FACULDADE DE TECNOLOGIA DE SOROCABA
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

ROSANE RAMOS DE ALMEIDA

SOROACQUA: PORTAL PARA DIVULGAÇÃO DE INFORMAÇÕES SOBRE A ÁGUA DE SOROCABA

Sorocaba, SP
Junho/2017

FACULDADE DE TECNOLOGIA DE SOROCABA

Relatório – Iniciação Científica

SOROACQUA: PORTAL PARA DIVULGAÇÃO DE INFORMAÇÕES SOBRE A ÁGUA DE SOROCABA

Aluna: Rosane Ramos de Almeida
6° Semestre - Tecnologia em Análise e Desenvolvimento de Sistemas
Orientador: Prof. Dr. Paulo Edson Alves Filho

Sorocaba, SP
Junho/2017

AGRADECIMENTOS

Agradeço ao meu orientador por todo apoio no desenvolvimento deste projeto e a todos os professores da Fatec Sorocaba pelo conhecimento que foi passado dentro e fora da sala de aula. Agradeço também a minha mãe por toda sua dedicação para que eu tivesse uma boa educação e por todo seu amor, e ao meu marido por estar ao meu lado me apoiando e incentivando.

RESUMO

A água é de fundamental importância para a vida de todos os seres vivos, sendo 80% do corpo humano é composto por ela. É considerada um solvente universal, auxilia na prevenção de doenças e proteção do organismo contra o envelhecimento. Sabendo que o organismo humano necessita de água para sobreviver, é indispensável que ela seja devidamente tratada, tornando-se ideal para o consumo. Sendo assim, as informações sobre sua qualidade devem ser disponibilizadas para a população para que ela possa cobrar dos responsáveis um tratamento mais adequado. Pensando nisso, este projeto teve o objetivo de criar um aplicativo móvel que permita a população consultar de forma clara e objetiva a qualidade da água que abastece seu bairro e a qualidade do rio Sorocaba, por meio de consultas personalizadas. O aplicativo contará também com a divulgação de notícias, eventos e projetos sobre o tratamento da água do rio Sorocaba e abrirá espaço para a divulgação de campanhas ecológicas visando a preservação da mata ciliar, fauna e flora. Este projeto também contempla a criação de um portal administrativo que servirá para alimentar o aplicativo com as informações que serão disponibilizadas pelo SAAE (Serviço Autônomo de Água e Esgoto) Sorocaba. O projeto completo está subdividido, e será realizado em equipe, sendo esta parte responsável pela análise de ferramentas, desenvolvimento do aplicativo e do portal e automação de testes para aplicativo híbrido.

Palavras-chave: Aplicativo. Qualidade. Água. Sorocaba. Híbrido. Desenvolvimento *Mobile*. Ionic 2. AngularJS. Testes Automatizados.

ABSTRACT

Water is of fundamental importance for the life of all living beings, and 80% of the human body is composed of it. It is considered a universal solvent, it aids in the prevention of diseases and protection of the body against aging. Knowing that the human body needs water to survive, it is indispensable that it be properly treated, making it ideal for consumption. Therefore, information about its quality should be made available to the population so that it can charge those responsible more appropriate treatment. With this in mind, this project aimed to create a mobile application that allows the population to consult clearly and objectively the quality of the water that supplies their neighborhood and the quality of the Sorocaba River, through personalized consultations. The application will also include the dissemination of news, events and projects on the water treatment of the Sorocaba River and will open space for the dissemination of ecological campaigns aimed at preserving the ciliary forest, fauna and flora. This project also includes the creation of an administrative portal that will serve to feed the application with the information that will be made available by the Sorocaba Autonomous Water and Sewage Service (SAAE). The complete project is subdivided and will be carried out as a team, and this part is responsible for tool analysis, application and portal development, and test automation for hybrid applications.

Keywords: Application. Quality. Water. Sorocaba. Hybrid. Mobile Development. Ionic 2. AngularJS. Automated Testing.

LISTA DE FIGURAS

Formatado ...

Formatado: Fonte parág. padrão

Formatado: Fonte parág. padrão

Formatado: Fonte parág. padrão

LISTA DE ABREVIATURAS E SIGLAS

SAAE - Serviço Autônomo de Água e Esgoto
DQO - Demanda Química de Oxigênio
DBO - Demanda Bioquímica de Oxigênio
OD - Oxigênio Dissolvido
NTK - Nitrogênio
PT - Fósforo
I/O - *Input* / *Output*
CVS - *Concurrent Version System*
HTML - *HyperText Markup Language*
CSS - *Cascading Style Sheets*
API - *Application Programming Interface*

SUMÁRIO

1 INTRODUÇÃO E JUSTIFICATIVA

O tratamento da água na cidade de Sorocaba utiliza um sistema moderno e de alta tecnologia, o que garante aos cidadãos a segurança do seu consumo.

O processo do tratamento da água começa com um pré-tratamento no qual, por meio de um gradeamento, retira os sólidos grosseiros do esgoto, depois é levado aos decantadores primários para a retirada dos sólidos sedimentares e então é enviado para os tanques de aeração onde o material é misturado com oxigênio para promover o crescimento da massa biológica antes de ser conduzido aos decantadores secundários, onde ocorre a separação da fase líquida da sólida. A fase líquida retorna ao rio Sorocaba, enquanto a fase sólida passará pelo processo de tratamento específico para sua disposição final.

Entretanto a população de Sorocaba não tem fácil acesso a informações sobre a qualidade e tratamento da água, o que resulta em uma falta de confiança sobre a água que está sendo consumida nas casas e também a qualidade da água do rio que contorna a cidade.

Hoje temos algumas informações no portal do SAAE, mas que não são de fácil acesso e entendimento para a população. Então, tivemos como objetivo fazer com que essa informação chegue às pessoas através de um aplicativo onde ela pode consultar a qualidade da água em qualquer região da cidade e também a qualidade da água do rio Sorocaba, levando em consideração alguns indicadores como DQO (Demanda Química de Oxigênio), DBO (Demanda Bioquímica de Oxigênio), pH, OD (Oxigênio Dissolvido), NTK (Nitrogênio) e PT (Fósforo), Resíduos, Cor e Turbidez, Condutividade e o Bacteriológico (Coliformes Termotolerantes).

A primeira etapa do projeto consistiu na análise e no estudo das tecnologias e ferramentas que foram utilizadas no projeto, suas boas práticas e padrões. Também foi desenvolvido um protótipo para o aplicativo e se deu início a implementação de suas principais funcionalidades.

Por se tratar de um projeto extenso e envolver desde questões de análise de ferramentas, novas tecnologias e desenvolvimento de software, ele foi assim subdivido:

Ana Paula Ribeiro: SoroAcqua: análise de interação humano e computador, análise e levantamento de requisitos, desenvolvimento da linguagem UML e desenvolvimento da aplicação móvel.

Rosane Ramos de Almeida: SoroAcqua: análise de ferramentas, modelagem de dados, desenvolvimento do aplicativo e do portal e automação de testes para aplicativo híbrido.

2 REVISÃO BIBLIOGRÁFICA

Para este projeto foi utilizado como base o Trabalho de Graduação realizado na Fatec Sorocaba chamado “Borgundfjorden.no: o portal para dados ambientais” (PESSUTO, 2016). Borgundfjorden.no é um portal que foi criado para hospedar informações ambientais sobre o fiorde Borgundfjorden na Noruega, bem como também reportar dados sobre a qualidade da água e flora do local.

Utilizou-se também como referência para os indicadores da qualidade da água o site do SAAE Sorocaba (SAAE SOROCABA, 2016) que contêm o resultado da qualidade da água que chega às residências da população de Sorocaba mês a mês e é divulgado através de um arquivo de texto.

Segundo Smith e Ribeiro (2016), o município de Sorocaba possui uma densa e perene malha hídrica composta com 2880 nascentes, além de córregos e rios, destacando-se o rio Sorocaba e o rio Pirajibu. Conforme os autores, é possível também verificar a diversidade da fauna e flora desta região, e a partir dele criamos no aplicativo uma área destinada para mostrar para a população informações como nome, descrição e ilustrações de alguns animais e da vegetação que contorna Sorocaba.

De acordo com o PORTAL DA QUALIDADE DAS ÁGUAS (2015), utilizam-se índices para medir a qualidade da água pela necessidade de sintetizar as informações a partir de vários parâmetros físico-químicos, que são utilizados para informar e orientar a população sobre a qualidade da água e sua gestão. Esses índices são simplificados para maior entendimento ao público leigo, mas para se obter uma análise mais detalhada deve-se analisar os parâmetros separadamente.

