INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO RIO GRANDE DO NORTE

KAYNA DAIZYANNE DA SILVA

MARIA LUIZA BARBOSA BEZERRA

UM SISTEMA DE APOIO À LIXEIRAS INTELIGENTES

Lajes/RN
2018

KAYNA DAIZYANNE DA SILVA
MARIA LUIZA BARBOSA BEZERRA

UM SISTEMA DE APOIO À LIXEIRAS INTELIGENTES

Relatório de Prática Profissional apresentado ao Curso Técnico de Nível Médio em Informática, modalidade subsequente, do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, campus Avançado de Lajes, em cumprimento às exigências legais como requisito parcial à obtenção do título de Técnico em Informática.

Orientador: Robercy Alves da Silva
Co-Orientador: Dannilo Martins Cunha

Lajes/RN
2018

KAYNA DAIZYANNE DA SILVA
MARIA LUIZA BARBOSA BEZERRA

UM SISTEMA DE APOIO À LIXEIRAS INTELIGENTES

Relatório de Prática Profissional apresentado ao Curso Técnico de Nível Médio em Informática, na modalidade subsequente, do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte, campus Avançado de Lajes, em cumprimento às exigências legais como requisito parcial à obtenção do título de Técnico em Informática.

Aprovado em __ de _____ de 2018.

BANCA EXAMINADORA

_____
Robercy Alves da Silva
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

_____
Dannilo Martins Cunha
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

_____
João Batista de Medeiros
Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte

AGRADECIMENTOS

Aos familiares pelo amor, apoio, paciência e pela valorização dos estudos como essencial ao crescimento.

A todos aqueles que acreditaram na realização deste trabalho e deram-nos forças e estímulo para dar prosseguimento a esta pesquisa e obter sucesso. Em especial, a docente, Profª. Ma. Pedrina Célia Brasil, por toda dedicação e paciência conosco.

Ao nosso orientador e co-orientador respectivamente, Prof. Robercy Alves da Silva, Prof. Me. Dannilo Martins Cunha e aos nossos colegas de turma.

A Deus criador dos céus e da terra, que nos deu vida e forças para alcançarmos nossos objetivos.

Se a educação sozinha não transforma a sociedade, sem ela tampouco a sociedade muda.

Freire (2000, p. 67).

RESUMO

Nos dias atuais a tecnologia vem sendo utilizada em grande parte da sociedade como um recurso positivo para mudança de comportamento. É através da interação entre tecnologia e tarefas que são realizadas diariamente, que surge o termo gamificação. Pensando em aproveitar essa integração, gamificação e educação ambiental, despertou-se a ideia do desenvolvimento de um *software* (sistema *web*) que possa ser adaptado a uma lixeira convencional transformando-a em lixeira com características especiais com intuito de incentivar e conscientizar a comunidade sobre a importância do descarte correto de lixo. Este trabalho tem como objetivo mostrar o gerenciamento dos dados obtidos nas lixeiras. No qual visa o gerenciamento de uma lixeira divertida, permitindo a visualização do seu uso por dia e hora, além do desempenho do nível da bateria e quantidade de lixo que a lixeira apresenta.

Palavras-Chave: Gamificação. Lixeira. Sistema. Gerenciamento.

ABSTRACT

Today, technology has been used in large part of society as a positive resource for behavior change. It is through the interaction between technology and tasks that are performed daily that the term gamification appears. Thinking about taking advantage of this integration, gamification and environmental education, the idea of the development of software (web system) that can be adapted to a conventional trash can be transformed into a garbage bin in order to encourage and make the community aware of the importance of the correct garbage disposal. The objective of this work is to show the management of the data obtained in the garbage dumps. In which it aims to manage a fun trash, allowing the visualization of its use by day and time, in addition to the performance of the battery level and amount of trash that the bin presents.

Key-words: Gamification. Trash can. System. Business Management.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE SIGLAS

ABRELPE	Associação Brasileira das Empresas de Limpeza Pública e Resíduos Especiais
ANSI	*American National Standard Institute*
API	*Application Programming Interface*
CRUD	*Create*, *Read*, *Update, Delete*
CSS	*Cascading Style Sheets*
GUI	*Graphical User Interface*
HTML	*Hypertext Markup Language*
IDE	*Integrated Development Environment*
IFRN	Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Norte
JDBC	*Java Database Connectivity*
JPA	*Java Persistence API*
JPQL	*Java Persistence Query Language*
MD5	*Message-Digest algorithm 5*
MOR	Mapeamento Objeto Relacional
MVC	*Model, View, Controller*
PDF	*Portable Network Graphics*
Play	*Play Framework*
PHP	*Personal Home Page*
PNG	*Portable Document Format*
POJOS	*Plain Old Java Objects*
RF	Requisitos Funcionais
RN	Rio Grande do Norte
RNF	Requisitos Não Funcionais
SGBD	Sistema de Gerenciamento de Banco de Dados
SQL	*Structured Query Language*
UML	*Unified Modeling Language*
XML	*eXtensible Markup Language*

SUMÁRIO

1. INTRODUÇÃO

Nos dias atuais a tecnologia vem sendo utilizada em grande parte da sociedade como um recurso positivo para mudança de comportamento.

De acordo com a pesquisa realizada pela ABRELPE/IBGE em relação ao Rio Grande do Norte (RN), no ano de 2015 foram coletados cerca de 0,783 (kg/hab/dia) de lixo gerando um total de 2.695 (t/dia). Neste mesmo ano a população era estimada em 3.442.175 (três milhões quatrocentos e quarenta e dois mil cento e setenta e cinco). Através dessa pesquisa nota-se que um dos grandes problemas da humanidade é o lixo produzido em aglomerações urbanas (ABRELPE, 2015).

Atualmente, a população tem a ideologia de que tudo pode ser descartável e isso faz com que a quantidade de lixo produzida aumente e estabeleça sérios problemas.

Numa cidade como Lajes/RN, onde a política pública de gerenciamento de resíduos sólidos é deficitária os transtornos causados pela poluição se fazem presentes no dia a dia da população. É notável que a poluição do município de Lajes não apresenta um nível elevado, comparados das grandes cidades. Mas exibe uma desorganização em relação ao momento de coletarem o lixo, pois não utilizam coleta seletiva, tendo como meio de controle de poluição as lixeiras públicas e carros de destinação inadequados sendo todos coletados sem separação.

Nesse contexto, para evitar problemas como aglomerações de lixos espalhados pelas ruas pode-se utilizar a tecnologia como forma de persuadir as pessoas, já que é perceptível grandes transformações em várias áreas da sociedade.

É através da interação entre tecnologia e tarefas que são realizadas diariamente, que surge o termo gamificação. Que nada mais é do que engajar as pessoas em uma iniciativa benéfica para elas mesmas (CRUZ JUNIOR, 2014).

Segundo Cruz Junior (2014), a gamificação está presente em realizar as tarefas que possam ser entediantes ou até mesmo despercebidas, sejam mais divertidas e proporcionem experiências mais prazerosas, aplicando técnicas de jogos ou intuitivas em ambientes que não sejam de jogos.

Sabendo-se disso é possível enriquecer o modelo tradicional de lixeiras através do uso da gamificação. Exemplos de lixeiras com esse propósito são:

- A lixeira *Bottle Bank Arcade* “Banco de Garrafas”, utiliza da sistemática de um jogo para arrecadação de garrafas. O jogador acumula pontos ao jogar as garrafas tornando o descarte divertido (TONÉIS, 2017, cap. 3).
- A *The World's Deepest Bin* “A lixeira mais funda do mundo”, é uma ação preparada para engajar a comunidade local no *Rolighetsteorin*, na sua instalação é utilizado sistemas com sensores de presença e caixas de som, tipo essas de computador (TONÉIS, 2017, cap. 3).

Pensando em aproveitar essa integração, gamificação e educação ambiental, despertou-se a ideia de desenvolver um sistema *web* que possa ser adaptado a uma lixeira convencional transformando-a em lixeira com características que possam incentivar e conscientizar a comunidade sobre a importância do descarte correto de lixo. Uma vez que o destino do lixo, em geral, deveria ser preocupação de todos os indivíduos, pois a má disposição do mesmo pode provocar graves consequências para todos.

Neste trabalho é proposto a criação de um *software* que utiliza a característica de gerenciamento de uma lixeira, permitindo a visualização do seu uso por dia e hora, além do desempenho do nível da bateria e quantidade de lixo que a lixeira apresenta.

1.1 Objetivo Geral

Visando envolver as pessoas de forma divertida no descarte de lixo e tornar o uso de uma lixeira convencional um ato agradável e sustentável, este projeto tem como objetivo geral propor o desenvolvimento de um sistema que mostre o desenvolvimento de uma lixeira.

Para o controle de dados da lixeira foi proposto a construção de um sistema *web* capaz de proporcionar simplicidade no gerenciamento, armazenando os dados apresentados e disponibilizando os mesmos aos usuários.

1.2 Objetivos Específicos

Os objetivos específicos foram importantes para que se tornasse capaz a realização do objetivo geral e consequentemente a conclusão do projeto. São eles:

a) Entregar o documento de requisitos;

b) Entregar os diagramas solicitados no prazo estabelecido;
c) Implementar os casos de usos para evolução do projeto.

1.3 Metodologia

A metodologia adotada por este trabalho consiste das fases teórica e experimental. A primeira baseia-se em pesquisar e adquirir o conhecimento necessário para a realização do objetivo deste trabalho. Nesta fase, é formada toda a base de conhecimento necessária para o desenvolvimento da ferramenta proposta. A fase experimental consiste em descrever os requisitos, modelar e desenvolver o sistema.

Para elaboração deste trabalho foram pesquisadas ferramentas que contribuíram no desenvolvimento do sistema *web*. Entre as tecnologias encontradas destacaram-se *Play Framework*, *Twitter Bootstrap* e a Linguagem Java.

Relacionado ao *Play Framework*, este trabalho utilizará o ambiente de desenvolvimento integrado ao *NetBeans IDE* 8.2, para edição de código; o banco de dados *MySQL* versão 6.3, para guardar os dados; a linguagem de programação Java J2EE, para implementar o painel de controle do servidor; HTML, CSS e *JavaScript* para implementar o painel de controle do usuário.

1.4 Estrutura do trabalho

Este documento está dividido em seis capítulos. O primeiro capítulo é composto pela introdução ao tema deste projeto, seus objetivos e metodologia. O segundo capítulo mostra a fundamentação teórica das tecnologias que contribuíram como base para realização deste trabalho. O terceiro capítulo é subdividido entre uma breve descrição e modelagem do projeto. O quarto capítulo é constituído de uma apresentação das áreas de interação do sistema. O quinto capítulo é apresentado os trabalhos futuros deste projeto. E no sexto capítulo é feita as considerações finais.

