Desenvolvimento de uma extensão para navegador que servirá de apoio para uma rotina saudável, utilizando o Framework Angular

Jéssica Fernanda de Carvalho[1], Maikon Delduca Batista[2], Fabrício Gustavo Henrique[3]

[1,2,3]Faculdade de Tecnologia de FATEC Ribeirão Preto (FATEC)
Ribeirão Preto, SP – Brasil

jessicafer.carvalho@gmail.com, maikondbatista@hotmail.com,
fabricio.henrique@fatec.sp.gov.br

Resumo. *Este artigo insere-se na área de Análise e Desenvolvimento de Sistemas e apresenta como eixo temático a construção de um software web utilizando o Framework Angular, com o intuito de ser usado como uma extensão de navegador. O objetivo principal desta extensão é apoiar as pessoas que ficam muito tempo em frente a tela do computador a se lembrarem de hábitos simples, mas que fazem toda diferença para saúde, como beber água, acertar a postura, piscar os olhos, entre outros. Será abordado as principais etapas do desenvolvimento da extensão e um breve estudo sobre o Framework Angular mostrando como o seu conjunto de elementos, ajudam a criar um sistema web de qualidade e com uma produtividade incrível.*

Abstract. *This article is part of the System Analysis and Development area and presents as thematic axis the construction of a web software using the Angular Framework, in order to be used as a browser extension. The main objective of this extension is to support people who spend a long time in front of the computer screen to remember simple habits, but that make all the difference for health, such as drinking water, correct posture, blinking, among others. The main stages of the extension development will be addressed and a brief study on the Angular Framework showing how its set of elements, help to create a web system of quality and with incredible productivity.*

1. Introdução

Atualmente, com avanço da pandemia, escolas tiveram que promover o ensino remoto e algumas empresas tiveram que adotar o *Home Office*. Se já era comum as pessoas ficarem horas em frente ao computador, isso só aumentou com o atual cenário.

Para aqueles que agora trabalham e estudam em casa, é importante se atentarem a alguns hábitos para não terem problemas de saúde. Ficam tão focados em frente à tela do computador que esquecem de beber água, piscar os olhos, se endireitar na cadeira ou se movimentarem de vez em quando.

Nesse sentido, este trabalho tem como objetivo desenvolver uma extensão para navegador que além de ativar os lembretes padrões citados acima, também dará ao usuário a liberdade de criar novos para não ficar preso apenas a estas opções pré-configuradas.

Para o desenvolvimento da extensão foi utilizado o Framework Angular, que foi

lançado em 2016 pelo Google e vem ganhando espaço no mercado desde então. Uma de suas vantagens é a tecnologia *Single-Page Applications* (SPA), que é a construção de aplicações em página única, não sendo necessário o seu recarregamento e assim fornecendo uma melhor experiência ao usuário. Além de ser open *source* e possuir uma grande gama de materiais de estudo para quem pretende aprimorar-se.

2. Métodos e ferramentas

As aplicações construídas em Angular tem uma principal característica que as difere de aplicações web comum: Os recursos usados para o funcionamento da aplicação (HTML, CSS, TypeScript e JavaScript) são todos carregados de uma vez ou dinamicamente conforme necessidade, sendo assim, não há transferência de controle entre páginas. E ao contrário de outros frameworks JavaScript, o Angular é completo, pois fornece tudo que é necessário para o desenvolvimento, desde ferramentas para testes até roteamentos.

2.1. JavaScript

O JavaScript é uma linguagem de programação que permite a manipulação de páginas web, sempre que uma página web faz algo a mais do que apenas mostrar seu conteúdo de forma estática, provavelmente está rodando JavaScript.

Essa linguagem permite criação e manipulação de variáveis, como por exemplo, o armazenamento dos dados de um formulário, também permite que o programador adicione *Event Listeners* por eventos, tais como clique, duplo clique, pressionar e soltar de teclas, entre outros inúmeros eventos. Além disso, ele também possui integrado ao núcleo da linguagem as APIs (*Application Programming Interfaces* – Interface de Programação de Aplicativos) que são uma espécie de conjuntos prontos de código, os quais permitem implementações complexas com pouquíssimas linhas de código. Essas APIs dividem-se em dois grupos, as APIs de navegador e APIs de terceiros, sendo:

a) APIs de Navegador: DOM (Document Object Model), Geolocalização, Serviço de Notificações, Alarmes, entre outras.

b) APIs de Terceiros: API de Login do Google, API de login do Facebook, entre outras.

