INSTITUTO DE INFORMÁTICA

ARQUITETURA DE SOFTWARE

LEONARDO RIBEIRO OLIVEIRA PALMEIRA

THÂMARA CORDEIRO DE CASTRO

LUIZ FELIPE PIRES

PROJETO FINAL:
Documentação Arquitetura de Software

LEONARDO RIBEIRO OLIVEIRA PALMEIRA

THÂMARA CORDEIRO DE CASTRO

LUIZ FELIPE PIRES

PROJETO FINAL:
Documentação Arquitetura de Software

Desenvolver o documento referente a arquitetura do software de gestão de academia - Haltere.

Professor: JACSON RODRIGUES BARBOSA

1. Introdução

1.1 Finalidade

O objetivo deste documento é descrever a arquitetura do software de gestão de academia, fornecendo uma visão geral da estrutura, componentes e interações do sistema. Ele serve como um guia para desenvolvedores, fornecendo diretrizes para o projeto, implementação e manutenção da aplicação.

1.2 Escopo

O software de gestão de academia é uma aplicação web projetada para auxiliar na administração e operação de academias. Ele oferece funcionalidades para criar e gerenciar treinamentos personalizados, cadastrar e buscar usuários, calcular o Índice de Massa Corporal (IMC), agendar treinamentos, registrar histórico de treinamento e gerar relatórios de desempenho. A arquitetura do software é baseada na linguagem de programação Elixir e no framework Phoenix para o Back-end, utilizando o banco de dados MySQL para armazenar os dados. Para o Front-End, será utilizado JavaScript, HTML e CSS, no framework ReactJS, com integração através da GraphQL. Por fim, o Ecto será utilizado como ORM para a comunicação com o banco de dados MySQL.

1.3 Definições, Acrônimos e Abreviações

- MVC: Model-View-Controller, padrão de arquitetura de software que separa as preocupações relacionadas ao modelo de dados, interface de usuário e lógica de controle.
- Elixir: Linguagem de programação funcional, concorrente e de propósito geral.
- Phoenix: Framework web para desenvolvimento em Elixir.
- Ecto: Biblioteca de mapeamento objeto-relacional (ORM) para banco de dados em Elixir.
- CSV: Comma-Separated Values, formato de arquivo de texto que representa dados tabulares, separando os valores por vírgulas.
- Front-End: Camada de um aplicativo ou sistema web que lida com a apresentação e interação com o usuário. Envolve o desenvolvimento de interfaces de usuário utilizando tecnologias como HTML, CSS e JavaScript.
- JavaScript: Linguagem de programação de alto nível, orientada a objetos e interpretada pelos navegadores web.
- HTML: HyperText Markup Language (Linguagem de Marcação de Hipertexto) é a linguagem padrão para a criação e estruturação de páginas web.
- CSS: Cascading Style Sheets (Folhas de Estilo em Cascata) é uma linguagem de estilo utilizada para definir a aparência e o layout de elementos HTML em uma página web.
- ReactJS: Biblioteca JavaScript de código aberto para construção de interfaces de usuário.
- Framework: Conjunto de ferramentas, bibliotecas e padrões que oferecem uma estrutura para facilitar o desenvolvimento de software.

- MySQL: Sistema de Gerenciamento de Banco de Dados Relacional de código aberto.
- GraphQL: Linguagem de consulta e manipulação de dados para APIs, que oferece uma maneira mais eficiente e flexível de interagir com os serviços de back-end.
- Ecto: Framework de banco de dados para a linguagem de programação Elixir, que fornece uma camada de abstração sobre o banco de dados, permitindo que os desenvolvedores interajam com o banco de dados de maneira mais eficiente e elegante.

1.4 Visão Geral

A arquitetura do software de gestão de academia é projetada com uma abordagem modular e escalável, o que possibilita a inclusão e atualização de funcionalidades de forma flexível e adaptável às necessidades em constante evolução. O sistema é baseado em um modelo cliente-servidor, onde os clientes (ou usuários) enviam solicitações ao servidor central. O servidor é responsável por processar essas solicitações, executar as operações necessárias e armazenar os dados relevantes em um banco de dados dedicado.

Model	Como camada de dados a model estará sempre fazendo as persistências e tratando os dados das requisições do usuário, que foram trabalhadas na regras de negócio na camada de controller.
Controller	Essa camada vem com o objetivo de tratar todas as regras de negócio, que vem através das solicitações do usuário que são tratadas nas rotas.
View	É uma das camadas do MVC que tem a função de representar as interfaces ao usuário final, toda ou qualquer interface são tratadas nessa camada.
Servidor	O servidor é responsável em processar as solicitações do cliente, ele é o responsável geral pela estrutura do sistema.
Cliente	A exibição das views por meio de um navegador é considerado o cliente. É através dele que as solicitações são requeridas ao servidor.

Essa arquitetura cliente-servidor proporciona uma separação clara de responsabilidades, onde os clientes (aplicações ou interfaces utilizadas pelos usuários) se comunicam com o servidor para obter acesso aos recursos e funcionalidades disponíveis. O servidor, por sua vez, gerencia a lógica de negócios, processando as requisições dos clientes e interagindo com o banco de dados para armazenar e recuperar informações.

Ao adotar uma abordagem modular, o software de gestão de academia é composto por módulos independentes e intercambiáveis, cada um responsável por uma funcionalidade

específica. Isso facilita a manutenção e atualização do sistema, permitindo que novas funcionalidades sejam incorporadas sem afetar negativamente as já existentes. Além disso, a escalabilidade da arquitetura possibilita que o software possa lidar com um aumento no número de usuários e demanda de recursos sem comprometer o desempenho.

O software utiliza a linguagem de programação Elixir juntamente com o framework Phoenix para o desenvolvimento web. Além de JavaScript, Html e CSS em paralelo com ReactJS, com integração através da GraphQL. Por fim, o Ecto será utilizado como ORM para a comunicação com o banco de dados MySQL .

O sistema é organizado de acordo com o padrão MVC. Os principais módulos do sistema incluem Controle de Treinamentos, Controle de Usuários, Cálculo de IMC, Teste de Usuário, Aplicação Principal, Exercício, Repositório, Treinamento, Usuário, Instrutor e Administrador. Cada módulo possui responsabilidades específicas e utiliza as tecnologias mencionadas.

O software apresenta um conjunto de requisitos funcionais que especificam as funcionalidades que ele deve oferecer. Esses requisitos incluem:

- Criação de treinamentos personalizados: O sistema deve permitir que os treinadores ou administradores criem treinamentos personalizados para os clientes da academia, adaptando-os às necessidades individuais de cada usuário.

- Criação de novos usuários: O software deve permitir que os administradores adicionem novos usuários ao sistema, incluindo informações como nome, idade, contato, entre outros dados relevantes.

- Cálculo automatizado do IMC (Índice de Massa Corporal): O sistema deve ser capaz de calcular automaticamente o IMC com base nos dados de altura e peso fornecidos pelos usuários.

- Solicitação de treinamentos: Os usuários devem poder agendar horários para suas sessões de treinamento, de acordo com a disponibilidade de horários e instrutores.

- Registro de histórico: O software deve manter um registro detalhado das atividades e interações dos usuários, como treinos realizados, frequência de participação e progresso ao longo do tempo.

- Geração de relatórios de desempenho: O sistema deve fornecer a capacidade de gerar relatórios que apresentem informações sobre o desempenho dos usuários, como resultados de avaliações físicas, objetivos alcançados e métricas de progresso.

Além dos requisitos funcionais, foram identificados atributos de qualidade importantes para o software:

- Desempenho: O sistema deve ser responsivo e eficiente, garantindo que as operações sejam executadas em tempo hábil, mesmo em momentos de alto tráfego e carga de trabalho.