2. FUNDAMENTAÇÃO TEÓRICA

Neste capítulo são apresentados os aspectos teóricos deste trabalho, foram estudados os conceitos de *Play Framework* (M.V.C, JPA, MOR, *Secure*, CRUD), Banco de Dados *MySQL* 6.3, *Bootstrap* (HTML, CSS, *JavaScript*), *Pencil Project*, Linguagem Java J2EE, *Astah Community* (UML) e *NetBeans*.

2.1 Play Framework

Em 2007 o *play* foi desenvolvido por Guillaume Bort. O *framework* teve sua primeira versão oficial lançada em outubro de 2009. Inicialmente foi escrito em Java e motivado a partir de *frameworks* modernos para o desenvolvimento ágil como o *Ruby on Rails* e *Django*. O *play* foi elaborado com o objetivo de tornar o desenvolvimento de aplicação *web* com Java ou *Scala* uma tarefa fácil, divertida e produtiva, mesmo para os programadores iniciantes nessas linguagens de programação (LEROUX; KAPER, 2014).

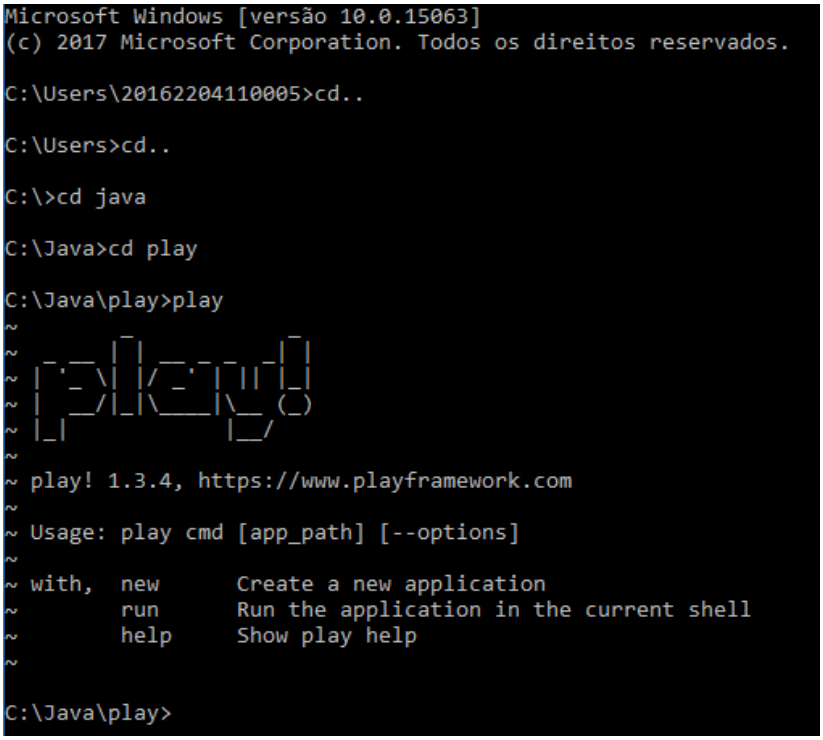


Figura 1 - Testando o play. Fonte: Elaboração própria (2018).

Com o *play* é possível integrar com qualquer banco de dados existente por meio de JPA, JDBC ou o programador pode adicionar sua própria camada de persistência. O *play* também apresenta integração com o Ambiente de Desenvolvimento Integrado (do

inglês *Integrated Development Environment - IDE*) e editores de textos, tais como: *Eclipse*, *NetBeans*, *IntelliJ*, *TextMate* e *Sublime Text* (LEROUX; KAPER, 2014). Atualmente o Play encontra-se na versão 2.6.15. A versão 1.3.4 foi utilizada no processo de construção da aplicação *web* deste trabalho.

2.1.1 M.V.C

O *Play Framework* é baseado na arquitetura M.V.C (Modelo, Visão e Controle), na qual foi a estrutura de organização das classes do sistema. O padrão M.V.C surgiu com o objetivo de possibilitar a compreensão do desenvolvimento de um código estruturado e simples. A ideia é separar todo o desenvolvimento de uma aplicação nestas três partes ou camadas (DURELLI; VIANA; PENTEADO, 2008):

- *Model* (Modelo): administra a atuação dos dados da aplicação;
- *View* (Visão): trabalha na apresentação gráfica e textual da aplicação ao usuário;
- *Controller* (Controle): executa as funcionalidades de comportamento da aplicação, e também controla a movimentação entre as camadas de visão e modelo, gerando um retorno ao usuário. Todas as requisições feitas pelo usuário são enviadas ao Controller. Este manipula os dados usando o Model e invoca a View correta, de acordo com a ação executada ou com os resultados vindos do Model.

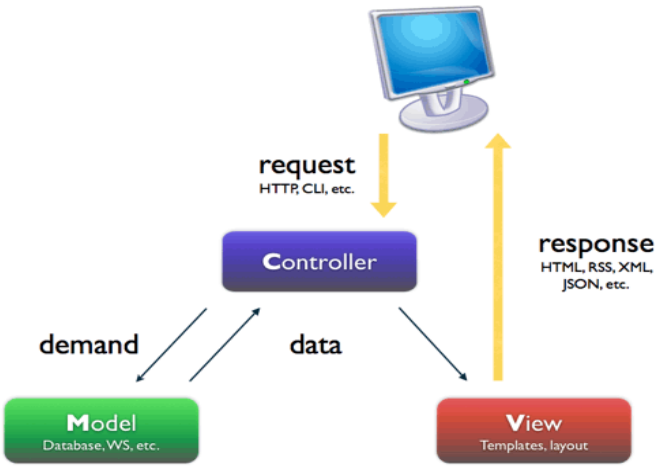


Figura 2 - Padrão M.V.C. Fonte: Webhozz[1]

[1] https://www.webhozz.com/blog/konsep-mvc-di-codeigniter/

A grande vantagem de se utilizar o padrão M.V.C é a separação de lógica e apresentação, sendo que isso favorece o trabalho em equipe.

2.1.2 Java Persistence API (JPA)

A *Java Persistence API* (JPA) é um *framework* para persistência em Java, que põe a disposição uma API de mapeamento objeto-relacional e soluções para agregar persistência com sistemas corporativos escaláveis. Gavin King foi líder da primeira versão da especificação, e também foi o criador do *Hibernate* junto a outros desenvolvedores (CAMPOS, 2010).

Segundo Campos (2010), para preencher as necessidades relacionadas ao mapeamento com JPA, surgiram vários *frameworks* e tecnologias de persistência que simplificam o trabalho do desenvolvedor e agregam a sua produtividade, concedendo poderosas APIs de 54 manipulação de dados. Cada ferramenta apresenta uma forma diferente de realizar o mapeamento objeto-relacional, porém os conceitos são iguais e relativamente simples, que baseiam-se em POJOs (*Plain Old Java Objects*). A implementação da JPA é executada por um *persistence provider* (provedor de persistência), que define como as coisas atuam, através da implementação das interfaces definidas pela especificação da JPA.

Ainda segundo o mesmo autor, para o mapeamento das entidades, o JPA manuseia de metadados ou *annotations*. Com relação a transações, o JPA libera apenas o uso de transações ou sem nenhuma transação que é o modo padrão sobre o qual ele funciona. Para carregar os dados, o JPA provê do suporte a *queries* SQL nativas, fornece JPQL e a consulta do tipo *criteria* por meio de uma API. A JPQL é uma linguagem de consulta orientada à objetos com base a funções do ANSI SQL. A API de *criteria* é inteiramente orientada a objetos, e possibilita a formação de objetos exemplo, inserção de expressões e aproveitamento de funções do SQL sem a necessidade de criação de uma *query* específica.

Os objetos que controlam o funcionamento do JPA, presentes no pacote *javax.persistence*, são: *EntityManagerFactory* e a *EntityManager*.

2.1.3 Mapeamento Objeto Relacional (MOR)

Segundo Silberschatz et al., (2016), Mapeamento Objeto Relacional é persistir de modo imediato e compreensível, os objetos de um aplicativo para tabelas em um banco de dados relacional. Em relação à conversão de dados entre banco de dados relacionais e linguagens de programação orientada a objetos.

Em banco de dados, entidades são apresentadas por tabelas, que dispõem de colunas que arquivam propriedades de diversos tipos. Uma tabela pode se unir com outras e criar relacionamentos variados.

Em uma linguagem orientada a objetos, entidades são classes, e objetos dessas classes representam elementos que encontram-se no mundo real.

Algumas vantagens de implementar um mapeamento objeto relacional (Silberschatz et al., 2016):

- Desempenho – o tempo poupado no desenvolvimento pode ser aplicado a programar melhorias do sistema.
- Manutenibilidade – por diminuir o número de linhas do código fonte do sistema, menor será o trabalho de correção do sistema.
- Eficiência – com a omissão dos códigos SQL no código fonte, as classes passam a ser mais clara e o sistema é ampliado em menor tempo.
- Autonomia de Fornecedor – por mais que um banco de dados empregue a mesma linguagem SQL, alguns comandos, tipos de dados, podem ser contrários de um banco para outro.

2.1.4 Secure

O módulo *Secure* simples auxilia a configurar o gerenciamento básico de autenticação e autorização em seu aplicativo. Ele fornece um controlador simples "*controllers.Secure*" que determina um grupo de interceptores que pode inserir com facilidade aos seus controladores usando a anotação "@*With*".

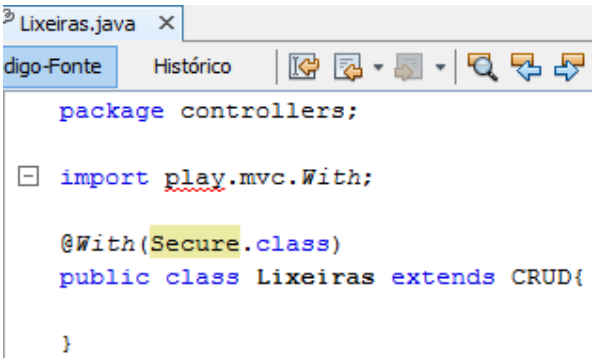


Figura 3 - Anotação @With. Fonte: Elaboração própria (2018).

Para a configuração, é necessário ativar o módulo de segurança para o aplicativo. No arquivo */conf/dependencies.yml*, ative o módulo *Secure* acrescentando uma linha após *require*:

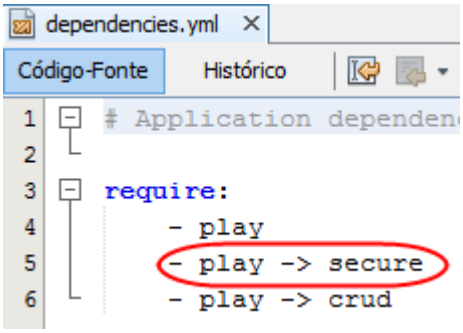


Figura 4 - Módulo Secure. Fonte: Elaboração própria (2018).

