

UATracking: Uma ferramenta de *Web Tracking* usando Redis

Gian Paulo Vieira¹, Edimar Manica¹

¹Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul - *Campus Ibirubá*
Rua Nelsi Ribas Fritsch, 1111 – CEP: 98200-000 – Ibirubá – RS – Brasil

Abstract. *The advancement of technology has enable companies and individuals to earn high visibility on the Internet for their business. In this scenario, these groups feel the need to adapt and personalize their websites, so that they can remain competitive. In order for such sites to be evaluated, it is necessary that the capture of the user actions don't interfere with the site usage. This paper proposes a tool to collect and monitor user actions. This tool posses flexible integration in HTML and works regardless of the language and database used in the back-end, with the possibility of scalability, with the usage of NoSQL database and containers. This paper also describes a case study to evaluate the tool with real users.*

Resumo. *O avanço da tecnologia possibilita que empresas e indivíduos possam ter alta visibilidade na Internet para os seus negócios. Neste cenário esses grupos sentem a necessidade de se adaptar e personalizar seus sites, para que possam se manter competitivos. Para que tais sites possam ser avaliados, é necessário a captação das ações dos usuários, de forma que tal coleta não interfira na interação do usuário. Nesse contexto, este trabalho de conclusão propõe uma ferramenta de coleta e armazenamento das ações de usuários. Essa ferramenta é de integração flexível no HTML e independente da linguagem e banco de dados utilizados, com a possibilidade de escalabilidade, pois utiliza banco de dados NoSQL e arquitetura de containers. Este trabalho também descreve um estudo de caso para avaliar a ferramenta com usuários reais.*

1. Introdução

Ao longo do ano de 2020, o mundo tem sofrido várias mudanças em decorrência da pandemia do coronavírus (SARS-CoV-2). O jeito como as pessoas trabalham, estudam, se divertem e fazem compras mudou. Vários negócios tem feito uma transição parcial ou total para a esfera online. Uma das consequências desse cenário é o crescimento de 47% do comércio eletrônico no Brasil no primeiro semestre do ano, segundo pesquisa da EBIT/Nielsen (EBIT I NIELSEN, 2020).

A pandemia de COVID-19 ocasionou o aumento das vendas online, incluindo novos usuários, que encontraram no comércio eletrônico uma maneira de manter o isolamento. Essa situação gerou novos desafios para as empresas de comércio eletrônico. Por exemplo, um usuário acessa pela primeira vez um determinado site de comércio eletrônico buscando por uma Webcam. O primeiro elemento que o usuário vê é a página inicial com a seção de produtos mais vendidos do site, porém, a Webcam não se encontra nesta seção. O usuário tenta utilizar os filtros, mas esses possuem muitos campos de preenchimento, que acabam por confundi-lo. Ao tentar pesquisar pelo nome específico do produto, o usuário também não o encontra, devido à digitação equivocada e/ou no campo

incorreto. O site é incapaz de identificar os produtos relevantes para o usuário, que acaba por desistir da navegação.

Para evitar essa situação, o administrador pode utilizar técnicas de *Web Tracking*, que permitem o rastreamento das ações dos usuários. A partir dos dados sobre essas ações, é possível analisar o comportamento dos usuários nos sites, como, por exemplo, os elementos que o usuário está utilizando, os erros que comete e as diferenças entre o uso esperado e o observado. Essa análise permite aperfeiçoar o site para melhorar a experiência do usuário.

Para que o rastreamento das ações do usuário não influencie no desempenho do site, é necessário que esses dados sejam coletados e armazenados independentemente das demais ações executadas. Por exemplo, o armazenamento das ações monitoradas não deve afetar o tempo de resposta das requisições do site. Uma ferramenta de rastreamento deve possuir flexibilidade para ser integrada independente da linguagem *Front-end* e *Back-end* utilizado no site monitorado. Além disso, a ferramenta deve possuir baixo acoplamento com o site, ou seja, devem haver poucas dependências de código a fim de obter ganhos de legibilidade e manutenibilidade.

Nesse contexto, este trabalho propõe a criação de uma ferramenta de *Web Tracking*, denominada *UATracking*, que utiliza um banco de dados *NoSQL* (*Not only SQL* - Não apenas SQL) com o objetivo de garantir a escalabilidade. Esta ferramenta possui flexibilidade e baixo acoplamento, o que permite a sua utilização em qualquer site independente das tecnologias utilizadas. Por fim, esta ferramenta foi testada utilizando um site de anúncios fictício com usuários reais por meio de um estudo de caso.

O restante deste trabalho está organizado da seguinte forma. Na Seção 2, são definidos os conceitos necessários para compreensão deste trabalho. Na seção 3, são apresentados três trabalhos relacionados, demonstrando as semelhanças e diferenças em relação ao presente trabalho. A seção 4 descreve a ferramenta proposta. Na seção 5, é detalhado um estudo de caso realizado neste trabalho. Por fim, na seção 6, são apresentadas as conclusões e os trabalhos futuros.

2. Fundamentação Teórica

Esta seção descreve os principais conceitos necessários para a compreensão deste Trabalho de Conclusão de Curso. A Subseção 2.1 aborda o conceito de *Web Tracking*, enquanto que a Subseção 2.2 descreve as questões de privacidade que surgem com o uso do *Web Tracking*. A Subseção 2.3 especifica o conceito de banco de dados *NoSQL*, bem como descreve os principais tipos.

2.1. Web Tracking

Web Tracking é a atividade de coleta de dados de usuários de um site para a análise de seus comportamentos e padrões. Essa coleta é feita através de ferramentas específicas que obtêm dados como, por exemplo, em quais links o usuário clicou (ERMAKOVA et al., 2018).