- Usabilidade: A interface do software deve ser intuitiva e amigável, permitindo que os usuários interajam facilmente com o sistema, mesmo sem experiência técnica avançada.

- Escalabilidade: O software precisa ser capaz de lidar com o crescimento do número de usuários e volume de dados sem comprometer o desempenho, garantindo que continue funcionando de forma eficaz à medida que a demanda aumenta.

- Manutenibilidade: A arquitetura deve permitir a manutenção do sistema de forma simples e rápida, com mínimos impactos ao introduzir novas funcionalidades ou corrigir problemas.

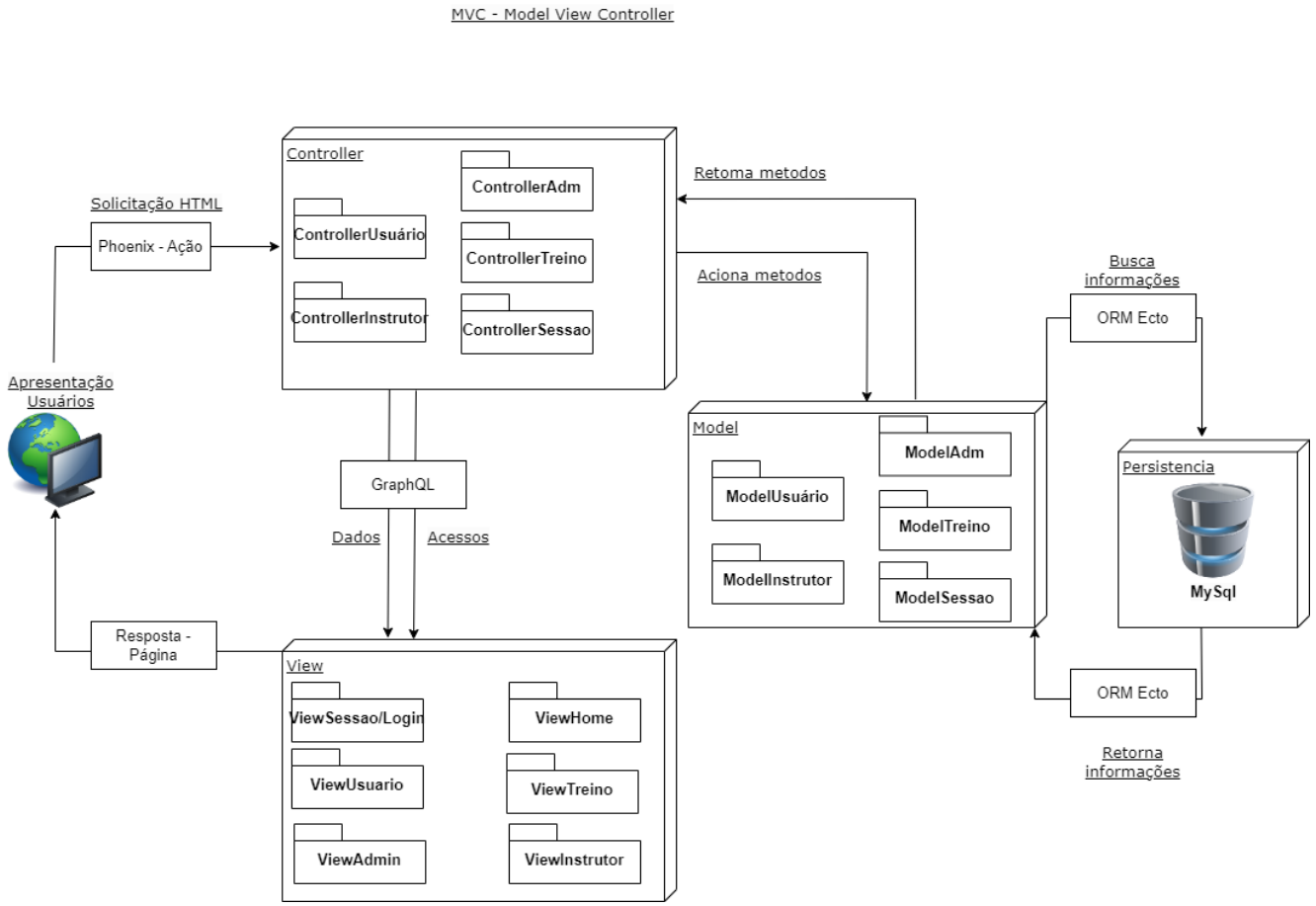
2. Representação Arquitetural

O software de gestão de academia adota uma arquitetura seguindo o padrão MVC (Model-View-Controller), que proporciona uma separação clara e organizada das responsabilidades no sistema. Nessa arquitetura, a lógica de negócios, a apresentação dos dados e o controle das interações são mantidos em módulos distintos, tornando o desenvolvimento e a manutenção mais eficientes.

A abordagem modular e escalável da arquitetura permite que novas funcionalidades possam ser adicionadas e atualizadas sem interferir no funcionamento do restante do sistema. Cada componente do MVC é independente, o que facilita a manutenção e evolução do software ao longo do tempo.

A arquitetura é baseada no modelo cliente-servidor, onde o servidor atua como o ponto central de processamento e armazenamento dos dados. Os clientes do sistema (aplicativo web) acessam o servidor para enviar solicitações, como a criação de novos usuários, agendamento de treinamentos ou geração de relatórios. O servidor, por sua vez, processa essas solicitações, realiza as operações necessárias no Model e retorna os resultados apropriados para serem apresentados na View.

A interface do usuário é implementada como um aplicativo web, o que permite que administradores da academia, instrutores e usuários finais acessem o sistema de qualquer dispositivo com acesso à internet. Essa abordagem torna o software mais acessível e facilita a distribuição de informações relevantes para os diversos usuários envolvidos na gestão da academia.



3. Metas e Restrições da Arquitetura

3.1 Metas da Arquitetura:

- Modularidade e Escalabilidade: A arquitetura deve ser modular para permitir o desenvolvimento independente de funcionalidades. Ela também deve ser escalável para acomodar o aumento do número de usuários e demandas de recursos.

- Flexibilidade e Adaptabilidade: A arquitetura deve permitir a inclusão e atualização de funcionalidades de forma flexível, para acompanhar as necessidades em constante evolução da academia.

- Separação de Responsabilidades: A arquitetura deve promover uma clara separação de responsabilidades entre os componentes do sistema, facilitando o gerenciamento e manutenção.

- Eficiência e Desempenho: O sistema deve ser eficiente em termos de processamento de solicitações e acesso aos recursos, garantindo um bom desempenho mesmo com aumento de carga.

- Integração com Tecnologias Modernas: A arquitetura deve ser compatível e integrada com as tecnologias escolhidas, como Elixir, Phoenix, JavaScript, HTML, CSS, ReactJS e GraphQL.

3.2 Restrições da Arquitetura:

- Linguagem de Programação: A arquitetura deve ser desenvolvida utilizando a linguagem de programação Elixir e o framework Phoenix.

- Tecnologias Front-end: O sistema deve utilizar JavaScript, HTML e CSS, com ReactJS como framework de escolha para a construção da interface do usuário.

- Integração através de GraphQL: A comunicação entre o front-end e back-end deve ser realizada por meio do protocolo GraphQL, que permite consultas flexíveis e eficientes aos dados.

- ORM Ecto e Banco de Dados MySQL: O sistema deve utilizar o ORM Ecto para a comunicação com o banco de dados MySQL.

- Padrão MVC: A arquitetura deve seguir o padrão Model-View-Controller (MVC) para organizar e separar a lógica de negócios, apresentação e manipulação de dados.

- Módulos Funcionais: O sistema será organizado em módulos específicos, como Controle de Treinamentos, Controle de Usuários, Cálculo de IMC, Teste de Usuário, entre outros. Cada módulo será responsável por funções bem definidas.

