****** PRELIMINARY VERSION ******

Assignment: Node Project 01 – SSD Yearbook

Course Value: 50%

Assignment Description:

Build a class yearbook application including role-based auth.

Teamwork:

Although not a requirement, I highly encourage all of you to try and work in teams to try and complete this assignment. Working in teams will allow you to pool your knowledge with your fellow classmates and to help each other solve some coding challenges.

Instructions:

1. Refactor your Mongo CRUD assignment to use Passport for Authentication, with sessions stored in your MongoDB Atlas cluster and role-based access controls (Authorization).
2. Users should be able to register themselves and login/logout as well as browse other profiles and edit their own profile including email, first name, last name, interests, and profile photo. They should not be able to edit their username or roles.
3. Users should be also be able to leave comments on profile pages.
4. Managers should be able to browse and edit the roles any user, but not delete or otherwise edit anyone other than themselves.
5. Admins should be able to perform all of the above as well as edit and delete users.
6. When your team has completed this assignment, only one of you needs to upload it to the Learning Hub, indicating the team members in your submission comment as well as the contributors property within package.json. Remember to delete the node_modules before zipping up and submitting your work.

Specifications

- Create a MongoDB, Express, Node.js web application that provides role-based functionality for viewing and administering users.
- To be considered complete, there must be:
  - A navigation bar and footer
  - A Home Page
  - A Register Page
  - A Login Page
  - A User listing page, showing each user’s name (first and last), sorted alphabetically by last name.
  - A User detail page
  - The ability to register as a new user with username, email, password, first name, last name, interests, and a profile photo.
  - Password encryption and user Authentication should be handled via PassportJS with local strategy for storage in your MongoDB Atlas cluster

- Users should have the ability to log in and out of the application with a 20 minute session timeout that re-saves with each request.

Unauthenticated Users

- Before logging in, users should only be able to see the home page, registration page, and login page.

Basic (Authenticated) Users

- Once logged in, a user should be redirected to their personal profile page
  - The detail page should display all of user fields, with the exception of _id, salt, hash, and __v.
  - There should be a sidebar containing links to all other profiles.
- Users should be able to edit their own profile, with the exception of their username and roles.
- Users should be able to browse other profiles, both via links in the sidebar on each profile page as well as from the user listing page.
- Users should be able to add comments on other profile pages.
- The user listing page should allow searching/filtering by any of the name fields, email address, or interests.
- Users cannot edit any aspect of other users unless they have additional permissions granted via roles.

Managers

- In addition to the functionality allowed as an Authenticated user, Managers should be able to edit the roles for themselves or other users.

Admins

- In addition to the functionality allowed as an Authenticated user and as a Manager, Admins should be able to edit and delete other users.

Starter Files

- You are free to use your code from Assignment 03 or from the Day 05 Demo as starter files, or you can create your application from scratch. Either option is fine.

Marking Criteria:

This project will be marked out of 50 and will be marked based on the following criteria:

- User Profile and comment CRUD:	**10**
- Passport Authentication:	**10**
- Session Management	**10**
- Role-based Authorization	**10**
- Search Functionality	**05**
- UI and Styling	**05**