Posteriormente, deve executar comando o *play dependencies* para agregar o módulo ao seu aplicativo.

Para a importação das rotas seguras padrão, é fundamental ir no arquivo *conf/routes*, importar as rotas do módulo padrão adicionando as seguintes linhas:

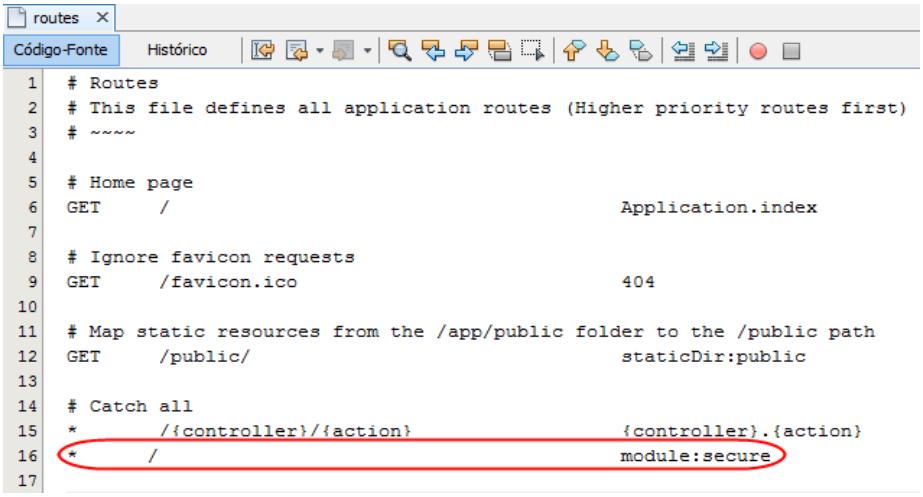


Figura 5 - Importando as rotas do módulo. Fonte: Elaboração própria (2018).

O módulo *Secure* gera um "*play secure:override*" que pode ser usado para sobrepor a página de login, para que possa ser personalizado para qualquer aplicativo. Isso funciona copiando o arquivo correspondente do módulo para um arquivo em seu aplicativo que será utilizado em seu lugar (PLAY, 2018).

2.1.5 CRUD

Classe que se comunica com o banco de dados, com métodos padrões que as entidades do sistema utilizam. É responsável pelas funções padrões de inserção, edição, exclusão e listagem.

O módulo CRUD (em inglês: "*Create*, *Read*, *Update and Delete*" ou seja, "Criar, Ler, Atualizar e Excluir"), gera uma interface *web* totalmente utilizável para seus objetos do modelo JPA (PLAY, 2018).

Na configuração, é necessário ativar o módulo para o aplicativo. No arquivo */conf/dependencies.yml*, ative o módulo CRUD acrescentando uma linha após *require*:

Figura 6 - Módulo CRUD. Fonte: Elaboração própria (2018).

Posteriormente, deve executar comando o *play dependencies* para agregar o módulo ao aplicativo.

O módulo CRUD fornece um comando "*crud: override"* que é usado na linha de comando para substituir o modelo padrão. Isso funciona porque o módulo CRUD carrega modelos de seu aplicativo, se estiverem presentes, em vez de seus próprios. Normalmente é utilizado o comando "*crud: ov*" em vez de "*crud: override"* (PLAY, 2018).

2.2 MySQL 6.3

Para configurar o banco de dados foi utilizada a ferramenta *MySQL* versão 6.3, desenvolvido pela empresa *Oracle Corporation*[2].

De acordo com Niederauer (2008), o *MySQL* 6.3 é um *software* livre que sua utilização possibilita a criação de diagramas que correspondem a visualização e uma melhor compreensão do banco de dados, também possui a facilidade de gerar o código SQL (*Structured Query language*) para criação do banco de dados no SGBD (Sistema de Gerenciamento de Banco de Dados). Entre os bancos de dados com código-fonte aberto, o *MySQL* é o mais popular.

Algumas das características desta ferramenta são (NIEDERAUER, 2008):

- ➢ Suportar qualquer plataforma atual;
- ➢ Excelente desempenho e segurança;
- ➢ Clareza no manuseio;
- ➢ É um *software* livre com base na Licença Pública Geral.

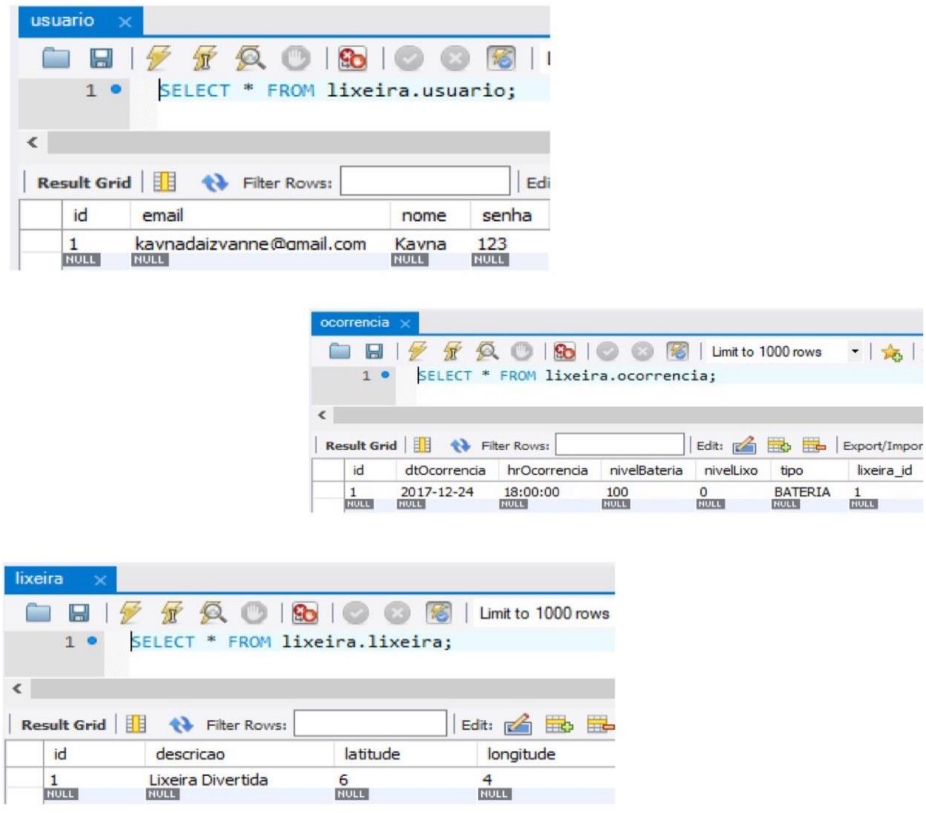


Figura 7 - Testando o banco de dados do sistema. Fonte: Elaboração própria (2018).

[2] https://www.oracle.com/index.html

O *download* do *MySQL* é feito através do site: http://www.mysql.com. Na seção de produtos, escolha "*Database Server*" e baixe o arquivo de instalação para o seu sistema operacional.

2.3 Twitter Bootstrap

Para a aplicação de HTML no nosso sistema foi utilizado o *Twitter Bootstrap* versão v4.1.1, que é uma ferramenta gratuita que tem como finalidade a criação de sites, e é bastante utilizado para esboçar telas em HTML que podem ser acessadas via navegador *web* ou dispositivo móvel.

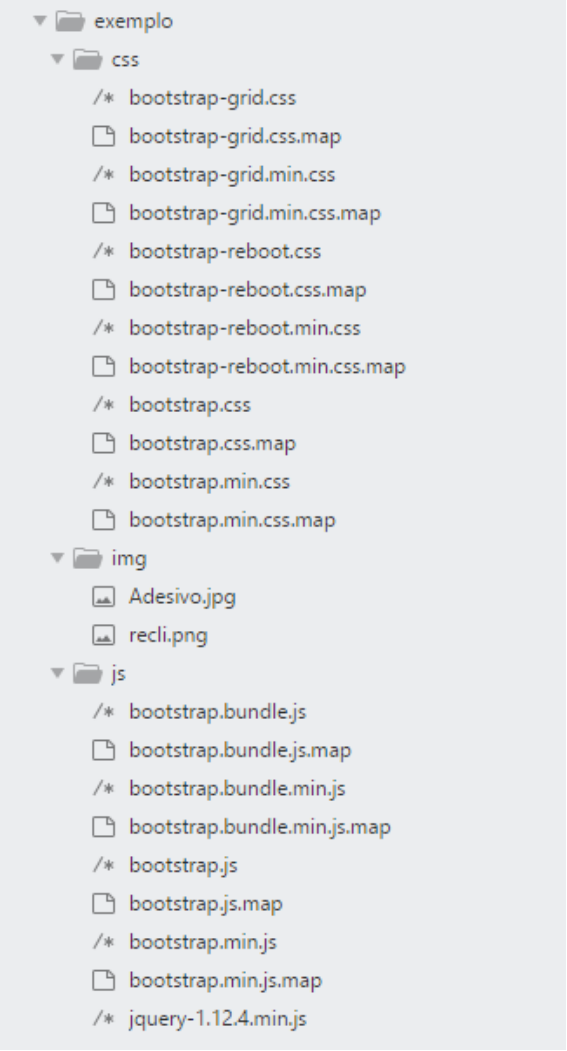


Figura 8 - Estrutura de arquivos. Fonte: Elaboração própria (2018).

O programa possui a inclusão de HTML e *designs* baseados em CSS para formulários, botões, navegação e outros componentes da interface. O uso deste

programa é simples e prático, basta apenas baixar o código fonte desejado e inserir na página HTML.

O *Twitter Bootstrap* foi desenvolvido por Jacob Thornton e Mark Otto, engenheiros do *Twitter*, e tem como objetivo simplificar o desenvolvimento ou manutenção de um projeto usando reaproveitamentos de códigos ou bibliotecas (BARBIERE, 2017).

2.3.1 Html, Css e JavaScript

HTML do inglês *Hypertext Markup Language* (Linguagem de Marcação de Hipertexto), foi criado por Tim Berners-Lee em 1990, e é composta de texto com *tags*, funções e atributos, marcando partes de texto que indicam ao navegador formatá-lo, incluir figuras, desenhar, etc. O HTML é habilitado a mostrar recursos dos aplicativos GUI, como os botões de rádio, listas de opções, caixas de texto multilinhas etc. Os navegadores são preparados para a exibição tais recursos (SILVA, 2008, p. 26).

