FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS

FATEC PROFESSOR Jessen Vidal

LUAN MATHEUS SATIRO DE OLIVEIRA

NLEARN: Um sistema web para aprendizado de linguagens de programação.

Orientador: Fabricio Galende Marques de Carvalho

São José dos Campos

2020

SUMÁRIO

[1 Introdução 3](#_Toc49841023)

[1.1 Definição do problema 3](#_Toc49841024)

[1.2 Objetivo 4](#_Toc49841025)

[1.3 Escopo 4](#_Toc49841026)

1.4 Ferramental tecnológico 6

[2 Modelagem 7](#_Toc49841027)

[2.1 Arquitetura 8](#_Toc49841028)

[2.2 Modelagem de Dados 12](#_Toc49841029)

[2.3 Modelagem dinâmica 14](#_Toc49841030)

[3 Resultados obtidos 16](#_Toc49841031)

[4 Considerações finais 21](#_Toc49841031)

# Introdução

O problema abordado neste trabalho consiste em uma dificuldade dos novos ingressantes nos meios de maratonas de programação, assim como 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 foi desenvolvido no escopo da disciplina laboratório de engenharia de software alinhado ao trabalho de graduação do curso de análise e desenvolvimento de sistemas da FATEC São José dos Campos.

## Definição do problema

As maratonas de programação são eventos realizados por instituições de ensino da área de TI. O intuito é prover uma competição onde o ganhador será aquele que acertar mais questões fornecidas pela banca.

Os monitores são alunos que obtiveram uma nota alta ao passar em determinada matéria e optam por auxiliar os novos alunos, fornecendo um tempo dedicado para monitoria na faculdade na qual estudam.

Um problema típico consiste na dificuldade de se localizar conteúdo específico para resoluções de determinados exercícios de programação. Outro problema consiste na falta de espaço exclusivo para monitores de matérias relacionadas ao curso de TI conseguirem adicionar conteúdo para os alunos.

As plataformas como URI e Ucoder fornecem 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.

O material para aprendizado de uma linguagem de programação pode ser de difícil compreensão. Inicialmente, isso pode gerar uma dificuldade e criar um desencorajamento para o aluno prosseguir neste caminho.

Um exemplo prático e simples: existe um algoritmo para resolver determinado problema, mas 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.

Esse trabalho foi desenvolvido no escopo da disciplina laboratório de engenharia de software e seu foco é na solução dos problemas anteriormente citados.

## Objetivo

O objetivo desse trabalho é desenvolver um sistema web para 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.

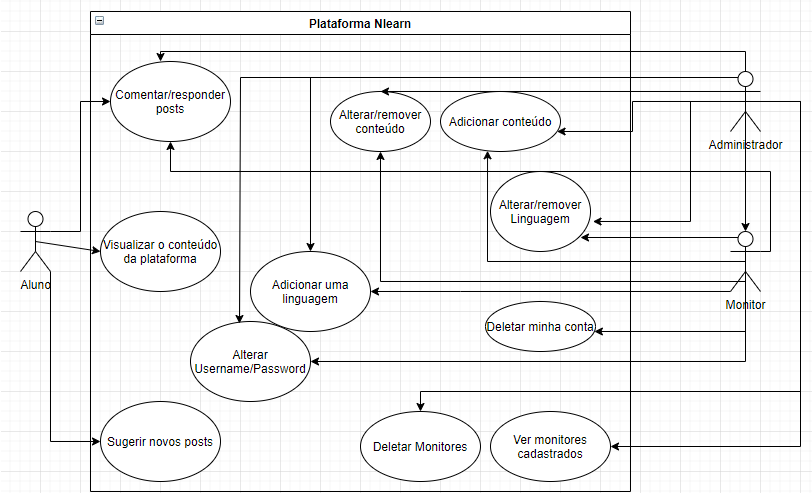
## Escopo

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. Os campos da seção conteúdo serão o título, subtítulo de um conteúdo, conteúdo, exercícios, key do tópico e key da linguagem.

Somente os usuários cadastrados no sistema poderão realizar essas alterações. Cada monitor só pode alterar o conteúdo que ele mesmo tenha postado. Também haverá um usuário administrador que além de poder também inserir, alterar ou deletar linguagens ou conteúdos, poderá deletar monitores do sistema. Ao deletar uma linguagem isso também deletará todo o conteúdo atribuído a ela.

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.

**Figura 1** – Caso de uso

Na figura 1 existem 3 atores. O aluno poderá visualizar o conteúdo na plataforma, comentar/responder posts e sugerir novos posts. O monitor poderá adicionar/alterar/remover linguagens ou conteúdo para linguagens. Um monitor pode apenas alterar ou excluir os conteúdos que cadastrou. O administrador tem os mesmos direitos dos monitores, com o acréscimo de poder visualizar todos os monitores cadastrados ou realizar a deleção deles. O usuário administrador pertence a mesma tabela do monitor, com a diferença que o administrador tem uma flag no código e isso permite acessar uma rota para administrar monitores

## Ferramental tecnológico

Nesse trabalho foram utilizas as seguintes tecnologias:

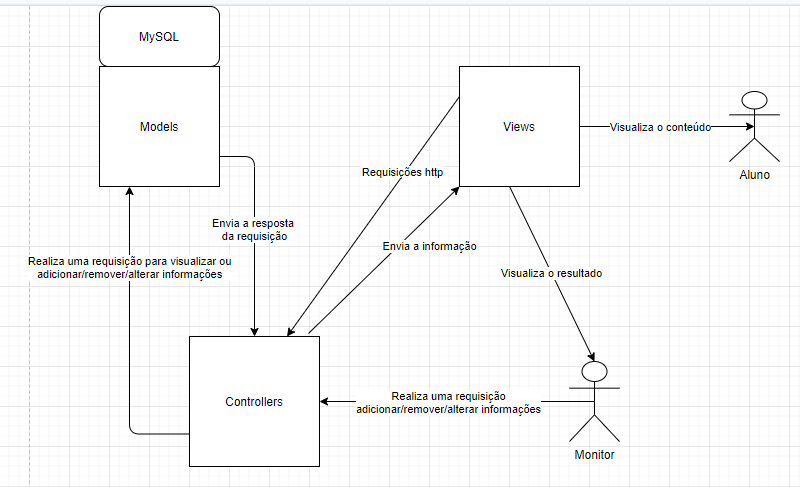
1. Python no Back-end;
2. Jinja2 para geração de páginas dinâmicas;
3. Gunicorn como servidor de implantação, em conjunto com o Flask;
4. SqlAlchemy para persistência de dados;
5. Flask micro framework para desenvolvimento web;
6. Bootstrap framework para o front-end;
7. Html5 para estruturar a página;
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;
11. MySql para o banco dados;
12. Git para versionamento do código;
13. Visual studio code para edição de código fonte.

# modelagem

Este Capítulo apresenta diferentes modelos e código-fonte representativos do sistema, desenvolvido 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

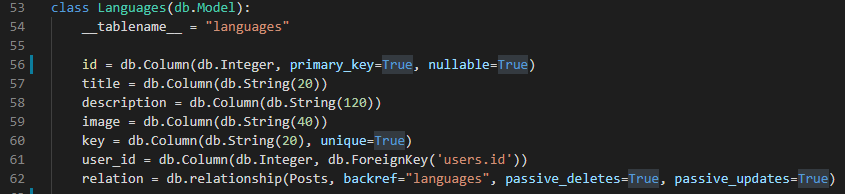
A figura 2 ilustra a arquitetura do sistema.

**Figura 2** – 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) como apresentado na figura 2.

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

Um exemplo de modelo é ilustrado na figura 3.

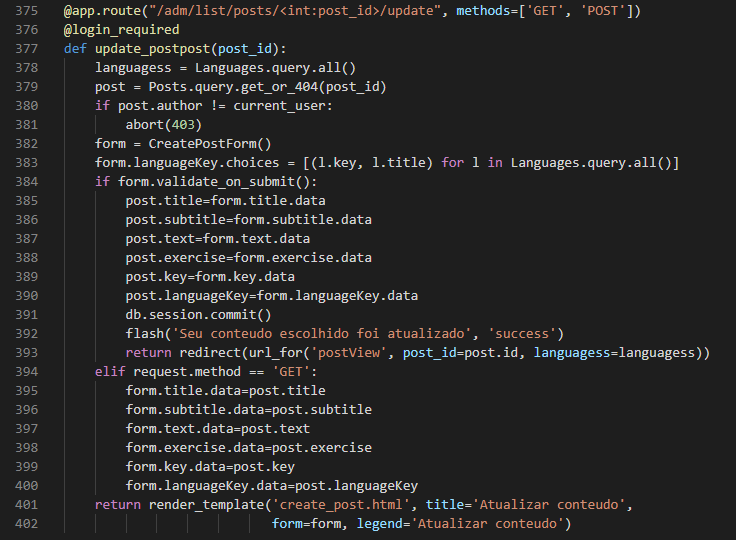
**Figura 3** – Classe Languages do model

Na figura 3 é realizado a descrição da estrutura de uma das tabelas do banco. 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.

No exemplo da figura 3 temos:

* Id (Linha 56): Para identificar a linguagem. Consiste em uma chave primária;
* Title (Linha 57): Para adicionar um título a linguagem. Consiste em um campo tipo texto;
* Description (Linha 58): Para adicionar uma descrição a linguagem. Consiste em um campo tipo texto;
* Image (Linha 59): Para adicionar uma imagem a linguagem. Consiste em um campo tipo texto;
* Key (Linha 60): Para definir o nome da linguagem. Consiste em um campo tipo texto único;
* User\_id (Linha 61): Uma chave estrangeira fazendo referência ao id da tabela usuário para verificar quem criou a linguagem;
* Relation (Linha 62): 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.

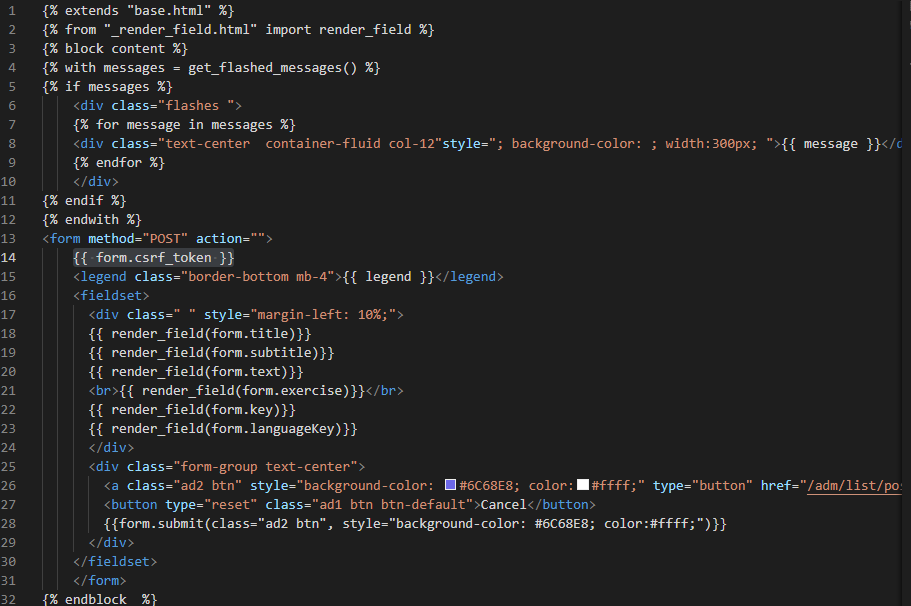
**Figura 4** – Rota para atualizar um conteúdo

Na figura 4 existem dois tipos de requisições: get e post. Essas requisições são utilizadas para receber e enviar informações ao banco.