Esses dados podem ser usados para uma variedade de análises, sendo comumente utilizados para avaliar o padrão de navegação de um usuário para fins comerciais de propaganda e publicidade. Por exemplo, para verificar se o *layout* de um site possui uma

boa usabilidade, é possível utilizar uma ferramenta que colete as rotas e os tempos que o usuários levam para chegar em uma determinada página. Com essa informação, é possível verificar a necessidade de atalhos e/ou mudanças na navegação do site. *Web Tracking* pode ser utilizada também para a análise detalhada de testes A/B. Um teste A/B é um experimento para analisar o comportamento dos usuários em duas versões diferentes de uma mesma página, podendo assim serem feitas análises de conversão e melhoria, por exemplo, qual página foi mais acessada (Neil Patel, 2020).

Existem ferramentas para *Web Tracking* já disponíveis no mercado como a GTM (*Google Tag Manager* - Gerenciador de Tags do Google)¹. O GTM é uma ferramenta do *Google* para automatizar e gerenciar a inserção de *tags* de monitoramento. Esta ferramenta pode monitorar as visitas a um site, assim como pode monitorar visitas em páginas específicas dentro dele. O GTM oferece também suporte a *tags* de *remarketing* como *Google ADS* e *Facebook Ads*, podendo assim monitorar quais produtos foram acessados para gerar propagandas relacionadas a esse produto. Ferramentas como o GTM facilitam a coleta de dados, ajudando na geração de ações de *marketing* precisa e conhecimento sobre perfil dos usuários dos sites monitorados. Essa ferramenta demonstra a relevância do tema.

Especificamente, para este Trabalho de Conclusão de Curso, é utilizada uma versão modificada da biblioteca *open-source Ahoy.js*² desenvolvida em JavaScript. Essa biblioteca permite a captação de cliques, acessos e submissões de formulários, porém não possui armazenamento dos dados coletados ou relatórios de análise. As funcionalidades de armazenamento e relatórios devem ser implementadas pelo administrador que utilizar em seu site. A ferramenta proposta utiliza a biblioteca *Ahoy.js* para a captura das ações dos usuários, armazena essas ações em um banco de dados *NoSQL* e disponibiliza os dados para a geração de relatórios.

2.2. Privacidade

Devido à natureza da coleta de dados e rastreamento de ações em *Web Tracking*, surgem as questões relacionadas à privacidade. Por um lado, organizações desejam obter o máximo possível de informações de seus usuários para análises de objetivo comercial. Por outro lado, os usuários se veem tendo suas experiências na *Internet* sendo limitadas devido às análises de comportamento, como suas vidas pessoais sendo analisadas sem consideração ou consentimento (BUJLOW et al., 2017).

São inúmeros os métodos que sites utilizam para identificar e coletar dados de seus usuários, sendo que os mais comuns são os *Cookies*, que são considerados variáveis de sessão, metadados do celular, dados únicos anexados a *URL*. Dados como, idade, faixa salarial, informações sobre a família e hábitos diários, podem ser coletados. Esses dados são utilizados nos sites, diretamente acessados, podendo ser eletronicamente compartilhados com terceiros, como agências de publicidade e sites de comércio, que armazenam os dados para o desenvolvimento de publicidade. A verificação de credibilidade financeira e determinação de preços do comércio eletrônico também podem ser analisadas. Há também a coleta governamental destes dados para vigilância de seus cidadãos, assim como

¹Disponível em: <https://marketingplatform.google.com/intl/pt-BR_br/about/tag-manager/>. Último acesso em: 26/12/2020

²Disponível em: <<https://github.com/ankane/ahoy.js>>. Último acesso em: 09/12/2020

hackers podem coletar dados para descobrir informações pessoais para fins ilícitos.

Existem alguns métodos de bloqueio parcial ou total da prática de *Web Tracking*:

- **Bloqueio de serviços de anúncios** - Usando ferramentas e extensões do navegador para bloquear anúncios e sites conhecidos por coletar dados;
- **Desativação de suporte ao JavaScript** - Desabilita completamente ferramentas que fazem uso do JavaScript para coletar dados;
- **Ferramentas específicas para navegação anônima** - O uso de ferramentas que podem criar canais seguros de comunicação como VPN (*Virtual Private Network* - Rede Privada Virtual) ou *proxys* anônimos. Cita-se também o uso de navegadores e sites de busca que não salvem ou passem informações privadas;
- **Remoção de *cookies* e histórico de navegação** - O usuário pode excluir *cookies* salvos no navegador, porém estes apenas funcionam com usuários não logados, uma vez que os sites podem estar guardando informações via servidor;

Nos últimos anos, organizações políticas como a União Europeia com a GDPR (*General Data Protection Regulation* - Regulamento Geral sobre a Proteção de Dados) têm movido ações e propostas para a proteção da privacidade (União Européia, 2016). No Brasil, foi sancionada a LGPD (Lei Geral de Proteção de Dados Pessoais) que regula as atividades de tratamento de dados pessoais (Brasil, 2018).

Essas leis ditam principalmente sobre os seguintes pontos: (1) Divulgar quais dados estão sendo armazenados, por qual motivo e obter o consentimento dos usuários; (2) Permitir que os usuários possam ver, editar, excluir suas informações e serem informados sobre manipulação e ou vazamento de suas informações. Esses pontos determinam que qualquer site que implemente coleta de dados deve informar seus usuários e fornecer meios de visualização e edição destas informações.

O administrador que utilizar a ferramenta UATracking deve informar aos usuários do site monitorado que suas ações estão sendo monitoradas. Devendo informar que informações, como, por exemplo, IP (*Internet Protocol address* - Endereço de Protocolo da Internet), páginas acessadas e utilização de elementos do site estão sendo coletadas e armazenadas.

