Projeto XPTO

Quem me ajuda(QMA)

Nome: Ricardo Magno do Nascimento Ferreira

Matrícula: 113112300

Disciplina:Desenvolvimento de Aplicações Corporativas Avançadas(DACA)

# Introdução

Projeto de QMA é um sistema que permite a alunos que precisam de apoio encontram alunos capacitados para oferecer ajuda. Foi utilizado Spring Boot no backend e Angular 6 para o frontend.

**Descrição do Projeto**

Os usuários serão capazes de se registrarem na plataforma como alunos e ou tutores, para estabelecer uma conexão entre aqueles que buscam ajuda e querem e podem ajudar em diversas disciplinas. Um aluno com alta proficiência em certa disciplina pode se tornar um tutor. Para maior compatibilidade com as necessidades dos alunos os tutores disponibilizam seus horários livres. Os alunos poderão avaliar os tutores assim como os tutores irão avaliar os alunos.

**User Stories**

***US1 - Cadastrar Alunos***

*Como sistema, deve ser possível cadastrar alunos para que os mesmos fiquem registrados e possam ser recuperados pela matrícula.*

*Cada aluno é identificado unicamente pela matrícula e tem um nome, código do curso, telefone e e-mail. Email é obrigatório e deve ser validado (ter uma arroba e ter algo antes e depois da arroba). Não deve ser possível nomes vazios ou em branco. O telefone é opcional. Todo aluno possui uma nota de avaliação (entre 0 e 5). Essa nota inicia como 5 e muda de acordo com as avaliações retornadas pelos tutores.*

## ***US2 - Definir papéis (Tutores)***

## *Como aluno, quero poder me disponibilizar como tutor para que eu possa atender ajudas de outros alunos de acordo com minha proficiência.*

## *Cada aluno no sistema pode se disponibilizar como tutor. Ao se cadastrar como tutor, é preciso indicar a disciplina na qual a pessoa pode ajudar mais (exemplo: “Cálculo 2”, “ICC”, “Programação 2”). O aluno deve indicar também uma proficiência (valor de 1-5), ou seja, o quão hábil ele se sente na disciplina. Todo tutor possui também uma nota de avaliação (entre 0 e 5). Essa nota inicía como 4 e muda de acordo com as avaliações retornadas pelos alunos. Cada tutor tem por fim, uma quantidade de dinheiro que foi recebida pelo sistema (começa em zero).*

## ***US3 - Disponibilizar e listar horários e locais***

## *Como tutor, quero disponibilizar meus locais e horários para que eu possa atender alunos de acordo com minha disponibilidade.*

## *Cada tutor deve cadastrar um conjunto de locais (identificados por nome) e um conjunto de dias disponíveis. O tutor deve indicar sua disponibilidade nos dias úteis da semana (seg, ter, qua, qui, sex). Importante: o local não é associado ao dia.*

## ***US4 - Cadastrar Pedidos de Ajuda (Presencial ou Online)***

## *Como aluno, quero pedir ajuda presencial ou online de forma que um tutor possa me atender de acordo com a minha necessidade.*

## *Um aluno (seja tutor ou não) pode cadastrar um pedido de ajuda. Existem dois tipos de pedidos, o pedido de ajuda presencial e o pedido de ajuda online.*

## *Na ajuda presencial, o aluno indica a disciplina e dia que quer uma ajuda e o local que tem interesse. Ao realizar um pedido no sistema, o sistema deve associar um tutor a esse pedido. O tutor escolhido precisa ter proficiência na disciplina e disponibilidade no local e no horário/dia indicado. Caso mais de um tutor esteja disponível naquele dia o de maior pontuação deve ser retornado (ou o primeiro aluno cadastrado em caso de empate). O mesmo tutor pode ser retornado para vários pedidos de ajuda diferentes.*

## *Na ajuda online, a interação irá acontecer via email (ou hangouts) sem necessidade de disponibilidade de horário. Há necessidade de especificar apenas a disciplina de interesse da ajuda. O tutor associado deve ter proficiência nessa disciplina e ser aquele de maior pontuação (ou o primeiro cadastrado em caso de empate).*

