
# Tô De Olho: Democratizando a Transparência da Câmara dos Deputados através de Dados Abertos

Trabalho de Conclusão de Curso

Pedro Batista de Almeida Filho

Pablo Vieira Florentino
Orientador

Instituto Federal da Bahia - IFBA
Curso de Análise e Desenvolvimento de Sistemas
Campus Salvador

Salvador, Bahia, Brasil
Junho de 2026

# Sumário

**1 Visão geral** **1**
1.1 Declaração do Problema . . . . . 1
1.2 Proposta de Solução de Software . . . . . 1
1.3 Tecnologias adotadas . . . . . 1
1.4 Trabalhos Relacionados . . . . . 1

**2 Requisitos** **1**
2.1 Requisitos Funcionais . . . . . 1
2.2 Requisitos Não-Funcionais . . . . . 1

**3 Design** **2**
3.1 Projeto UML . . . . . 2
3.2 Visão arquitetural . . . . . 2
3.3 Modelo de Banco de Dados . . . . . 2

**4 Testes de Software** **3**
4.1 Projeto de Testes . . . . . 3

**5 Implantação** **3**
5.1 Projeto de Implantação . . . . . 3

**6 Manual do Usuário** **3**

# 1 Visão geral

## 1.1 Declaração do Problema

A transparência governamental é um pilar fundamental para o fortalecimento da democracia, e a Lei de Acesso à Informação (LAI), instituída em 2011, representa um marco regulatório importante no Brasil ao estabelecer diretrizes para que órgãos públicos disponibilizem informações de interesse coletivo. No entanto, embora os dados da Câmara dos Deputados sejam públicos e estejam disponíveis através de uma API oficial (⟨https://dadosabertos.camara.leg.br/api/v2/⟩), eles encontram-se dispersos e em formatos que não favorecem a análise por cidadãos leigos.

O problema central reside na complexidade de acesso e interpretação dessas informações. A disponibilidade de dados abertos não elimina a dificuldade; ela apenas transfere o desafio do acesso para o processamento, organização e apresentação dos dados de forma compreensível. O cidadão comum não possui conhecimento técnico para consumir APIs, processar grandes volumes de dados ou correlacionar informações sobre despesas parlamentares, votações e proposições legislativas.

Esse cenário gera um distanciamento entre eleitores e seus representantes, prejudicando a fiscalização cidadã e o exercício pleno da democracia. Com as eleições federais de 2026 no horizonte, torna-se ainda mais urgente a necessidade de ferramentas que permitam aos brasileiros acompanhar e avaliar a atuação de seus deputados de forma informada e acessível.

## 1.2 Proposta de Solução de Software

O *Tô De Olho* é uma plataforma *web* de transparência política concebida para democratizar o acesso aos dados da Câmara dos Deputados do Brasil. A solução organiza, processa e apresenta dados públicos oficiais de forma acessível, visando fomentar uma democracia digital mais participativa em âmbito nacional.

A plataforma centraliza informações sobre os 513 deputados federais, incluindo:

- **Perfil do Parlamentar**: dados biográficos, partido, estado, links para redes sociais e resumo de atividades;
- **Transparência Financeira**: visualização clara e pesquisável dos gastos da Cota Parlamentar;
- **Atividade Legislativa**: lista de proposições apresentadas e histórico de votações nas principais pautas;
- **Fórum de Discussão**: espaço para debate cívico entre cidadãos, com validação via API do TSE para garantir a participação de eleitores reais.

O *Tô De Olho* agrega valor ao transformar dados brutos em informação útil, permitindo que o cidadão comum fiscalize seus representantes sem necessidade de conhecimento técnico.

A plataforma atua como uma ponte entre os dados abertos governamentais e a população, tornando a transparência efetivamente acessível.

## 1.3 Tecnologias adotadas

A arquitetura do *Tô De Olho* foi projetada seguindo o padrão de *microservices*, escolhido por sua escalabilidade, manutenibilidade e resiliência. As principais tecnologias adotadas são:

**Backend:**

- **Golang**: linguagem principal para desenvolvimento dos microsserviços, escolhida por seu alto desempenho, eficiência em concorrência e baixo consumo de recursos;
- **PostgreSQL**: banco de dados relacional para persistência dos dados legislativos;
- **Redis**: sistema de cache em memória para otimização de consultas frequentes;
- **RabbitMQ**/**Google Pub**/**Sub**: *message queue* para comunicação assíncrona entre microsserviços e desacoplamento do processo de ingestão de dados.

**Frontend:**

- **Next.js 15**: *framework* React para renderização híbrida (SSR/SSG);
- **TypeScript**: tipagem estática para maior segurança e manutenibilidade do código;
- **Tailwind CSS**: *framework* CSS utilitário para estilização responsiva e *mobile-first*.

**Infraestrutura e DevOps:**

- **Docker**: containerização dos serviços para portabilidade e consistência entre ambientes;
- **Kubernetes (GKE)**: orquestração de containers na Google Cloud Platform;
- **GitHub Actions**: pipelines de CI/CD para integração e deploy contínuos.

