



Desenvolvimento Web II

Framework Laravel 5

*Todo Conteúdo Laravel do 1º Bimestre
(Disciplina / Atividade / Conteúdo / Peso)*

Gil Eduardo de Andrade





Framework Laravel

Migrations:

(php artisan make:model DisciplinaModel -m)

```
public function up() {  
  
    Schema::create('disciplina_models', function (Blueprint $table) {  
        $table->increments('id');  
        $table->string('nome');  
        $table->string('abreviatura');  
        $table->integer('carga_horaria');  
        $table->integer('id_curso');  
        $table->string('periodo');  
        $table->integer('ativo');  
        $table->timestamps();  
    });  
}
```

Indica o curso ao qual a disciplina está vinculada.





Framework Laravel

Migrations:

(php artisan make:model AtividadeModel -m)

```
public function up() {  
    Schema::create('atividade_models', function (Blueprint $table) {  
        $table->increments('id');  
        $table->string('nome');  
        $table->string('tipo');  
        $table->date('prazo');  
        $table->integer('id_disciplina');  
        $table->integer('bimestre');  
        $table->integer('ativo');  
        $table->timestamps();  
    });  
}
```

Indica o prazo de entrega da atividade – data limite.

Indica a disciplina ao qual a atividades está vinculada.

Indica em qual bimestre a atividades deve ser contabilizada.





Framework Laravel

Migrations:

(php artisan make:model ConteudoModel -m)

```
public function up() {  
    Schema::create('conteudo_models', function (Blueprint $table) {  
        $table->increments('id');  
        $table->string('nome');  
        $table->date('data');  
        $table->integer('id_disciplina');  
        $table->integer('bimestre');  
        $table->integer('ativo');  
        $table->timestamps();  
    });  
}
```

Indica a data na qual o conteúdo será lecionado.

Indica a disciplina ao qual o conteúdo está vinculado.

Indica a qual bimestre o conteúdo pertence.



Framework Laravel

Migrations:

(php artisan make:model PesoModel -m)

```
public function up() {  
    Schema::create('peso_models', function (Blueprint $table) {  
        $table->increments('id');  
        $table->integer('id_disciplina');  
        $table->double('trabalho', 3, 2);  
        $table->double('avaliacao', 3, 2);  
        $table->double('parcial01', 3, 2);  
        $table->double('parcial02', 3, 2);  
        $table->double('parcial03', 3, 2);  
        $table->double('parcial04', 3, 2);  
        $table->timestamps();  
    });  
}
```

Indica a disciplina
ao qual o peso das
atividades e da
avaliação está
vinculado.

Peso dos trabalhos
e da avaliação, três
dígitos inteiros e
duas casas
decimais depois da
vírgula.

Peso de cada um
dos 4 bimestres.




Framework Laravel

Rotas:

(arquivo: *web.php* – *Disciplinas e Conteúdos*)

```
34 Route::get('/disciplina', 'DisciplinaController@listar');
35 Route::get('/disciplina/cadastrar', 'DisciplinaController@cadastrar');
36 Route::get('/disciplina/editar/{id}', 'DisciplinaController@editar');
37 Route::post('/disciplina/salvar/{id}', 'DisciplinaController@salvar');
38 Route::get('/disciplina/remover/{id}', 'DisciplinaController@remover');
39 Route::get('/disciplina/confirmar/{id}', 'DisciplinaController@confirmar');
40
41 Route::get('/conteudo/cadastrar/{id}', 'ConteudoController@cadastrar');
42 Route::get('/conteudo/editar/{id}', 'ConteudoController@editar');
43 Route::post('/conteudo/salvar/{id}', 'ConteudoController@salvar');
44 Route::get('/conteudo/remover/{id}', 'ConteudoController@remover');
45 Route::get('/conteudo/confirmar/{id}', 'ConteudoController@confirmar');
```



Rotas para manipulação dos dados das disciplinas e dos conteúdos cadastrados. Repare que não há uma rota para listagem dos conteúdos. Essa dinâmica foi adotada porque os conteúdos são apresentados junto com as disciplinas num *ComboBox* adaptado, como pode ser visto no slide a seguir.

