**UNIVERSIDADE POSITIVO**

**BACHARELADO EM SISTEMAS DE INFORMAÇÃO**

**DESEMPREGADOS**

**CURITIBA**

**SETEMBRO/2021**

**LUCAS RODRIGUES LEITE**

**THALES FUCKNER**

**VINÍCIUS KRUCHELSKI GUGELMIN**

**DESEMPREGADOS**

Trabalho de conclusão de disciplina Análise e Projeto de Sistemas e Desenvolvimento de Software Visual ao curso Bacharelado em Sistemas de Informação da Universidade Positivo.

Orientadores: Evandro Antunes Zatti e Diogo Deconto

**CURITIBA**

**SETEMBRO/2021**

**RESUMO**

Dentre as principais preocupações de um ingresso na faculdade a mais crítica é em relação a forma que irá arranjar dinheiro para pagar seu devido curso. Isto se agrava ainda mais se tratando de um país onde a desigualdade social é imensa, e comumente há em uma sala de aula de nível superior um aluno, que acabou de sair do Ensino Médio e está à procura de seu primeiro estágio, e outro, que batalha todo dia para sustentar sua família em um emprego de área distinta de seu curso. Com base nisto, este projeto foi realizado avaliando desde os perfis mais juvenis, e adaptando-o também para as necessidades adultas e mais definidas. Foi selecionada uma ferramenta já existente, site de busca de empregos, o qual é frequentemente usado cotidianamente por pessoas que necessitam de um emprego ou até mesmo por quem já trabalha e almeja uma função mais valorizada, e trabalhada para constituir uma inteligência mais complexa e eficaz para não necessitar o gasto de tempo normalmente necessário para efetuar tarefas simples – cadastrar perfil universitário; procurar vagas relacionadas; candidatar-se e aguardar – a fim de facilitar a vida tanto de quem está à procura de estudantes para trabalhar em sua empresa, quanto de quem está na busca de uma nova oportunidade.

Palavras-chave: faculdade, curso, emprego, ferramenta, inteligência.

**LISTA DE ILUSTRAÇÕES**

FIGURA 1 – BPMN - "LOGAR" ESTUDANTES 5

FIGURA 2 – BPMN - EDITAR DADOS DE CURSOS 6

FIGURA 3 – BPMN - CADASTRAR VAGAS DE EMPRESAS 6

FIGURA 4 – DIAGRAMA DE CLASSE - UC001 7

FIGURA 5 – DIAGRAMA DE CLASSE - UC002 7

FIGURA 6 – DIAGRAMA DE CLASSE - UC003 8

SUMÁRIO

[1. INTRODUÇÃO 1](#_Toc83755025)

[1.1. APRESENTAÇÃO DO TEMA 1](#_Toc83755026)

[1.2. PROBLEMAS A SOLUCIONAR 1](#_Toc83755027)

[1.3. JUSTIFICATIVA 1](#_Toc83755028)

[1.4. OBJETIVO GERAL 2](#_Toc83755029)

[1.5. OBJETIVOS ESPECÍFICOS 2](#_Toc83755030)

[2. DEFINIÇÃO DE REQUISITOS 3](#_Toc83755031)

[2.1. REQUISITOS FUNCIONAIS 3](#_Toc83755032)

[2.2. REQUISITOS NÃO-FUNCIONAIS 3](#_Toc83755033)

[3. ARQUITETURA DO SISTEMA 5](#_Toc83755034)

[3.1. DIAGRAMAS BPMN 5](#_Toc83755035)

[3.2. DIAGRAMAS DE CASO DE USO 7](#_Toc83755036)

[4. ESPECIFICAÇÃO DOS REQUISITOS DE SISTEMA 9](#_Toc83755037)

[4.1. REQUISITOS FUNCIONAIS 9](#_Toc83755038)

[4.2. REQUISITOS NÃO-FUNCIONAIS 23](#_Toc83755039)

[5. MODELOS DO SISTEMA 30](#_Toc83755040)