2.3. Bancos de Dados *NoSQL*

Banco de dados *NoSQL* é um conjunto de tecnologias de armazenamento de dados que implementam conceitos de alta velocidade para escrita e leitura rápidas e a não obrigatoriedade de estruturação dos dados, possibilitando mudanças dinâmicas e escalabilidade horizontal descentralizada, ou seja, a estrutura do banco de dados pode ser facilmente replicada para outros computadores, tanto para *backups* como para aumentar o desempenho. Por não possuir um esquema rígido para modelagem dos dados, bancos de dados do tipo *NoSQL* requerem que grande parte da interpretação dos dados seja feita na aplicação. Este conceito de banco de dados não garante algumas característica como consistência e não-redundância de dados, um sacrifício em prol de uma alta velocidade de operações e fácil escalabilidade (DAVOUDIAN; CHEN; MENGCHI, 2018).

De acordo com Freitas, Souza e Salgado (2016), os quatro tipos básicos de banco de dados *NoSQL* são:

- **Chave-valor** - é o banco de dados mais simples, onde os valores são ligados a uma chave única, não possui nenhum tipo de estrutura adicional para organizar os dados. Um exemplo dessa categoria é o *Redis* (REDIS LABS, 2020);
- **Orientado a documentos** - são usadas estruturas chamadas documentos que guardam valores em chave-valor no formato *JSON* (*JavaScript Object Notation* - Notação de Objeto em JavaScript), sendo um documento uma coleção de objetos relacionados a uma informação. O SGBD (Sistemas de Gerenciamento de Banco de Dados) *MongoDB* (MONGODB INC, 2020) é um exemplo dessa categoria;
- **Orientado a grafos** - são utilizados nodos para armazenar os dados, sendo que esses possuem conexões que mantêm as relações de interesse. Um exemplo de aplicação desse banco de dados é para armazenar relações de amizade em redes sociais, e o *Neo4j* (NEO4J, 2020) é um exemplo desse tipo de banco de dados;
- **Orientado a colunas** - são utilizadas famílias de colunas para organizar dados semelhantes, possuindo similaridades com o modelo relacional, por organizar dados em linhas e colunas. O *Cassandra* (Apache Software Foundation, 2020) pode ser citado como um banco de dados desse tipo.

Neste trabalho de conclusão, é utilizado um banco de dados do tipo chave-valor com o SGBD Redis por oferecer alta velocidade de escrita devido a sua estrutura simples de chave-valor. O Redis é capaz de trabalhar com vários tipos de dados, desde tipos básicos até vetores, listas, *hashes*, e tipos especiais de dados para arquivos como *bitmap* e *hyperlog*. Construído com foco no desenvolvimento rápido de aplicações, o *Redis* inclui diversos módulos como, por exemplo, módulo de pesquisa *full-text*, uso de *JSON* como dado nativo e uso de matrizes e gráficos.

Um dos principais diferenciais quando comparado com outros SGBDs é sua capacidade de oferecer armazenamento em memória RAM (*Random Access Memory* - Memória de acesso aleatório) ao mesmo tempo que grava os dados de forma permanente automaticamente, podendo assim oferecer uma alta velocidade (REDIS LABS, 2020). Esse ponto é interessante para ferramentas de *Web Tracking*, pois é necessário que a armazenagem não exija demasiado processamento do sistema, durante a gravação e leitura.

3. Trabalhos Relacionados

Esta seção descreve os trabalhos relacionados. Foram selecionados três trabalhos: um que realiza rastreamento visual, outro que faz rastreamento de ações em dispositivos móveis e o último que realiza o rastreamento de ações em sites Web. Esses trabalhos foram obtidos a partir de uma busca pela expressão “Web Tracking” no *Google Acadêmico*³.

Dong *et al.* (2018) propõem um processo de combinação de questionários e rastreamento visual para determinar a usabilidade e legibilidade de mapas de fluxo. Nesse trabalho os autores utilizaram o equipamento *Tobii T120 Eye Tracker*⁴ para captar o movimento dos olhos do usuário com o objetivo de determinar o fluxo de visualização do mapa e áreas de interesse. Em conjunto com o *Tobii T120 Eye Tracker*, foram utilizados questionários manuais para avaliar os resultados da interpretação dos usuários sobre os referidos mapas. Esse trabalho concluiu que mapas de fluxo que utilizam linhas curvas

³Disponível em:<<https://scholar.google.com.br/?hl=pt>>, Último acesso em: 09/12/2020

⁴Disponível em:<<https://www.tobii.com/>>, Último acesso em: 09/12/2020

possuem maior precisão de acerto, assim como foi possível observar que linhas com cores diferentes são mais fáceis de serem interpretadas do que linhas de diferentes larguras.

Em Soojin, Sungyong e Kyeongwook (2018), foi proposto um processo para detecção de erros de usabilidade em dispositivos móveis. Para isso foram utilizados *frameworks* existentes para criar um processo de leitura de *logs* do uso do aplicativo. Este processo foi capaz de gerar modelos de atividades dos usuários, os quais foram comparados com os modelos esperados de uso criados pelos desenvolvedores. Esse processo foi capaz de não só detectar erros de uso como detectar problemas de usabilidade na aplicação.

No trabalho de Solís-Martínez *et al.* (2020), é descrita uma ferramenta baseada em *JavaScript uxJS*⁵ capaz de coletar as interações do usuário com determinado site. Essa ferramenta utiliza o NPM (*Node Package Manager* - Gerenciador de pacotes do Node) para sua instalação. Baseando-se nos conceitos de velocidade, facilidade de uso, foi criado um processo de coleta de informações para a geração de relatórios automatizados com análises quantitativas. A ferramenta é capaz de coletar cliques, ações de rolagem do mouse, tempo de visita em cada página do site e armazena estas informações em um banco de dados. Essas informações ficam disponíveis em uma tela especial para o administrador do site monitorado visualizar em forma de relatórios.