A partir do estudo sobre os índices que verificam a qualidade da água deu-se início ao desenvolvimento do aplicativo para dispositivos móveis.

Com o aumento das vendas de dispositivos móveis, a demanda por aplicações móveis também aumentou (CEVALLOS, 2014), e por isso optou-se por desenvolver um aplicativo para essa plataforma.

Para aplicativos móveis temos duas opções de desenvolvimento o nativo e o híbrido. Aplicações nativas são aplicações desenvolvidas para uma

plataforma específica e somente irá funcionar nela. Para esse desenvolvimento utiliza-se ferramentas como a linguagem de desenvolvimento e ambiente disponibilizados pela própria plataforma. Já aplicações híbridas funcionam em qualquer que seja o dispositivo utilizando apenas um código fonte. (PREZOTTO e BONIATI, 2014).

Conforme Bezerra (2016), no desenvolvimento híbrido o ambiente é diferente do disponibilizado pela plataforma, no caso do nativo, e tem como vantagem que desenvolvemos apena uma vez, em uma linguagem e é executado para várias plataformas, custando menos tempo.

Para o desenvolvimento do aplicativo foi utilizada a documentação do *framework* Ionic 2 (IONIC 2 *FRAMEWORK*, 2016). Essa documentação se encontra no site oficial do *framework* e oferece todo suporte necessário para se devenvolver um aplicativo híbrido.

Para a criação do Portal Administrativo foi utilizado o framework AngularJS, que é um framework muito poderoso mantido pelo Google. SCHMITZ e LIRA (2016) afirmam que o AngularJs fornece um conjunto de funcionalidades, tais como o DataBinding, templates, fácil uso do Ajax, controllers, que tornam o desenvolvimento web muito mais fácil e empolgante. Ele possui algumas particularidades interessantes, uma delas é funcionar como uma extensão ao documento HTML que permite adicionar novos parâmetros e interagir de forma dinâmica com vários elementos, sem necessitar programar em JavaScript, sendo quase uma linguagem declarativa. Esses parâmetros, que são chamados de diretivas, alteram o comportamento padrão do HTML.

Formatado: Cor da fonte: Preto

São chamado de aplicativos híbridos aqueles que são desenvolvidos utilizando linguagens *web* como: HTML, CSS e JavaScript. Esses aplicativos são comercializados e avaliados nas lojas virtuais da mesma forma que os aplicativos nativos, com a vantagem de serem disponibilizados para Android, Windows e IOS ao mesmo tempo, sem a necessidade de se criar três aplicativos diferentes cada um com a linguagem nativa da plataforma onde será disponibilizado. WEYL (2014) complementa afirmando que

> (...) as aplicações híbridas são HTML5, CSS e aplicativos baseados em JavaScript que são convertidos ou compilados em um aplicativo nativo, muitas vezes simples usando uma visualização de tela inteira. Como o recipiente de aplicação. Usando as tecnologias (...) HTML, CSS, JavaScript, podemos empacotar e compilar nosso código fonte em um aplicativo nativo para os vários sistemas operacionais de

> dispositivos. Uma vez que é um aplicativo nativo, podemos distribuí-lo nas várias lojas de aplicativos. (autor, ano, p.tradução nossa).

De acordo com o W3C (2016), o CSS é utilizado para formatar a informação que será entregue pelo HTML e essa informação pode ser qualquer elemento como: imagem, texto, vídeo, áudio ou qualquer elemento criado, sendo ele visual ou não. Ele permite formatar características como: cores, background, font, posição, etc. O CSS3 surgiu trazendo muito poder de formatação na criação de designer web.

JavaScript é a linguagem mais popular no desenvolvimento *web,* sendo suportada por todos os navegadores. É uma linguagem de programação interpretada, baseada em objetos e foi escolhida para o desenvolvimento deste projeto por ser leve e muito poderosa.

> JavaScript é um recurso criado pela equipe da Code School para a comunidade JavaScript. Como o JavaScript é uma ótima linguagem para iniciantes de codificação, reunimos alguns dos melhores recursos de aprendizagem e construímos um curso de JavaScript para ajudar os novos desenvolvedores a se instalarem.
> Com a ajuda de membros da comunidade que contribuem com conteúdo para o site, JavaScript.com pretende também manter mais avançados desenvolvedores atualizados sobre notícias, quadros e bibliotecas. (JAVASCRIPT, 2017, p.Tradução nossa).

MongoDB é um banco de dados NoSQL e orientado a documentos, possuindo uma estrutura flexível em relação a inserção e consulta dos dados. Antes de escolher o banco de dados a ser utilizado, foi realizada uma análise para escolher entre usar um banco SQL ou NoSQL. Ao final da análise, percebeu-se que seria mais viável utilizar uma tecnologia NoSQL na modelagem dos dados, pois algumas estruturas de dados seriam flexíveis.

MongoDB é um banco de dados orientado a documentos que armazena dados em coleções semelhantes a JSON. É um projeto NoSQL e se distingue por possuir uma poderosa linguagem de consulta baseada em documento. Seu armazenamento de dados não possui estruturas rígidas, como tabelas, mas sim documentos vagamente definidos que permite acrescentar novos atributos aos documentos individuais sem que outros sejam alterados (IBM, 2017).

JSON é um formato utilizado para troca de dados entre sistemas, sendo fácil de ser interpretado tanto por humanos quanto por máquinas pois é construído com texto simples e legível.

> JSON (JavaScript Object Notation - Notação de Objetos JavaScript) é uma formatação leve de troca de dados. Está baseado em um subconjunto da linguagem de programação JavaScript, Standard ECMA-262 3a Edição -Dezembro - 1999. JSON é em formato texto e completamente independente de linguagem, pois usa convenções que são familiares às linguagens C e familiares, incluindo C++, C#, Java, JavaScript, Perl, Python e muitas outras. Estas propriedades fazem com que JSON seja um formato ideal de troca de dados. (JASON, 2017, p.).

3 OBJETIVOS

Tivemos como objetivo desenvolver um aplicativo híbrido através de um só código disponíveis às plataformas iOS, Android e Windows. Assim obtivemos uma ferramenta que oferece informação, de forma fácil e clara, sobre a qualidade da água que chega nas torneiras das casas e sobre a qualidade da água do rio Sorocaba.

Almejamos também o desenvolvimento de um portal administrativo que permitiu adicionar os valores dos indicadores da qualidade da água obtidos nos testes realizados mensalmente pelo SAAE e exibidos no aplicativo mobile, bem como também notícias, eventos e projetos.

4 MATERIAIS E MÉTODOS

Inicialmente foi realizado um estudo sobre as ferramentas e tecnologias que foram utilizadas durante o desenvolvimento do projeto. Para esse estudo foi utilizado o curso realizado na Fatec Sorocaba titulado Ionic 2 e o curso disponibilizado na plataforma Alura sobre AngularJS, onde foi possível conhecer essas tecnologias e também implementar exemplos que posteriormente viriam a ser úteis no desenvolvimento da aplicação.

Em seguida foi realizada a instalação de todos os programas e pacotes necessários para se criar uma aplicação com o *framework* Ionic 2 e AngularJs.

Após o período inicial de instalação, partiu-se para a criação dos arquivos do projeto e início do desenvolvimento do protótipo.

O protótipo foi desenvolvido de acordo com os requisitos e pesquisas levantados pela aluna Ana Paula Ribeiro como é descrito em seu relatório (RIBEIRO, 2016).

Logo partiu-se para a modelagem dos dados e a criação do portal administrativo para permitir criar e alimentar os dados no banco de dados e consultá-los no aplicativo mobile.

Por fim, foi realizado um teste automatizado no aplicativo utilizando a ferramenta Appium. Para a realização deste teste criou-se um projeto na linguagem Java, onde foi criada uma Classe que descreve em qual parte do aplicativo será realizado o teste. Foi escolhida a função de consulta da qualidade da água para ser testada. Primeiro informou-se qual a plataforma da aplicação, no nosso caso foi Android, em seguida o dispositivo que será executado os testes, que foi o Appium e por fim o caminho do aplicativo Android. Após informar essas configurações, partiu-se para escrever o código de automação utilizando os três métodos a seguir:

sendKeys() – preenche o campo com a informação que for passada por parâmetro.
click() – executa a ação de clique no aplicativo.
getText() – pega o texto do componente.

