Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Computação e Sistemas

Elaboração e construção de um protótipo mínimo viável para o Tingoram : Um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos

Jonathan Wagner Guimarães

TRABALHO DE CONCLUSÃO DE CURSO

ORIENTAÇÃO:
Igor Muzetti Pereira

Julho, 2018
João Monlevade–MG

Jonathan Wagner Guimarães

Elaboração e construção de um protótipo mínimo viável para o Tingoram : Um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos

Orientador: Igor Muzetti Pereira

Monografia apresentada ao curso de Engenharia de Computação do Instituto de Ciências Exatas e Aplicadas, da Universidade Federal de Ouro Preto, como requisito parcial para aprovação na Disciplina "Trabalho de Conclusão de Curso II".

Universidade Federal de Ouro Preto

João Monlevade

Julho de 2018

G963e Guimarães, Jonathan Wagner.
Elaboração e construção de um protótipo mínimo viável para o Tingoram [manuscrito]: um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos / Jonathan Wagner Guimarães. - 2018.

52f.: il.: color; grafs; tabs.

Orientador: Prof. MSc. Igor Muzetti Pereira.

Monografia (Graduação). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Departamento de Computação e Sistemas de Informação.

1. Sistemas de informação. 2. Mineração de dados (Computação). I. Pereira, Igor Muzetti . II. Universidade Federal de Ouro Preto. III. Titulo.

CDU: 004.62

Catalogação: ficha@sisbin.ufop.br

UNIVERSIDADE FEDERAL DE OURO PRETO
INSTITUTO DE CIÊNCIAS EXATAS E APLICADAS
COLEGIADO DO CURSO DE ENGENHARIA DE COMPUTAÇÃO

ANEXO IV – Folha de Aprovação

Curso de Engenharia de Computação

FOLHA DE APROVAÇÃO DA BANCA EXAMINADORA

Elaboração e construção de um protótipo mínimo viável para o Tingoram : um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos

Jonathan Wagner Guimarães

Monografia apresentada ao Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto como requisito parcial da disciplina CSI496 – Trabalho de Conclusão de Curso II do curso de Bacharelado em Engenharia de Computação e aprovada pela Banca Examinadora abaixo assinada:

Igor Muzetti Pereira
Prof. MSc. Igor Muzetti Pereira
Departamento de Computação e Sistemas – Universidade Federal de Ouro Preto
Professor Orientador

Prof. MSc. Vicente José Peixoto de Amorim
Departamento de Computação e Sistemas – Universidade Federal de Ouro Preto
Professor Convidado

Prof. Dr. George Henrique Godim da Fonseca
Departamento de Computação e Sistemas – Universidade Federal de Ouro Preto
Professor Convidado

João Monlevade, 12 de julho de 2018

Se todo fruto vem de uma árvore, a raiz da árvore que produziu como fruto este trabalho, começa na Dona Stella. Minha avó, guerreira que me ensinou que a vida é feita de dar a volta por cima, mantendo a ótica mais positiva e simples da vida que já vi alguém ter. Como ela mesmo diz, nosso olhar tem a capacidade de transformar lixo em luxo. Raiz forte essa. Outra parte dessa raiz é a outra guerreira da minha vida, minha mãe Daniela, responsável pela minha educação, sempre me apoiando e me motivando a seguir em frente com a vida. Nunca vou me esquecer quando eu falei que havia passado na faculdade mas não tinha dinheiro pra morar em outra cidade e ela me disse : " Vai arruma um emprego lá e eu arrumo outro aqui. Dividimos suas contas se precisar, fome você não passa." Das tantas coisas que minha mãe me ensinou, a mais forte foi o seu exemplo com atitudes, mostrando disposição e vontade pra encarar qualquer trabalho com disposição e empenho de uma forma que eu nunca vi em alguém. Sem palavras pela gratidão por vocês duas. Essa conquista não é minha, é de vocês. Obrigado pela criação, educação e apoio. Nada disso seria sem vocês. A vocês duas dedico esse trabalho.

Agradecimentos

Uma conquista com porte de uma graduação em uma Universidade Federal nunca é uma conquista meritocrática, mas antes uma conquista de uma família, de uma cultura, de um povo. Em um país de desigualdades absurdas, graduar-se é um privilégio. Parafraseando uma música de Bia Ferreira "Existe muita coisa que não te disseram na escola. Cota não é esmola." Primeiramente agradeço a minha família mais próxima (Stella, Daniela, Wagner, João , Davyd, Daniel, Matheus e Carlinhos) alicerce básico da construção da minha personalidade. Aos irmãos da República Vira-Lata pelo acolhimento e irmandade diária, sem os quais eu não chegaria aqui. Um agradecimento especial ao "Dino", que me aturou por quase 6 anos na mesma casa. À Banda Dunada, que me proporcionou momentos de muita música e amizade dentro e fora da universidade, me permitindo respirar música mesmo em um curso de engenharia. À "Japa", companheira de acampamentos e discussões filosóficas durante boa parte da graduação. Você também tem construção importante neste trabalho. Agradeço a todos os mestres e professores que passaram pela minha vida durante minha jornada na educação pública, desde a infância até o presente momento, existe um pedaço de cada um neste diploma. Aos bons professores pela inspiração e motivação a continuar aprendendo, e por evidenciar como nunca sabemos nada. Aos não tão bons assim, por me incentivar a aprender por outras fontes e não depender de uma pessoa para a construção do nosso próprio conhecimento.

Barriga vazia não pensa.

Resumo

Segundo o (FOOD; ORGANIZATION(FAO), 2017), 8,6 milhões de brasileiros passam fome. Segundo (IBGE, 2014) em 2013, 22,6% da população sofria algum tipo de inseguridade alimentar . Segundo a Embrapa, a quantia de desperdício de alimentos nesse país, caso servisse de alimento, supriria a demanda diária de 19 milhões de pessoas. Recentemente, um Projeto de Lei foi aprovado na Câmara dos Deputados incentivando a doação de alimentos através da possibilidade de redução de impostos, penalizando desperdiçadores com multa. Esse trabalho surge nesse contexto e propõe o planejamento e construção de Protótipo Mínimo Viável para o Tingoram, um sistema de mineração de dados web que tem por finalidade buscar dados de estabelecimentos que sejam potenciais doadores e/ou recebedores de alimentos e realizar contato eles, a fim de criação de vínculo de doação, utilizando para isso técnicas de mineração de dados, web scraping e recuperação da informação. Essa sugestão é feita tendo por base o georreferenciamento dos estabelecimentos. As informações foram buscadas do site Telelistas.net, um site de listas telefônicas, bem como da API do Google Maps (a Places API) e também do a do Facebook. Para realização de contato, o principal meio foi a plataforma web do WhatsApp, através de um script de envio semi automatizado. Como estudo de caso o sistema "percorreu" a cidade de Mariana, Minas Gerais, encontrando 583 estabelecimentos para realização de contato inicial, dos quais aproximadamente 14% possuíam número de celular. Os estabelecimentos que não houveram dados o suficiente para que o contato fosse realizado de forma automatizada, foram persistidos para futura ação de voluntários. Alguns estabelecimentos responderam ao contato via WhatsApp, mas até o término deste documento ainda não respondemos a nenhum deles.

Palavras-chaves: Mineração de dados Web. Web Scraping. Sistema de sugestão. Envio de mensagens semi-automáticas.

Abstract

According to (FOOD; ORGANIZATION(FAO), 2017), 8.6 million Brazilians go hungry. According to (IBGE, 2014) in 2013, 22.6 % of the population suffered some kind of food insecurity. According to Embrapa, the amount of food waste in that country, if it were food, would meet the daily demand of 19 million people. Recently, a Bill was passed in the Chamber of Deputies encouraging the donation of food through the possibility of reducing taxes, penalizing wasteful with a fine. This work arises in this context and proposes the planning and construction of Minimum Viable Prototype for Tingoram, a web data mining system whose purpose is to search data from establishments that are potential donors and / or food recipients and to make contact with them end of creation of donation link, using for this techniques of data mining, web scraping and information retrieval. This suggestion is made based on the georeferencing of the establishments. The information was searched from the Telelistas.net site, a phone book site, as well as the Maps API (the Places API) as well as the Facebook API. For contact, the main means was the WhatsApp web platform, through a semi automated submission script. As a case study, the system "crossed" the city of Mariana, Minas Gerais, finding 583 establishments for initial contact, of which approximately 14 % had a cell number. Establishments that did not have sufficient data to have the contact done in an automated way were persisted for future action by volunteers. Some establishments responded to the contact via WhatsApp, but until the end of this document we have yet to respond to any of them.

Keywords: Web Data Mining. Web Scraping. Suggestion system. Semi-automatic sending of messages.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

ACID - Atomicity, Consistency, Isolation and Durability

API - Application programming interface

AWS - Amazon Web Services

BASE - Basically Available, Soft state, Eventually consistent

BD -Banco de dados

CAP - Consistency, Availability and Partition tolerance

CRUD - Create, read, update e delete

CSS - Cascading Style Sheet

HTML - HyperText Markup Language

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

JSON - JavaScript Object Notation

MTV - Modelo Template View

MVC - Model-view-controller

MDW - Mineração de Dados Web

ONU - Organização das Nações Unidas

ORM - Object-relational mapping

PL - Projeto de Lei

RI - Recuperação da Informação

SQL - Structured Query Language

XML - eXtensible Markup Language

WS - Web Scraping

Sumário

1 Introdução

Um relatório oficial da Organização das Nações Unidas, (FOOD; ORGANIZATION(FAO), 2013) aponta com clareza a intensidade do problema relacionado à fome no mundo : nosso planeta desperdiça anualmente 1.3 bilhão de toneladas de alimentos . Outro relatório da Food and Agriculture Organization até 2014 haviam mais de 805 milhões de pessoas estão "cronicamente subalimentadas" (FAO,2014). Um terceiro relatório "Panorama de Insegurança Alimentar e Nutricional na América Latina" (FOOD; ORGANIZATION(FAO), 2017), da mesma instituição, afirma que cerca de 8,6 milhões de brasileiros passam fome. O Instituto Brasileiro de Geografia e Estatística em (IBGE, 2014) afirma que 52 milhões de pessoas, o equivalente a 22,6% da população sofrem algum tipo de inseguridade alimentar. Ainda segundo a ONU no site da FAO, o mundo desperdiçou em 2017 a quantidade de alimentos referente a nada menos que 1,4 bilhões de hectares de terra plantada. Esse número significa 28% da área de agricultura do mundo todo. Essa quantidade de desperdício possui extensão territorial maior que o Canadá, Estados Unidos, Brasil e China, por exemplo, perdendo em extensão somente para a Rússia. Isso significa que, territorialmente falando, o segundo maior país do mundo em área geográfica é o desperdício de alimentos. A Empresa Brasileira de Pesquisa Agropecuária (Embrapa) publicou em seu site (EMBRAPA, 2016), que a quantidade de alimentos desperdiçados no país seria suficiente para alimentar 19 milhões de pessoas diariamente. Como evidenciam os dados, a quantidade de 'alimentos desperdiçados, caso fosse distribuída ao invés de ir para o lixo, resolveria a questão da fome nesse país. Como podemos ver, o problema da fome no Brasil atualmente é estrutural, causado pela má distribuição de recursos e de renda. Será que se houvesse devida realocação dos alimentos desperdiçados, esse problema seria resolvido, ou pelo menos amenizado? Baseado nesses dados e tendo em vista a necessidade de realocação dos recursos alimentícios que atualmente são desperdiçados, esse trabalho se propõe a auxiliar na transformação da logística dos alimentos, construindo um Protótipo Mínimo Viável para o Tingoram, um sistema de sugestão de doação de alimentos baseado em georreferenciamento, a fim de evitar que esse recursos continuem indo para a lixeira. A proposta é minerar na Web informações sobre os locais que sejam potenciais doadores e/ou recebedores desses alimentos, bem como diversos contatos dos responsáveis por estes estabelecimentos, a fim de articulação da comunicação entre as partes, criando a possibilidade da criação de vínculo de doação entre elas com a finalidade de doação dos alimentos desperdiçados. O sistema é dividido em duas partes: o sensor, que busca informações na web e salva de forma estruturada e o atuador, que realiza o envio de mensagens de forma semi automatizada para os estabelecimentos. O próximo capítulo se incumbe dos referenciais teóricos e técnicos nos quais esse trabalho foi embasado, cons-

truindo embasamento necessário às áreas às quais podemos referi-lo como pertencente. Ele também cita o estado da arte em mineração web, bem como trabalhos semelhantes a este. No terceiro capítulo temos a explanação de como o sistema foi construído, como e quais tecnologias foram usadas para essa construção, passando brevemente pelas bibliotecas, interfaces e APIS que foram importante para o mesmo. É aqui que temos referências às decisões de projeto, questões analíticas e métodos de implementação e funcionamento do fluxo de informações do Tingoram, como um todo e em específico. O capítulo 4 traz a análise dos resultados desse trabalho. Nele teremos informações quanto ao número de estabelecimentos obtidos via cada meio de scraping, quantos foram unificados em uma só coleção de documentos, com quantos lugares conseguimos estabelecimento de contato e por qual meio de comunicação esse contato foi feito, bem como uma análise explanatória sobre o conteúdo dos dados obtidos e potencialidades do sistema.

2 Revisão bibliográfica

Como podemos observar empiricamente em nosso cotidiano e confirmar na literatura em (', 2011), o método dominante de busca de informações das pessoas da atualidade tem sido indiscutivelmente a Web. Segundo Liu, a busca na Web tem sua raiz na Recuperação da Informação (do inglês Information Retrieval), campo de estudo que se incumbe de auxiliar os usuários a encontrarem as informações que necessitam a partir de uma grande coleção de documentos de texto, genericamente falando.