Cascade Style Sheets ou Folha de Estilo é uma linguagem para incluir estilo a páginas *web*, determinando a apresentação de conteúdos e aparências. A sintaxe da linguagem é um pouco diferente de HTML e exige um esforço de aprendizagem, pois é formada sempre por uma regra, e cada regra é formada por duas partes principais (MUNZLINGER, 2010):

- Seletor: onde é estabelecido o componente que sofrerá a formatação;
- Declaração: composta por propriedade que é o nome de onde será alterado; e valor, onde é o conteúdo concedido à propriedade.

Pode-se usar três modos de Folhas de Estilo CSS:

- Incorporado;
- Vinculado;
- In-Line.

JavaScript é uma linguagem embutida dentro de arquivos HTML, que surgiu em 1995, ela proporciona a criação de animações, botões e funções para validação de campos em formulários HTML. Seu principal recurso é a capacidade de fazer com que a página HTML seja dinâmica, de forma que o usuário possa exercer interação. Os

scripts documentados em *JavaScript* permitem o controle dos objetos de uma página HTML, tendo a simplicidade para alterá-los (MILETTO; BERTAGNOLLI, 2014, p. 96).

Ambas as tecnologias foram utilizadas neste trabalho para desenvolvimento da interface cliente do sistema.

2.4 Pencil Project

Para auxiliar na criação dos nossos esboços foi utilizado o *Pencil Project* versão 3.0.4, que é um *software* gratuito desenvolvido em 2008 pela empresa *Evolus*, e é sobretudo uma ferramenta gratuita de prototipagem que permite criar interfaces gráficas, e possui uma acessível utilização e simplicidade no manuseamento.

Sua maior característica é onde o usuário tem a opção de escolher entre diversos formatos para exportar seus arquivos, nos quais se destaca: HTML, PDF e PNG. As ferramentas deste programa executam no modo *drag and drop*, onde é necessário somente arrastar e soltar na barra (EVOLUS, 2008-2012).

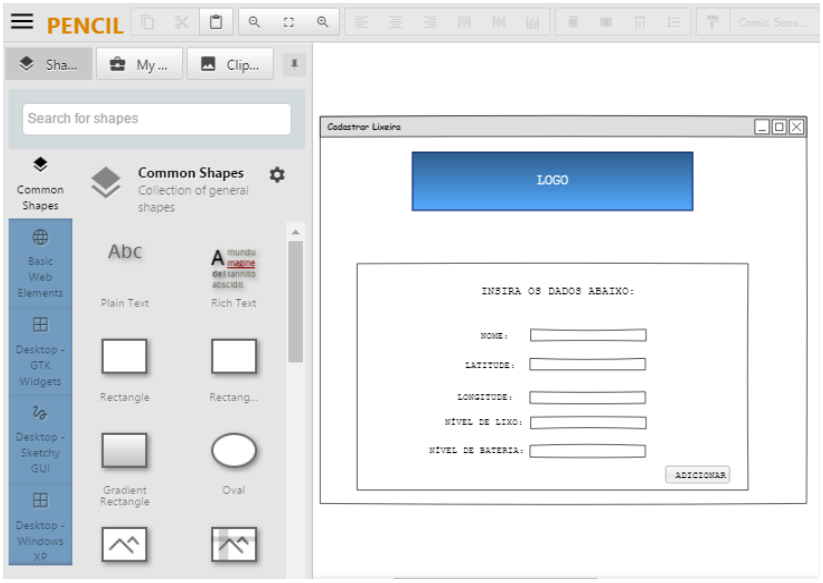


Figura 9 - Utilizando o Pencil Project. Fonte: Elaboração própria (2018).

A ferramenta *Pencil* permite que seja modeladas soluções de software fazendo uso de *templates*, que podem ser usados em diversos cenários, para reproduzir painel de controle. Ela apresenta um conjunto de visões que possibilita a criação de diversos modelos de interfaces. Possui por padrão modelos baseados no paradigma *desktop*, modelos de interfaces para dispositivos móveis e modelos *web* (NETO, 2017).

2.5 Linguagem Java

A linguagem utilizada no nosso sistema foi a Linguagem Java, desenvolvida na década de 90 pela *Sun Microsystem* e atualmente pertence a *Oracle*, é uma linguagem de programação completa apropriada para a elaboração de aplicações, e estar ligada aos grandes avanços de computação mundial. Sua multifuncionalidade possibilita o programador recursos capazes de uma variedade de aplicativos que podem ou não ter a necessidade do uso de recursos de conectividade (SEBESTA, 2018, p. 87).

Algumas das suas características são (SEBESTA, 2018, p. 88):

- Naturalidade e competência, pois abrange um conjunto de bibliotecas que oferecem partes das funções básicas da linguagem, envolvendo criação de interface gráfica e práticas de acesso à rede;
- Linguagem compilada e interpretada, só é necessária escrever e compilar o código somente uma vez, pois os *bytecodes* gerados serão executados em qualquer plataforma de *hardware* e *software*;
- Segurança, já que as possibilidades de erro são mínimas e a linguagem tem um esquema para assegurar a integridade de código.

Os programas desenvolvidos em Java podem possuir dois tipos: o *applets*, que são os que dependente de um navegador *web* para ser executados; e o *applications*, que são os programas que o interpretador identifica e executa.

2.6 Astah Community

Para especificar os requisitos e ajudar na visualização do sistema, utilizamos a ferramenta *Astah Community*, aplicado no desenvolvimento de diagramas necessários para representar os requisitos.

O *Astah* foi desenvolvido pela *Change Vision* e é um *software* gratuito para modelagem de diagramas UML, o mesmo exibe uma interface de fácil navegação dividida em várias seções, cada uma com sua respectiva finalidade (LIMA, 2016).

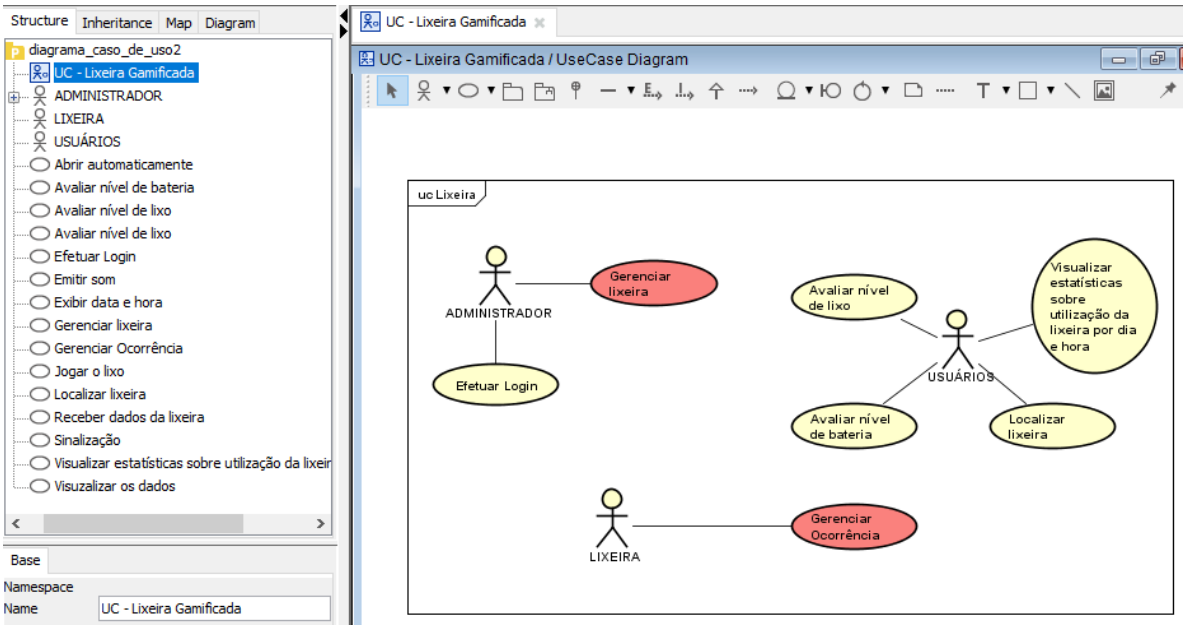


Figura 10 - Utilizando o Astah Community. Fonte: Elaboração própria (2018).

O *Astah Community* permite a criação de nove tipos de diagramas, alguns deles são: diagramas de classe, diagramas de sequência, diagramas de caso de uso, diagramas de componentes, entre outros.

2.6.1 UML

Segundo Gudwin (2010), a *Unified Modeling Language* (UML) é uma linguagem gráfica aplicada para detalhar, criar e documentar os elementos de sistemas e caracteriza-se como o protocolo fundamental para comunicação da indústria de software. A UML é livre de linguagens de programação. É estabelecido mecanismo extensível que permite a criação de projetos com novos conceitos e limitações específicas.

Através da UML foi possível criar os nossos diagramas proposto neste trabalho, como:

O diagrama de caso de uso: devem representar as funcionalidades que o sistema se submete a resolver do ponto de vista dos atores externos. Um ator é qualquer entidade externa, como: um usuário, um dispositivo ou outro sistema. Esse diagrama revela a interação entre um usuário e o sistema que ele está manuseando (GUEDES, 2018).

O diagrama de classe: é parte principal dos métodos orientados a objetos, onde expõe os tipos de objetos do sistema e os relacionamentos estáticos que ocorrem entre eles. Ele é estático, pois sua estrutura determinada por ele é sempre válida em qualquer instante do ciclo de vida de um projeto. Ele exibe os atributos, as operações de uma classe e as restrições associadas a ela (GUEDES, 2018).

2.7 NetBeans IDE 8.2

Para a implementação do sistema foi utilizado o *NetBeans IDE* (Ambiente de Desenvolvimento Integrado) 8.2, fundado pela *Sun Microsystem* em 2000, é um conjunto de bibliotecas, módulos e APIs (*Application Program Interface*) que possibilita ao usuário escrever, compilar, depurar e executar programas, oferecendo uma documentação de configuração bem desenvolvida (ORACLE, 2018).

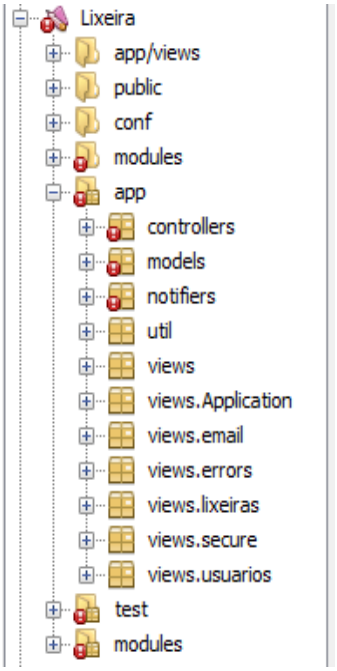


Figura 11 - Estrutura de arquivos. Fonte: Elaboração própria (2018).

