FACULDADE DE EDUCAÇÃO TECNOLÓGICA
DO ESTADO DO RIO DE JANEIRO

BRUNO MOURA JACCOUD DE ALMEIDA
JAMILE CELENTO ESPERANÇA

PLATAFORMA DE CATALOGAÇÃO DE LIXO ELETRÔNICO COM SUGESTÕES DE REUSO

RIO DE JANEIRO
2018

BRUNO MOURA JACCOUD DE ALMEIDA E JAMILE CELENTO ESPERANÇA

PLATAFORMA DE CATALOGAÇÃO DE LIXO ELETRÔNICO COM SUGESTÕES DE REUSO

Projeto de trabalho de conclusão de curso apresentado como parte das atividades para obtenção do título de Tecnólogo em Análise e desenvolvimento de sistemas da Faculdade de educação tecnológica do estado do Rio de Janeiro.
Orientadores:
Prof. Ricardo Marciano dos Santos
Prof. Roberto Cardoso Freire da Silva

RIO DE JANEIRO
2018

BRUNO MOURA JACCOUD DE ALMEIDA E JAMILE CELENTO ESPERANÇA

PLATAFORMA DE CATALOGAÇÂO DE LIXO ELETRONICO COM SUGESTÕES DE REUSO

Projeto de trabalho de conclusão de curso apresentado como parte das atividades para obtenção do título de Tecnólogo em Análise e desenvolvimento de sistemas da Faculdade de educação tecnológica do estado do Rio de Janeiro.
Orientadores:
Prof. Ricardo Marciano dos Santos
Prof. Roberto Cardoso Freire da Silva

Aprovado em:

BANCA EXAMINADORA

_____
Prof. Ricardo Marciano dos Santos
Orientador
FAETERJ - RIO

_____
Prof. Roberto Cardoso Freire da Silva
Co-orientador
FAETERJ - RIO

_____
Prof(a). Maria Cláudia Roenick Guimarães
Banca
FAETERJ - RIO

_____
Prof(a). Rosangela de Sena Almeida
Banca
FAETERJ - RIO

RIO DE JANEIRO
2018

AGRADECIMENTOS

Aos nossos pais

Agradecemos, primeiramente aos nossos pais por terem ajudado tanto nessa etapa e estarem sempre conosco.

Aos professores (as)

Agradecemos a todos os professores que nos acompanharam nessa jornada enquanto universitários e foram essenciais à nossa formação como profissionais e participaram de nosso crescimento como indivíduos.

Aos nossos amigos (as)

Por fim, agradecemos todos os amigos que estiveram conosco nessa jornada, vocês com certeza são parte dessa vitória.

RESUMO

Este trabalho tem como finalidade, disponibilizar pela internet informações sobre componentes eletrônicos – que normalmente são descartados no lixo comum – o que pode ser evitado, utilizando-os para os mais diversos fins, desde pequenos projetos pessoais utilizando IoT ou até mesmo como material de ensino para o estudo de ciências e tecnologia. A disponibilidade destes componentes será catalogada na rede por meio de uma aplicação web utilizando JavaScript, e C#. A aplicação desenvolvida trata-se de um sistema onde é possível incluir e consultar em um banco de dados, registros de componentes eletrônicos disponíveis, sua origem e sugestões de projetos que podem ser desenvolvidos através do reuso de seus componentes.

Palavras-chaves: E-Waste. Lixo Eletrônico. Reuso. Componentes Eletrônicos. Sustentabilidade.

ABSTRACT

The purpose of this work is to make information about electronic components available on the internet, which are usually discarded in the common garbage, this can be avoided by using them for a variety of purposes, from small personal projects using IoT or even teaching material for the study of science and technology. The availability of these components will be catalogued on the network through a web application using JavaScript, and C #. The application developed is a system where it is possible to include and consult in a database, electronic records of available components, their origin and suggestions of projects that can be developed through the reuse of its components.

Keywords: E-Waste. Junk. Reuse. Electronic Components. Sustainability.

LISTA DE IMAGENS

LISTA DE ABREVIATURAS E SIGLAS

AJAX	Asynchronous Javascript and XML
API	Application Programming Interface
CIL	Common Intermediate Language
CLR	Common Language Runtime
COOL	C-Like Object Oriented Language
DTO	Data Transfer Object
ECMA	European Computer Manufacturers Association
E-WASTE	Eletronic Waste
FGV	Fundação Getúlio Vargas
GSMA	Global System Mobile Association
MVC	Model-View-Controller
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
IBGE	Instituto Brasileiro de Geografia e Estatística

INEP Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira

IOT Internet of Things

JSON Javascript Object Notation

ONU Organização das Nações Unidas

POCO Plain Old CLR Object

UNU-IAS United Nations University Institute for the Advanced Study of Sustainability

VB Visual Basic

XML Extensible Markup Language

SUMÁRIO

INTRODUÇÃO

O contexto atual de acelerada produção voltada à informação torna indispensável o uso da tecnologia para a vida em sociedade. Os computadores, tablets e smartphones são hoje parte de praticamente todos os âmbitos da vida da maioria dos cidadãos.

Verificando a extrema importância dessas ferramentas na contemporaneidade, deve-se pensar também, na quantidade existente desses aparelhos e em seu descarte. Tomando como exemplo os smartphones, cuja quantidade ultrapassa 7 bilhões de aparelhos ativos no mundo (União Internacional de Telecomunicações, 2015), nos fazendo imaginar as montanhas de lixo que tais aparelhos inutilizados possam gerar.

Segundo a Associação de Empresas da Indústria Móvel (2014), o Brasil é atualmente o país da América Latina que mais gera lixo eletrônico, chegando a 1,4 milhão de toneladas de resíduos, considerando somente o ano de 2014. De acordo com o Portal Exame (2010), cada brasileiro descarta 0,5 kg de lixo eletrônico em média por ano .Mais absurdo que esses números é a grande quantidade de lixo eletrônico não revertido de nenhuma maneira ao desenvolvimento de novos sistemas e dispositivos, que poderiam ser produzidos a um preço acessível - possibilitando até mesmo seu uso em escolas e universidades públicas para fins educativos e de instrução no estudo de novas tecnologias de modo sustentável e amigável à natureza.

Pretende-se por meio deste trabalho propor um sistema de catalogação de lixo eletrônico facilitando o seu uso na criação de novos produtos voltados à IoT com a total e única finalidade de ser objeto de estudo de instituições de ensino público, de modo que o custo não será um obstáculo para a aquisição de material e instrução nesse tema.

Tratando-se de um dos assuntos de maior visibilidade no mundo atual da tecnologia e também um mercado promissor para pesquisas científicas e criação de novos sistemas e aplicações, a possibilidade de levar esse tipo de conhecimento, ainda que em nível básico, a alunos de instituições públicas serve de porta de entrada para os futuros pesquisadores e desenvolvedores ao contato com esse tipo de conhecimento.

1.1. OBJETIVOS

Desenvolver um software que possibilite o compartilhamento de informações sobre componentes eletrônicos para descarte, visando evitar o desperdício de material no meio ambiente, permitindo a utilização das partes encontradas em novos projetos e soluções. Para tal, espera-se que a aplicação proposta esteja implementada ao final deste trabalho, além de estar hospedada na internet para posterior acesso e validação.

1.2. MOTIVAÇÃO E JUSTIFICATIVA

O Brasil está entre os países que mais gera lixo eletrônico, segundo dados do Global E-Waste Monitor 2017, em 2016 o Brasil gerou 1,5 milhão toneladas de lixo eletrônico (EXAME, 2018).

O Global E-Waste Monitor é um relatório internacional elaborado pela Universidade das Nações Unidas em parceria com a União Internacional das Telecomunicações e a International Solid Waste Association, que no Brasil é representada pela Associação Brasileira de Empresas de Limpeza Pública e Resíduos Especiais.

De acordo com a pesquisa, grande parte desse volume não é reciclado e tem destinação inadequada, contaminando o meio ambiente e trazendo riscos à saúde.

Como alternativa na diminuição do impacto causado, está a reutilização desse lixo eletrônico, como objeto de aprendizagem dentro de um contexto pedagógico e uma forma de incentivo ao ensino, de uma forma sustentável.

1.3. MÉTODO DE PESQUISA

Para elaboração deste trabalho será utilizada a pesquisa bibliográfica, baseada em livros, revistas, sites e artigos relacionados com o tema em questão, a fim de promover o desenvolvimento de novos produtos criados a partir do lixo eletrônico, através de sua catalogação e centralização do conhecimento, para que possam ser

utilizados nas salas de aula para despertar o interesse dos alunos e incentivar a reutilização de e-waste.

1.4. PROBLEMA

Tendo o posto do segundo país que mais abandona lixo eletrônico dentre os países emergentes, ficando atrás apenas da China, segundo dados da ONU, o Brasil ainda não tem nenhuma estratégia para amenizar essa situação.

Em 2015 o Brasil contava com 154 milhões de smartphones e 152 milhões de computadores (FGV, 2015), número que só aumenta a cada ano e consequentemente a produção de e-waste também.

Apenas 20% dos resíduos eletrônicos descartados em 2016 foram reciclados, a despeito do alto valor agregado dos materiais que compõem alguns equipamentos, como cobre, platina, prata, ouro e outros matérias recuperáveis (Global E-waste Monitor, 2017).

A expectativa é que em 2018 a América Latina produzirá 4,8 mil toneladas de lixo eletrônico, 10% do total global segundo a GSMA e do Instituto Universitário das Nações Unidas para o Estudo Avançado da Sustentabilidade (UNU-IAS).

A produção de lixo eletrônico global deve aumentar em torno de 17% e superar a estimativa de 50 milhões de toneladas por ano até 2021 (Global E-waste Monitor, 2017).

2. TECNOLOGIAS, CONCEITOS E FRAMEWORKS

Neste capítulo são citadas as tecnologias, os conceitos e os frameworks que são utilizados no escopo deste trabalho, durante o processo de desenvolvimento da aplicação.

2.1. APLICAÇÃO WEB

Uma aplicação web é caracterizada pela distribuição de informação no modelo Cliente/Servidor, é camada interativa baseada no princípio da hiperligação entre documentos.

Uma aplicação web é formada por dois tipos de arquivos:

- HTML puro ou com scripts;
- Arquivos diversos como bases de dados, código executável e imagens.

Nessa arquitetura distribuída temos um servidor que é capaz de receber requisições que utilizam o protocolo HTTP e as tratar conforme seu algoritmo interno, retornando o resultado desse processo através de uma resposta HTTP para o browser do cliente que executou a requisição.

Apesar de termos o HTML como formato de resposta mais comum para requisições, é possível encontrar sistemas que enviam variados formatos de resposta, como os formatos JSON e XML, que são mais comuns nas respostas de aplicações para integração e comunicação entre sistemas, essas aplicações são chamadas de Web Services.

Atualmente a arquitetura de projetos mais utilizada continua sendo a arquitetura Multicamadas (SEBESTA, 2012), onde a aplicação é dividida em camadas com responsabilidades especificas na execução da aplicação.

O padrão de desenvolvimento de aplicações web mais popular continua sendo o modelo Model-View-Controller (MVC), que tem como característica a separação da aplicação em três partes.

- Model: Representa os dados da aplicação.

- View: É a representação da interface visual da aplicação.

- Controller: Componente responsável por gerenciar a interação do usuário com a aplicação.

Apesar de ser o mais utilizado o desenvolvimento web evolui continuamente, e tem diversos outros tipos de modelos, arquiteturas, paradigmas e padrões.

2.2. C#

Em 1999 ocorria o desenvolvimento da plataforma .NET Framework, uma iniciativa da Microsoft do que passaria a ser uma plataforma única para de desenvolvimento e execução de aplicações, de forma que qualquer código gerado utilizando .NET Framework possa ser executado em qualquer dispositivo que possua o mesmo.

Dessa forma o código deixa de ser desenvolvido para um sistema especifico ou dispositivo especifico e sim para a plataforma .NET.

O .NET Framework é baseado em um dos princípios utilizados pelo Java, os programas desenvolvidos para .NET são compilados duas vezes, na primeira gerando um código Common Intermediate Language (CIL), que é compilado pela Common Language Runner (CLR) para gerar o código que será lido pela máquina de acordo com a plataforma.

