**FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA**

CARLOS EDUARDO MENDONÇA DA SILVA

EDUARDO TOSHIO ROCHA OKUBO

KAUÊ ALEXANDRE DE OLIVEIRA

MATEUS VINICIUS DA CONCEIÇÃO SILVA

VITOR MACHADO MIRANDA

**DOCUMENTAÇÃO DO PROJETO VIVABEM**

CARLOS EDUARDO MENDONÇA DA SILVA

EDUARDO TOSHIO ROCHA OKUBO

KAUÊ ALEXANDRE DE OLIVEIRA

MATEUS VINICIUS DA CONCEIÇÃO SILVA

VITOR MACHADO MIRANDA

**DOCUMENTAÇÃO DO PROJETO VIVABEM**

Trabalho acadêmico apresentado à disciplina de Domain Driven Design do Curso Análise e Desenvolvimento de Sistemas da Faculdade de Informática e Administração Paulista como requisito de nota da Global Solution da Turma 1TDSPV. Requerido pelo prof. Leonardo Gasparini Romão.

**SUMÁRIO**

**1 INTRODUÇÃO .................................................................................................................... 4**

**2 NOSSA SOLUÇÃO................................................................................................................ 5**

**3 PROTÓTIPO DAS TELAS........................................................................................................6**

**3.1 HOME PAGE..........................................................................................................6**

**3.2 LOG-IN E CADASTRO.............................................................................................7**

**3.3 TELA DO PERFIL....................................................................................................9**

**3.4 TELA DOS TREINOS...............................................................................................9**

**3.5 TELAS DO PLANO DE DIETA.................................................................................10**

**4 COMO FAZER CHAMADAS A API.........................................................................................12**

**4.1 RECUPERANDO TODOS OS DADOS......................................................................13**

**4.2 RECUPERANDO UM DADO ESPECÍFICO................................................................14**

**4.3 ADICIONANDO UMA NOVA INFORMAÇÃO..........................................................15**

**4.4 ATUALIZANDO UMA INFORMAÇÃO.....................................................................16**

**4.5 DELETANDO UMA INFORMAÇÃO........................................................................16**

**4.6 VERIFICAÇÃO DE LOGIN......................................................................................17**

**5 MODELO DO BANCO DE DADOS.........................................................................................18**

**6 UML DAS CLASSES .............................................................................................................19**

**1 INTRODUÇÃO**

Como alunos da turma 1TDSPV, nos foi proposto como desafio pela FIAP, em parceria com a empresa Hapvida NotreDame Intermédica, desenvolver uma solução tecnológica que contribua para enfrentar os desafios ligados ao Objetivo de Desenvolvimento Sustentável (ODS) 3 da ONU, referente a “Boa Saúde e Bem-Estar”.

A área da saúde teve grandes benefícios proporcionados pelos avanços tecnológicos dos últimos anos, o que refletiu de maneira geral em um aumento da qualidade de vida e bem-estar para as pessoas. Contudo, contrapondo essas melhorias, as novas tecnologias também trouxeram uma notável crescente dos problemas de saúde decorrentes do sedentarismo. Tendo em vista essa questão, o projeto VivaBem tem como objetivo desenvolver um auxílio inteligente e tecnológico que ajude as pessoas a terem uma vida mais ativa e se prevenirem de danos a sua saúde.

Para alcançar esse objetivo, usaremos das tecnologias que foram aprendidas ao longo do ano para o desenvolvimento do Front-End e do Back-End, para modelagem de um banco de dados e para a utilização de modelos de classificação com IA. O resultado desse projeto será uma melhoria significativa na qualidade de vida das pessoas, que terão a sua disposição uma tecnologia para as auxiliares nas práticas de uma vida mais saudável.

**2 NOSSA SOLUÇÃO**

Para que então consigamos melhorar a vida das pessoas o projeto VivaBem será uma aplicação onde o usuário terá a sua disposição diversos treinos para um melhor condicionamento físico e a dietas bem planejadas, tudo sobe o auxílio de IA.

