

DESENVOLVIMENTO DE APLICATIVO PARA ADOÇÃO DE ANIMAIS ABANDONADOS NAS RUAS

Marco Antônio Galdino Dias¹, Tony Hudyson Alves Campos², Vinicius Melo Pena³, Vítor Hugo Rodrigues⁴, José Roberto de Almeida⁵.

Universidade de Uberaba - UNIUBE, Uberaba - Minas Gerais

marco.agd.2013@edu.uniube.br; jose.almeida@uniube.br

Resumo

Atualmente, é possível perceber que nas cidades brasileiras há um grande número de animais nas ruas, seja porque foram abandonados ou seja porque estão perdidos. Há algumas ONGs e projetos que têm a finalidade de acolher esses animais, oferecendo abrigo, comida e o necessário até que seja adotado. Porém, ainda existe uma dificuldade por parte desses grupos, visto que existem muitos *Pets* nas ruas. Nesse sentido, o presente artigo trata sobre o desenvolvimento de uma aplicação para ajudar a divulgar animais para adoção, encontrar os respectivos donos que perderam seus pets, ou até mesmo, pessoas que queiram adotar. Para isso, será usada uma tecnologia que está em alta no mercado, que é o desenvolvimento de aplicativos móveis multiplataforma utilizando React Native que é capaz de fazer *build* para ambas plataformas iOS e Android.

Palavras-chave: Google Maps, React Native, Pets.

1 Introdução

O aplicativo *Android* tem a função de conectar as pessoas resgatantes, ou seja, pessoas que resgataram algum animal na rua, juntamente ao usuário final no qual gostaria de adotar um *pet*. Durante a elaboração do projeto, foi elaborado um *schema* de um aplicativo com o objetivo de mapear locais onde existam *pets* nas ruas ao redor da cidade. Apesar de o objetivo principal é retirá-los das ruas, é importante também encaminhá-los para um lugar seguro e responsável.

A relação entre humanos e seus animais de estimação se torna cada vez mais evidente no nosso dia-a-dia. Esse fenômeno não é novo, porém observa-se um grande aumento da importância dada à essa relação nas famílias de todo mundo, inclusive no Brasil. Mesmo com esse fenômeno, observa-se grandes quantidades de cães abandonados nos centros urbanos. (SILVA FILHO, 2017, p. 10),

Ainda sobre o abandono de animais, Silva afirma que

de acordo com a organização não governamental Sociedade Protetora dos Animais (SPAC), em Curitiba há, aproximadamente, 450 mil cães, sendo que destes 229 mil estão nas ruas. Até 2005, os animais recolhidos pela Prefeitura de Curitiba eram sacrificados. (2006 apud SILVA FILHO, 2017, p. 10).

Com a nova lei, a prefeitura não mais recolhe e nem sacrifica os animais na câmara de gás do Centro de Controle de Zoonoses (CCZ). Em anos anteriores, os sacrifícios passavam de 15 mil ao ano (VICENTE, 2007). Atualmente a prefeitura somente

realiza o recolhimento de animais agressivos, doentes ou que possam passar alguma enfermidade a seres humanos. Sendo assim, torna-se extremamente importante que haja um controle sobre esses animais, tanto para a saúde deles, como para a da população (SILVA FILHO, 2017, p. 10).

Com o objetivo de desburocratizar e facilitar o processo de entrega ao novo dono, foi criada uma API para fazer a comunicação com o Banco de Dados e um aplicativo móvel para o usuário, no qual será possível visualizar todos os *pets* disponíveis. Diante a isso, o cadastro dos *pets* será realizado em um cliente *Web*.

2 Fundamentação Técnica

Baseado no modelo de negócio e requisitos do projeto, o desenvolvimento de um aplicativo móvel foi-se demonstrado viável já que a maior parte da população tem ao menos um dispositivo móvel nos dias atuais, seja o mesmo Android ou iOS.

Devido às diferenças de *design* e tecnologias, o desenvolvimento nativo para Android e iOS se dá em linguagens e ambientes diferentes, obrigando praticamente todos os desenvolvedores, que queiram atingir o maior número de usuários possíveis, a desenvolver para ambas as plataformas. Outra possibilidade existente é a utilização de plataformas de desenvolvimento híbridas, que trazem benefícios em questões de reaproveitamento de código e manutenção, porém pecam em aspectos como *design* exclusivo para cada plataforma, desempenho (SILVA FILHO, 2017, p. 17).

