VenturaHR

Documento de Requisitos

versão 3.0

Histórico de Revisões

Data	Versão	Descrição	Autores
24/06/2021	1.0	Versão inicial revisada pela equipe liderada por Armênio Torres	Julia Teixeira da Paz Pinheiro
16/08/2021	2.0	Versão inicial revisada pela equipe liderada por Armênio Torres	Julia Teixeira da Paz Pinheiro
27/09/2021	3.0	Versão inicial revisada pela equipe liderada por Armênio Torres	Julia Teixeira da Paz Pinheiro

Repositório no GitHub: jupazpinheiro/venturaHR

Introdução

O RH 2.0 ganhou protagonismo e está sendo promovido a área estratégica, essencial para a tomada de decisão e crescimento do negócio.

Problemas antigos como a dificuldade em medir o retorno sobre o investimento (ROI) das ações, ineficiência do recrutamento e baixo desempenho em treinamentos já têm suas soluções tecnológicas sob medida.

As HRTechs são startups que desenvolvem soluções tecnológicas para a área de Recursos Humanos (Human Resources), agregando inteligência aos processos. A missão dessas empresas é levar a inovação ao RH, mostrando que a tecnologia pode reduzir custos, aumentar a eficiência e agilizar o crescimento do setor.

Cenário Atual

VenturaSoft é uma HRTech que atua no segmento de recolocação de profissionais de TI. Devido às peculiaridades desse mercado, os requisitos para contratação têm um dinamismo vertiginoso, pois as tecnologias e as “stacks” adotadas pelas empresas estão em constante evolução.

A empresa necessita que seja construída uma solução de software, chamado de VenturaHR, que tenha abrangência de todos os fluxos operacionais da sua atividade fim.

A VenturaSoft tem como clientes empresas que precisam fazer processos seletivos para vagas em aberto.

Solução desejada

VenturaHR precisa atualizar o processo de recrutamento afim de manter a relação empresa X vaga X candidato o mais ágil possível por meio de cálculo pelo sistema respeitando os critérios pré-definidos pela empresa.

As empresas cadastram suas vagas definindo um texto e requisitos da vaga atribuindo pesos a cada critério, o candidato responde as perguntas marcando os níveis de conhecimento em cada item. No final do período de anúncio estipulado pela empresa sistema com base nas informações cedidas calcula os pontos de cada candidato e entrega para a empresa um ranking de todos os candidatos aptos para a vaga.

Requisitos do Sistema

Requisitos Funcionais

ID	Descrição	Prioridade
RF-001	Criação de contas de usuário: Administrador, Empresa e Candidato. As contas de Empresa e Candidato poderão ser criadas pelo aplicativo. As contas de Administrador só podem ser criadas por administradores.	Essencial
RF-002	Usuário administrador deve visualizar um resumo operacional na sua Caixa de Entrada.	Desejável
RF-003	Usuário administrador pode pesquisar contas de usuários.	Essencial
RF-004	Usuário administrador pode bloquear / desbloquear contas de usuários que pesquisou.	Essencial
RF-005	Contas de usuário não poderão conter o mesmo e-mail	Essencial
RF-006	Usuário candidato deve possuir uma área de alertas e produtos que possam ser adicionados como cursos e palestras.	Desejável
RF-007	Contas de usuário e não usuário poderão ter acesso as vagas ativas em uma tabela oferecendo um filtro de pesquisa	Essencial
RF-008	Usuário empresa deve possuir um histórico de vagas anunciadas	Desejável
RF-009	Usuários devem ser notificados caso estejam inativos por	Desejável

	6 meses e excluídos em 1 ano de inatividade	
RF-010	Todas as vagas anunciadas devem conter ao menos um critério de seleção para ranking dos candidatos	Essencial

Requisitos Não Funcionais

ID	**Descrição**	**Prioridade**
NF-001	Todas as funcionalidades acessadas pela web	Essencial
NF-002	Execução de processos batch com hora marcada	Essencial
NF-003	Todas as funcionalidades devem ser acessíveis (WCAG)	Desejável
NF-004	Todas as funcionalidades da web devem ser responsivas	Desejável
NF-005	Todas as funcionalidades devem permanecer sempre disponíveis	Essencial
NF-006	Todos os usuários devem aceitar o Termo de Privacidade	Desejável
NF-007	Todos os usuários em seu cadastro devem conter uma senha segura (letras maiúsculas, minúsculas, números e caracteres especiais)	Desejável