Ao contrário de Dong *et al.* (2018), a ferramenta *UATracking* não utiliza tecnologias e métodos que não possam ser facilmente incorporados a um site ou equipamentos avançados, como para o rastreamento visual. O trabalho de Soojin, Sungyong e Kyeongwook (2018) é voltado para a área *mobile*, enquanto este trabalho propõe uma ferramenta de utilização em sites Web, que pode ser também utilizada em navegadores *mobile*. A maior similaridade deste trabalho é com o trabalho de Solís-Martínez *et al.* (2020) que propõe um processo similar, porém este artigo descreve a ferramenta *UATracking*, cuja maior diferença é a possibilidade de descrever seções específicas de elementos HTML (*HyperText Markup Language* - Linguagem de Marcação de Hipertexto) que devem ser monitorados e o baixo acoplamento, podendo ser utilizada independente da tecnologia do site monitorado.

4. UATracking

Esta seção descreve a ferramenta *UATracking* que realiza o rastreamento de ações do usuário em um site Web. A Subseção 4.1 descreve a arquitetura da ferramenta. Por fim, a Subseção 4.2 descreve as instruções para instalação, configuração e implementação da *UATracking*.

4.1. Arquitetura

A ferramenta *UATracking* rastreia as ações do usuário por meio de uma versão adaptada da biblioteca de *Web tracking* *Ahoy.js* e armazena os registros dessas ações utilizando o banco de dados *NoSQL* Redis. A biblioteca *Ahoy.js* foi escolhida por ser constituída de um único arquivo capaz de ser incluído em qualquer site, independentemente da tecnologia *back-end* do mesmo. Além disso, *Ahoy.js* pode ser configurada para captar dados dos eventos da página, tais como mudanças e submissão de formulários, cliques

⁵Disponível em:<<https://github.com/7891011/uxjs>>. Último acesso em: 09/12/2020

em botões e links, páginas acessadas, além do tempo entre todas essas ações. Também, a configuração desta biblioteca requer a inclusão de um único arquivo no *template* principal do site.

Foi escolhido trabalhar com o Redis devido ao mesmo possuir código livre, sendo capaz de armazenamento em memória e persistência de dados. Um banco de dados de chave-valor se faz útil para atingir o objetivo de alta velocidade de escrita, para que a etapa de gravação dos dados não comprometa o desempenho do site.

Na Figura 1, é ilustrado a visão geral da *UATracking*. Um usuário executa uma ação no site monitorado, por exemplo, um clique. A pela ferramenta *UATracking* detecta a ação. Caso a ação esteja configurada para ser rastreada, a ferramenta irá armazenar as informações sobre essa ação. O administrador do site monitorado pode consultar essas informações através dos relatórios disponibilizados pela *UATracking*.

A Figura 2 apresenta a arquitetura da ferramenta proposta. Destaca-se que essa arquitetura possui dois componentes principais: (i) o site monitorado e (ii) a ferramenta *UATracking*. O site monitorado pode utilizar qualquer linguagem de programação e SGBD. A ferramenta *UATracking* utiliza a estrutura de *containers* através do Docker⁶ para facilitar a instalação e configuração, possuindo um servidor Web Apache, o SGBD Redis, linguagem de programação PHP (*Hypertext Preprocessor* - Pré-processador de hipertexto) e *framework* Lumen⁷, que é utilizado para receber, gravar e retornar os dados monitorados.

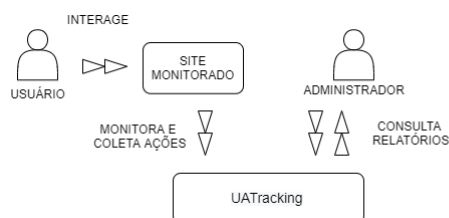


Figura 1. Visão geral da UATracking

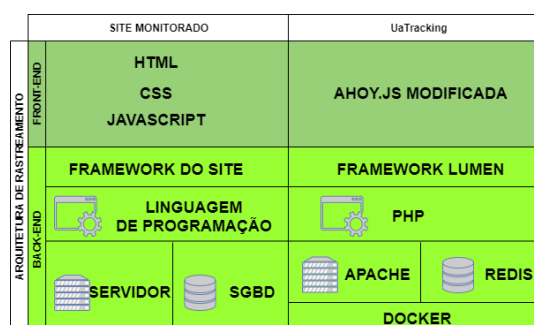


Figura 2. Arquitetura da UATracking

O Docker foi utilizado para simplificar o desenvolvimento de aplicações através do uso de *containers*, permitindo a instalação de todas as dependências de uma aplicação, de forma autônoma do restante do sistema. Dentre os arquivos da ferramenta se encontra uma versão modificada da biblioteca *Ahoy.js*, que foi alterada para incluir a configuração de itens monitorados, utilizado para identificar as seções de elementos HTML nos relatórios e a função de consulta de relatórios.

Quando um usuário executa uma ação que tenha sido configurada para ser monitorada, a *UATracking* coleta as informações dessa ação, a converte no formato JSON e envia para a camada de *back-end*. No *back-end*, esse arquivo JSON é recebido pelo *framework* Lumen, lido e os dados são armazenados no banco de dados Redis. Os dados armazenados no Redis são posteriormente acessados pelas rotas configuradas para os re-

⁶Disponível em: <<https://www.docker.com/>>. Último acesso em: 30/12/2020

⁷Disponível em: <<https://lumen.laravel.com/>>. Último acesso em: 09/12/2020

latórios, para que então possam fornecer informações aos administradores sobre as ações realizadas pelos usuários no site.

As seguintes ações são monitoradas na ferramenta proposta: acessos às páginas, cliques em filtros, cliques em paginações e submissões de formulários de busca. De cada ação, são coletadas as seguintes informações: tipo e horário da ação, link da página onde ocorreu a ação e atributos de identificação do elemento HTML que disparou a ação.

