CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

ETEC ZONA LESTE

Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas – AMS

Danillo Fonseca Cruz

Gustavo Grorossi Muniz

TOMANAGE: Gerenciador de Tarefas

São Paulo

2022

Danillo Fonseca Cruz

Gustavo Grorossi Muniz

TOMANAGE: Gerenciador de Tarefas

Trabalho de Conclusão de Curso apresentado ao Curso do Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas AMS da Etec Zona Leste, orientado pela Professora Vilma Cardoso dos Santos como requisito final para obtenção do título de Técnico em Desenvolvimento de Sistemas.

São Paulo

2022

DEDICATÓRIA

Dedico este trabalho à minha família, que muito me apoiou e aos desenvolvedores pelo esforço dedicado ao mesmo. Por fim à IBM, pela oportunidade de crescimento oferecida.

AGRADECIMENTOS

A Deus, pela vida nos dada, e por nos ajudar a ultrapassar todos os obstáculos encontrados ao longo do curso.

Agradecemos aos nossos pais e irmãos, por terem nos dado força e sustentabilidade financeira no início do curso para chegar a este momento. Aproveitamos também a oportunidade de agradecer todo o apoio dado durante todos os anos das nossas vidas e todo aporte que nos deram em casa, além de auxiliar com sabedoria e com muita paciência.

Agradecemos aos nossos professores Ediney Ciasi Barreto, Jeferson Roberto de Lima, Rogério Bezerra Costa e Vilma Cardoso dos Santos por todo tempo que lhes coube, por conselhos passados, todas as correções e ensinamentos que nos permitiram apresentar um melhor desempenho e além da oportunidade de nos orientar na conclusão deste trabalho.

EPÍGRAFE

“A verdadeira motivação vem de realização, desenvolvimento pessoal, satisfação no trabalho e reconhecimento.”

FREDERICK HERZBERG

RESUMO

O presente projeto se propõe a produzir uma aplicação institucional para o Curso do Ensino Médio com Habilitação Profissional de Técnico em Desenvolvimento de Sistemas – AMS do Centro Estadual de Educação Tecnológica Paula Souza. Para tanto, além da produção da aplicação em si, contempla também a produção de um site para a divulgação dos desenvolvedores e a promoção da aplicação criada. A aplicação produzida para o curso de Análise e Desenvolvimento de Sistemas tem o intuito de gerenciar a rotina diária de usuários de forma acessível e intuitiva de uma maneira simples.

Palavras-Chave: Projeto. Aplicação institucional. Desenvolvimento de Sistemas.

ABSTRACT

The present project proposes to produce an institutional application for the High School Course with Professional Qualification of Technician in Systems Development - AMS of the State Center for Technological Education Paula Souza. Therefore, in addition to the production of the application itself, it also includes the production of a website for the dissemination of developers and the promotion of the created application. The application produced for the Systems Analysis and Development course is intended to manage the daily routine of users in an accessible and intuitive way in a simple way.

Keywords: Project. Institutional application. Systems development.

LISTA DE ILUSTRAÇÕES

Figura 1 - Pirâmide de Aprendizagem 15
Figura 2 - Técnica Pomodoro 18
Figura 3 - Ferramenta Trello 21
Figura 4 - Trello Premium 22
Figura 5 - Ferramenta Notion 23
Figura 6 - Prototipação Figma 25
Figura 7 - Documento Básico HTML 28
Figura 8 - Login HTML 29
Figura 9 - Exemplo Regras CSS 32
Figura 10 - Exemplo Regras CSS 32
Figura 11 - Exemplo Página com BootStrap 34
Figura 12 - Principais Tecnologias 35
Figura 13 - Exemplo de documento em JavaScript 37
Figura 14 - Execução JavaScript 37
Figura 15 - Conjunto de scripts em PHP 40
Figura 16 - Conexão com a Base de Dados 41
Figura 17 - Conexões dos Bancos de Dados 45
Figura 18 - Exemplo Diagrama de Caso de Uso 49
Figura 19 - Exemplo Diagrama de Atividades 51
Figura 20 - Simbologia Diagrama de Atividades 52
Figura 21 - Exemplo Diagrama de Classes 54
Figura 22 - Exemplo de um Processo de Hereditariedade 56
Figura 23 - Funcionamento das Associações 56
Figura 24 - Exemplo Diagrama de Sequência 58
Figura 25 - Tela de Login 59
Figura 26 - Tela de Cadastro 60
Figura 27 - Tela de Dicas 61
Figura 28 - Tela de Ferramentas 62
Figura 29 - Tela de Criação de Quizzes 63
Figura 30 - Tela de Resultados 64
Figura 31 - Website no Computador 65
Figura 32 - Website no Smartphone 65

LISTA DE SIGLAS E ABREVIATURAS

Banco de Dados (BD)

Cascading Style Sheets (CSS)

Geography Markup Language (GML)

HyperText Markup Language (HTML)

HyperText Preprocessor (PHP)

Interface de Programação de Aplicações (API)

Relational Database Management System (RDBMS)

Sistemas Gerenciadores de Banco de Dados (SGBD)

Standard General Markup Language (SGLM)

Structure Query Language (SQL)

Unified Modeling Language (UML)

User Interface (UI)

*World Wide Web (*WWW)

SUMÁRIO

1 INTRODUÇÃO 12

2 REFERENCIAL TEÓRICO 13

2.1 Estudo Sobre uso do Tempo 13

2.1.1 Pirâmide de Glasser 14

2.1.2 Técnica Pomodoro 16

2.1.3 O Poder do Hábito 19

2.2 Softwares Gerenciadores de Tarefas 20

2.2.1 Trello 20

2.2.2 Notion 23

2.3 Figma 24

2.3.1 Funcionamento 26

2.3.2 Compartilhamento 26

2.4 HTML 26

2.5 CSS 30

2.6 BootStrap 33

2.7 JavaScript 35

2.8 PHP 39

2.9 React Native 42

2.10 Firebase 43

2.11 UML 46

2.11.1 Levantamento de Análise 48

2.11.2 Diagrama de Caso de Uso 49

2.11.3 Diagrama de Atividades 50

2.11.4 Diagrama de Classes 53

2.11.5 Diagrama de Sequência 57

3 DESENVOLVIMENTO 59

3.1 Aplicação Mobile 59

3.2 Website.....65

4 CONCLUSÃO 67

REFERÊNCIAS.....68

1 INTRODUÇÃO

Atualmente muitas pessoas têm dificuldade de gerenciar seu tempo em relação aos estudos, o que é um pilar essencial para o desenvolvimento pessoal e acadêmico. As pessoas estão completamente sobrecarregadas com afazeres diários, então ter uma boa administração sobre seu tempo se torna algo necessário. Utilizando um sistema de gerenciamento de tarefas acadêmicas, o usuário irá adquirir uma maneira prática e eficiente para estudar durante longos períodos.

Aplicando a técnica da Pirâmide de Glasser, os níveis de retenção de aprendizagem de um indivíduo vão ser evidenciados de acordo com a forma que ele se relaciona com o conteúdo estudado. A Técnica Pomodoro se baseia na ideia de que dividindo o fluxo de trabalho em blocos de concentração intensa, o indivíduo consegue melhorar a agilidade do cérebro e estimular o foco. O objetivo do estudo é explicar por que fazemos o que fazemos. Sendo assim, O Poder do Hábito é uma reflexão que aborda como a capacidade de fazermos coisas de forma subconsciente, com um menor esforço cerebral, tem a tendência de criar padrões em nossas vidas.

Ao inserir as tecnologias na aplicação, proporcionou-se uma nova visão para os usuários, trazendo consigo a facilidade de utilização e praticidade para a aquisição de conhecimento e administração de tempo. Dessa forma, os avanços da tecnologia trouxeram consigo novos sistemas de gerenciamento e suporte para quem busca o aperfeiçoamento nos estudos. O *software* possui inúmeras vantagens em relação aos outros, como por exemplo, *quizzes* que o próprio usuário pode criar com a finalidade de reforçar ainda mais o conteúdo estudado.

Visando a criação de uma aplicação *Mobile* e um *Website* para divulgação e suporte de usuários, foram empregadas as tecnologias: *HyperText Markup Language* (HTML) para dar o formato do *Website; Cascading Style Sheets* (CSS) para fazer a estilização; *BootStrap* que será usado como *Framework; JavaScript* e *Hypertext Preprocessor* (PHP) para adicionar interatividade ao *Website; React Native* que será utilizado para o desenvolvimento da aplicação *Mobile; Firebase* que será utilizado como banco de dados, fazendo assim a autenticação e armazenamento das informações do usuário.

2 REFERENCIAL TEÓRICO

Este capítulo contém o embasamento teórico das tecnologias que serão utilizadas para a elaboração do projeto de pesquisa do aplicativo gerenciador de tarefas toManage.

2.1 Estudo Sobre uso do Tempo

Diz Moura (2013), por muitas vezes, qualquer distração já pode ser uma boa desculpa para pararmos de realizar a tarefa. Com o excesso de afazeres diários, existe a sensação de que não há tempo suficiente para realizar todas as tarefas, observa-se que o tempo está cada vez mais curto. Atualmente fica evidente a relevância de uma boa administração do tempo na escola e na vida pessoal. Ouvimos muito que precisamos ter uma alta produtividade para não adiar tarefas importantes, entre outras dicas, contudo, quando estamos sem foco, qualquer distração é uma excelente desculpa para procrastinarmos.

Segundo Globo (2012), falta de tempo é o principal causador de estresse entre os brasileiros. O desenvolvimento pessoal é um processo, focando no progresso dos hábitos diários. Estou sem tempo. Está é uma frase que costumamos dizer, é indiscutível que o dia contém exatamente 24 horas, o que podemos fazer é gerenciar o nosso tempo.

A partir do contexto apresentado, observa-se que o mau gerenciamento diário tem se tornado um óbice cada vez maior. É indiscutível que cada vez mais, alunos tem seus rendimentos menores, isso deixa claro a importância do gerenciamento na rotina. Como observado, existe uma dificuldade em gerenciar o tempo e ter mais produtividade em tarefas pendentes. Essa sobrecarga dá origem à sensação de que não há tempo suficiente, acarretando quadros de ansiedade. O estudo sobre o uso do tempo é essencial para o melhor aproveitamento desse recurso na vida das pessoas.

De acordo com Chaves (1998), administrar o Tempo é Planejar a Vida. Diante disso, surgiu o seguinte questionamento: como garantir o aumento da produtividade com uma boa gestão do tempo?

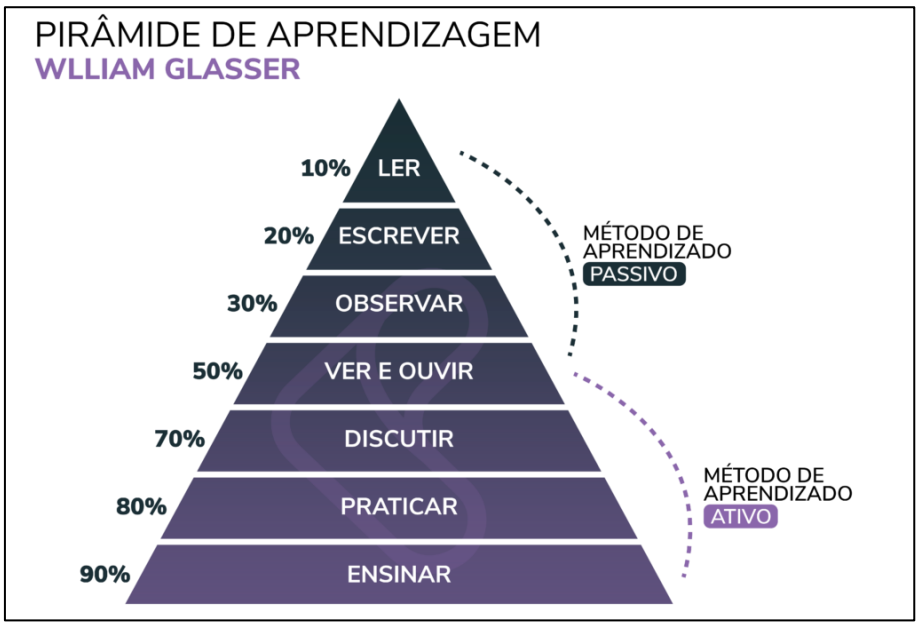
2.1.1 Pirâmide de Glasser

De acordo com Miranda (2016), a competência engloba um conjunto de habilidades, atitudes e conhecimentos necessários aos profissionais para desenvolverem de forma eficiente suas ações. O psiquiatra americano William Glasser aplicou sua teoria da pirâmide de aprendizagem para a educação. A Teoria explica que o professor é um guia para o aluno e não um chefe. Glasser explica que não se deve trabalhar apenas com a memorização, porque a maioria dos alunos simplesmente esquecem os conceitos após a aula. Segundo Glasser, “A boa educação é aquela em que o professor pede para que seus alunos pensem e se dediquem a promover um diálogo para promover a compreensão e o crescimento dos estudantes.” Em vez disso, o psiquiatra sugere que os alunos aprendam efetivamente com você, fazendo. Além disso, Glasser também explica o grau de aprendizagem de acordo com a técnica utilizada.

> As pirâmides e cones de aprendizagem são recursos costumeiramente utilizados no meio escolar para defender que as práticas pedagógicas que favorecem a participação mais ativas dos alunos são as mais eficientes. Na maioria das vezes, essas pirâmides e cones são associadas aos estudos de William Glasser e Edgar Dale. (DA SILVA & MUZARDO, 2018, p.169)

A pirâmide de aprendizagem mostra o grau em que compreendemos de acordo com o método, podemos observar esse procedimento na figura 1 a seguir.

Figura 1 - Pirâmide de Aprendizagem



Fonte: Keeps, 2022.

De acordo com a pirâmide, o processo de aprendizagem do ser humano ocorre da seguinte forma:

- **Ler:** Glasser acredita que, quando você tem contato com algum tema pela primeira vez durante a leitura, seu cérebro é capaz de reter apenas 10% daquele assunto. Por isso, geralmente a tendência é que acabemos lendo e revisando trechos que julgamos mais importantes para auxiliar na retenção do conteúdo.
- **Ouvir:** Quando Apenas ouvimos o conteúdo, o cérebro atua assimilando apenas 20% do que foi explanado. Isso ocorre porque, muitas vezes, você acaba fazendo outras coisas enquanto absorve o som do ambiente.
- **Ver:** Se ao estudar determinado conteúdo você dispuser de imagens que associem o conteúdo, conseguirá, de acordo com a pirâmide de aprendizagem, guardar em torno de 30% do que foi repassado.
- **Ver e Ouvir:** No caso de ver e ouvir, o conteúdo consegue ser fixado em até 50%. Às vezes o aluno lê e não consegue assimilar tão bem, mas após assistir uma aula, já com a bagagem literária, é capaz de assimilar tudo com maior facilidade.

- **Debater:** Ao debatermos o conteúdo com outra pessoa ou com um grupo de pessoas, conseguimos reter até 70% do aprendizado. Esse debate pode acontecer presencialmente ou, em caso de estudos EAD, através de fóruns e chats, proporcionando um ciclo de análises críticas acerca dos temas abordados.
- **Ensinar:** Quando você ensina, a capacidade de assimilação do conteúdo, segundo a teoria de Glasser, é de 95%. Você pode dar uma aula a si mesmo, pode gravar um vídeo repassando o assunto, pode explicá-lo ao seu colega, tudo de modo a aumentar seu nível de compreensão a respeito daquele tema.

Segundo Carotenuto e Pereira (2020), metodologias ativas têm sido centro de muitos debates educacionais, mas elas só fazem sentido quando são primeiramente internalizadas pelo docente, quebrando seu próprio paradigma de ensino. A teoria de William Glasser vem amplamente sendo divulgada e aplicada por professores e pedagogos mundo afora, é uma das muitas teorias de educação existentes, e uma das mais interessantes, pois ela demonstra que ensinar, é aprender!

2.1.2 Técnica Pomodoro

A técnica Pomodoro foi inicialmente desenvolvida no final dos anos 80 pelo italiano Francesco Cirillo. Ele buscava uma maneira eficiente de aumentar significativamente sua produtividade nos estudos durante o período em que cursava seus primeiros anos de universidade. Francesco utilizou um timer de cozinha para fazer a organização de suas tarefas acadêmicas. O timer tinha o formato de um tomate, que significa pomodoro em italiano, ele girava durante 25 minutos e emitia um efeito sonoro forte quando o tempo acabava. Após a pausa do timer, Francesco concentrava-se em suas tarefas sem interrupções, mantendo assim seu foco em determinado assunto. A técnica Pomodoro foi oficialmente divulgada em 1992.