# INTRODUÇÃO

É inegável que com o gradual crescimento e melhorias da tecnologia, significantes partes e atividades manuais da sociedade perdem espaço para tarefas mais automatizadas, inteligentes e eficientes. Desde a facilitação de trabalhos industriais realizados por máquinas até pesquisas mais assertivas e verídicas, podemos contar com ajuda da tecnologia no nosso dia a dia. Entretanto, apesar da tecnologia está em qualquer lugar, o domínio e manuseio da mesma está fragmentado nas mãos de muitos, e aqueles poucos que detém vasto conhecimento dela, não se sentem dispostos em compartilhar todo o seu conhecimento para então fazer tudo fluir e evoluir. Todo esse enredo foi contado para chegar ao ponto do esquecimento da tecnologia, o detalhe de que muitos abandonam ou as vezes realmente deslembram de que existe a tecnologia para melhorar as coisas exaustivas de todo dia. Pensando em automatizar e inovar, este projeto procura eliminar o máximo de gasto de tempo possível preenchendo formulários e dados pessoais para a simples procura de emprego, dados, aqueles que são frequentemente preenchidos no cadastro para ingressos nas universidades, e estes, os fundamentais futuros clientes desde sistema.

## APRESENTAÇÃO DO TEMA

O objeto de estudo proposto neste trabalho é o desenvolvimento de uma ferramenta automatizada que auxilie os universitários no cadastramento de informações para a procura de vagas de emprego e elimine a necessidade de ter de gastar tempo olhando setores, requisitos e valores para vagas de emprego que realmente lhe interessam e se encaixam eu seu perfil.

## PROBLEMAS A SOLUCIONAR

Como propor às universidades a implementação de um sistema inovador e exclusivo para facilitar a obtenção de novos alunos trabalhadores e o interesse de empresas em cadastrar vagas no mesmo.

## JUSTIFICATIVA

O emprego, para a maior parte dos estudantes universitários, é a principal fonte de renda para pagar o curso superior ou a porta de entrada para encaminhar e direcionar sua carreira. Este sistema incorporaria todas as funcionalidades de um site padrão de ofertas de emprego de modo que o principal tempo necessário de interação do estudante com o mesmo fosse a inserção de seu número de matrícula e senha fornecida pela universidade para gerar ter acesso aos seus dados.

## OBJETIVO GERAL

O objetivo geral deste trabalho é desenvolver uma ferramenta automatizada por meio de pesquisas de campo que permitam o entendimento dos maiores problemas na busca de emprego pelos estudantes de ensino superior, buscando sempre inserir todas as funcionalidades já existentes em sites de busca de emprego, e criando um layout agradável e mais amigável, tanto para estudantes – que irão acessar o sistema para “logar” e visualizar seus resultados de vagas – quanto para empresas – no cadastramento de suas propostas.

## OBJETIVOS ESPECÍFICOS

Os objetivos específicos pertinentes a este estudo que auxiliarão para atingir o objetivo geral já citado são:

1. Realizar pesquisas com estudantes universitários a fim de entender as principais frustações na hora de procurar vagas de emprego;
2. Pesquisar quais as principais ferramentas e funcionalidades que existem para auxiliar na busca de emprego;
3. Formar ideias iniciais para a construção e automatização do sistema;
4. Definir uma metodologia ágil a ser implementada para o desenvolvimento do projeto;
5. Construir a estrutura do projeto através de diagramas e documentação em tópicos;
6. Criar um protótipo com layout básico, mas funcional, para iniciar a criação da inteligência por trás;
7. Realizar testes manuais e automáticos de funcionamento da ferramenta.

