**CENTRO PAULA SOUZA**

**ETEC ZONA LESTE**

**Desenvolvimento de Sistemas**

**Henrique Cesar Fonseca Alves**

**Kaua Alves Seppe**

**Rafael Moriya**

**Luciano**

**Alexandre**

**AUTOMATIZAÇÃO NA CRIAÇÃO E ATRIBUIÇÃO DE TREINOS PARA CLIENTES DE ACADEMIA**

**São Paulo**

**2022**

**Alexandre Vieira dos Santos**

**Henrique Cesar Fonseca Alves**

**Kaua Alves Seppe**

**Luciano Rodrigues Campos Vitor**

**Rafael Moriya Oliveira**

**AUTOMATIZAÇÃO NA CRIAÇÃO E ATRIBUIÇÃO DE TREINOS PARA CLIENTES DE ACADEMIA**

Trabalho de Conclusão de Curso apresentado ao Curso Técnico em Desenvolvimento de Sistemas da Etec Zona Leste, orientado pelo Prof. Tamara, como requisito parcial para obtenção do título de técnico em Desenvolvimento de Sistemas.

**São Paulo**

**2022**

**DEDICATÓRIA**

Não há exemplo maior de dedicação do que o da nossa família. À nossa querida família, que tanto admiramos e amamos, dedicamos o resultado deste trabalho tão árduo realizado ao longo deste percurso. Dedicamos, este trabalho aos nossos docentes que sempre estiveram ao nosso lado disponibilizando-nos tempo e auxílio, dedicamos, também aos amigos e familiares.

**AGRADECIMENTO**

Agradecemos ‘a todos que nos auxiliaram no desenvolvimento deste trabalho de conclusão de curso, onde fez-se imperiosa a ajuda, a compreensão e a dedicação de todos os participantes. Durante o desenvolvimento tivemos momentos de crises e dificuldades. Todavia, conseguimos superar os entraves com êxito e afinco. Ao que concerne aos mestres, transpareço nosso respeito, afeto e admiração. Por todo apoio e ensinamento, somos muito gratos. Aos nossos amigos somos gratos e exultamos pela paciência e confiança. Por derradeiro e não menos importante agradecemos as nossas famílias, por todo amor, apoio, carinho, benevolências e bem-aventurança.

“Nenhum cidadão tem o direito de ser um amador em matéria de treinamento físico. Que desgraça é para o homem envelhecer sem nunca ver a beleza e a força do que o seu corpo é capaz”

SÓCRATES

**RESUMO**

Em primeira análise, neste trabalho buscamos automatizar o sistema em uma academia específica com atualizações no treino, cadastro de clientes, facilitar o contato entre professores e alunos e amenizar as dúvidas recorrentes aos exercícios com vídeos de explicativos e de demonstração. De maneira que supra todas às necessidades do usuário, seja com requisitos funcionais ou requisitos não funcionais do sistema.

**PALAVRAS-CHAVE:**

**ABSTRACT**

**SUMÁRIO**

# 1 INTRODUÇÃO

Em 11 de março de 2020, a Organização Mundial da Saúde (OMS) declarou que o mundo enfrentava uma nova pandemia (Wikipedia, 2020), desta vez causada pelo SARS-CoV-2 (*severe acute respiratory syndrome coronavirus* 2), o qual é o agente etiológico da COVID-19, uma doença infecciosa de caráter respiratório, causando uma calamidade em todo o globo. À fim de conter a disseminação do vírus, o governo brasileiro investiu como medida preventiva a quarentena, isolamento social, métodos que incentivavam a população a ficar em casa em conjunto a campanhas de vacinação, evitando assim, a propagação em massa da doença. Esse período de quarentena prejudicou ativamente a economia do país, o portal de notícias g1 (2020) compilou dados relacionados à pandemia, onde haveria tido uma queda de 9,7% do PIB brasileiro, afetando tanto grandes empresas quanto pequenas, sendo as academias um dos setores mais atingidos pela pandemia (Agência Brasil, 2021), alcançando um patamar de 52% abaixo do que seria normal para o mês de maio de 2021. Em face disso, torna –se valido ressaltar que todos os métodos preventivos utilizados foram de extrema importância à queda dos índices dos casos de COVID-19 e a crescente procura as academias no país. Nesse sentido, foi adotado um novo plano de retoma onde se pretendia maximizar a presença de alunos e docentes, em segurança (Academia Militar, 2020c). Possibilitando, então, o acesso aos aparelhos e o retorno das atividades em grupo, como, zumba, pilates, natação, artes marciais etc. Mas, utilizando-se das medidas preventivas. Desse modo, o aumento do interesse nesse setor deve-se muito pela disseminação dos benefícios das atividades físicas proporcionam à saúde. A IHRSA (*International Health Racquet & Sportsclub Association*), afirma que atualmente, existem 35 mil unidades oficiais que colocam o Brasil como o segundo país com maior número de academias no mundo, atrás apenas dos EUA com mais de 40 mil. Nessa perspectiva, as academias são de extrema importância para a economia brasileira, principalmente, de pequeno porte que movimenta a maior parte dos recursos e lucros no país. Superando grandes franquias como SMART FIT e RED FIT.