* Variável languagess (linha 378): 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 (linha 379): Essa variável é responsável por pegar o id do post que o usuário está no momento;
* Form (linha 382): Essa variável é responsável por criar o formulário da classe CreatePostForm;
* Linha 383: Essa linha é responsável por pegar as linguagens e as exibir em um campo do tipo select;
* Primeiro if (linha 380): 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 (linha 384): Esse if realiza uma validação. Se o botão de finalizar for acionado, os dados abaixo serão inseridos numa variável que realiza o commit para o banco. Dessa forma, salvará as informações obtidas nos forms exibindo um flash. O flash exibira uma mensagem informando que a linguagem foi atualizada. O return desse if retornará para outra função que exibira todos os posts existentes;
* Return (linha 393): Caso entre no if retornará à função postView para visualizar todos as linguagens existentes;
* Elif (linha 394): Esse elif pega as informações existentes no banco para que o monitor visualize o conteúdo já existente e decida o que de fato quer atualizar em cada tópico;
* Return (linha 401): o return da função update\_postpost serve para que caso nenhuma condicional seja executada retorne a página de criação de conteúdo.

**View**: É responsável por tudo que o usuário final visualiza, toda a interface, informação.

Um exemplo é ilustrado na figura 5.

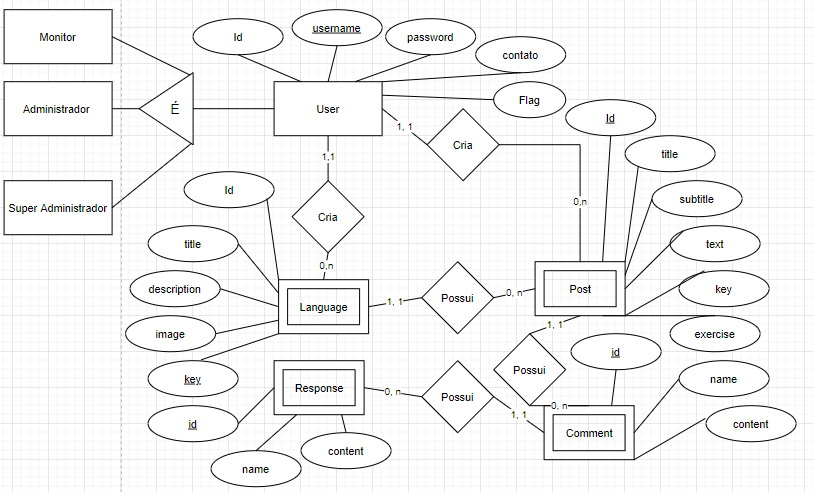
**Figura 5** – Template para criação de conteúdo

Na figura 5 é apresentado a utilização de jinja, html, css e boostrap.

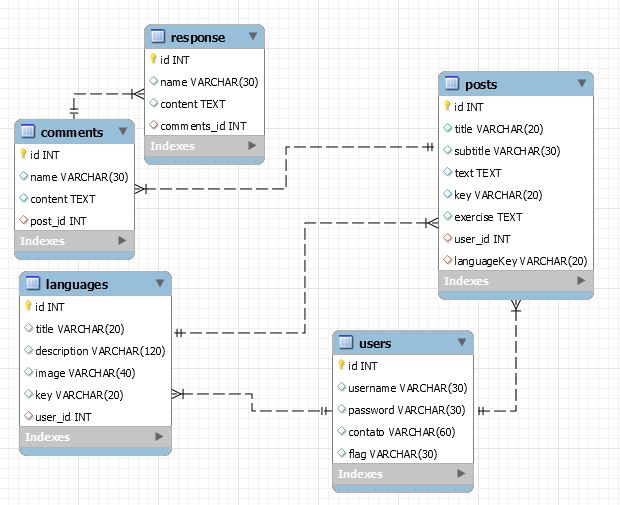
* Parâmetro block (linhas 1 e 3): Extensão do template base (template base é a estrutura geral do site);
* Mensagem flash (linha 4): Serve para exibir mensagens programadas nos controllers, como por exemplo: linguagem cadastrada;
* Tag <div> ou containers (linha 8): Possui as configurações de estilo;
* Tag <form> (linha 13): Serve para estabelecer as ações que possibilitam inserir as variáveis de formulário;
* Tag <fieldset> (linha 16): É encarregada de receber as variáveis do formulário para que os monitores possam digitar e enviar essas informações aos controllers;
* Botões (linhas 26, 27 e 28): Utilizados para realizar ações como: retornar para página anterior, cancelar escrita, enviar formulário.

## Modelagem de Dados

Na figura 6 será ilustrado um diagrama ER.

