

PONTIFÍCIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
NÚCLEO DE EDUCAÇÃO A DISTÂNCIA
Pós-graduação *Lato Sensu* em Arquitetura de Software Distribuído

Fábio Balbino Ribeiro

SISTEMA DE GESTÃO DA QUALIDADE

Belo Horizonte
2023

Fábio Balbino Ribeiro

SISTEMA DE GESTÃO DA QUALIDADE

Trabalho de Conclusão de Curso de Especialização
em Arquitetura de Software Distribuído como
requisito parcial à obtenção do título de especialista.

Orientador(a): Luiz Alberto

Belo Horizonte

2023

Para minha incansável e amada mãe

AGRADECIMENTOS

Agradeço aos brasileiros invisíveis, que com seus trabalhos sustentam tantas outras profissões, e infelizmente, quase nunca têm o seu devido reconhecimento financeiro ou social.

RESUMO

Este projeto diz respeito a uma solução de gestão da qualidade genérica, pensada para ser um *SaaS* a ser utilizado como um sistema de apoio por empresas de diversos setores, tais como construção civil, mineração e indústria em geral. As principais funcionalidades dizem respeito à gestão de não-conformidades e incidentes em geral.

Por ter como característica a responsividade, o sistema pode ser acessado tanto por dispositivos móveis quanto por computadores, preservando a experiência de uso. Sua arquitetura baseada em serverless permite o escalonamento horizontal e habilita sua utilização em diferentes cargas de trabalho.

A definição deste projeto leva em consideração a seguinte abordagem: estabelecimento dos requisitos funcionais e não funcionais, elaboração e projeto da arquitetura, realização de uma prova de conceito e avaliação arquitetural. Ao final, o objetivo é demonstrar a viabilidade da arquitetura proposta, destacando suas principais características e sua compatibilidade com o escopo originalmente definido.

Palavras-chave: arquitetura de software, projeto de software, requisitos arquiteturais, serverless, .Net, C#, Azure, Angular, RxJS, NgRx, JWT, API, JSON, REST, Azure Functions, Azure App Insights, Azure Notification Hub, Azure Service Bus.

SUMÁRIO

1. Objetivos do trabalho.....	7
2. Descrição geral da solução.....	8
2.1. Apresentação do problema.....	8
2.2. Descrição geral do software (Escopo).....	8
3. Definição conceitual da solução.....	9
3.1. Requisitos Funcionais.....	9
3.2 Requisitos Não-Funcionais.....	15
3.3. Restrições Arquiteturais.....	19
3.4. Mecanismos Arquiteturais.....	20
4. Modelagem e projeto arquitetural.....	21
4.1. Modelo de componentes.....	21
4.2. Modelo de implantação.....	24
5. Prova de Conceito (POC) / protótipo arquitetural.....	25
5.1. Implementação e Implantação.....	25
6. Avaliação da Arquitetura.....	32
6.1. Análise das abordagens arquiteturais.....	32
6.2. Cenários.....	33
6.3. Avaliação.....	35
6.4. Resultado.....	49
7. Conclusão.....	50
APÊNDICES.....	51
CHECKLIST PARA VALIDAÇÃO DOS ITENS E ARTEFATOS DO TRABALHO...52	

1. Objetivos do trabalho

O objetivo deste trabalho é realizar um levantamento arquitetural abrangente para o sistema de gestão da qualidade (SGQ), com o propósito de definir uma estrutura sólida que sustente as funcionalidades necessárias para o gerenciamento eficaz da qualidade da empresa contratante. A solução aqui detalhada possibilita que colaboradores e gestores da qualidade, mesmo estando geograficamente dispersos, compartilhem a mesma plataforma de trabalho cotidiano. Com isso, é possível a identificação e tratamento de não-conformidades e planejamento das auditorias.

Os objetivos específicos do projeto são:

1. Construir um módulo de controle de usuários, onde o acesso seja controlado mediante diferentes perfis de uso e a autenticação seja feita utilizando JWT;
2. Construir um módulo de acessibilidade, onde seja possível ao usuário alterar o layout do app para um modo de alto contraste, possibilitando que usuários com dificuldades visuais possam utilizar o sistema com mais facilidade;
3. Construir um módulo de visualização de checklists, onde, em caso de alteração da informação, seja disparada um evento que é lido por um serviço responsável por notificações ao usuário;
4. Construir as funções serverless que darão suporte aos objetivos descritos acima, utilizando JWT, banco de dados NoSql e fila de eventos.
5. Construir o sistema de maneira que o layout seja adaptável a diversos tamanhos de dispositivo, tais como tablets, computadores e celulares.