Então, com os métodos acima foi criado o teste que permitiu preencher o campo do bairro com o nome Centro, clicar no botão consultar e verificar se o resultado do teste foi igual ao resultado de comparação que foi informado no código, se os valores forem iguais o teste foi bem-sucedido.

Exemplo:

Bairro selecionado: Centro
Estação de comparação: ETA Éden
Estação resultante do teste: ETA Éden
Resultado: Teste bem-sucedido

Segue a classe criada para realizar o teste:

```
public class TesteExemplo {

	@Test
	public void test() throws MalformedURLException {

		File diretorioAplicacao = new File("C:\\app");
		File arquivoAplicacao = new File(diretorioAplicacao, "SoroAcqua.apk");

		DesiredCapabilities capacidade = new DesiredCapabilities();
		capacidade.setCapability(MobileCapabilityType.PLATFORM_NAME,
MobilePlatform.ANDROID);
		capacidade.setCapability(MobileCapabilityType.DEVICE_NAME,
"Android Emulator");
		capacidade.setCapability(MobileCapabilityType.APP,
arquivoAplicacao.getAbsolutePath());

		@SuppressWarnings("rawtypes")
		AndroidDriver driver = new AndroidDriver(new
URL("http://127.0.0.1:4723/wd/hub"), capacidade);

	driver.findElement(By.id("com.exemplo.soroacqua:id/cmbxBairro")).sendKeys("
Centro");
	driver.findElement(By.id("com.exemplo.soroacqua:id/btnConsultar")).click();
	Assert.assertEquals("ETA Éden",
driver.findElement(By.id("com.exemplo.soroacqua:id/txtEstacao")).getText());

	}

}
```

5 FERRAMENTAS TECNOLÓGICAS

5.1 Tecnologias de versionamento de códigos

5.1.1 Git

Esta ferramenta foi utilizada como um repositório para o projeto para trabalhar no controle de versões, o que permite que várias pessoas consigam contribuir ao mesmo tempo para o projeto sem o problema de implementar novas funcionalidades em versões desatualizadas.

O Git é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com tudo, desde projetos pequenos a muito grandes, com velocidade e eficiência.

Ele supera as ferramentas de Sistema de controle de versões como *Subversion*, CVS (*Concurrent Version System* – Sistema de versionamento concorrente), *Perforce* e *ClearCase* com recursos como ramificação local barata, áreas de armazenamento convenientes e vários fluxos de trabalho. (GIT, 2016).

5.2 Tecnologias de desenvolvimento

5.2.1 Ionic 2

O Ionic foi utilizado em nosso projeto para criar toda a parte visual utilizada na aplicação mobile, oferecendo diversas bibliotecas de CSS (*Cascading Style Sheets)* que é um mecanismo para adicionar estilo como por exemplo cores, fontes e espaçamentos a um documento *web*, JavaScript que é uma linguagem de programação interpretada e AngularJS que é um *framework* JavaScript.

A tecnologia Ionic é uma estrutura de desenvolvimento de aplicativos para dispositivos móveis híbridos, utilizando a linguagem de marcação HTML5 (*HyperText Markup Language*) que é uma linguagem de marcação utilizada na construção de páginas na *web*. Aplicativos híbridos são, essencialmente,

pequenos sites executados em um *shell* de navegador em um aplicativo que tem acesso à camada de plataforma nativa. *Shell* é um programa que recebe, interpreta e executa os comandos de usuário, aparecendo na tela como uma linha de comandos, representada por um interpretador de comandos, que aguarda na tela os comandos do usuário. Aplicativos híbridos têm muitos benefícios sobre aplicativos nativos puros, especificamente em termos de suporte de plataforma, velocidade de desenvolvimento e acesso a código de terceiros. (IONIC, 2016).

5.2.2 Apache Cordova

O plugin do Apache Cordova permitiu que fossemos capazes de acessar, através de códigos, as funções nativas do dispositivo, como a câmera e o GPS.

O Apache Cordova é um *framework* de desenvolvimento móvel de código aberto. Ele permite que se use tecnologias *web* padrões, tais como: HTML5, CSS3 e JavaScript para desenvolvimento de plataformas cruzadas. Os aplicativos são executados em ambientes isolados segmentados para cada plataforma e dependem de ligações API *(Application Programming Interface)* compatíveis com padrões para acessar as capacidades de cada dispositivo, como sensores, dados, *status* da rede. (CORDOVA, 2016).

O Cordova é utilizado para criar uma aplicação nativa capaz de abrir uma *webview* (nome dado a um tipo especial de navegador que começa a ser executado assim que a aplicação híbrida é aberta pelo usuário) para a execução do HTML, CSS e JavaScript criados pelo desenvolvedor. É dentro desse navegador que a aplicação é executada. Paralelo a isso, ele também tem o conceito de *plugins* que são usados para acessar os recursos do dispositivo.

Através do comando *build* do cordova, o código não nativo (HTML, CSS e JavaScript) da aplicação é empacotado dentro de uma aplicação nativa. O *build* pode ser considerado a principal funcionalidade do Cordova. A entrada do *build* são os arquivos HTML, CSS e JavaScript além de alguns arquivos de configuração do projeto, e a saída é uma aplicação nativa que, ao ser

executada, abre a *webview* que vai renderizar (processo pelo qual se obtém o produto final de um processamento digital qualquer) a aplicação.

Através do comando *plugin* do Cordova podemos acessar os recursos nativos da aplicação que podem ser adicionados e removidos.

5.2.3 NodeJS

Essa plataforma foi utilizada para a execução do JavaScript facilmente em servidores e porque para utilizar o Ionic 2 e desenvolver aplicações móveis com o Cordova é necessário tê-la instalada.

Node.js é uma plataforma construída sobre o motor JavaScript do Google Chrome (navegador de internet, desenvolvido pela companhia Google) para facilmente construir aplicações de rede rápidas e escaláveis. Node.js usa um modelo de I/O (*Input*/*output*) direcionada a evento não bloqueante que o torna leve e eficiente, ideal para aplicações em tempo real com troca intensa de dados através de dispositivos distribuídos (NODEJS, 2016).

5.2.4 ***NPM***

O NPM teve a finalidade de instalar e gerenciar o AngularJS e também gerenciar suas dependências. Exemplo: npm install angular.

NPM (*Node Package Manager* - Gerenciador de Pacotes do Node) é um repositório online para publicação de projetos de código aberto para o Node.js e é um utilitário de linha de comando que interage com este repositório online, que ajuda na instalação de pacotes, gerenciamento de versão e gerenciamento de dependências (NPM, 2016).

5.2.5 TypeScript

Para a criação de uma aplicação é necessário a utilização de uma linguagem de programação. Uma linguagem possui um conjunto de regras que possibilitam realizar as operações solicitadas pelo usuário, e para este projeto utilizou-se a linguagem TypeScript.

TypeScript é uma linguagem de programação criada e mantida pela Microsoft. Foi projetada para o desenvolvimento de grandes aplicações em JavaScript tanto no lado cliente ou servidor (Node.js). Basicamente é um subconjunto de JavaScript que acrescenta tipagem de objetos, estrutura de classes e é totalmente baseada em programação orientada a objeto.

A tipagem estática, que exige a declaração de quais dados poderão ser associados a cada variável antes de sua utilização, lhe permite realizar muitas operações de refatoração avançadas, como renomeação global, pesquisa de referência e auto complete de código.

Com TypeScript podemos escrever código utilizando uma estrutura fortemente tipada e ter este código compilado para JavaScript puro. (TypeScript, 2016).

O TypeScript traz o poder da tipagem para o seu código, com o intuito de minimizar erros, simplificar a injeção de dependências e facilitar testes.

5.2.6 Atom

O Atom foi escolhido para esse projeto por ser uma ferramenta leve e prática para a edição e criação dos arquivos (CSS, HTML, TypeScript) que contém os códigos fonte do projeto.

O Atom é um editor de texto moderno, acessível e de fácil usabilidade. É uma ferramenta que permite personalizar para fazer qualquer coisa, mas também usar de forma produtiva sem a necessidade de utilizar um arquivo de configuração (ATOM, 2016).

5.2.7 AngularJS

AngularJS foi o *framewowrk* utilizado no desenvolvimento do portal administrativo para a criação do design das páginas.