# DEFINIÇÃO DE REQUISITOS

Pensando no desenvolvimento do sistema, as funções gráficas, funções automatizadas, telas e layout ficaram divididas entre a equipe para cada um se responsabilizar pelo funcionamento inteiro de cada estrutura. Para organizar isto, foi utilizado como ferramenta inicial de comunicação o WhatsApp, podendo manter o contato dos membros em qualquer local e situação. Implementando conceitos de metodologia ágil, o grupo optou por utilizar o sistema de mapeamento e planejamento de execuções de tarefa do Kanban, conseguindo dar feedbacks sempre que em progresso de tarefas e términos. Outros conceitos individuais foram utilizados do Scrum, sempre utilizando o Discord – aplicativo de chamadas de voz e texto em grupo – para demarcar dúvidas e impedimentos na realização de tarefas. Concluindo o planejamento de requisitos funcionais e não funcionais, obteve-se os seguintes itens:

## REQUISITOS FUNCIONAIS

1. Cadastrar estudantes (simulando base de dados de uma Universidade);
2. “Logar” estudantes;
3. Editar dados de estudantes;
4. Remover estudantes;
5. Cadastrar administradores;
6. “Logar” administradores;
7. Editar dados de administradores;
8. Remover administradores;
9. Editar dados de cursos;
10. Cadastrar empresas;
11. Cadastrar vagas de empresas;
12. Editar dados de empresas;
13. Remover vagas de empresas;

## REQUISITOS NÃO-FUNCIONAIS

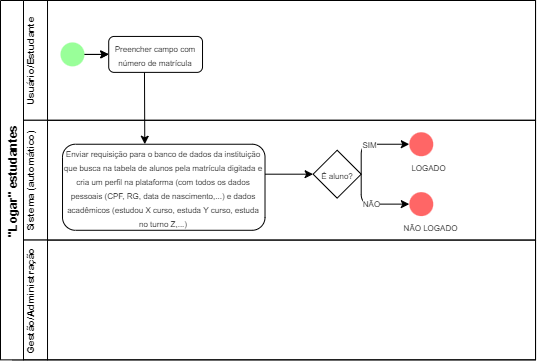
1. A base de dados deve ser protegida para acesso apenas de usuários autorizados, e esses, devem ser alunos da Universidade detentora do sistema;
2. O sistema precisa ser compatível com todos os sistemas operacionais e navegadores;
3. O tempo de desenvolvimento não pode ultrapassar 6 meses;
4. O sistema deve haver um cadastro seguro na validação de dados de acordo com a base de dados da Universidade;
5. Deve haver um script diário para verificar mudança no cadastro dos usuários ou no cancelamento dos mesmos;
6. O sistema deve conter respostas automáticas sem falhas na execução;
7. A administração do sistema precisa ser ágil e ter tempo de solução de problemas curto;
8. Precisa haver uma tela de cadastro simples, e que permita apenas o preenchimento do número de matrícula equivalente a que está ativa na Universidade detentora do sistema, e uma senha gerada pela própria Universidade;
9. Deve haver uma página de informações do usuário, onde as informações serão retornadas pela própria base de dados da Universidade, e um botão de requisitar alterações de determinados dados;
10. O sistema precisa ter um administrador geral, podendo gravar relatórios periódicos de movimentação de usuários;
11. O site precisa ter uma central de notificações que retornem dados de possíveis chamadas de emprego e status do curso.

# ARQUITETURA DO SISTEMA

Para desenvolvimento de diagramas e documentos de texto não programáveis, foi optado pela utilização de programas do pacote Office – Word e Excel – e programas online de compartilhamento via Google Drive – como o Draw.io.