Figura 6 – Diagrama entidade relacionamento (modelo conceitual)

Na figura 6 existem 5 entidades. Users possui 5 atributos e duas relações, o username é sua determinante, assim como possui 3 elementos estendidos: monitor, administrador e super administrador. A entidade users pode criar nenhuma ou muitas linguagens, assim como consegue criar nenhum ou muitos conteúdos. A seguir, a entidade languages possui 5 atributos e uma relação com o conteúdo, o atributo key é seu determinante. A language pode conter nenhum ou muitos conteúdos e só existe se um monitor ou administrador a cadastrar. Após isso, posts com 6 atributos, o id é seu determinante e possui relação com comments. Um post só pode existir caso exista uma language e um monitor/administrador. Um post pode existir em um ou nenhum monitor/administrador, assim como existe em uma ou nenhuma language. A seguir, comments com 3 atributos e id é seu determinante. Um comments pode existir em um ou nenhum post. E por fim, renponse com 3 atributos e id é seu determinante. Um comment/reponse pode existir caso exista um post. Um response pode existir em um ou nenhum comments.

**Figura 7** – Modelo lógico

Na figura 7, existem cinco tabelas. A tabela de posts(conteúdo), languages(linguagens), users(monitores/administrador), comments(comentários) e response(respostas).

Na tabela users existem cinco atributos:

* ID: Uma chave primária que serve para identificar o usuário;
* Username: Para o nome;
* Password: Para fazer as validações de login na plataforma;
* Contato: Para inserir o contato do monitor;
* Flag: Para definir se o usuário poderá administrar ou não os monitores.

Na tabela posts existem 8 atributos:

* ID: Uma chave primária que serve para identificar 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: Uma chave estrangeira para atribuir conteúdo para o usuário que o criou;
* LanguageKey: Uma chave estrangeira para atribuir cada post para cada linguagem escolhida.

Na tabela languages existem 6 atributos:

* ID: Uma chave primária que serve para identificar a linguagem;
* 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: Uma chave estrangeira para atribuir a linguagem para o usuário que a criou.

Na tabela comments existem 3 atributos:

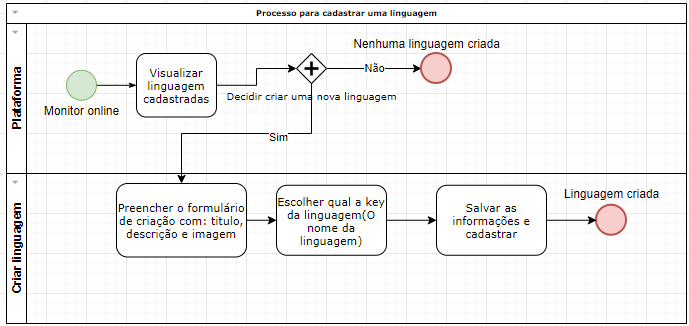
* ID: Uma chave primária que serve para identificar o comentário;
* Name: Para inserir o nome de quem realiza o comentário;
* Content: Para inserir o conteúdo do comentário.

Na tabela response existem 3 atributos:

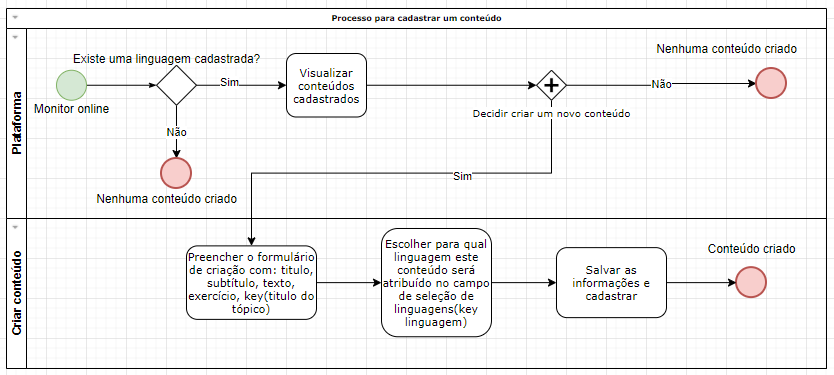
* ID: Uma chave primária que serve para identificar a resposta;
* Name: Para inserir o nome de quem realiza a resposta;
* Content: Para inserir o conteúdo da resposta.

## Modelagem dinâmica

A Figura 8 ilustra o processo para cadastrar uma linguagem.

