Laravel is a free, open-source web framework based on [PHP.](https://www.simplilearn.com/why-learn-php-article) It is developed by Taylor Otwell, who began building it in 2011. Laravel supports the [MVC](https://www.simplilearn.com/tutorials/dot-net-tutorial/mvc-architecture) (Model-View-Controller) architectural pattern, which makes it easy for you to create an elegant and expressive syntax for your web application.

It can be difficult to find your way around Laravel interview questions, especially if you want to work in web development. Even though Laravel is a PHP framework, it's important to comprehend its nuances and how it works with different technologies. A [java full stack developer course](https://www.simplilearn.com/mobile-and-software-development/java-javaee-soa-development-training) could be the ideal next step for developers who want to expand into comprehensive web programming and move beyond PHP.

Here are some Laravel questions that will help ace your Laravel technical interviews.

## Top Laravel Interview Questions

### 1. What is the latest Laravel version?

The latest Laravel version is 9. It was launched on February 8th, 2022.

### 2. Define Composer.

Laravel is a popular web application framework that allows you to build dynamic websites and applications.

A composer is a tool that includes all the dependencies and libraries. It helps the user to develop a project concerning the mentioned framework. Third-party libraries can be installed easily using composer.

Composer is used to managing its dependencies, which are noted in the composer.json file and placed in the source folder.

### 3. What is the templating engine used in Laravel?

The Laravel Blade templating engine is a powerful piece of the framework that allows you to easily create powerful templates with a syntax that's simple and intuitive.

The Blade templating engine provides structure, such as conditional statements and loops. To create a blade template, you just need to create a view file and save it with a .blade.PHP extension instead of a .php extension. The blade templates are stored in the /resources/view directory. The main advantage of using the blade template is that we can create the master template, which other files can extend.

### 4. What are available databases supported by Laravel?

Laravel has you covered. The database configuration file is app/config/database.php. You can define your database connections in this file and specify which you should use reference. Examples for all of the supported database systems are provided in this file.

Laravel supports four database systems: [MySQL](https://www.simplilearn.com/tutorials/mysql-tutorial), [Postgres](https://www.simplilearn.com/tutorials/sql-tutorial/postgresql-tutorial), [SQLite](https://www.simplilearn.com/tutorials/sql-tutorial/what-is-sqlite), and [SQL Server.](https://www.simplilearn.com/what-is-microsoft-sql-server-architecture-article)

### 5. What is an artisan?

The artisan script is a command-line interface included with Laravel. It's the first thing you'll see when you run composer create-project, or PHP artisan serve.

Artisan is made up of commands and is one of your best friends for developing and managing your Laravel applications. You can view a list of all available Artisan commands by running PHP artisan list.

### 6. How to define environment variables in Laravel?

 In Linux, you have probably become familiar with environment variables. You can check the available environment variables with the printenv command.

To define an environment variable in Linux, use the export command followed by your new variable name: export name=Simplilearn.

The .env file holds your env variables for your current environment. The DotEnv Library powers it.

As the .env file often holds sensitive information like API keys or database credentials, you should never commit it to Git and push it to GitHub.

### 7. Can we use Laravel for Full Stack Development (Frontend + Backend)?

Laravel is a great choice for building full-stack web applications. With Laravel, you can create a backend that will be scalable, and the frontend can be built using blade files or SPAs using [Vue.js,](https://www.simplilearn.com/vue-js-interview-questions-article) which is provided by default. But it can also be used to just provide APIs for a SPA application.

### 8. How to put Laravel applications in maintenance mode?

Laravel makes it easy to manage your deployment with minimal effort. Laravel allows you to quickly and easily disable your application while updating or performing maintenance when you need to make changes to your server or database.

To enable maintenance mode, the following are some helpful laravel commands related to maintenance mode:

# enable maintenance mode

php artisan down

# disable maintenance mode

php artisan up

# if you want the client to refresh

# page after a specified number of seconds

php artisan down --retry=60

### 9. What are the default route files in Laravel?

You can define Laravel routes in your routes/web.php file or create a separate file for other types of routes.

All routes are defined in your route files, located in the routes directory. The Laravel framework automatically loads these files. The routes/web.php file defines routes for your web interface. These routes are assigned to the web middleware group, providing features like session state and CSRF protection. The routes in routes/api.php are stateless and set in the API middleware group.

For most applications, you will begin by defining routes in your routes/web.php file. You may access the routes described in routes/web.php by entering the designated route's URL in your browser or through one of your controllers' actions or methods (explained later).

### 10. What are migrations in Laravel?

Migration is a feature of Laravel that allows you to modify and share the application's database schema. It will enable you to alter the table by adding a new column or deleting an existing column.

If you have ever had to tell a teammate to add a column to their local database schema manually, you've faced the problem that database migrations solve. Migrations are like version control for your database, allowing your team to modify and share the application's database schema. Migrations are typically paired with Laravel's schema builder to build your application's database schema.

The Laravel Schema facade provides database agnostic support for creating and manipulating tables across all of Laravel's supported database systems.

### 11. What are seeders in Laravel?

Laravel's database seeding feature allows you to quickly insert data into your database. It is helpful for development environments where you may not have access to your production database.

Laravel includes the ability to seed your database with data. By default, a Database seeder class is defined for you. You may use the call method from this class to run other seed classes. All seed classes are stored in the database/seeders directory.

A seeder class only contains one method: run. This method is called when the db:seed Artisan command is executed. You may use the query builder to insert data or Eloquent model factories.

### 12. What are the factories in Laravel?

Laravel has an excellent model factory feature that allows you to build fake data for your models. It is beneficial for testing and seeding counterfeit data into your database to see your code in action before any accurate user data comes in.

By default, Laravel's database seeding feature will create a new row in the database table and insert the value into each field. But sometimes, you might want a few extra areas or some sort of random string instead of a numeric value. That's where model factories come in handy!

Model Factories allow you to create a new model instance using their rules. You can do anything from creating an empty model instance to creatinbuildingth all fields filled out with values or even random ones!

### 13. How to implement soft delete in Laravel?

 Laravel 5.6 has a new feature called soft deletes. When soft deleted models, they aren't removed from our database. Instead, a deleted\_at timestamp is set on the record.

To enable soft deletes for a model, you have to specify the soft delete property on the model like this:

Use Illuminate\Database\Eloquent\SoftDeletes;

Use SoftDeletes; in our model property.

After that, when you use the delete() query, PHP will not remove records from the database. Then a deleted\_at timestamp is set on the record.

### 14. What are Models?

 Laravel is a framework that follows the Model-View-Controller design pattern. All your models, views, and controllers are stored in their directories, making it easy to keep track of everything.

You'll use controllers to handle user requests and retrieve data by leveraging models. Models interact with your database and recover your objects’ information. Finally, views render pages.

Laravel comes with a fantastic, built-in command line interface called Artisan CLI that provides complete commands to help you build your application.

### 15. What is the Laravel Framework?

Laravel is an open-source [PHP framework](https://www.simplilearn.com/tutorials/php-tutorial/php-framework), which is robust and easy to understand. It follows a model-view-controller design pattern. Laravel reuses the existing components of different frameworks which helps in creating a web application. The web application thus designed is more structured and pragmatic.

Laravel offers a rich set of functionalities that incorporates the basic features of PHP frameworks like CodeIgniter, Yii, and other [programming languages](https://www.simplilearn.com/best-programming-languages-start-learning-today-article) like [Ruby on Rails.](https://www.simplilearn.com/tutorials/java-tutorial/what-is-ruby-on-rails) Laravel has a very rich set of features that will boost the speed of web development.

With Laravel, you can build applications for any type of business or organization. Whether it’s eCommerce, social media marketing, or an online ticketing system, you can create any type of web application with Laravel because it’s flexible and scalable enough to accommodate any size project easily.

### 16. What's New in Laravel 8?

Laravel 8.0 is the latest version of Laravel Framework. The Laravel 8.0 was released on September 8th, 2020, with the latest and unique features making it one of the top frameworks in today's world. This latest version brings many new features, such as Laravel Jetstream, model directory, migration squashing, rate limiting improvements, model factory classes, time testing helpers, dynamic blade components, and more.

Following are some of the new features in Laravel 8

* Time Testing Helpers
* Models Directory
* Migration Squashing
* Laravel Jetstream
* Rate Limiting Improvements
* Model Factory Classes
* Dynamic Blade Components

### 17. How to enable query log in laravel?

Our first step should be

DB::connection()->enableQueryLog();

After our query, you should place it

$querieslog = DB::getQueryLog();

After that, you should place it

dd($querieslog)

### 18. What is Middleware in Laravel?

Middleware in laravel is a platform that works as a bridge between the request and the response. The main aim of middleware is to provide the mechanism for investigating HTTP requests entering into your application. For instance, middleware in laravel ensures that the user of your particular application is authenticated. If they find that the user is not authenticated, it will redirect the user to the main login page of the application.

Middleware in laravel also helps you to handle a request from a user who has already been authenticated. For example, if you want to display information about a user who has already been established, then middleware will help you by providing this functionality within your application.

### 19. What is reverse Routing in Laravel?

Reverse routing is the process used to generate the URLs based on the names or symbols. It is also known as backtracking or reverse mapping.

You can do reverse routing in two ways:

1. Using route parameters in routes
2. Using route names in routes

When you use the URL structure, you can simply add a symbol or name at the end of your URL. It will let you create more readable URLs in your application and make it easier for users to understand them.

Route:: get(‘list’, ‘blog@list’);

{{ HTML::link\_to\_action('blog@list') }}

### 20. What is a Service container?

A Service container is one of the most powerful tools used to manage dependencies over the class and perform dependency injections.

* A service container can be used as a registry to keep track of all the classes in use within your application.
* In addition, it also helps in binding interfaces to concrete classes.

### 21. What is Auth? How is it used?

Laravel Auth is an in-built functionality provided by Laravel to identify the user credentials with the database. It takes input parameters like username and password for user identification. If the settings match, then the user is said to be authenticated.

If you want to authenticate your laravel application, then you can use the auth function.

### 22. How to mock a static facade method in Laravel?

 Laravel facades provide a static interface to classes available inside the application's service container. They're used to provide a simple way to access complex objects and methods, and they're often used to centralize the configuration of those objects.

Facades can be mocked in Laravel using the shouldRecieve method, which returns an instance of a facade mock.

$value = Cache::get('key');

Cache::shouldReceive('get')->once()->with('key')->andReturn('value');

### 23. What is composer lock in laravel?

After you run composer install in your project directory, the Composer will generate a composer.lock file. It will record all the dependencies and sub-dependencies installed by the composer.json.

### 24. What is Dependency injection in Laravel?

In Laravel, dependency injection is a term used for the activity of injecting components into the user application. It’s a critical element of agile architecture. The Laravel service container is a powerful tool that manages all class dependencies and performs dependency injection.

The Laravel service container is a tool that manages all class dependencies and performs dependency injection.

public function \_\_construct(UserRepository $data)

{

    $this->userdata = $data;

}

### 25. How to use skip() and take() in Laravel Query?

 If you're looking for a way to limit the number of results in your query, skip() and take() are two great options!

skip() is used to skip over several results before continuing with the query. Take() is used to specify how many results you want from the query.

For example, if you want to get only five results from a query that usually returns 10, you'd use skip(5). If you're going to start at the 6th result in the question and get everything after it (skipping 1), then you'd use take(1).

### 26. What is the Repository pattern in laravel?

The repository pattern decouples data access layers and business logic in our application. It allows objects without having to know how these objects persisted. Your business logic does not need to understand how data is retrieved. The business logic relies on the repository to get the correct data.

This pattern makes our code cleaner and easier to maintain because we can change the implementation of our data layer without changing any related code.

### 27. What is the Singleton design pattern in laravel?

 The Singleton Design Pattern in Laravel is one where a class presents a single instance of itself. It is used to restrict the instantiation of a class to a single object. It is useful when only one example is required across the system. When used properly, the first call shall instantiate the object. After that, Laravel shall return all calls to the same instantiated object.

When working with an application that requires multiple instances of an object, it can be hard to manage them all and ensure another part of your codebase is not changing them.

There are many ways you could handle this issue:

* Use a static variable inside your class and update it manually each time you want to use it
* Use a global variable and check if it's set before using it
* Use dependency injection (you'll need this if you want to inject services into your classes)

### 28. What are the advantages of Queue?

Laravel queues are a great way to handle time-consuming tasks. They allow you to offload work from your web server, so your users don't have to wait for a response from your API before getting the next page.

Queues are also helpful if your application has multiple servers and you want to use them all without having jobs interfere with each other. For example, if one server is running PHP code while another is running Python code, they shouldn't be able to interfere with each other's processes.

### 29. What is tinker in Laravel?

Laravel Tinker is a powerful REPL tool used to interact with the Laravel application with the command line in an interactive shell. Tinker came with the release of version 5.4 and is extracted into a separate package.

Tinker is an excellent tool for debugging and exploring your code. You can use it to inspect variables, models, classes, and methods at runtime. You can change anything in the console, and it will instantly update the page you are working on.

For Installing tinker, you can use composer require laravel/tinker.To execute tinker, we can use the php artisan tinker command.

### 30. What is a REPL?

 REPL stands for Read—Eval—Print—Loop. It's a type of interactive shell that takes in single user inputs, processes them, and returns the result to the client.

If you've ever used a programming language like Ruby or Python, you've probably used a REPL. They're extremely useful for testing snippets of code and running little experiments on your computer.

### 31. What are the basic concepts in laravel?

* Blade Templating
* Routing
* Eloquent ORM
* Middleware
* Artisan(Command-Line Interface)
* Security
* In-built Packages
* Caching
* Service Providers
* Facades
* Service Container

### 32. What is a lumen?

 Lumen is a micro PHP framework introduced by Taylor Otwell, the creator of Laravel. It is a faster, smaller, and leaner version of a full web application framework that uses the same components as Laravel, but especially for microservices.

Lumen is a lightweight framework for building web applications in PHP. It is built on top of Laravel and follows the same best practices that you have come to know in Laravel. In fact, Lumen was designed to be used as an alternative to Laravel when you only need to build small applications and services.

Lumen is ideal for both new projects and existing projects that require a microservice architecture or fast prototypes.

To install Lumen you can use the following command.

composer global require "laravel/lumen-installer=~1.0"

### 33. How do I stop Artisan service in Laravel?

If you're having trouble with your server, here are a few steps to help you troubleshoot.

First, try pressing Ctrl + Shift + ESC. This will open up the task manager, where you can locate the php system walking artisan process and kill it with a proper click.

Next, reopen your command line and begin again the server.

Note that it may be possible to kill the process just by sending it a kill sign with Ctrl + C.

### 34. What are the features of Laravel?

* Offers a rich set of functionalities like Eloquent ORM, Template Engine, Artisan, Migration system for databases, etc
* Libraries & Modular
* It supports MVC Architecture
* Unit Testing
* Security
* Website built in Laravel is more scalable and secure.
* It includes namespaces and interfaces that help to organize all resources.
* Provides a clean API.

### 35. What is validation in Laravel?

Laravel provides several different ways to validate your application's incoming data. The most common way is by creating a Form Request.

Form Requests allow you to validate incoming data with ease, so you don't have to worry about creating validation rules or manually checking for errors. Form Requests also support custom validation rules and custom error messages, which means you can build your own validations that are specific to your application.

### 36. What is a yield in Laravel?

@yield is a Blade directive that allows you to pull content from a child page into a master page, and it's used in Laravel to define sections in a layout. When the Laravel framework performs the blade file, it first checks to see if you have extended a master layout. If you have, it will move on to the master layout and start grabbing content from @sections.

### 37. What is Nova?

Laravel Nova is an admin panel built on the Laravel Framework. It's perfect for managing your database records, and it's easy to install and maintain.

Laravel Nova comes with features that have the ability to administer your database records using Eloquent.

### 38. Explain ORM in Laravel.

ORM stands for Object-Relational Mapping. It is a programming technique that is used to convert data between incompatible type systems in object-oriented programming languages.

The ORM is used to map objects in the application's domain model to relational database tables, and vice versa. In this way, the ORM lets you work with your domain objects as if they were an old-fashioned collection of fields and properties while keeping the more recent advantages of a relational database.

### 39. Explain MVC Architecture

MVC stands for Model View Controller. It segregates domain, applications, business and logic from the user interface. This is achieved by separating the application into three parts:

* Model: Data and [data management](https://www.simplilearn.com/what-is-data-management-article) of the application
* View: The user interface of the application
* Controller: Handles actions and updates

### 

### 40. What is Routing?

[Routing](https://www.simplilearn.com/tutorials/angular-tutorial/angular-routing) refers to accepting requests and sending them to the relevant function of the controller. The route is a method of creating the request URL of the application. These URLs are readable and also SEO-friendly. Routes are stored under the /routes folder inside files. The default Routes in Laravel are used for registering:

* web.php: web routes
* api.php: API routes
* console.php: console-based commands
* channel.php: broadcasting channels supported by the application

### 41. What is the use of Bundles in Laravel?

Popularly known as packages, Bundles are a convenient way to group code. Bundles extend the functionality of Laravel since they can have views, configuration, migration, tasks and more. A Bundle can range from database ORM to an authentication system.

### 42. Explain Seeding.

Seeding refers to introducing test data to the database for testing the application. Developers can add dummy data to their database tables using a database seeder. Different data types allow the developer to detect bugs and improve performance.

### 43. How do you check the installed Laravel version of a project?

Using the command PHP Artisan --version or PHP Artisan -v

### 44. Which Artisan command gives a list of available commands?

PHP Artisan list

### 45. What is the difference between the Get and Post method?

Both Get and [Post](https://www.simplilearn.com/tutorials/php-tutorial/php-post-method) are used to retrieve input values in Laravel. A limited amount of data in the header is allowed in the Get method, whereas the Post method allows sending large amounts of data in the body.

