PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS

ESCOLA POLITÉCNICA E DE ARTES

GRADUAÇÃO EM ENGENHARIA DE COMPUTAÇÃO

ARTUR SOUZA DIAS

Aplicativo para análise de feedback de clientes para identificar padrões de satisfação

GOIÂNIA
2025

ARTUR SOUZA DIAS

Aplicativo para análise de feedback de clientes para identificar padrões de satisfação

Trabalho de Conclusão de Curso apresentado à Escola de Ciências Exatas e da Computação, da Pontifícia Universidade Católica de Goiás, como parte dos requisitos para a obtenção do título de Bacharel em Engenharia de Computação.

Orientador: Professor Me. André Luiz Alves

GOIÂNIA
2025

ARTUR SOUZA DIAS

Aplicativo para análise de feedback de clientes para identificar padrões de satisfação

Trabalho de Conclusão de Curso aprovado em sua forma final pela Escola de Ciências Exatas e de Computação, da Pontifícia Universidade Católica de Goiás, para obtenção do título de Bacharel em Engenharia de Computação,
Aprovado em: _____/_____/_____ .

_____

Orientador: Prof. Me. André Luiz Alves

_____

Defensor: Prof. Me. Daniel Corrêa da Silva

_____

Defensor: Prof. Me. Eugênio Júlio M.C Carvalho

GOIÂNIA
2025

RESUMO

Neste trabalho se apresenta a implementação de um sistema web para análise de feedbacks de clientes, com o foco de identificar padrões de satisfação e pontos que necessitam de análise. Essa aplicação permite que os clientes avaliem o atendimento recebido por meio de notas e comentários, cujos dados são armazenados em tempo real no Firebase Firestore.Os feedbacks são categorizados como positivos ou negativos e analisados para detectar padrões recorrentes, o que possibilita a geração de insights sobre a qualidade do atendimento. A repetição de termos-chave negativos permite indicar tendências de insatisfação. O sistema conta com visualizações gráficas dinâmicas através da biblioteca Chart.js, além de uma aba de filtragem por funcionário, exportação de dados em CSV e acesso restrito ao painel de feedbacks, disponível apenas pelo gestor. A aplicação foi desenvolvida com HTML, CSS e JavaScript, utilizando o Firebase para armazenamento de dados, autenticação de usuários e hospedagem.

Palavras-chave: Análise de feedbacks; Padrões de satisfação; Engenharia de software.

ABSTRACT

This work presents the implementation of a web-based system for analyzing customer feedback, with a focus on identifying satisfaction patterns and areas that require further analysis. The application allows customers to evaluate the service received through ratings and comments, with the data being stored in real time using Firebase Firestore.The feedback is categorized as either positive or negative and analyzed to detect recurring patterns, enabling the generation of insights into the quality of service. The repetition of negative keywords helps indicate trends of dissatisfaction.The system features dynamic graphical visualizations using the Chart.js library, as well as a filtering tab by employee, CSV data export, and restricted access to the feedback panel, which is only accessible by the manager. The application was developed using HTML, CSS, and JavaScript, with Firebase handling data storage, user authentication, and hosting.

Keywords: Feedback analysis; Satisfaction patterns; Engineering software.

LISTA DE FIGURAS

SUMÁRIO

1.INTRODUÇÃO

Nas empresas atualmente, a análise de feedbacks de clientes tornou-se essencial para identificar padrões de satisfação e pontos que necessitam de atenção como por exemplo os feedbacks negativos dos clientes. Com o crescimento das plataformas digitais, as empresas enfrentam desafios cada vez maiores para analisar de forma eficiente as opiniões dos consumidores. Pois cada um tem uma forma de pensar diferente.

A coleta e a análise estruturada desses dados tem como objetivo monitorar as interações com os clientes, revelando tendências positivas e negativas. No entanto, não basta apenas reunir essas informações: é fundamental contar com um sistema, capaz de organizar, categorizar e interpretar os feedbacks desses clientes, transformando-os em insumos estratégicos.

Pensando assim, foi desenvolvido um aplicativo web para a coleta e análise de opiniões dos clientes, com foco na identificação de padrões de atendimento e análises de como o colaborador atendeu esse cliente.Aplicação permite que os usuários avaliem o desempenho dos colaboradores por meio de notas e comentários. Essas informações são armazenadas no Firebase e apresentadas em um dashboard, implementado com HTML, JavaScript e a biblioteca Chart.js. O gestor tem acesso a gráficos dinâmicos, pode filtrar os dados por funcionário e exportar os resultados em formato CSV.

Tal proposta representa uma solução funcional para empresas que desejam acompanhar a experiência do cliente de maneira mais estratégica. O principal objetivo é fornecer uma ferramenta capaz de aprimorar a compreensão sobre a satisfação dos clientes, contribuindo para a melhoria dos serviços prestados e auxiliando os gestores, para assim o mesmo ter uma melhor análise de como seus colaboradores estão fazendo as tratativas com o cliente.

Esse trabalho, busca ajudar as empresas a compreenderem melhor as necessidades, expectativas e feedbacks negativos dos clientes, usando dados coletados diretamente de suas opiniões.

2. ENGENHARIA DE SOFTWARE

A engenharia de software é definida por Roger Pressman (2016) como a aplicação de uma abordagem sistemática, disciplinada e quantificável ao desenvolvimento, operação e manutenção de software. O objetivo dessa disciplina é garantir a criação de sistemas de alta qualidade que atendam às expectativas dos usuários e clientes, combinando práticas técnicas com gerenciamento eficiente de projetos. Ian Sommerville (2011) complementa, afirmando que a engenharia de software abrange desde a especificação dos requisitos até a manutenção do sistema**.**

Especificação de Requisitos**:** Nesta fase, define-se o que o sistema deve fazer, além de identificar as restrições e funcionalidades essenciais. Uma especificação clara e precisa é fundamental para garantir que o produto final atenda às expectativas do cliente (SOMMERVILLE, 2011).

Design e Arquitetura**:** A arquitetura do sistema é planejada de forma modular para facilitar a implementação e a manutenção. Um design bem estruturado garante que o sistema seja escalável e compreensível ao longo do tempo (PRESSMAN, 2016).

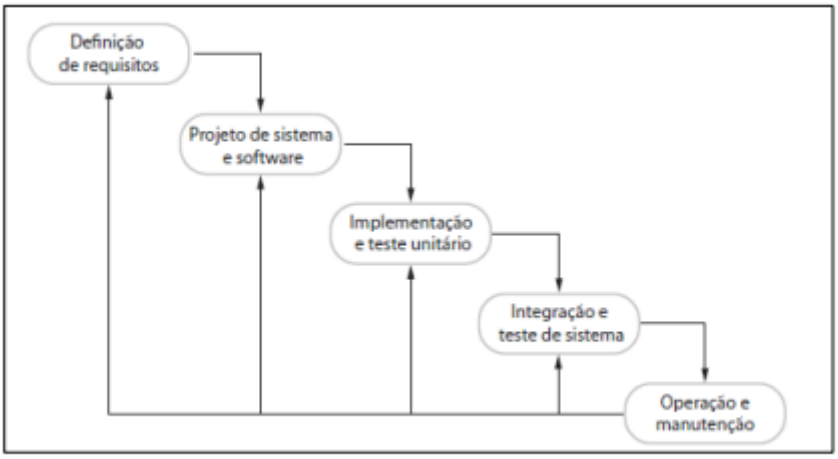
Implementação e Codificação: Baseando-se no design, esta fase envolve a programação e construção do software. O uso de padrões de codificação e testes contínuos são fundamentais para garantir um código de alta qualidade (SOMMERVILLE, 2011).

Validação e Testes: Os testes garantem que o software funcione corretamente e atenda aos requisitos estabelecidos. É essencial planejar e executar essa fase de forma sistemática para identificar e corrigir defeitos (Pressman, 2016).

Manutenção**:** Após a entrega, o software pode passar por correções e atualizações para adaptar-se às novas necessidades ou resolver problemas. A manutenção pode ser corretiva, adaptativa ou evolutiva (Sommerville, 2011).

