sqrl.htm)
  + [Google OAuth2](https://developers.google.com/identity/protocols/OAuth2)

### UI/UX

* Use of Responsive Design (where applicable) (ITMD 361, ITMD 362)
* Use of user authentication (ITMD 411)
  + Must use HTTP Session
  + Different UI for Unauthenticated users
  + Have read/only features for unauthenticated users
  + Different UI for Authenticated users
  + Have a user account management page (EDIT Page)
  + Different UI for Administrative users
  + Have features for Operations in application
    - database dump
    - database restore feature
    - turn any/an upload feature to read-only
  + UI is modified per authenticated user via CSS
* Layout design (ITMD 361, 362, ITMM 471)
  + Diagrams of site functionality using layout tool
  + Diagrams of colors, fonts, and other usability features

### Project Management

* Management of project progress (ITMM 471)
  + Trello (not provided) or JIRA (account will be provided)
  + Slack
  + GitHub
  + GitHub Issues to resolve bug posts from UI/UX tester
* User story - how the user will experience the site
* Focus on allocation of resources to help others meet their goals

### IT Operations

* Application needs to include:
  + A load balancer
  + minimum of 2 front-ends
  + A memory caching layer, such as [Redis](https://redis.io/) or [Memcached](https://www.memcached.org/)
* Database needs to have 15 users with actual data contained at deploy time
  + Needs to have Master Slave replication and or 3 nodes for replication
* Creation of Dev Environment on local laptop (ITMO 356) (ITM 301)
  + Application must always be in a working state
  + Each team member must be able to deploy the entire environment via script on their own hardware
  + Environment must be configurable via a scriptable deploy (suggested to use Packer and Vagrant)
  + No manual editing or installing
    - Note - Test and Production environments will be built after sprint 03

## Written Report Content

At the start of each sprint each person will contribute their deliverables. This list of atomic ~~events~~ tasks will be used to grade each person’s contribution to the overall project for each sprint. The project manager will be responsible to collecting the team’s input and prepare a report. You will receive feedback from me on this report. The report should state the following at the minimum with additional information will be released per sprint.

* List the team members and describe their functions during the spring
* Describe the groups stated (atomic) goals for the sprint
  + Describe how each of these goals were accomplished
  + Explain and show how bugs were found, tracked, and delegated
  + Explain any security assumptions made and explain how these were discovered and mitigated
* List the goals for the next sprint
* Each team member should submit a small comment of their own accomplishments and describe what they did and explain any decisions referencing any lectures or material from the text book or web
* Reference decisions from any of the videos, guest lectures, and class presentations, or text book
* Also in the main directory of the team GitHub repo, there needs to be an **install.md** file with instructions and assumptions of how to install and run your project

## Report

This story will weave together the contributions of each team member into a single narrative.

## Deliverable

The Project Manager will submit the URL to the sprint-02.md on GitHub, it will be a group submission so only one is needed. Don’t forget!