2. Descrição geral da solução

2.1. Apresentação do problema

A ausência de um sistema de gestão da qualidade informatizado pode acarretar em uma série de desafios para as empresas nos dias de hoje. Primeiramente, a falta de automação pode levar a processos manuais demorados e propensos a erros, o que resulta em desperdício de tempo e recursos valiosos. A rastreabilidade de dados também se torna uma tarefa árdua, dificultando a identificação rápida de problemas e a tomada de decisões informadas.

Além disso, a falta de um sistema informatizado de gestão da qualidade pode comprometer a conformidade com normas e regulamentações específicas da indústria. A documentação manual pode ser desorganizada e sujeita a perdas, tornando difícil a comprovação do cumprimento de padrões de qualidade. Isso, por sua vez, coloca a empresa em risco de penalidades legais e perda de credibilidade no mercado. Em resumo, a adoção de um sistema de gestão da qualidade informatizado é essencial para otimizar processos, melhorar a conformidade e garantir um desempenho consistente e competitivo no mundo dos negócios atual.

2.2. Descrição geral do software (Escopo)

Nosso Software como Serviço (SaaS) de Gestão da Qualidade é uma solução inovadora que oferece simplicidade e eficiência. Ele é construído com uma arquitetura serverless, reduzindo preocupações com infraestrutura e proporcionando escalabilidade sob demanda. Além disso, é responsivo, adaptando-se a dispositivos móveis, tablets e desktops, permitindo que a equipe acesse e gerencie dados críticos de qualidade em qualquer lugar.

Também é possível a disponibilização de dados à consultorias externas mediante acesso controlado via API (também serverless), de forma que essas consultorias externas possam se munir das informações necessárias às suas análises.

Além dos microsserviços em questão, o front-end acessa diretamente um serviço externo responsável pelas normas que serão utilizadas pelo sistema.