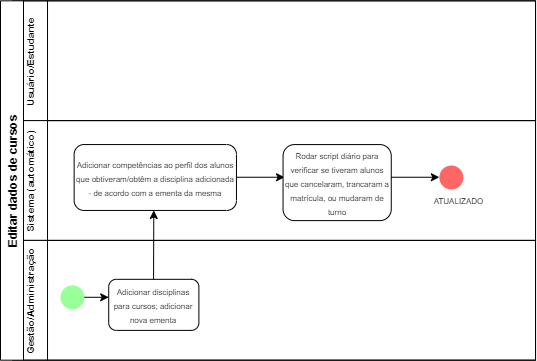
## DIAGRAMAS BPMN

FIGURA 1 – BPMN - "LOGAR" ESTUDANTES



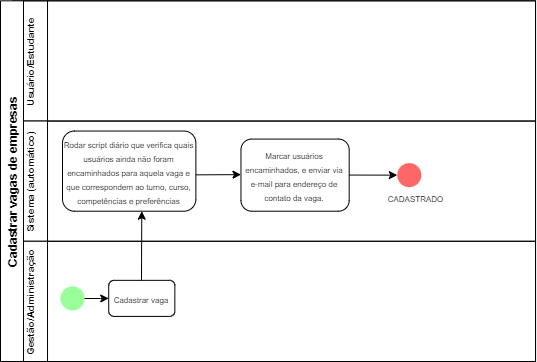
Fonte: OS AUTORES (2021)

FIGURA 2 – BPMN - EDITAR DADOS DE CURSOS



Fonte: OS AUTORES (2021)

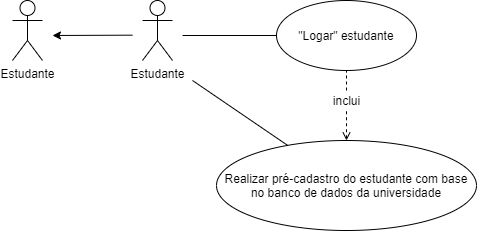
FIGURA 3 – BPMN - CADASTRAR VAGAS DE EMPRESAS



Fonte: OS AUTORES (2021)

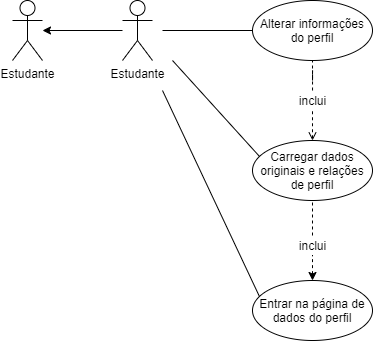
## DIAGRAMAS DE CASO DE USO

FIGURA 4 – DIAGRAMA DE CLASSE - UC001



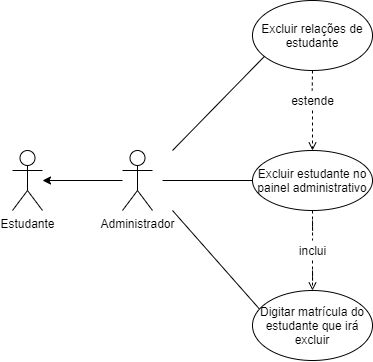
Fonte: OS AUTORES (2021)

FIGURA 5 – DIAGRAMA DE CLASSE - UC002



Fonte: OS AUTORES (2021)

FIGURA 6 – DIAGRAMA DE CLASSE - UC003



Fonte: OS AUTORES (2021)

# ESPECIFICAÇÃO DOS REQUISITOS DE SISTEMA

## REQUISITOS FUNCIONAIS

1. Cadastrar estudantes (simulando base de dados de uma Universidade);
   * Função

Cadastrar estudantes na base de dados do sistema.

* + Descrição

Cadastrar estudantes na base de dados do sistema através da leitura de arquivo JSON – simulando API de retorno de estudantes da base de dados da universidade.

* + Entradas

Leitura de arquivo JSON de estudantes.

* + Origem

Arquivo JSON de estudantes – simulando API de retorno de estudantes da base de dados da universidade.

* + Saídas

Mensagem de sucesso na inserção de itens – estudantes – na tabela de estudantes do banco de dados.

* + Destino

Tabela “estudante” no banco de dados; preenchimento da base de dados principal.

* + Ação

Arquivo JSON de estudantes criado com base em dados de alguns alunos é lido e inserido item por item no banco de dados para formar a tabela de estudantes do sistema.

* + Requer

Criação do arquivo JSON através de pesquisa dos alunos e código automático para ler o mesmo arquivo.

* + Pré-condição

Banco de dados deve estar operante, arquivo deve conter uma quantidade significante de dados para desenvolvimento e funcionamento do sistema.

* + Pós-condição

O sistema passa a conter dados significantes.

