Introdução ao Laravel

Framework PHP moderno
Desenvolvido para tornar o processo de criação web mais agradável.

Baseado no padrão MVC
Organiza o código em Model, View e Controller.

Facilita o desenvolvimento
Recursos prontos: autenticação, rotas e sessões.

Sintaxe expressiva
Código limpo e fácil de entender.

Ampla comunidade e documentação
Suporte ativo e recursos de aprendizado abundantes para desenvolvedores.

Por que usar o Laravel?

Agilidade no desenvolvimento
Reduz o tempo de desenvolvimento com componentes prontos e estrutura organizada.

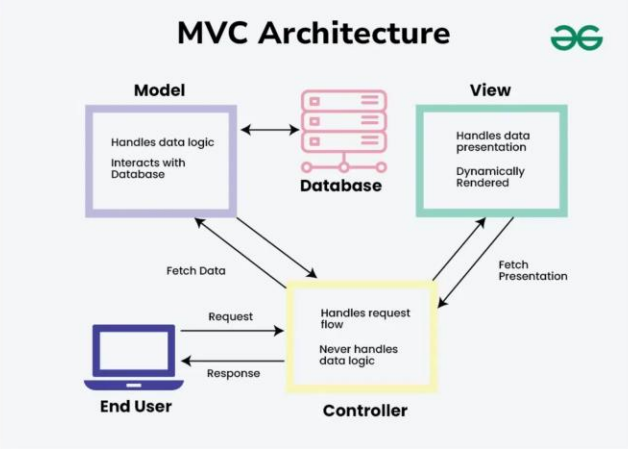
Ferramentas integradas
Artisan (CLI), Blade (templates) e Eloquent (ORM) facilitam o trabalho.

Segurança integrada
Proteção contra CSRF, hash de senhas e sistema de autenticação.

Suporte para APIs
Facilidade para criar APIs RESTful com recursos nativos.

Manutenção e escalabilidade
Código organizado e modular facilita manutenção e crescimento do projeto.

Arquitetura MVC



Model

Representa os dados e a lógica de negócios. No Laravel, são classes que interagem com o banco de dados através do Eloquent ORM.

View

Responsável pela interface do usuário. No Laravel, são arquivos Blade que combinam HTML com código PHP de forma elegante.

Controller

Gerencia as requisições do usuário, processa dados através dos Models e retorna as Views. É o intermediário entre Model e View.

Fluxo de dados no Laravel MVC:

1. Usuário faz uma requisição através do navegador
2. A rota direciona a requisição para o Controller apropriado
3. O Controller interage com o Model para acessar/modificar dados
4. O Controller passa os dados para a View
5. A View renderiza a página HTML que é enviada ao navegador

Requisitos para Laravel 11

Antes de começar, você precisa:

PHP >= 8.2
Laravel 11 requer PHP versão 8.2 ou superior.

Composer instalado
Gerenciador de dependências para PHP.

MySQL
Banco de dados relacional para armazenamento.

Node.js e npm (opcional)
Para compilação de assets frontend.

Editor de código
VSCode é recomendado para desenvolvimento.

Criando um Projeto Laravel 11

1 Criar o projeto

```
composer create-project laravel/laravel nome-do-projeto
```

2 Entrar na pasta do projeto

```
cd nome-do-projeto
```

3 Iniciar o servidor local

```
php artisan serve
```

Isso iniciará o servidor em http://localhost:8000

Estrutura básica criada:

- **app/** - Contém os controllers, models e outros arquivos PHP
- **config/** - Arquivos de configuração do projeto
- **database/** - Migrações e seeders
- **resources/** - Views, assets e arquivos de linguagem
- **routes/** - Definições de rotas da aplicação

Dica: Use o comando `php artisan` para ver todos os comandos disponíveis no Laravel.

Configuração do Banco de Dados

```
DB
```

Importante:

Crie o banco de dados manualmente no MySQL antes de usar.