> Quando você se sente perdido, um Pomodoro pode ser dedicado à exploração final de definir as suas prioridades em linha reta e traçar um novo plano. Se suas ideias são claras, mas algo está faltando – talvez determinação, talvez um pouco de coragem – encerre o Pomodoro e comece a trabalhar sobre ele, sem esperar pelo tempo. (DE OLIVEIRA, 2020, p.34)

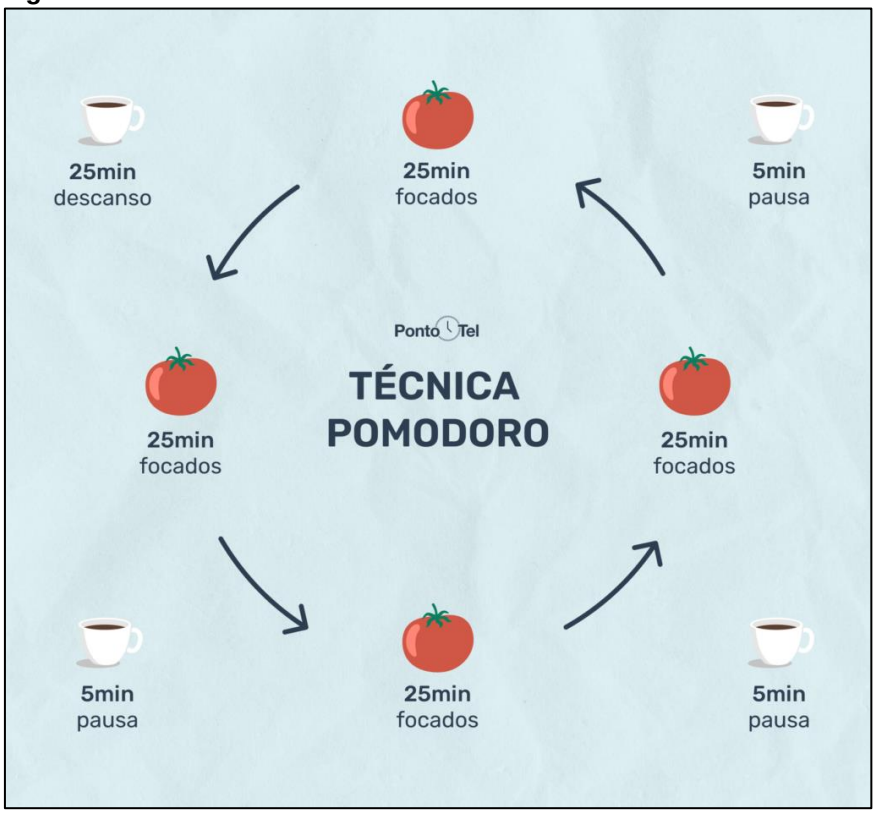
A técnica funciona da seguinte maneira: ela se baseia na ideia de que fazendo a divisão do nosso fluxo de trabalho em blocos de concentração intensa, podemos melhorar a agilidade em que o cérebro trabalha e estimulamos nossa concentração, ou seja, melhoramos a nossa gestão de tempo e ficamos mais competentes. Para colocar a técnica em prática você irá precisar de:

- Uma lista de tarefas (“to-do-list”);
- Lápis e borracha;
- Um timer para fazer a contagem regressiva do tempo.

Para começar, você deve criar uma lista de tarefas que tem de ser cumpridas naquele determinado dia. Após isso, é necessário fazer a divisão do seu tempo em pequenos períodos de 25 minutos, que são os “pomodoros” e assim estudar e/ou trabalhar sem interrupção alguma em suas tarefas durante esses períodos. Após o timer tocar, marque com um “X” as tarefas que já foram concluídas ou anote o a porcentagem concluída, para em seguida terminar, faça um intervalo de 5 minutos. Nesse momento é importante que você faça outras coisas como: tomar um café, relaxar e ir ao banheiro, por exemplo.

Segundo Carmo (2020), durante o período de estudo a criança e a pessoa que está acompanhando devem focar na tarefa e manter distantes distratores externos. A cada quatro ciclos, é necessário fazer uma pausa entre 15 e 30 minutos, ou seja, faça uma pausa maior para descansar. Na técnica Pomodoro esses intervalos entre os pomodoros são de extrema importância para “oxigenar o seu cérebro” e aumentar a sua agilidade de raciocínio. Os objetivos da técnica Pomodoro são: o aumento do foco e aumento da concentração em tarefas que tem de ser cumpridas, evitando assim o desperdício de tempo, o que não é uma boa alternativa para quem busca ter resultados e avanços. Podemos observar o funcionamento da Técnica Pomodoro na figura 2 a seguir.

Figura 2 - Técnica Pomodoro



Fonte: PontoTel, 2022.

Ao decorrer do tempo você irá perceber alguns benefícios que a técnica Pomodoro trouxe, como:

- Quanto tempo você leva para concluir determinada tarefa;
- Quais distrações geralmente você tem;
- As suas principais interrupções no trabalho.

A técnica Pomodoro pode ser utilizada para a organização de tarefas de todos os tipos, diversas pessoas a usam como método de estudo para concursos. Além disso, ela também é pode ser usada para a organização de atividades domésticas, projetos de trabalho, entre outras atividades. Uma pessoa pode através de análises e testes, alcança sua própria produtividade e criar um ritmo de estudo mais adequado e que atende melhor as suas expectativas em relação aos estudos.

Segundo Mark (2021), a jornada de trabalho tem mais a ver com minutos seguidos trabalhados do que com horas trabalhadas. A professora cronometrou rotinas de alguns funcionários e chegou à conclusão de que em média, eles são capazes de

realizar de três a cinco minutos de trabalho sem interrupções ou distrações. Para conseguir administrar com mais responsabilidade o seu tempo, você pode implementar medidas práticas e que sejam eficazes, como por exemplo, desativar as notificações do celular e deixá-lo na mochila, por exemplo. A técnica Pomodoro é uma ótima escolha para se organizar durante os estudos e obter resultados incríveis a longo prazo, a produtividade é uma habilidade essencial, tanto durante os estudos quanto no trabalho.

2.1.3 O Poder do Hábito

Segundo Murphy (2021), a cura pessoal será sempre a prova mais convincente dos poderes de nosso subconsciente. O Poder do Hábito, é uma obra escrita por Charles Duhigg e foi publicada em fevereiro de 2012, ela ganhou o prêmio Pulitizer por ser uma obra de grande sucesso na época. O livro discursa o porquê fazemos determinadas coisas na vida, seja nos negócios ou na vida pessoal. A obra revela como os hábitos determinam as maneiras como consumimos e vivemos, além de conseguirmos compreendê-los e mudá-los potencialmente.

> Entendemos como fazer as pessoas comerem menos, se exercitarem mais, trabalharem de forma mais eficiente e levarem vidas mais saudáveis. Transformar um hábito não é necessariamente fácil nem rápido. Nem sempre é simples. Mas é possível. (DUHIGG, 2012, p.16)

O livro é dividido em três partes: a primeira trata dos hábitos e das rotinas das pessoas; a segunda aborda os hábitos das empresas e das grandes organizações; a terceira trata de uma forma mais apurada os hábitos sociais. O autor apresenta em sua obra diversas pesquisas bem interessantes como forma de argumento e concretização de suas afirmações. O escritor também buscou estudar mais afundo sobre a mente humana, encontrando assim, padrões, que mesmo após sofrer com problemas de saúde, algumas pessoas mantinham hábitos os quais não eram possíveis serem explicados por meio da lógica.

Segundo James (1892), toda a nossa vida, na medida em que tem forma definida, não é nada além de uma massa de hábitos. Para entender esse padrão Duhigg mostrou

um estudo realizado no final do século XX pelo *Massachusetts Institute of Technology* (MIT), no qual foi descoberto pelos pesquisadores um ciclo neurológico em todos os hábitos, denominado como o *Loop do Hábito.* Esse ciclo era constituído por três partes, sendo elas: a deixa, a qual representa o estímulo que manda o cérebro entrar no modo automático; a rotina, que é a forma de executarmos a deixa; a recompensa, a qual auxilia o cérebro a saber se vale a pena memorizar o hábito. Sendo assim, foi observado que com a ausência da recompensa, é gerada uma certa ansiedade em relação ao resultado, podendo esse fato ser remetido às pessoas que fumam, bebem ou jogam determinado jogo compulsivamente sem ter o controle de quando devem parar.

2.2 Softwares Gerenciadores de Tarefas

Para o desenvolvimento do sistema proposto, foram analisadas algumas aplicações disponíveis no mercado, cujas funcionalidades se assemelham com as almejadas para esse projeto, tais como gerenciamento de tarefas, gerenciamento de tempo através de técnicas de foco e rastreador de hábitos. Dessa forma, nesta seção serão apresentadas as ferramentas apuradas pela equipe, expondo também pontos positivos e negativos encontrados nas aplicações, além de fazer uma análise comparativa final.

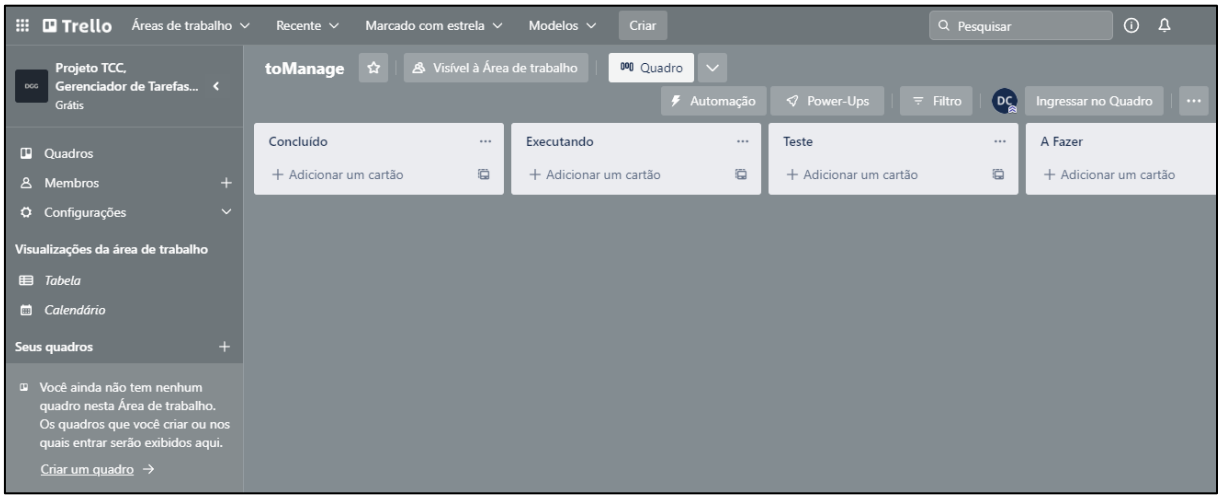
2.2.1 Trello

O Trello é um aplicativo criado pela Atlassian, ele faz parte de um conjunto de ferramentas (juntamente com o Jira, o Confluence e o Team Central), basicamente eles fazem a gestão dos mais diversos projetos para que possa levar qualquer projeto até o fim, ou seja, até a sua conclusão. A ferramenta possibilita ao time de gerenciamento um fluxo de trabalho ou monitoramento de tarefas, nele você encontra algumas funções como: acrescentar arquivos, fazer listas do que é necessário fazer para a conclusão do projeto e até mesmo automatizar o funcionamento.

Segundo Damascena (2019), o uso de tecnologias educacionais digitais como ferramenta didática possibilita o desenvolvimento de um processo dinâmico, interativo e contextualizado com a realidade vivenciada. Para utilizar o Trello é preciso criar uma conta e consequentemente começar a usar a ferramenta. A forma mais simples de

começar a utilizá-lo é criando algumas listas, elas lhe darão um caminho para a continuação do projeto. Em seguida você poderá criar quadros com cartões, mas antes de tudo pense e se organize em relação ao fluxo de trabalho do seu quadro. Na figura 3 a seguir, podemos observar um quadro com listas inseridas, elas indicam o que já foi concluído, utilizando a ferramenta Trello.

Figura 3 - Ferramenta Trello



Fonte: Autoria Própria, 2022.

Entretanto, o Trello não é apenas um quadro branco analógico, nele existem ferramentas que o deixam mais eficaz e completo em relação a outros *softwares.* Imagens, textos e comentários podem ser usados sem moderação em cada tarefa, isso facilita muito a consulta de dados do usuário. Essas informações são armazenadas e agrupadas em cartões Trello para futuramente serem usadas ao seu favor.

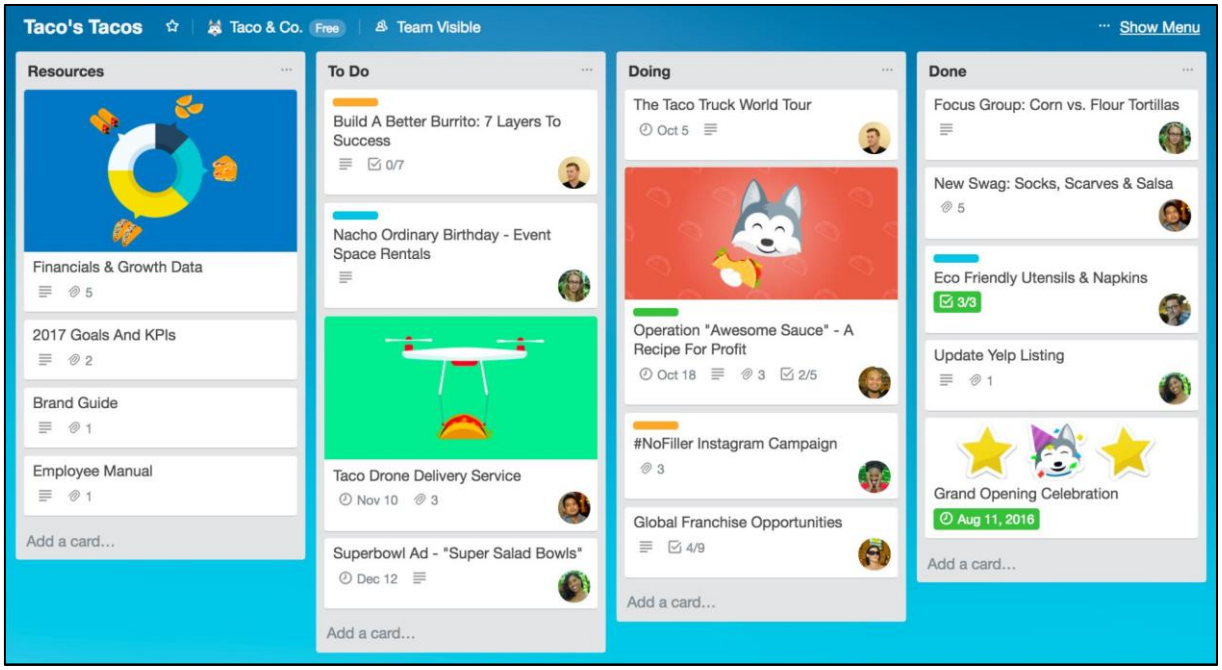
> Nessa perspectiva, como alternativa de suporte para realização de trabalhos de forma colaborativa no projeto de extensão, sem a necessidade de numerosas reuniões para acompanhar o andamento das tarefas, foi utilizada a ferramenta Trello®, criado para auxiliar equipes na organização, monitoramento e controle de projetos, permitindo a compreensão do todo, a comunicação transparente e visual. (MELLO & SOUZA, 2018)

Também existem os Power ups Trello, eles são vigorosas integrações de aplicativos que fazem a conexão de toda desordem de outros ativos próximos a você. É possível

também integrar os seus quadros criados com a ferramenta Figma, isso facilita bastante o processo de desenvolvimento do projeto. Além dessas vantagens, o usuário também pode fazer comunicações com o Slack ou Google Drive de forma gratuita para armazenar os dados do projeto em desenvolvimento.

A plataforma Trello foi criada com o intuito de fortalecer todas as equipes envolvidas no projeto, isso inclui as equipes de marketing, recursos humanos, desenvolvimento, vendas e design. A ferramenta pode ser utilizada no próprio escritório do usuário, seja de forma remota ou híbrida, isso é uma vantagem significativa em relação a outras ferramentas. O desenvolvimento de forma híbrida por exemplo funciona da seguinte maneira: o usuário pode colocar notas adesivas em seu escritório e só depois passar tudo a limpo para o Trello, esse processo facilita muito a gestão do projeto, pois nem sempre o usuário estará com um computador ou um notebook em mãos. Na figura 4 a seguir, podemos observar a ferramenta Trello sendo utilizada na versão Premium, o que possibilita ao usuário mais funções de organização.

Figura 4 - Trello Premium



Fonte: Vida Organizada, 2022.