4. Visão de Processos

A visão de processos descreve como os diferentes componentes e módulos interagem para realizar as funcionalidades do sistema. Ela mostra como os casos de uso e a visão lógica se traduzem em fluxos de trabalho e ações concretas.

<u>Processo de Autenticação e Autorização:</u>

- O usuário acessa a página de login e insere suas credenciais.
- As credenciais são validadas pelo sistema.
- Se as credenciais forem válidas, o sistema autentica o usuário e gera um token JWT.

- Com base nas permissões do usuário, ele é redirecionado para a página principal ou área apropriada do sistema.
- O usuário pode navegar nas funcionalidades permitidas de acordo com as permissões.
- Quando o usuário escolhe fazer logout, o token de autenticação é invalidado.

Processo de Gerenciamento de Usuários:

- Um administrador acessa a interface de gerenciamento de usuários.
- O administrador cria um novo usuário, fornecendo os detalhes necessários.
- Os detalhes são validados e salvos no banco de dados.
- O administrador pode editar informações de usuários existentes ou excluí-los.
- As edições são validadas e atualizações são feitas no banco de dados.
- O histórico de treinamentos é registrado para novos usuários.

Processo de Gerenciamento de Treinamentos:

- Um instrutor acessa a interface de gerenciamento de treinamentos.
- O instrutor cria um novo treinamento personalizado, definindo detalhes como exercícios e intensidade.
- Os detalhes do treinamento são validados e salvos no banco de dados, associados ao usuário específico.
- O instrutor pode editar treinamentos existentes, validando as edições e atualizando os dados no banco de dados.
- Os treinamentos disponíveis são listados para instrutores e administradores, que podem selecionar treinamentos para usuários.

Processo de Cálculo do IMC:

- Um administrador ou instrutor acessa a interface de cálculo de IMC.
- Eles selecionam um arquivo CSV contendo os dados dos usuários.
- O sistema processa o arquivo, calcula os IMCs e os associa aos usuários correspondentes no banco de dados.
- Os resultados do cálculo são armazenados para referência futura.

Processo de Geração de Relatórios de Progresso:

- Um usuário acessa a interface de geração de relatórios.
- O usuário escolhe as métricas relevantes para o relatório.
- O sistema processa os dados do histórico de treinamentos e gera um relatório personalizado.
- O relatório é exibido para o usuário, mostrando detalhes sobre seu progresso e desempenho.

Processo de Listagem de Treinamentos Disponíveis:

- Instrutores ou administradores acessam a interface de listagem de treinamentos.
- A lista de treinamentos disponíveis é apresentada, recuperada da camada de armazenamento.
- Instrutores ou administradores selecionam os treinamentos que desejam oferecer aos usuários.

- As seleções são registradas no banco de dados, permitindo que os usuários visualizem e participem dos treinamentos.

Processo de Visualização do Histórico de Treinamentos:

- Os usuários acessam a interface de visualização de histórico.
- O sistema recupera os dados do histórico de treinamentos do banco de dados.
- Os usuários podem navegar e visualizar os treinamentos anteriores, incluindo detalhes relevantes.
- Se necessário, os usuários podem gerar relatórios personalizados para avaliar seu progresso.

5. Visões de caso de uso

Visão de caso de uso visa descrever como o sistema interage com os atores externos e os principais cenários de uso. Ela ajuda a definir e comunicar como os diferentes componentes do sistema se encaixam para atender aos requisitos funcionais do software.

Módulo de Autenticação e Autorização:

Realizar Login:

Ator: Usuário (Administrador, Instrutor, Usuário)
Descrição: O usuário fornece suas credenciais (nome de usuário e senha) para acessar o sistema.
Fluxo:

- O usuário acessa a interface de login.
- Insere suas credenciais.
- As credenciais são validadas.
- O sistema autentica o usuário e gera um token JWT.
- O usuário é redirecionado para a página principal do sistema.

Realizar Logout:

Ator: Usuário (Administrador, Instrutor, Usuário)
Descrição: O usuário encerra sua sessão no sistema.
Fluxo:

- O usuário seleciona a opção de logout.
- O sistema invalida o token de autenticação.
- O usuário é redirecionado para a página de login.

Módulo de Usuários:

Criar Novo Usuário:

Ator: Administrador
Descrição: O administrador cria um novo usuário no sistema.
Fluxo:

- O administrador acessa a interface de gerenciamento de usuários.
- Seleciona a opção de criar um novo usuário.
- Insere os detalhes do novo usuário.

- Os dados são validados.
- O novo usuário é criado e registrado no sistema.

Editar Informações de Usuário:

Ator: Administrador

Descrição: O administrador edita as informações de um usuário existente.

Fluxo:

- O administrador acessa a interface de gerenciamento de usuários.
- Seleciona o usuário que deseja editar.
- Realiza as edições necessárias nos detalhes do usuário.
- As edições são validadas.
- As informações do usuário são atualizadas no sistema.

Visualizar Histórico de Treinamentos:

Ator: Usuário (Administrador, Instrutor, Usuário)

Descrição: O usuário visualiza o histórico de treinamentos.

Fluxo:

- O usuário acessa a interface de visualização de histórico.
- O sistema valida o acesso ao histórico de treinamento.
- O sistema exibe os treinamentos anteriores do usuário, com detalhes relevantes.

Módulo de Treinamentos:

Solicitar Treinamento Personalizado:

Ator: Usuário

Descrição: O usuário solicita um treinamento novo para si.

Fluxo:

- O usuário acessa a interface de visualização de histórico.
- Seleciona a opção de solicitar novo treinamento.
- Define os detalhes do treinamento (exercícios, intensidade, duração, etc.).
- Os dados são validados.
- O treinei será direcionado ao instrutor para ser aprovado

Criar Treinamento Personalizado:

Ator: Instrutor

Descrição: O instrutor cria um novo treinamento personalizado para um usuário.

Fluxo:

- O instrutor acessa a interface de gerenciamento de treinamentos.
- Seleciona a opção de criar um novo treinamento.
- Define os detalhes do treinamento (exercícios, intensidade, duração, etc.).
- Os dados são validados.
- O treinamento é associado ao usuário específico.

Editar Treinamento Existente:

Ator: Instrutor

Descrição: O instrutor edita um treinamento personalizado existente.

Fluxo:

- O instrutor acessa a interface de gerenciamento de treinamentos.
- Seleciona o treinamento que deseja editar.
- Realiza as edições necessárias nos detalhes do treinamento.
- As edições são validadas.
- As informações do treinamento são atualizadas.

Listar Treinamentos Disponíveis:

Ator: Instrutor, Administrador

Descrição: O instrutor ou administrador visualiza a lista de treinamentos disponíveis.

Fluxo:

- O instrutor ou administrador acessa a interface de listagem de treinamentos.
- O sistema exibe a lista de treinamentos disponíveis.

Módulo de IMC:

Calcular IMC a partir de Arquivo CSV:

Ator: Administrador, Instrutor e Usuário

Descrição: O administrador, instrutor ou usuário realiza a exportação do cálculo do IMC dos usuários a partir de um arquivo CSV.

Fluxo:

- O ator acessa a interface de cálculo de IMC.
- Seleciona o arquivo CSV contendo os dados dos usuários.
- O sistema processa os dados e calcula os IMCs.
- Os resultados são associados aos respectivos usuários.