Diagramas

Diagramas de Casos de Uso por Ator

Usuário

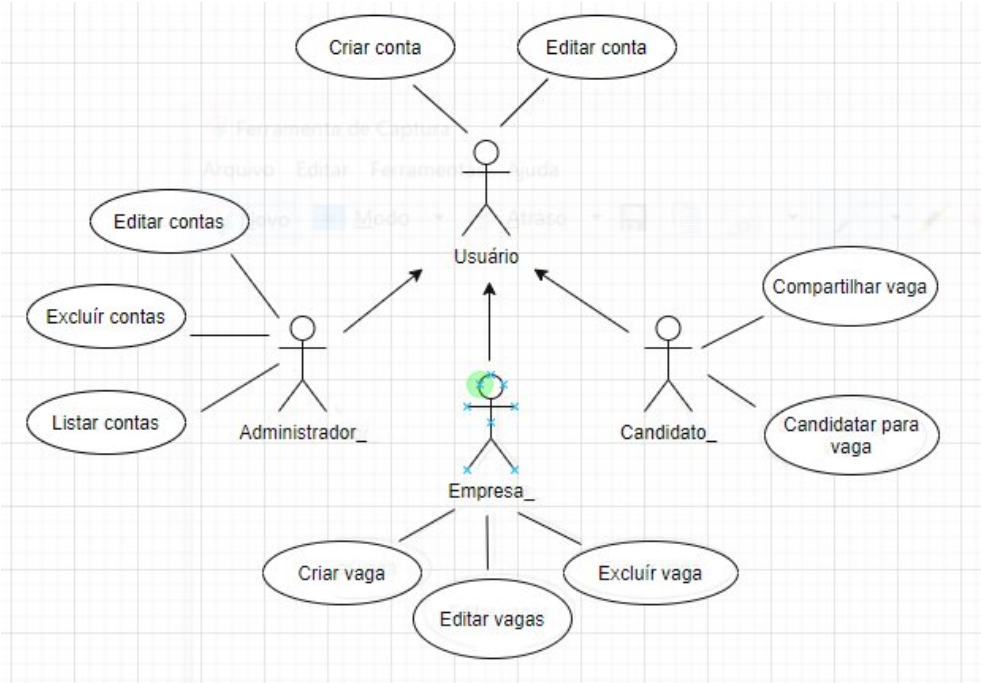


Diagrama de casos de uso dos atores exemplificando algumas das funções essencial

Administrador

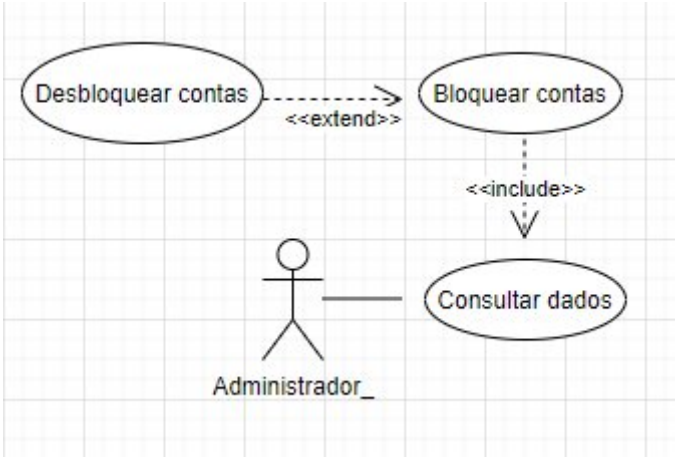


Diagrama de caso de uso de gerenciamento de contas de usuários

Empresa

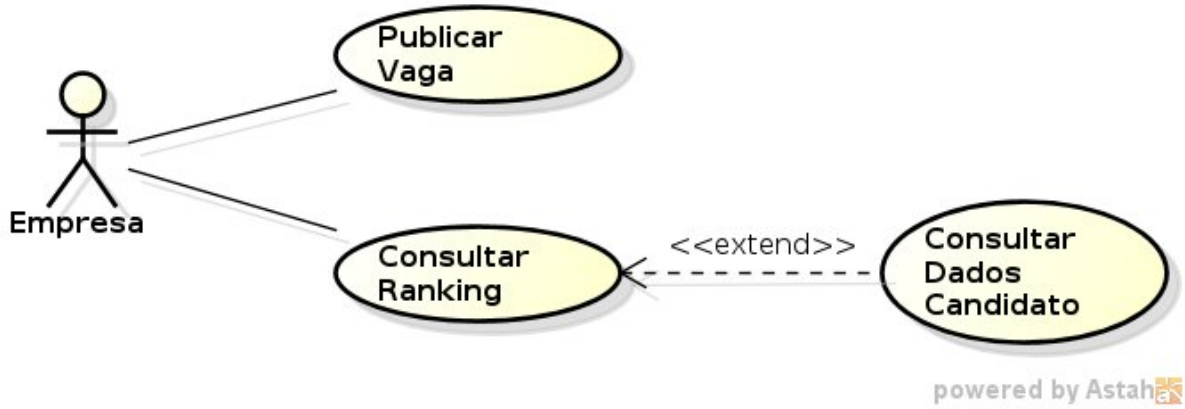


Diagrama de caso de uso para esquema de preenchimento das vagas

Candidato

Diagrama de caso de uso de candidatura de vaga pelo usuário, método exclusivo deste ator

