Tô De Olho: Democratizando a Transparência do Senado Federal 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
Fevereiro de 2026

Sumário

1 Visão geral **1**
- 1.1 Objetivos . . . 1
  - 1.1.1 Objetivo Geral . . . 1
  - 1.1.2 Objetivos Específicos . . . 1
- 1.2 Declaração do Problema . . . 1
- 1.3 Proposta de Solução de Software . . . 2
- 1.4 Tecnologias adotadas . . . 2
- 1.5 Trabalhos Relacionados . . . 3

2 Fundamentação Teórica **4**
- 2.1 Transparência Pública e Dados Abertos . . . 4
- 2.2 Democracia Digital e Participação Cidadã . . . 5
- 2.3 Métricas de Efetividade Legislativa . . . 6
- 2.4 Visualização de Dados e Retórica Visual . . . 6
- 2.5 Arquitetura de Microsserviços . . . 6
- 2.6 Engenharia de Dados: APIs e Processos ETL . . . 7

3 Metodologia **7**
- 3.1 Abordagem de Desenvolvimento . . . 7
- 3.2 Fontes de Dados . . . 8
  - 3.2.1 API Legislativa do Senado . . . 8
  - 3.2.2 API Administrativa do Senado . . . 8
  - 3.2.3 Portal da Transparência (CGU) . . . 8
- 3.3 Estratégia de Ingestão . . . 9
- 3.4 Arquitetura do Sistema . . . 9
- 3.5 Stack Tecnológico . . . 9
- 3.6 Infraestrutura e Implantação . . . 10
- 3.7 Algoritmo de Ranking . . . 10

4 Requisitos **11**
- 4.1 Requisitos Funcionais . . . 11
- 4.2 Requisitos Não-Funcionais . . . 11

5 Design **12**
- 5.1 Projeto UML . . . 12
  - 5.1.1 Diagrama de Classes . . . 13
  - 5.1.2 Diagrama de Implantação . . . 13
- 5.2 Visão arquitetural . . . 14
- 5.3 Modelo de Banco de Dados . . . 15

6 Testes de Software **16**
- 6.1 Projeto de Testes . . . 16

7 Implantação **16**

7.1 Projeto de Implantação 16

8 Manual do Usuário Simplificado **17**

9 Considerações Finais **17**

1 Visão geral

O *Tô De Olho* é uma plataforma *web* de transparência parlamentar focada no Senado Federal. Sua proposta é aproximar cidadãos dos dados legislativos oficiais, convertendo informação dispersa em conhecimento fiscalizável. O projeto vai além dos dados abertos básicos, integrando fontes complexas como a Cota para o Exercício da Atividade Parlamentar dos Senadores (CEAPS) e as ”emendas PIX”. Ao combinar arquitetura de *microservices*, ingestão híbrida (API do Senado, Scrapers e Portal da Transparência) e um front-end em Next.js 15, a plataforma busca reduzir a assimetria de informação sobre os 81 senadores da República [1].

A literatura de democracia digital evidencia que TICs ampliam possibilidades de participação, mas só geram valor quando articuladas a contextos de uso e inclusão. Avelino et al. mapeiam iniciativas de dados abertos, reforçando que tecnologias precisam ser mediadas por visualizações claras para efetivação do controle social [2]. Com um corpo legislativo menor e mais ”caro”per capita que a Câmara, o Senado carece de ferramentas focadas que cruzem votações nominais com a execução orçamentária de emendas. À luz desses estudos, o *Tô De Olho* procura transformar a transparência passiva em *accountability* ativa, oferecendo rankings e métricas objetivas de desempenho parlamentar [3, 4].

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver uma plataforma *web* de transparência política que centralize, organize e simplifique o acesso aos dados públicos do Senado Federal, fomentando a fiscalização cidadã e o debate qualificado sobre a atuação dos 81 senadores, com ênfase no monitoramento de gastos e emendas parlamentares.

1.1.2 Objetivos Específicos

- Implementar uma arquitetura de *microservices* em Golang para ingestão de dados da API do Senado e do Portal da Transparência;
- Desenvolver rotinas ETL para consumir a API Administrativa do Senado (CEAPS) e utilizar *Web Scraping* apenas para dados não estruturados, como remuneração de gabinete;
- Criar algoritmos de *Ranking* para avaliar senadores com base em assiduidade, transparência de gastos e disciplina partidária;
- Construir uma interface *front-end* responsiva utilizando Next.js 15, permitindo a visualização intuitiva de perfis, despesas e ”scorecards”de fiscalização.

1.2 Declaração do Problema

Embora o Senado Federal disponibilize dados via API e Portal da Transparência, estas fontes sofrem de fragmentação. A API de Dados Abertos Legislativa foca no processo

legislativo (matérias, votações), enquanto a API Administrativa contém dados da CEAPS em formato estruturado. Contudo, informações de remuneração de gabinete são disponibilizadas apenas em arquivos CSV dinâmicos. Além disso, a recente explosão das ”Emendas PIX”(Transferências Especiais) criou um vácuo de transparência, onde o rastro do dinheiro público se perde. Sem uma ferramenta que consolide mandato, gastos de gabinete e destinação de emendas em uma visão unificada, o cidadão não possui insumos para avaliar qualitativamente seus representantes [2].