4.2. Implementação

Para utilizar a ferramenta *UATracking* é necessário realizar as seguintes etapas: instalação de serviços Web de monitoramento, configuração de itens monitoráveis, configuração de páginas monitoráveis e configuração de exibição dos relatórios. A seguir cada etapa é descrita em uma subseção.

4.2.1. Instalação de Serviços Web de Monitoramento

O objetivo dessa etapa é instalar os serviços Web de monitoramento. Esses serviços são responsáveis por receber os dados rastreados e armazená-los em um banco de dados NoSQL, bem como fornecer os dados para relatórios.

Para a instalação desses serviços, é necessário instalar o Docker e fazer o *download* da *UATracking*⁸. O *download* pode ser feito pelo navegador Web. Nesse caso, é necessário baixar o código-fonte compactado e descompactá-lo em qualquer diretório do sistema operacional utilizado. Também, é possível realizar o *download* via ferramenta de versionamento GIT⁹, utilizando o seguinte comando:

```
git clone git@gitlab.com:gian.vieira/tracking_api.git
```

O próximo passo é configurar as portas do Apache e do Redis, bem como a senha do Redis. Por padrão, o Apache está configurado nas portas 81 e 8443, enquanto o Redis está configurado na porta 6379. É necessário apenas modificar essas portas se elas já estiverem em uso por outro serviço. Além disso, o Redis vem configurado com a senha “api”. Para garantir a segurança do sistema, recomenda-se fortemente a sua substituição por outra mais forte. A configuração das portas e da senha é realizada modificando os seguintes arquivos: (i) arquivo “.env” localizado na pasta raiz do projeto baixado, exemplificado na Figura 3; e (ii) arquivo “.env” localizado na pasta `www/tracking`, exemplificado na Figura 4. No primeiro arquivo, as linhas 8 e 9 referem-se às portas do Apache, enquanto a linha 10 refere-se à porta do Redis e a linha 13 define a senha desse banco de dados. No segundo arquivo, a linha 13 refere-se à porta do Redis, enquanto a linha 15 define a sua senha.

⁸Disponível em: <https://gitlab.com/gian.vieira/tracking_api>. Último acesso em: 09/12/2020.

⁹Disponível em: <<https://git-scm.com/>>. Último acesso em: 09/12/2020.


```

1 DOCUMENT_ROOT=../www
2 VHOSTS_DIR=./config/vhosts
3 APACHE_LOG_DIR=./logs/apache2
4 PHP_INI=./config/php/php.ini
5
6
7
8 HOST_MACHINE_UNSECURE_HOST_PORT=81
9 HOST_MACHINE_SECURE_HOST_PORT=4443
10 HOST_MACHINE_REDIS_PORT=6379
11
12 # Senha do redis
13 REDIS_PASSWORD=api
14

```

Figura 3. Arquivo .env da pasta raiz do ferramenta

```

1 APP_NAME=Tracking
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6 APP_TIMEZONE=UTC
7
8 LOG_CHANNEL=stack
9 LOG_SLACK_WEBHOOK_URL=
10
11 DB_CONNECTION=redis
12 DB_HOST=redis
13 DB_PORT=6379
14 DB_DATABASE=0
15 DB_PASSWORD=api
16
17 CACHE_DRIVER=redis
18 QUEUE_CONNECTION=sync

```

Figura 4. Arquivo .env da pasta www/tracking

Após executadas as instruções descritas, deve-se executar o seguinte comando dentro da pasta raiz do projeto para realizar a instalação inicial:

```
docker-compose up -d --build
```

Caso o servidor seja reiniciado, deve-se apenas executar o seguinte comando dentro da pasta raiz do projeto para iniciar o serviço:

```
docker-compose up -d
```

4.2.2. Configuração de Itens Monitoráveis

Essa etapa tem o objetivo de informar para a ferramenta como os itens que devem ser monitorados estão implementados no site. Mais especificamente, é necessário informar os elementos utilizados no site para filtros, campo de busca, carrossel e paginação, bem como a página e o elemento utilizados para descrever os produtos. Essa configuração permite maior flexibilidade ao site, uma vez que não exige a utilização de elementos específico e permite identificar o contexto da ação realizada pelo usuário.

Para realizar essa configuração, é necessário criar um arquivo *Javascript* e configurar a função *ahoy.configure()*. Os seguintes argumentos precisam ser configurados:

1. **urlPrefix** - o IP e a porta do servidor onde os serviços Web de monitoramento foram instalados;
2. **trackingId** - o valor do atributo *id* do elemento HTML que contém a descrição do produto na página que o descreve;
3. **trackingIdElem** - se a descrição do produto está no texto do elemento deve-se definir o valor *text*, se a descrição do produto está no valor de um atributo, especificar o nome desse atributo;
4. **filtros** - o valor do atributo *data-section* do elemento que engloba os botões de navegação e as categorias de produtos;
5. **campo_busca** - o valor do atributo *data-section* do elemento que contém os campos de busca que devem ser monitorados;

6. **carrossel** - o valor do atributo *data-section* do elemento que contém o carrossel de produtos;
7. **paginacao** - o valor do atributo *data-section* do elemento que engloba a paginação dos resultados da busca;
8. **pagina_produtos** - o nome da página que descreve um anúncio de produto selecionado pelo usuário. É necessário que o site contenha uma página que exibe detalhes de um produto selecionado pelo usuário.

A Figura 5 apresenta a configuração dos argumentos da função *ahoy.configure()* a partir do site utilizado como exemplo.

```

9  ahoy.configure({
10  urlPrefix: "http://localhost:81",
11  trackingId: "tracking_anuncio",
12  trackingIdElem: "text",
13  filtros: "filtros",
14  campo_busca: "campo_busca",
15  carrossel: "carrossel",
16  paginacao: "paginacao",
17  pagina_produtos: "pagina_anuncio",
18  });

```

Figura 5. Exemplo de configuração dos argumentos da função

A seguir é apresentado um exemplo de site e a configuração de seus itens monitoráveis. São exibidas três páginas do site: a página principal (Figura 6), a página de resultados (Figura 7) e a página de detalhe do produto selecionado (Figura 8). Para cada página, são apresentados os trechos de código HTML relevantes para a configuração e destacadas em negrito as informações que devem ser utilizadas para definir os argumentos da função *ahoy.configure()*.

