FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS

FATEC PROFESSOR Jessen Vidal

LUAN MATHEUS SATIRO DE OLIVEIRA

NLEARN (PLATAFORMA DE APRENDIZADO)

Orientador: Fabricio Galende Marques de Carvalho

São José dos Campos

2020

SUMÁRIO

[1 Introdução 3](#_Toc49841023)

[1.1 Definição do problema 4](#_Toc49841024)

[1.2 Objetivo 4](#_Toc49841025)

[1.3 Escopo 5](#_Toc49841026)

1.4 Ferramental tecnológico 6

[2 Desenvolvimento 6](#_Toc49841027)

[2.1 Arquitetura 7](#_Toc49841028)

[2.2 Modelo de Dados 12](#_Toc49841029)

[2.3 Processo dinamico 16](#_Toc49841030)

[3 Resultados e Discussão 17](#_Toc49841031)

# Introdução

O problema abordado neste relatório consiste em uma dificuldade dos novos ingressantes nos meios de maratonas de programação, assim como, sobre uma falta de espaço exclusivo para monitores. O objetivo consiste em fornecer uma maneira alternativa e direta de encontrar conteúdo para as resoluções dos exercícios, assim como, disponibilizar um espaço exclusivo para os monitores adicionarem conteúdo. Com base nos tópicos apresentados a seguir, é possível visualizar uma contribuição para a comunidade de maratonas de programação, auxiliando e facilitando o trajeto de novos ingressantes.

Este projeto será desenvolvido para a matéria de laboratório de engenharia de software alinhado ao trabalho de graduação do curso de analise e desenvolvimento de sistemas da FATEC.

## Definição do problema

O problema localizado está sobre a dificuldade de localizar conteúdo específico para resoluções de determinados exercícios, assim como, um espaço focado para monitores de matérias de programação, onde os mesmos não possuem um local exclusivo para adicionar conteúdo para os alunos, dessa forma, os monitores podem avaliar conteúdos pertinentes de determinada matéria e adicionar na plataforma para que os alunos possam visualizar aquelas informações e terem um melhor entendimento de determinado assunto.

As plataformas como URI e Ucoderfornecem exercícios de maratonas de programação e um sistema que recebe a resolução para verificar se a resposta é válida baseado no modelo de maratona, as plataformas também fornecem um fórum para conversar sobre dificuldades encontradas, no entanto, o foco do fórum é para esclarecer dúvidas especificas das resoluções e não de fato sobre o caminho necessário para chegar em determinada resolução.

Se situar no conteúdo de algoritmo de uma linguagem de programação pode ser extenso, inicialmente isso pode gerar uma dificuldade e criar um desencorajamento para o aluno em prosseguir neste caminho.

Um exemplo prático e simples: existe um algoritmo para resolver determinado problema, no entanto, o aluno não possui o conhecimento da existência do mesmo e não teve um direcionamento eficiente, dessa forma o aluno irá perder horas para realizar a resolução do exercício de uma forma ‘’alternativa’’ e isso tornará o exercício bem mais complexo, dessa forma, o aluno se sentirá frustrado em não conseguir encontrar a resolução adequada.

## Objetivo

O objetivo é desenvolver uma plataforma de aprendizado separado por tópicos com guias provenientes da própria documentação da linguagem ou conteúdos criados pelos monitores de programação, orientando as etapas de aprendizado para conseguir resolver problemas de maratona de programação ou algo relacionado a programação.

## Escopo

O sistema possuirá uma página administrativa na qual se encarregara de adicionar, alterar ou remover informações. Nessas funcionalidades haverá duas seções, adicionar linguagem e adicionar conteúdo para a linguagem, para a seção linguagem os campos referentes são adicionar título, descrição, imagem, e key e os campos da seção conteúdo serão o título, subtítulo de um conteúdo, conteúdo, exercícios, key do tópico e key da linguagem.