1.3 Proposta de Solução de Software

O *Tô De Olho* é uma plataforma *web* concebida para centralizar a fiscalização do Senado Federal. A solução integra múltiplas APIs oficiais e utiliza *crawlers* apenas para dados não estruturados (como remuneração de gabinete), agregando informações de três dimensões do mandato: Atividade Legislativa, Gestão de Recursos (CEAPS) e Articulação Orçamentária (Emendas).

A plataforma centraliza informações sobre os 81 senadores, incluindo:

- **Ranking de Desempenho**: avaliação baseada em métricas objetivas como assiduidade, economia de cota e fidelidade partidária;
- **Raio-X da CEAPS**: análise detalhada dos gastos com passagens, divulgação e consultorias, identificando padrões de uso;
- **Rastreio de Emendas**: cruzamento de dados do Portal da Transparência para evidenciar o destino das emendas parlamentares, com foco nas modalidades de transferência especial.

O sistema atua como um ”auditor digital”, automatizando o cruzamento de dados que, manualmente, seria inviável para o eleitor comum.

1.4 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 escolhida para os serviços de ingestão e API, ideal para processamento concorrente de grandes volumes de dados (ETL dos CSVs de despesas);
- **PostgreSQL**: banco de dados relacional para persistência estruturada (Senadores, Votações, Despesas);
- **Redis**: cache para rankings e totalizadores de gastos;
- **Go-Colly**: framework de *web scraping* para captura de dados não estruturados (ex: sites de detalhes de emendas quanto necessário).

Frontend:

- **Next.js 15**: (App Router) para renderização eficiente e SEO;
- **Recharts**: biblioteca para visualização de dados (gráficos de evolução de gastos);
- **Tailwind CSS**: estilização responsiva.

Infraestrutura e DevOps:

- **Docker**: containerização;
- **Google Cloud Run** / **GKE**: orquestração serverless/container;
- **GitHub Actions**: CI/CD.

Fontes de Dados:

- **API do Senado Federal**: dados legislativos (votações, matérias);
- **Portal Administrativo do Senado**: fontes de arquivos CSV para a CEAPS;
- **API do Portal da Transparência (CGU)**: dados de emendas parlamentares e transferências da união.

1.5 Trabalhos Relacionados

Diversas iniciativas no Brasil e no mundo buscam promover a transparência política por meio da tecnologia. À luz da Escada de Participação de Arnstein [5], podemos classificar essas ferramentas conforme o grau de poder que conferem ao cidadão:

Nível Informação:

Portal da Transparência (CGU): Principal ferramenta oficial do governo federal para acesso a dados de gastos públicos, servidores e transferências. Embora abrangente, sua interface é voltada para consultas técnicas, exigindo conhecimento prévio sobre a estrutura orçamentária para navegação efetiva. O cidadão tem acesso aos dados, mas sem ferramentas de análise comparativa.

Portal de Dados Abertos da Câmara e Senado: Fontes oficiais que disponibilizam APIs e arquivos para download. Seguem os princípios de *Open Government Data*, porém exigem conhecimento técnico para consumo. Situam-se no degrau mais básico da informação — dados brutos sem interpretação.

Nível Consulta:

Ranking dos Políticos: Iniciativa da sociedade civil que avalia parlamentares com base em critérios predefinidos de votação. Diferencia-se por adotar uma abordagem de *ranking* com critérios fixos focados em posicionamento ideológico. Avança em relação aos portais oficiais ao oferecer interpretação, porém a metodologia fechada limita a autonomia do cidadão.

Nível Fiscalização Qualificada:

Operação Serenata de Amor: Projeto de código aberto que utiliza inteligência artificial para detectar irregularidades em gastos parlamentares da Câmara dos Deputados. Albuquerque, Almeida e Costa [6] analisaram o projeto sob a ótica de **valor público**, demonstrando como tecnologias desenvolvidas com dados governamentais abertos podem potencializar o controle social.

O projeto é composto por dois componentes principais: a *Rosie*, algoritmo de *machine learning* que analisa notas fiscais em busca de padrões anômalos; e o *Jarbas*, interface web que permite ao cidadão consultar as suspeitas identificadas. O estudo aponta que, entre 2016 e 2018, a Rosie analisou mais de 3 milhões de reembolsos e gerou milhares de denúncias formais ao Congresso.

Diferencial do Tô De Olho:

O *Tô De Olho* posiciona-se como evolução dessas iniciativas, com foco específico no **Senado Federal** — casa legislativa menos coberta por ferramentas de fiscalização. A proposta diferencia-se por:

1. **Consolidação Multi-Fonte**: Integra três APIs distintas (Legislativa, Administrativa e Portal da Transparência) em interface única, superando a fragmentação dos portais oficiais;
2. **Metodologia Transparente**: Inspirado no *Legislative Effectiveness Score* de Volden e Wiseman [7], o algoritmo de ranking expõe claramente critérios e pesos, permitindo que o cidadão compreenda — e questione — a avaliação;
3. **Design Orientado à Ação**: Seguindo princípios de visualização retórica [8], a plataforma contextualiza dados absolutos com médias e comparativos, reduzindo a manipulação e estimulando conclusões informadas;
4. **Foco em Emendas PIX**: Rastreia transferências especiais (modalidade criada em 2020), preenchendo lacuna de transparência não coberta por ferramentas anteriores.

2 Fundamentação Teórica

Esta seção apresenta os conceitos fundamentais que embasam o desenvolvimento do *Tô De Olho*, abrangendo transparência pública, democracia digital e arquitetura de sistemas distribuídos.

