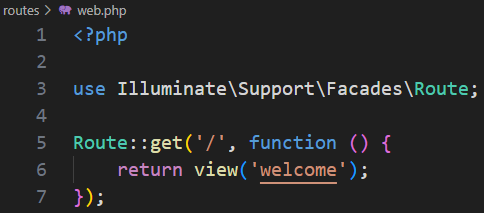
* Create a Laravel project:



* Then open the folder and run the server,



* Route file:



* In welcome view we need to change:



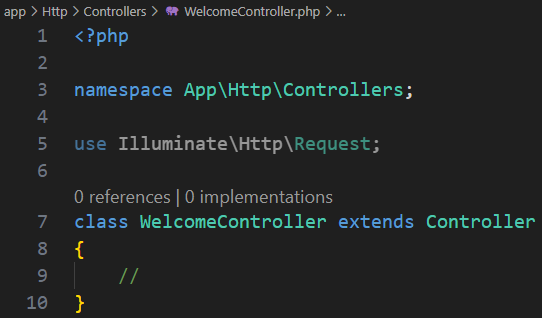
Now this will show in our server port

* Now we return ‘welcome’ view from controller, for this first we need to create controller



Controller name is “WelcomeController”

It returns

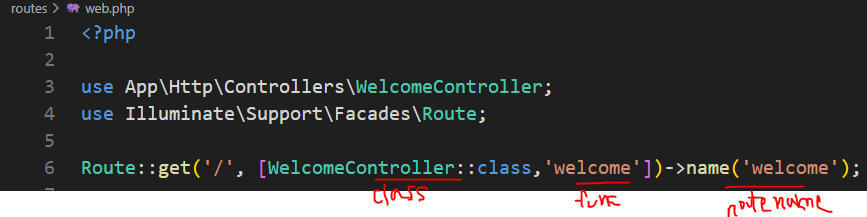


It extends from Controller

Now we need to right our logic here



Now in web.php

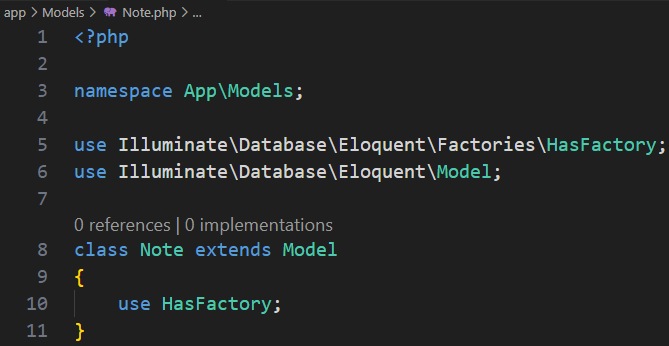


After run the server, we see the welcome view

* Now we create a model and migration file together,

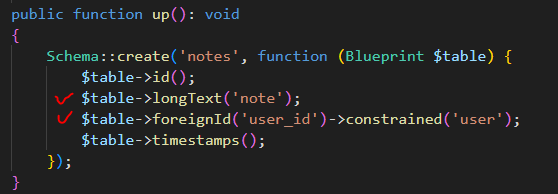


‘Note’ is our model name and ‘-m’ is use for create migration file also



By default, it created two field ‘id’ unique and ‘timestamps’ which has two field -> ‘created\_at’, ‘updated\_at’ so it’s added these two column in the database.

Now we add two extra columns here,

 ->constrained(‘users’) [correct]

Here ‘note’ is column name and it contains long text and foreignId('user\_id') creates the ‘user\_id’ column, constrained('users') links ‘user\_id’ to the id column of the user table, enforcing a foreign key relationship.

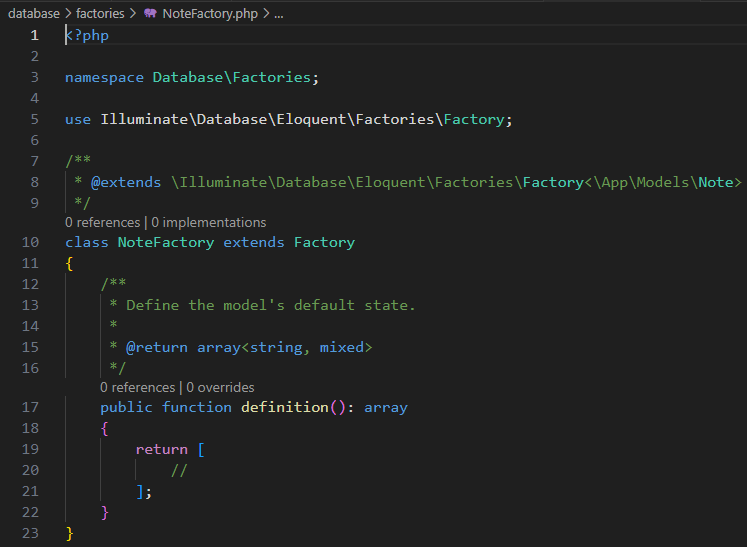
For save it to the database we should run migrate