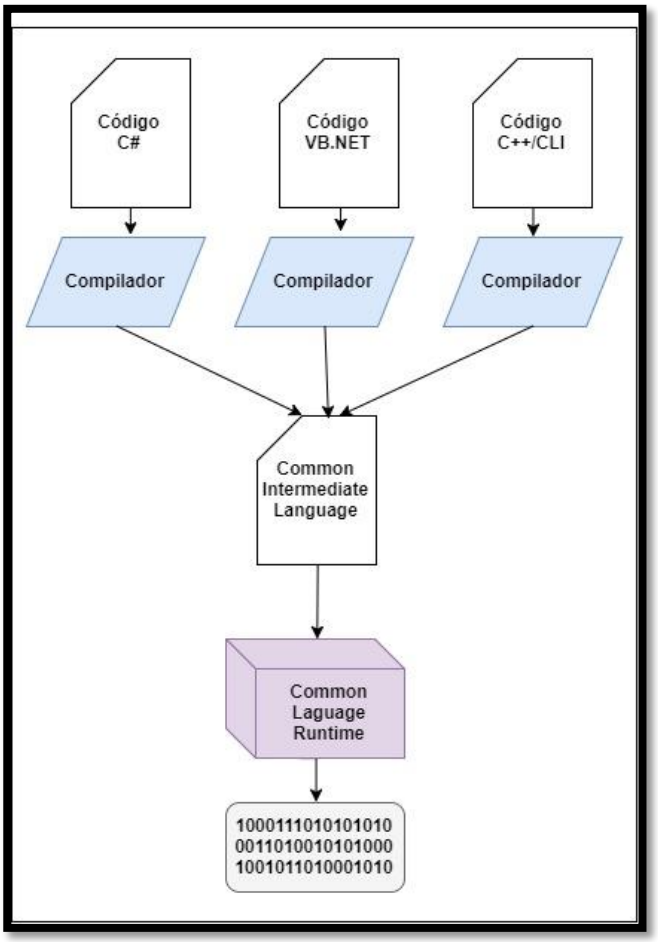

Figura 1 - Funcionamento do .NET Framework

No decorrer do desenvolvimento do .NET Framework foi iniciado o projeto que originaria a linguagem C# com o nome de projeto COOL (C-like Object Oriented Language). Tendo como engenheiro chefe Anders Hejlsberg.

Segundo Hejlsberg a CLR, que é o componente de máquina virtual da plataforma .NET Framework foi fundamentada em falhas das linguagens mais utilizadas na época como Java, C, C++, SmallTalk, Delphi e VB. O que acabou por basear o design da linguagem.

Três anos depois, em 2002 o projeto COOL foi lançado com o nome de C# juntamente com o .NET Framework.

O C# é uma linguagem orientada a objetos fortemente tipada, que permite a criação dos mais diversos tipos de aplicações que rodam no .NET Framework. Por ter uma sintaxe muito parecida com C, C++ e Java acaba sendo de fácil aprendizado.

Atualmente o C# tem sido utilizado para criação de aplicações web, aplicativos mobile e aplicações desktop.

2.3. JAVASCRIPT

O Javascript é uma linguagem interpretada, pois seu código fonte é executado por um interpretador e após isso executado pelo sistema operacional, e é mais conhecida como uma linguagem de script para páginas web. Tem a capacidade de uma linguagem orientada a objetos baseada em protótipos, ou seja, o processo de reutilização de dados também conhecido como herança, é aplicado através da clonagem de objetos existentes, os protótipos. Por se tratar de uma linguagem de script dinâmica acaba sendo multiparadigma, suportando orientação a objetos, programação imperativa e funcional.

Em 1995 o Javascript foi criado por Brendan Eich, na época engenheiro na Netscape. Ele foi lançado juntamente com o Netscape 2 no início de 1996. Originalmente seu nome seria Livescript, mas foi renomeado na tentativa de aproveitar a popularidade da linguagem da Sun Microsystems, o Java, o que gera até hoje uma certa confusão com relação ao envolvimento das linguagens.

Alguns meses depois a Netscape inseriu o Javascript na ECMA International, uma organização de padronização europeia, o que resultou na primeira edição do ECMAScript, a versão padronizada e estabilizada pela European Computer Manufacturer’s Association (ECMA), segundo a especificação ECMA-262 esse é o nome oficial da linguagem, apesar de não ser amplamente conhecido.

Ao contrário do que o nome pode indicar Javascript não é Java interpretado.

Objetos são criados programaticamente em Javascript, incluindo métodos e propriedades em tempo de execução, diferente do que acontece em linguagens compiladas como C++ e Java. Uma vez que um objeto é construído ele pode servir como protótipo para criar objetos similares.

O JavaScript é executado no lado do cliente, e pode ser usado para definir o design e programar como a página deve se comportar na ocorrência de um evento, por

se tratar de uma linguagem que também compartilha características de sintaxe com Java e C++ é de fácil aprendizado.

Com o desenvolvimento da linguagem diversos recursos foram adicionados, como a padronização de uma notação para seus objetos (JSON), a capacidade de efetuar requisições assíncronas (AJAX), além disso por ter sido adotada por uma grande comunidade de desenvolvedores foram e continuam sendo criadas diversas bibliotecas e frameworks.

2.4. OUTRAS TECNOLOGIAS E ARQUITETURA

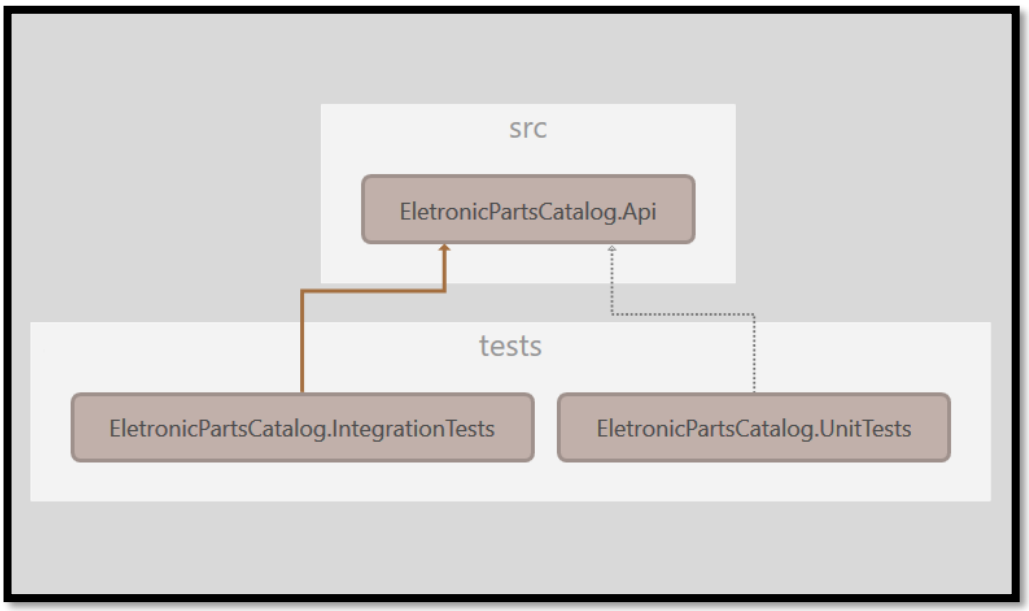

Figura 2 - Diagrama de dependências do back-end

Para o desenvolvimento do sistema de catalogação e todos os seus componentes foram utilizadas as linguagens de programação: C# e Javascript. Para armazenar os dados foi utilizado o SQL Server e para a criação de testes unitários foi utilizado o XUnit.

Mais especificamente foram utilizados os frameworks: .Net Core, React, Redux, Asp Net MVC e Entity Framework Core.

O projeto foi arquitetado pronto para microsserviços, sendo a aplicação final dividida em duas partes, o back-end e o front-end.

O front-end é tudo envolvido com o que o usuário vê e interage, sendo geralmente uma interface gráfica. Já o back-end é a parte da aplicação responsável por fazer o trabalho de acesso ao banco e processamento de dados, e não é visível ao usuário.

O estilo de arquitetura de microsserviços é uma abordagem que desenvolve um aplicativo único como uma suite de pequenos serviços, cada um executando seu próprio processo e se comunicando através de mecanismos leves, muitas vezes em uma API com recursos HTTP.

No back-end temos a API que contém algumas seções, como a seção de acesso ao banco de dados. Foi decidida a utilização de Code First no desenvolvimento da estrutura de banco de dados, por meio do Entity Framework Core.

O Entity Framework faz uma ponte entre as classes POCO e o banco de dados utilizando um container que chamamos de Contexto. O contexto é o responsável por mapear as classes com o banco de dados e também por informar ao Entity Framework quem é o banco de dados, através da string de conexão, sendo necessário trocar somente a string de conexão para mudar de banco. Nenhum tipo de alteração no código seria necessário.

A seção de domínio contém as classes que representam as entidades da aplicação, que são utilizadas na API, ou seja componentes do domínio de negócio da aplicação, por exemplo, usuários, projetos e componentes eletrônicos.

API é o acrônimo de Application Programming Interface. Trata-se de um conjunto de métodos e padrões de programação para acesso a um software ou plataforma baseado na Web. Uma API permite a troca de informações entre um ou mais sistemas.

Há ainda a camada de segurança, responsável pela autenticação da aplicação dessa forma, somente usuários cadastrados podem fazer alterações em dados da base da aplicação. Isso se torna necessário para que haja um controle mínimo da qualidade das informações compartilhadas.

A seção de funcionalidades é responsável por manipular o acesso ao banco de dados e definir mensagens de erro caso a operação iniciada não tenha sucesso

acessando os dados armazenados por meio de interface do contexto e sendo intermediaria entre o acesso ao banco e a interface gráfica.

Também no back-end foram feitas implementações de controladores, e a utilização do Swagger para facilitar o consumo de dados por outras aplicações.

O Swagger é uma especificação aberta para APIs. O documento do Swagger especifica a lista de recursos que estão disponíveis na API e as operações que podem ser chamadas nesses recursos.

