## **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.

1. We could utilize a database, which sometimes needs to exchange data through multiple different applications dynamically.

2. PHP can accept data from forms on the website server, and it allows the server to receive information from the client.

3. PHP can provide implemented authentication, to store personal information which can be saved and used in different devices.

* **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.

#### **1. Prevent injections from unwanted input that could be used as SQL code.**

#### **2. Check to see if there are no sensitive data, such as database passwords stored in repositories.**

#### **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'])) {

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']);

}  
 }

The first line checks to see if a last name GET variable was set after a user has submitted an HTML form with method “=get”.

The second line checks to see if the variable was empty.

If it was not empty, it prepares the sql insertion statement. The insertion basically means select only customers from the database that have a last name provided. Then, the prepared variable $pstmt will bind a parameter, the whatever the value from $\_GET[‘lname’] was, to the ‘ln’ column. If the string was empty, then it will print out “lname not given, outputting entire file”, and it will prepare a sql insertion statement that will select all customers from the database. The statement is executed, the results are all fetched, and the results are printed out, from first to last name.

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

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

});

});  
 });

The code first adds an event listener click function to every element that has id = ‘trigger’. When it is clicked, it runs an AJAX request get json data to fetch a file called ‘people.json’. The code iterates through the the ‘people’ object of the json data, calling an alert which pops out a popup box that contains the iterated property index’s name and profession.

#### **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.