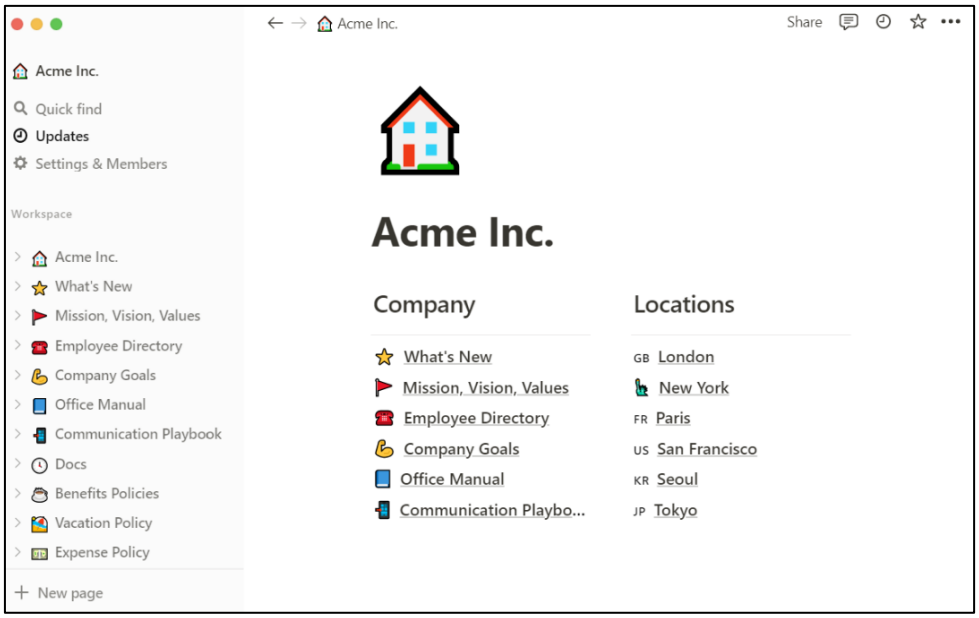
Segundo Trello (2019), a possibilidade de inclusão de anexos, prazos finais, fotos, comentários, checklist, calendário que permite a visualização dos cartões, etiquetas simbolizando prioridades, conclusão das atividades e outros. O Trello foi criado para

todos, ou seja, para todas as faixas etárias e para as mais diversas equipes, seu uso pode se basear de um calendário editorial ou até mesmo para realizar reuniões mais organizadas. É uma ferramenta gratuita, porém ele pode ser estendido para uma versão Premium que tem mais funções que permitem que as equipes vejam os projetos em uma linha do tempo, como um calendário, em tabelas ou integrados a mapas, todos essas funções ajudam muito projetos maiores a se desenvolverem.

2.2.2 Notion

Segundo Notion (2022), ferramenta colaborativa, com o intuito de integrar diversas funcionalidades em um só lugar. Apresenta foco na organização de dados e notas, além de disponibilizar diversas formas de configuração e exibição desses dados. Podemos observar na figura 5 a seguir, a interface de usuário da ferramenta Notion.

Figura 5 - Ferramenta Notion



Fonte: Notion, 2022.

Segundo Pink (2013), a proposta foi a construção de uma ferramenta de *codesign*, pautada na fotoelicitação. O Notion é uma ferramenta de organização de tarefas, mas não apenas isso! Utilizando-o é possível fazer, por exemplo, a gestão de projetos, compartilhar tarefas com outras pessoas e acompanhar o progresso de cada atividade. O *software* é destinado tanto para a organização pessoal quanto para a profissional, oferecendo diversas funcionalidades para quem o escolhe. A ferramenta

pode ser acessada pelo computador via *browser*, instalado no *desktop* ou até mesmo usado como uma aplicação no *smartphone* do usuário.

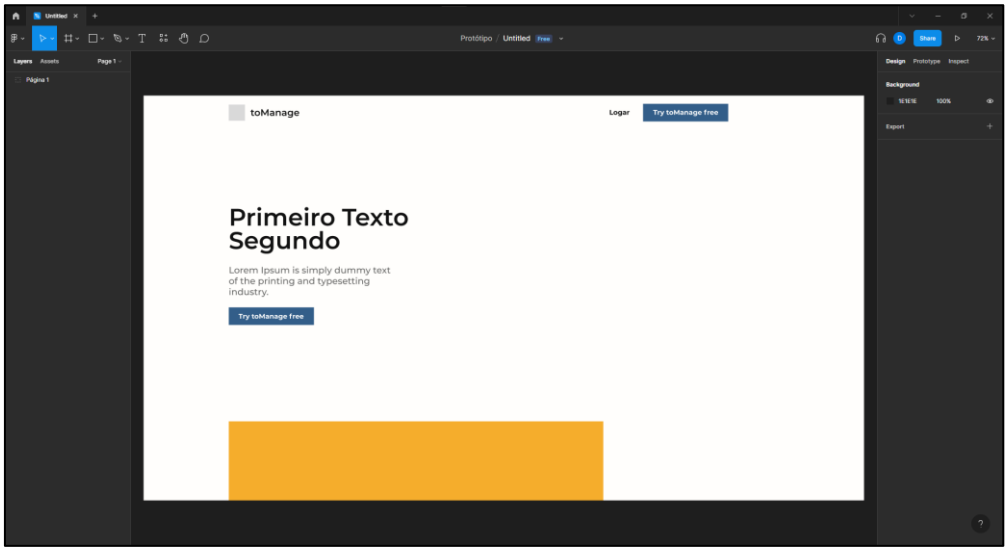
> (...) o ensino e a aprendizagem ultrapassaram os muros da sala de aula tradicional, permitindo que as formas desse ensinar e aprender possibilitassem maior flexibilidade de tempo e espaço, sem depender da presença física do professor e do aluno. Nessa direção, temos o apoio da Educação a Distância (EaD), também conhecida como *e-learning* e *on-line learning.* (MOORE & KEARSLEY, 2013)

O *software* tem diversas funcionalidades, como por exemplo: agenda, calendário, lista de tarefas, tabelas, notas, espaço para armazenamento de conteúdo, criação de texto e *checklists.* O usuário pode utilizar a ferramenta Notion para distribuir tarefas para sua equipe, criando uma página dentro de seu espaço de trabalho , construindo *cards* e colunas que permitem o avanço das atividades.

2.3 Figma

Segundo Oliveira (2022), a construção de produtos digitais é permeada por diversos fatores relacionados diretamente aos seus usuários. O Figma é um editor de gráficos de vetor e prototipagem de projetos de design baseado principalmente no navegador *web*. O Figma é uma das melhores ferramentas de *UI Design*, ou seja, de interface do usuário. Muitos projetos nascem de novas ideias ou como uma melhoria e inovação para o que já existe. No entanto, você há de concordar que criar algo do início demanda tempo e dedicação, e é indiscutível a necessidade do Figma no desenvolvimento de projetos, pois com ele, criaremos uma prototipação do *site*, facilitando o desenvolvimento do toManage. É possível observar uma prototipação do *site* toManage na figura 6 a seguir.

Figura 6 - Prototipação Figma



Fonte: Autoria Própria, 2022.

O desenvolvimento da parte visual é importantíssimo, um protótipo permite com que os desenvolvedores analisem detalhadamente todas as partes e funcionalidades de um objeto.

> No desenvolvimento de *software*, a fase de especificação dos requisitos é uma das mais importantes, visto que erros não detectados nesta são propagados para as fases posteriores. Quanto mais avançado estiver o desenvolvimento, mais caro custa reparar um erro introduzido nas fases iniciais, pois isto envolve reconsiderar vários estágios do desenvolvimento. (RANGEL & SALUM, 2003)

De acordo com Heberle (2017), cada vez mais as empresas de *software* recebem novas demandas de requisitos de sistemas, sendo que há uma demanda cada vez maior. Para o desenvolvimento de um *site*, exige muito tempo e a facilidade que o Figma traz é essencial para a entrega final do projeto. E por fim, o uso dessa ferramenta será necessário para economizar tempo com futuros erros, pois já teremos uma base da funcionalidade de cada objeto.

2.3.1 Funcionamento

A tendência da maioria das ferramentas online é facilitar o uso para usuários mais leigos, o Figma ajuda o usuário a se programar e fazer um escopo do projeto que futuramente será programado. A plataforma é de fácil entendimento, primeiramente, é necessário criar uma conta e escolher um plano para começar a utilizar a ferramenta. Logo em seguida, você criará um projeto dentro do *software web*.

Após criar o projeto, a ferramenta irá apresentar diversos *templates* que poderão ser usados para começar o seu trabalho. O usuário poderá escolher um modelo específico para a criação do arquivo baseado no iPhone, ou simplesmente criar uma página *web* em branco utilizando os sistemas operacionais Windows ou macOS. Após fazer a escolha o usuário poderá iniciar o design do seu *site*, as ferramentas de edição já estarão disponíveis para utilizá-las e aplicá-las em seu projeto. A ferramenta Figma é sincronizada com a nuvem a todo momento, ou seja, o usuário não precisa se preocupar em salvar o seu projeto toda hora, todas as alterações feitas são salvas automaticamente.

2.3.2 Compartilhamento

O Figma tem um sistema de compartilhamento muito interessante e essencial para o desenvolvimento do projeto, pois, com ele, nós podemos compartilhar e editar um mesmo arquivo, facilitando o desenvolvimento do protótipo. A ferramenta conta com uma espécie de grupo, adicionando pessoas ao projeto. O usuário pode expandir seu uso para trabalho corporativo, clicando no botão do canto superior direito da tela para compartilhar seu arquivo com outra pessoa. Ao fazer o acesso do *link*, será solicitado ao seu colega um cadastro no Figma, ou simplesmente fazer login com sua conta local para fazer a edição do projeto compartilhado.

2.4 HTML

Segundo Carril (2012), o HTML é uma linguagem de marcação, utilizada para formatar páginas *Web* com texto e informações separadamente. A principal linguagem para o desenvolvimento de *sites* atualmente é *HyperText Markup Language* (HTML), a linguagem é o componente base da *web*, sendo uma das principais para o desenvolvimento *Front-End* e a principal encontrada na *internet*. A Linguagem de

Marcação de Hipertexto é uma das principais tecnologias utilizadas na *World Wide Web*, utilizada para a codificação de hipertexto.

- *Front-End*: Ele está muito relacionado com a interface gráfica do projeto. Ou seja, é onde se desenvolve a parte que irá interagir diretamente com o usuário. Por exemplo, um formulário de registro.
- *Back-End*: Já esse está relacionado a parte de trás do *site*, é tudo o que envolve como o projeto irá funcionar. Ou seja, aquilo que o usuário não vê, por exemplo, o banco de dados.

As atuais linguagens de marcação têm como ancestral comum a *Standard Generalized Markup Language* (SGLM) que, por sua vez, evoluiu da *Generalized Markup Language* (GML) desenvolvida pela IBM.

> As linguagens de marcação remontam à época em que os profissionais revisores de texto marcavam indicações nos documentos, de maneira que estes fossem facilmente reconhecidos dentro do texto final que seria entregue para o leitor. (FLATSCHART, 2011)

O HTML é uma linguagem de marcação, e não uma linguagem de programação. Pois com ele, não é possível criar funcionalidade dinâmicas, trata-se de uma linguagem de marcação. A linguagem HTML é utilizada para a criação e desenvolvimento de milhares de páginas, pois se trata de uma linguagem versátil e ampla, sendo capaz de criar diversas estruturas com simples linhas de código. Para o desenvolvimento de um documento HTML, nós utilizamos as tags, elas são caracterizadas inicialmente em dois tipos, em *block* e *inline*.

Além de podermos criar simples títulos, parágrafos e *links*, é possível desenvolver códigos mais complexos, como demonstrado na figura 7 a seguir. No exemplo, mostramos um formulário de *login* feito em um documento HTML.

Figura 7 - Documento Básico HTML

```html
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="UTF-8">
    <link rel="stylesheet" href="css/style.css">
    <title>Login</title>
  </head>
  <body>

    <section class="area-login">
        <div class="login">
            <form method="POST">
                <input type="text" name="nome" placeholder="Email" autofocus>
                <input type="password" name="senha" placeholder="Senha">
                <input type="submit" value="Entrar">
            </form>

            <p>Ainda não tem uma conta? <a href="#">Criar Conta</a></p>

        </div>
    </section>

  </body>
</html>
```

Fonte: Autoria Própria, 2022.

Agora que já demos um exemplo de um documento HTML, vamos entender mais a fundo a funcionalidade acerca das *tags* fundamentais abordadas acima e suas funções de acordo com o W3Schools (2022). Para que futuramente, até você consiga criar uma página HTML.

- ***Doctype*:** Todos os documentos devem começar com uma <!DOCTYPE> declaração. A declaração não é uma *tag* HTML. É uma “informação” para o navegador sobre qual tipo de documento esperar.
- ***Html*:** A *tag* <html> representa a raiz de um documento HTML, é o contêiner para todos os outros elementos HTML.
- ***Head*:** É o elemento contêiner para metadados (dados sobre dados) e é colocado entre a *tag* <html> e a *tag* <body>.
- ***Meta*:** A *tag* define metadados sobre um documento HTML, os metadados são uados por navegadores, mecanismos de pesquisa e outros serviços da *web*.
- ***Link*:** Define o relacionamento entre o documento atual e um recurso externo, a *tag* é usada com mais frequência para vincular a folha de estilo externas ou para adicionar um *favicon* ao *site*.
- ***Title*:** Define o título do documento. É mostrado na barra de título do navegador ou na aba da página. É uma *tag* obrigatória em um documento HTML.
- ***Body*:** A *tag* define o corpo do documento, todo o conteúdo da página.

As linguagens de marcação, como o HTML, aceitam formatação semântica, permitindo de que maneira a informação será mostrada para o usuário final. A maioria

dos elementos possui uma *tag* de abertura e de fechamento, uma *tag* é uma palavra específica, definida em HTML, envolta por sinais de menor que < e maior que >. De um modo geral, as *tags* aparecem em pares, uma indicando o início e a outra indicando o fim da marcação. Entretanto alguns não precisam fechar a *tag* para funcionar. Esse é o caso dos elementos vazios, como usados nas linhas 4 e 5.

Uma das partes mais importantes do documento HTML é o cabeçalho, que contém as informações mais importantes do *site*, como é o caso do título, que é inserido dentro da tag <title>. Os títulos são mostrados na barra de título do navegador e na área em que aparecem as páginas já visitadas. A *tag* <title> é fundamental para os mecanismos de busca da *internet*. O corpo do documento é constituído por imagens, texto, *links* e formulários mostrado pela janela do *browser*. No código, o corpo é tudo aquilo inserido entre as *tags* <body> e </body>.

De acordo com Costa (2007), não é necessário um compilador para obter um resultado dos códigos HTML, tudo o que é preciso é um editor de textos comum e de um *browser* que se encarregará de interpretar o código. A diferença básica que fará com que o *browser* identifique que se trata de uma página HTML é a extensão do ficheiro. Ou seja, o documento HTML deve ser gravado com extensão ".htm" ou ".html" enquanto um ficheiro de texto é gravador com ".txt". Na figura 8 a seguir, podemos observar um exemplo dos códigos desenvolvidos, e como as *tags* agem, criando um formulário simples.

Figura 8 - Login HTML

Fonte: Autoria Própria, 2022.

Um par de *tags*, com abertura e fechamento, também pode ser denominado elemento, diz Silva (2015) que explica mais algumas *tags* do HTML:

- ***Br*:** elemento vazio, ou seja, somente com *tag* de abertura, destinado a quebra de linha.
- ***P*:** do tipo nível de bloco, é um elemento destinado a marcar parágrafos.

- ***B e strong*****:** elemento do tipo *inline*, causam efeito de renderização em negrito. A diferença é que o elemento b dá o aspecto visual negrito e o elemento *strong* dá uma forte ênfase.
- ***Div*****:** é um elemento destinado a criar um container geral para os outros elementos, também do tipo nível de bloco.
- ***Form*****:** elemento que informa a criação do formulário.
- ***Fieldset*****:** cria uma borda ao redor do formulário.
- ***Legend*****:** legenda do *fieldset*, um título entre a borda.
- ***Input*****:** controla o que será adicionado no formulário, cria campos para entrada de dados, juntamente com o atributo “type”.
- ***Label*****:** legenda para um item do formulário, identificação dos campos.
- ***Select*****:** *tag* para criação de uma lista com opções predefinidas.
- ***Option*****:** *tag* interna da *select*, cria os itens da lista de opções.

A criação de um formulário é essencial para uma página HTML, de fato o formulário é um dos principais pontos de interação entre um usuário e um *site*. Eles permitem que o usuário envie dados para a página. Em um *site*, a maioria dos pontos em que se coloca dados são formulários, como um login. O HTML é a principal linguagem de marcação encontrada na internet, cada página tem diversos elementos que cria a estrutura dos conteúdos de uma página. Se engana quem pensa que o HTML é complicado, pois trata-se de uma linguagem amigável para iniciantes, que possui bastante suporte e é principalmente usada para páginas estáticas. Quando falamos "HTML" estamos referindo-se a tecnologia como um todo, porém o HTML já está na sua quinta versão, o HTML5 é realmente apenas uma atualização para o padrão antigo. Hoje, o HTML é a principal linguagem de marcação para desenvolvimento de *sites*, sendo de fácil aprendizado, portanto será essencial o uso no desenvolvimento.