2.2. TypeScript

O TypeScript é como um *superst* (super conjunto) da linguagem JavaScript, ele oferece todas as funcionalidades do JavaScript adicionado, principalmente os tipos, que dão nome a ele. E suas grandes vantagens frente ao JavaScript são:

a) Criação de tipos, facilitando o uso da POO (Programação Orientada a Objetos);

b) Ambiente rico em detecção de erros durante a digitação;

c) IntelliSense (Inteligência), ou seja, a capacidade do código de se autocompletar;

d) Métodos, funções entre outras implementações simplificadas;

2.3.1 Html

O HTML(*HyperText Markup Language* ou Linguagem de Marcação de Hipertexto) é a

base da programação WEB, ela define a estrutura da página, como o próprio nome diz, o HTML é uma linguagem de marcação, a qual é constituída de elementos, sendo estes elementos todas as marcações delimitadas pelas *tags*, que são os caracteres "<"(Abertura da Marcação) e ">" (Fechamento da Marcação), um documento HTML, possui a seguinte estrutura básica:

```
1  <!DOCTYPE html>
2  <html lang="pt-br">
3    <head>
4      <title>Título da página</title>
5      <meta charset="utf-8">
6    </head>
7    <body>
8    </body>
9  </html>
```

Figura 1 - Exemplo de uma estrutura HTML

A Linha 1 define o tipo do documento;

As linhas 2 e 10 definem a estrutura como HTML;

As Linhas 3 e 6 definem o cabeçalho, a primeira parte do HTML que será interpretada pelo browser, aqui deve ser colocado todo o código que deve ser lido primeiro pelo navegador, como blocos de código JavaScript, bibliotecas entre outros dados que devem ser tratados com prioridade;

A linha 4 define o Título da página (que aparecerá na aba do navegador);

A linha 5 específica o tipo de encoding, que é a codificação dos caracteres;

As linhas 7 e 8 definem o corpo, ou seja, onde será efetivamente construída a página WEB.

2.3.2 CSS

CSS (Cascading Style Sheets ou Folhas de Estilo em Cascata) assim como o HTML, o CSS não é considerado uma linguagem de programação, e sim uma linguagem de folha de estilos.

Com o CSS é possível aplicar estilos apenas aos elementos HTML que se deseja. Para aplicar estilo a um elemento, é necessário seguir um conjunto composto de quatro regras, conforme destacado na figura 2.

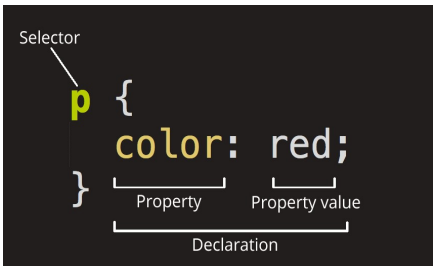

Figura 2 - Exemplo de uma estrutura CSS

a) Selector: Ele define há quem será aplicado o estilo, o seletor pode se aplicar à classe, elemento (p, div, fieldset, etc.), id entre outras combinações.

b) Declaration: Especifica quais propriedades do elemento deve ser estilizada.

c) Property: Especifica qual propriedade do elemento deve ser estilizada.

d) Property Value: Especifica qual valor será dado à propriedade.

3. Desenvolvimento

Neste momento, será mostrado as principais etapas do desenvolvimento da extensão.

