



**UNIVERSIDADE
FEDERAL DO CEARÁ**
CAMPUS DE QUIXADÁ

CURSO DE ENGENHARIA DE SOFTWARE

**RELATÓRIO – PROJETO INTEGRADO II
MAISELF**

Equipe:

Eric Rodrigues - 470296

Gustavo Colombo - 470168

Ítalo Lima - 470651

Marcos Gênesis - 471549

Professor:

Camilo Camilo Almendra

QUIXADÁ

Maiο, 2021

SUMÁRIO

1	INTRODUÇÃO	2
2	SISTEMAS RELACIONADOS	2-6
2.1	Evernote	2-3
2.2	7 Weeks	3-4
2.3	Notion	4-5
2.4	Mobilis	5-6
3	DESCRIÇÃO DO SISTEMA	6
3.1	DIFERENCIAL DO NOSSO SISTEMA	7
4	MODELO DE FEATURES	7-8
5	REQUISITOS	8-23
5.1	Requisitos Funcionais	8-15
5.2	Requisitos Não Funcionais	16-24
6	PROTOTIPAÇÃO DO PROJETO	25
6.1	Fluxo de Navegação	25
7	FRAMEWORKS, LINGUAGENS DE PROGRAMAÇÃO E ARQUITETURA DE SOFTWARE UTILIZADOS NO PROJETO	26
8	BANCO DE DADOS	27
8.1	Esquema de dados	27-28
8.2	Padrão de acesso de dados	28-29
9	TESTES	29-31

1 INTRODUÇÃO

Um tema que sempre cerca as pesquisas periféricas dos adolescentes e jovens adultos nos últimos anos é o mundo do desenvolvimento pessoal. Autores como Jordan Patterson, David Goggins e Tony Robbins inflamaram esse tema com seus livros lançados nos últimos 3 anos. Observando o mercado de aplicativos mobile e web voltados a esse tema, é possível notar uma situação curiosa: a grande maioria destes, tem como foco sempre um subtema dessa categoria. Por exemplo, um subtema que existem bastantes opções no mercado de aplicativos é gerenciador financeiro, com grandes nomes no mercado como Mobills, Organizze, entre outros. Para hábitos existem bastante opções também, assim como para a organização diária, como Todoist, Notion, Trello e etc. Porém é nítido o espaço existente para uma aplicação que centralize esses dados em uma só plataforma, assim como o Rappi fez para o mercado de delivery no Brasil e América Latina. Observando esta oportunidade, decidimos então criar a Maiself - Alusão a palavra *myself* em inglês, que pode significar “eu”.

O projeto Maiself visa criar uma base centralizada de informações, para que seja possível visualizar de maneira mais clara, objetiva e organizada o processo de desenvolvimento pessoal de nossos usuários, respeitando o tempo de cada um, para uma vida mais organizada, produtiva e feliz.

2 SISTEMAS RELACIONADOS

Nesta seção serão descritos 3 sistemas que encontramos em pesquisa e disponíveis na internet, que se aproximam ao máximo do domínio do projeto com objetivos do sistema, possíveis funcionalidades e tecnologias.

2.1 Evernote

Sistema para anotação de tarefas/notas: Nesse sistema, o usuário pode escrever suas notas sem ficar restrito ao acesso desktop, pode baixar uma versão mobile da solução e acessar o que escreveu e ainda anotar o que desejar, fora navegar entre blogs, sites, guardar documentos.

Funcionalidades:

1. CRUD notas/anotações.

2. Cadastro de lembretes dos itens anotados.
3. Cadastro de etiquetas dos itens anotados.
4. Criar um ‘caderno’ para criar notas - deixar mais organizado suas notas.
5. Upload de anexos a notas como imagens, vídeos, áudios.
6. Criar listas

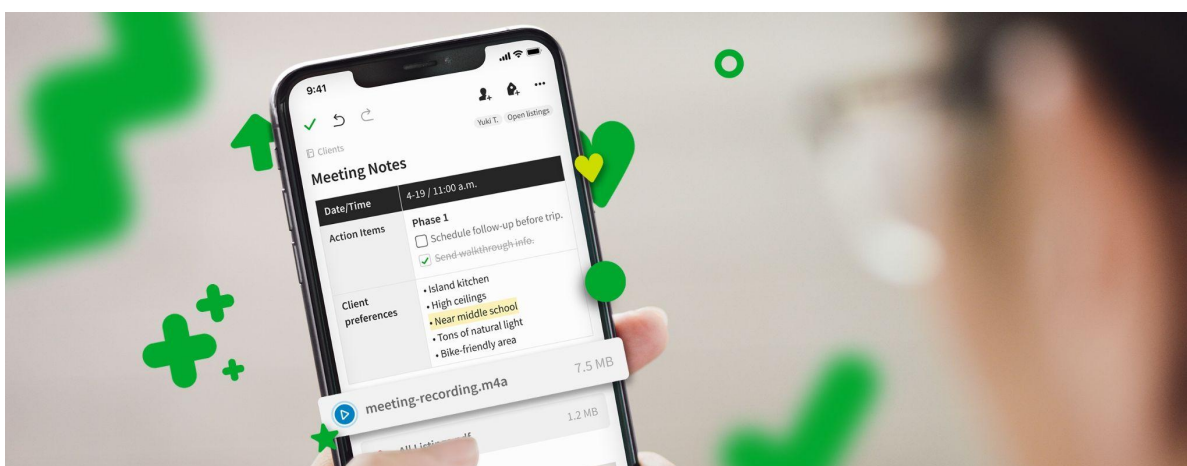


Figura 1 – Tela do Evernote

Fonte: [Link para o Evernote](#)

2.2 7 Weeks:

Aplicativo para gerenciamento/criação de novos hábitos: O aplicativo disponível de forma gratuita para Android é para a criação de novos hábitos, atos que são repetidos até que tornam-se regra na rotina da pessoa. Estudiosos dizem que para a criação de um novo hábito temos que incorporá-lo durante 49 dias (7 semanas) até que ele seja costume. O aplicativo consiste em criar um novo hábito, o motivo por que você está criando ele e quando pretende terminá-lo e partir disso o aplicativo o mantém informado do progresso.

Funcionalidades:

1. Cadastro do hábito.
2. Cadastro da justificativa do hábito.
3. Cadastro da data limite do hábito para incorporá-lo
4. Visualização da estatística do progresso no hábito, com porcentagem, a partir dos dias feitos, da data limite e dos dias que o usuário não fez
5. Emitir alarmes para realizar o ato do hábito
6. Marcar os dias concluídos no calendário de realização do hábito

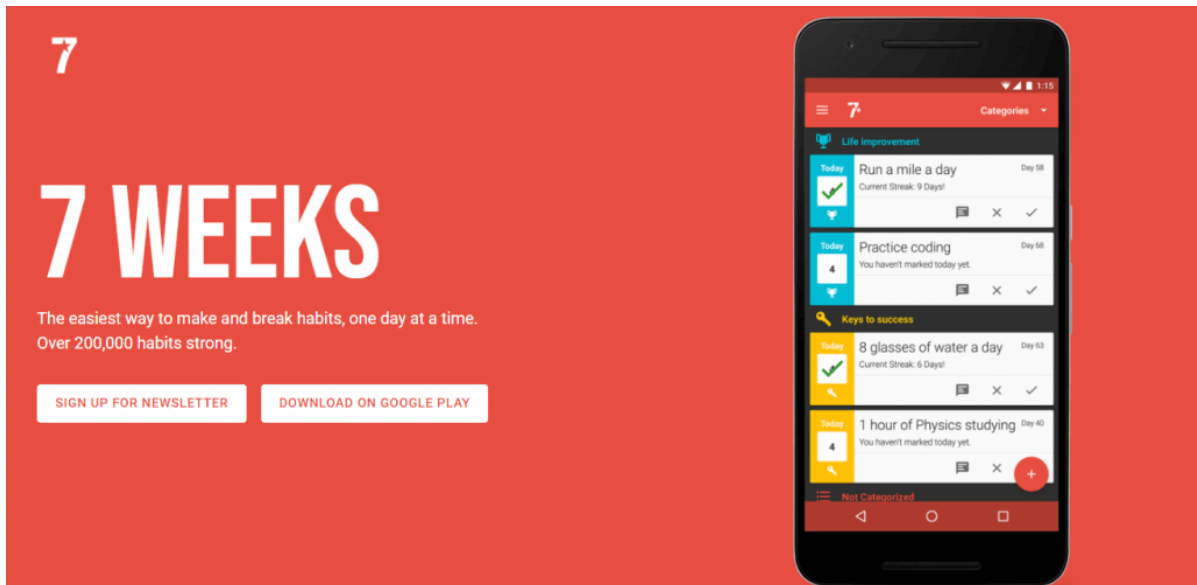


Figura 2 – Tela inicial do aplicativo 7 weeks.

