**Projeto Peppa Educações**

Integrantes: Taís Bruno, Rafael Carrilho, Natália Bruno, Antônio Vinícius e Pedro Xavier

1. **Introdução**
   1. **Objetivo**

O Peppa Educações é uma plataforma de cursos com o objetivo de facilitar o acesso ao conhecimento através da internet. Através dela, pessoas podem criar e compartilhar cursos de diversas áreas do conhecimento, que poderão ser acessados por outros usuários que tiverem interesse em adquirir novas habilidades.

* 1. **Escopo**

O Peppa Educações é uma plataforma em que os usuários podem publicar e acessar cursos de diversas áreas de conhecimento. Um usuário pode se cadastrar no sistema com seu email, nome e senha protegida. O sistema apresenta módulos para cadastro, alteração, consulta e remoção de cursos e aulas para os cursos.

O cadastro de um curso exige somente o nome e as categorias relacionadas ao curso. Após a criação do curso, o usuário deve estipular aulas a serem publicadas junto ao curso. Um curso está relacionado a um instrutor e este tem permissão para editar o curso e suas aulas, além de poder anexar material à aula. O instrutor pode publicar múltiplos cursos e pode visualizar a lista dos cursos publicados.

No sistema, um usuário pode buscar cursos por nome, filtrá-los por avaliações e categorias. Ao selecionar um curso na lista, o mesmo visualiza um resumo do curso com a lista de aulas e pode se inscrever no curso. Uma vez inscrito, o usuário pode assistir às aulas do curso, comentar no curso, comentar nas aulas do curso e avaliar o curso.

A plataforma salva o progresso das aulas para que, caso interrompidas, os usuários possam retomar de onde pararam. O usuário pode se inscrever em diversos cursos e visualizar a lista de cursos nas quais está inscrito. O sistema mantém um histórico de atividades realizadas pelos usuários, tal qual criar curso, criar aula, permitindo um acompanhamento eficiente de suas contribuições à plataforma.

1. **Requisitos Arquiteturais**
   1. **Restrições**

As seguintes restrições de requisito e de sistema possuem uma relação significativa com a arquitetura:

* Acesso via internet: A aplicação poderá ser acessada por meio da internet através de requisições web.
* Linguagem e framework: A linguagem utilizada no desenvolvimento do sistema será o Ruby, adotando o framework Ruby on Rails.
* Somente usuários cadastrados e autenticados no sistema têm acesso a suas funcionalidades.
* Sendo um sistema Web, ele deverá funcionar com os navegadores Internet Explorer 8 ou superior, Firefox 20 ou superior e Chrome 24 ou superior.
  1. **Requisitos e Soluções**

| **Requisito** | **Solução Arquitetural** |
| --- | --- |
| Autenticação e Autorização | Utilização de autenticação baseada em Basic Token com JWT, e autenticação baseada em papéis, garantindo maior segurança e controle de acesso aos recursos da plataforma. |
| Paginação de Dados | Ao carregar listas de conteúdo com mais de 20 itens, a resposta será paginada através do parâmetro ‘page’, proporcionando melhor desempenho e experiência do usuário. |
| Armazenamento de Arquivos | O armazenamento de arquivos na aplicação poderá ser realizado em sistema de arquivos ou banco de dados. A estratégia de armazenamento deverá ser determinada de acordo com o volume de dados, as características dos arquivos armazenados e o ciclo de vida dos arquivos no sistema. |
| Segurança do Sistema | As senhas dos usuários serão criptografadas ao completar o cadastro. Para acessar os registros da aplicação, os usuários deverão informar um token de autorização que só será gerado após um login bem-sucedido com o email e senha correspondentes. |
| Exclusão lógica de dados | Ao excluir um recurso da aplicação, o mesmo não será apagado do banco de dados. Ao invés disso, será atribuída uma flag de ‘deleted\_at’, indicando que o recurso foi apagado e está inativo. |
| Utilização do padrão MVC | A arquitetura do sistema será dividida em camadas, cada uma com funções específicas, seguindo o padrão Model-View-Controller(MVC) para maior organização e manutenibilidade do código. |
| Portabilidade de SGBDs | Devido ao uso do framework Ruby on Rails, é possível conectar qualquer SGBD à plataforma através de sua interface de queries dinâmicas, garantindo flexibilidade e adaptabilidade. |
| Interfaces de usuário  amigáveis e dinâmicas | A interface de usuário será desenvolvida considerando a usabilidade da aplicação. |