O *NetBeans* é uma ferramenta gratuita e não tem nenhuma restrição de uso, conta com duas ferramentas integradas que são os *frameworks JasperReports* e o *iReports*, que tem como finalidade facilitar o uso de forma integrada. Suas duas bases são, o *NetBeans IDE* e a Plataforma *NetBeans*. O suporte em Java não é o único, o *NetBeans* também fornece suporte nas linguagens C, C++, XML, HTML, PHP e outras (MERLIN, 2009).

3. MODELAGEM DO PROJETO

Para atingir os objetivos deste trabalho foi projetado e implementado o sistema “A lixeira divertida”, que nada mais é do que um sistema *web* que tem por objetivo dar suporte e mensurar a utilização de lixeiras. Dessa forma é previsto a criação de módulos que auxiliam no armazenamento e apresentação dos dados capturados pelo dispositivo. Os usuários poderão visualizar informações, como: uso por dia e hora, desempenho do nível da bateria e quantidade de lixo que a lixeira apresenta.

3.1 Atores do sistema

Esta seção é composta por uma tabela que apresenta os usuários e suas respectivas ações relacionadas ao sistema.

Tabela 1 - Descrição dos usuários

Lixeira	A lixeira gerencia as ocorrências e envia os dados para o administrador.
Usuários	Os usuários poderão descartar seus lixos de uma forma divertida, e acessar os dados para obter informações da lixeira.
Administrador	O administrador irá gerenciar os dados da lixeira.

Fonte: Elaboração própria (2018).

3.2 Levantamento de requisitos

Esta seção é subdividida entre requisitos funcionais e requisitos não funcionais. Os requisitos apresentados pelo professor Robercy Alves da Silva, orientador deste trabalho.

3.2.1 Requisitos funcionais

Os requisitos funcionais apresentam as principais funcionalidades que o sistema deve efetuar. Eles são representados pela tabela 2:

Tabela 2 - Lista de requisitos funcionais

Cód.	Descrição	Prioridade
RF01	O sistema exibirá um formulário para o administrador	Alto

Localizar lixeira	informar a localização das lixeiras para seus usuários, através da latitude e longitude.	
RF02 Exibir estatísticas	O sistema vai permitir ao usuário verificar data e hora da utilização da lixeira através de um *real time* programado no Arduino.	Baixo
RF03 Avaliar nível de lixo	A partir do momento que os usuários forem depositando os seus lixos, irá conter um sensor de proximidade para detectar o nível de lixo e apresentar através das cores amarelo (quantidade de lixo mediano) e vermelho (cheia).	Baixo
RF04 Avaliar nível de bateria	O usuário poderá detectar o total de carga que se encontra a bateria, caso esteja com a potência baixa, irá emitir um sinal de alerta.	Baixo
RF05 Receber dados da lixeira	A lixeira poderá enviar ao sistema dados sobre dia e hora que foi utilizada, nível de lixo e nível de bateria em que ela apresenta.	Médio
RF06 Realizar cadastro	O sistema possibilitará aos usuários realizar cadastro no sistema, informando seu: nome, e-mail e senha.	Alto
RF07 Efetuar Login	O sistema deve permitir aos usuários realizar login para permissão a área restrita do sistema. O usuário precisará informar seu: e-mail e senha.	Alto
RF08 Recuperar Senha	O sistema deve conceder aos usuários que eles recuperem sua senha informando o e-mail cadastrado.	Alto

Fonte: Elaboração própria (2018).

3.2.2 Requisitos não funcionais

Os requisitos não funcionais estão associados às limitações do sistema quanto a sua disponibilidade, satisfação e segurança. Eles são representados pela tabela 3:

Tabela 3 - Lista de requisitos não funcionais

Cód.	Descrição	Prioridade
RNF01 Tempo de resposta	O usuário ao aproximar-se da lixeira poderá inserir seu lixo por cerca de 10 segundos, após sua utilização será enviado ao sistema o tipo de ocorrência.	Médio
RNF02 Usabilidade	O sistema deverá prover de uma interface gráfica simples e intuitiva, de fácil navegação para facilitar o uso do mesmo por parte do usuário.	Alto
RNF03 Segurança	O sistema deve dispor de mecanismo de segurança para autenticação do administrador e controle de acesso às funcionalidades do sistema apenas para administradores cadastrados.	Alto
RNF04 Criptografia de senha de acesso	O sistema deverá prover a criptografia da senha de usuário. Deverá ser criptografado em MD5.	Baixo
RNF05 Validação de e-mail	O usuário deverá incluir no endereço de e-mail o caractere especial “@”.	Alto
RNF06 Acesso Restrito	O sistema não deverá autorizar o acesso de usuários não cadastrados a área restrita do sistema.	Alto

Fonte: Elaboração própria (2018).

3.3 Diagrama de Caso de Uso

Como mostrado na Figura 12, o sistema possui 3 atores, e sete casos de uso.

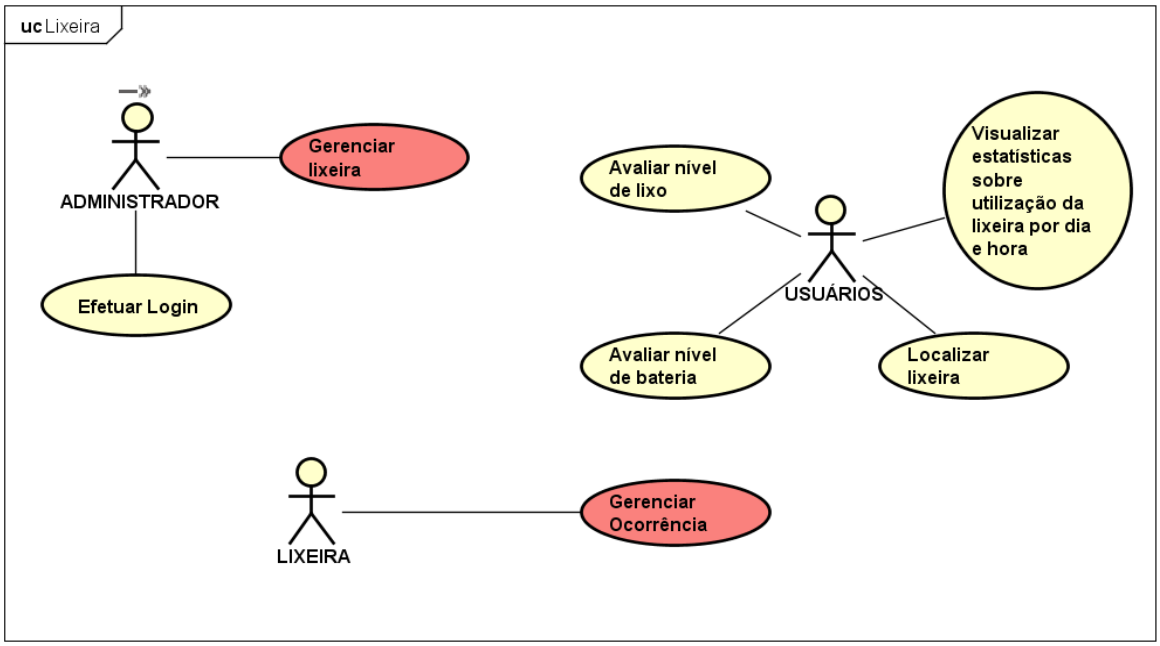


Figura 12 - Diagrama de caso de uso. Fonte: Elaboração própria (2018).

3.4 Caso de uso expandido

Neste subcapítulo são apresentadas as expansões dos casos de usos de maior risco.

Tabela 4 - Expansão de cadastrar lixeira

➢ **Cadastrar lixeira** **Descrição:** Este caso de uso especifica a ação do administrador. Após a realização do cadastro e login ao sistema, o administrador fica habilitado a fazer alterações referentes a parte de exibição das lixeiras. O mesmo poderá cadastrar, editar e remover lixeiras. Apenas administradores cadastrados podem fazer alterações no sistema. **Ator**: Administrador. **Pré-condições:** 1. O ator precisa realizar o cadastro e fazer login no sistema para efetuar o cadastramento da lixeira. **Pós-condições:** 1. O ator fica habilitado a realizar ações na área restrita do sistema.

Requisitos funcionais:

1. **RF06**. O portal deve prover uma interface gráfica para administradores se cadastrarem no sistema.

2. **RF07**. O portal deve prover uma interface para os administradores se autenticarem no sistema.

Requisitos não funcionais:

1. **RNF06**. O sistema não permitirá o acesso a usuários não cadastrados no sistema.

2. **RNF02**. O sistema deverá prover de uma interface gráfica simples e intuitiva, de fácil navegação.

Fluxo Básico:

1) O ator clica em entrar no sistema.

2) O ator clica em “Gerenciar Lixeiras”.

3) O ator clica em “Cadastrar-se”.

4) O sistema solicitará as informações necessárias para efetuar cadastro:

   a) Nome;

   b) E-mail; e

   c) Senha.

5) O ator informa os dados de cadastramento.

6) Os dados acima são armazenados.

7) O sistema solicita as informações obrigatórias para efetuar login:

   a) E-mail; e

   b) Senha.

8) O ator informa os dados de autenticação.

9) O sistema valida os dados de autenticação.

10) O sistema exibe o painel de controle no qual o ator pode fazer ações relacionadas ao gerenciamento das lixeiras.

Fluxo Alternativo A: No passo 9 do Fluxo Básico, caso haja algum erro na autenticação relacionado aos dados informados:

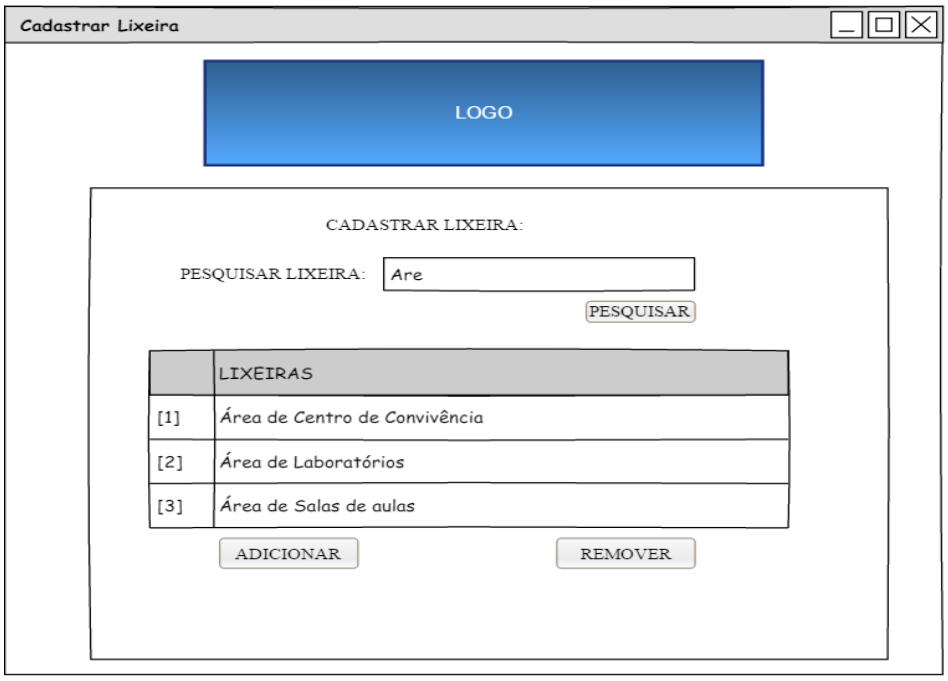
1. O sistema informa o erro ao ator.
2. O fluxo retorna ao passo 7 do fluxo básico.