Somente os monitores do sistema poderão realizar essas alterações, e cada monitor só pode alterar o conteúdo que ele mesmo tenha postado, também haverá um usuário administrador, e esse usuário administrador além de poder também inserir, alterar ou deletar linguagens ou conteúdos, poderá deletar monitores do sistema, e ao deletar um monitor isso também deletará todo o conteúdo atribuído ao mesmo

Funcionalidades:

1. Exclusão de um monitor: o usuário administrador pode excluir monitores do sistema.
2. Cadastrar nome do monitor: O administrador poderá adicionar um nome para o monitor.
3. Cadastrar senha do monitor: O administrador poderá cadastrar uma senha para o monitor.
4. Cadastrar contato do monitor: O administrador poderá adicionar o contato do monitor.
5. Adicionar título da linguagem: O monitor pode adicionar o título de uma linguagem pela pagina administrativa.
6. Adicionar descrição da linguagem: O monitor pode adicionar a descrição de uma linguagem pela pagina administrativa.
7. Adicionar imagem da linguagem: O monitor pode adicionar uma imagem para linguagem pela pagina administrativa.
8. Adicionar a key da linguagem: O monitor pode adicionar a key de uma linguagem pela pagina administrativa.
9. Alterar título da linguagem: O monitor pode alterar o título de uma linguagem pela pagina administrativa.
10. Alterar descrição da linguagem: O monitor pode alterar a descrição de uma linguagem pela pagina administrativa.
11. Alterar imagem da linguagem: O monitor pode alterar uma imagem para linguagem pela pagina administrativa.
12. Alterar a key da linguagem: O monitor pode alterar a key de uma linguagem pela pagina administrativa.
13. Excluir linguagem: O monitor pode excluir a linguagem do sistema.
14. Adicionar título do conteúdo: O monitor pode adicionar o título de um post pela pagina administrativa.
15. Adicionar subtítulo do conteúdo: O monitor pode adicionar o subtítulo de um post pela pagina administrativa.
16. Adicionar conteúdo: O monitor pode adicionar o conteúdo de um post pela pagina administrativa.
17. Adicionar exercício: O monitor pode adicionar o exercício de um post pela pagina administrativa.
18. Adicionar key do tópico do conteúdo: O monitor pode adicionar a key do tópico de um post pela pagina administrativa.
19. Adicionar key da linguagem para atribuir ao conteúdo: O monitor pode adicionar a key da linguagem de um post pela pagina administrativa.
20. Alterar título do conteúdo: O monitor pode alterar o título de um post pela pagina administrativa.
21. Alterar subtítulo conteúdo: O monitor pode alterar o subtítulo de um post pela pagina administrativa.
22. Alterar conteúdo: O monitor pode alterar o conteúdo de um post pela pagina administrativa.
23. Alterar exercício: O monitor pode alterar o exercício de um post pela pagina administrativa.
24. Alterar key do tópico do conteúdo: O monitor pode adicionar a key do tópico de um post pela pagina administrativa.
25. Alterar key da linguagem para atribuir ao conteúdo: O monitor pode adicionar a key da linguagem de um post pela pagina administrativa.
26. Excluir conteúdo: O monitor pode excluir um conteúdo da página administrativa.

## Ferramental tecnológico

A seguir está descrito as tecnologias utilizadas no sistema, assim como, para que os mesmos servem.

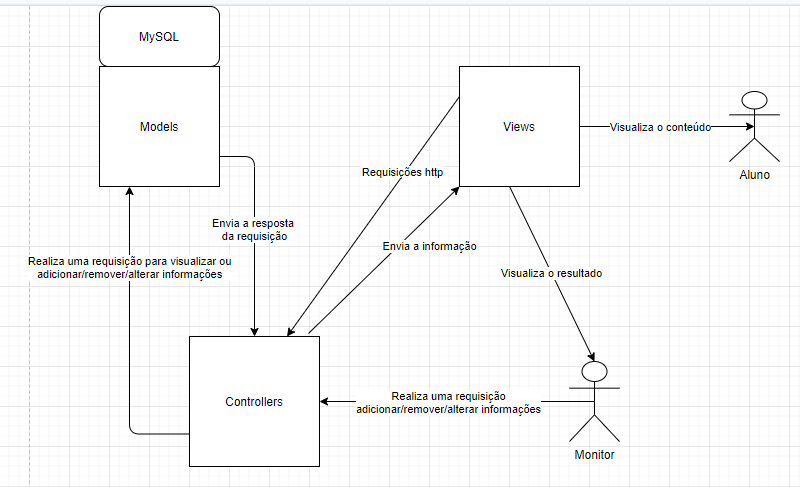
Escopo tecnológico:

1. Python no Back-end.
2. Jinja2 para geração de páginas dinâmicas.
3. Gunicorn como servidor de implantação, em conjunto com o Flask.
4. SqlAlchemy para persistência de dados.
5. Flask para implantar o sistema web.
6. Bootstrap framework para o front-end
7. Html5 para estruturar a página.
8. Css3 para estilizar a página.
9. Virtualenv para isolamento de ambiente de desenvolvimento e obtenção de pacotes.
10. Javascript para funcionalidades voltadas ao front-end.

