**Web programozás**

**13.Laborgyakorlat**

**Támakör:**

1. Composer-alapú alkalmazás készítés
2. LaravelCRUD alaklamazás Tailwind stílussal
3. Laravel Livewire bevezető

**Feladatok:**

1. **Composer-alapú alkalmazás lépései**
   1. Hozzunk létre egy PHP projektet, majd nyissunk konzolt a gyökér mappában
   2. A *composer init* paranccsal hozzunk létre egy *composer.json* fájlt (ezt manuálisan is lehet)
   3. A composer require „csomag” paranccsal pedig telepítsük a szükséges külső csomagot (pld. composer require amenadiel/jpgraph)
      1. A *composer.json* fáljba bekerül egy bejegyzés
      2. A *vendor* mappába letöltődik a külső csomag
   4. A telepített csomagot teszteljük (fontos a vendor mappa importálása)

Példa:

require\_once '../vendor/autoload.php';

use Amenadiel\JpGraph\Graph;

use Amenadiel\JpGraph\Plot;

// Create the Pie Graph.

$graph = new Graph\PieGraph(350, 250);

$graph->title->Set("A Simple Pie Plot");

$graph->SetBox(true);

$data = array(40, 21, 17, 14, 23);

$p1 = new Plot\PiePlot($data);

$p1->ShowBorder();

$p1->SetColor('black');

$p1->SetSliceColors(array('#1E90FF', '#2E8B57', '#ADFF2F', '#DC143C', '#BA55D3'));

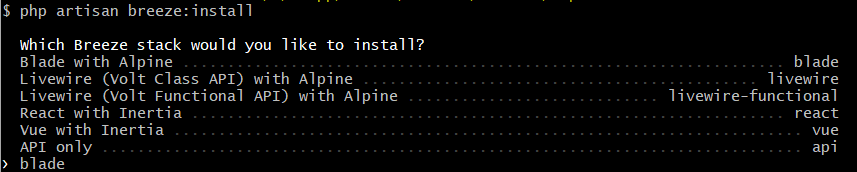
$graph->Add($p1);

$graph->Stroke();

1. **Install Tailwind for design**
   1. Breeze telepítés

composer require laravel/breeze

php artisan breeze:install



php artisan migrate:fresh --seed

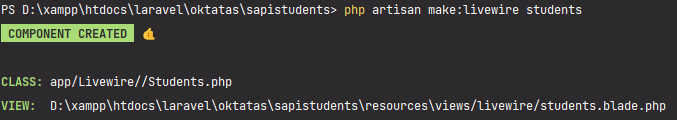
npm run dev

1. **Blade fájlok frissítése**
   1. Az *index.blade.php*, *create.blade.php*, *edit.blade.php* fájlok felülírása a megadott változatokkal (*tailwind* formázásokat tartalmaznak)
   2. Hozzunk létre egy *show.blade.php* fájlt egy hallgató megjelenítésére
   3. Az *edit.blade.php* fájlt módosítsuk, úgy hogy a „grade” mezőt is lehessen megváltoztatni
   4. A hallgatói lista legyen lapozható
2. **Livevewire CRUD alkalmazás**
   1. Livewire telepítés

composer require livewire/livewire

* 1. Student komponens létrehozása (Livewire/Students.php, views/livewire/students.blade.php)

php artisan make:livewire student



* 1. Hallgatók lekérése (Student.php)

public function render()

{

$this->students = Students::all();

return view('livewire.student');

}

* 1. Hallgatók megjelenítése (student.blade.php)

<div class="table-responsive">

<table class="table">

<thead>

<tr>

<th>Id</th>

<th>Name</th>

<th>Email</th>

<th>Grade</th>

<th>Action</th>

</tr>

</thead>

<tbody>

@if (count($students) > 0)

@foreach ($students as $student)

<tr>

<td>{{$student->id}}</td>

<td>{{$student->name}}</td>

<td>{{$student->email}}</td>

<td>{{$student->grade}}</td>

<td>

<button wire:click="edit({{$student->id}})" class="btn btn-primary btn-sm">Edit</button>

<button wire:click="delete({{$student->id}})" class="btn btn-danger btn-sm">Delete</button>

</td>

</tr>

@endforeach

@else

<tr>

<td colspan="3" align="center">

No Students Found.

</td>

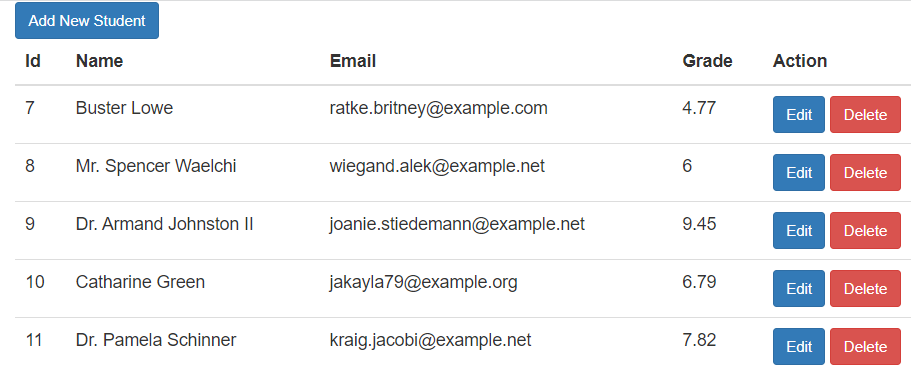
</tr>

@endif

</tbody>

</table>

</div>



* 1. Hozzunk létre egy [home.blade.php](http://home.blade.php) fájlt a *views* mappában, majd hívjuk be a *student* komponenst.

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<meta http-equiv="X-UA-Compatible" content="ie=edge">

<title>Livewire Crud</title>

<!-- Latest compiled and minified CSS -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/css/bootstrap.min.css">

<!-- jQuery library -->

<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.7.1/jquery.min.js"></script>

<!-- Latest compiled JavaScript -->

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.4.1/js/bootstrap.min.js"></script>

**@livewireStyles**

</head>

<body>

<div class="container">

<div class="row justify-content-center mt-3">

**@livewire('student')**

</div>

</div>

**@livewireScripts**

</body>

</html>

* 1. Implementáljuk a *Create*, *Update*, *Delete* műveleteket.