**Figura 8** – Diagrama de processo

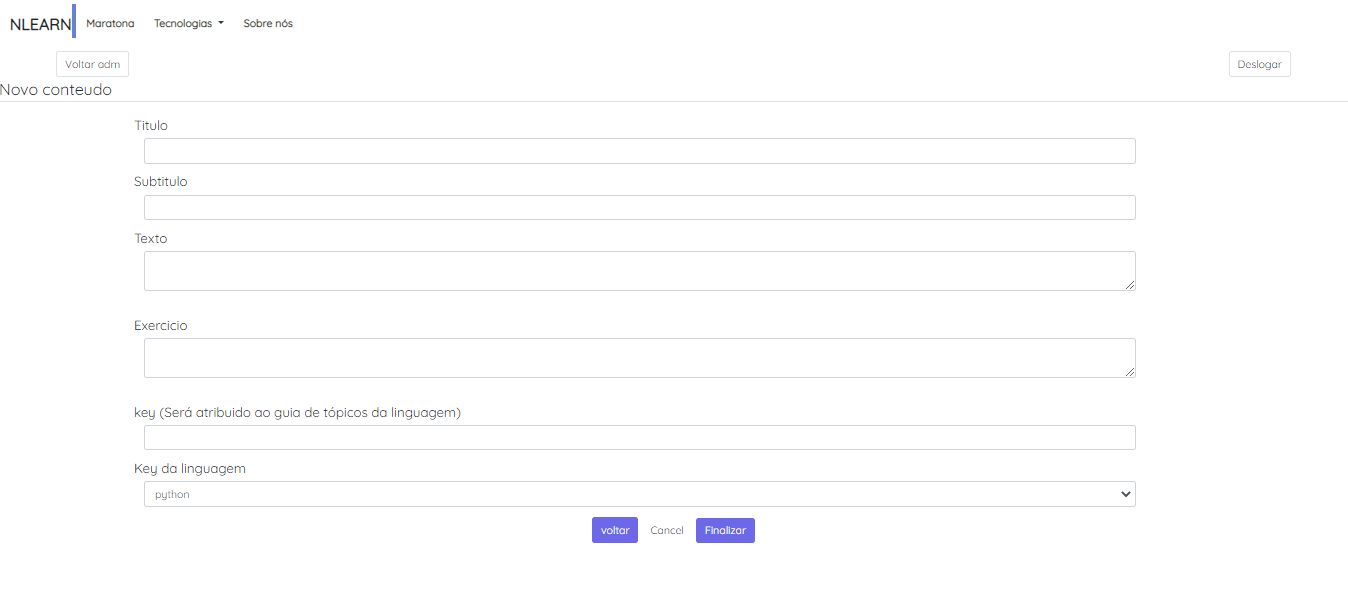
Na figura 8 é apresentado o que um monitor precisa fazer para criar uma linguagem. Inicialmente ele visualiza se existe alguma linguagem criada e se quer criar uma nova. Após isso, caso queira criar uma nova linguagem, irá preencher as informações necessárias no formulário. Após o término do preenchimento basta somente clicar no botão finalizar e visualizar a nova linguagem criada na seção todas linguagens listadas.

**Figura 9** – Diagrama de processo

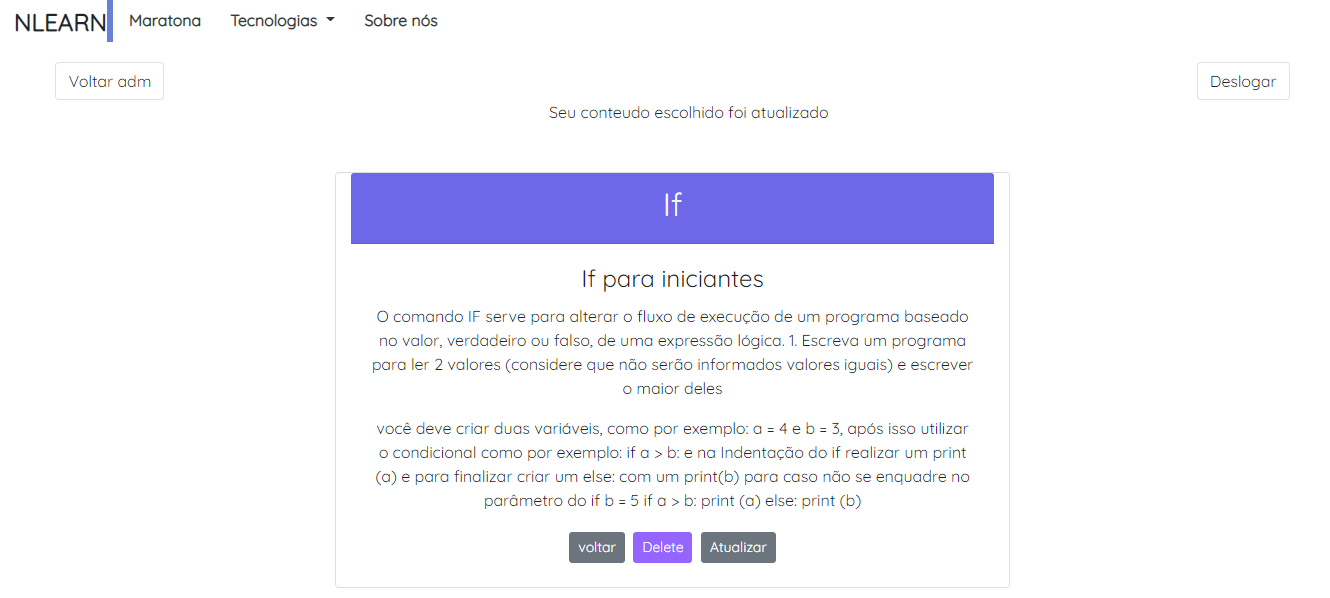
Na figura 9 é apresentado o que um monitor precisa fazer para criar um conteúdo. Inicialmente ele visualiza se existe alguma linguagem criada e se não houver, não será possível continuar com o processo. Após isso, caso queira criar um novo conteúdo, irá preencher as informações necessárias no formulário. Após o término do preenchimento basta somente clicar no botão finalizar e visualizar o novo conteúdo criado na seção todos conteúdos listados.