Fluxo Alternativo B: No passo 7 do Fluxo Básico, caso o ator tenha esquecido a

senha:

1. O sistema informa o erro ao ator.
2. O ator clica em “Esqueceu Senha?”.
3. O ator informa o e-mail para recuperar senha.
4. O fluxo retorna ao passo 7 do fluxo básico.

Protótipos



Fonte: Elaboração própria (2018).

Tabela 5 - Expansão de cadastrar ocorrência

➢ **Cadastrar Ocorrência**

Descrição:
Este caso de uso especifica a ação da lixeira para cadastrar ocorrências. Após o cadastramento, é possível visualizar o nível do lixo, nível de bateria, data e hora da ocorrência.

Ator:

Lixeira.

Pré-condições:

1. O ator precisa estar cadastrado no sistema para envio de informações.

Pós-condições:

1. O ator fica habilitado a enviar as ocorrências para o sistema.

Requisitos funcionais:

1. **RF05**. A lixeira poderá enviar ao sistema dados sobre dia e hora que foi utilizada, nível de lixo e nível de bateria em que ela apresenta.

Requisitos não funcionais:

1. **RNF01**. O usuário ao aproximar-se da lixeira poderá inserir seu lixo por cerca de 10 segundos, após sua utilização será enviado ao sistema o tipo de ocorrência.

Fluxo Básico:

1) O administrador acessa a interface de cadastramento de lixeiras.
2) O administrador clica em “Adicionar Lixeira”.
3) O sistema solicitará as informações necessárias para efetuar cadastro:
    a) Descrição;
    b) Latitude; e
    c) Longitude.
4) O administrador cadastra o ator no sistema.
5) O ator torna-se habilitado a enviar os seguintes dados ao sistema:
    a) Data da ocorrência;
    b) Hora da ocorrência;
    c) Tipo da ocorrência;

d) Nível de bateria; e

e) Nível de Lixo.

6) Os dados acima são armazenados.

7) O sistema exibe o painel de controle no qual apresenta as lixeiras e suas respectivas informações.

Fluxo Alternativo A: No passo 5 do Fluxo Básico, caso haja algum erro no envio de dados, é informado:

1. É enviado uma mensagem:

a) "400", "Requisição inválida", "Data e/ou hora inválido(s)";

b) "400", "Requisição inválida", "Não foi possível realizar o cadastro da ocorrência";

c) "400", "Requisição inválida", "A lixeira não está cadastrada".

Fonte: Elaboração própria (2018).

3.5 Diagrama de Classe

A Figura 13 mostra o diagrama parcial das classes modeladas deste trabalho.

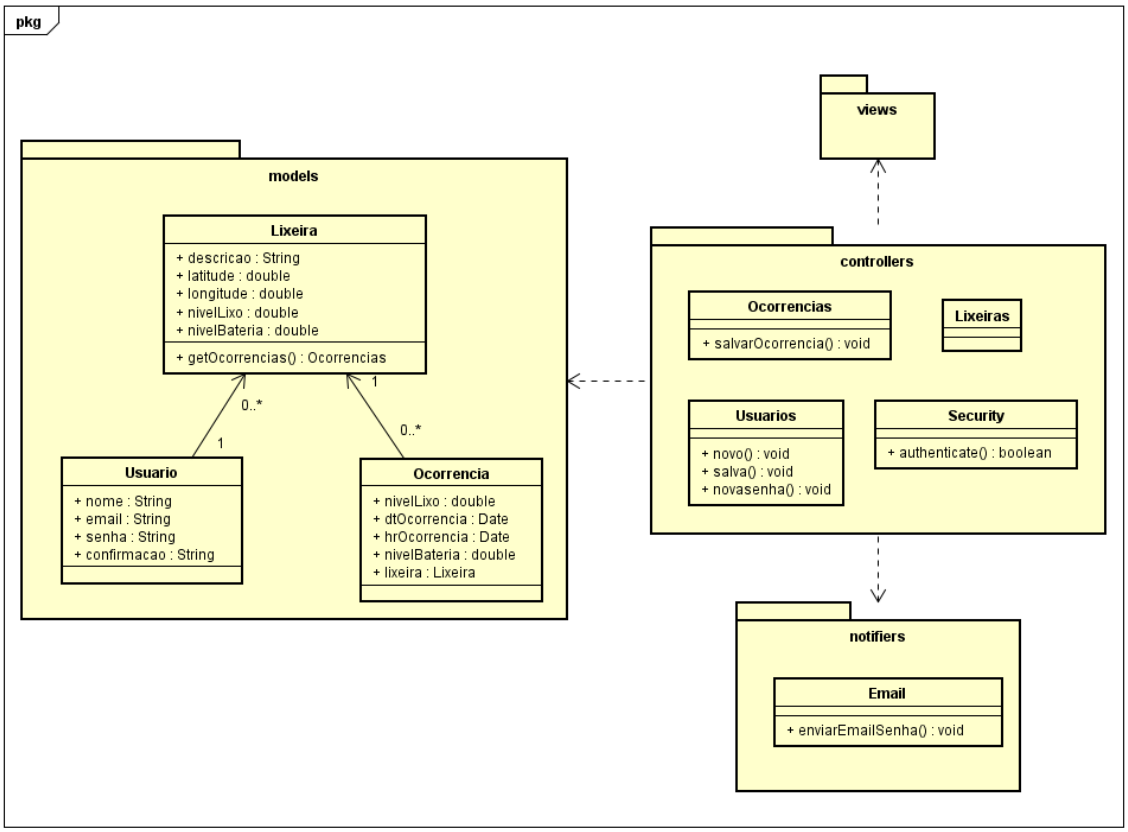


Figura 13 - Diagrama de classe. Fonte: Elaboração própria (2018).

4. RESULTADOS

Nesta seção, é feita a apresentação das funcionalidades do sistema implementadas para este trabalho.

4.1 Gerenciar Lixeira

Sendo o usuário um administrador e estando logado no sistema, ele terá acesso ao gerenciador de lixeiras. Onde é possível adicionar, editar os dados do dispositivo, e também a opção de deletar as lixeiras, uma vez escolhida essa opção o dispositivo e todos os registros a ele associados serão excluídos do banco de dados.

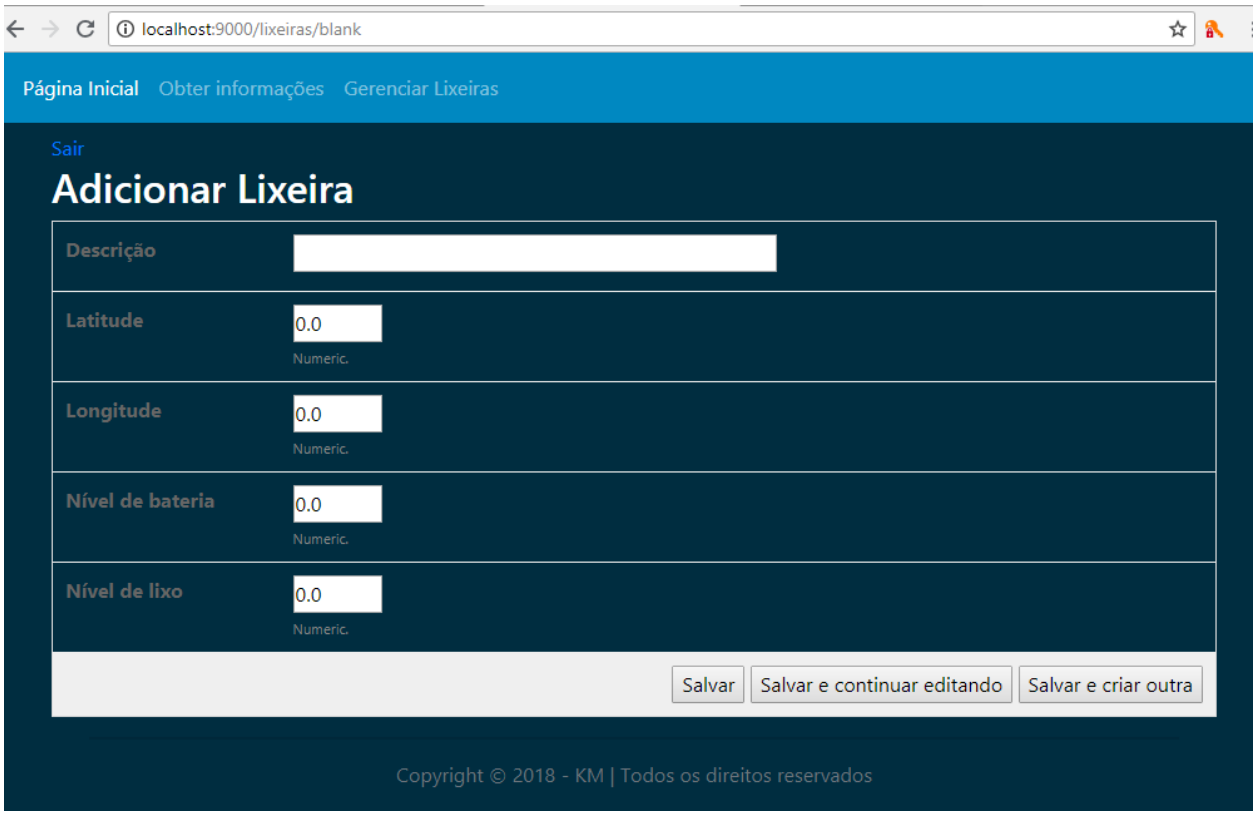


Figura 14 - Funcionamento do adicionar lixeira. Fonte: Elaboração própria (2018).

O usuário sendo um administrador, ele terá acesso a lista de lixeiras. Onde será possível detalhar a visualização dos dados contidos nelas.