### 46. What are some common Artisan commands in Laravel?

make:controller – Creates a new Controller file in App/Http/Controllers folder

make:model – Creates a new Eloquent model class

make:migration – Creates a new migration file

make:seeder – Creates a new database seeder class

make:request – Creates a new form request class in App/Http/Requests folder

make:command – Creates a new Artisan command

make:mail – Creates a new email class

make:channel – Creates a new channel class for broadcasting

### 47. Explain the project structure in Laravel.

The following list of directories/folders shows the typical project structure in Laravel

* app: source code of the application resides here, and it has the following sub-folders:

1. Consoler
2. Exceptions
3. Http
4. Providers

* bootstrap: contains files required to bootstrap an application and configure auto-loading
* config: configuration files such as app.php, auth.php, broadcasting.php, cache.php, database.php and so on are stored in this folder
* database: holds the database files and has .gitignore file and the following sub-folders:

1. factories
2. migration
3. seeds

* public: contains files used to initialise the web application
* resources: The resource directory contains files to enhance the web application and has the following sub-folders:

1. Assets
2. Lang
3. Views

* routes: includes route definitions
* storage: stores the cache and session files and has sub-folders:

1. app
2. framework
3. logs

* test: holds the automated unit test cases
* vendor: contains the composer dependency packages

### 48. Give an example to describe how a Route is created.

Routes are created in the routes folder. The files web.php and api.php have the routes that have been created for websites and API, respectively.

Route::get('/', function () {

   return view('welcome');

});

The above Route is defined for the homepage, and it returns the view “Welcome” every time it gets a request for /.

### 49. Name the template engine used in Laravel?

The name of the template engine in Laravel is Blade. It enables you to use the mustache-like syntax with plain PHP. It then gets compiled and cached until the blade file changes. The file has a .blade.PHP extension.

### 50. What is soft delete in Laravel?

You don't delete data in soft delete but add a deleted tag. This way, you can quickly restore data when needed.

### 51. Describe localization in Laravel?

Serving the content according to the client's language preference is localization.

### 52. What are Laravel's registries?

You use Requests to interact with HTTP requests and session cookies. For doing this, you can use the class illuminate\Http\Request. After submitting a request to a Laravel route, the request object is available within the method using dependency injection.

### 53. How does request validation happen in Laravel?

You can either create a request validation class or a controller method for request validation. For example:

public function store (Request $request)

{

$validated = $request->validate([ 'title' => 'required|unique:posts|max"255', 'body' => 'required',]);

}

### 54. Describe the service provider in Laravel.

You can register services, events, etc., using a service provider before booting the application. They help inject Laravels services or your application services and dependencies.

### 55. Explain register and boot methods in the service provider class.

The register method in the Service Provider class binds classes and services to Service Containers. And the boot method runs after the program includes all the dependencies in the container. Then you can create routes and view composers in the boot method.

### 56. How to create a middleware in Laravel?

Using a middleware, you can filter and inspect HTTP requests. You can create a middleware using the following code.

PHP artisan make middleware AllowSmallFile

### 57. Define collections in Laravel.

A collection is an API wrapper for PHP array functions; you can generate it from an array. Using a collection, you can reduce or map arrays.

### 58. Explain queues in Laravel?

Time-consuming tasks can make an interface unresponsive. You must run such tasks in the background. In Laravel, queues are a way to run these tasks in the background so that your main threads will remain responsive.

### 59. Define accessors and mutators.

Accessors allow you to change the data after acquiring it from the database. And mutators enable you to modify data before saving it to a database.

### 60. What are relationships in Laravel?

If you're familiar with working with Eloquent models, you know that relationships are defined as methods in your Eloquent model classes. Since relationships also serve as influential query builders, defining relationships as methods provides powerful method chaining and querying capabilities.

Following are the types of relationships in Laravel

* One To One relationship.
* One To Many relationships.
* Many To Many relationships.
* Has Many Through relationships.
* Polymorphic relationships.
* Many To Many Polymorphic relationships.

### 61. What is Eloquent in Laravel?

 Laravel is a PHP framework that allows you to develop web applications quickly. It is easy to learn, comes with a lot of prebuilt functionality, and makes it easy for developers to create complex websites and other applications.

It's built on top of Symfony components, which you can use for different websites, including e-commerce sites and business apps. The framework also includes an ORM (Object Relational Mapper) called Eloquent, which is used to communicate with the database.

### 62. What is throttling and how to implement it in Laravel?

Throttling is a great way to rate-limit requests from a particular IP. This can be used to prevent DDOS attacks as well. Laravel provides a middleware that can be applied to routes, which can be added to the global middleware list as well.

You can implement throttling as below:

Route::middleware('auth:api', 'throttle:60,1')->group(function () {

    Route::get('/user', function () {

        //

    });

});

### 63. What are facades?

A facade is a way to access classes available in the application's service container. The service container holds all of your application's business logic, and facades provide a "static" interface to those classes.

There are lots of Laravel facades, and they're all over the place in the framework! You can see them in any controller or view file.

You can access a facade as mentioned below

use Illuminate\Support\Facades\Cache;

use Illuminate\Support\Facades\Route;

Route::get('/cache', function () {

   return Cache::get('key');

});

### 64. What are Events in Laravel?

Events are great for decoupling different parts of your application. For example, suppose you want to send a Slack notification each time an order has been shipped instead of coupling your order processing code to your Slack notification code. In that case, you can raise an App\Events\OrderShipped event that the listener can receive and use to dispatch a Slack notification.

Each time you generate an event or listener, the event will store it in its directory.

Events are typically stored in the app/Events directory, while their listeners are stored in app/Listeners. You can see these directories by running `PHP artisan make: event OrderShipped.`

You don't have to worry about these directories if you don't see them in your application. They'll be created for you as you generate events and listeners using Artisan console commands.

### 65. Explain logging in Laravel?

Logging can be a powerful tool for tracking down bugs, but it can also be overwhelming to manage. Laravel helps you make sense of your log messages by providing an easy-to-use logging system that allows you to write log messages to files, the system error log, Slack, and more.

Laravel logging is based on "channels." Each channel represents a specific way of writing log information. Log messages may be written to multiple channels based on their severity.

Under the hood, Laravel utilizes the Monolog library, which supports a variety of powerful log handlers. Laravel makes it a cinch to configure these handlers, allowing you to mix and match them to customize your application's log handling.

## Frequently Asked Questions

### 1. Why is Laravel used?

Using Laravel, you can simplify the development process. It simplifies the regular tasks such as authentication, routing, and sessions. This is one of the most common Laravel Interview questions.

### 2. What is MVC in Laravel?

MVC means Model View Controller and is the architecture developers use when they build an application. MVC architecture will enable you to understand how the application's data flow works.

### 3. What is namespace in Laravel?

Namespaces are an element's class where each element has a unique name. You can share them with elements in other classes.

### 4. Does Laravel support Bootstrap?

Yes.

### 5. Name the aggregate methods of the Query Builder.

The Query Builder provides variety of aggregate methods such as:

count(), max(), min(), avg() and sum().

### 6. How is a Blade template file identified?

Blade template files have .blade.php extension and are located in the resources/views folder.

### 7. Name the ORM used in Laravel.

Eloquent is the ORM (Object Relational Mapper) used in Laravel.

### 8. What is Vapor?

A completely serverless deployment and auto-scaling platform for Laravel. It is powered by Amazon Web Services (AWS) Lambda.

### 9. Name some common tools used to send emails in Laravel.

* SwiftMailer
* SMTP
* Mailgun
* Mailtrap
* Mandrill
* Postmark
* Amazon SES (Simple Email Service)
* Sendmail

### 10. What is Forge?

Serve Management & Application Deployment Service.

### 11. Name a few competitors of Laravel.

* CodeIgniter
* Angular
* Symfony
* Slim Framework
* Modx
* Yii
* CakePHP
* Placon
* Scriptcase

## What is Laravel Framework?

Laravel is an open-source [PHP web application framework](https://www.interviewbit.com/blog/php-frameworks/). It is a very well documented, expressive, and easy to learn framework. Laravel is very developer-friendly as the framework can help beginners as well as advanced users. As you grow as a developer you can go more deep into Laravel functionalities and give more robust and enterprise solutions.

Additionally, the framework is very scalable as you can use packages like Vapor to handle hundreds of thousands of requests using AWS serverless technology.

This article will walk you through basic Laravel interview questions to advanced questions.

## Laravel Interview Questions For Freshers

### 1. What are migrations in Laravel?

In simple, Migrations are used to create database schemas in Laravel. In migration files, we store which table to create, update or delete.

Each migration file is stored with its timestamp of creation to keep track of the order in which it was created. As migrations go up with your code in GitHub, GitLab, etc, whenever anyone clones your project they can run `PHP artisan migrate` to run those migrations to create the database in their environment. A normal migration file looks like below:

<?php

use Illuminate\Database\Migrations\Migration;

use Illuminate\Database\Schema\Blueprint;

use Illuminate\Support\Facades\Schema;

class CreateUsersTable extends Migration

{

/\*\*

\* Run the migrations.

\*

\* @return void

\*/

public function up()

{

Schema::create('users', function (Blueprint $table) {

$table->id();

$table->string('name');

// Create other columns

});

}

/\*\*

\* Reverse the migrations.

\*

\* @return void

\*/

public function down()

{

Schema::dropIfExists('users');

}

}

The up() method runs when we run `php artisan migrate` and down() method runs when we run `php artisan migrate:rollback`.

If we rollback, it only rolls back the previously run migration.

If we want to rollback all migrations, we can run 'php artisan migrate:reset`.

If we want to rollback and run migrations, we can run `PHP artisan migrate:refresh`, and we can use `PHP artisan migrate:fresh` to drop the tables first and then run migrations from the start.

### 2. What is the latest Laravel version?

The latest Laravel version is 8.x.

### 3. What are Models?

With Laravel, each database table can have a model representation using a model file which can be used to interact with that table using Laravel Eloquent ORM.

We can create a model using this artisan command:

class Post extends Model

{

/\*\*

\* The attributes that are mass assignable.

\*

\* @var array

\*/

protected $fillable = [];

/\*\*

\* The attributes that should be hidden for arrays.

\*

\* @var array

\*/

protected $hidden = [];

}

### 4. How to implement soft delete in Laravel?

Soft Delete means when any data row is deleted by any means in the database, we are not deleting the data but adding a timestamp of deletion.

We can add soft delete features by adding a trait in the model file like below.

use Illuminate\Database\Eloquent\Model;

use Illuminate\Database\Eloquent\SoftDeletes;

class Post extends Model {

use SoftDeletes;

protected $table = 'posts';

// ...

}

### 5. What are factories in Laravel?

Factories are a way to put values in fields of a particular model automatically. Like, for testing when we add multiple fake records in the database, we can use factories to generate a class for each model and put data in fields accordingly. Every new laravel application comes with database/factories/UserFactory.php which looks like below:

<?php namespace Database\Factories; use App\Models\User; use Illuminate\Database\Eloquent\Factories\Factory; use Illuminate\Support\Str; **class** **UserFactory** **extends** **Factory** { /\*\* \* The name of the factory's corresponding model. \* \* **@var** string \*/ protected $model = User::class; /\*\* \* Define the model's default state. \* \* **@return** array \*/ public **function** **definition**() { return [ 'name' => $this->faker->name, 'email' => $this->faker->unique()->safeEmail, 'email\_verified\_at' => now(), 'password' => '$2y$10$92IXUNpkjO0rOQ5byMi.Ye4oKoEa3Ro9llC/.og/at2.uheWG/igi', // password 'remember\_token' => Str::random(10), ]; } }

The above command will create a new factory class for the User model. It is just a class that extends the base Factory class and makes use of the Faker class to generate fake data for each column. With the combination of factory and seeders, we can easily add fake data into the database for testing purposes.

### 6. What are seeders in Laravel?

Seeders in Laravel are used to put data in the database tables automatically. After running migrations to create the tables, we can run `php artisan db:seed` to run the seeder to populate the database tables.

We can create a new Seeder using the below artisan command:

It will create a new Seeder like below:

<?php use App\Models\Auth\User; use Illuminate\Database\Eloquent\Model; use Illuminate\Database\Seeder; **class** **UserTableSeeder** **extends** **Seeder** { /\*\* \* Run the database seeds. \*/ public **function** **run**() { factory(User::class, 10)->create(); } }

The run() method in the above code snippet will create 10 new users using the User factory.

Factories will be explained in the next question.

7. What are the default route files in Laravel?

Below are the four default route files in the routes folder in Laravel:

* web.php - For registering web routes.
* api.php - For registering API routes.
* console.php - For registering closure-based console commands.
* channel.php - For registering all your event broadcasting channels that your application supports.

### 8. How to put Laravel applications in maintenance mode?

Maintenance mode is used to put a maintenance page to customers and under the hood, we can do software updates, bug fixes, etc. Laravel applications can be put into maintenance mode using the below command:

php artisan down

And can put the application again on live using the below command:

php artisan up

Also, it is possible to access the website in maintenance mode by whitelisting particular IPs.

### **9. Can we use Laravel for Full Stack Development (Frontend + Backend)?**

Laravel is the best choice to make progressive, scalable full-stack web applications. Full-stack web applications can have a backend in laravel and the frontend can be made using blade files or SPAs using Vue.js as it is provided by default. But it can also be used to just provide rest APIs to a SPA application.

Hence, Laravel can be used to make full-stack applications or just the backend APIs only.

### **10. How to define environment variables in Laravel?**

The environment variables can be defined in the .env file in the project directory. A brand new laravel application comes with a .env.example and while installing we copy this file and rename it to .env and all the environment variables will be defined here.

Some of the examples of environment variables are APP\_ENV, DB\_HOST, DB\_PORT, etc.

### **11. What is an artisan?**

Artisan is the command-line tool for Laravel to help the developer build the application. You can enter the below command to get all the available commands:

PHP artisan list: Artisan command can help in creating the files using the make command. Some of the useful make commands are listed below:

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

### **12. What are available databases supported by Laravel?**

The supported databases in laravel are:

* PostgreSQL
* SQL Server
* SQLite
* MySQL

### **13. 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.

### **14. Define Composer.**

Composer is the package manager for the framework. It helps in adding new packages from the huge community into your laravel application.

For example, one of the most used packages for authentication will be Passport, for including that into your project, you can run the below command on your terminal:

composer requires laravel/passport

It generates a file(composer.json) in your project directory to keep track of all your packages. A default composer.json file of your laravel project will look something like below:

{

"name": "laravel/laravel",

"type": "project",

"description": "The Laravel Framework.",

"keywords": [

"framework",

"laravel"

],

"license": "MIT",

"require": {

"php": "^7.3|^8.0",

"fideloper/proxy": "^4.4",

"fruitcake/laravel-cors": "^2.0",

"guzzlehttp/guzzle": "^7.0.1",

"laravel/framework": "^8.12",

"laravel/tinker": "^2.5"

},

"require-dev": {

"facade/ignition": "^2.5",

"fakerphp/faker": "^1.9.1",

"laravel/sail": "^1.0.1",

"mockery/mockery": "^1.4.2",

"nunomaduro/collision": "^5.0",

"phpunit/phpunit": "^9.3.3"

}

}

The “require” and “require-dev” keys in composer.json specify production and dev packages and their version constraints respectively.

## Advanced Laravel Interview Questions

### 1. What are Relationships in Laravel?

Relationships in Laravel are a way to define relations between different models in the applications. It is the same as relations in relational databases.

Different relationships available in Laravel are:

* One to One
* One to Many
* Many to Many
* Has One Through
* Has Many Through
* One to One (Polymorphic)
* One to Many (Polymorphic)
* Many to Many (Polymorphic)

Relationships are defined as a method on the model class. An example of One to One relation is shown below.

<?php

**namespace** **App**\**Models**;

**use** **Illuminate**\**Database**\**Eloquent**\**Model**;

**class** **User** **extends** **Model**

{

/\*\*

\* Get the phone associated with the user.

\*/

**public** **function** **phone**()

{

**return** **$this**->hasOne(Phone::class);

}

}

The above method phone on the User model can be called like : `$user->phone` or `$user->phone()->where(...)->get()`.

We can also define One to Many relationships like below:

<?php

**namespace** **App**\**Models**;

**use** **Illuminate**\**Database**\**Eloquent**\**Model**;

**class** **User** **extends** **Model**

{

/\*\*

\* Get the addresses for the User.

\*/

**public** **function** **addresses**()

{

**return** **$this**->hasMany(Address::class);

}

}

Since a user can have multiple addresses, we can define a One to Many relations between the User and Address model. Now if we call `$user->addresses`, eloquent will make the join between tables and it will return the result.

### **2. What is Eloquent in Laravel?**

Eloquent is the ORM used to interact with the database using Model classes. It gives handy methods on class objects to make a query on the database.

It can directly be used to retrieve data from any table and run any raw query. But in conjunction with Models, we can make use of its various methods and also make use of relationships and attributes defined on the model.

Some examples of using the Eloquent are below:

* `DB::table(‘users’)->get()`
* `User::all()`
* `User::where(‘name’, ‘=’, ‘Eloquent’)->get()`

### **3. What is throttling and how to implement it in Laravel?**

Throttling is a process to rate-limit requests from a particular IP. This can be used to prevent DDOS attacks as well. For throttling, Laravel provides a middleware that can be applied to routes and it can be added to the global middlewares list as well to execute that middleware for each request.

Here’s how you can add it to a particular route:

Route::middleware('auth:api', 'throttle:60,1')->group(function () {

Route::get('/user', function () {

//

});

});

This will enable the /user route to be accessed by a particular user from a particular IP only 60 times in a minute.

### **4. 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.

The below image could help you understand why Facades are used for:

### 5. 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.

For creating a new Event in laravel, we can call below artisan command:

php artisan make:event UserLoggedIn

This will create a new event class like below:

<?php

**namespace** **App**\**Events**;

**use** **App**\**Models**\**User**;

**use** **Illuminate**\**Broadcasting**\**InteractsWithSockets**;

**use** **Illuminate**\**Foundation**\**Events**\**Dispatchable**;

**use** **Illuminate**\**Queue**\**SerializesModels**;

**class** **UserLoggedIn**

{

**use** **Dispatchable**, **InteractsWithSockets**, **SerializesModels**;

/\*\*

\* The user instance.