O AngularJS é um *framework* estrutural para aplicativos *web* dinâmicos. Ele permite que você use o HTML como linguagem de modelo e permite estender a sintaxe de HTML para expressar os componentes do aplicativo de forma clara e sucinta. A ligação de dados de AngularJS e a injeção de

dependência eliminam muito do código que você teria que escrever. E tudo acontece dentro do navegador, tornando-se um parceiro ideal com qualquer tecnologia de servidor (ANGULARJS, 2016).

5.2.8 MongoDB

Utilizamos o banco de dados MongoDB para o armazenamento dos dados que serão utilizados no aplicativo SoroAcqua.

MongoDB é um banco de dados orientado a documentos e possui multiplataforma NoSQL que armazena dados em documentos JSON com esquema dinâmico. Isso significa que você pode armazenar seus registros sem se preocupar com a estrutura de dados, como o número de campos ou tipos de campos para armazenar valores. Os documentos do MongoDB são semelhantes aos objetos JSON (MONGODB, 2017).

5.2.9 Appium

Utilizou-se o Appium para executar testes no aplicativo com a finalidade de achar algum erro em seu desenvolvimento que não seria facilmente detectado com testes manuais.

O Appium é uma estrutura de automação de teste de código aberto para uso com aplicativos nativos, híbridos e móveis.

Ele dirige aplicativos iOS, Android e Windows usando o protocolo WebDriver (APPIUM, 2017).

6 PROJETO DETALHADO DE SOFTWARE

6.1 Desenvolvimento

O portal foi desenvolvido em AngularJs e TypeScript integrado com o banco de dados MongoDB, ferramentas que permitiram uma maior produtividade e eficiência na codificação.

6.2 Código fonte

A seguir temos algumas imagens do código fonte de algumas das principais telas do portal:

~~6.2.1~~ *Código do Login*

~~6.2.1~~

A figura 1 ilustra o código utilizado para gerar a página de login onde é controlado o aceso ao portal por usuário. Nas figuras 2 e 3 é possível visualizar o código onde buscamos os dados através do JSON.

[S1] Comentário: Todas as figuras precisam ser chamadas no texto

Formatado: Fonte: (Padrão) Calibri, 11 pt, Não Itálico

Formatado: Normal, Espaçamento entre linhas: simples

Formatado: Fonte: Não Negrito, Cor da fonte: Automática

Formatado: Justificado, Recuo: Primeira linha: 1,27 cm, Espaçamento entre linhas: 1,5 linhas

Formatado: Fonte: Não Negrito, Cor da fonte: Automática

Figura 1- Login

```
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { UsuarioService } from '../usuario.service';
import { Usuario } from '../usuario';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.css']
})
export class LoginComponent implements OnInit {
  constructor(private usuarioservice:UsuarioService,private _router:Router) { }
  user:Usuario;
  ngOnInit() {
    this.user = new Usuario();
  }

  onSubmit(){
    this.usuarioservice.logIn(this.user).subscribe(respostafinal=>{
      if(respostafinal){
        this._router.navigate(['/noticia']);
      }
    },
    err=>{
      console.log(err);
    });
    this.user = new Usuario;
  }
}
```

Fonte: A autora (2017).

Figura 2- Login Service - Parte 1

```
@Injectable()
export class UsuarioService {
  logado = false;
  urlDestino = 'https://servidorsoroacqua.herokuapp.com/api/user/';

  constructor(private http:Http) {
    this.logado = !!localStorage.getItem('token');
  }

  logIn(data){
    var headers = new Headers({'Content-Type':'application/json'});
    //headers.append('Authorization','Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino+"login",data,options).
    map((res)=>{
      if (res.ok) {
          localStorage.setItem('token',res.json().token);
          localStorage.setItem('id',res.json().id);
          localStorage.setItem('nome',res.json().nome);
          this.logado = true;
      }
      return res.ok;

    })
    .catch((error:any)=>Observable.throw(error.json().error || 'Server error'));
  }
```

Fonte: A autora (2017).

Figura 3- Login Service - Parte 2

```
  signUp(data){
    var headers = new Headers({'Content-Type':'application/json'});
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino+"cadastrar",data,options)
                .map(res=>{
                  if (res.ok) {
                      localStorage.setItem('token',res.json().token);
                      localStorage.setItem('id',res.json().id);
                      localStorage.setItem('nome',res.json().nome);
                      this.logado = true;
                  }
                  return res.ok;
                }).catch(error=>Observable.throw(error.json().error||'Erro'));
  }

  isLoggedIn(){
    return this.logado;
  }

  logOut(){
    localStorage.removeItem('token');
    localStorage.removeItem('id');
    localStorage.removeItem('nome');
    this.logado = false;
  }
}
```

Fonte: A autora (2017).

6.2.2 Cadastro das Estações

A seguir serão apresentados nas figuras 4 e 5 os códigos utilizados para a criação da tela de cadastro de estações.

Figura 4- Cadastro das Estações

```
import { Component, OnInit } from '@angular/core';
import { Estacao } from '../estacao';
import { EstacaoService } from '../estacao.service';

@Component({
  selector: 'app-estacao',
  templateUrl: './estacao.component.html',
  styleUrls: ['./estacao.component.css']
})
export class EstacaoComponent implements OnInit {
  estacao:Estacao;
  constructor(private estacaoservice: EstacaoService) {  }

  ngOnInit() {
    this.estacao = new Estacao();
  }

  onSubmit(){
    this.estacaoservice.novaEstacao(this.estacao).subscribe(respostafinal=>{
      console.log(respostafinal);
      this.estacao = new Estacao();
    },
    err=>{
      console.log(err);
    }
  );
  }
}
```

Fonte: A autora (2017).

Formatado: Normal, Justificado, Recuo: À esquerda: 0 cm, Primeira linha: 1,27 cm

Formatado: Cor da fonte: Automática

Formatado: Cor da fonte: Automática

Formatado: Cor da fonte: Automática

Figura 5 - Estações Service

```
export class EstacaoService {
  urlDestino = 'https://servidorsoroacqua.herokuapp.com/api/estacao/';
  constructor(private http:Http) { }

  novaEstacao(estacao:Estacao){
    var headers = new Headers({'Content-Type':'application/json'});
    headers.append('Authorization','Bearer '+localStorage.getItem('token'));
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino,estacao,options).
    map(res=>{
      return res.ok;
    }).catch(error=>Observable.throw(error.json().error||'ERRO'));
  }

  todasEstacoes(){
    var headers = new Headers({'Content-Type':'application/json'});
    headers.append('Authorization','Bearer '+localStorage.getItem('token'));
    var options = new RequestOptions({headers:headers});
    return this.http.get(this.urlDestino,options).
    map(res=>{
      return res.json()
    }).catch(error=>Observable.throw(error.json().error || 'Erro'));
  }
}
```

Fonte: A autora (2017).

Formatado: À esquerda, Recuo: À esquerda: 0 cm

6.2.3 *Inserir Novos Valores para a Qualidade da Água*

É possível visualizar nas figuras 6 e 7 o código que insere os valores para a qualidade da água.

Formatado: Fonte: (Padrão) Arial, 12 pt, Cor da fonte: Automática

Formatado: Normal, Justificado, Recuo: À esquerda: 0 cm, Primeira linha: 1,27 cm, À direita: 0 cm

Formatado: Fonte: (Padrão) Arial, 12 pt, Cor da fonte: Automática

Figura 6 - Inserir Qualidade da Água

```
import { Component, OnInit } from '@angular/core';
import { EstacaoService } from '../estacao.service';
import { QualidadeService } from '../qualidade.service';
import { Estacao } from '../estacao';
import { Qualidade } from '../qualidade';

@Component({
  selector: 'app-qualidade',
  templateUrl: './qualidade.component.html',
  styleUrls: ['./qualidade.component.css']
})
export class QualidadeComponent implements OnInit {
  qualidade:Qualidade;
  estacaoes:Array<Estacao>;
  estacao:string;
  constructor(private estacaoservice:EstacaoService, private qualidadeservice:QualidadeService) {

  }

  ngOnInit() {
    this.estacaoservice.todasEstacoes().subscribe(respostafinal=>{
      this.estacaoes = respostafinal;
    });
    this.qualidade = new Qualidade();
  }

  onSubmit(){
    this.qualidadeservice.novaQualidade(this.qualidade).subscribe(respostaparcial=>{
      this.qualidadeservice.novaQualidadeEstacao(this.estacao,respostaparcial).subscribe(respostafinal=>{
          console.log(respostafinal);
          this.qualidade = new Qualidade();
      });
    });
  }
}
```

Fonte: A autora (2017).