2.5 CSS

Também conhecido como *Cascading Style Sheets* (CSS), traduzindo para o português, Folha de Estilo em Cascatas desenvolvido pelo W3C (World Wide Web Consortium). É um mecanismo usado para adicionar estilização a um documento HTML e é utilizado na formatação da aparência das páginas. O CSS é facilmente

utilizado com as linguagens de Marcação de Hiper Textos, porém um documento HTML passa a ser utilizado somente como elemento para estruturar as páginas, pois não foi projetado para ter *tags* que ajudariam a formatação da página.

Segundo Miletto e Bertagnolli (2014), antes de o CSS ser utilizado para a formatação de páginas *Web,* era necessário utilizar *tags* HTML específicas. Assim, quando tinha a alteração na aparência de um elemento, todas as páginas deviam ser reformuladas e em alguns casos até mesmo refeitas. Porém com o CSS, é possível definir um único local de formatação e foi criada com o intuito de auxiliar o HTML, criando uma estilização para a página estática, assim, substituindo as inúmeras *tags* que tinham que ser colocadas para criar layouts eficientes. Tornando assim a segunda camada do desenvolvimento, a camada da formatação.

> A construção de uma folha de estilos pode ser realizada através de qualquer editor de textos como o Bloco de notas, ou editores HTML. Para facilitar a construção das folhas de estilos, é recomendável utilizar um editor de HTML, como, por exemplo, Aptana, Adobe Dreamweaver, Front Page ou outro à sua escolha. (JOBSTRAIBIZER, 2009)

As regras de estilo CSS podem ser definidas de três formas. Inline, as regras são definidas dentro de uma *tag* HTML.

- ***Inline***: as regras são definidas dentro de uma *tag* HTML.
- **Interna**: as regras são definidas no cabeçalho do documento HTML.
- **Externa**: as regras são definidas em um documento separado, fora de todos os documentos HTML.

Figura 9 - Exemplo Regras CSS

Fonte: DevMedia, 2022.

A estrutura da sintaxe CSS é bem simples, estipulando regras para o arquivo em HTML. Com cada regra é possível estilizar o conteúdo todo ou somente determinados elementos, como demonstrado na figura 9 acima. Foi utilizado a regra interna, definindo assim a estilização no cabeçalho do documento.

A diferença entre um *site* que implementa CSS e outro que não o usa é gigantesca, na figura 10 mostra uma página HTML usando a estilização, a mesma página da figura 9, podendo assim notar a gigantesca diferença.

Figura 10 - Exemplo Regras CSS

Fonte: Autoria Própria, 2022.

Segundo Silva (2007), ao usuário é facultado criar folhas de estilos personalizadas de acordo com suas preferências e necessidades. A figura mostra como é essencial o uso do CSS nos dias de hoje, e como ele é indispensável na criação de um documento bonito e organizado.

2.6 BootStrap

Framework é um termo em inglês que significa estrutura. De maneira geral, eles são conjuntos de códigos composto por uma estrutura, por sua vez as estruturas são adicionadas aos projetos para facilitar e acelerar o desenvolvimento.

Segundo Jain (2014), atualmente existem várias técnicas que facilitam o desenvolvimento de *sites* responsivos, uma delas é utilizar *frameworks.* Assim, um *framework* tem como objetivo oferece recursos que facilitam o desenvolvimento ao abstrair conceitos, normalmente de mais baixo nível, viabilizando o reuso de código. Funcionando assim como uma espécie de *template* ou modelo que, quando utilizado, oferece certas melhorias para elementos estruturais básicos. A principal função de um *framework* é facilitar o processo de desenvolvimento de um determinado *site*, ou *software*. Pois o conjunto de sintaxes de tarefas específicas que eles têm, que inclui código fonte, compiladores, bibliotecas, classes abstratas, *APIs* e ainda outros elementos que oferecem suporte geral ao desenvolvimento, permitem com que construam *sites* muito mais rapidamente, pois não é preciso se preocupar com comando básicos e funções adicionais.

> Um dos problemas que facilmente é detectado na versão *desktop* deste *site* são as medidas fixas (*pixels*). Se redimensionarmos a janela do navegador, perderemos parte do *site*, obrigando ao usuário visualizar o *site* sempre em tela cheia. Já na versão *mobile*, temos o problema ainda maior. (TOMAZINI & LOPES, 2020)

BootStrap é um *framework web* com código-fonte aberto, sendo um dos mais utilizados para desenvolvimento *front-end*. Ele fornece estruturas utilizando HTML, CSS e JavaScript e seu principal objetivo é criar *sites* responsivos, que permita a interface de um *site* seja otimizada para qualquer tamanho de tela, fazendo com que

os desenvolvedores não tenham que criar muitas versões de um mesmo *site* para todos os tipos e tamanhos de telas, melhorando assim a experiência do usuário em um *site* amigável e responsivo.

Figura 11 - Exemplo Página com BootStrap

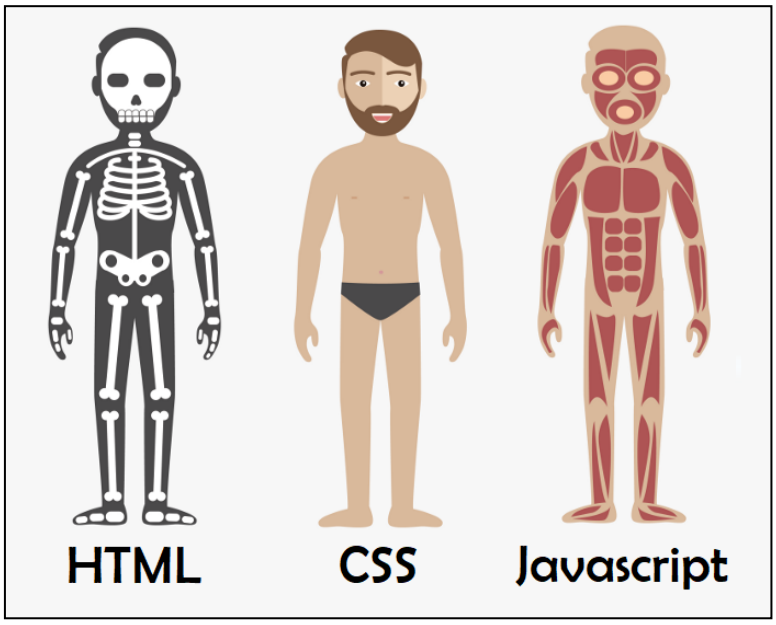
Fonte: Autoria Própria, 2022.

Segundo Matos e Zabot (2010), o Bootstrap é uma coleção de ferramentas de código aberto para desenvolvimento de *sites* e aplicativos *web*. Na figura 11 acima, podemos observar uma página desenvolvida com o *BootStrap*, é feita a comparação entre uma página no computador e no celular mostrando como a responsividade é o principal foco. Com isso o desenvolvimento é facilitado, e o projeto acelerado. Porém, como se trata de um *framework*, é necessário com que alguns códigos sejam importados para o projeto, são eles divididos em dois arquivos primários. Bootstrap.css é o primeiro *framework* CSS que organiza e gerencia o layout de um *site*, tudo o que você precisa fazer é referenciar as páginas *web* no arquivo CSS. O *bootstrap.js* é o segundo arquivo a ser importado, ele é a parte central do *BootStrap*, que consiste em arquivos JavaScript, responsáveis pela interatividade de um *site*. Com esses dois arquivos nós já conseguimos desenvolver páginas complexas e responsivas com simples linhas de código.

2.7 JavaScript

Ao implementar itens complexos em páginas *web* se usa o JavaScript, esta que é uma das principais linguagens de programação, oferecendo uma variedade muito grande de *frameworks* e aplicações para os mais diversos dispositivos. Apesar de ter *Java* no nome, *Java* e *JavaScript* não possuem nenhuma ligação direta uma com a outra. Sendo a terceira camada das principais tecnologias padrões da *web*, duas das quais (*HTML* e *CSS*) nós já falamos. Segundo a *Mozilla Foundation*, atual nome da antiga *Netscape Communications Corporations,* empresa responsável pela criação do *JavaScript*, "*JavaScript* é uma linguagem de programação, leve, interpretada, orientada a objetos, baseada em protótipos e em *first-class functions,* mais conhecida como a linguagem de *script* da internet." Na figura 12 a seguir é apresentado como as tecnologias funcionam, tendo como embasamento o corpo humano.

Figura 12 - Principais Tecnologias



Fonte: Alura, 2022.

O *JavaScript* é uma linguagem de programação de alto nível, sendo criada pelo programador Brendan Eich, a linguagem foi criada com a finalidade de fornecer um meio de adicionar interatividade à uma página *web*. Ao usar essa linguagem de programação para a criação de páginas *web*, é possível gerar diversos itens interativos à página, permitindo a você criar conteúdo que se atualiza dinamicamente, controlar multimídias, imagens animadas, e tudo o mais que há de interessante.

De acordo com Cruz e Valente (2016), *JavaScript* é uma linguagem muito conhecida e cada vez mais popular. Ela representa, por exemplo, 15% dos repositórios do *GitHub*. Isso ocorre porque a tecnologia mostra grandes vantagens aos programadores, em razão de que ela funciona como um grande facilitador de diversos procedimentos e pode ser integrada a outros idiomas para fazer a comunicação com bancos de dados.

Para entendermos a funcionalidade do *JavaScript*, vamos primeiramente saber o que é *script*, para em seguida dermos continuidade no que realmente é a linguagem em questão. *Script* é uma série de instruções que, quando executadas de maneira ordenada e seguindo um conjunto de instruções, realizam determinada tarefa em um *software*, automatizando uma execução de tarefas que seriam executadas, uma de cada vez, por um operador humano. A expressão “linguagem de *script*” é também eventualmente empregada para se referir a linguagens de propósitos diversos de alto nível dinâmicas.

Segundo Flanagan (2004), *JavaScript* é uma linguagem de alto nível, dinâmica, interpretada e não tipada, conveniente para estilos de programação orientados a objetos e funcionais. Uma linguagem de programação não tipada é uma linguagem onde podemos declarar variáveis sem definir o seu tipo, ou seja, se você é uma pessoa que já é um pouco experiente no ramo da programação, talvez consiga codificar e produzir seu código muito mais rapidamente. Na figura 13 a seguir, é representado um código-fonte de um documento criado com a linguagem *JavaScript*. Para intercalar um trecho de código *JavaScript* em uma página *web*, é necessário utilizar a *tag* <*script*> e </*script*>. Observando a imagem, podemos acatar a ideia de que a linguagem tem a função de instruir o programa a seguir determinadas tarefas que estão incluídas em um *script*.

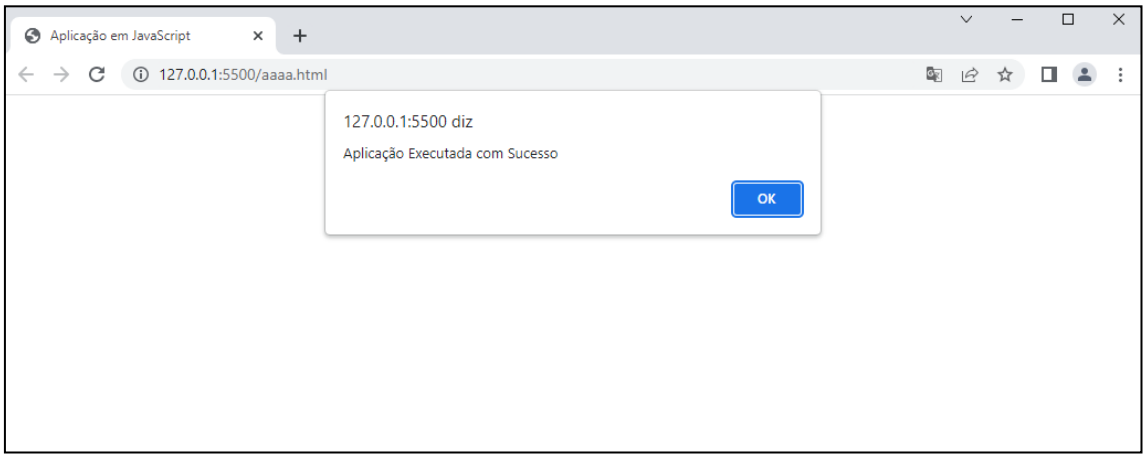
Figura 13 - Exemplo de documento em JavaScript

```
1   <!DOCTYPE html>
2   <html lang="en">
3       <head>
4           <meta charset="utf-8">
5           <title>Aplicação em JavaScript</title>
6           <script type="text/javascript">
7               alert('Aplicação Executada com Sucesso!');
8           </script>
9       </head>
10  </html>
```

Fonte: Autoria Própria, 2022.

O *JavaScript* é uma linguagem de programação *front-end*, assemelhando-se assim com *HTML* e *CSS*. A principal função da linguagem é consentir a criação de conteúdos dinâmicos com controle de animações e mídias, deixando o *site* mais interativo com o usuário. A linguagem foi criada inicialmente para a programação *client-side*, ou seja, ela é uma linguagem que é executada no lado cliente, podendo ser utilizada no computador do próprio usuário. Na figura 14 a seguir, podemos observar uma página *web* interativa e dinâmica, construída com a linguagem de programação *JavaScript*.

Figura 14 - Execução JavaScript



Fonte: Autoria Própria, 2022.

Ao adicionar o *JavaScript*, nós podemos deixar o *site* muito mais dinâmico e mais apresentável, desenvolvendo diversas maneiras de fazer uma página interativa com o usuário. Ao longo dos anos foram desenvolvidas algumas aplicações em cima da linguagem. Ao longo dos anos foram desenvolvidas algumas aplicações em cima da

linguagem. O *Node.js* é uma das mais importantes, dado que ele permite a criação de aplicações *server-side,* além disso é possível criar aplicações *Mobile*. Ao programar utilizando a linguagem *JavaScript* também é viável desenvolver aplicações *Desktop* e além do mais pode ser usada para a criação de jogos, sendo assim um ecossistema muito versátil. Usando o *JavaScript* também é possível desenvolver aplicativos muito poderosos e úteis. Por ser uma linguagem muito difundida, várias empresas a utilizam para programar aplicativos, tanto para uso interno, quanto para o uso usuário final no mercado.

> É uma linguagem de programação usada para manipular, personalizar e automatizar as funcionalidades de um sistema existente. Em tais sistemas, as funcionalidades já se encontram disponíveis por meio de uma interface de usuário e a linguagem de *script* provê um mecanismo para acessá-las. (SILVA, 2020)

O *JavaScript* possui inúmeras vantagens que fazem da linguagem uma excelente escolha entre outras linguagens presentes no mercado atualmente. Mas uma vantagem que é muito importante citar é que ao programar com ela não é necessário utilizar um compilador, visto que os navegadores da internet interpretam a linguagem com *HTML*. Esse fator já faz com que a linguagem se sobressaia das outras, porque geralmente ao programar com outras linguagens é necessário o uso de um compilador. Vamos entender mais a fundo algumas das vantagens acerca da linguagem *JavaScript* de acordo com o Hostinger (2022).

- Você não precisa de um compilador porque os navegadores de internet o interpretam com HTML.
- É mais fácil de aprender do que as outras linguagens de programação.
- Erros são bem mais fáceis de localizar e, por conta disso, de serem corrigidos.
- Ele pode ser designado a certos elementos de páginas de internet ou eventos específicos, como cliques e rolagens de mouse personalizados.
- É totalmente compatível com várias plataformas e navegadores.
- Você pode validar entradas e reduzir a necessidade de verificações manuais de dados

- Ele faz com que os *sites* sejam bem mais interativos e segurem a atenção dos visitantes por mais tempo, características que define a experiencia do usuário (UX).
- Ele é mais rápido e mais leve que as outras linguagens de programação.

Entretanto o *JavaScript* também possui falhas como qualquer outra linguagem. Por exemplo, ela é vulnerável a brechas de segurança, em outras palavras, isso quer dizer que o usuário corre riscos de serem executados códigos maliciosos em seu aparelho.