\*

\* **@var** \App\Models\User

\*/

**public** $user;

/\*\*

\* Create a new event instance.

\*

\* **@param** \App\Models\User $user

\* **@return** void

\*/

**public** **function** **\_\_construct**(User $user)

{

**$this**->user = $user;

}

}

For this event to work, we need to create a listener as well. We can create a listener like this:

php artisan make:listener SetLogInFile --event=UserLoggedIn

The below resultant listener class will be responsible to handle when the UserLoggedIn event is triggered.

use App\Events\UserLoggedIn;

class SetLogInFile

{

/\*\*

\* Handle the given event.

\*

\* @param \App\Events\UserLoggedIn

\* @return void

\*/

public function handle(UserLoggedIn $event)

{

//

}

}

### **6. Explain logging in Laravel?**

Laravel Logging is a way to log information that is happening inside an application. Laravel provides different channels for logging like file and slack. Log messages can be written on to multiple channels at once as well.

We can configure the channel to be used for logging in to our environment file or in the config file at config/logging.php.

### **7. 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.

### **8. What are Requests in Laravel?**

Requests in Laravel are a way to interact with incoming HTTP requests along with sessions, cookies, and even files if submitted with the request.

The class responsible for doing this is Illuminate\Http\Request.

When any request is submitted to a laravel route, it goes through to the controller method, and with the help of dependency Injection, the request object is available within the method. We can do all kinds of things with the request like validating or authorizing the request, etc.

### **9. How to do request validation in Laravel?**

Request validation in laravel can be done with the controller method or we can create a request validation class that holds the rules of validation and the error messages associated with it.

One example of it can be seen below:

/\*\*

\* Store a new blog post.

\*

\* @param \Illuminate\Http\Request $request

\* @return \Illuminate\Http\Response

\*/

public function store(Request $request)

{

$validated = $request->validate([

'title' => 'required|unique:posts|max:255',

'body' => 'required',

]);

// The blog post is valid...

}

### **10. What is a Service Container in Laravel?**

Service Container or IoC in laravel is responsible for managing class dependencies meaning not every file needs to be injected in class manually but is done by the Service Container automatically. Service Container is mainly used in injecting class in controllers like Request object is injected. We can also inject a Model based on id in route binding.

For example, a route like below:

Route::get('/profile/{id}', 'UserController@profile');

With the controller like below.

public function profile(Request $request, User $id)

{

//

}

In the UserController profile method, the reason we can get the User model as a parameter is because of Service Container as the IoC resolves all the dependencies in all the controllers while booting the server. This process is also called route-model binding.

### **11. 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.

If we are creating our application, we can register our facades in provider classes.

### **12. What is the register and boot method in the Service Provider class?**

The register method in the Service Provider class is used to bind classes or services to the Service Container. It should not be used to access any other functionality or classes from the application as the service you are accessing may not have loaded yet into the container.

The boot method runs after all the dependencies have been included in the container and now we can access any functionality in the boot method. Like you can create routes, create a view composer, etc in the boot method.

### **13. How to define routes in Laravel?**

Laravel Routes are defined in the routes file in routes/web.php for web application routes. Routes can be defined using Illuminate\Support\Facades\Route and calling its static methods such as to get, post, put, delete, etc.

use Illuminate\Support\Facades\Route;

Route::get('/home', function () {

return 'Welcome to Home Sweet Home';

});

A typical closure route looks like the above, where we provide the URI and the closure function to execute when that route is accessed.

Route::get('/hello', 'HomeController@index');

Another way is like above, we can directly give the controller name and the method to call, this can again be resolved using Service Container.

### **14. What are named routes?**

A named route is a route definition with the name assigned to it. We can then use that name to call the route anywhere else in the application.

Route::get('/hello', 'HomeController@index')->name('index');

This can be accessed in a controller using the following:

return redirect()->route('index');

### **15. What are route groups?**

Route Groups in laravel is used when we need to group route attributes like middlewares, prefixes, etc. we use route groups. It saves us a headache to put each attribute to each route.

Syntax:

Route::middleware(['throttleMiddleware'])->group(function () {

Route::get('/', function () {

// Uses throttleMiddleware

});

Route::get('/user/profile', function () {

// Uses throttleMiddleware

});

});

### **16. What is Middleware and how to create one in Laravel?**

Middleware gives developers the ability to inspect and filter incoming HTTP requests of our application. One such middleware that ships with laravel are the authentication middleware which checks if the user is authenticated and if the user is authenticated it will go further in the application otherwise it will throw the user back to the login screen.

We can always create a new middleware for our purposes. For creating a new middleware we can use the below artisan command:

php artisan make:middleware CheckFileIsNotTooLarge

The above command will create a new middleware file in the app/Http/Middleware folder.

### **17. How to create a route for resources in laravel?**

For creating a resource route we can use the below command:

Route::resource('blogs', BlogController::class);

This will create routes for six actions index, create, store, show, edit, update and delete.

### **18. What is dependency Injection in Laravel?**

The Laravel Service Container or IoC resolves all of the dependencies in all controllers. So we can type-hint any dependency in controller methods or constructors. The dependency in methods will be resolved and injected in the method, this injection of resolved classes is called dependency Injection.

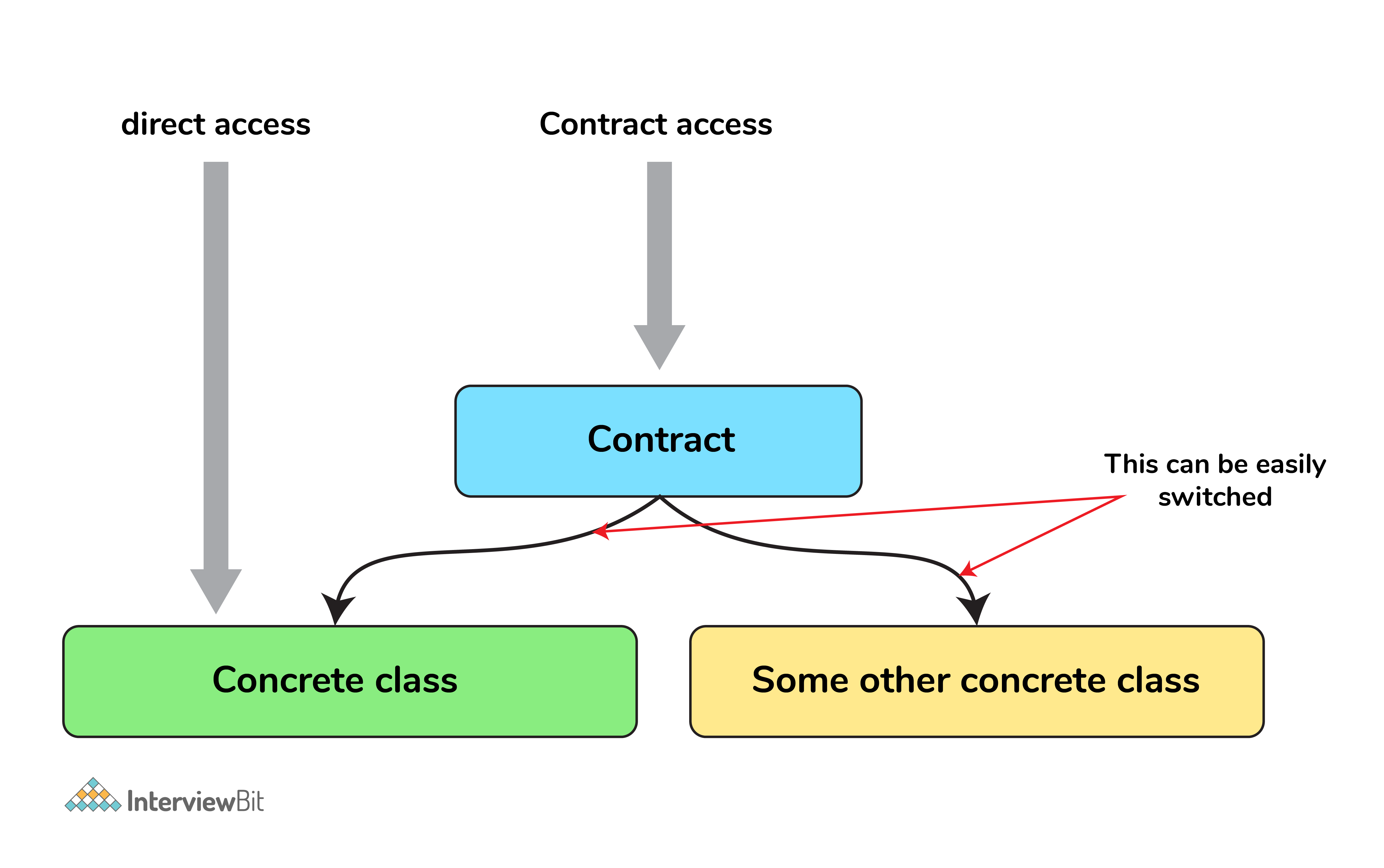
### **19. What are collections?**

Collections in laravel are a wrapper over an array of data in Laravel. All of the responses from Eloquent ORM when we query data from the database are collections (Array of data records).

Collections give us handy methods over them to easily work with the data like looping over data or doing some operation on it.

### **20. What are contracts?**

Laravel Contracts are a set of interfaces with implementation methods to complete the core tasks of Laravel.



Few examples of contracts in Laravel are Queue and Mailer. Queue contract has an implementation of Queuing jobs while Mailer contract has an implementation to send emails.

### **21. 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.

### **22. What are accessors and mutators?**

Accessors are a way to retrieve data from eloquent after doing some operation on the retrieved fields from the database. For example, if we need to combine the first and last names of users but we have two fields in the database, but we want whenever we fetch data from eloquent queries these names need to be combined.

We can do that by creating an accessor like below:

public function getFullNameAttribute()

{

return $this->first\_name . " " . $this->last\_name;

}

What the above code will do is it will give another attribute(full\_name) in the collection of the model, so if we need the combined name we can call it like this: `$user->full\_name`. Mutators are a way to do some operations on a particular field before saving it to the database.

For example, if we wanted the first name to be capitalized before saving it to the database, we can create something like the below:

public function setFirstNameAttribute($value)

{

$this->attributes[‘first\_name’] = strtoupper($value);

}

So, whenever we are setting this field to be anything:

$user->first\_name = Input::get('first\_name');

$user->save();

It will change the first\_name to be capitalized and it will save to the database.

### **Laravel Basic Interview Questions**

#### **1. What is Laravel and why is it used?**

Ans: Laravel is a popular PHP framework designed for building modern web applications using the MVC (Model-View-Controller) architecture. It offers features like routing, authentication, session management, and caching, which streamline the development process. Developers choose Laravel for its elegant syntax and robust ecosystem, which includes tools such as Eloquent ORM, Blade templating, and Laravel Mix. Laravel promotes clean and maintainable code, which makes it ideal for both small projects and large enterprise applications. Its community and regular updates make it a top choice for developers worldwide.

#### **2. What is the role of Artisan in Laravel?**

Ans: Artisan is Laravel’s built-in command-line interface (CLI) that automates common development tasks. It simplifies actions such as generating controllers, models, migrations, and seeders through simple commands. Artisan can also be used to run tests, clear caches, and manage database migrations, making it a powerful tool for developers. Beyond built-in commands, developers can also create custom commands, schedule tasks, and interact with the application’s database via Artisan. By leveraging Artisan, developers can improve productivity and ensure consistency across the project.

#### **3. Explain Middleware in Laravel.**

Ans: Middleware in Laravel serves as a filtering mechanism that sits between the incoming HTTP request and the application’s response. It provides a way to perform tasks like authentication, logging, and CORS handling before the request reaches the controller. Laravel comes with several built-in middleware, such as CSRF protection and session management. Middleware can be assigned globally, or it can be assigned to specific routes, allowing for flexible request handling. Developers can also create custom middleware to address specific requirements, improving application security and maintainability.

#### **4. What is Eloquent ORM?**

Ans: Eloquent ORM (Object-Relational Mapping) is Laravel’s database interaction layer that simplifies working with databases. It allows developers to define models that correspond to database tables and interact with them using expressive PHP syntax instead of raw SQL queries. Eloquent supports all common database operations, such as creating, reading, updating, and deleting records, and it also simplifies working with relationships like one-to-one, one-to-many, and many-to-many. Eloquent also supports advanced features like eager loading, query scopes, and custom attributes, making it a powerful tool for database management in Laravel applications.

### **Laravel Interview Questions for 5 Years Experience**

#### **5. How does Laravel handle database migrations?**

Ans: Laravel provides a simple and effective way to manage and version control database schema changes through migrations. Migrations are PHP classes that define changes to the database schema, such as creating or modifying tables and columns. Using Artisan commands, developers can apply migrations to update the database schema or roll back changes. This ensures consistency across different environments (development, staging, production) by storing migration history in the database. Laravel’s migrations also support database seeding, which allows you to populate your tables with sample data for testing purposes.

#### **6. What is the use of Service Providers in Laravel?**

Ans: Service providers are a fundamental part of Laravel’s service container and the bootstrapping process. They are responsible for registering services, event listeners, middleware, and bindings in the application’s service container. Every service in Laravel is registered through a service provider, including core services like routing, authentication, and caching. Developers can also create custom service providers to encapsulate application-specific logic, allowing for cleaner and more modular code. Service providers help in managing dependencies and ensure that services are only loaded when needed, optimizing the performance of the application.

#### **7. What are Laravel Facades?**

Ans: Laravel facades provide a simple and expressive way to interact with services in the service container. Facades act as “static proxies” to underlying classes, offering a familiar syntax while maintaining the benefits of dependency injection and service container resolution. For example, Cache::put() is a facade that interacts with Laravel’s caching service. Facades simplify access to core services, such as database connections, email, and file storage, making the code cleaner and more readable. Behind the scenes, facades resolve the actual service instances from the container, ensuring that dependencies are managed efficiently.

#### **8. What is CSRF protection in Laravel?**

Ans: CSRF (Cross-Site Request Forgery) protection is a security feature built into Laravel to prevent unauthorized users from performing actions on behalf of an authenticated user. Laravel automatically generates a CSRF token for every session, which must be included in every POST, PUT, PATCH, or DELETE request. This token is then validated to ensure that the request originated from the user’s session, preventing malicious attacks. The @csrf Blade directive is used to include the token in HTML forms, making CSRF protection easy to implement without additional configuration.

#### **9. How does Laravel support RESTful API development?**

Ans: Laravel offers a comprehensive suite of tools for building RESTful APIs with ease. It provides features like resource controllers, API routes, and automatic JSON responses, which make structuring APIs straightforward. Laravel supports token-based authentication via Laravel Passport and Sanctum, ensuring secure access to APIs. API responses can be formatted using Laravel’s built-in response macros, and data can be transformed using Eloquent API resources. The framework also supports rate limiting, versioning, and API testing, enabling developers to build robust, scalable, and secure APIs quickly.

#### **10. What is the difference between require and include in Laravel?**

Ans: Both require and include are used in PHP to include external files, but they behave differently when the file cannot be found. require results in a fatal error, halting the execution of the script if the file is missing, whereas include issues a warning but allows the script to continue running. Although not exclusive to Laravel, understanding these differences is important when working with legacy code or managing file inclusions. In Laravel, however, autoloading via Composer and Laravel's service container is typically preferred over manually including files, ensuring a cleaner and more efficient application structure.

### **PhP Laravel Interview Questions**

#### **11. What are Laravel Blade Templates?**

Ans: Blade is Laravel's lightweight templating engine that allows developers to separate logic from views in a clean, intuitive way. It provides useful features such as template inheritance, sections, and control structures like loops and conditionals. Blade templates compile down to plain PHP code, which makes them fast and efficient. It also allows for easy integration with Laravel’s route and controller systems, enabling the creation of dynamic, reusable views with minimal overhead.

#### **12. What is Dependency Injection in Laravel?**

Ans: Dependency Injection (DI) is a design pattern that Laravel uses to manage class dependencies through the service container. Instead of creating dependencies manually inside a class, Laravel’s service container automatically injects them when the class is resolved. This promotes loose coupling, improves code maintainability, and makes it easier to test classes. DI is often used for injecting services like database connections, mailers, and repositories into controllers and other classes.

#### **13. What is the purpose of the routes/web.php file in Laravel?**

Ans: The routes/web.php file in Laravel is where the application’s web routes are defined. These routes are responsible for handling HTTP requests that are associated with views or HTML responses. Routes in this file can be tied to controllers, closures, or middleware. This is the default location for defining routes that are intended to render webpages, and it typically works alongside routes/api.php for API-specific routes.

#### **14. How can you handle file uploads in Laravel?**

Ans: Laravel provides a simple API for handling file uploads, allowing you to easily store files on the local disk or cloud storage. You can use the store or storeAs methods to upload files to a specific disk, defined in the config/filesystems.php configuration file. Laravel supports various file validation rules, such as file type, size, and mime type, and also offers the ability to retrieve uploaded files via the File or Storage facade for further processing or storage.

### **Laravel Interview Questions for 2 Year Experience**

#### **15. How does Laravel handle authentication?**

Ans: Laravel provides a simple and flexible authentication system out of the box. It includes features for user login, registration, and password reset, all managed via controllers and routes. Authentication is handled using Laravel’s built-in Auth facade, and it supports multiple methods of user verification, including traditional session-based authentication, API token authentication via Laravel Passport or Sanctum, and two-factor authentication. The system can be easily customized to meet the unique needs of an application.

#### **16. What are Laravel Jobs and Queues?**

Ans: Laravel’s job and queue system allows developers to defer the processing of time-consuming tasks, such as sending emails or processing images, to be handled later. Jobs are simply classes that represent specific tasks, and these tasks are placed onto queues that are processed asynchronously. Laravel supports multiple queue drivers, including database, Redis, and Amazon SQS. Using queues ensures that your application remains responsive by offloading heavy operations, and it can significantly improve the performance of large-scale applications.

