Crc Development Timeline

March 26:

As of March 26, I have decided to keep documentation of technical detail regarding the progress of this application. I began this project roughly around March 17.

**The following information will contain progress and development from March 17 to March 26**

List of created important files:

* **Index.php**
* **Loginpage.php**
* **InputForm.php**
* **InputFormData.php**
* **InputFormHelper.php**
* **Session.php**
* **Db.php**
* **InputForm.css**
* **Loginpage.css**
* **inputForm.js**
* **inputForm2.js**
* **inputFormJQuery.js**
* **date.js (external library)**

List of important features that have been implemented within these files:

* Index.php
  + Original bootstrap template modified
    - Map section removed
    - About section removed
    - Download section removed
    - Default pictures replaced
    - Navigation bar tabs changed to linking to other php files instead of referring to a class within the same page
    - Admin Login page created in the Navigation bar
    - Contact page created in the Navigation bar
* Loginpage.php
  + Following (created) files are linked to this page:
    - Loginpage.css
      * Used for styling the div’s and inputs
      * Used for referring a background image
      * Hovering animation effect
    - Session.php
      * Used for creating a “Global” session variable so that other files can access inputted information from user
    - Db.php
      * Used for establishing a connection to a database to validate username and password
  + Within this file, php code is written, awaiting an input from the user.
  + PHP code will validate the inputted username and password or even if they have inputted information.
  + Sends a request to a server and extracts information and compares username and password
* inputForm.php
  + Following (created) files are linked to this page:
    - Session.php
      * This is included so that it can check if Session is set. If it is, then the page will execute as it should. Otherwise, it will bring the user back to the login page.
    - inputFormHelper.php
      * This file contains all the titles (Zones to the CRC) and all the times in an array.
      * This is used to create tables, as well as having a reference to keys to accessing certain inputs
    - inputFormData.php
      * contains methods select(), insert(), and delete() to interact with the database depending on what is selected from the inputForm page.
      * In insert() this file will automatically insert only the inputs that have been “set”, therefore ignoring the inputs that have been “disabled”
      * Select() will take all the rows associated with the current date (or the inputted date) and by using the associative properties (Zone, Time), map the “People” value into the inputted box. At this time, inputForm.js will change the type of input box to disabled and indicate to the user that it has been successfully inputted.
    - inputForm.css
      * Formats the inputForm.php
    - inputForm.js
      * Monitors the time and date of the page and updateClock() is executed every second.
      * changeType() (on the client side) changes the attributes of the inputted boxes from the table
    - inputForm2.js
      * Responsible for changing between tables in inputForm.php
    - inputFormJQuery.js
      * Animation effects
  + This php file is responsible for inserting and retrieving information about the number of people into and from a database.

**March 26 Update:**

* Played around with AJAX: Created the following files for testing (not implemented at all due to lack of purpose)
  + inputFormTime.php
  + inputFormAJAX.js
  + inputFormDataAJAX.php
* Files changed/added:
  + inputForm.php
    - Overall objective was to be able to switch between tables (based on their dates) and access the variables or data stored in the database specific to the table.
      * First attempt was to do both client and server side code, one where Javascript was used to update the visuals, while the server side was keeping track of a count so that it can retrieve from the database behind the scenes. However, the biggest problem was how to keep a global variable within another file….
      * Another problem was that “Submit” refreshed the page so that the server side code will be executed when data is being submitted from other dates other than the current. At the same time, I wanted to update dynamically the table and assign the new dates to a server variable and extract database information. This was impossible to do.
      * Decided to wipe out all the JavaScript that updates the date, and instead, use server side code to update the date.
      * “Next” and “Previous” were changed to type=”Submit” under a form of method=”POST” so that server side code can be updated when POST[‘action’] is set.
    - Biggest problem was to keep track of a global “date” so that when previous or next is pressed, the date can be automatically updated and read from certain methods so that the correct data corresponding to the date can be retrieved from the database.
      * This was done by creating a SESSION[‘Date’] where this would only execute once based on a SESSION[‘key’] where ‘key’ is just a random session to account for accessing this more than once if the page refreshes especially when next or previous was pressed.
      * Example: if(!isset($\_SESSION['function\_ran'])){
      * date\_default\_timezone\_set("America/New\_York");
      * $\_SESSION['Date'] = date("m/d/Y");
      * $\_SESSION['function\_ran'] = true;
      * }
    - Second objective was to create a logout function so that the user can successfully “end the session” if need to without having to close the browser.
      * To fully implement this functionality, the logout button was implemented into the navigation bar, which also required importing a JavaScript file to allow for consistency in animation.
      * Problems also occurred with the formatting, and how to account for detecting when the logout button was pressed.
      * Originally, it was in the form <a href> where a href links to the loginpage.php, however, this could not be detected by PHP
      * To fix this, a <button></button> was created within <a></a> and given a “name”, under the form with the method “post”
      * Session ends by session\_destroy() and session\_unset();
      * While href=”loginpage.php” will no longer work, instead, within the php file, header(location: /loginpage.php) could be used.
  + inputFormNavigation.php
    - This is simply the php file that will detect whether or not “logout” is pressed. If it is pressed, then it will end the session and take the user back to loginpage.php
  + inputFormData.php
    - A lot of changes were made to this so that it can be easier to read as well as following a little bit of Object-oriented design.
      * A class was defined as “InputFormData”
      * Methods were defined for each use of the database: Select(), Insert(), and Delete()
      * AS well as the next and previous functions where it will increment or decrement the date.
      * Learned a new way of incrementing or decrementing the date using $date1 = strtotime("+1 day", strtotime($date));
      * return date("m/d/Y", $date1);
      * An instance of the class is created within the page, and it should be executed after each submit or reload of the page.
      * Many if conditions are placed in here to account for the buttons pressed on the page. They will perform the specific method for the button that is pressed in the inputForm.php
    - Methods take in a parameter of the date so that it can USE it to retrieve , insert, or delete depending on the date that is provided.
  + inputForm.css
    - added a class: navigationButton for styling logout and other buttons in navigation bar
* Files No Longer needed as of **May 26:**
  + inputForm2.js
  + inputFormTime.php
  + inputFormAJAX.js
  + inputFormDataAJAX.php

