

Automatização do Acompanhamento do Andamento de Processos e Projetos de Lei

Ary Andrade¹, Felipe Schreiber¹, Gabriel Oliveira¹,
Henrique Pan¹, Paulo Mattos¹, Victor Raposo¹

¹ Universidade Federal do Rio de Janeiro - UFRJ
Rio de Janeiro, Brazil

[{arynetto, schreiber.felipe, gabriel.oliveira}@poli.ufrj.br

[{henriquepan, paulo.mattos, victor.ravaglia}@poli.ufrj.br

Abstract. *In a law firm, one of the weekly workloads consists of monitoring lawsuits, checking for updates. In general, due to the lack of automation involved in the activity, this is done inefficiently, consulting various court sites manually. This document describes a web platform that aims to speed up and increase efficiency in the monitoring process, automating most of the workload.*

Resumo. *Em escritórios de advocacia, uma das cargas de trabalho envolve o acompanhamento de processos e projetos de lei. Em geral, devido a falta de automatização envolvida na atividade, isso é feito de forma ineficiente, consultando diversos sites de tribunais manualmente. Neste documento, é descrita uma plataforma web que visa acelerar e aumentar a eficiência no acompanhamento de processos, automatizando a maior parte da atividade.*

1. Introdução

O acompanhamento de processos e projetos de lei em firmas de advocacia é uma tarefa que em geral é feita manualmente e pode levar horas para ser concluída. O objetivo dessa atividade, é condensar as informações sobre todas as atualizações que ocorreram em um determinado período de tempo, possibilitando a elaboração de relatórios.

A aplicação descrita neste documento se refere ao Menor Produto Viável (MVP) para a solução final. Ela consiste de uma aplicação web que visa automatizar parte do processo, reduzindo a carga horária e o pessoal necessários para o acompanhamento do andamento de processos. Segue o link do repositório da aplicação: Github¹.

Este documento está dividido como se segue. A Seção 2 descreve todo o processo de acompanhamento de atualizações usualmente adotado. Na Seção 3 a plataforma proposta bem como as funções presentes no MVP são apresentadas. A Seção 4 detalha as tecnologias, metodologias e detalhes técnicos associados ao desenvolvimento da solução. Finalmente, na Seção 5, conclusões e propostas para a aplicação finalizada são discutidas.

2. Definição do Problema

No ramo jurídico, parte da carga de trabalho envolve o acompanhamento de processos, em busca de atualizações que possam indicar possíveis resoluções ou empecilhos no andamento do mesmo. Esse acompanhamento é feito visitando sites de tribunais regionais,

¹ <https://github.com/netoary/programacao-avancada>

estaduais e federais. Um advogado pode estar encarregado de mais de um cliente simultaneamente. Cada cliente pode ter um ou mais processos associados a diferentes tribunais.

A carga de trabalho convencional envolve a construção de tabelas de acompanhamento legislativo (TAL), semelhantes a apresentada na Figura 1. Nelas, são contidas as últimas atualizações de cada processo ou projeto de lei que está sendo acompanhado. Para que a tabela seja atualizada, para cada linha da tabela, o encarregado deve visitar o site do tribunal associado, se identificar ou se cadastrar quando necessário, e ler os relatórios fornecidos pelo tribunal na procura de atualizações no andamento. O objetivo final das TAL é resumir o progresso dos processos e projetos de lei para facilitar a elaboração de relatórios. As TAL podem conter dezenas de entradas, e sua atualização manual pode durar horas.