2.1 Mineração de dados

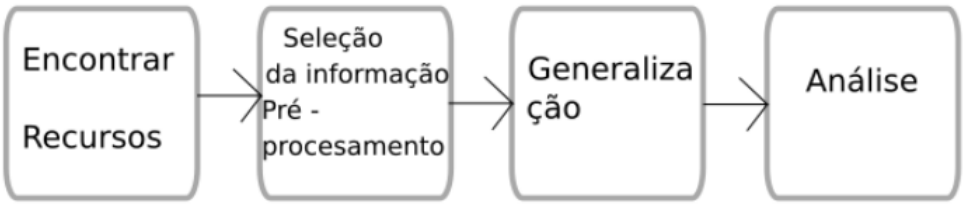
Mineração de dados é a busca de informações valiosas em grandes bancos de dados, é a exploração e análise desses dados por meios automáticos ou semiautomáticos , ou ainda o processo de extração ou mineração de conhecimento em grandes quantidades de dados, ou nas palavras do autor "um processo altamente cooperativo entre homens e máquinas que visa a exploração de grandes bancos de dados, com o objetivo de extrair conhecimentos através do reconhecimento de padrões e relacionamentos entre as variáveis" (CORTÊS SéRGIO DA COSTA. PORCARO, 2002) Uma das subáreas da mineração de dados é a mineração de dados na web, um caso particular de mineração onde o banco de dados, assim como na RI tradicional, não é estruturado. Esse trabalho pode ser definido como sendo desse tipo de mineração.

2.2 Mineração de dados Web

A Mineração de Dados Web (MDW) pode ser vista como parte do processo de uma recuperação de informação na (', 2011). Segundo (KOSALA RAYMOND BLOCKEEL, 2000), MDW é o uso de técnicas de mineração de dados para automaticamente descobrir e extrair informação da Web dos seus serviços . Eles afirmam que o trabalho de MDW pode ser decomposto em quatro sub tarefas, como podemos ver na Figura 1.São elas:

1. Encontrar recursos : obter páginas Web com possível conteúdo de interesse.
2. Seleção da informação e pré-processamento : automaticamente selecionar e pré-processar informações específicas desses websites.
3. Generalização : descobrir padrões tanto de um único site como de uma gama dos mesmos.
4. Análise : Validação e interpretação dos padrões descobertos.

Figura 1 – Sequência de sub-tarefas da Mineração de dados



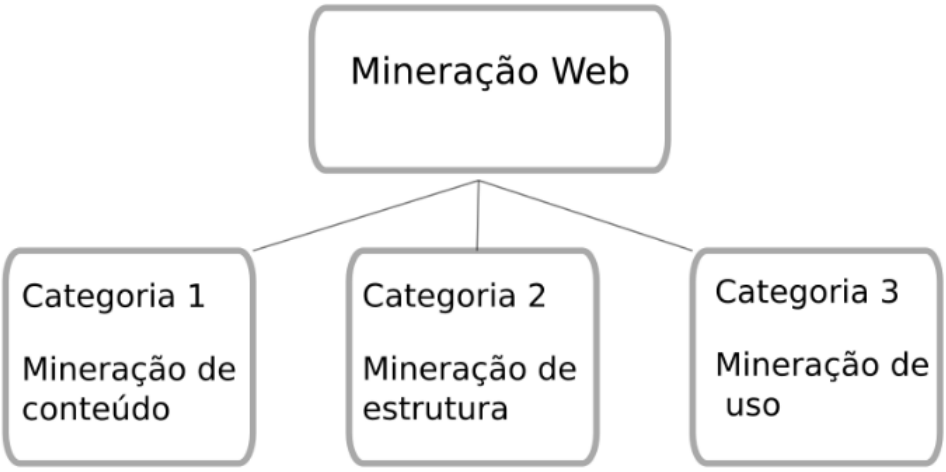
Fonte: Elaborado pelo autor

Enquanto as duas primeiras etapas se encarregam de buscar, selecionar e filtrar os dados brutos com objetivo de persisti-los no sistema, as duas últimas se incumbem de encontrar padrões nesses dados préviamente minerados e armazenados para extrair deles conhecimento útil. Como afirmam (BORGES J. E LEVENE, 1999) e também (BHARANIPIRYA V. PRASAD, 2011) , a MDW também pode ser dividida em três subáreas de interesse, de acordo com a parte da Web a ser minerada : mineração Web de conteúdo, de estrutura e de uso, como podemos evidenciar na Figura 2. Por mineração Web de conteúdo, entendemos a busca de informação entre os códigos htmls, ou seja, no conteúdo visualizado pelo usuário de um web site. A mineração de estrutura, é a busca de informações dentro dos próprios htmls, como classes e atributos. Já a mineração de dados Web de uso, se refere ao acesso dos usuário dado um determinado site ou aplicativo Web, no que se refere ao número de clicks, de acessos e de visualizações dessas informações e sites. Neste fizemos uso de MDW estrutural e de conteúdo.

2.3 Web Scraping

De acordo com (NEIL, 2016), de uma forma genérica um sistema de Web Scraping (WS) é qualquer software que extrai informações em sites, geralmente capaz de simular o comportamento da navegação humana na web, seja através do protocolo HTTP ou incorporando ações automatizadas em um navegador Web. De acordo com Milev , o WS pode ser visto como parte da Mineração de Dados e essa parte da Inteligência de Negócios. As razões pelas quais uma empresa adota o WS como ferramenta são, dentre outras, construir um motor de busca vertical específico, buscar produtos e preços para comparação, recrutamento de talentos , monitoramento de marca, verificação de anúncios (marketing),

Figura 2 – Categorias da Mineração de dados Web



Fonte: Elaborado pelo autor

recolha de anúncios imobiliários, para fins de pesquisa, coleta de grande volume de dados em sites de mídia, scraping para criar novos sites,geração de leads em sites de vendas .

Neil também afirma que o WS pode ser visto como uma subárea da mineração de dados web, concentrando-se na busca e transformação dos dados que estão disponíveis on-line mas de uma forma não estruturada, em dados estruturados, armazenado-os em seguida em algum banco de dados para posterior análise e utilização dos mesmos, vindo a ser uma alternativa para o antigo processo enfadonho de coletar manualmente grandes quantidades de dados. Segundo pesquisa feita pela (SOLUTIONS, 2017) mais de 19% dos principais usos do Web Scraping é para raspagem de contatos é feito por bots, extratores de dados automatizados. O propósito desse tipo de scraping é obter o endereço de e-mail do consumidor para fins de marketing em futuro empreendimentos e formação de mailing list. Ainda segundo este trabalho, as principais dificuldades em buscar e-mails são que e-mails importantes não são facilmente encontrados, a taxa de rejeição (número de e-mails que não puderam ser entregues) aumenta significativamente dado a quantidade de e-mails abandonados buscados e principalmente o fato de que bons sites geralmente ocultam a informação do e-mail de seus usuários, o que torna esse trabalho um pouco árduo. Ainda existe o fato de que, se o seu scraping capturou um e-mail com sucesso, provavelmente outros web scrapings podem também encontrar esse e-mail e enchê-lo de spam, o que faria com que sua mensagem também não fosse vista pelo usuário final, podendo então ser considerada como não recebida. Concluindo,um sistema de Web scraping pode ser enxergado como um sistema que automaticamente acessa um ou mais sites (geralmente via HTTP), faz parsing do conteúdo HTML , extrai seu conteúdo e constrói uma saída

formatada (GLEZ-PEÑA DANIEL, 2013).

2.4 Web Crawler

De acordo com (GUPTA,2010) , um *Web Crawler* é um programa recursivo com intuito de entrar nas subpáginas de uma dada página qualquer de maneira recursiva, com o intuito de realizar a busca de informações em toda extensão de um domínio (ou até que ele obtenha o dado desejado). Usaremos essa tecnologia na obtenção de dados do site Telelistas.net.

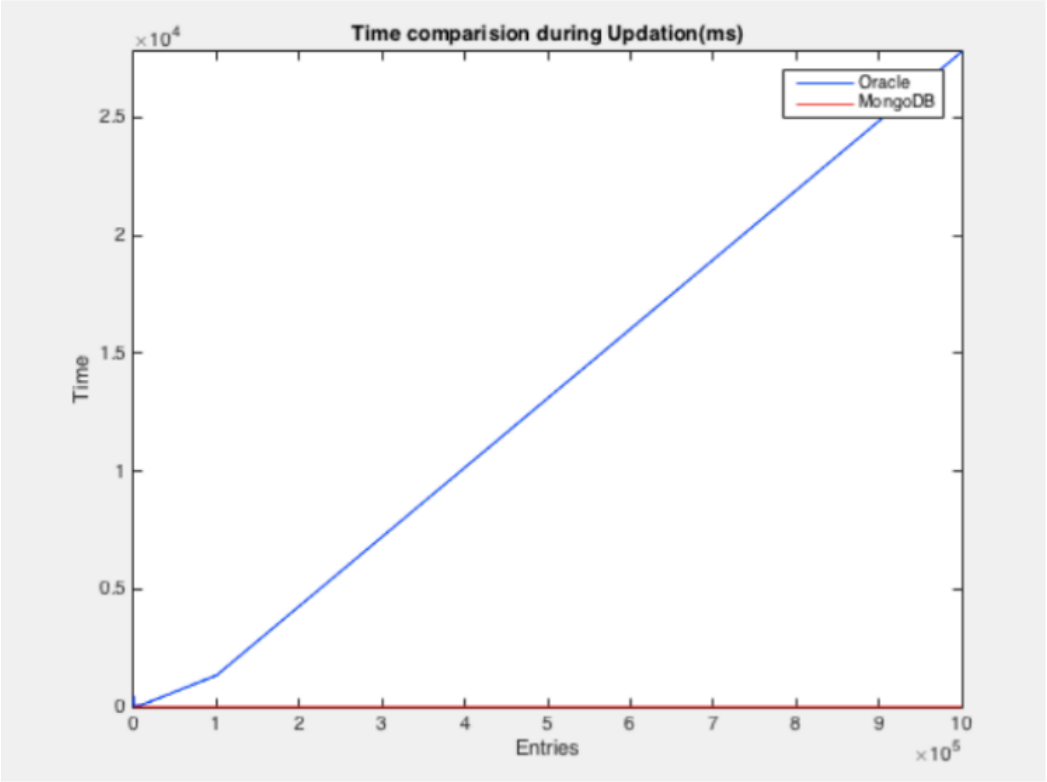
2.5 Mineração de dados no Facebook

Segundo (RUSSEL, 2014), a Graph API do Facebook "apresenta uma forma simples e consistente de enxergar o grafo social do Facebook, representando de maneira uniforme os objetos nesse grafo (por exemplo, pessoas, fotos, eventos e páginas) e a relação entre esses objetos (por exemplo, relacionamentos com amigos, conteúdo compartilhado e foto tags) ". O nosso interesse nesse trabalho é a busca de informações referentes ao objeto do tipo Página pública , obtendo informações úteis para o Tingoram sobre os contatos das instituições previamente capturadas com a finalidade de efetivar o estabelecimento de comunicação com elas.

2.6 Bancos de dados não relacionais

Segundo (SINGH, 2015), em 1998 Carlo Strozzi utilizou o termo ”NoSQL” a primeira vez. Podemos enxergar o termo como sendo “Not Only SQL”. Sua capacidade de escalonar horizontalmente associada a uma alta disponibilidade o trouxe à cena dos bancos de dados modernos. Ainda segundo ele, “os datastores NoSQL têm ampla aceitação em diversos setores variando de manufatura, petróleo e gás, energia, serviços bancários e de saúde”. Devido a sua arquitetura, NoSQLs possuem em geral, larga usabilidade em Big Data e em sistemas que interagem constantemente com páginas web. Uma informação importante sobre os bancos de dados NoSQL é que eles não garantem as propriedades ACID, apesar de alguns frameworks prometerem cumpri-las. Como informa (UPADHYAY AVIRAL .JAIN, 2017), o tempo de resposta de um banco NoSQL dado o crescimento das entradas (em proporções de Big Data) é absurdamente inferior ao de um banco de dados relacional. A Figura 3 mostra o banco de dados não relacional Mongo DB (utilizado nesse trabalho) em comparação com um banco de dados tradicional (relacional), o Oracle. Como podemos ver eles são úteis na recuperação de documentos. Segundo (SINGH, 2015), isso se deve em grande parte ao processo de Map Reduce.

Figura 3 – Gráfico de comparação entre update no Oracle (relacional) e no Mongo DB (não-relacional).



Fonte: (UPADHYAY AVIRAL .JAIN, 2017)

(UPADHYAY AVIRAL .JAIN, 2017), são MongoDB (utilizado neste trabalho), o SimpleDB e o CouchDB. Os datastores de armazenamentos de dados de registros extensíveis possuem linhas e colunas que podem se dividir em vários nós. HBase, HyperTable e PNUTS são alguns exemplos, todos motivados pelo modelo BigTable do Google.

2.7 Bots no Messenger Facebook

Segundo a documentação oficial do Facebook muitos tipos de conteúdos não estruturados podem ser enviados utilizando o Messenger. A própria empresa fornece uma série de mensagens modelos prontas para uso. O nosso interesse nesse trabalho é o de envio de mensagens de texto para as páginas de possíveis doadores e/ou recebedores de alimentos.