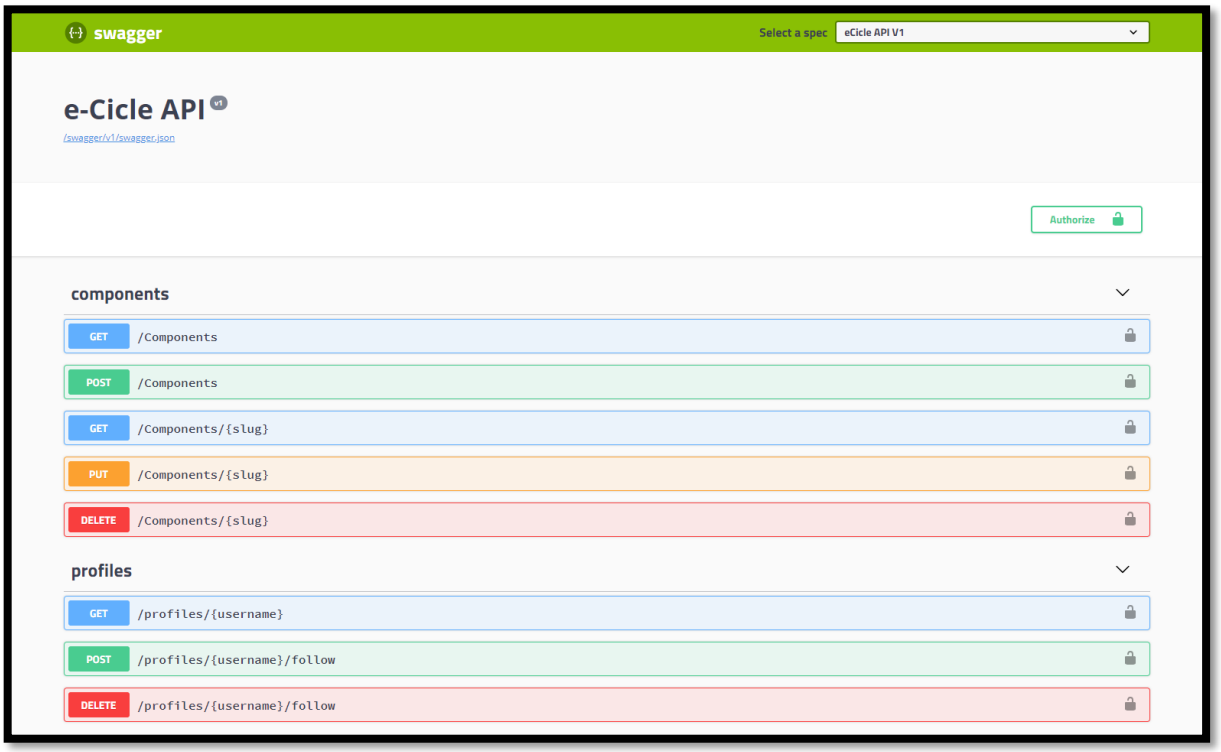

Figura 3 - Página inicial do Swagger

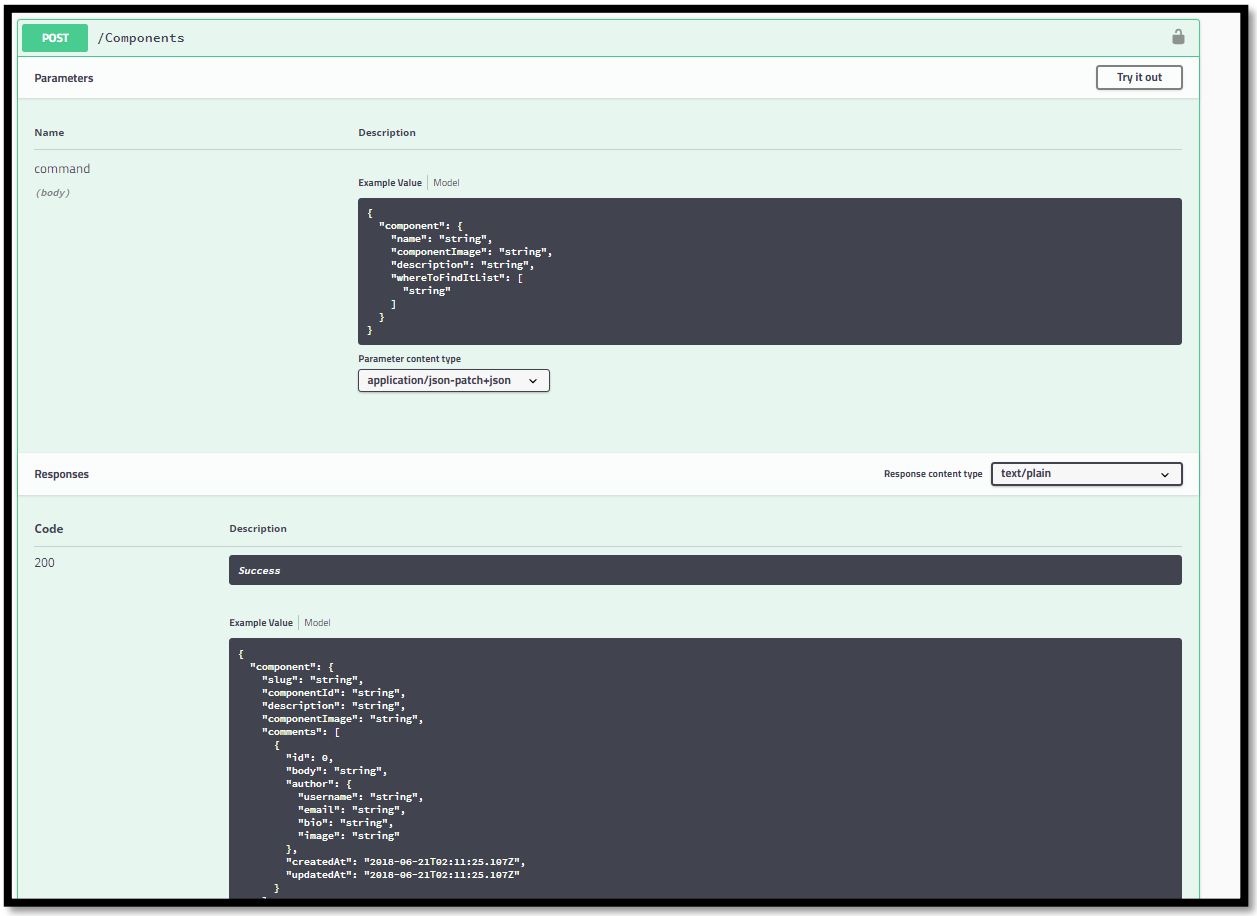

Figura 4 - Exemplos de requisição e resposta no Swagger

A camada de testes é detentora dos testes unitários e de integração, ela serve para nos dar segurança caso tenhamos que fazer uma atualização no software sem perder um comportamento da versão anterior, uma vez que caso os comportamentos se alterem os testes falharão.

O objetivo dessa arquitetura é manter o software o mais expansível possível de forma que não seria um problema por exemplo fazer as alterações necessárias para que a API possa ser consumida por outros sites e serviços disseminando as informações contidas na base de dados da aplicação.

No front-end utilizamos a biblioteca React, que facilitou o desenvolvimento separando todo o código em pequenas partes (em arquivos diferentes), que se comportam como componentes reutilizáveis. Além de usar o paradigma de programação declarativo que no caso do React tem como foco apenas o que é exibido na interface de usuário.

O paradigma de programação (estilo de construção da estrutura de elementos em programas de computadores) declarativo expressa a lógica computacional sem se importar em descrever os comandos necessários para execução de determinada tarefa.

Para gerenciar os estados globais dos componentes React foi utilizado o Redux, que ficou responsável pela troca de estados dos componentes por meio de ações recebidas pela aplicação. O Redux é uma biblioteca Javascript open-source para gerenciamento de estados de aplicações.

Open-source é o termo utilizado para indicar que o projeto tem o seu código aberto para modificações e leitura por qualquer pessoa interessada em contribuir.

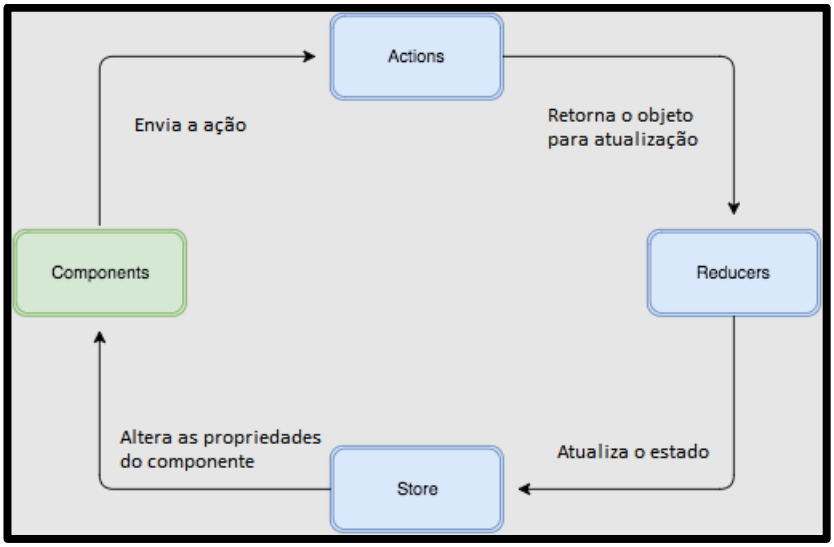

Figura 5 - Fluxo de mudança de estado de um componente

Na documentação do sistema foram utilizados: Diagrama de Casos de Uso, Diagrama de Classes, Diagrama de Sequência e seus respectivos dicionários e descrições.

Nos próximos tópicos estão os diagramas e todos os seus anexos.

3. DIAGRAMAS DO SISTEMA

3.1. Diagramas de Caso de Uso

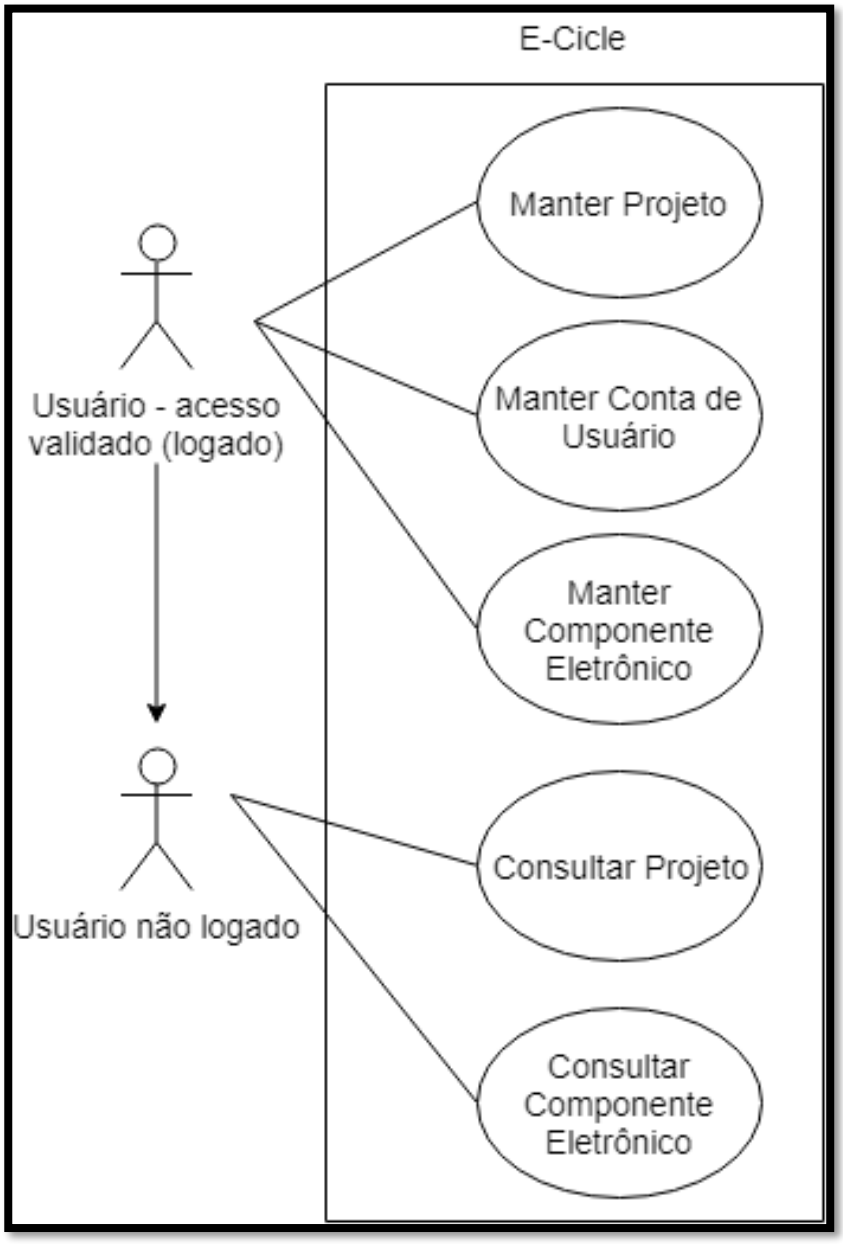

Figura 6 - Diagrama de Caso de Uso

3.1 Diagramas de Caso de Uso – Cenário

O E-Cicle – é utilizado abertamente por usuários previamente cadastrados. No sistema, o usuário pode realizar a inclusão de um novo componente, projeto ou qualquer outra informação, basta estar em uma sessão ativa e com seu login validado pelo sistema. O usuário também pesquisa componentes, projetos e outros usuários e os visualiza em modo de consulta apenas.

3.2 Diagrama de Classes

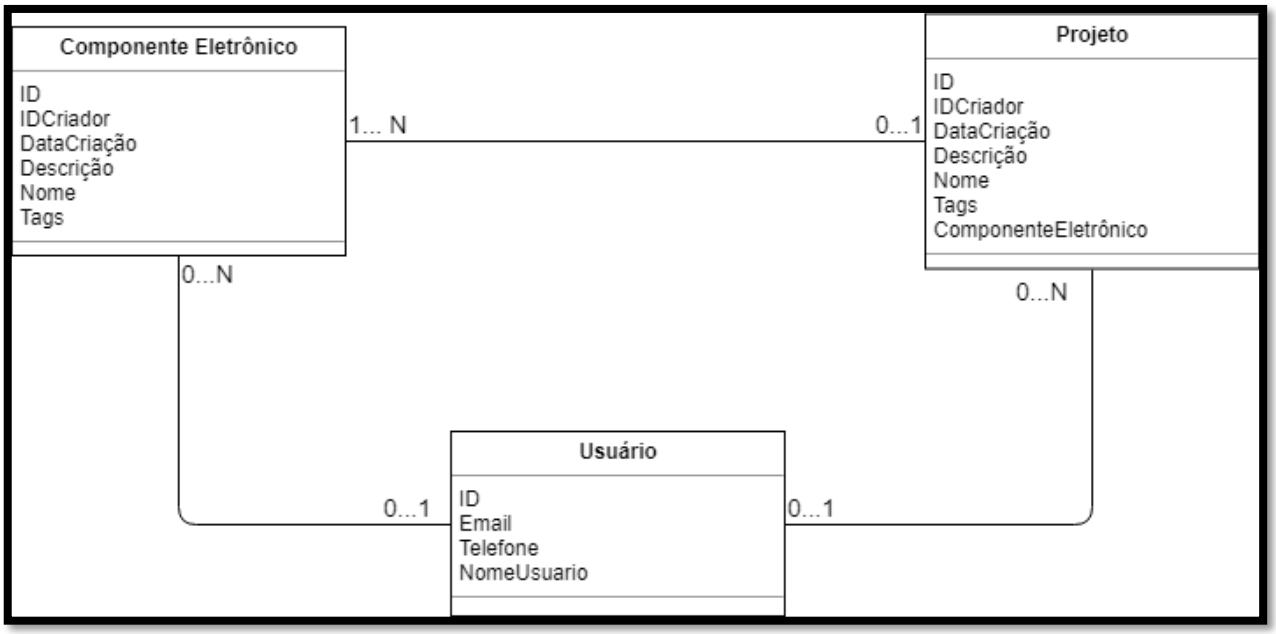

Figura 7 - Diagrama de Classes

3.2.1 Diagramas de Classes – Descrição

As classes do sistema se resumem a Usuário, que possui os atributos: ID, E-mail, Telefone e NomeUsuário. Um Usuário pode ter 0 ou mais Projetos ou Componentes Eletrônicos cadastrados por ele mesmo e consequentemente atrelados ao seu perfil. A classe Componentes Eletrônicos representa os objetos cadastrados com possibilidade de utilização em projetos. A classe Projeto representa os mesmos atributos das outras classes, como um ID, uma descrição, um nome, uma data de criação e o nome do criador.