Módulo de Histórico de Treinamentos:

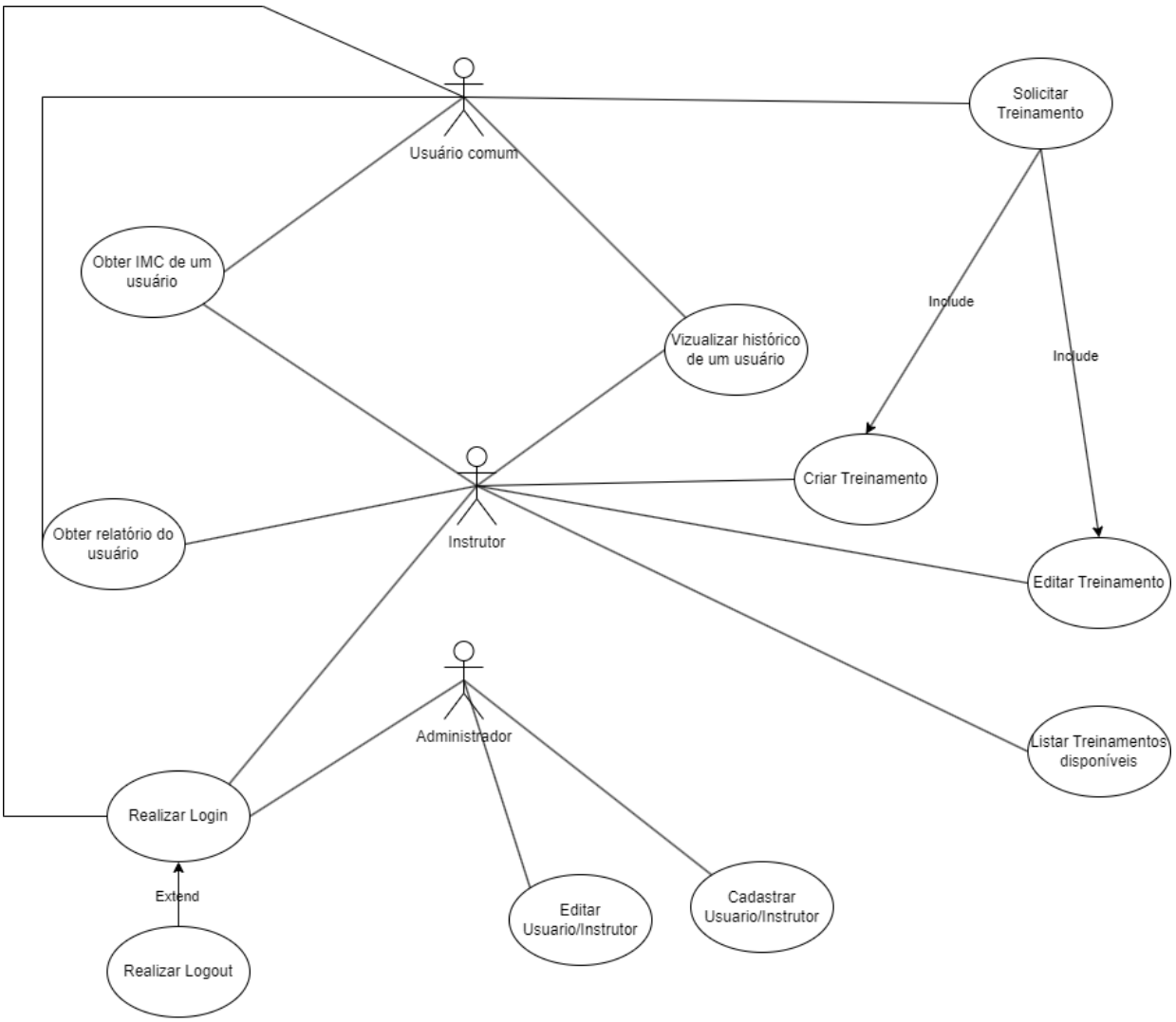
Visualizar Relatório de Progresso:

Ator: Usuário (Administrador, Instrutor, Usuário)

Descrição: O usuário gera um relatório personalizado para acompanhar seu progresso.

Fluxo:

- O usuário acessa a interface de geração de relatórios.
- Seleciona as métricas relevantes para o relatório.
- O sistema processa os dados do histórico de treinamentos.
- O relatório é exibido com as métricas selecionadas.



6. Visão Lógica

O sistema será composto por módulos interconectados que trabalharão juntos para oferecer as funcionalidades necessárias para a gestão da academia. Vamos detalhar os principais componentes:

Módulos:

Módulo de Autenticação e Autorização:

-Responsável pelo controle de acesso dos usuários ao sistema.
-Permite o login de administradores, instrutores e usuários.
-Gerencia as permissões de cada tipo de usuário.

Login:

- Interface para os usuários inserirem suas credenciais de login (nome de usuário e senha).
- Envia as credenciais para validação.

Validação de Credenciais:

- Valida as credenciais fornecidas pelo usuário em relação aos registros do banco de dados.
- Verifica se as credenciais são válidas para um usuário existente e registrado no sistema.

Autenticação:
- Após a validação das credenciais, o sistema autentica o usuário.
- Gera um token de autenticação (JSON Web Token) contendo informações sobre o usuário e suas permissões.

Gerenciamento de Permissões:
- O sistema mantém uma lista de permissões associadas a cada tipo de usuário (administradores, instrutores, usuários).
- Quando um usuário é autenticado, suas permissões são carregadas a partir do banco de dados ou de um serviço dedicado.

Controle de Acesso:
- Com base nas permissões do usuário autenticado, o módulo controla o acesso às funcionalidades do sistema.
- Permite ou nega o acesso a diferentes partes do sistema com base nas permissões definidas.

Logout:
- Fornecer uma opção para que os usuários possam encerrar sua sessão de forma segura.
- Invalida o token de autenticação, tornando-o inválido para futuras solicitações.

Fluxo de Autenticação e Autorização:
- Um usuário acessa a interface de login e insere suas credenciais.
- As credenciais são enviadas para validação.
- O sistema verifica se as credenciais são válidas e autentica o usuário.
- Com base nas informações de autenticação, o sistema carrega as permissões do usuário.
- O usuário autenticado pode acessar as funcionalidades apropriadas do sistema de acordo com suas permissões.
- Quando o usuário deseja sair, o logout é realizado, invalidando o token de autenticação.

Módulo de Usuários:
-Permite visualização de treinos vinculados ao usuário.
-Permite solicitação de novos treinos.
-Mantém informações detalhadas sobre os usuários, incluindo histórico de treinamentos.

Interface de Usuário:
- Esta é a interface pela qual os administradores interagem com o Módulo de Usuários.

- Ela permite a inserção de dados de novos usuários, edição de informações existentes e exclusão de usuários.

Serviço de Gerenciamento de Usuários:

- Este serviço lida com todas as operações relacionadas aos usuários.
- Ele recebe e processa as solicitações da interface de usuário e realiza a validação dos dados antes de interagir com a camada de armazenamento.

Camada de Armazenamento:

- Esta camada é responsável por persistir os dados dos usuários, incluindo suas informações pessoais e histórico de treinamentos.
- Ela se comunica com o serviço de gerenciamento de usuários para salvar e recuperar informações.

Mecanismo de Validação:

- Antes que os dados dos usuários sejam processados, eles passam por um mecanismo de validação que verifica a integridade e precisão das informações inseridas.
- Se os dados forem inválidos, a interface de usuário é notificada com mensagens de erro apropriadas.

Fluxo de Funcionamento:

- Um administrador acessa a interface de usuário do Módulo de Usuários.
- O administrador escolhe a opção de criar um novo usuário e fornece os detalhes necessários, como nome, idade, gênero, etc.
- Os dados inseridos são enviados para o Serviço de Gerenciamento de Usuários.
- O Serviço de Gerenciamento de Usuários valida os dados usando o Mecanismo de Validação, garantindo que todas as informações necessárias sejam fornecidas e que elas estejam em um formato válido.
- Após a validação, o Serviço de Gerenciamento de Usuários solicita à Camada de Armazenamento que salve as informações do novo usuário.
- O histórico de treinamentos é iniciado para o novo usuário.
- Quando um administrador deseja editar as informações de um usuário existente, a interface de usuário encaminha a solicitação ao Serviço de Gerenciamento de Usuários.
- O Serviço de Gerenciamento de Usuários verifica a validade dos dados de edição e atualiza as informações na Camada de Armazenamento, mantendo o histórico de treinamentos intacto.
- A exclusão de um usuário segue um processo semelhante, onde o Serviço de Gerenciamento de Usuários remove as informações do usuário e histórico de treinamentos da Camada de Armazenamento.