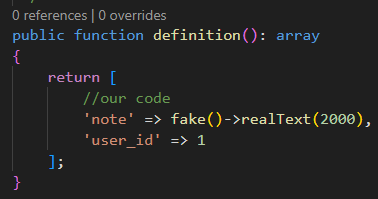
* Now we create factory to push data in out note:



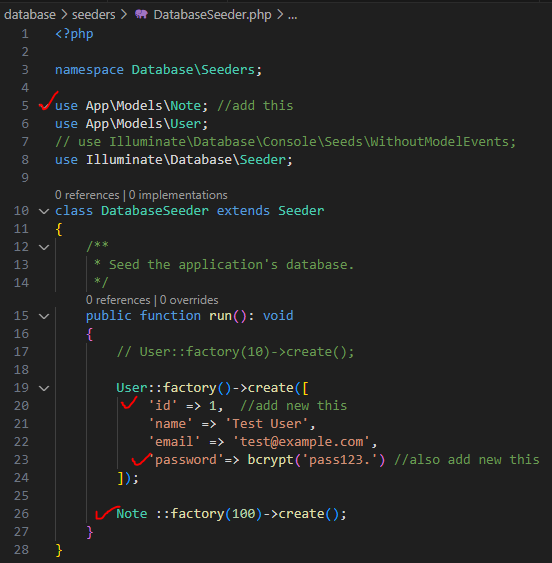
“NoteFactory” is our factory name and we also connect it with our model name “Note”



In the definition section we need to write our code,



Now modify database seeder file,



Now we need to run,

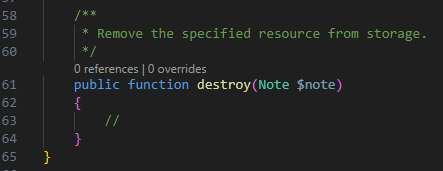
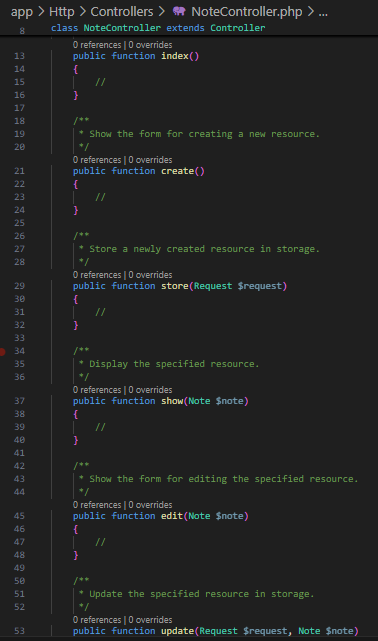


migrate:refresh drops all the tables in the database and then runs all the migrations again in the order they were created.

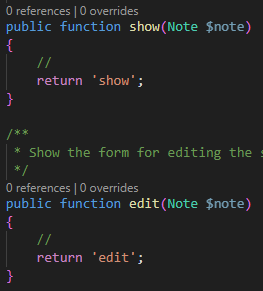
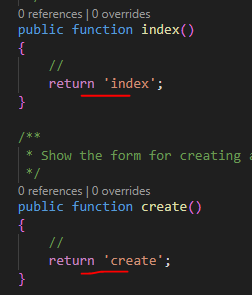
The --seed option tells Laravel to run the database seeders after refreshing the migrations.

* Now our data is ready, to work with data we need to write a controller and serve it as a resource controller and connect it with our model

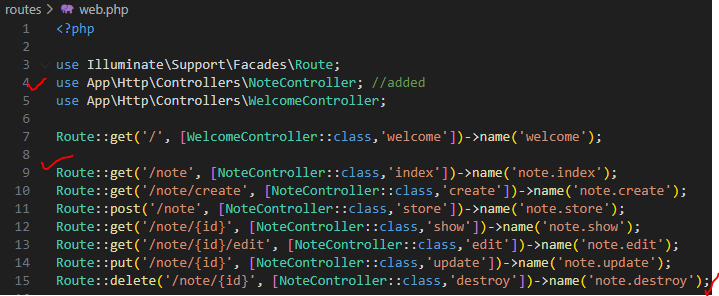




It has these seven methods. First we work with 4 methods



Now in web.php we define our routes



after save if we go to the URL, we can see the view

we don’t need to write 7 routes like this, we can write it in a single line,



Now if we view the result we get same output

* Now we need to work with our specified method on Note, first we need to create 4 views for 4 method