**1.1 PROBLEMA**

Sabe-se que a maior parcela das pessoas tem grande dificuldade de manter-se em forma com uma rotina de exercícios, condicionar e balancear de maneira correta uma alimentação saudável. Isso porque grande parte da sociedade atual tem costumes que prejudicam a saúde acarretando doenças como: sedentarismo, triglicérides, colesterol, obesidade, gordura no fígado, infarto etc. Com a aparição da pandemia devido ao vírus (COVID-19) houve um aumento no sedentarismo e a diminuição à procura de academias no país.

**Hipótese**

Para solucionar as entraves pós-pandemia e a grande procura as academias, tornou-se necessária à automação dos treinos, facilitar a comunicação entre aluno e professor, com o fito de mitigar as dificuldades ao manusear os aparelhos dentro das academias.

# 2 REFERENCIAL TEÓRICO

# 2.1 HTML

O HTML (Linguagem de Marcação de Hipertexto), originou-se em 1991, pelo então considerado pai da internet Tim Berners-Lee, na Suíça. De acordo com o autor CJ Costa (2007). A linguagem HTML é construída por textos e códigos especiais chamadas marcas ou *tags* (comandos de linguagem), o HTML foi projetado inicialmente para interligar situações de pesquisas próximas e compartilhar arquivos com facilidade. Atualmente é a linguagem de marcação mais utilizada para formatar páginas web, sites e aplicativos mobiles em conjunto com linguagens de programação como o PHP (*Hypertext Preprocessor*) e o JavaScript. As páginas web possuem um sistema eficaz de edição e modelagem, necessitando de uma IDE de modelagem como Visual Studio Code, Not Pad++, etc.

Para criar qualquer projeto em HTML é necessário seguir algumas regras, tais regras são as *tags*, as *tags* são responsáveis pela criação de objetos identificáveis como: listas, links, parágrafos, textos e formulários.

Para FFlatschart (2011). Um hipertexto do HTML é um documento ou sistema formado por grupos de dados como (textos, imagens, vídeo, áudio) conectados por links. São exemplos de elos de associação a tag <href> utilizada para conectar uma página a outra, a tag imagem que salva uma imagem à página web. Estes recursos estão presentes em todos os sites e em toda parte visual de sistemas e até mesmo televisores e celulares.

O HTML está presente no dia a dia de todas as pessoas, contudo a maior parte das pessoas não sabem da existência desta tecnologia fundamental.

(exemplo)

(aplicaçao)

# 2.2 CSS

Cascading Styly Sheets (CSS) em português significa folha de estilo em cascata. Surgiu no final de 1996 para revolucionar a web com seu o designer, formatação e leveza, visto que esse era o trabalho do HTML de marcar e modelar. Contudo, de acordo com o autor no JR Lewis (2010) no livro CSS Aavançado PG18. (“**citação grande: arrumo depois” Não demorou muito para as pessoas perceberem que essa abordagem não funcionaria no futuro porque ela era fundamentalmente limitante. Em vez de tentar disponibilizar um documento monolítico aos navegadores web, fazia muito mais sentido dar aos navegadores os blocos de construção do conteúdo em si e, então, deixar que o navegador cuidasse de juntar tudo. Esse princípio é conhecido como separação de interesses (separation of concerns).**

Para JR Lewis (????) a inovação feita com o surgimento do CSS inovou e facilitou a forma de trabalhar na web.