Figura 7- Qualidade da Água Service

```
export class QualidadeService {
  urlDestino = 'https://servidorcorocoroa.herokuapp.com/api/';
  constructor(private http:Http) { }

  novaQualidade(qualidade:Qualidade){
    var headers = new Headers({'Content-Type':'application/json'});
    headers.append('Authorization','Bearer '+localStorage.getItem('token'));
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino+'qualidade/',qualidade,options).
    map(res=>res.json()._id).catch(error=>Observable.throw(error.json().error||'ERRO'));
  }

  novaQualidadeEstacao(estacao:string,qualidade:string){
    var headers = new Headers({'Content-Type':'application/json'});
    headers.append('Authorization','Bearer '+localStorage.getItem('token'));
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino+'estacao/'+estacao+'/novaqualidade/',{qualidade:qualidade},options).
    map(res=>{
      return res.json();
    }).catch(error=>Observable.throw(error.json().error||'ERRO'));
  }
}
```

Fonte: A autora (2017).

Formatado: Figura, Centralizado, À direita: 2 cm, Espaçamento entre linhas: simples

6.2.4 Inserir Notícia

Para inserir uma notícia utiliza-se os códigos demonstrados nas figuras 8, 9 e 10.

Formatado: Fonte: (Padrão) Arial, 12 pt, Cor da fonte: Automática

Formatado: Normal, Justificado, Recuo: À esquerda: 0 cm, Primeira linha: 1,27 cm, À direita: 0 cm

Figura 8- Inserir Notícia - Parte 1

```
import { Component, OnInit } from '@angular/core';
import { ImgurService } from '../imgur.service';
import { NoticiaService } from '../noticia.service';
import { Noticia } from '../noticia';
@Component({
  selector: 'app-noticia',
  templateUrl: './noticia.component.html',
  styleUrls: ['./noticia.component.css']
})
export class NoticiaComponent implements OnInit {

  constructor(private imgurservice:ImgurService, private noticiaservice:NoticiaService) { }
  noticia:Noticia;

  ngOnInit() {
    this.noticia = new Noticia();
  }

  file: File;
  res:string;
  caminho:string;
  onChange(event: EventTarget) {
        let eventObj: MSInputMethodContext = <MSInputMethodContext> event;
        let target: HTMLInputElement = <HTMLInputElement> eventObj.target;
        let files: FileList = target.files;
        let reader: FileReader;
        reader = new FileReader();
        this.file = files[0];
        var that = this;
        reader.onload = function(){
          that.res = reader.result.split(',')[1];
        };
        if (this.file) {
            reader.readAsDataURL(this.file);
        }
    }
```

Fonte: A autora (2017).

Figura 9 - Inserir Notícia - Parte 2

```
  salvar(){
    this.imgurservice.novaImagem(this.res).subscribe(resposta=>{
      this.caminho = resposta.data.link;
      this.noticia.imagem = this.caminho;
      this.noticia.data = new Date();
      this.noticiaservice.novaNoticia(this.noticia).subscribe(resfinal=>{
        console.log(resfinal);
        this.noticia = new Noticia();
      },
      err=>{
        console.log(err);
      });
    },
    err=>{
      console.log(err);
    });
  }
}
```

Fonte: A autora (2017).

Figura 10 - Notícia Service

```
export class NoticiaService {
  urlDestino = 'https://servidorsoroacqua.herokuapp.com/api/noticia/';
  constructor(private http:Http) { }

  novaNoticia(noticia:Noticia){
    var headers = new Headers({'Content-Type':'application/json'});
    headers.append('Authorization','Bearer '+localStorage.getItem('token'));
    var options = new RequestOptions({headers:headers});
    return this.http.post(this.urlDestino,noticia,options).
    map(res=>{
      return res.ok;
    }).catch(error=>Observable.throw(error.json().error||'ERRO'));
  }
}
```

Fonte: A autora (2017).

6.3 Estrutura do Banco de Dados

Criação das coleções:

```
db.createCollection("user")
db.createCollection("bairro")
db.createCollection("evento")
db.createCollection("projeto")
db.createCollection("noticia")
db.createCollection("diretor")
```

Exemplo de inserção de dados às coleções:

```
db.user.insert({
        "_id": "58bf3fae8272720011eb747c",
    "email": "soroacqua@soroacqua.com",
        "senha":
"{\"hash\":\"BL24tvstmuZ2OvHls3tfBK38HgK5mSJ6qFrM9f3VupZX5AyBV5bnU
A/yjnUt98zFK9CZwcRMp0HpOWkGFNHy4IXH\",\"salt\":\"DsrpG+tPfWBmSuHU
CZKP6dZE9I0TNkld4EZFjdLqeQvujxsrFbl3Sge4pqHdT09gdLhhw9xCrwKRPY/
y6fKavemF\",\"keyLength\":66,\"hashMethod\":\"pbkdf2\",\"iterations\":381022}",
    "nome": "Soroacqua",
    "usuario": "soroacqua"
})
db.noticia.insert({
        "_id": "58bf40e38272720011eb747d",
        "titulo": "Água",
        "descricao": "Água faz bem para a mente",
        "imagem":"http://www.unicompra.com.br/wp-
content/uploads/2016/02/1431100917aguacorrienteyaguamineraldiferenciasese
nciales01.jpg",
    "data": new Date(2017,02,26)
})
```

Formatado: Inglês (EUA)

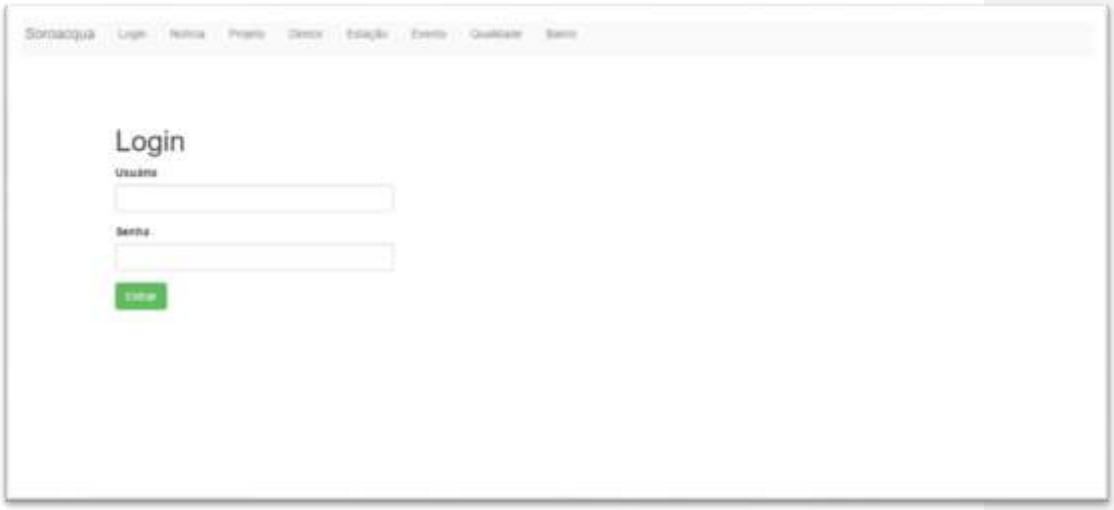
```
db.projeto.insert({
        "_id": "58cf5e9fd9f20e14dc6de213",
   "titulo": "Salvando lobo-guará",
        "descricao": "Evento para proteção desse animal tão relevante para a fauna sorocabana",
        "imagem":"http://amigodolobo.org/wp-content/uploads/2015/03/08_lobo-didatico-1024x695.jpg",
   "data": new Date(2017,05,06),
   "local": "Zoológico Quinzinho de Barros"
})
db.evento.insert({
        "_id": "58cf5d40d9f20e14dc6de211",
        "titulo": "Encontro ecológico",
        "descricao": "Encontro para biólogos e amantas da natureza",
        "imagem":"https://www.revistamissoes.org.br/wp-content/uploads/ecologia-e-desenvolvimento-sustentavel-7.jpg",
   "data": new Date(2017,05,15),
        "local": "Rua da Penha, 209 - Centro",
        "taxa": 5,
        "inscricao": "http://globo.com"
})
db.diretor.insert({
        "_id": "58cf5b267dacbd1d6c9dd8f9",
   "data_inicio": new Date(2017,02,05),
   "data_final": new Date(2017,01,30),
        "diretor": "Pedro Pereira",
   "prefeito": "Luis Mario"
})

db.bairro.insert({
        "_id": "58c59d0ea446e62d58742c4e",
        "descricao": "Júlio de Mesquita Filho",
        "estacao": [
          {
```

```
                "_id": "58c5f827c717971674ec77fb",
                "descricao": "Estação Júlio de Mesquita Filho",
                "endereco": "Rua 1, nº27 - Júlio de Mesquita",
                "qualidades": [
                        {
                                "_id": "58c5f0af119a911620532d17",
                                "oxigenio_dissolvido": "2%",
                                "coliformes_termotolerantes": "0.8%",
                                "ph": "3.4%",
                                "dbo": "1.3",
                                "temperatura": "31ºC",
                                "nitrogenio": "0.95%",
                                "fosforo": "2.35%",
                                "turpidez": "0.3%",
                                "residuo": "0.015%",
                                "aluminio": "2",
                                "alcalinidade_carbonatos": "1.3",
                                "alcalinidade_hidróxidos": "2.3",
                                "cloreto": "1.6",
                                "cor": "truva",
                                "dureza": "6.9",
                                "ferro": "5.6",
                                "fluoreto": "1.2",
                                "cloro_residual": "3.6",
                                "dqo": "5.6",
                                "dbo": "1.3",
                                "condutividade": "2.3",
                                "data": new Date(2017,03,20)
                        }
                ]
        }
    ]
})
```