* + Efeitos colaterais

Criação de item na tabela “estudante”.

1. “Logar” estudantes;
   * Função

“Logar” estudantes na plataforma.

* + Descrição

“Logar” estudantes na plataforma através do preenchimento do formulário de login.

* + Entradas

Preenchimento de formulário de login com código de matrícula e senha fornecida pela instituição de ensino.

* + Origem

Formulário de login.

* + Saídas

Dados do estudante.

* + Destino

Acesso ao sistema e visualização dos dados na tela de perfil.

* + Ação

Inserção dos dados do usuário, validação dos mesmos, e redirecionamento de página para tela de perfil com dados restantes preenchidos.

* + Requer

Estudante estar com status ativo.

* + Pré-condição

Banco de dados deve estar operante e estudante deve existir e estar ativo.

* + Pós-condição

Os dados do usuário devem estar acessíveis a visualização pelo mesmo.

* + Efeitos colaterais

Nenhum.

1. Editar dados de estudantes;
   * Função

Editar estudantes na plataforma.

* + Descrição

Editar dados de estudantes dentro da plataforma.

* + Entradas

Inserção de dados de estudante.

* + Origem

Tela de perfil do estudante.

* + Saídas

Dados do estudante atualizados.

* + Destino

Tabela “estudante” no banco de dados; visualização dos dados atualizados na tela de perfil.

* + Ação

Inserção de novos dados do usuário, validação dos mesmos, e atualização de dados na página de perfil.

* + Requer

Estudante estar com status ativo.

* + Pré-condição

Banco de dados deve estar operante, estudante deve existir e estar ativo e dados devem ser válidos.

* + Pós-condição

Os dados antigos não devem existir mais.

* + Efeitos colaterais

Atualização de item da tabela “estudante”.

1. Remover estudantes;
   * Função

Remover estudantes.

* + Descrição

Remover estudantes da plataforma.

* + Entradas

Identificação do estudante a ser deletado e do administrador que irá deleta-lo.

* + Origem

Administrador da plataforma.

* + Saídas

Mensagem de sucesso na remoção do usuário.

* + Destino

Tabela “estudante” no banco de dados; painel administrativo.

* + Ação

Remoção do item na tabela de “estudante” do banco de dados.

* + Requer

Administrador estar logado.

* + Pré-condição

Banco de dados deve estar operante e administrador deve existir.

* + Pós-condição

O estudante não deve existir mais.

* + Efeitos colaterais

Remoção de item da tabela “estudante”.

1. Cadastrar administradores;
   * Função

Cadastrar administradores na base de dados do sistema.

* + Descrição

Cadastrar administradores na base de dados do sistema através de formulário de cadastro no “admin”.

* + Entradas

Dados do administrador a ser criado.

* + Origem

Cadastro no “admin” da plataforma.

* + Saídas

Dados do administrador.

* + Destino

Tabela “administrador” no banco de dados; “admin” da plataforma.

* + Ação

Dados do formulário de cadastro de administrador são lidos, validados e inseridos na tabela de “administrador” no banco de dados.

* + Requer

Cadastro prévio de administrador pelo código.

* + Pré-condição

Banco de dados deve estar operante e administrador deve estar logado para criar um novo.

* + Pós-condição

O administrador é registrado na tabela de “administrador” e pode “logar” no “admin”.

* + Efeitos colaterais

Criação de item na tabela “administrador”.

1. “Logar” administradores;
   * Função

“Logar” administradores na plataforma administrativa.

* + Descrição

“Logar” administradores na plataforma através do preenchimento do formulário de login no “admin”.

* + Entradas

Preenchimento de formulário de login do “admin” com e-mail e senha cadastrada.

* + Origem

Formulário de login do “admin”.

* + Saídas

Dados do administrador.

* + Destino

Acesso ao “admin”.

* + Ação

Inserção dos dados do administrador, validação dos mesmos, e redirecionamento de página para tela inicial do “admin”.

* + Requer

Administrador deve estar cadastrado.