#### **17. What is the difference between hasOne, hasMany, and belongsTo in Eloquent relationships?**

Ans: In Eloquent, relationships define how models interact with each other in a database. hasOne defines a one-to-one relationship, where one model has a related model. hasMany defines a one-to-many relationship, where one model can have multiple related models. belongsTo, on the other hand, defines an inverse one-to-one or one-to-many relationship, where the current model is the child in the relationship. These relationships are used to query related data easily and efficiently in Laravel.

#### **18. What is Laravel’s config directory used for?**

#### Ans: The config directory in Laravel contains all the configuration files for the application, including database connections, mail services, session handling, caching, and more. Each configuration file is an array of settings that can be easily modified depending on the environment (local, production, etc.). The config directory allows developers to manage application settings in a centralized, organized way, promoting cleaner code and making it easier to maintain and update application configurations.

#### **19. What are Events and Listeners in Laravel?**

Ans: Events and Listeners in Laravel provide a simple observer pattern implementation. An event is an action or occurrence in your application that can trigger certain behaviors. For example, an event could be triggered after a user logs in or when a new post is created. A listener responds to these events and handles the specific logic, such as sending an email or updating related models. Laravel’s event system allows for decoupled, modular code that makes it easy to extend the application’s functionality without modifying core logic.

#### **20. What is the purpose of the artisan migrate:rollback command?**

Ans: The artisan migrate:rollback command is used to reverse the most recent database migration in Laravel. When you run migrations to modify your database schema, you can use this command to roll back those changes, which is useful when testing or fixing issues. Laravel allows you to specify the number of steps to roll back using the --step option, enabling partial rollbacks if necessary. This command helps ensure that database changes can be easily reversed without manually managing the schema.

Laravel Interview Questions for 3 Year Experience

#### **21. What is the purpose of Laravel’s session?**

Ans: Laravel’s session is used to store user data across requests. It can hold data like user preferences, login status, or flash messages temporarily. Laravel provides an easy-to-use API for managing session data, which can be stored in various backends such as files, cookies, or a database. The session allows web applications to remember information about a user between requests, ensuring a seamless experience like keeping users logged in or displaying previous search results.

#### **22. What is the difference between get() and first() in Eloquent?**

Ans: The get() method retrieves multiple records from the database as a collection, while first() retrieves only the first result of a query. If no records are found, first() returns null, whereas get() returns an empty collection. Use get() when you expect multiple results and first() when you only need a single record. Both methods are used for querying Eloquent models, but their return types differ, making them suitable for different use cases.

#### **23. How does Laravel handle caching?**

Ans: Laravel’s caching system provides a simple and unified API for interacting with different caching backends, such as file, database, Redis, or Memcached. Caching is used to store expensive or frequently accessed data temporarily to improve performance. Laravel offers a variety of caching methods, such as storing data with Cache::put() and retrieving it with Cache::get(). The framework also supports cache tags and automatic cache clearing, making it easy to manage cached data efficiently and boost application speed.

#### **24. What is the purpose of service container in Laravel?**

Ans: The service container in Laravel is a powerful tool for managing class dependencies and performing dependency injection. It allows for the automatic resolution of class dependencies, making it easier to manage complex applications. The container stores services and can inject them into controllers, jobs, events, or other classes at runtime. By leveraging the service container, Laravel promotes loose coupling between components, making the application more modular and easier to maintain.

#### **25. What are Laravel Collections?**

Ans: Laravel Collections are an enhanced version of PHP arrays that offer a variety of methods for working with data sets. Collections allow developers to perform common operations like filtering, mapping, and reducing arrays in a fluent and expressive manner. They can be used to handle query results, or any other data array, and provide methods like filter(), map(), pluck(), and sortBy(). Collections simplify data manipulation, improving readability and making code more elegant.

### **Laravel Interview Questions for 10 Year Experienced Professionals**

#### **26. What is the purpose of Laravel’s validation feature?**

Ans: Laravel provides a robust validation system that allows you to easily validate incoming request data. You can use the Validator class or form request validation to ensure that user input meets specific criteria such as required fields, correct data types, and length restrictions. Validation can be done directly in the controller or through custom request classes, making it easy to handle different forms of input validation across the application. Laravel also provides custom validation rules, allowing developers to create tailored rules for specific application needs.

#### **27. How does Laravel handle database relationships?**

Ans: Laravel provides an elegant and powerful ORM (Eloquent) for managing relationships between models. It supports several types of relationships, including one-to-one, one-to-many, many-to-many, and many-to-many polymorphic relationships. By defining relationships in the model classes, developers can easily retrieve related data using methods like hasOne(), hasMany(), and belongsTo(). Laravel also supports eager loading to optimize query performance by loading related data in a single query, reducing the number of database queries and improving efficiency.

#### **28. What is the difference between include and require in Laravel?**

Ans: In PHP, both include and require are used to include files, but the key difference lies in how they handle errors. If the included file cannot be found, include will only generate a warning and continue execution, while require will throw a fatal error and stop the script. While this behavior is common in PHP, Laravel primarily uses its autoloading mechanism with Composer for class loading, making include and require less common in modern Laravel applications.

#### **29. What is Laravel's Queue system?**

Ans: Laravel’s Queue system is a mechanism for handling tasks asynchronously, such as sending emails, processing images, or performing API calls. By pushing jobs onto a queue, Laravel allows tasks to be processed in the background, freeing up the main thread to handle user requests. Laravel supports several queue drivers, including database, Redis, and Amazon SQS. The queue system helps to improve application performance by offloading heavy tasks and ensuring that the application remains responsive, even during resource-intensive operations.

#### **30. What is the difference between Route::get() and Route::post() in Laravel?**

Ans: In Laravel, Route::get() and Route::post() are used to define routes for handling HTTP GET and POST requests, respectively. Route::get() is typically used for retrieving data from the server or displaying resources, while Route::post() is used for sending data to the server, such as when submitting forms or making API requests. GET requests are generally used for reading or retrieving resources, while POST requests are used for creating or updating resources in a Laravel application.

### **Laravel Advanced Interview Questions**

#### **31. What is the difference between composer install and composer update in Laravel?**

Ans: composer install is used to install all the dependencies listed in the composer.json file. It installs the exact versions of dependencies that are locked in the composer.lock file, ensuring the same environment across different machines. On the other hand, composer update updates all the dependencies to their latest compatible versions based on the version constraints in the composer.json file and updates the composer.lock file. Generally, composer install is used when setting up a new project, while composer update is used to update the project’s dependencies.

#### **32. How can you optimize the performance of a Laravel application?**

Ans: Laravel provides several techniques for optimizing performance, such as:

Caching: Utilize caching for database queries, views, and route caching to reduce execution time.  
Database Optimization: Use database indexing, optimized queries, and eager loading to avoid N+1 query problems.  
Route Caching: Use php artisan route:cache to cache all the application routes, reducing response time.  
Config Caching: Use php artisan config:cache to cache configuration files.  
Autoloader Optimization: Use composer dump-autoload -o to optimize Composer’s autoloader performance.  
Queueing: Offload time-consuming tasks to queues (e.g., sending emails, processing images) to improve user experience.

By utilizing these strategies, you can greatly enhance the performance of a Laravel application.

#### **33. What is the use keyword in Laravel?**

Ans: The use keyword in Laravel is used to import classes, traits, or namespaces into the current PHP file. This allows you to use short class names without having to write out their full namespace each time. For example, to use Laravel’s Carbon class for date handling, you would write use Carbon\Carbon; at the top of the file. It’s an essential part of working with PHP’s namespaces and helps keep the code clean and readable.

### **Laravel Interview Questions for Experienced**

#### **34. What is Laravel's built-in Artisan command for database migrations?**

Ans: Laravel provides several Artisan commands to manage database migrations. The most common ones are:

php artisan migrate:  Applies any pending migrations to the database.  
php artisan migrate: rollback: Rolls back the last database migration.  
php artisan migrate:reset:  Rolls back all migrations.  
php artisan migrate:refresh: Rolls back all migrations and re-applies them, useful for testing purposes.  
php artisan migrate:fresh: Drops all tables and re-runs all migrations, used to start with a fresh database state.

These commands help manage and version control database schema changes efficiently.

#### **35. How do you perform database seeding in Laravel?**

Ans: Database seeding is used to populate your database with test or default data. In Laravel, you can create seeders using php artisan make:seeder SeederName, and you define the logic for inserting data inside the run() method. After creating a seeder, you can run it using the php artisan db:seed command, which will populate the database with the defined data. You can also use php artisan migrate:refresh --seed to run migrations and seed data at once.

#### **36. Explain how to use the DB facade in Laravel?**

Ans: The DB facade in Laravel provides a simple and expressive interface for working with the database directly. You can use it to perform raw SQL queries or interact with the database using the query builder. For example:

Raw SQL Queries: DB::select('SELECT \* FROM users WHERE id = ?', [1]);  
Query Builder: DB::table('users')->where('id', 1)->get();

The DB facade is useful when you need more control over SQL execution or need to perform complex database operations outside of Eloquent’s ORM.

### **Laravel Interview Questions for Senior Developer**

### 

#### **37. What is Laravel’s route model binding?**

#### Ans: Route model binding in Laravel provides a convenient way to inject model instances directly into your route callbacks or controllers. Instead of manually fetching a model using the ID from the URL, Laravel automatically resolves the model for you. For example, in a route, you can specify a model binding like this:

php  
Copy  
Edit  
Route::get('user/{user}', function (App\Models\User $user) {  
    return $user;  
});

Laravel will automatically retrieve the User model by ID and inject it into the route, simplifying the code and ensuring that the model is properly resolved.

#### **38. What are Laravel Policies?**

Ans: Policies in Laravel are used to authorize actions in the application. They provide a simple, organized way to manage user permissions based on their roles or other attributes. Policies are typically used for controlling access to models, such as checking if a user can update a post or delete a comment. To create a policy, you can use php artisan make:policy, and then define methods corresponding to different actions (e.g., update, delete). You can then use the authorize method or middleware to check permissions before executing actions.

#### **39. How does Laravel’s Queue system differ from a regular background job system?**

Ans: Laravel’s Queue system is built on the concept of pushing jobs into a queue and processing them asynchronously. It’s more structured and integrated within Laravel’s ecosystem, making it easier to manage, monitor, and scale background tasks. A regular background job system may require more manual configuration and handling, while Laravel’s Queue system is optimized for Laravel applications, supporting various backends like Redis, Beanstalkd, and database, and offering features like delayed jobs, retries, and job chaining.

#### **40. How would you handle multiple database connections in Laravel?**

Ans: Laravel supports multiple database connections through its configuration file config/database.php. To set up multiple connections, you define them in the connections array with different database configurations. You can then specify which connection to use within the application using DB::connection('connection\_name'). Additionally, Eloquent models can be configured to use a specific connection by setting the $connection property. This allows you to work with multiple databases within a single Laravel application, ideal for scenarios like microservices or multi-tenant systems.

## Laravel Basic Interview Questions

Here are some commonly asked basic Laravel framework interview questions and answers.

1. What is Laravel and why is it popular among developers?

Laravel is a PHP framework built to simplify common web development tasks like routing, authentication, sessions, and caching. Developers like Laravel because it has clean syntax, built-in tools, and strong community support. It speeds up development without sacrificing structure or flexibility.

1. How does Laravel handle routing?

Laravel uses a file called routes/web.php for web routes and routes/api.php for API routes. You define routes using expressive methods like Route::get(), Route::post(), etc. Routes can be grouped, named, and protected with middleware, making the application easy to manage.

1. What is the purpose of artisan in Laravel?

Artisan is Laravel’s command-line tool. It helps automate repetitive tasks like creating controllers, running migrations, seeding the database, and clearing caches. For example, php artisan make:controller generates a new controller class in seconds.

1. How do you define a controller in Laravel?

You can create a controller using the Artisan command php artisan make:controller UserController. Then you define methods inside the class for handling requests, like index(), store(), or update(). These methods are linked to routes in web.php or api.php.

1. What is the difference between get(), first(), and find() in Eloquent?

* get() returns a collection of records.
* first() returns only the first result.
* find() looks for a specific record by primary key.

So, User::get() gets all users, User::first() returns the first user, and User::find(1) fetches the user with ID 1.

## Laravel Interview Questions for Freshers

Here is a list of Laravel interview questions and answers for freshers.

1. What are service providers in Laravel?

Service providers are the central place for registering services, bindings, and event listeners. Laravel loads them during the bootstrapping process. They live in the app/Providers folder and include methods like register() and boot() to set up things like custom classes or observers.

1. What is middleware and how is it used?

Middleware acts as a filter for HTTP requests. It is used for tasks like checking authentication, logging, or modifying requests before they hit the controller. You can create middleware using php artisan make:middleware, and apply it in route definitions or globally in Kernel.php.

1. How do you create a migration file in Laravel?

Use the command php artisan make:migration create\_users\_table. This creates a file in the database/migrations directory. You then define the table schema using Laravel’s Schema builder. Migrations are version-controlled and help sync databases across environments.

1. What is the use of .env file?

[See also  Top 25+ Unix Interview Questions and Answers](https://www.hirist.tech/blog/top-25-unix-interview-questions-and-answers/)

The .env file stores environment-specific configuration like database credentials, mail settings, and API keys. Laravel reads these values using env() or via config/ files. It helps keep sensitive data out of the codebase and allows easy switching between environments.

1. How does blade templating work in Laravel?

Blade is Laravel’s built-in templating engine. It lets you write clean views with features like template inheritance (@extends), sections (@section), and control structures like @if, @foreach. Blade compiles templates into plain PHP for fast rendering.

## Laravel Interview Questions for Experienced

Let’s go through some important Laravel interview questions and answers for experienced candidates.

1. How do you use Laravel queues in a real project?

Laravel queues are used to delay time-consuming tasks like sending emails or processing uploads. You push jobs to the queue using dispatch(), and they’re processed by workers running in the background. You can use database, Redis, or SQS as queue drivers, depending on your project needs.

1. What are some ways to optimise Laravel performance?

You can speed up Laravel apps by caching routes and config using Artisan (php artisan route:cache, config:cache), using eager loading to reduce queries, and minimising middleware. Tools like Laravel Octane can also help in high-performance cases. Database indexing and using queues properly also improve performance.

1. How do you manage large Laravel projects?

“In large projects, I use a modular structure with separate folders for each domain or feature. I also rely on service classes, repository patterns, and proper naming conventions. Route grouping, custom traits, and policy-based authorization keep things clean and organized.”

1. How does Laravel handle database transactions?

Laravel offers the DB::transaction() method to wrap operations that need to succeed or fail together. If anything goes wrong inside the closure, all changes are rolled back. You can also manually begin and commit transactions using DB::beginTransaction(), commit(), and rollBack().

1. Can you explain repository pattern in the context of Laravel?

“The repository pattern separates business logic from data access. Instead of calling Eloquent directly in controllers, I create a repository class to handle all model interactions. This makes the code easier to test, maintain, and switch to another data source if needed.”

### Laravel Interview Questions for 1 Year Experience

These interview questions for Laravel are mostly asked to candidates with 1 year of experience.

* What kind of Laravel projects have you worked on so far?
* Tell me about a time you struggled with a bug in Laravel and how you fixed it.
* A user report says their data isn’t saving—how would you troubleshoot this in a Laravel form submission?

### Laravel Interview Questions for 2 Year Experience

If you have 2 years of experience, you might come across such Laravel interview questions.

* How has your understanding of Laravel evolved over the past two years?
* Describe a situation where you had to meet a tight deadline on a Laravel project.
* You are asked to implement a new feature. How would you structure the controller and routes?

### Laravel Interview Questions for 3 Year Experience

Here are some interview questions for Laravel for professionals with three years of experience.

* What Laravel tools or packages do you regularly use?
* Share an experience where you improved the performance of a Laravel app.
* You are migrating a legacy PHP app to Laravel. What’s your approach?

Also Read - [**Top 20 PHP OOPs Interview Questions and Answers**](https://www.hirist.tech/blog/top-20-php-oops-interview-questions-and-answers/)

### Laravel Interview Questions for 4 Year Experience

Candidates with 4 years of experience might face these Laravel interview questions.

* What architecture patterns do you follow when building Laravel applications?
* Have you mentored junior developers in Laravel? How did you do it?
* You need to integrate a third-party API. How would you structure it in a Laravel project?

### Laravel Interview Questions for 5 Year Experience

These interview questions for Laravel are for professional with 5 years of experience.

* How do you stay updated with changes in Laravel?
* Describe a conflict you had during a Laravel project and how you handled it.
* A Laravel app is taking too long to load. What would you check first?

### Laravel Interview Questions for 6 Year Experience

If you have around 6 years of experience, you might be asked such Laravel interview questions.

* What’s your approach to writing scalable Laravel code?
* Tell me about a project where you led the Laravel development end-to-end.
* You are asked to build a multi-tenant Laravel app. How would you go about it?

[See also  Top 45+ Database Testing Interview Questions and Answers](https://www.hirist.tech/blog/top-45-database-testing-interview-questions-and-answers/)

### Laravel Interview Questions for 10 Years Experienced

These interview questions for Laravel are mostly asked to senior professionals with 10 years of experience.

* How have you seen Laravel evolve over the last decade, and how did you adapt?
* Tell me about a time when you had to overhaul an existing Laravel codebase.
* You are tasked with auditing a large Laravel codebase. What are your first steps?

## Laravel Advanced Interview Questions

Here are some advanced PHP Laravel interview questions and answers.

1. How would you handle caching strategies in Laravel for high-traffic applications?

“I use route, view, and config caching with Artisan commands like php artisan config:cache. For data caching, I prefer Redis or Memcached. I cache frequently accessed queries using Cache::remember() and apply tags to manage cache groups. I also avoid caching dynamic content and set appropriate TTLs.”

1. What is Laravel Octane and when should you use it?

“Laravel Octane speeds up applications by serving requests through Swoole or RoadRunner. It keeps the app in memory between requests, which reduces boot time. I use Octane for high-performance apps with many requests per second, especially when working with APIs or real-time services.”

1. How do you implement custom service containers and bindings?

“In AppServiceProvider, I use the register() method to bind classes or interfaces to the container using $this->app->bind(). For example, binding a repository interface to its implementation makes the code loosely coupled and easier to test.”