2.8 Trabalhos relacionados

Em (NEIL, 2016) temos a análise de um software de web scraping. O sistema utilizado como modelo base para esta pesquisa foi o Data Toolbar®. No artigo, o autor usou essa ferramenta para extrair dados do domínio Amazon.com . Esse sistema é instalado no browser. O usuário entra no site desejado, preenche um formulário de busca indicando os termos a serem "scrapeados" e o sistema retorna uma tabela (que pode ser convertida em template html genérico) para visualização dos dados, preenchido com as respostas do scraping do sistema. No exemplo do artigo Neil configurou a busca para livros do site Amazon. Segundo o mesmo, o Data Toolbar funciona para a grande maioria dos casos mas peca na obtenção do detalhamento de alguns dados, como o nome dos autores dos livros buscados. É possível ainda, escolher vários formatos desejados para salvar os dados obtidos. (ELOISA URRU, 2013) também fazem um estudo de caso de um sistema que, segundo eles, é o primeiro a usar soluções de web scraping para fazer Publicidade Web (área que tem como principal missão sugerir produtos e serviços , baseada em IR, machine learning , otimização e macroeconomia). Eles abordam a Publicidade Web como sendo uma tarefa de filtragem de informações e realizam filtragem colaborativa para fazer scraping dos banners de anúncios das páginas (utilizando Python e Beaultiful Soup, que serão introduzidos na Seção 3). O sistema entra nos primeiros 10 links contidos no site que não redirecionem para o próprio domínio do site, captura os banners de cada um deles, embaralha-os e randomicamente seleciona três anúncios dessa coleção para sugerir para a página inicial. Os autores esperam futuramente aplicar este algoritmo para sugestão de anúncios baseados em gostos dos usuários de redes sociais, capturados via web scraping. (KUMAR, 2014) não desenvolveram nenhum sistema de WS mas analisaram 5 ferramentas existentes no mercado, a saber: Automation Anywhere, Web Info Extractor, Web Content Extractor, Screen Scraper e Mozenda. Em uma tabela comparativa, mostraram que todas as ferramentas fazem extração de dados web, tanto estruturados quanto não estruturados, e que somente a ferramenta Automation Anywhere possui persistência e suporte ao uso de inteligência artificial. As ferramentas analisadas são voltadas para a raspagem de conteúdos de textos, imagens e mídias, algumas delas possibilitando exportar views em forma de relatórios, cada uma escrita em uma linguagem.

2.9 A doação de alimentos perante a legislação brasileira

Segundo (DEPUTADOS, 2010)(site oficial da Câmara dos Deputados), a Comissão de Seguridade Social e Família (CSSF), da Câmara dos Deputados, analisou e aprovou por unanimidade no dia 06 de junho de 2018, o Projeto de Lei 5958/2013, que regulamenta a doação de alimentos industrializados, embalados ou in natura que tenham perdido a condição de comercialização (amassados, com aparência pouco atraente, por exemplo),

mas estejam dentro do prazo de validade. De acordo com esse Projeto de Lei, "fica proibido o descarte de alimentos que estejam dentro do prazo de validade para venda e próprios para consumo, sujeitando o infrator a multa a ser definida em regulamento" . A esse projeto de lei foram apensados dezenas de Projetos de Lei com o mesmo intuito propostos paralelamente, todos aceitos em votação unânime.

Figura 4 – Artigos do PL 5958/13.

Art. 13. O art. 13 da Lei nº 9.249, de 26 de dezembro de 1995, passa a vigorar com a seguinte redação:

"Art. 13. .....

.....

§ 3º Nas doações de alimentos com antecedência mínima de 5 (cinco) dias do vencimento do prazo de validade previsto na embalagem, o limite da dedução prevista no inciso III do § 2º será de 5% (cinco por cento)." (NR)

Art. 14. A Lei nº 9.605, de 12 de fevereiro de 1998, passa a vigorar acrescida do seguinte art. 61-A:

"Art. 61-A. Descartar alimentos processados ou industrializados, embalados ou não, dentro do prazo de validade para venda, alimentos in natura ainda próprios para consumo, segundo as normas sanitárias vigentes, ou em desacordo com as disposições da Lei nº 12.305, de 2 de agosto de 2010, que institui a Política Nacional de Resíduos Sólidos.

Pena – multa.

Parágrafo único. Os critérios técnicos de avaliação do cumprimento do disposto no caput serão definidos em regulamento."

Fonte: PL 5958/13

Dentre suas deliberações, as principais são A penalização com multa ao desperdício de alimentos e do outro lado, a redução de 5% do imposto de renda mediante efetivação de doação. Podemos ver os principais artigos da PL na Figura 4.

3 Metodologia

O Tingoram se propõe a ser um sistema de mineração de dados Web e de posterior estabelecimento de contato semi-automático, utilizando pra isso os dado obtidos previamente. Sua finalidade é buscar na rede as informações como nome, endereço, coordenada geográfica e principalmente informações de contatos (telefone, celular, e-mail, id da página do Facebook) dos estabelecimentos que sejam potenciais doadores e/ou recebedores de alimentos, tentando após obtenção desses, realizar o contato com os estabelecimentos com a finalidade de criação de vínculo entre as partes, visando a doação de alimentos. Desde que o sistema possua o georreferenciamento dos locais, a sugestão de doação preza o estabelecimento de vínculo entre os lugares geograficamente mais próximos. Para tal, o sistema foi projetado separado em 2 aplicações dentro do mesmo projeto,a saber, as aplicações sensor e atuador . A primeira é responsável pela parte de busca de contatos dos potenciais doadores e recebedores de alimentos na Internet, utilizando para isso uma combinação ferramental de APIs de serviços Web com algoritmos de Web Scraping implementados pelo autor do sistema, utilizando como bibliotecas específicas para tal, como a Beautifulsoup. Já a aplicação atuador é responsável pelo estabelecimento de contato com os estabelecimentos e criação de vínculo entre as partes envolvidas na doação. O atuador é considerado semi-automático pelo fato de que nem sempre o sensor conseguirá informações de contatos que permitem meios automáticos de contato. Por exemplo, nos casos em que só há telefone fixo e/ou endereço do estabelecimento ele é marcado como estabelecimento a se enviado para a ação de voluntários. Esses voluntários podem acessar o sistema e realizar cadastro prévio, ajudando o Tingoram através de ligações e/ou visitas aos estabelecimentos armazenados em banco, a fim de criação de vínculo de doação.

3.1 Ambiente

Para controle de desenvolvimento e engenharia de software do sistema em questão foi utilizada a ferramenta Trello, implementando colunas de atividades de forma a simular o sistema de controle de tarefas encontrado no método Scrum. aborda a utilização dessa ferramenta de alinhada com a filosofia Scrum . O Tingoram foi escrito tendo como base a linguagem Python, dada sua robustez e simplicidade. Um dos principais motivos de escolha da mesma foi a decisão de utilizar o framework Django de desenvolvimento Web rápido escrito em Python, com a filosofia de programação intitulada “não repita a si mesmo”. A biblioteca requests foi utilizada como a responsável pela meio de comunicação HTTP (get e post) entre a aplicação e os links que a mesma possuía necessidade de acesso/ comunicação. A biblioteca djongo (detalhada na seção 3.1.2) foi utilizada como ponte entre o Django

(Python Web Framework) e o Mongo DB (CHODOROW, 2013)

3.1.1 Django

Django é um framework para desenvolvimento de aplicações Web criado em 2003 por Adrian Holovaty e Simon Willison, escrito em Python, tornando-se open source em 2005. Ele segue um padrão de programação que segundo (HOLOVATY MOSS, 2015) é considerado um padrão de projeto do tipo MVC (modelo, visão e controle), batizado por eles como MTV (modelo,template e view).

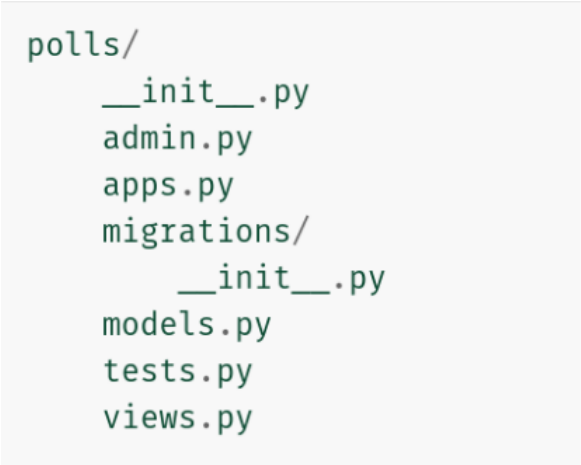
3.1.1.1 Modelo Visão e Controle (MVC) versus Modelo Template e View (MTV)

Como afirmam (CRIZEL VINICIUS DA S., 2016), o padrão MVC segue o fluxo de comunicação semelhantes, sendo que cada parte é modularizada e independente. Por Modelo entendemos a camada de definição, manipulação e validação dos dados , responsável pelas operações CRUD e comunicação com o banco de dados. O Controlador, como próprio nome já diz, é responsável por controlar esses dados intermediados pelo Modelo. É nele que é construída a lógica de negócio da aplicação, os processamentos, tratamentos e modificações dos dados vindos do modelo, bem como a renderização das telas, fazendo o intermédio entre o Modelo e a Visão, definindo suas ações de acordo com os códigos que tratam as requisições feitas pelos usuaŕios. A Visão é o módulo responsável pela exibição dos dados armazenados e processados para o usuário da aplicação . Isso significa que o processamento é totalmente transparente para essa camada. Ela pode ser interpretada como o frontend do sistema, geralmente composto de códigos HTML, CSS e Javascript com suas inúmeras ferramentas de visualização de dados e interação com o usuário. Segundo a documentação oficial do framework Django , o MTV é um padrão similar ao MVC, porém com uma filosofia que descreve de forma mais fiel o funcionamento do Django. No MTV, o Modelo possui exatamente a mesma definição que seu “pai” MVC. Já a Visão pode ser “interpretada” como Template e o Controlador “interpretado” como sendo Visão. As diferenças não são somente uma questão de nomenclatura, por isso o uso do termo acima, “interpretado", entre aspas. A principal abstração que o Django fornece é exatamente a implementação de parte do Controlador para o programador . Na interpretação do Django do MVC, a Visão é responsável pelos dados que são apresentados ao usuário mas não em relação a como eles são exibidos e sim a em relação a quais serão esses dados. O Template é a parte responsável exatamente pela exibição desses dados. De acordo com os autores da documentação Oficial Django (DJANGO, 2018), o mais importante é compreender conceitualmente cada camada do framework.

3.1.1.2 Estrutura do Framework Django

Um sistema web é interpretado no Django como sendo um projeto composto de aplicações.A estrutura de arquivos mostrada na Figura 5 é o padrão implementado pelo Django, contendo um arquivo intitulado manage.py, que segundo os autores do framework, é "um utilitário de linha de comando para configurações gerais que lhe permite interagir com o projeto Django de várias maneiras" . Em settings.py definimos todas as configurações gerais do projeto, tais como aplicações instaladas, bancos de dados utilizados, administrador(es), configurações de segurança, entre outros.

Figura 5 – Estrutura de uma aplicação Django.



Fonte: (DJANGO, 2018)

Em models.py escrevemos nossas definições de modelos de dados. Escrevemos nossas "visões", responsáveis pelo tratamento e processamento dos dados em views.py. As classes que forem definidas em models.py e posteriormente declaradas em admin.py terão uma página dedicada à manipulação de seus dados criada automaticamente pelo framework no endereço admin/nome_classe.

3.1.2 Mongo DB

A primeira coisa que devemos saber sobre o MongoDB é que ele é um banco de dados orientado a documentos e não um banco de dados relacional tradicional. Segundo afirma (CHODOROW, 2013) , a principal razão para tal é torná-lo facilmente escalável, além de tornar sua modelagem mais flexível, substituindo as rígidas "colunas" de tabelas por documentos, tendo suporte a aninhamento de documentos (*embedded documents*) . No Mongo não há nenhum esquema fixo pré-definido, podendo o documento que representa uma mesma abstração de dados, assumir várias formas diferentes, tornando o banco sensível ao contexto. Podemos enxergar um documento do Mongo como o análogo de uma linha

Tabela 1 – Comparação entre bancos de dados relacionais e o Mongo

Bancos de Dados Relacionais Tradicionais (MySql, PostgreSql,etc)	MongoDB (Não relacional)
Tabelas	Coleções
Registros/Colunas	Documentos/Objetos
Consultas retornam gravações	Queries retornam cursores

(registro) em um banco de dados relacional e uma coleção como o análogo a uma tabela, como é mostrado na Tabela 1.

3.1.3 Djongo

O Djongo é a ponte entre o Django e o Mongo DB. Ele cria toda camada de abstração da comunicação entre as partes, permitindo ao programador criar e acessar os documentos do banco de maneira simples e eficiente, exatamente da mesma maneira que o ORM do Django. Um ORM permite a representação dos bancos de dados/ coleções em classes e registros/documentos em instâncias de suas respectivas classes.

