PROGRAMAÇÃO WEB USANDO O FRAMEWORK LARAVEL

APRESENTAÇÃO GERAL E BREVES CONSIDERAÇÕES

Aula 1

- **FORMADOR**
- **FORMANDOS**

Objetivos:

Ao final da formação, os participantes deverão ser capazes de:

1. Desenvolver sites dinâmicos, Profissionais e escaláveis usando Laravel.
2. Aplique boas práticas de segurança e organização de código em Laravel.

Introdução ao Desenvolvimento Web

- Conceitos de Frontend e Backend
- Apresentação do Laravel e suas vantagens
- Ferramentas:VSCode, Composer, Laragon, Laravel 11, MySQL.

Instalação e Configuração do ambiente de desenvolvimento

- Instalação do VSCode, Laragon, Composer e Laravel via Composer
- Estrutura de pastas e arquivos do Laravel
- Configuração do ambiente (.env)

  Acesse o site oficial do Laragon: https ://laragon .org
  Acesse o site oficial do Composer: https ://getcomposer .org

Introdução ao MVC (Model-View-Controller)

- O padrão MVC no Laravel
- Controladores: Criação e manipulação de dados
- Modelos: Comunicação com o banco de dados
- Views: Blade templates para renderização de interfaces

Rotas e Controladores

- Definição e configuração de rotas
- Agrupamento de rotas, middleware e prefixos
- Passagem de parâmetros nas rotas
- Controladores e métodos: criação e manipulação de dados nas visualizações

Conceitos de Frontend e Backend

- **Frontend (Cliente):** O frontend é uma parte do sistema que o usuário final interage diretamente. Ele envolve tudo relacionado à interface gráfica do usuário (GUI) e à experiência do usuário (UX). No desenvolvimento web, o frontend refere-se ao conteúdo exibido no navegador, incluindo:
    - **HTML:** Linguagem de marcação utilizada para estruturar o conteúdo de uma página.
    - **CSS:** Utilizado para estilizar e melhorar a aparência visual do conteúdo.
    - **JavaScript:** Linguagem de programação responsável por adicionar interatividade às páginas web (animações, validações de formulários, etc.).

O frontend é geralmente responsável por capturar as interações do usuário e enviá-las ao backend para processamento.

- **Backend (Servidor):** O backend é uma parte do sistema que não é visível para o usuário. Ele lida com toda a lógica de negócios, gerenciamento de banco de dados, autenticação de usuários e outras funções que são processadas no servidor. O backend recebe as comissões do frontend, processa os dados e retorna uma resposta respondida.

O backend funciona como o “cérebro” do sistema, executando as ações corretas com base nas interações feitas no frontend e retornando as respostas certas.

Apresentação do Laravel e suas Vantagens

O nome "Laravel" foi escolhido pelo seu criador, Taylor Otwell, como uma referência ao Castelo de LaraVelha ("LavraVelha"), uma cidade fictícia inspirada na arquitetura europeia medieval. Otwell queria um nome que soasse elegante e tivesse uma conotação de estrutura, força e sofisticação, características que ele desejava para o framework.

- **O que é o Laravel:** Laravel é um framework PHP open-source, baseado no padrão MVC (Model-View-Controller), que facilita o desenvolvimento de aplicações web robustas e escaláveis. Ele é amplamente utilizado pela sua simplicidade e elegante na escrita de código, além de oferecer uma vasta gama de funcionalidades já prontas para uso.
- **Vantagens do Laravel:**
    1. **Estrutura baseada no padrão MVC:** Laravel segue o padrão MVC, que separa a lógica da aplicação em três partes:
        - **Model (Modelo):** Representa os dados e interação com o banco de dados.
        - **View (Visão):** Responsável pela interface e pela forma como os dados são apresentados ao usuário.
        - **Controller (Controlador):** Controla a comunicação entre o Model e o View.
    2. **Eloquent ORM:** O Eloquent é o ORM (Object-Relational Mapping) embutido no Laravel, que facilita a interação com o banco de dados, permitindo realizar consultas e manipulação de dados utilizando uma sintaxe fluida e fácil de entender.
    3. **Blade Templating Engine:** O Laravel utiliza o Blade, um mecanismo de templates simples, mas poderoso, que permite criar layouts reutilizáveis para visualizações, facilitando a organização e a manutenção do frontend.
    4. **Rotas simples e intuitivas:** O sistema de rotas do Laravel é altamente flexível e permite criar rotas com facilidade, além de poder agrupar rotas, aplicar middlewares e definir intervalos dinâmicos.
    5. **Segurança:** O Laravel oferece várias funcionalidades de segurança embutidas, como proteção contra injeção de SQL, Cross-Site Scripting (XSS) e Cross-Site Request Forgery (CSRF). Ele também possui um sistema robusto de autenticação e autorização de usuários.
    6. **Facilidade no gerenciamento de dependências:** O Laravel faz uso do Composer, um gerenciador de pacotes que permite integrar bibliotecas e pacotes externos de maneira fácil e eficiente, mantendo o projeto modular e organizado.
    7. **Documentação e Comunidade:** O Laravel possui uma excelente documentação e uma comunidade ativa, o que facilita encontrar soluções para problemas

comuns, além de contar com uma grande variedade de tutoriais e pacotes prontos para uso.