## Laravel Technical Interview Questions

Let’s cover some important technical interview questions for Laravel.

1. What is the difference between hasOneThrough and hasManyThrough in Eloquent?

hasOneThrough defines a one-to-one relationship across two models, while hasManyThrough defines one-to-many. For example, if a country has many users and users have posts, then Country can access posts via hasManyThrough.

1. How does Laravel handle CSRF protection?

Laravel uses a CSRF token stored in a session and injected into forms via @csrf. When a POST, PUT, PATCH, or DELETE request is made, Laravel compares the token with the session to prevent cross-site attacks.

1. What are the steps to create and use a custom Artisan command?

Use php artisan make:command MyCommand. Then, set a signature and logic inside the handle() method. Register it in Kernel.php under the commands array. Run it using php artisan my:command.

## Laravel Logical Interview Questions

These are some logical Laravel interview questions and answers.

1. How would you avoid N+1 query problems in Eloquent?

“I use eager loading with with() to load relationships ahead of time. For example, Post::with(‘comments’)->get() prevents multiple queries for each comment. I also use load() when I already have the parent model.”

1. How would you validate deeply nested array inputs in a Laravel request?

“I use dot notation or wildcard rules like ‘items.\*.name’ => ‘required|string’ in form request validation. This works well when dealing with arrays of objects from the frontend.”

1. What logic would you use to handle dynamic role-based permissions?

“I define roles and permissions in database tables. Then I use Laravel Gates or Policies to check permissions at runtime. I usually create a hasPermission() method on the User model and check permissions through middleware or inside controllers.”

## Laravel Developer Interview Questions

Here are some commonly asked Laravel interview questions and answers for developers.

1. What is the difference between Auth::user() and auth()->user()?

Both return the authenticated user. Auth::user() is the facade version, while auth()->user() is a helper function. The helper is shorter and preferred in recent Laravel versions, but both are valid.

1. How do you handle file uploads securely in Laravel?

“I validate files using the mimes or file rule and restrict file size. Then I store uploads using store() or storeAs() in Laravel’s storage system. I never trust client file names and always store files outside the public directory unless explicitly needed.”

1. How would you debug a failed job in Laravel queues?

“First, I check the failed\_jobs table to see the error message. I also log errors inside the job’s failed() method. Laravel also supports retrying failed jobs with php artisan queue:retry. I use Horizon for monitoring when Redis is the queue driver.”

## Laravel Interview Questions for Senior Developer

If you are a senior developer, you might come across these interview questions for Laravel.

1. How would you structure a Laravel project for a team of 10 developers?

[See also  Top 35+ Rest Assured Interview Questions for Answers](https://www.hirist.tech/blog/top-35-rest-assured-interview-questions-for-answers/)

“I use a domain-based folder structure and break logic into service classes and repositories. I enforce strict naming conventions and use contracts for shared logic. I also document APIs and business rules in a shared workspace like Notion or Wiki.”

1. What’s your process for setting up CI/CD in a Laravel application?

“I use GitHub Actions or GitLab CI to automate testing, linting, and deployments. The pipeline runs tests on every push and deploys to staging or production if successful. I handle .env management and database migrations as part of the workflow. For deployment, I prefer Laravel Forge or Ploi, as they simplify server setup and work well with Laravel apps.”

1. How do you manage environment-specific configurations across staging and production?

“Each environment has its own .env file. I never commit them to the repo. I use environment variables for API keys, DB credentials, and queue settings. Laravel reads from .env or system environment variables at runtime.”

## Laravel API Interview Questions

Here are some frequently-asked Laravel interview questions and answers on API.

1. How do you build RESTful APIs using Laravel?

“I define API routes in routes/api.php and return JSON responses from controllers. I use form request classes for validation and Resource classes for formatting responses. For versioning, I prefix routes like v1/users.”

1. What tools or packages do you use to document Laravel APIs?

“I use tools like Laravel Swagger, Scribe, or Postman collections. They help generate API documentation from annotations or routes. Some even support auto-generation from request/response structures.”

1. How do you handle API authentication and rate limiting in Laravel?

“For authentication, I use Laravel Sanctum or Passport. Rate limiting is managed using throttle middleware, and can be configured in RouteServiceProvider or directly on routes using ->middleware(‘throttle:60,1’).”

Also Read - [**Top 50+ REST API Interview Questions and Answers**](https://www.hirist.tech/blog/top-20-rest-api-interview-questions-and-answers/)

## Laravel Coding Interview Questions

These are some common coding interview questions for Laravel.

1. Write a query to fetch the latest 5 posts created by a user with ID 10.

$posts = Post::where(‘user\_id’, 10)

             ->orderBy(‘created\_at’, ‘desc’)

             ->take(5)

             ->get();

1. Create a route that returns a JSON response with user details and their posts.

use App\Models\User;

Route::get(‘/user/{id}’, function ($id) {

    $user = User::with(‘posts’)->findOrFail($id);

    return response()->json($user);

});

1. Create a validation rule for an input that looks like:

“items”: [

  { “product\_id”: 1, “quantity”: 2 },

  { “product\_id”: 5, “quantity”: 1 }

]

This kind of input is commonly used in cart submissions or bulk order forms. Here’s how you’d validate it in Laravel:

$request->validate([

    ‘items’ => ‘required|array|min:1’,

    ‘items.\*.product\_id’ => ‘required|integer|exists:products,id’,

    ‘items.\*.quantity’ => ‘required|integer|min:1’

]);

This checks that each item has a valid product\_id and a quantity of at least 1.

1. Use Eloquent to get all users who haven’t posted anything.

$users = User::doesntHave(‘posts’)->get();

Build a custom validation rule to check if a date is in the future.

$request->validate([

    ‘event\_date’ => [‘required’, ‘date’, function ($attribute, $value, $fail) {

        if (strtotime($value) <= time()) {

            $fail(‘The event date must be in the future.’);

        }

    }]

]);

### Interview Questions for PHP:

#### \*\*: What is method overriding in PHP ?

Method overriding in PHP is a fundamental concept of OOP, allowing a subclass to provide it's own implementation for a method that is inherited from the parent class(superclass). Here are the rules for method overriding:

* Have to have same method name.
* Have to have same number of parameters.
* Have to have same types of parameters.

Here is an example of how method overriding works in PHP:

class Animal {

public function speak() {

echo 'Animal speak';

}

}

class Dog extends Animal {

/\*\*

\* Here is the method overriding happens.

\* You cannot have any params.

\*/

public function speak() {

echo 'Dog Barks';

}

}

#### \*\*: What is method overloading in PHP ?

Method overloading allows you define multiple methods with the same name in a class but with different parameter list. This means you can create different versions of a method that performs different actions depending on the parameters passed to the method. However, unlike other programming languages, PHP does not natively support method overloading by changing the number or types of arguments. Instead you can manually handle the parameters passed to method to handle the method overloading in PHP.

Here is an example:

class Math {

public static function sum(int $a): int {

if(func\_num\_args() == 1) {

return $a;

}

// or whatever you want to do with the parameters

return array\_sum(func\_get\_args());

}

}

#### \*\*: What is PEAR in PHP?

PEAR in PHP stands for "PHP Extension and Application Repository." It is a framework and distribution system for reusable PHP components, including libraries, packages, and extensions.

#### 1: Is PHP a case sensitive language?

Yes, PHP is a case-sensitive language, meaning that it distinguishes between uppercase and lowercase letters in variable names, function names, and other identifiers.

#### 2: What is the meaning of ‘escaping to PHP’?

"Escaping to PHP" refers to the process of embedding PHP code within an HTML document using PHP tags (). This allows you to mix PHP code with HTML to create dynamic web pages.

#### 3: What are the different types of PHP variables?

In PHP, variables can be categorized into different types based on their data types or what they store. The main types of PHP variables are:

1. Scalar Variables: - Integer: Stores whole numbers, e.g., $num = 42. - Float (or Double): Stores floating-point numbers (numbers with decimal points), e.g., $pi = 3.14. - String: Stores text or characters, e.g., $name = "John". - Boolean: Stores either true or false, e.g., $is\_active = true.
2. Compound Variables: - Array: Stores multiple values in an ordered list, e.g., $fruits = array("apple", "banana", "cherry"). - Object: Stores instances of user-defined classes, e.g., $car = new Car().
3. Special Variables: - Resource: A special type used to hold references to external resources like database connections, file handles, etc. - NULL: Represents a variable with no value or an undefined value, e.g., $empty\_var = null.
4. Superglobals: - Variables that are predefined in PHP and are accessible from any part of the script. These include $\_GET, $\_POST, $\_REQUEST, $\_SESSION, $\_COOKIE, $\_SERVER, $\_ENV, and more. They are used to handle HTTP requests, manage sessions, and access server-related information.
5. Variables with Special Prefixes: - These variables have special meanings:
   * $GLOBALS: A superglobal that allows access to variables in the global scope.
   * $\_SESSION: A superglobal used for session variables.
   * $\_POST and $\_GET: Superglobals for handling form data sent via HTTP POST and GET methods.
   * $\_COOKIE: A superglobal for handling cookies.
   * $\_SERVER: A superglobal containing server-related information.

These are the main types of PHP variables. Each type serves a specific purpose and is used in different situations within PHP scripts.

#### 4: What are the rules for naming a PHP variable?

Rules for naming a PHP variable:

* Variable names must start with a letter or underscore.
* Variable names can only contain letters, numbers, and underscores.
* Variable names are case-sensitive.
* Variable names should not be PHP reserved words.

#### 5: What are the rules to determine the “truth” of any value which is not already of the Boolean type?

In PHP, values are considered "truthy" if they are not:

* Empty strings
* Zero (0 or 0.0)
* NULL
* False (false boolean)
* An empty array

#### 6: What is NULL?

NULL is a special value in PHP that represents the absence of a value or a variable that has not been assigned any value.

#### 7: How do you define a constant in PHP?

To define a constant in PHP, you can use the define() function. For example:

define("MY\_CONSTANT", 42);

#### 8: What is the purpose of constant() function?

The constant() function in PHP is used to retrieve the value of a constant by providing its name as a string. For example, constant("MY\_CONSTANT") would return the value of the constant named "MY\_CONSTANT."

#### 9: What are the differences between PHP constants and variables?

Differences between PHP constants and variables:

* Constants are defined using define() and cannot be changed after definition, while variables can change their values.
* Constants are typically used for values that should not be altered during the script's execution, like configuration settings.
* Variables are used for data that can change during the script's execution.

#### 10: Name some of the constants in PHP and their purpose.

Some common PHP constants and their purposes:

* PHP\_VERSION: Holds the current PHP version
* PHP\_OS: Represents the name of the operating system.
* DIRECTORY\_SEPARATOR: Contains the directory separator for the current OS.
* E\_ERROR, E\_WARNING, etc.: Error reporting constants for different error levels.

#### 11: What is the difference between PHP4 and PHP5?

PHP5 introduced significant improvements and new features compared to PHP4, including:

* Enhanced object-oriented programming support.
* Introduction of visibility keywords (public, private, protected).
* Improved support for exceptions.
* Better performance and memory management
* Support for new data types, like the SimpleXML extension for parsing XML.

#### 12: What is the meaning of a final class and a final method?

In PHP, a final class cannot be extended or subclassed, and a final method within a class cannot be overridden in any subclass. It's used to prevent further modification or extension of the class or method.

#### 13: How can you compare objects in PHP?

To compare objects in PHP, you can use the == operator to check if they are equal in terms of properties and values, or the === operator to check if they are the same instance. The == operator compares values, while the === operator compares both values and data types.

#### 14: How can PHP and Javascript interact?

PHP and JavaScript can interact through AJAX (Asynchronous JavaScript and XML) requests, which allow JavaScript to make requests to a PHP script on the server, retrieve data, and update the webpage without a full page refresh. They can also communicate through cookies, sessions, and embedding JavaScript code within PHP-generated HTML.

#### 15: What are the data types in PHP?

Common data types in PHP include:

* Integer (int)
* Float (float)
* String (string)
* Boolean (bool)
* Array
* Object
* NULL
* Resource
* Callable
* Iterable (introduced in PHP 7.1)

#### 16: What are constructor and destructor in PHP?

In PHP, a constructor is a special method called when an object of a class is created. It is used to initialize object properties. A destructor is a method that is called when an object is no longer referenced or when the script ends. It can be used to perform cleanup tasks.

#### 17: What are include() and require() functions?

include() and require() are PHP functions used to include external files in a script. They include and evaluate the specified file. If the file is not found or has an error, require() will produce a fatal error, while include() will only produce a warning.

#### 18: What is the main difference between require() and require\_once()?

The main difference between require() and require\_once() is that if the same file is included multiple times in a script, require\_once() will ensure that it is included only once, preventing duplication.

#### 19: What are different types of errors available in Php ?

Different types of errors in PHP:

* Parse errors (syntax errors)
* Fatal errors
* Warnings
* Notices
* Deprecated errors
* User-generated errors using trigger\_error()

#### 20: What are the different types of Array in PHP?

Different types of arrays in PHP:

* Indexed arrays (numeric keys)
* Associative arrays (key-value pairs)
* Multidimensional arrays (arrays within arrays)
* Constant arrays (created with define())
* Typed arrays (introduced in PHP 8.0, with predefined data types for array elements)

#### 21: What is the difference between “echo” and “print” in PHP?

In PHP, both echo and print are used to output text or data to the screen. However, there are some differences:

* echo is a language construct, while print is a function. This means that echo is slightly faster and more versatile in terms of what it can output.
* echo can output multiple values separated by commas, while print can only output one value and always returns 1.
* echo does not have a return value, while print returns 1.
* echo is often used for simple output, while print is less commonly used in modern PHP code.

#### 22: Name some of the functions in PHP.

PHP has a wide range of built-in functions. Some commonly used functions include strlen, strpos, substr, date, array, explode, implode, file\_get\_contents, file\_put\_contents, mail, and many more.

#### 23: What is the main difference between asp net and PHP?

ASP.NET is a web application framework developed by Microsoft, while PHP is a server-side scripting language. Some key differences include:

* ASP.NET is typically used with the C# or Visual Basic.NET languages, while PHP has its own scripting language.
* ASP.NET is often used with Microsoft technologies like IIS and SQL Server, while PHP is platform-agnostic and can run on various web servers and databases.
* ASP.NET is integrated with the Windows ecosystem, whereas PHP is more open-source and cross-platform.
* ASP.NET uses a compiled approach, while PHP is interpreted.

#### 24: What is the use of session and cookies in PHP?

Sessions and cookies are used to maintain state information between HTTP requests in PHP:

* Sessions are server-side mechanisms for storing and retrieving data for a specific user across multiple pages during their visit to a website. They are typically more secure but require server resources.
* Cookies are small pieces of data stored on the client-side (user's browser). They can be used to store user-specific information and are sent with each HTTP request. They are less secure, as the user can view and modify them.

#### 25: What is the difference between $message in PHP?

* $message is a variable that holds a single value.
* $$message is a variable variable. It takes the value of $message and treats it as the name of another variable whose value it represents. For example, if $message contains the value "varName", then $$message is equivalent to $varName.

#### 26: How can we create a database using PHP and MySQL?

To create a database using PHP and MySQL, you typically need to perform the following steps:

1. Connect to your MySQL server using PHP (e.g., mysqli\_connect or PDO).
2. Execute an SQL query to create a new database (e.g., CREATE DATABASE database\_name).
3. Check if the database creation was successful.
4. Close the database connection. Here's a simplified example using mysqli:

$servername = "localhost";

$username = "username";

$password = "password";

// Create a connection

$conn = mysqli\_connect($servername, $username, $password);

// Check connection

if (!$conn) {

die("Connection failed: " . mysqli\_connect\_error());

}

// Create a new database

$sql = "CREATE DATABASE mydb";

if (mysqli\_query($conn, $sql)) {

echo "Database created successfully";

} else {

echo "Error creating database: " . mysqli\_error($conn);

}

// Close the connection

mysqli\_close($conn);

#### 27: What is GET and POST method in PHP?

GET and POST are two HTTP request methods used to send data to a web server:

* GET: Data is appended to the URL as query parameters. It's primarily used for retrieving data. It has a limitation on the amount of data that can be sent (the data is visible in the URL).
* POST: Data is sent in the body of the HTTP request. It's used for sending larger amounts of data, like form submissions, and is more secure as the data is not visible in the URL.

#### 28: What is the use of callback in PHP?

A callback in PHP is a reference to a function or method that can be passed as an argument to another function. Callbacks are often used for customizing the behavior of functions, especially in scenarios like sorting arrays or handling asynchronous operations.

For example, the usort function can take a callback function to customize the sorting behavior of an array.

#### 29: What are PHP Magic Methods/Functions?

PHP magic methods are special methods with double underscores (e.g., \_\_construct, \_\_destruct, \_\_get, \_\_set, etc.) that provide predefined functionality in classes. They are automatically called by the PHP interpreter in response to specific events. Magic methods are commonly used for object initialization, property access, and more.

#### 30: How can you encrypt password using PHP?

You should never store plain text passwords in a database. Instead, you should hash the passwords using a secure hashing algorithm like bcrypt. Here's a simplified example of how to hash and verify passwords in PHP:

// Hashing a password

$password = "user\_password";

$hashed\_password = password\_hash($password, PASSWORD\_BCRYPT);

// Verifying a password

$entered\_password = "user\_input\_password";

if (password\_verify($entered\_password, $hashed\_password)) {

// Password is correct

} else {

// Password is incorrect

}

#### 31: How to connect to a URL in PHP?

To connect to a URL in PHP, you can use functions like file\_get\_contents() or cURL. Here's an example using file\_get\_contents():

$url = "https://example.com";

$data = file\_get\_contents($url);

For more advanced requests with options like setting headers and handling cookies, you can use the cURL library.

#### 32: What is Type hinting in PHP?

Type hinting in PHP is a feature that allows you to specify the data type of a function's parameters or return value. This helps improve code clarity and can catch type-related errors early. For example:

function add(int $a, int $b): int {

return $a + $b;

}

#### 33: What is the difference between runtime exception and compile time exception?

Runtime exceptions (also called runtime errors) occur during the execution of a program, while compile-time exceptions (also called compile-time errors) occur during the compilation of the code. Compile-time errors prevent the program from being compiled and executed, whereas runtime exceptions can occur after the program has started running.