3.1.4 Selenium

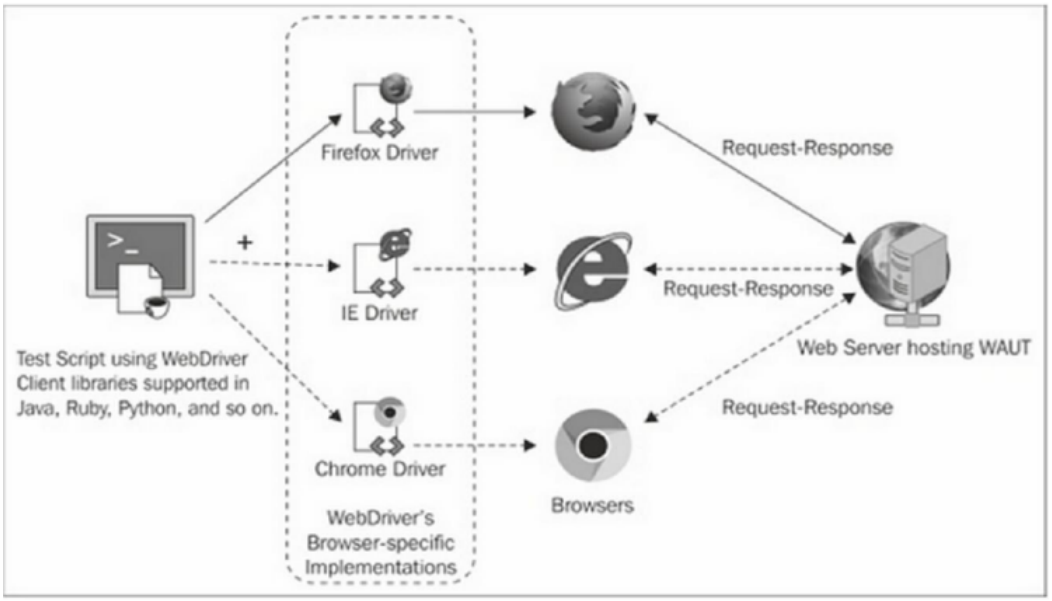
Segundo (RAHMAN,2014), o Selenium "é uma estrutura de teste de software para aplicativos da Web", possuindo suporte a diversas linguagens, cobrindo funcionamento conjunto a quase todos os navegadores atuais. Ele é composto por várias ferramentas de teste de software web . Nesse trabalho temos interesse específico no uso do Selenium Webdriver, um controlador de navegadores que possibilita ao programador implementar casos reais de uso de qualquer aplicação web sob o ponto de vista do usuário final . O Selenium é um software de código aberto sob licença Apache 2.0.

3.1.4.1 Selenium Web Driver

Segundo (AVASSARALA,2014) ,O Webdriver é uma "ferramenta de automação de testes de navegador para aplicativos da Web que é usada para compilar testes de ponta a ponta". Essa ferramenta utiliza a aplicação em questão exatamente da mesma maneira como esperado de um usuário final, ou seja por meio de um navegador , como podemos ver na Figura 6.

A função do programador que utiliza o Webdriver é implementar algoritmos que possibilitem o controle da navegação no sistema de interesse, de maneira automática. Neste trabalho seu uso se refere ao envio de mensagens através do WhatsApp Web, uma plataforma alternativa de acesso à rede social, disponibilizada pela própria empresa.

Figura 6 – Estrutura e funcionamento do Selenium Web Driver.



Fonte: (AVASSARALA,2014)

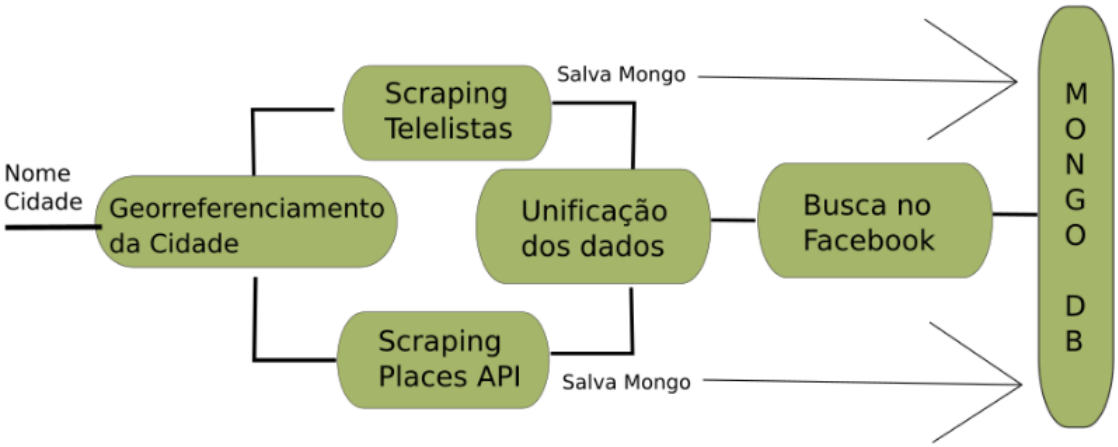
3.1.5 Homologação e Produção

Para controle de versões e execução de testes, foram mantidas duas versões distintas do projeto Tingoram, uma localhost , simulando um ambiente de homologação e a outra em nuvem, na AWS, simulando um ambiente de produção. Na homologação local foi utilizada uma ferramenta Python chamada virtualenv , que permite isolar um ambiente em determinada pasta , de forma que todos os pacotes e versões de bibliotecas instaladas nesse ambiente permanecem nele, não alterando o sistema ao qual está contido. Para ativá-lo, basta executar o source "/bin/activate", dentro da pasta do ambiente virtual criada, e essa pasta estará isolada . Todas as views, antes de serem acopladas no Django foram escritas antes como scripts Python e testadas nesse ambiente. Após funcionamento correto desses scripts os códigos são transcrição para views e essas incorporadas ao servidor local do Django, testadas, homologadas e então copiadas para a produção. O ambiente de produção deste trabalho foi uma máquina na AWS, com IP de número 18.220.228.137. As conexões com a mesma foram feitas por intermédio de uma chave criptográfica ".pem", utilizando a ferramenta de conexões remotas ssh do Linux. Para transferência de arquivos foi utilizado o programa scp , também do Linux e utiliza a .pem para autenticação da conexão com a máquina'.

3.2 Sensor

O Sensor do Tingoram consiste em uma aplicação criada dentro do projeto Django, responsável pela busca e mineração de dados dos estabelecimentos na Web e pela persistência desses dados em disco. Primeiramente são feitas duas buscas independentes: a busca no serviço de mapas do Google Places utilizando-se a API fornecida pela empresa e a busca feita pelo "Web scraper" construído neste trabalho em um site de listas telefônicas. Como utilizamos o Mongo como banco de dados , e por consequência o JSON como formato padrão de persistência, aproveitamos a semelhança desses com os dicionários Python para usá-los como padrão de saída de todas as funções de sensoriamento da web, o que nos permite persistir os estágios intermediários da informação, ou seja, temos coleções de documentos no banco para ambas as buscas, respectivamente nomeadas Lugar_Places e Lugar_Telelist. O nome "sensor" foi escolhido dada a função que essa parte do sistema terá em versões posteriores, "farejando" em um loop uma lista de cidades, "sensoriando-as" na rede.

Figura 7 – Estrutura da aplicação Sensor, do Tingoram



Fonte: Elaborado pelo autor

Após essas buscas o sistema faz a unificação entre os documentos salvos, responsável por detectar quais informações das buscas paralelas são referentes ao mesmo estabelecimento, salvando todos os dados obtidos até então em uma nova camada de dados , criando a coleção Lugar_Unificado. Essa coleção foi criada utilizando as propriedades ORM do Django, que permite tratar o banco de dados como se fosse um objeto. Isso significa que essa coleção possui uma "herança" em forma de documentos (Embedded Documents). Para uma melhor compreensão, vejamos sua estrutura na Figura 8. Essa estrutura hierárquica foi escolhida apenas para representar as camadas de dados , uma abstração referente à

sequência que o fluxo de execução do sensor, de acordo com quais dados que o mesmo gera, como podemos ver na Figura 7.

Figura 8 – Coleções Lugar_Unificado e Lugar_Unificado_Completo

```
class Lugar_Unificado(models.Model):
    lugar_telelist = models.EmbeddedModelField(model_container=Lugar_Telelist)
    lugar_places = models.EmbeddedModelField(model_container=Lugar_Places)


class Lugar_Unificado_Completo(models.Model):
    lugar_unificado = models.EmbeddedModelField(model_container=Lugar_Unificado)
    lugar_facebook = models.EmbeddedModelField(model_container=Lugar_Facebook)
```

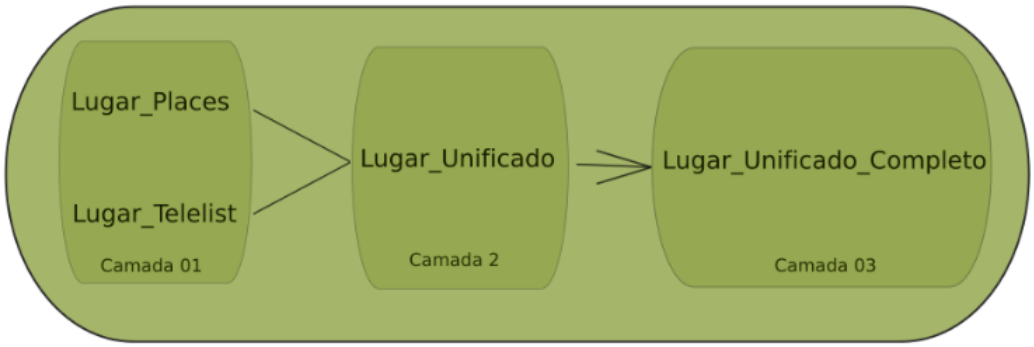
Fonte: Elaborado pelo autor

Após unificação, o fluxo segue, buscando o nome do estabelecimento de cada documento persistido e unificado e realizando uma consulta no Facebook pela página pertencente à mesma. Essa página, além de nos fornecer informações úteis sobre o estabelecimento, servirá posteriormente para estabelecimento de contato automatizado via bot do Facebook Messenger. O filtro de verificação e confirmação da identidade do estabelecimento é tanto o nome quanto a cidade do mesmo. O sensor termina sua ação fazendo uma requisição das informações públicas disponíveis no Facebook sobre páginas encontradas referentes aos estabelecimentos previamente obtidos, utilizando para tal a Graph API do Facebook . Finalmente o sistema persiste esses documentos agregados por estabelecimentos, na tabela Lugar_Unificado_Completo, na terceira e última camada de dados do sensor. As camadas de coleções de dados (documentos) do Tingoram são evidenciadas na Figura 9. Como dito anteriormente, essas camadas são puramente abstratas (no sentido de que não são armazenados dados redundantes mas sim adicionada uma relação entre os documentos). Essas camadas refletem o grau de unificação dos dados obtidos pelo sistema, bem como a etapa do sensor que a gerou.

3.2.1 Google Places

Como podemos ver na documentação oficial do Google Places, a Places Web Service API nos permite consultar informações de locais em uma variedade de categorias, com tipos de estabelecimentos diferentes, pontos de interesse proeminentes e localizações geográficas. É possível pesquisar locais por proximidade ou com uma string de texto. Existem dois tipos de chamadas. Uma Place Search retorna uma lista de locais com informações resumidas sobre cada local. A busca retorna dados semelhantes aos oriundos das pesquisas utilizando o Google Maps, mostradas na Figura 10 , porém com resposta em JSON ou XML. O segundo tipo de chamadas à API é a que utiliza requisições do tipo Place Details , que

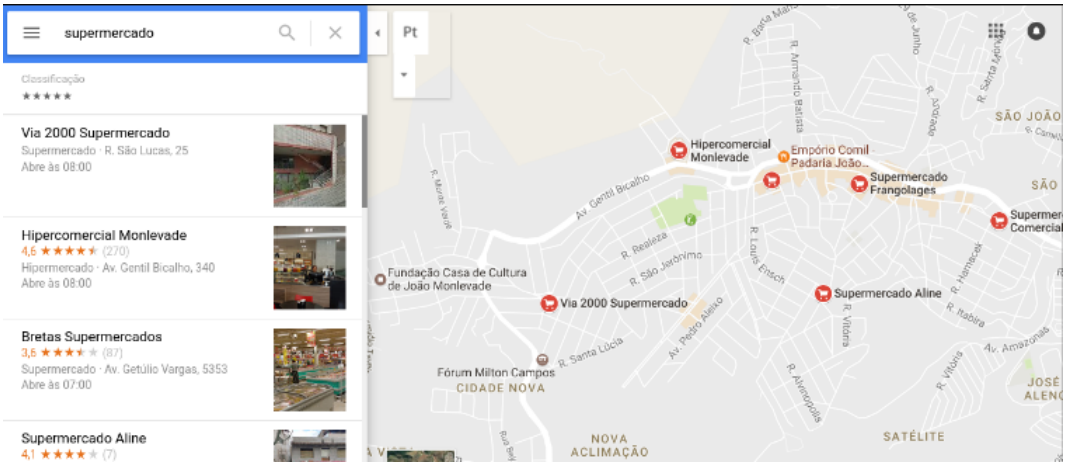
Figura 9 – Camadas abstratas de persistência (coleções de documentos no Mongo DB).



Fonte: Elaborado pelo autor

recebe como parâmetro obrigatório o place_id de um local específico e retorna informações adicionais sobre esse local.

Figura 10 – Exemplo de consulta no site https://maps.google.com



Fonte: https://maps.google.com.