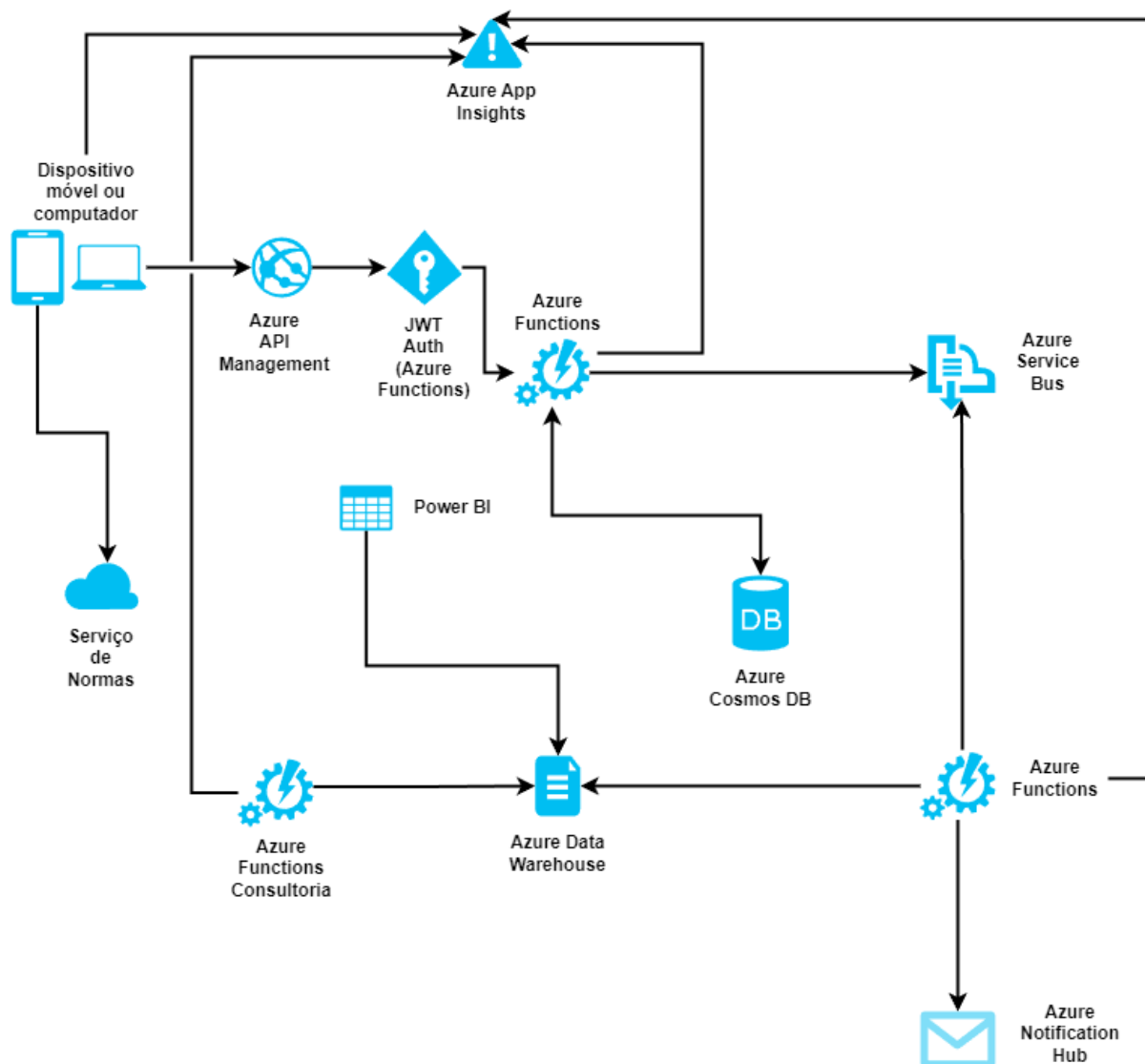


Figura 1: Diagrama informal da arquitetura

3. Definição conceitual da solução

3.1. Requisitos Funcionais

Módulo de incidentes e problemas (ocorrências)

- **Controle de não-conformidades (NC)**

A aplicação deve oferecer a funcionalidade de possibilitar que qualquer usuário logado registre não conformidades. Além disso, é necessário que o gestor da qualidade tenha a capacidade de incluir dados relacionados à causa da não conformidade, bem como os prazos para sua conclusão e uma análise final sobre a eficácia do plano de ação. Após a investigação da causa, a implementação do plano de ação e a inclusão da análise final, a não conformidade deve ser finalizada pelo gestor da qualidade.

- **Controle de ações do SGQ**

A aplicação deve capacitar o gestor da qualidade a registrar diferentes tipos de ações, que podem ser correções, ações corretivas ou ações preventivas. É importante que o gestor possa especificar o usuário encarregado de executar a ação, a data estimada para sua conclusão e o resultado após a implementação.

Módulo de controle de processos

- **Controle de auditorias**

A aplicação deve viabilizar o registro de auditorias pelo gestor de qualidade, com a capacidade de incluir informações como a data de início da auditoria, o auditor encarregado, o tipo (interno/externo) e os elementos auditados.

- **Controle de resultados de checklists**

A aplicação deve oferecer a funcionalidade de registrar os resultados obtidos ao verificar os checklists de qualidade. Deve ser possível indicar a data da verificação e os resultados específicos de cada item presente no checklist em análise.

Módulo de compliance

- **Consulta de normas**

A aplicação deve habilitar qualquer usuário logado a acessar um catálogo de normas externo, provido por uma empresa especializada. É necessário que os usuários possam consultar os detalhes de cada norma e, se estiver disponível, realizar o download do arquivo da norma.

- **Solicitações de análise**

A aplicação deve possibilitar que o gestor da qualidade envie solicitações para análise de não conformidades e planos de ação às empresas de consultoria. Essa integração tem como objetivo fornecer ao gestor da qualidade informações sobre a eficácia do plano de ação desenvolvido.

Módulo de configuração

- **Controle de usuários**

A aplicação deve possibilitar que o gestor da qualidade gerencie os usuários do sistema, incluindo a administração dos perfis de acesso de cada usuário.

- **Controle de empresas**

A aplicação deve permitir que o gestor da qualidade administre as empresas autorizadas para operar, juntamente com seus respectivos setores, processos e produtos.