Assim que o usuário, o nosso cliente, acessar o site ele terá a possibilidade de se fazer seu log-in (caso já possua seu cadastro) ou de se cadastrar-se. No cadastro ele irá informar seus dados como nome, e-mail, idade, etc, e em seguida ele nos informará seu objetivo, se será a perda de gordura ou o ganho de massa muscular, além de poder especificar o peso que pretende chegar e a data em que deseja atingir essa meta. Adiante ele poderá informar seu peso, estatura e nível de atividade para que possamos calcular seu metabolismo basal, por meio de IA, e podermos oferecer uma dieta mais precisa. Enfim, ele poderá nos informar seu biotipo ou até mesmo descobrir dentro do próprio site, também com o auxílio de IA, e então para finalizar o usuário escolherá um nível de treino, seja iniciante, intermediário ou avançado.

Já conectado ao site, o usuário terá uma tela onde poderá visualizar suas informações e acompanhar sua evolução, além de alterar essas informações caso precise e adicionar medidas de seu corpo que poderão ajudar no seu acompanhamento. A hora que ele quiser, poderá também o nível de seu treino caso ache que está muito fácil ou difícil e mudar o foco da sua dieta, seja para emagrecimento ou ganho de músculos. Quando for começar o treino, ele pode escolher qual parte do corpo pretende focar (peitoral, costas, bíceps, etc) e então lhe será apresentado os exercícios ideais para ele, com o número de repetições e séries, além do tempo de descanso. Em relação as dietas, ele terá a sua disposição um plano pensado para seu café da manhã, almoço, lanche da tarde e jantar.

Para garantir o sucesso do projeto, será necessário contar com uma equipe multidisciplinar de desenvolvedores e analistas de dados. Além disso, será de ser necessário investir em treinamento e capacitação dos nossos operadores para garantir que eles possam utilizar o sistema com eficiência e precisão. Dessa forma, com a implementação desse projeto, esperamos melhorar significativamente a qualidade de vida das pessoas, as proporcionando uma rotina mais saudável e ativa.