Como o modelo de negócio da aplicação requer disponibilizar o aplicativo em ambas plataformas, ou seja, tanto para o Android quanto para o iOS, então optamos por utilizar uma tecnologia muito utilizada ultimamente, chamado React Native que realiza o processo de *build* para ambas plataformas.

3 Método

Para organizar as tarefas deste projeto, foi utilizado o sistema Kanban. Esse sistema foi escolhido em razão de ser prático e de ser considerado um dos métodos mais utilizados pelas empresas na atualidade.

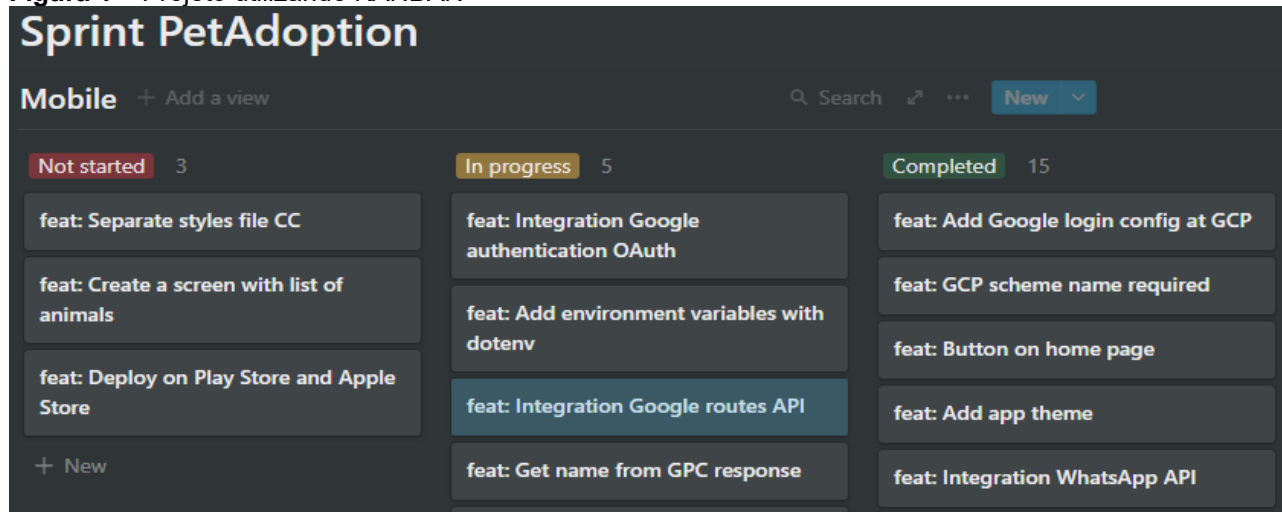
3.1 Kanban

Kanban é um sistema de controle e gestão do fluxo de produção em empresas e projetos que usa cartões coloridos (*post-its*) e também recebe o nome de gestão visual, em razão do uso de cores como sinalizadores (HERTER, 2021). Um exemplo de *site* que utiliza esse

método é o Trello. Nele temos colaborações, gerência de projetos e para alcançar novos picos de produtividade.

Na imagem a seguir demonstra-se como fica um projeto utilizando o Kanban:

Figura 1 – Projeto utilizando KANBAN

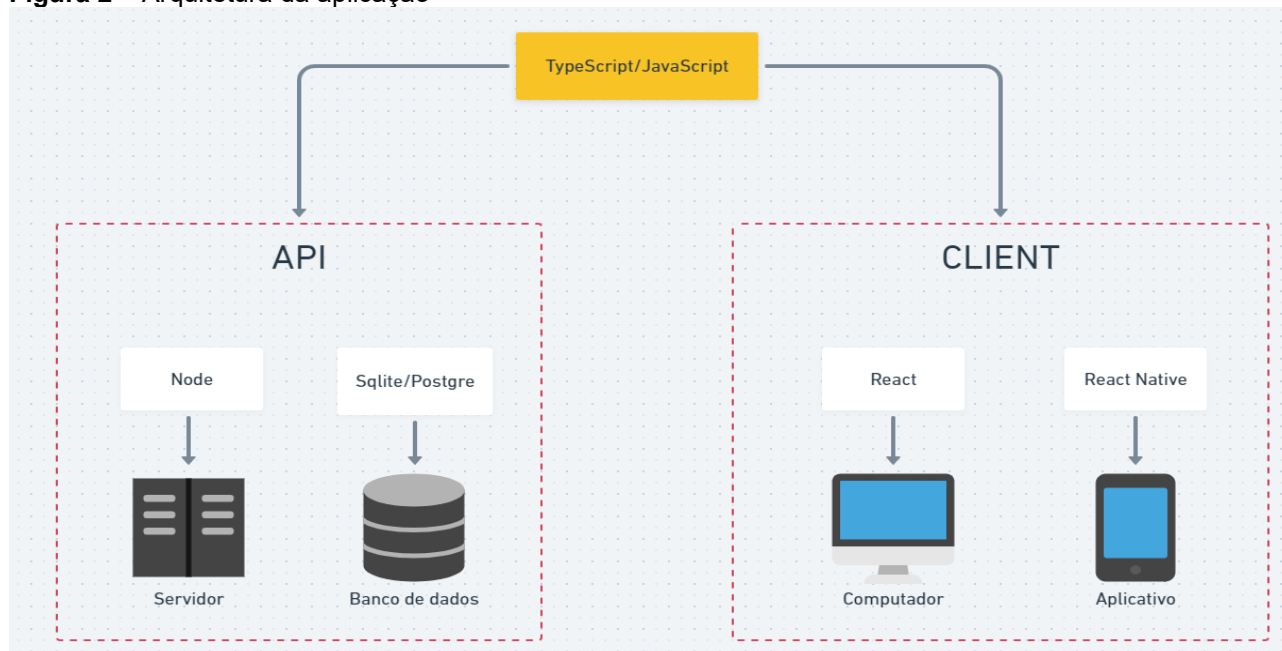


Fonte: Elaborada pelos autores.

3.2 Arquitetura

Conforme foi modelado e mediante as demandas do projeto, a arquitetura escolhida para suprir as necessidades da aplicação são basicamente três aplicações, sendo elas: camada cliente *web*, cliente *mobile* e outra de servidor, como visto na Figura 2.

Figura 2 – Arquitetura da aplicação



Fonte: Elaborada pelos autores

A primeira camada é a camada da API, na qual envolve o *Backend* da aplicação que foi desenvolvido em NodeJS e a parte do banco de dados que o SQLite gerencia. A camada da API é responsável por armazenar e disponibilizar os dados da aplicação em um arquivo conhecido como JSON (*JavaScript Object Notation*).

Já a segunda camada é a camada cliente, na qual envolve a aplicação mobile e também a aplicação *web*, o *mobile* é o aplicativo desenvolvido em React Native, e a *web* foi desenvolvida em ReactJS e ambas consomem as informações que estão contidos na camada do servidor. Importante lembrar que estas camadas operam de forma independente, ou seja, a camada *web* pode operar sozinha e no caso de falha em algum dos servidores estas camadas estarão mais seguras por estarem em servidores isolados e desacoplados um do outro.

3.3 Estimativa de horas

No início do projeto foi feita uma estimativa de atividades e horas a serem utilizadas no desenvolvimento de cada uma das atividades. No Quadro 1, temos uma relação das horas previstas a serem utilizadas para a execução do projeto. Observa-se que, apesar de alguma diferença entre as horas previstas e horas utilizadas, o erro ficou próximo de 10%, que é um valor aceitável para projetos desta natureza.

Quadro 1 – Estimativa de horas

Atividades	Horas previstas	Horas utilizadas
Estudo de viabilidade	8	8
Estudo das tecnologias	20	10
Modelagem dos objetos e relacionamentos	15	15
Criação da estrutura do projeto no GitHub	20	10
API - <i>Endpoint</i> listar todos os <i>pets</i>	20	20
API - <i>Endpoint</i> cadastrar uni pet	20	20
API - <i>Endpoint</i> filtrar <i>pets</i> por ID	20	20
API - <i>Endpoint</i> listar <i>pets</i> com o filtro	20	20
Tela <i>Web</i> - Criação da tela de boas vindas	20	20
Tela <i>Web</i> - Criação da tela de cadastro do doador e <i>pet</i>	20	20
Tela <i>Web</i> - Integração com API IBGE	30	30
Tela <i>Web</i> - Integração com API Leaflet	30	30
Tela <i>Mobile</i> - Criação da tela de boas vindas	20	20
Tela <i>Mobile</i> - Integração com Google Maps	30	30
Tela <i>Mobile</i> - Criação da tela da renderização do pet no mapa por filtro	20	20
Tela <i>Mobile</i> - Criação da tela das informações do <i>pet</i> e do doador	20	20
Tela <i>Mobile</i> - Integração com API IBGE	20	10
Tela <i>Mobile</i> - Integração com o telefone do celular	20	20
Tela <i>Mobile</i> - Integração com o Google Maps Routes	20	20
Documentação da Aplicação	30	30
Total	423	393