Fonte: [Link para o 7 Weeks](#)

2.3 Notion

Sistema/aplicativo para anotações em geral: Esse sistema, bastante popular entre os programadores é disponível em forma de site, aplicativo móvel e aplicação desktop, ele serve para o gerenciamento e armazenamento de notas, textos em geral, basicamente tudo o que você anota está guardado pra sempre em infinitas páginas e em quantos workspaces quiser. Uma das coisas que o torna tão incrível é a sua alta integração com diversos meios de escrita, onde o mesmo dispõe integração de escrita com o *Markdown*, *LáTeX* e etc, tudo isso dentro da própria ferramenta. A mesma também dispõe de integração com o calendário, quadros *Kanban* e etc.

Funcionalidades:

1. **CRUD de workspaces.**
2. **CRUD de páginas dentro dos workspaces**
3. **CRUD de páginas dentro das páginas.**
4. **Anotações, listas, to-list, bloco de notas e muito mais podem ser cadastrados nas páginas.**
5. **Anexo de imagens as páginas, menções a outras páginas.**
6. **Compartilhar páginas com a web ou com quem apenas quiser**

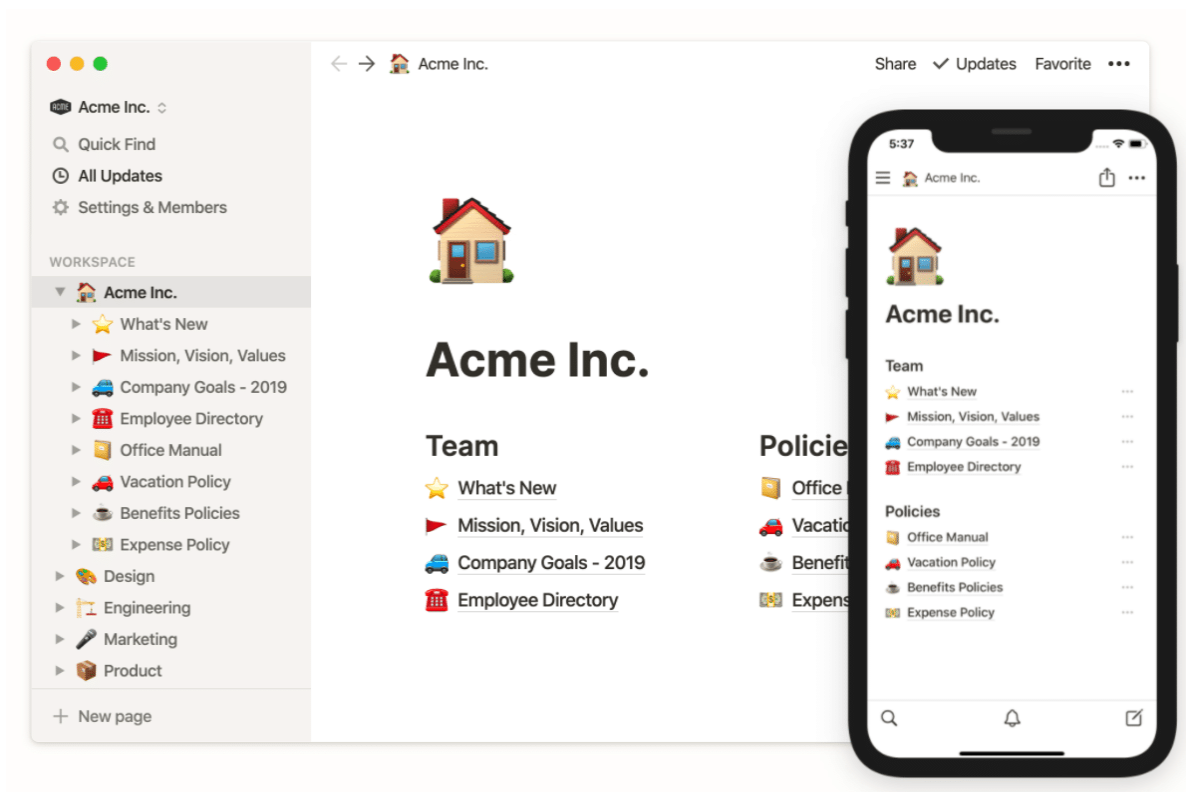


Figura 3 – Tela do Sistema e aplicativo Notion.

Fonte: [Link para o Notion](#)

Vídeo demonstrativo: [Demonstração Notion](#)

2.4 Mobills

Sistema/aplicativo para gerenciamento e organização financeira: Mobills é o aplicativo de gerenciamento de custos pessoais mais bem avaliado do país, mas com também disponibilidade no sistema web. O aplicativo é muito útil para quem deseja ter um maior controle de gastos, para quem deseja estabelecer teto de gastos ou apenas controlar seus gastos, com o montante que entra subdividido em categorias e o valor de cada entrada e o montante que sai subdividido em categorias também e com valores resultando no valor total da carteira.

Funcionalidades:

1. **Cadastrar Transações.**
2. **Classificar Transações como entrada ou saída**
3. **Classificar Transações de entrada/saída em categorias pré-prontas.**
4. **Nomear categorias para Transações.**

5. Dar valor inicial a carteira todo mês para ter o balanço no final do mês.
6. Criar objetivos para alcançar com a economia de dinheiro.



Figura 4 – Tela do Sistema Mobills.

Fonte: [Link para a Mobills](#)

3 DESCRIÇÃO DO SISTEMA

O sistema proposto pela nossa equipe é denominado de **Maiself** e propõe como solução principal a integração de funcionalidades divididas em distintos módulos e presente em variadas aplicações no mercado, ou seja, a proposta é fornecer ao usuário final uma plataforma web que o proporcione o total controle das principais áreas da sua vida. através do gerenciamento dos aspectos que as compõem, como a organização financeira para o domínio das finanças, o acompanhamento dos hábitos para a rotina individual, a utilização de ferramentas para a automatização e apoio nas atividades do seu dia-a-dia, registros referentes à evolução e cuidado com a sua saúde e auxiliares para memorização, controle e documentação de tarefas, compromissos e lembretes, através de um histórico completo que funciona como um diagnóstico de atividades, no qual pode ser visto em diferentes perspectivas, textuais e gráficas, tudo isso garante uma integridade dos “dados” resultantes de tudo que você faz, todos os dias, desde aquela pedalada na terça-feira pela manhã, passando pelo check-list de atividades diárias que você tem a cumprir no trabalho, até o acompanhamento do seu IMC no último período de 6 meses, que pode indicar que você perdeu peso, quando decidiu economizar nos lanches noturnos que costumava pedir na sua lanchonete preferida, todas as nossas ações têm como resultado (direto ou indireto) a produção de dados e ter o

versionamento, a visualização, o registro e à aquisição deles pode ser crucial para aumentar a eficiência na busca pelo controle e equilíbrio da própria existência.

3.1 Nosso Diferencial

Como anteriormente citado, nosso objetivo tem como unir diferentes módulos de organização pessoal em uma única aplicação web, esse é um dos nossos diferenciais dos sistemas apresentados anteriormente também. Outro diferencial é que para soluções web não há sistemas de controle de hábitos ou de controle de finanças(não de graça pelo menos), então estamos trazendo dois segmentos de controle pessoal que quase toda pessoa hoje em dia no mundo da tecnologia visa, podendo dentro do sistema navegar entre esses módulos e usar qual quiser totalmente de graça.

4 Modelo de Features

- **Cadastro, autenticação, alteração e remoção do usuário.**
 - Cadastro do usuário na plataforma.
 - Confirmação de conta do usuário cadastrado na plataforma.
 - Login do usuário na plataforma.
 - Edição das informações do usuário na plataforma.
 - Exclusão da conta do usuário na plataforma.
- **Cadastro, listagem, remoção e edição de hábitos do usuário.**
 - Definição de hábitos do usuário, na plataforma.
 - Listagem dos hábitos do usuário, cadastrados na plataforma.
 - Edição dos hábitos do usuário, cadastrados na plataforma.
- **Criação, edição e remoção de lembretes para os hábitos do usuário.**
 - Definição de lembretes do usuário, para os hábitos cadastrados na plataforma.
 - Edição dos lembretes do usuário, para os hábitos cadastrados na plataforma.
- **Cadastro e edição de informações referentes ao corpo do usuário.**
 - Cadastro do peso do usuário.
 - Cadastro da altura do usuário.
 - Cálculo mensal do IMC, a partir das informações de peso e altura, cadastradas na plataforma.
- **Cadastro, edição e remoção das entradas e saídas (finanças) do usuário.**

- Cadastro das entradas (do mês, trimestre, semestre ou ano) financeiras do usuário.
- Edição das entradas financeiras do usuário.
- Remoção das entradas financeiras do usuário.
- Produção do balanço (mensal, trimestral, semestral ou anual) das finanças do usuário.
- **Criação de notas (bloco de notas digital) do usuário.**
 - Registro de atividades, pensamentos, anotações importantes, etc.
- **Criação e edição de check-lists de atividades.**
- **Produção de gráficos de estabilidade estabelecida para a assiduidade dos hábitos do usuário.**
 - Cálculo a partir da fórmula de estabilidade.
 - Heat-Map para a representação da estabilidade.
 - Gráfico de frequência de atividades.