**3 PROTÓTIPO DAS TELAS**

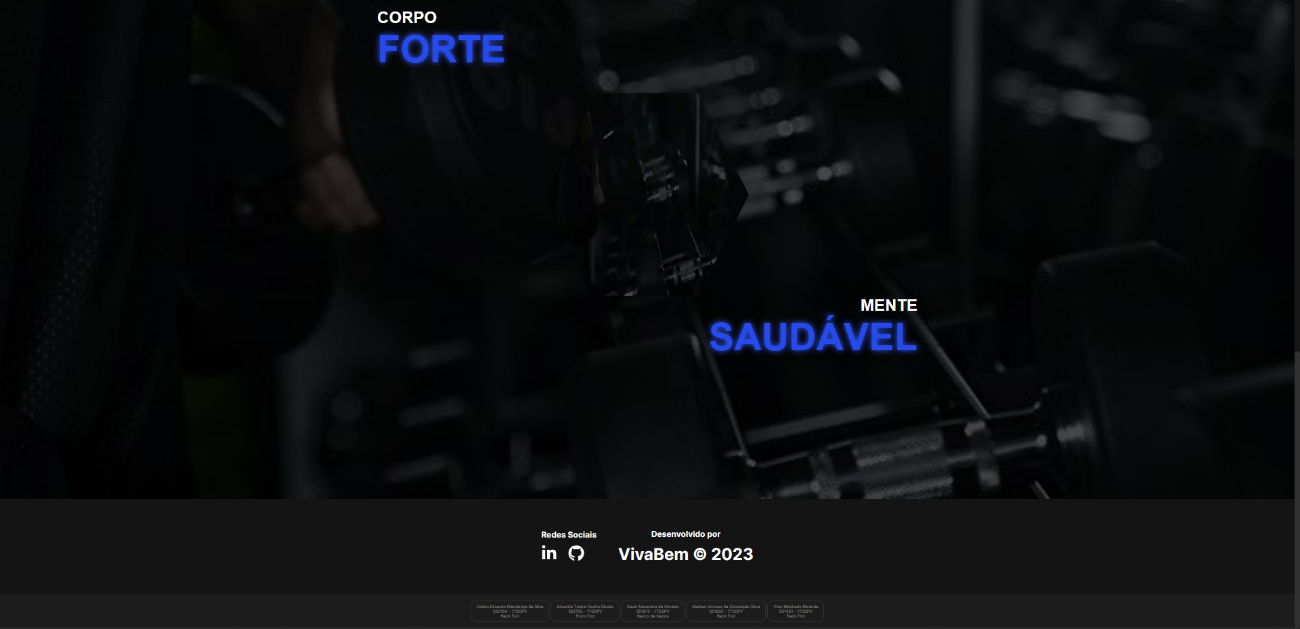
**Home Page**

**Homem sentado no chão

Descrição gerada automaticamente com confiança baixa**

**Uma imagem contendo no interior, mesa, homem, computador

Descrição gerada automaticamente**

****

**Log-in e Cadastro**

**Tela de computador com fundo preto

Descrição gerada automaticamente**

**Tela de computador com fundo escuro

Descrição gerada automaticamente**

**Tela de computador com texto preto sobre fundo escuro

Descrição gerada automaticamente**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

**Interface gráfica do usuário, Site

Descrição gerada automaticamente**

**Interface gráfica do usuário, Site

Descrição gerada automaticamente**

**Tela de computador com texto preto sobre fundo escuro

Descrição gerada automaticamente**

**Tela do Perfil**

**Tela de computador com jogo

Descrição gerada automaticamente**

**Tela dos Treinos**

**Tela de celular com aplicativo aberto

Descrição gerada automaticamente**

**Tela do Plano de Dieta**

**Tela de celular com aplicativo aberto

Descrição gerada automaticamente**

**Tela de computador com fundo preto

Descrição gerada automaticamente**

**Tela de computador com fundo preto

Descrição gerada automaticamente**

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

**4 COMO FAZER CHAMADAS A API**

Como requisição da matéria de Domain Drive Design, foi desenvolvido uma API RESTful na linguagem Java usando o modelo de arquitetura JAX-RS. A aplicação foi pensada para que possa integrar o Front-End com o banco de dados, tornando possível a interação com os dados armazenados.

A API desenvolvida possui 8 endpoints que possibilitam operações de CRUD com as entidades do banco de dados, ou seja, cada um desses endpoints possuem os métodos GET (Um para trazer todos os dados da respectiva tabela e outro para trazer um específico por meio do seu ID), POST, PUT e DELETE. Alguns ainda possuem algumas outras formas de realizar o método GET.

Os endpoints podem ser usados para interagir com as seguintes entidades do banco de dados:

|  |  |  |
| --- | --- | --- |
| Entidade | Endpoint | Sobre |
| Biotipo | “/biotipo” | Recupera e manipula informações sobre os biotipos. |
| Cliente | “/cliente” | Recupera e manipula informações dos clientes usuários da aplicação. |
| Dieta | “/dieta” | Recupera e manipula informações sobre as dietas. |
| Exercício | “/exercicio” | Recupera e manipula informações sobre os exercícios. |
| Medida | “/medida” | Recupera e manipula informações das medidas dos usuários. |
| Objetivo | “/objetivo” | Recupera e manipula informações dos objetivos dos usuários. |
| Tipo de Treino | “/tipo-treino” | Recupera e manipula informações sobre os tipos de treinos. |
| Treino | “/treino” | Recupera e manipula informações sobre os treinos. |

**Exemplos de como fazer a chamada**

Com o programa rodando, basta usar a seguinte o URL para acessar o endpoint desejado:

http://localhost:8080/api/{endpoint}

**Recuperando todos os dados**

Para recuperar todos os dados presentes na tabela do banco basta informar o endpoint ao qual você deseja acessar as informações e realizar uma requisição GET. Você receberá como retorno um JSON com todos os dados.

URL para a chamada:

http://localhost:8080/api/exercicios

Retorno:

[

    {

        "id": 1,

        "nome": "Supino Máquina",

        "tipoTreino": {

            "id": 1,

...

        },

...

    },

    {

        "id": 2,

        "nome": "Barra Fixa",

        "tipoTreino": {

            "id": 1,

            ...

        },

...

    },

    {

        "id": 3,

        "nome": "Elevação Lateral",

        "tipoTreino": {

            "id": 1,

            ...

        },

...

    },

...

