Trent University
COIS 3420H
Culminating Project – Winter 2020

The culminating project for this course is designed to give you experience implementing the concepts discussed all together in a full web application. The application itself is based on the following premise:

You have been hired to build a web application that allows users create an online "bucket list". The application allows users to create, organize and manage one (or more) lists of bucket items. As bucket list items are completed, the user can mark them off by adding a completion date, details, and a picture. The user can chose to make their list private, or have a publicly viewable link to share their progress with others. Your application must have the following general functionality:

- The ability to for a user to register, edit and delete their own account
- The ability to manage (view, edit) list information, which at minimum should include name, description, and if they list is publicly viewable
    - If you chose to allow multiple lists, you would also need add/delete
    - If the app only has 1 list per user, you could include this information on the account page
- The ability to add, edit, view and delete bucket list items from a *manage list* page
    - Adding an item should be done using a modal window, which only the basics (like name and public view status)
    - Editing the item should allow them to change the title and view status as well as allow items to be marked as completed with (at minimum) completion date, description, image upload, along with anything else you think would be good.
    - Usually the cleanest way to incorporate (edit, delete, info/view) links on an item-by-item basis is using icons (I recommend FontAwesome as an icon library for its comprehensiveness and ease of use)
    - Viewing a particular bucket item should generate a modal window with all the details for an item displayed in a nicely formatted manner
- A *display list* page, which provides a public facing view of the list for sharing (no edit or delete buttons) where each completed item should be a clickable link to the details page for that item
    - From a security perspective, this page should only be viewable if the list is flagged as public, or if the person who owns it is logged it.
- Login functionality, with a password reset, and remember-me feature, as well as a logout
- All pages except the application home (splash page), login and register, and public views of user's lists should redirect back to login if the user isn't logged in
- The ability to search other people's publicly viewable list items for ideas. This should also include something similar to Google's "I feel lucky" which randomly choses something off another users bucket list (which is publicly viewable) as a suggestion

Other Requirements:

- Your menu should be dynamic so users only see links that are relevant (i.e logged in users should see logout, logged out users should see login)
- Usernames in your system should be unique, and your register page should include an ajax call when the field loses focus to verify this before form submit (we'll do something similar in lab)
- You must use at least 2 javascript/jquery plugins (I'd suggest one for password strength)
- Edit pages must pre-populate all existing data so the user can **edit**.
- Delete links must include a JavaScript confirmation dialog for verification before continuing
- Security considerations must include (at minimum) how you deal with passwords and protecting against XXS and SQL injection with collected form data
- Details collected about the list item should also be "HTML-ifed" so that it displays correctly on the details page
- Design your forms keeping in mind the concepts of usability and data integrity. Also keep these concepts in mind when deciding what form fields are required (Everything and nothing are both bad choices)
- Every form should have both PHP and JavaScript validation. Form fields that aren't required should still be validated if something is entered (when appropriate)
- **Note: all front-facing forms should be self-processing to improve usability of error display**

Like with the final question of Assignment #1, the actual design of your application is up to you. You're welcome to find a design you like and try to replicate it, but you should not be copying large chunks of code from anywhere. Your design should not include the use of any external libraries, frameworks or templates (except an icon library like Font Awesome). I may approve the use of another library if you ask. (But will say no to Bootstrap)

Note: Attempting things on the listed ***Advanced Topics***, will get you extra complexity marks (particularly if you're working in a big group where expectations are slightly higher). Since you're not actually being marked at Check-in, you can complete Advanced Topics at any time.

Check-In One

For the first check-in period you should have finalized your design and completed the HTML/CSS for all the front-facing pages.

Although pages like List, Search and Details will eventually be populated using database information, for Check-in One, hard-code enough data to finalize what your design should look like (you can then replace the hard-coded data with the relevant PHP later).

You can ignore pages like edit account (which from a design perspective identical to add) and delete and logout (which have no front-end at all)

You're not required to have any php or javascript at this check-in

Check-in Two

For the second check-in period you should have created your database tables and completed the back-end functionality for all pages. This should include the login/session component, all php to process your forms (add/edit/search), replacing the hardcoded data on display pages with database calls, and the implementation of the delete and logout pages. You are not required to have any JavaScript completed at this check-in