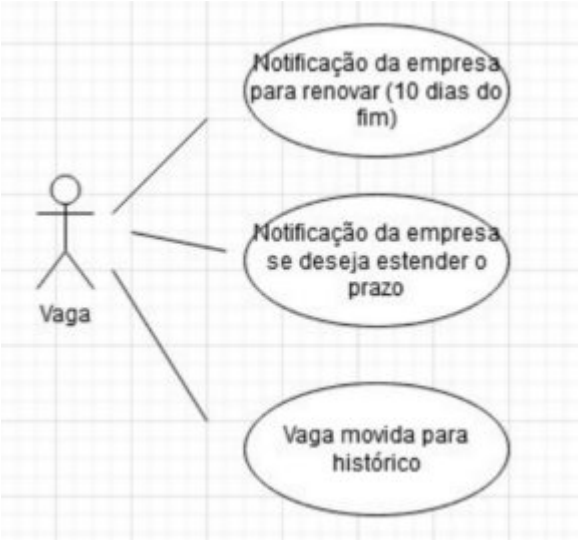


Diagrama de caso de uso de tempo

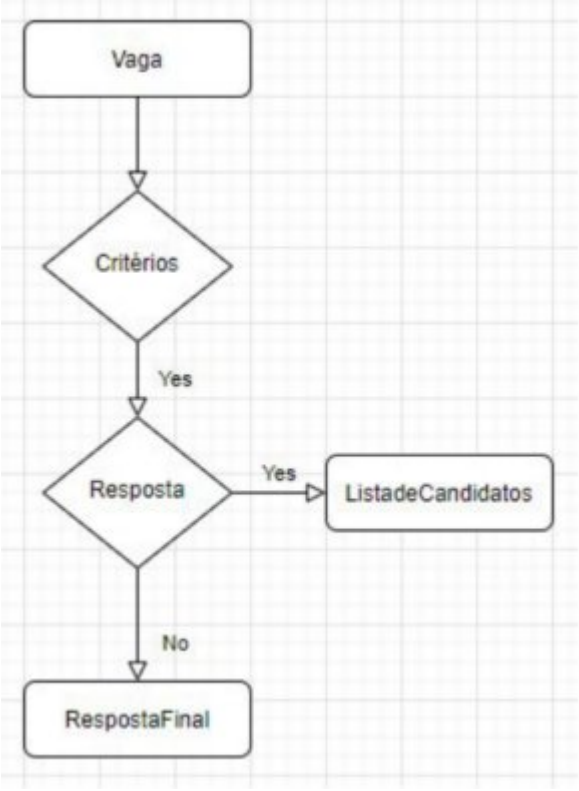


Diagrama de fluxo de vaga

Detalhamento dos Casos de Uso por Ator

Empresa

Nome: Publicar Vaga.

Descrição: A Empresa pública uma vaga de emprego, relacionando seus dados e seus critérios de seleção.

Atores: Empresa.

Pré-condições: Empresa devidamente autenticada (login), com seu contrato comercial em dia e com sua conta desbloqueada.

Pós-condições: Uma vaga de emprego será publicada e poderá ser pesquisada e receber respostas de Candidatos.

Fluxo Principal:

Empresa inicia a publicação de vaga de emprego.

Sistema solicita os dados da vaga:

* Título.

* Descrição.

* Forma de Contratação: CLT, PJ, Free lancer, Intermitente etc.

* Local de contratação: cidade, estado, bairro.

Empresa fornece os dados da vaga.

Sistema solicita o cadastramento dos critérios de seleção:

* Nome do Critério.

* Descrição.

* PMD: 1 - Desejável, 2 - Diferencial, 3 - Relevante, 4 - Muito Relevante e 5 - Obrigatório.

* Peso desse critério em relação aos outros critérios.

Empresa relaciona os critérios da vaga de emprego, preenchendo todos os dados.

Empresa finaliza a publicação da vaga.

Sistema publica a vaga de emprego.

Fluxos Alternativos:

Passo 4: Dados da Vaga inválidos

Sistema valida os dados e mostra mensagens de erro.

Volta ao passo 2.

Candidato

Nome: Responder Vaga

Descrição: O candidato pesquisa as vagas e se candidata respondendo aos critérios pré definidos pela empresa.

Atores: Candidato e Empresa

Pré-condição: Candidato autenticado para realizar a candidaturas.

Pós-condição: E-mail enviado para o candidato confirmando a candidatura.