5 REQUISITOS

Requisitos de software podem ser descritos brevemente como a interseção entre uma necessidade real de alguém e a documentação estratégica que possibilita a implementação da funcionalidade. O processo começa sempre com um problema, uma necessidade, uma ideia ou até mesmo uma melhoria necessária, alguém tem isso definido no campo da ideia e um dia, “magicamente” tal pensamento vira parte de artefatos produzido pelo processo de desenvolvimento de software, seja de forma documental ou em uma linguagem de programação, por isso é tão importante entender e buscar realizar com assertividade as operações referentes à requisitos.

Aqui estão dispostos e documentados os Requisitos do Maiself, sendo eles divididos entre funcionais e não-funcionais, com a expectativa de todos estarem idealizados e descritos da melhor maneira possível, visando agregar o maior valor possível ao cliente, que é o usuário final da aplicação.

5.1 Requisitos Funcionais

[RF01] - O usuário poderá realizar seu cadastro no sistema

Identificador	[RF01] - O usuário poderá realizar seu cadastro no sistema
Descrição	A partir dos dados pessoais (Nome, sobrenome, email, senha, data de nascimento, altura e peso) requisitados no

	formulário de cadastro, o sistema deve enviar um email de confirmação para o endereço de email fornecido.
Prioridade	Essencial.

Tabela 1 – Tabela do Requisito Funcional 01.

[RF02] - Confirmação de conta do usuário que preencheu o formulário de cadastro, na plataforma.

Identificador	[RF02] - Confirmação de conta do usuário que preencheu o formulário de cadastro, na plataforma.
Descrição	O usuário deve confirmar o cadastro na plataforma, através do acesso ao token enviado no endereço de email fornecido e inserir suas informações no banco de dados.
Prioridade	Essencial.

Tabela 2 – Tabela do Requisito Funcional 02.

[RF03] - O usuário poderá efetuar login no sistema, passando os dados de Email e Senha

Identificador	[RF03] - O usuário poderá efetuar login no sistema, passando os dados de Email e Senha
Descrição	O usuário deve fornecer o email e senha que usou no cadastro, no formulário de login, dado à validade desses dados (conta ativa e dados compatíveis), o sistema deve fornecer o acesso à plataforma.
Prioridade	Essencial.

Tabela 3 – Tabela do Requisito Funcional 03.

[RF04] - O usuário poderá editar suas informações pessoais no sistema

Identificador	[RF04] - O usuário poderá editar suas informações pessoais no sistema.
Descrição	A partir do momento que o usuário possui sua conta confirmada e está logado na aplicação, ele deve poder editar as informações do seu perfil.
Prioridade	Essencial

Tabela 4 – Tabela do Requisito Funcional 04.

[RF05] - O usuário poderá excluir sua conta a partir do momento que não estiver mais satisfeito com o site ou desejar apenas excluir a conta

Identificador	[RF05] - O usuário poderá excluir sua conta a partir do momento que não estiver mais satisfeito com o site ou desejar apenas excluir a conta.
Descrição	A partir do momento que o usuário possui sua conta confirmada e está logado na aplicação, ele deve ser capaz de apagar sua conta, feito isso, o sistema deve remover suas informações do banco de dados.
Prioridade	Essencial

Tabela 5 – Tabela do Requisito Funcional 05.

[RF06] - O usuário deverá ser capaz de cadastrar os seus hábitos definidos, na plataforma.

Identificador	[RF06] - O usuário deverá ser capaz de cadastrar os seus hábitos definidos, na plataforma.
Descrição	O usuário deve poder cadastrar hábitos referentes à sua rotina, preenchendo as suas informações (nome, descrição, cor da etiqueta de importância, frequência, pontuação de relevância).
Prioridade	Essencial

Tabela 6 – Tabela do Requisito Funcional 06.

[RF07] - O usuário deverá ser capaz de visualizar a listagem de todos os seus hábitos definidos, na plataforma.

Identificador	[RF07] - O usuário deverá ser capaz de visualizar a listagem de todos os seus hábitos definidos, na plataforma.
Descrição	O usuário deverá ser capaz de verificar todos os seus hábitos cadastrados na plataforma, como também as informações referentes a eles.
Prioridade	Essencial

Tabela 7 – Tabela do Requisito Funcional 07.

[RF08] - O usuário deverá ser capaz de editar os seus hábitos cadastrados na plataforma.

Identificador	[RF08] - O usuário deverá ser capaz de editar os seus hábitos cadastrados na plataforma.
Descrição	O usuário deverá ser capaz de editar os seus hábitos cadastrados na plataforma, alterando as informações referentes a eles.
Prioridade	Essencial

Tabela 8 – Tabela do Requisito Funcional 08.

[RF09] - O usuário poderá priorizar os seus hábitos definidos.

Identificador	[RF08] - O usuário poderá priorizar os seus hábitos definidos.
Descrição	O usuário deverá ser capaz de estabelecer uma relação de prioridade entre os hábitos que ele definiu, aumentando assim a importância daqueles mais necessários.
Prioridade	Importante

Tabela 9 – Tabela do Requisito Funcional 09.

[RF10] - O usuário poderá cadastrar lembretes para os hábitos que definiu, na plataforma.

Identificador	[RF10] - O usuário poderá cadastrar lembretes para os hábitos que definiu, na plataforma.
Descrição	O usuário pode definir qual ou quais de seus hábitos cadastrados na plataforma, ele quer que a plataforma o lembre sobre os mesmos.
Prioridade	Importante

Tabela 10 – Tabela do Requisito Funcional 10.

[RF11] - O usuário poderá cadastrar informações referentes ao seu corpo.

Identificador	[RF11] - O usuário poderá cadastrar informações referentes ao seu corpo.
Descrição	O usuário deve ser capaz de realizar o cadastro de

	informações do seu corpo, no sistema a fim de que haja cálculos em determinados hábitos de determinado módulo, o de Hábitos. Informações que podem ser por exemplo de Peso, Altura.
Prioridade	Essencial

Tabela 11 – Tabela do Requisito Funcional 11.

[RF12] - O usuário poderá verificar o cálculo do IMC com os seus dados registrados de Peso e Altura

Identificador	[RF12] - O usuário poderá verificar o cálculo do IMC com os seus dados registrados de Peso e Altura
Descrição	O usuário uma vez que cadastrou seu peso e altura na plataforma poderá no módulo de hábitos verificar o cálculo de IMC seguindo a regra da OMS adotada aqui que é $IMC = peso / (altura * altura)$.
Prioridade	Essencial

Tabela 12 – Tabela do Requisito Funcional 12.

Requisitos referentes ao módulo financeiro

[RF13] - O usuário poderá cadastrar montantes de dinheiro na categoria de Entrada.

Identificador	[RF13] - O usuário poderá cadastrar montantes de dinheiro na categoria de Entrada.
Descrição	O usuário neste módulo terá como objetivo o controle e organização de gastos pessoais. Ele poderá cadastrar neste módulo uma quantidade de dinheiro na categoria Entrada que é o dinheiro de forma positiva representando suas entradas de dinheiro, seus ganhos.
Prioridade	Essencial

Tabela 13 – Tabela do Requisito Funcional 13.

[RF14] - O usuário poderá cadastrar montantes de dinheiro na categoria de Saída.

Identificador	[RF14] - O usuário poderá cadastrar montantes de dinheiro na categoria de Saída.
Descrição	O usuário neste módulo terá como objetivo o controle e organização de gastos pessoais. Ele poderá cadastrar neste módulo uma quantidade de dinheiro na categoria Saída que é o dinheiro de forma negativa, que representa os seus gastos.
Prioridade	Essencial

Tabela 14 – Tabela do Requisito Funcional 14.

[RF15] - O usuário poderá editar o valor das Entradas financeiras cadastradas

Identificador	[RF15] - O usuário poderá editar o valor das Entradas financeiras cadastradas.
Descrição	Uma vez que o usuário tiver uma entrada financeira cadastrada ele poderá alterar o seu valor para mais, ajustando sua entrada de dinheiro de acordo com a alteração do recebimento
Prioridade	Essencial

Tabela 15 – Tabela do Requisito Funcional 15.

