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 Vue.js 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 Vue.js client application) to Heroku.

resources

• Passport documentation

• bcrypt package

submission

1. Submit your project using Git and GitHub. Start by creating a repo for this assignment here.

2. To pass off your project (required to receive credit), choose one of the following two options:
  • Show your completed assignment to the instructor during class or office hours using Google Meet.
  • Alternatively, you may record and submit to Canvas an audio/video screen capture that demonstrates ALL of the following aspects of your completed project. Use your recorded voice to narrate your demonstration. If one or more aspects described below are not shown, points will be deducted accordingly.

Using Google Chrome, with the Network tab open and the first two columns visible (name and status), do the following:

1. Navigate to your deployed application and refresh the page to show an initially logged-out state.
2. Explain the restrictions in place when the user is logged out. (e.g. any data and/or UI elements that are intentionally not displayed or otherwise not functioning).
3. Attempt to register using an existing email address, and show that it fails with an error message.
4. Attempt to register using a new email address, and show that it succeeds.
5. Attempt to login using this new email address and an **incorrect** password, and show that it fails with an error message.
6. Attempt to login using this new email address and the correct password, and show that it succeeds.
7. Demonstrate that your application now functions differently once the user has authenticated. (e.g. any previously hidden or non-functioning data and/or UI elements that are now revealed or functioning).
8. Reload the page again to show that the user remains authenticated and that the UI state is unchanged.
9. Briefly, show several examples of **authorization** (not authentication) logic/rules within your **server** code.