Módulo de Treinamentos:

-Permite a criação e edição de treinamentos personalizados.
-Validação dos dados dos treinamentos, como exercícios e intensidade.
-Associa treinamentos a usuários específicos.

Interface de Usuário:

- A interface permite que os administradores da academia interajam com o Módulo de Treinamentos.
- Eles podem criar novos treinamentos, definir parâmetros, associar treinamentos a usuários e realizar edições.

Serviço de Gerenciamento de Treinamentos:

- Este serviço cuida de todas as operações relacionadas a treinamentos.
- Ele recebe e processa solicitações da interface de usuário, garante a validação dos dados e interage com a camada de armazenamento.
-

Camada de Armazenamento:

- Esta camada persiste nas informações sobre os treinamentos, mantendo registros detalhados sobre os exercícios, intensidade, duração e a associação com os usuários.

Mecanismo de Validação:

- Antes de processar os dados dos treinamentos, eles passam pelo mecanismo de validação, que garante que todos os detalhes estejam corretos e em um formato válido.

Fluxo de Funcionamento:

- Um administrador acessa a interface de usuário do Módulo de Treinamentos.
- O administrador escolhe a opção de criar um novo treinamento personalizado e fornece os detalhes necessários, como exercícios, intensidade, duração, etc.
- Os dados inseridos são enviados para o Serviço de Gerenciamento de Treinamentos.
- O Serviço de Gerenciamento de Treinamentos valida os dados usando o Mecanismo de Validação, garantindo que todas as informações sejam fornecidas de maneira correta.
- Após a validação, o Serviço de Gerenciamento de Treinamentos solicita à Camada de Armazenamento que salve as informações do novo treinamento.
- O treinamento é associado ao usuário específico para o qual foi criado.
- Quando um administrador deseja editar um treinamento existente, a interface de usuário encaminha a solicitação ao Serviço de Gerenciamento de Treinamentos.
- O Serviço de Gerenciamento de Treinamentos verifica a validade dos dados de edição e atualiza as informações na Camada de Armazenamento, mantendo a associação com o usuário intacta.

Módulo de IMC:

-Processa os dados de um arquivo CSV para calcular o IMC dos usuários.
-Fornece resultados precisos e rápidos do cálculo do IMC.

Interface de Usuário:

- A interface permite que os administradores ou usuários acessem o Módulo de IMC e escolham o arquivo CSV contendo os dados dos usuários para calcular o IMC.

Serviço de Processamento de IMC:

- Este serviço é responsável por processar os dados do arquivo CSV e calcular o IMC para cada usuário.
- Ele se comunica com a camada de armazenamento e a interface de usuário.

Camada de Armazenamento:
- Esta camada é responsável por persistir os resultados do cálculo do IMC e associá-los aos usuários correspondentes.
- Isso permite que os resultados sejam acessados posteriormente, se necessário.

Fluxo de Funcionamento:
- Um administrador ou usuário acessa a interface de usuário do Módulo de IMC.
- A interface permite que o usuário escolha um arquivo CSV que contém os dados dos usuários para calcular o IMC.
- O arquivo CSV é enviado para o Serviço de Processamento de IMC.
- O Serviço de Processamento de IMC lê os dados do arquivo CSV, aplica as fórmulas apropriadas para calcular o IMC de cada usuário e armazena os resultados na Camada de Armazenamento.
- Os resultados do cálculo do IMC são associados aos usuários específicos, permitindo que seus IMCs sejam rastreados individualmente.
- Quando necessário, os resultados do cálculo do IMC podem ser acessados por administradores, instrutores ou usuários, dependendo das permissões de acesso.

Módulo de Listagem de Treinamentos:
-Lista todos os treinamentos disponíveis na academia.
-Permite aos instrutores ou administradores selecionarem treinamentos para que os usuários participem.

Interface de Usuário:
- A interface exibe a lista de treinamentos disponíveis na academia.
- Ela também permite que instrutores ou administradores selecionem treinamentos para os usuários.

Serviço de Listagem de Treinamentos:
- Este serviço é responsável por recuperar a lista de treinamentos disponíveis do banco de dados e apresentá-la na interface de usuário.
- Ele também facilita a seleção de treinamentos pelos instrutores ou administradores.

Camada de Armazenamento:
- Esta camada contém os dados relacionados aos treinamentos disponíveis na academia.
- O serviço de listagem de treinamentos acessa esses dados para apresentá-los aos usuários.

Fluxo de Funcionamento:
- Os instrutores ou administradores acessam a interface de usuário do Módulo de Listagem de Treinamentos.

- A interface exibe uma lista de todos os treinamentos disponíveis na academia, recuperando os dados do Serviço de Listagem de Treinamentos.
- Os instrutores ou administradores têm a opção de selecionar os treinamentos que desejam oferecer aos usuários.
- Ao selecionar um treinamento, a interface comunica essa seleção de volta ao Serviço de Listagem de Treinamentos.
- O Serviço de Listagem de Treinamentos registra a seleção e pode atualizar o estado dos treinamentos selecionados no banco de dados, para que os usuários saibam quais treinamentos estão disponíveis para eles.
- Os usuários acessam a interface de usuário, que agora exibe os treinamentos disponíveis para participação.
- Os usuários selecionam os treinamentos que desejam participar e podem reservar horários, se necessário.

Módulo de Histórico de Treinamentos:
-Registra e mantém um histórico de treinamentos de cada usuário.
-Permite o acompanhamento do progresso e desempenho individual.
-Gera relatórios personalizados para cada usuário.
-Inclui métricas relevantes para avaliar o progresso e desempenho.

Interface de Usuário:
- A interface exibe o histórico de treinamentos de cada usuário e permite que os usuários visualizem seu progresso e desempenho.
- Ela possibilita a geração de relatórios personalizados.

Serviço de Histórico de Treinamentos:
- Este serviço é responsável por registrar os treinamentos de cada usuário e armazená-los na camada de armazenamento.
- Ele também gera relatórios personalizados com base nos dados do histórico.

Camada de Armazenamento:
- Esta camada contém os dados do histórico de treinamentos de cada usuário.
- O Serviço de Histórico de Treinamentos acessa e atualiza esses dados conforme necessário.

Fluxo de Funcionamento:
- Os usuários acessam a interface de usuário do Módulo de Histórico de Treinamentos.
- A interface exibe o histórico de treinamentos do usuário, recuperando os dados do Serviço de Histórico de Treinamentos.
- Os usuários podem navegar pelo histórico e visualizar os treinamentos que realizaram ao longo do tempo.
- Os usuários também têm a opção de gerar relatórios personalizados para avaliar seu progresso e desempenho. Ao selecionar essa opção, a interface se comunica com o Serviço Histórico de Treinamentos.

- O Serviço de Histórico de Treinamentos processa os dados do histórico do usuário e gera um relatório personalizado com base nas métricas relevantes, como frequência de treinamento, progresso em relação aos objetivos, calorias queimadas, etc.
- O relatório personalizado é retornado à interface de usuário, onde o usuário pode visualizá-lo, salvá-lo ou imprimir conforme necessário.