Os processos de software podem seguir diferentes modelos. Um dos mais conhecidos é o modelo cascata, que segue uma abordagem sequencial. Nesse

modelo, cada fase do desenvolvimento é iniciada somente após a conclusão da anterior. Embora fácil de entender e aplicar, ele é mais adequado para projetos onde os requisitos são estáveis desde o início, pois é difícil realizar alterações significativas nas fases anteriores (PRESSMAN, 2016).



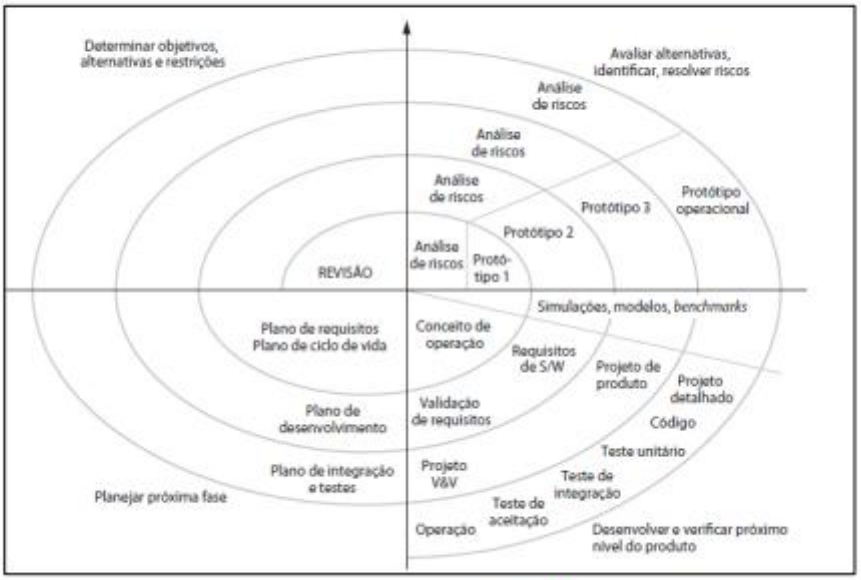
Fonte: Sommerville, 2011.

Figura 1 - Modelo em Cascata

O modelo em espiral é um modelo similar ao modelo cascata, porém envolve uma série de ciclos em que as atividades de desenvolvimento são repetidas e refinadas em cada iteração como ilustrado na Figura 2. É adequado para projetos com requisitos completos ou incertos. Uma das vantagens de se usar esse modelo é sua flexibilidade e adaptação, onde permite avaliação constante dos riscos e ajustes no processo e a comunicação com o cliente é facilitada. No entanto, entre suas desvantagens estão, pode ser complexo e exigir uma equipe experiente para codificar, pode ter atrasos e custos mais altos devido à avaliação constante dos riscos (PRESSMAN, 2016).

Figura 2 - Modelo em Espiral

Figura 2 – Modelo em espiral.



Fonte: Sommerville, 2011.

2.1 Engenharia de Requisitos

Engenharia de requisitos é uma disciplina focada na coleta, análise, especificação, validação e gerenciamento dos requisitos necessários para o desenvolvimento de um sistema. Segundo Sommerville (2011), a elicitação de requisitos é uma fase crítica desse processo, na qual se busca compreender as necessidades e expectativas dos stakeholders, garantindo que o sistema final esteja alinhado com as demandas e restrições do projeto.

Assim como a Engenharia de software, a Engenharia de Requisitos tem várias etapas sendo essas:

Elicitação de Requisitos**:** A elicitação envolve a coleta de informações de diversas partes interessadas por meio de entrevistas, questionários, workshops, observação e prototipagem. O objetivo é identificar requisitos funcionais e não funcionais. Durante essa fase, a documentação também pode ser analisada para obter informações adicionais ou identificar restrições.

Análise e Negociação: Após a coleta, os requisitos são analisados para que esteja tudo claro, consistentes e relevantes. A negociação entre as partes

interessadas é fundamental para resolver conflitos e alinhar expectativas. Pode-se usar ferramentas como diagramas e matrizes de rastreabilidade, que permitem rastrear a correspondência entre requisitos e funcionalidades do sistema.

Especificação de Requisitos: Consiste em formalizar e documentar os requisitos de maneira precisa, permitindo que a equipe de desenvolvimento os utilize como base para projetar e implementar o sistema. A especificação pode incluir requisitos em formato textual, casos de uso ou histórias de usuário.

Validação de Requisitos: A validação é essencial para confirmar que os requisitos documentados refletem corretamente as necessidades do cliente e que o sistema desenvolvido atenderá às por completo as exigências do cliente.

Gerenciamento de Requisitos**:** Como os requisitos podem evoluir durante o projeto, o gerenciamento contínuo é necessário para garantir que mudanças sejam controladas e documentadas.

2.2 Projeto de Software

Projeto de software envolve a definição da arquitetura, componentes, interfaces e outros elementos que compõem o sistema. Nesta etapa, foram elaborados os diagramas de caso de uso, fluxogramas e a arquitetura do sistema, conforme os requisitos identificados. O sistema foi projetado com uma estrutura composta por Frontend, Backend e banco de dados em nuvem.

No Frontend, foram utilizados HTML, CSS e JavaScript, linguagens que viabilizaram o desenvolvimento da interface e das interações com o usuário. Para a visualização gráfica dos resultados, foi adotada a biblioteca Chart.js, que permite a criação de gráficos interativos e responsivos (CHART.JS, 2023).

O sistema conta com uma arquitetura baseada em nuvem, utilizando o Firebase, que oferece serviços de banco de dados (Firestore), autenticação de usuários e hospedagem da aplicação (GOOGLE, 2023). A plataforma Firebase possibilitou uma estrutura Backend simplificada e escalável, eliminando a necessidade de infraestrutura de servidores locais.

O projeto de software é a base da engenharia de software, pois é nesta fase que se define como os requisitos serão transformados em uma solução técnica. O projeto de software é a base da engenharia de software, pois o mesmo explica de como os requisitos foram implementados Pressman (2016)

2.3 Produção

Na fase de produção corresponde ao desenvolvimento prático do sistema. Nessa etapa, foram criados os arquivos HTML, CSS e JavaScript que contêm a interface e funcionalidades do sistema.
Foram implementadas funções como o formulário de feedback com estrelas, a autenticação para clientes e gestores, o envio de dados ao Firestore e a visualização de resultados por meio de dashboards.
Aplicação foi testada durante o processo de codificação, e o Firebase foi configurado para o deploy via Firebase Hosting (GOOGLE, 2023).
Conforme Sommerville (2011), a produção deve seguir boas práticas de codificação e versionamento, permitindo uma implementação estruturada e sustentável.

2.4 Testes

A fase de testes foi realizada com o objetivo de garantir a funcionalidade, a usabilidade e a confiabilidade do sistema. Foram utilizados testes exploratórios manuais com diferentes perfis de usuários (cliente e gestor) para simular o uso .
Foram testadas as principais funcionalidades, incluindo:

- Envio de feedbacks com diferentes notas e comentários;
- Filtragem por funcionário no dashboard;
- Exportação para CSV;
- Restrições de acesso ao gestor;
- Login do cliente com nome e data.

3. ESTUDO DE CASO

Muitas empresas não possuem esse tipo de sistema de coleta de feedback dos clientes, o que prejudica muito, pois a empresa não sabe se está tendo um bom atendimento ou um péssimo atendimento, e também não sabe como melhorar a partir desses erros.

Esse trabalho será um aplicativo que irá ajudar as empresas que possuem contato direto com o cliente, sendo esses os principais autores para a coleta do feedback. Assim, isso melhoraria a experiência da empresa, pois aumentaria as oportunidades de crescimento.

3.1 Referencial Teórico

O referencial teórico deste trabalho apresenta as bases para o desenvolvimento do sistema de análise de feedbacks de clientes. Ele abrange conceitos fundamentais sobre engenharia de software, processamento de linguagem natural (PLN) e visualização de dados, com o objetivo de fornecer uma base sólida para a implementação da solução proposta

3.1 Análise de Feedback de Clientes

Análise de feedbacks é uma prática de suma importância para empresas que buscam melhorar sua relação com os clientes e otimizar seus serviços. O feedback dos clientes é uma das principais fontes de informação para identificar problemas e áreas de melhoria, além de possibilitar uma melhor compreensão das expectativas dos consumidores.