# Desenvolvimento

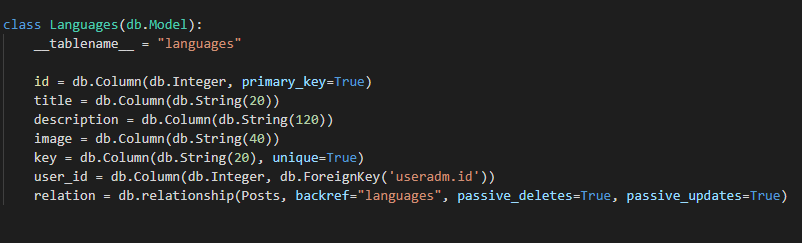
Este Capítulo apresenta detalhes sobre o desenvolvimento do sistema para auxiliar os alunos da FATEC São José dos Campos, oferecendo uma plataforma para visualização de assuntos pertinentes aos cursos Análise e Desenvolvimento de Sistemas e Banco de Dados.

## Arquitetura

**Figura 1** – Arquitetura do sistema

Neste projeto foi utilizado o formato de estrutura MVC (MVC é o acrônimo de Model-View-Controller é um padrão de projeto de software, ou padrão de arquitetura de software) explicarei a seguir por etapas:

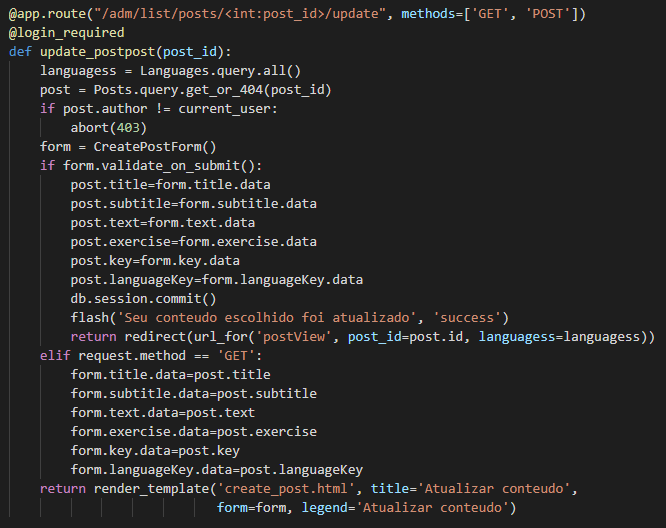
**Model:** É responsável pela leitura e escrita de dados, e também de suas validações.

**Figura 2** – Classe Languages do model

No código demonstrado acima é realizado a descrição da estrutura de uma das tabelas do banco, como demonstrado, é necessário definir seus atributos, tipos e relações, dessa forma, ao executar os comandos provenientes da tecnologia Flask, as tabelas serão criadas no banco de acordo com as especificações definidas no código.

Neste exemplo temos um id que é a chave primária e é constituído de um inteiro, após isso, temos o título, descrição, imagem e key, todas do tipo texto, com uma particularidade onde a key é um atributo único, ou seja, não pode existir valores iguais em linguagens diferentes, a seguir temos uma chave estrangeira fazendo referencia ao id da tabela usuário e por fim uma relação na qual implica que quando uma linguagem for excluída ou atualizada, todos os seus filhos também serão deletados ou atualizados.