Funções e Métodos:

Controllers:

UserController (Controller/Usuario.ex):

- acess_training
- solicit_training
- delete_training

AdminController (Controller/Administrador.ex):

- create_user
- edit_user
- delete_user
- delete_training
- create_instructor
- delete_instructor

InstructorController (Controller/Instrutor.ex):

- create_user
- edit_user
- create_training
- edit_training
- delete_user
- delete_training

TrainingController (Controller/Treino.ex):

- link_user_to_training
- report_Imc
- export_report

SessionController (Controller/Sessao.ex):

- login
- logout
- expiration_time

Models:

User (Model/Usuario.ex):

- id
- nome
- idade
- altura
- peso

- TreinoVinculado
- InstrutorVinculado

Admin (Model/Administrador.ex):
- id
- nome
- email
- senha

Instructor (Model/Instrutor.ex):
- id
- nome
- email
- senha
- especialidade

Training (Model/Treino.ex):
- id
- nome
- exercícios
- dias
- duração

TrainingHist (Model/Treino.ex):
- id
- data_realizacao
- avaliacao
- observacoes
- UsuarioAssociado
- TreinoAssociado

Views:
SessionView:
- render_token
- render_error

HomeView:
- render_home

UserView:
- render_user
- render_users

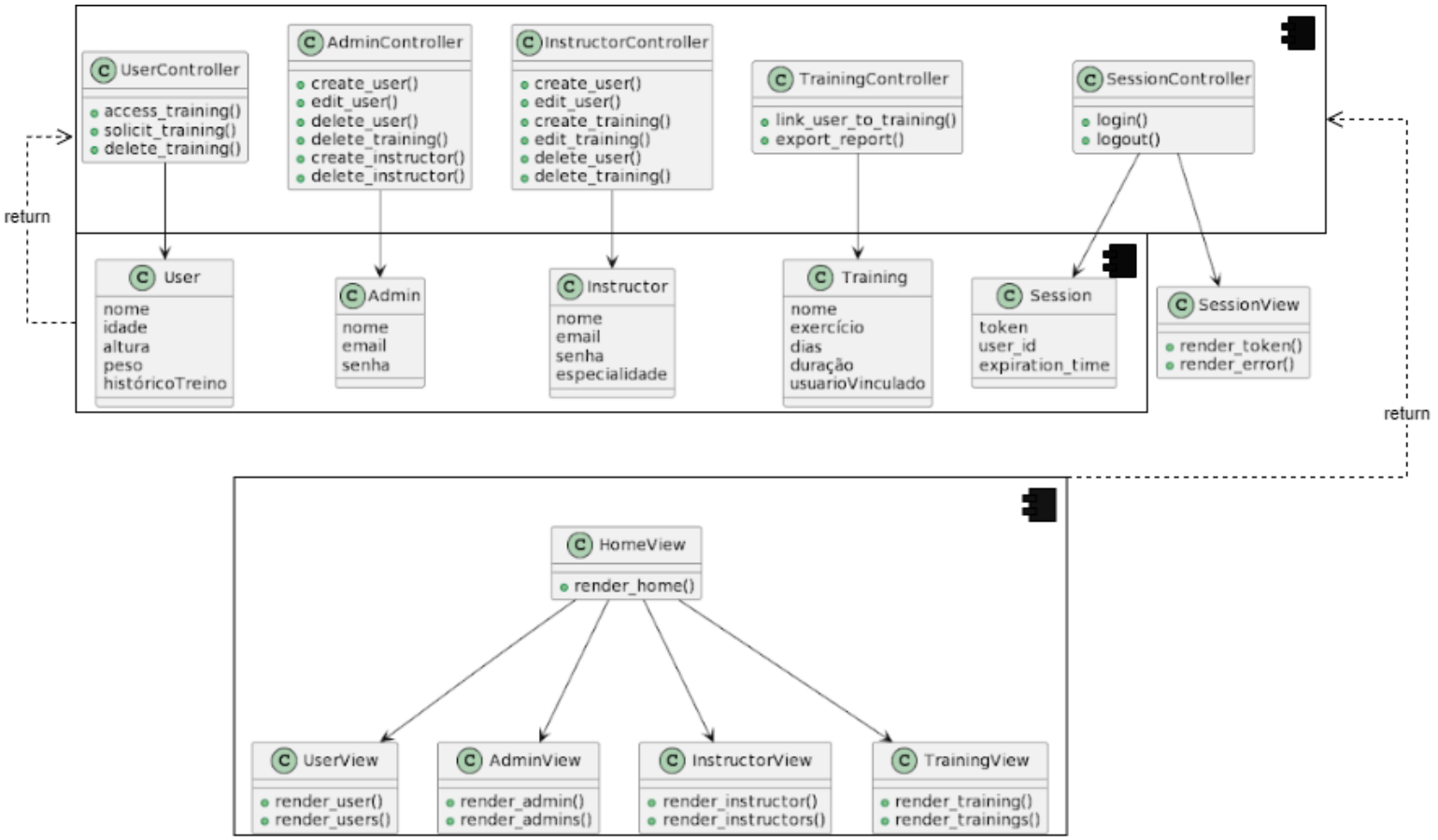
AdminView:
- render_admin
- render_admins

InstructorView:

- render_instructor
- render_instructors

TrainingView:

- render_training
- render_trainings
- render_Alltrainings



Associações:

- As linhas que conectam as classes representam associações entre elas.
- Por exemplo, os controladores (UserController, AdminController, etc.) interagem com os modelos correspondentes (por exemplo, User, Admin) para executar ações como acessar e gerenciar dados.
- Da mesma forma, os controladores interagem com as visões para exibir informações aos usuários.

Herança:

- A seta de herança indica que certas classes herdam propriedades e comportamentos de classes pai.
- Por exemplo, o controlador SessionController herda os métodos associados à classe pai Session.

Relacionamentos:
- Os relacionamentos entre controladores e modelos representam como os controladores manipulam dados nos modelos.
- Por exemplo, o UserController interage com o User para gerenciar operações relacionadas a usuários.

