# How Faculty Will Mark your Milestone 2

Faculty marking workflow:

* Installed XAMPP to run MySQL (you could install MySQL separate but it’s a lot more involved – up to you, although you do get MySQL Workbench with it)
  + If you don’t use MySQL Workbench, you can use mysqldump at the command line/terminal
* Installed: MongoDB (along with the MongoDB Compass client app that comes with it)
* Students are providing the SQL code in a .sql file if they choose MySQL
* Students are providing the JSON code in a .json file if they choose MongoDB
* Copy/paste/import the data store records into the data store
* Next, run node at the command line to start the web app

How faculty check everything:

1. Connect to localhost:8000 (they should be using 8000, if not deduct a mark)
   1. With Node.js, you’ll have to ensure that you have all of the libraries that they are using installed globally (e.g., express, express-routing, etc.); you install once and it always works for all assignments. You may need to install the odd package that they are using but most students will use the ones prescribed
   2. You will be able to see the user accounts and the passwords from the data store code (MySQL or JSON)
   3. Keep the developer tools console up in your browser and look for errors or console.log output
   4. In either account, it’s okay if they are two separate paths, but it’s more elegant if it’s the same path, just different content displayed based on the type of account being used
2. In both paths (login/landing page, profile/dashboard page), see if the are utilizing media queries – specifically, does the layout change at all? In the very least fonts should be different as should images (e.g., smaller fonts with a more narrow layout)
3. Login to an admin account and add a user, edit another user, and delete a user, refresh and see if the changes were saved to the DB
4. Log out of the admin account and try to log into the account that was deleted (you should see a message stating no credentials of that type are found)
5. Login to a regular user account and change the avatar, change the password, change the email address, refresh the page and show that those changes were made on the server/DB side
6. Spot check the server code to ensure that they are using sessions, also ensure that they are using the data store (either MySQL or MongoDB) to grab user data and that they’ve not hardcoded user credentials into their server code
7. They should also be naming their tables/collections <name of group>\_user and <name of group> \_timeline. So for example BBY\_24\_user for group 24, and so on. This is so that we don’t have delete a million databases in MySQL and MongoDB – all of their tables will be within one database with no naming collisions, the DB name should be COMP2800

**Note:** *once you’ve marked a couple of these, it should flow fairly quickly. Doing spot checks on the server code to ensure that they are using sessions and a data store is really all that is needed. The rest of the code you shouldn’t need to spend too much time on since the app simply won’t work otherwise.*

# Milestone 2 Rubric

|  |  |  |
| --- | --- | --- |
| **Component** | **Marks weight** | **What team earned** |
| Web design template (HTML/CSS/JS) is high fidelity and contains components such as header, footer, logo, text that is specific to the app – or any combination. They’ve imported their own fonts (no default browser fonts), color has been added, at least one image has been added, responsive (so at least one media query), layout looks good in both resolutions | 7 | 7 |
| Datastore code in separate file so that we can put that code into the datastore (e.g., .sql for MySQL and .json for MongoDB) | 4 | 4 |
| CSS is in an external stylesheet | 1 | 1 |
| JS is in an externa stylesheet | 1 | 1 |
| Admin user dashboard allows for CRUD | 6 | 6 |
| Regular user page allows for editing of the following credentials: add/update avatar, password, email address, name | 6 | 6 |
| **Total earnable marks** | **25** | **25** |

# Additional Details

What you may lose marks on (each one up to a total of 3 being removed from the final grade):

* Runtime errors anywhere
* Not using “strict”
* HTML/CSS not valid
* Links to various resources are broken
* Using variable names that are from the example applications (e.g., session-app, assignment4.js, etc.)
* Boilerplate content (e.g., placeholder images, lorem ipsum)
* Dead code (code that is from examples that was left in place, commented out sections of code)
* Console.log messages
* Not including a readme.txt file
* Readme.txt file is misrepresentative of the work submitted (e.g., 100% complete when it’s very clearly less than 50% complete)
* Not on port 8000