**Fontes de Dados:**

- **API RESTful v2 da Câmara dos Deputados**: fonte primária de dados atualizados;
- **Arquivos em Massa (JSON/CSV)**: utilizados para carga inicial histórica (*backfill*);
- **API do TSE**: validação de eleitores para acesso ao fórum.

## 1.4 Trabalhos Relacionados

Diversas iniciativas no Brasil e no mundo buscam promover a transparência política através da tecnologia. Entre os trabalhos relacionados, destacam-se:

**Portal da Transparência (CGU):** Principal ferramenta oficial do governo federal para

transparência, disponibiliza dados sobre gastos públicos, servidores e convênios. Embora abrangente, sua interface é voltada para consultas técnicas e não oferece visualizações consolidadas por parlamentar ou fórum de discussão cidadã.

**Portal de Dados Abertos da Câmara:** Fonte oficial dos dados legislativos, oferece API e arquivos para download. Contudo, exige conhecimento técnico para consumo e não provê interpretação ou contextualização das informações.

**Ranking dos Políticos:** Iniciativa da sociedade civil que avalia parlamentares com base em critérios predefinidos. Diferencia-se do *Tô De Olho* por adotar uma abordagem de *ranking* com critérios fixos, enquanto nossa plataforma permite que o próprio cidadão analise os dados e forme sua opinião.

**Serenata de Amor:** Projeto de *data science* que utiliza inteligência artificial para detectar irregularidades em gastos parlamentares. Foca em análise automatizada de anomalias, enquanto o *Tô De Olho* prioriza a apresentação acessível dos dados para fiscalização cidadã direta.

**Diferencial do Tô De Olho:** Nossa solução se diferencia por: (1) consolidar múltiplas fontes de dados em uma interface única e acessível; (2) oferecer um fórum validado para discussão cívica; (3) utilizar arquitetura de microsserviços escalável para suportar picos de acesso em períodos eleitorais; (4) focar na experiência do usuário leigo, seguindo princípios de acessibilidade (WCAG 2.1).

# 2 Requisitos

## 2.1 Requisitos Funcionais

**Módulo de Deputados:**

- **[RF01]** Como cidadão, eu preciso visualizar a lista de todos os deputados federais para conhecer meus representantes.
- **[RF02]** Como cidadão, eu preciso filtrar deputados por estado, partido ou nome para encontrar parlamentares específicos.
- **[RF03]** Como cidadão, eu preciso acessar o perfil completo de um deputado para conhecer suas informações biográficas e de contato.

**Módulo de Despesas:**

- **[RF04]** Como cidadão, eu preciso visualizar os gastos da cota parlamentar de cada deputado para fiscalizar o uso de recursos públicos.
- **[RF05]** Como cidadão, eu preciso filtrar despesas por período, tipo e fornecedor para análises específicas.
- **[RF06]** Como cidadão, eu preciso ver o total de gastos por categoria para entender como

os recursos são utilizados.

**Módulo de Atividade Legislativa:**

- **[RF07]** Como cidadão, eu preciso visualizar as proposições apresentadas por um deputado para acompanhar sua produtividade.
- **[RF08]** Como cidadão, eu preciso ver como um deputado votou nas principais pautas para avaliar seu posicionamento.
- **[RF09]** Como cidadão, eu preciso acompanhar a tramitação de projetos de lei para entender o processo legislativo.

**Módulo de Fórum:**

- **[RF10]** Como eleitor validado, eu preciso criar discussões sobre temas políticos para debater com outros cidadãos.
- **[RF11]** Como eleitor validado, eu preciso comentar em discussões existentes para contribuir com o debate.
- **[RF12]** Como usuário, eu preciso validar meu CPF junto ao TSE para participar do fórum como eleitor verificado.

**Módulo de Usuário:**

- **[RF13]** Como usuário, eu preciso criar uma conta na plataforma para personalizar minha experiência.
- **[RF14]** Como usuário, eu preciso salvar deputados favoritos para acompanhá-los mais facilmente.

## 2.2 Requisitos Não-Funcionais

**Desempenho:**

- **[RNF01]** O sistema deve responder a requisições de consulta em até 2 segundos sob condições normais de uso.
- **[RNF02]** A arquitetura deve suportar escalabilidade horizontal para lidar com picos de acesso em períodos eleitorais.

**Usabilidade e Acessibilidade:**

- **[RNF03]** O sistema deve ser acessível via navegadores *web* em dispositivos *desktop* e *mobile*.
- **[RNF04]** A interface deve seguir o padrão *mobile-first* para garantir boa experiência em dispositivos móveis.
- **[RNF05]** O sistema deve seguir as diretrizes de acessibilidade WCAG 2.1 nível AA.

**Confiabilidade:**

- **[RNF06]** Os dados devem ser sincronizados diariamente com a API oficial da Câmara dos Deputados.
- **[RNF07]** O sistema deve manter disponibilidade mínima de 99% durante o período eleitoral.