JavaScript é uma linguagem de programação de excelente reputação, por isso, ter conhecimento e experiência nela é um diferencial, já que a linguagem possui infraestrutura para o desenvolvimento de projetos deste tipo, por meio de tecnologias *web*. Uma vez que já está instalado em todos os navegadores da *web* hoje, *JavaScript* poupa o tempo de configurar um ambiente de desenvolvimento, podendo simplesmente "começar a codificar". Além disso, se você sabe programa em *JavaScript*, faz parte de uma grande comunidade que oferece suporte e conselho. Para aprender de fato *JavaScript*, é essencial ter conhecimento sobre lógica de programação, além de ter uma boa noção sobre as linguagens de programação *HTML* e *CSS*. A linguagem em resumo é usada para implementar em seu *site* funcionalidades mais complexas. Aprender a programar em *JavaScript* é uma excelente escolha para quem busca melhorar suas habilidades em designar objetos de páginas *web*, como cliques personalizados e rolagens de mouse.

2.8 PHP

A linguagem de programação *Hypertext Preprocessor* (PHP), originalmente chamada de *Personal Home* Page, utilizada para o desenvolvimento de aplicações e *web* para a criação de *sites*, uma linguagem frequentemente usada para a comunicação do lado servidor ou *back-end* do projeto. A linguagem é capaz de lidar com várias funções de *back-end,* como, coletar formulários de dados, gerenciar arquivos do servidor, modificar bases de dados e muito mais. De grosso modo, podemos dizer que é uma linguagem de *script* que serve para automatizar a execução de tarefas e deixá-las dinâmicas.

De acordo com os autores Converse e Park (2003), o PHP é uma linguagem para a criação de *scripts* para *Web* do lado servidor embutidos em HTML, cujo código-fonte

é aberto, e que é compatível com os mais importantes servidores *web*. Uma linguagem *open source* (código aberto) se trata de um código projetado para ser acessado abertamente pelo público, fazendo com que todas as pessoas possam vê-lo e modificá-lo. O PHP é uma linguagem de código aberto, de uso geral e muito utilizada, ela acaba trazendo transparência e integração entre ferramentas, é adequada para o desenvolvimento *web* e que pode ser embutida dentro do HTML.

> O PHP passou por várias reescritas de código ao longo do tempo e nunca parou de conquistar novos adeptos. Uma segunda versão foi lançada em novembro de 1997, sob o nome PHP/FI 2.0. Naquele momento, aproximadamente 60 mil domínios, ou 1% da internet, já utilizavam PHP, que era mantido principalmente por Rasmus. (DALL'OGLIO, 2015)

Na figura 15 a seguir, podemos observar um código simples em PHP, formando um conjunto de *scripts*, que quando executadas de maneira ordenada realizam a tarefa proposta pelo usuário.

Figura 15 - Conjunto de scripts em PHP

```
1   <?php
2
3   $varialvel = '
4   <table class="table" width="100%">
5       <thead width="100%">
6       </thead>
7   </table>
8
9   echo '
10  <table class="table" width="100%">
11      <thead width="100%">
12      </thead>
13  </table>
```

Fonte: Stack Overflow, 2022.

As páginas PHP contêm HTML em código mesclado, ele é executado no servidor. Gerando HTML que é então enviado diretamente para o navegador, que recebe os resultados da execução desse *script*. Um outro benefício desse recurso é que quando você está utilizando marcações HTML repetidamente, você pode implementar o código em um simples arquivo PHP. Você consegue criar aplicações que efetuam alguma tarefa determinada pelo usuário, essas aplicações são compiladas dentro de um servidor, chamado de *server-side* (*script* do lado servidor), esse termo é tradicional

e muito utilizado pelos programadores. Posteriormente você encontra uma lista das principais *tags* em PHP e suas funções:

- **strlen()**: Esta função retorna à quantidade de caracteres de uma *string*, na qual devemos passar por parâmetro.
- **substr()**: Esta função retorna uma parte de uma *string*, na qual podemos informar o início e término ou apenas de onde deve-se iniciar.
- **strtolower()**: Esta função converte todos os caracteres da *string* para minúscula.
- **strtoupper()**: Esta função converte todos os caracteres da *string* para maiúscula.
- **strip_tags()**: Esta função remove *tags* HTML de uma *string*.
- **str_replace/str_ireplace()**: Esta função remove determinada ocorrência de toda uma *string*. O str_replace diferencia maiúscula de minúscula e o str_ireplace não faz essa diferenciação.
- **explode()**: Esta função quebra uma *string* em um ponto especificado e retorna um *array*, na qual podemos acessar um de seus índices.

Na figura 16 a seguir, podemos observar a conexão entre um *site* e um banco de dados, desenvolvido em PHP. Que é dinâmico e tem alta compatibilidade, sem falhas e com agilidade.

Figura 16 - Conexão com a Base de Dados

```php
<?php
    $link = mysqli_connect("HOST", "USUARIO", "SENHA", "BASE");

    if (!$link) {
        echo "Error: Falha ao conectar-se com o banco de dados MySQL." . PHP_EOL;
        echo "Debugging errno: " . mysqli_connect_errno() . PHP_EOL;
        echo "Debugging error: " . mysqli_connect_error() . PHP_EOL;
        exit;
    }

    echo "Sucesso: Sucesso ao conectar-se com a base de dados MySQL." . PHP_EOL;

    mysqli_close($link);
?>
```

Fonte: Autoria Própria, 2022.

Segundo Niederauer (2017), um programa PHP pode ser escrito em qualquer editor de texto, como, por exemplo, o Bloco de Notas (Notepad), do Windows, ou o Vi, do

Linux. Algumas outras vantagens do PHP é que ele é fácil de aprender, tem um baixo custo e integração com bases de dados. A linguagem também é altamente popular devido à sua natureza de código aberto e suas funcionalidades versáteis. Aprender a programar em PHP certamente é uma ótima escolha para quem busca melhorar suas habilidades como programador, seja iniciante ou mais experiente.

2.9 React Native

O *framework React Native* é uma estrutura de aplicativo móvel bastante popular entre os desenvolvedores. Durante a sua criação ele foi baseado na linguagem JavaScript, linguagem essa que permite criar aplicativos móveis renderizados efetivamente para iOS e Android. Inicialmente, ele foi lançado no ano de 2015 pelo Facebook em um projeto de código aberto. Jordan Walke, desenvolvedor do Facebook fez uma descoberta inovadora anos depois. Através de muitas pesquisas ele encontrou definitivamente um método que gerasse elementos de interface do usuário, ou seja, nesse momento era possível desenhar uma tela de app utilizando a linguagem JavaScript. Resumidamente, o *framework* foi desenvolvido somente para a plataforma iOS e mais adiante o Facebook desenvolveu também o suporte para plataformas Android.

Segundo Neves e Junior (2020), é natural o surgimento de uma grande demanda por desenvolvimento de aplicativos para resolver diversos tipos de problemas. Código aberto é um termo que se refere a um *software* que o código está disponível para download por qualquer pessoa, ou seja, qualquer um pode acessar seu código-fonte. Ao adquirir o *software*, o usuário tem total liberdade de alterar o código ou criar uma versão pessoal. O React Native tornou-se uma das principais soluções para o desenvolvedor *Mobile*, atualmente a linguagem é utilizada por muitos aplicativos de grande porte, como por exemplo: Discord, Tesla, Instagram e Facebook. Dentre as vantagens de se utilizar o *framework*, podemos citar uma bastante interessante, que é o fato de poder criar o código somente uma vez e o utilizar para alimentar seus aplicativos, seja da plataforma iOS ou Android.

> O React Native é um *framework* escrito em JavaScript que busca simplificar o desenvolvimento multiplataforma de aplicativos *mobile*, possibilitando desenvolver e criar aplicativos tanto para Android como para iOS apenas com uma única fonte de código. (React Native, 2021)

O React Native foi baseado no React, que é uma biblioteca JavaScript que já foi muito popular durante o lançamento do *framework mobile*. Além disso a linguagem capacitou os desenvolvedores front-end a terem mais agilidade, pois antigamente eles tinham que trabalhar com tecnologias baseadas na *web*, esse fator dificultava muito o processo de criação do *software*. O principal foco do React Native é tornar a experiência do usuário com uma interface mais eficiente. Além disso, ele também permite a reutilização de componentes que tenham sido desenvolvidos em outras plataformas e que tenham basicamente a mesma função.

De acordo com o Facebook (2021), um dos frameworks híbridos utilizado por grandes empresas é o React Native, uma biblioteca desenvolvida com JavaScript criado em 2015. A linguagem também é fácil de escrever pois ela usa JSX, que é uma extensão de sintaxe opcional para a linguagem de programação JavaScript, essa ferramenta permite a combinação das linguagens HTML e JavaScript. O React Native melhora de uma forma eficiente o processo de atualização do Document Object Model (DOM). Essa ferramenta permite que você construa um Virtual DOM e o hospede na memória, toda vez que ocorre uma mudança no DOM real, o virtual faz uma modificação automaticamente. Além de todas essas vantagens o framework também proporciona a criação de uma interface de usuário que pode ser facilmente encontrada e acessa diversos motores de busca, este recurso traz uma vantagem imensa, pois nem todos os frameworks de JavaScript são compatíveis a Search Engine Optimization (SEO). Usando o React Native você resumidamente pode escrever um código HTML em JavaScript.

2.10 Firebase

O Firebase foi criado em 2011 pela plataforma digital do Google, que por sua vez é utilizada para favorecer o desenvolvimento dos mais diversos aplicativos, seja *web* ou móveis, de uma forma simplificada. A ferramenta possui diversas funcionalidades, ela

é usada como uma técnica de Marketing Digital com a finalidade de aumentar o número de usuários e consequentemente gerar maiores rendas para a plataforma Google. O principal objetivo da plataforma é melhorar significativamente o rendimento de aplicativos por meio de implementações das mais diversas funcionalidades presentes nela.

Podemos utilizar as seguintes ferramentas do Firebase:

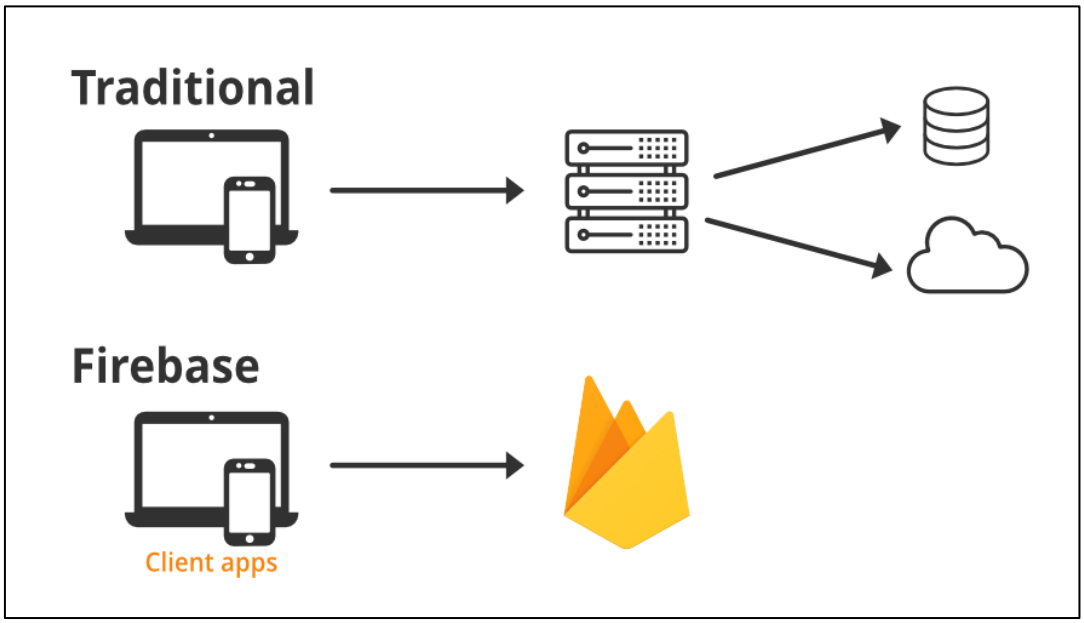
- **Firebase Authentication:** essa ferramenta proporciona uma interação dos usuários de uma forma dinâmica.
- **Dynamic Links:** essa ferramenta possibilita que os aplicativos criados pelos usuários possam ser visualizados e utilizados.
- **Cloud Messaging:** essa ferramenta envia notificações a várias outras plataformas.
- **Analytics:** essa ferramenta basicamente cria análises de resultados obtidos pelos usuários.

Segundo Silva (2018), o Firebase é uma plataforma da Google, que oferece vários serviços de *backend* para aplicações móveis e *web*, tais como: Authentication, Firebase Database e Firebase Storage. Utilizando a plataforma Firebase por exemplo, você pode fazer a comunicação com seus respectivos usuários utilizando o *In-app messaging*. Com ele é possível aperfeiçoar a experiência do público e futuramente convertê-los em potenciais clientes. Também é possível implementar funções que podem ser testadas antecipadamente em um subconjunto base de usuários para fazer a verificação mediante ao aplicativo. A utilização da plataforma é de extrema importância, pois os desenvolvedores podem fazer uma adaptação de acordo com as suas necessidades. O Firebase possui diversas características que o tornam exclusivo, por exemplo, a plataforma se encontra disponível para as principais plataformas móveis, tanto Android quanto iOS, ela também se encontra disponível na *web*. Além disso é possível ganhar dinheiro na plataforma, o uso da ferramenta Admob possibilita que o usuário ganhe dinheiro através de anúncios e publicidade.

> Firebase é um banco de dados na nuvem disponibilizado pela Google e o utilizando a estrutura NoSQL, que consiste em um banco de dados que não utiliza da linguagem SQL orientada a tabelas, mas sim, nesse caso, a documentos. A Orientação a documento, também conhecida como semi-estruturada, é um banco de dados que utiliza JSON (JavaScript Object Notation - Notação de Objeto JavaScript) para salvar e buscar dados. (MEZZARI; LEAL & VIEGAS, 2019, p.50-55)

A plataforma permite fazer o desenvolvimento gratuito dos aplicativos, sendo assim, os usuários preferem utilizá-lo. Além do mais a ferramenta Firebase gera crescimento e desenvolvimento nos aplicativos, essa é uma grande vantagem pois através de serviços oferecidos, é possível sim desenvolver suas aplicações de qualquer dispositivo de uma forma segura. Fazendo o uso da plataforma Firebase, não é necessário criar um banco de dados tradicional, que faz a conexão dos dispositivos, servidores e consequentemente ao banco de dados e nuvem. A plataforma faz a conexão direta dos dispositivos com o banco de dados. Na figura 17 a seguir, podemos observar como é feita essa conexão.

Figura 17 - Conexões dos Bancos de Dados



Fonte: Acervo Lima, 2022.

O Firebase oferece serviços que podem ser divididos em três categorias, sendo elas:

- Desenvolvimento;
- Crescimento de aplicativos e/ou negócio;
- Analytics próprio.

Segundo a Rockcontent (2021), o Firebase foi criado para que o rendimento dos aplicativos aumente, por meio da execução de diferentes tipos de funcionalidades. A plataforma inicialmente, oferece ao usuário a opção de começar utilizando as ferramentas com um plano gratuito e que pode posteriormente ser expandido para um plano pago, caso seja necessário o uso de ferramentas mais avançadas. Basta o usuário escolher o que é melhor para o desenvolvimento de seu aplicativo no momento.

2.11 UML

A *Unified Modeling Language* (UML) é uma poderosa linguagem de comunicação entre os programadores. Ela é uma linguagem de notação, ou seja, é uma forma de escrever, se comunicar e ilustrar os mais diversos projetos de sistemas. Para desenvolver um *software* é necessário seguir algumas etapas para que ele tenha uma boa qualidade, isso reduz significativamente as chances de erros. Fazer essas etapas, além de facilitar o trabalho dos envolvidos, ainda garante ao usuário uma boa experiência durante o uso. Para realizar este processo é preciso fazer a elaboração de uma documentação contendo a estrutura do projeto em questão.

> Uma empresa de *software* bem-sucedida é aquela que fornece *software* de qualidade e capaz de atender às necessidades dos respectivos usuários. Uma empresa que consiga desenvolver esse *software* de maneira previsível e em determinado período, com utilização eficiente e eficaz de recursos, será uma empresa com um negócio viável. (BOOCH, 2006)

Ao desenvolver um *software*, um programador pode utilizar os mais diversos elementos gráficos, como: retângulos, setas, linhas, entre outros elementos. A UML permite que o desenvolvedor crie diagramas que representem as áreas de um *software* e as interações e mudanças que podem ocorrer no mesmo. A equipe do

projeto faz uma espécie de “desenho” que auxilia a visualização dos aspectos do programa, facilitando assim a construção do *software*. Fazendo a criação da UML, os desenvolvedores podem ter uma visão geral do progresso do trabalho através de diagramas padronizados. Fazendo desta maneira pode-se evitar problemas que muitas vezes são comuns no desenvolvimento, como erros ao implementar ou comunicar os envolvidos. Como a UML é uma linguagem padrão entre os programadores, ela é eficiente e cumpre seu papel, que é fazer a organização do produto. Na linguagem UML são usados diagramas que se dividem em dois grupos, são os diagramas estruturais e os diagramas comportamentais.