Para se ter uma modelagem faz-se mister ter todo o escopo do projeto CSS dentro da *tag* (*style*), as funcionalidades do CSS são definidas em *tags* assim como no html. Pois o CSS puxa a *tag* utilizada no HTML, como as *tags* <div>, <h1>, <p> e modela como no exemplo de código abaixo:



O CSS está presente em todos as páginas web e aplicativos mobile, deixando-o mais leve e bonito visualmente.

(Mais exemplos)

(aplicações)

# 2.3 BOOTSTRAP

# 2.8 JavaScript

O JavaScript (frequentemente abreviado como JS) é uma linguagem de programação voltada para o desenvolvimento web, criada por Brendan Eich tendo como base a linguagem Java em 1995 para a empresa Netscape. No decorrer do tempo de acordo com Flanagan (2004) A grande maioria dos navegadores e sites web comportam interpretadores JavaScript tornando-a a linguagem de programação mais onipotente da história. O JavaScript está presente na tríade de tecnologias web que todo programador web deve conhecer sendo eles o HTML, CSS e o JavaScript.

O JavaScript é uma linguagem de alto nível. Segundo Flanagan (2004) uma linguagem de alto nível é interpretada, dinâmica e não tipada conveniente para programação orientada a objetos. Ou seja, está mais próxima a linguagem do programador a linguagem do computador o que a torna uma linguagem mais intuitiva e fácil de se trabalhar. A linguagem JS é muito utilizada hodiernamente

para

# 2.4 PHP

O *Hypertext Preprocessor* (PHP), antes chamado de *Personal Home Page Tools,* é uma linguagem de programação originada no outono de 1994 por Rasmus Lerdorf. Segundo DALL’OGLIO (2015), no início a linguagem era um conjunto de scripts escritos em linguagem C com foco na criação de páginas dinâmicas, utilizada por Rasmus para acompanhar o acesso ao seu currículo na internet. Após um tempo, a linguagem passou a se tornar conhecida e atrair novos usuários. Motivado por isso, Rasmus adicionou diversos mecanismos, sendo um deles a interação com bancos de dados. No ano de 1995, foi liberado o código-fonte do PHP, possibilitando que outros desenvolvedores se juntassem ao projeto para aprimorar ainda mais a linguagem. Nesse período, o PHP foi chamado de FI (*Forms Interpreter),* sofrendo diversas reescritas de código e conquistando ainda mais adeptos (DALL’OGLIO, 2015).

De acordo com CONVERSE e PARK (2003), PHP se tornou uma linguagem para a criação de scripts para a *Web* sendo utilizada juntamente ao *HyperText Markup Language* (HTML), possuindo compatibilidade com diversos servidores *Web* importantes (principalmente o Apache). Uma das funcionalidades do PHP é a possibilidade de incorporar fragmentos de códigos em páginas padrão HTML, sendo interpretado à medida que as páginas são apresentadas aos usuários. CONVERSE e PARK (2003) retratam o PHP como uma linguagem que facilita a conexão das páginas *Web* com o banco de dados.

O PHP é um módulo oficial de servidor HTTP Apache, um dos maiores servidores *Web* do mercado. De acordo com o levantamento de dados feita em agosto de 2021 pelo site W3Techs, empresa responsável por monitorar o uso de tecnologias na *Web,* a linguagem de programação PHP é utilizada por cerca de 79,1% dos sites disponíveis na internet. Devido o PHP ser uma linguagem de código livre, diversos mecanismos de facilitação de manuseio do código foram criados, alguns deles chamados de *frameworks,* sendo Laravel o mais conhecido e utilizado.

(exemplo)

(aplicação)

# 2.5 LARAVEL

Lançado em 2011, o *framework* Laravel é apontado por Neto (2020) como versátil e robusto. A equipe desenvolvedora fez uso de múltiplas ferramentas e soluções já consolidadas e utilizadas na comunidade PHP, juntando-as dentro de uma estrutura que aproveita da melhor forma os artifícios fornecidos por essas ferramentas.