# Resultados OBTIDOS

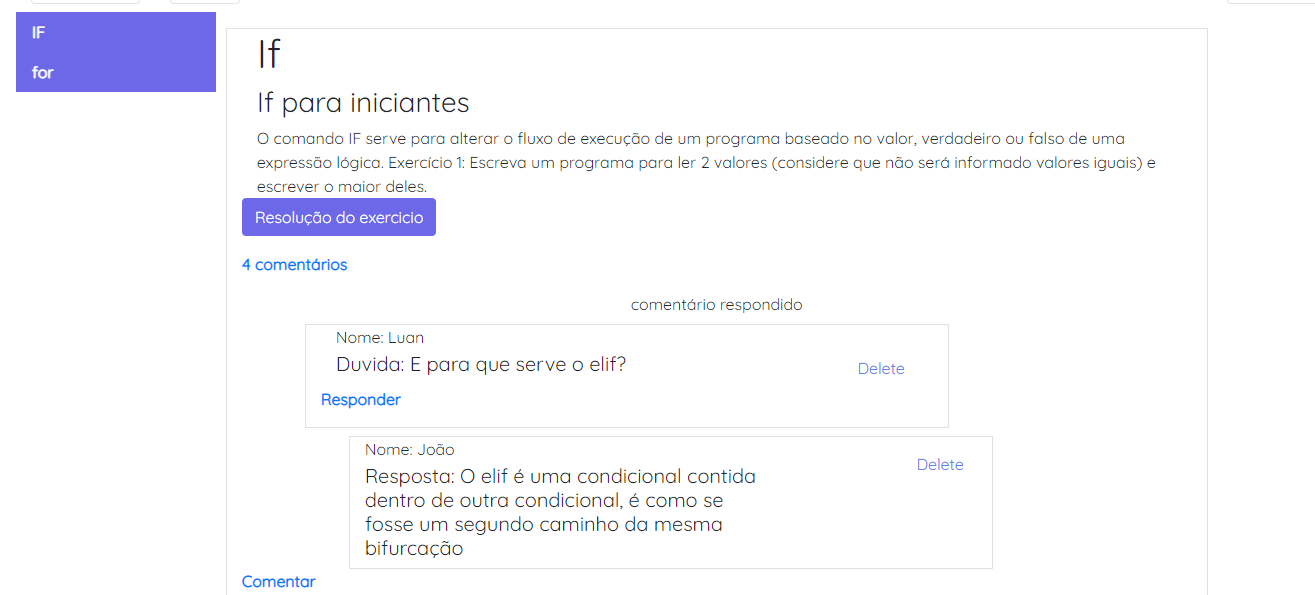
Nas figuras 10, 11, 12, 13, 14, 15, 16 e 17 são apresentados alguns dos resultados obtidos.

**Figura 10** – View para criação de conteúdo

Como ilustrado na figura 10, os campos para criação de um novo conteúdo são: titulo, subtítulo, texto, exercício, key e a key da linguagem para qual o conteúdo será atribuído.

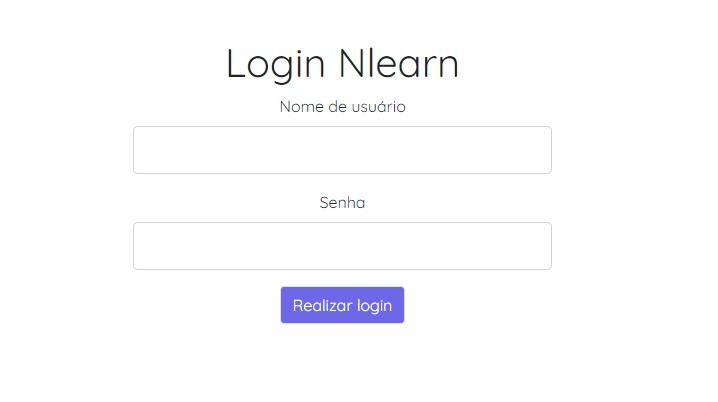
 **Figura 11** – View de exibição de conteúdo para alterar/deletar

Na figura 11 é apresentado um conteúdo especifico atribuído a uma linguagem para exemplificação. Nesta seção é possível atualizar ou deletar este conteúdo.

**Figura 12** – View de exibição de conteúdos cadastrados nas linguagens

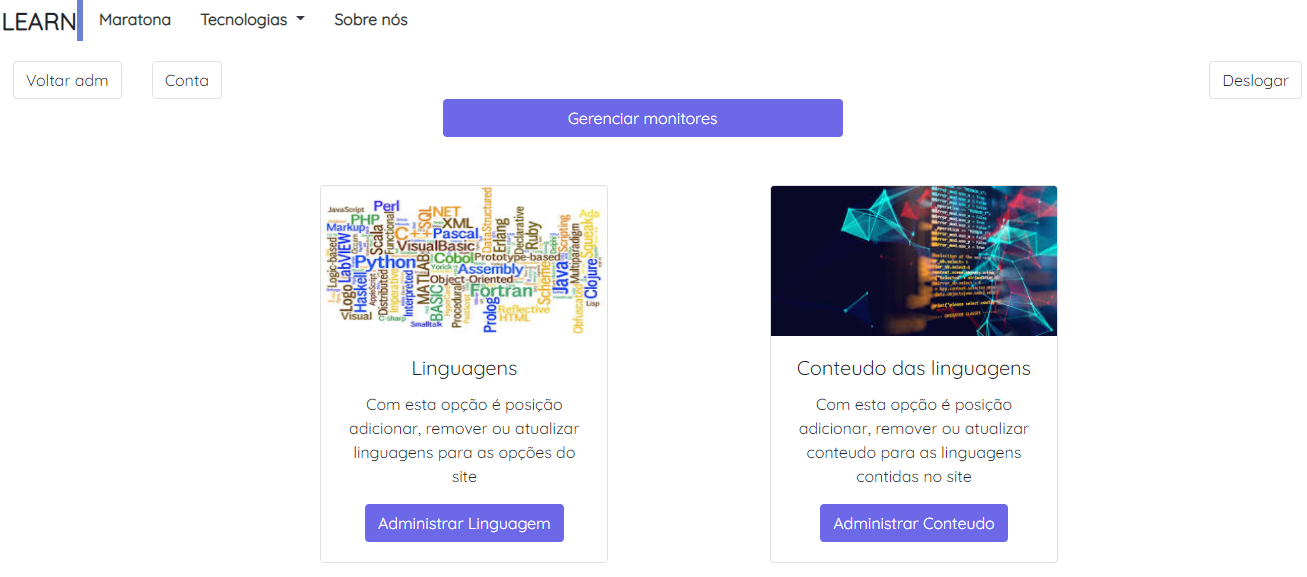
Na figura 12 é ilustrado a tela que o aluno poderá visualizar após o monitor ter realizado o cadastro de uma linguagem e um conteúdo. O aluno poderá comentar ou responder comentários sobre cada conteúdo. O intuito dessa tela é auxiliar o aluno na compreensão de cada tópico cadastrado pelo monitor.