Segundo Guedes (2018), seu principal enfoque está em permitir a visualização das classes que comporão o sistema com seus respectivos atributos e métodos. Inicialmente vamos entender o que são os diagramas estruturais: eles são utilizados para especificar os detalhes da estrutura de um sistema (parte estática). Se encaixam nesses diagramas, as classes, métodos, *interfaces*, *namespaces* e serviços, por exemplo: componentes que devem ser instalados e como deve ser a arquitetura do sistema. Agora vamos entender o que são os diagramas comportamentais: eles são utilizados para especificar os detalhes do comportamento de um sistema (parte dinâmica). Nesse diagrama as funcionalidades devem funcionar como um processo de negócio que deve ser tratado pelo sistema que está em processo de desenvolvimento. Um exemplo disso são os componentes estruturais que trocar mensagens e como eles respondem às chamadas.

A UML ajuda a esclarecer o escopo do projeto, pelo simples motivo de centralizá-lo em uma única visão, que é o diagrama. Utilizando uma linguagem que todos podem entender é essencial, pois um dos maiores problemas ao se produzir um *software* é a má comunicação. A linguagem é universal, ou seja, programadores do mundo todo a utilizam, isso evita problemas de incompatibilidade. As pessoas que geralmente são envolvidas neste projeto são os Analistas de Negócio, Product Owner, Scrum Master, Arquitetos, Desenvolvedores, Gerentes de Projeto/Produto. Alguns profissionais de diferentes níveis de senioridade consideram a utilização da UML uma perca de tempo, isso é uma grande farsa, pois a UML pode definir e organizar os requisitos funcionais no sistema e especificar o contexto e os requisitos do sistema.

De acordo com Larman (2000), durante a análise orientada a objetos, há uma ênfase em encontrar e descrever os objetos – ou conceitos – no domínio do problema. A UML pode ser utilizada quando é necessário fazer a especificação do desejo do cliente que será materializado no *software* a ser desenvolvido. Quando os integrantes de uma equipe precisam ter uma visão geral do projeto. E quando se comunicam para o mundo externo protocolos, ou seja, quando as interfaces do sistema devem ser consumidas por terceiros ou ilustrar as topologias arquiteturas físicas/lógicas. Podemos assim concluir que a linguagem UML é essencial para a produção de um *software*, pois ela é como um idioma que auxilia as equipes de produção a ter mais eficiência e obter melhores resultados. Ela possibilita a clareza, o que diminui diretamente o desperdício de tempo na produção de um *software*. Nos dias de hoje as coisas precisam ser desenvolvidas com agilidade e facilidade, isso se aplica também ao se produzir um *site* ou aplicativo.

2.11.1 Levantamento de Análise

O levantamento de análise de requisitos de um *software* tem a função de obter diretamente as necessidades específicas do cliente em questão, com o intuito de solucioná-la. Dessa maneira os problemas solucionados pelo sistema serão problemas reais, e não apenas problemas que foram imaginados pelo desenvolvedor. Podemos classificar os requisitos em três tipos:

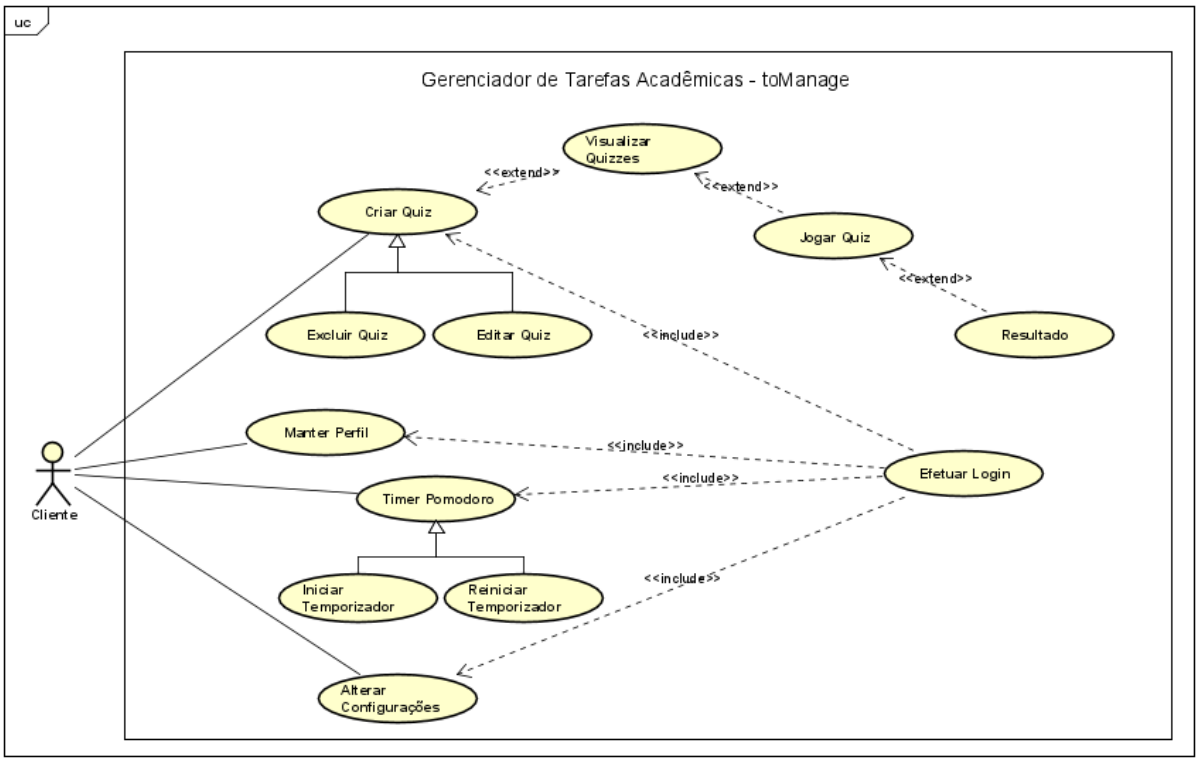
- Capacidades operacionais que o usuário precisa para solucionar problemas ou realizar atividades específicas;
- Capacidades operacionais cruciais para que um *software* consiga satisfazer um contrato, seja padrão ou exigência formal;
- Requisitos que serão a base de desenvolvimento do sistema ou futuros componentes.

Sinteticamente, ao realizar o levantamento de requisitos de um produto o que a empresa precisa fazer é fazer a identificação das necessidades dos clientes, levando em conta do que é preciso para a gestão empresarial.

2.11.2 Diagrama de Caso de Uso

O objetivo do diagrama de caso de uso é representar como os casos de uso interagem entre si no sistema e com os usuários (atores), ou seja, como as funcionalidades se relacionarão umas com as outras e como serão utilizadas pelos clientes durante o uso do *software*. Neste diagrama, basicamente temos três principais elementos, que são: o ator, o caso de uso e o relacionamento. Na figura 18 a seguir, podemos observar um exemplo de um diagrama de caso de uso.

Figura 18 - Exemplo Diagrama de Caso de Uso



Fonte: Autoria Própria, 2022.

- **Ator:** O boneco.
- **Casos de Uso:** São as elipses formadas.
- **Relacionamentos:** São as setas que ligam os casos de uso entre si e ligam os usuários aos casos de uso.

A função do ator é fazer a execução do caso de uso, ou seja, ele quem executará a funcionalidade que está especificada no caso de uso. O ator pode ser definido em dois tipos: humano e sistêmico. O diagrama é composto por simples desenhos que descrevem de uma maneira objetiva e que escrito manualmente ficaria muito extenso.

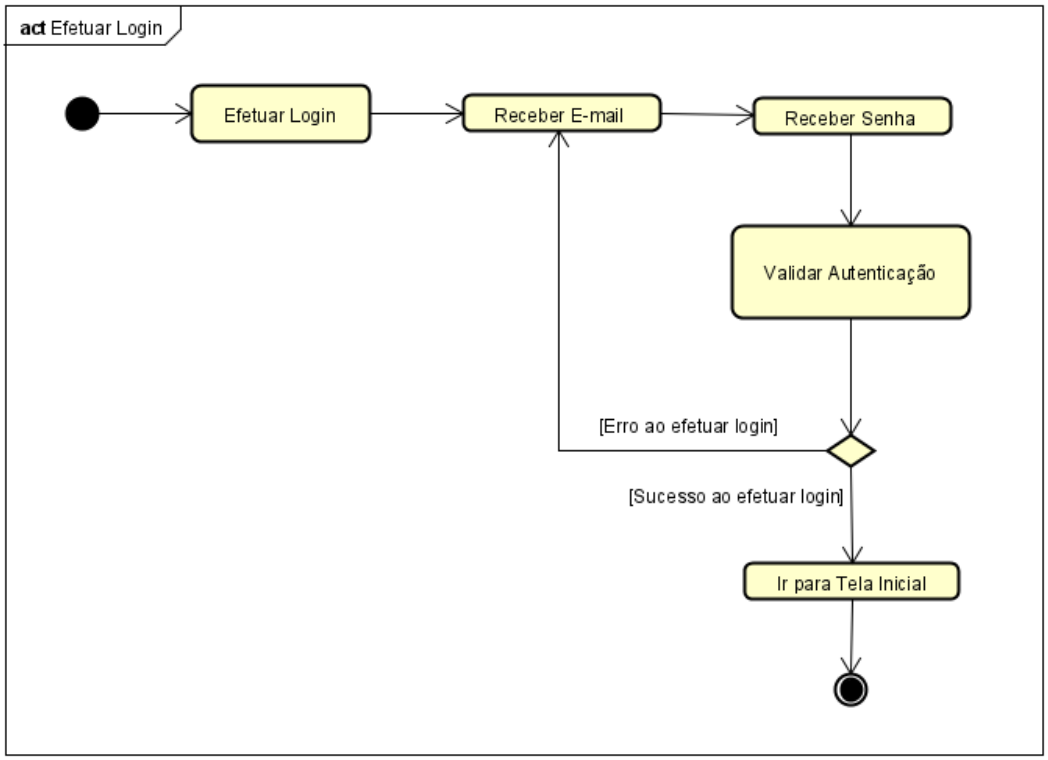
Nele também existe o conceito de *include* e *extend* que são as relações entre os casos de uso.

O *include* é a relação de um determinado caso de uso que para ser executado precisa chamar outro caso de uso. A relação *extend* quer dizer que um determinado caso de uso extendido irá funcionar exatamente como o caso de uso base, porém alguns passos novos serão inseridos no caso de uso extendido.

2.11.3 Diagrama de Atividades

A Linguagem de modelagem unificada inclui diversos subconjuntos de diagramas, isso inclui: diagramas de estrutura, de interação e de comportamento. Os diagramas de atividade, juntamente com os diagramas de caso de uso e de máquina de estados, são considerados diagramas de comportamento pois descrevem o que é necessário ocorrer no sistema sendo modelado. É essencial se comunicar com clareza e coesão, porque as partes interessadas lidam com muitas questões. Diagramas de atividade ajudam a unir as pessoas das mais diversas áreas de negócios de desenvolvimento de uma empresa para entender o mesmo processo e comportamento. Os diagramas de atividades podem representar e facilitar como as ações são efetuados dentro de um processo ou sistema, o que pode ajudar na comunicação dos processos de negócios e no *design* e desenvolvimento bem-sucedidos de sistemas. Na figura 19 a seguir, podemos observar um exemplo de um diagrama de atividades.

Figura 19 - Exemplo Diagrama de Atividades



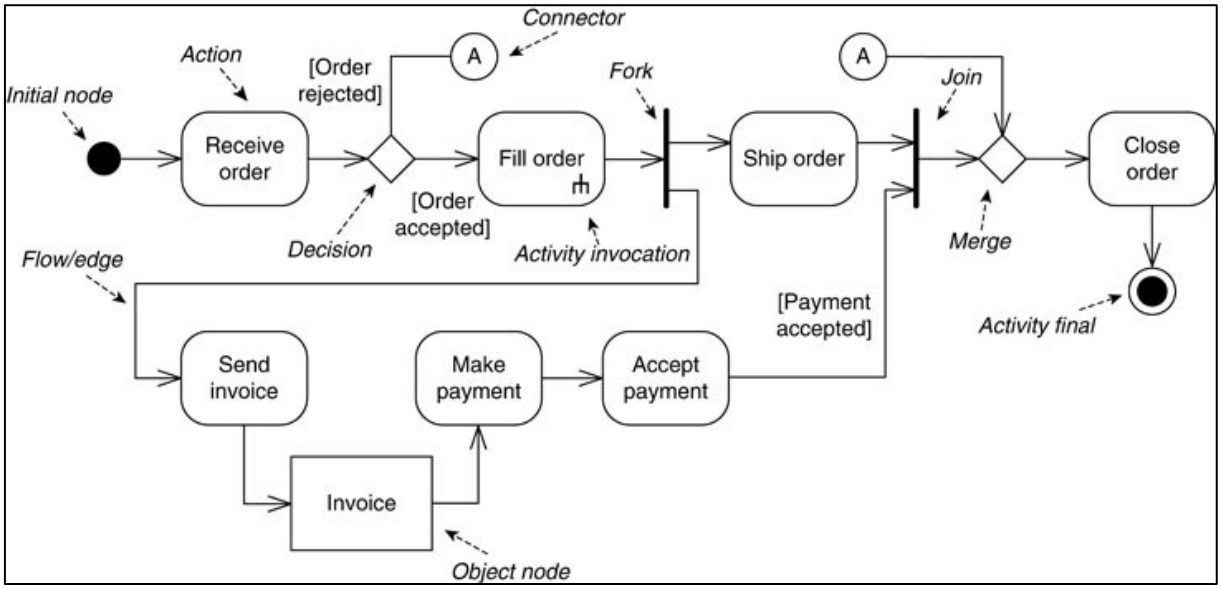
Fonte: Autoria Própria, 2022.

Para fazer a criação de um diagrama de atividade, é necessário um conjunto de símbolos especiais, incluindo aqueles para dar início, finalizar, fundir ou receber maias etapas no fluxo. Os diagramas de atividades trazem à uma organização diversos benefícios pois eles oferecem:

- Demonstram a lógica de um algoritmo.
- Descrevem as etapas que foram realizadas em um caso de uso UML.
- Ilustram um processo de negócio ou fluxo de trabalho entre usuários e sistema.
- Facilitam e melhoram qualquer processo ao esclarecer casos de uso complicados.
- Modelam elementos de arquitetura de *software*, como um método, função e operação.

Antes de fazer a criação de um diagrama de atividades, é necessário entender os componentes básicos dele. Os componentes mais comuns de um diagrama de atividade incluem: ações, nó de decisão, fluxos de controle, nó inicial e por fim nó final. Na figura 20 a seguir, podemos observar a simbologia de um diagrama de atividades.

Figura 20 - Simbologia Diagrama de Atividades



Fonte: Venngage, 2022.

- **Nó inicial:** é um círculo pequeno preenchido que simboliza o estado inicial ou o início de uma atividade.
- **Nó de atividade:** é um retângulo com cantos arredondados que simboliza uma atividade ou estado de uma ação.
- **Nó de ação:** é um símbolo em forma de estádio ou cápsula que é usado para fazer a representação de uma ação.
- **Fluxo de ação:** é uma seta que representa a modificação de uma atividade ou ação para outra. É chamado também de borda de atividade ou fluxo de controle.
- **Nó de objeto:** é um retângulo que faz a representação de um objeto criado ou usado na atividade.
- **Fluxo de objetos:** é um ângulo ou seta tracejada e colocada em seguida de uma ação para demonstrar a criação de um objeto ou antes de uma ação para demonstrar que ela requer um objeto.
- **Nó de bifurcação ou nó de junção:** eles são representados por uma linha horizontal grossa que faz a divisão de uma ação em fluxos simultâneos (nó de bifurcação) ou faz a junção de fluxos simultâneos em uma única e exclusiva ação (nó de junção).
- **Evento temporal:** é um símbolo de ampulheta que faz a representação de um intervalo de tempo dentro de uma atividade.