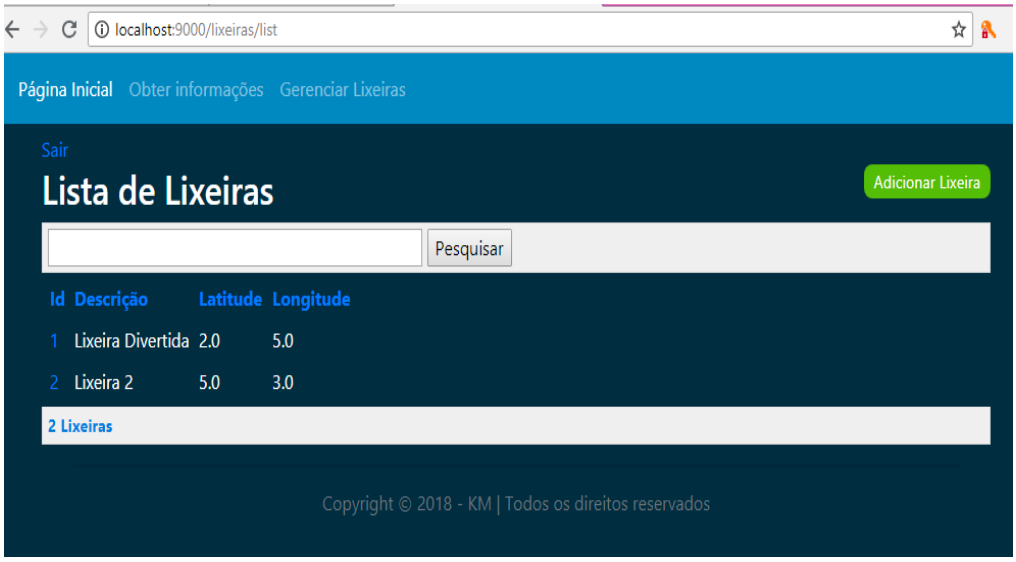


Figura 15 - Funcionamento do listar lixeira. Fonte: Elaboração própria (2018).

4.2 Obter Informações

Ao usuário, é dada a opção de obter as informações da lixeira, nela será possível visualizar os dados contidos e ele não precisará efetuar login para ter acesso. Outra opção para análise dos dados coletados é a visualização por meio de gráfico.

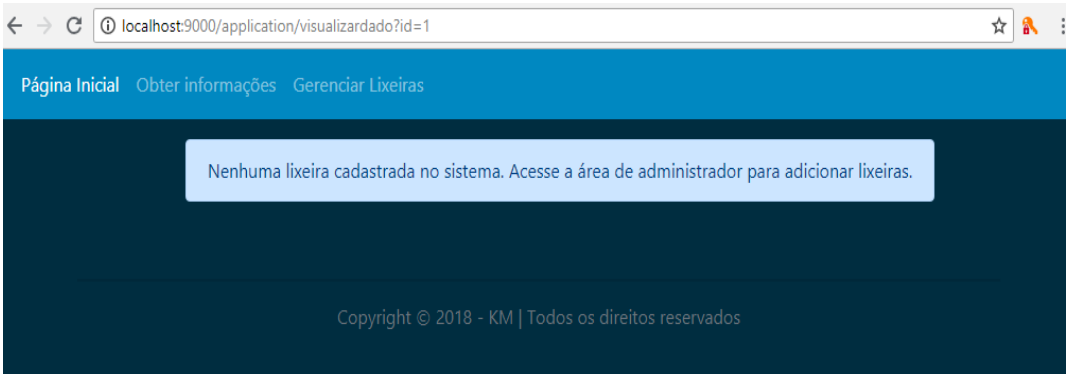


Figura 16 - Funcionamento do obter informações. Fonte: Elaboração própria (2018).

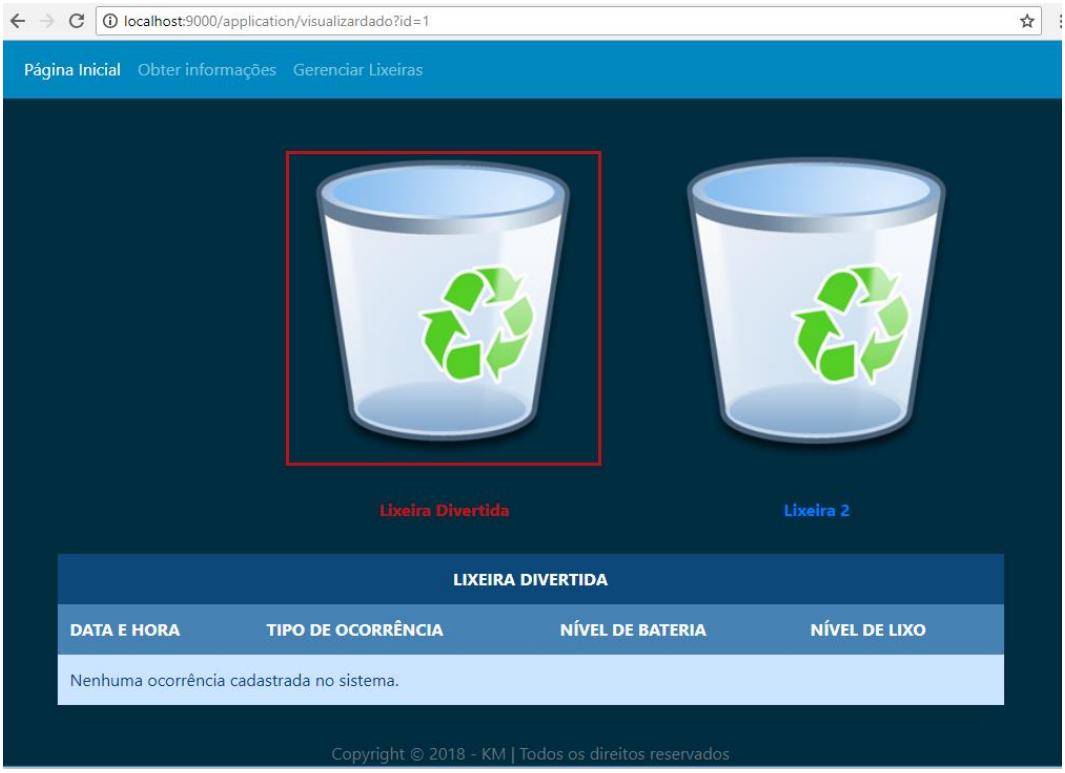


Figura 17 - Funcionamento com uma lixeira cadastrada. Fonte: Elaboração própria (2018).

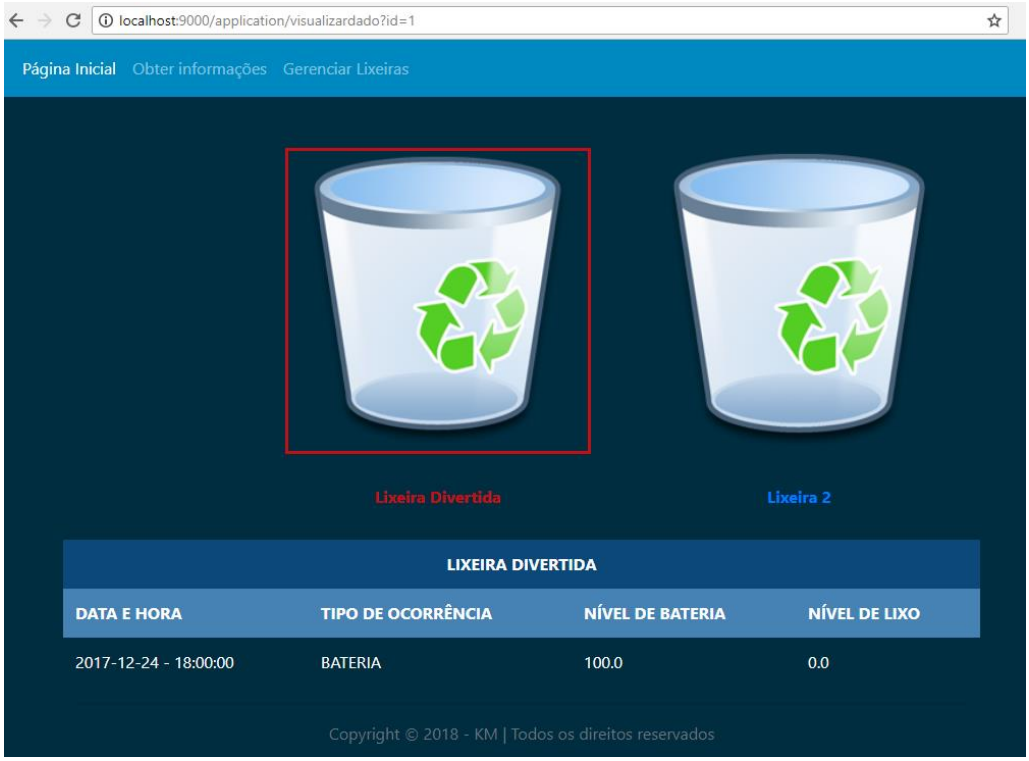


Figura 18 - Funcionamento com ocorrência cadastrada. Fonte: Elaboração própria (2018).

4.3 Realizar Autenticação

Para ter acesso ao sistema o usuário deve informar seu e-mail de login e senha. Caso um dos itens apresentados for incorreto, o sistema exibirá uma mensagem de erro e rejeitará o acesso do usuário ao sistema.

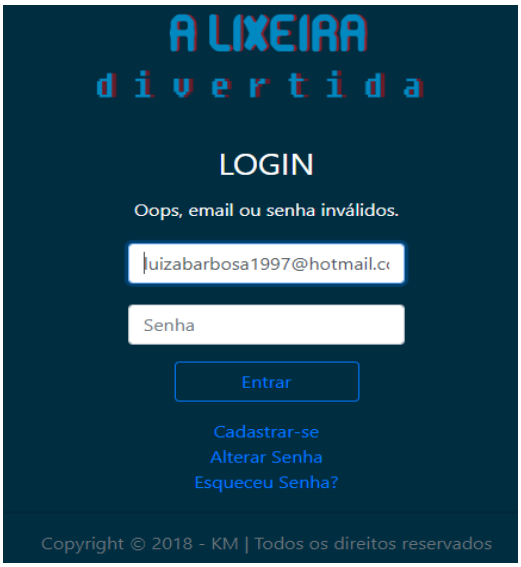


Figura 19 - Realizar autenticação. Fonte: Elaboração própria (2018).

Caso sejam informados conforme solicitado, o sistema encaminhará o usuário à página inicial do sistema, onde lhe serão apresentadas as informações sobre a lixeira divertida.

