**Aluno: Jonas Felicetti**

# Explicação de pastas/classes

## /core

É onde se encontram as classes bases para suas respectivas ‘filhas’, que herdam seus atributos e funções; sendo elas /App, /Controller e /Database

### /core/App

Faz o controle ‘geral’, lendo a URL, as classes e funções nela, junto com os parâmetros que serão utilizados nas respectivas funções.

### /core/Controller

Classe ‘pai’ das classes contidas em /controllers

### /core/Database

Classe que controla a conexão com o BDO

## /models

É onde se encontram as classes de regra de negócio e CRUDE com o banco de dados

## /views

É onde se encontram as telas, que serão mostradas para o usuário, ou cliente, após serem chamadas e receberem os dados necessários das classes de controle

## /controllers

É aqui que se encontram as classes que receberão as requisições dos clientes e informarão os dados a classe View especifica da requisição com os dados coletados pelas classes de Model

# Funcionamento do Sistema

Ao acessar o link /user no navegador será listado todos os usuários cadastrados no banco de dados, informando seus nomes e roles.

Ao clicar no botão de edição, no lado direito da tabela, você será encaminhado ao URL /user/show/#, sendo ‘#’ o ID do usuário que será editado, nesta tela você pode alterar, ou definir, caso não tenha sido definida ainda, a role do usuário e ao clicar em salvar a alteração será realizada no banco de dados e você será encaminhado para a url /user/save/#/#, sendo ‘#’ o ID do usuário que foi editado.

No final da página /user/save há um botão que o encaminhará para a pagina inicial(/user) novamente

# Benefícios e problemas do Framework

Principal beneficio que eu percebi ao trabalhar com o MVC é a questão de organização, tornando o código extremamente modular e excluindo quase que completamente a reescrita de códigos, por fazer uso do paradigma de PDO.

Mas encontrei uma certa dificuldade de início, evidenciando a curva de aprendizado que este modelo apresenta, sem falar que o projeto todo se torna relativamente complexo, não sendo muito interessante para aplicações simples, como aquelas desenvolvidas apenas com o intuito de aprender uma linguagem e etc. fazendo com que seja necessário mais tempo para desenvolver o projeto caso não esteja familiarizado com o modelo, mas em aplicações praticas ou maiores, principalmente quando tem mais de uma pessoa desenvolvendo, este tipo de modelo se torna quase indispensável, pelos motivos citados anteriormente.