Uma “view” do Tingoram é responsável pela implementação das buscas de dados utilizando a Places API chama primeiramente uma requisição do tipo Place Search chamada Nearby Search. Ela recebe como parâmetro de entrada o nome da cidade e estado onde deseja realizar a consulta, chamando um terminal específico do Google maps para realizar consulta de geolocalização dessa cidade. Um dos parâmetros da busca à API do Google é a georreferência do local de pesquisa. Para capturar as informações , utilizamos esse terminal descrito, que atende a requisições que possuem o formato

Tabela 2 – Parâmetros de busca na Places API para possíveis doadores e recebedores de alimentos

Parâmetros de busca a possíveis recebedores de alimentos pela Places API	Parâmetros de busca a possíveis doadores de alimentos pela Places API
	bakery
	cafe
	restaurant
church	food
school	grocery_or_supermarket
hospital	lodging
lodging	meal_delivery
	meal_takeaway
	colloquial_area
	geocode

https://maps.googleapis.com/maps/api/place/nearbysearch/[output]?[parameters], onde [output] pode ser uma das siglas , json ou xml, e [parameters] é a sequência de parâmetros passados para a API, como a chave de acesso à API, a latitude e longitude do espaço de busca desejado, o tipo de local buscado e o raio de alcance da consulta. Para fins de pesquisa, definimos um raio de 20 quilômetros da latitude e longitude das ruas centrais da cidade, também como parâmetro do terminal . A Tabela abaixo mostra quais foram os tipos de lugares pesquisados para cada categoria de estabelecimento do sistema: doadores e recebedores. A tabela mostra nomes em inglês pois são os nomes aceitos pela API segundo documentação oficial. Uma "view" do Tingoram é responsável pela implementação das buscas de dados utilizando a Places API chama primeiramente uma requisição do tipo Place Search chamada Nearby Search. Ela recebe como parâmetro de entrada o nome da cidade e estado onde deseja realizar a consulta, chamando um terminal específico do Google maps para realizar consulta de geolocalização dessa cidade. Um dos parâmetros da busca à API do Google é a georreferência do local de pesquisa. Para capturar as informações , utilizamos esse terminal descrito, que atende a requisições que possuem o formato https://maps.googleapis.com/maps/api/place/nearbysearch/[output]?[parameters], onde [output] pode ser uma das siglas , json ou xml, e [parameters] é a sequência de parâmetros passados para a API, como a chave de acesso à API, a latitude e longitude do espaço de busca desejado, o tipo de local buscado e o raio de alcance da consulta. Para fins de pesquisa, definimos um raio de 20 quilômetros da latitude e longitude das ruas centrais da cidade, também como parâmetro do terminal .

Como resultado dessa chamada à API da forma exibida na Tabela 2. Temos a busca de uma série informações sobre diversos estabelecimentos que podem ser poten-

ciais doadores e/ou desperdiçadores de alimentos bem como seus respectivos place_ids . Esses ids são úteis pois nos permite fazer uma segunda chama, utilizando agora o webservice do Google chamado de Place Details, também pertencente à mesma API. Utilizamos em sequência cada place_id obtido como parâmetro de entrada da função de obtenção de detalhes dos lugares, via requisição no terminal Place Details, com endereço https://maps.googleapis.com/maps/api/place/details/[output]?[parameters] , onde [output] é semelhante à chamada anterior e o principal parâmetro (além da chave para acesso) é o id do local que deseja obter informações detalhadas. Antes do término da execução da view responsável pela parte do sensor que realiza a busca no Google Places, uma função contendo os comandos do Djongo com Django para persistência dessas informações no Mongo DB é chamada.

3.2.2 Scraping Telelistas

De acordo com uma das definições anteriores, um sistema de Web Scraping é aquele que busca dados na rede de forma automatizada ou semi automatizada , frequentemente simulando a interação de um usuário comum, ou seja utilizando-se de HTMLs, tanto em termos de informações contidas nos mesmos quanto em formulários, fazendo uso conjunto do protocolo HTTP. A este trabalho incumbiu-se também a tarefa de implementação de um algoritmo de Web Scraping no site mais conhecido de listas telefônicas, o Telelistas. Em relação à legalidade de obtenção desses dados, além do fato de serem dados públicos, com conteúdo disponibilizado na Web podendo ser acessado por qualquer pessoa que detenha o link de acesso, o arquivo robots.txt do site não proíbe a obtenção de forma automatizada dos dados que o Tingoram usa. O primeiro passo realizado foi o estudo do site de listas telefônicas, mais precisamente os padrões de comportamentos de seus redirecionamentos e de suas urls, principalmente as geradas após envio de preenchimento do formulário da página inicial do site, conforme mostrado na Figura 11.

Figura 11 – Página inicial do site TeleListas.net

Fonte: Telelistas.net

Tabela 3 – Padrões de geração de URLs descobertos com base em análise de comportamento do site Telelistas.net

Padrões de busca de possíveis doadores no Telelistas	Padrões de busca de possíveis recebedores no Telelistas
/pizzarias /entregas+rapidas /buffets /armazens+emporios+e+mercearias /centrais+de+abastecimento /supermercados+e+hipermercados /restaurantes	/creches

Foi observado pelo autor que, após preenchimento do formulário com do campo UF preenchido com MG e Cidade preenchido com a Cidade Mariana, a URL da nova página tornava-se https://www.telelistas.net/mg/mariana. Quando o teste era feito com o campo Cidade preenchido com o valor "João Monlevade" a URL de resposta possuía formato https://www.telelistas.net/mg/joao+monlevade , o que confirmou o padrão de comportamento de mapeamento de URLs do site, nos levando a chegar na generalização

[ENRERECO_TELELSITAS] / [SIGLA_ESTADO] / [CIDADE]

onde no campo cidade, os espaços sempre são substituídos pelo caractere '+ '. Essa generalização nos levou à possibilidade de criação de links de acesso ao site, possibilitando o acesso automatizado via crawler.

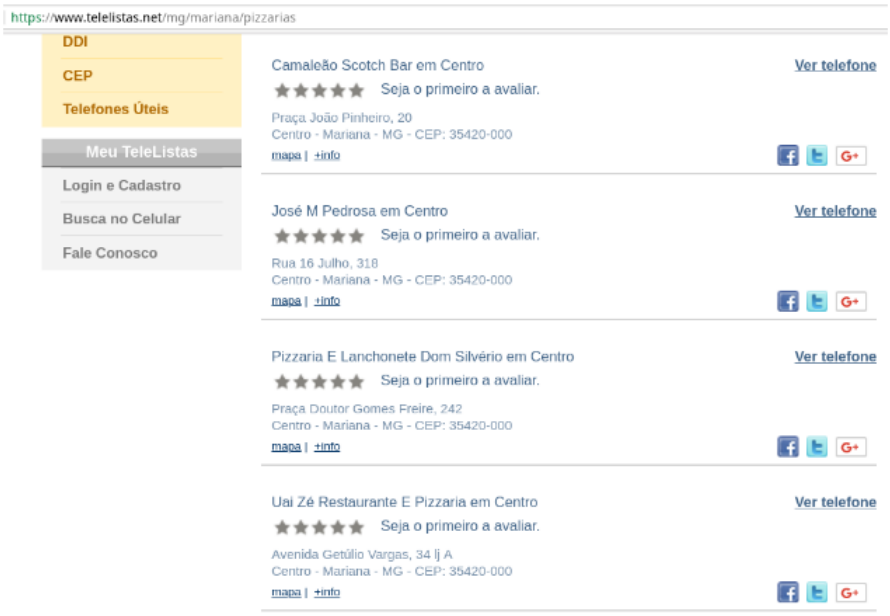
Semelhantemente a esse estudo de resposta do formulário inicial preenchendo-o com nome das cidades a fim de descoberta de padrão, foi testado o preenchimento desse mesmo formulário porém de maneira completa, por inúmeras vezes, variando os campos Nome e Palavras-chave, com intuito de estudar as URLs de resposta. Essa análise permitiu-nos chegar à generalização do padrão de comportamento de redirecionamento de rotas do site Telelistas.net. A view que realiza o web scraping nesse site utiliza um crawler construído com base nessas análises, especificamente para esse site. Ela recebe como entrada uma string com o padrão 'nome_cidade,estado', vinda de um formulário preenchido pelo autor para dispará-la. Busca então, baseado nesses padrões de comportamento das URLs do site Telelistas, doadores e recebedores. Vemos no Quadro 3, o conjunto de padrões de sufixos dos links que foram descobertos por inspeção, separados em doadores e recebedores. Cada um desses padrões foi somado à URL pré-fixada ao mesmo. A URL final possui seguinte estrutura : http://telelistas.net/[estado]/[cidade]/[sufixo _quadro_3].

Ao entrar em cada um desses links, gerados para uma dada cidade, o scraping

depara com uma resposta HTML contendo uma lista de estabelecimentos daquele setor de estabelecimentos da cidade. Armazenamos o nome e o endereço do local nessa etapa do scraping. Na busca em primeiro nível, já conseguimos estruturar pelo menos os dados referentes ao nome , tipo e link para detalhamento de informações de cada estabelecimento.

Se clicarmos em "+info", como mostrado na Figura 12, teremos uma informação detalhada sobre cada estabelecimento. Sequencialmente, o algoritmo entra em cada um dos links de detalhamento obtidos e efetua o scraping de informações detalhadas sobre esses estabelecimentos.

Figura 12 – Página contendo estabelecimentos (pizzarias, no caso) do Telelistas.net



Fonte: Telelistas.net

A view construída funciona como um Web Crawler (', 2011), específico para o site em questão, entrando sequencialmente em cada link de uma "área" de estabelecimentos daquela cidade. Várias dessas páginas possuem em seu rodapé uma paginação com diversas outras abas de lugares-resposta. A view percorre todas essas abas, já armazenando em estrutura de dicionários Python (em memória), o nome, o endereço e o link de detalhamento daquele lugar. Os algoritmos foram construídos similarmente aos anteriores, baseados em inspeção e análise dos códigos HTML para descoberta de padrões de resposta. Por exemplo, a div com id 'contato_1_a' sempre possuía em seu conteúdo os telefones dos estabelecimentos (que possuem telefone cadastrado na parte de contatos), sendo necessário "apenas" tratamento prévio para "raspagem" do lixo de informação. Lembrando que , baseado em testes empíricos os "lixos" de informação, ou seja, tudo aquilo que não era número de telefone (como no exemplo da imagem os prefixos "Tel" e "WhatsApp"),

foi removido durante o scraping, adicionando os números de telefone que restaram na estrutura JSON daquele documento (criada na etapa anterior do scraping, supracitada). Esses prefixos, antes de serem removidos foram utilizados para auxílio na categorização da natureza daqueles números de telefone, permitindo armazená-los com a diferenciação se são números referentes a um aparelho celular ou a um telefone fixo.

3.2.3 Unificando os documentos e salvando-os na segunda camada de dados

A próxima etapa no fluxo da aplicação sensor é a execução de uma função responsável pela unificação das informações até agora buscadas. Seu funcionamento é simples porém sua importância é enorme no sistema. Ela se baseia nos nomes dos endereços da coleção Lugar_Places, para procurar por meio de consultas na coleção Lugar_Telelists, nomes semelhantes aos citados, com intuito de encontrar dados de diferentes fontes sobre um mesmo local . Como a resposta da Google Places API é dada em inglês, foi utilizada a biblioteca Python TextBlob para realizar a tradução dos nomes dos locais para português , para casamento entre esses nomes. A unificação é feita consultando-se os nomes de todos os documentos das coleções Lugar_Places e Lugar_Telelist , passando esses nomes por um pré-processamento que inclui remoção de stopwords (palavras que possam atrapalhar a comparação, como preposições, por exemplo), colocando-os em caso minúsculo. Para comparação entre os nomes dos lugares, também tornou-se necessário deixá-los em letras minúsculas e remover (mediante análise dos dados e inclusão de tratamento particularizado) dados que não eram referentes aos nomes dos lugares, mas sim à localização como o termo “ no Centro”, na Figura 12, acima. Feito isso, comparam- se os nomes. Encontrando ou não nomes semelhantes os documentos criados são persistidos na coleção Lugar_Unificado. O nome dessa coleção foi escolhido tendo em vista a unificação de informações do mesmo estabelecimento, oriundas de várias fontes de scraping.

3.2.4 Busca de informações no Facebook

Após unificação dos dados “scrapeados”, que antes eram “independentes” o sistema busca nessa coleção de documentos que foi “cruzada” , a Lugar_Unificado, o nome de cada estabelecimento e realiza nome a nome a busca por página)s) de estabelecimento(s) no Facebook que seja(m) daquela cidade e que possua(m) nome semelhante ao vigente à pesquisa. Essa funcionalidade foi construída utilizando a Facebook Graph API Explorer, mais especificamente a parte que tange à pesquisa de nomes de páginas e posterior busca de algumas informações adicionais sobre essas páginas respondidas, utilizando o terminal com da API com link que referencia, o id do Facebook daquela página, como mostrado na Figura 13. Independente da view encontrar ou não a página pesquisada, no fim do fluxo os dados serão persistidos na coleção que cria uma terceira camada de abstração no nível de armazenamento do sensor, referenciando uma camada de dados com mais

um grau processamento nos dados obtidos inicialmente.Essa coleção foi chamada de Lugar_Unificado_Completo para dar o entendimento de que este é o último estágio de busca que o sensor realiza, ou seja essa tabela está completa em termos de buscas que o sistema realiza sobre um estabelecimento, e portanto, pronta para ser utilizada pela aplicação Atuador. O primeiro passo para criação de uma aplicação que busca dados do Facebook de acordo com a documentação oficial disponibilizada em site, foi a criação de uma conta na rede para o nosso sistema, com posterior cadastro dessa conta como uma conta de desenvolvedor do Facebook. Esse cadastro nos concede um "segredo de cliente" (client secret) e um "ID de cliente" (client id), utilizados na obtenção do token de acesso aos dados do Facebook, fornecidos via Graph API ,um parâmetro necessário para cada requisição de dados que for feita utilizando a mesma . Foi testado empiricamente que quando a rede social encontra o resultado esperado da pesquisa realizada, este resultado é na grande maioria das vezes, o primeiro item dessa lista. Em algumas vezes ou o resultado estava na segunda colocação da lista ou então nem estava entre os resultados retornados. Por isso , por definição a view descrita acima faz uma nova requisição com o ID da primeira resposta da pesquisa do nome do local. Se a resposta possuir como endereço a mesma cidade que a já definida para aquele local, ele salva adicionalmente no documento desse local as informações detalhadas sobre o ele fornecidas pelo Facebook. Caso não seja a mesma cidade ele tenta executar o mesmo procedimento com o segundo resultado da pesquisa do nome. Se essa segunda tentativa falhar, a view considera que não foi encontrado na rede social nenhum estabelecimento com aquele nome.