Fonte: Elaborado pelos autores

4 Desenvolvimento

Todo o projeto foi desenvolvido utilizando Typescript, no qual foi criado pela Microsoft e é um conjunto de ferramentas e formas mais eficientes para escrever códigos em Javascript. Ele adiciona recursos, como tipagem e orientação a objetos que não estão presentes no Javascript (MELO, 2021).

Na aplicação é utilizado para tipar as variáveis, sendo utilizado tanto no *backend*, *frontend* e na aplicação *mobile*.

4.1 Backend

O *backend* das aplicações foi construído utilizando o Node.js, que é um interpretador JavaScript assíncrono *open source*, ou seja, o Node.js consegue executar mais de uma tarefa ao mesmo tempo.

4.1.1 NodeJS

Node.js é atualmente uma das tecnologias mais utilizadas no desenvolvimento do lado do servidor, ou seja, *server side*, no qual é possível construir aplicações backend tanto para web quanto para mobile.

Moreira (2013) entende que o objetivo número um do Node é fornecer uma maneira fácil para construir programas de rede escaláveis. Em vez de criar um *thread* de conexão, cada requisição dispara um evento que executa dentro da engine, evitando *deadlock*, não permitindo bloqueios. Suportando assim dezenas de milhares de conexões simultâneas (apud CAMPOS *et al.*, 2016, p. 3).

Node.js tornou-se muito popular por utilizar a linguagem JavaScript, ou seja, uma linguagem comumente utilizada no *client side*, portanto isso permite com que o desenvolvedor possa construir aplicações *Full Stack* utilizando apenas JavaScript.

4.1.2 API (Interface de programação de aplicações)

A API (Interface de programação de aplicações) é basicamente um conjunto de padrões, que fazem parte da interface, permitindo criar uma plataforma para desenvolvedores de uma forma simples e prática. Uma API basicamente expõe *endpoints* nos quais são arquivos no formato JSON ou XML, o modelo JSON é mais aceitável no mercado de trabalho já que é um basicamente um objeto. Diferentes softwares, aplicativos, programas e plataformas podem ser criados a partir de uma API, ou seja, os aplicativos desenvolvidos para dispositivos Android e iOS são construídos de acordo com padrões definidos do modelo de negócio e são fornecidos pelas APIs através de *endpoints*.

Com as APIs pode-se desenvolver vários softwares e aplicações que possuem capacidade de se comunicar com outras plataformas. Essa comunicação acontece por exemplo em aplicações que utilizam mapas com a API do Google Maps, ou outras APIs como a do Instagram que permite publicar faixas e informações nos Stories.

O uso de uma API é simplesmente para facilitar no desenvolvimento dos projetos para programadores e ainda existe a possibilidade de criar novas plataformas através de padrões. Ao usar a API, não há necessidade de criar código personalizado para cada função que o programa executa. Descartando mais trabalho desnecessário e facilitando no desenvolvimento de novas aplicações para diferentes plataformas.

Vale salientar que “as APIs também possuem papel fundamental quando o assunto é segurança, já que também são capazes de bloquear acesso e permissões a dados de *software* e *hardware* que algumas aplicações não podem usar.” (GIGAOUTLET, 2021).

A figura 3 demonstra um retorno de uma rota da API da aplicação em formato JSON, que demonstra uma listagem de uma ligação de um dado, na qual é a ligação entre o doador e a referência do pet e a categoria na qual se enquadra.