3.1. Criando o projeto

Para criar a aplicação "Be Healthier" (nome dado a extensão), usa-se o comando 'ng new BeHealthier' e em seguida temos esta estrutura de pastas e arquivos:

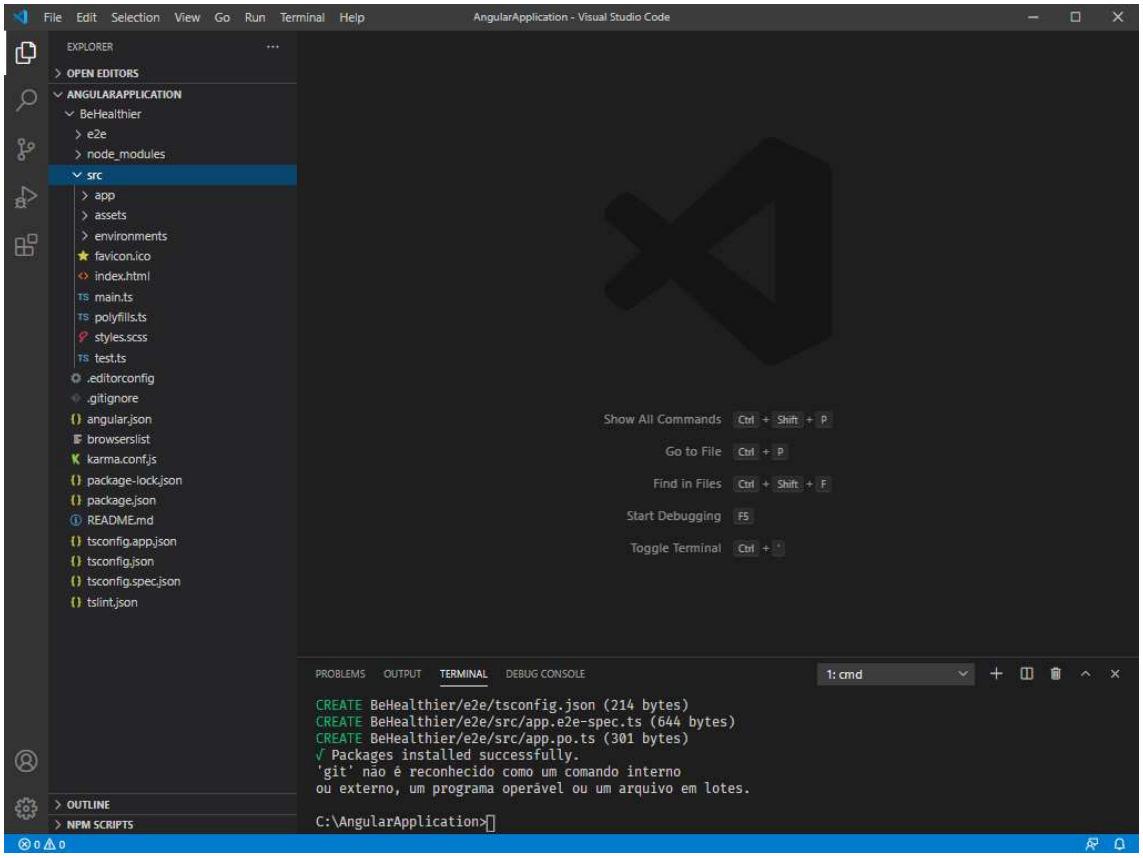

Figura 3 - Estrutura de um projeto Angular

As pastas têm as seguintes finalidades:

a) E2e – Testes End-To-End usando *proctractor*.

b) Node_modules – Dependências do Node.JS para o projeto.

c) Src/assets – Recursos externos como arquivos diversos, imagens, etc.

d) Src/environments – Configuração de ambientes (desenvolvimento/produção).

e) Src/app – Pasta criada junto do aplicativo, que contém o código padrão da aplicação.