Figura 13 – Exemplo de busca através da Graph API Explorer

Fonte: graph.facebook.com

A Figura 13 se refere à Graph API Explorer, uma ferramenta online para simulação das requisições HTTPSs feitas aos servidores do Facebook. Como podemos ver a resposta a uma chamada feita utilizando um ID capturado, é um JSON contendo alguns dos dados solicitados (os que a página com ID passado como parâmetro possui).

A view chamada view_busca_nomes_facebook é responsável por obter os nomes

dos estabelecimentos contidos nos documentos já unificados e persistidos e utilizá-los como entrada em uma função de busca de nomes de estabelecimentos no Facebook. Essa função retorna uma lista de nomes de estabelecimentos com seus respectivos ids. Após comparação com o nome do estabelecimento já armazenado em banco, o id correto é utilizado para a obtenção de informações detalhadas da página daquele estabelecimento, caso encontre-a.

No fim do fluxo, a aplicação sensor do Tingoram transforma e salva os dados não-estruturados, que foram capturados e pré-processados pelo sensor em forma de coleções no Mongo BD (ou seja, de forma estruturada), para que o posteriormente a aplicação atuador possa trabalhar com esses dados.

3.3 Atuador

Ao término da execução da aplicação sensor do Tingoram, todos os dados obtidos estarão unificados e centralizados na coleção Lugar_Unificado_Completo. A próxima fase do sistema deste trabalho consiste em utilizar esses dados para criação de vínculo entre doadores e recebedores de forma georreferenciada e posterior comunicação entre as partes. Com a finalidade de estabelecimento de contato com os potenciais doadores e recebedores de alimentos obtidos pelos sensor, foi criada uma aplicação dentro do projeto Django (projeto Tingoram) chamada Atuador. Essa aplicação contém dentro de si os códigos responsáveis pelo envio de mensagens semi-automatizadas para envio de mensagens via WhatsApp para os números obtidos, para as páginas do Facebook encontradas e envio de e-mail.

3.3.1 Indexação da sugestão de doação

A primeira função do atuador sobre os dados , é indexar os dados georreferenciados, ou seja, ordenar os doadores e os recebedores de uma dada cidade ou região com intuito de criar conexões entre as partes envolvidas no processo de doação, geograficamente mais próximas, com posterior sugestão de criação de vínculo entre os estabelecimentos. Nos casos em que algum estabelecimento contactado responda a alguma mensagem do Tingoram, o sistema gera a sugestão de doação/recebimento. Essa sugestão é uma lista rankeada de estabelecimentos, com seus endereços e contatos. O ranking de sugestão é construído baseado nos estabelecimentos mais próximos geograficamente da instituição que respondeu à mensagem enviada pelo nosso sistema. Para realização do cálculo, foi utilizada a biblioteca Python pyproj, utilizada na manipulação de coordenadas geográficas. A limitação para a lista de doação foi de 10 estabelecimentos, tendo em vista que dificilmente algum estabelecimento terá contato de doação com muitas empresas, devido à própria limitação da validade dos alimentos. No caso do lugar ser um potencial doador, serão sugeridos potenciais recebedores, e vice-versa. Por último na lista, aparecem os locais sem

georreferenciamento, mas na mesma cidade que o usuário. A fim de otimizar o sistema e como a primeira mensagem não envia informações sobre estabelecimentos ao seu redor, essa indexação só é feita em caso de resposta do estabelecimento em questão ao primeiro contato feito com o mesmo.

3.3.2 Efetivação da comunicação do sistema com as Empresas

Com os dados unificados em tabela e com as respectivas informações sobre os estabelecimentos buscados pelo sensor no Google Places, no Telelistas e no Facebook, o Tingoram tentará enviar um mensagem inicial de primeiro contato com a empresa. A finalidade do estabelecimento desse primeiro contato com as entidades doadoras e recebedoras é o envio da seguinte mensagem :

"Olá [ESTABELECIMENTO]. Você foi selecionado automaticamente como potencial DOADOR/RECEBEDOR de alimentos pelo Tingoram. Recentemente a PL 5958/13 foi aprovada. Ela implica em REDUÇÃO DE IMPOSTO para quem doar alimentos, bem como MULTA para quem NÃO doar. Responda TENHO INTERESSE ou NÃO,OBRIGADO, para evitar o envio de spam. Obrigado. Equipe Tingoram."

3.3.2.1 Mensagens via WhatsApp

Para conseguirmos realizar o envio de mensagens automáticas para os números de WhatsApp pré-capturados, utilizamos a plataforma Web disponibilizada pela empresa para seus usuários no endereço https://web.whatsapp.com/ . Como essa é uma plataforma destinada a usuários finais e não uma API, foi necessário utilizar o Selenium Web Driver para simular de maneira automatizada a interação de usuário com o WhatsApp Web. O grande desafio da construção dessa automatização se deu quanto à simulação exata do comportamento do usuário na plataforma citada, por meio da construção do código com sequência específica, utilizando o Selenium como ferramenta. Para isso o autor utilizou a plataforma em questão (WhatsApp Web) por diversas vezes, com o navegador em modo desenvolvedor para analisar o comportamento do sistema bem como seus códigos HTMLs. Foi observado o fluxo de comunicação em resposta às interações com o sistema e as URLs de redirecionamento. Gradativamente a interação foi simulada . Por fim foi construído uma view que abre um fluxo onde o usuário que deseja enviar uma mensagem em "massa" precisa é notificado da necessidade de instalar o aplicativo previamente em um celular e seguir o procedimento padrão da empresa para se conectar na plataforma Web (tirar uma foto do QRCode para se logar). Feito isso o WhatsApp do número cadastrado no celular que logou na web começa a enviar a mensagem do Tingoram, citada acima, para a lista de telefones cadastrados na base de dados do nosso sistema. Para executar esse procedimento foi comprado um número de telefone especificamente para o Tingoram. Esse número foi cadastrado na plataforma em questão e aí sim utilizado para estabelecimento de contato

com as empresas, para tal seguindo os passos descritos anteriormente. Esse método de contato foi considerado semi-automático por necessitar de prévio cadastro, instalação de aplicativo e login pelo administrador do sistema.

3.3.3 Armazenamento de contatos a serem feitos por voluntários

Os estabelecimentos que possuíam apenas endereço físico e/ou telefone fixo foram armazenados no momento da criação de vínculo de contato com um booleano marcando-os como estabelecimentos a serem contactados via voluntários. Isso foi feito tendo em vista quando o sistema estiver aberto para a comunidade, momento em que estarão abertos os cadastros de voluntários e esses estabelecimentos poderão receber visitas e/ou ligações desses voluntários pré-cadastrados no Tingoram. Nesse protótipo, voluntários puderam se cadastrar em um formulário criado para a aplicação, no endereço do Django admin do sistema. A idéia é que futuramente o cadastro dos voluntários seja feito mediante a ferramenta de "login com um click" da Facebook Graph API, já citada préviamente nesse documento e também já utilizada no nosso sistema.

4 Resultados

Para estudo de caso e escrita desse documento foi escolhida como base a cidade de Mariana , Minas Gerais, cidade de atual residência do autor deste trabalho. Com o servidor do Django no ar, colocamos o sistema para executar, chamado manualmente cada trecho do processo, a fim de observarmos os resultados.

4.1 Resultados da execução da aplicação sensor

A aplicação sensor percorreu as etapas já citadas nesse trabalho e produziu como resultado a captura dos dados, como podemos ver nessa seção.

4.2 Busca via Google Places

A primeira parte da execução do fluxo do sensor é a busca de informações através da utilização API Places. Ao digitarmos no navegador o link cadastrado para acionar a view que executa o sensor com os códigos que utiliza a Google Places API para a cidade de Mariana, o Tingoram executou a busca para essa cidade e obteve 483 lugares. A maior vantagem dos dados oriundos da Places API é que eles possuem georreferenciamento, ou seja, poderão ser usados na indexação.

Figura 14 – Debug da View que conta os lugares obtidos via Places API

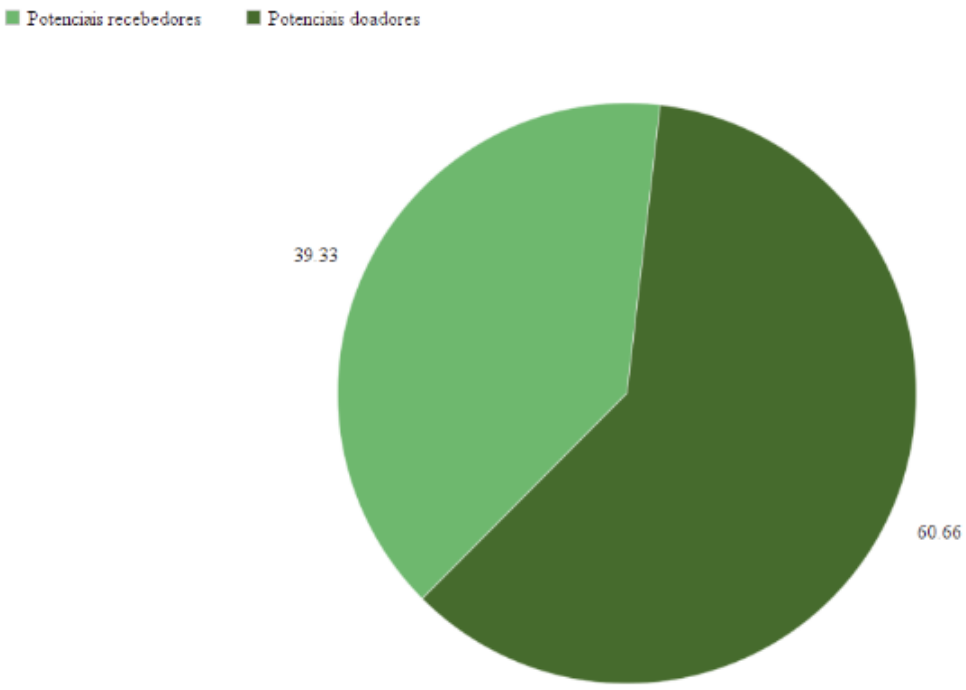
```
NUMERO DE POTENCIAIS LUGARES PLACES >>> 483
 NUMERO DE POTENCIAIS LUGARES DOADORES >> 293
 NUMERO DE POTENCIAIS LUGARES RECEBERDORES >> 190
[03/Jul/2018 23:51:06] "GET /view_conta_places/ HTTP/1.1" 200 139
```

Fonte: graph.facebook.com

A Figura 14 mostra a resposta de uma view construída com o intuito de contar quantos tipos de estabelecimentos de cada natureza foram buscados através da API Places.

Podemos observar que existem 293 potenciais doadores, o que implica em 60,66% dos lugares buscados via Google Places. Isso implica em 39,33% dos lugares buscados por esse meio são potenciais recebedores de alimentos, ou seja, 190 estabelecimentos. O gráfico abaixo ilustra tal situação.