1. **Padrões arquiteturais adotados**

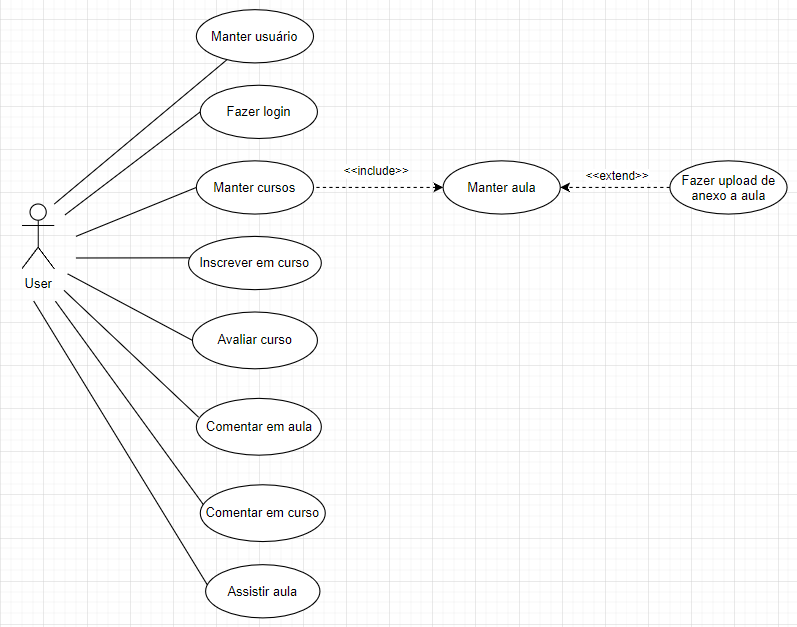
A arquitetura do sistema será fundamentada nos padrões MVC (Model-View-Controller) e Cliente-Servidor. A adoção do Cliente-Servidor envolverá a implementação de uma API REST, que irá dispor objetos JSON e status HTTP como resposta às requisições web realizadas pelos usuários.

Optamos pelo padrão MVC devido a sua ampla utilização em sistemas de informação e no mercado, bem como pela familiaridade e experiência prévia dos integrantes do grupo com este tipo de arquitetura. Além disso, ao dividir as responsabilidades entre Models, Controllers e Views, a arquitetura proposta promove escalabilidade, manutenibilidade, testabilidade e modularidade do sistema, facilitando seu desenvolvimento e evolução ao longo do tempo.

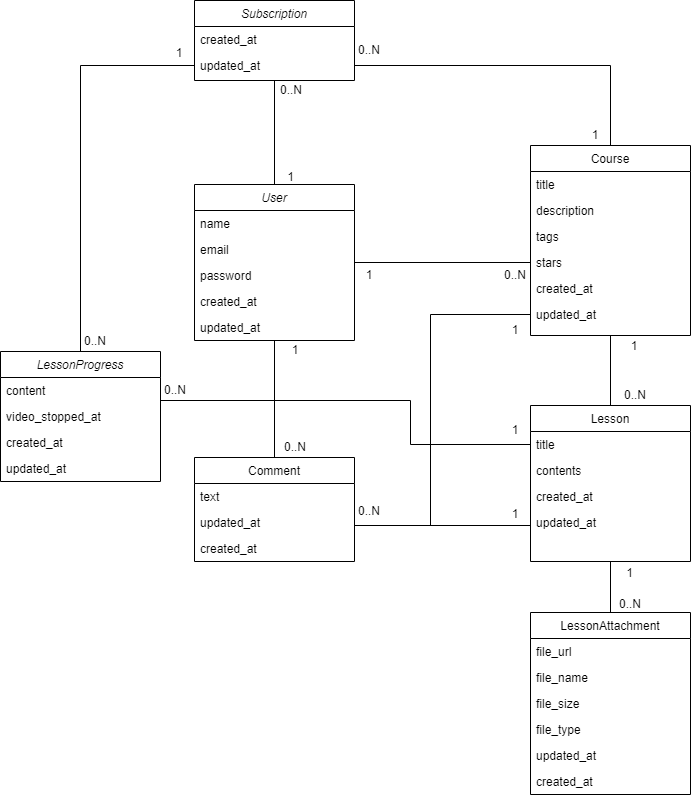
1. **Visão arquitetural**

Os diagramas de análise do projeto proporcionam uma visão simplificada das classes a serem implementadas, abrangendo o diagrama de casos de uso, modelo conceitual e o diagrama de sequência do sistema. Esses diagramas serão anexados como imagens ao repositório do projeto, para fácil acesso de todos os desenvolvedores envolvidos.