Nem todos os sites de tribunal tem a mesma interface, dificultando a navegação. Alguns tribunais oferecem a possibilidade de atualizações automáticas através de e-mails, no entanto, em geral, as repostas não são padronizadas, podendo conter todo o progresso do processo ou projeto de lei, desde seu início, em vez de só a ultima atualização. Além disso, há a necessidade de condensar informações para que as TALs sejam preenchidas, resultando na ação de usar os e-mails apenas como aviso de que há uma nova atualização, e conseqüentemente, os sites de tribunais ainda serem visitados independentemente.

| Número | Assunto | Andamento mais recente | Observação |
|------------------|-------------------------------|--|---|
| Processo nº 0059 | Questiona nova lei de IPTU/RJ | 21.10.2020 - Remessa do escrivão para 3VP (Divisão de Comunicação Externa e Gestão) 21.10.2020 - A desembargadora proferiu a decisão de não reatuação 19.10.2020 - Conclusão ao 3º Vice-Presidente para reatuação. | Publicada decisão proferida pelo relator (19.07.2020.01.2019); o município intimado para exarar par juízo de admissibilidade; 21.11.2019 foi proferida para a 3ªVP. Em 18.02.2020 Remessa do Escrivão/Dir |
| ADIN nº 5 | ISS (LC) | 23.10.2020 - Ocorreu a juntada de procuração/substabelecimento 15.10.2020 - A Prefeitura de São Paulo apresentou seus memoriais / 08.10.2020 - Conclusos ao relator após juntada petição da CONFEDERAÇÃO NACIONAL | Publicação no Dje das de Campo (20.02.2019); jur calendário de julgamento; Inclua-se em pauta, Julg; |
| ADPF 4 | ISS (LC) | 23.10.2020 - Ocorreu a juntada de procuração/substabelecimento 08.10.2020 - Conclusos ao relator após juntada petição da CONFEDERAÇÃO NACIONAL DE SAÚDE, HOSPITAIS E ESTABELECIMENTOS E SERVIÇOS | Juntada de mandado de |

Figura 1. Exemplo de uma Tabela de Acompanhamento Legislativo. As linhas representam processos ou projetos de leis sendo acompanhados. O objetivo é preencher a coluna "Observação" com uma compilação de andamentos.

3. Plataforma Proposta

Seguindo os moldes estabelecidos pelos requisitos da disciplina, a solução proposta foi uma *Single Page Application* (SPA, ou em português, aplicação de página única). O objetivo da aplicação é automatizar a extração da informação de processos e projetos de lei.

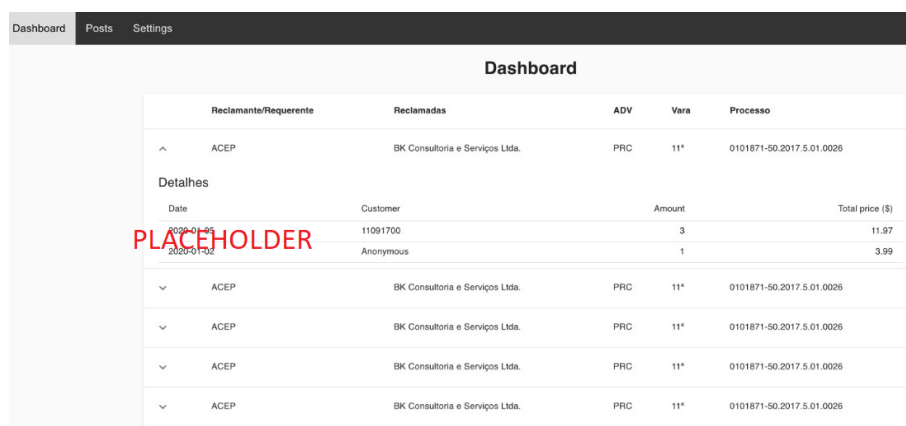
Para o MVP, haverão dois diretórios "não-cadastrados" e "cadastrados", cujo diretório é descrito com mais detalhes no *readme* do projeto. Apenas os projetos contidos na pasta "cadastrados" serão exibidos ao usuário inicialmente. A Figura 2 apresenta a página principal da aplicação, o *dashboard* (painel de controle). No *dashboard* esta contida uma tabela que concentra as informações úteis de processos e projetos de lei cadastrados. Os

processos são agrupados pelo seu respectivo número de identificação, de modo que, todas as atualizações feitas desde o cadastro ficam associadas a ele.