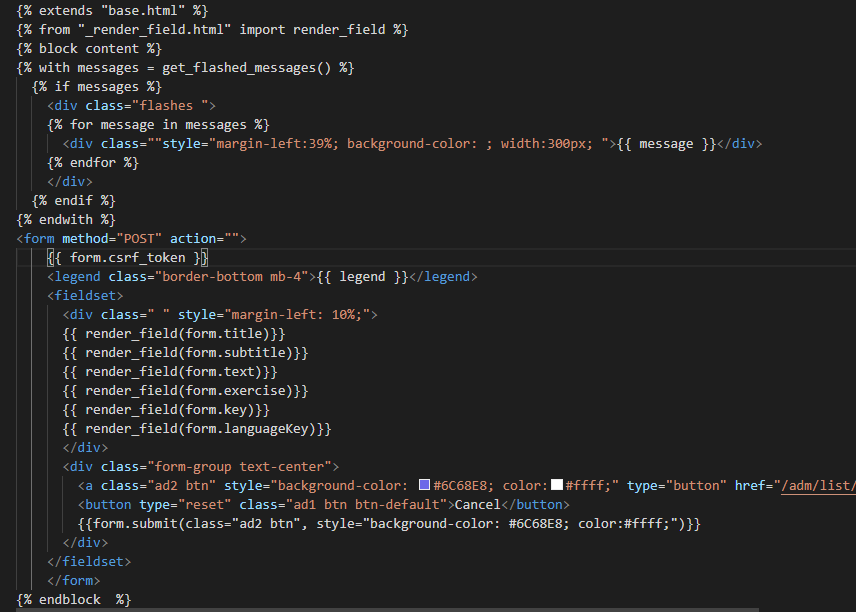
**Controller:** Um controlador é responsável por controlar a maneira como um monitor interage com uma aplicação MVC e possui o fluxo de controle lógico para uma aplicação.

**Figura 3** – Rota para atualizar um conteúdo

No exemplo acima é demonstrado dois tipos de requisições, get e pot, para pegar e enviar conteúdo para o banco, agora descreverei as funções seguidos por uma lista:

* Variável languagess: Essa variável é responsável por pegar todo o conteúdo existente na tabela Languages, para que possa ser exibido os títulos das linguagens no menu de navegação do site.
* Variável post: Essa variável é responsável por pegar o id do post que o usuário está no momento.
* Form: Essa variável é responsável por criar o formulário da classe CreatePostForm.
* Primeiro if: Esse if serve para validar se o monitor logado é o mesmo que criou aquele post, caso não seja, ele abortara a ação, pois somente quem criou o post pode o atualizar ou deletar.
* Segundo if: Esse if serve para realizar uma validação, ou seja, se o botão de finalizar for acionado, os dados abaixo serão inseridos numa variável que realizara o commit para o banco, ou seja, salvará as informações obtidas nos forms e se isso for feito retornara um flash, que é uma mensagem que a linguagem foi atualizada, o return desse if retornará para outra função na qual exibirá todos os posts existentes.
* Elif: Esse elif serve para pegar as informações existentes no banco para que o monitor visualize o conteúdo já existente para decidir o que de fato quer atualizar em cada tópico.
* Return: o return da função update\_postpost serve para que casa nenhuma condicional seja executada, retorne a página de criação de conteúdo.

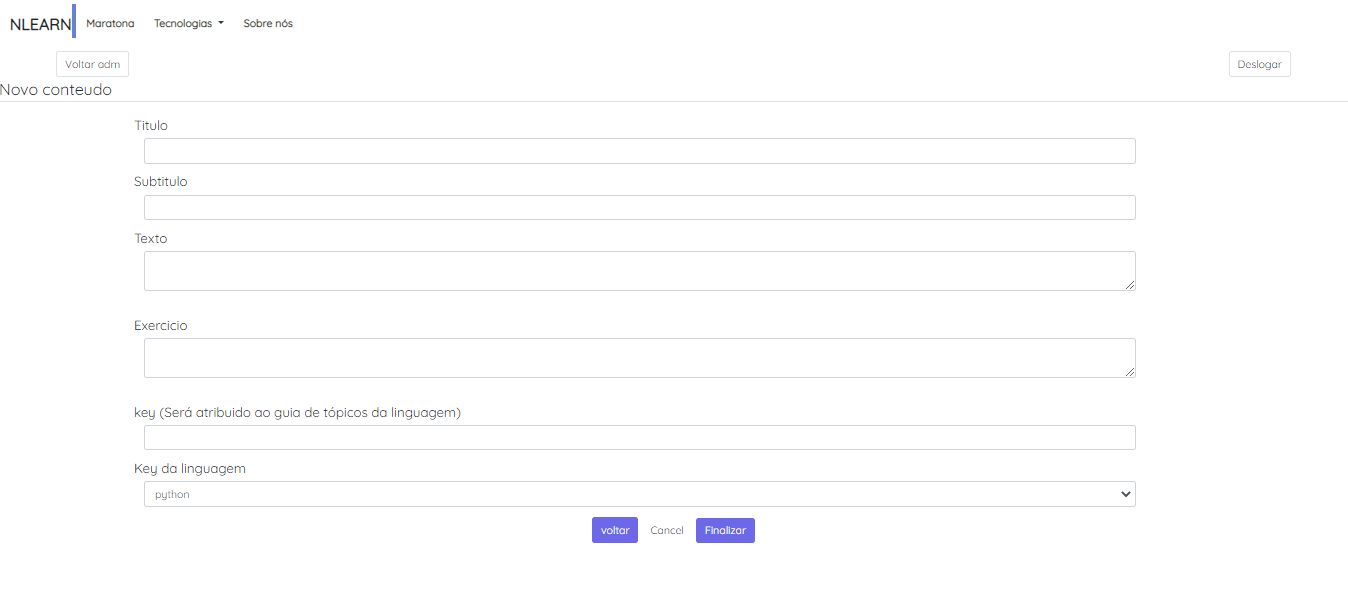
**View:** Consiste na camada de interação com o usuário onde é exibido os dados