A Figura 6 ilustra a página principal do site, onde o usuário pode encontrar os produtos por meio de um campo de busca (Área 01), de um carrossel de produtos (Área 02) ou dos filtros (Área 03). Para cada área, é necessário identificar o valor do atributo *data-section* do elemento que engloba todos os elementos que compõem a área.

The image shows a screenshot of a website's main page with three specific areas highlighted by yellow dashed boxes and labeled as Área 01, Área 02, and Área 03. Each area is associated with a snippet of HTML code that defines its configuration for the tracking function.

- ÁREA 01:** Points to a search bar in the top navigation. The associated code is:


```
<form data-section="campo_busca" class="navbar-form navbar-left">
...
</form>
```
- ÁREA 02:** Points to a product carousel displaying a backpack. The associated code is:


```
<div id="carousel_anuncios" data-section="carrossel" class="carousel slide"
data-ride="carousel" data-interval="2000">
<!-- Wrapper for slides -->
<div class="carousel-inner">
...
</div>
</div>
```
- ÁREA 03:** Points to a sidebar containing various filter categories. The associated code is:


```
<div data-section="filtros" class="col-sm-2" id="actions-sidebar">
<ul class="nav nav-pills nav-stacked">
...
</ul>
</div>
```

Figura 6. Exemplo de página principal

A Figura 7 ilustra a página de resultados. Apenas vinte resultados são exibidos em cada página. Por isso, há uma paginação (Área 04). É necessário identificar o valor do atributo *data-section* do elemento que engloba todos os elementos que compõem a paginação.

Anúncios

Imagem	Categoria	Título	Preço	Ações
	Trilha e Acampamento	Cartão de Sobrevivência Super Ferramenta com 11 funções	RS 28	Ver
		Camping Lazer Pesca Pescaria Aventura Acampamento		
		Escotismo Promoção Barato Frete Grátis todo Brasil Outlet Top		

<< primeiro < anterior 1 2 3 4 5 6 próximo >

Página 6 de 6, mostrando 1 registro(s) de 101


```
<div class="paginator" data-section="paginacao">
<ul class="paginator">
<?=$this->Paginator->first('<_.__(primeiro) ?>') ?>
<?=$this->Paginator->prev('<_.__(anterior) ?>') ?>
<?=$this->Paginator->numbers() ?>
<?=$this->Paginator->next('<_.__(proximo) . '>') ?>
<?=$this->Paginator->last('<_.__(ultimo) . '>>') ?>
</ul>
</div>
```

ÁREA 04

Figura 7. Exemplo de página de resultados

A Figura 8 ilustra a página de detalhe do produto selecionado, ou seja, a página com todas as informações do produto selecionado pelo usuário. É necessário identificar, nessa página, o elemento ou atributo que descreve/identifica o produto.

ID: 35



```
<tr>
<th scope="row"><?=$this->Paginator->first('<_.__(primeiro) ?>') ?>
<td id="tracking_anuncio"><?=$this->Paginator->prev('<_.__(anterior) ?>') ?>
</tr>
```

ÁREA 05

Criado por	gian.vieira
Categoria	Trilha e Acampamento
Título	Mochila Camping Trilha 70 L
Preço	113
Validade	1/1/20

Figura 8. Exemplo de página de detalhe do produto selecionado

4.2.3. Configuração das páginas monitoráveis

Essa etapa tem o objetivo de configurar o site cliente para que possa monitorar os itens configurados na etapa anterior. A arquitetura do processo de monitoramento foi planejada para requerer a menor alteração possível no site alvo do rastreamento. Especificamente, em cada página que deve ser monitorada, é necessário apenas importar a biblioteca Ahoy.js modificada e o arquivo de configuração definido na seção anterior, bem como chamar duas funções da biblioteca.

Após a instalação do serviço, a biblioteca de rastreamento modificada fica disponível por meio do link "IP_INSTALLADO": "PORTA_INSTALLADA/tracking/public/js/ahoy.js". Este link deverá ser referenciado em cada página do site que se deseja rastrear através da tag

`<script>`. Também, é necessário importar o arquivo Javascript (definido na seção anterior) onde são especificados os itens monitorados.

Por fim, é necessário chamar duas funções da biblioteca de rastreamento modificada, como exemplificado na figura 9. A primeira função a ser chamada é a *configure*, onde é necessário definir no argumento *page* um identificador para a página a fim de permitir a identificação da página onde a ação do usuário ocorreu. A segunda função a ser chamada é a *trackAll*, que inicia o rastreamento das ações do usuário na página.

4.2.4. Configuração da exibição dos relatórios

Essa etapa tem o objetivo de consultar os dados rastreados a fim de gerar relatórios sobre a utilização do site. É possível obter informações sobre as ações dos usuários, as ações produtivas realizadas pelos usuários, os produtos mais procurados e as principais buscas improdutivas.

O relatório das ações dos usuários exibe a quantidade de vezes que os usuários utilizaram o campo de busca, os filtros, o carrossel e a paginação. Esse relatório não considera se a ação levou o usuário até um anúncio de seu interesse. A finalidade desse relatório é mostrar aos administradores quais as formas de navegação são mais procuradas pelos usuários do site. Essa informação pode orientar a reorganização do site explorando o comportamento dos usuários.