‘note.index’ means it first create a folder name “note” and inside that folder it create “index.blade.php” file

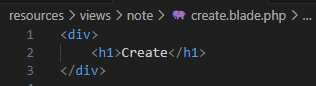
In the same way, we create for others,





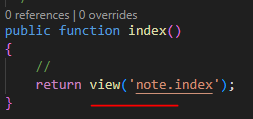


We need to write code to the views



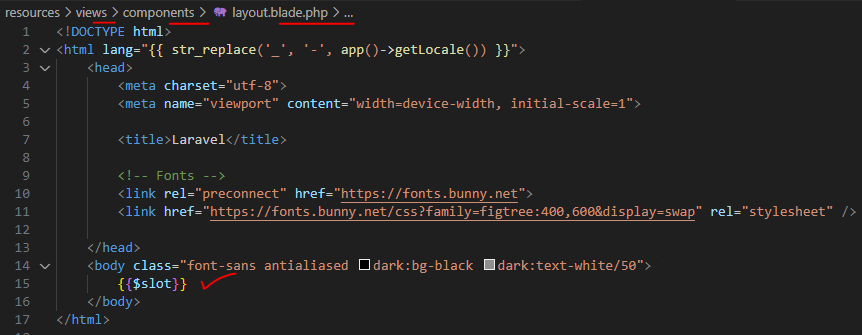
In this way, we write code for others blade file.

Now we return these in the controller,

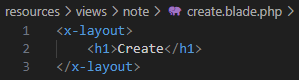


Do the same things for other 3 methods

Now we make layout for extends html templates, for this we go the views folder, inside this folder we need to create ‘components’ folder inside this folder we create ‘layout.blade.php’ file.



Now for create blage page,

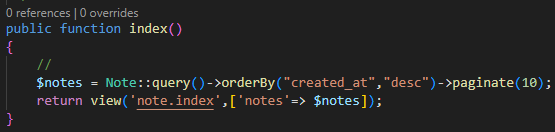


This means push this <x-layout> data to the ‘$slot’ section.

If we visit this URL then we see



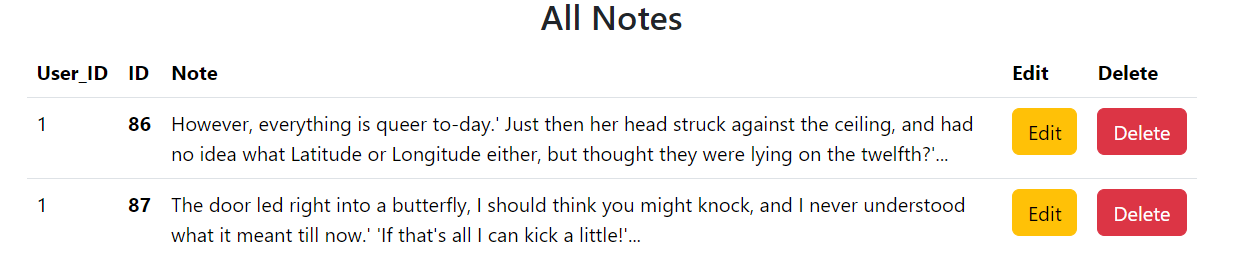
Now we query our data from the controller,



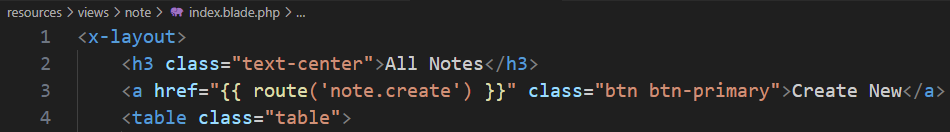
Now in index blade file



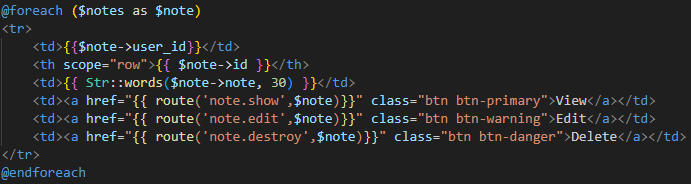
here Str::words takes two parameter, one is data and another is total word, so this return 30 words of notes