- **Controle de planos de auditoria**

A aplicação deve possibilitar que o gestor da qualidade gerencie os planos anuais de auditoria, incluindo a capacidade de registrar itens de auditoria (os requisitos a serem verificados) e distribuí-los ao longo do tempo para a composição de cada plano de auditoria.

- **Controle de checklists**

A aplicação deve permitir a personalização de checklists de acordo com a necessidade da empresa contratante do SaaS, tendo suas métricas personalizadas. Também deve permitir a associação dos checklists às auditorias cadastradas.

- **Controle de empresas de consultorias**

A aplicação deve permitir que o gestor da qualidade gerencie as empresas de consultoria habilitadas para integração. Deve ser possível informar o nome, a URL e o token de acesso para a integração.

Módulo de conta

- **Meus dados**

A aplicação deve permitir que o usuário edite seus dados próprios: nome, sobrenome, e-mail e idioma.

- **Alterar senha**

A aplicação deve permitir que o usuário altere sua senha de acesso.

- **Excluir conta**

Por conta das modificações de legislação pós LGPD, deve ser possível ao usuário realizar a exclusão da sua própria conta, assim como seus dados. Caso o usuário seja o administrador, ele poderá também excluir todas as informações da empresa.

Módulo política de privacidade e termos de uso

- **Exportar dados armazenados do usuário**

O usuário pode exportar quais dados seus estão armazenados nos servidores do sistema.

- **Exportar dados armazenados da empresa (Somente administradores)**

O usuário pode exportar quais dados da empresa estão armazenados nos servidores do sistema.

- **Consultar política de privacidade**

O usuário pode consultar quando quiser a política de privacidade do sistema.

- **Consultar termos de uso do sistema**

O usuário pode consultar quando quiser os termos de uso do sistema.

Módulo acessibilidade

- **Modo alto contraste**

O usuário pode aplicar um modo de alto contraste no sistema para pessoas com dificuldades oculares.

Módulo de relatórios

- Relatório de análises de consultoria
- Relatório de auditoria
- Relatório de controle de processo (checklist)
- Relatório de não conformidades

Módulo de Cadastro

- **Cadastrar uma empresa**

Esse é o módulo em que se provê o acesso ao sistema. O usuário que realizar o auto-cadastro da empresa terá o acesso administrador e conseguirá inserir mais informações sobre a empresa.

- **Excluir uma empresa**

Semelhante à funcionalidade “Excluir conta”, porém diretamente no contexto da empresa. Permite que todo o cadastro de uma empresa seja excluído.

- **Solicitar uma demonstração**

Será criado um perfil do tipo “caixa de areia” onde o usuário poderá simular a utilização do sistema dentro da sua empresa pelo período de sete dias.

A tabela a seguir apresenta as prioridades de cada requisito funcional, tendo por critério a importância da funcionalidade para o sistema. (Essencial / Importante / Desejável)

Módulo	RF	Prioridade
Módulo de incidentes e problemas (ocorrências)	Controle de não-conformidades (NC)	Essencial
Módulo de incidentes e problemas (ocorrências)	Controle de ações do SGQ	Essencial
Módulo de controle de processos	Controle de auditorias	Essencial
Módulo de controle de processos	Controle de resultados de checklists	Essencial
Módulo de compliance	Consulta de normas	Importante
Módulo de compliance	Solicitações de análise	Importante
Módulo de configuração	Controle de usuários	Essencial
Módulo de configuração	Controle de empresas	Importante
Módulo de configuração	Controle de planos de auditoria	Importante
Módulo de configuração	Controle de checklists	Essencial
Módulo de configuração	Controle de empresas de consultorias	Importante
Módulo de conta	Meus dados	Essencial
Módulo de conta	Alterar senha	Essencial
Módulo de conta	Excluir conta	Essencial