[RF16] - O usuário poderá editar o valor das Saídas financeiras cadastradas

Identificador	[RF16] - O usuário poderá editar o valor das Saídas financeiras cadastradas.
Descrição	Uma vez que o usuário tiver uma saída financeira cadastrada ele poderá alterar o seu valor, para menos, ajustando sua entrada de dinheiro de acordo com a alteração do gasto.
Prioridade	Essencial

Tabela 16 – Tabela do Requisito Funcional 16.

[RF17] - O usuário poderá remover Entradas financeiras cadastradas

Identificador	[RF17] - O usuário poderá remover Entradas financeiras cadastradas.
----------------------	---

Descrição	O usuário quando tiver uma entrada financeira cadastrada e por algum motivo, seja por não ocorrer a entrada do dinheiro ou algo que o impeça de tê-lo, ou alguém pagou alguma conta por ele, poderá remover essa entrada dos cadastros para ter melhor organização.
Prioridade	Essencial

Tabela 17 – Tabela do Requisito Funcional 17.

[RF18] - O usuário poderá remover Saídas financeiras cadastradas

Identificador	[RF18] - O usuário poderá remover Saídas financeiras cadastradas.
Descrição	O usuário quando tiver uma saída financeira cadastrada e por algum motivo externo, para organização de suas finanças o usuário poderá remover a saída financeira
Prioridade	Essencial

Tabela 18 – Tabela do Requisito Funcional 18.

[RF19] - O usuário poderá verificar o balanço de suas finanças.

Identificador	[RF19] - O usuário poderá verificar o balanço de suas finanças.
Descrição	Com a soma das entradas positivas e a soma das entradas negativas(saídas), iremos subtrair o resultados das duas somas anteriores para obter o Balanço, que pode ser mensal por padrão, mas poderá ser trimestral, semestral ou anual.
Prioridade	Essencial

Tabela 19 – Tabela do Requisito Funcional 19.

Requisitos referentes à Organização pessoal**[RF20] - O usuário poderá cadastrar anotações de seu interesse, registrar suas atividades, pensamentos, etc...**

Identificador	[RF20] - O usuário poderá cadastrar anotações de seu interesse.
Descrição	O usuário dentro deste módulo cadastrar anotações, pensamentos, qualquer que seja o texto de seu interesse na opção de bloco de notas.

Prioridade	Importante
-------------------	------------

Tabela 20 – Tabela do Requisito Funcional 20.

[RF21] - O usuário poderá criar checklists de atividades.

Identificador	[RF21] - O usuário poderá criar checklists de atividades.
Descrição	O usuário poderá criar checklists para suas tarefas e marcá-las como concluídas ao longo do desenvolvimento delas.
Prioridade	Importante

Tabela 21 – Tabela do Requisito Funcional 21.

[RF22] - O usuário poderá editar checklists de atividades.

Identificador	[RF22] - O usuário poderá editar checklists de atividades.
Descrição	O usuário poderá editar os checklists que criou, caso haja a necessidade, evitando que se faça necessária a criação de um novo, no caso da ocorrência de mudanças.
Prioridade	Importante

Tabela 22 – Tabela do Requisito Funcional 22.

[RF23] - O usuário de acordo com sua assiduidade na realização dos hábitos deverá poder visualizar um gráfico que ilustre a sua estabilidade, de acordo com a fórmula definida pelas regras de negócio.

Identificador	[RF23] - O usuário de acordo com sua assiduidade na realização dos hábitos deverá poder visualizar um gráfico que ilustre a sua estabilidade, de acordo com a fórmula definida pelas regras de negócio.
Descrição	De acordo com a fórmula de estabilidade definida pelas regras de negócio e baseado na assiduidade do usuário em seus hábitos, o sistema deverá gerar diferentes gráficos(heat-map, gráfico de frequência) que ilustram esses resultados.
Prioridade	Essencial

Tabela 23 – Tabela do Requisito Funcional 23.

5.2 Requisitos Não Funcionais

Um requisito não-funcional é uma descrição ou característica que um sistema deve exibir, ou mesmo uma restrição que deve ser respeitada pelo mesmo.

Requisitos não-funcionais são atributos de qualidade para um software. Estes atributos podem ser classificados de diferentes formas. A abordagem deste trabalho foca em um exemplo de classificação binária, onde os atributos podem ser de qualidade externa ou interna.

“Atributos de qualidade diferenciam os sistemas que simplesmente fazem o suposto daqueles que encantam os seus usuários”

- Os atributos de qualidade externa são mais voltados para o usuário, e influenciam a experiência e percepções do mesmo, sobre a qualidade de um sistema.
 - São normalmente avaliados quando o sistema está em execução.
- Os atributos de qualidade interna são voltados para os desenvolvedores e mantenedores do sistema.
 - São normalmente percebidos nas etapas de design e codificação.

Atributos de qualidade interna impactam os de qualidade externa, por isso, apesar das diferenças, os dois tipos estão conectados e são cruciais para o sucesso de um software, algo que vai além de simplesmente fazer o que o cliente pediu.

GLOSSÁRIO

- **Definição dos atributos**
 - **Requisitos não-funcionais derivados**

Requisitos de Usabilidade

- **É a capacidade de quão facilmente o sistema pode ser compreendido, aprendido e atraente para quem usa, no caso o usuário. Esse RNF não se restringe a interfaces visuais mas a qualquer sistema que tenha comunicação ou interação com o usuário como um terminal do linux por exemplo e se subdivide em outras categorias.**
- **Requisitos Não-Funcionais Derivados:**

[RNF 01] - Os usuários deverão se familiarizar com a interface em pouco tempo (Média de 2 execuções de fluxo)

Identificador	[RNF 01] - Os usuários deverão se familiarizar com a interface em pouco tempo (Média de 2 execuções de fluxo).
----------------------	--

Descrição	Os usuários que irão usar o Maiself deverão em pouco tempo de uso aprender/compreender o fluxo de execução e como medida de tempo usamos a média de 2 vezes cada execução de fluxo dentro do sistema.
Prioridade	Essencial

Tabela 27 – Tabela do Requisito não Funcional 01.

[RNF 02] - Os usuários não precisarão de treinamento prévio para compreender o fluxo de execução do Maiself

Identificador	[RNF 02] - Os usuários não precisarão de treinamento prévio para compreender o fluxo de execução do Maiself.
Descrição	O objetivo da interface e do sistema é ser o mais simples e descritivo possível, com isso não precisaremos obrigar o usuário a ter conhecimento aprofundado de tecnologia para usar o sistema, apenas seguir um fluxo de execução simples e descritivo.
Prioridade	Essencial

Tabela 28 – Tabela do Requisito não Funcional 02.

[RNF03] - A interface deve possuir design simples e *Flat* para melhor familiarização dos usuários

Identificador	[RNF03] - A interface deve possuir design simples e <i>Flat</i> para melhor familiarização dos usuários
Descrição	Para que o fluxo de atividades e interface sejam simples e descritivos, por convenção, usaremos o design Flat que busca o design minimalista, priorizando mais a informação em tela e mantendo a simplicidade evitando a poluição visual.
Prioridade	Essencial

Tabela 29 – Tabela do Requisito não Funcional 03.

[RNF 04] - Menus existentes dentro do Maiself não devem possuir mais que um nível de ramificação interna

Identificador	[RNF 04] - Menus existentes dentro do Maiself não devem possuir mais que um nível de ramificação interna.
Descrição	Para que não haja poluição visual nem confusão de informações para o usuário, definimos que caso haja menus eles não podem ter mais de um nível de ramificação interna para não poluir a tela de informações nem armazenar muitas informações em um só lugar.
Prioridade	Importante

Tabela 30 – Tabela do Requisito não Funcional 04.

Requisitos de Eficiência

- **Esse tipo de RNF se refere ao desempenho do sistema desenvolvido, se o tempo de execução/resposta e recursos são compatíveis com o que foi desenvolvido, este também se divide em categorias;**
- **Requisitos Não-Funcionais Derivados:**

[RNF05] - O sistema Maiself deverá tratar dados como Altura e Peso de usuários em ponto flutuante com maior precisão possível de no máximo 5 casas decimais após a vírgula

Identificador	[RNF05] - O sistema Maiself deverá tratar dados como Altura e Peso de usuários em ponto flutuante com maior precisão possível de no máximo 5 casas decimais após a vírgula.
Descrição	Os cálculos dentro do sistema ou armazenamento de informações que sabe-se que são números não inteiros devem ser guardados do tipo numérico float com precisão de 5 casas decimais.
Prioridade	Essencial

Tabela 31 – Tabela do Requisito não Funcional 05.

[RNF 06] - Os cálculos de IMC de cada usuário deverão ser feitos com base na fórmula disponibilizada pela Organização Mundial da Saúde (OMS)

Identificador	[RNF 06] - Os cálculos de IMC de cada usuário deverão ser feitos com base na fórmula disponibilizada pela Organização Mundial da Saúde (OMS).
Descrição	O cálculo de IMC que irá haver dentro do Maiself no

	módulo de hábitos por exemplo seguirá a fórmula da OMS que é IMC = peso/(altura * altura) .
Prioridade	Essencial

Tabela 32 – Tabela do Requisito não Funcional 06.

