**PHP**

* **Differentiate between variables and constants in PHP**

The value of a variable can be changed during the execution.

The constant value can’t be changed during script execution.

* **What is a session in PHP?**

Session variables hold single user information and are available to all pages in one application.

* **What is the difference between “echo” and “print” in PHP?**

echo can output one or more strings.

print can only output one string and it always returns 1.

* **What is the purpose of @ in PHP?**

In PHP, @ is used for suppressing error messages.

* **What are the different types of Array in PHP?**

There are 3 main types of arrays that are used in PHP:

* + **Indexed Array**

An array with a numeric key is known as the indexed array. Values are stored and accessed in linear order.

* + **Associative Array**

An array with strings for indexing elements is known as the associative array. Element values are stored in association with key values rather than in strict linear index order.

* + **Multidimensional Array**

An array containing one or more arrays within itself is known as a multidimensional array. The values are accessed using multiple indices.

* **What are traits?**

Traits are a mechanism that lets you create reusable code in PHP and similar languages where multiple inheritances are not supported. It’s not possible to instantiate it on its own.

A trait is intended to reduce the limitations of single inheritance by enabling a developer to reuse sets of methods freely in many independent classes living in different hierarchies of class.

* **Does JavaScript interact with PHP?**

JavaScript is a client-side programming language, whereas PHP is a server-side scripting language. PHP has the ability to generate JavaScript variables, and this can be executed easily in the browser. Thereby making it possible to pass variables to PHP using a simple URL

* **What is NULL?**

NULL is a special data type which can have only one value. A variable of data type NULL is a variable that has no value assigned to it.

* **Name some of the constants in PHP and their purpose.**

\_LINE\_ – It represents the current line number of the file.

\_FILE\_ – It represents the full path and filename of the file. If used inside an include, the name of

the included file is returned.

\_FUNCTION\_ – It represents the function name.

\_CLASS\_ – It returns the class name as it was declared.

\_METHOD\_ – It represents the class method name.

* **What is overloading and overriding in PHP?**

Overloading is defining functions that have similar signatures, yet have different parameters. Overriding is only pertinent to derived classes, where the parent class has defined a method and the derived class wishes to override that method. In PHP, you can only overload methods using the magic method \_\_call.

* **What is the use of callback in PHP?**

PHP callback are functions that may be called dynamically by PHP. They are used by native functions such as array\_map, usort, preg\_replace\_callback, etc. A callback function is a function that you create yourself, then pass to another function as an argument. Once it has access to your callback function, the receiving function can then call it whenever it needs to.

* **What is the use of count() function in PHP?**

The PHP count() function is used to count total elements in the array, or something an object.

* **What does isset() function?**

The isset() function checks if the variable is defined and not null.

* **What is the difference between indexed and associative array?**

The indexed array holds elements in an indexed form which is represented by number starting from 0 and incremented by 1. For example:

* + $season=array("summer","winter","spring","autumn");

The associative array holds elements with name. For example:

* + $salary=array("Sonoo"=>"350000","John"=>"450000","Kartik"=>"200000");
* **What is Object-Oriented Programming?**
  + **Data** **encapsulation**: Wrapping up chunks of data under a single main title to make the development process convenien
  + **Data** **abstraction**: Hiding the background working of methods and functions helps on reducing development complexity by exposing only the operation details and hiding the implementation details.
  + **Data** **inheritance**: Transferring important data from a parent class to a child class, which helps in reusability.
  + **Polymorphism** - The ability of a code to have multiple inputs and a common output.
* **What is a class?**

Classes are basically a structure for an object, where it helps the user to store both variables as well as functions, and altogether they form a collection like a package, which is called an object.

* **What is an object?**

Objects are made from Classes; it can be considered as an occasion of a class that is made dynamic. Object in computer programs is like the genuine word object. Each programming object has a few properties and ways of behaving.

* **What are the different types of visibility? OR What are the access modifiers?**

There are three types of access modifiers in PHP.

* + **Public**: Public strategy or variable can be open from any place, which means a public technique or variable of a class can be called beyond the class or in a subclass.
  + **Protected**: A safeguarded strategy or variable must be brought into that class and its subclass.
  + **Private**: A confidential technique or variable of a class must be called inside that class just in which it is proclaimed.

