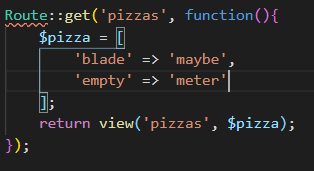
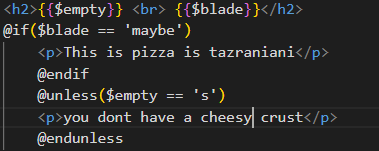
Uri = a string of characters used to identify a resource on a computer network, of which the best known type is the web address or URL.

First parameter of Route::get, you define the URL path of the php file.   
The return views first parameter (‘pizzas’) is the filename of the php file. The second parameter is for data ($pizza)



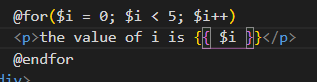
In this case, $pizza is an array. You can access the arrays values with {{$array\_name}} in other files.

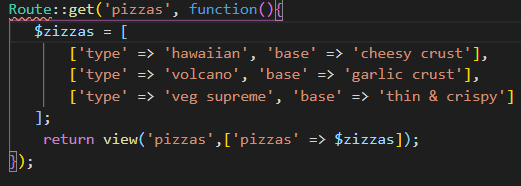
Here $empty => ‘meter’  
$blade => ‘maybe’.

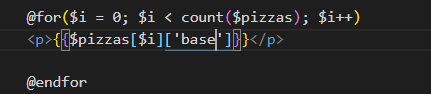
You can also use if() statements.   
Blade doesn’t follow the traditional way of loops.

If you want to declare a variable, use @if and end with @endif.

@unless($condition) means whatever is the condition is **NOT** true, the code below will run.   
So in this case $empty = ‘meter’. But in the @unless statement we put if empty = ‘s’, which is not true, echo out ‘you don’t have a cheesy crust’. End the condition with @endunless.

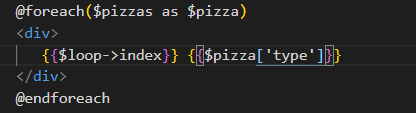
You can use normal php vanilla code if you declare your code inside the @php and @endphp tags.  
  
To make a for loop you declare your code within @for and @endfor  
This puts out:   
the value of I is 1 , 2, 3 etc.

To pass more advanced array data you need to do it like shown below.  
So the ‘pizzas’ in the array in the View is eventually the variable that will be used in the other page. The arrow near the ‘pizzas’ is pointing towards the $zizzas variable with is the array on top of the view.



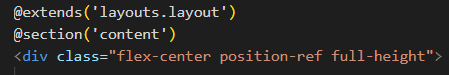
So what this loop will do is echo out the base values of the   
$zizzas array. As you can see, you can get the array values by declaring them in brackets.

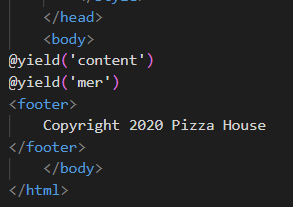
The @foreach loop cycles through the $pizzas array and stores is in the $pizza variable.

You need to define a key to the $pizza variable or else you will get an error.  
The loop index simply gives the output data numbers.

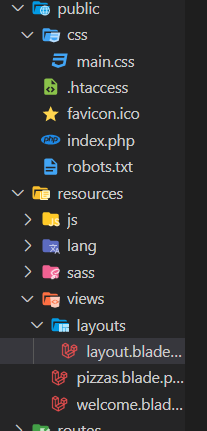
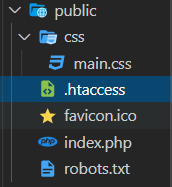
Use @yield(‘@section\_name’) to add a section. This is how to define a section:

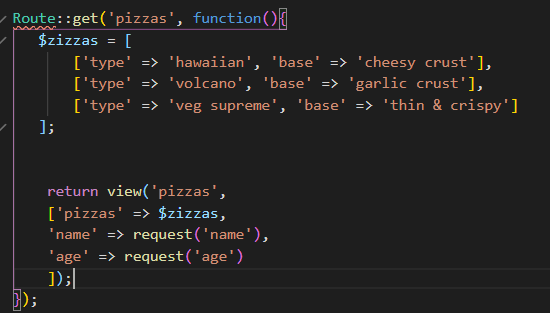
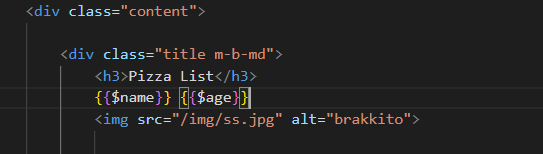
The section has the name ‘mer’  
this section ends with @endsection.

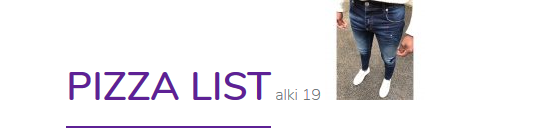
For this to work, you need to add @extends(‘file.location\_name’). To go through directories, you use . instead of /

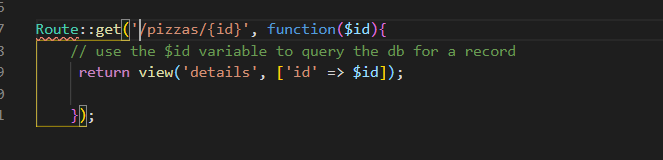
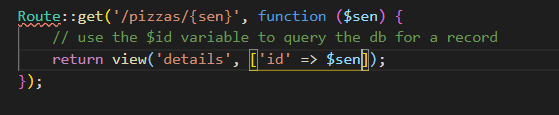
If you go over to the page where you want this section to be added to, simply use @yield() to add the section  


In this case the section is added in the body of this file.