3.2 Visualização de Dados com Chart.js

A visualização de dados é uma etapa fundamental para facilitar a interpretação dos resultados. Neste sistema, a biblioteca Chart.js (CHART.JS, 2023) foi empregada para a construção de gráficos de barra que exibem a quantidade de avaliações negativas, neutras e positivas. A interface do dashboard permite ao gestor filtrar os resultados por funcionário, tornando possível identificar padrões de atendimento e auxiliar em decisões gerenciais.

3.3 Análise de Feedback e Processamento de Linguagem Natural (PLN)

O sistema realiza uma análise textual simples para identificar padrões de melhoria nos comentários. Isso é feito por meio da contagem de palavras negativas mais frequentes, como “demorado”, “ruim” ou “lento”. Apesar de não utilizar técnicas avançadas de PLN, esse tipo de análise básica já permite extrair informações úteis para melhoria do atendimento. Essa abordagem foi implementada diretamente em JavaScript e integrada ao sistema sem uso de bibliotecas externas de PLN

3.4 Armazenamento de Dados com Firebase

Para o armazenamento dos feedbacks, utilizou-se o Firebase Firestore (GOOGLE, 2023), um banco de dados NoSQL em tempo real mantido pela Google. O Firestore permite armazenar e recuperar documentos JSON organizados em coleções, de forma escalável e eficiente, eliminando a necessidade de configuração de servidores. Esta escolha viabiliza a persistência dos dados do feedback, incluindo nome do cliente, data, funcionário avaliado, nota e comentário textual.

3.4.1 Funcionalidade do firestore

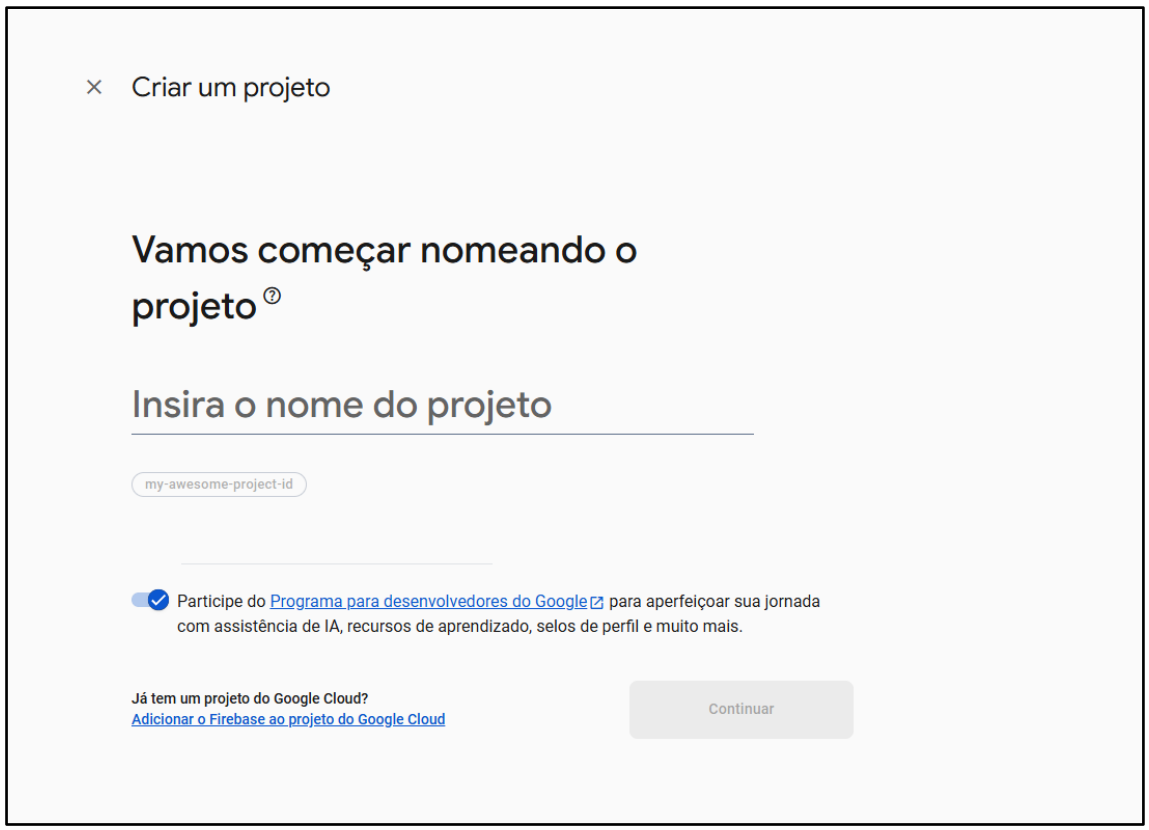
O primeiro acesso a plataforma se inicia com uma guia, onde mostra como criar o primeiro projeto no software. É feita uma pergunta de como será o nome dos projetos, as primeiras tarefas, os status primário de cada uma delas

Figura 3 - Página inicial do Firebase

Fonte: Elaborado pelo autor

Nessa segunda parte a plataforma pede para o usuário inserir um nome do projeto

Figura 4 - Página inicial do Firebase de configuração



Fonte: Elaborado pelo autor

Ela possui uma assistência de IA da google (gemini) que o usuário pode ativar ou não no seu projeto

Figura 5 - IA do Firebase

Fonte: Elaborado pelo autor

Além da IA a plataforma conta com um sistema de análise que segmenta em gerar diversos relatórios. Assim como demostra na imagem anterior o usuário pode ou não aceitar ativar o google Analytics. Caso ele aceite ser necessários fazer as devidas configurações

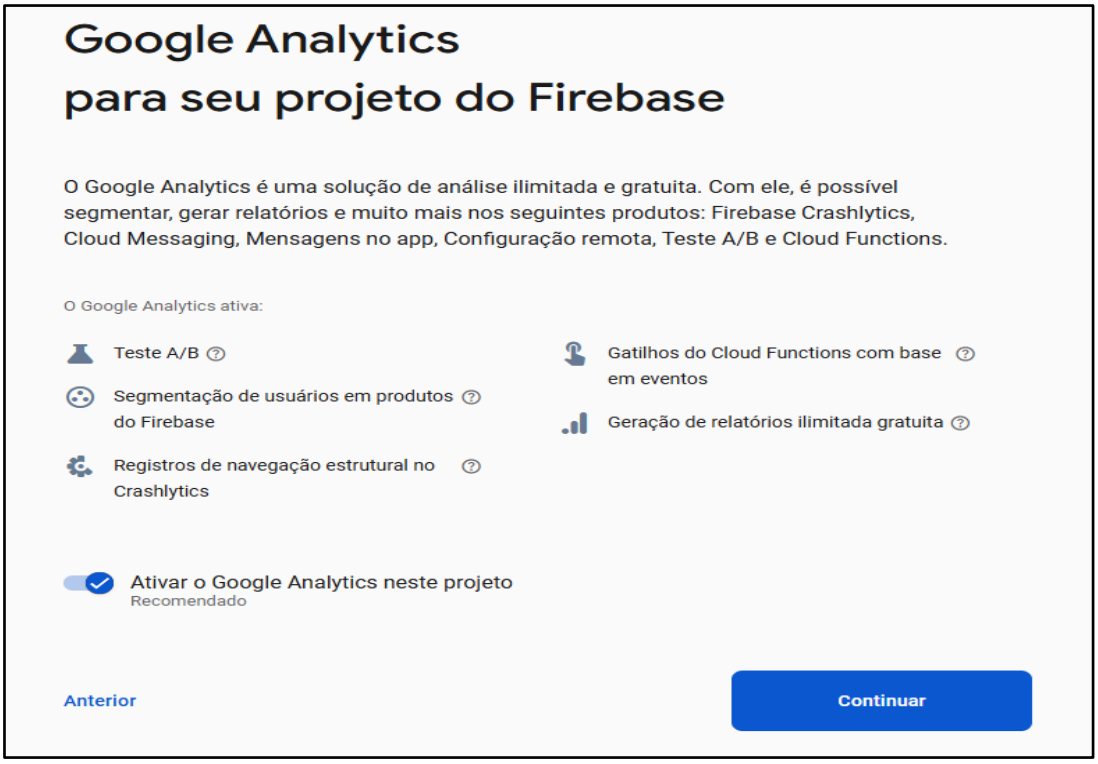
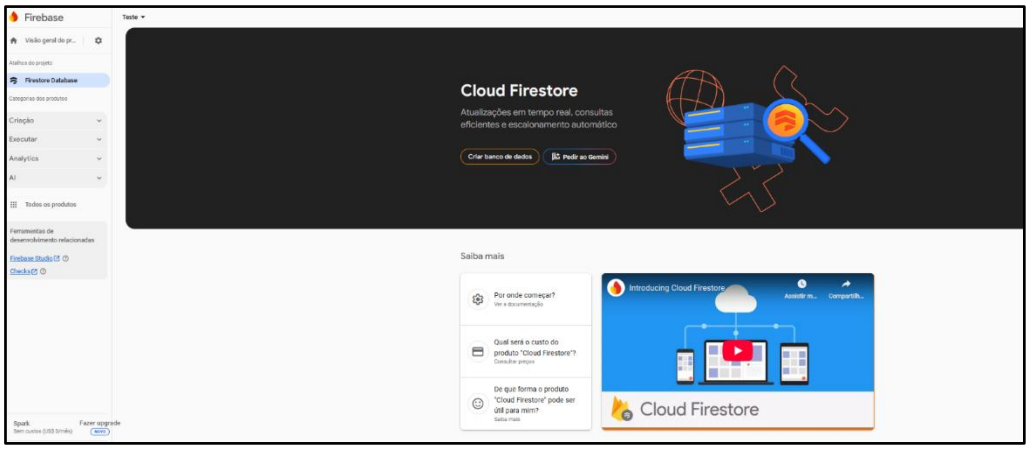