[RNF 07] - As faixas indicativas de Índice de Massa Corpórea devem também seguir as medidas da OMS

Identificador	[RNF 07] - As faixas indicativas de Índice de Massa Corpórea devem também seguir as medidas da OMS.
Descrição	Quando houver cálculo de IMC para os usuários, eles serão alocados em faixas que os representarão, de acordo com sua situação corpórea: Baixo peso muito grave = abaixo de 16 kg/m². Baixo peso grave = entre 16 e 16,99 kg/m². Baixo peso = entre 17 e 18,49 kg/m². Peso normal = entre 18,50 e 24,99 kg/m². Sobrepeso = entre 25 e 29,99 kg/m². Obesidade grau I = entre 30 e 34,99 kg/m². Obesidade grau II = entre 35 e 39,99 kg/m². Obesidade grau III (obesidade mórbida) = maior que 40 kg/m².
Prioridade	Essencial

Tabela 33 – Tabela do Requisito não Funcional 07.

[RNF 08] - O maiself deve ser capaz de tratar chamadas simultâneas a rotas diferentes em sua API

Identificador	[RNF 08] - O maiself deve ser capaz de tratar chamadas simultâneas a rotas diferentes em sua API.
Descrição	O front-end do Maiself deverá se integrar com o back-end em formato API REST e deve garantir que as requisições que serão feitas ao sistema deverão ser executadas simultaneamente com outros usuários que também fizeram requisições.
Prioridade	Essencial

Tabela 34 – Tabela do Requisito não Funcional 08 .

Requisitos de Confiabilidade

- O Requisito de Confiabilidade refere-se a se manter em um nível de desempenho específico quando posto em situações de desempenho específicas, suas subdivisões são: Tolerância a Falhas, Recuperabilidade, Conformidade e outros;
- Requisitos Não-Funcionais Derivados:

[RNF 09] - O sistema deve estar disponível 97% do tempo, 24 horas por dia , 7 dias por semana

Identificador	[RNF 09] - O sistema deve estar disponível 97% do tempo, 24 horas por dia , 7 dias por semana.
Descrição	O sistema deve ser capaz de ficar no ar em média 97% do tempo de um calendário mensal, fora os períodos de manutenção e melhorias, isso são 24 horas por dia o dia que ficar no ar e 7 dias as semanas que fica no ar o sistema, ou seja, o quanto ele para resulta no percentual de 3%.
Prioridade	Importante

Tabela 35 – Tabela do Requisito não Funcional 09.

[RNF 10] - O sistema deve entrar em manutenção nos períodos de noite/madrugada. Precisamente das 00:00 AM às 03:30 AM todo final de mês entre dia 28 e 30

Identificador	[RNF 10] - O sistema deve entrar em manutenção nos períodos de noite/madrugada. Precisamente das 00:00 AM às 03:30 AM.
Descrição	As manutenções do sistema devem ocorrer todo final de mês com a finalidade de correção de bugs pequenos, caso não tenham sido descoberto bugs críticos. Caso as duas alternativas sejam inexistentes, será usado o tempo para implementação de melhorias no sistema. Esse período é o que estimamos que a minoria dos usuários seriam prejudicados com a parada do sistema.
Prioridade	Essencial

Tabela 36 – Tabela do Requisito não Funcional 10.

Requisitos de Portabilidade

- É a facilidade que um sistema tem de ser transferido para outro ambiente computacional sem sofrer alterações, ou seja, poder rodar em ambientes distintos.
- **Requisitos Não-Funcionais Derivados:**

[RNF 11] - O sistema deve executar em qualquer navegador que suporte as mais recentes atualizações do ECMAScript

Identificador	[RNF 11] - O sistema deve executar em qualquer navegador que suporte as mais recentes atualizações do ECMAScript.
Descrição	O sistema deve executar em qualquer navegador que suporte as mais recentes atualizações do ECMAScript e nas mais antigas também.
Prioridade	Importante

Tabela 37 – Tabela do Requisito não Funcional 11.

Requisitos de Implementação

- São os requisitos que delimitam o escopo quanto à questão da implementação, seja através da especificação de tecnologias, definição do banco de dados, frameworks, etc.
- **Requisitos Não-Funcionais Derivados:**

[RNF 12] - O back-end que compõe a API do Maiself deve ser implementado na tecnologia Nest.Js

Identificador	[RNF 12] - O back-end que compõe a API do Maiself deve ser implementado na tecnologia NestJS.
Descrição	O back-end que o front-end irá consumir terá de ser construído, todas as regras de negócio e implementações com a tecnologia e especificações de NestJS.
Prioridade	Importante

Tabela 38 – Tabela do Requisito não Funcional 12.

[RNF 13] - A documentação da API do Maiself deverá seguir os padrões e ser documentada em Swagger

Identificador	[RNF 13] - A documentação da API do Maiself deverá seguir os padrões e ser documentada em Swagger.
Descrição	A API do Maiself após concluída deverá ser documentada na plataforma Swagger, onde será

	documentado cada endpoint consumível e disponível da API e o corpo da Requisição.
Prioridade	Importante

Tabela 39 – Tabela do Requisito não Funcional 13.

[RNF 14] - O banco de dados utilizado no back-end da API será o PostgreSQL e usará o ORM TypeOrm para ajudar na construção do back-end

Identificador	[RNF 14] - O banco de dados utilizado no back-end da API será o PostgreSQL e usará o ORM TypeOrm para ajudar na construção do back-end.
Descrição	O banco de dados que será utilizado na nossa implementação será o PostgreSQL em conjunto com o TypeOrm para realizar queries no banco de dados em formato JavaScript.
Prioridade	Importante

Tabela 40 – Tabela do Requisito não Funcional 14.

[RNF 15] - O front-end que compõe a interface UI/UX do Maiself deve ser implementado na tecnologia Next.Js + React

Identificador	[RNF 15] - O front-end que compõe a interface UI/UX do Maiself deve ser implementado na tecnologia Next.js + React.
Descrição	O front-end deve ser implementado na tecnologia Next.js por opção ser uma opção que potencializa os sites com React e facilita no SEO.
Prioridade	Importante

Tabela 41 – Tabela do Requisito não Funcional 15.

[RNF 16] - Os testes de unidade do Maiself deverão ser produzidos e testados com Jest

Identificador	[RNF 16] - Os testes de unidade do Maiself deverão ser produzidos e testados com Jest.
Descrição	Os testes de unidade do Maiself deverão ser produzidos e testados com Jest, sendo responsabilidade de quem programa.
Prioridade	Importante

Tabela 42 – Tabela do Requisito não Funcional 16.

Requisitos de Padrões

- Esses requisitos determinam quais padrões serão utilizados no desenvolvimento do sistema e na definição do projeto. Então, referem-se às formas utilizadas de acordo com a metodologia adotada, a forma de programação, etc.
- Requisito Não-Funcional.

[RNF 17] - A Api do Maiself deve se apoiar nos fundamentos SOLID para sua construção.

Identificador	[RNF 17] - A Api do Maiself deve se apoiar nos fundamentos SOLID para sua construção.
Descrição	A API seguirá o padrão SOLID para desenvolvimento visando melhor organização de componentes e estruturação do código.
Prioridade	Essencial

Tabela 43 – Tabela do Requisito não Funcional 17.

Requisitos de Interoperabilidade

- Esse requisito é referente a necessidade da aplicação se conectar com outra aplicação, aqui deve ser especificado necessidades do sistema de implementação de webservices ou outros tipos de interfaces com sistemas externos.
- Requisitos Não-Funcionais.

[RNF 18] - O Maiself deve ser modelado para consumir outras APIs caso necessário

Identificador	[RNF 18] - O Maiself deve ser modelado para consumir outras APIs caso necessário.
Descrição	O front-end que fará a comunicação com o back-end em formato API deve ser construído de forma que possa facilitar a integração e a realização de requisições com outras APIs não se prendendo apenas a nossa.
Prioridade	Essencial

Tabela 44 – Tabela do Requisito não Funcional 18.

[RNF 19] - O Maiself terá uma API única tanto para versão web para versão mobile

Identificador	[RNF 19] - O Maiself terá uma API única tanto para versão Web quanto para Versão mobile.
Descrição	O Maiself terá uma API única tanto para versão Web quanto para versão Mobile, utilizando um conceito bastante aplicado no mercado atualmente.
Prioridade	Essencial

Tabela 45 – Tabela do Requisito não Funcional 19.

Requisitos de Ética

- **Adicionamos esses requisitos extras com o intuito somente de documentar decisões que estão relacionadas com a visão e com os aspectos burocráticos do sistema.**
- **Requisitos Não-Funcionais Derivados:**

[RNF 20] - O Maiself não divulgará nem negociará quaisquer dados ou estatísticas relacionadas ao usuário.