## *Ao cadastrar um pedido, é retornado um ID referente a ajuda marcada. Deve ser possível pegar informações sobre esse pedido de ajuda (qual foi a disciplina, dia e local de interesse). Deve ser possível pegar a matrícula do tutor que atenderá aquela ajuda, bem como pegar informações sobre a ajuda em si (horário, dia, local e disciplina).*

## *US5 - Avaliar Tutor*

## *Como aluno, desejo avaliar o tutor de forma que ele possa ser priorizado nos atendimentos.*

## 

## **Arquitetura**

## 

## Através do Spring foi utilizada a arquitetura MVC com o layer service para comunicação com o repositório. O **framework Spring**, é um dos frameworks Java mais difundidos e utilizado. Com ele é possível fazer injeção de dependência, persistência de dados e uma implementação para o padrão MVC, ajudando na criação de aplicações WEB robustas e flexíveis.

## 

## 

http://blog.harbinger-systems.com/2015/08/service-layer-in-rails-application/

## Os quais,

## 1. É acessada uma URL no browser que envia a requisição HTTP para o servidor que roda a aplicação web com Spring MVC. Quem recebe a requisição é o controlador do framework, o Spring MVC.

## 2. O controlador do framework irá procurar qual classe é responsável por tratar essa requisição, entregando a ela os dados enviados pelo browser. Essa classe faz o papel do controller.

## 3. O controller passa os dados para o model, que por sua vez executa todas as regras de negócio como validações, cálculos e acessos ao banco de dados.

## 3.b O controller passa os dados para o service, que por sua vez executa acessos ao banco de dados.

## 4. Os resultados das operações realizadas pelo model e pelo service são retornados ao controller.

## 5. O controller retorna o nome da view, junto com os dados que ela precisa para renderizar a página.

## 6. O Framework encontra a view que processa os dados, transformando o resultado em um HTML.

## 7. Finalmente, o HTML é retornado ao browser do usuário.

## 

## 

## 

## 

## Componentes

## 

## ● BackEnd:

## Entity

## Ajuda - Representa entidade ajuda.

## Student - Representa entidade aluno.

## Tutor - Representa entidade tutor.

## Service

## StudentService - Implementação do service do Aluno realizando a comunicação entre repositório e controller.

## TutorService - Implementação do service do Tutor realizando a comunicação entre repositório e controller.

## Controller -

## StudentRestController - Comunicação do modelo do Aluno com o front-end.

1. Dto -
   1. StudentDTO -Usado no envio de dados do Aluno ao usuário
   2. TutorDTO - Usado no envio de dados do Tutor ao usuário
   3. TutorRequest - Utilizado no envio de dados Tutor ao Controller
2. Repository
   1. StudentRepository - Armazenar Alunos
   2. TutorRepository - Armazenar Tutores

# 

# **Framework Spring Boot**

## Como descrito na arquitetura, o Spring Boot auxilia na interação Back-end~Front-end, dentro da classe StudentRestController.java podemos ver alguns dos seguintes exemplos de uso:

## @RequestMapping(value = "/aluno/", method = RequestMethod.POST)

## public ResponseEntity<?> cadastrarAluno(@RequestBody Student aluno, UriComponentsBuilder ucBuilder) throws StudentNotFoundException, StudentAlredySaveException {

## studentService.cadastrarAluno(aluno.getNome(), aluno.getMatricula(), aluno.getCode(), aluno.getTelefone(), aluno.getEmail(), aluno.getSenha());

## return new ResponseEntity<Student>(aluno, HttpStatus.CREATED);

## **Métodos Post são na intenção de algum tipo de cadastro. O front manda informações que são manipuladas pelo back.**

## @SuppressWarnings("unchecked")

## @RequestMapping(value = "/aluno/{matricula}", method = RequestMethod.GET)

## public ResponseEntity<?> recuperaAlunoDTO(@PathVariable("matricula") String matricula) throws StudentNotFoundException, Exception {

## StudentDTO aluno = studentService.recuperaAlunoDTO(matricula);

## if (aluno == null) {

## return new ResponseEntity(new CustomErrorType("Aluno with matricula " + matricula + " not found"), HttpStatus.NOT\_FOUND);

## }

## return new ResponseEntity<StudentDTO>(aluno, HttpStatus.OK);

## }

## Por outro lado, métodos Get são usados para fazer consultas, o front manda alguma informação e o back devolve outra a partir dela.

## 

## 

## 

## 