#### 34: Is PHP a loosely typed language?

Yes, PHP is a loosely typed language. In PHP, you don't need to explicitly declare data types for variables, and variables can change their data type during runtime. This flexibility can lead to unexpected behaviors if not handled carefully.

#### 35: What is required to use the image function?

To use image functions in PHP, you need the GD (Graphics Draw) library extension enabled on your PHP server. You can enable it in your PHP configuration or compile PHP with GD support.

#### 36: Mention the popular Content Management Systems (CMS) in PHP?

Some popular PHP-based Content Management Systems (CMS) include WordPress, Joomla, Drupal, and Magento.

#### 37: Mention some of the constants in PHP and their purpose.

PHP has many predefined constants. Some common ones include: \_\_LINE\_\_, \_\_DIR\_\_, \_\_CLASS\_\_ and more.

#### 38: Explain the uses of explode() and implode() functions.

* explode(): This function is used to split a string into an array based on a delimiter. For example, you can split a comma-separated string into an array of values.
* implode(), also known as join(): This function is used to join the elements of an array into a single string using a specified delimiter.

#### 39: How to make a connection with MY SQL server?

To connect to a MySQL server in PHP, you can use functions like mysqli\_connect, PDO, or the newer MySQLi (MySQL Improved) functions. Here's an example using MySQLi:

$servername = "localhost";

$username = "username";

$password = "password";

$database = "dbname";

$conn = mysqli\_connect($servername, $username, $password, $database);

if (!$conn) {

die("Connection failed: " . mysqli\_connect\_error());

}

#### 40: Explain the difference between mysqli\_connect and mysqli\_pconnect.

* mysqli\_connect: This function establishes a regular (non-persistent) connection to the MySQL server. The connection is closed when the PHP script finishes execution.
* mysqli\_pconnect: This function establishes a persistent connection to the MySQL server, which remains open even after the PHP script finishes execution. Persistent connections can reduce connection overhead but may have some limitations and drawbacks.

#### 41: How do we access the data transfer through the URL with the POST method?

Data transferred through the POST method is accessed using the $\_POST superglobal in PHP. For example:

$value = $\_POST['input\_name'];

Here, input\_name is the name of the input field in an HTML form that was submitted with the POST method.

#### 42: Explain the unlink() function in PHP.

The unlink() function in PHP is used to delete a file from the server. For example:

$file\_path = "path/to/file.txt";

if (unlink($file\_path)) {

echo "File deleted successfully.";

} else {

echo "Error deleting file.";

}

#### 43: Explain what does the unset() function mean?

The unset() function in PHP is used to destroy a variable, making it no longer available for use in the script. It can also be used to remove elements from an array. For example:

$variable = "Hello";

unset($variable); // $variable is now undefined

$array = [1, 2, 3];

unset($array[1]); // Removes the element at index 1

#### 44: What is the process to automatically run off incoming data?

It's not clear what you mean by "run off incoming data." If you are referring to processing incoming data automatically, you typically use server-side scripts to handle and process data received from web forms or other sources.

#### 45: Describe what the function get\_magic\_quotes\_gpc() means.

get\_magic\_quotes\_gpc() was a function used to check if PHP's "Magic Quotes" feature was enabled. Magic Quotes automatically added backslashes to incoming data to protect against SQL injection and other security issues. However, this feature is deprecated in modern PHP versions, and it's no longer available. You should not use it in new code.

#### 46: Can we remove the HTML tags from the data?

Yes, you can remove HTML tags from data in PHP using functions like strip\_tags(). For example:

$text = "<p>This is <b>some</b> text.</p>";

$clean\_text = strip\_tags($text);

echo $clean\_text; // Output: "This is some text."

#### 47: What is the procedure to cast types in PHP?

You can cast types in PHP using explicit type casting. For example, to cast a variable to an integer, you can use (int) or (integer):

$string = "123";

$integer = (int)$string; // Cast to integer

#### 48: Explain what accessing a class via:: means.

In PHP, you can access class properties and methods using the :: operator. It's used to access static properties and methods or constants of a class. For example:

class MyClass {

const MY\_CONSTANT = 42;

public static function myMethod() {

// ...

}

}

$constant\_value = MyClass::MY\_CONSTANT;

MyClass::myMethod();

#### 49: In PHP, objects used are passed by value or passed by reference.

In PHP, objects are passed by reference. When you pass an object to a function or assign it to another variable, you are working with a reference to the same object. Any changes made to the object inside the function or through the new variable will affect the original object.

#### 50: Explain what Persistent Cookie is.

A persistent cookie (also known as a long-term cookie) is a type of HTTP cookie that remains on the user's device for an extended period, even after the browser is closed. It has an expiration date set in the future. Persistent cookies are typically used to remember user preferences or login sessions over an extended period.

#### 51: At what phase do sessions end in PHP?

Sessions in PHP typically end when the user's browser is closed, or when the session is explicitly destroyed using session\_destroy() or when it times out due to inactivity. The session timeout is determined by the session.gc\_maxlifetime setting in the PHP configuration.

#### 52: Explain what $GLOBALS is.

$GLOBALS is a superglobal array in PHP that is used to access global variables from within functions or methods. It allows you to access variables that are outside the current scope.

#### 53: What does $\_SERVER mean?

$\_SERVER is a superglobal array in PHP that provides information about the server environment, request, and other server-related data. It includes information like headers, file paths, and server information.

#### 54: What do $\_FILES mean?

$\_FILES is a superglobal array in PHP that is used to access and manage file uploads submitted via HTML forms with the enctype="multipart/form-data" attribute. It provides information about uploaded files, such as file names, MIME types, and file sizes.

#### 55: Explain the use of the header() function in PHP.

The header() function in PHP is used to send HTTP headers, such as setting cookies, redirecting to a different page, specifying content type, and more. It must be called before any actual output is sent to the browser. It allows you to control various aspects of the HTTP response.

#### 56: How do you execute a PHP script from the command line?

To execute a PHP script from the command line, you can use the php command followed by the script's filename, like this:

php script.php

#### 57: What are the different PHP array functions?

PHP provides a variety of array functions for manipulating arrays. Some common ones include array\_push(), array\_pop(), array\_shift(), array\_unshift(), count(), array\_merge(), array\_slice(), array\_key\_exists(), and many more.

#### 58: Explain the difference between the functions strstr() and stristr().

* strstr(): This function is used to find the first occurrence of a substring within a string. It is case-sensitive, so it will only match if the case matches.
* stristr(): This function is also used to find the first occurrence of a substring within a string, but it is case-insensitive, meaning it will match regardless of the case.

$string = "Hello, World!";

$result = strstr($string, "world"); // $result is FALSE

$result = stristr($string, "world"); // $result is "World!"

#### 59: What rules determine the “truth” of any value not already of the Boolean type?

The "truthiness" of a value in programming languages like PHP is often determined by its evaluation in a boolean context. In PHP, values that are considered "truthy" include non-empty strings, non-zero numbers, and non-empty arrays. Values that are considered "falsy" include an empty string, zero, and empty arrays.

#### 60: What is the difference between a single-quoted string and a double-quoted string?

The main difference between single-quoted strings and double-quoted strings in PHP is that variables and escape sequences inside double-quoted strings are interpolated and evaluated, while in single-quoted strings, they are treated as literal characters.

#### 61: Which function can be used to exit from the script after displaying the error message?

The die() or exit() function can be used to exit from a script after displaying an error message in PHP.

#### 62: Which function is used in PHP to check the data type of any variable?

The gettype() function is used in PHP to check the data type of a variable.

#### 63: How can you increase the maximum execution time of a script in PHP?

You can increase the maximum execution time of a script in PHP using the set\_time\_limit() function or by modifying the max\_execution\_time directive in the PHP configuration (php.ini) file.

#### 64: What is meant by ‘passing the variable by value and reference’ in PHP?

Passing a variable by value in PHP means that you are passing a copy of the variable to a function or assignment, so changes made to the variable within the function or assignment do not affect the original variable outside. Passing a variable by reference means that you are passing a reference to the original variable, and changes made to it within the function or assignment will affect the original variable.

#### 65: Explain type casting and type juggling.

Type casting is the explicit conversion of one data type to another, while type juggling is the automatic conversion of data types in expressions. For example, in type casting, you might explicitly convert a string to an integer using (int) like (int)$str, while in type juggling, PHP might automatically convert a string to an integer if it's used in an arithmetic operation.

#### 66: How can you retrieve data from the MySQL database using PHP?

To retrieve data from a MySQL database using PHP, you can use functions provided by the MySQLi extension or PDO (PHP Data Objects). You'll typically establish a database connection, create SQL queries, execute them, and fetch results using functions like mysqli\_query(), mysqli\_fetch\_assoc(), or prepared statements in PDO.

#### 67: How can you create a session in PHP?

To create a session in PHP, you can use the session\_start() function. This function initializes a session or resumes the current session if one exists.

#### 68: Which function you can use in PHP to open a file for reading or writing or for both?

In PHP, you can use the fopen() function to open a file for reading, writing, or both, depending on the mode you specify. For example, fopen('file.txt', 'r') opens a file for reading, and fopen('file.txt', 'w') opens a file for writing.

#### 69: What is the difference between include() and require()?

include() and require() are both used to include and execute the content of another PHP file in the current script. The main difference is in how they handle errors. include() will generate a warning and continue executing the script if the file is not found, while require() will generate a fatal error and stop script execution. Use include() when the included file is not critical, and use require() when it's essential.

#### 70: Which function is used in PHP to delete a file?

The unlink() function is used in PHP to delete a file.

#### 71: What is the use of strip\_tags() method?

The strip\_tags() method is used to remove HTML and PHP tags from a string. It can be used to sanitize user input or to extract plain text from an HTML document.

#### 72: How can you send an HTTP header to the client in PHP?

You can send an HTTP header to the client in PHP using the header() function. For example, header("Location: http://example.com"); can be used to redirect the user to another URL.

#### 73: Which functions are used to count the total number of array elements in PHP?

The count() and sizeof() functions are commonly used to count the total number of elements in an array in PHP.

#### 74: How can you upload a file using PHP?

To upload a file using PHP, you typically create an HTML form with an <input type="file"> element, and then use PHP to process the uploaded file. You can use the $\_FILES superglobal to access the uploaded file's information and move it to the desired location on the server using functions like move\_uploaded\_file().

#### 75: Which function is used in PHP to search a particular value in an array?

You can use the in\_array() function in PHP to search for a particular value in an array. It returns true if the value is found in the array, false otherwise.

#### 76: What is the use of the $\_REQUEST variable?

The $\_REQUEST variable in PHP is a superglobal array that is used to collect data from both HTTP GET and POST requests. It contains data from the $\_GET, $\_POST, and $\_COOKIE superglobal arrays. However, it's important to use this variable with caution as it can potentially lead to security vulnerabilities, such as data injection and should be sanitized or validated before use.

#### 77: How long does a PHP session last for?

The duration of a PHP session is determined by the session settings in the PHP configuration. By default, a session lasts until the user closes their browser, or until a specific timeout is reached. You can configure the session's lifetime using the session.gc\_maxlifetime directive in your php.ini file.

#### 78: What is the use of mysqli\_real\_escape\_string() function?

The mysqli\_real\_escape\_string() function is used to escape and sanitize user input that is going to be used in SQL queries. It helps prevent SQL injection by escaping special characters, making the input safe to use in database queries.

#### 79: Which functions are used to remove whitespaces from the string?

PHP provides several functions to remove whitespaces from a string, including trim(), ltrim(), and rtrim(). These functions remove whitespace characters (spaces, tabs, newlines) from the beginning, end, or both ends of a string, respectively.

#### 80: What is a persistence cookie?

A persistence cookie, often referred to as a persistent or long-term cookie, is a type of HTTP cookie that doesn't expire when the user closes the web browser. Instead, it persists on the user's computer for a specified duration, typically set by the website, and can be used to remember user preferences or login information over extended periods.

#### 81: How can a cross-site scripting attack be prevented by PHP?

Cross-site scripting (XSS) attacks can be prevented in PHP by sanitizing and validating user input, escaping output data when displaying it in web pages, and using security mechanisms like input validation, output encoding, and content security policies (CSP). Additionally, using functions like htmlspecialchars() to escape user input when rendering it in HTML can help prevent XSS attacks.

#### 82: What is meant by public, private, protected, static and final scopes?

Public, private, protected, static, and final are access modifiers and scopes in PHP:

* public: Members marked as public can be accessed from anywhere, both inside and outside the class.
* private: Members marked as private are only accessible within the class itself.
* protected: Members marked as protected can be accessed within the class and its subclasses.
* static: Static members are associated with the class rather than instances of the class.
* final: Final classes or methods cannot be extended or overridden by subclasses.

#### 83: How can image properties be retrieved in PHP?

To retrieve image properties in PHP, you can use functions like getimagesize() or image-related functions from the GD or Imagick extensions. getimagesize() returns an array with information about the image, including its size, width, height, and MIME type.

#### 84: What is the difference between abstract class and interface?

An abstract class in PHP is a class that cannot be instantiated and is meant to be extended by other classes. It can contain abstract methods (methods without implementation) that must be implemented in the child classes. An interface, on the other hand, is a contract that defines a set of methods that implementing classes must provide, but it cannot contain any method implementations. Classes can implement multiple interfaces but can only extend one abstract class.

#### 85: What is garbage collection?

Garbage collection in PHP is the process of automatically identifying and cleaning up memory that is no longer in use, releasing it back to the system. PHP uses a reference counting mechanism and a cyclic garbage collector to manage memory automatically. This helps prevent memory leaks and ensures efficient memory usage.

#### 86: What is PDO?

PDO (PHP Data Objects) is a PHP extension that provides a consistent interface for interacting with databases, regardless of the specific database system being used. PDO allows you to work with multiple database systems (e.g., MySQL, PostgreSQL, SQLite) using the same set of functions, making it a more portable and secure way to work with databases in PHP.

#### 87: What does isset() function?

The isset() function in PHP is used to check if a variable or array element is set and is not null. It returns true if the variable exists and is not null, and false if it does not exist or is null.

#### 88: Explain some of the PHP array functions?

Some of the commonly used PHP array functions include count(), array\_push(), array\_pop(), array\_shift(), array\_unshift(), array\_merge(), array\_reverse(), array\_slice(), array\_splice(), in\_array(), and array\_search(). These functions allow you to perform various operations on arrays, such as adding, removing, and searching for elements.

#### 89: What are the functions to be used to get the image’s properties (size, width and height)?

To get an image's properties such as size, width, and height in PHP, you can use the getimagesize() function. It returns an array that contains information about the image, including these properties.

#### 90: What is the difference between Session and Cookie?

The main differences between sessions and cookies in PHP are as follows:

* Sessions are typically stored on the server, while cookies are stored on the client's browser.
* Sessions are more secure for storing sensitive data, as they are not exposed to the client.
* Cookies have an expiration date and can be long-term (persistent), while sessions usually expire when the user closes the browser.
* Sessions are often used to store user-specific data across multiple pages, while cookies are often used for tracking user preferences or login information.

#### 91: How to set cookies in PHP?

To set cookies in PHP, you can use the setcookie() function. Here's an example of how to set a cookie:

setcookie("user", "John Doe", time() + 3600, "/");

This code sets a cookie named "user" with the value "John Doe" that will expire in 1 hour (3600 seconds) and is accessible from the root directory ("/").

#### 92: What is sql injection ?

SQL injection is a security vulnerability that occurs when an attacker inserts malicious SQL code into input fields or parameters, which are then executed by a web application's database. This can allow the attacker to manipulate the database, retrieve, modify, or delete data, and potentially gain unauthorized access to a system.

#### 93: Distinguish between urlencode and urldecode?

urlencode() and urldecode() are functions used to encode and decode URLs in PHP. urlencode() is used to convert special characters in a string to URL-encoded format, making it safe to include in a URL. urldecode() is used to reverse the process and decode a URL-encoded string back to its original form.

#### 94: What is the use of htmlentities() function in PHP?

The htmlentities() function in PHP is used to convert characters with special meaning in HTML to their corresponding HTML entities. This is often used to prevent cross-site scripting (XSS) attacks by encoding user-generated content before it is displayed in a web page.

#### 95: What is the difference between mysqli and PDO?

The main differences between mysqli and PDO in PHP are:

* mysqli is specific to MySQL databases, while PDO is more universal and can work with multiple database systems.
* mysqli provides both procedural and object-oriented interfaces, while PDO primarily uses object-oriented methods.
* PDO supports prepared statements and named placeholders, making it more secure against SQL injection.
* mysqli has better support for stored procedures and multi-query execution.
* PDO is considered more extensible and suitable for working with various database backends.

#### 96: What is MIME in PHP?

MIME (Multipurpose Internet Mail Extensions) is a standard used in PHP and other programming languages to define the type and structure of data files, especially for email and the internet. MIME types describe the format of a file, such as text, image, audio, or video, and help applications determine how to handle or display the data.

#### 97: Explain session\_start() function and session\_destroy() function in PHP?

* session\_start() is used to initialize a session or resume the current session. It must be called at the beginning of a script that intends to use sessions. It creates a unique session identifier for each user and can store session data on the server.
* session\_destroy() is used to destroy the current session. It deletes all session data associated with the user and effectively logs them out. It is often used when a user wants to log out of a website.

#### 98: What is the command line for executing a PHP script?

To execute a PHP script from the command line, you can use the php command followed by the script's filename. For example:

php myscript.php

#### 99: What are \_\_wakeup and \_\_sleep methods in PHP?

* \_\_wakeup() is called when an object is unserialized. You can use it to perform any necessary cleanup or initialization when an object is recreated from its serialized form.
* \_\_sleep() is called before an object is serialized. You can specify which object properties should be included in the serialization process by returning an array of property names from this method.