Fluxo Principal:

1. Candidato filtra na aba de vagas pela palavra-chave.

2. Candidato completa as perguntas quanto aos critérios de seleção

3. Sistema solicita confirmação dos dados enviados

4. Candidato revisa e confirmando

5. Sistema envia e-mail para o Candidato com a confirmação de candidatura

6. Sistema avalia a pontuação e adiciona à lista de candidatos a ser enviada para a Empresa

Fluxo Alternativo:

Passo 4 Candidato desiste da vaga

1. Candidato seleciona o cancelar

2. Sistema retorna para o passo 1

Passo 6

1. Candidato não atinge a nota mínima estabelecida pela empresas

2. Candidato recebe um e-mail negando a candidatura.

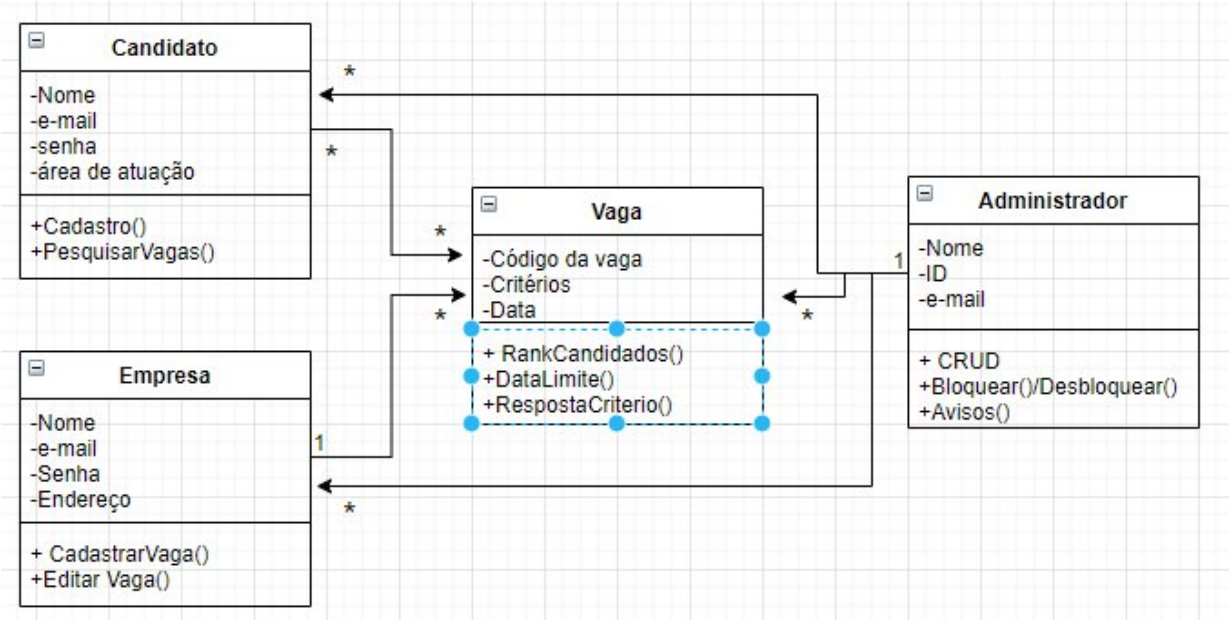


Diagrama de relação de classes

Diagrama de Atividade

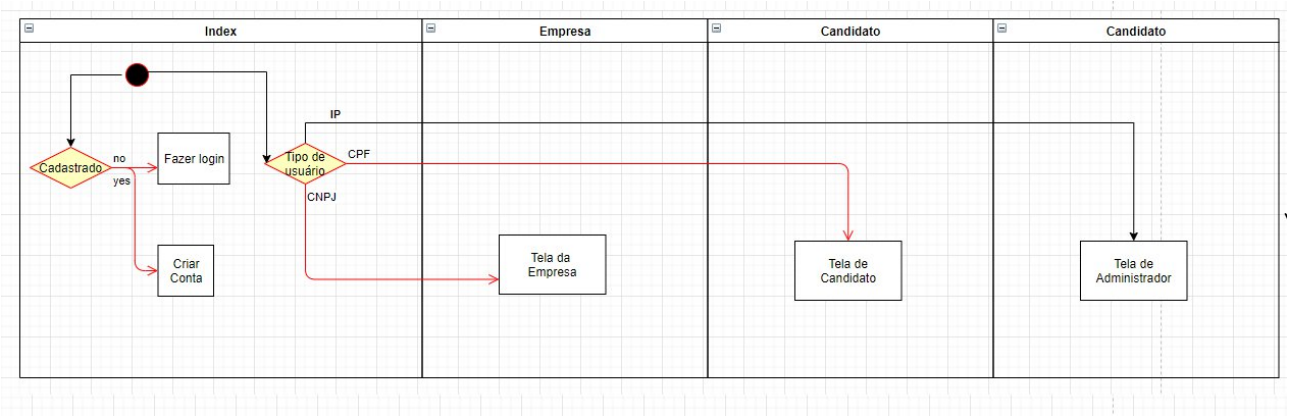


Diagrama do fluxo de login

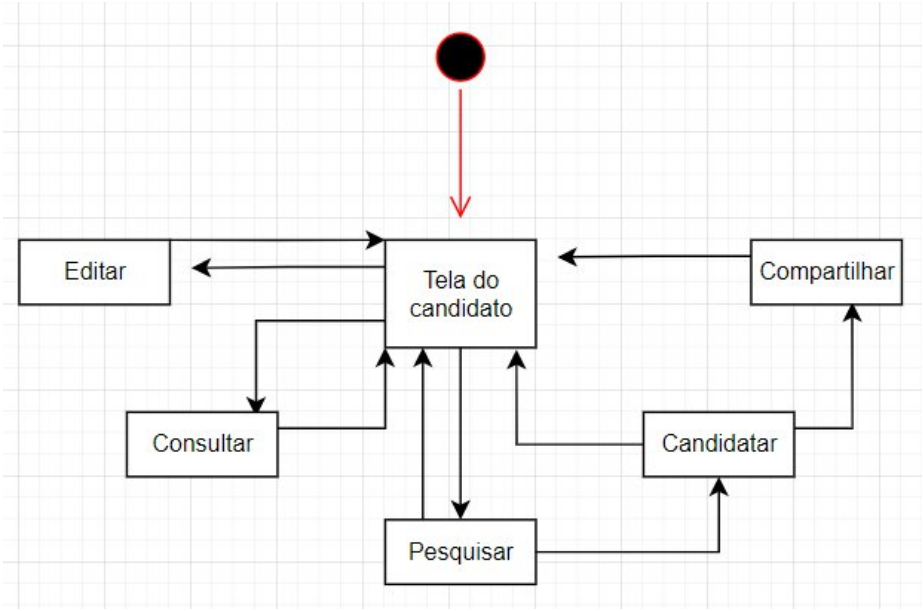


Diagrama de fluxo do candidato

Breve Descrição das Classes

Candidato

Classe dentro do pacote de models possui o perfil mais acesso a ele e mais restrições no sistema.

Vaga

Classe dentro do pacote de negócios/controle possui características informadas somente pela empresa e a regra de negócio a ser aplicada (critérios e data limite)

Administrador

Classe mãe dentro do pacote de models e possui acesso ilimitado ao sistema

Empresa

Classe dentro do pacote models possui o perfil a ser buscado com poucas restrições no sistema.

Fundamentos de Orientação a Objetos

Descrição das Classes Envolvidas

Usuário: representa as contas de login dos usuários do sistema. É a classe-base para Administração do sistema tendo como base os níveis de acesso a áreas destinadas do sistema catalogados em Candidatos, Empresa e Administrador.

Vaga: representa a publicação de uma Vaga de Emprego. Todas as vagas devem conter critérios formando um ranking de usuários:candidatos pelo próprio sistema.

Comportamentos:

Polimorfismo

Logar como Usuário e acessar a caixa de entrada correta.

Pesquisa de vagas oferece ações conforme o usuário. Só o candidato pode responder vaga.

Abstração – transformação de uma ação real em máquina

Método calcular no classe vaga.

Método pesquisar na classe usuário.

Encapsulamento

Resposta aos critérios da vaga, onde cada usuário:candidato só consegue prosseguir com a seleção se corresponder aos requisitos mínimos da vaga cadastrado pelo usuário:empresa

Herança

Características de cadastro são herdadas da classe mãe usuário:administrador que tem acesso total ao sistema, enquanto usuário:empresa e usuário:candidato herdam características e consequentemente acesso a áreas que destinam.