Figura 6- Criação do Banco de Dados

Fonte: Elaborado pelo autor

Após todas as configurações o usuário pode criar seu banco de dados no cloud Firestore (banco de dados na nuvem).

Figura 7 - Layout principal do firebase



Fonte: Elaborado pelo autor

3.5 Segurança e Controle de Acesso

O controle de acesso é realizado por meio de um login exclusivo para o gestor, com credenciais fixas. O painel de resultados (dashboard) é protegido e visível apenas ao gestor. Os clientes, por sua vez, acessam apenas a interface de envio de feedbacks, sem visualizar os dados coletados. Essa separação garante a privacidade dos dados e mantém o foco de cada tipo de usuário.

3.6 Banco de Dados e Visualização de Dados

Foi usado o Firebase Firestore que é um de banco de dados NoSQL em tempo real da plataforma Google Firebase. Firestore foi escolhido por ser uma solução escalável, de fácil integração com aplicações web e que dispensa a necessidade de um servidor backend dedicado, sendo ideal para sistemas de empresas de pequeno porte

Além disso, a utilização de dashboards permite a visualização clara e intuitiva das tendências e padrões identificados. Shneiderman (2016) afirma que gráficos e relatórios visuais são ferramentas indispensáveis para transformar dados complexos em informações úteis, ajudando gestores a tomar decisões estratégicas com maior eficiência.

4. PROPOSTA DE SOLUÇÃO

Esta seção descreve a solução desenvolvida para enfrentar o problema da ausência de mecanismos adequados de coleta e análise de feedbacks dos clientes. A proposta consiste em um sistema web que permite aos clientes avaliarem o atendimento recebido por meio de notas e comentários, os quais são armazenados em tempo real no banco de dados Firebase Firestore.

O sistema possui dois tipos de usuários: cliente e gestor. Os clientes acessam uma interface simples, onde informam nome e data, atribuem uma nota ao atendimento e deixam um comentário. Os gestores, por sua vez, acessam um painel administrativo protegido por autenticação, onde visualizam os feedbacks por meio de gráficos dinâmicos e podem filtrar os dados por funcionário, exportar os resultados em formato CSV e apagar registros antigos

4.1 Objetivo do Estudo de Caso

Este estudo de caso demonstra como o sistema proposto pode transformar o processo de gestão de feedbacks promovendo:

- Centralização de feedbacks em um banco de dados unificado.
- Processamento automático para identificar padrões e tendências.
- Fornecimento de insights estratégicos por meio de dashboards interativos.
- Melhoria da experiência do cliente e aumento da satisfação geral

4.2 Requisitos Funcionais

Os requisitos funcionais definem o comportamento esperado do sistema, ou seja, as funcionalidades que ele deve oferecer aos usuários para atender aos objetivos propostos. Abaixo temos alguns dos requisitos funcionais:

- ID01: Permitir que clientes enviem feedbacks com nome e data
- ID02: Permitir login de gestores com credenciais específicas*
- ID03: Exibir feedbacks com gráfico filtrável por funcionário
- ID04: Permitir limpeza de feedbacks (somente por gestores)

- ID05: Exibir nome e data do cliente no dashboard

Tabela completa de requisitos está disponível no Apêndice A.

4.3 Requisitos não funcionais

Os requisitos não funcionais determinam as características de qualidade do sistema, influenciando diretamente na experiência do usuário, segurança dos dados e performance da aplicação. Abaixo estão descritos os principais requisitos não funcionais do sistema desenvolvido:

- IDN01 :O sistema deve processar até 50 feedbacks por dia sem degradação no desempenho.
- IDN02: Deve garantir que dados sensíveis sejam armazenados com criptografia.
- DN03: O sistema deve ter interface responsiva para ser acessado em dispositivos móveis.

Tabela completa de requisitos não funcionais está disponível no Apêndice B.

4.2 Aplicação do Sistema

Integração de Dados:

- O sistema é acessado por dois tipos de usuários: cliente e gestor.
- Apenas o gestor tem acesso ao dashboard, e o login é feito com credenciais fixas.
- O cliente informa nome e data antes de enviar um feedback.
- A análise é feita com base nas palavras dos comentários
- O sistema possui botões de exportação dos dados em CSV e limpeza dos feedbacks (restrito ao gestor).
- Dashboards foram desenvolvidos para mostrar:
- Porcentagem de feedbacks por tipo.
- Tendências de feedbacks negativos ao longo do tempo.

4.3 Resultados Esperados

A implementação proposta do sistema tem o potencial de trazer melhorias significativas para empresas, como:

Redução no Tempo de Análise: Possibilidade de processar feedbacks automaticamente em minutos.

Maior Eficiência: Identificação mais rápida de problemas prioritários, como atrasos no atendimento.

Tomada de Decisão Baseada em Dados: Relatórios mais claros e organizados para embasar decisões estratégicas.

4.4 Impacto na Empresa

Satisfação do Cliente: O sistema proposto permite o acompanhamento em tempo real dos feedbacks, o que poderá possibilitar ações rápidas para resolver problemas críticos.

Indicadores de Qualidade: Com a utilização contínua, espera-se um crescimento na pontuação de satisfação geral dos clientes ao longo do tempo.

Oportunidades de Crescimento: A análise automatizada dos comentários pode gerar insights valiosos, auxiliando empresas na identificação de sugestões de melhorias em produtos e serviços.