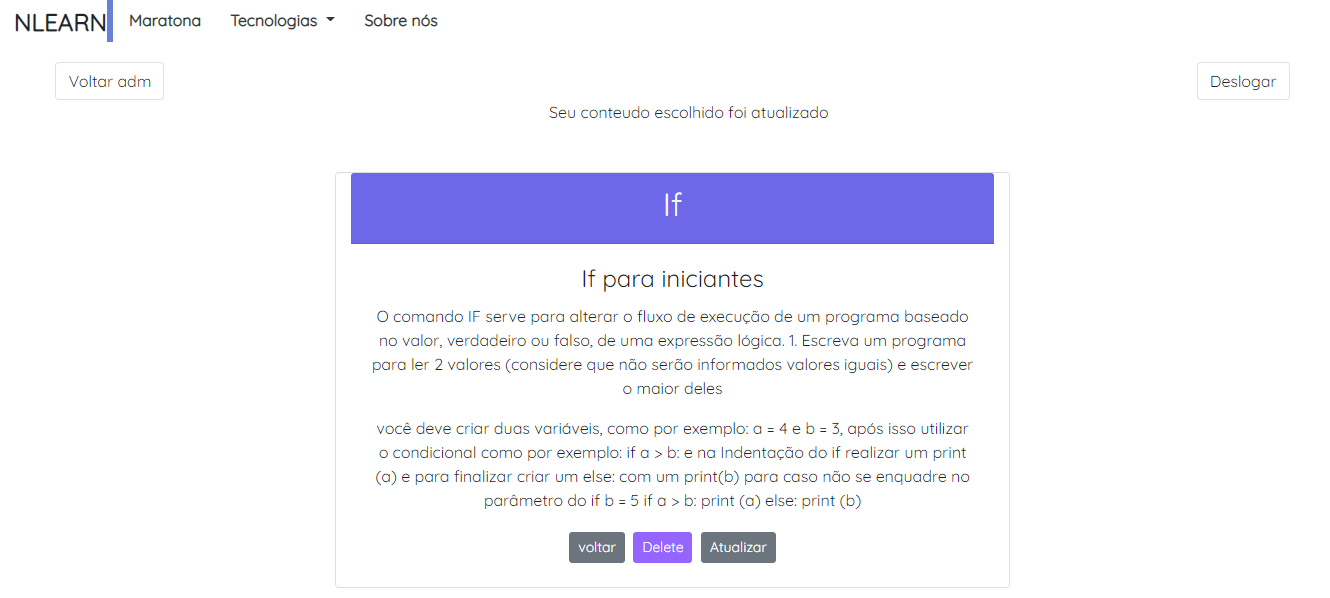
**Figura 4** – Template para criação de conteúdo