Diagrama de Sequência

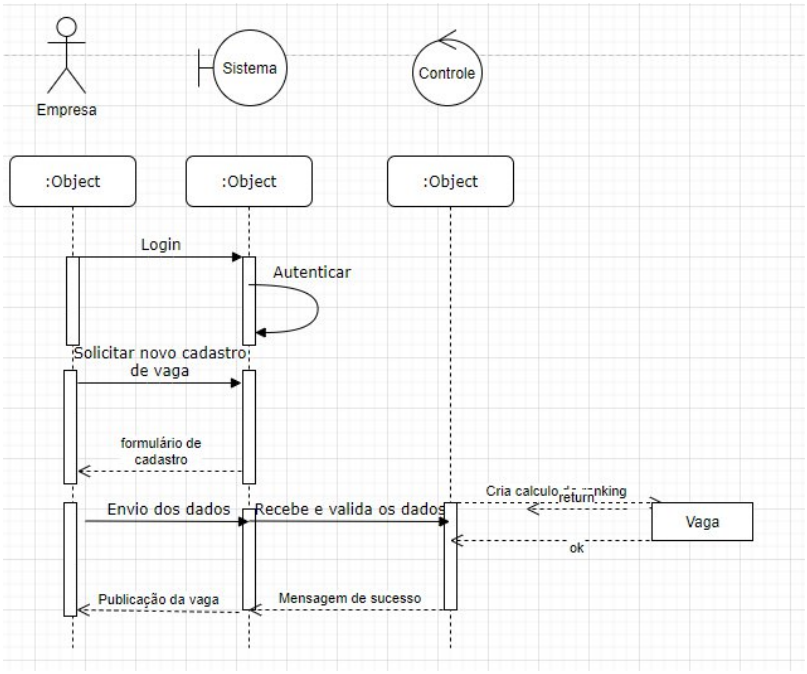


Diagrama de Sequência do Caso de uso publicar vaga

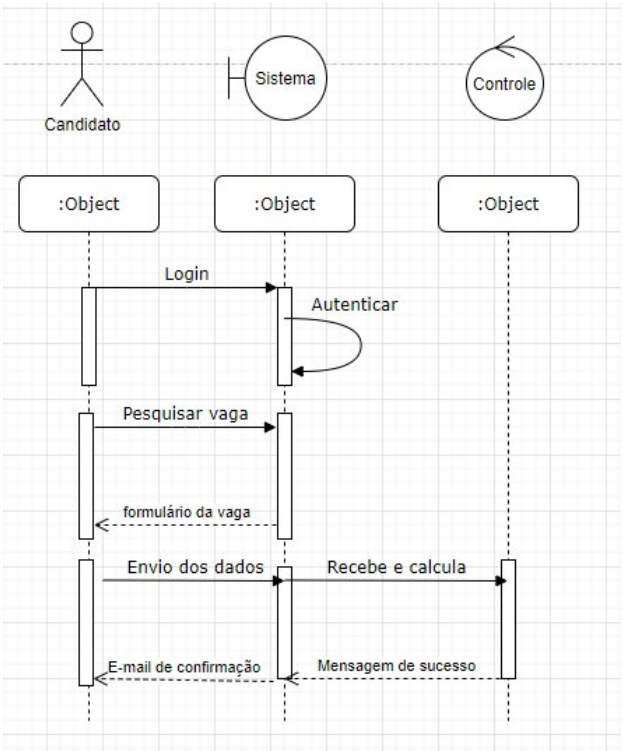


Diagrama de Sequência do Caso de Uso Responder Vaga

Diagrama de Estados

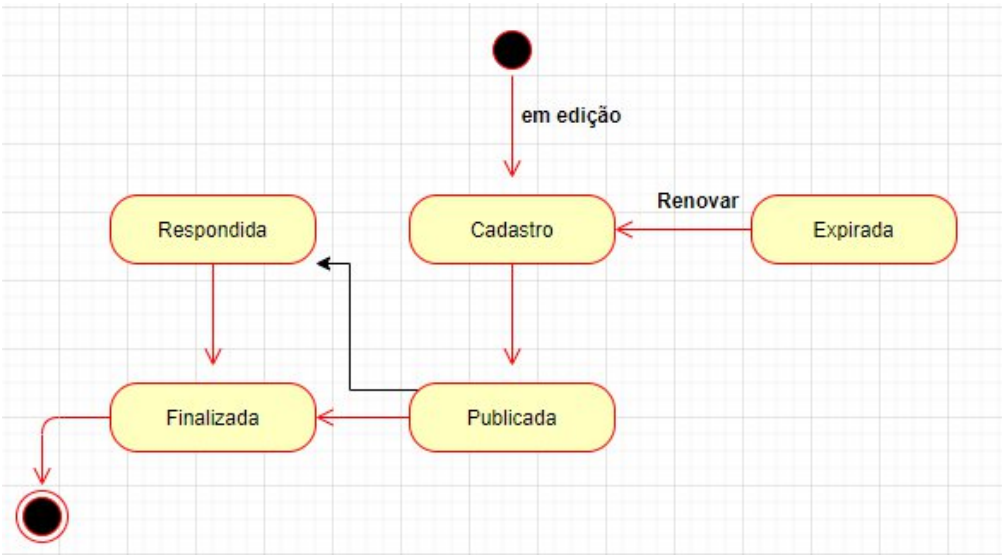


Diagrama de estados da vaga

Diagrama de ocorrência

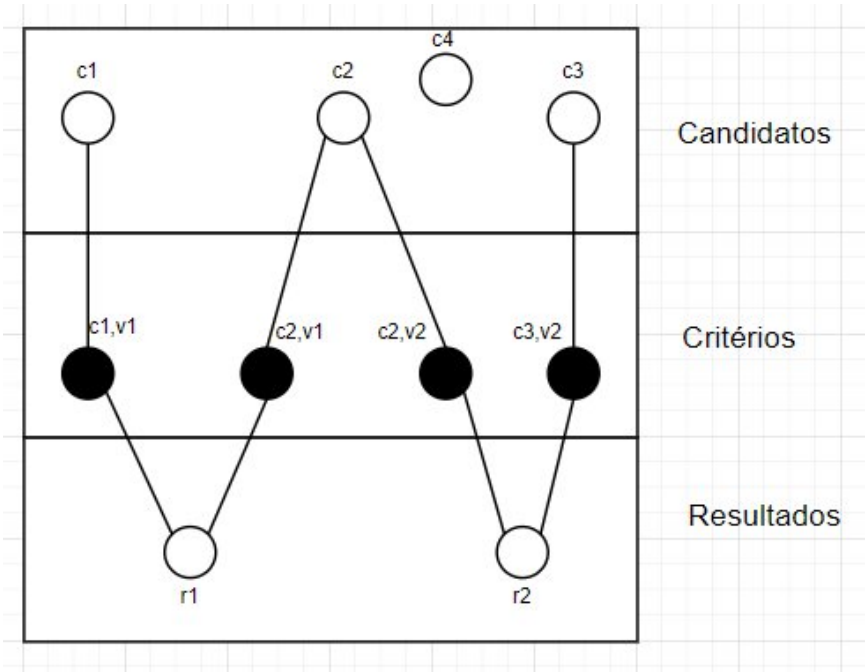


Diagrama de ocorrência de relacionamento de candidatos viáveis e suas candidaturas

Considerações Finais:

Tecnologias para desenvolvimento:

	Funções:
MySQL Workbench	Banco de dados
Intelli J	IDE
Postman	Teste das WebApi

Construção do site

Embasamento:

APIs - podem ser vistas como um mediador entre usuário ou clientes e os recursos ou serviçosweb que eles querem obter. A api ajudará a comunicar o que você quer no sistema para recuperar informações ou executar uma função.

Rest - é um conjunto de restrições de arquitetura e deve estar em conformidade com os seguites critérios

Ter uma arquitetura cliente/servidor (cliente - servidores-recursos) com solicitações gerenciadas por HTTP

Armazenamento de dados em cache para otimização das interações cliente-servidor

Ter um sistema em camadas

GET

Por default todos os métodos são get mas são armazenados no cache e os dados permanecem no histórico do navegador. Método usado apenas para solicitar dados.

POST

Utilizado para enviar dados a um servidor para criar ou atualizar um recurso

Versões

Após a etapas de documentações concluídas foi dada a partida na construção do projeto.

Uma primeira versão foi apresentada como monolíto pois houveram complicações em entender e chamar as rotas referentes a cada funcionamento dos objetos envolvidos. A

solução apresentada foi considerar todos usuários da aplicação realizando uma lógica para direcionar cada ator (candidato, empresa e administrador) a sua área pessoal.

