CIT416 Assignments – Spring 2021

Web Technology Presentation (100 points)  
There are literally hundreds of topics related to web development and various technologies used in desktop browser and mobile development that are not covered in IST/CIT263, CIT313, IST430/CIT410, or IST436/CIT416. It is nearly impossible to cover in depth every single topic. However, other topics than what have been covered may be of importance to you depending on the job you apply for, and ultimately, acquire. Choose a topic of interest to you that has not been covered in one of these three classes, research it, and prepare a 10 minute presentation to introduce the class to your topic. By research, I do not mean go to a single web site and copy information. In your presentation, include information on what the technology/topic is, how it could be integrated or implemented into web sites or mobile applications, what benefit(s) it provides, etc. You will need a minimum of three sources for your presentation. This project/presentation will be in place of a regular final exam.

Now that you know the minimum requirements, here is the kicker. You have the opportunity to score up to 100% on this presentation, or up to 80% on this presentation. If you deliver your presentation in person during the final exam slot, the HIGHEST you can score is an 80% out of 100%. If you choose to do a “teach it back” video in place of your presentation (same concept as a presentation, but recorded for others to watch), you can score up to 100% for your grade.

## Docker Setup w/Web Server and DBMS installation (125 points)

For this project, you will gain experience in installing and creating Docker container images (<https://www.docker.com/resources/what-container>) that contain all of the apps necessary to create your semester-based project using React. This means you will be responsible for setting up a shareable Docker image with HTTP, PHP, mySQL, and React support. A key: read up on Docker, what a Container is, and search the Hub for predeveloped container images that can be added to your project (and customized): <https://www.docker.com/products/docker-hub>. In the end, this actually should be the system you use to complete all of your project work this semester.

This will not be a complete exercise in web site and/or DBMS setup and security, but will provide you with a feel for the basics. Specifically, you will install and configure a base install of Docker. You cannot mess up your actual machine that you install Docker on as the containers will operate independently of your operating system, much the same way that a Virtual Machine works. You will then add a web server and DBMS to your installation (see above for specifics). You will have to research how to install and configure each container to operate together (HINT: YouTube, tutorials). You may be required to install several dependencies as well, but do your research.

Tip #1: Use Visual Studio Code and Add Docker as an extension

Tip #2: Use DockerFiles. Best practices for writing Dockerfiles: https://docs.docker.com/develop/develop-images/dockerfile\_best-practices/

Tip #3: Be sure your data is “persistent” after set up

Tip #4: Popular Quick Commands

docker ps -a (list of all containers and state)

docker stop [container] OR docker stop $(docker ps -a)

You will also have to install a PHP 7.4.x (<https://www.php.net/downloads.php>) acontainer image and can be a bit tricky (that’s a hint to not wait until the last minute to start on this project).

You will submit your Dockerfile, any files you customized, and a zip of your local storage folder, and a detailed report that includes each of the steps that you took for the successful installation and implementation of each piece of your server, including screenshots (be detailed). To test your installation of PHP, you will create a PHP page with the EXACT following content (which is also a hint that you will have to change some settings from the ‘out of the box’ installation) and place it in the root folder of your web site, naming the file index.php:

<?  
 phpinfo();  
?>

To test your installation of your database, create a PHP page within your web server’s root folder with the following content (exactly as shown, no changes allowed), access the PHP page, then show me the results with screenshots. Your screen shots should include the MySQL database itself after executing the page and the browser’s output of the page’s execution. Think…. There may be a management tool that exists that will also need to be installed to show and manipulate the database outside of simply issuing PHP commands. As a matter of fact, you will need to have a database established (no tables) prior to accessing the PHP page. All of the information you need in order to create the database with the username and password necessary to have this work are all within the code below (hint hint). Questions? Ask

<?