Para criar um novo componente, usa-se o comando "ng generate component [nomecomponente]". Abaixo tem um exemplo do componente *settings* criado para montar a tela de configuração da extensão:

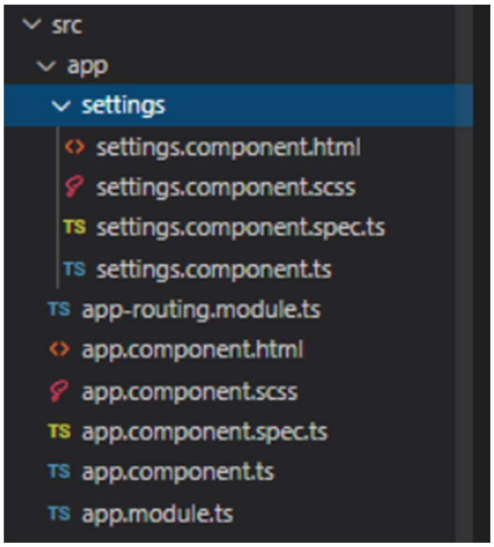

Figura 5 – Estrutura de um componente no Angular

O Componente se subdivide em 4 arquivos:

a) [NomeComponente].html – Código HTML do componente.

b) [NomeComponente].SCSS – Código SCSS (Estilização) do componente.

c) [NomeComponente].spec.ts – Arquivo para teste unitário usando Jasmine.

d) [NomeComponente].ts – Arquivo para o código Typescript.

Para extensões no navegador Google Chrome, é necessário configurar um arquivo do tipo 'JSON' chamado 'manifest.json', este arquivo provê importantes dados sobre a extensão, como qual página abrir quando o usuário clicar em seu link, permissões, configuração de tarefas *background* e entre outras configurações importantes.

O arquivo de manifesto pode ser visualizado na figura 6.

3.2. JSON

JSON (JavaScript Object Notation) é um formato de arquivo muito usado para interações entre sistemas, ele serve para armazenar e transmitir informações, além disso, ele pode ser facilmente lido por humanos. Este formato permite a transmissão de dados de maneira eficiente, pois permite transmitir apenas o necessário, tornando-o uma alternativa mais simples e com menor tamanho comparado ao seu principal "concorrente", o XML.

src > {} manifest.json > ...

```
{
  "name": "Be Healthier",
  "version": "1.0",
  "description": "Build an Extension with Angular",
  "manifest_version": 2,
  "browser_action": {
    "browser_style": true,
    "default_title": "Be Healthier",
    "default_popup": "index.html?#/settings"
  },
  "background": {
    "scripts": ["background.js"],
    "persistent": false
  },
  "permissions": ["notifications", "alarms", "background"]
}
```

Figura 6 - Estrutura de um arquivo JSON

Analisando a estrutura de um JSON, entende-se que da linha 1 a 5 temos "Propriedade": "valor".

Da linha 6 até a 10, define-se que aquela key (chave) é um object (objeto), ou seja, o 'browser_action' não possui um valor diretamente, e sim outros três pares de key e value, que estão definidas entre a linha 7 e 9.

Na linha 11 o caso da linha 6 se repete.

Da linha 12 e na 15, temos um exemplo de key do tipo Array (Arranjo ou Lista), ou seja, pode-se ter um ou mais valores para estas propriedades.

Portanto, conclui-se que os caracteres '[' e ']' definem o início e fim de um array e os caracteres '{' e '}' definem um objeto.

3.3. Permissões

Para o correto funcionamento da extensão, é necessário declarar corretamente no arquivo de manifesto as permissões que ela precisa.

Notifications – Essa permissão faz com que a extensão possa enviar notificações para o Sistema Operacional, onde são enviadas pela extensão através do Google Chrome. As notificações são exibidas na bandeja do sistema.

Alarms – Essa permissão permite que a extensão crie agendamentos, estes agendamentos são feitos com base no tempo de repetição definido nos alertas, quando o agendamento atinge o tempo especificado, o Chrome dispara um evento e a extensão "ouve" este evento e aplica a notificação necessária.

Background – Essa permissão permite que seja definido um script (background.js neste caso) que rodará mesmo quando a extensão não estiver aberta, tal permissão se faz necessária para que a extensão possa "ouvir" e lidar com os eventos enviados pelo Alarm do Google Chrome, sem essa permissão, a extensão só conseguiria enviar notificações caso ficasse aberta.

3.4. Módulo

A base de um projeto Angular são os módulos, pois eles são responsáveis por carregar as dependências necessárias para o funcionamento da aplicação, cada módulo atribui uma configuração de injeção de dependência que o Angular irá utilizar, além de gerenciar os pacotes baixados pelo cliente (browser).