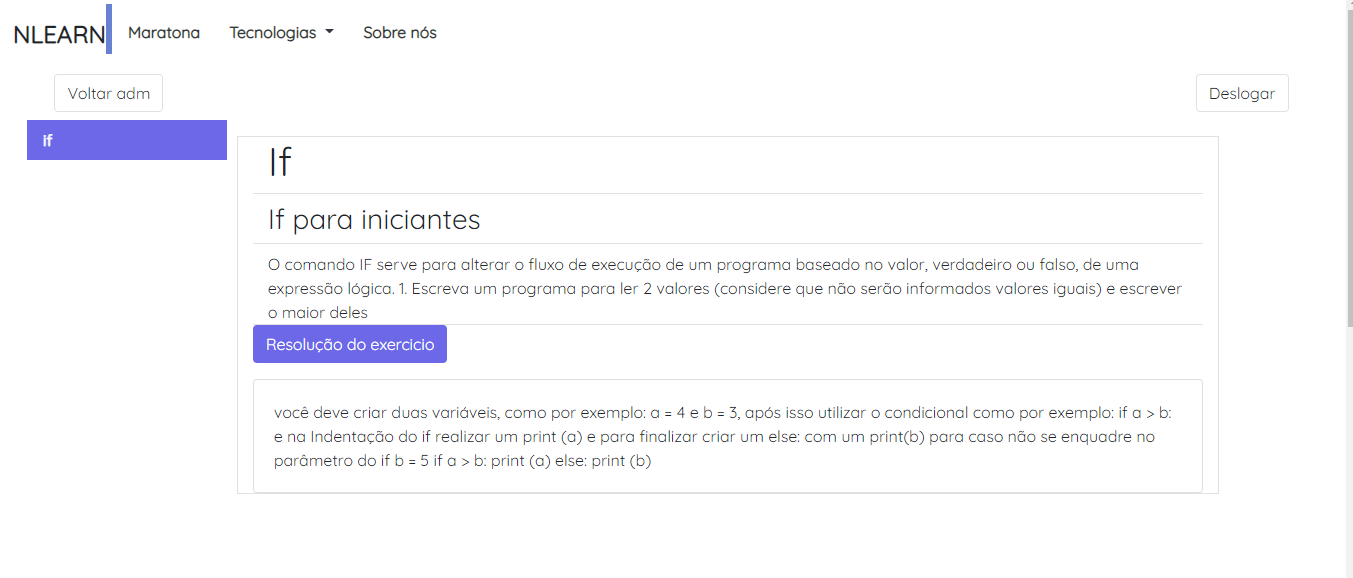
No exemplo acima é demonstrado a utilização de jinja, html, css e boostrap, eu tenho um parâmetros block que é estendido do template base que é a estrutura geral do site, após isso temos parâmetros chamando minha mensagem flash e a seguir tenho containers(<div>) com as configurações de estilo para determinado container, abaixo tenho a tag <form> onde eu passo a tag <fieldset> que é basicamente onde eu passo as variáveis do meu formulário para que o monitor possa digitar e enviar para os controllers.

A seguir será demonstrado o que o monitor poderá visualizar de acordo com o template acima.

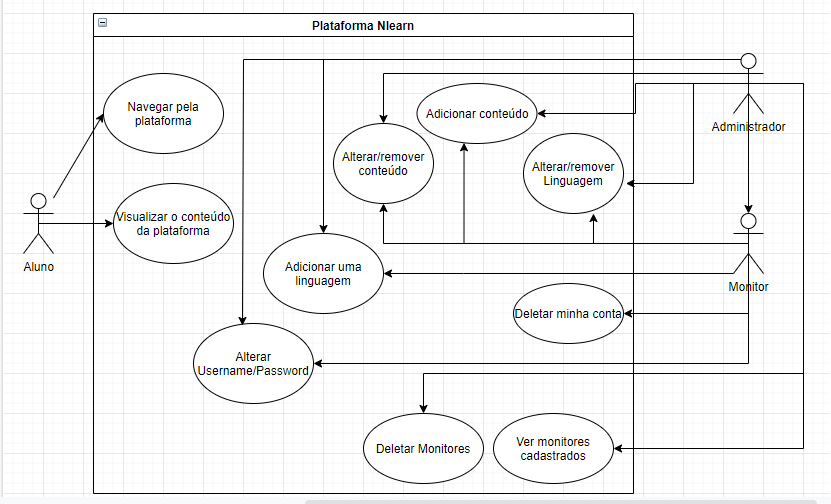
.

