Based on 100 points total, roughly 1.3 per check box unless noted as a %. Extra credit evaluated separately.

Lab 4 - Updating Records.

The purpose of this exercise is to:

> Create an admin portion of the site. This includes insert update and deleting of reccords. You will also need to use version control software. If that is not working you cannot receive a grade higher than 50% even if you have completed more than half of the check boxes.

As a reminder all labs are due by noon. Many classes have them due by midnight but I have extended all the due dates to noon the next day to allow you a little more time.

NOTE: When you make a commit (there are LOTS of commits) it is assumed that the code is working at that point.

Set up version control for this lab.

- [ ] Make a copy of your dev-lab3 folder naming it dev-lab4. i.e. just copy the whole folder and rename the copy as this will have all the files to start.
- [ ] Be sure to have a README.md with just your First and Last Name.
- [ ] Create a new private GitHub project in the class Repo named Lab-4-FirstName-LastName. For example mine would be Lab-4-Robert-Erickson.
- [ ] Verify that you have used the case and hypens in your repo name as detailed above.
- [ ] Keep the content of the first GitHub page saving them in gitcommands.txt overwriting the commands from lab 3. This file should contain only the commands for lab 4.
- [ ] Erase all previous commits from your dev-lab4 folder so we can start new. Be sure you are in the correct folder dev-lab4

```
rm -rf .git
```

- [ ] Set up your dev-lab4 folder to be a git repo (git init) on your computer.

```
git init
```

- [ ] Check if a remote is connected and remove it. It should already be deleted but it does not hurt to check.

```
git remote -v (will show you if a remote is there)

git remote rm origin (if one is there delete it)
```

- [ ] Now connect your local computer to your git repo:

```
git remote add origin command
```

- [ ] Make a commit after you sFTP your files and they are working as before.

Set up the live site.

- [ ] Using Terminal or git bash shell, connect to the server and get into your class folder creating the live folder (live-lab4)
- [ ] Initialize live-lab4 as a git repo.
- [ ] Connect your live folder with the GitHub repo.
- [ ] Pull your files live.
- [ ] After you pull always verify the site is working.

Setup a new database.

- [ ] Using the web site webdb.uvm.edu (at the botttom of the page) create a new database named cs148_lab4
- [ ] Copy the datea base tables (strucutre and data) from cs148_lab3 to LCARRERA_cs148_lab4
- [ ] update constants.php to the new database name.
- [ ] Make a commit for setting up the database connection.
- [ ] Make sure you ftp lib/pass.php to the live site.

Add to the admin menu

- [ ] Create an admin folder in your lab4 folder to hold the files for the admin links.
- [ ] Delete ADMIN from your .gitignore file. NOTE: it seems sometimes case does not matter and sometimes it does. So its best to just take ADMIN out of gitignore.
- [ ] You may use two div elements like w3schools example: https://www.w3schools.com/howto/howto_css_dropdown_navbar.asp to create the admin drop down links. NOTE: I suggest adding the css slowly as several rules you do not need and several you will need to modify to match your design.
- [ ] Create a sub link to allow the admin to add wildlife records (insert).
- [ ] Create a sub link to allow the admin to edit wildlife records (udpate and delete. NOTE: You can create a seperate link for update and delete.

Set up an admin folder.

- [ ] Set up the admin folder with top, header, nav and footer. NOTE: there are ways to not have extra files but setting up the paths can be tricky. It is easier to just have new files where you will still link to the same lib folder and css folder but you will need to add ../ to many paths. You can create a new nav as it is just for admins. Be sure to allow an option to exit admin and go back to the public site.

- [ ] Edit Database.php so the path to pass.php will work.

```php
$path = 'lib/';

if (substr(BASE_PATH, -6) == 'admin/'){
    $path = '../' . $path;
}

include  $path . 'pass.php';
```

The includes need to account for the current path.

Add new methods to your class file.

- [ ] Create a method to update records (its functionally the same as insert).

- [ ] Test the update method to make sure it works. Just make the function call sending it a valid sql update with the correct array.

```php
// testing your update
$sql  = 'UPDATE tblWildlife SET  fldCommonName = ? WHERE pmkWildlifeId = ?';
$data = array('Bob the Beaver', 1);
// first print out the sql:
print $thisDatabaseReader->displayQuery($sql, $data);

//after confirming the above query works then do the update call

$saved = $thisDatabaseWriter->update($sql, $data);
```

Testing to make sure your method works.