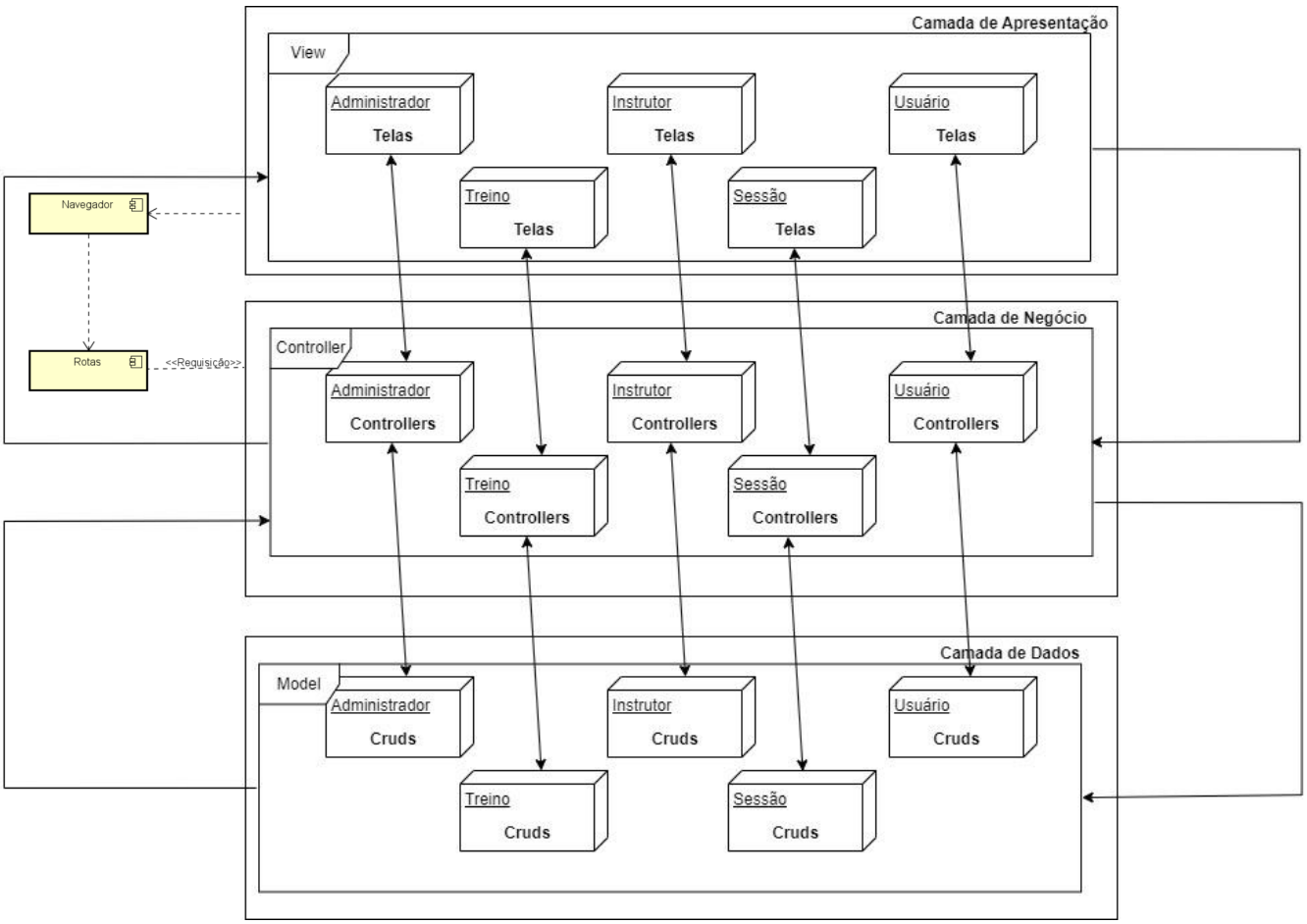
Composição/Agregação:
- As classes como User e Training têm atributos (como históricoTreino e usuarioVinculado) que implicam algum tipo de composição ou agregação.
- Esses atributos são outras classes/entidades que fazem parte da estrutura da classe principal.

Visões Conectando-se aos Controladores:
- As visões (SessionView, HomeView, etc.) interagem com os controladores para solicitar e exibir dados relevantes aos usuários.
- Essa interação ajuda na renderização do conteúdo apropriado na interface do usuário.

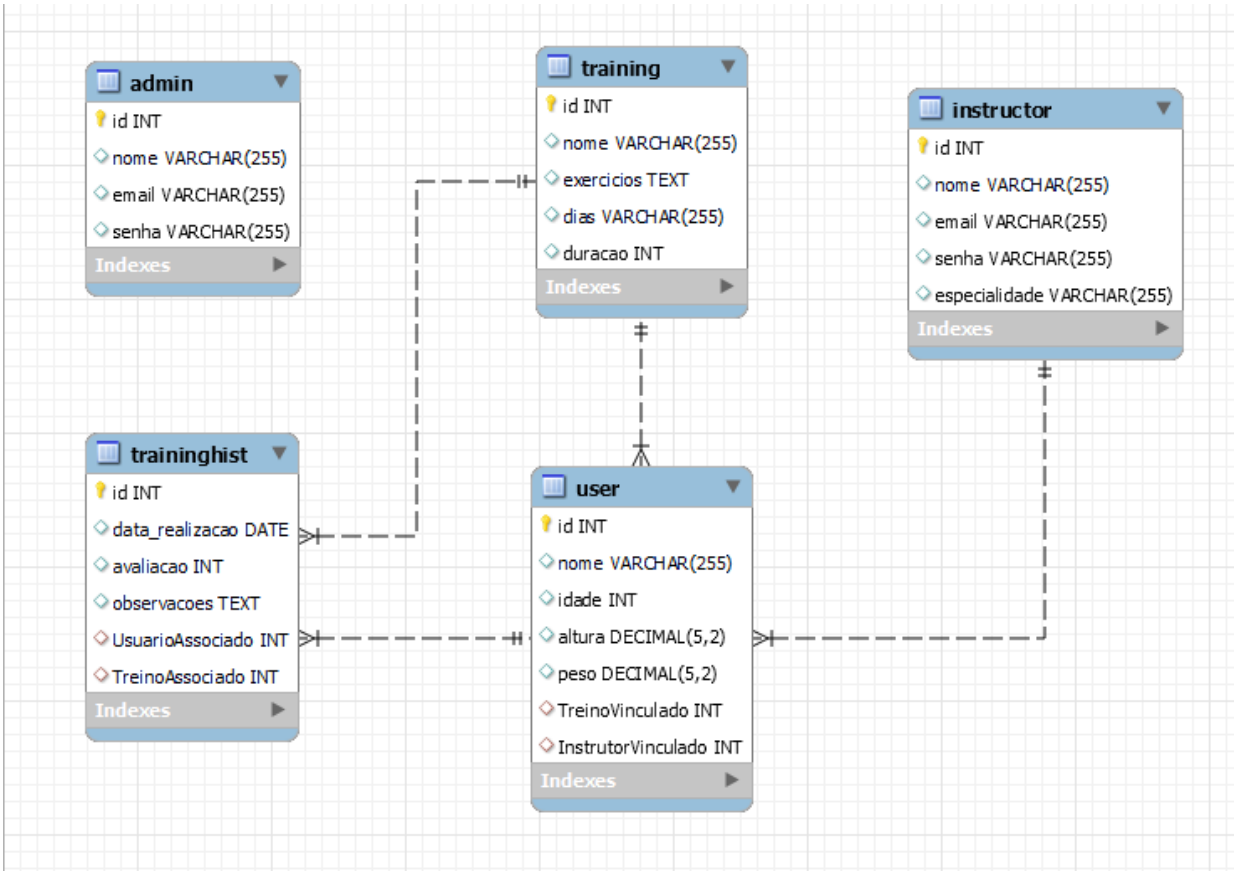
7. Visão de Implementação

O sistema terá três níveis de acesso, cada um deles dividido em duas abordagens arquiteturais. O primeiro nível segue o modelo cliente-servidor, onde são aplicadas as regras de apresentação, lógica de negócios e manipulação de dados. O segundo nível de arquitetura segue o padrão MVC. Ele adere às regras de roteamento e é composto por três camadas: visão, modelo e controlador.

As rotas têm a responsabilidade de encaminhar os usuários para as visualizações solicitadas, por meio da comunicação com o controlador, que por sua vez se comunica com o modelo. O modelo é responsável por lidar com a persistência dos dados no banco de dados.