Identificador	[RNF 20] - O Maiself não divulgará nem negociará quaisquer dados ou estatísticas relacionadas ao usuário.
Descrição	Visando a integridade ética e a segurança dos nossos usuários é essencial que não iremos divulgar/vender, nenhum dado do usuário ou relacionado a ele.
Prioridade	Essencial

Tabela 46 – Tabela do Requisito não Funcional 20.

6 PROTOTIPAÇÃO DO PROJETO

Esta seção tem como objetivo apresentar as telas produzidas do protótipo da nossa solução com base nos conceitos vistos na disciplina Interação Humano-Computador(IHC) e boas práticas de UI.

6.1 Fluxo de Navegação

Abaixo, está representada a modelagem de tarefas de algumas das ações - as tarefas - e o fluxo que é desejado que o usuário siga em cada uma delas. Este é um passo bastante importante para que junto com as fases de IHC suprimidas nesse documento como a criação de personas, e cenários, sirvam de base para uma das últimas fases da fase de prototipação, junto com os wireframes.

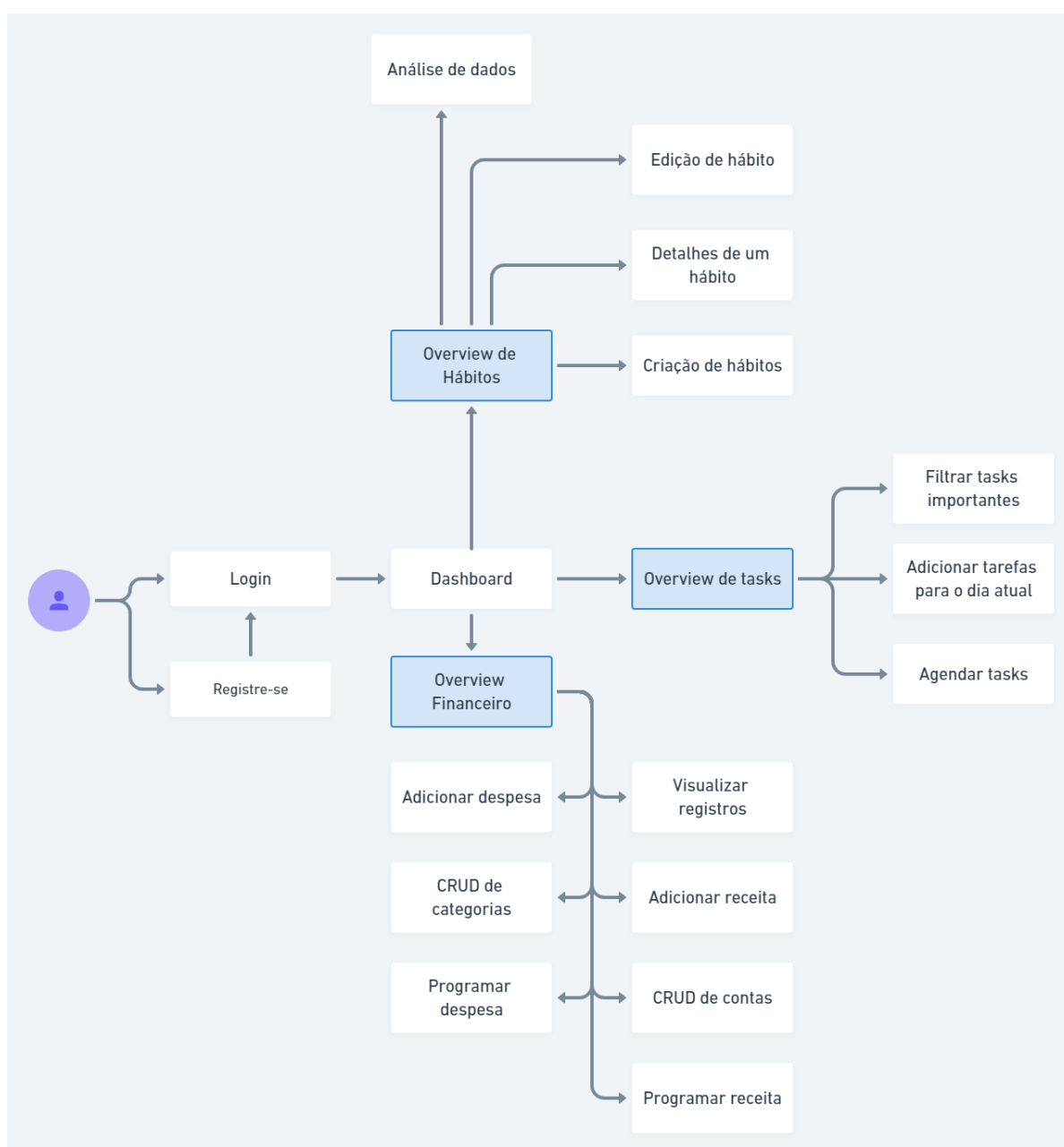


Figura 5 – Fluxo de possíveis navegações do usuário.

7 FRAMEWORKS, LINGUAGENS DE PROGRAMAÇÃO E ARQUITETURA DE SOFTWARE UTILIZADOS NO PROJETO

A linguagem de programação principal utilizada no nosso projeto será JavaScript, devido a maior familiarização dos participantes do grupo com ela, a facilidade de entendimento e o tempo de uso da mesma. Já no tocante aos frameworks que iremos utilizar, tanto no front-end quanto no back-end optamos por frameworks muito utilizados no mercado de trabalho atualmente, todos os participantes já utilizaram ou criaram projetos com estes frameworks que são ReactJS + NextJS para o desenvolvimento do front-end de SPA com SEO e para o back-end, NestJS + NodeJS

No tocante às Arquiteturas seguidas no nosso projeto a explicação será dividida em duas partes, primeiro a do front-end e depois a do back-end.

Arquitetura do Front-end: A arquitetura do projeto front-end que será construído em ReactJs + NextJs será baseado em componentes, ou seja, todo trecho de código que possa possivelmente ser utilizado em outros trechos de código, isolamos e transformamos em componentes.

Arquitetura do Back-end: A arquitetura do projeto back-end será construído com NestJS e NodeJS, seguindo a orientação a módulos. Partes importantes para a aplicação viram módulos, assim como pode existir submódulos destes módulos maiores e assim por diante. A estrutura de pasta foi pensada para que de acordo com a escalabilidade do projeto seja incrementada, não seja preciso grandes mudanças estruturais. Camadas de infra e http são criadas para distribuir corretamente as responsabilidades, assim como possíveis providers, DTO (data transfers objects) possuem pastas reservadas.

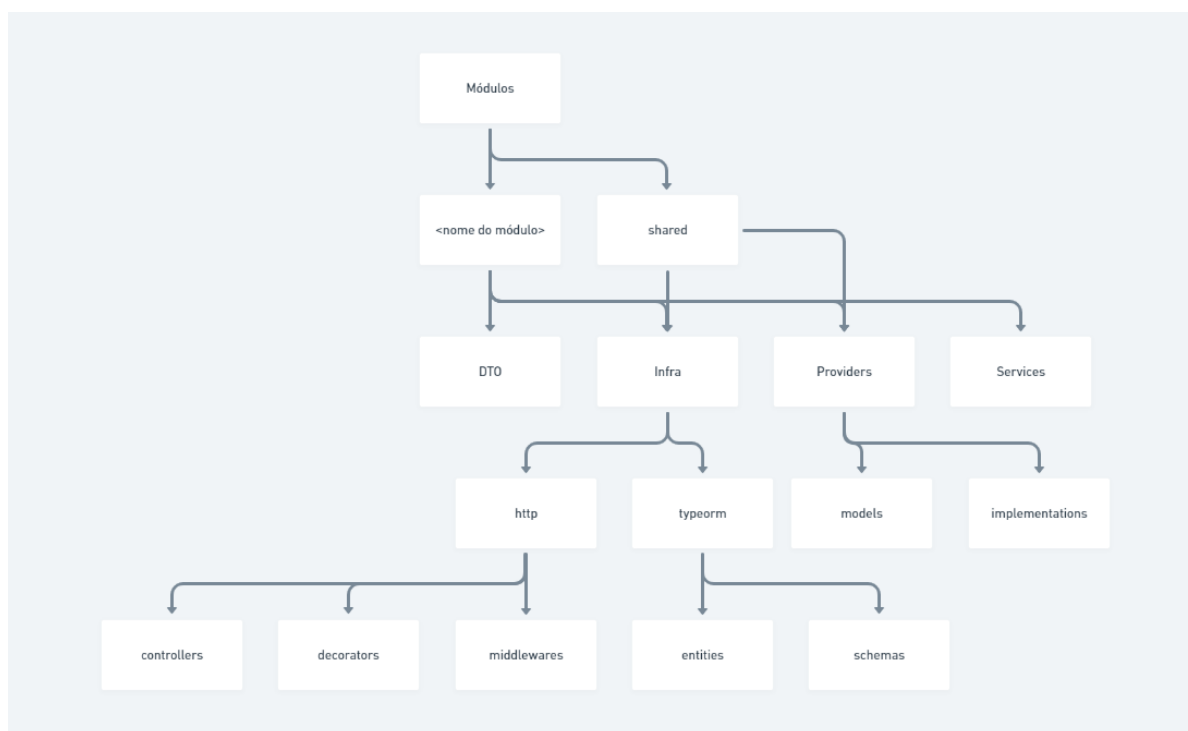


Figura 6 – Arquitetura do Back-end e suas separações.

