Requirements

- Within a new or existing Express application, implement user registration, user authentication, and basic user authorization.
  - For registration, create a simple user model using Mongoose, and use the bcrypt library for Node.js to hash user passwords.
  - For authentication, use the Passport middleware along with its “local” strategy to enable session-based authentication.
  - For authorization, leverage Passport’s built-in session deserialization to implement basic logic for determining your own rules for authorization. For example, you might simply restrict access to all data unless the user is authenticated, or you might established record ownership for a resource and restrict access to only records which are owned by the authenticated user.
- Within a new or existing AngularJS application, implement user interfaces and user flows for all of the above, including registration, authentication, and authorization.
  - Implement the requests necessary to communicate with your web service for registration, authentication, and retrieving authorized resource data.
  - Implement forms which allow the user to register and authenticate, as well as UI elements to display them as appropriate.
  - Display appropriate messaging for any error conditions, including any validation errors and authentication failures.
  - Implement the logic necessary to reveal (or hide) UI elements as part of the authentication and authorization user flows.
- Deploy your finished application (including both the Node.js server application and the AngularJS client application) to Heroku.

Resources

- Passport documentation
- bcrypt package

Submission

1. Show your completed assignment to the instructor during class or office hours to receive credit.
2. Submit your project using Git and GitHub. Start by creating a repo for this assignment here.