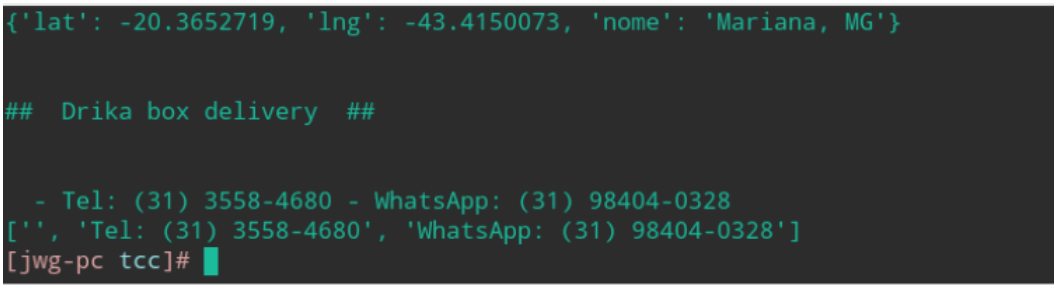
Figura 15 – Percentual de locais buscados na Places API, segundo sua natureza (doador ou recebedor)



Elaborado pelo autor

4.3 Busca via scraping em Telelistas.net

Figura 16 – Print dos debugs do sistema evidenciando o scraping de números de telefone no Telelistas.net



Elaborado pelo autor

Já as buscas do sensor através do site Telelistas.net rodou a cidade de Mariana , procurando por estabelecimentos dentro dos links descritos no Quadro 3. As páginas que possuíam cursor de navegação foram percorridas recursivamente até o fim. Cada link pode possuir N cursores de navegação de páginas sucessivas, percorridas pelo crawler do

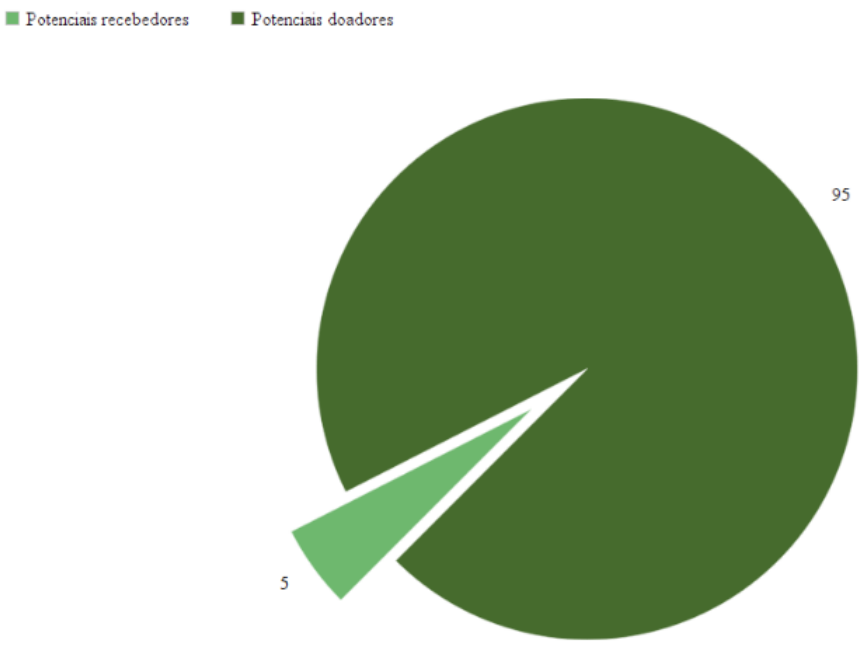
Figura 17 – Resultado da view criada para contabilização de quais lugares sejam potenciais doadores e recebedores.

```
NUMERO DE POTENCIAIS LUGARES TELELISTAS >>> 100
 NUMERO DE POTENCIAIS LUGARES DOADORES >> 95
 NUMERO DE POTENCIAIS LUGARES RECEBERDORES >> 5
```

Elaborado pelo autor

Tingoram que armazena todas as informações, tanto dos estabelecimentos das páginas quanto entrando no link referente ao botão "+ info" de cada um desses estabelecimentos, como foi mostrado na Figura 12 .

Figura 18 – Potenciais doadores e recebedores de alimentos oriúndos do scraping em Telelistas.net .



Elaborado pelo autor

A Figura 16 mostra o resultado intermediário do pré-processamento das informações contidas na div referente aos contatos, após o crawler entrar na página referente ao link em "+info". Podemos ver o nome do estabelecimento,a fins de debug,seguido dos telefones obtidos na div 'contato_1_a', citado anteriormente neste trabalho.A busca no Telelistas.net retornou 100 estabelecimentos, como podemos ver no cursor do banco de dados da imagem dos quais Y são doadores e Z recebedores. Essa diferença se deu devido ao site ser voltado

ao comércio e portanto, mais focado em potenciais desperdiçadores. Como ilustra o gráfico da Figura 18 , 95% dos contatos buscados nesta etapa do scraping foram de potenciais doadores de alimentos, contra 5% de potenciais recebedores.

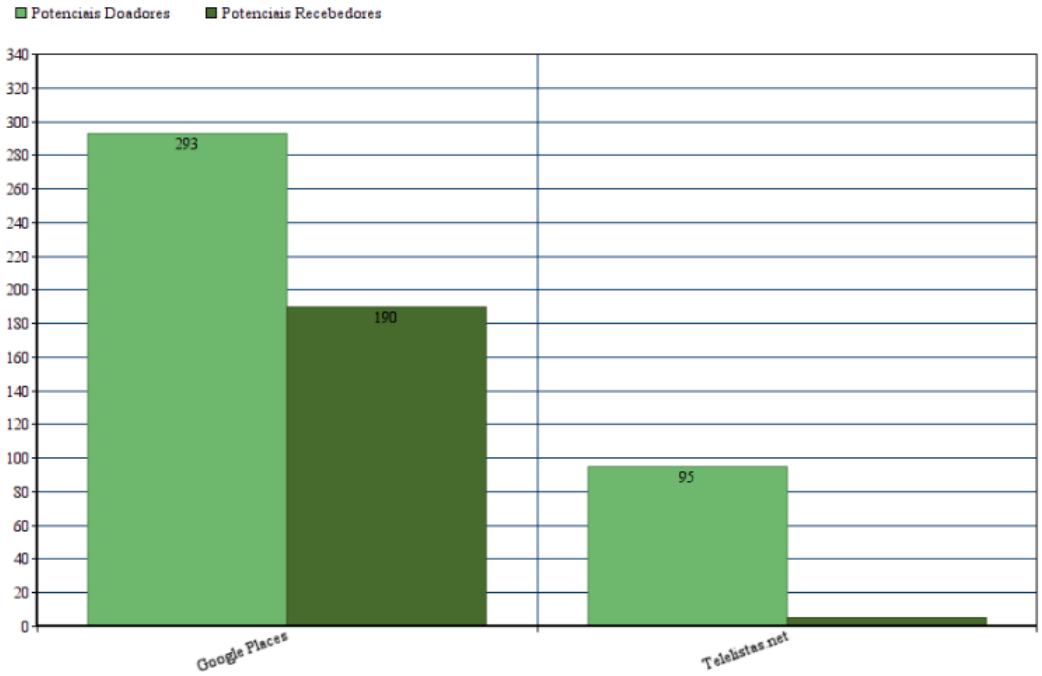
4.4 Busca via Facebook

Na busca via Facebook, recebemos um JSON referente à pesquisa efetuada, semelhante ao da Figura 13, na Seção 3.2.4. Confirmado o nome da página respondido ao comparar com o já persistido que originou a consulta no Facebook, obtemos as informações detalhadas, referentes à segunda consulta à Graph API. Vemos a resposta no segundo JSON, contendo endereço , telefone e imagem de capa.

Como a busca através do facebook não se deu na primeira camada e sua função era apenas de complemento de dados, ou seja, não foi executada busca visando novos lugares no mesmo, não foi necessária construção funções de contabilização desse volume de dados.

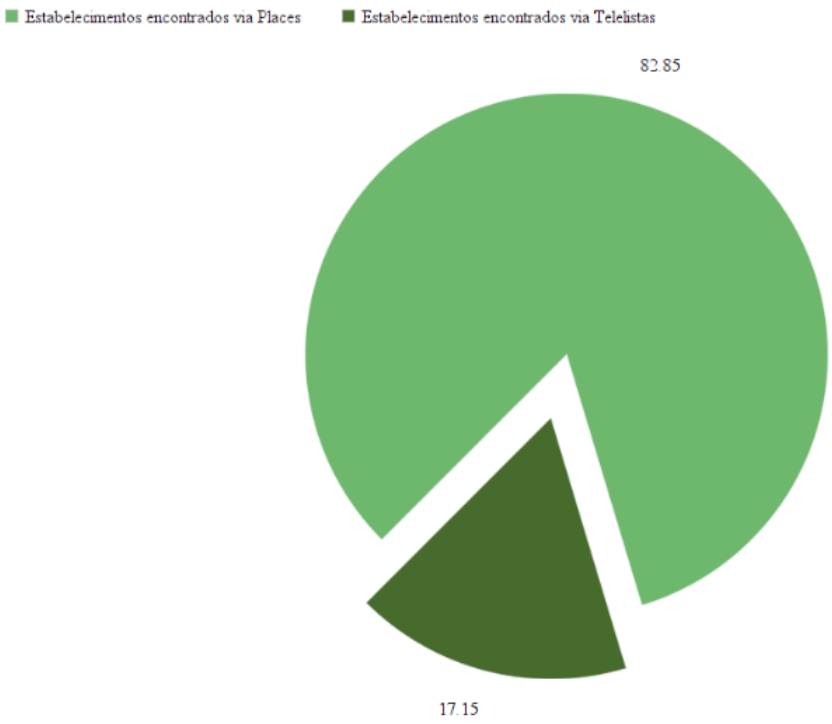
4.4.1 Comparação entre as duas principais fontes de dados: Google Places e Telelistas

Figura 19 – Potenciais doadores/recebedores de alimentos e de onde vieram esses dados



Elaborado pelo autor

Figura 20 – - Gráfico que mostra de onde vieram os dados de cada estabelecimento



Elaborado pelo autor

Como pudemos evidenciar nas seções anteriores e confirmar no gráfico abaixo, tivemos um total de 583 lugares obtidos, dos quais 82.85% foram provenientes do Google Places como mostram os dados dos gráficos.

4.5 Comparação segundo a natureza do contato buscado

A Figura 21 é proveniente do resultado da execução da view responsável por contar cada tipo de contato obtido pelo sensor.

Como podemos ver, conseguimos a captura de 100% dos endereços dos lugares. Em 3 estabelecimentos do tipo Telelist conseguimos obter especificamente números de WhatsApp.

De um total de 368 números de telefone genéricos, conseguimos identificar, através de tratamento de padrão, 81 números de celulares, somados aos 3 contatos de WhatsApp encontrados. Os gráficos demonstram isso visualmente:

Como podemos ver , 85,59% dos estabelecimentos não possuem dados suficientes para contato via celular (Figura 33) e 36,36% deles não possui nenhuma forma de contato

Figura 21 – Execução da view que gera metadados sobre os dados produzidos

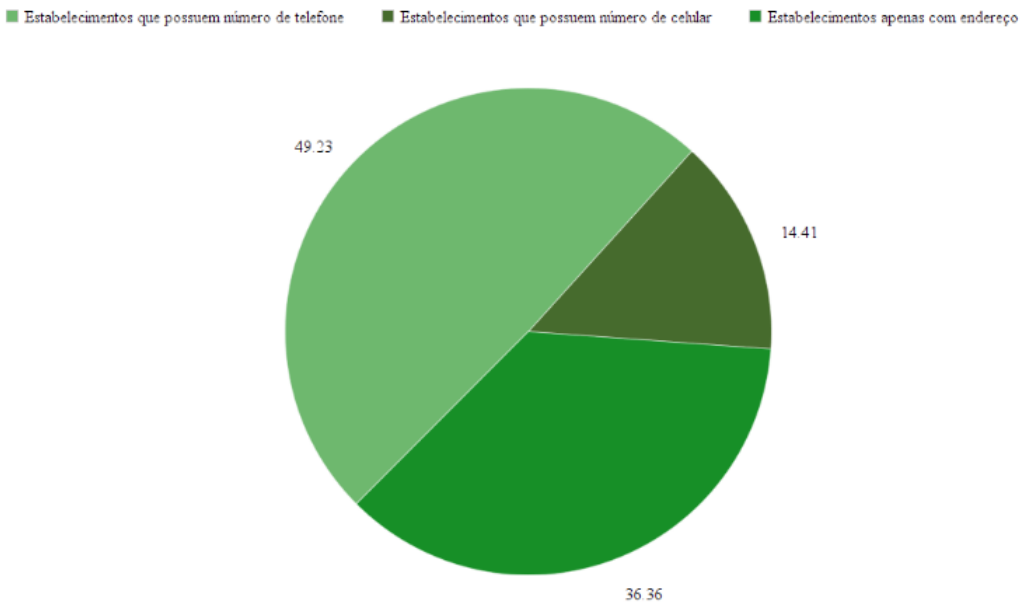
```
NUMERO ENDERECOS telelists >>> 100
 NUMERO DE TELEFONES telelists >> 9
 NUMERO DE WHATSAPPs telelists >> 3
NUMERO ENDERECOS palces >>> 483
 NUMERO DE TELEFONES palces >> 359
 NUMERO DE WHATSAPPs palces >> 0

NUMERO ENDERECOS TOTAL >>> 583
 NUMERO DE TELEFONES TOTAL >> 368
 NUMERO DE WHATSAPPs TOTAL >> 3


 NUMERO DOS TELEFONES QUE SÃO CELULAR >> 81
```

Elaborado pelo autor

Figura 22 – Número de totais de estabelecimento de acordo com a natureza dos contatos buscados



Elaborado pelo autor

além do endereço físico, sendo portanto necessária a atuação nesses locais via voluntariado.

4.6 Bot Facebook

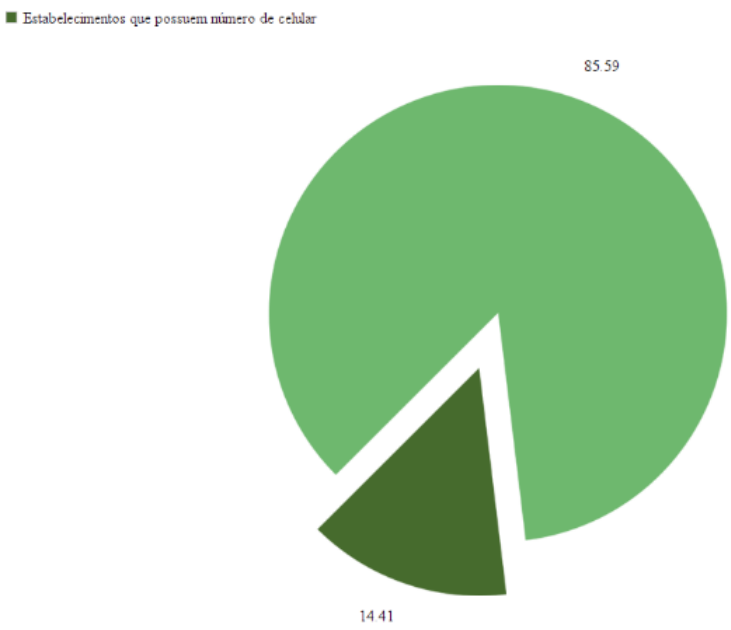
Construímos um bot para o envio de mensagens para o Facebook das páginas "scrapeadas" , mas até o presente momento, a política da Rede Social só nos permite o

envio automático de mensagens passivas, ou seja somente quando alguém entra na nossa página ou clica em algo relacionado à mesma. Segundo documentação oficial, existe uma exceção onde é possível enviar mensagens de forma ativa, o que o ainda não conseguimos. A investigação prossegue para além deste documento, como imprescindível futuro trabalho a ser feito.