6.4 Funcionalidades do portal administrativo

6.4.1 Login

Figura 11- Login



Fonte: A autora (2017).

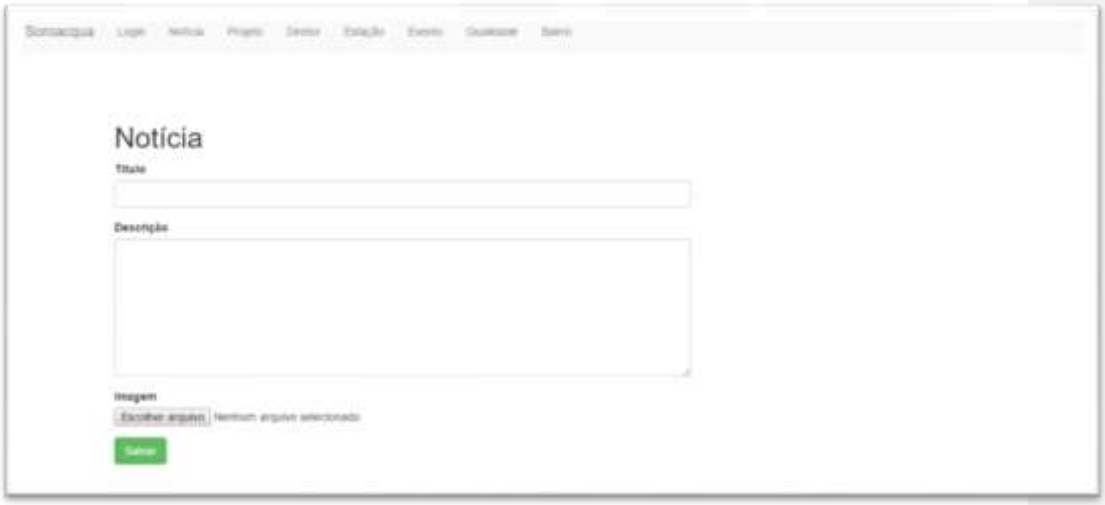
A figura 11 ilustra o login onde o usuário deverá informar um usuário e uma senha pré-cadastrados para ter acesso ao portal administrativo. Todas as telas do portal só poderão ser acessadas após essa autenticação de entrada.

6.4.2 Cadastros

As telas a seguir foram criadas a partir de formulários básicos onde é possível efetuar os cadastros com suas informações e em muitas telas também é possível inserir imagens que aparecerão no aplicativo.

6.4.2.1 *Cadastro de Notícias*

Figura 12 - Cadastro de Notícias



Fonte: A autora (2017).

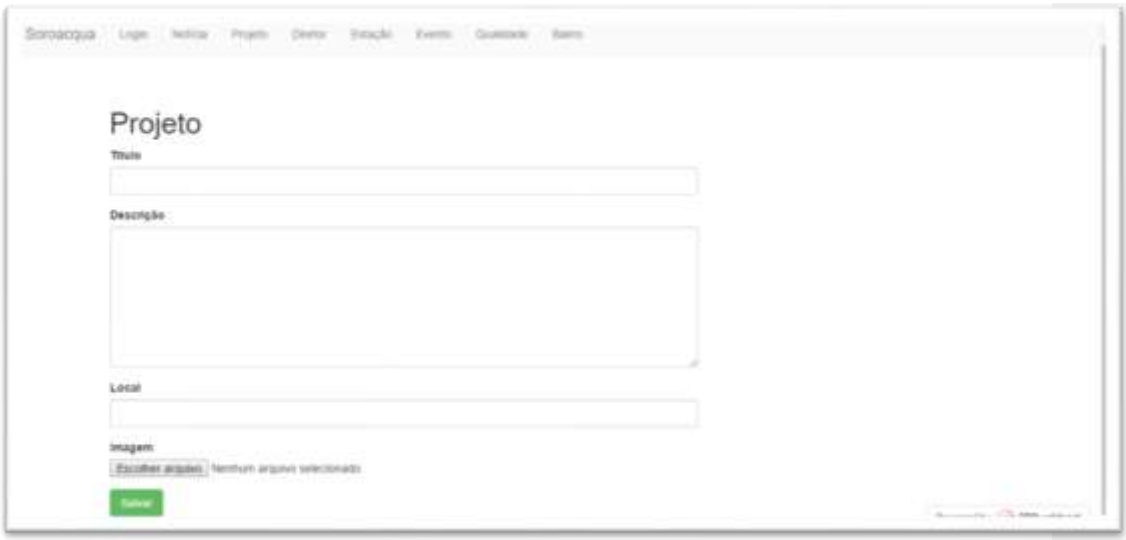
A figura 12 ilustra os campos necessários para realizar o cadastro de notícias que aparecerão na página de notícias do aplicativo como é mostrado na figura 13.

Figura 13- SoroAcqua - Tela de Notícias

Fonte: A autora (2017).

6.4.2.2 *Cadastro de Projetos*

Figura 14- Cadastro de Projetos



Fonte: A autora (2017).

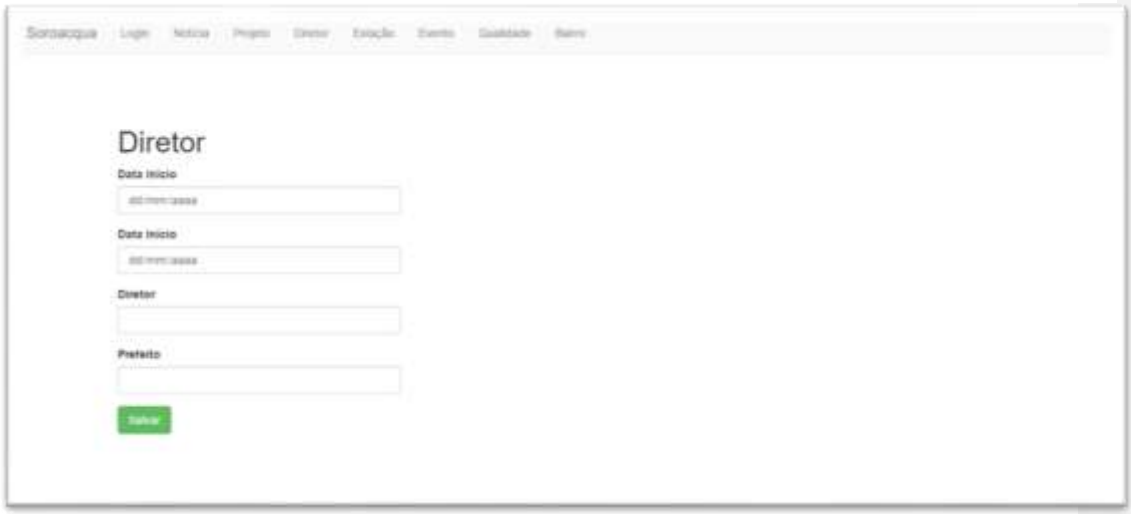
A figura 14 demonstra a tela que permite realizar o cadastro dos projetos com suas informações e uma imagem que aparecerá na parte de projetos do aplicativo como mostrado na figura 15.