Considerado um *framework* volátil, sendo útil tanto para projetos simples quanto de grande porte. Um de seus atributos mais característicos é a praticidade que ele oferece para o desenvolvimento de funcionalidades comuns, através da ferramenta artisan, executada por meio de linhas de comando.

A obra Laravel para ninjas (GABARDO, 2017) retrata o Laravel como um *framework* intuitivo, claro e com diversas funcionalidades. Sendo um *framework PHP Model View Control (*MVC*),* sob a norma de Programação Orientada a Objetos (PÓO).

Nos tempos atuais *frameworks* não são novidade no meio de desenvolvimento de *softwares.* Grande parte das linguagens possuem *frameworks* dos mais variados tipos. De acordo com McCool (2012), um *framework* assimila um conjunto de classes ou funções implantadas em uma linguagem de programação específica, facilitando o processo de desenvolver um *software*. Um *framework*, então, é um conceito, uma estrutura que serve como início para o desenvolvimento de um projeto específico.

Os *frameworks* possuem diversos propósitos, como diminuir ou deixar de escrever um código-fonte por meio da reutilização de métodos, classes e funções, forçar a utilização de um padrão de design de projetos, como por exemplo o MVC, que oferece diversos meios de uso comum no meio de desenvolvimento *Web* como roteamento, criação de formulários, sessões e outros artifícios já feitos ou semiprontos, evitando assim, a necessidade de reescrever tais funções em cada projeto.

Segundo Gabardo (2017), o MVC é um padrão de design de projetos que realiza a separação do HTML da lógica e das regras de negócio. Para o manuseio correto do framework, Gabardo (2017) afirma que é necessário compreender como o MVC funciona, pois é a base de um projeto Laravel.

A figura X representa uma ilustração do modelo MVC.

**Figura 1 – Fluxo de dados padrão do Modelo MVC**

Diagrama

Descrição gerada automaticamente

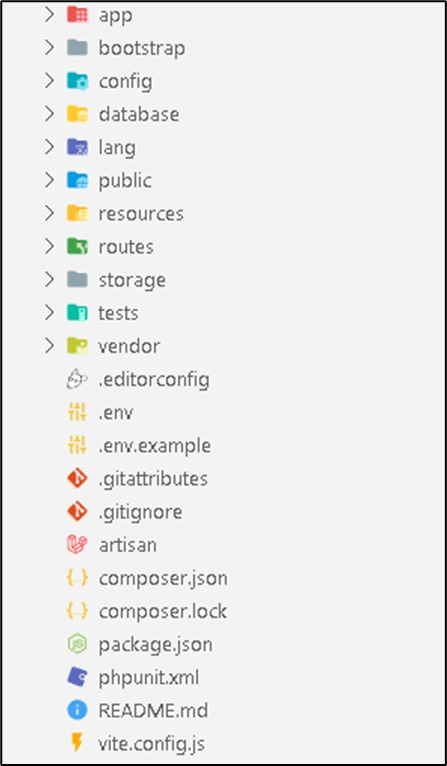
Fonte: Gabardo (2017)

Conforme apresentado na Figura X, o fluxo de dados do programa tem início através de uma requisição HTTP. Uma rota mapeia e envia essa requisição para um *controller*, que é responsável pelo processamento da requisição, sendo capaz de aplicar e enviar dados a um *model*, para então apresentar uma *view,* e possuindo a capacidade de encaminhar dados para a *view*, ou apenas exibir um arquivo HTML estático. (GABARDO, 2017) aponta que o modelo MVC, possui as seguintes camadas:

* *Routes*: mapeiam as requisições HTTP e as enviam para o *controller* adequado. Apesar de não ser uma camada própria do modelo MVC, é o procedimento mais comum para abordar requisições HTTP. Certos *frameworks* realizam a liberação de acesso direto aos métodos dos *controllers*, já outros obrigam a formação de rotas para cada método.
* *Model*: são representações de objetos, tem como utilidade modelar os objetos que serão utilizados na aplicação. São responsáveis por definir quais são e de que tipo são os campos de um objeto.
* *Views*: arquivos da camada de exibição da aplicação, é o que apresenta os resultados das requisições para o usuário. Podendo ser HTML, PHP ou fazer uso de algum tipo de *template*.
* *Controllers*: responsáveis por coletar as requisições através das rotas, processam as requisições de acordo com a lógica de negócios empregada, consumindo e enviando dados para os *models*, também carregando arquivos de visualização.