#### 100: What is the role of the lambda function used in PHP?

In PHP, a lambda function (also known as an anonymous function) is a small, anonymous function that can be used as an argument to higher-order functions or assigned to variables. It is typically used for short, simple operations where defining a full function is unnecessary. Lambda functions can be created using the fn keyword (available in PHP 7.4 and later) or by using function(). They are often used in array functions like array\_map(), array\_filter(), and usort(). Lambda functions provide a more concise way to define functions on the fly without explicitly naming them.

#### 101: What do you mean by object-oriented programming?

Object-oriented programming (OOP) is a programming paradigm that uses objects, which are instances of classes, to structure and organize code. In OOP, the key concepts are classes and objects. Classes define the blueprint for creating objects, and objects are instances of classes that encapsulate data and behavior (methods). OOP promotes the principles of encapsulation, inheritance, and polymorphism to create modular, reusable, and maintainable code.

#### 102: What are the characteristics of OOPs?

The characteristics of object-oriented programming (OOP) include:

* Encapsulation: This refers to the bundling of data (attributes or properties) and methods (functions) that operate on the data into a single unit called a class. Objects of the class can access and manipulate the data through defined interfaces.
* Inheritance: Inheritance allows a class (subclass or derived class) to inherit properties and behaviors from another class (superclass or base class). It promotes code reuse and hierarchy.
* Polymorphism: Polymorphism enables objects of different classes to be treated as objects of a common superclass. It allows for method overriding and dynamic method dispatch, making code more flexible.
* Abstraction: Abstraction is the process of simplifying complex reality by modeling classes based on essential properties and behaviors while hiding unnecessary details.
* Modularity: OOP promotes the creation of self-contained modules (classes) that can be developed and maintained independently, enhancing code modularity and reusability.

#### 103: What is a class and object?

* Class: A class is like a blueprint or template for creating objects. It defines the structure and behavior of objects.
* Object: An object is an instance of a class. It is a concrete realization of the class's blueprint, with its own data and behavior.

#### 104: What do you mean by Inheritance?

When a class inherits attributes or methods from another class, this is called inheritance.

There are different types of inheritance in object oriented programming:

* Single Inheritance: Single inheritance occurs when a class inherits from only one parent class.
* Multiple Inheritance(Interface Inheritance): Multiple inheritance refers to a class inheriting properties and methods from more than one parent class. PHP does not supports multiple inheritance. But you can implements multiple interfaces or you can use traits for this purpose.
* Multi-level Inheritance: Multiple inheritance involves a chain of classes where one class inherits from another, and subsequent classes inherits from those classes.
* Hierarchical Inheritance: Hierarchical inheritance occurs when multiple child classes inherit from a single parent class. Each child class my have it's own unique properties and methods, but they share the common behavior from the parent class.
* Hybrid inheritance: Hybrid inheritance is a combination of two or more types of inheritance within the same program. While PHP supports only single and interface | traits inheritance, combining them can lead to hybrid inheritance.

Here are some examples:

/\*\*

\* Single Inheritance Example:

\*/

class Vehicle

{

protected $brand;

protected $model;

protected $year;

public function \_\_construct($brand, $model, $year)

{

$this->brand = $brand;

$this->model = $model;

$this->year = $year;

}

public function getInfo()

{

return "Brand: $this->brand, Model: $this->model, Year: $this->year";

}

}

class Car extends Vehicle

{

private $fuel\_type;

public function \_\_construct($brand, $model, $year, $fuel\_type)

{

parent::\_\_construct($brand, $model, $year);

$this->fuel\_type = $fuel\_type;

}

/\*\*

\* Method overriding

\*/

public function getInfo()

{

return parent::getInfo() . " Fuel type: $this->fuel\_type";

}

}

$aVehicle = new Vehicle('Bus', 'No model', 2020);

// echo $aVehicle->getInfo();

// $car = new Car('Toyota', 'Carmy', 2022, 'Gasoline');

// echo $car->getInfo();

/\*\*

\* Multiple Inheritance: In PHP, you cannot use multiple inheritance. You have to use interfaces or traits to implement multi-inheritance in PHP.

\*/

interface Communication

{

public function makeCall($phone\_number);

public function sendText($phone\_number, $message);

}

interface Multimedia

{

public function takePhoto();

public function playMusic($song);

}

class Smartphone implements Communication, Multimedia

{

public function makeCall($phone\_number)

{

echo "Calling $phone\_number" . PHP\_EOL;

}

public function sendText($phone\_number, $message)

{

echo "Sending text to $phone\_number: $message" . PHP\_EOL;

}

public function takePhoto()

{

echo "Taking a photo" . PHP\_EOL;

}

public function playMusic($song)

{

echo "Playing the song named: $song";

}

}

$iPhone = new SmartPhone();

// $iPhone->sendText('01309900000', 'Hi there!');

/\*\*

\* Multi-level inheritance: Multi-level inheritance is a concept where a chain of classes is created, where each child class inherits from it's parent, forming a hierarchy. Like: Your grand-father -> your-father -> you 😀

\*/

class Employee

{

protected $name;

protected $employeeId;

public function \_\_construct($name, $employeeId)

{

$this->name = $name;

$this->employeeId = $employeeId;

}

public function getDetails()

{

return "Employee ID: $this->employeeId, Name: $this->name";

}

}

class Manager extends Employee

{

protected $department;

public function \_\_construct($name, $employeeId, $department)

{

parent::\_\_construct($name, $employeeId);

$this->department = $department;

}

/\*\*

\* Method overrides

\*/

public function getDetails()

{

return parent::getDetails() . ", Department: $this->department (Manager)";

}

}

class Director extends Manager

{

protected $responsibilities;

public function \_\_construct($name, $employeeId, $department, $responsibilities)

{

parent::\_\_construct($name, $employeeId, $department);

$this->responsibilities = $responsibilities;

}

/\*\*

\* Method overrides

\*/

public function getDetails()

{

return parent::getDetails() . ", Responsibilities: $this->responsibilities (Director)";

}

}

$employee = new Employee("John Doe", "E123");

$manager = new Manager("Alice Smith", "M456", "Marketing");

$director = new Director("Eve Johnson", "D789", "Finance", "Financial strategy");

// Calling methods

echo "Employee Info: " . $employee->getDetails() . PHP\_EOL;

echo "Manager Info: " . $manager->getDetails() . PHP\_EOL;

echo "Director Info: " . $director->getDetails() . PHP\_EOL;

#### 105: What is Polymorphism?

Polymorphism is one of the fundamental concepts in object oriented programming(OOP). It allows objects of different classes to be treated as objects of a common superclass. It enables you to write more flexible and extensible codes.

In PHP, Polymorphism is achieved through method overriding and interfaces. Let's look at an example:

interface Animal {

public function speak();

}

class Dog implements Animal {

public function speak() {

echo 'Woof! Woof!';

}

}

class Cat implements Cat {

public function speak() {

echo 'Meow! Meow!';

}

}

Now you create instances of these classes and use polymorphism to treat them as instances of the common interface Animal;

#### 106: How is the static keyword used in the program?

The static keyword is used to define class-level variables and methods that belong to the class itself, not to instances of the class.

#### 107: Name the types of constructors.

There are two types of constructors:

* Default Constructor (No-Argument Constructor)
* Parameterized Constructor (accepts arguments)

#### 108: What do PHP namespaces mean?

PHP namespaces are used to organize code elements (classes, functions, constants) into logical groups, preventing naming conflicts and improving code organization.

#### 109: Describe the types of access modifiers in PHP.

PHP has three main access modifiers:

* Public: Accessible from anywhere.
* Protected: Accessible within the class and its subclasses.
* Private: Accessible only within the class.

#### 110: Is operator overloading supported by PHP?

PHP does not support operator overloading, which means you can't define custom behaviors for operators like + or - for your classes.

#### 111: What is the difference between encapsulation and abstraction?

* Encapsulation: Protecting data and controlling access to it within a class.
* Abstraction: Simplifying complex systems by focusing on essential features and ignoring unnecessary details.

#### 112: What is the purpose of the yield keyword in PHP?

The yield keyword in PHP is used to create generators, allowing you to iterate over large data sets efficiently by pausing and resuming a function's execution to save memory and processing time.

#### 113: What are the types of Polymorphism?

There are two main types of polymorphism in OOP:

* Compile-time (Static) Polymorphism: This is achieved through method overloading and operator overloading. Method overloading is when a class has multiple methods with the same name but different parameters. The correct method to call is determined at compile time based on the method's signature.
* Runtime (Dynamic) Polymorphism: This is achieved through method overriding. Method overriding occurs when a subclass provides a specific implementation for a method that is already defined in its superclass. The correct method to call is determined at runtime based on the actual object's type.

#### 114: What is the use of traits in PHP?

Traits in PHP allow you to reuse code in multiple classes without inheritance. Traits provide a mechanism for code reuse and composition by allowing the inclusion of methods and properties from a trait into a class. This is useful when multiple classes need to share common functionality, but you want to avoid the limitations of single inheritance.

#### 115: What is the use of the finalize method?

The finalize method is not a standard method in PHP or most modern programming languages. It may be a concept from older languages like Java, where it was used for resource cleanup and is related to garbage collection. In modern PHP, resource management is typically handled automatically by the language, and you don't need a finalize method.

#### 116: Explain Polymorphism with the help of an example.

Polymorphism is best explained through an example of method overriding, which demonstrates runtime polymorphism. Consider a class hierarchy with a base class Shape and two subclasses, Circle and Rectangle. All of them have a calculateArea method. When you call calculateArea on a Shape object, the specific version of the method in the actual subclass will be executed.

class Shape {

public function calculateArea() {

// Common code for all shapes

}

}

class Circle extends Shape {

public function calculateArea() {

// Calculate area specific to a circle

}

}

class Rectangle extends Shape {

public function calculateArea() {

// Calculate area specific to a rectangle

}

}

$circle = new Circle();

$rectangle = new Rectangle();

echo $circle->calculateArea(); // Calls the Circle's method

echo $rectangle->calculateArea(); // Calls the Rectangle's method

This demonstrates polymorphism because the same method name is used, but the actual behavior is determined by the type of the object.

#### 117: What is the use of the final keyword in PHP?

The final keyword in PHP is used to restrict the inheritance and overriding of methods and classes. When you declare a class or method as final, it means that it cannot be extended or overridden by any subclass. It's used to prevent further modification or specialization of a class or method.

#### 118: What is Interface in PHP?

An interface in PHP defines a contract that a class must adhere to. It specifies a set of method signatures that a class implementing the interface must provide. Interfaces provide a way to achieve abstraction and ensure that classes follow a certain structure without specifying the implementation details. A class can implement multiple interfaces, enabling it to define multiple contracts.

#### 119: What is method overloading and how do you use it in PHP OOPs?

Method overloading in PHP allows you to define multiple methods with the same name but different parameters in a class. PHP does not support traditional method overloading as seen in some other languages. Instead, the latest defined method with a particular name and set of parameters will be the one used. It's more about method replacement than overloading. You can achieve similar behavior by using default parameter values or variable-length argument lists (variadic functions) in your methods.

#### 120: What is hybrid inheritance?

Hybrid inheritance is a combination of different types of inheritance, typically involving multiple inheritance (a class inheriting from more than one class) and hierarchical inheritance (a class with multiple subclasses). While some languages support hybrid inheritance, PHP does not directly support multiple inheritance, so achieving hybrid inheritance may require using interfaces and composition.

#### 121: What is static polymorphism?

Static polymorphism, also known as compile-time polymorphism, is achieved through method overloading or operator overloading. The correct method or operator to use is determined at compile time based on the method or operator signature.

#### 122: What is dynamic polymorphism?

Dynamic polymorphism, also known as runtime polymorphism, is achieved through method overriding. The correct method to call is determined at runtime based on the actual object's type, allowing objects of different classes to be treated as objects of a common superclass.

#### 123: Differentiate between overloading and overriding.

* Method Overloading: Method overloading involves having multiple methods with the same name in a class, but with different parameters. The correct method to call is determined at compile time based on the method's signature. It is a form of compile-time (static) polymorphism.
* Method Overriding: Method overriding occurs when a subclass provides a specific implementation for a method that is already defined in its superclass. The correct method to call is determined at runtime based on the actual object's type. It is a form of runtime (dynamic) polymorphism.

#### 124: What is data abstraction?

Data abstraction is the process of simplifying complex systems by modeling classes based on essential properties and behaviors while hiding unnecessary details. It allows you to focus on the high-level structure of data and operations without getting into the specifics of how data is stored or processed. Data abstraction is a key concept in object-oriented programming, and it promotes modularity and maintainability.

#### 125: How to achieve data abstraction?

Data abstraction can be achieved through the use of classes and objects in object-oriented programming. You create classes that represent abstract data types, define the public interface (methods) that should be accessible, and hide the internal details (private properties and methods). By doing this, you encapsulate the data and provide a way for other code to interact with the data only through well-defined, abstract methods.

#### 126: Differentiate between data abstraction and encapsulation.

* Data Abstraction: Data abstraction is about simplifying complex systems by defining high-level structures, focusing on what data and operations are needed, and ignoring unnecessary details. It is a broader concept that includes encapsulation.
* Encapsulation: Encapsulation, on the other hand, is one of the techniques used to achieve data abstraction. It involves bundling the data (attributes) and methods that operate on the data into a single unit (a class) and controlling access to that data. Encapsulation is a specific way to implement data abstraction.

#### 127: What is the difference between OOP and SOP?

* OOP (Object-Oriented Programming): OOP is a programming paradigm that uses objects and classes to structure and organize code. It promotes concepts like encapsulation, inheritance, and polymorphism, making it easier to create modular, reusable, and maintainable code.
* SOP (Structured or Procedural Programming): SOP is a programming paradigm that uses procedures or functions to organize code. It is based on a top-down approach where the code is structured around procedures that perform specific tasks. It does not emphasize the use of objects and classes.

The main difference is the approach to structuring code. OOP focuses on objects and their interactions, while SOP organizes code around procedures or functions.

#### 128: What is the difference between dependency injection and factory design pattern?

* Dependency Injection (DI): Dependency injection is a design pattern that focuses on providing a class with its dependencies (usually other objects or services) from the outside, rather than having the class create its dependencies. It promotes loose coupling between classes, making them more modular and easier to test.
* Factory Design Pattern: The Factory Design Pattern is used to create objects without specifying the exact class of object to be created. It provides a method (the factory method) for creating objects of a particular class or its subclasses. This pattern allows you to abstract the process of object creation.

The key difference is that DI is about providing dependencies to a class, while the Factory Design Pattern is about creating objects without necessarily knowing their concrete types.

#### 129: What libraries would be useful for OOP development in PHP?

For OOP development in PHP, you can consider using various libraries and frameworks, depending on your project requirements. Some popular libraries and frameworks that promote OOP principles in PHP development include:

* Symfony: A high-performance PHP framework that encourages the use of OOP and provides a wide range of components for various tasks.
* Laravel: A popular PHP framework that utilizes OOP principles for building web applications. It includes an elegant and expressive syntax.
* Doctrine: An Object-Relational Mapping (ORM) library for PHP that allows you to work with databases using OOP concepts.
* Composer: While not a library, Composer is a dependency management tool that helps you install and autoload PHP libraries in your OOP projects.
* PHPStan: A static analysis tool for PHP that helps ensure type safety and adherence to OOP principles.
* PHPUnit: A testing framework for PHP that facilitates unit testing in an object-oriented context.

#### 130: If a class inherits two methods with the same name from its parent class and both have the same number of parameters, then what happens when you try to instantiate this class?

If a class inherits two methods with the same name from its parent class, and both methods have the same number of parameters, the behavior can vary depending on the programming language and how it handles method resolution. In some languages, the method from the most specific subclass is called, while in others, you might encounter an ambiguity error, and you would need to explicitly specify which method to call using parent class names. The exact behavior depends on the language's rules for method resolution and may vary between programming languages.

### Interview Questions for MySQL:

#### 0: How to get the current MySQL version?

To get the current MySQL version, you can use the following methods:

Command Line (Linux/Unix): Use the command mysql --version. Command Line (Windows): Use the command mysql -V. MySQL Client: Log in to the MySQL client and run the SQL query SELECT VERSION(); phpMyAdmin: The MySQL version is typically displayed on the initial page MySQL Workbench: You can find the MySQL version in the "Server Status" section.

#### 1: What is the MySQL server’s default port?

The default port for MySQL server is 3306.

#### 2: How many different tables are present in MySQL?

The number of different tables in a MySQL database can vary widely and depends on the specific database schema and the data model used. There is no fixed or standard number of tables in MySQL. The number of tables is determined by the design and requirements of the database. Some databases may have just a few tables, while others can have many tables to represent different data entities and relationships.

#### 3: What is Difference between CHAR\_LENGTH and LENGTH?

CHAR\_LENGTH counts the number of characters, while LENGTH counts the number of bytes in a string.

#### 4: What do you understand by % and \_ in the like statement?

% matches any sequence of characters (including zero). \_ matches any single character.

#### 5: How many index columns can be created in a table?

There's no strict limit; you can create multiple indexes on a table, but be cautious of performance implications.

#### 6: What are string types available for columns?

Common string types include VARCHAR, CHAR, TEXT, and ENUM.

#### 7: Explain the main difference between FLOAT and DOUBLE?

FLOAT is a single-precision floating-point number, while DOUBLE is a double-precision floating-point number. DOUBLE provides higher precision but uses more storage.

#### 8: Explain the difference between having and where clause in MySQL.

* WHERE filters rows before aggregation.
* HAVING filters results after aggregation.

#### 9: How can we add a column in MySQL?

Use the ALTER TABLE statement with the ADD COLUMN clause to add a new column to an existing table.

#### 10: How to delete columns in MySQL?

Use the ALTER TABLE statement with the DROP COLUMN clause to delete a column from an existing table.

#### 11: How to delete a table in MySQL?

Use the DROP TABLE statement followed by the table name to delete a table in MySQL.

#### 12: How to get the top 10 rows?

Use the SELECT statement with LIMIT 10 to retrieve the top 10 rows from a table.

