Web Development

1. php database connection

- MySQL procedural connection
    1. Connect to the database.
    2. Handle connection errors.
    3. Prepare a query string
    4. Execute the SQL query.
    5. Process the results.
    6. Free resources and close connection.

```
// Create connection Using mysqli Extension
$conn = mysqli_connect($host, $user, $password, $database);
```

2. url parameters

- also known as query string parameters, are a way to pass data between web pages by appending data to the end of a URL.
- URL parameters are represented by key-value pairs, separated by an equals sign (=) and separated from each other by an ampersand (&). For example, in the URL http://example.com/page.php?name=John&age=30, name=John and age=30 are the URL parameters.

Using PHP code:

```
<a href = “<?php echo “show.php?id= “ .$result [‘id’]; ?>”   show page </a>
```

3. get vs post

- GET: The GET method sends data in the URL as a query string. This means that the data is visible in the URL bar of the browser. GET requests are typically used for fetching data FROM the server.
    - http://example.com/page.php?name=John&age=30
- POST: The POST method sends data in the HTTP request body, which is not visible in the URL bar of the browser. POST requests are typically used for submitting data TO the server.

4. superglobal variable

- $_GET: an array containing the values of all the GET parameters passed to the script.
- $_POST: an array containing the values of all the POST parameters passed to the script.
- $_FILES: an array containing information about the uploaded files, such as the name, type, and temporary location.
- $_COOKIE: an array containing the values of all the cookies sent to the script.
- $_SESSION: This variable is an array containing the values of all the session variables.

5. session and cookie compare

compare	cookies	sessions
1	Stored client-side	Store user data on the server (main advantage than cookie)
2	Limited in size by browser	Limited only by server space
3	Can be disabled	Cannot be modified by users Expire when the browser is closed
4	use SSL to ensure data security	identified by a session id: often a small COOKIE;

```
//Before you can use session variables, you need to create a PHP session with:
session_start();

//set a session to an empty array
$_SESSION = array();

//destroy all the data associated with the current session.
session_destroy():

//create a session variable
$_SESSION['myvar'] = 5;?
```

6. variable name in js and php

variable	JavaScript	PHP
case senstive	yes	PHP variable case-sensitive, but function names, class names, keywords not
Variable Declaration	var, let, or constv	$
dynamically typed	yes	yes, but PHP is more strict
assignment	JS: let myVar = 10	PHP: $myVar = 10;
constant	const MY_CONST = 'Hello'	define('MY_CONST', 'Hello')
boolean	var myvar = true	$myvar = true
data type		

7. What does move_uploaded_file() return?

- move_uploaded_file() function is used in PHP to move uploaded files to a new location.

- It returns a boolean value indicating whether the operation was successful. If the file is successfully moved, the function returns true, otherwise it returns false.

```
//  function moves an uploaded file to a new destination
move_uploaded_file(file, dest)

//This function returns true on success and false on failure.
if (isset($_FILES['uploaded_file']) && $_FILES['uploaded_file']['error'] ==
UPLOAD_ERR_OK) {
        $upload_folder = '/uploads/';
        $file_name = basename($_FILES['uploaded_file']['name']);
        $destination = $upload_folder . $file_name;
     if (move_uploaded_file($_FILES['uploaded_file']['tmp_name'], $destination)) {
                echo "File uploaded successfully.";
          } else { echo "An error occurred while uploading the file."; }
  } else {  echo "No file was uploaded or an error occurred during upload.";}
```

8. loosely type vs dynamic type

- loosely typed languages allow variables to hold values of any type and the type can be changed at any time,
- dynamically typed languages determine the type of a variable at runtime based on the value assigned to it.
- a language can be bot loosely and dynamically typed, like js and php
- In some cases, loosely typed and dynamically typed are used interchangeably, but they are not the same thing.

9. is double a data type in js?

- JavaScript represents all numbers, including integers and floating-point numbers, as 64-bit floating-point values

```
let x = 5; // integer
let y = 3.14; // floating-point number
let z = x + y; // result is a floating-point number
console.log(z); // outputs 8.14
```

- Note that in JavaScript, you don't need to specify the type of a variable when you declare it. JavaScript is a dynamically typed language, which means that the type of a variable is determined at runtime based on the value assigned to it.