2.1 Transparência Pública e Dados Abertos

A transparência governamental constitui pilar fundamental do Estado Democrático de Direito. No Brasil, a Lei de Acesso à Informação (LAI – Lei nº 12.527/2011) estabelece que o acesso é a regra e o sigilo, a exceção, garantindo aos cidadãos o direito de solicitar e receber informações públicas sem necessidade de justificativa [2].

A literatura distingue duas modalidades de transparência: a **transparência ativa**, na qual

o Estado disponibiliza informações de forma proativa em portais e bases de dados; e a **transparência passiva**, que responde às solicitações dos cidadãos via canais específicos. O Portal de Dados Abertos do Senado Federal exemplifica a primeira modalidade, disponibilizando APIs e arquivos para consulta pública.

O conceito de *Open Government Data* (Dados Governamentais Abertos) preconiza que as informações públicas devem ser disponibilizadas em formatos abertos, processáveis por máquina e livres de licenças restritivas. Tim Berners-Lee propôs uma escala de cinco estrelas para avaliar a qualidade dos dados abertos, sendo o nível máximo aquele em que os dados são linkados (*Linked Open Data*), permitindo cruzamentos entre diferentes fontes [2].

2.2 Democracia Digital e Participação Cidadã

O conceito de democracia digital refere-se ao emprego de tecnologias de informação e comunicação (TICs) para produzir "mais democracia e melhores democracias" [1]. Gomes identifica três fases históricas neste campo: a teledemocracia (anos 1970-90), marcada por experimentos com televisão interativa; a fase da internet (1995-2005), caracterizada pelo debate sobre potenciais e limites da rede; e a autonomização contemporânea, onde subtemas como governo aberto, *smart cities* e parlamento digital desenvolvem-se de forma independente [4].

A participação cidadã mediada por tecnologia pode assumir diferentes níveis de profundidade. Sherry Arnstein, em seu trabalho seminal de 1969, propõe a "Escada da Participação Cidadã", uma tipologia de oito degraus que classifica o grau de poder real conferido aos cidadãos [5]. Os degraus inferiores — manipulação e terapia — representam formas de **não-participação**, onde o objetivo é "educar" ou "curar" os participantes em vez de ouvi-los. Os degraus intermediários — informação, consulta e pacificação — constituem níveis de **participação simbólica**, nos quais cidadãos podem ouvir e ser ouvidos, mas sem garantia de que suas vozes influenciem decisões. Apenas nos degraus superiores — parceria, delegação de poder e controle cidadão — observa-se redistribuição efetiva de poder decisório.

Ferramentas de transparência como o *Tô De Olho* situam-se primariamente no degrau da **informação**: proveem ao cidadão dados estruturados sobre a atuação parlamentar, condição necessária — mas não suficiente — para o exercício pleno da fiscalização. Como adverte Arnstein, "participação sem redistribuição de poder é um processo vazio e frustrante para os desprovidos de poder" [5]. Reconhecendo essa limitação, o projeto busca ir além da mera disponibilização de dados, oferecendo *rankings*, comparativos e visualizações que **empoderam** o cidadão para uma fiscalização mais qualificada.

No contexto brasileiro, Avelino et al. mapeiam iniciativas de governo aberto no âmbito federal, identificando avanços significativos na disponibilização de dados, porém alertando para desafios como a brecha digital e o analfabetismo funcional, que limitam o acesso efetivo da população às informações disponibilizadas [2].

2.3 Métricas de Efetividade Legislativa

A avaliação quantitativa do desempenho parlamentar constitui tema relevante na ciência política contemporânea. Volden e Wiseman desenvolveram o *Legislative Effectiveness Score* (LES), uma métrica que mensura a capacidade de parlamentares em conduzir suas proposições através do processo legislativo [7]. O modelo considera múltiplas dimensões: quantidade de projetos apresentados, taxa de aprovação em comissões, progressão para votação em plenário e conversão em lei.

Os autores identificaram fatores correlacionados à maior efetividade: senioridade no mandato, posição em comissões estratégicas, pertencimento ao partido majoritário e experiência prévia em cargos legislativos estaduais. Embora desenvolvida para o contexto norte-americano, a metodologia oferece um *framework* adaptável para avaliar parlamentares brasileiros.

No *Tô De Olho*, o "Score" do senador inspira-se nesta abordagem, combinando indicadores objetivos como presença em votações, produtividade legislativa (proposições de autoria e relatorias), economia na utilização da cota parlamentar e participação em comissões. A transparência metodológica — expor claramente os critérios e pesos utilizados — é fundamental para que o *ranking* seja percebido como ferramenta de informação, não de manipulação.

2.4 Visualização de Dados e Retórica Visual

A apresentação de dados ao cidadão não é neutra: escolhas de *design* influenciam a interpretação das informações. Hullman e Diakopoulos investigaram os "efeitos de enquadramento" (*framing effects*) em visualizações narrativas, demonstrando que técnicas retóricas como seleção, omissão, ênfase e sequenciamento podem direcionar a leitura do público [8].

Os autores identificam quatro categorias de técnicas retóricas em visualizações: (1) **proveniência** — identificar a origem e credibilidade dos dados; (2) **mapeamento visual** — como elementos gráficos representam variáveis; (3) **anotações linguísticas** — textos que guiam a interpretação; e (4) **interatividade** — controles que permitem ao usuário explorar os dados por conta própria.

