Vaga Livre - Backend

Serviço de backend do projeto desenvolvido para Maratona técnologica da Uvv.

Arquitetura do projeto

A runtime do backend sendo utilizada será o Node.js (https://nodejs.org).

A API será construída utilizando o módulo Express (https://expressjs.com).

A API será responsável por manipular requisições HTTP seguindo o estilo REST (Representional State Transfer). A lógica responsável pela interpretação e respostas dessas requisições segue o padrão MVC.

O formato padrão para negociação de dados será o tipo MIME `application/json`.

As requisições devem ser **semanticas**:

- Cabeçalhos das requisições devem conter as seguintes chaves:
    - `Content-Type: application/json`
    - `Accepts: application/json`
- Cabeçalhos das respostas devem conter as seguinte chaves:
    - `Content-Type: application/json`
- Cabeçalhos das respostas devem conter status de acordo com o que realmente aconteceu:
    - Ao acessar um recurso: `HTTP/1.1 200 OK`
    - Ao criar um recurso: `HTTP/1.1 201 Created`
    - Ao destruir um recurso: `HTTP/1.1 204 No Content`
    - Ao atualizar um recurso: `HTTP/1.1 201 Created`
    - Ao acessar um recurso sem permissão: `HTTP/1.1 403 Forbidden`
    - Ao acessar recursos que não existam: `HTTP/1.1 404 Not Found`
    - Ao receber requisição sem `Content-Type: application/json`: `HTTP/1.1 406 Not Acceptable`

Estrutura de Pastas

```
|-- docs              // Documentos sobre o projeto
|-- infrastructure    // Arquivos sobre a infraestrutura de desenvolvimento
|-- server            // Codigo fonte do webservice
  |-- index.js        // Arquivo principal do serviço
  |-- package.json    // Definição de dependencias e comandos pre-definidos
  |-- config          // Configurações do projeto
  |-- controllers     // Controllers do projeto
  |-- models          // Modelos das entidades do banco de dados
  |-- routes          // Definição de rotas da aplicação
  |-- seeders         // Seed de dados limpos para aplicação
  |-- tests           // Testes automatizados
```

Persistencia de Dados

A arquitetura de persistencia de dados será criada utilizando o RDBMS PostgreSQL 9.5 (http://www.postgresql.org/). Por ser rápido, simples, opensource e de facil configuração, foi a opção mais aceita pelo grupo.

Ambiente de Desenvolvimento

O ambiente de desenvolvimento é criado e montado utilizando o Vagrant (https://vagrantup.com).

Vagrant é uma ferramenta para construir e configurar maquinas virtuais de forma reproduzível.

Seguindo o padrão de IAC (Infrastructure as Code) (https://www.thoughtworks.com/insights/blog/infrastructure-code-reason-smile), iremos garantir que a infraestrutura utilizada por todos no ambiente de desenvolvimento, testes e produção será a mesma, evitando os desenvolvedores de terem que lembrar daquela configuração manual que eles passaram horas pesquisando no Google passando dores de cabeça.

No root deste projeto é encontrado um arquivo `Vagrantfile`. É um arquivo em ruby com instruções de qual imagem deve ser clonada, configurações de rede e memória e quais comandos serão executados no provisionamento dessa maquina.

Mais informações sobre o `Vagrantfile` podem ser encontradas em https://www.vagrantup.com/docs/vagrantfile/ (https://www.vagrantup.com/docs/vagrantfile/)

No ambiente de desenvolvimento estaremos utilizando a distribuição Ubuntu (http://www.ubuntu.com/), presente em uma imagem mínima no Atlas (https://atlas.hashicorp.com/) (hub de imagens do Vagrant, suportado pela Hashicorp): hashicorp/precise64 (https://atlas.hashicorp.com/hashicorp/boxes/precise64).

No momento, o ambiente necessita de 128mb de memória disponível.

Dependências:

- Virtualbox (https://www.virtualbox.org/wiki/Downloads)
- Vagrant (https://vagrantup.com)

Iniciando Ambiente

Com o terminal na pasta `./infrastructure` do projeto, inicie o ambiente de desenvolvimento.

```
$ vagrant up
```

Localização do Código Fonte

O diretório `./server` deste reposítorio é sincronizado com a pasta `/server` no ambiente virtualizado. Todas as alterações feitas são sincronizadas em tempo real para dentro/fora da maquina virtual.

Iniciando o servidor

- Acesse a maquina virtual:
  `/ $ vagrant ssh`
- Acesse o diretório:
  `/ $ cd /server`
- Instale as dependências:
  `/server $ npm install`
- Inicie o servidor:
  `/server $ npm run start-dev`