4.4 Efetuar Login

O administrador para ter acesso ao sistema terá interação com a tela de login. Nesse caso, qualquer usuário é considerado um usuário desconhecido. Caso o usuário não possua um cadastro no sistema, é oferecido a possibilidade de realizar um cadastro. Uma vez que o cadastro é finalizado, o usuário terá acesso a página para visualizar a lista de lixeiras.

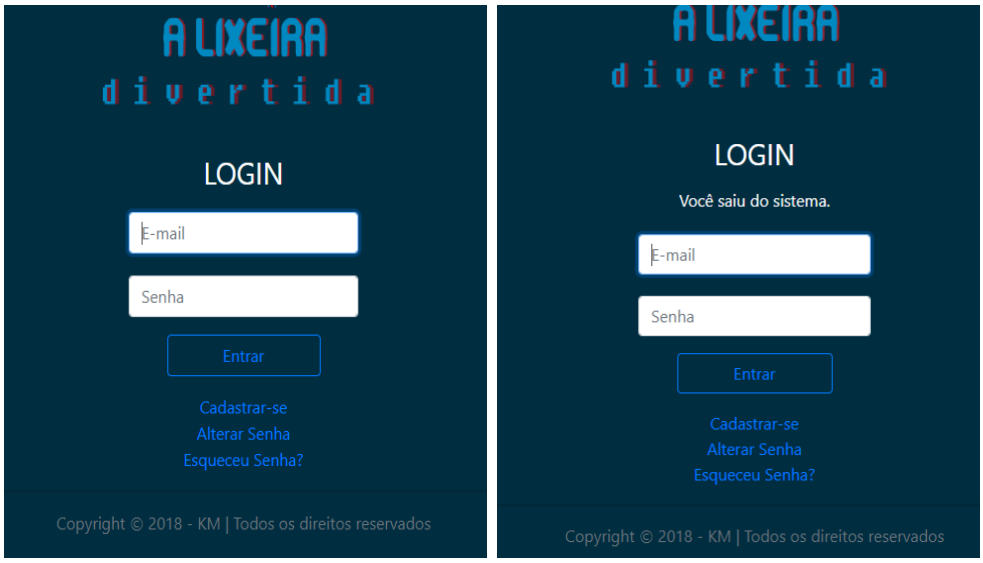


Figura 20 - Funcionamento do efetuar login. Fonte: Elaboração própria (2018).

Quando o usuário solicitar sua saída do sistema, irá direcioná-lo a página de login e o sistema exibirá uma mensagem.

4.5 Recuperar Senha

A tela de recuperação de senha é onde o usuário pode recuperar sua senha com cadastro previamente feito no sistema. É disponível quando aciona a opção de “Esqueceu Senha?”, na tela de login. Um formulário será exibido com o campo e-mail, onde o usuário preencherá com seu e-mail cadastrado para que seja enviado sua senha.

Figura 21 - Recuperar senha. Fonte: Elaboração própria (2018).

4.6 Cadastrar Usuário

Caso o usuário não seja administrador ainda, o sistema permite o usuário preencher um cadastro, fornecendo os dados: nome de usuário, e-mail e senha, para ter acesso aos privilégios da lixeira. Caso o usuário tenha realizado o cadastro com sucesso, será redirecionado para a tela de login.

Figura 22 - Cadastrar usuário. Fonte: Elaboração própria (2018).

4.7 Página Inicial

A tela inicial é a interface principal do sistema, pois é dividida em *tabs* onde o usuário é redirecionado para telas diferentes. As *tabs* são compostas por: Obter Informações e Gerenciar Lixeiras.

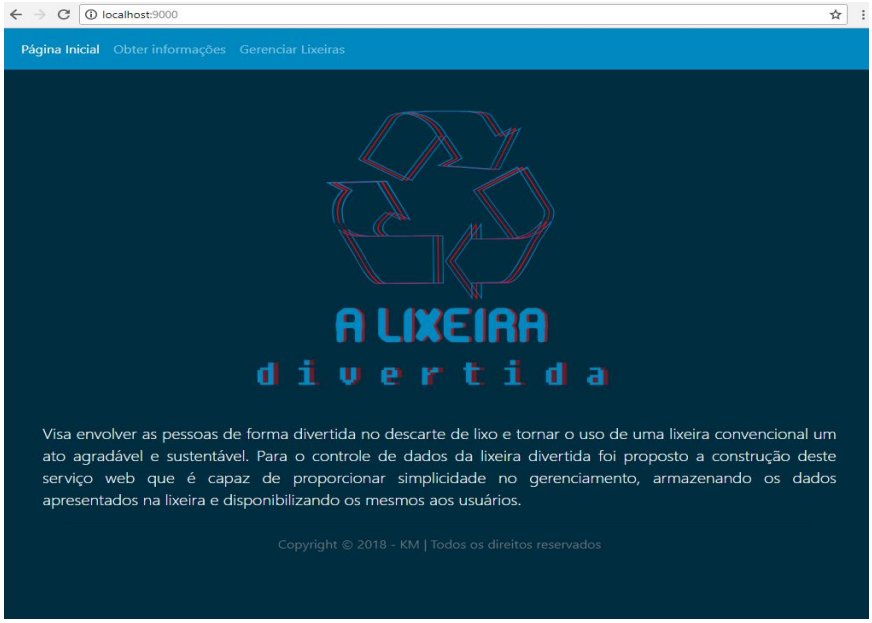


Figura 23 - Tela inicial. Fonte: Elaboração própria (2018).

5. TRABALHOS FUTUROS

Para futuras versões do sistema são sugeridos os seguintes aprimoramentos:

I. Implementação do hardware (lixeira) para encorajar e esclarecer a comunidade sobre a importância do descarte correto de lixo.

II. A aplicação de um sistema mobile de notificações a partir do qual o administrador possa receber em tempo real informações de seu interesse, como por exemplo, alertas de nível de lixo e alertas de nível de bateria.

III. Implementar os seguintes métodos para aprimoramento do sistema: Promover Usuário e Trocar Senha.

IV. Gráficos de análise das ocorrências.

6. CONCLUSÃO

Ao final deste trabalho foi desenvolvido um sistema web que propõem atingir os objetivos aqui estabelecidos. Foi desenvolvida uma ferramenta voltada para o controle de dados, com o intuito de prover uma plataforma alternativa para o usuário. O sistema foi devidamente projetado para mostrar o desenvolvimento de uma lixeira com qualidades específicas.

A fundamentação teórica gerou o entendimento do assunto que seria abordado, apresentando uma introdução ao padrão de arquitetura MVC (*Model-View-Controller*) aplicado no desenvolvimento de *softwares* e ao *Play Framework* que foram as principais ferramentas utilizadas neste trabalho.

Dessa forma, novos conhecimentos foram adquiridos ao longo do projeto, superando algumas dificuldades encontradas como a falta de conhecimento de parte da equipe sobre algumas plataformas escolhidas, e não possuir o *hardware* essencial para a elaboração completa do projeto.

Após analisar o sistema, é possível visualizar algumas vantagens, tais como: simplicidade no acesso, utilidade, fácil gerenciamento e também a visualização dos dados obtidos.

REFERÊNCIAS

ABRELPE. **PANORAMA DOS RESÍDUOS SÓLIDOS NO BRASIL 2015**. Disponível em: <http://www.abrelpe.org.br/Panorama/panorama2015.pdf>. Acesso em: 30 de maio de 2018.

BARBIERE, Lu. **O Que é Bootstrap e Para Que Serve?**. Disponível em: <https://www.ciawebsites.com.br/dicas-e-tutoriais/o-que-e-bootstrap/>. Acesso em: 18 de maio de 2018.

CRUZ JUNIOR, Gilson. **Burlando o círculo mágico: o esporte no bojo da gamificação**. Movimento, v. 20, n. 3, 2014.

DE CAMPOS, RAPHAEL BARRETO PALHARES et al. **ANÁLISE COMPARATIVA DE FRAMEWORKS DE PERSISTÊNCIA**. Links, p. 1, 2010.

DURELLI, Vinícius HS; VIANA, Matheus C.; PENTEADO, Rosângela AD. **Uma proposta de reuso de interface gráfica com o usuário baseada no padrão arquitetural MVC**. Simpósio Brasileiro de Sistemas de Informação, Rio de Janeiro–RJ, anais SBSI, p. 48-59, 2008.

Evolus. **Pencil Project: Top features of Pencil**. Disponível em: <http://pencil.evolus.vn/Features.html>. Acesso em: 17 de junho de 2018.

GUDWIN, Ricardo R. **Introdução à Linguagem UML**. Disponível em: <http://www.dca.fee.unicamp.br/~gudwin/ftp/ea976/Estruturais2010.pdf> . Acesso em: 20 de maio de 2018.

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

LEROUX, Nicolas; KAPER, Sietse de. **Play for Java: Covers Play 2**. Manning Publications Co., 2014.

LIMA, Davi de. **Modele softwares com Astah Community**. Disponível em: <http://www.techtudo.com.br/tudo-sobre/astah-commmunity.html>. Acesso em: 25 junho de 2018.

MERLIN, RODRIGO. **IMPLEMENTAÇÃO DE UM MÓDULO EM JAVA PARA GERAR LISTAGENS E RELATÓRIOS EM APLICAÇÕES DESKTOP E WEB**. 2009.

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

NETO, Moacyr Franco. **Tutorial da ferramenta de prototipação Pencil Project**. Disponível em:

<https://www.uaberta.unisul.br/sgc/downloadArquivoConteudo.processa?ead=8.822588450502341E111523271227482&arquivoId=41149&comunidadeId=44>. Acesso em: 28 de Abril de 2018.

NIEDERAUER, Juliano. **Integrando PHP 5 com MySQL**. São Paulo. Novatec Editora Ltda , 2005-2008.

Oracle Corporation. **Bem-vindo à comunidade NetBeans**. Disponível em: <https://netbeans.org/about/index.html>. Acesso em: 12 de junho de 2018.

Play. **Documentação**. Disponível em:<https://www.playframework.com/documentation/1.2/home>. Acesso em: 05 de maio de 2018.

SEBESTA, Robert W. **Conceitos de Linguagens de Programação-11**. Bookman Editora, 2018.

SHEETS, CSS–Cascading Style. Introdução à Tecnologia Web. 2010.

SILBERSCHATZ, Abraham; KORTH, Henry; SUNDARSHAN, S. **Sistema de banco de dados**. Elsevier Brasil, 2016.

SILVA, Maurício Samy. **Criando sites com HTML: sites de alta qualidade com HTML e CSS**. Novatec Editora, 2008.

TONÉIS, Cristiano N. **Os games na sala de aula: Games na educação ou a gamificação da educação**. Bookess Editora LTDA-ME, 2017.