4.7 Contatos feitos via Wpp

Como podemos ver na terceira seção deste capítulo, a aplicação sensor obteve 84 números de telefones, o que, de acordo com o gráfico da Figura 23 representa 14,41% de todos os estabelecimentos buscados.

Figura 23 – Porcentagem de estabelecimentos que contém número de celular

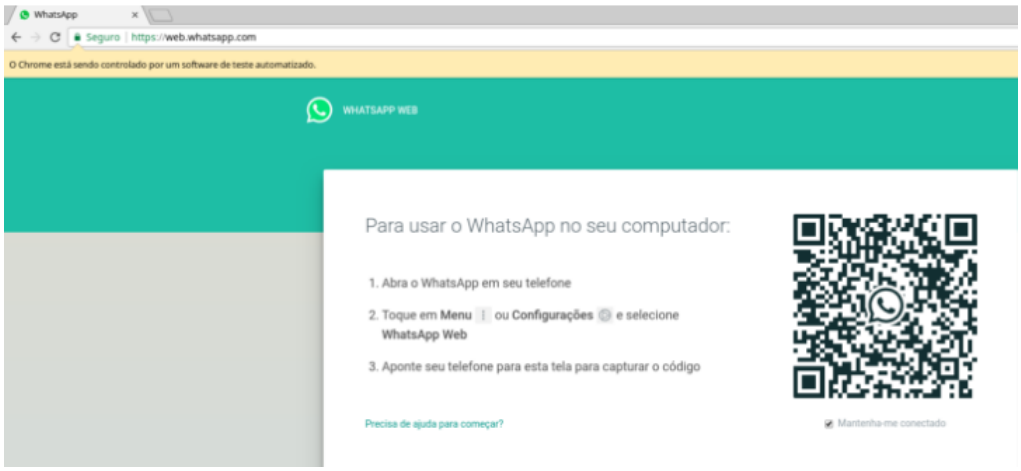


Elaborado pelo autor

Não descobrimos uma maneira de verificar se o número possuía cadastro no WhatsApp, por isso enviamos em laço uma mensagem a todos os estabelecimentos com campos referentes ao número de whatsapp ou número de celular, utilizando o número específico do Tingoram.

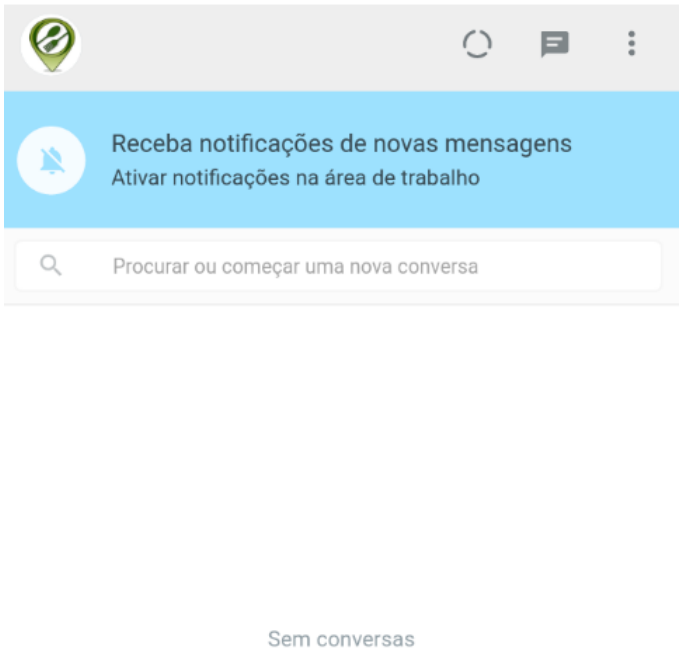
A Figura 24 mostra o redirecionamento para login na plataforma web da rede social, antes e a Figura 25 após usuário tirar a foto necessária para realizar o login. Como podemos ver, antes da execução não haviam conversas com nenhum contato, dado que o número é novo e foi comprado apenas para o trabalho em questão.

Figura 24 – Parte do processo que não é automatizado, o login na plataforma WhatsApp Web



Web Whatsapp

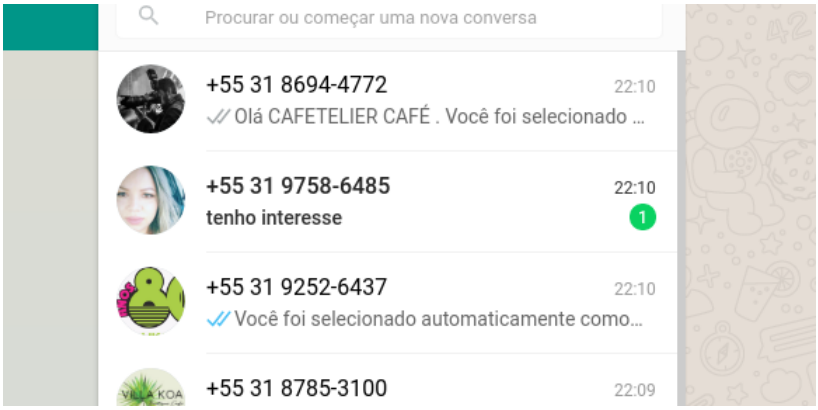
Figura 25 – Não haviam conversas antes de rodarmos a view que envia as mensagens



Elaborado pelo autor

A Figura 26 mostra a caixa de contatos após rodar o script que envia as mensagens para os números de celular da base. Foram enviadas mensagens para X números. As respostas não foram contabilizadas até a data de término desse documento.

Figura 26 – Imagem do envio automático. Podemos ver uma resposta positiva



Elaborado pelo autor

5 Conclusão

Vivemos em um cenário globalizado, com um terço dos recursos alimentares sendo desperdiçados. Existem informações referentes aos contatos dos estabelecimentos de alimentos na Web, de forma não estruturada. Para suprir a necessidade da logística da doação desses alimentos, o Tingoram surge como uma ferramenta de auxílio à realocação desses recursos através da sugestão de doação criação de vínculo de doação de alimentos baseada em georreferenciamento. Este trabalho consistiu na implementação do Protótipo Mínimo Viável desse sistema, bem como execução de um estudo de caso na cidade de Mariana, Minas Gerais. Como mostrado, ainda existem inúmeros desafios para a efetivação de um sistema que realize de maneira automatizada uma tarefa com essa complexidade, mas já conseguimos obter dados referentes a 587 estabelecimentos, entre doadores e recebedores, somente na cidade de Mariana, Minas Gerais, dos quais aproximadamente 14Foram enviadas mensagens para cada um desses 84 números de maneira semi automatizada via WhatsApp. Até o término desse documento, alguns estabelecimentos haviam respondido à mensagem , o que ainda não foi contabilizado. Duas contribuições técnicas deste trabalho foram a busca semi automatizada de números no Telelistas.net, com posterior estruturação dessas informações e a construção do código com o envio de mensagens em massa via WhatsApp. O desafio prossegue, com a esperança de que o Tingoram possa futuramente auxiliar a efetivamente matar a fome de muita gente, através da criação de vínculos de doação entre estabelecimentos e entidades recebedoras de alimentos.

5.1 Limitações

Dentre as principais limitações deste trabalho,podemos destacar : a ausência de respostas aos estabelecimentos que responderam às mensagens, ausência de envio de SMS, uma fraca integração entre os componentes e o fato do bot do Facebook implementado ser passivo.

5.2 Trabalhos futuros

Esse trabalho possui em frente uma série de desafios a serem enfrentados, dentre eles uma melhoria na unificação dos dados e criação de um laço que utiliza todas as funções de forma a colocar o Tingoram disponível na rede, “farejando “ cidades o tempo todo. Uma demanda prioritária é a criação de cadastro de voluntários, para soltar o sistema para a comunidade e a mesma poder auxiliar através de ligações e visitas. Outra é deixar o bot do facebook ativo e não passivo. A idéia é conseguir setar os parâmetros do envio de

mensagem de forma a fazer com que a rede social interprete nossa mensagem como uma mensagem não promocional e, se possível como filantrópica. Se isso ocorrer, poderemos refinar a busca no Facebook e enviar mensagens de solicitação de criação de vínculo. Futuramente seria interessante uma maneira de realizar o envio de uma mensagem de áudio automática, pré armazenada para os telefones fixos dos estabelecimentos, dado o volume de dados de telefones fixos encontrados. Também seria interessante trabalhar com envios de mensagens SMS .

Referências

', B. *Web Data Mining: Exploring Hyperlinks, Contents and Usage Data.* [S.l.]: Springer, 2011. Citado 2 vezes nas páginas 16 e 34.

BHARANIPIRYA V. PRASAD, V. K. Web content mining tools: A comparative study. 2011.

BORGES J. E LEVENE, M. Data mining of user navigation patterns. WEBKDD'99, 1999. Citado na página 17.

CHODOROW, K. *MongoDB: The Definitive Guide.* [S.l.]: O'Reilly, 2013. Citado na página 17.

CORTÊS SéRGIO DA COSTA. PORCARO, R. M. L. S. Mineração de dados - funcionalidades, técnicas e abordagens. In: . [S.l.: s.n.], 2002. p. 11. Citado 2 vezes nas páginas 24 e 25.

CRIZEL VINICIUS DA S., N. M. U. e. a. Repositório digital: instrumento de aprendizagem na educação profissional. V Congresso Brasileiro de Informática na Educação (CBIE 2016), 2016. Citado na página 16.

DEPUTADOS, C. dos. *Ordem do dia.* 2010. Disponível em: <http://www.camara.leg.br/internet/ordemdodia/ordemDetalheReuniaoCom.asp?codReuniao=52855>. Citado na página 24.

DJANGO, E. *Django Documentation.* [S.l.], 2018. Disponível em: <https://docs.djangoproject.com/>. Citado na página 21.

ELOISA URRU, M. V. Exploiting web scraping in a collaborative filtering-based approach to web advertising. Barcelona Digital Technology Centre, 2013. Citado 2 vezes nas páginas 24 e 25.

EMBRAPA. *Redução do desperdício de Alimentos.* [S.l.], 2016. Disponível em: <https://www.embrapa.br/>. Citado na página 21.

FOOD; ORGANIZATION(FAO), A. *Food wastage footprint : Impacts on natural resources.* [S.l.]: ONU, 2013. Citado na página 14.

FOOD; ORGANIZATION(FAO), A. *Panorama de Insegurança Alimentar e Nutricional na América Latina .* [S.l.]: ONU, 2017. Citado na página 14.

GLEZ-PEÑA DANIEL, L. A. e. a. Web scraping technologies in an api world. 2013. Citado 3 vezes nas páginas 7, 8 e 14.

HOLOVATY MOSS, J. K. The definitive guide to django: Web development done right. 2015. Citado na página 19.

IBGE. *Inseguridade alimentar cai nos domicílios.* 2014. Disponível em: <https://agenciadenoticias.ibge.gov.br/agencia-noticias/2013-agencia-de-noticias/releases/14735-asi-pnad-inseguranca-alimentar-nos-domicilios-cai-de-302-em-2009-para-226-em-2013.html>. Citado na página 24.

KOSALA RAYMOND BLOCKEEL, H. Web mining research:a survey, 2000. Department of Computer Science,Katholieke Universiteit Leuven, 2000.

KUMAR, T. S. A. M. S. N. A comparative analysis of different web content mining tools. 2014.

NEIL, Y. . Web scraping: the easy way. Department of Information Systems. Georgia Southern University, 2016. Citado 3 vezes nas páginas 7, 8 e 14.

RUSSEL, M. A. *Mining the Social Web, Second Edition*. [S.l.]: O'Reilly, 2014.

SINGH, K. Survey of nosql database engines for big data. 2015. Citado na página 16.

SOLUTIONS, B. Web data scraping. market research. bizzbee, 2017. Citado na página 21.

UPADHYAY AVIRAL .JAIN, V. Mongodb and nosql databases. International Journal of Computer Applications, 2017. Citado 2 vezes nas páginas 17 e 21.

Citado na página 19.

Citado na página 19.

Citado na página 18.

Citado 2 vezes nas páginas 19 e 20.

TERMO DE RESPONSABILIDADE

Eu, **Jonathan Wagner Guimarães** declaro que o texto do trabalho de conclusão de curso intitulado "***Elaboração e construção de um protótipo mínimo viável para o Tingoram : Um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos***" é de minha inteira responsabilidade e que não há utilização de texto, material fotográfico, código fonte de programa ou qualquer outro material pertencente a terceiros sem as devidas referências ou consentimento dos respectivos autores.

João Monlevade, 12 de julho de 2018

Jonathan Wagner Guimarães

Certifico que o aluno **Jonathan Wagner Guimarães**, autor do trabalho de conclusão de curso intitulado "**Elaboração e construção de um protótipo mínimo viável para o Tingoram : Um sistema de mineração de dados web baseado em georreferenciamento para sugestão semi automatizada de doação de alimentos**", efetuou as correções sugeridas pela banca examinadora e que estou de acordo com a versão final do trabalho.

Igor Muzetti Pereira

Igor Muzetti Pereira

Orientador

Ouro Preto, 26 de Julho de 2018.