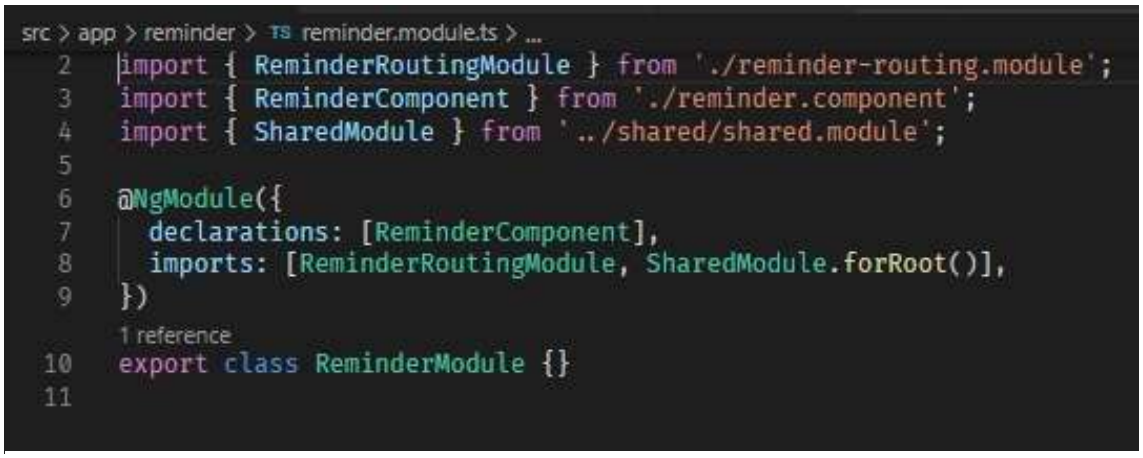

Figura 7 - Módulo criado na aplicação

O "ReminderModule" é o módulo responsável pela tela de criação/alteração de alertas, ele carrega o "SharedModule", que é responsável por carregar dependências comuns entre as demais aplicações, como "CommonModule", "NgxBootstrapSliderModule", entre outros módulos, componentes e serviços.

3.5. Roteamento

Por padrão no Angular, os módulos são carregados todos de uma vez, ou seja, um pacote com todas as páginas da aplicação é enviado para o cliente em apenas uma requisição, para casos onde é previsto que o usuário não vai usar toda aplicação, sempre que acessá-la, é possível utilizar sua funcionalidade de *Lazy-loading*, que permite enviar ao *client* (navegador) apenas os dados necessários para exibição de determinada página, por exemplo, em nossa aplicação o usuário nem sempre vai acessar o gerenciamento de alertas, portanto foi implementado o *Lazy-loading* para que o *client* apenas receba esse pacote caso necessário.

src > app > TS app-routing.module.ts > ...

```
1   import { NgModule } from '@angular/core';
2   import { Routes, RouterModule } from '@angular/router';
3
4   const routes: Routes = [
5     {
        6 references
6       path: 'settings',
        3 references
7       loadChildren: () =>
8         import('./settings/settings.module').then(
9           (m) => m.SettingsModule
10        ),
11    },
12    {
        6 references
13      path: 'manage',
        3 references
14      loadChildren: () =>
15        import('./manage/manage.module').then((m) => m.ManageModule),
16    },
17    {
        6 references
18      path: 'reminder',
        3 references
19      loadChildren: () =>
20        import('./reminder/reminder.module').then((m) => m.ReminderModule),
21    },
22  ];
23
24  @NgModule({
25    imports: [RouterModule.forRoot(routes, { useHash: true })],
26    exports: [RouterModule],
27  })
      2 references
28  export class AppRoutingModule {}
29
```

Figura 8 - Rotas da aplicação

Na linha 18 é definida a URL que deve ser acessada, deste modo ao acessar: “localhost:4200/reminder” (que no caso da extensão, quando o usuário acessar a criação ou alteração de alertas) é carregado o módulo “ReminderModule" que contém todos os dados relativos a esta aplicação.