Framework Laravel

Disciplinas e Conteúdos *(ComboBox adaptado)*



Disciplinas Cadastradas

Cadastrar Nova Disciplina

buscar

Q

ID	NOME DA DISCIPLINA	CURSO	CARGA	PERÍODO	CONTEÚDOS	ATIVIDADES	ATIVO	EVENTOS
6	LINGUAGEM DE PROGRAMAÇÃO	EMI-INFO	160 H/A	BIMESTRAL	<div><div>+</div><div></div></div>	<div><div>+</div><div></div></div>	SIM	  
7	PROGRAMAÇÃO DE COMPUTADORES I	TADS	80 H/A	SEMESTRAL	<div><div>+</div><div></div></div>			

1º Bimestre

06/02/2018 - CONCEITOS INICIAIS - (ATIVO)

14/02/2018 - LÓGICA DIGITAL PROG - (ATIVO)

06/03/2018 - COMANDOS E/S - (ATIVO)

13/03/2018 - COMANDOS DE CONDIÇÃO - (ATIVO)

2º Bimestre

22/05/2018 - VETORES - (ATIVO)

3º Bimestre

15/05/2018 - XXXXX - (ATIVO)

©2018 » Gil Eduardo de Andrade » versão 3.0

Os conteúdos são apresentados num **ComboBox** adaptado que permite adicionar novas informações as

Os conteúdos são apresentados num **ComboBox** adaptado que permite adicionar novas informações ao clicar no “+”.



Framework Laravel

Rotas:

(arquivo: *web.php* – *Disciplinas e Conteúdos*)

```
47 Route::get('/atividade/cadastrar/{id}', 'AtividadeController@cadastrar');
48 Route::get('/atividade/editar/{id}', 'AtividadeController@editar');
49 Route::post('/atividade/salvar/{id}', 'AtividadeController@salvar');
50 Route::get('/atividade/remover/{id}', 'AtividadeController@remover');
51 Route::get('/atividade/confirmar/{id}', 'AtividadeController@confirmar');
52
53 Route::get('/peso/editar/{id}', 'PesoController@editar');
54 Route::post('/peso/salvar/{id}', 'PesoController@salvar');
```



Rotas para manipulação dos dados das atividades cadastradas. Repare que assim como o conteúdo, não há uma rota para listagem das atividades que são apresentadas junto com as disciplinas num *ComboBox* adaptado. Os pesos possuem são adicionados com valores padrões quando uma disciplina é cadastrada, por isso só possuem duas rotas, uma para alteração dos valores e outra confirmação e salvamento no banco de dados.





Framework Laravel

Controle: (*DisciplinaControle.php – Método “listar”*)

```
public function listar() {  
  
    $disciplinas = DisciplinaModel::orderBy('nome')->get();  
    $cursos = CursoModel::select('id', 'nome', 'abreviatura')->get();  
    $atividades = AtividadeModel::orderBy('bimestre', 'ASC')->orderBy('prazo', 'ASC')->get();  
    $conteudos = ConteudoModel::orderBy('bimestre', 'ASC')->orderBy('data', 'ASC')->get();  
  
    return view('disciplina')->with('disciplinas', $disciplinas)  
        ->with('cursos', $cursos)  
        ->with('atividades', $atividades)  
        ->with('conteudos', $conteudos);  
}
```



Seleciona, além das informações das disciplinas, dados das atividades, conteúdos e cursos. Repare a utilização dos recursos do *Eloquent*: ***select()*** – permite especificar os campos da tabela a serem selecionados e ***order by()*** – permite especificar o campo da tabela que será utilizado como referência para ordenar os dados selecionados. Quando essas diretivas são necessárias devemos usá-las juntamente com o ***get()***, e não com o ***all()***.