Figura 15- SoroAcqua - Tela de Projetos

Fonte: A autora (2017).

6.4.2.3 *Cadastro de Diretores*

Figura 16- Cadastro de Diretores



Fonte: A autora (2017).

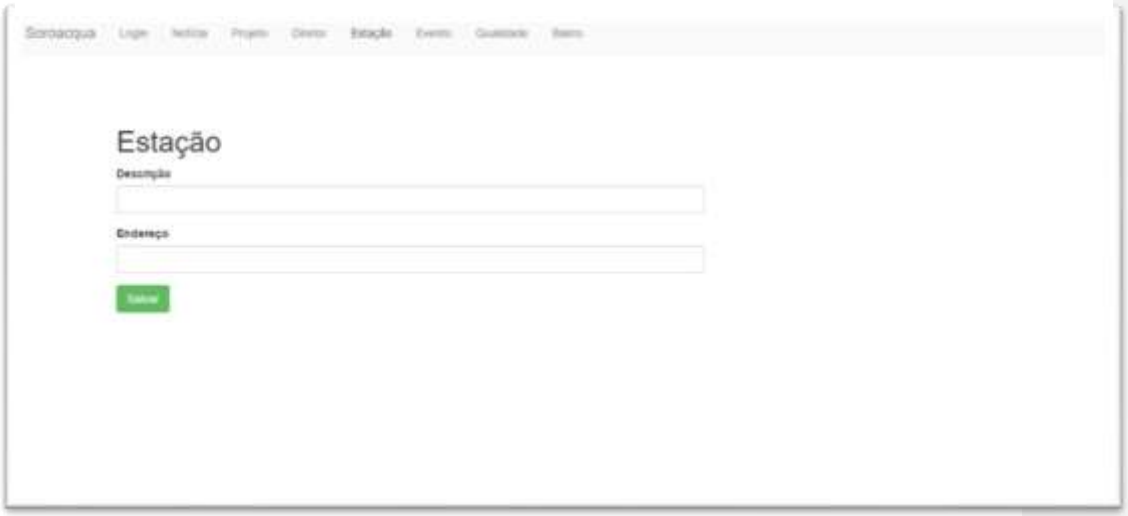
A figura 16 ilustra o cadastro de diretores que serão listados na tela do SAAE no aplicativo conforme exemplificado na figura 17.

Figura 17- SoroAcqua – Tela Sobre o SAAE

Fonte: A autora (2017).

6.4.2.4 *Cadastro de Estações*

Figura 18- Cadastro de Estações

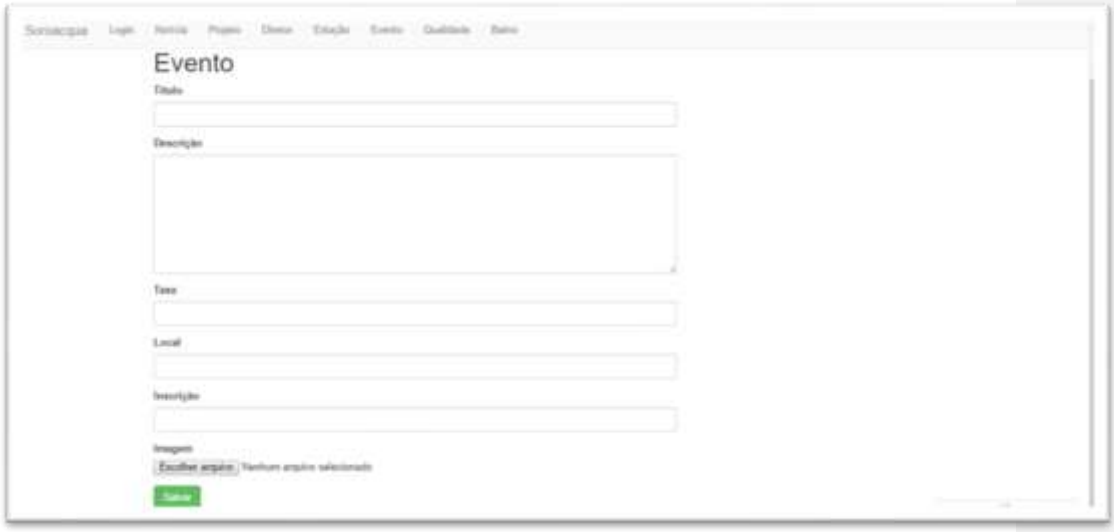


Fonte: A autora (2017).

A figura 18 ilustra o cadastro de estações que serão utilizadas para pesquisar a qualidade da água a partir do bairro do usuário.

6.4.2.5 *Cadastro de Eventos*

Figura 19- Cadastro de Eventos



Fonte: A autora (2017).

A figura 19 demonstra o cadastro dos eventos que serão divulgados no aplicativo como mostra a figura 20.

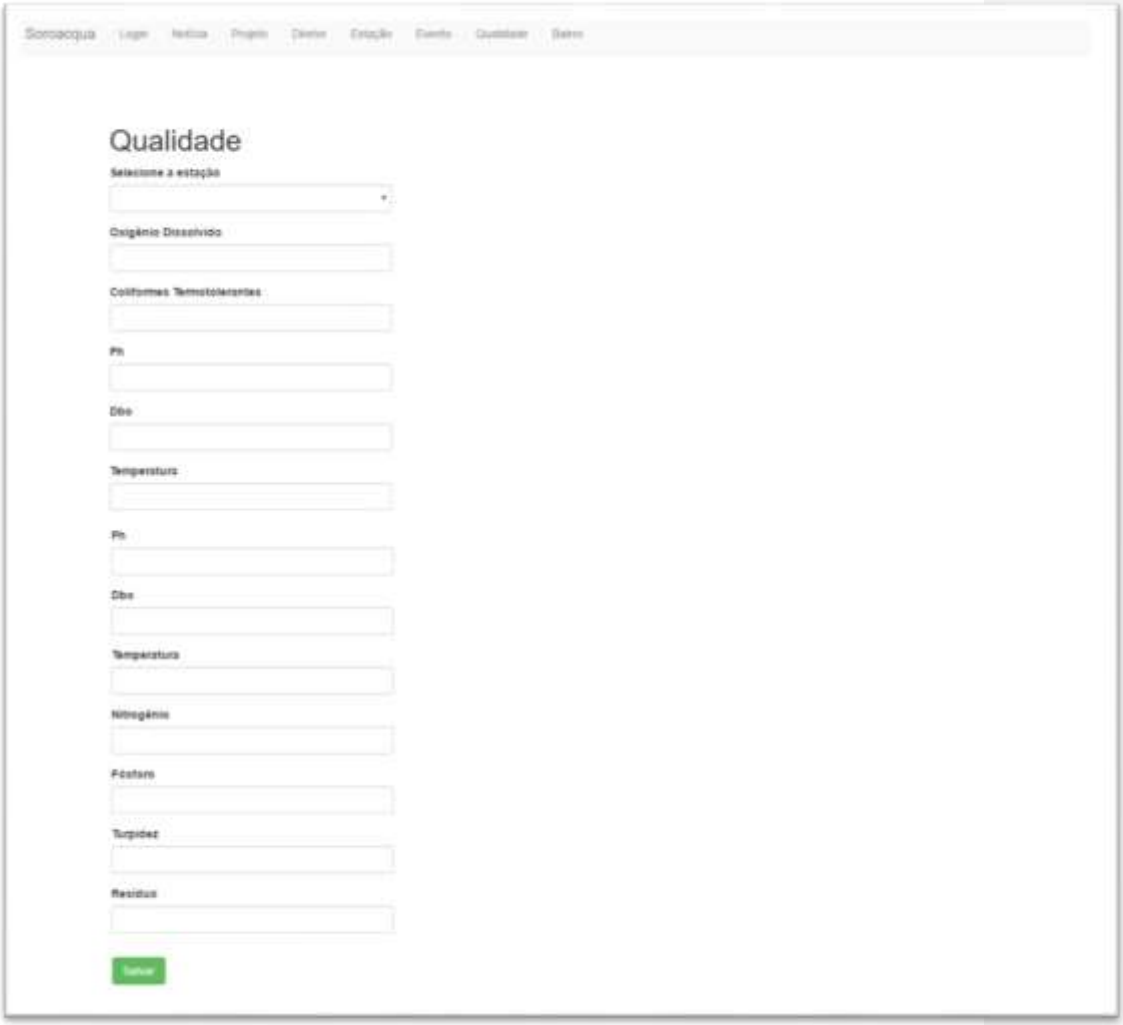
Figura 20- SoroAcqua – Tela de eventos

Fonte: A autora (2017).