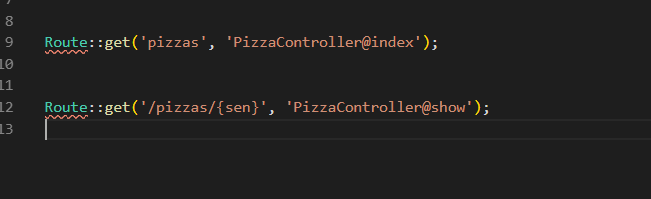
Everything inside the public folder (the folder where all the production files go) is made out of root level direction.  
So if you do not add /public/ if you are redirecting.   


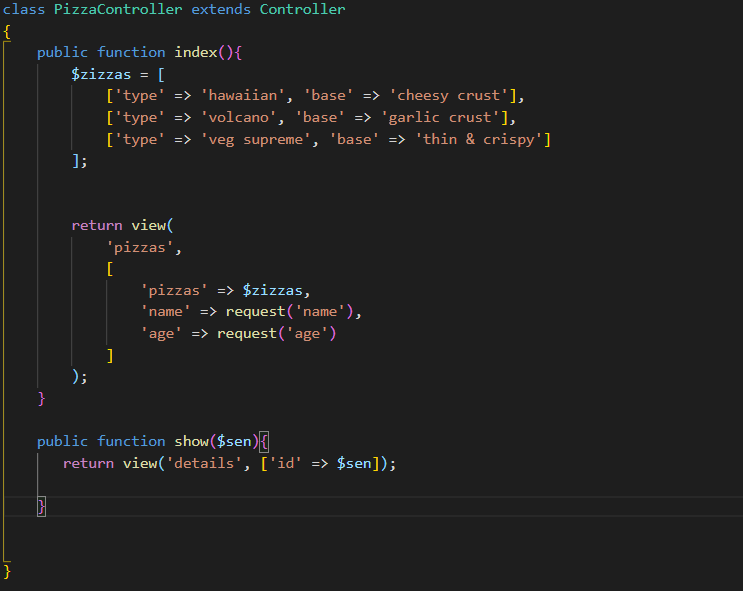
To get the data out of the URL use the request() method in web.php the following will output an echo  




To include routing in your URL, use the following script:  
The Route::get() method has an id.  
the function($id) gets passed in the Route() id.  
and the view(‘details’) = a new php file.   
and in the array, the first ‘id’ is the variable that you are going to use to declare it into the details.blade.php page. The second id is where the url data is stored.   
This still loades the page!

There is also another way to declare routes.

 In this example, PizzaController Is the controller located in:  
app/Http/Controllers. @show is the action (method) located in the controllers file. The controller file is NOT a blade file.



In this case, in the route you still declare the argument  
in the route section of the Route::get() method.

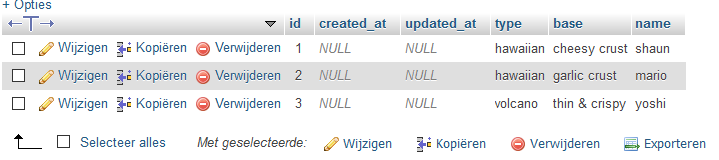
Models

|  |
| --- |
| $pizzas = Pizza:all(); = Gets all data from the pizza table. |

|  |
| --- |
| $pizza = Pizza:orderBy(‘\*name of row/value\*’, ‘desc, asc’)->get(). Without the get() it won’t return anything. |

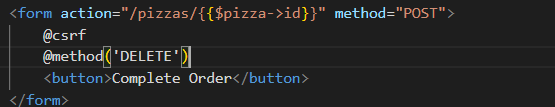
|  |
| --- |
| $pizza = Pizza:where(‘\*name of row\*’, \*value of data\*)->get(). |

|  |
| --- |
| $pizzas = Pizza::latest()->get(); = This will get the latest data and so on ascending.  So if you have a date for example: 14 juni 2020, 21 april, 2014, 21 januari, 2002.  It will put it like below: 1. 14 Juni 2020 2. 21 april 2014 3. 21 januari 2002.  So from recent date descending. |



Remove records

Use @csrf to prevent crosside scripting. Add @csrf to EVERY FORM

To overwrite a method use @method and then post,get,delete,update to overwrite the original method declared in the form brackets

These are all the available routes in Laravel:

Route::get();

Route::post();

Route::patch();

Route::put();

Route::delete();

Route::any();

You can register multiple HTTP requests with the Route::match method.  
The first parameter is the HTTP routes and the second parameter is the route to the file you want to get a response from



The Route::any method is used to detect all HTTP requests. The first parameter in this case is the route to the file.

For redirecting to another route, you can use Route::redirect.  
  
  
The default value of the third parameter is 302. But you may customize this. (For whatever reason)



You may define as many route parameters as required by your route:



With the where() method, you can declare regular expression to match your route action:  
  


You can define a regular expression in a route in the RouteServiceProvider.php file.   
To do that you simply use the Route::pattern();   
  


When you do this, (in this example) the route::pattern method will be applied to every routing using that first parameter (in this case the ‘id’);