* + Pré-condição

Banco de dados deve estar operante e administrador deve existir e estar ativo.

* + Pós-condição

O administrador deve ter controle do site pela plataforma administrativa.

* + Efeitos colaterais

Nenhum.

1. Editar dados de administradores;
   * Função

Editar administradores na plataforma administrativa.

* + Descrição

Editar dados de administradores dentro do “admin”.

* + Entradas

Inserção de dados de administrador.

* + Origem

Tela de edição de administradores no “admin”.

* + Saídas

Dados do administrador atualizados.

* + Destino

Tabela “administrador” no banco de dados; visualização dos dados atualizados na lista de administradores no “admin”.

* + Ação

Inserção de novos dados do administrador, validação dos mesmos, e atualização de dados na lista de administradores no “admin”.

* + Requer

Administrador estar com status ativo.

* + Pré-condição

Banco de dados deve estar operante, administrador deve existir e estar ativo e dados devem ser válidos.

* + Pós-condição

Os dados antigos não devem existir mais.

* + Efeitos colaterais

Atualização de item da tabela “administrador”.

1. Remover administradores;
   * Função

Remover administradores.

* + Descrição

Remover administradores do “admin”.

* + Entradas

Identificação do administrador a ser deletado e do administrador que irá deleta-lo.

* + Origem

Administrador da plataforma.

* + Saídas

Mensagem de sucesso na remoção do administrador.

* + Destino

Tabela “administrador” no banco de dados; painel administrativo.

* + Ação

Remoção do item na tabela de “administrador” do banco de dados.

* + Requer

Administrador estar logado; administrador ser o que foi registrado pelo código – principal.

* + Pré-condição

Banco de dados deve estar operante e administrador deve existir.

* + Pós-condição

O administrador não deve existir mais.

* + Efeitos colaterais

Remoção de item da tabela “administrador”.

1. Editar dados de cursos;
   * Função

Editar cursos na plataforma.

* + Descrição

Editar dados de cursos dentro da plataforma.

* + Entradas

Inserção de dados de cursos.

* + Origem

Administrador da plataforma.

* + Saídas

Dados do curso atualizados.

* + Destino

Tabela “curso” no banco de dados; visualização dos dados atualizados na listagem no administrador.

* + Ação

Inserção de novos dados do curso, validação dos mesmos, e atualização de dados.

* + Requer

Curso estar com status ativo.

* + Pré-condição

Banco de dados deve estar operante, curso deve existir e estar ativo e dados devem ser válidos.

* + Pós-condição

Os dados antigos não devem existir mais.

* + Efeitos colaterais

Atualização de item da tabela “curso”.

1. Cadastrar empresas;
   * Função

Cadastrar empresas na base de dados do sistema.

* + Descrição

Cadastrar empresas na base de dados do sistema através de formulário de cadastro na plataforma.

* + Entradas

Dados da empresa a ser criado.

* + Origem

Cadastro na plataforma.

* + Saídas

Dados da empresa.

* + Destino

Tabela “empresa” no banco de dados; tela de cadastro na plataforma.

* + Ação

Dados do formulário de cadastro de empresa são lidos, validados e inseridos na tabela de “empresa” no banco de dados.

* + Requer

Nenhum.

* + Pré-condição

Banco de dados deve estar operante e dados devem ser válidos.

* + Pós-condição

A empresa deve conseguir “logar” no sistema para a área de empresas.

* + Efeitos colaterais

Criação de item na tabela “empresa”.

1. Cadastrar vagas de empresas;
   * Função

Cadastrar vagas de empresas na base de dados do sistema.

* + Descrição

Cadastrar vagas de empresas na base de dados do sistema através de formulário de cadastro dentro da empresa.

* + Entradas

Dados da vaga de empresa a ser criado.

* + Origem

Cadastro na plataforma.

* + Saídas

Dados da vaga de empresa.

* + Destino

Tabela “vaga” no banco de dados; tela de cadastro na plataforma.

* + Ação

Dados do formulário de cadastro de vaga de empresa são lidos, validados e inseridos na tabela de “vaga” no banco de dados.