O relatório das ações produtivas realizadas pelos usuários contabiliza a forma utilizada pelos usuários para chegar até os produtos de seu interesse. Considera-se que o produto era do interesse do usuário, se ele acessou a página detalhe desse produto. Esse relatório contabiliza apenas a última ação realizada pelo usuário para chegar na página detalhe do produto. Caso essa ação seja utilizar a paginação, então contabiliza-se também a penúltima ação, de forma a identificar qual ação levou o usuário até a paginação. A finalidade deste relatório é apresentar aos administradores a forma que permitiu aos usuários encontrar os produtos de seu interesse, permitindo planejar alternativas para deixar essa forma ainda mais acessível.

O relatório dos produtos mais procurados exibe os produtos que os usuários possuem mais interesse. Considera-se que o usuário teve interesse pelo produto, se ele abriu a página detalhe que descreve o produto. Esse relatório é útil para que os gerentes de vendas possam saber quais produtos são de interesse do usuário.

O relatório de buscas improdutivas exibe as palavras-chave utilizadas em buscas que não retornaram resultados. Esse relatório é útil para que os administradores do site possam verificar quais são as dificuldades que os usuários tem para encontrar um produto de seu interesse. Além disso, permite identificar se existe a necessidade de implementar as funções de autocompletar, expansão de consultas e sugestões de correção de palavras.

Para obter os dados necessários para gerar os relatórios descritos, deve-se chamar a função "*ahoy.relatorio()*", como exemplificada no trecho de código apresentado na Figura 10, passando os seguintes argumentos:

1. **Número do relatório** - 1 para o relatório das ações dos usuários; 2 para o relatório das ações produtivas dos usuários; 3 para o relatório dos produtos mais procurados

- e 4 para o relatório das buscas improdutivas;
2. **Data de início** - data a partir da qual os dados são considerados, no formato "DD/MM/YYYY", ou o valor *null* para utilizar os dados desde o início do rastreamento;
 3. **Data de fim** - data final do período considerado para a obtenção dos dados, no formato "DD/MM/YYYY", ou o valor *null* para utilizar os dados até o momento atual;
 4. **Função *callback*** - função que recebe os dados consultados e fornece o tratamento necessário;

```

1  <script>                                ahoy.relatorio(1,null,null,function(dados){
2    ahoy.configure({page: "ver_anuncios"}) /* LÓGICA DA EXIBIÇÃO DOS DADOS */
3    ahoy.trackAll();
4  </script>                                });
5

```

Figura 9. Trecho para iniciar o monitoramento em cada página

Figura 10. Trecho de código exemplificando a consulta de dados para a geração de relatórios

Os dados dos relatórios (parâmetro **dados** da função *callback*) estão estruturados como um vetor de objetos Javascript, com as propriedades informação e quantidade, sendo respectivamente a informação dentro do contexto do relatório e a quantidade de vezes que essa informação foi contabilizada pelo relatório. É possível utilizar esse vetor para gerar relatórios de diversos formatos, como, por exemplo, *PDF*, *HTML* e *XLS* ou ser utilizado para montar gráficos utilizando bibliotecas como a *Google Charts*¹⁰.

5. Estudo de caso

Para avaliar a ferramenta proposta foi realizado um estudo de caso com usuários reais em um site de anúncios fictício utilizando um teste A/B. O site fictício foi desenvolvido com o *framework* CakePHP¹¹ utilizando um servidor Web Apache, o banco de dados relacional *MySQL*, a linguagem de programação PHP, HTML, CSS (*Cascading Style Sheets* - Folha de Estilo em Cascatas) e JavaScript. Este foi populado com anúncios retirados do site da *Amazon*¹² através de um *script* automatizado que navegou pelo site da *Amazon*, coletou o nome e a descrição dos anúncios e fez o *download* das imagens desses.

O teste A/B consistiu em dividir os acessos à página em dois *layouts* diferentes, logo, foi possível exemplificar um dos usos da utilização do *Web Tracking* por meio da ferramenta apresentada neste trabalho.

O estudo foi realizado ao longo do ano de 2020, no qual foi enviado um formulário para trinta voluntários contendo quinze anúncios que deveriam ser encontrados em um site de anúncios utilizado para este estudo de caso. Cada pessoa deveria encontrar o anúncio e preencher no formulário a *ID* do anúncio e uma breve descrição informando como este foi encontrado.

O formulário solicitava que os participantes encontrassem quinze produtos, sendo fornecida um dos três tipos de informações: (1) Apenas a imagem do produto; (2) Apenas

¹⁰Disponível em: <<https://developers.google.com/chart>>. Último acesso em: 09/12/2020

¹¹Disponível em: <<https://cakephp.org/>>. Último acesso em: 09/12/2020.

¹²Disponível: <<https://www.amazon.com.br/>>. Último acesso em: 09/12/2020.

uma descrição baseada em sinônimos; (3) O anúncio com imagem e nome. O usuário teve que encontrar o anúncio que continha cada produto no site utilizando o campo de busca, filtros de categorias e/ou carrossel.

O formulário foi limitado para que trinta pessoas respondessem, sendo divididas em quinze para o *layout A* e quinze para o *layout B*. Assim este estudo pode verificar quais elementos do site estavam sendo usados em cada *layout*. O *layout A* possui um campo de pesquisa por texto no canto esquerdo da barra de navegação superior, filtros de categorias no canto esquerdo e não contém nenhum filtro de preços na página inicial. Enquanto o *layout B* foi modificado para dar ênfase aos filtros de categorias, expondo-as na parte mais central da página inicial acima do carrossel, também adicionando filtros de preço na página principal. Neste último *layout* o campo de pesquisa por texto foi colocado para o lado mais direito da página, para que os participantes não percebessem sua presença em um momento inicial. A figura 11 exibe os resultados do teste A/B, demonstrando em porcentagem quais elementos gráficos foram utilizados em cada *layout*.