6.4.2.6 *Cadastro de Qualidade*

A figura 21 ilustra a tela de cadastro da qualidade da água que tem como objetivo cadastrar as qualidades da água de acordo com os índices vindos diretamente do portal do SAAE, onde o usuário poderá acessá-las através da tela de consulta da qualidade da água (figura 22) e de acordo com o bairro selecionado.

Figura 21- Cadastro de Qualidade



Fonte: A autora (2017).

Figura 22- SoroAcqua – Tela de Consulta da Qualidade da Água

Fonte: A autora (2017).

6.4.2.7 *Cadastro de Bairros*

Figura 23- Cadastro de Bairros

Fonte: A autora (2017).

A figura 23 ilustra o cadastro dos bairros que serão listados na opção de consultada da qualidade da água como mostrado na figura 24.

Figura 24- Cadastro de Notícias

Fonte: A autora (2017).

7 RESULTADOS E DISCUSSÕES

Os resultados obtidos neste projeto foram a modelagem de dados e criação da base de dados do sistema, desenvolvimento do portal, desenvolvimento do aplicativo e realização de testes automatizados. Para garantir a melhor forma de desenvolvimento e um portal e aplicativo funcional, foram estudadas várias metodologias de desenvolvimento e ferramentas, tais como: Ionic 2, AngularJs, Node.js, Atom, Codova, TypeScript, Git, MogoBD e Appium que auxiliaram e garantiram o desenvolvimento e conclusão do projeto.

8 CONCLUSÕES

A partir do início do desenvolvimento deste projeto espera-se que essa aplicação seja de grande ajuda para a população de Sorocaba. Trabalhamos com o desenvolvimento de um aplicativo híbrido, que é desenvolver um aplicativo que possui somente um código fonte e pode ser compilado para várias plataformas, sem a necessidade de gerar três projetos separados.

Do ponto de vista educacional, trabalhar com uma ferramenta de desenvolvimento hibrida foi uma experiência nova e agregadora, pois foi possível criar uma tela da aplicação e realizar testes nas plataformas IOS, Android e Windows ao mesmo tempo. No início da pesquisa, surgiu a dificuldade em relação a tecnologia a ser utilizada, entretanto, a partir de um curso do *framework* Ionic 2 realizado na Fatec Sorocaba percebeu-se que essa seria uma tecnologia que agregaria valor ao desenvolvimento.

Por fim, a pesquisa se finaliza com a elaboração de um portal administrativo no qual é possível preencher os dados que serão exibidos no aplicativo SoroAcqua, como também seu banco de dados. Os testes automatizados realizados utilizando a ferramenta Appium foram satisfatórios de acordo com o objetivo inicial de entregar um aplicativo funcional e híbrido.

9 SUGESTÕES PARA TRABALHOS FUTUROS

Como este trabalho está em sua primeira versão, existem algumas melhorias que poderão ser realizadas em uma segunda fase de desenvolvimento deste projeto. Entre elas estão: integração direta com o Banco de Dados do SAAE, realização de consultas ao aplicativo sem a necessidade de utilizar a internet e implementação de segurança sobre os arquivos do aplicativo de modo a proteger seu código fonte.

REFERÊNCIAS

ANGULARJS. 2016. Disponível em: <https://pt.wikipedia.org/wiki/AngularJS>. Acesso em: 12 out. 2016.

APACHE Cordova. 2015. Disponível em: <https://cordova.apache.org/>. Acesso em: 12 out. 2016.

APPIUM: automation for apps. [2017]. Disponível em: <http://appium.io/>. Acesso em: 02 fev. 2017.

ATOM: a hackable text editor for de 21st century. [2016]. Disponível em: <https://atom.io/>. Acesso em: 12 out. 2016.

BEZERRA, F. T. **Desvendando o desenvolvimento de aplicativos**. 2016. Disponível em: <http://congressodemobile.com.br/wp-content/ uploads/ 2016/06/Desvendando-o-Desenvolvimento-de-Aplicativos-para-iOS.pdf> Acesso em: 20 jan. 2017.

CEVALLOS, E. A. **Case study on mobile applications UX: effect of the usage of a cross platform development framework.** 2014. 108 p. Thesis (Master Degree in Software Engineering) — Universidad Politécnica De Madrid, Madrid, jun. 2014. Disponível em:<http://oa.upm.es/30422/1/EMSE-2014-05_Esteban_Angulo-1.pdf>. Acesso em: 16 fev. 2017

GIT: distributed-even-if-your-workflow-isnt. [2016]. Disponível em: <https://git-scm.com/> Acesso em: 14 set. 2016.

IONIC. [2016]. Disponível em: <http://ionicframework.com>. Acesso em: 12 out. 2016.

IBM. **Explore o MongoDB.** 2017. Disponível em: <https://www.ibm.com/ developerworks/br/library/os-mongodb4/ >. Acesso em: 28 maio 2017.

INTRODUÇÃO ao JSON. [s.d.]. Disponível em: < http://www.json.org/json-pt.html >. Acesso em: 10 abr. 2017.

JAVASCRIPT. 2016. Disponível em: <https://www.javascript.com >. Acesso em 10 set. 2016.

MONGODB. **MongoDB atlas database as a service.** 2016. Disponível em: <https://www.mongodb.com/>. Acesso em: 12 out. 2016.

NODE.JS. 2016. Disponível em: <https://nodejs.org/>. Acesso em: 12 out. 2016.

NPM. 2016. Disponível em <http://nodebr.com>. Acesso em: 14 de out. 2016.

PESSUTTO, D. **Borgundfjorden.no:** o portal para dados ambientais. 2016. 46f. Trabalho de Conclusão de Curso (Tecnologia em Análise e Desenvolvimento de Sistemas) – Faculdade de Tecnologia José Crespo Gonzales, Sorocaba, 2016.

.

PORTAL DA QUALIDADE DAS ÁGUAS. **Indicadores de qualidade:** índice de qualidade das águas (IQA). Brasília, DF: Agência Nacional de Águas, 2016. Disponível em: <http://portalpnqa.ana.gov.br/indicadores-indice-aguas.aspx>. Acesso em: 10 nov. 2016.

PREZOTTO, E.; BONIATI, B. Estudo de frameworks multiplataforma para desenvolvimento de aplicações mobile híbridas. In: ENCONTRO ANUAL DE TECNOGIA DA INFORMAÇÃO, 5., 2014, Frederico Westphalen, RS. **Anais...** Santa Maria: UFMS, 2014.

RIBEIRO, A. **SoroAcqua:** aplicativo para divulgação de informações sobre a água de Sorocaba. 20f. Sorocaba: Faculdade de Tecnologia José Crespo Gonzales, 2016.

SAAE SOROCABA. **Qualidade da água de Sorocaba.** Sorocaba, 2016. Disponível em: <http://saaesorocaba.com.br/>. Acesso em: 15 de set. 2016.

SAAE SOROCABA. **Tratamento do esgoto em Sorocaba**. Sorocaba, 2016. Disponível em: <http://saaesorocaba.com.br/site/?page_id=205>. Acesso em: 12 set. 2016.

SCHMITZ, D; LIRA, D. **AngularJS na prática**. [S.l.]: Leanpub, c2013.

SMITH, W. S.; RIBEIRO, C. A. (Org.). **Parque Natural Municipal corredores de biodiversidade:** pesquisas e perspectivas futuras. Sorocaba: Secretaria do Meio Ambiente, 2015. Disponível em: <http://meioambiente.sorocaba.sp.gov.br/wp-content/uploads/2016/06/livro-parque-da-bio-verso-completa.pdf >. Acesso em: 20 set. 2016.

TYPESCRIPT Preview. 2016. Disponível em <http://www.diullei.com/TypeScript-ptBR/>. Acesso em: 12 out. 2016.

WEYL, E. **Mobile HTML5:** usando o que há de mais moderno atualmente. São Paulo: Novatec, 2014.

W3C Brasil. Disponível em: <http://www.w3c.br/Home/WebHome>. Acesso em: 15 out. 2016.

~~Todas as referências são sites?~~