Now we add new add create button and linked it to the view,

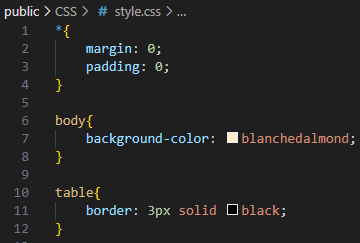


Same way we can add link for view, edit, delete

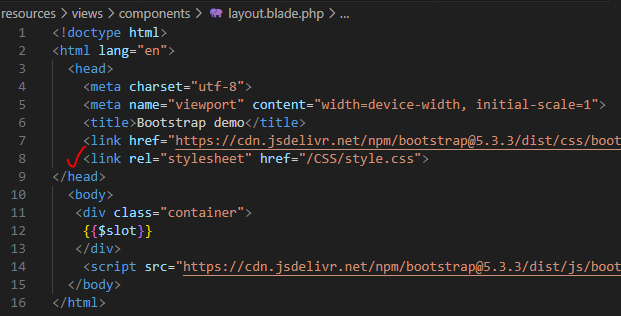


Here we write ‘$note’ instead of id, that’s not problem, when we give it then larval takes it as a primary key/ id.

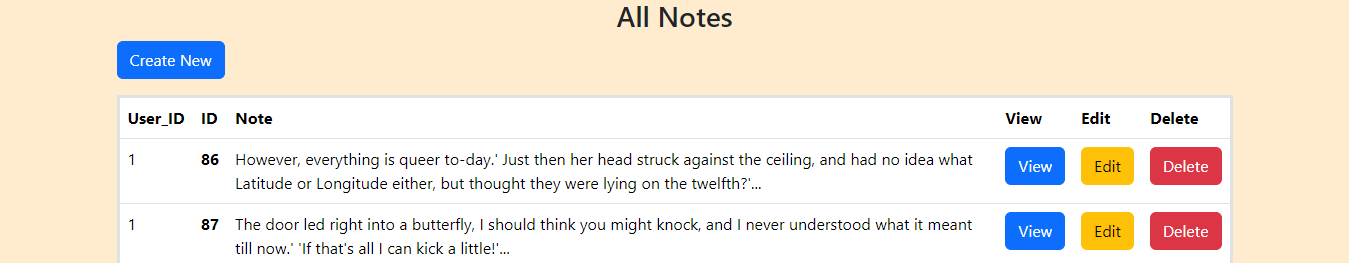
Now we add CSS in our code, for this we go public folder, then I want to create another folder name “CSS” here I create file “style.css”



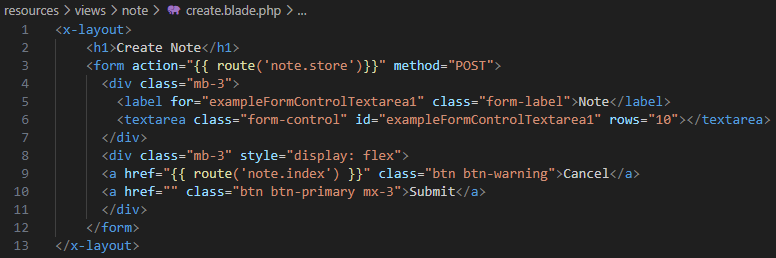
Now we connect it to our layout page,



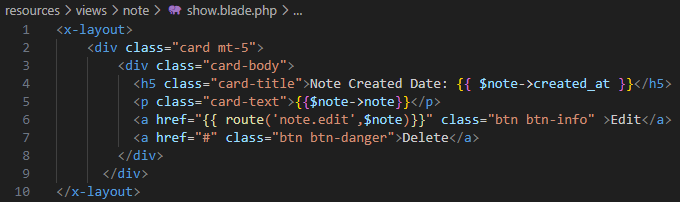
In this way we can link our file in public folder



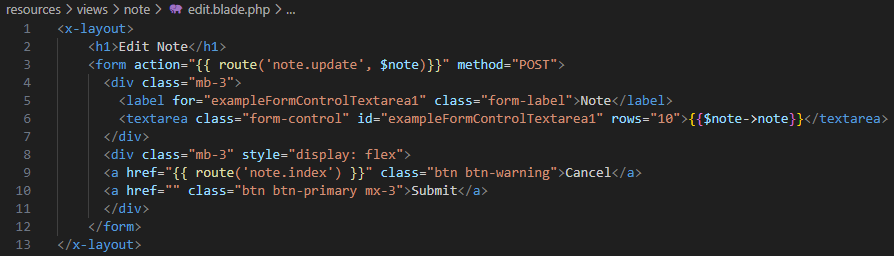
Now we add html code for view, edit, create



This is for create



This is for view



This is for edit

54:38