## **Quiz 2**

Same caveats as quiz 1: read everything really carefully, then read everything really carefully again, then read everything really carefully a third time, then start answering questions.

When you are finished, you are free to leave. Quiz is open notes, open Internet. Only things you can’t do are talk to each other and post the questions on StackExchange and the like.

#### **Part 1**

* **1.1**. **Explain three possible features** of a web application that require (or, at least, made easier by) a server-side component written in a language such as PHP. Don't just mention the feature, explain what it involves.

Three possible features of a web application that require a server-side component are databases, authentication, and dynamic pages.

Server-side components are usually written in PHP or Python and allow for faster load times. The way they work is that dynamic HTML pages are generated by running a script directly on the web server after a request is made by the user. PHP is able to connect to the database using the MySQLi extension and PDO (php data objects). If a website were to be storing information in a database, they would use a server-side component in their application. A lot of applications require authentication, which occurs through the serverside component. Dynamic pages that are written in HTML also require a server-side component because the script is run directly on the web server after a request has been made, after which HTML code is sent directly to the client browser in the form of a web page.

* **1.2**. **Explain two actions** that can be taken to **secure** a web application. These may be related to user-authentication & authorization, server configuration, codebase, and/or network infrastructure.

Two actions that can be taken to secure a web application are implementing SSL/TLS encryption and creating a maximum script execution time to define how long a particular script can run on the server. The SSL/TLS encryption encrypts all the data that occurs in transit through the secure HTTPS protocol. Utilizing SSL can seriously be used well to secure a web application. A maximum script execution time simply ensures that the time allotted to run a script on the server. Shortening this time would narrow the time possibilities for attackers to attack the main page.

#### 

#### **Part 2**

Explain each code segment in two different ways: first, explain the overall picture without using any technical jargon, as if you were explaining the code to someone who doesn’t understand any programming, and; second, explain in as exacting detail as possible, line by line, what the code is doing. If there are any mistakes or errors in the code, fix them inline using a different color.

**2.1**.  
 if (isset($\_GET['lname'])) {

// Check connection

if ($conn->connect\_error) {

die("Connection failed: " . $conn->connect\_error);

}

if ($\_GET['lname'] != '') {

$pstmt = $conn->prepare('SELECT \* from customers WHERE lname = :ln');

$pstmt->bindParam('ln', $\_GET['lname'], PDO::PARAM\_STR);

} else {

echo "lname not given, outputting entire file";

$pstmt = $conn->prepare('SELECT \* from customers');

}

$pstmt->execute();

while ($row = $pstmt->fetch()) {

printf("%s %s",$row['fname'],$row['lname']);

}  
 }

1. Essentially what this code is doing is that it is extracting the lastnames from the database and printing firstname and lastnames in a row.
2. The isset statement checks to ensure that the value of $\_GET[‘lname’] is valid. If it is not equal to nothing, then the code selects the lastnames from the database. The $conn connects the file to the database. The SQL command SELECT \* selects the column for the lastname where appropriate. The next line then binds variables to a prepared statement as parameters. The next line is a else statement that only occurs if the lanme is not given in which case it will select all the contents from the customers database. The code will execute and during this the print block will print both the first name and lastname of everyone with a lastname in the database.

**2.2**.  
 $('#trigger').click(function(e) {

$.getJSON('people.json', function(data) {  
 $.each(data, function(key, val) {  
 alert(val.name + ", " + val.profession);

});  
 });

1. Here this block of code takes an input of data and for each person within the data creates an alert box containing the person’s name separated by profession.
2. First, this is a trigger function that is set to complete an action. From the data, people are pulled with the information of defining that into a key, val pair. The people’s contents are then displayed through an alert box.

#### **Part 3**

**3.1**. In MySQL, create a database named websys\_quiz with the following 2 tables and link the 2 tables using a foreign key constraint:  
items

-id int(11) primary key

-name varchar(255)

-price decimal(6,2)

discounts

-id int primary key

-item\_id int(11)

-discount decimal(3,2)

Run the following code in the SQL tab to insert test data. As in Part 2, if this code is wrong, fix it before running it and document your fix(es) in your README.md file.  
INSERT INTO `items` (`id`, `name`, `price`) VALUES (1, 'MacBook Pro', '2499'),

(2, 'OpenBSD Tshirt, '20.0'),(3, 'Amazon echo', '99.99'),(4, 'Nvidia GTX 3080', '1999.99'),(5, 'AMD Ryzen 9 3900X’, '549.99');  
INSERT INTO `discounts` (`id`, `item\_id`, `discount`) VALUES (1, 1, 0.25), (2, 2, 0.5),(3, 3, 0.75),(4, 5, 0.1);

**3.2**. Create a PHP page with 3 buttons that connects to the previously created database and performs each of the following queries based on the button pressed:

* + 1) List **all** items ordered by price from lowest to highest, **before applying discount**.
  + 2) List **all** items ordered by price from lowest to highest, **after applying discount**.
  + 3) Find the average price of **only** the items that have a discount, **after applying discount**.

Creativity counts for this! Don’t just stop once this works. Showcase all your talents in HTML, CSS, Javascript, PHP, and MySQL.

**README.md** Don't forget a readme! Briefly explain your solution and any issues you faced. Don’t forget to include any MySQL fixes you needed at the start of Part 3.

#### 

#### **Submission**

* Create a **new branch** named **quiz2** in your personal repo
* Put all quiz materials in that branch
* **DO NOT MERGE INTO MAIN**
* **-5 points for each submission step not followed**

#### **Rubric**

* **Part 1** 20 Points
* **Part 2** 20 Points
* **Part 3:**
* **Database** 15 Points
* **PHP+queries** 25 Points
* **Creativity** 10 Points
* **readme** 10 Points

**Extra Credit (+5 points)**

Name the chat protocol developed at RPI in the 1990s.

Telnet