DOMOMAKER - E

This assignment is due by the due date on the dropbox.

This exercise is the fifth part of a multipart assignment to build a small MVC-based login system with some basic per-user functionality.

Congrats, you built an MVC powered app. Now it's your chance to shine ☺

In this section, nothing is given to you. You need to build more features for the app yourself. Some of these features or the architecture may feed directly into how you do your second project.

For this assignment you need to:

Part 1:

- Make a new attribute in a Model (Account, Domo or a new model)

- Make a new form input that matches the attribute you made in a Model. This input may be in the Signup, Login, App or a new page.

- Have a controller send back your new attribute for display on the page.

There are other steps that will be necessary to hook this all together (such as a controller accepting the new attribute and saving it with the model).

Part 1 Example: Adding an attribute so that Domos have a level (like in a game) or a height. The user would be able to enter this on the form and see it displayed on each Domo.

Note: Entries already in the database will not have this new attribute. You can clear old entries out using MongoDB Compass.

Part 2:

- Add an entirely new feature to the app. This may be a small feature such as just a new page with some content. Alternatively, it could be a more advanced feature, such as a adding the ability to delete a Domo or alter a Domo.

  Again, you may need to clear out a Mongo database collection for this to work correctly. Otherwise, if you are editing models for your feature, you may end up with old models that don't fit the new format.

  I'd suggest testing a feature you think you might do in your second project. Make use of the React and Mongoose documentation as needed.

ESLint & Code Quality

You are required to use ESLint with the AirBnB configuration for code quality checks. The eslintrc config file was given to you in the starter files. Your **server** code must pass ESLint to get credit for this part. Warnings are okay, but errors must be fixed. I won't require ESLint on the client-side code in this assignment. ***If you cannot fix an error or need help, please ask us!***

You are also required to use the "pretest" script in order to run ESLint. I should be able to run "*npm test*" and have ESLint scan your code for me.

Continuous Integration

You are required to use CircleCI for continuous integration. Your build must succeed and be handed in with the assignment. The circle.yml config file was given to you in the starter files. CircleCI will build based on your *npm test* and ESLint configuration.

For instructions on how to do this, look at the ***CircleCI Setup*** instructions on mycourses.

Build & Bundling Scripts

Your app should have custom scripts to automatically restart node upon server changes (I suggest using nodemon) and create a client-side JS bundle (Webpack is installed and configured to do this with the "npm run webpack" command).

You should write your JS in separate files and create a bundled JS file from them. The only JS file that should go to the client is the bundled file.

DOMOMAKER - E

Rubric

DESCRIPTION	SCORE	VALUE %
App works correctly – all pages, forms and functions work correctly. App.js starts and runs correctly.		**25**
Updated Model – One of the models has a new field or a new model has been created appropriately. Model can successfully save and load from the database.		**15**
Working Form with new attribute – Form should allow for the new updated model fields. Form successfully submits the new field to the server.		**15**
New Data is Shown in the App – Data is pulled from the updated model and shown to the user. The new attribute(s) is displayed to the user.		**20**
New Feature on the App – App should have an entirely new feature. The new feature should work correctly every time.		**25**
Git Penality – If your code is not in a Git repo or the link is not handed in, there will be a penalty.		**-30% (this time)**
Heroku Penalty – If your app is not functional on Heroku, there will be a penalty. This means you need to correctly hookup the addons on Heroku.		**-30% (this time)**
CircleCI Penalty – If your build is not successful on CircleCI, there will be a penalty.		**-30% (this time)**
Build and Bundling Penalty - If your package.json does not include properly working automatic build scripts for Node (nodemon or other) and client-side bundling scripts (babel, webpack, browserify, rollup or other), there will be a penalty.		**-20% (this time)**
ESLint Penalties – I should be able to run 'npm test' and have ESLint scan your code without errors. Warnings are acceptable. **You may not alter the rules in the .eslintrc file provided.** **Check your code with ESLint often during development!** **If you cannot fix an error, please ask us!**	1 Error 2 Errors 3 Errors 4 Errors 5+ Errors	**-10%** **-15%** **-20%** **-25%** **-35%**
Additional Penalties – These are point deductions for run time errors, poorly written code or improper code. There are no set values for penalties. The more penalties you make, the more points you will lose.		
TOTAL		**100%**

Submission:

By the due date please submit a zip of your project to the dropbox. **Do not** include the node_modules folder in the zip or in your git repo.

In the submission comments include the following:

- *a link to your Heroku App (not the dashboard but the working web link from 'open app'),*
- *a link to your Github repo (if private, please add the TA and myself).*
- a link to your circleCI build *in the submission comments.*