Essas vantagens fazem do Laravel uma excelente escolha para o desenvolvimento web moderno, pois ele combina simplicidade com poder, permitindo o desenvolvimento rápido e eficiente de aplicações robustas e escaláveis.

Definição e Configuração de Rotas

As rotas no Laravel são responsáveis por direcionar as operações que chegam ao servidor para os controladores ou ações corretas. Quando um usuário acessa uma URL, o sistema de rotas define qual função ou driver será executado para processar essa solicitação.

- **Definição básica de uma rota:** As rotas são definidas no arquivo routes/web.php para rotas web (que retornam visualizações). Exemplo básico de rota:

```
Route::get('/cursos', function () {
  return 'Listando todos os cursos';
});
```

Isso define uma rota que responde à URL /curso se retorna o texto "Listando todos os cursos".

- **Rotas para controladores:** Rotas também podem ser associadas diretamente a métodos de controladores:

```
Route::get('/cursos', [CourseController::class, 'index']);
```

Agrupamento de Rotas, Middleware e Prefixos

O agrupamento de rotas permite que você aplique configurações em várias rotas ao mesmo tempo, como aplicar um middleware ou definir um prefixo comum para uma série de rotas.

- **Agrupamento com middleware:** Um middleware é um filtro que pode ser aplicado às rotas para verificar a autenticação, permissões ou outras condições antes de permitir que uma solicitação seja processada.

Exemplo de agrupamento com middleware de autenticação:

```
Route::middleware(['auth'])->group(function () {
  Route::get('/perfil', [UserController::class, 'show']);
  Route::get('/configuracoes', [SettingsController::class, 'index']);
});
```

Isso aplica o middleware autha todas as rotas dentro do grupo, garantindo que apenas usuários autenticados possam acessá-los.

- **Agrupamento com prefixo:** Um prefixo pode ser usado para definir um segmento comum de URL para um grupo de rotas. Por exemplo, as rotas administrativas podem ter o prefixo /admin:

```
Route::prefix('admin')->group(function () {
```

```
    Route::get('/usuarios', [AdminController::class, 'users']);

    Route::get('/relatorios', [AdminController::class, 'reports']);
});
```

Aqui, as rotas acessíveis serão /admin/usuariose /admin/relatorios.

Passagem de Parâmetros nas Rotas

O Laravel permite a passagem de parâmetros diretamente nas URLs, que podem ser usadas dentro dos drivers para capturar informações dinâmicas.

- **Parâmetros obrigatórios:** Os parâmetros obrigatórios são especificados entre chaves {}na definição da rota.

```
Route::get('/cursos/{id}', [CourseController::class, 'show']);
```

Nesse exemplo, {id}é um parâmetro dinâmico que pode ser acessado no driver. Uma URL /cursos/1chamaria o método showcom o valor 1para o parâmetro id.

- **Parâmetros adicionais:** Parâmetros adicionais podem ser definidos usando o ponto de interrogação ?, e um valor padrão pode ser passado no método do controlador.

```
Route::get('/cursos/{categoria?}', [CourseController::class, 'index']);
```

Se a categoria não for passada na URL, o método indexainda será executado, mas sem valor para o parâmetro.

Controladores e Métodos: Criação e Manipulação de Dados nas Views

Os controladores no Laravel são responsáveis por processar a lógica de negócios da aplicação e responder às respostas adequadas, como dados para uma visualização. Eles contêm métodos chamados pelas rotas para manipular os dados necessários.

- **Criação de controller:** Um controller pode ser criado com o comando Artisan:

```
php artisan make:controller CourseController
```

- **Métodos do** controller**:** Um controller normalmente contém métodos como index(), show(), create(), store(), edit(), e update(), que incluem as operações CRUD (Create, Read, Update, Delete).

Exemplo de controlador:

```
class CourseController extends Controller
{
    public function index()
    {
        $courses = Course::all();
        return view('courses.index', compact('courses'));
    }
```

```
    public function show($id)
    {
        $course = Course::find($id);
        return view('courses.show', compact('course'));
    }
}
```

- **index():** Retorna todos os cursos e os envia para uma view chamada courses.index.
- **show($id):** Recebe um parâmetro $id, busca o curso correspondente no banco de dados e envia para uma view chamada courses.show.

Os controladores permitem separar a lógica da aplicação, facilitando a organização do código e a manutenção. Eles recebem os dados das rotas, interagem com os modelos para acessar o banco de dados e enviam os dados processados para as visualizações.