**Segurança:**

- **[RNF08]** As comunicações devem ser criptografadas utilizando HTTPS/TLS.
- **[RNF09]** A validação de eleitores deve ser realizada através da API oficial do TSE.
- **[RNF10]** O sistema deve estar em conformidade com a LGPD (Lei Geral de Proteção de Dados).

**Manutenibilidade:**

- **[RNF11]** A arquitetura de microsserviços deve permitir atualizações independentes de cada módulo.
- **[RNF12]** O código deve seguir padrões de desenvolvimento e estar documentado.
- **[RNF13]** O sistema deve possuir *pipelines* de CI/CD para integração e deploy contínuos.

# 3 Design

## 3.1 Projeto UML

[Insira os seguintes Diagramas de UML para o seu projeto:

1. Diagrama de Classe
2. Diagrama de Atividades da principal atividade do sistema
3. Diagrama de Sequência da principal atividade do sistema
4. Diagrama de Estado do principal objeto do sistema
5. Diagrama de Componentes
6. Diagrama de Implantação

OBS: Como cada sistema tem particularidades específicas, a escolha de qual a principal atividade e qual o principal objeto do sistema deve ser discutida com o orientador. Consideramos que existem alguns diagramas básicos que devem existir em qualquer projeto. São eles: Diagrama de Classes, Componentes e Implantação. A elaboração dos outros diagramas devem ser discutidos com seu orientador.]
[Obrigatório - A seção é obrigatória, os diagramas de classes e diagrama de casos de uso

são obrigatórios. Os outros diagramas são opcionais a depender do tipo do sistema a ser desenvolvido. Deve ser discutido com o orientador quais diagramas devem constar no trabalho.]

## 3.2 Visão arquitetural

O *Tô De Olho* adota uma arquitetura de **microsserviços**, onde cada componente é responsável por um domínio específico do negócio e pode ser desenvolvido, implantado e escalado de forma independente. Essa escolha arquitetural foi motivada pelos seguintes fatores:

- **Escalabilidade**: permite escalar individualmente serviços com maior demanda, como o de consulta de despesas durante períodos de maior fiscalização;
- **Manutenibilidade**: atualizações em um serviço (ex: despesas) não afetam os demais (ex: votações);
- **Resiliência**: falhas em um microsserviço não comprometem todo o sistema.

**Microsserviços Principais:**

- `servico-deputados`: gestão de dados dos parlamentares;
- `servico-despesas`: transparência financeira e cota parlamentar;
- `servico-atividades`: proposições e votações legislativas;
- `servico-usuarios`: autenticação e gerenciamento de perfis;
- `servico-forum`: discussões e participação cidadã;
- `servico-ingestao-dados`: ETL (*Extract, Transform, Load*) de dados da Câmara.

**Estratégia de Ingestão de Dados:**

A plataforma utiliza uma estratégia híbrida de ingestão:

1. **Backfill**: carga inicial histórica através de arquivos em massa (JSON/CSV) disponibilizados pela Câmara;
2. **Sincronização Contínua**: *CronJobs* diários que consomem a API RESTful v2 para manter os dados atualizados.

**Comunicação entre Serviços:**

Os microsserviços comunicam-se através de:

- **APIs REST**: para comunicação síncrona entre serviços;
- **Message Queue (RabbitMQ)**: para comunicação assíncrona e desacoplamento, especialmente no processo de ingestão de dados.

**Infraestrutura:**

Todos os serviços são containerizados com Docker e orquestrados pelo Kubernetes (GKE - Google Kubernetes Engine), permitindo auto-scaling, balanceamento de carga e alta disponibilidade.

## 3.3 Modelo de Banco de Dados

[Descreva o modelo de Dados Físico e Lógico]
[Obrigatório - Somente se o sistema possui banco de dados.]

# 4 Testes de Software

## 4.1 Projeto de Testes

[Descreva em detalhes as estratégias de Testes que utilizou para desenvolvimento do seu software. Se utilizou casos de testes, testes automatizados ou manuais, ferramentas, etc.]
[Obrigatório]

# 5 Implantação

## 5.1 Projeto de Implantação

[Descreva a Plataforma de Hardware e Software requeridas para instalação e operação do seu software.]
[Obrigatório]

# 6 Manual do Usuário

[Descreva o Manual de Usuário, como utilizar o seu software. Sugerimos que coloque figuras com as telas do seu sistema se necessário para melhorar o entendimento do usuário.]

[Obrigatório]

# Agradecimentos

[Agradecimentos a colaboradores do seu projeto]
[Opcional]

[Forneça uma lista completa de todos os documentos mencionados ou que foram utilizados como referência na elaboração deste documento. Todos os documentos devem ser identificados por título, data, nome e organização responsável por sua publicação. Especifique as fontes dessas referências. Utilize o padrão ABNT para o formato das referências.]
[Obrigatório]

**[Dica: insira suas referencias no arquivo ”referencias.bib”e utilize o comando ”cite”para utilizá-la]**

# Apêndices

# Glossário, Siglas e Abreviações

[Forneça as definições de todos os termos, siglas e abreviações necessárias à compreensão deste documento.] [Opcional]