Zachary DeVita
CS 496
Assignment#2

URL: http://assignment2.z9cqkpjzc2.us-west-2.elasticbeanstalk.com/

My website contains a registration page. It contains much of the commonly required personal information which is found on any registration page. Personal info required includes: user name, email, password for account, first and last name, address information, gender, and it also requires the user to "accept the terms and conditions" of the website. Required information for the user to enter is denoted by an asterisk, and includes the user name, password, a repeated entry of the password, first and last name, gender, and the checkbox for accepting the terms must be checked. The address information is optional on my registration page, and this includes street info, city, state and zip code.

When the register button is clicked, and the information is valid, the user info gets stored into an sql database. This is done using AJAX calls from a client side javascript, storing the information in JSON objects and sending it to server side javascript. Server side javascript then uses the appropriate sql query to store the info into the database. The data is retrieved much the same way; through an AJAX request on client side, then returned by the server javascript after obtaining the info from the database.

Test Cases:

1. Input: Save user with no user name.
   Expected result: user is not saved and a message indicating user name is required is displayed.

2. Input: Save user with no email.
   Expected result: user is not saved and a message indicating email is required is displayed.

3. Input: Save user with no password.
   Expected result: user is not saved and a message indicating password is required is displayed.

4. Input: Save user with password and repeated password not matching.
   Expected result: user is not saved and a message indicating passwords do not match is displayed.

5. Input: Save user with no first name.
   Expected result: user is not saved and a message indicating first name is required is displayed.

6. Input: Save user with no last name.
   Expected result: user is not saved and a message indicating last name is required is displayed.

7. Input: Save user with no street.
   Expected result: user is saved and no error is displayed.

8. Input: Save user with no city.
   Expected result: user is saved and no error is displayed.

9. Input: Save user with no state selected.

Expected result: user is saved and no error is displayed.

10. Input: Save user with no zip.
    Expected result: user is saved and no error is displayed.

11. Input: Save user without checking the “accept terms and conditions” checkbox.
    Expected result: user is not saved and a message indicating you must accept the terms and conditions is displayed.

12. Input: Save user with checking the “accept terms and conditions” checkbox.
    Expected result: user is saved and no error is displayed.

13. Input: Save user with numbers and symbols in user name.
    Expected result: user is saved and no error is displayed.

14. Input: Save email with multiple ‘@’ symbols.
    Expected result: user is not saved and a message indicating that the correct email format must be used is displayed.

15. Input: Save email with period as first character.
    Expected result: user is not saved and a message indicating that the correct email format must be used is displayed.

16. Input: Save email with period as last character.
    Expected result: user is not saved and a message indicating that the correct email format must be used is displayed.

17. Input: Save email that contains consecutive periods that do not include the first or last characters.
    Expected result: user is not saved and a message indicating that the correct email format must be used is displayed.

18. Input: Save email that ends with four letters after the last period.
    Expected result: user is not saved and a message indicating that the correct email format must be used is displayed.

19. Input: Save user where password contains symbols.
    Expected result: user is saved and no error is displayed.

20. Input: Save user where gender is male.
    Expected result: user is saved and gender is displayed correctly

21. Input: Save user where gender is female.
    Expected result: user is saved and gender is displayed correctly

22. Input: click on “Remove Account” button.

Expected result: the account is no longer displayed.

23. Input: click on the “edit” button.
Expected result: form is repopulated with the appropriate account’s information.

24. Input: click the “edit” button and change user name before saving.
Expected result: user is saved and user name is updated in account info.

25. Input: click the “edit” button and change email before saving.
Expected result: user is saved and email is updated in account info.

26. Input: click the “edit” button and change only the first password field before saving.
Expected result: user is not updated and a message indicating passwords don’t match is displayed.

27. Input: click the “edit” button and change only the second password field before saving.
Expected result: user is not updated and a message indicating passwords don’t match is displayed.

28. Input: click the “edit” button and change both of the password fields to matching values before saving.
Expected result: user is saved and password is updated in account info.

29. Input: click the “edit” button and change first name before saving.
Expected result: user is saved and first name is updated in account info.

30. Input: click the “edit” button and change last name before saving.
Expected result: user is saved and last name is updated in account info.

31. Input: click the “edit” button and change street before saving.
Expected result: user is saved and street is updated in account info.

32. Input: click the “edit” button and change city before saving.
Expected result: user is saved and city is updated in account info.

33. Input: click the “edit” button and change state before saving.
Expected result: user is saved and state is updated in account info.

34. Input: click the “edit” button and change zip before saving.
Expected result: user is saved and zip is updated in account info.

35. Input: click the “edit” button and change gender before saving.
Expected result: user is saved and gender is updated in account info.

Test Results:

1. success

2. success
3. success
4. success
5. success
6. success
7. success
8. success
9. success
10. success
11. success
12. success
13. success
14. failed: the invalid format of email was saved because the email validation does not take into consideration whether '@' symbols are used. The entire password could be comprised of '@' symbols.
15. failed: the invalid format of email was saved because the email validation does not check whether a 'period' is in the first character of the email.
16. failed: the invalid format of email was saved because the email validation does not check whether a 'period' is in the last character of the email.
17. failed: the invalid format of email was saved because the email validation does not check whether there are consecutive 'periods' in the email.
18. failed: the invalid format of email was saved because the email validation does not check whether the domain name is reasonable
19. success
20. success
21. success
22. success
23. success
24. success
25. success
26. success
27. success
28. success
29. success
30. success
31. success
32. success
33. success
34. success
35. success

I used a node.js environment for my webpage and I fully utilized the template system. I have a single layout which utilizes different views depending on which page is being displayed. I did only use a single view for this assignment so the templates were not necessary, but using this system early will make it more portable for the final project. Eventually, when multiple views are used, I will also incorporate some of the node.js conditionals with partials that are appropriate for the individual pages.

If I had to do this over, and I could make time stop for a few days, I would increase the data validation to consider valid formatting for emails, require at least 6 characters for the password, have the password show up as asterisks, revalidate password when attempting to change the password, set up an email option for users who forgot their password. I would also make the entire registration page a big modal, and have it only referenced by the homepage and saved as a partial. I would also show the account information on a user account page, which would only show their specific info, and would be displayed on a different view than the 'home' view.