Para adicionar um novo processo, o usuário deve clicar no botão "adicionar processo", onde será oferecida a oportunidade de inserir o número do processo desejado. O programa irá buscar na lista de processos disponíveis no diretório "não-cadastrados" a procura de uma correspondência. E finalmente, o *dashboard* será atualizado.

Para o MVP, as seguintes funcionalidades principais estão disponíveis:

- Cadastro de processos e projetos de lei. No MVP, isso será feito de forma offline.
- Consulta de estado (movimentação) de processos cadastrados.
- Painel de controle com resumo dos processos cadastrados.
- Busca por processos cadastrados.
- Exportação das informações contidas no painel de controle para planilha.



| | Reclamante/Requerente | Reclamadas | ADV | Vara | Processo |
|----------|-----------------------|---------------------------------|-----|--------|---------------------------|
| ^ | ACEP | BK Consultoria e Serviços Ltda. | PRC | 11* | 0101871-50.2017.5.01.0026 |
| Detalhes | | | | | |
| | Date | Customer | | Amount | Total price (\$) |
| | 2020-01-05 | 11091700 | | 3 | 11.97 |
| | 2020-01-02 | Anonymous | | 1 | 3.99 |
| ▼ | ACEP | BK Consultoria e Serviços Ltda. | PRC | 11* | 0101871-50.2017.5.01.0026 |
| ▼ | ACEP | BK Consultoria e Serviços Ltda. | PRC | 11* | 0101871-50.2017.5.01.0026 |
| ▼ | ACEP | BK Consultoria e Serviços Ltda. | PRC | 11* | 0101871-50.2017.5.01.0026 |
| ▼ | ACEP | BK Consultoria e Serviços Ltda. | PRC | 11* | 0101871-50.2017.5.01.0026 |

Figura 2. Página principal da aplicação, onde um *dashboard* condensa as informações de processos e projetos de lei cadastrados.

4. Implementação

Para construir a aplicação nos baseamos na pilha MERN -MongoDB², Express³, React⁴ e Node⁵. Essa pilha é essencialmente baseada na linguagem Javascript. Além disso, o projeto está construído dentro de um container Docker, para facilitar o seu *deploy*. A seguir, dividiremos a implementação em 3 subseções: Dependências, componentes e rotas. As dependências explicitam as bibliotecas utilizadas no desenvolvimento dos componentes, que por sua vez descrevem as funcionalidades da aplicação. As rotas descrevem a navegação do usuário.

4.1. Dependências

Optamos por utilizar o material-ui, que é um framework de interface para usuário em React. Dela, importamos:

²<https://www.mongodb.com/>

³<https://expressjs.com/>

⁴<https://pt-br.reactjs.org/>

⁵<https://nodejs.org/en/>

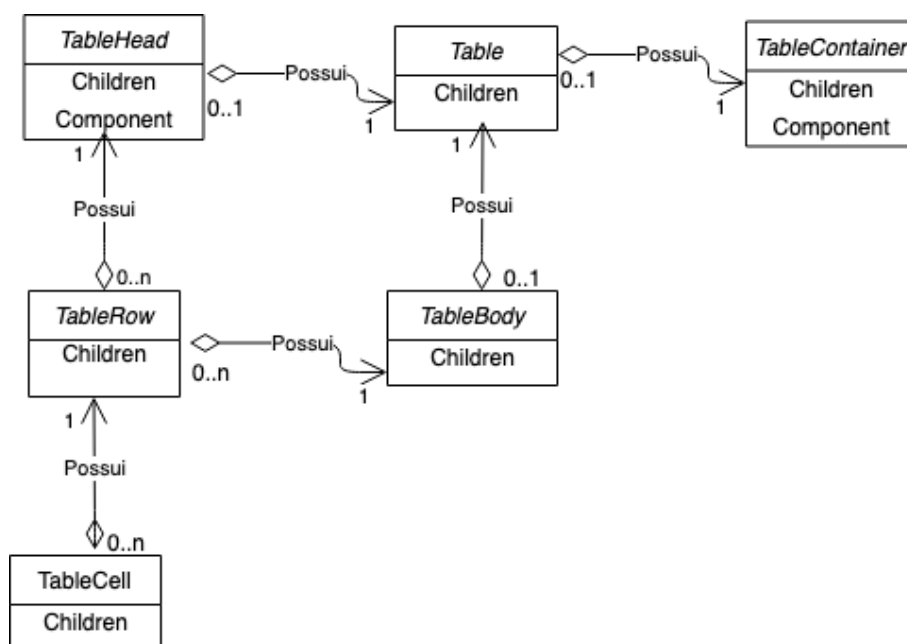


Figura 3. Diagrama UML das classes importadas

- **Table** - Ela fornece um conjunto de ferramentas que possibilitam não apenas visualizar o conteúdo mas também prover os dados ao usuário de forma iterativa com recursos navegação. Além disso, ela facilita a implementação de propriedades css nas linhas de forma independente do header, que contém o nome de cada coluna das informações. A principal propriedade dessa classe é children, que é o conteúdo da tabela em si. A propriedade children se divide em TableBody e TableHead, que serão detalhados a seguir.
- **TableBody** - Essa classe contém os dados da tabela em si. Sua principal propriedade é children, que contém as linhas da tabela. Cada linha da tabela também é uma classe, denominada TableRow. TableBody portanto é um agregado de várias instâncias de TableRow.
- **TableCell** - Cada coluna, seja ela do header da tabela ou das linhas em si, é encapsulada por essa classe. Ou seja, ela renderiza a tag HTML `<th>` quando a classe pai for TableHead, ou a tag `<td>` quando for TableBody.
- **TableContainer** - Ela contém a propriedade children, que é classe Tabela em si. Além disso, outra propriedade importante é o component, que define o estilo da renderização.
- **TableHead** - Essa é a classe que guarda o nome das colunas essencialmente. Duas propriedade importante são o children, que normalmente é da classe TableRow, e o component que define o estilo da renderização.
- **TableRow** - Essa é classe que representa cada linha da tabela, que pode ser tanto do cabeçalho ou dos dados. A principal propriedade é children, que contem tags `<th>` ou `<td>` dependendo se for do cabeçalho ou de cada linha da tabela. A propriedade children pode ser ainda da classe TableCell.

4.2. Componentes

Basicamente temos 3 componentes implementados:

- Row - Classe que explicita como cada linha da nossa tabela está organizada e contém os atributos claimed, court, lawyer, history, name, value e process number.
- History - Classe que explicita o histórico de um determinado processo. Contém os atributos dateTime, message e documentId.
- Dashboard - Função responsável por gerar o dashboard em si.

4.3. Rotas

- Overview - Essa classe/rota faz duas etapas: Primeiramente pega o arquivo xml que é devolvido da api do tribunal. Em segundo lugar efetua o parse dos dados e os retorna para que o Dashboard possa efetuar o display.

5. Conclusão

Devido a falta de automatização na busca por atualizações de processos e projetos de lei, muito do tempo de trabalho de advogados é utilizado de forma ineficiente. Neste trabalho, propomos uma solução em forma de uma página web que tem a expectativa de automatizar a maior parte dessa carga de trabalho. No MVP implementou-se boa parte da interface com o usuário provendo: cadastro de processos, painel de consulta (em um arquivo simulado), painel de controle dos processos e exportação das informações no painel.

Para projeto futuro ainda pretendemos implementar: busca de processos na API do tribunal, um painel personalizado para cada usuário, cadastro dos usuários, agrupamento dos processos por etiquetas e implementar o backend propriamente dito.