$mysqli = new mysqli("localhost", "my\_user", "my\_password", "userdb");

/\* check connection \*/

if (mysqli\_connect\_errno())

{

echo "Connect failed: " . mysqli\_connect\_error() . "<br />";

exit();

}

/\* Create table and Insert rows \*/

if (!$mysqli->query("CREATE TABLE friends (id int, name varchar(20),   
 primary key(id))"))

$mysqli->error;

if (!$mysqli->query("INSERT INTO friends VALUES (1,'Jen Lynn'), (2,

'Brian'),(3, 'Michael'), (4, 'Katherine'), (5, 'Alex'), (6, 'Drew'),

(7, 'Olivia')"))

$mysqli->error;

/\* select all rows \*/

$result = $mysqli->query("SELECT \* FROM friends");

echo "Affected rows by (SELECT): " . $mysqli->affected\_rows . "<br />";

while($row = $result->fetch\_assoc())

echo stripslashes($row['name']) . "<br />";

/\* close connection \*/

$mysqli->close();

?>

## GITting Started (75 points)

Welcome to the real world of development for this one. You will be using a GIT repo for your code this semester, as many shared development teams do. For this project, you are to:

* Setup a GitHub education account – free private repos and more (<https://education.github.com/pack>)
* Setup a private repo. When done, let someone in the class know so that they can clone your repo, make changes, and submit a new branch and pull request (can be a single page of code). You only have to share and work with one other student. If there is an odd # of students in the course, I’ll be the even man in. My git username is morgan16.
* You will then have to accept the partner’s pull request
* Then, clone the partner’s repo, make a change, and submit a pull request
* After the initial back and forth above is complete, you will setup a private repo for your ReactJS project to reside in.
* You get to decide what GIT client to use or use the command line git tools under Ubuntu.

When complete, provide a document detailing the steps you took to complete this project and send me an invite (morgan16) to your repo so that I can grade it. Within Blackboard, upload your document and the URL for your repo.

## Small”er” React Application (/125)

This will be your first application developed using React. For this project, you are to use a theme (use a free one) from <https://material-ui.com/store/?utm_source=docs&utm_medium=referral&utm_campaign=home-store> and develop a small dashboard. This will give you some experience creating a single page application with React as well as using Axios for fetching data. This dashboard when loaded will contain the following information:

* Current weather information (use openweathermap APIs to display temp, forecast and city information (Huntington))
* Current stock information (use <https://www.alphavantage.co/> and include Apple, Microsoft, and Amazon stock in dashboard boxes)
* Display at least 5 comics by title in a dropdown that once a user selects a comic, its image is displayed within the dashboard (<https://xkcd.com/json.html>). The dropdown MUST be populated using the API.

## Intro to Semester Project (575 total points including template, DB creation and coding)

Just what is that project you ask? This spring, you will be building an online course scheduler application for students to select sections of courses for registration, much like the Marshall University Course Scheduler application for students (CSG). We will call our project CSG+. Specifically, you will create a course selection database system that will containing the following functionality:

* User account creation/login
  + A student can create an account on the first access, or login to return to their account
    - Should be able to review courses/sections without logging in
  + Account data stored should include: Name, email, password
  + HINT: Database design needs to account for the system storing data for multiple users
  + MUST create an admin role and provide a means for that admin to be able to login.
* Ability to Add Classes and Sections
  + Each class entered into the database needs a course alpha designator (CIT), course number (416), and title (Advanced Web Programming).
  + Each section of a class entered into the database will need to be associated (related) to a class and have: days offered, times offered, and instructor name. You can obviously have more than one section of a given course. ALSO, we are going to assume this is for a single semester (to make things a bit easier).
  + Upload syllabi for a course (relate to the course, not a section). To test this later, you can use a single page, small PDF.
  + Limit add/delete/update access to courses/sections to the admin
* Display of Courses/Sections offered
  + All courses being offered will need to be displayed on the homepage (can have a simple list of courses with alpha, number and title.
  + Have an individual viewing page for each course that displays each section being offered and a link to open the syllabus.
* Favorite a Section
  + User MUST be logged in to add a particular section to their schedule for the semester.
  + Must keep track of sections favorited between browser sessions (HINT: DB)
* Schedule Review Page
  + Student will be able to review their favorited sections
  + Students can remove a section from this page.