Para o *Tô De Olho*, esses princípios orientam decisões de *design*: exibir sempre a fonte oficial e data de atualização (**proveniência**); utilizar escalas consistentes em gráficos comparativos (**mapeamento**); contextualizar valores absolutos com médias e percentis (**anotação**); e permitir filtros por partido, estado e período (**interatividade**). O objetivo é maximizar a transparência metodológica, evitando que a plataforma seja percebida como veículo de viés político.

2.5 Arquitetura de Microsserviços

A arquitetura de *microservices* representa uma abordagem de desenvolvimento na qual uma aplicação é estruturada como conjunto de serviços independentes, cada qual responsável por uma funcionalidade específica do negócio. Taibi, Lenarduzzi e Pahl investigaram empiricamente os processos de migração para esta arquitetura, identificando motivações como escalabilidade e manutenibilidade, além de desafios técnicos e organizacionais [9].

Os principais benefícios desta arquitetura incluem: escalabilidade granular, que permite alocar recursos apenas aos serviços mais demandados; manutenibilidade, uma vez que alterações em um serviço não afetam os demais; e resiliência, já que a falha de um componente não compromete todo o sistema.

No contexto de e-governo, Maureal et al. demonstram a aplicabilidade desta arquitetura em sistemas públicos, relatando redução de 40% no tempo de processamento e aumento de 25% na satisfação dos usuários após a implementação do sistema NTC-EDGE nas Filipinas [10]. Para o *Tô De Olho*, esta abordagem permite que os serviços de ingestão de dados (APIs do Senado, Portal da Transparência) operem independentemente dos serviços de apresentação (*frontend*), garantindo que picos de acesso em períodos eleitorais possam ser absorvidos mediante escalamento horizontal seletivo.

2.6 Engenharia de Dados: APIs e Processos ETL

A estratégia de ingestão de dados do *Tô De Olho* fundamenta-se no padrão ETL (*Extract, Transform, Load*): extração dos dados brutos das fontes oficiais, transformação para normalização e enriquecimento, e carga no banco de dados da aplicação.

A abordagem híbrida adotada combina dois mecanismos: *backfill*, que realiza carga inicial massiva através de arquivos históricos disponibilizados em formato CSV; e sincronização contínua, que consome as APIs RESTful oficiais via tarefas agendadas (*cron jobs*) para manter os dados atualizados.

Ulbricht discute as implicações da coleta automatizada de dados públicos para projetos democráticos, argumentando que técnicas de *web scraping* podem democratizar o acesso à informação ao superar as limitações de interfaces oficiais pouco amigáveis [11]. No contexto do *Tô De Olho*, o *scraping* é empregado de forma complementar às APIs, capturando dados não estruturados como valores de remuneração de gabinete que não estão disponíveis via interface programática.

3 Metodologia

Esta seção descreve a metodologia adotada para o desenvolvimento do *Tô De Olho*, detalhando a abordagem de desenvolvimento, as fontes de dados utilizadas, a arquitetura do sistema e a infraestrutura de implantação.

3.1 Abordagem de Desenvolvimento

O desenvolvimento do projeto seguiu uma abordagem iterativa e incremental, inspirada em práticas ágeis. O trabalho foi organizado em ciclos de desenvolvimento focados em entregas funcionais, permitindo validação contínua das funcionalidades implementadas.

A divisão do trabalho ocorreu em cinco fases principais:

1. **Fundação**: Estruturação do projeto em Golang, implementação do cliente para a API

Legislativa do Senado, criação das *migrations* do banco de dados e configuração inicial do *frontend* em Next.js;

2. **Ingestão de Dados**: Implementação do cliente para a API Administrativa, configuração do *scheduler* para tarefas agendadas, coleta de votações nominais e carga de dados históricos;

3. **Ranking e API**: Desenvolvimento do serviço de cálculo de rankings, criação dos *endpoints* REST para consumo pelo *frontend*, configuração do cache Redis e implementação de testes automatizados;

4. **Frontend**: Desenvolvimento do *dashboard* principal, interface de ranking interativo e páginas de perfil dos senadores;

5. **Emendas e Polimento**: Integração com o Portal da Transparência para dados de emendas parlamentares, visualizações de dados e preparação para *deploy*.

3.2 Fontes de Dados

O sistema integra três fontes de dados governamentais oficiais, cada uma com características e formatos distintos:

3.2.1 API Legislativa do Senado

Disponível em `legis.senado.leg.br/dadosabertos`, esta API RESTful fornece dados do processo legislativo: lista de senadores em exercício, histórico de mandatos, votações nominais em plenário e comissões, proposições de autoria parlamentar, discursos proferidos e composição das comissões. Os dados são retornados em formato JSON, com paginação para grandes volumes.

3.2.2 API Administrativa do Senado

Acessível em `adm.senado.gov.br/adm-dadosabertos`, esta interface disponibiliza dados administrativos: lançamentos da Cota para o Exercício da Atividade Parlamentar dos Senadores (CEAPS), informações sobre auxílio-moradia, escritórios de apoio, lista de servidores por lotação e remunerações mensais. Alguns dados sensíveis são disponibilizados apenas em arquivos CSV para *download*.

3.2.3 Portal da Transparência (CGU)

A API do Portal da Transparência (`api.portaldatransparencia.gov.br`) fornece dados de emendas parlamentares e transferências da União. Para acesso, é necessária autenticação via chave de API. O filtro `tipoEmenda=Transferência Especial` permite identificar as chamadas "emendas PIX", modalidade de repasse que dispensa convênio e apresenta menor rastreabilidade.