Figura 3 – Retorno da API em formato JSON

```
1  [
2    "petpoint2": {
3      "id": 1,
4      "image": "266ffc73f28c-96bd6e8891ff-berkay-gumustekin-ngqyo2AYYnE-unsplash.jpg",
5      "username": "Vitor",
6      "petname": "Rex",
7      "description": "Rex e ",
8      "email": "vitor@gmail.com",
9      "whatsapp": "35999607755",
10     "latitude": -19.751679005289407,
11     "longitude": -47.96656608581544,
12     "city": "Uberaba",
13     "uf": "Minas Gerais",
14     "image_url": "http://192.168.0.2:3333/uploads/266ffc73f28c-96bd6e8891ff-berkay-gumustekin-ngqyo2AYYnE-unsplash.jpg"
15   },
16   "category": [
17     {
18       "title": "Cachorro"
19     }
20   ]
21 ]
```

Fonte: Elaborada pelos autores

4.2 Cliente Web

O *frontend* da aplicação, no qual é uns dos clientes do projeto, foi decidida a utilização do ReactJS, que é uma biblioteca JavaScript *open source* desenvolvida para a criação de interfaces de usuário (UIs) em páginas web. O Cliente *Web* ficará responsável pela realização do cadastro no banco de dados de todas as informações necessárias referente aos *pets* e aos doadores.

4.2.1 ReactJS

ReactJS é uma tecnologia desenvolvida pela a empresa Facebook, que tem como principal objetivo construir interfaces de usuário, ou seja páginas *front-end* web. Atualmente, é utilizado em várias *big techs*, também conhecidas como FANGs(Facebook, Amazon, Netflix e Google), pois essa tecnologia tem muitas vantagens em se utilizar a mesma, tais como: biblioteca *open source*, curva de aprendizado baixa comparado às demais tecnologias disponíveis no mercado de trabalho, componentização e reutilização de código.

4.3 Cliente Mobile

O desenvolvimento do aplicativo foi utilizado a tecnologia também desenvolvida pela a empresa Facebook, nomeada de React Native. O React Native irá permitir com que possamos desenvolver uma aplicação nativa e multiplataforma, ou seja, no final do desenvolvimento do aplicativo poderá ser realizada a *build* tanto para Apple (IPK) quanto para Android (APK). O cliente *mobile* ficará responsável por renderizar a localização do *pet* de acordo com o filtro, após clicar no *pet* haverá uma outra tela na qual conterà todas as informações necessárias referente ao *pet* e ao doador responsável pelo o mesmo.

4.3.1 React Native

React Native combina as melhores partes do desenvolvimento nativo com React, a melhor biblioteca JavaScript da classe para construir interfaces de usuário. React Native é escrito em JavaScript, porém renderizado com código nativo, ou seja, o React native é renderizado para a IU da plataforma nativa, o que significa que seu aplicativo usa as mesmas APIs de plataforma nativa que outros aplicativos usam.

É possível criar versões de componentes específicos da plataforma para que uma única base de código possa compartilhar o código entre as plataformas. Com o React Native, uma equipe pode manter duas plataformas e compartilhar uma tecnologia comum, ou seja, React.

O React Native permite que você crie aplicativos verdadeiramente nativos e não comprometa a experiência dos usuários. Ele fornece um conjunto de plataformas componentes nativos agnósticos como *View*, *Text Image* que o mapa diretamente para blocos de UI construção nativa da plataforma.

4.3.2 Build

O processo de *Build* ou também conhecido como empacotamento do projeto é um dos últimos processos a ser realizado antes de publicar o aplicativo na playstore. Nesta fase poderá ser realizada a *build* tanto para Apple (IPK) quanto para Android (APK) e o processo de build ocorre quando há uma nova versão do código, ou seja, uma alteração no código fonte da aplicação.

Geralmente este processo é manual, mas podendo ser automatizado em caso de necessidade extrema, ou seja, duas ou mais *builds* por dia. O processo irá gerar uma versão compilada da aplicação na qual é hospedada em um servidor dedicado.

5 Resultados

Avalia-se que este projeto trouxe bons resultados. Primeiramente do ponto de vista acadêmico, pois a proposta de trazer as tecnologias Node.js, React JS e React native, para uma aplicação desenvolvida na Universidade, já é um grande desafio.

Observando o caráter social do aplicativo, tem-se uma contribuição muito importante para auxiliar diversos usuários nos quais gostariam de ajudar um animal de rua, possibilitando um canal de fácil visualização e utilização, para encontrar o antigo ou novo dono.