5 ARTEFATOS DO PROJETO

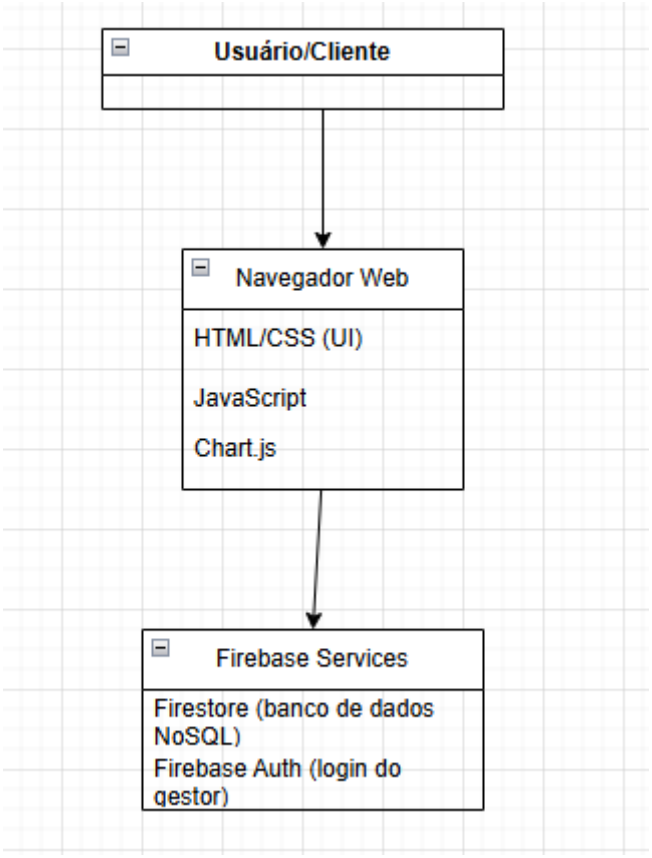
Durante o desenvolvimento do sistema, foram produzidos diversos artefatos de engenharia de software, os quais serviram como base para garantir clareza no entendimento dos requisitos e na estrutura da solução

5.1 Tabela de Arquitetura do Sistema

Componente	Descrição	Tecnologia
Interface do Usuário	Interface web para envio de feedbacks e visualização de resultados (dashboard).	HTML, CSS e JavaScript
Banco de Dados	Armazena os feedbacks, nomes, datas e usuários.	Firebase Firestore
Análise de Feedback	Identifica padrões negativos por repetição de palavras.	JavaScript
Autenticação	Controla o acesso ao dashboard (apenas o gestor pode acessar).	Firebase Authentication

Imagem demonstrando o diagrama de Arquitetura baseado no item 5.3.1

Figura 8 - Diagrama da Arquitetura



Fonte: Elaborado pelo Autor

5.2 Modelo de Dados

Entidade	Atributos	Descrição
Feedback	ID, funcionário, nota, comentário, cliente, data	Armazena as informações de cada feedback recebido
Cliente	Nome, data (informada manualmente pelo cliente no formulário)	Detalha informações do cliente que enviou o feedback.

5.3 Fluxograma de Processamento de Dados

Etapa	Descrição
Coleta de Feedbacks	Usuário preenche formulário com nota, comentário, nome e data.
Armazenamento	Os dados são armazenados automaticamente no Firebase Firestore.
Análise de Comentários	Script verifica palavras negativas recorrentes para sugerir melhorias.
Geração de Dashboard	Gera gráfico de barras com filtro por funcionário e classifica por cores.

5.4 Tecnologias Adotadas

Componente	Tecnologia	Motivação
Frontend	HTML + JavaScript	Criação de interfaces dinâmicas e responsivas.
Backend	Firebase Realtime Database	Armazenamento em nuvem integrado e escalável com banco de dados.
Banco de Dados	Chart.js	Criação de gráficos interativos com simplicidade
Processador de NLP	Firebase Hosting	Deploy rápido e gratuito. Análise textual avançada para categorizar e identificar

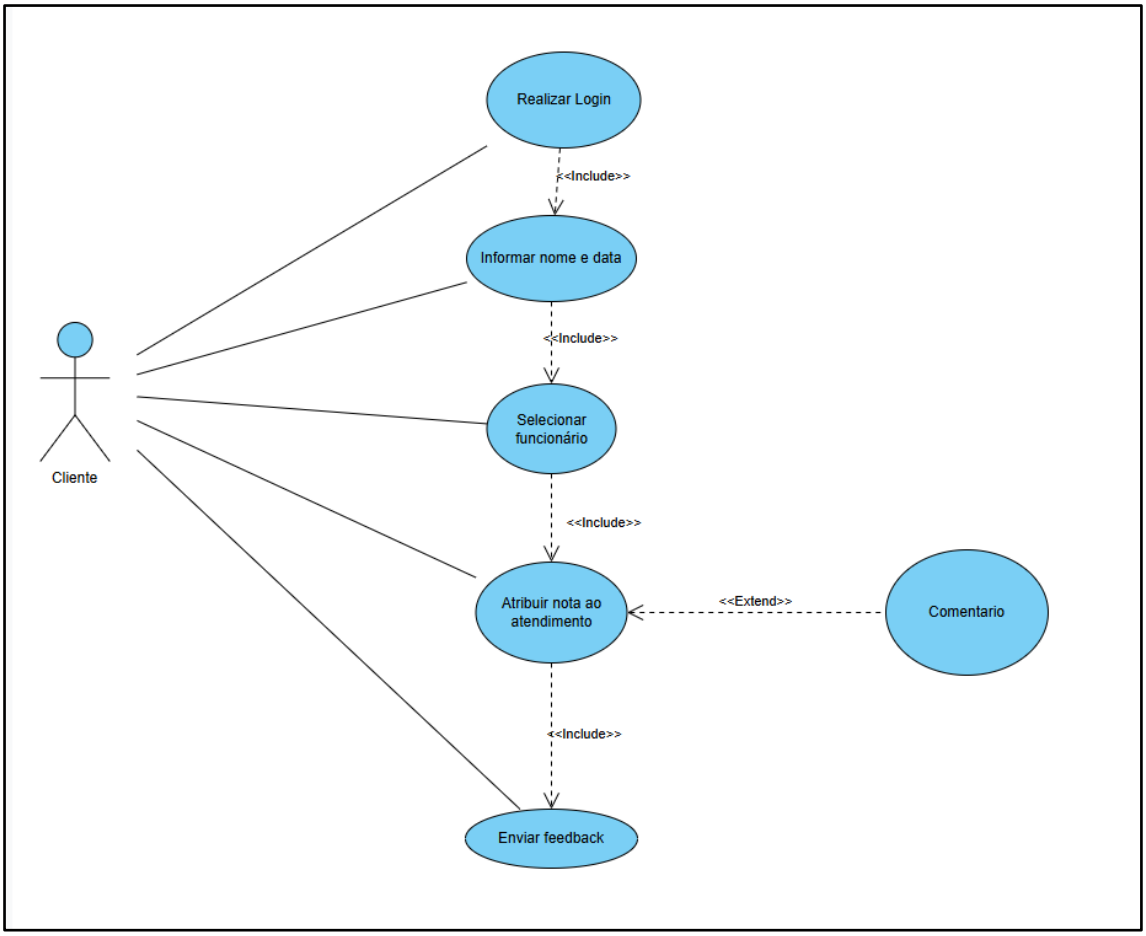
		padrões nos feedbacks.

5.5 CASOS DE TESTE

Nome	**Descrição**
Envio de Feedback	Garante que os clientes enviam o feedback por meio do site criado sendo os dados devem ser corretamente armazenados no banco de dados
Teste de Performance	Tempo de Resposta: o sistema processa um grande volume de feedback e quando estourar a meta o mesmo salva os feedbacks e apaga os atuais da memória
Teste de Usabilidade	Interface de feedback: testa se a interface e fácil de interagir e fácil de manusear para assim enviar um feedback claro para o cliente