3.3 Diagrama de Sequência

3.3.1 Cadastrar Usuário

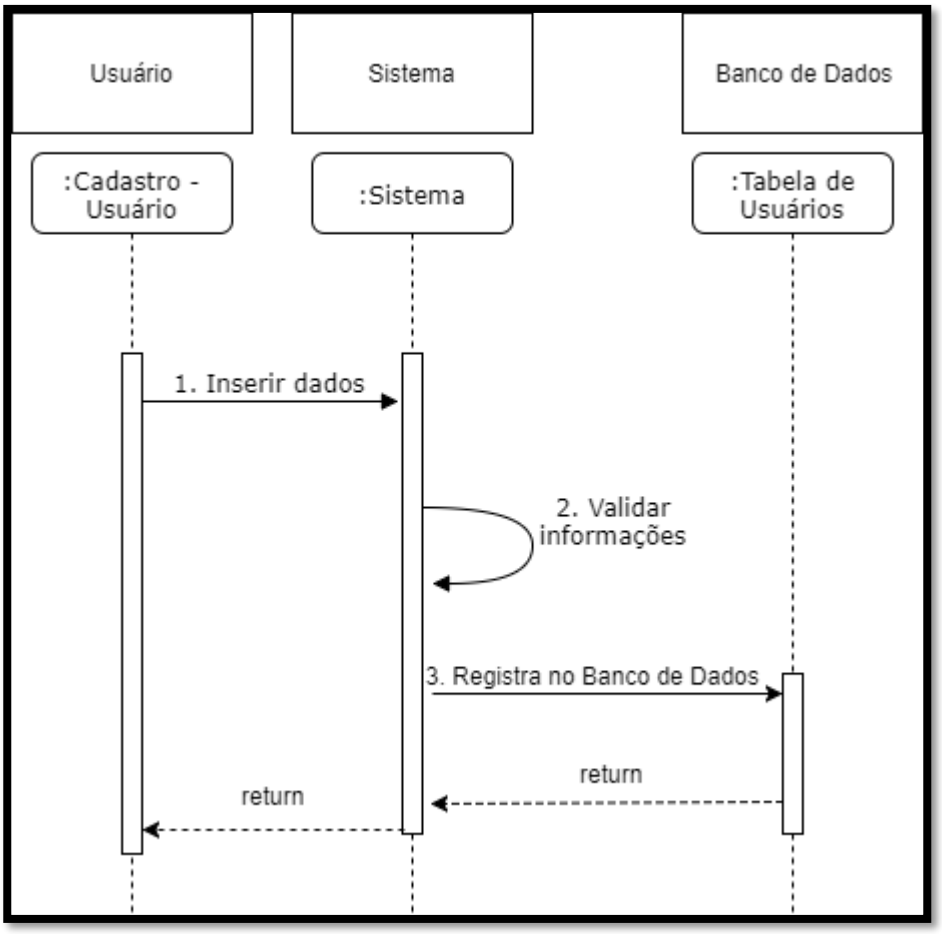

Figura 8 - Cadastrar Usuário

3.3.1.1 Fluxo – Cadastrar Usuário

1. Usuário insere os seus dados na tela de cadastro de usuário;
2. Sistema realiza a validação;
3. Sistema registra as informações de usuário logo;
4. Sistema retorna sucesso para o usuário.

3.3.2 Cadastrar objeto por tipo selecionado

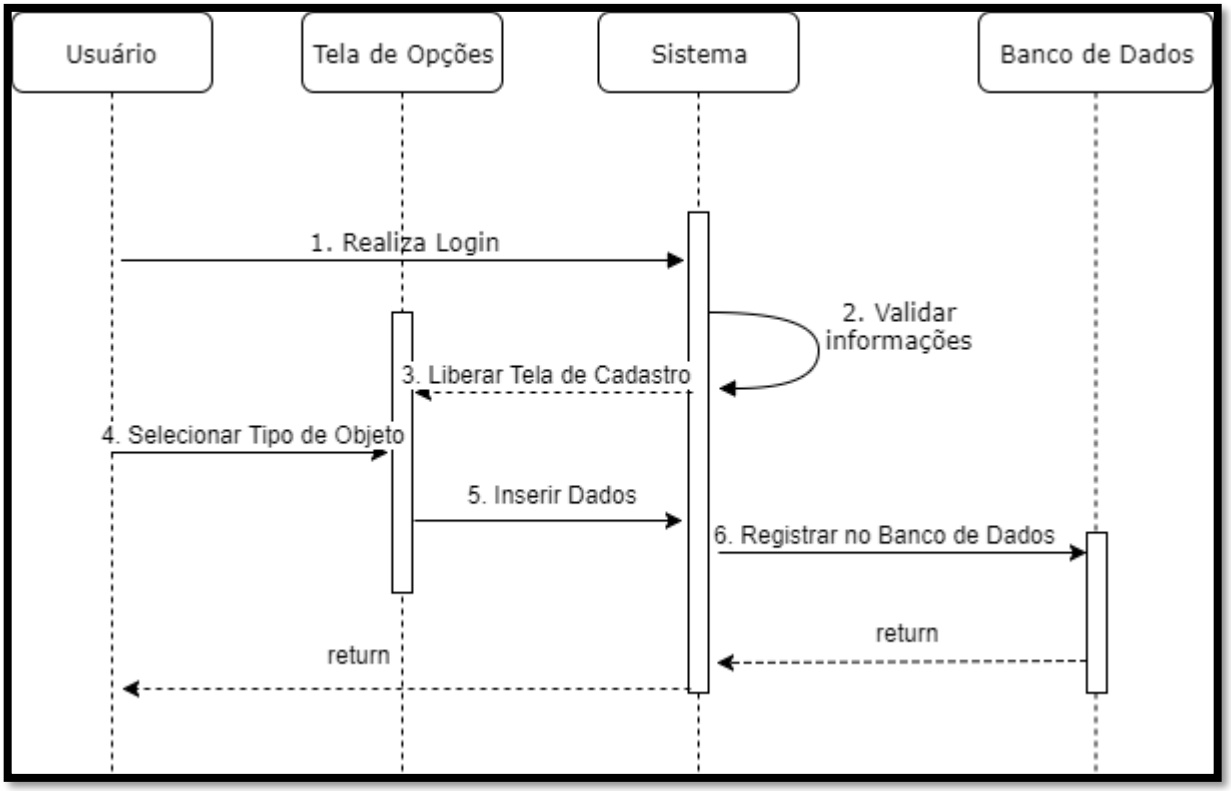

Figura 9 - Cadastrar objeto por tipo selecionado

3.3.2.1 Fluxo - Cadastrar objeto por tipo selecionado

1. Usuário realiza login no sistema;
2. Sistema valida informações do usuário;
3. Sistema libera tela de cadastramento;
4. Usuário seleciona o tipo de objeto a ser acessado;
5. Usuário insere dados através da tela de cadastro;
6. Sistema registra envio no banco de dados;
7. Sistema exibe mensagem de sucesso na tela de cadastro.

3.4. Diagrama Entidade – Relacionamento

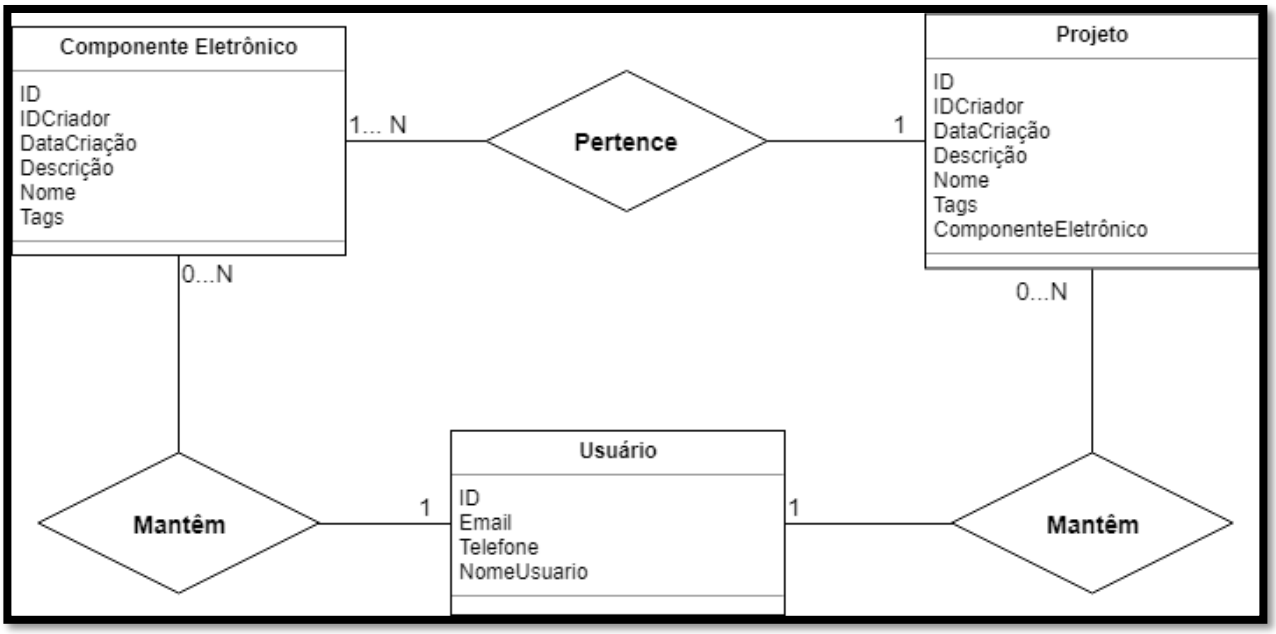

Figura 10 - Diagrama Entidade – Relacionamento

3.4.1 Diagrama Entidade – Relacionamento – Descrição

Uma entidade Componente Eletrônico pertence a um Projeto e a entidade Usuário mantêm zero ou mais Componentes Eletrônicos e Projetos.