Caminhos de Administrador e Candidato não foram desenvolvidos a fundo por razões de complexibilidade de sistema. Previsão para desenvolvimento destas partes em futuras versões.

A página inicial se encontra recursiva e deverá ser trabalhada para melhor acessibilidade.

Entendimento dos problemas

Imagens não apareciam pois o tomcat deve ler uma lógica em localhost e por isso as pastas devem ser salvas na raíz dos jsp.

Relações OtM (@OneToMany) foi difícil de entender e reproduzir no cógido.

Páginas mapeadas para novas vagas em todos os usuários, porém para acesso as vagas publicadas pelo usuário, somente com o login de segurança

Publicação das vagas sofreram mudanças de uma página separada para a página index da empresa por erros de rota.

As funções de Http session foram amplamente utilizadas para gerenciar as sessões. Quando um usuário loga a primeira vez, este recebe um Id por meio de um request.getSession()

```
//recursividade para addcriterio sem voltar pro index
ModelAndView resposta = new ModelAndView( viewName: "/empresa/addVaga");
HttpSession session = request.getSession();
```

-curiosidade- Caso não declarado, o tempo da session é determinado em 30min

As informações de usuário são armazenadas no .setAttribute para posteriormente serem utilizadas pelo usuário

```
//diferença em criterios
session.setAttribute( s: "vaga", vaga);
```

Para obtermos as informações fornecidas utilizamos o método getAttribute() da interface HttpSession

```
//recepção de OtM para criterios-varios possiveis-
List criterios = (List) session.getAttribute( s: "criterios");
if(criterios == null){
    criterios = new ArrayList<>();
}
criterios.add(criterio);
session.setAttribute( s: "criterios", criterios);
```