Framework Laravel

Controle: (*DisciplinaControle.php – Método “salvar” – INSERT*)

```
// Dados - Tabela Peso (iniciais - padrão)
$objPesoModel = new PesoModel();
$objPesoModel->id_disciplina = $objDisciplinaModel->id;
$objPesoModel->trabalho = 0.5;
$objPesoModel->avaliacao = 0.5;

if(Request::input('periodo') == 'ANUAL') {
    $objPesoModel->parcial01 = $objPesoModel->parcial02 = $objPesoModel->parcial03 = $objPesoModel->parcial04 = 0.25;
}
else {
    $objPesoModel->parcial01 = $objPesoModel->parcial02 = $objPesoModel->parcial03 = $objPesoModel->parcial04 = 0.50;
}
```



Ao salvar o cadastro de uma nova disciplina já são definidos os valores dos pesos que os trabalhos e a avaliação terão para o cálculo do conceito bimestral. Também já são definidos, de acordo com o período de duração da disciplina (anual ou semestral) os pesos que cada bimestre terão dentro do cálculo do conceito final do aluno. Esses dados podem ser alterado posteriormente pelo usuário, mas num primeiro momento são definidos com um valor padrão.



Framework Laravel

Controle: (*DisciplinaControle.php – Método “remove”*)

```
if(is_numeric($id)) {  
    $disciplina = DisciplinaModel::find($id);  
    if(empty($disciplina)) {  
        $msg = "Disciplina não encontrada para o ID=$id!";  
        return view('messagebox')->with('tipo', 'alert alert-warning')  
            ->with('titulo', 'OPERAÇÃO INVÁLIDA')  
            ->with('msg', $msg)  
            ->with('acao', "/disciplina");  
    }  
}
```

Verifica se o parâmetro vindo pela URL é válido, é um número (*id*).

Busca no banco a disciplina que possui o *id* informado (*\$id*).

Verifica se existe uma disciplina para o *id* informado. Caso não seja encontrada no banco, uma *view* em formato de caixa de mensagem (criada para esse tipo de situação) é invocada! O título, a mensagem, o tipo do alerta e a ação a ser tomada após a caixa de mensagem ser fechada são definidas antes da sua chamada.

Framework Laravel

Controle: (*DisciplinaControle.php – Método “remove” – continuação*)

```
$total_atv = AtividadeModel::where('id_disciplina', $id)->count();
$total_con = ConteudoModel::where('id_disciplina', $id)->count();

if($total_atv == 0 and $total_con == 0) {
    return view('disciplinaRemover')->with("disciplina", $disciplina);
}
else {
    $msg = "Existem Conteúdos e/ou Atividades vinculadas a disciplina";

    return view('messagebox')->with('tipo', 'alert alert-danger')
        ->with('titulo', 'OPERAÇÃO INVÁLIDA')
        ->with('msg', $msg)
        ->with('acao', "/disciplina");
}
```

Verifica se a disciplina possui conteúdos e/ou atividades vinculadas.

Caso não possua, então a disciplina pode ser removida, e a *view* para confirmação da remoção é invocada.

Caso possua, a disciplina não pode ser removida, e a caixa de mensagem padrão é invocada.

Framework Laravel

Controle: (*DisciplinaControle.php* – Método “confirmar”)

```
$objDisciplinaModel = DisciplinaModel::find($id);  
  
if(empty($objDisciplinaModel)) {  
    $msg = "Disciplina não encontrada para o ID=$id!";  
    return view('messagebox')->with('tipo', 'alert alert-danger')  
        ->with('titulo', 'OPERAÇÃO NÃO-CONCLUÍDA')  
        ->with('msg', $msg)  
        ->with('acao', "/disciplina");  
}  
  
$objPesoModel = PesoModel::where('id_disciplina', $objDisciplinaModel->id)->first();  
  
$objDisciplinaModel->delete();  
$objPesoModel->delete();
```

Verifica se há uma disciplina com *id* = *\$id*.

Caso não haja invoca caixa de mensagem para informar o erro.

Se houver seleciona os pesos configurados para a disciplina para que também sejam removidos.

Observe que o método *first()* do *Eloquent* foi utilizado ao invés do método *get()*. Isso foi feito porque precisamos da referência do registro na tabela de pesos para que sua remoção seja efetuada, e o método *get()* retorna apenas um *collection*.