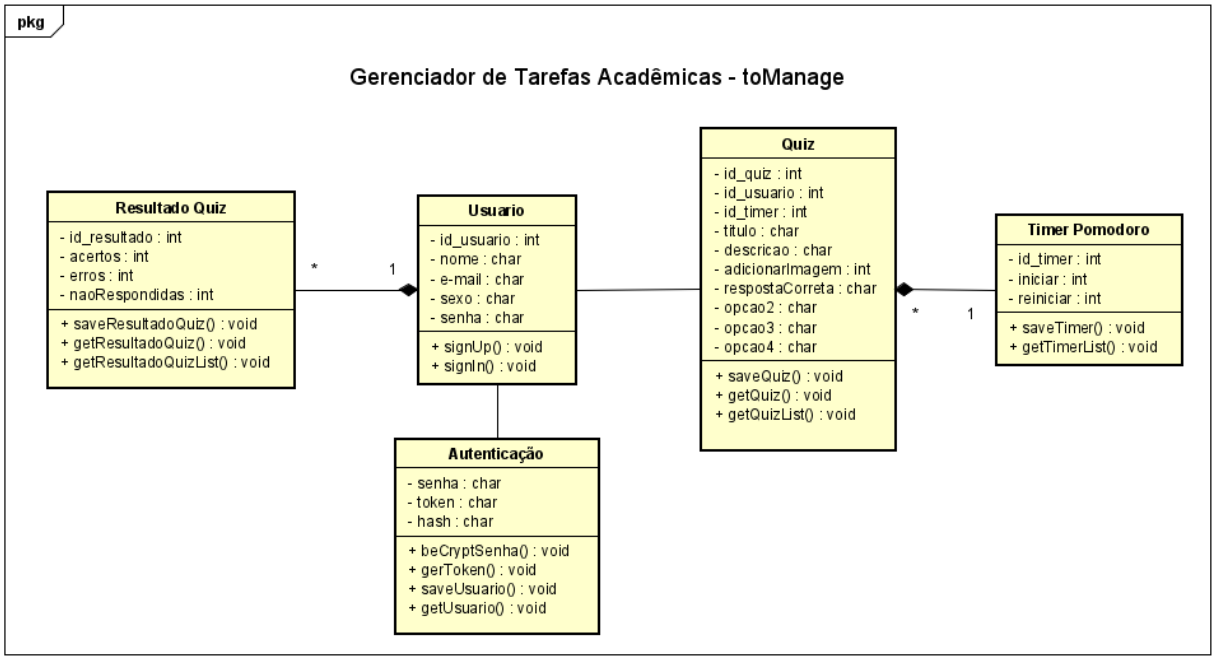
- **Sinais enviados e recebidos:** é uma forma de pentágono em forma de seta, que é o sinal enviado e indica que um sinal precisa ser apurado para concluir uma determinada ação, enquanto uma forma de bandeira representa o sinal recebido e indica que o sinal foi apurado.
- **Nó final:** é um círculo pequeno preenchido dentro de outro círculo que sinaliza a fase final ou o fim da atividade.

2.11.4 Diagrama de Classes

Os diagramas de classes são um dos tipos mais importantes de diagramas UML, porque eles mapeiam de forma objetiva a estrutura de um determinado sistema ao modelar as suas classes, atributos, operações e relações entre objetos. A linguagem de modelagem unificada (UML) ajuda o usuário a fazer a modelagem de sistemas de inúmeras maneiras. Esse tipo de diagrama é muito usado por engenheiros de *software* ao realizar a documentação de arquiteturas de *software*, eles são um tipo de diagrama da estrutura pois descrevem o que realmente deve estar presente no sistema a ser modelado.

A UML foi elaborada como um modelo padronizado para descrever uma abordagem de programação orientada a objetos. As classes são os componentes básicos dos objetos, os diagramas de classes são os componentes básicos da UML. Esse tipo de diagrama representam as classes que serão utilizadas para programar os objetos ou as interações entre classes e objetos. Na figura 21 a seguir, podemos observar um exemplo de diagrama de classes construído.

Figura 21 - Exemplo Diagrama de Classes



Fonte: Autoria Própria, 2022.

A forma de classe em si é basicamente constituída por um retângulo com três linhas. A linha superior contém o nome da classe, a linha do meio os atributos da classe e a linha inferior contém os métodos ou operações que a classe pode empregar. As classes e subclasses são agrupadas juntas para mostrar a relação estática entre cada objeto.

Mas afinal, quais os benefícios que os diagramas de classes trazem? Bom, eles oferecem uma sucessão de benefícios para qualquer organização, a seguir estão listados alguns deles:

- Fazem com que o usuário tenha uma melhor visão geral dos esquemas de uma determinada aplicação.
- Ilustram modelos de dados para sistemas de informação, seja ele simples ou complexo.
- Expressam visualmente as necessidades específicas de um *software* e divulgam essas informações por toda a organização.
- Fornecem uma descrição apartada de implementação de tipos utilizados em um sistema e em seguida são passados entre seus componentes.

- Criam gráficos em detalhe que ressaltam qualquer código necessário para ser programado e implementado na estrutura descrita.

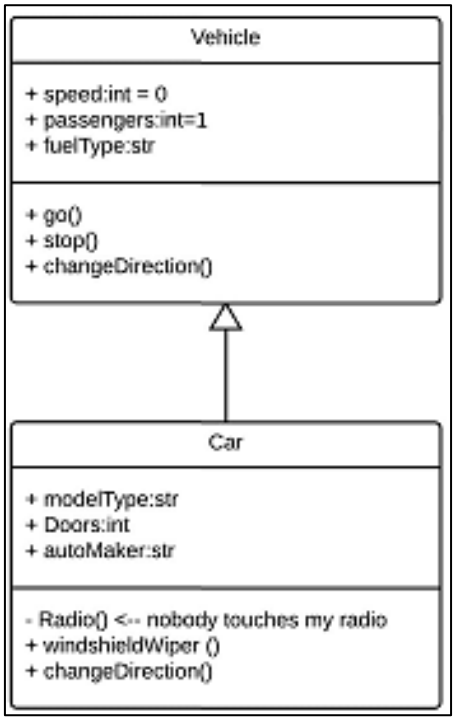
Como padronizado, o diagrama de classes é composto por três partes, sendo elas: a parte superior que contém o nome da classe, seja falando do classificador ou de um objeto; a parte do meio que contém os atributos da classe, esta partição descreve as qualidades da classe e é necessária somente quando se descreve uma instância específica de uma classe; a parte inferior que contém as operações da classe (métodos), é exibido em formato de lista e descreve como uma classe interage com os dados.

Todas as classes têm níveis de acesso distintos, dependendo do modificador de acesso (visibilidade). A seguir estão listados os níveis de acesso com seus respectivos símbolos:

- Público (+)
- Privado (-)
- Protegido (#)
- Pacote (~)
- Derivado (/)
- Estático (sublinhado)

Existem dois escopos para membros, os classificadores e as instâncias. Os classificadores são membros estáticos, à medida que as instâncias são as instâncias específicas da classe. A hereditariedade é o processo de uma subclasse, assumindo assim a funcionalidade de uma classe primária, esse processo é simbolizado por uma linha conectada reta com uma ponta de seta fechada apontando para a classe primária. Na figura 22 a seguir, podemos observar o processo de hereditariedade.

Figura 22 - Exemplo de um Processo de Hereditariedade



Fonte: Lucidchart, 2022.

A associação bidirecional é a relação padrão entre duas classes, onde ambas as classes estão cientes uma da outra e da relação existente entre elas, essa associação é representada por uma linha reta entre duas classes. Já a associação unidirecional é uma relação entre duas classes, onde uma classe está ciente da existência da outra classe e interagem entre si, ela é modelada por uma linha reta de ligação com uma ponta de seta aberta da classe conhecimento à classe conhecida. Na figura 23 a seguir, podemos observar o funcionamento desses dois tipos de associações.

Figura 23 - Funcionamento das Associações

Fonte: Macoratti, 2022.

2.11.5 Diagrama de Sequência

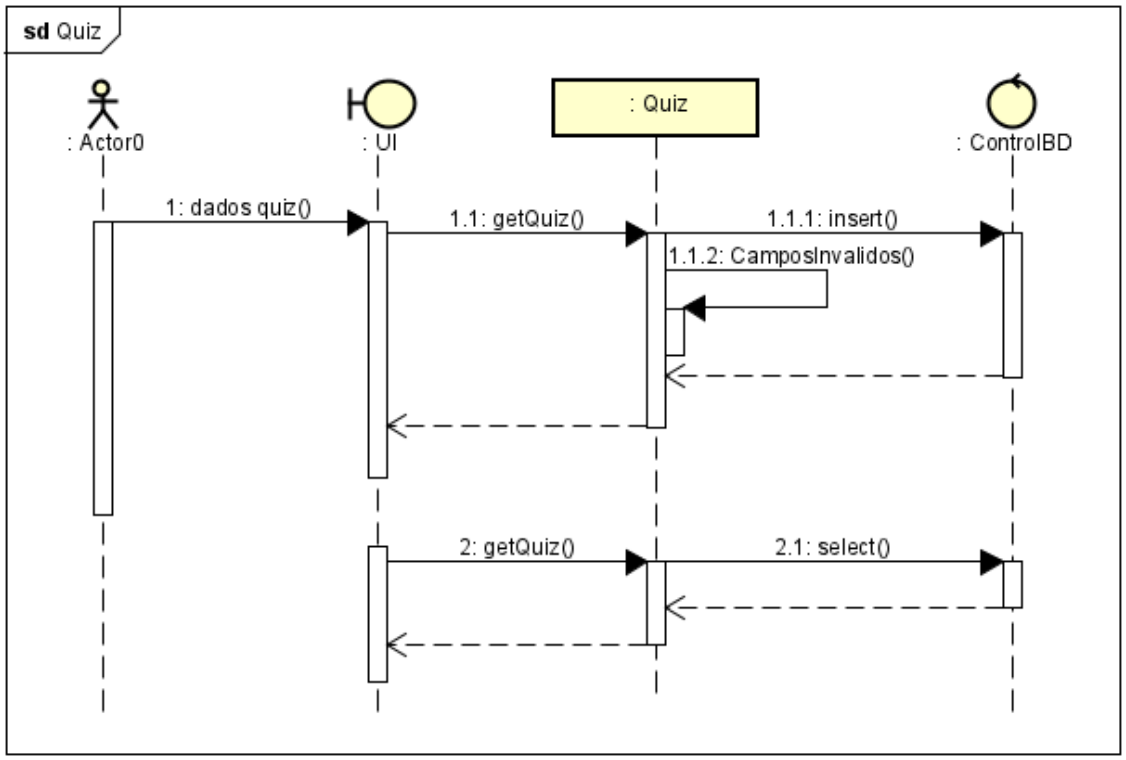
O diagrama de sequência é muito usado pois ele incide especificamente sobre os processos e objetos que trabalham simultaneamente. Ele é uma espécie de diagrama de interação, pela razão dele descrever como e em qual ordem um grupo de objetos trabalha em conjunto. Os diagramas de sequência são usados por desenvolvedores de *software* e profissionais de negócios para entender as necessidades de uma nova aplicação ou simplesmente para documentar um processo já existente. Esse tipo de diagrama também é conhecido como diagrama de eventos ou cenário de eventos.

Os diagramas de sequência têm diversos benefícios, eles podem ser referências úteis para organizações, abaixo estão listados alguns de seus benefícios:

- Modelam a lógica de um processo, função ou operação sofisticado.
- Representam os detalhes de um caso de uso UML.
- Manifestam como objetos e componentes interagem uns com os outros para concluir um processo.
- Planejam e compreendem a funcionalidade detalhada de um cenário existente ou futuro.

Os diagramas de sequência apresentam casos de uso ideais para cada cenário. O cenário de uso é um diagrama de como seu *software* poderia ser utilizado, ele é perfeito para certificar-se de que o usuário compreendeu a lógica de cada cenário de uso do sistema. Assim como o usuário manipularia um diagrama de sequência UML para fazer a exploração da lógica de um caso de uso, ele também pode utilizar a lógica do método para explorar a lógica de qualquer função, procedimento ou algum processo complexo. A lógica de serviço são métodos de alto nível usados pelos clientes, é basicamente um diagrama de sequência ideal para mapeá-lo. Na figura 24 a seguir, podemos observar detalhadamente como é o funcionamento de um diagrama de sequência através de um exemplo.

Figura 24 - Exemplo Diagrama de Sequência



Fonte: Autoria Própria, 2022.

3 DESENVOLVIMENTO

Neste capítulo será abordado a forma como os desenvolvedores se organizaram para assim dar funcionalidade tanto para a aplicação *mobile*, quanto para o *website*. A maneira como foram desenvolvidos e as ferramentas neles presente estão detalhadas e especificadas de maneira que o usuário consiga entender a proposta deles.

3.1 Aplicação Mobile

A aplicação *mobile* é formada por telas de navegação para assim obter uma melhor organização e poder ter uma melhor noção de onde se encontram as ferramentas. A tela de *login* é a primeira delas, onde o usuário consegue entrar em sua conta e começar a usar a aplicação, esse processo é seguro, pois ao realizar o *login* seus dados ficarão salvos e autenticados na nuvem do Firebase. Nessa tela o usuário irá preencher os campos “E-mail” e “Senha” e em seguida clicar no botão “Entrar”. Podemos observar na figura 25 a seguir a tela de *login* da aplicação toManage e sua aparência.

Figura 25 - Tela de Login

Fonte: Autoria Própria, 2022.

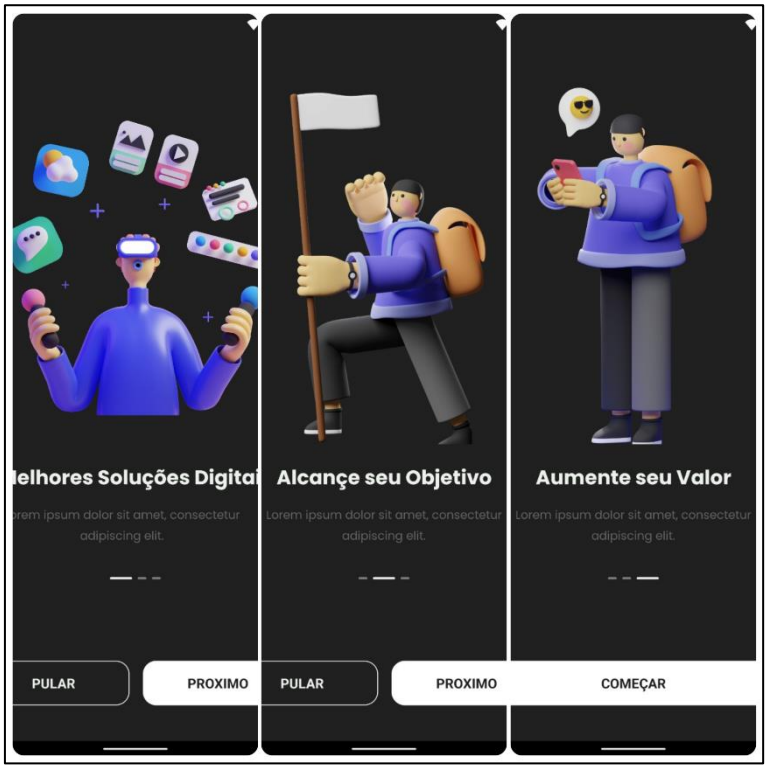
Caso seja a primeira vez que o usuário esteja utilizando a aplicação, ele poderá clicar sobre “*Criar Conta*”, dessa forma ele será direcionado para a tela de cadastro, onde ele irá informar seus dados nos campos “E-mail” e “Senha” e em seguida clicar no botão “Cadastrar”, para assim começar a utilizar a aplicação. Podemos ver isso na tela de cadastro, apresentada na figura 26.

Figura 26 - Tela de Cadastro

Fonte: Autoria Própria, 2022.

Logo após o usuário fazer *login* na aplicação, ele será direcionado para a tela de dicas, onde são apresentadas algumas informações sobre a aplicação e suas funcionalidades. Nessa tela existem os botões: “Pular” – que será utilizado caso o usuário já tenha usado a aplicação alguma vez e já sabe como funciona; “Próximo” – para navegar entre as telas; “Começar” – para ser direcionado para a próxima tela. Podemos observar esta ação na figura 27 a seguir, na tela de dicas.

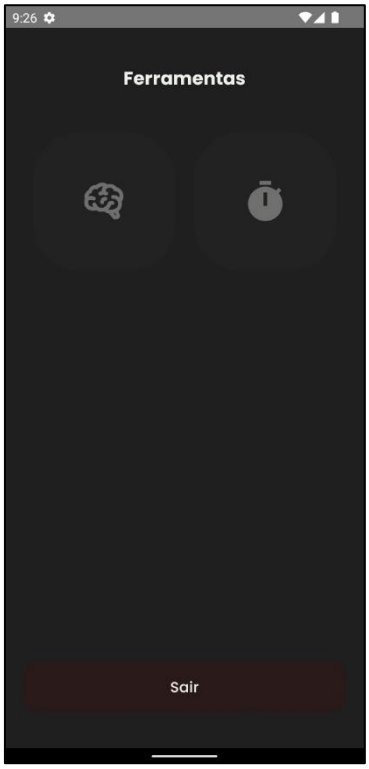
Figura 27 - Tela de Dicas



Fonte: Autoria Própria, 2022.