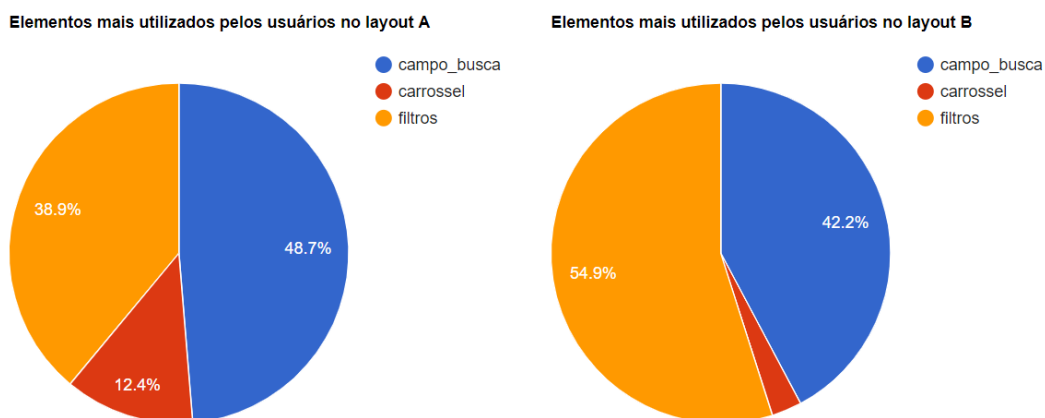


Figura 11. Resultado do teste A/B

O formulário não obteve as trinta pessoas necessárias para o teste, sendo que quinze responderam para o *layout A* e cinco responderam para o *layout B*. Apesar de não possuir um número equivalente de pessoas, é possível observar um aumento do uso de filtros no *layout B*. Sendo este o objetivo desse *layout*, de que o usuário utilizasse os filtros de categorias e/ou preços em vez do campo de busca.

6. Conclusão

O presente trabalho de conclusão de curso propôs a ferramenta *UATracking* para a coleta e armazenamento de ações dos usuários em sites de forma escalável por meio da utilização do Redis. Através da utilização da estrutura de *containers* foi possível manter um baixo acoplamento no site monitorado. Com a utilização da versão modificada da biblioteca *Ahoy.js*, foi possível manter um alto grau de flexibilidade, ou seja, é possível utilizar qualquer linguagem de programação e SGBD no site monitorado.

Um estudo de caso com usuários reais em um site fictício mostrou que a ferramenta proposta foi capaz de coletar e armazenar as ações dos usuários. Através da análise parcial do resultado, foi possível observar que houve diferenças entre os *layouts*, em um nível de elementos *HTML*, o que pode contribuir para a melhoria da usabilidade do site.

Como trabalhos futuros destacam-se: (1) a realização de experimentos com testes de carga para comprovar a escalabilidade da ferramenta; (2) a disponibilização de relatórios na própria ferramenta; (3) a disponibilização da ferramenta como um serviço de API online.

Referências

- Apache Software Foundation. *What is Cassandra?* 2020. Disponível em: <<https://neo4j.com/top-ten-reasons/>>. Último acesso em: 09/12/2020.
- Brasil. *Lei Geral de Proteção de Dados Pessoais*. 2018. Disponível em: <https://lcpd-brasil.info/http://www.planalto.gov.br/ccivil_03/_Ato2015-2018/2018/Lei/L13709.htm>. Último acesso em: 09/12/2020.
- BUJLOW, T. et al. A survey on web tracking: Mechanisms, implications, and defenses. *Proceedings of the IEEE*, v. 105, n. 8, p. 1476–1510, Aug 2017. ISSN 0018-9219.
- DAVOUDIAN, A.; CHEN, L.; MENGCHI, L. A survey on nosql stores. *Journal ACM Computing Surveys (CSUR)*, v. 51, 2018.
- EBIT I NIELSEN. *EBIT I NIELSEN: E-commerce no Brasil cresce 47% no primeiro semestre, maior alta em 20 anos*. 2020. Disponível em: <<https://static.poder360.com.br/2020/08/EBIT-ecommerce-Brasil-1semestre2020.pdf>>. Último acesso em: 09/12/2020.
- ERMAKOVA, T. et al. Web tracking – a literature review on the state of research. *Proceedings of the 51st Hawaii International Conference on System Sciences*, p. 4732–4741, 2018.
- FREITAS, M.; SOUZA, D.; SALGADO, A. Conceptual mappings to convert relational into nosql databases. *Proceedings of the 18th International Conference on Enterprise Information Systems*, p. 174–181, 2016.
- MONGODB INC. *Why and When to Use MongoDB*. 2020. Disponível em: <<https://www.mongodb.com/why-use-mongodb>>. Último acesso em: 09/12/2020.
- Neil Patel. *A Beginner's Guide To A/B Testing: An Introduction*. 2020. Disponível em: <<https://neilpatel.com/blog/ab-testing-introduction/>>. Último acesso em: 09/12/2020.
- NEO4J. *Top Ten Reasons for Choosing Neo4j*. 2020. Disponível em: <<https://neo4j.com/top-ten-reasons/>>. Último acesso em: 09/12/2020.
- REDIS LABS. *Why Redis*. 2020. Disponível em: <<https://redislabs.com/why-redis/>>. Último acesso em: 09/12/2020.
- SOOJIN, P.; SUNGYONG, P.; KYEONGWOOK, M. An automatic user activity analysis method for discovering latent requirements: Usability issue detection on mobile applications. *Sensors*, v. 18, p. 2963, 2018. ISSN 1424-8220.
- União Européia. *Regulamento (ue) 2016/679 do parlamento europeu e do conselho*. 2016. Disponível em: <<https://eur-lex.europa.eu/legal-content/PT/TXT/HTML/?uri=OJ:L:2016:119:FULL&from=PT>>. Último acesso em: 09/12/2020.