3.5. Diagrama de Implantação

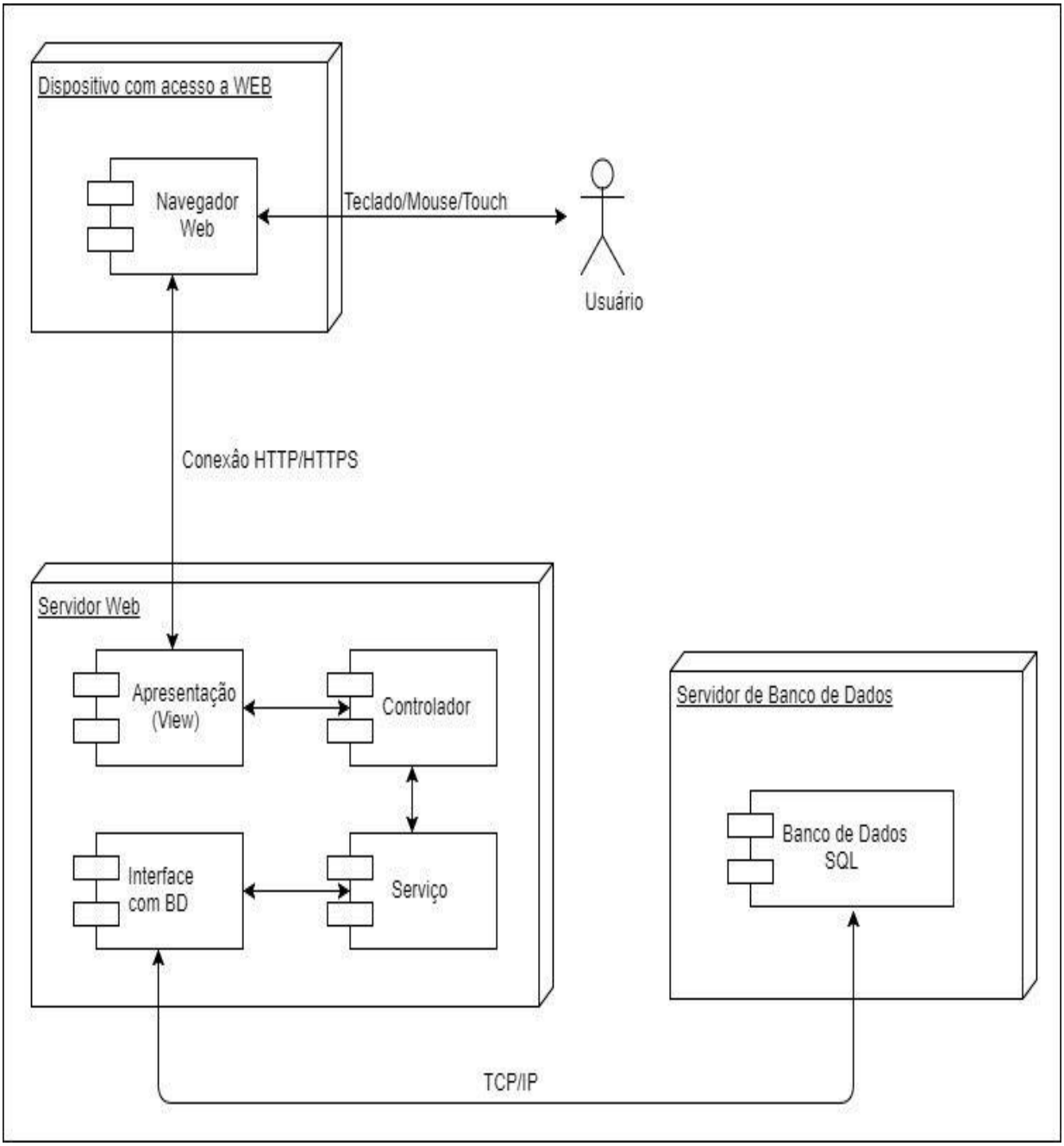

Figura 11 - Diagrama de Implantação

3.6. Diagrama de Pacotes

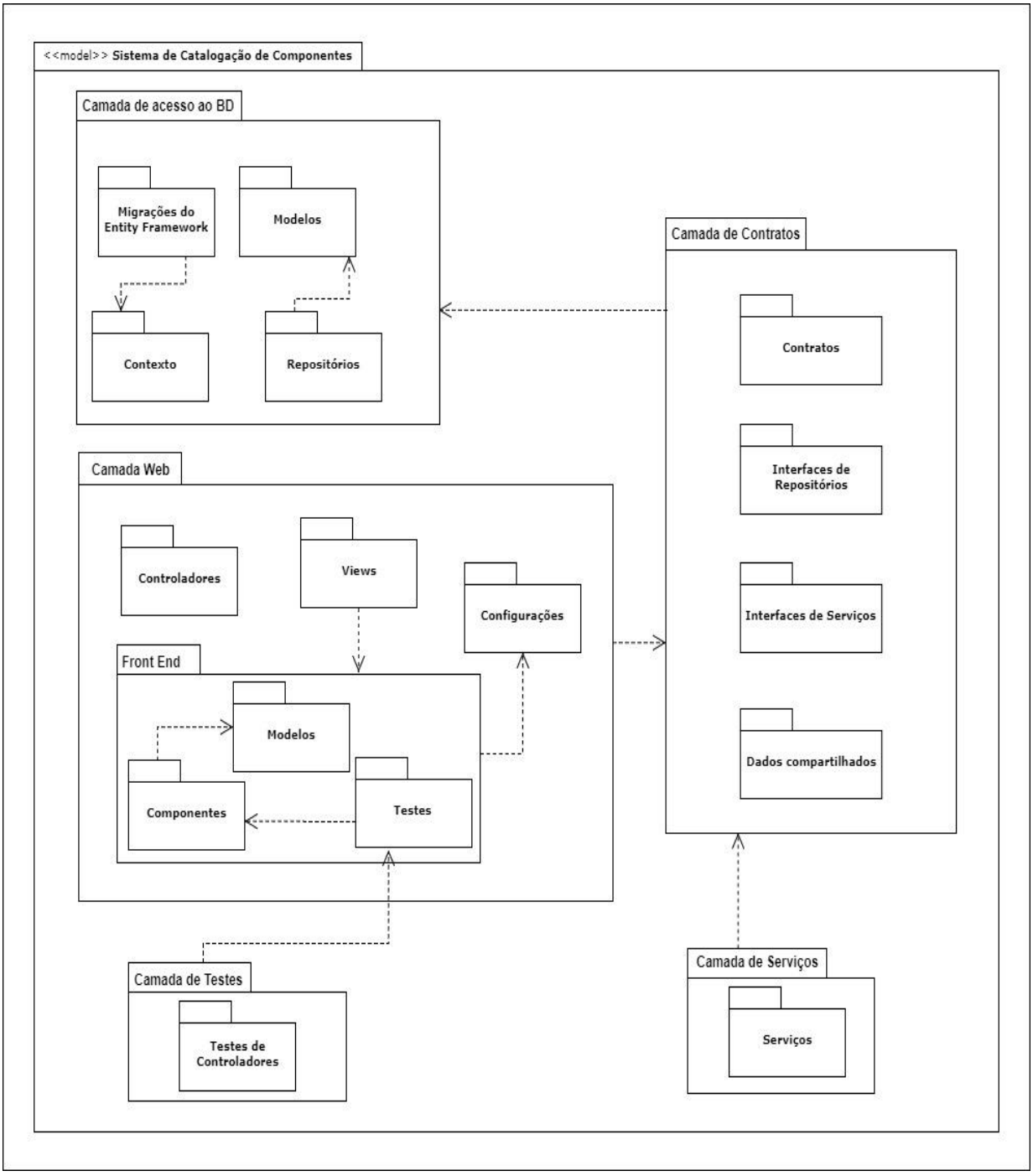

Figura 12 - Diagrama de Pacotes

3.7 Diagramas de Atividades

3.7.1 Criar usuário

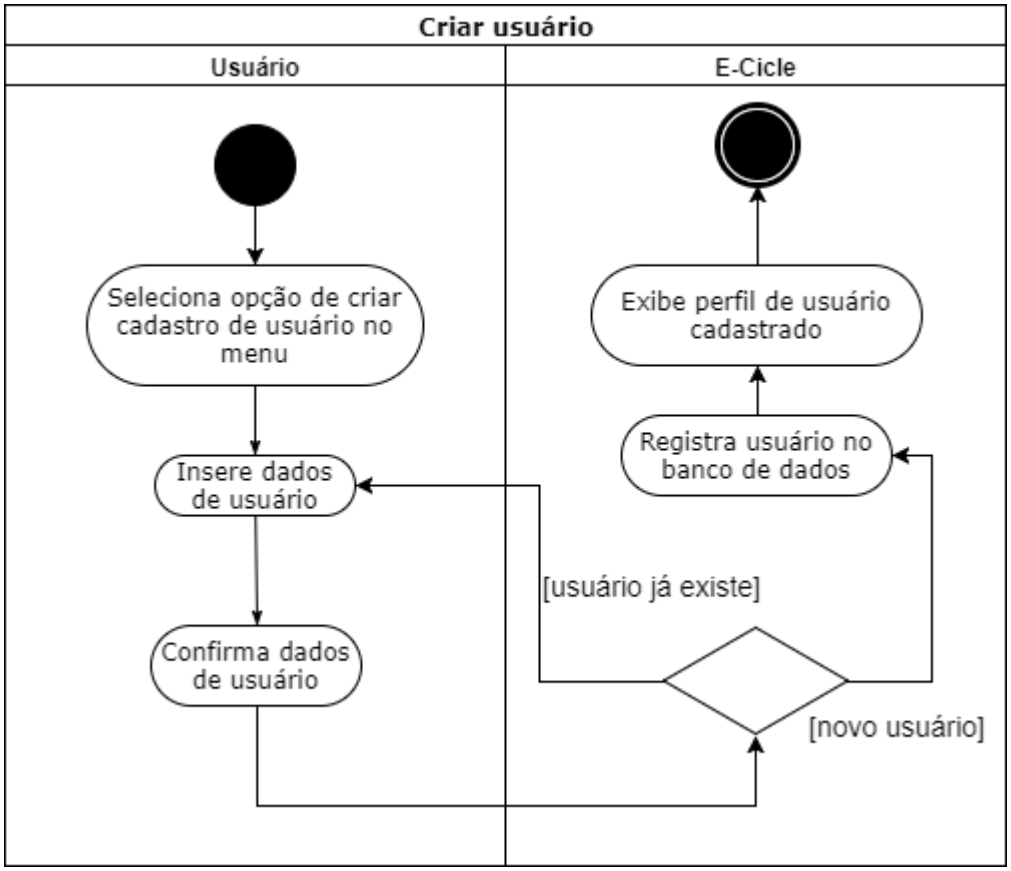

Figura 13 - Diagrama de atividades – Criar Usuário

3.7.2 Criar componente

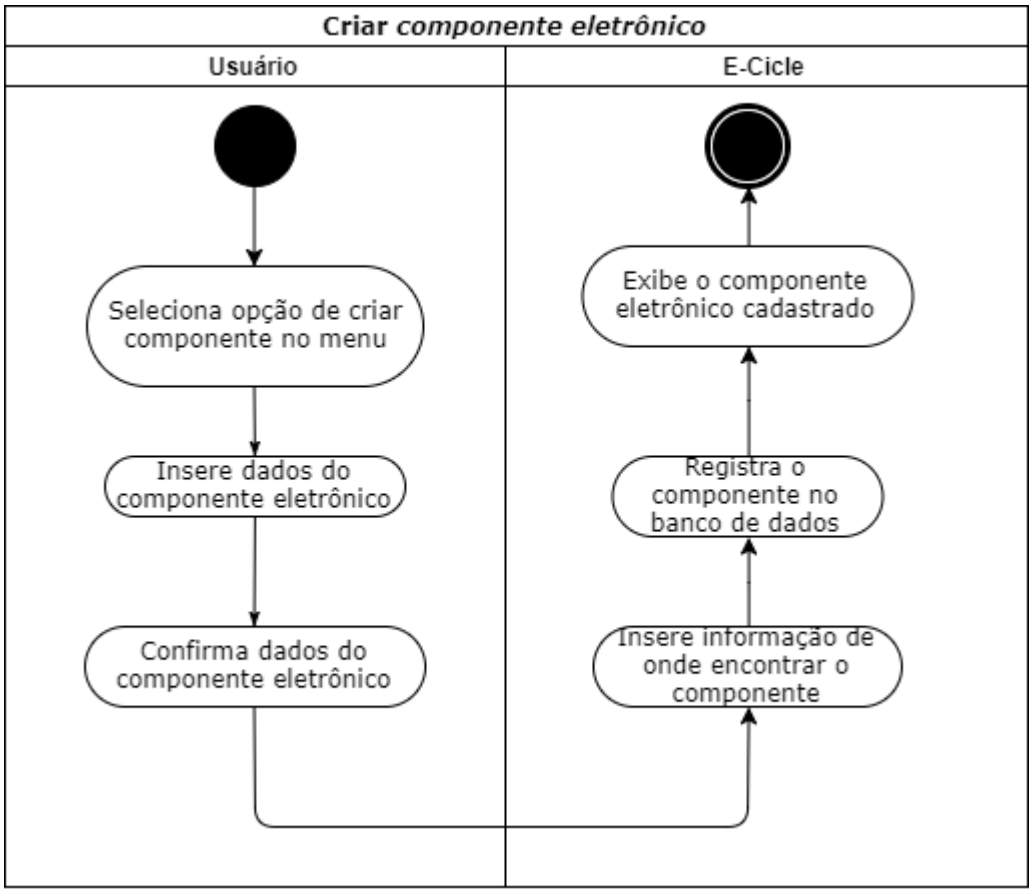

Figura 14 - Diagrama de atividades – Criar Componente eletrônico

3.7.3 Criar Projeto

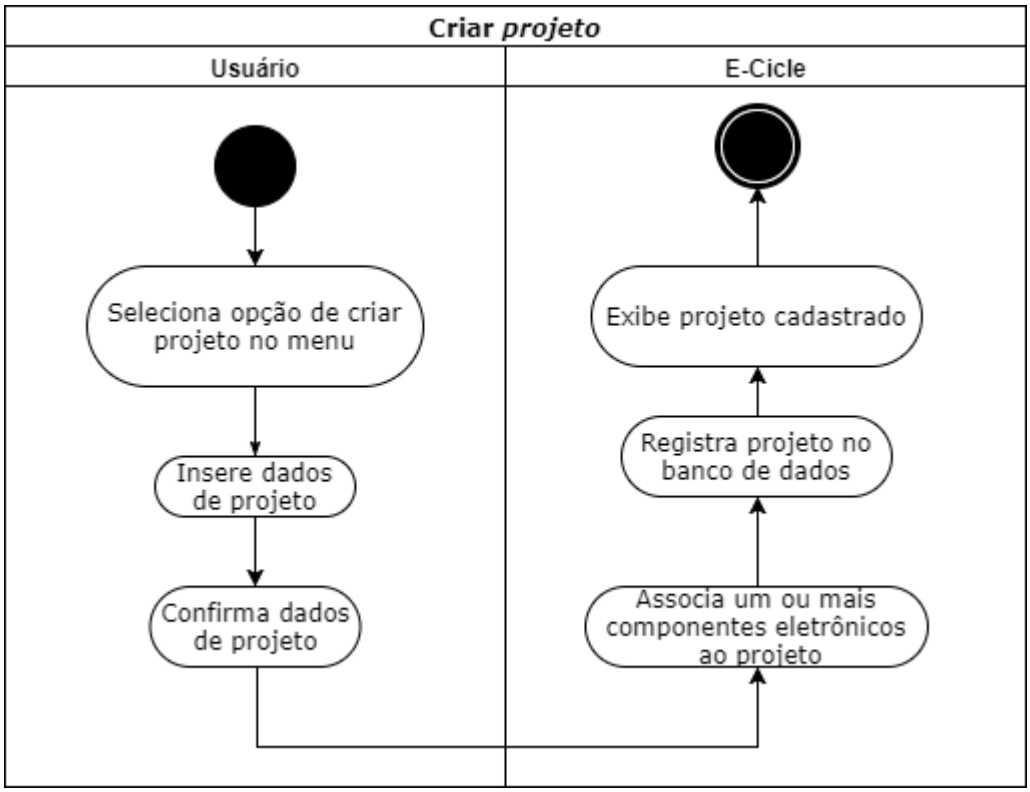

Figura 15 - Diagrama de atividades – Criar Projeto

3.7.4 Buscar Informação na Aplicação

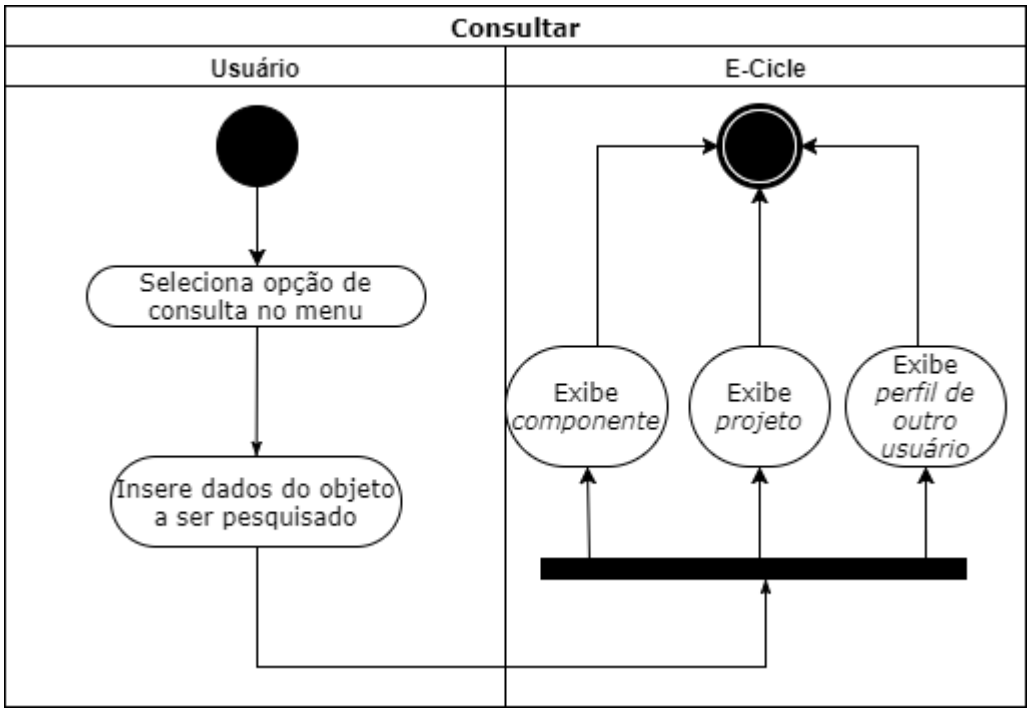

Figura 16 - Diagrama de atividades – Buscar

3.7.5 Editar Informação

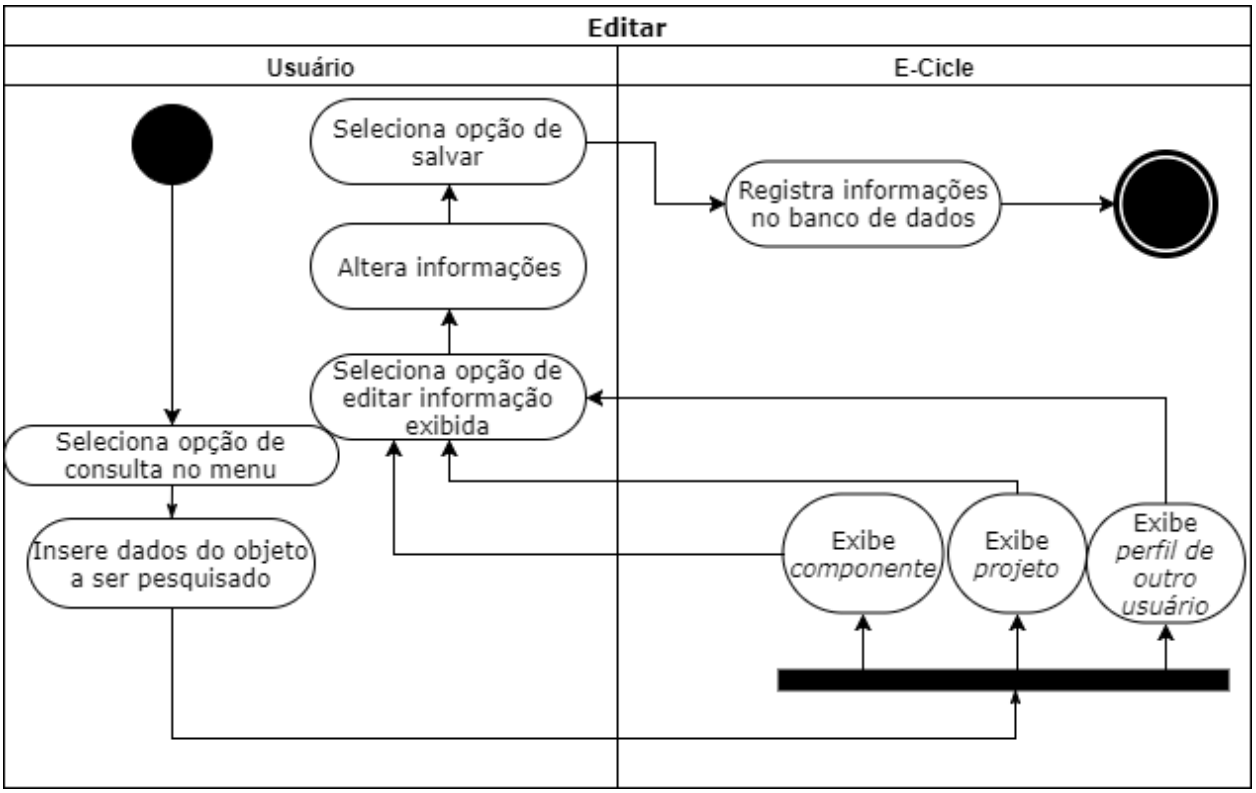

Figura 17 - Diagrama de atividades – Editar

3.7.6 Deletar

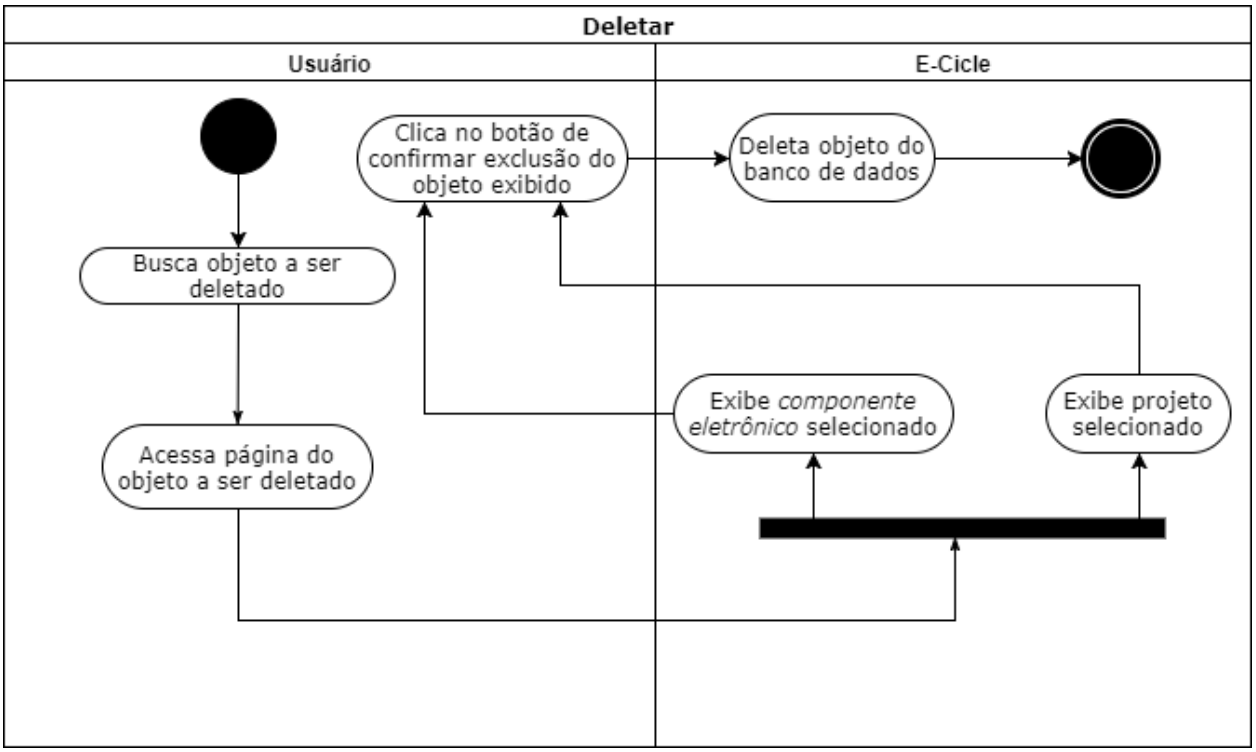

Figura 18 - Diagrama de atividades – Deletar

4. TESTES E IMPLEMENTAÇÃO

Com o objetivo de garantir a qualidade e correto funcionamento do sistema, antes de sua implementação - foram executados testes de integração entre os módulos de back-end e front-end e testes de interface, com a finalidade de garantir a melhor usabilidade, através de uma interface amigável e intuitiva para o usuário.

4.1 Teste unitário

O teste de unidade tem a finalidade de verificar o funcionamento da menor parte do sistema, um módulo ou fração, garantindo a qualidade do determinado componente, explica BARTIÉ (2002).

Assim como ressalta BOAS (2003), a execução dos testes unitários possibilita a correção dos eventuais problemas que podem acarretar em grades atrasos no desenvolvimento e testes futuros do sistema.

Os testes unitários foram executados em paralelo ao desenvolvimento, possibilitando a correção imediata de problemas nos módulos da aplicação.

