PHP - Laravel - Migrations – Industry

Migrations are an essential part of Laravel, which is a PHP web application framework. It allows developers to manage database schema changes easily and consistently across different environments. With Laravel migrations, developers can create, modify, and delete database tables and columns using a version control system. This ensures that all database schema changes are tracked, and developers can easily roll back changes if necessary. In this article, we will discuss the basics of Laravel migrations and how to use them.

- Requirements for running migrations:

To run Laravel migrations, you need to have the following:

1. Laravel installed on your system
2. A database configured in Laravel configuration files
3. Permissions to create, modify, and delete database tables and columns

Artisan migration command:

The Artisan command-line interface in Laravel provides a simple way to manage migrations. To create a new migration file, run the following command:

php artisan make:migration create_users_table

This command will create a new migration file in the database/migrations directory. The name of the file will include a timestamp and the name of the migration. For example, 20220330123010_create_users_table.php.

Migration structure:

Each Laravel migration file has two methods: up() and down(). The up() method defines the changes that need to be made to the database schema, while the down() method defines how to revert those changes.

How to create a table using a migration:

To create a new table using a migration, you can use the Schema facade in Laravel. For example, to create a users table with id, name, and email columns, use the following code in the up**()** method:

```
public function up()
{
   Schema::create('users', function (Blueprint $table) {
      $table->id();
      $table->string('name');
      $table->string('email')->unique();
      $table->timestamps();
   });
}
```

- **Laravel migration rollback:**

  If you need to revert a migration, you can use the following Artisan command:

  php artisan migrate:rollback

  This command will revert the last batch of migrations. If you need to rollback multiple batches, use the --step option:

  php artisan migrate:rollback --step=3

  This command will rollback the last three batches of migrations.

- **Database Seeding:**

  Database seeding is the process of populating a database with sample data. Laravel provides a simple way to seed a database using seeders. To create a new seeder, use the following Artisan command:

php artisan make:seeder UsersTableSeeder

This command will create a new seeder file in the **database/seeders** directory. To run a seeder, use the following Artisan command:

php artisan db:seed

This command will run all the seeders in the database/seeders directory.

Migrations for our project database:

When working on a Laravel project, you will typically create several migration files to manage database schema changes. To run all the migrations, use the following Artisan command:

php artisan migrate

This command will run all the migrations in the database/migrations directory.

Pagination:

Laravel provides a simple way to paginate database results. To paginate a query result, use the paginate() method:

$users = DB::table('users')->paginate(10);

This code will retrieve 10 records from the users table and paginate the results. To display the pagination links in a Laravel view, use the following code:

{{ $users->links() }}

This code will display