6 DIAGRAMA DE CASO DE USO

Figura 8 - Diagrama de caso de uso



Fonte: Elaborado pelo autor

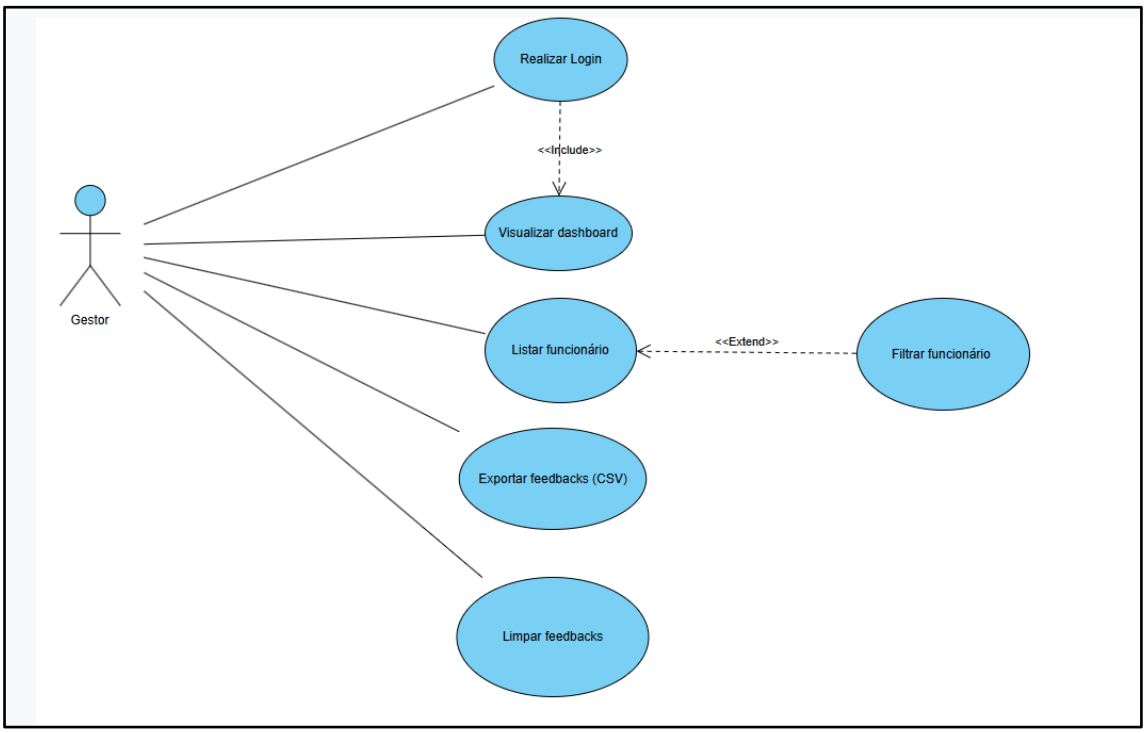
O diagrama acima representa os principais casos de uso do sistema do ponto de vista do cliente, que é o ator envolvido nesta interação. O foco é o processo de login e envio de feedback sobre o atendimento recebido por parte dos funcionários da empresa.

Descrição do caso de uso:

- Realizar login: o cliente inicia o uso do sistema acessando a tela de login, onde deve informar nome e data antes de acessar a funcionalidade de feedback.
- Informar o nome e data: esses dados são coletados para registro e identificação básica do cliente no momento da avaliação.
- Selecionar o funcionário: após o login, o cliente deve escolher o funcionário que realizou o atendimento a ser avaliado.

- Atribuir nota ao atendimento: o cliente fornece uma nota quantitativa sobre o atendimento recebido, utilizando um sistema de estrelas ou deslizador.
- Enviar feedback: o cliente insere um comentário opcional e envia a avaliação. Os dados são então armazenados no banco de dados Firebase Firestore e posteriormente utilizados para análise no dashboard do gestor.

Figura 9 - Diagrama de caso de uso do Gestor



Fonte: Elaborado pelo autor

O diagrama acima apresenta os casos de uso relacionados às ações realizadas pelo gestor do sistema. O gestor é o ator responsável por acessar e analisar os feedbacks fornecidos pelos clientes, com o intuito de tomar decisões baseadas em dados reais de atendimento.

Descrição dos casos de uso:

- Realizar login: o gestor inicia a sessão informando seu email e senha cadastrados, tendo acesso exclusivo ao painel de resultados.

- Visualizar dashboard: ao entrar no sistema, o gestor pode acessar o dashboard com gráficos e análises dos feedbacks.
- Listar funcionário: os dados exibidos no dashboard incluem feedbacks de todos os funcionários que foram avaliados.
- Filtrar funcionário: extensão do caso "Listar funcionário".
- Baixar CSV: função que exporta os feedbacks em formato CSV.

7. CONCLUSÃO

Este trabalho teve como objetivo desenvolver uma aplicação web capaz de coletar e analisar feedbacks de clientes, visando identificar padrões de

satisfação dos clientes. Foi construído um sistema utilizando tecnologias como HTML, CSS, JavaScript, Firebase e a biblioteca Chart.js.

No processo de desenvolvimento, foram abordados conceitos da engenharia de software, banco de dados em nuvem e visualização de dados. O sistema criado permite que clientes avaliem o atendimento recebido com base em uma escala de 0 a 5, além de escreverem comentários de como foi a experiencia do atendimento. As avaliações são armazenadas no banco de dados Firestore e exibidas em um dashboard, sendo esse acessível apenas ao gestor do departamento, assim facilitando para o gestor uma melhor forma de administrar seus colaboradores.

A análise dos comentários é realizada por meio de um processo de identificação de palavras negativas recorrentes, o que permite ao gestor analisar como funcionário está tratando os clientes. O sistema tem recursos como exportação dos dados em CSV para abrir em planilhas do Excel, filtro por funcionário e limpeza de feedbacks antigos, garantindo ao gestor uma maior facilidade de análise.

Ao final da implementação do trabalho, foi feita a validação de todos os objetivos, e todos foram atendidos. O sistema desenvolvido cumpre seu papel como uma ferramenta de apoio à gestão da qualidade no atendimento, promovendo uma abordagem baseada em dados reais e contribuindo para a melhoria contínua dos serviços prestados.

REFERÊNCIAS BIBLIOGRÁFICAS

CHART.JS. *Chart.js Documentation.* 2023. Disponível em: https://www.chartjs.org/docs/latest/. Acesso em: 27 maio 2025.

GOOGLE. *Firebase Documentation.* 2023. Disponível em: https://firebase.google.com/docs. Acesso em: 27 maio 2025.

JURAFSKY, Daniel; MARTIN, James H. *Speech and Language Processing.* Pearson, 2022.

MOZILLA DEVELOPER NETWORK (MDN). *JavaScript Reference.* Disponível em: https://developer.mozilla.org/pt-BR/docs/Web/JavaScript. Acesso em: 27 março 2025.

PRESSMAN, R. S. *Engenharia de software: uma abordagem profissional.* 8. ed. McGraw-Hill, 2016.

SOMMERVILLE, I. *Engenharia de Software.* São Paulo: Pearson, 2011.

W3SCHOOLS. *HTML, CSS and JavaScript Tutorials.* Disponível em: https://www.w3schools.com/. Acesso em: 27 março 2025.

ZEITHAML, Valarie A.; BITNER, Mary Jo; GREMLER, Dwayne D. *Marketing de Serviços: A empresa com foco no cliente.* McGraw-Hill, 2014.

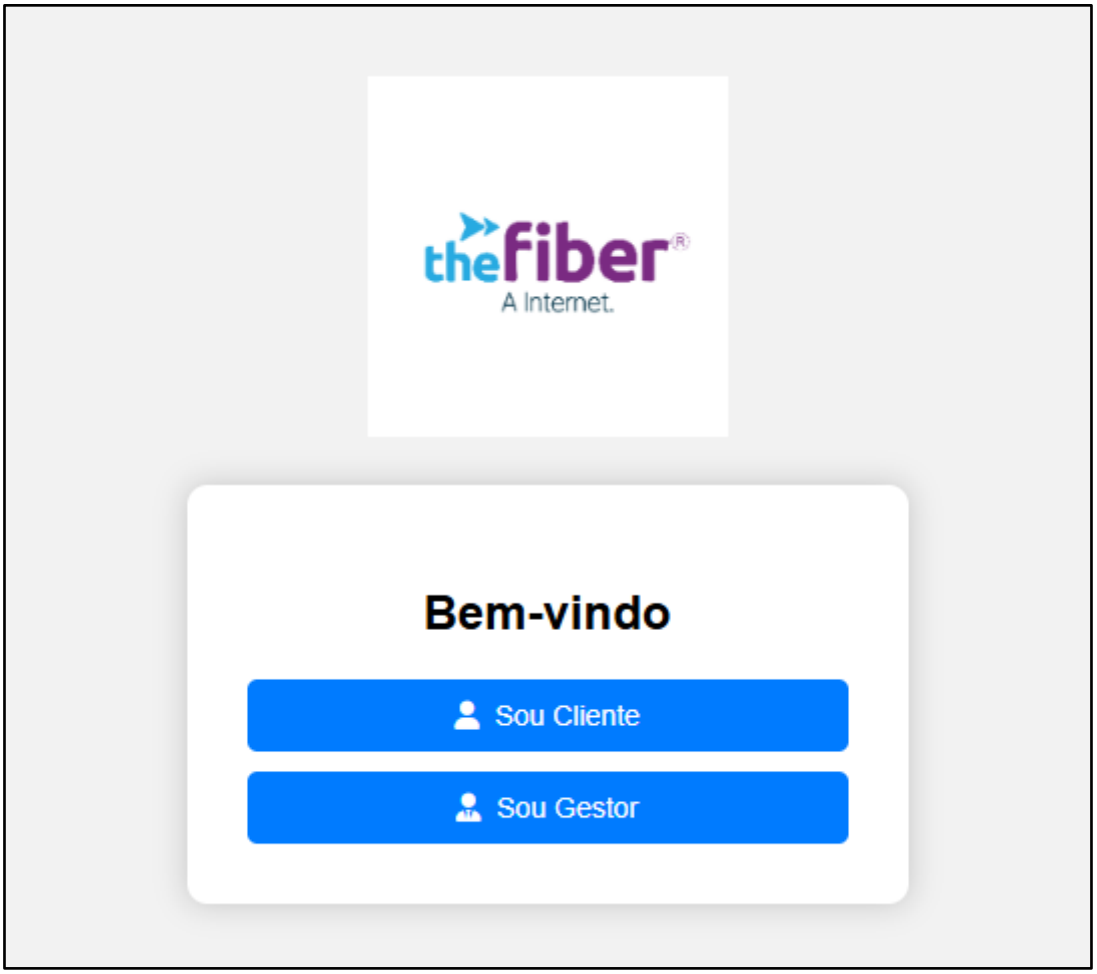
Apêndice A – TABELA DE REQUISITOS FUNCIOANAIS