Passos para configuração:

1. **Crie o banco de dados**
   Use o MySQL Workbench ou linha de comando para criar o banco.
2. **Edite o arquivo .env**
   Configure as credenciais do banco de dados no arquivo .env na raiz do projeto.

O que é CRUD?

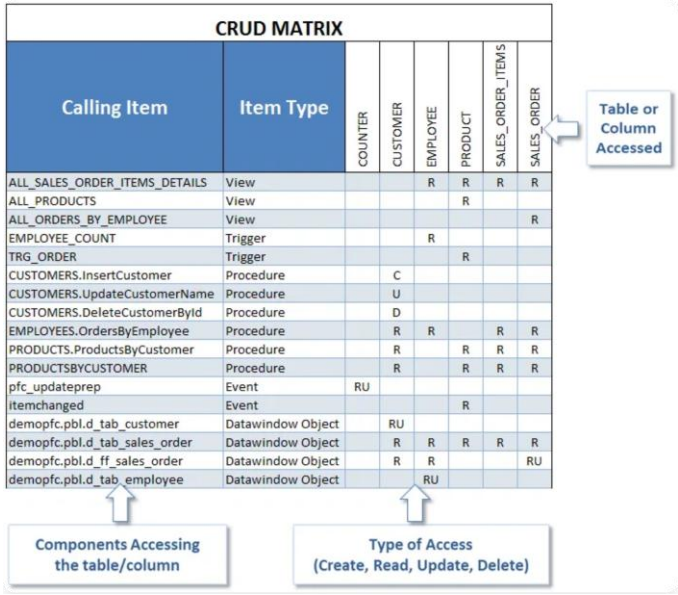
CRUD é um acrônimo para as quatro operações básicas usadas em aplicações que utilizam banco de dados:

Create (Criar)
Inserir novos registros no banco de dados

Read (Ler)
Consultar e exibir registros existentes

Update (Atualizar)
Modificar registros existentes

Delete (Excluir)
Remover registros do banco de dados



No Laravel:

Create: `$aluno = new Aluno(); $aluno->save();`

Read: `Aluno::all();` ou `Aluno::find($id);`

Update: `$aluno->update($request->all());`

Delete: `$aluno->delete();`

Criando Model e Controller

Crie o model e o controller do CRUD:

php artisan make:model Aluno

php artisan make:controller AlunoController --resource

app/Models/Aluno.php

```php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Aluno extends Model
{
protected $table = 'alunos';
protected $fillable = [
'matricula',
'nome',
'email',
'data_nascimento',
];
}
```

Sobre o Model:

Sobre o Controller:

index()	- Listar todos	create()	- Form de criação
store()	- Salvar novo	show()	- Exibir um
edit()	- Form de edição	update()	- Atualizar
destroy()	- Excluir registro		

Definindo Rotas do CRUD

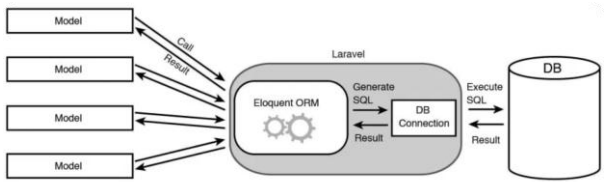
```
php artisan make:controller AlunoController --resource
```

O que isso faz?

O método `resource()` cria automaticamente todas as rotas necessárias para um CRUD completo, seguindo as convenções RESTful.