3.3 Estratégia de Ingestão

A ingestão de dados segue uma estratégia híbrida que combina carga inicial massiva com atualização contínua:

Backfill (Carga Histórica): Para a população inicial do banco de dados com dados históricos, podem ser utilizados arquivos em massa CSV quando disponíveis. A API Administrativa do Senado fornece endpoint REST para consulta de despesas CEAPS por ano, sendo a fonte principal para sincronização contínua.

Sincronização Contínua: Tarefas agendadas (*cron jobs*) executam diariamente a coleta de atualizações via APIs. O serviço de ingestão verifica a data da última atualização de cada entidade e solicita apenas os registros novos ou modificados, otimizando o consumo de recursos e respeitando os limites de requisição das APIs.

3.4 Arquitetura do Sistema

O *Tô De Olho* adota uma arquitetura de *microservices*, na qual cada componente é responsável por um domínio específico do negócio:

- `servico-senadores`: Gerencia dados cadastrais dos parlamentares e cálculo de rankings;
- `servico-ceaps`: Processa e totaliza despesas da Cota Parlamentar;
- `servico-emendas`: Integra dados de emendas e transferências do Portal da Transparência;
- `servico-legislativo`: Coleta e armazena votações, discursos e proposições;
- `servico-ingestao`: Orquestra as tarefas agendadas de coleta de dados.

A comunicação entre serviços ocorre de duas formas: APIs REST para chamadas síncronas (consultas do *frontend*) e filas de mensagens para operações assíncronas (processamento de grandes volumes durante a ingestão).

3.5 Stack Tecnológico

A escolha das tecnologias foi orientada por critérios de desempenho, manutenibilidade e adequação ao domínio do problema:

Backend: Golang foi selecionada pela sua eficiência em processamento concorrente, característica essencial para a ingestão paralela de múltiplas fontes de dados. O framework Gin provê roteamento HTTP de alto desempenho, enquanto GORM oferece mapeamento objeto-relacional com suporte a migrações.

Frontend: Next.js 15 com *App Router* possibilita renderização híbrida (servidor e cliente), otimizando o tempo de carregamento inicial e o SEO. A biblioteca Recharts é utilizada para visualização de dados, permitindo gráficos interativos de evolução de gastos e comparativos entre senadores.

Banco de Dados: PostgreSQL armazena os dados de forma estruturada, com índices otimizados para as consultas mais frequentes (agregações por senador, período e tipo de despesa). Redis atua como cache para rankings pré-computados e resultados de consultas complexas.

Scraping: Go-Colly é empregado para captura de dados não disponíveis via API, como informações de remuneração de gabinete que são publicadas apenas em páginas HTML dinâmicas.

3.6 Infraestrutura e Implantação

Todos os serviços são containerizados com Docker, utilizando *multi-stage builds* para otimização das imagens. A orquestração ocorre via Google Kubernetes Engine (GKE), que gerencia o ciclo de vida dos containers, balanceamento de carga e escalamento automático baseado em métricas de CPU e memória.

O *pipeline* de CI/CD, implementado com GitHub Actions, automatiza as etapas de *build*, testes e *deploy*:

1. **Build**: Compilação dos binários Go e verificação de erros de sintaxe;
2. **Test**: Execução de testes unitários e de integração, utilizando *testcontainers* para instâncias efêmeras de PostgreSQL e Redis;
3. **Publish**: Construção da imagem Docker e envio para o Google Container Registry;
4. **Deploy**: Aplicação dos manifestos Kubernetes para atualização dos serviços em produção.

3.7 Algoritmo de Ranking

O cálculo do *score* de cada senador baseia-se em cinco critérios objetivos, ponderados conforme sua relevância para a fiscalização cidadã:

- **Presença em Votações (25%)**: Percentual de participação nas votações nominais em plenário;
- **Produtividade Legislativa (25%)**: Quantidade de proposições de autoria e relatorias assumidas;
- **Economia na Cota (20%)**: Relação entre o valor utilizado e o teto disponível da CEAPS;
- **Participação em Comissões (15%)**: Envolvimento efetivo nas comissões permanentes e temporárias;
- **Transparência (15%)**: Disponibilidade de informações públicas sobre o mandato.

Cada métrica é normalizada em uma escala de 0 a 100 antes do cálculo final, permitindo comparabilidade entre critérios de natureza distinta. A fórmula resultante é:

$$Score = (Presença \times 0.25) + (Produtividade \times 0.25) + (Economia \times 0.20) + (Comissões \times 0.15) + (Transparência \times$$

Os rankings são recalculados diariamente após a conclusão das tarefas de ingestão e armazenados em cache Redis, garantindo resposta imediata às consultas do *frontend*.

4 Requisitos

4.1 Requisitos Funcionais

Módulo de Senadores:

- **[RF01]** O sistema deve apresentar a lista atualizada dos 81 senadores com foto, partido e estado.
- **[RF02]** O sistema deve permitir a busca de senadores por nome, sigla partidária ou UF.
- **[RF03]** O sistema deve exibir o ”Score”do senador baseado no algoritmo de ranking do projeto.

Módulo de Transparência Financeira (CEAPS):

- **[RF04]** O sistema deve importar os lançamentos da Cota Parlamentar (CEAPS) através de arquivos CSV/Dados Abertos.
- **[RF05]** O sistema deve permitir visualizar o gasto acumulado por tipo de despesa (passagens, correios, consultorias).
- **[RF06]** O sistema deve exibir os fornecedores que mais receberam recursos de um determinado senador.