Ao criar um projeto com o Laravel, os seguintes arquivos e pastas mostrados na figura X fazem parte da pasta raiz do projeto.

**Figura 2 - Pastas Padrões de um Projeto Laravel 9**



Fonte: Autoria própria, 2022

Como aponta Gabardo (2017), cada pasta e arquivo tem sua funcionalidade para promover e facilitar a criação da aplicação em questão. Esses arquivos e pastas tem os seguintes papéis e funcionalidades no projeto apontadas por Gabardo (2017):

• app: é justamente a pasta da aplicação. Nesta pasta se encontra os models e a subpasta Http/Controllers onde serão os controladores serão construídos.

• bootstrap: pasta que possui scripts que carregam, inicializam a aplicação e retornam a solicitação da aplicação.

• config: pasta onde estão os arquivos de configuração, como as configurações de conexão com banco de dados e outros.

• database: esta pasta armazena classes específicas das interações com banco de dados sendo elas: migrations, factories e seeds.

• public: nesta pasta que a alocação de arquivos de acesso público como imagens, arquivos estáticos é realizada. Esta pasta possui o arquivo index.php, arquivo de configuração do servidor Apache.htaccess, e diversos outros arquivos de configuração

• routes: pasta aonde as rotas serão definidas, seja para uma API, aplicação Web etc.

• storage: pasta destinada a manter arquivos gerados pelo framework, como logs, sessões, caches etc.

• resources: armazena as views do projeto, arquivos para compilação de CSS, arquivos de linguagem e outros recursos.

• tests: pasta com a finalidade de armazenar arquivos de teste unitários.

• vendor: possui arquivos do framework propriamente dito. Não havendo necessidade de alterar a pasta.

• .env: arquivo responsável por possuir as variáveis da aplicação incluindo a chave criada no momento da criação do projeto.

• .env-example: arquivo de exemplo para configuração das variáveis da aplicação.

• .gitattributes: arquivo que possui orientações para o servidor de versionamento.

• .gitignore: arquivo responsável por impedir outros arquivos de serem adicionados a repositórios GIT, como o arquivo detentor da chave da aplicação e a pasta do próprio framework.

• artisan: arquivo encarregado de carregar recursos do framework automaticamente.

• composer.json: responsável por determinar as dependências do projeto. Este arquivo possui as orientações que informam aos gerenciadores de pacotes quais são os pacotes e as bibliotecas das quais o projeto é dependente.

• composer.lock: arquivo criado automaticamente pelo gerenciador de dependências Composer. Após a instalação das dependências, o Composer guarda a lista de versões exatas dos pacotes instalados nesse arquivo, travando o projeto a essas versões específicas

• package.json: este arquivo possui as dependências para fazer uso do Gulp (ferramenta de automação de tarefas em JavaScript) com o Laravel.

• phpunit.xml: arquivo responsável pela configuração de testes unitários com a linguagem PHP.

• README.md: arquivos .md são arquivos de marcação que possibilita a utilização de tags e marcações comumente adotados para gerar documentação. Este arquivo possui informações sobre o framework, links, para a documentação, informações sobre a licença do framework etc.

Como exemplo, as figuras a seguir apresentam a criação de um formulário de cadastro e listagem de endereços através de *routes, models*, *views*, *controllers* e *migrations*, utilizando bootstrap para a estilização.

**Figura X – *Migration* dos Dados de Endereço, Código PHP**

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

**Figura X – *Model* dos Dados de Endereço**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

Fonte: Autoria própria, 2022.

**Figura X – Cadastro de Endereço, Código HTML**

*Linha do tempo

Descrição gerada automaticamente com confiança média*

Fonte: Autoria própria, 2022.