- [ ] Make a commit for this update method.

- [ ] Create a method to delete records (its functionally the same as insert).

- [ ] Test the delete method to make sure it works. Just like testing an update.

- [ ] Make a commit for this delete method.

Form to Insert New Wildlife Records

- [ ] Follow the guidelines to add one form elment at a time as shown below. Repeat for each form element.

Adding a form element guidlines

- [ ] A. Create and initialize (default value) a php variable for the form element.
- [ ] B. Add the new form object (input etc) to your form.
- [ ] C. Have the form element set to the default value as defined in A.
- [ ] D. In your form processing if statement sanitize the data from your form element.
- [ ] E. In your form processing if statement validate the data from your form element. Remember to create a flag if you should save the data.
- [ ] F. Make a commit for this stage of the form after verifing it works.
- [ ] Repeat A-F until your form is complete.

Display Wildlife Records to allow me to choose which record to update or delete.

- [ ] Create a page that will pass the primary key information to the insert form page for an update.
- [ ] Make a commit for this stage of the code after verifying your code works. Working means the page displays, you click a link it goes to form, add a print statement on the form page to verify the primaty key made it. After confirming just delete the print statement.
- [ ] Create a page that will pass the primary key information to a delete form page. Deletes must be a form submission so this page would be like a confirm you want to delete.
- [ ] Make a commit for this stage of the code after verifying your code works.

Update Wildlife Records - using the same insert form.

- [ ] Follow the guidelines to add functionality one step at a time. The form will need some minor additions to allow insert or update.
- [ ] Make a commit for this stage of the code after verifying your code works.
- [ ] My display critter page needed to be updated with nl2br (new line to br element) function in order to display the line breaks.

```
print '<h3>Description</h3>';
print nl2br($animal['fldDescription']) . PHP_EOL;
```

This preserves the blank lines from your text area input elements

- [ ] Make a commit for this stage of the code after verifying your code works.

Make Admin Login

- [ ] Create a table to hold the admins netid.
- [ ] Anyone can log in but only admins can see the pages to let them add update or delete.
- [ ] Make sure you copy admin/.htaccess to the live site.
- [ ] admin web pages need to use an .htaccess file allowing only rerickso, klee14 and yourself (who are in your adminstrator database).
- [ ] Create an admin report to display each animals common name, adotpers first name, adopters last name, donation amount and total amount this animal received in dontations. Sort the data by the common name.
- [ ] Create a sub link to allow the admin to see this Wildlife Dontation report.
- [ ] Create an admin report to display each adotpers first name, adopters last name, wildlifes common name, , donation amount and total amount this adoptered dontated. Sort the data by the adopters last name, first name.
- [ ] Create a sub link to allow the admin to see this Adopter Dontation report.

Standard Requirements

- [ ] Main goals of assignment met.
- [ ] Be sure DEBUG is set to false.
- [ ] Make sure you copy lib/pass.php to the live site.
- [ ] Update your main index.php file with links to all the files in the correct place (public and supporting).
- [ ] Do not use any banned elements (div, span, br).
- [ ] Be sure your text is still legible with and without the Color blindness tool.
- [ ] All files must use proper coding guidelines.

- [ ] Make sure paragraph, lists, forms text is not centered. Centering boxes is ok.
- [ ] (5% of your grade) Proper use of version control (commits on a regular basis, commits small with good message about what was accomplished with this commit) jigsaw.w3.org/css-validator/
- [ ] (5% of your grade) Make sure your files all pass w3c HTML validation: validator.w3.org/
- [ ] (5% of your grade) Make sure your files all pass w3c CSS validation: jigsaw.w3.org/css-validator/
- [ ] sFTP your files to your silk account which is our web server
- [ ] Submit your assignment only once in blackboard. See next direction first.
- [ ] Copy this code as a text submission in blackboard, look for the Write Submission button. You will need to click on the html button once you are in the editor and paste this code there:

```
<a target="_blank"
href="//lcarrera.w3.uvm.edu/cs148/live-lab4">
http://lcarrera.w3.uvm.edu/cs148/live-lab4</a>
```