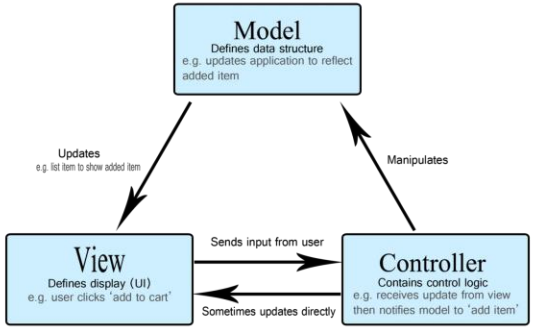
Isso é equivalente a criar manualmente:

- GET /alunos (index) - Listar todos
- GET /alunos/create (create) - Formulário de criação
- POST /alunos (store) - Salvar novo registro
- GET /alunos/{id} (show) - Visualizar um registro
- GET /alunos/{id}/edit (edit) - Formulário de edição
- PUT/PATCH /alunos/{id} (update) - Atualizar registro
- DELETE /alunos/{id} (destroy) - Excluir registro



Método	URI	Nome	Ação

Criando Views com Blade



Estrutura de arquivos:

- resources/views/alunos/
  - index.blade.php
  - create.blade.php
  - edit.blade.php
  - show.blade.php

Exemplo de View com Blade:

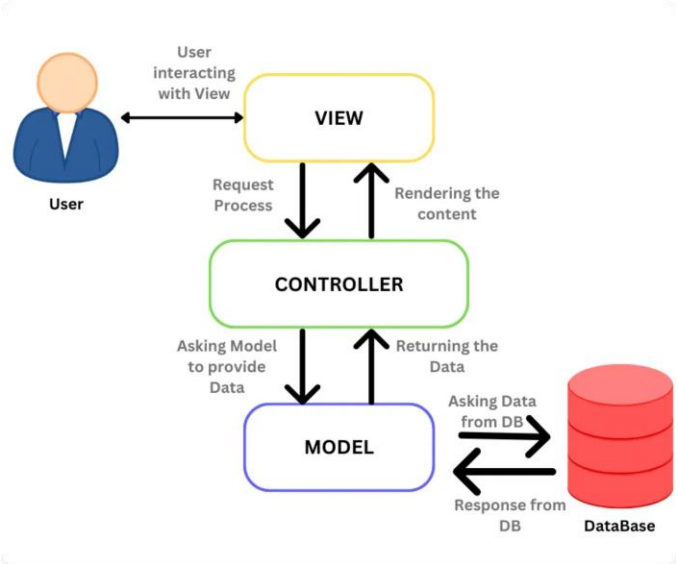
```
">Novo Aluno</a>
 <thead>
 <tr>
 <th>Matrícula</th>
 <th>Nome</th>
 <th>Ações</th>
 </tr>
 </thead>
 <tbody>
@foreach ($alunos as $aluno)
 <tr>
 <td>{{ $aluno->matricula }}  </td>
 <td>{{ $aluno->nome }}  </td>
 <td>
 <a href="https://private-us-east-
1.manuscdn.com/sessionFile/J3Up4TW2WPZGhKLe0Cipi8/sandbox/slides_resource_
05b3820d-e2b-prod-
aws_1758663785133_na1fn_L2hvbWUvdWJ1bnR1L2xhcmF2ZWxfcHJlc2VudGF0aW9uL
x-oss-
```

Implementação Completa do CRUD

Model | Controller | View

app/Models/Aluno.php

```php
namespace App\Models;
use Illuminate\Database\Eloquent\Model;
class Aluno extends Model
{
protected $table = 'alunos';
protected $fillable = [
'matricula',
'nome',
'email',
'data_nascimento',
];
}
```



Fluxo completo do CRUD:

1. **Model (Aluno.php):** Define a estrutura dos dados e as regras de negócio
2. **Controller (AlunoController.php):** Gerencia as requisições e coordena o fluxo de dados
3. **Views (*.blade.php):** Exibem os formulários e listagens para o usuário
4. **Rotas (web.php):** Conectam as URLs aos métodos do controller

Conclusão

Resumo do que foi feito:

- Criamos um projeto Laravel 11
- Conectamos ao MySQL com .env
- Criamos a tabela alunos via SQL
- Criamos Model, Controller e Rotas
- Implementamos um CRUD básico com views Blade

Próximos passos:

- Implementar validação de formulários
- Adicionar autenticação de usuários
- Criar APIs RESTful com Laravel
- Explorar recursos avançados do Eloquent ORM