Por fim, o aplicativo inicial abriu portas para projetos futuros, não só voltados à adoção, mas também à contribuição para esses seres que merecem uma vida melhor.

6 Discussão

Pode-se notar que o aplicativo possibilita proporcionar praticidade e facilidade para o usuário na busca por algum animal perdido ou para adoção. Visto que o aplicativo mapeia a localização de onde o *pet* está, disponibiliza a rota pelo Google Maps e também traz informação para contato, tal como o *e-mail* e o redirecionamento para o WhatsApp de quem realizou o cadastro do *pet*.

Um ponto importante que pode ser citado é a interface amigável que o *software* apresenta, esta foi totalmente desenvolvida com foco em uma fácil usabilidade, pensando em todos os níveis de conhecimento dos usuários.

7 Conclusão

O presente trabalho apresentou uma demonstração de aplicativo para mapear animais perdidos ou disponíveis para adoção. Tendo como objetivos principais propor uma solução para acolher e achar novos lares para animais de rua, assim como ajudar os donos a acharem seus pets perdidos. Para isso, foram utilizadas tecnologias atuais do mercado de desenvolvimento de *software* que são: ReactJS, Node.js e React Native.

Portanto, ressalta-se assim, a importância desse aplicativo com cunho social, possibilitando a todos os usuários encontrar seu *pet* perdido ou algum *pet* para adoção.

Referências

ARAÚJO, Taís Montenegro. **A responsabilidade do município na proteção e guarda dos animais abandonados**. Orientador: Jossano Mendes de Amorim. 2016. 44 f.

Dissertação (Trabalho de Conclusão de Curso em Direito) – Faculdade de Direito, Universidade Estadual da Paraíba, Guarabira, 2016.

CAMPOS, Diego P. G. et al. APLICAÇÃO DE ARMAZENAMENTO EM NUVEM UTILIZANDO A PLATAFORMA NODE.JS. **RE3C - Revista Eletrônica Científica de Ciência da Computação**, v. 11, n. 1, 2016. Acesso em: 05 out. 2021.

GIGAOUTLET. **O que é uma API e para que serve? Cinco perguntas e respostas.** Disponível em: <https://gigaoutlet.com.br/o-que-e-api-e-para-que-serve-cinco-perguntas-e-respostas-utilitarios-blog-gigaoutlet/>. Acesso em: 11 nov. 2021.

HERTER Duda. **Kanban, o que é, como usar + exemplo para aplicar.** Curitiba: Planning Is Cool, 2021. Disponível em: <https://www.planningiscool.com.br/kanban/>. Acesso em: 07 nov. 2021.

MELO, D. **O que é TypeScript?** Disponível em: <https://tecnoblog.net/426754/o-que-e-typescript-guia-para-iniciantes>. Acesso em: 05 out. 2021.

PORTAL COMUNICARE. **229 mil cães circulam pelas ruas de Curitiba.** Disponível em: <http://www.portalcomunicare.com.br/229-mil-caes-circulam-pelas-ruas-de-curitiba/>. Acesso em: 07 nov. 2021.

SILVA FILHO, Gerson Luis Ferreira da. **Desenvolvimento de aplicativo para adoção de animais abandonados utilizando a linguagem de programação kotlin e programação reativa.** Orientador: Rubens Alexandre de Faria. 2017. 64 f. Trabalho de Conclusão de Curso (Engenharia da Computação) - Departamento Acadêmico de Eletrônica, Departamento Acadêmico de Informática, Universidade Tecnológica Federal do Paraná, Curitiba, 2017. Disponível em: http://repositorio.roca.utfpr.edu.br/jspui/bitstream/1/8462/1/CT_COENC_2017_1_4.pdf. Acesso em: 02 de nov. 2021.

SILVA, M. E. T. da. **Resumo executivo do projeto rede de defesa e proteção animal da cidade de Curitiba.** 2006.

VICENTE, Marcos Xavier. Fim da carrocinha poupa 15 mil cachorros por ano em Curitiba. **Gazeta do Povo**, Curitiba, 11 jun. 2007. Vida e Cidadania. Disponível em: <https://www.gazetadopovo.com.br/vida-e-cidadania/fim-da-carrocinha-poupa-15-mil-cachorros-por-ano-em-curitiba-aibmur9pkyek23kwkjp95ja6m/>. Acesso em: dez. 2021.