Módulo de Emendas e Orçamento:

- **[RF07]** O sistema deve integrar com o Portal da Transparência para buscar emendas de autoria do senador.
- **[RF08]** O sistema deve destacar valores destinados via ”Transferências Especiais”(emendas PIX).

Módulo de Atividade Legislativa:

- **[RF09]** O sistema deve listar as votações nominais recentes e o voto de cada senador (Sim/Não/Abstenção).

4.2 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 as APIs oficiais do Senado Federal e Portal da Transparência.
- **[RNF07]** O sistema deve manter disponibilidade mínima de 99% durante o período eleitoral.

Segurança:

- **[RNF08]** As comunicações devem ser criptografadas utilizando HTTPS/TLS.
- **[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.

5 Design

5.1 Projeto UML

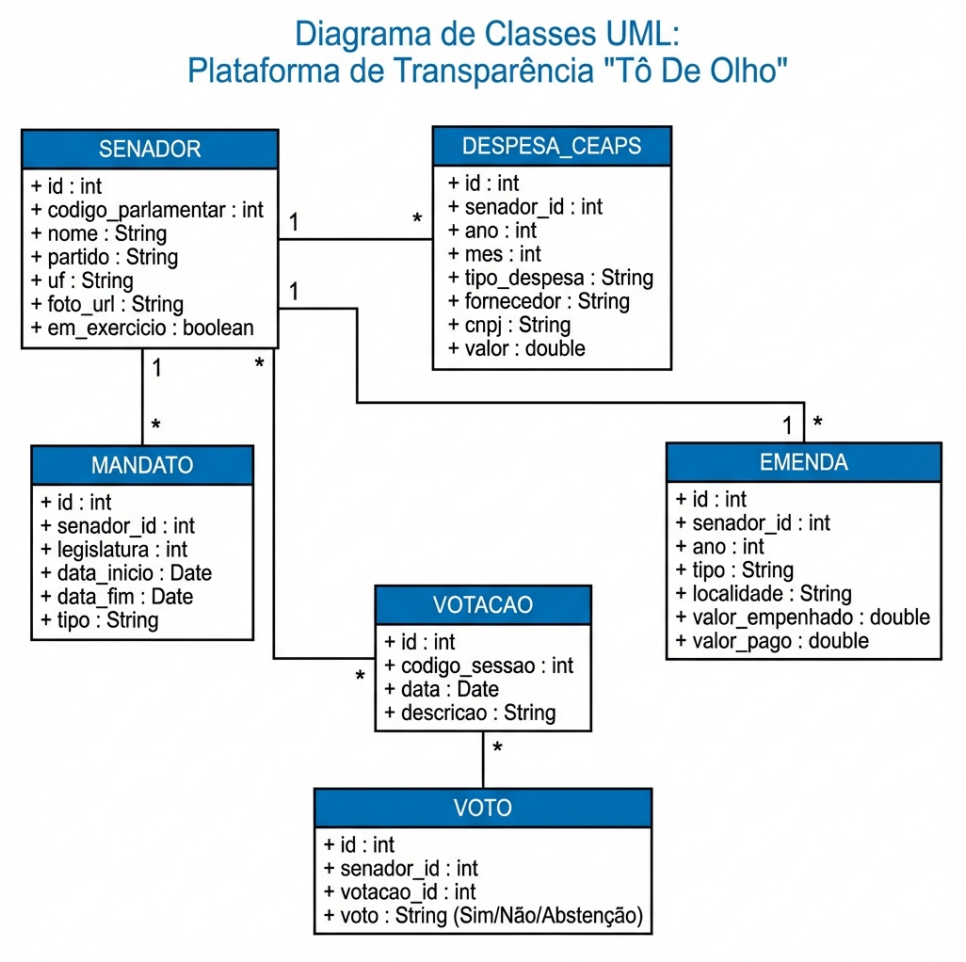
O projeto utiliza a Unified Modeling Language (UML) para documentar visualmente a estrutura e o comportamento do sistema. Abaixo são apresentados os diagramas essenciais para o entendimento da arquitetura proposta.

5.1.1 Diagrama de Classes

O Diagrama de Classes modela a estrutura estática do domínio. As principais entidades identificadas são:

- **Senador**: Representa o parlamentar, contendo atributos como Identificador, Nome, Partido e Estado.
- **Despesa**: Representa um lançamento na CEAPS, associada a um Senador e um Fornecedor.
- **Votacao**: Representa uma sessão deliberativa no Plenário, composta por múltiplos Votos.
- **Emenda**: Representa verba orçamentária destinada pelo senador, contendo Valor, Ano e Beneficiário.

Figura 1: Diagrama de Classes Simplificado (Conceitual)