ID	**Descrição**	Prioridade	Critério de Aceitação
ID01	Permitir que clientes enviem feedbacks com nome e data	Alta	Feedback deve ser armazenado corretamente no Firebase.
ID02	Permitir login de gestores com credenciais específicas	Alta	Apenas gestores autenticados podem acessar o dashboard
ID03	Exibir feedbacks com gráfico filtrável por funcionário	Alta	O gráfico deve mudar ao selecionar um funcionário
ID04	Permitir limpeza de feedbacks (somente por gestores)	Média	Botão limpa os dados armazenados no banco
ID05	Exibir nome e data do cliente no dashboard	Alta	Nome e data devem aparecer ao lado dos comentários

Apêndice B – TABELA DE REQUISITOS NÃO FUNCIOANAIS

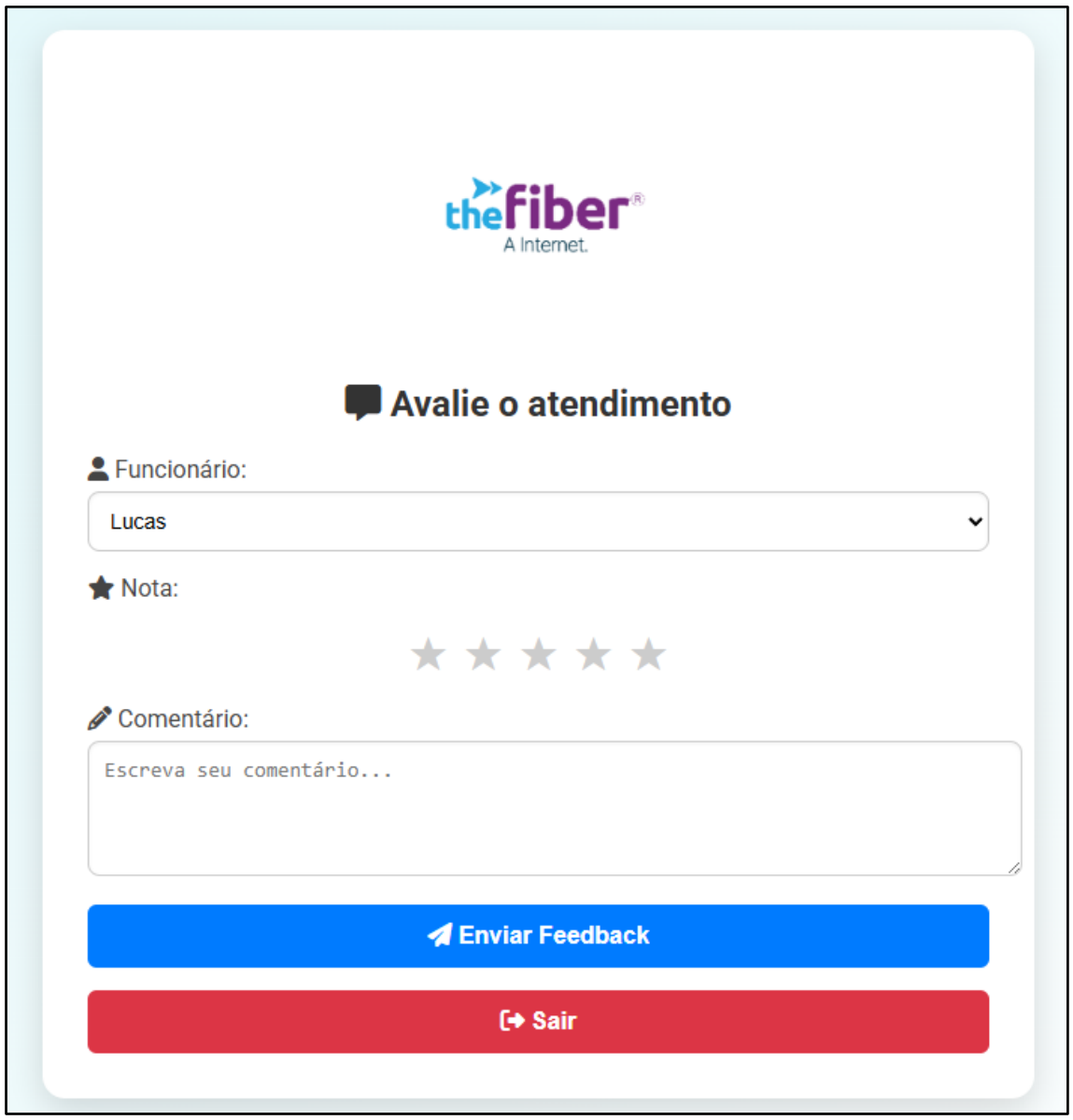
ID	Descrição	Critério de Aceitação
IDN01	O sistema deve processar até 50 feedbacks por dia sem degradação no desempenho.	Feedbacks enviados em lotes grandes devem ser processados em menos de 2 segundos.
IDN02	Deve garantir que dados sensíveis sejam armazenados com criptografia.	Dados no banco devem estar criptografados (verificados com ferramentas de auditoria).
IDN03	O sistema deve ter interface responsiva para ser acessado em dispositivos móveis.	Todos os componentes do site devem se ajustar para visualização em telas menores.
IDN04	Os dados devem ser armazenados com backup automático diário.	O banco deve registrar um backup a cada 24 horas, com logs de confirmação.

