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.

One feature of a web application that uses PHP is enabling the access of data in a database, which one can change as need be. This involves the creation of MySQL tables and the selecting, inserting, adding, and deleting of data in these tables. Another feature would be that PHP can be used to send and receive both files and cookies between servers. Also, PHP greatly aids in the use of sign-up and login features for web applications, as it helps mitigate roles and features users have access to.

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.

One action we can take to secure a web application is to use password salting to protect passwords stored in a database. Password salting involves adding a string of between 32 or more characters to a password and then performing a hash function. Password salting is one of the most secure ways to protect passwords without the chance of them being leaked/accessed illegally. Another action is to use permission-based authorization to manage who has what access when it comes to back-end access. By doing this, we can give select people, mainly the admins but it could be other individuals, more control of the web application. By simply including some form of check-in in the SQL table upon a user logging in, it increases the security dramatically.

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'])) {
          $conn = new mysqli($servername, $username, $password,
$dbname);
          if ($_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']);
         }
      }
```

==========

General:

If a last name is inputted into the system, it is used to search through a list of names. All individuals' names with the same last name that are stored in the system are printed out. If the last name cannot be found, then all names are printed out.

Technical:

The first two if statements check to see if an element with the tag 'lname' is first, set to a value (not blank), and second, checking to make sure the value is not null. If we do not get past the first if statement, then the code will not run. We create a SQL connection upon passing it. With the second if, if 'lname' is not null, we search the MySQL table for all of the entries that feature 'lname' in the 'ln' column. We then check to make sure all of the entries we return match our parameters. If 'lname' is null, then we select all of the entries in the table. We then loop through "$pstmt", and output all of the rows that match the previous elements. However, the results are printed out as names, first name and last name.

==========

2.2.

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

==========

General:

Upon the user clicking a button, the system will print out a list of people. Each line of the list will feature a name and the said person's profession.

Technical:
When a button with the id tag "trigger" is clicked, a JS function is run to retrieve the JSON information stored in the "people.json" file. The .each() method is then used to loop through the JSON file, retrieve the values stored, and then print out the values that correspond to "name:" and "profession:".
==========

Extra Credit:

PlayNet? (HipChat was from the 2000s)