* + Requer

Empresa deve criar e ser ativa.

* + Pré-condição

Banco de dados deve estar operante e dados devem ser válidos.

* + Pós-condição

A vaga deve estar disponível a candidaturas automáticas de estudantes por parte do sistema.

* + Efeitos colaterais

Criação de item na tabela “vaga”.

1. Editar dados de empresas;
   * Função

Editar empresas na plataforma.

* + Descrição

Editar dados de empresa dentro da plataforma.

* + Entradas

Inserção de dados da empresa.

* + Origem

Tela de edição de empresa.

* + Saídas

Dados da empresa atualizados.

* + Destino

Tabela “empresa” no banco de dados; visualização dos dados atualizados.

* + Ação

Inserção de novos dados da empresa, validação dos mesmos, e atualização de dados.

* + Requer

Empresa estar com status ativo.

* + Pré-condição

Banco de dados deve estar operante, empresa deve existir e estar ativa e dados devem ser válidos.

* + Pós-condição

Os dados antigos não devem existir mais.

* + Efeitos colaterais

Atualização de item da tabela “empresa”.

1. Remover vagas de empresas;
   * Função

Remover vagas de empresas.

* + Descrição

Remover vagas de empresa.

* + Entradas

Identificação da vaga a ser deletada e da empresa que irá deleta-la.

* + Origem

Página de empresa.

* + Saídas

Mensagem de sucesso na remoção da vaga.

* + Destino

Tabela “vaga” no banco de dados; página de empresa.

* + Ação

Remoção do item na tabela de “vaga” do banco de dados.

* + Requer

Empresa estar logada; empresa ser dona da vaga.

* + Pré-condição

Banco de dados deve estar operante e vaga deve existir.

* + Pós-condição

A vaga não deve existir mais.

* + Efeitos colaterais

Remoção de item da tabela “vaga”.

## REQUISITOS NÃO-FUNCIONAIS

1. A base de dados deve ser protegida para acesso apenas de usuários autorizados, e esses, devem ser alunos da Universidade detentora do sistema;
   * Finalidade

Proteção e autorização.

* + Usabilidade

Rápido de implementar.

* + Confiabilidade

Falhas apenas em desenvolvimento.

* + Eficiência

Alta chance de escalabilidade e confiança no projeto.

* + Manutenibilidade

Esforço moderado para alteração seguindo os requisitos.

* + Portabilidade

Demanda uma certa dificuldade para se adequar a outros ambientes.

* + Segurança

Extremamente alta.

* + Compatibilidade

Normal.

1. O sistema precisa ser compatível com todos os sistemas operacionais e navegadores;
   * Finalidade

Compatibilidade.

* + Usabilidade

Rápido de implementar.

* + Confiabilidade

Falhas podem ser mais fáceis de existir.

* + Eficiência

Alta chance de escalabilidade

* + Manutenibilidade

Esforço alto-moderado para alteração.

* + Portabilidade

Demanda uma certa dificuldade para se adequar a outros ambientes.

* + Segurança

Média.

* + Compatibilidade

Extrema.

1. O tempo de desenvolvimento não pode ultrapassar 6 meses;
   * Finalidade

Agilidade e execução eficaz.

* + Usabilidade

Exaustivo.

* + Confiabilidade

Falhas são solucionadas mais rapidamente.

* + Eficiência

Alta confiança.

* + Manutenibilidade

Esforço inicial extremo.

* + Portabilidade

Demanda alta dificuldade inicial.

* + Segurança

Extremamente alta.

* + Compatibilidade

Dificuldade extrema.

1. O sistema deve haver um cadastro seguro na validação de dados de acordo com a base de dados da Universidade;
   * Finalidade

Proteção e autorização.

* + Usabilidade

Rápido de implementar.

* + Confiabilidade

Poucas falhas.

* + Eficiência

Confiança no projeto.

* + Manutenibilidade

Baixo esforço para alteração.

* + Portabilidade

Baixa dificuldade.