Apêndice C – TELA DE LOGIN

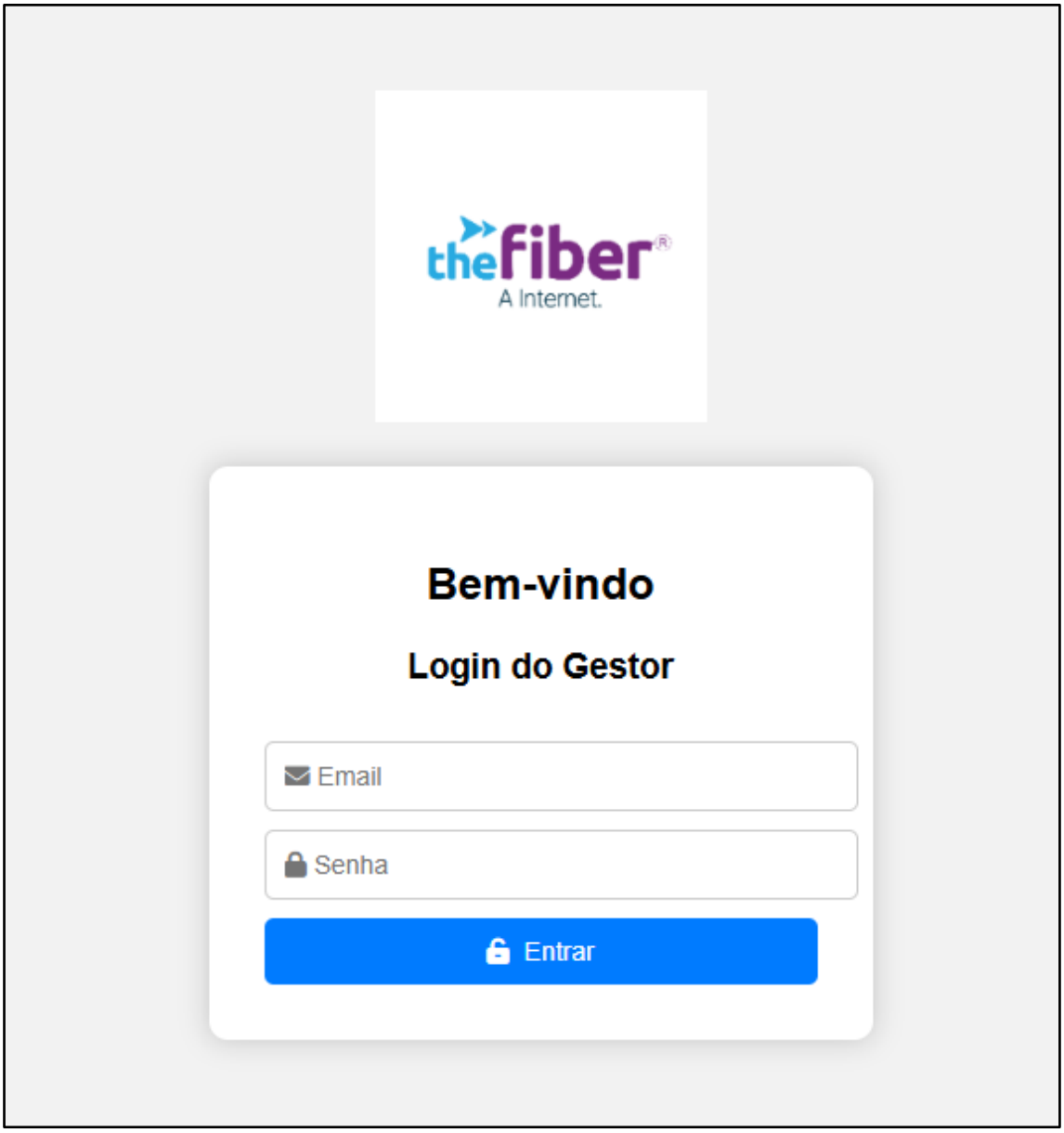


Apêndice D – TELA DE LOGIN DO CLIENTE

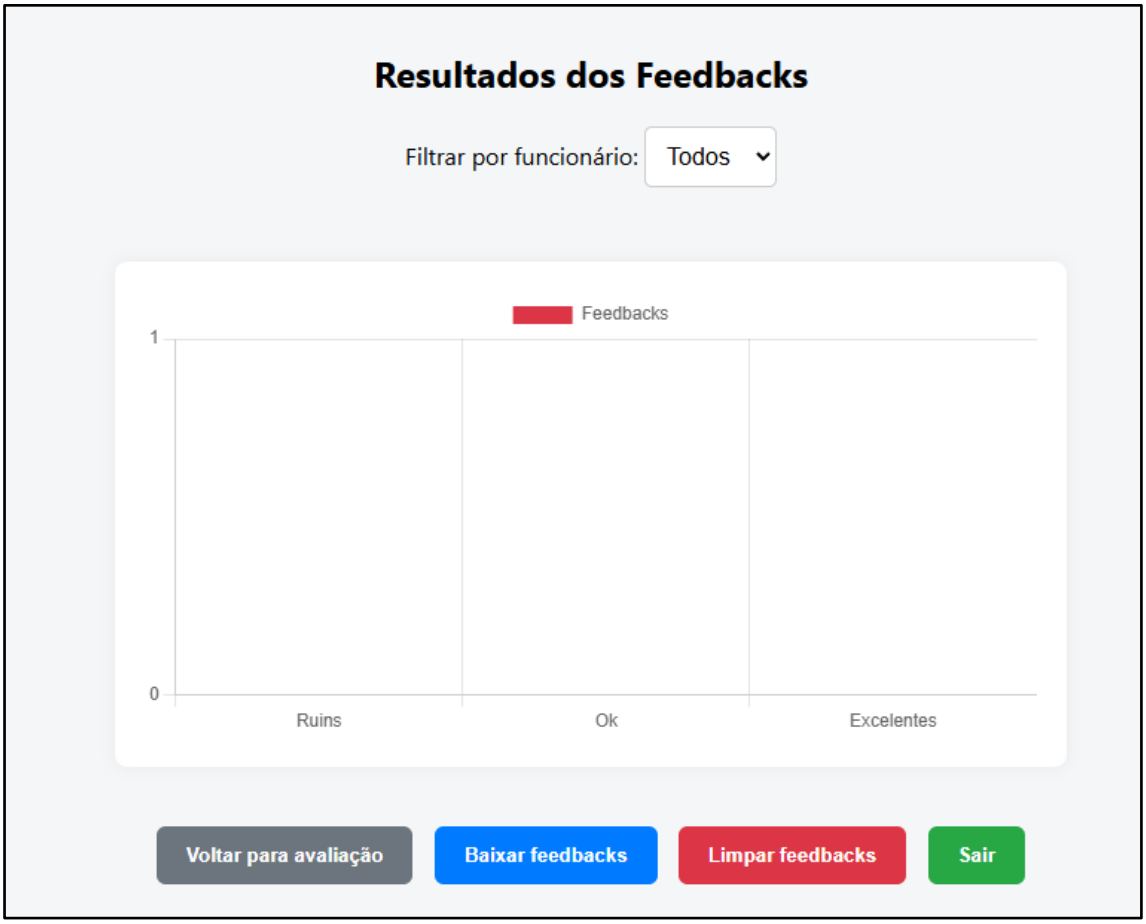
Apêndice E – TELA DE AVALIÇÃO PARA O CLIENTE



Apêndice F – TELA DE LOGIN DO GESTOR



Apêndice G – GRAFICO DOS FEEDBACKS DOS FUNCIONARIOS



PONTIFÍCIA UNIVERSIDADE CATÓLICA DE GOIÁS
GABINETE DO REITOR
Av. Universitária, 1069 ● Setor Universitário
Caixa Postal 86 ● CEP 74605-010
Goânia ● Goás ● Brasil
Fone: (62) 3946.1000
www.pucgoias.edu.br ● reitoria@pucgoias.edu.br

RESOLUÇÃO n° 038/2020 – CEPE

ANEXO I

APÊNDICE ao TCC

Termo de autorização de publicação de produção acadêmica

O(A) estudante Artur Souza Reis
do Curso de Engenharia da Computação matrícula 2018100330690,
telefone: 62982253397 e-mail arturreis1234567@gmail.com na qualidade de titular dos direitos autorais, em consonância com a Lei nº 9.610/98 (Lei dos Direitos do autor), autoriza a Pontifícia Universidade Católica de Goiás (PUC Goiás) a disponibilizar o Trabalho de Conclusão de Curso intitulado Aplicativo para gerenciar de Feedback de clientes para identificar problemas gratuitamente, sem ressarcimento dos direitos autorais, por 5 (cinco) anos, conforme permissões do documento, em meio eletrônico, na rede mundial de computadores, no formato especificado (Texto (PDF); Imagem (GIF ou JPEG); Som (WAVE, MPEG, AIFF, SND); Vídeo (MPEG, MWV, AVI, QT); outros, específicos da área; para fins de leitura e/ou impressão pela internet, a título de divulgação da produção científica gerada nos cursos de graduação da PUC Goiás.

Goiânia, 03 de abril de 2025.

Assinatura do(s) autor(es): Artur Souza Reis

Nome completo do autor: Artur Souza Reis

gov.br Documento assinado digitalmente
ANDRE LUIZ ALVES
Data: 03/04/2025 23:36:41-0300
Verifique em https://validar.iti.gov.br

Assinatura do professor-orientador:

Nome completo do professor-orientador: André Luiz Alves