**Figura 13** – Tela de login



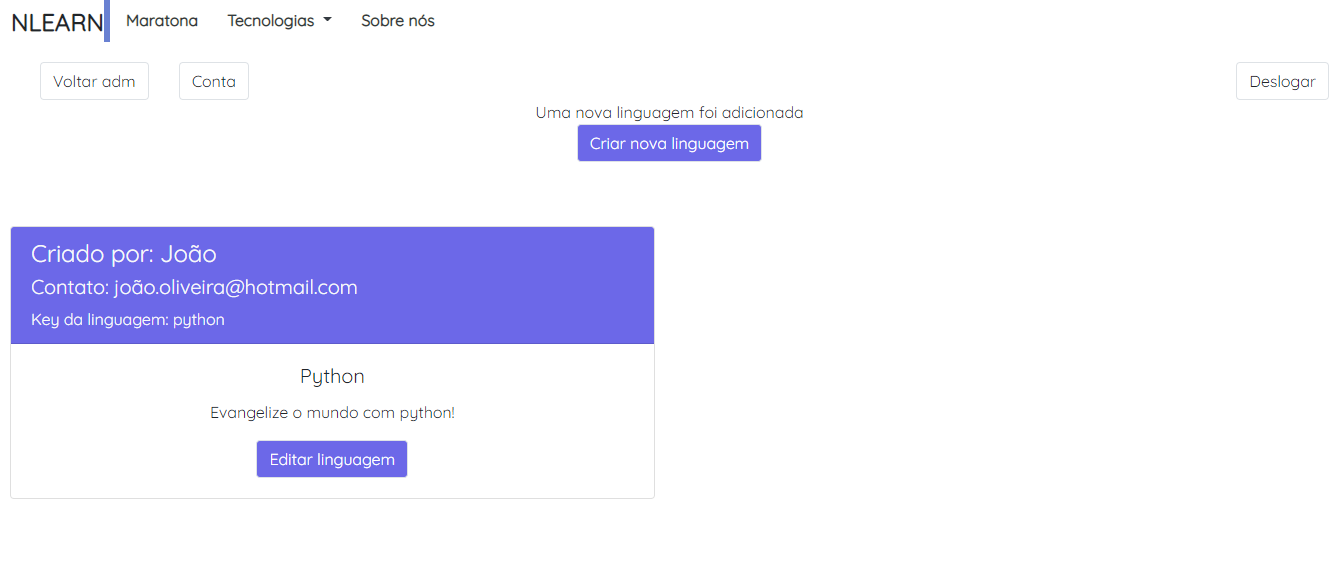
.

Na figura 13 é apresentado a tela de login. Se o usuário não tiver acesso ao sistema será retornado uma mensagem de login inválido. Para ter acesso ao sistema é necessário contatar algum administrador. Somente o administrador pode cadastrar novos usuários.

**Figura 14** – Tela para selecionar a seção linguagem ou conteúdo

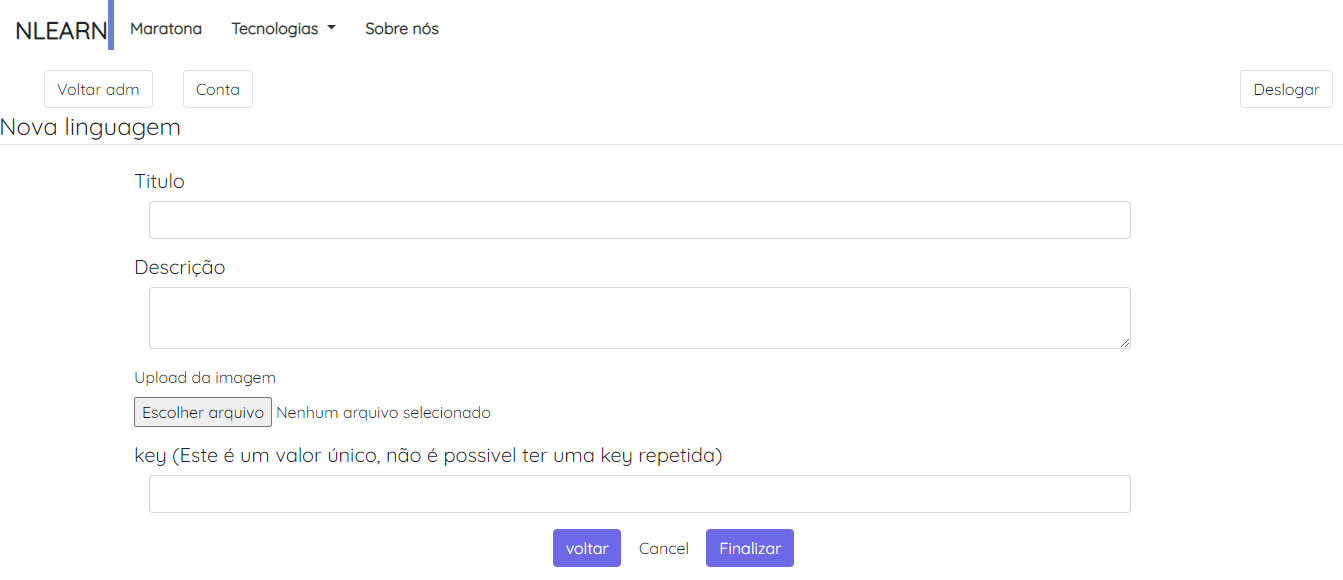
Na figura 14 é apresentado a primeira seção após realizar o login. Nesta seção é possível selecionar as opções: administrar linguagem, administrar conteúdo, deslogar da conta e alterar informações da conta.