**Figura X – *Controller* doFormulário, Código PHP**

**Texto

Descrição gerada automaticamente**

Fonte: Autoria própria, 2022.

**Figura X – Listagem de Endereços Cadastrados, Código HTML**

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

**Figura X – Listagem de Endereços Cadastrados, Código HTML (2)**

Texto

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

**Figura X – *Controller* Endereços Cadastrados, Código PHP**

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Fonte: Autoria própria, 2022.

**Figura X – Cadastro de Endereço, Página *Web*.**

**Uma imagem contendo Tabela

Descrição gerada automaticamente**

Fonte: Autoria própria, 2022.

**Figura X – Listagem de Endereços Cadastrados, Página *Web*.**

**Interface gráfica do usuário, Aplicativo, Word

Descrição gerada automaticamente**

Fonte: Autoria própria, 2022.

# 2 .6 Banco De Dados

# 2.6.1 Abordagem Relacional

# 2.6.2 Normalização

# 2.6.3 Dicionário de Dados

# 2.7 UML

# 2.7.1 Levantamento de Requisitos

# 2.7.2 Diagrama de Casos de Uso

# 2.7.3 Diagrama de Atividades

# 2.7.4 Diagrama de Classes

# 2.7.5 Diagrama de Sequência

# 

# 3 DESENVOLVIMENTO

# 4 CONCLUSÃO

REFERENCIAS

GABARDO, Ademir C. Laravel para ninjas. Novatec Editora, 2017.

MCCOOL, Shawn. **Laravel starter**. Packt Publishing, 2012.

CONVERSE, Tim; PARK, Joyce. **PHP: a bíblia**. Gulf Professional Publishing, 2003.

DALL’OGLIO, Pablo. **PHP Programando com Orientação a Objetos** 3ª Edição. Novatec Editora, 2015.

DOUGLAS, Michael; MARABESI, Matheus. **Aprendendo Laravel: O framework PHP dos artesãos da web**. Novatec Editora, 2017.

CUCINOTTA, Domenico; VANELLI, Maurizio. WHO Declares COVID-19 a Pandemic. **WHO Declares COVID-19 a Pandemic**, Parma, ano 2020, v. Vol. 91 No. 1, p. 157-158, 19 mar. 2020. DOI https://doi.org/10.23750/abm.v91i1.9397. Disponível em: <https://pubmed.ncbi.nlm.nih.gov/32191675/> . Acesso em: 25 ago. 2022.

NETO, Jaime. **Laravel - Escolhendo Um Framework Php**. Rio de Janeiro: Clube de Autores, 2020, 138 p.

Organização Mundial de Saúde declara pandemia do novo Coronavírus: Mudança de classificação obriga países a tomarem atitudes preventivas. **UNA-SUS**, Brasília, 11 mar. 2020. Saúde. Disponível em: <https://www.unasus.gov.br/noticia/organizacao-mundial-de-saude-declara-pandemia-de-coronavirus#:~:text=Organização%20Mundial%20de%20Saúde%20declara%20pandemia%20do%20novo%20Coronavírus,-Mudança%20de%20classificação&text=Tedros%20Adhanom%2C%20diretor%20geral%20da,Sars-Cov-2>). Acesso em: 04 set. 2022.

AVARENGA, Darlan e SILVEIRA, Daniel. PIB tem tombo recorde de 9,7% no 2 º trimestre e Brasil entra de novo em recessão. **g1,** Rio de Janeiro, 01 set. 2020. Economia. Disponível em: <https://g1.globo.com/economia/noticia/2020/09/01/pib-tem-queda-recorde-de-97percent-no-2o-trimestre-e-brasil-entra-de-novo-em-recessao.ghtml>. Acesso em: 25 ago. 2022.

GUERRA, Antônio. Sebrae: setor de academias é um dos mais afetados pela pandemia: Faturamento está 52% abaixo do normal. **Agência Brasil**, Brasília, 08 jul. 2021.Economia.

Disponível em:<https://agenciabrasil.ebc.com.br/economia/noticia/2021-07/sebrae-setor-de-academias-e-um-dos-mais-afetados-pela-pandemia>. Acesso em: 25 ago. 2022.