March 27 Update:

Files changed:

* inputFormData.php / inputForm.php
  + Had to move the if conditional statements back into inputForm because $\_SESSION[‘Date’] apparently is undefined. (inputFormData seems to be executed before the script in inputForm)
  + Within method insert(), added conditional branch that selects from the database and see if it exists. If it doesn’t, then it will insert it, if it does exist, then it will update.
* inputForm.js
  + Created method typeToEdit()
    - Turns off “Disable” to allow for editing
    - Change background color to blue to indicate the ability to edit

April 4:

Using this website: <http://www.cybexintl.com/solutions/gymplanner.aspx>

Created a Gym floor plan that will be implemented into Index.php (Main Page)

Files changed:

* Index.php
  + Within section <id=map>, altered tags such that only a <div> contains a link to the gym floor plan image

Files added:

* Index.css
  + The attributes contained within this file allows the image of the gym floor plan to scale dynamically with the window size.

Required future implementations:

* Incorporating Javascript library, ImageMapster
  + With this library, allow users to hover over a section of the gym, and have it display certain values that’s specific to its zone
  + Must allow coordinates to resize dynamically using “resize” method provided.
  + Also must account for how it will display when accessing with a mobile device.

April 8:

Files changed:

* Index.php
  + Successfully added map and area coordinates indicating which sections are designated in the gym floor
  + Successfully added jQuery code with library ImageMapster that allows hovering effects
  + Learned that jqeury.min.js is for production only while jquery.js is used to test and develop
  + These two cannot be imported at the same time as it would take out functions
  + Had a hard time determining how to resize image proportionally as well as centering it
    - Solution was to tie <img> to a background url in index.css, and centering and resizing there. So when javascript code executes, the css will restrain it.

April 9:

Files changed:

* Index.php
  + Jquery.min conflicted with jquery.js (jquery.min was executed first in header, and jquery.js was executed last)
    - Problem was that menu bar no longer toggled
    - Solution: created separate jquery file that manually toggles the menu

Files Added:

* Toggle.js

**April 10, 2016:**

Files Changed:

* Index.php
  + Added header titles
* InputForm.php
  + Added Total Row
* InputFormData
  + Fixed Insert function by accounting for Selecting ‘Date’
  + Total row successfully calculates the total of the corresponding column
    - One problem overcame was that an old array name was used as a key reference (but was not successfully retrieved because it had spaces in it)
  + Current Problem: Cannot access reference to label values in php. If I change them to inputs, existing javascript code will turn the inputs into “disabled”, which will not be eligible to be put into database.
    - Next step is to have javascript code only do the actual input values, and ignore the Time values
    - Fixed step above
  + Created function “insertTotal” that extracts database values using Select query and adds up all previous values associated with column in order to add or update “Time” row in database. This is to assume that insertTotal() gets executed AFTER insert() as the new inputted information must be accounted for in the sum of the “Time”

Database:

Added primary key constraints to date, zone, and time

Brainstorm for new features:

Index.php will use php date as a reference to extract from Database

Total Row from InputForm.php will automatically calculate the total for each column and add to database if any of the corresponding elements in the column is added, or updated if element in column was changed.