10. php comparsion operator in the code can be trick, loops

- php for loop:

```
for ($i = 0; $i < 10; $i++) {
  echo $i . "<br>";
}
```

- php while loop:

```
$i = 0;
while ($i < 10) {
    echo $i . "<br>";
    $i++;
}
```

- php foreach loop:

```
$colors = array("red", "green", "blue");
foreach ($colors as $color) {
  echo $color . "<br>";
}
```

11. ip6 vs ip4

- ip6: 128 bits, 16 bits * 8 (2001:0db8:85a3:0000:0000:8a2e:0370:7334.)
- ip4: 32 bits, 8bits * 4 (192.0.2.1.),

12. Which range of http response status codes indicates successful responses?

- 200-299

13. HTML textarea

```
<label for="message">Message:</label>
<textarea id="message" name="message" rows="4" cols="40"> Enter your message
here...</textarea>
```

14. form element collect data

```
<form action="submit.php" method="post">
  <label for="name">Name:</label>
  <input type="text" id="name" name="name"><br>

  <label for="email">Email:</label>
  <input type="email" id="email" name="email"><br>
```

```
  <label for="message">Message:</label>
  <textarea id="message" name="message" rows="4" cols="40"></textarea><br>

  <input type="submit" value="Submit">
</form>
```

15. how to include js in html and php

include js in html

```
<html>
  <head>
    <script src="myscript.js"></script>
  </head>
</html>
```

passing data from php to js using `<?php ?>` tag

```
<script>
  var x = “<?php echo “$hi” ?>
  document.write(x);
</script>;
```

Including a `<script>` tag using the `echo` statement

```
<?php
    echo '<script>' ;
  echo 'console.log("Hello from JavaScript!"); ';
  echo '</script>' ;
?>

<?php
  echo "<script>alert('Hello, world!');</script>";
?>
```

16. which attribute is mandatory to for img and which is optional

- src mandatory and alt, width, height, style is optional.

17. stylesheet which one will apply?

- the priority of a style rule is determined by the specificity of the selector used to apply the rule.
- specificity: Inline styles > ID selectors > class selectors > element selectors.
- Universal selectors (*), combinators (+, >, ~), and pseudo-classes (:first-child, :hover, etc.) have a lower specificity than ID selectors, class selectors, or element selectors.

- If two or more selectors have the same specificity, the one that appears last in the stylesheet takes precedence.
- you should use specific selectors to target the elements you want to style.
- Inline >Internal >External Style Sheet >Browser's Default Behavior
- !important > inline >id# >class . >tag

18. rowspan--table?

```
<table>
  <tr>
    <th colspan="2">Name</th>
    <th>Age</th>
  </tr>
  <tr>
    <td>Jill</td>
    <td>Smith</td>
    <td rowspan="2">30</td>
  </tr>
  <tr>
    <td>Eve</td>
    <td>Jackson</td>
  </tr>
</table>
```

Name		Age
Jill	Smith	30
Eve	Jackson	

19. inline vs block

- block elements are those that take up the full width of their parent container, and create a new line after themselves. Examples of block-level elements include `div, p, h1`
- inline elements are those that only take up as much width as their content requires, and do not create a new line after themselves. Examples of inline-level elements include `<a> <span> <em> <strong> <img>`

20. Elements that do not require a closing tag include:

- img, br, input are self-closing.
- meta, link and other elements that do not define content.

21. DOM event

```
<button id="myButton">Click me</button>

<script>
  var button = document.getElementById("myButton");
```

```
  button.addEventListener("click", function(event) {
    console.log("Button clicked");
  });
  button.addEventListener("mouseover", function(event) {
    console.log("Mouse over button");
  });
</script>
```

```
document.querySelector('input[name="email"]').myEventHandler();
```

```
<input type="text" name="email" onfocus="this.style.backgroundColor='yellow'" />
```

22. most common JS events

- click - This event is triggered when an element is clicked.
- mouseover - This event is triggered when the mouse pointer is moved over an element.
- keydown - This event is triggered when a key is pressed down.
- load - This event is triggered when a web page has finished loading.
- submit - This event is triggered when a form is submitted.
- change - This event is triggered when the value of an element changes, such as when a checkbox is checked or unchecked.
- keyup is an event that is triggered when the user releases a key on the keyboard. For example, when the user types something into a text field or other input element.

