Ministério da Educação
Secretaria de Educação Profissional e Tecnológica
Instituto Federal Catarinense
Campus Videira

JONAS FELICETTI

RELATÓRIO DESCRITIVO
SOBRE O PROJETO FINAL DA MATÉRIA DE LPGII

Videira
2022

JONAS FELICETTI

RELATÓRIO DESCRITIVO
SOBRE O PROJETO FINAL DA MATÉRIA DE LPGII

Trabalho apresentado para a disciplina de LPGII do curso de Bacharelado em Ciência da Computação do Instituto Federal Catarinense – Campus Videira
Orientador(a): Fabricio Bizotto

Videira
2022

Introdução

A proposta do trabalho era desenvolver uma aplicação web integrada com um banco de dados com pelo menos 3 tabelas, possuísse funcionalidade de login, utilizasse de métodos GET e POST e também sessions. Para este fim resolvi desenvolver uma aplicação,através do framework Laravel, framework este que nunca tinha utilizado, para gerenciamento de ACC's(atividades curriculares complementares), podendo ser realizado o login ou o cadastro de um novo usuário, nas primeiras telas, e, posteriormente, o cadastro e consulta de ACC's pertencentes ao usuário logado. A primeira ideia era utilizar diferentes 'roles' para os usuários, tais como, aluno e servidor, para que pudesse ser realizada a aprovação das ACC's, mas, infelizmente, não conseguir realizar esta implementação, tornando assim a tabela 'roles' apenas uma chave estrangeira da tabela 'users'.

Desenvolvimento

O primeiro passo para o desenvolvimento da aplicação foi a criação das rotas:

```
Route::controller(\App\Http\Controllers\HomeController::class)
    ->prefix('/home')
    ->middleware(Login::class)
    ->group(function (){
        Route::get('/', 'show')->name('site.home');
    });

    Route::controller(\App\Http\Controllers\ActivityController::class)
    ->prefix('/atividade')
    ->middleware(Login::class)
    ->group(function (){
        Route::get('/', 'create')->name('site.atividade.create');
        Route::post('/', 'form')->name('site.atividade.create.form');
        Route::get('/show', 'show')->name('site.atividade.show');
    });

Route::controller(\App\Http\Controllers\LoginController::class)
    ->prefix('/login')
    ->group(function (){
        Route::get('/', 'index')->name('site.login.get');
        Route::post('/', 'form')->name('site.login.post');
        Route::get('/logout', 'logout')->name('site.logout');
    });

Route::controller(\App\Http\Controllers\SinginController::class)
    ->prefix('/singin')
    ->group(function (){
        Route::get('/', 'index')->name('site.singin.get');
        Route::post('/', 'form')->name('site.singin.post');
    });
```

Que atraves de metodos GET e POST chamavam os seus respectivos Controllers:

```php
class LoginController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('Login.index');
    }

    public function form(Request $request)
    {
        if (Auth::attempt(['user' => $request->user, 'password' => $request->password])) {
            $request->session()->regenerate();
            return redirect()->route('site.home');
        }else{
            return redirect()->route('site.login.get');
        }
    }

    /**
 * Log the user out of the application.
 *
 * @param  \Illuminate\Http\Request  $request
 * @return \Illuminate\Http\Response
 */
    public function logout(Request $request)
    {
        Auth::logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        return redirect()->route('site.login.get');
    }
}
```

```php
class SinginController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('Singin.index');
    }


    public function form(Request $request){

        // dd($request->all());
        $user = [
            'user' => $request->user,
            'role_id' => 1,
            'name' => $request->name,
            'password' => Hash::make($request->password)
        ];
        // dd($user);
        User::create($user);
        return redirect()->route('site.login.get');
    }
}
```

```php
class HomeController extends Controller
{
    /**
     * Display a listing of the resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        return view('Home.index');
    }

    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */
    public function show()
    {

        return view('Home.index', ['userName'=>Auth::user()->name, 'activities'=>Activity::where('user_id', Auth::user()->id)->get()]);
    }

}
```

```php
class ActivityController extends Controller
{

    /**
     * Show the form for creating a new resource.
     *
     * @return \Illuminate\Http\Response
     */
    public function create()
    {
        return view('Activity.create', ['userName'=>Auth::user()->name]);
    }


    public function form(Request $request){
        $activity = [
            'user_id' => Auth::user()->id,
            'activity' => $request->activity,
            'institution' => $request->institution,
            'time' => $request->time,
            'description' => $request->description,
        ];
        Activity::create($activity);
        return redirect()->route('site.home');
    }
}
```

Após ter estabelecido as rotas e criado os Controllers foram criadas as Views, esta foi sem dúvidas uma das partes mais demoradas do projeto, tornando óbvia minha falta de prática e conhecimento quando estou tratando do front end, característica esta que pretendo melhorar o quanto antes.

Após as views estarem criadas me deparei com a funcionalidade das migrations, que, junto das Models, já realizam a criação do banco de dados e todos os CRUD's através de funções nativas do framework, foi um processo demorado por conta da falta de conhecimento sobre estas funções e de como o Laravel realizava tudo de forma automática, resultando em poucas horas codando, mas diversas horas pesquisando, vendo videos e lendo a própria documentação, que por sinal é bem completa e de fácil entendimento. após isso veio o próximo obstáculo, entender como aplicar e como funcionava a validação de usuários no Laravel, obstáculo esse que, novamente, foi superado por mais pesquisa do que código, o Laravel já trata as relações de usuário através de sua classe 'Auth', só foi necessário entender como utilizá la corretamente, junto da função 'hash' para criptografar a senha, depois de tudo

isso só restava entender como seria mostrado na tela do usuário as informações contidas no banco e, novamente me repetindo, mais estudo do que código, por conta do framework já ter as funções nativas para este tipo de funcionalidade.

Conclusão

Sinceramente, sei que foi irresponsabilidade da minha parte deixar este projeto para última hora, mas ao mesmo tempo foi muito empolgante, ter q aprender uma tecnologia do zero e implementá-la em 72h, varando 2 noites a fio pra conseguir desenvolver a aplicação. gostei muito de trabalhar com o laravel, principalmente com a parte de migrations e estudar sobre as seeds e factorys, que já criam e populam o banco através de código PHP. Com certeza vou continuar estudando este framework e com toda certeza vou olhar este código daqui a algum tempo e encontrar diversos erros e afins, mas foi muito satisfatório concluir este desafio, principalmente pela pressão de tempo.

Qualquer dúvida sobre o código estou à disposição para prestar explicações, desde já agradeço a paciência, mesmo comigo entregando o trabalho depois do prazo.

Videira

2022