**Figura 5** – View para criação de conteúdo

 **Figura 6** – View de exibição de conteúdo para alterar/deletar

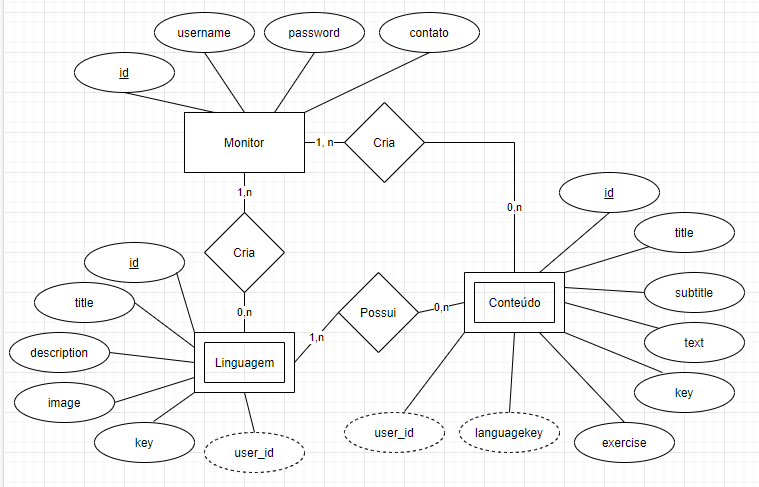
**Figura 7** – View de exibição de conteúdos cadastrados nas linguagens

## Modelo de Dados

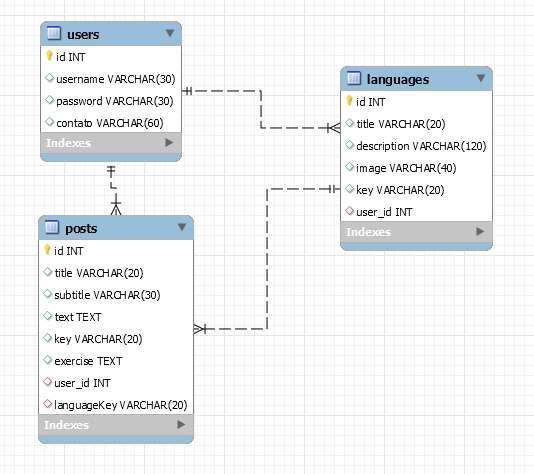
**Figura 8** – Caso de uso

Como demonstrado no caso de uso acima, tenho 3 atores, o aluno poderá somente visualizar o conteúdo na plataforma ou navegar entre as páginas, o monitor poderá adicionar/alterar/remover linguagens ou conteúdo para linguagens, sendo que, os mesmos só podem realizar ações sobre o conteúdo que os mesmo criaram e também podem atualizar seus dados como usuário e senha ou deletar a própria conta, e por fim, temos o administrador, no caso, o administrador tem os mesmos direitos dos monitores com o acréscimo de poder visualizar todos os monitores cadastrados ou realizar a deleção deles, é importante salientar que o usuário administrador pertence a mesma tabela do monitor, com a diferença que o administrador tem uma flag no código onde o permite acessar uma rota para administrar monitores

Figura 9 – Diagrama entidade relacionamento (modelo conceitual)



Como demonstrado no diagrama entidade relacionamento acima eu tenho o monitor com 4 atributos e duas relações, no caso, o monitor pode criar nenhuma ou muitas linguagens, assim como, pode criar nenhum ou muitos conteúdos, a seguir tenho a linguagem com 6 atributos e uma relação com o conteúdo, onde uma linguagem possui nenhum ou muitos conteúdos, sendo que a linguagem só pode existir se um monitor ou administrador a cadastrar, e por fim, temos a entidade conteúdo, onde a mesma possui 8 atributos e a mesma só pode existir caso exista uma linguagem e um monitor/administrador, logo, um conteúdo pode existir em um ou nenhum monitor/administrador e um conteúdo pode existir em uma ou nenhuma linguagem.

**Figura 10** – Modelo lógico

No modelo lógico acima, eu tenho três tabelas, a tabela de posts(conteúdo), languages(linguagens), users(monitores/administrador).

Na tabela users, tenho quatro atributos:

* ID: É uma chave primária que serve para identificar qual é o usuário.
* Username: Para o nome.
* Password: Para fazer as validações de login na plataforma.
* Contato: Para inserir o contato do monitor.