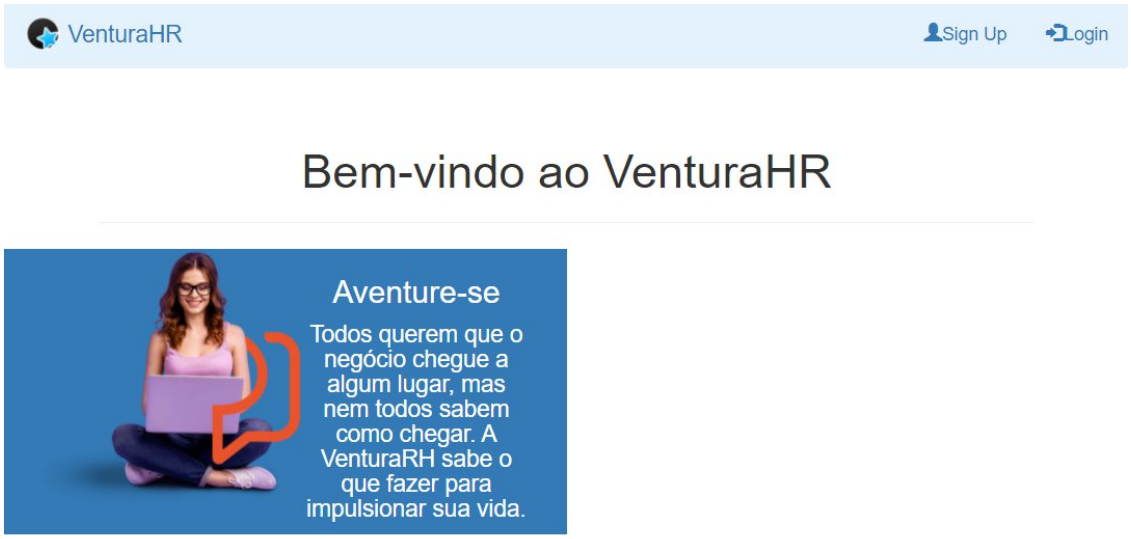
Para cadastro de outro atributo do objeto, era necessário remover-lo para iniciar uma nova atribuição

```
//limpeza da sessão para publicar mais de uma vaga
session.removeAttribute( s: "vaga");
session.removeAttribute( s: "criterios");
return "/empresa/index";
```

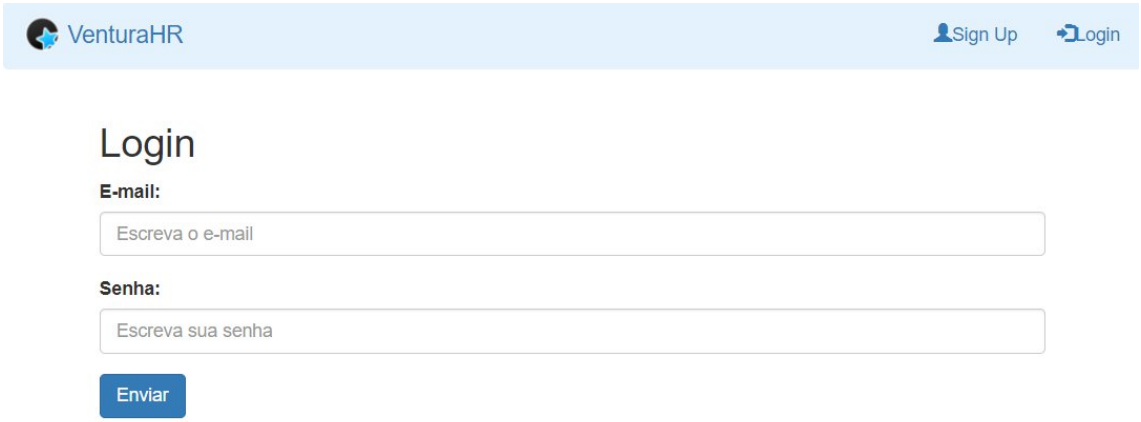
As APIs sofreram conflito e por isso foram modificadas as portas 8080/8081 e 8082 utilizando a anotação Feign

Resultado apresentado

Página Inicial



Página de Login



VenturaHR

v.1.0

Página inicial da empresa listando vagas

Conclusão

Apesar dos desafios enfrentados no desenvolvimento do projeto. A linguagem Java mostrou-se ser modelo para aprendizagem dos fundamentos de Programação Orientada a Objetos. O estudo dos fundamentos misturado a aplicação na prática puderam dar um vislumbre de como uma linguagem bem estruturada pode alcançar.