Framework Laravel

Controle: *(ConteudoControle.php / AtividadeControle.php / PesoControle.php)*

- As classe de controle para conteúdo, atividade e peso possuem conceitos que já foram vistos e trabalhados anteriormente. Sendo assim não serão abordadas porque podem ser compreendidas facilmente pelos alunos.



Framework Laravel

View: (*disciplina.blade.php* – *ComboBox conteúdo* – 1ª parte)

```
<a href="{{ action('ConteudoController@cadastrar', $dados->id) }}" type="button" class="btn btn-primary">
    <span class='glyphicon glyphicon-plus'></span>
</a>
<ul class="dropdown-menu" role="menu">
    @php ($flag = 1)
    @foreach($conteudos as $data)
        @if($data->id_disciplina == $dados->id)

            @if($data->bimestre == 1 and $flag == 1)
                <li class="dropdown-header">1º Bimestre</li>
                @php ($flag = 2)
            @elseif ($data->bimestre == 2 and $flag == 2)
                <li class="divider"></li>
                <li class="dropdown-header">2º Bimestre</li>
                @php ($flag = 3)
```

Observe que o *id* da disciplina que está sendo listada é configurada para o botão “+” que permite adicionar novos conteúdos. Isso permite vincular o novo conteúdo a disciplina selecionada para recebê-lo.

Ao percorrer o *array \$conteudos* que contém as informações de todos os conteúdos cadastrados no sistema, aqueles que possui vínculo com a disciplina que está sendo listada no momento são adicionados no ComboBox. Uma verificação para separá-los por bimestre é feita e um divisor dentro do ComboBox é adicionado.



Framework Laravel

View: (*disciplina.blade.php – ComboBox conteúdo – 2ª parte*)

```
@elseif ($data->bimestre == 3 and $flag == 3)
    <li class="divider"></li>
    <li class="dropdown-header">3º Bimestre</li>
    @php ($flag = 4)
@elseif ($data->bimestre == 4 and $flag == 4)
    <li class="divider"></li>
    <li class="dropdown-header">4º Bimestre</li>
    @php ($flag = 0)
@endif
```

```
<li>
    <a href="{{ action('ConteudoController@editar', $data->id) }}">
        @if($data->ativo == 1)
            <i>{{ \Carbon\Carbon::parse($data->data)->format('d/m/Y') }}</i> - {{ $data->nome }} - (ATIVO)
        @else
            <i>{{ \Carbon\Carbon::parse($data->data)->format('d/m/Y') }}</i> - {{ $data->nome }} - (INATIVO)
        @endif
    </a>
</li>
```

Ao adicionar os dados dos conteúdos em cada item do ComboBox já o define como selecionável para edição, apontando para o método editar da classe de controle de conteúdo, definindo como parâmetro o *id* do conteúdo que será editado. Ao adicionar o pacote `\Carbon\` do blade é utilizado para formatar a data vinda do banco no padrão “d/m/y”.

O ComboBox de atividades, que vem na sequência do código, segue o mesmo padrão.





Framework Laravel

View: (*disciplinaEditar.blade.php – Ativo / Inativo*)

```
<div class="col-sm-2">
  <label>Status: </label>
  <select name="ativo" class="form-control">
    @if($disciplina->ativo == 1)
      <option selected="true">ATIVO</option>
      <option> INATIVO</option>
    @else
      <option>ATIVO</option>
      <option selected="true"> INATIVO</option>
    @endif
  </select>
</div>
```

Em comparação com o cadastro, a *view* de edição de disciplinas possui um ComboBox a mais que permite ativar ou desativar uma disciplina.

Ao montar o Combo o status da disciplina é verificado (0 – inativa / 1 – ativa) com intuito de deixar o item “ativo” ou “inativo” previamente selecionado tendo como base os dados lidos do banco .





Conceitos Iniciais

Exemplos Utilizados no Documento

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_exdoc09.zip

Código-fonte da Aplicação SETA

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_dica09.zip

Exercício sobre o Conteúdo

http://www.gileduardo.com.br/ifpr/dwii/downloads/dwii_pratica09.pdf