In Index.php, total number of people will read from the Total row, and it will look at current time, and round down to the nearest time column that it corresponds to for that date.

April 11:

Files Changed:

* Index.php
  + Label will show total number of people

Files Added:

* IndexTotal.php
  + Created method called retrieveLatestTime() that will traverse down a time array and find the latest time where there exists an “Time” row in the database and show it.
  + Within retrieveLatestTime(), it will find the earliest index of the time array (aka the latest time) and the corresponding number of people to that time. This information is retrieved from the database.
  + Accounted for new days : once a new day starts, the date will change at midnight, which, using my old code, will automatically generate a null because it will only look at values for that date and the first input will not be put until 6:30AM.
    - This was solved by accounting for displaying the latest time from the previous day.

April 12th:

Database change:

* Added new column “InsertTime” where it will keep track of when data is inserted
* NOW() and DATE\_ADD / DATE\_SUB will be used in queries to extract current time. Because timezone for database is different from ours, I used DATE\_SUB (minus 4 hours) to get our correct time.

Files Changed:

* Index.php
  + Prints out the updated time for the latest time column
* IndexTotal.php
  + Function returns an array that consists of the Time column and its corresponding total value and the time it was inserted
* InputFormData.php
  + Inserting will now have another element, “InsertTime” which will insert the time that it was inserted
  + Reformatted the date format of mysql

**Things to implement:**

Primary Key **DONE**

SSL

Better Username and Password Authentication

Edit Button **DONE**

Maybe Delete Button for each element

Floor Plan **DONE**

Retrieving values for previous dates **DONE**

Need to edit php code within the value= for input so that if the value is not “submitted”, it will not be displayed right after the page reloads.

Link to another page where you can see thumbnails of all the tables

Exporting table to pdf / excel

Button that automatically emails information to inputted email.

CSS routing

Visitor Tracker

Another row in table that shows total for each time. **DONE**

Add time of insertion into database (and update) **DONE**

Possible Scenarios and how to address them (April 1):

1. *Username/Password gets exposed to other users (not staff members)*
   1. **Solution**: Have a page where it will send a session with a link to manager’s email -> If manager decides to click on link, then within a time frame, he/she must provide new username or password. At this instant, it will check if the session is the same and if it still exists, and if so, username and password will be changed. It is then the manager’s responsibility to notify Crc Staff members the new Username and Password.
2. *What if all data from database gets deleted?* 
   1. **Solution**: Application will have an export function so that the manager can still retrieve PDF / Excel versions of the data that is being inputted/displayed. It should then be the manager’s responsibility to extract PDF before database malfunctions
3. *What if random data was inserted into other tables?* 
   1. **Solution**: Currently, you can delete an entire table simply through the GUI provided in the Input Form page. I will soon implement a function where you can delete each element of the table.
4. *What if a Crc Staff member decides to use the “delete” function and remove important data?*
   1. **Solution**: I can restrict the use of the delete button to only the manager. If this were the case, and if a Crc Staff member accidently inputs wrong data, he/she can still use the edit button to change the mistake
5. *Currently, the display on the front page replicates the exact lay-out of the gym, including specific equipment; what if these equipment move?*
   1. **Solution:**
      1. I used an online platform to design the floor plan, I could provide the account information to allow for manual changes, but this would then require a lot of time from the manager, as well as the manager’s responsibility to implement new changes into the system
      2. Ignore the equipment placement changes, as long as the zones for designated areas do not change. The point of the equipment is to allow for users to understand how the layout is positioned and how it resembles the floor plan of the gym
6. *What if I want to change the designated areas of the gym?*
   1. **Solution:** Currently, I do not have a solution for this problem, as it will not only require changes in the display (image) and jquery code, but also the structure of the table that is used to input data. If this is a **serious** issue, I can come up with alternatives.

Personal Questions:

1. If this application gets fully implemented, do I still have the right to alter code/make any changes?
2. Does the Crc have the right to claim this application as theirs? Do I still have the right to claim it as mine?
3. Application currently requires host plan + domain name / I can currently pay for these so long as I am working on the application, but will the Crc manage this after I leave? If so, and if the Crc does pay for hosting + domain name, will I, by then, still have the right to claim the application as mine?