Módulo política de privacidade e termos de uso	Exportar dados armazenados do usuário	Essencial
Módulo política de privacidade e termos de uso	Exportar dados armazenados da empresa (Somente administradores)	Essencial
Módulo política de privacidade e termos de uso	Consultar política de privacidade	Essencial
Módulo política de privacidade e termos de uso	Consultar termos de uso do sistema	Essencial
Módulo acessibilidade	Modo alto contraste	Desejável
Módulo de relatórios	Relatório de análises de consultoria	Importante
Módulo de relatórios	Relatório de auditoria	Importante
Módulo de relatórios	Relatório de controle de processo (checklist)	Importante
Módulo de relatórios	Relatório de não conformidades	Importante
Módulo de Cadastro	Cadastrar uma empresa	Essencial
Módulo de Cadastro	Excluir uma empresa	Essencial
Módulo de Cadastro	Solicitar uma demonstração	Desejável

3.2 Requisitos Não-Funcionais

- **Usabilidade – A aplicação deve ser de fácil utilização**

Estímulo	Registro de não conformidade.
Fonte do estímulo	Usuário acessando a funcionalidade de registro de não conformidade.
Ambiente	Funcionamento com carga normal.
Artefato	Módulo de incidentes e problemas.
Resposta	O frontend permite uma utilização flúida e intuitiva.
Medida da resposta	Usuário foi capaz de realizar o registro de uma não conformidade e seu plano de ação em até 3 minutos.

- **Confiabilidade – A aplicação deve ser resiliente a falhas**

Estímulo	Acesso a um serviço externo offline
Fonte do estímulo	Usuário acessando um serviço externo que está inacessível
Ambiente	Acesso a algum serviço externo comprometido
Artefato	Módulo de compliance
Resposta	O frontend deve ser resiliente à falha do serviço e retornar ao funcionamento normal de forma transparente tão logo o acesso seja

	reestabelecido.
Medida da resposta	O frontend exibe uma mensagem de erro para cada chamada à API que não pode ser efetuada, sem contudo impedir a navegação do usuário para outras telas.

- **Disponibilidade – A aplicação deve permitir uma taxa de disponibilidade de no mínimo 98%**

Estímulo	Manutenção em um dos códigos das Azure Functions do sistema enquanto usuários acessam ao sistema
Fonte do estímulo	Manutenção corretiva
Ambiente	Funcionamento com carga normal, com usuários realizando operações de custo operacional normal
Artefato	Aplicação
Resposta	A aplicação deve continuar disponível para os usuários
Medida da resposta	A aplicação continua respondendo normalmente, sem impacto nos tempos de resposta

- **Portabilidade – A aplicação deve ser acessível seja por computadores, seja por dispositivos móveis**

Estímulo	Consulta de checklists.
Fonte do estímulo	Usuário acessando o sistema a partir de um smartphone.
Ambiente	Funcionamento com carga normal, com usuários realizando operações de custo operacional normal
Artefato	Módulo de checklists
Resposta	O Front-end da aplicação deve se ajustar ao layout do dispositivo
Medida da resposta	O front-end foi responsivo ao layout do dispositivo

- **Segurança – A aplicação deve proteger dados e funcionalidades com um mecanismo de segurança confiável**

Estímulo	Acesso a uma página privada sem login prévio
Fonte do estímulo	Usuário acessando uma funcionalidade protegida sem a especificação do token de segurança.
Ambiente	Funcionamento com carga normal.
Artefato	Todos os módulos da aplicação, exceto página de cadastro
Resposta	O acesso deve ser bloqueado.
Medida da resposta	O frontend impede o acesso à funcionalidade

	e exibe o formulário de login
--	-------------------------------

- **Acessibilidade – A aplicação deve ser configurável para mostrar cores alternativas/contrastantes para pessoas com deficiências na visualização de cores**

Estímulo	Alteração da configuração de acessibilidade de cores para cores alternativas/contrastantes
Fonte do estímulo	Usuário com deficiência na visualização de cores
Ambiente	Funcionamento com carga normal.
Artefato	Todos os módulos da aplicação, exceto página de cadastro
Resposta	O novo esquema de cores deve ser disponibilizado
Medida da resposta	Novo esquema de cores aplicado, possibilitando a visualização por pessoas com deficiências na visualização de cores