If you do not yet have a firm working knowledge of JSON, you will need to brush up on those skills this semester. The “server” side of your project can be any language you choose, but since each of you should be good with PHP, I would HIGHLY recommend going with it. Here is a good PHP->JSON tutorial that contains multiple PHP examples converted to JSON: <https://www.w3schools.com/js/js_json_php.asp>

## React Project: Database (75 points)

For this project, you will create the database (mySQL or FireBase – but if you choose FireBase, you are on your own) that will be required to support the functionality of CSG+ as described above. Be sure to create all of the necessary tables and attributes and relate the tables as necessary. (Hint: many to many relationships are NOT acceptable). As a submission, please submit a FULL ERD in your ERD-generating application of choice (MySQL Workbench, Visio, LucidChart, or even https://sqldbm.com/)

## React Project: Setup/Bootstrap Theme (75 points)

Now that you have a web server up and running within Docker which supports PHP, MySQL, and React, you can and add a template (theme) for your larger semester project. For this deliverable, you are to:

* Ensure React is installed and working (see textbook/provided notes)
* Choose a reactJS template appropriate for viewing our items from the schedule database. Be sure to search for a template that you will be able to use for the entire project (think section selection, viewing, editing)
* Create a repo on your GitHub account for code management
* Don’t forget to merge everything to your Master Branch before building for distribution
  + Dist file will go on your web server
* When complete, provide a document detailing the steps you took to complete this project and send me an invite (morgan16) to your repo so that I can grade it. Within Blackboard, upload your document and the URL for your repo.

## React Project: User Account Creation/Login (100 points)

For this deliverable, you will need to integrate a login/account creation interface into your theme. As a reminder from the project description, this project will have the following requirements:

* Courses/Sections can be viewed by anyone, and edited/deleted by admin only
* Use a login template to make your life easier (but hey, if you don’t like to make things easy, have at it). You can discuss your template’s login with me if it has one to see if it’s okay.
* TEST THOROUGHLY that your register and login is WORKING before turning it in.
  + Once a person registers, it should log them in automatically.
* Provide error handling messages for incorrect username and password.
* HINT: Use state. Database design needs to account for the system storing data for multiple users

Once logged in, the application should show the user’s name in the upper nav bar of the browser. Be sure to use form validation on all login or account creation table fields. When finished, merge your new code into a new branch on your git repository and submit the repository’s URL to Blackboard when complete.

## React Project: Adding Classes and Sections (125 points)

For this deliverable, you will create a form where the admin will be able to add courses, sections, and syllabi, and a view where non-admins can see all courses in a format that is readable and makes courses distinct from one another in a catchy format (think cards, tiles, etc.). As a reminder from the project description, this project will have the following requirements:

* Each course entered into the database needs a course alpha designator, course number, and title.
  + Have the ability to associate a syllabus with a course (not all courses have a syllabus)
* Each section associated with a course needs days, times, and instructor
* Display courses individually on home page (nice design/spacing)
* Have an individual page for each course where you can review all of the sections associated with that course, see the syllabus, etc.
  + No favoriting yet, just listing and reviewing

Be sure that for all data entry you are using form validation. Send your form submissions (new entries) via Axios ajax APIs to be stored in the database and the images in a folder on the server. When finished, wash, rinse, and repeat…. upload your new code to your git repository and submit the address to Blackboard when ready.

## React Project: Favoriting Sections/Schedule Review Page (125 points)

For this deliverable, the user is to create a view that supports the student’s ability to favorite sections of courses. As a reminder from the project description, this project will have the following requirements:

* + User MUST be logged in to select their sections
  + A user should only be able to favorite a section of a course
  + Create a page on which the student can review all of their favorited sections

Same rules apply: Axios ajax API requests. Wash, rinse, repeat your submission process.

## Admin Viewing Schedules (75 points)

For this deliverable, the admin will be able to view all students and their favorited sections.

* Display a listing of all students in a dropdown
  + Only an admin should be able to access this page (page content should at least be hidden from other users as well)
* Selecting a user will then load their schedule for review
* Admin can delete a section from a user’s schedule
* Both viewing the schedule and deleting a section should be Axios ajax API based
  + The pages should NOT reload, but appropriate messages should be displayed

When finished, submit your last project the same as the others.

Questions? Please ask early and ask often.

“Choosing to not use an MVC framework means that we will have to keep the Model and the View in sync manually, which looks simple at first sight but very quickly we will end up with an unmaintainable program.”