**4.1 Diagrama de casos de uso**

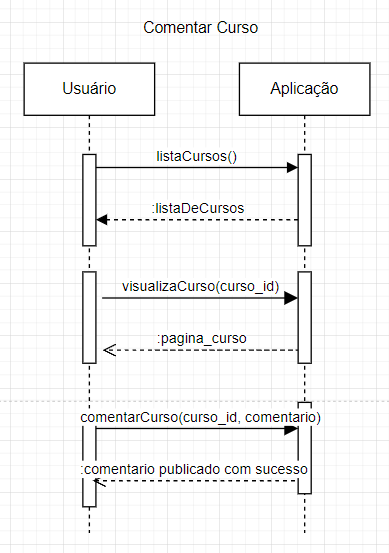


**4.2 Modelo conceitual**

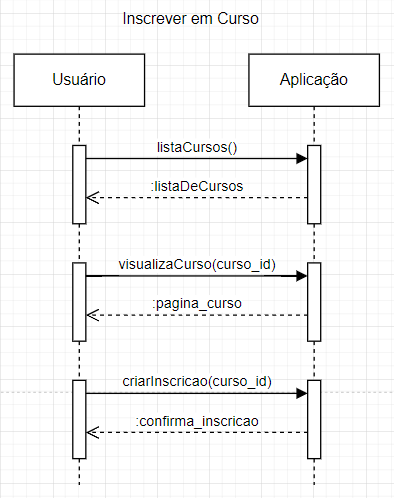
****

**4.3 Diagrama de sequência do sistema**

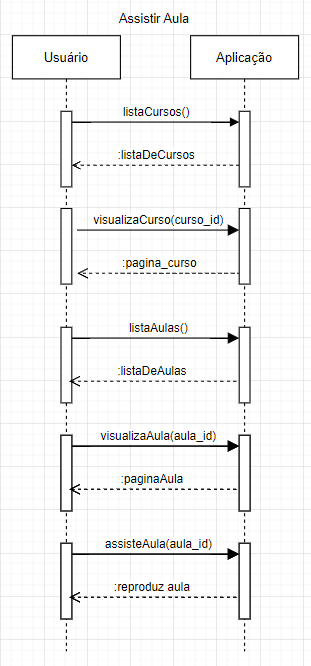
Comentar em curso:

****

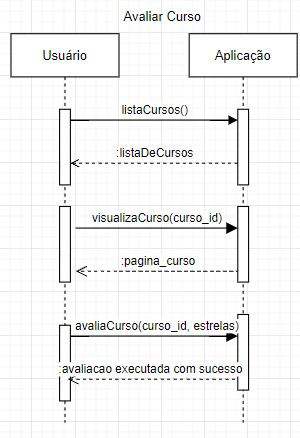
Inscrever em curso:

****

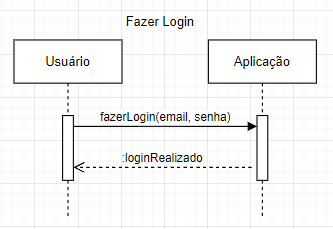
Assistir aula:

****

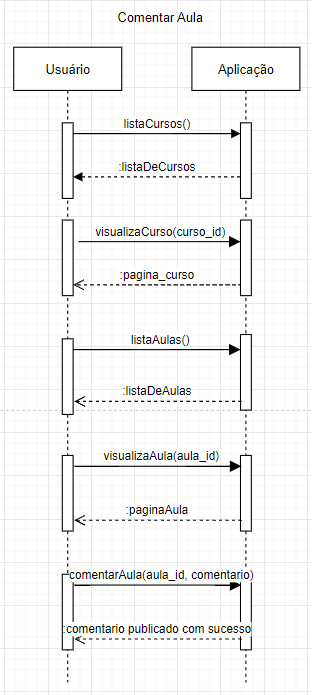
Avaliar curso:

****

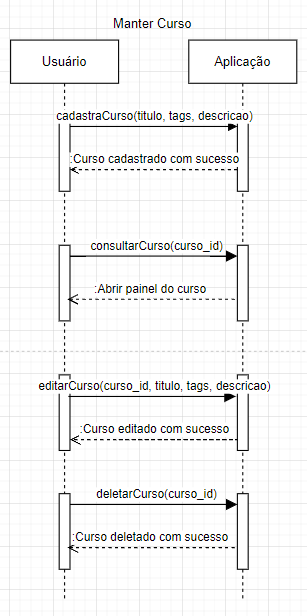
Fazer login:

****

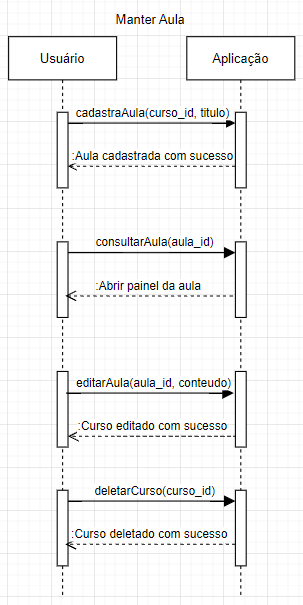
Comentar em aula:

****

Manter curso:

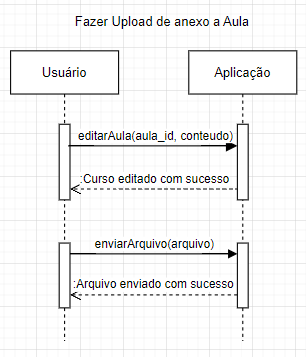
****

Manter aula:

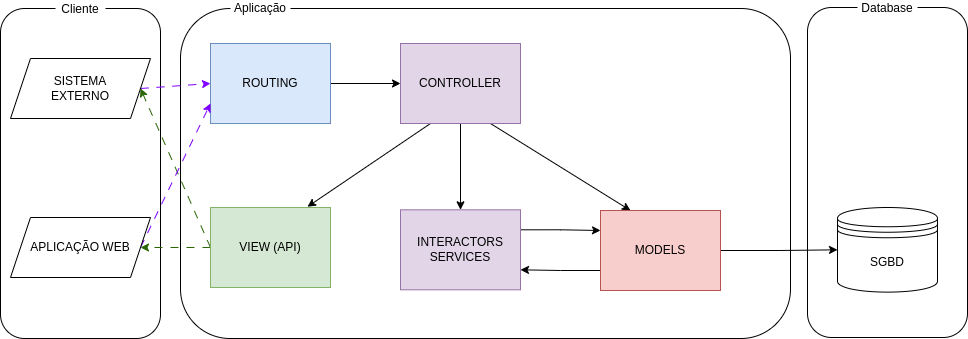
****

Manter usuário: Análogo a Manter Curso

Fazer upload de anexo a aula:

****

**4.4 Diagrama de arquitetura**



A aplicação Web representa a interface do front end desenvolvido em … que será conectado a API através de requisições HTTP. A linha tracejada verde representa a resposta da API, enquanto a linha tracejada roxa representa a requisição HTTP em si. A representação do SGBD explicita o banco de dados da nossa aplicação, que pode ser o MySQL, Postgres, etc. Models representa a classe que lidará com ações referentes ao banco de dados e que fará a interface com o SGBD. A classe Routing irá mapear os endpoints da aplicação para uma ação em um controller. O Controller é uma classe que recebe um chamado através do Routing, e funcionará como um intermediador entre o usuário e as lógicas de serviço. Os interactor services são as classes que vão extrair as lógicas de serviço dos controllers, tornando-os mais modularizados. Ao finalizar a execução das lógicas de serviço, os Controllers irão retornar uma resposta em formato JSON (View da API) para o serviço externo que fez a requisição.

Foi adotado o estilo arquitetural em camadas e Cliente-Servidor com o propósito de dividir as responsabilidades, facilitar a manutenção e evolução do sistema e aumentar a flexibilidade.

1. **Qualidade**

Com a elaboração desse documento de arquitetura, podemos garantir que o sistema Peppa Educações atenderá aos atributos de qualidades desejados. Ao adotar o padrão MVC, garantimos o atributo de Manutenibilidade por deixarmos as classes com responsabilidades divididas e reduzidas, facilitando a compreensão e a modificação do código.

A arquitetura de conexão com o banco de dados, garantida pelo uso do Ruby on Rails, nos dá a flexibilidade de conectar com qualquer SGBD que seja de interesse do desenvolvedor. Isso nos proporciona Portabilidade, permitindo a migração para outros sistemas de banco de dados, se necessário, sem grandes impactos para o sistema.

A implementação de criptografia de senha e role based authentication são medidas essenciais para garantir os atributos de Segurança e Adequação Funcional do sistema. Essas estratégias evitam o acesso não autorizado a informações sensíveis, garantindo que apenas usuários com as devidas permissões possam acessar determinadas partes do sistema.