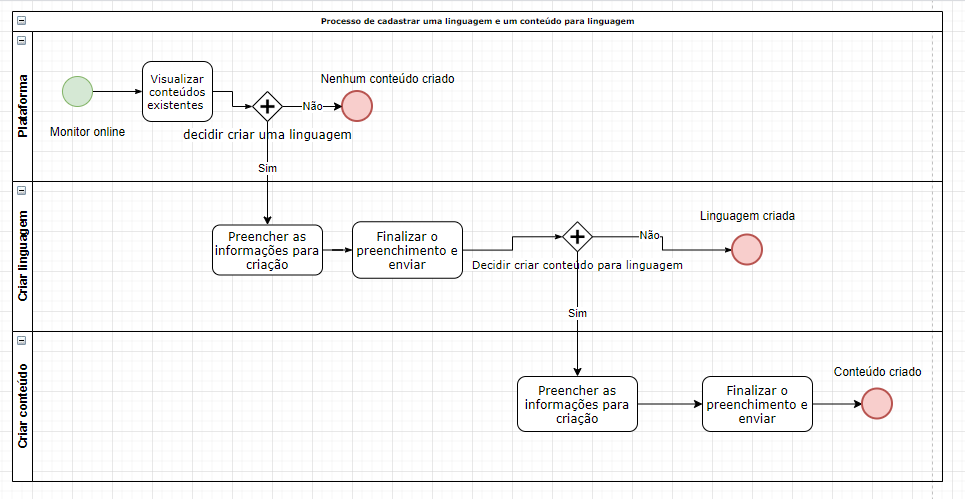
Na tabela posts tenho 8 atributos:

* ID: É uma chave primária que serve para identificar qual é o conteúdo.
* Title: Para inserir um título.
* Subtitle: Para inserir um subtítulo.
* Text: Para inserir explicações.
* Exercise: Para inserir a resolução do exercício.
* User\_id: É uma chave estrangeira para atribuir conteúdo para o usuário que o criou.
* LanguageKey: É uma chave estrangeira para atribuir cada post para cada linguagem escolhida.

Na tabela languages tenho 6 atributos:

* ID: É uma chave primária que serve para identificar qual é o conteúdo.
* Title: Para inserir um título.
* Description: Para inserir uma descrição.
* Image: Para inserir imagens para os cards da página inicial.
* Key: Para definir qual linguagem será.
* User\_id: É uma chave estrangeira para atribuir a linguagem para o usuário que o criou.

## Processo dinâmico

**Figura 11** – Diagrama de processo

No diagrama de processo acima é demonstrado as ações que um monitor precisará fazer para criar uma linguagem ou um conteúdo.

# Resultados e Discussão

O foco neste trabalho foi na utilização da linguagem Python e da tecnologia Flask. Além disso, esse trabalho foi desenvolvido com um editor de código-fonte Virtual Studio *Code*, SQLAlchemy (ORM) e o banco de dados MySQL.

Junto com o Flask, é possível utilizar o Jinja2, uma engine de templates para criação de HTML que é retornado ao usuário através de requisições HTTP. O SQLAlchemy (Object Relational Mapper) é responsável por traduzir as classes em tabelas no banco de dados (MySQL).

O Flask, é um micro-framework, e por isso é de certa forma simples, leve e rápido, recomendado para pequenos projetos e podendo atingir até um certo grau de complexidade, oferecendo maior flexibilidade e controle para o desenvolvedor. É escalável, mas por ser single threadedpode não apresentar uma boa performance em algumas circunstâncias.

Atualmente Python já é utilizado por grandes empresas, tais como, Netflix, Google, Spotify, Globo dentre outros, e é escolhido por ser uma linguagem de programação de alto nível, imperativa e de scripts. Sem esquecer que é orientado a objetos e de interpretação fácil, isso são fatores que fazem esta linguagem ser amplamente utilizada por muitas empresas.

Ao finalizar este projeto eu conclui que é possível o transformar em algo muito útil para a comunidade de maratonas de programação se utilizado de forma correta, em tempos de pandemia onde as relações escolares estão EAD, também pode ter um valor importante para agregar ao âmbito escolar, pois o mesmo fornece um ambiente virtual de interação direta, ou seja, é possível criar uma ponte entre o conhecimento e os alunos de programação por meio dos monitores.