4. Funcionamento da extensão

A extensão, quando instalada possui quatro alertas cadastrados por padrão:

a) Piscar, com intervalo de 7 minutos;

b) Beber água, com intervalo de 120 minutos;

c) Corrigir a postura, com intervalo de 30 minutos;

d) Movimentar-se, com intervalo de 120 minutos;

Ao abrir a extensão, o usuário vai visualizar a seguinte tela:

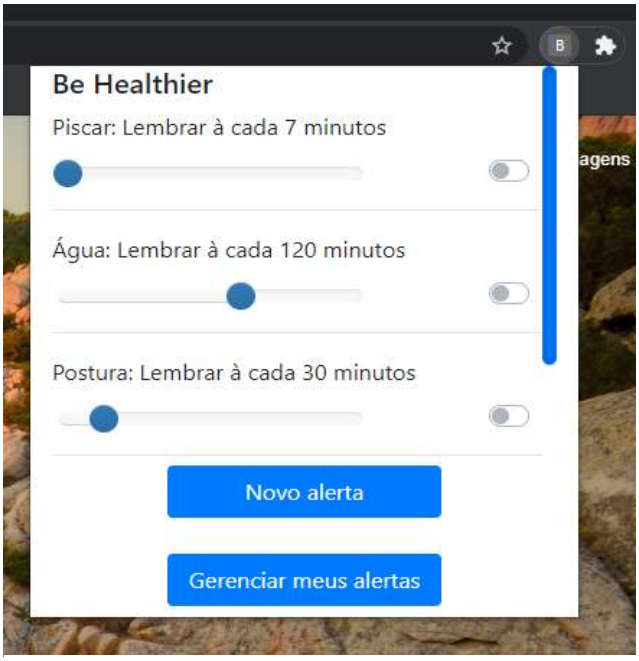

Figura 9 - Tela principal da extensão

Para que a extensão não tome muito espaço da tela, é exibida uma barra de rolagem que permite a visualização rápida de até 3 alertas por vez.

Na tela inicial, permite-se que o usuário faça configurações rápidas, como ativar/desativar alarmes e alterar o tempo entre as repetições dos mesmos. Para alterar outras configurações do alerta, como ícone, descrição e título, deve-se ser acessar a página 'Gerenciar meus alertas', que será da seguinte maneira:

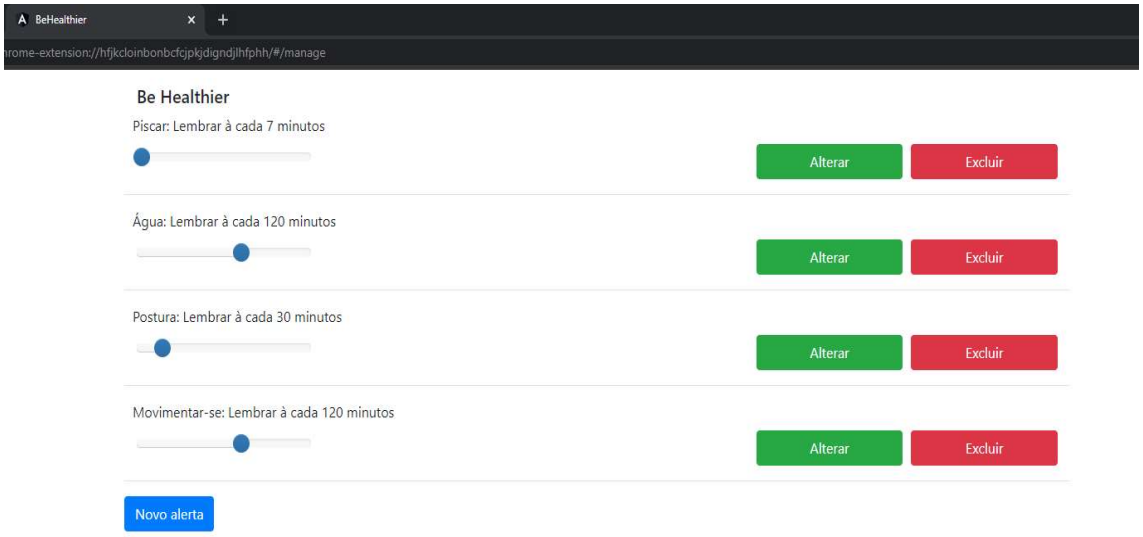

Figura 10 - Tela para gerenciar os alertas

Nesta tela o usuário terá permissão de alterar e excluir os alertas, tanto os

padrões como os criados por ele.