4.2 Teste de Integração

O teste de integração foi executado à fim de montar e integrar os módulos (unidades do sistema), validando a sua compatibilidade para fornecer um pacote de software, da maneira como explica BARTIÉ (2002). UKUMA (2002) ressalta a importância destes conjuntos de testes, pois mesmo se cada módulo já foi testado isoladamente, pode apresentar algum problema quando houver a interação das unidades.

Os testes de integração permitiram um melhor desenvolvimento na conexão e funcionamento da interface entre back-end e front-end da aplicação.

4.3 Teste de Interface

O teste de interface foi executado com a finalidade de verificar se a navegabilidade dos objetos e telas do sistema, funcionariam de acordo com a proposta e de forma intuitiva.

Na execução do teste foram simuladas as condições de uso do programa sob a visão do usuário para verificar se o software é intuitivo e simples. Através de tarefas previamente definidas foi verificado: a facilidade de navegação na aplicação, clareza dos textos, acesso simplificado de mecanismos de apoio ao usuário, volume de ações e seu tempo de execução para realizar as tarefas.

Antes da execução das tarefas, com o objetivo de verificar se a apresentação do sistema e sua finalidade estava clara, os usuários responderam à pergunta: “Antes de começarmos o nosso teste, só de observar a aplicação, você consegue dizer o que ela é? Para que serve?”. Com isso foi possível melhorar a percepção do usuário quanto à finalidade do sistema.

As tarefas passadas para os usuários envolveram: a criação de um cadastro pessoal na aplicação, realização de login e logoff da aplicação, criação e edição de um componente, criação e edição de um projeto, interação com o conteúdo criado por outros usuários da aplicação e deleção dos objetos criados.