Laravel

* **What is the templating engine used in Laravel?**

The templating engine used in Laravel is Blade. The blade gives the ability to use its mustache-like syntax with the plain PHP and gets compiled into plain PHP and cached until any other change happens in the blade file. The blade file has .blade.php extension.

* **What are available databases supported by Laravel?**
  + PostgreSQL
  + SQL Server
  + SQLite
  + MySQL
* **What is an artisan?**
  + php artisan make:controller - Make Controller file
  + php artisan make:model - Make a Model file
  + php artisan make:migration - Make Migration file
  + php artisan make:seeder - Make Seeder file
  + php artisan make:factory - Make Factory file
  + php artisan make:policy - Make Policy file
  + php artisan make:command - Make a new artisan command
* **How to put Laravel applications in maintenance mode?**
  + php artisan down
  + php artisan up
* **What are facades?**

Facades are a way to register your class and its methods in Laravel Container so they are available in your whole application after getting resolved by Reflection.

The main benefit of using facades is we don’t have to remember long class names and also don’t need to require those classes in any other class for using them. It also gives more testability to the application

* **What are Events in Laravel?**

In Laravel, Events are a way to subscribe to different events that occur in the application. We can make events to represent a particular event like user logged in, user logged out, user-created post, etc. After which we can listen to these events by making Listener classes and do some tasks like, user logged in then make an entry to audit logger of application.

* **What is Localization in Laravel?**

Localization is a way to serve content concerning the client's language preference. We can create different localization files and use a laravel helper method like this: `\_\_(‘auth.error’)` to retrieve translation in the current locale. These localization files are located in the resources/lang/[language] folder.

* **What is a Service Provider?**

A Service Provider is a way to bootstrap or register services, events, etc before booting the application. Laravel’s own bootstrapping happens using Service Providers as well. Additionally, registers service container bindings, event listeners, middlewares, and even routes using its service providers.

* **What are queues in Laravel?**

While building any application we face a situation where some tasks take time to process and our page gets loading until that task is finished. One task is sending an email when a user registers, we can send the email to the user as a background task, so our main thread is responsive all the time. Queues are a way to run such tasks in the background

**VUE**

* **What exactly is the Prop component?**

Data is passed from the parent component to the child component using Component Prop. A component can have as many props as it wants. The prop is a custom attribute that becomes a property on the component instance when a value is supplied.

* **Describe the lifecycle hooks in Vue.js?**
  + Before Create - This is the first lifecycle hook that is called when a Vue instance has been created.
  + Created - It's called immediately after the 'Before creates' hook, but the Vue instance has already set initial properties, data, etc.
  + Before mount - invoked right before the instance is mounted on the DOM. The template has been completed at this time.
  + Mounted - This is the name given to the template once it has been filled with data and is fully functional.
  + Before Updated - When any changes to data require the DOM to be changed, this method is called before the update.
  + Updated - It's called after the DOM has been updated.
  + Before destroy - It's a location where you can clean up your resources before terminating the Vue instance.
  + Destroyed - When all Vue instances have been destroyed, this method is called. When you call the destruct method on an object in code, it will be activated.
* **What are the most essential elements of the State Management Pattern?**
  + Our app is driven by the state, which is the source of truth.
  + The view is nothing more than a declarative state mapping.
  + The actions are the various ways in which the state could change due to user input from the view.
* **What are mixins?**

Mixins are a versatile approach to sharing reusable Vue component functionality. Any component choices can be included in a mixin object. When a component employs a mixin, all of the mixin's options are "mixed" into the component's own.

* **What is Virtual DOM?**
  + The Document Object Model (DOM) is an interface that allows languages like JavaScript to interact with and change HTML documents. Nodes in a tree represent elements, and the interface lets us manipulate them. This interface, however, comes with a cost, and the page will be slowed down by a considerable number of persistent DOM operations.
  + Vue overcomes this problem by storing a virtual representation of the document's structure in memory, where a virtual node (vnode) represents nodes in a document's structure. When manipulation is required, the computations and operations are executed in memory on the Virtual DOM rather than on the real DOM. This is faster, and it allows the Virtual DOM algorithm to determine the most efficient method of updating the actual DOM structure.
  + This is then applied to the actual DOM tree after it has been computed. This improves performance, which is why Virtual DOM-based frameworks like Vue and React have become so popular.