8 BANCO DE DADOS

Toda aplicação tem a necessidade de armazenar dados, sejam estes dos mais diversos tipos, independente do âmbito em que ela está inserida, na nossa plataforma, existem dados de usuário, dados relacionados aos hábitos desses usuários, dados produzidos pelo relacionamento entre diferentes usuários, etc... Para persistir esses dados, na era caracterizada pela imersão de dados, devemos tomar decisões baseado nos critérios que consideramos prioritários, e isso pode significar a co-utilização de diferentes tipos de bancos de dados, neste caso, pensamos na elevada quantidade de dados que deverá ser armazenada, diariamente, devido à utilização e os logs resultantes das atividades desempenhadas pelos hábitos usuários, dentro da nossa plataforma, pensando neste armazenamento e na integridade desses dados, somente para o módulo de hábitos, optamos por futuramente escolher o MongoDB, um SGBD para bancos não relacionais, baseado na noção de documentos, onde os atributos são armazenados como uma coleção, sem esquemas. O MongoDB proporciona uma maior leveza no armazenamento de dados, em detrimento com a nossa outra opção utilizada para o restante da aplicação, o Postgresql, além disso, há uma maior facilidade para lidar com os relacionamentos entre o módulo e outras entidades, visto que esses acontecem de maneira implícita. No entanto, pensando na aplicação neste momento, utilizamos o modelo relacional em sua totalidade desenvolvida, onde os dados são modelados através de tabelas, isso garante uma maior consistência para os mesmos, o SGBD definido foi o Postgresql.

Esquema dos Dados

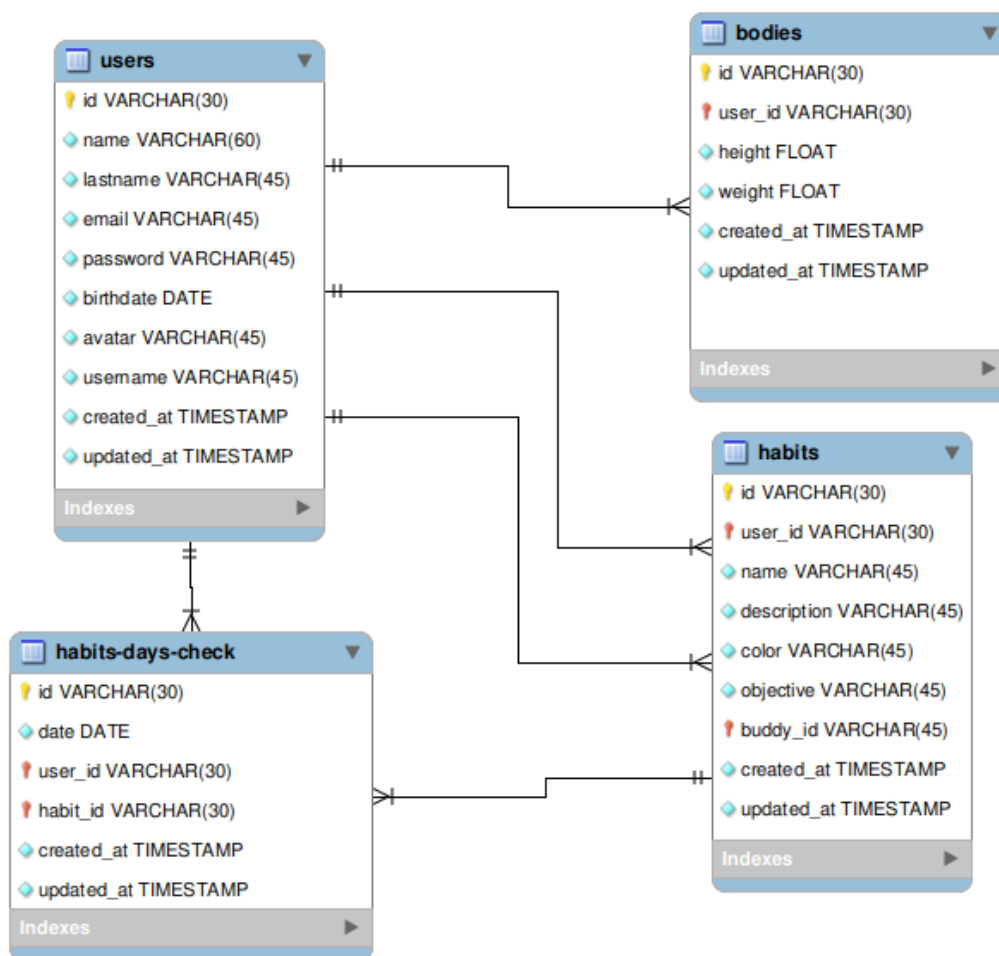


Figura 7 – Modelo Entidade Relacionamento da aplicação.

Padrão de Acesso aos Dados

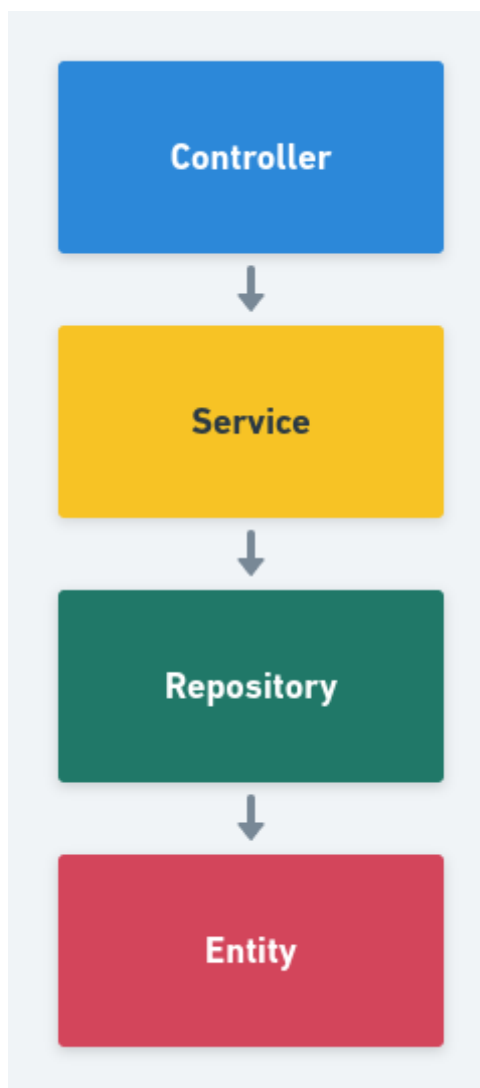


Figura 8 – Padrão de acesso a dados dentro do Back-end.

9 TESTES

O que será testado de forma automatizada, na nossa aplicação ?

Os módulos de usuário, hábitos e financeiro serão o que serão testados de forma automatizada, faremos testes unitários para as entidades e para os services de cada módulo (cada service representa uma unidade a ser testada), isso pode ser melhor visualizado a partir da imagem abaixo:

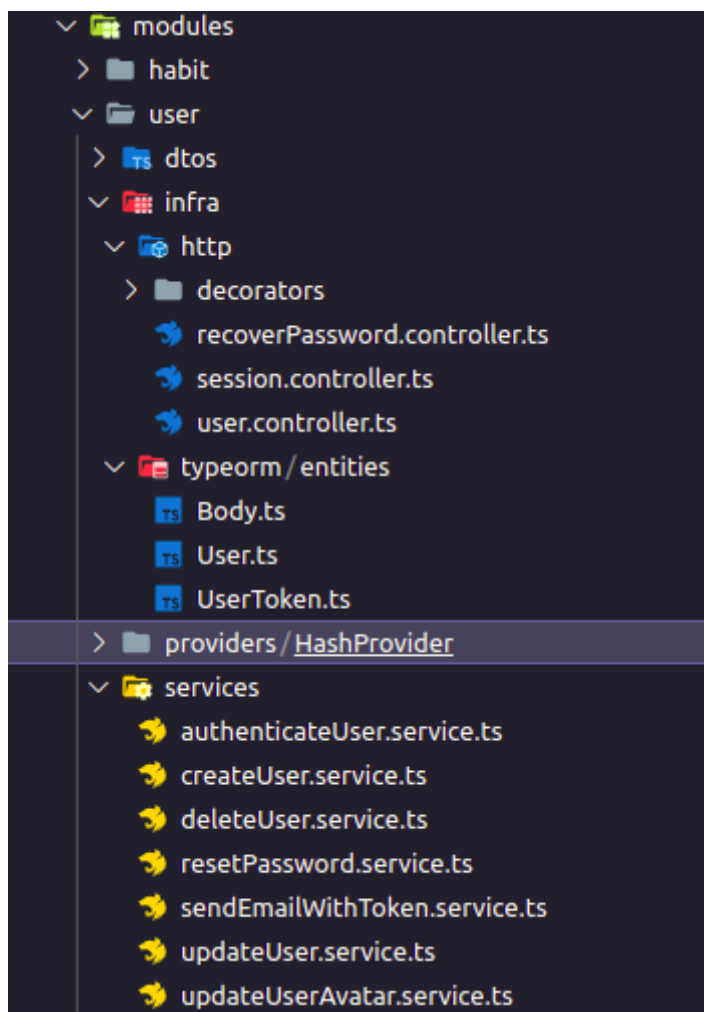


Figura 9 - Arquitetura de pastas que exhibe os services.

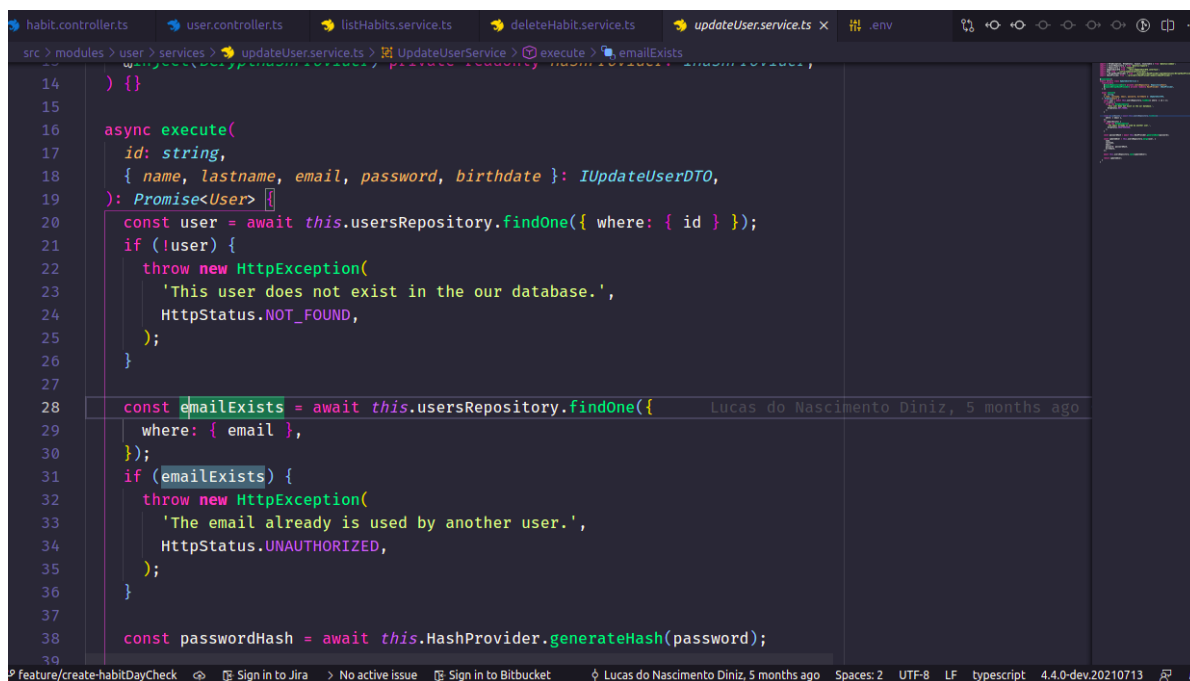
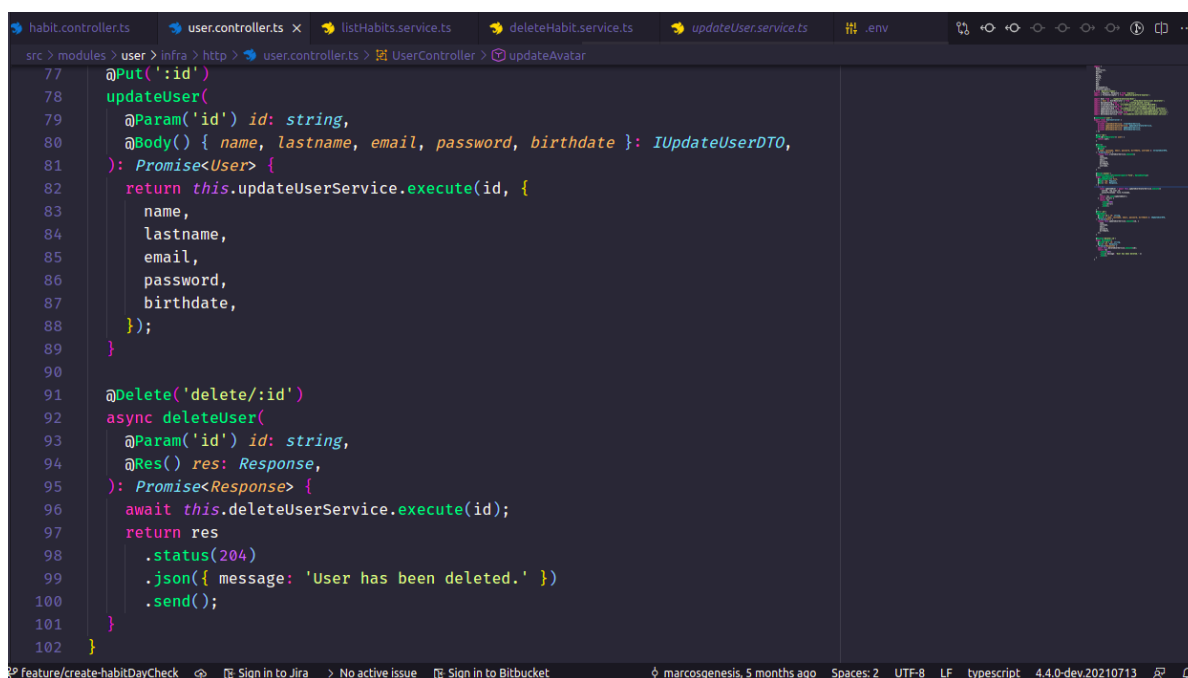


Figura 10 - Representação de um service.

Para realizar os testes de integração, dividimos estes em duas partes, primeiramente, os testes dentro de cada módulo, estes serão feitos nos controllers deste módulo, arquivos que contêm as rotas, que “chamam” cada service, estes testes validarão o funcionamento das unidades testadas, quando funcionam em conjunto com às outras. Por exemplo: individualmente, dos testes unitários, iremos testar se é possível editar um usuário, no teste de integração, um dos comportamentos poderá ser o de tentar editar um usuário que foi deletado do banco, isso não deve ser possível. Um exemplo de controller da nossa aplicação está disposto abaixo.



```

77  @Put('/:id')
78  updateUser(
79    @Param('id') id: string,
80    @Body() { name, lastname, email, password, birthdate }: IUpdateUserDTO,
81  ): Promise<User> {
82    return this.updateUserService.execute(id, {
83      name,
84      lastname,
85      email,
86      password,
87      birthdate,
88    });
89  }
90
91  @Delete('delete/:id')
92  async deleteUser(
93    @Param('id') id: string,
94    @Res() res: Response,
95  ): Promise<Response> {
96    await this.deleteUserService.execute(id);
97    return res
98      .status(204)
99      .json({ message: 'User has been deleted.' })
100      .send();
101  }
102  }

```

Figura 11 - Representação de um controller.

A segunda parte dos testes de integração servirá para testar as interações entre os módulos supracitados, esses testes têm como intuito validar se a aplicação não irá “quebrar” quando os módulos estiverem “conectados” entre si, com o acesso à funcionalidades que dependem dessa interação. Os módulos podem ser vistos abaixo.

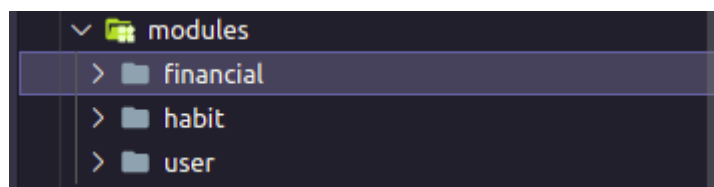


Figura 12 - Arquitetura de pastas que exhibe os módulos da aplicação.