Mouse Events	Keyboard Events	Form Events	Document/Window
click / dblclick, mouseenter, mouseleave	keypress, keyup / keydown	submit / change, focus / blur	load / resize, scroll / unload

23. JSON conversion

```
//json
{
  "name": "John",
  "age": 30,
  "email": "john@example.com"
}
```

```
// js object
var person = {
  name: "John",
  age: 30,
  email: "john@example.com", //trailling comma is allowed
```

```
};
```

- Key-Value pair just like object in JS
- JSON compare with JS object
    - The keys must be surrounded by double quotes (")
    - Strings must be surrounded by double quotes (")
    - Trailing commas are not allowed
- JSON to PHP
    - php function json_decode()
- js to json
    - JSON.stringify()

```
const myObj = {
  name: "John",
  age: 30,
  city: "New York"
};
const jsonString = JSON.stringify(myObj);
console.log(jsonString);
// {"name":"John","age":30,"city":"New York"}
```

- JSON to JS
    - JSON.parse()

```
const jsonString = '{"name":"John","age":30,"city":"New York"}';
const myObj = JSON.parse(jsonString);
console.log(myObj);
// { name: 'John', age: 30, city: 'New York' }
```

24. AJAX

- AJAX is a combination of several web technologies, including HTML, CSS, JavaScript, and XML (or JSON).
- AJAX allows web pages to update data dynamically without requiring a full page refresh.
- AJAX requests are typically initiated by JavaScript code running in the browser.
- AJAX requests are sent to the server using the XMLHttpRequest (XHR) object in JavaScript.
- AJAX can be used to fetch data from a server (e.g. a database), send data to a server (e.g. form data), or update a web page dynamically.
- AJAX responses can be in a variety of formats, including HTML, XML, JSON, and plain text.

- AJAX requests and responses are typically asynchronous, which means that they occur independently of the rest of the web page's code.
- AJAX can be used to create interactive and responsive web applications, such as web-based email clients, real-time chat applications, and online shopping carts.
- AJAX requires careful consideration of security issues, such as cross-site scripting (XSS) attacks and cross-site request forgery (CSRF) attacks.
- AJAX is widely used in modern web development and is an important tool for creating dynamic and interactive web pages.

Others

- use `defer` or put js to the end to avoid the js error "cannot set property"?
- `isset()` method to check if a variable is set in your script.
- the build-in function that outputs information about PHP's configuration is: `phpini()`;
- `position:relative` style will Move the element's relative to the same position
- absolute value: px, pt, pc, in, cm
- relative value: em, rem, %, vw
- in js, concate a number with a string, boolean, and a number with double quote, and a number with single quote

```
console.log(1 + "1"); // Output: "11"
console.log(1 + '1'); // Output: "11"
console.log("1" + 1); // Output: "11"
console.log(+"1" + 1); // Output: "2"
console.log(1 + "hahaha"); // Output: "1hahaha"
console.log(1 + false); // Output: 1 (false is converted to 0)
console.log(1 + true); // Output: 2 (true is converted to 1)
```

- requive vs include: Require will give a fatal error upon failure.]
- an option box that allows for multiple values to be selected need to use `multiple` attribute

```
<select name="colors" multiple>
  <option value="red">Red</option>
  <option value="green">Green</option>
  <option value="blue">Blue</option>
</select>
```

- js array methods

```
// create an array of numbers
var numbers = [1, 2, 3, 4];

// add a number to the end of the array
numbers.push(5);

// remove the last number from the array
```

```
var lastNumber = numbers.pop();

// add a number to the beginning of the array
numbers.unshift(0);

// remove the first number from the array
var firstNumber = numbers.shift();

// log the results to the console
console.log(numbers);     // [0, 1, 2, 3]
console.log(firstNumber); // 1
console.log(lastNumber);  // 5
```

- PHP array

```
// declare a array using SQUARE brackets
$products = ['Tires', 'Oil', 'Spark Plugs'];

// use array() method
$products = array('Tires', 'Oil', 'Spark Plugs');
```

- PHP array methods explode:
    - returns an array of strings, each of which is a substring of main string formed by splitting it on boundaries formed by a delimiter.

```
$string = "Hello, World!";
$delimiter = ",";
$words = explode($delimiter, $string);
```

- in js, array can hold a mix of different data type

```
// In this example, myArray contains a number (1), a string ("two"), a boolean
(true), an object ({name: "John", age: 30}), and an array ([4, 5, 6]). You can
access each element of the array using its index:
var myArray = [1, "two", true, {name: "John", age: 30}, [4, 5, 6]];
```

- JS Object

```
var bird = {
  genus : 'corvus',
  species : 'corvax',
  commonName: 'raven',
  maxOffspring : 5,
```

```
  noisy : true,
  deadly : false
};

// access the bird's value
bird."species"
```

- HTML Form fieldset

```
<form>
  <fieldset>
    <legend>Contact Information</legend>
    <label for="name">Name:</label>
    <input type="text" id="name" name="name"><br>
    <label for="email">Email:</label>
    <input type="email" id="email" name="email"><br>
    <label for="phone">Phone:</label>
    <input type="tel" id="phone" name="phone"><br>
  </fieldset>
  <button type="submit">Submit</button>
</form>
```