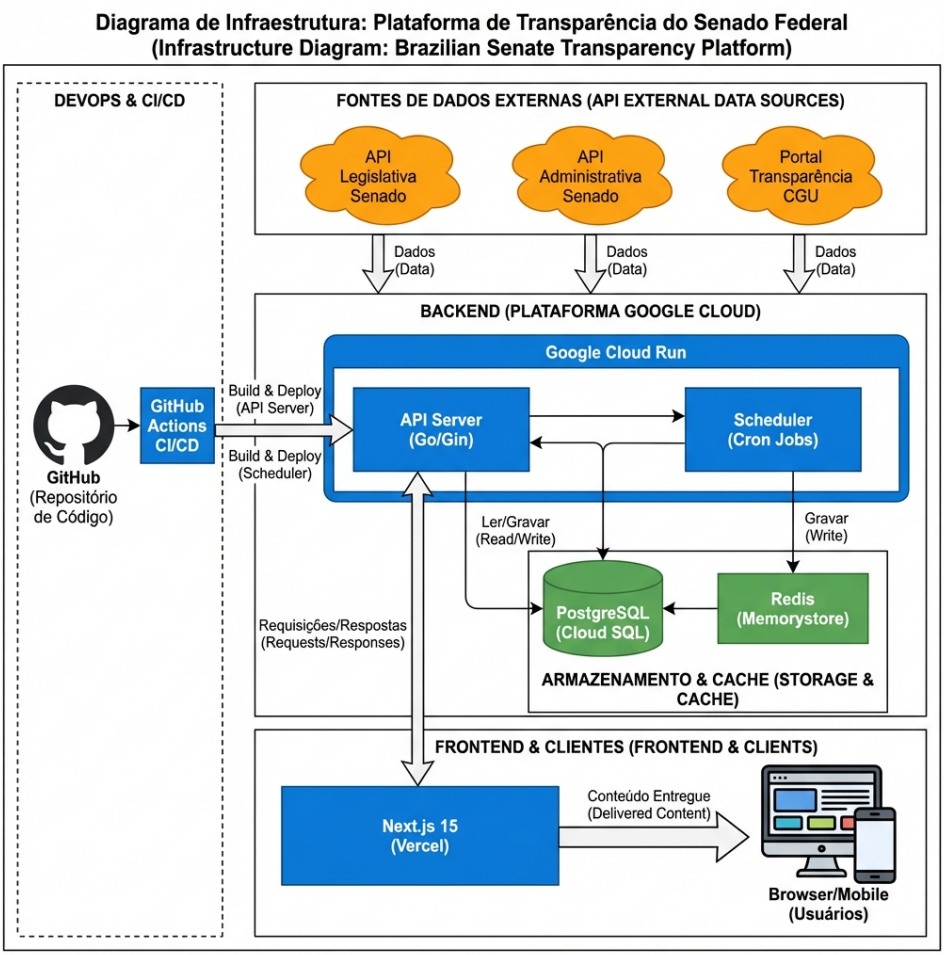


Fonte: Autoria Própria

5.1.2 Diagrama de Implantação

O sistema é implantado em nuvem (GCP), utilizando orquestração de containers.

Figura 2: Diagrama de Implantação e Infraestrutura



Fonte: Autoria Própria

5.2 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-senadores`: Core do domínio, gerencia dados cadastrais e ranking;
- `servico-ceaps`: Responsável pelo processamento e totalização da Cota Parlamentar (ETL dos CSVs);

- `servico-emendas`: Integração com API da Transparência para buscar transferências e convênios;
- `servico-legislativo`: Coleta dados de votações e matérias das APIs do Senado;
- `servico-ingestao`: Workers agendados para atualização diária dos dados.

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 pelo Senado;
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.

5.3 Modelo de Banco de Dados

O modelo de dados relacional foi projetado para garantir integridade e eficiência nas consultas analíticas. As tabelas principais são:

- **TB_SENADORES**: Tabela mestre. PK: `codigo_senado`. Colunas: nome, partido, uf, url_foto.
- **TB_MANDATOS**: Histórico de legislaturas. FK para `TB_SENADORES`.
- **TB_DESPESAS_CEAPS**: Armazena cada nota fiscal reembolsada. FK para `TB_SENADORES`. Colunas: valor, data, tipo_despesa, cnpj_fornecedor, url_documento. Indexada por senador e ano para performance em buscas.
- **TB_VOTACOES**: Cabeçalho das votações. PK: `codigo_sessao`.
- **TB_VOTOS**: Tabela de junção (*many-to-many*) entre Senadores e Votações, registrando o voto individual (Sim/Não).

- **TB_EMENDAS**: Registra valores destinados. Colunas: valor_empenhado, valor_pago, beneficiario, modalidade (ex: Pix).

6 Testes de Software

6.1 Projeto de Testes

A estratégia de qualidade do *Tô De Olho* combina testes em diferentes níveis, aproveitando o ferramental nativo da linguagem Go:

- **Testes Unitários**: Validam regras de negócio isoladas, como o cálculo do ”Score”do senador e parsers de CSV. Implementados com o pacote padrão `testing` do Go, utilizando a técnica de *Table-Driven Tests* para cobrir múltiplos cenários de borda.
- **Testes de Integração**: Validam a comunicação entre os componentes e o banco de dados. Utiliza-se a biblioteca `testcontainers-go` para subir instâncias efêmeras do PostgreSQL e Redis durante a execução da pipeline, garantindo que as queries e a persistência funcionem como esperado num ambiente controlado.
- **Testes de Contrato (API)**: Asseguram que as respostas dos serviços externos (Senado/Transparência) continuam respeitando os formatos esperados, alertando sobre ”Quebras de API”em dependências externas.

7 Implantação

7.1 Projeto de Implantação

A implantação do sistema segue as práticas de *GitOps* e Infraestrutura como Código.

Ambiente de Execução: Os microsserviços são empacotados em imagens Docker otimizadas (*multi-stage builds*) e implantados em um cluster Kubernetes (Google Kubernetes Engine - GKE). O cluster gerencia automaticamente o ciclo de vida dos containers, escalando réplicas (*Horizontal Pod Autoscaling*) com base no uso de CPU/Memória.