Além dos alertas já cadastrados ao instalar a extensão, o usuário tem total liberdade para criação de novos e alteração dos existentes, eles podem ser facilmente criados ao navegar para a tela de 'Novo alerta' ou 'Alterar alerta' e preencher o formulário da imagem abaixo:

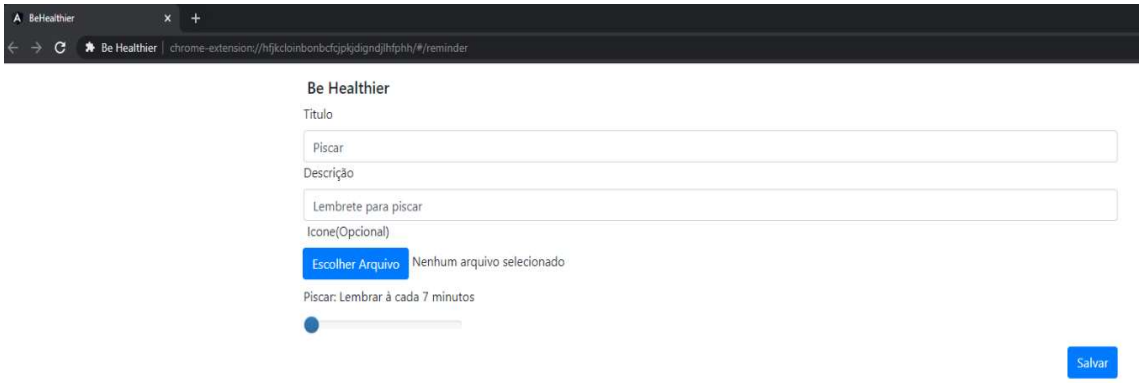

Figura 11 - Tela para cadastrar ou alterar alertas

Ao clicar em salvar, o *timer* do alerta será reiniciado e o ele passará a contar com os novos valores associados a ele.

Quando o alerta estiver ativo e a extensão ativada, de acordo com o tempo definido para as suas repetições, o usuário receberá notificações através do navegador Google Chrome, exibindo o ícone, título e descrição definidos no momento do cadastro do alerta, como mostra na figura 12.

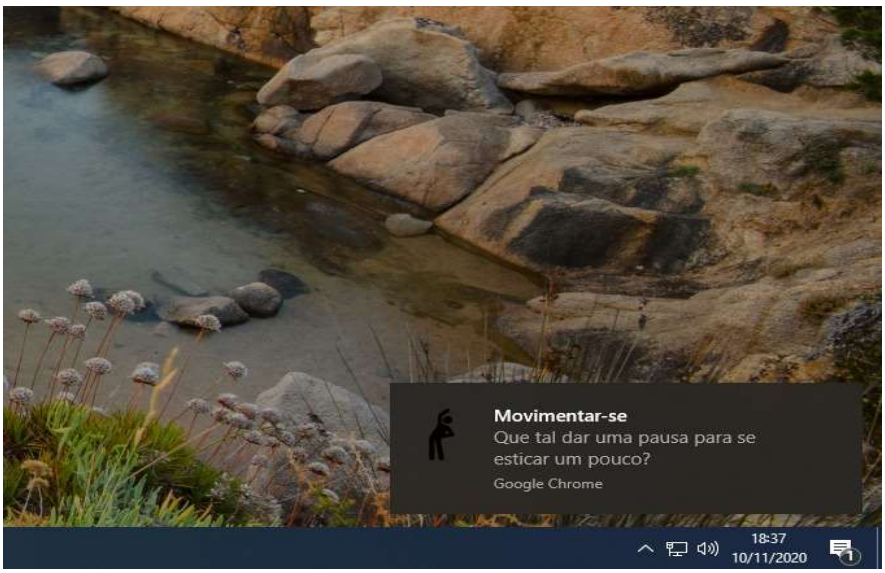

Figura 12 - Notificação da extensão

5. Conclusão

Durante o desenvolvimento da extensão foi apresentado o poderoso Framework Angular que se mostrou ser muito eficaz. E para quem nunca teve contato com a ferramenta, foi mostrado alguns de seus recursos e especificações.