**Figura 15** – Visualizar linguagens existentes



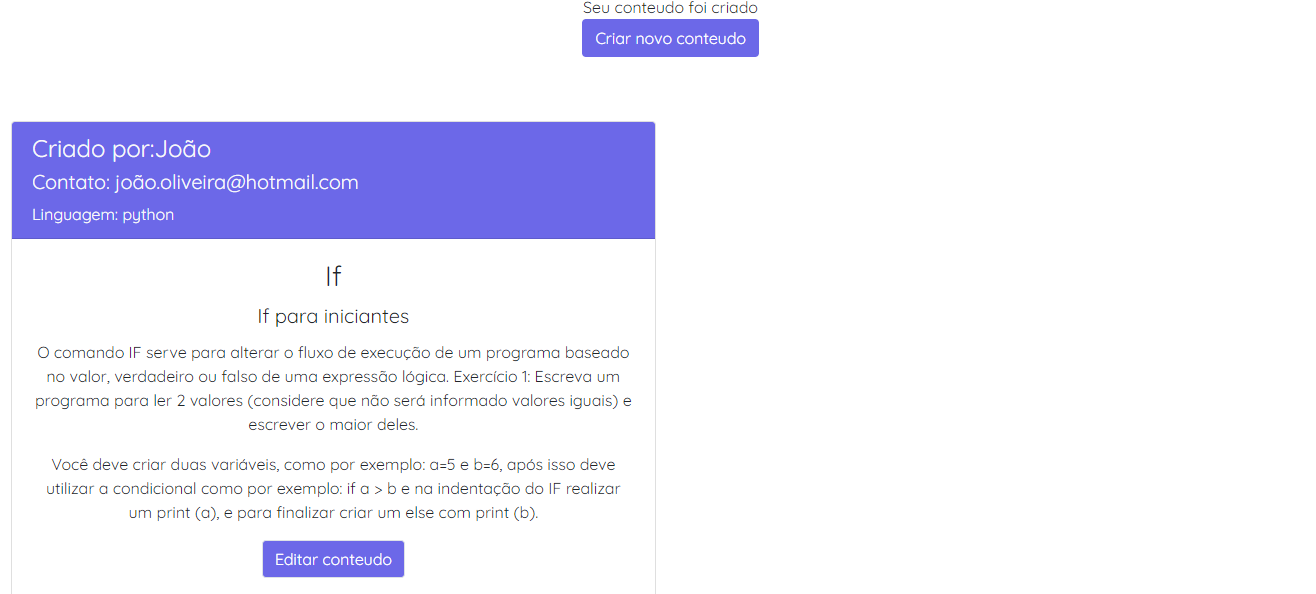
Na figura 15 é apresentado a seção de linguagem cadastradas. Um card é utilizado para estruturar as informações. No exemplo, a linguagem foi criada por João. A linguagem criada foi python.

**Figura 16** – Criar nova linguagem



Na figura 16 é apresentado a seção para criação de uma nova linguagem. O usuário pode informar as seguintes informações: título, descrição, imagem e a key para identificar qual linguagem será.

**Figura 17** – Visualizar conteúdos existentes



Na figura 17 é apresentado a seção de conteúdos cadastrados. Um card é utilizado para estruturar as informações. No exemplo, o conteúdo foi criado por João. O conteúdo criado foi sobre condicionais.

# CONSIDERAÇÕES FINAIS

O foco neste trabalho foi na utilização da linguagem Python e da tecnologia Flask, assim como na utilização de Mysql para banco de dados e SqlAlchemy para a persistência de dados.

Na versão atual do sistema é possível realizar o cadastramento de três tipos de usuários: usuário administrador, que não pode ser deletado do sistema, usuário administrador, que pode ser deletado, e o monitor. Deve sempre existir um usuário no sistema, pois só é possível cadastrar novos usuários a partir de um usuário administrador existente. Esses três usuários podem cadastrar linguagens e conteúdo para o sistema ou comentar/responder posts. Os usuários administradores podem remover linguagens e conteúdos criados por monitores, assim como realizar a deleção das contas ou qualquer comentário/resposta. Apenas os criadores de suas respectivas linguagens e conteúdos podem realizar a atualização dessas informações.

As limitações da atual versão do sistema são: não é possível adicionar conteúdo interativo apenas estático.

Como sugestões para trabalhos futuros tem-se:

* Adicionar seção para alunos criarem contas;
* Adicionar um fórum para alunos conversarem sobre os conteúdos abordados;
* Adicionar opção para inserção de vídeos.