4.3.1 Interface de usuário antes da execução dos testes

As telas antes da realização do teste de interface eram:

Figura 19 - Tela principal da aplicação

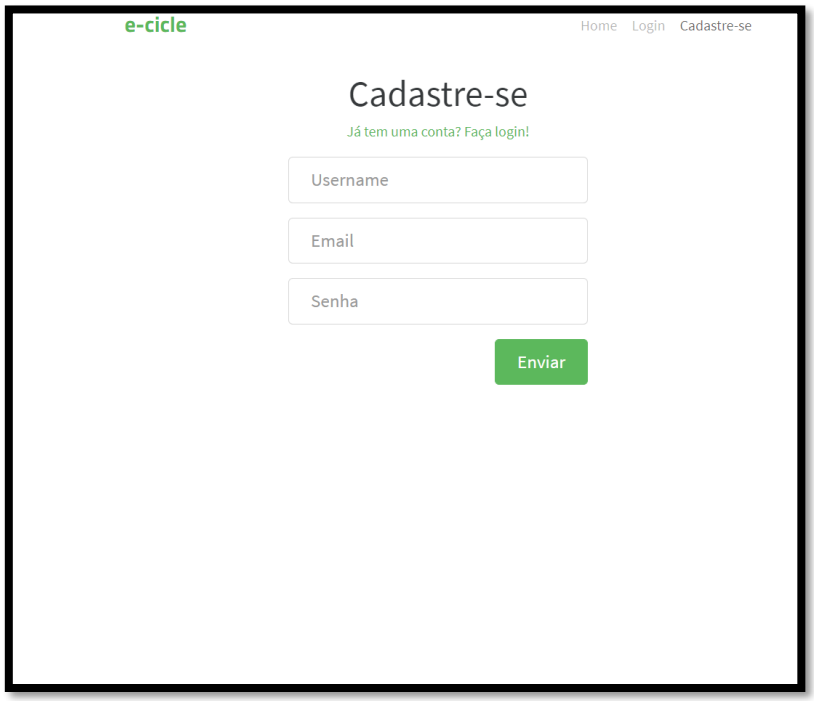

Figura 20 - Tela de Cadastro de Usuário

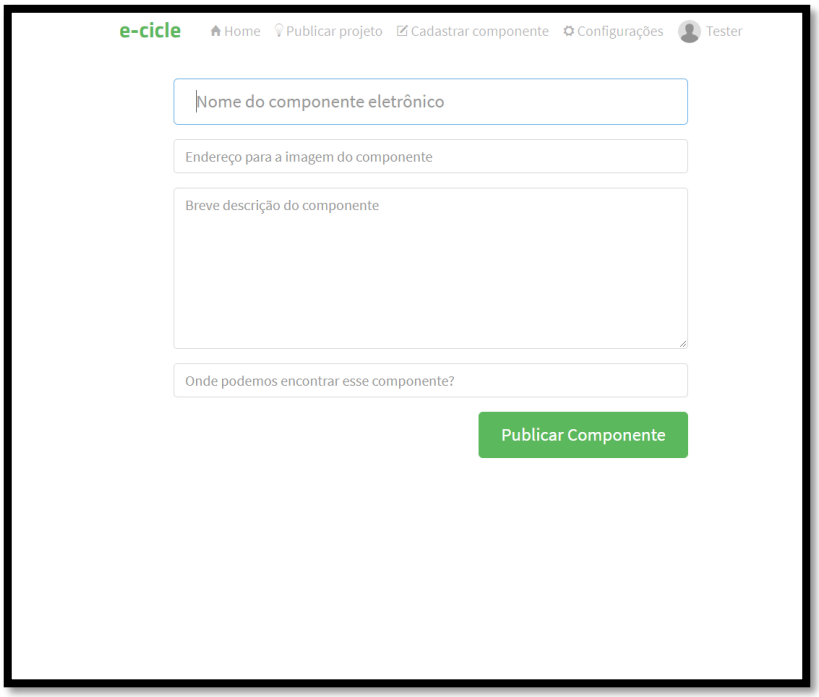

Figura 21 - Cadastro de Componente

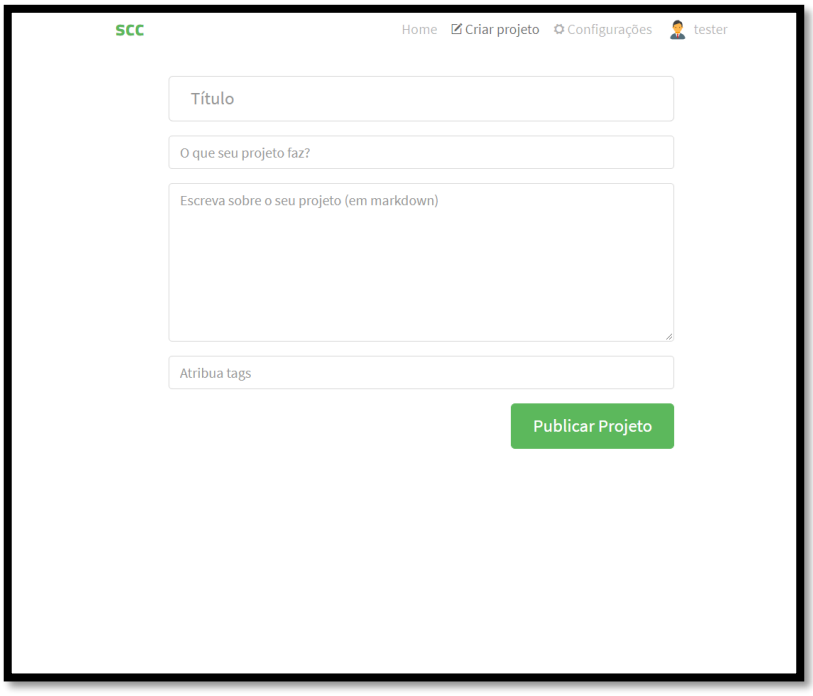

Figura 22 - Cadastro de Projeto

4.3.2 Interface de usuário após execução do teste de interface

Após a análise dos resultados dos testes de interface, foi identificada a necessidade de desenvolvimento das atualizações abaixo com a finalidade de melhorar a experiência final do usuário:

- Alterar a inclusão de palavras-chave ou tags, tanto em projetos quanto em componentes;
- Melhorar a diferenciação de um componente para um projeto na aplicação, através da inclusão de ícones para cada um;
- Melhorar a descrição do sistema na tela inicial, incluindo a baixo do banner em destaque uma descrição direta e objetiva do sistema;
- Melhorar a forma de logout da aplicação;
- Permitir a realização de login por username (antes era possível apenas utilizando o e-mail cadastrado);
- Inclusão de uma confirmação de senha no ato do cadastro;
- Destacar mais o botão de deletar um projeto ou componente;
- Alterar a tela inicial após o login para ser a de feed global e não de feed do usuário de forma que os usuários terão sempre a informação mais nova cadastrada na aplicação;
- Permitir acesso à tela de projeto e componente via link na imagem do mesmo.