Ao clicar no botão “Começar” o usuário será direcionado para a aba de ferramentas da aplicação. Nessa tela ele poderá escolher entre criar um quiz (ícone de cérebro) ou cronometrar seu tempo de estudo através do timer Pomodoro (ícone de relógio). É apresentado também nessa tela o botão “Sair”, caso o usuário queira parar de utilizar a aplicação *mobile* toManage. Na figura 28 a seguir podemos observar a tela de ferramentas e sua aparência.

Figura 28 - Tela de Ferramentas

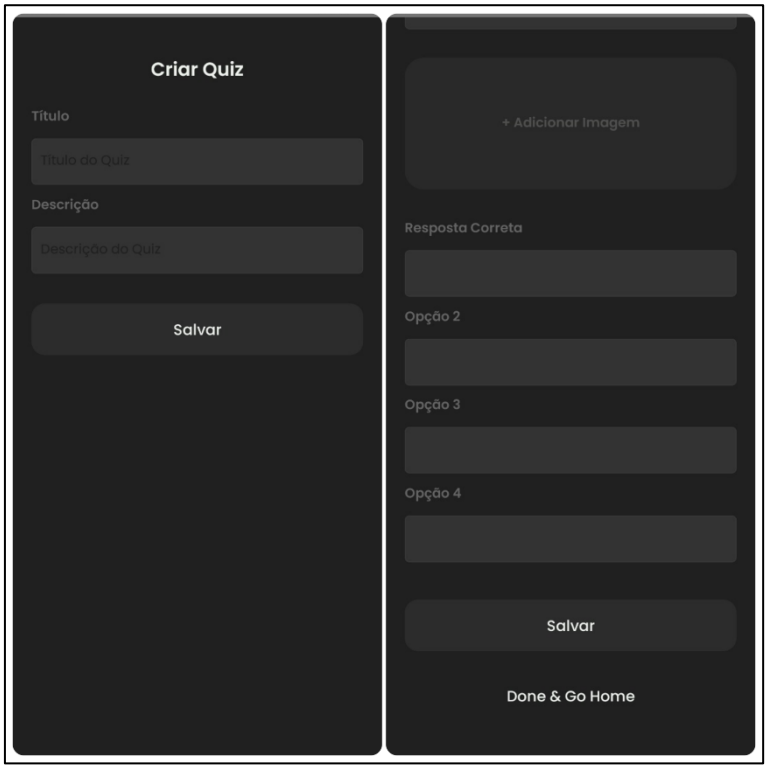


Fonte: Autoria Própria, 2022.

Se por exemplo, o usuário escolher opção de criar um quiz, ele será direcionado para uma tela onde é possível preencher os campos: “Título” e “Descrição” para começar a editar as suas devidas respostas, preenchendo os campos; “Resposta correta”; “Opção 2”; “Opção 3”; “Opção 4”. É disponibilizado para o usuário a inserção de uma imagem para seu quiz ficar ainda melhor.

Ao terminar de preencher os devidos campos o usuário poderá finalmente salvar seu quiz clicando no botão “Salvar”, dessa maneira os dados inseridos nos campos ficarão salvos no banco de dados para posteriormente serem usados. É concedido a opção de voltar à página inicial no rodapé da tela, caso o usuário queira voltar ao início. Na figura 29 a seguir é possível observar a tela “Criar Quiz” e sua aparência.

Figura 29 - Tela de Criação de Quizzes



Fonte: Autoria Própria, 2022.

Com o quiz criado o usuário poderá respondê-lo após o término do seu tempo de estudo. Ele pode obter a resposta correta ou a resposta incorreta de acordo com a escolha feita. É apresentado ao mesmo um *Pop-up* com o resultado obtido, tanto com o número de acertos, quanto com o número de erros. O usuário terá a opção de jogar novamente clicando no botão “Jogar Denovo” ou até mesmo voltar para o menu principal clicando no botão “Menu”. Na figura 30 a seguir, podemos observar essas ações sendo executadas e a aparência da tela de resultados.

Figura 30 - Tela de Resultados

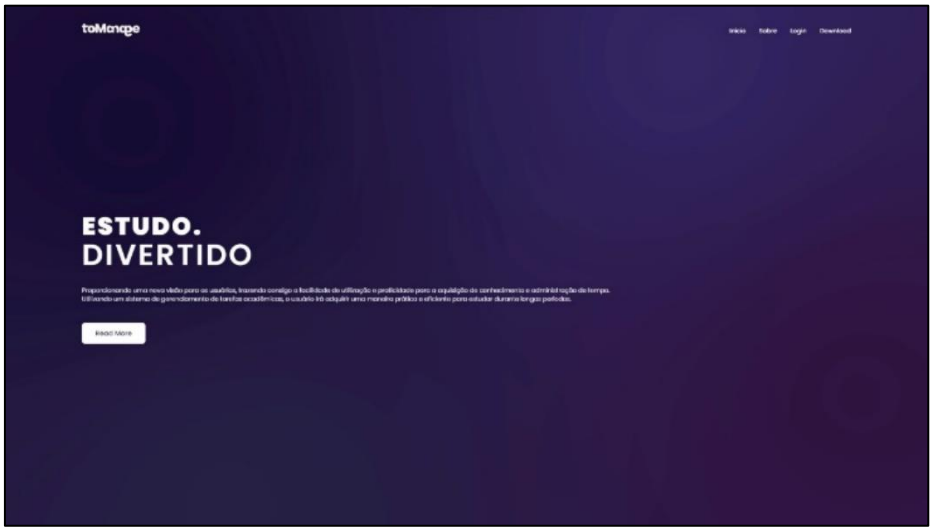
Fonte: Autoria Própria, 2022.

Essas são as funcionalidades da aplicação *mobile* toManage. O usuário poderá conhecer mais afundo como ela funciona instalando a aplicação em seu *smartphone* e assim desfrutar o melhor gerenciamento de tempo relacionado aos estudos.

3.2 Website

Para a divulgação da aplicação *mobile* foi desenvolvido um *website,* onde o usuário consegue obter mais informações sobre os desenvolvedores, conhecer ainda mais o projeto toManage e obter todo suporte necessário. Na figura 31 a seguir podemos observar a estrutura do *site* no computador e suas funcionalidades.

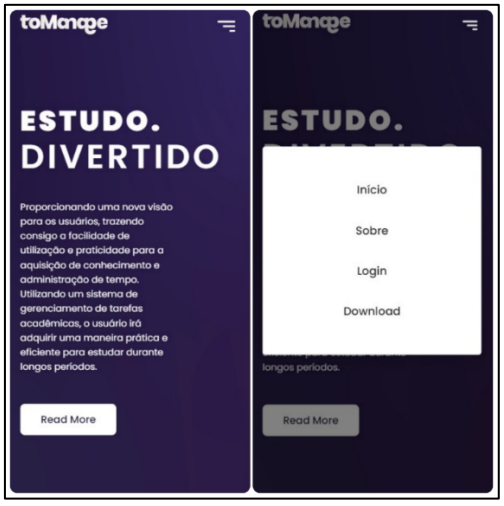
Figura 31 - Website no Computador



Fonte: Autoria Própria, 2022.

A visualização do *website* para celular foi desenvolvida de uma forma responsiva utilizando Bootstrap, para assim parte do *website* não ser corrompida, perdendo a qualidade do *software*. Na figura 32 a seguir é possível observar o *website* em um *smartphone*.

Figura 32 - Website no Smartphone



Fonte: Autoria Própria, 2022.

Ao utilizar o *site* o usuário pode escolher a opção de *menu*, onde encontra mais guias para navegar: “Início” – retorna à página inicial do *site*; “Sobre” – o usuário obtém informações sobre o projeto toManage; “Login” – para entrar em sua conta; “Download” – para fazer o *download* da aplicação *mobile*, como podemos observar na figura 32 acima. O principal objetivo da criação do *website* foi a divulgação da aplicação para assim não restarem possíveis dúvidas do funcionamento da aplicação *mobile*.

4 CONCLUSÃO

A partir dos resultados encontrados neste projeto, concluímos que as pessoas conseguem obter um melhor gerenciamento de tempo relacionado aos estudos com um auxílio de uma aplicação que atenda às necessidades de uma forma simples e intuitiva. Recomenda-se que sejam feitos estudos longitudinais que avaliem de maneira prospectiva se as pessoas ao usar a aplicação toManage serão mais produtivas nos setores acadêmico e pessoal nos meses seguintes.

REFERÊNCIAS

ARAÚJO, Marcus de Souza. **EaD em tela: docência, ensino e ferramentas digitais.** Revista Brasileira de Linguística Aplicada, v. 14, p. 735-741, 2014.

BOOCH, Grady. **UML:** guia do usuário. Elsevier Brasil, 2006.

CARMO, Bárbara Kéfere do et al. **A trilha Pomodoro.**

CAROTENUTO, Felipo Maluf; PEREIRA, Otaviano José. **Professores, metodologias ativas e a EAD:** uma proposta prática da inversão da sala de aula utilizando a Pirâmide de William Glasser. Setor Educacional: Educação Média e Tecnológica, Educação Superior. Tipo: Investigação científica (IC). Natureza: Planejamento de pesquisa. Categoria: Conteúdos e Habilidades. Uberaba/MG, 2020.

CARRIL, Marly. **HTML-Passo a Passo.** Clube de Autores, 2012.

CHAVES, Eduardo OC. **Administrar o tempo é planejar a vida.** Rio de Janeiro: Aymará, 1998.

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

COSTA, Carlos J. **Desenvolvimento para web.** ITML press/Lusocredito, 2007.

CUNHA, André. **React Native:** o que é e tudo sobre o Framework. [S. l.], 20 out. 2022. Disponível em: https://www.alura.com.br/artigos/react-native#:~:text=React%20Native%20(também%20conhecido%20como,a%20mesma%20base%20de%20código. Acesso em: 4 dez. 2022.

DA SILVA CRUZ, Vitor; PETRUCELLI, Erick Eduardo; SOTTO, Eder Carlos Salazar. **A LINGUAGEM JAVASCRIPT COMO ALTERNATIVA PARA O DESENVOLVIMENTO DE APLICAÇÕES MULTIPLATAFORMA.** Revista Interface Tecnológica, v. 15, n. 2, p. 39-49, 2018.

DA SILVA, Fábio Luiz; MUZARDO, Fabiane Tais. **Pirâmides e cones de aprendizagem:** da abstração à hierarquização de estratégias de aprendizagem. Dialogia, n. 29, p. 169-179, 2018.

DA SILVA, Leandro Saggiomo et al. **Formação Continuada em Educação a Distância:** Percepções sobre as competências na atuação do professor tutor. EmRede-Revista de Educação a Distância, v. 3, n. 2, p. 252-265, 2016.

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

DAMASCENA, Samia Carine Castro et al. **Uso de tecnologias educacionais digitais como ferramenta didática no processo de ensino-aprendizagem em enfermagem.** Brazilian Journal of Development, v. 5, n. 12, p. 29925-29939, 2019.

DE OLIVEIRA, Elaine Cristina Rocha. **A Técnica Pomodoro (O Pomodoro).**

DUHIGG, Charles. **O poder do hábito:** por que fazemos o que fazemos na vida e nos negócios. Objetiva, 2012.

FLANAGAN, David. **JavaScript:** o guia definitivo. Bookman Editora, 2004.

FLATSCHART, Fábio. **HTML 5-Embarque Imediato.** Brasport, 2011.

GIBB, Lygia Sabbag Fares. **A tendência de despadronização da jornada de trabalho:** configuração no Brasil e impacto nas mulheres. Unpublished doctoral thesis, UNICAMP, Campinas, 2017.

GUEDES, Gilleanes TA. **UML 2-Uma abordagem prática.** Novatec Editora, 2018.

HEBERLE, Moisés. **Ferramenta para prototipação de interfaces e apoio ao mapeamento de requisitos de sistema.** 2017. Trabalho de Conclusão de Curso.

JAMES, William. **Massa de Hábitos.** Matsumota, 1892.

JOBSTRAIBIZER, Flávia. **Criação de sites com o CSS.** Universo dos Livros Editora, 2009.

LARMAN, Craig. **Utilizando UML e padrões.** Bookman Editora, 2000.

MATOS, ECIVALDO DE SOUZA; ZABOT, Diego. **Aplicativos com Bootstrap e Angular Como Desenvolver Apps Responsivos:** Como Desenvolver Apps Responsivos. Saraiva Educação SA.

MELLO, Anna Carolina; SOUZA, Luiz Henrique Gomes de. **Solução Simplificada para o Monitoramento e Controle de Projetos Utilizando a Ferramenta Trello.** Boletim do Gerenciamento, [S.l.], v. 2, n. 2, out. 2018. ISSN 2595-6531. Disponível em: <https://nppg.org.br/revistas/boletimdogerenciamento/article/view/35>. Acesso em: 04 dez. 2022.

MEZZARI, Lucas Torres; LEAL, Eduardo Henrique Viva; VIEGAS, Silvio Cesar. **INTERNET DAS COISAS:** Arduino. Firebase e Android. REFAQI-Revista Eletrônica em Gestão e Tecnologia, v. 5, n. 1, p. 50-55, 2019.

MILETTO, Evandro Manara; DE CASTRO BERTAGNOLLI, Silvia. **Desenvolvimento de Software II:** Introdução ao Desenvolvimento Web com HTML, CSS, JavaScript e PHP-Eixo: Informação e Comunicação-Série Tekne. Bookman Editora, 2014.

MOURA, Dionatan. **O Mantra da Produtividade:** aprimore sua produtividade com técnicas de foco e organização pessoal. Editora Casa do Código, 2016.

MURPHY, Joseph. **O poder do subconsciente.** Editora Best Seller, 2021.

NEVES, Jonathan; JUNIOR, Vilmar Mendes. **Uma análise comparativa entre flutter e react native como frameworks para desenvolvimento híbrido de aplicativos mobile:** Estudo de caso visando produtividade. Ciência da Computação-Tubarão, 2020.

NIEDERAUER, Juliano. **PHP para quem conhece PHP.** Novatec Editora, 2017.

OLIVEIRA, George Moreno de. **Desenvolvimento e avaliação do plugin para o Figma para Documentação de Acessibilidade para Interfaces-DAI.** 2022.

PORTELA, Raiama Lima et al. **CORRESPONDÊNCIAS POR MEIO DE FERRAMENTAS DE DESIGN:** artesanato e empoderamento (ou aprisionamento?). 2018.

RANGEL, Guilherme Salum. **ProTool:** uma ferramenta de prototipação de software para o ambiente PROSOFT. 2003.

ROCK CONTENT, Redator. **Conheça Firebase:** a ferramenta de desenvolvimento e análise de aplicativos mobile: Diante do incremento significativo do valor do mercado de aplicativos digitais nos últimos anos, surgem problemas que devem ser resolvidos rapidamente, ao ritmo da sua evolução.. [S. l.], 21 ago. 2019. Disponível em: https://rockcontent.com/br/blog/firebase/. Acesso em: 4 dez. 2022.

RODRIGUES, Alexsandro Sutil et al. **Gestão do tempo aplicada à produtividade, qualidade de vida e desempenho:** análise de publicações do banco de dados da CAPES e do Google Acadêmico. In: Congresso Internacional de Administração Sucre: UEPG. 2018.

SANTOS JÚNIOR, William Pereira dos; OLIVEIRA, Atirson Fabiano Barbosa; ASSIS, Iago Gonçalves de. **As vantagens e desvantagens do React Native comparado ao Kotlin.** 2021.

SILVA, Ludmila. **Notion:** o que é e como essa ferramenta pode ajudar sua empresa na prática?. [S. l.], 24 jun. 2022. Disponível em: https://pluga.co/blog/notion-o-que-e/#:~:text=O%20Notion%20é%20uma%20ferramenta,pessoal%20quanto%20para%20a%20profissional. Acesso em: 4 dez. 2022.

SILVA, Maurício Samy. **Construindo sites com CSS e (X) HTML:** sites controlados por folhas de estilo em cascata. Novatec Editora, 2007.

SILVA, Maurício Samy. **JavaScript-Guia do Programador:** Guia completo das funcionalidades de linguagem JavaScript. Novatec Editora, 2020.

SILVA, Werliton Carlos Sousa da. **Aplicações móveis nativas com react native e firebase:** um estudo de caso. 2018.

SOLOMON, Karen. **Para que serve o Trello? Explicação do software de gerenciamento de projetos favorito.** [S. l.], 23 jun. 2022. Disponível em: https://blog.trello.com/br/para-que-serve-o-trello. Acesso em: 4 dez. 2022.

SOUZA, Leonardo Patrocinio; DO ESPÍRITO SANTO, Felipe. **COMPARATIVO ENTRE FRAMEWORKS DE CSS BOOTSTRAP E BULMA PARA DESENVOLVIMENTO DE PROJETOS WEB.** Revista Interface Tecnológica, v. 17, n. 1, p. 140-152, 2020.

TOMAZINI, Marcos; LOPES, Luiz Fernando Braga. **Web design responsivo-Bootstrap.**