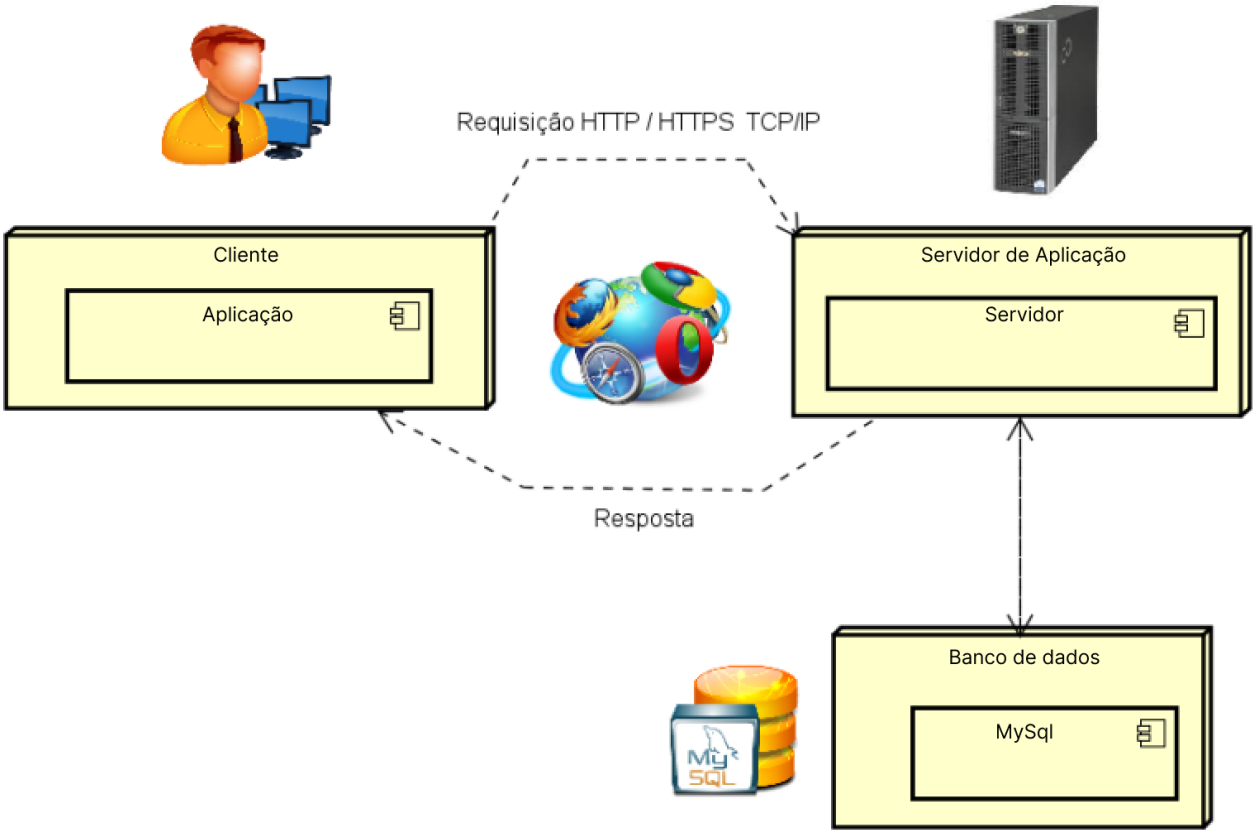
7.1 Estrutura de dados:



8. Visão de Implantação

O Software de Gestão de Academia - Haltere é uma aplicação WEB o qual seguirá um padrão de Cliente-Servidor, toda sua implantação é composta em três fases: Cliente, que faz acesso da aplicação; Servidor da Aplicação, onde a partir de uma requisição HTTP/HTTPS ele faz a interação com a aplicação e assim como suas regras de negócios, tendo acesso aos recursos do mesmo e a camada de Banco de dados que é o local onde estarão armazenados todos os dados que nele foi salvo, todos se utilizam da camada TCP/IP para comunicação.

Para um bom desempenho, exige-se alguns requisitos mínimos para uma melhor performance parte de princípios do lado do Cliente onde o mesmo necessita de um Hardware com Processador de 1Ghz; 1GB de Memória Ram; 16GB de espaço em Disco e uma conexão com a internet de 1 megabits de velocidade, já do lado do Servidor da Aplicação/Negócio um Hardware com Processador AMD EPYC™ 7343 3.2GHz, 16C/32T, 128M Cache (190W) DDR4-3200; 16GB RDIMM, 3200MT/s, duas fileiras, BCC; 3.84TB SSD SAS RI 24Gbps 512e 2.5in Hot-Plug, AG Drive 1DWPD e uma conexão com a internet de 100 megabits de velocidade.



9. Decisões Arquiteturais

A arquitetura do software de gestão de academia foi projetada com base em várias decisões importantes, cada uma delas com justificativas específicas:

Escolha da Arquitetura MVC:

A decisão de adotar a arquitetura MVC foi tomada para separar claramente as preocupações relacionadas ao modelo de dados, lógica de negócios e apresentação. Isso torna o sistema mais organizado, facilita a manutenção e permite o desenvolvimento independente de cada componente.

A adoção do padrão MVC proporciona uma organização clara das responsabilidades no sistema, tornando mais fácil para os desenvolvedores entenderem, implementarem e manterem as diferentes partes do software.

Ademais permite que novas funcionalidades sejam adicionadas de forma independente, sem afetar as partes já existentes do sistema. Essa decisão foi tomada para

garantir a facilidade de manutenção e a possibilidade de evolução do software de acordo com as necessidades em constante mudança.

Utilização de Elixir e Phoenix:

A escolha da linguagem Elixir e do framework Phoenix para o back-end se deve à natureza concorrente e escalável do Elixir, que é adequada para sistemas que precisam lidar com muitas solicitações simultâneas, como uma aplicação web de academia. O Phoenix oferece recursos de web framework que facilitam o desenvolvimento e o gerenciamento de rotas, controladores e views.

Escolha do ReactJS e GraphQL para o Front-End:

O ReactJS foi selecionado para a construção do front-end devido à sua eficiência na criação de interfaces de usuário dinâmicas e reativas. A integração com GraphQL permite consultas flexíveis aos dados do back-end, reduzindo a sobrecarga de excessivas solicitações HTTP e melhorando a eficiência da comunicação entre cliente e servidor.

Banco de Dados MySQL e ORM Ecto:

A escolha do banco de dados MySQL foi feita considerando sua ampla adoção e confiabilidade como um sistema de gerenciamento de banco de dados relacional. O ORM Ecto proporciona uma maneira eficiente e elegante de interagir com o banco de dados a partir da linguagem Elixir, abstraindo a complexidade das consultas SQL e facilitando a persistência dos dados.

Padrão Cliente-Servidor:

A escolha do modelo cliente-servidor se baseia na necessidade de separar as responsabilidades entre as interfaces de usuário (clientes) e a lógica de processamento (servidor). Isso permite que os clientes solicitem recursos e funcionalidades ao servidor, que por sua vez gerencia as operações e interações com o banco de dados.

Abordagem Web para Acessibilidade e Distribuição:

A decisão de implementar o sistema como uma aplicação web foi motivada pela acessibilidade e facilidade de uso. Usuários de diferentes funções (administradores, instrutores, usuários finais) podem acessar o sistema de qualquer dispositivo com acesso à internet, o que aumenta a distribuição das informações e a capacidade de gerenciamento remoto.

10. Conclusão

Em conclusão, a elaboração deste documento de Arquitetura Final de Sistema para o Software de Academia (Haltere) representa um marco significativo no desenvolvimento de uma solução tecnológica que visa transformar a maneira como alunos e instrutores abordam o treinamento físico. Ao longo deste processo, examinamos cuidadosamente os requisitos funcionais e não funcionais, exploramos opções arquiteturais, modelamos os componentes e

fluxos de dados, e delineamos as decisões-chave de design que moldaram a implementação e operação do sistema.

A arquitetura proposta, baseada em uma cliente-servidor, demonstra a capacidade de atender às demandas variadas e em constante evolução de uma academia moderna. A divisão clara entre os módulos de criação de perfil, geração de programas de treinamento e agendamento flexível proporciona uma base sólida para a personalização das experiências dos alunos, ao mesmo tempo que permite uma administração eficiente por parte dos instrutores.

Em suma, o Software de Academia (Haltere), conforme delineado nesta arquitetura final, tem o potencial de revolucionar a forma como a educação física é abordada. Ao proporcionar personalização, interatividade e análise de dados detalhada, o sistema se tornará um aliado valioso para a busca do bem-estar e condicionamento físico eficazes. Estamos confiantes de que a implementação bem-sucedida dessa arquitetura resultará em um software altamente eficiente, capaz de atender às necessidades tanto dos alunos quanto dos instrutores, moldando positivamente o cenário da educação física moderna.