Pipeline de CI/CD: Utiliza-se GitHub Actions para automação. A cada *push* na branch principal:

1. *Build*: Compilação dos binários Go e checagem de erros;
2. *Test*: Execução dos testes unitários e de integração;
3. *Publish*: Criação da imagem Docker e envio para o Registry (GCR);
4. *Deploy*: Atualização dos manifestos Kubernetes no cluster de produção.

8 Manual do Usuário Simplificado

A plataforma é pública e não requer cadastro para consulta, maximizando a transparência.

1. **Acesso**: Navegue para ⟨https://todeolho.org.br⟩.
2. **Ranking**: Na página inicial, visualize os ”Top 3”senadores em Economia e Presença.
3. **Busca**: Utilize a barra superior para digitar o nome de um senador.
4. **Detalhes**: Ao clicar em um senador, navegue pelas abas ”Gastos”(para ver notas fiscais detalhadas) e ”Emendas”(para ver destino das verbas).
5. **Fiscalização**: Use o botão ”Compartilhar”para enviar ficha do senador nas redes sociais.

9 Considerações Finais

O projeto *Tô De Olho* atingiu seu objetivo principal de desenvolver uma plataforma *web* capaz de centralizar e democratizar o acesso aos dados do Senado Federal. A arquitetura de *microservices* em Golang, aliada a uma estratégia híbrida de ingestão de dados (API do Senado, Crawlers e Portal da Transparência), provou-se robusta para lidar com a complexidade e a dispersão das informações. A interface desenvolvida em Next.js priorizou a usabilidade e a acessibilidade, oferecendo ao cidadão ferramentas intuitivas para fiscalizar despesas da CEAPS, acompanhar votações e conhecer o perfil de seus 81 representantes.

Entretanto, é importante destacar uma limitação no escopo final da implementação. Inicialmente, planejou-se um módulo de fórum para debate cívico validado. Devido a restrições de tempo hábil para desenvolvimento e testes de segurança rigorosos, este recurso não foi incluído na versão atual. Conforme aponta Costa, a gestão pública digital apoia-se em pilares que incluem não apenas a transparência e a prestação de contas, mas também a participação direta [12]. Embora o *Tô De Olho* avance significativamente na transparência ativa e na *accountability*, a ausência do fórum limita, neste momento, a dimensão da participação deliberativa direta na plataforma.

Como trabalhos futuros, sugere-se a implementação deste espaço de discussão (Fórum de Cidadania), bem como a expansão do escopo para incluir dados da Câmara dos Deputados (tornando o sistema bicameral) e a aplicação de técnicas de aprendizado de máquina para identificar padrões anômalos de gastos na cota parlamentar, fortalecendo ainda mais o controle social.

Agradecimentos

[Agradecimentos a colaboradores do seu projeto]
[Opcional]

Referências

1 GOMES, W. Democracia digital: Que democracia? **Compolítica**, 2010. Disponível em: ⟨http://www.compolitica.org/home/wp-content/uploads/2011/01/gt_ip-wilson.pdf⟩.

2 AVELINO, D. P. de; POMPEU, J. C. B.; FONSECA, I. F. da. Democracia digital: mapeamento de experiências em dados abertos, governo digital e ouvidorias públicas. **Instituto de Pesquisa Econômica Aplicada (Ipea)**, 2021.

3 PATEMAN, C. **Participation and Democratic Theory**. [S.l.]: Cambridge University Press, 1970.

4 GOMES, W. **A democracia no mundo digital**. [S.l.]: Sesc, 2019.

5 ARNSTEIN, S. R. A ladder of citizen participation. **Journal of the American Institute of Planners**, v. 35, n. 4, p. 216–224, 1969.

6 ALBUQUERQUE, O. J. d.; ALMEIDA, B. d.; COSTA, L. V. Valor público por meio de tecnologias desenvolvidas com dados governamentais abertos: o caso da operação serenata de amor. **Revista de Administração Pública**, SciELO Brasil, v. 52, n. 4, p. 610–629, 2018.

7 VOLDEN, C.; WISEMAN, A. E. Legislative effectiveness in the american states. **American Political Science Review**, Cambridge University Press, v. 112, n. 4, p. 1–17, 2018. Metodologia do State Legislative Effectiveness Score (SLES).

8 HULLMAN, J.; DIAKOPOULOS, N. Visualization rhetoric: Framing effects in narrative visualization. **IEEE Transactions on Visualization and Computer Graphics**, IEEE, v. 17, n. 12, p. 2231–2240, 2011.

9 TAIBI, D.; PAHL, C.; LENARDUZZI, V. Processes, motivations, and issues for migrating to microservices architectures: An empirical investigation. **IEEE Cloud Computing**, IEEE, v. 4, n. 5, p. 22–32, 2017.

10 MAUREAL, A. L. et al. Enhancing e-governance through microservices—the development and impact of the ntc-edge system. **Mindanao Journal of Science and Technology**, v. 22, n. Special Issue, p. 260–276, 2024.

11 ULBRICHT, L. Scraping the demos: Digitalization, web scraping and the democratic project. **Democratization**, Taylor & Francis, v. 27, n. 3, p. 426–442, 2020.

12 COSTA, E. A. D. Gestão pública digital: O poder das tic na democracia brasileira. **Revista de Administração Pública**, v. 25, p. 123–145, 2019.

[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 referências no arquivo "referencias.bib" e utilize o comando "cite"]

Apêndices

Glossário, Siglas e Abreviações

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