#### 13: What is the use of the ‘DISTINCT’ keyword in MySQL?

The DISTINCT keyword is used to retrieve unique values from a column in a table. It ensures that only distinct values are returned, eliminating duplicates.

#### 14: Which storage engines are used in MySQL?

Common storage engines in MySQL include InnoDB, MyISAM, and MEMORY. Each engine has unique features and is suitable for different use cases.

#### 15: How to create a table in MySQL?

Use the CREATE TABLE statement to define the table structure, specify column names and data types, and set constraints like primary keys and foreign keys.

#### 16: What types of relationships are used in MySQL?

Common types include one-to-one, one-to-many, and many-to-many relationships. These are established using keys like primary and foreign keys.

#### 17: How to insert Date in MySQL?

You can insert a date into a MySQL table using the INSERT statement with a valid date format, like YYYY-MM-DD.

#### 18: What is join? Tell different join in MySQL.

A join is used to combine rows from two or more tables based on a related column between them. Common joins include INNER JOIN, LEFT JOIN (or LEFT OUTER JOIN), RIGHT JOIN (or RIGHT OUTER JOIN), and FULL JOIN (or FULL OUTER JOIN).

#### 19: What is a primary key? How to drop the primary key in MySQL?

A primary key is a unique identifier for each record in a table. To drop a primary key, use the ALTER TABLE statement with the DROP PRIMARY KEY clause.

#### 20: What is InnoDB?

InnoDB is a popular storage engine in MySQL known for its support of transactions, foreign keys, and ACID compliance. It's widely used for reliable and transaction-safe database operations.

#### 21: What is the difference between UNION and UNION ALL in MySQL?

* UNION removes duplicate rows.
* UNION ALL includes all rows, even duplicates

#### 22: What is a timestamp in MySQL?

A timestamp in MySQL is a data type used to store date and time values with a time zone.

#### 23: What is the use of ENUMs in MySQL?

ENUMs are used to represent a set of predefined values for a column. They help ensure data consistency and limit possible values.

#### 24: How can you control max size of heap in MySQL?

You can set the max heap size using the --max-heap-table-size and --tmp-table-size parameters in your MySQL configuration.

#### 25: What is a view? How to create a view?

A view is a virtual table created from the result of a SQL query. To create a view, use the CREATE VIEW statement, specifying the query defining the view.

#### 26: Where MyISAM table will be stored and also give MyISAM formats of storage?

MyISAM tables are stored in the database directory as separate files. Common MyISAM storage formats include .MYD (data), .MYI (index), and .frm (table format) files.

#### 27: How can we save images in MySQL?

Images can be saved in MySQL by storing them in a BLOB (Binary Large Object) column. BLOBs can store binary data, such as images, as part of a table's record.

#### 28: What are trigger and how many TRIGGERS are available in MySQL table?

Triggers are database objects that automatically perform actions in response to predefined events. In MySQL, you can create AFTER INSERT, AFTER UPDATE, and AFTER DELETE triggers for tables.

#### 29: What are Access Control Lists?

Access Control Lists (ACLs) are lists of permissions that specify which users or system processes are granted access to objects, as well as what operations are allowed on given objects in a system.

#### 30: What are various ways to create an index?

You can create indexes using CREATE INDEX statements, by defining primary keys, or by using the ALTER TABLE statement. Common index types are B-tree, hash, and full-text indexes.

#### 31: What are a clustered index and a non clustered index?

In MySQL, the concept of a clustered index is mainly associated with the InnoDB storage engine. A clustered index determines the physical order of rows in a table and is typically based on the primary key. A non-clustered index doesn't affect the physical order of rows and is separate from the data.

#### 32: How to validate emails using a single query?

You can validate emails using regular expressions in a query. For example, SELECT email FROM users WHERE email REGEXP '^[A-Za-z0-9.\_%+-]+@[A-Za-z0-9.-]+\\.[A-Za-z]{2,4}$';

#### 33: How can you handle the –secure-file-priv in MySQL?

You can set the --secure-file-priv option in the MySQL configuration file to specify a directory where MySQL can write secure files for data import/export operations.

#### 34: How do you create a database in MySQL?

Use the CREATE DATABASE statement to create a new database in MySQL.

#### 35: How do you create a table using MySQL?

Use the CREATE TABLE statement to define and create a new table in a MySQL database. Specify the table's structure, column names, data types, and constraints.

#### 36: What is BLOB in MySQL?

BLOB stands for Binary Large Object. It's a MySQL data type used to store binary data, such as images, audio, or other non-text data.

#### 37: How to add users in MySQL?

You can add users in MySQL using the CREATE USER statement or through a client application. You need to specify their username and password.

#### 38: What are MySQL Triggers?

MySQL triggers are database objects that automatically execute actions (e.g., SQL statements) in response to specific events, such as INSERT, UPDATE, or DELETE operations on a table.

#### 39: How many Triggers are possible in MySQL?

In MySQL, you can create multiple triggers for each table. The number of triggers you can create is not strictly limited, but it's good practice to keep them manageable for clarity and maintainability.

#### 40: What is the MySQL server?

The MySQL server is the core component of the MySQL database management system. It handles database operations, such as querying, updating, and managing data.

#### 41: What are the MySQL clients and utilities?

MySQL provides various client applications and utilities like the command-line client, MySQL Workbench, phpMyAdmin, and others. These tools allow you to interact with and manage MySQL databases.

#### 42: Can you explain the logical architecture of MySQL?

The logical architecture of MySQL includes components like the Query Optimizer, Storage Engines, and the SQL Layer. The Query Optimizer optimizes queries, the Storage Engines handle data storage and retrieval, and the SQL Layer manages SQL parsing and execution.

#### 43: What is Scaling in MySQL?

Scaling in MySQL refers to the process of increasing the capacity and performance of a MySQL database system to handle growing workloads and user demands. It can involve techniques like sharding, replication, and clustering to distribute and manage data across multiple servers.

#### 44: What is the purpose of the SHOW TABLES command in MySQL?

The SHOW TABLES command is used to list the tables in the current database.

#### 45: What is normalization and denormalization in the context of database design?

Normalization is the process of organizing data in a database to reduce redundancy and improve data integrity. Denormalization is the opposite, where data is intentionally duplicated to optimize query performance.

#### 46: What is the ACID properties in the context of database transactions?

ACID stands for Atomicity, Consistency, Isolation, and Durability. These properties ensure that database transactions are reliable, and data remains consistent even in the face of errors.

#### 47: How can you prevent SQL injection in MySQL?

To prevent SQL injection, use parameterized queries or prepared statements. These techniques ensure that user inputs are treated as data and not executable SQL code.

#### 48: What is the difference between a LEFT JOIN and a RIGHT JOIN in MySQL?

A LEFT JOIN retrieves all records from the left table and matching records from the right table, while a RIGHT JOIN retrieves all records from the right table and matching records from the left table.

#### 49: What is a stored procedure in MySQL?

A stored procedure is a set of SQL statements that can be executed as a single unit. It is stored in the database and can be called multiple times with different parameters.

#### 50: How do you export data from a MySQL table to a CSV file?

You can export data to a CSV file using the SELECT...INTO OUTFILE statement or by using MySQL client utilities like mysqldump with the --tab option.

### Interview Questions for Laravel:

#### 0: Explain the MVC architecture and how Laravel implements it.

Laravel follows the Model-View-Controller (MVC) architecture. Models represent the data, Views display it, and Controllers handle user requests and manage the flow between Models and Views.

#### 1: What is Eloquent ORM, and how does it work in Laravel?

Eloquent is Laravel's ORM (Object-Relational Mapping). It allows you to work with databases using object-oriented syntax, making database interactions more convenient.

#### 2: What is Blade templating in Laravel?

Blade is Laravel's templating engine. It simplifies the creation of views by allowing you to write clean, readable template files with dynamic content.

#### 3: How does routing work in Laravel, and what are the different types of routes?

Laravel's routing maps URLs to controller actions. It includes route definition and handling for various HTTP request methods. There are named, resource, and wildcard routes, among others.

#### 4: What are middleware in Laravel, and how are they used?

Middleware is code that filters HTTP requests entering your application. It can be used for authentication, logging, CORS, and more.

#### 5: Explain dependency injection in Laravel.

Dependency injection is a technique used in Laravel to resolve dependencies for classes and functions. The Laravel service container manages this, making it easy to inject dependencies.

#### 6: What are Laravel migrations, and why are they important?

Migrations are version control for your database schema. They allow you to modify the database structure using PHP code and can be rolled back.

#### 7: How does authentication and authorization work in Laravel?

Laravel provides built-in tools for user authentication and role-based authorization. It's handled through middleware and policies.

#### 8: What are seeders in Laravel?

Seeders are used to populate database tables with sample data. They help in database testing and development.

#### 9: What are factories in Laravel?

Factories are used to generate fake data for testing and seeding databases. They are particularly useful for testing.

#### 10: How to implement soft delete in Laravel?

Soft delete is a feature in Laravel that allows you to mark records as deleted without actually removing them from the database. It's useful for data retention and recovery.

#### 11: What are Models?

In Laravel, Models represent the data structure and business logic of your application's database tables. They are used to interact with the database.

#### 12: What are Relationships in Laravel?

Relationships in Laravel define how different models are related to each other in terms of database associations, such as one-to-one, one-to-many, and many-to-many.

#### 13: What is Eloquent in Laravel?

Eloquent is Laravel's ORM (Object-Relational Mapping) system that enables you to work with databases using object-oriented syntax and models.

#### 14: What is throttling and how to implement it in Laravel?

Throttling is a rate-limiting mechanism used to control the number of requests a user can make in a specified time frame. It can be implemented in Laravel using middleware or route middleware to protect routes.

#### 15: What are facades?

Facades in Laravel provide a simple and consistent interface to various services in the application. They offer an easy way to access services like the database, caching, and more.

#### 16: What are Events in Laravel?

Events in Laravel are used to announce and listen for specific events in your application. They are useful for decoupling components and responding to various activities.

#### 17: What is Localization in Laravel?

Localization in Laravel is the process of translating your application's content into multiple languages. It allows your application to be used by speakers of different languages.

#### 18: What are Requests in Laravel?

Requests in Laravel handle HTTP requests. They provide validation, authorization, and input handling for your application.

#### 19: How to do request validation in Laravel?

Request validation in Laravel is done by creating request classes. These classes define the rules for validating incoming HTTP requests.

#### 20: What is a Service Container in Laravel?

The Service Container in Laravel is a powerful tool for managing class dependencies and performing dependency injection. It's used to resolve, bind, and manage class instances in the application.

#### 21: What is a Service Provider?

A Service Provider in Laravel is responsible for binding classes and services in the service container, registering components, and performing any bootstrapping needed for your application.

#### 22: What is the register and boot method in the Service Provider class?

The register method is used to bind services into the container, while the boot method is used for any additional actions needed after all service providers have been registered.

#### 23: How to define routes in Laravel?

Routes in Laravel are defined in the routes/web.php or routes/api.php files using the Route facade or closure functions.

#### 24: What are named routes?

Named routes in Laravel allow you to define a unique name for a route. This makes it easier to reference the route in your application, such as for URL generation or redirection.

#### 25: What are route groups?

Route groups in Laravel are used to group multiple routes together and apply common middleware or other attributes to those routes.

#### 26: What is Middleware and how to create one in Laravel?

Middleware in Laravel is a filter that can be applied to HTTP requests entering the application. You can create middleware using the make:middleware Artisan command.

#### 27: What are collections?

Collections in Laravel provide a convenient way to work with arrays of data. They offer various methods for data manipulation and transformation.

#### 28: What are contracts?

Contracts in Laravel define the methods that a class must implement, providing a way to ensure that classes follow specific interfaces.

#### 29: What are queues in Laravel?

Queues in Laravel enable delayed or background execution of tasks. They are used to handle time-consuming operations outside the regular request cycle.

#### 30: What are accessors and mutators?

Accessors are used to format and retrieve attributes from models. Mutators are used to set or modify attribute values before saving to the database in Laravel models.

#### 31: Explain the concept of eager loading in Laravel.

Eager loading is a technique in Laravel to load related models (e.g., for relationships like belongsTo or hasMany) to avoid the "N+1 query problem" and improve performance.

#### 32: How do you handle AJAX requests in Laravel?

Laravel provides built-in support for handling AJAX requests. You can use the Request object and return JSON or other responses in your controller methods.

#### 33: What are macros in Laravel, and how do you define them?

Macros in Laravel allow you to extend existing classes with additional methods. You can define them using the macro method provided by Laravel's Macroable trait.

#### 34:What are the common tools used to send emails in Laravel?

Common tools for sending emails in Laravel include the built-in mail driver, SMTP, and third-party services like Mailgun or SendGrid.

#### 35: Explain validations in laravel?

Validations in Laravel are rules and filters applied to user input to ensure it meets specific criteria, such as required fields, email format, and custom rules. Laravel provides a convenient way to define and enforce these rules in your application.

#### 36: How to install laravel via composer ?

To install Laravel via Composer, use the command: composer create-project --prefer-dist laravel/laravel project-name.

#### 37: Explain Laravel’s service container?

Laravel's service container is a tool for managing class dependencies and performing dependency injection. It automatically resolves and injects dependencies into your classes, facilitating better code organization and testability.

#### 38: How to enable query log in Laravel?

You can enable query log in Laravel by calling DB::enableQueryLog() before executing queries and then retrieve the logged queries using DB::getQueryLog().

#### 39: How to use custom table in Laravel Model?

In a Laravel Model, you can specify a custom table by setting the $table property to the desired table name. For example: protected $table = 'custom\_table';.

#### 40: List types of relationships available in Laravel Eloquent?

Laravel Eloquent supports relationships like belongsTo, hasOne, hasMany, belongsToMany, morphTo, and morphMany, among others.

#### 41: How to clear cache in Laravel?

You can clear the cache in Laravel using the php artisan cache:clear command.

#### 42: What do you understand by Unit testing?

Unit testing is a software testing technique in which individual components or units of code are tested in isolation to ensure they perform as expected. In Laravel, PHPUnit is often used for writing unit tests to validate the correctness of specific parts of the application.

#### 43: Explain the Service container and its advantages.

The Service container in Laravel is a powerful tool for managing class dependencies and performing dependency injection. Its advantages include facilitating cleaner, more maintainable code, improving testability, and enabling the resolution of dependencies automatically, making it easier to manage and extend your application.

#### 44: What is the use of PHP compact function?

The compact function in PHP is used to create an array from variables. In Laravel, it's often used to pass data to views by compacting variables into an array for use in Blade templates.

#### 45: What do you understand by ORM?

ORM stands for Object-Relational Mapping. It's a technique used to map database tables and records to objects in object-oriented programming languages, such as Laravel's Eloquent ORM. ORM simplifies database interactions by allowing developers to work with database data as if they were working with objects and classes.

#### 46: How can someone change the default database type in Laravel?

To change the default database type in Laravel, you can edit the DB\_CONNECTION value in the .env file to the desired database type (e.g., mysql, pgsql, sqlite, etc.).

#### 47: In which directory controllers are kept in Laravel?

Controllers in Laravel are typically stored in the app/Http/Controllers directory.

#### 48: What do you know about Closures in Laravel?

Closures are anonymous functions used in Laravel to define small, reusable code blocks. They are often used in routes and middleware to perform specific actions at runtime.

#### 49: How will you describe Fillable Attribute in a Laravel model?

The fillable attribute in a Laravel model is an array that defines which model attributes can be mass-assigned when using methods like create or update. It helps protect against overwriting sensitive attributes.

#### 50:How can we check the Laravel current version?

You can check the current Laravel version installed in your project by running the command php artisan --version in the command line.

#### 51: How can we get data between two dates using Query in Laravel?

You can retrieve data between two dates in Laravel using the whereBetween method in a query. For example: $data = Model::whereBetween('date\_column', [$startDate, $endDate])->get();.

#### 52: How do you do soft deletes?

In Laravel, soft deletes are implemented by adding the use SoftDeletes trait to your Eloquent model and defining the deleted\_at column in the corresponding database table. Soft deleted records are not removed from the database but marked as deleted by setting the deleted\_at timestamp. You can use the withTrashed and onlyTrashed methods to retrieve soft deleted records.

#### 53: How do you generate migrations?

You can generate a migration in Laravel using the artisan command php artisan make:migration. For example: php artisan make:migration create\_table\_name. This will create a new migration file in the database/migrations directory, where you can define the schema for your database table.

#### 54: How do you mock a static facade methods?

To mock a static facade method in Laravel for testing, you can use a package like Mockery or PHPUnit. You can create a mock object of the facade and define the expected behavior using these tools. Example:

use Illuminate\Support\Facades\Facade;

Facade::shouldReceive('staticFacadeMethod')->andReturn('mocked result');

#### 55: List some Aggregates methods provided by query builder in Laravel?

Some aggregate methods provided by Laravel's query builder include count(), sum(), avg(), min(), and max(). These methods allow you to perform calculations on data columns in your database tables.

#### 56: What is Closure in Laravel?

In Laravel, a Closure is an anonymous function that can be used as a callback or as a parameter for various methods. Closures are often used in routes, middleware, and as callback functions for various operations within the application.

#### 57: What is autoloading classes in PHP?

Autoloading classes in PHP is the process of automatically including the necessary class files when they are needed, without requiring manual require or include statements. Autoloading simplifies code organization and makes it more maintainable.

#### 58: What is CSRF protection and CSRF token?

CSRF (Cross-Site Request Forgery) protection is a security feature in Laravel that helps prevent malicious websites from making unauthorized requests on behalf of a user. Laravel generates and verifies CSRF tokens to ensure that the requests are coming from trusted sources and not from potentially harmful external sites.

#### 59: What template is used by the Laravel engine?

Laravel uses the Blade template engine for building dynamic views. Blade provides a clean and expressive way to write templates with features like control structures, template inheritance, and more.

#### 60: What is reverse Routing in Laravel?

Reverse routing in Laravel allows you to generate URLs for named routes. Instead of hardcoding URLs in your application, you can use route names to generate URLs dynamically. This makes it easier to maintain and update URLs throughout your application, especially when routes change.