# Criação de Aplicativo Finanças com o CakePHP

Criaremos um pequeno aplicativo para controle de despesas pessoais, contendo apenas duas tabelas, despesas e receitas.

## Criação do banco (financas)

```
CREATE TABLE 'receitas' (
 `id` int(11) NOT NULL AUTO_INCREMENT,
 'descricao' varchar(100) NOT NULL,
 `mes` char(7) NOT NULL,
 `valor` int(11) NOT NULL,
 `created` DATETIME DEFAULT NULL,
 `modified` DATETIME DEFAULT NULL,
 PRIMARY KEY ('id')
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
CREATE TABLE 'despesas' (
 'id' int(11) NOT NULL AUTO INCREMENT,
 `descricao` varchar(100) NOT NULL,
 'valor' int(11) NOT NULL,
 'mes' char(8) NOT NULL,
 `created` DATETIME DEFAULT NULL,
 `modified` DATETIME DEFAULT NULL,
 `receita_id` int(11) NOT NULL,
 PRIMARY KEY ('id')
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

#### Criação do aplicativo:

cd /var/www/html composer create-project --prefer-dist cakephp/app financas

### Configuração do Aplicativo

Banco de dados – config/app.php

Rotas – config/routes.php

## Geração do Código Básico com o bake

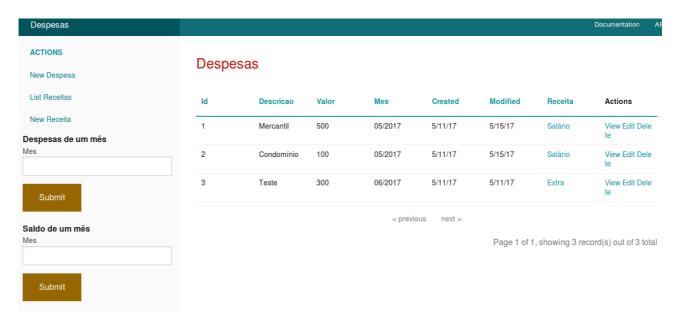
cd /var/www/html cd financas

bin/cake bake all receitas bin/cake bake all despesas

**ALERTA** – quando colando o código cuidado com caracteres especiais, especialmente aspas.

Nosso objetivo é criar um aplicativo que some as despesas do mês sugerido. E que encontre o saldo entre as somatórias de despesas e somatórios de receitas do mês.

#### Mais ou menos assim:



O mês tem um formato próprio, texto sendo dia/ano (dd/aaaa).

Vamos criar uma **função na DespesasTable.php**, que retorne a soma das receitas do mês:

```
public function despesasMes($mes){
          $connection = ConnectionManager::get('default');
          $results = $connection->execute("SELECT sum(valor) FROM despesas WHERE mes = '$mes'")->fetchAll();
          return $results[0][0];
}
```

Agora criarei uma outra **função no respectivo DespesasController.php**, que retorne o resultado da função acima:

Agora criaremos uma função no Receitas Table.php, que retorne a soma das receitas do mês:

```
Adicionar ao início:
use Cake\Datasource\ConnectionManager;
public function receitaMes($mes){
       $connection = ConnectionManager::get('default');
       $results = $connection->execute("SELECT sum(valor) FROM receitas WHERE mes =
'$mes''')->fetchAll();
       return $results[0][0];
}
Veja que existe grande semelhança entre esta e a das despesas.
Agora vamos fazer algo diferença. Vamos trabalhar com dois models com o mesmo controller. Para
isso vamos usar a função LoadModel(). Criaremos outra função no DespesasController.php
Vamos criar uma função que retorne o saldo do mês entre receitas e despesas
public function saldoMes()
       $mes = $this->request->data('mes');
       $this->LoadModel('Receitas');
       $receitas = $this->Receitas->receitaMes($mes);
       $despesas = TableRegistry::get('Despesas');
       $total = $despesas->despesasMes($mes);
       $saldo = $receitas - $total;
       $this->set('saldo',$saldo);
}
Vamos customizar a src/Template/Despesas/index.ctp e criar duas views novas:
src/Template/Despesas/despesas_mes.ctp:
<?php
print $total;
?>
<br>>
<br>
<br>
<!= $this->Html->link(__('Voltar'), ['action' => 'index']) ?>
src/Template/Despesas/saldo_mes.ctp:
<?php
print $saldo;
?>
<br>
<br>
<!= $this->Html->link(__('Voltar'), ['action' => 'index']) ?>
```

Agora vejamos a customização do index.ctp

Logo abaixo da linha:

```
<!= $this->Html->link(__('New Receita'), ['controller' => 'Receitas', 'action' => 'add']) ?>
```

Adicione os dois pequenos forms:

```
<b>Despesas de um mês</b>
</php
echo $this->Form->create("Despesas",array('url'=>'/despesas/despesas_mes'));
echo $this->Form->input('mes');
echo $this->Form->button('Submit');
echo $this->Form->end();
?>
<b>Saldo de um mês</b>
</php
echo $this->Form->create("Saldo",array('url'=>'/despesas/saldo_mes'));
echo $this->Form->input('mes');
echo $this->Form->button('Submit');
echo $this->Form->end();
?>
```

Depois de pronto podemos efetuar as operações através dos pequenos forms.

- O form chama o action do controller
- O controller chama o model para que devolva o resultado de uma função
- Recebendo o resultado o controller devolve para a view/template respectiva