Pode-se concluir que o objetivo proposto inicialmente foi alcançado. E os conceitos apresentados neste projeto podem ser aplicados para desenvolver diversos outros tipos de aplicações em Angular.

Embora a extensão atenda ao que foi proposto no projeto, algumas melhorias podem ser exploradas no futuro, como:

a) Implementar a funcionalidade de lembretes diários, semanais, mensais e anuais.

b) Publicar a extensão na Chrome Web Store para que todos tenham acesso.

c) Colocar a extensão para rodar em outros navegadores, além do Chrome.

Referências

MOZILLA. “O que é JavaScript?” Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/First_steps/O_que_e_JavaScript>. Acesso em 26 de outubro 2020.

LUIZ, ANDREY. “JavaScript #1 - Uma breve história da linguagem.” Disponível em: <http://shipit.resultadosdigitais.com.br/blog/javascript-1-uma-breve-historia-da-linguagem/#:~:text=O%20JavaScript%20foi%20criado%20na,era%20o%20Mosaic%2C%20da%20NCSA.&text=Em%201995%2C%20a%20Netscape%20contratou,uma%20linguagem%20que%20proporcionasse%20isso>. Acesso em 26 de outubro 2020.

MACORATTI, JOSÉ CARLOS. “O que é TypeScript e quais os seus benefícios?” Disponível em: <http://www.macoratti.net/16/05/net_typscr1.htm. Acesso em 26/10/2020>. Acesso em 28 de outubro 2020.

DEVMEDIA. “Introdução ao TypeScript.” Disponível em: <https://www.devmedia.com.br/introducao-ao-typescript/36729 Acesso em 26/10/2020>. Acesso em 28 de outubro 2020.

W3SCHOOLS. “What is HTML?” Disponível em: <https://www.w3schools.com/whatis/whatis_html.asp>. Acesso em 26/10/2020. Acesso em 28 de outubro 2020.

MOZILLA. HTML: “Linguagem de Marcação de Hipertexto.” Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/HTML Acesso em 26/10/2020>. Acesso em 29 de outubro 2020.

TABLELESS. “Estrutura Basica – Iniciando o código básico de HTML.” Disponível em: <https://tableless.github.io/iniciantes/manual/html/estruturabasica.html>. Acesso em 29 de outubro 2020.

MOZILLA. “CSS.” Disponível em: < https://developer.mozilla.org/pt-BR/docs/Web/CSS Acesso em 26/10/2020. >. Acesso em 30 de outubro 2020.

ANGULAR. “Setting up the local environment and workspace.” Disponível em: <https://angular.io/guide/setup-local#setting-up-the-local-environment-and-workspace>. Acesso em 30 de outubro 2020.

ANGULAR. “Lazy-loading feature modules.” Disponível em: <https://angular.io/guide/lazy-loading-ngmodules>. Acesso em 01/11/2020. Acesso em 30 de outubro 2020.

ANGULAR. “Introduction to modules.” Disponível em: <https://angular.io/guide/architecture-modules>. Acesso em 01 de novembro 2020.

CHROME. “Manifest File Format.” Disponível em: <https://developers.chrome.com/extensions/manifest>. Acesso em 01 de novembro 2020.

GUBITOSO, MARCO DIMAS. “CSS e SCSS.” Disponível em: <https://www.ime.usp.br/~gubi/aulas/218/SASS.html>. Acesso em 01 de novembro 2020.

L. ANDREI. “O Que É JSON?” Disponível em https://www.hostinger.com.br/tutoriais/o-que-e-json/#:~:text=O%20JSON%20(JavaScript%20Object%20Notation,para%20que%20ele%20%C3%A9%20usado. Acesso em 07 de novembro 2020.