JSON

To get an overview of the JSON data format, see here (<https://towardsdatascience.com/an-introduction-to-json-c9acb464f43e>. ). It will give you background information and the reasoning we are using JSON over other formats.

Json manipulation is very simple in PHP. Use the documentation here (<https://www.php.net/manual/en/function.json-decode>. ) and here (<https://www.php.net/manual/en/function.json-encode>. ) to get started.

I have created a full example on the University Database, so you can have an idea of how to create a dataset in JSON.

full\_example\_datagen.php

<?php

/\*

This example will generate data for two database tables:

student and deparment. It will store all data in a single

json file which will be then read into the database.

\*/

// require Faker code

require\_once 'composer/vendor/autoload.php';

// use the factory to create a Faker\Generator instance

$faker = Faker\Factory::create();

/\* We'll generate 50 random words as department names,

25 random words as buildings, and

Budgets ranging from 50,000 to 200,000 in whole tens of thousands \*/

// generate 25 random words for department buildings

$buildings = $faker->words(25);

for ($i = 0; $i < 50; $i++){

//generate a random word for a department name

$department[$i]["dept\_name"] = $faker->word();

//assign each building to two departments

$department[$i]["building"] = $buildings[$i%25];

$department[$i]["budget"] = $faker->numberBetween("5","20")."0,000";

} // we now have 50 random departments in the $department array

// declare a student array for clarity

$student = array();

// generate 2,000 students

for ($i = 0; $i < 2000; $i++){

//generate a sequential ID starting at 1

//(this might be skipped if we have auto\_increment)

$student[$i]["ID"] = $i+1;

//assign a random name to each student

$student[$i]["name"] = $faker->firstName()." ".$faker->lastName();

//assign a random existing department to each student

$student[$i]["dept\_name"] = $department[rand(0,49)]["dept\_name"];

$student[$i]["tot\_cred"] = $faker->numberBetween(0,150);

} // we now have 2000 random students in the $student array

// now, we add everything to the $alldata array

// careful to add data in order to ensure referential

// contraints integrity

$allData = array(

"department" => $department,

"student" => $student

);

// create new json file (it will overrite if it already exists)

$fp = fopen("datadump.json","w");

if(!$fp)

die("Couldn't open file");

//write the json encoded data to the file

fwrite($fp, json\_encode($allData));

//close the file

fclose($fp);

?>

I didn't have time to create the reading from the JSON file, but it is very straightforward. Stack overflow has an example (<https://stackoverflow.com/questions/19763757/insert-json-decoded-data-into-mysql-with-pdo>. ) that contains all that is needed to decode and insert into the database.

Copying files to the server

As explained in class, you should copy your files to your public\_html folder using a secure copying tool. That is scp on MacOS/Linux and pscp.exe on Windows. Here are guides on how to use each of these tools:

Location of your public\_html folder: /home/username/public\_html, where username is your username (that you used to connect through PuTTY or ssh).

Windows: use pscp.exe. Guide here (<https://www.poftut.com/how-to-use-pscp-command-on-windows/>. ). Note: you may use other tools, such as sftp, but you'll have to dig that out on your own; pscp works just as well.

MacOS/Linux: use scp. Guide here (<https://linuxize.com/post/how-to-use-scp-command-to-securely-transfer-files/#copy-a-local-file-to-a-remote-system-with-the-scp-command>. ). Note: again, you may use other tools, such as sftp, but you'll have to dig that out on your own; scp works just as well.

Example:

(p)scp FILE username@uncg-cs-dblab.eastus.cloudapp.azure.com:/home/username/public\_html

The above example copies FILE to the respective user's public\_html, and can be accessed on a web browser through:

https://uncg-cs-dblab.eastus.cloudapp.azure.com/~username/FILE