]

Status HTTP:

|  |  |
| --- | --- |
| 200 OK | Retornado com sucesso. |
| 404 NOT FOUND | Nenhum dado encontrado. |

**Recuperando um dado específico**

Além do endpoint, adicione também o ID da informação que você deseja recuperar na URL e realize uma requisição GET, assim você terá como retorno um único objeto JSON do dado com respectivo ID.

URL para a chamada:

http://localhost:8080/api/exercicio/2

Retorno:

[

    {

        "id": 2,

        "nome": "Barra Fixa",

        "repeticoes": 15,

        "series": 3,

        "tempoDescanso": 2,

        "tipoTreino": {

            "id": 1,

            ...

        }

    }

]

Status HTTP:

|  |  |
| --- | --- |
| 200 OK | Retornado com sucesso. |
| 404 NOT FOUND | Nenhum dado encontrado. |

As entidades que armazenam dados dos Tipos de Treinos e de Exercícios possuem mais uma forma de recuperá-los. É possível recuperar os Tipos de Treino por meio do ID de um Treino e recuperar os Exercícios por meio do ID dos Tipos de Treinos.

URLs para as chamadas:

http://localhost:8080/api/exercicio/tipo-treino/2

http://localhost:8080/api/tipo-treino/treino/2

**Adicionando uma nova informação**

Para adicionar uma nova informação você precisará enviar, por meio de uma requisição POST, um objeto JSON com os atributos da entidade, respeitando as regras e restrições do banco de dados, no endpoint respectivo a tabela. Não é necessário passar o ID, pois o banco se encarregará de colocá-lo.

URL para a chamada:

http://localhost:8080/api/exercicio

Exemplo de objeto que deve ser enviado:

[

    {

        "nome": "Nome de Exemplo",

        "repeticoes": 10,

        "series": 3,

        "tempoDescanso": 2,

        "tipoTreino": {

            "id": 1

        }

    }

]

Status HTTP:

|  |  |
| --- | --- |
| 201 CREATED | Criado com sucesso. |
| 400 BAD REQUEST | Envio incorreto. |

**Atualizando uma informação**

Caso precise atualizar os dados é possível usar uma requisição PUT e usar a URL com endpoint e o ID da informação que você deseja alterar. Você também precisará enviar um objeto JSON com todos os atributos da entidade, alterando aqueles que devem ser atualizados. Não é necessário enviar o ID, pois ele será recuperado da URL pela própria API.

URL para a chamada:

http://localhost:8080/api/exercicio/5

Exemplo de objeto que deve ser enviado:

[

    {

        "nome": "Nome Alterado",

        "repeticoes": 20,

        "series": 3,

        "tempoDescanso": 2,

        "tipoTreino": {

            "id": 2

        }

    }

]

Status HTTP:

|  |  |
| --- | --- |
| 200 OK | Atualizado com sucesso. |
| 404 NOT FOUND | Nenhum dado encontrado. |

**Deletando uma informação**

O método DELETE necessita apenas de informar o endpoint e o ID do dado que se deseja deletar na URL e a informação será apagada no banco de dados.

URL para chamada:

http://localhost:8080/api/exercicio/3

Status HTTP:

|  |  |
| --- | --- |
| 200 OK | Deletado com sucesso. |
| 404 NOT FOUND | Nenhum dado encontrado. |

**Verificação de Login**

A API ainda possui um endpoint para fazer verificação do login do cliente. Essa verificação de login pode ser acessada nessa URL:

|  |
| --- |
| http://localhost:8080/api/cliente/login |

Os dados podem ser recuperados de um formulário no Front-End e enviados em um JSON com a seguinte estrutura:

{

"email ": "exemplo@email.com",

       "senha ": "ExemploDeSenha",

}

O e-mail será verificado no banco de dados, no caso de existir a senha será verificada. Se tudo estiver de acordo, será retornado um JSON com os dados desse cliente ou colaborador.

Status HTTP:

|  |  |
| --- | --- |
| 202 ACCEPTED | Retornado com sucesso. |
| 404 NOT FOUND | Email não encontrado. |
| 406 NOT ACCEPTABLE | Senha incorreta. |

**5 MODELO DO BANCO DE DADOS**

Uma imagem contendo Diagrama

Descrição gerada automaticamente

**6 UML DAS CLASSES**

**Diagrama

Descrição gerada automaticamente**