3.3. Restrições Arquiteturais

- O front-end deve ser desenvolvido utilizando Angular, com observabilidade via RxJS e NgRX como gerenciador de estado
- O back-end deve ser serverless, desenvolvido com a suíte .Net e a linguagem C#
- O sistema deve utilizar Azure como sua suíte de cloud
- O front-end deve ser responsivo
- Deve ser utilizado o padrão REST com dados serializados em JSON para as API's
- A autenticação da aplicação deve utilizar JWT
- A segurança do acesso à paginas deve ser feita mediante a utilização de guardas de rota
- O banco de dados principal da aplicação deve ser em Azure Cosmos DB utilizando tecnologia NoSQL
- O monitoramento da aplicação deverá ser realizado via Azure App Insights
- O armazenamento em massa de dados para ser disponibilizado para as ferramentas de BI e relatórios deve utilizar o Azure Data Warehouse
- O envio de notificações ao usuário deve ser realizado via Azure Notification Hub

3.4. Mecanismos Arquiteturais

Análise	Design	Implementação
Persistência	ORM	EntityFramework
Persistência	ORM	Sqlite
Persistência	Banco de Dados NoSQL	Azure CosmosDB
Front end	Web	Angular
Front end	Mobile	Angular
Back end	Serverless	Azure Functions
Monitoramento	Telemetria	Azure App Insights
Teste de Software	Testes unitários	xUnit
Notificações	Notificação por Push	Azure Hub Notification
Eventos	Service Bus	Azure Service Bus
Autenticação e autorização	Azure Functions com JWT	Azure Functions com JWT
Documentação	Documentação de API's	Swagger
Integração com sistemas externos	Interfaces padronizadas (OpenAPI)	API's REST JSON
Deploy	Orquestração da Stack	Azure Functions
Versionamento	Gestão de alterações	Git