* + Segurança

Extremamente alta.

* + Compatibilidade

Normal.

1. Deve haver um script diário para verificar mudança no cadastro dos usuários ou no cancelamento dos mesmos;
   * Finalidade

Automatização.

* + Usabilidade

Dificuldade média de implementar.

* + Confiabilidade

Probabilidade de falhas na produção.

* + Eficiência

Alta chance de escalabilidade, rapidez e praticidade do projeto.

* + Manutenibilidade

Esforço demasiado para manutenção do projeto.

* + Portabilidade

Demanda uma certa dificuldade para se adequar a outros ambientes.

* + Segurança

Média.

* + Compatibilidade

Normal.

1. O sistema deve conter respostas automáticas sem falhas na execução;
   * Finalidade

Automatização e segurança.

* + Usabilidade

Dificuldade extrema de implementação.

* + Confiabilidade

Probabilidade de falhas na produção.

* + Eficiência

Alta chance de escalabilidade e confiança no projeto.

* + Manutenibilidade

Esforço moderado para manutenção.

* + Portabilidade

Demanda uma certa dificuldade para se adequar a outros ambientes.

* + Segurança

Extremamente alta.

* + Compatibilidade

Extrema.

1. A administração do sistema precisa ser ágil e ter tempo de solução de problemas curto;
   * Finalidade

Administração e coordenação.

* + Usabilidade

Dificuldade moderada.

* + Confiabilidade

Erro humano não padronizado.

* + Eficiência

Alta-moderada.

* + Manutenibilidade

Baixo esforço.

* + Portabilidade

Dificuldade inexistente.

* + Segurança

Baixa segurança.

* + Compatibilidade

Dificuldade inexistente.

1. Precisa haver uma tela de cadastro simples, e que permita apenas o preenchimento do número de matrícula equivalente a que está ativa na Universidade detentora do sistema, e uma senha gerada pela própria Universidade;
   * Finalidade

Praticidade e simplicidade.

* + Usabilidade

Baixa dificuldade de implementação.

* + Confiabilidade

Falhas quase inexistentes.

* + Eficiência

Utilização geral do sistema.

* + Manutenibilidade

Baixo esforço para manutenção.

* + Portabilidade

Fácil de implementação em outros ambientes.

* + Segurança

Normal.

* + Compatibilidade

Extrema.

1. Deve haver uma página de informações do usuário, onde as informações serão retornadas pela própria base de dados da Universidade, e um botão de requisitar alterações de determinados dados;
   * Finalidade

Controle de dados.

* + Usabilidade

Baixa dificuldade de implementação.

* + Confiabilidade

Probabilidade de falhas na produção.

* + Eficiência

Confiança no projeto.

* + Manutenibilidade

Esforço normal.

* + Portabilidade

Fácil adequação.

* + Segurança

Normal.

* + Compatibilidade

Normal.

1. O sistema precisa ter um administrador geral, podendo gravar relatórios periódicos de movimentação de usuários;
   * Finalidade

Administração, coordenação e automatização.

* + Usabilidade

Dificuldade moderada de implementação.

* + Confiabilidade

Probabilidade de falhas na produção.

* + Eficiência

Alta chance de escalabilidade e praticidade do projeto.

* + Manutenibilidade

Esforço normal para manutenção.

* + Portabilidade

Normal.

* + Segurança

Alta.

* + Compatibilidade

Normal.

1. O site precisa ter uma central de notificações que retornem dados de possíveis chamadas de emprego e status do curso.
   * Finalidade

Automatização e praticidade.

* + Usabilidade

Dificuldade extrema de implementação.

* + Confiabilidade

Probabilidade de erros internos não visíveis para o usuário.

* + Eficiência

Alta chance de escalabilidade e valor ao projeto.

* + Manutenibilidade

Grande esforço exigido.

* + Portabilidade

Demanda uma certa dificuldade para se adequar a outros ambientes.

* + Segurança

Extremamente alta.

* + Compatibilidade

Média.

# MODELOS DO SISTEMA