Code for Grading

Sam Edson, Sabrina Drammis, Grant Gunnison, Danny Sanchez

Basic Coding

Modularity

util methods super useful.

Verification

Mongoose validation

- validator package - emails, escaping
- Won't save unless passes validator
- Prevent empty / too short / not formatted correctly

Authenticated and certain type of user for different routes

- util methods super useful

Future plans

- more robust
- check all requests and make sure we don't let invalid requests modify our database.

Security

We take a number of steps to address security issues. For starters, we equip our database with an escaping mechanism to make our text in HTML format. Luckily, Mongoose has a *pre* middleware method that functions on the input of a database action. We also use the Node package called Validator that has an escape() function that replaces special characters with their percent escaped equivalents. Together, we use them, for example, in a Mongoose *pre* save on an Employer's name and company name. This would mess up most Javascript code that could be injected, and also makes our data HTML friendly.

We use the Node Module, Helmet, to enforce our Content Security Policy. We make sure no inline Javascript is allowed. Also, Helmet has an xssFilter that our app can use to prevent cross-site scripting by setting the *X-XSS-Protection* HTTP header. This isn't a complete CSP, so we define our own cross site scripting rules, which only let code run from jquery, ajax, and self. We do the same for style, allowing jquery, bootstrapcdn, and self. This does a lot to prevents code from other web sites being run, preventing many cross-site scripting attacks. Lastly through Helmet, we tell our app to set *frameguard* to *deny*. This ensures that our site does not become a div or encapsulated by another site, restricting our content to our own site.

We originally wanted to use *csurf* to prevent cross site scripting, but it was proving difficult to test with because it prevented our ajax requests to connect to the server. We may want to use it for the final product, but for now we are leaving it out because much of what it does is prevented by Helmet.

We also put some protection on our cookies. We make the session cookie httpOnly because it ensures that clients cannot access the cookie from their own code.

Lastly, we disable x-powered-by so that hackers don't know we are using Express with Node.