4.3.2.1 Telas após melhorias de interface

Figura 23 - Nova tela principal

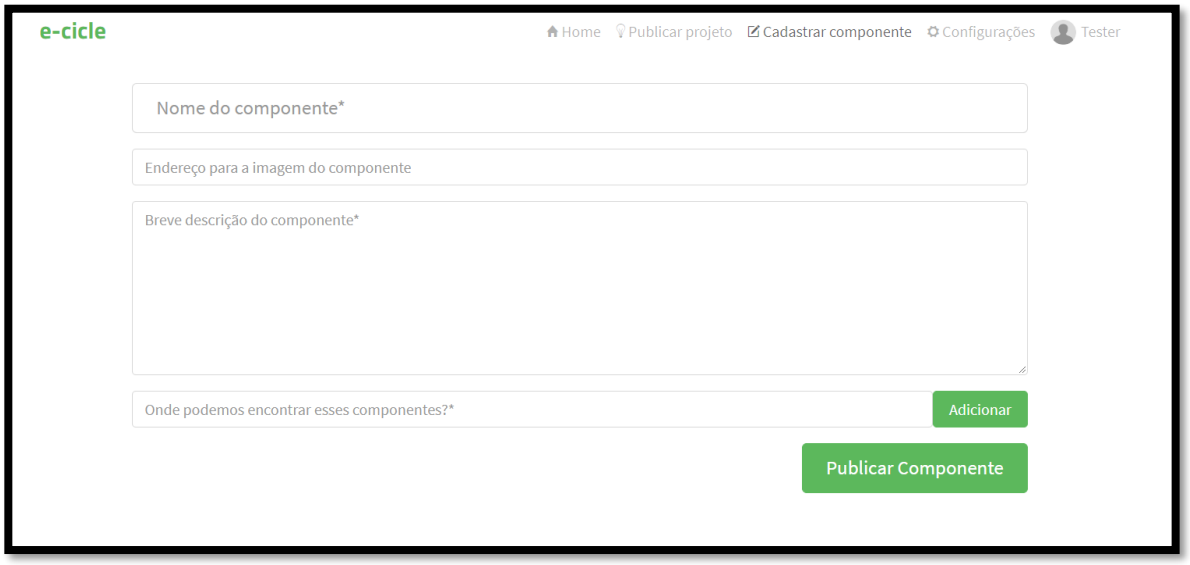

Figura 24 - Nova tela de cadastro de componente

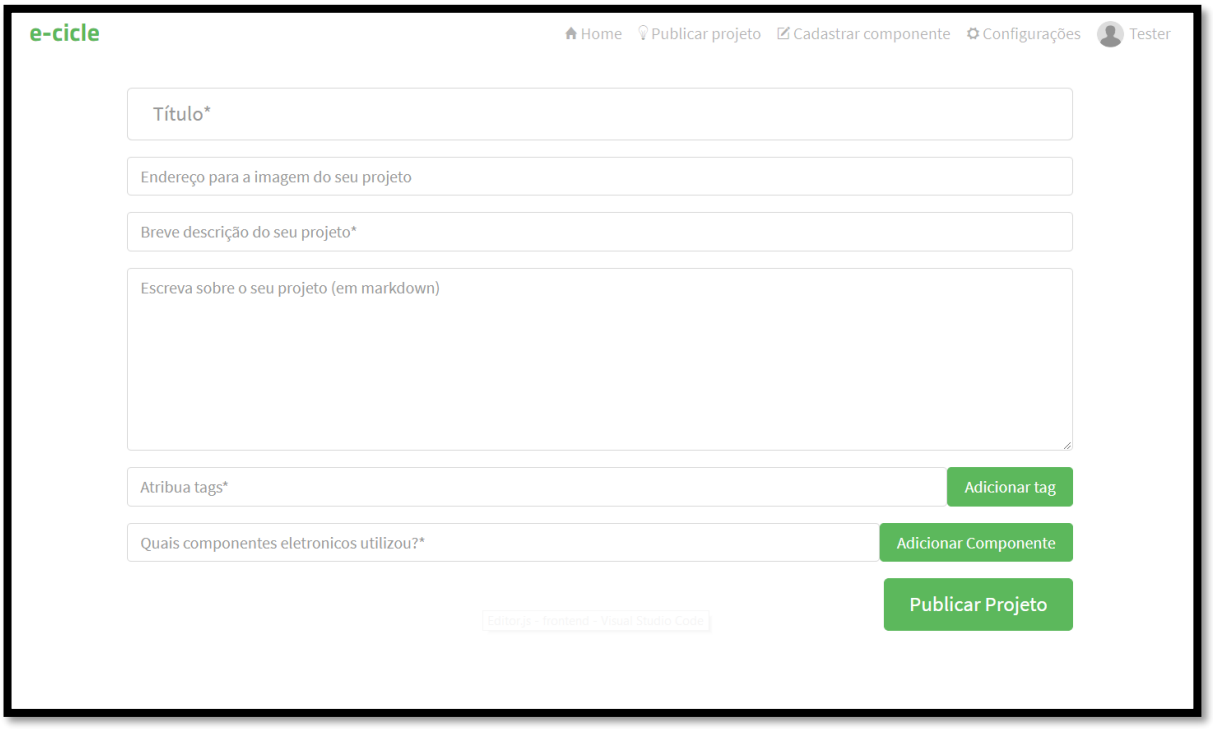

Figura 25 - Nova tela de cadastro de componente

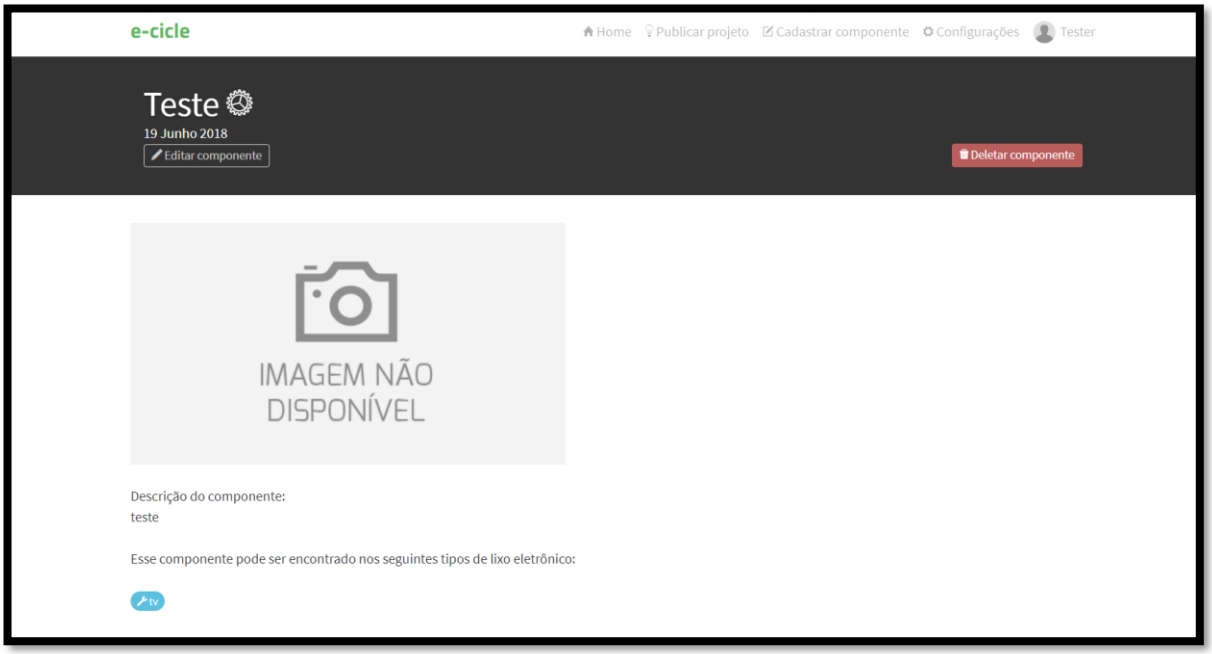

Figura 26 - Nova tela de componente com o novo ícone e novo botão de deleção

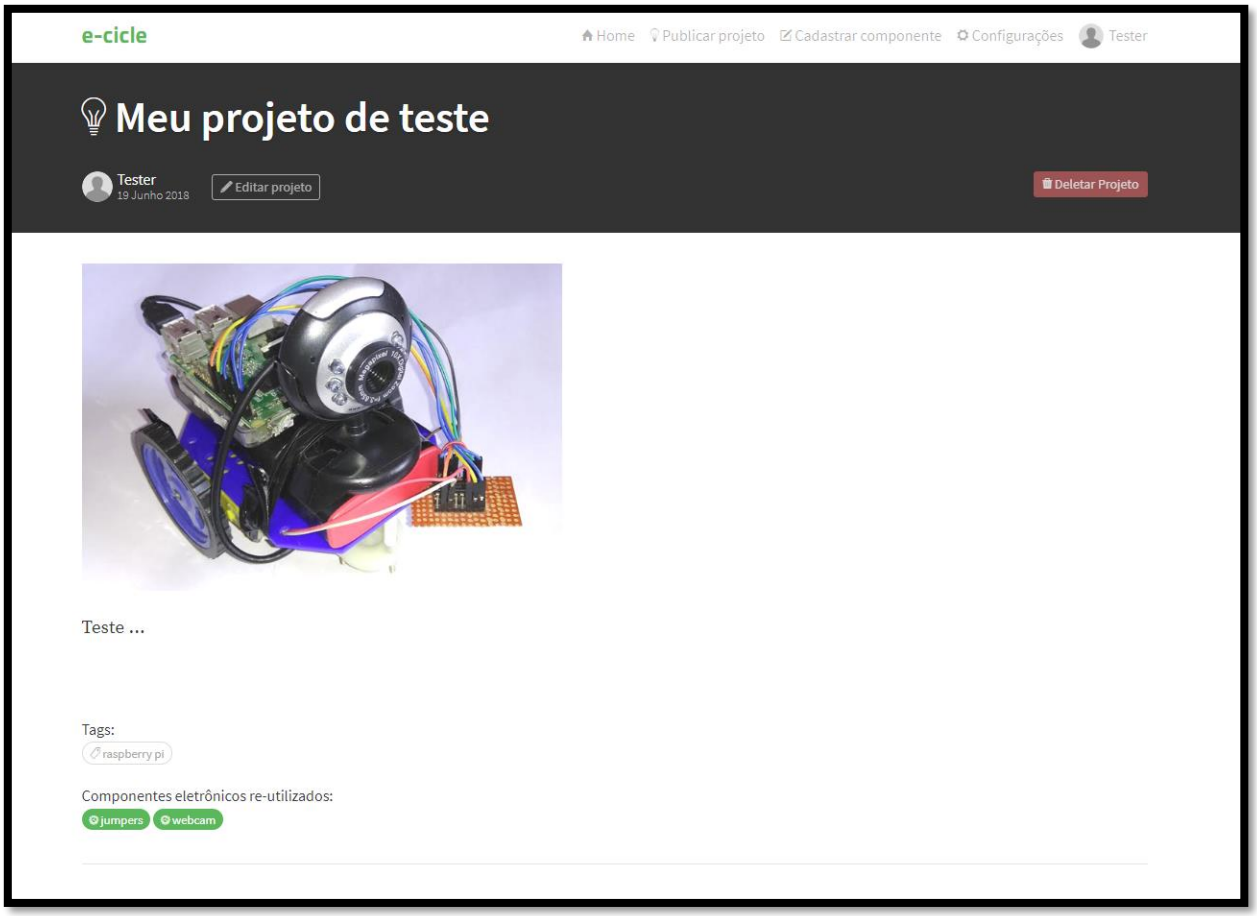

Figura 27 - Nova tela de projeto com o novo ícone e novo botão de deleção

5. DEFINIÇÃO DA INTERFACE – CORES E PADRÕES

Através do estudo de interface foram definidos os diversos componentes da interface, as cores e os padrões utilizados, incluindo: os símbolos da aplicação, cores e aspectos textuais.

Na escolha as cores, foi priorizada a aproximação com o tema da a aplicação, - sustentabilidade pelo reuso de lixo eletrônico -, com sua representação no mundo real, as cores verde e azul foram as cotadas para utilização na interface, porém o verde prevaleceu por ser comumente associado à natureza.

Os símbolos utilizados na aplicação, também foram utilizadas no contexto da realidade, visando também possibilitar que os usuários possam intuitivamente compreender as diferentes representações e conceitos presentes na aplicação.

Em “Projetos” por exemplo, é utilizado o ícone de uma lâmpada, remetendo à uma “ideia nova e criativa” para reuso de lixo eletrônico, para os componentes são utilizadas engrenagens, trazendo o conceito de “peças” ou “partes” avulsas a serem utilizadas nos projetos e por último, a logo da aplicação, que foi desenhada para remeter às ideias de eletrônicos + lixo + reciclagem.

O uso de fontes sem serifa foi escolhido com a finalidade de transmitir a ideia de atualidade, remetendo à tecnologia, além de facilitar a leitura na web. As fontes foram combinadas em padrão harmônico concordante, ou seja, apenas uma fonte, havendo diferenciações de peso (negrito) quando necessário estabelecer contraste.

6. CONSIDERAÇÕES FINAIS

Apenas 2% do lixo eletrônico é reciclado no país, o que é não somente um problema ecológico mas também um problema econômico. Quando passamos a investir em reciclagem destes componentes, transformamos problemas em oportunidades, amenizando o impacto no meio ambiente e criando alternativas para a geração de renda

O desenvolvimento do presente estudo possibilitou uma análise de como um software de compartilhamento de informações sobre projetos utilizando lixo eletrônico pode ajudar e incentivar a diminuição da quantidade de material eletrônico descartado através de seu reuso.

Além disso, também permitiu um teste de interface para obter uma boa experiência de usuário de forma acessível aos diferentes graus de conhecimento em informática do público alvo. No estudo de interface, foi executado um teste em ambiente controlado onde os participantes receberam algumas tarefas e tiveram o tempo máximo de quarenta segundos para conclui-las, ao não concluir uma tarefa no tempo estipulado a próxima tarefa era apresentada, assim sucessivamente até que todas as tarefas estabelecidas no roteiro do teste fossem anunciadas ao participante.

Durante o teste os participantes tiveram suas interações com a aplicação, voz e rosto gravados, para que fosse possível detectar todos os pontos de dúvida ou dificuldade na utilização da aplicação. Além disso seus resultados foram salvos para análise estatística e tomada de decisão.

Após a realização do teste de interface, verificou-se que a parte mais complexa do processo era o registro de componentes, devido a mecânica de inserção de "tags" que não é tão difundida em usuários menos ligados diretamente a área de tecnologia da informação.

Ao analisar os resultados foram ainda encontradas outras dificuldades menores dos participantes do teste.

Com essas informações a interface da aplicação foi alterada a fim de facilitar a interação do usuário. Permitindo assim, que os objetivos propostos fossem realmente alcançados.

Dada a importância do assunto, torna-se necessário o desenvolvimento de formas de facilitar o compartilhamento de conhecimento relacionado a reutilização de componentes eletrônicos e tornar fácil de ser feito digitalmente para pessoas, independentemente do nível de conhecimento em informática.

Nesse sentido, a utilização de recursos digitais permite às escolas e à sociedade em geral reutilizar material eletrônico. Diminuindo o impacto ambiental e tornando o estudo utilizando componentes eletrônicos mais acessível.

REFERÊNCIAS

BARTIÉ, Alexandre. **Garantia da qualidade de software**. Rio de Janeiro: Editora Campus, 2002.

BOAS, André Luiz de Castro Villas. **Gestão de configuração para teste de software**. Dissertação de mestrado. Universidade Estadual de Campinas, 2003.

CERATTI, Mariana. **Lixo eletrônico: um mercado com potencial milionário**. Disponível em: <https://brasil.elpais.com/brasil/2017/02/18/politica/1487418470_101918.html>. Acesso em: 24 Ago. 2017.

CROCKFORD, Douglas. **JavaScript The Good Parts.** 1. ed. Sebastopol: O'Reilly Media, 2008.

Departamento de Sistemas de Informação UDESC. **Quanto que o Brasil produz de lixo eletrônico?** Disponível em: <http://nti.ceavi.udesc.br/e-lixo/index.php?makepage=quanto_o_brasil_produz>.

Developer Mozilla. **A re-introduction to JavaScript (JS tutorial)**. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/JavaScript/A_re-introduction_to_JavaScript/>. Acesso em: 17 Out. 2017.

Developer Mozilla. **What is JavaScript?** Disponível em: https://developer.mozilla.org/en-US/docs/Web/JavaScript/About_JavaScript/>. Acesso em: 15 Out. 2017.

FOWLER, Martin. **Microservices.** Disponível em: < <https://martinfowler.com/articles/microservices.html>. Acesso em: 19 Jun. 2018.

FGV. **Pesquisa Anual do Uso de TI.** Disponível em: <http://eaesp.fgvsp.br/ensinoeconhecimento/centros/cia/pesquisa/>. Acesso em: 24 Ago. 2017.

GIUSTO, D. *et al.* (Ed.). **The Internet of Things: 20th Tyrrhenian Workshop on Digital Communications***.* 1. ed. Nova York: Spring-Verlag, 2010.

HELLER, Eva. **A Psicologia das Cores: Como as Cores Afetam a Emoção e a Razão.** 1ª Ed. São Paulo: GG, 2012.

HONORINE, Solenn. **E-Waste Creates Economic, Environmental Problem for Developing Nations.** Disponível em: <http://eaesp.fgvsp.br/ensinoeconhecimento/centros/cia/pesquisa/>. Acesso em: 25 Jun. 2018.

JUNIOR, Edgard. **UIT diz que número de celulares no mundo passou dos 7 bilhões em 2015**. Disponível em: <http://www.ebc.com.br/tecnologia/2015/05/uit-diz-que-numero-de-celulares-no-mundo-passou-dos-7-bilhoes-em-2015>. Acesso em: 25/04/2018.

LEE, Gyu. **The Internet of Things - Concept and Problem Statement***.* Disponível em: <https://tools.ietf.org/html/draft-lee-iot-problem-statement-00>. Acesso em: 30 Ago. 2017.

NIELSEN, Jakob. **Usability Engineering**. 1 ed. Boston: Academic Press, 1993. 1 vol.

PACIEVICTH, Thais**. Evasão escolar***.* Disponível em: <http://www.infoescola.com/educacao/evasao-escolar/>. Acesso em: 24 Ago. 2017.

PORTAL EXAME. **Número de smartphones supera o de computadores no Brasil.** Disponível em: <http://exame.abril.com.br/tecnologia/numero-de-smartphones-supera-o-de-computadores-no-brasil/>. Acesso em: 26 Ago. 2017.

PORTAL G1. **Brasil produz 36% do lixo eletrônico da América Latina, mostra estudo**. Disponível em: <http://g1.globo.com/tecnologia/noticia/2015/12/brasil-produz-36-do-lixo-eletronico-da-america-latina-mostra-estudo.html>. Acesso em: 25 Ago. 2017.

SEBESTA, Robert W. **Programming the world wide web.** 7. ed. Boston: Pearson, 2012.

UKUMA, Luciano Hayato. **Uma estratégia para desenvolvimento de componentes de software autotestáveis**. Dissertação de mestrado, Universidade Estatual de Campinas, 2002.

VERMESAN, O. FRIESS, P. **Internet of Things - Global Technological and Societal Trends from Smart Environments and Spaces to Green Ict**. 1. ed. Aalborg: River Publishers, 2011.

WILLIAMS, Robin. **Design Para Quem Não É Designer: Princípios de Design e Tipografia para iniciantes**. 5ª Ed. São Paulo: Callis, 2013.