**• How to do config database in laravel.**

Laravel uses the PDO (PHP Data Objects) extension to connect to databases. In order to configure your database in Laravel, you will need to update the config/database.php file with your database credentials.

1. First, ensure you have a database set up and have the credentials (username, password, hostname, etc.) ready. Laravel supports several database systems, including MySQL, PostgreSQL, and SQLite.
2. Open the config/database.php file and find the connections array. This array contains all of the available connection configurations.
3. Choose a connection type from the array (e.g., mysql, sqlite, pgsql, etc.) and update the corresponding configuration options with your database credentials. For example, to configure a MySQL database, you might update the Default Database Connection Name:

'default' => env('DB\_CONNECTION', 'mysql'),

1. Open the .env file in the root directory of your Laravel project. Set the DB\_CONNECTION variable in this file to match the connection type you chose in the previous step. For example, if you are using a MySQL database, you would set the DB\_CONNECTION variable to mysql.
2. DB\_CONNECTION=mysql
3. DB\_HOST=127.0.0.1
4. DB\_PORT=3306
5. DB\_DATABASE=laravel
6. DB\_USERNAME=root

DB\_PASSWORD=

1. Set the DB\_HOST variable to the hostname of your database server. For example, if you are using a local development server, you might set this to localhost.
2. Set the DB\_DATABASE variable to the name of your database.
3. Set the DB\_USERNAME and DB\_PASSWORD variables to the username and password for your database.
4. If you are using a MySQL or PostgreSQL database, you may also need to set the DB\_PORT variable in your .env file to the correct port number for your database server.
5. If you are using a MySQL database, you may also need to set the DB\_SOCKET variable in your .env file if your MySQL server uses a socket other than the default.
6. Save the .env file and exit.

Once you have configured your database connection, you can use the Laravel query builder or Eloquent ORM to perform CRUD operations on your database.

For example, to retrieve all rows from a table called users, you could use the following code:

Example:

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

• Call MySQLi Store Procedure from Laravel.

**• Apply Curd Operation through Query Builder for Employee Management.**

* use Illuminate\Support\Facades\DB;

**// fetch all data**

function view\_employee()

{

$data= DB::table(employees)->get();

return view('view employee',['alldata'=>$data]);

}

**// fetch where data**

function view\_employee()

{

$data= DB::table(employees)

->where(‘eid,10)

->get();

return view(‘view\_employee ',['alldata'=>$data]);

}

**• Create All Migration for Employee management**