4. Modelagem e projeto arquitetural

4.1. Modelo de componentes

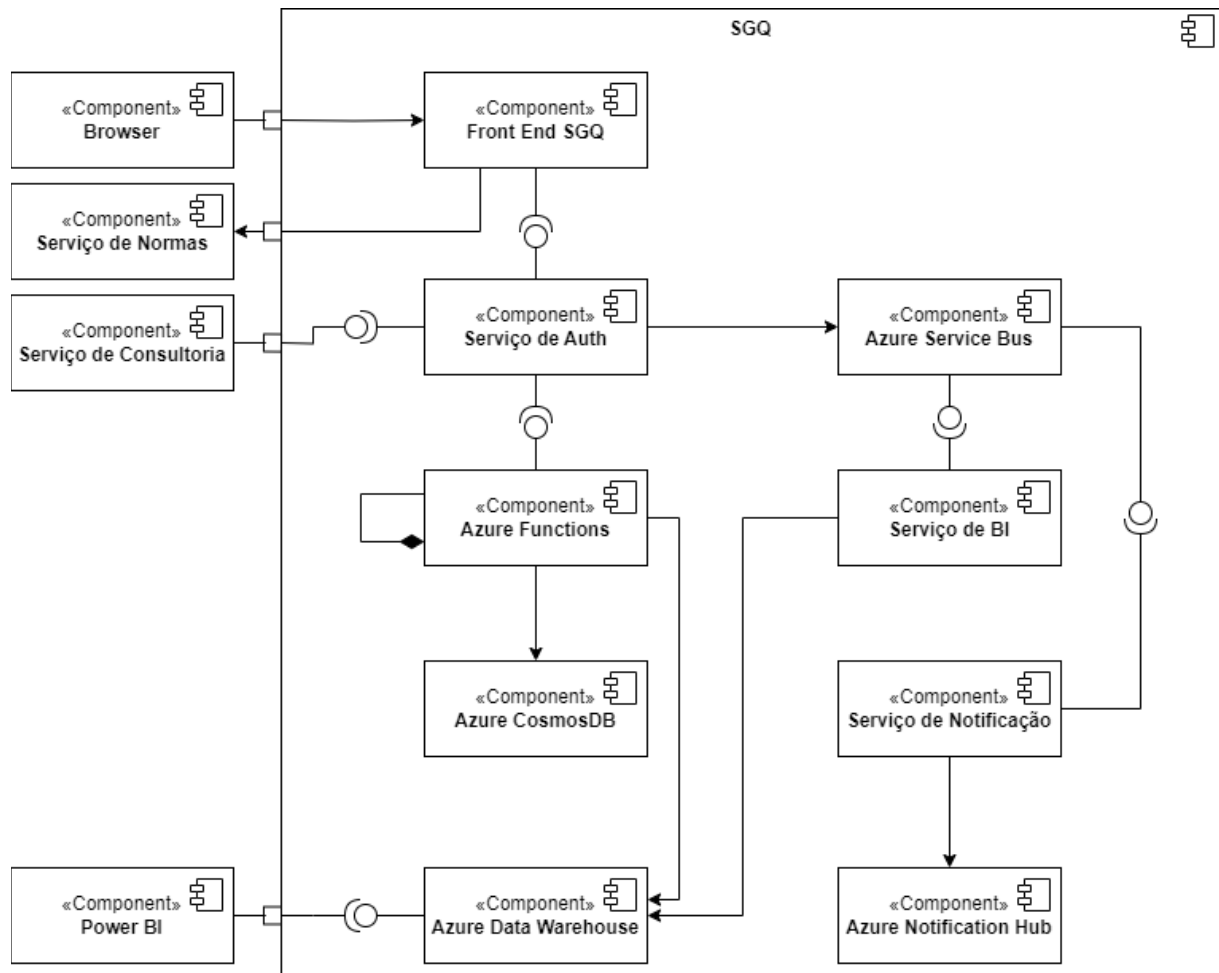


Figura 2: Diagrama de componentes.

Browser: Meio de acesso ao sistema. Pode ser utilizado tanto por dispositivos móveis quanto por computadores.

Serviço de Normas: Serviço que o Front End acessa para obter as informações relativas às normas disponíveis. Precisa ser desenvolvido.

Serviço de Consultoria: Serviço externo que acessa os dados disponibilizados pelo sistema. Esses dados são os dados tratados pela aplicação e que possuem relevância para os serviços de auditoria externa, consultoria externa, etc. Precisa ser desenvolvido.

Front end SGQ: O sistema utiliza um front-end Angular para criar uma interface de usuário moderna e responsiva. Este front-end realiza chamadas JSON para se comunicar com o as

funções serverless por meio de JSON. O front também pode executar chamadas ao servidor de normas.

Power BI: Aplicação Microsoft responsável por acessar os dados disponibilizados na Azure Data Warehouse, onde podem ser criados relatórios personalizados.

Serviço de Auth: Serviço responsável pela autenticação e autorização, mediante a utilização de JWT. Para o Front End acessar as demais áreas do sistema, é necessário estar autenticado e ter autorização para utilização da respectiva área. Esse controle é feito via perfil de acesso, também configurado no JWT. Feito utilizando-se Azure Functions.

Azure Functions: O Azure Functions é um serviço de computação serverless oferecido pela Microsoft Azure, que permite executar código de forma escalável e sob demanda, sem a necessidade de gerenciar a infraestrutura subjacente. As Azure Functions são utilizadas por muitas áreas dentro do sistema, tais como notificar equipes sobre não conformidades, agendar e realizar ações corretivas ou preventivas, processar dados de auditorias e acionar alertas com base em eventos críticos.

Azure CosmosDB: Banco de dados NoSQL oferecido pela Microsoft Azure que será utilizado para registrar as informações do sistema. Por ser um sistema com uma necessidade alta de personalização, o banco de dados NoSQL acaba por ter uma menor fricção para se chegar a esse cenário customizável, mantendo alta performance de escrita e leitura.

Azure Data Warehouse: Serviço Microsoft Azure para armazenamento e consulta de grandes volumes de dados para os sistemas de BI e consultorias externas.

Azure Service Bus: Serviço de mensagens Microsoft Azure responsável pela integração entre as diversas partes do serviço distribuído.

Serviço de BI: Serviço responsável por uma estratificação intermediária e envio ao Azure Data Warehouse, onde é disponibilizada para a construção de relatórios, acesso via Power BI, etc. Feito utilizando-se Azure Functions.

Serviço de Notificação: Serviço responsável pelo tratamento de notificações e preparação para envio ao usuário quando há informações relevantes, tais como alterações em não-conformidades, análises concluídas, etc. O usuário precisa aceitar o envio de notificações para que seja possível o recebimento. Feito utilizando-se Azure Functions.

Azure Notification Hub: Serviço Microsoft Azure que recebe o tratamento das informações do serviço de notificação e envia web push notifications aos usuários que subrescreveram ao serviço.

4.2. Modelo de implantação

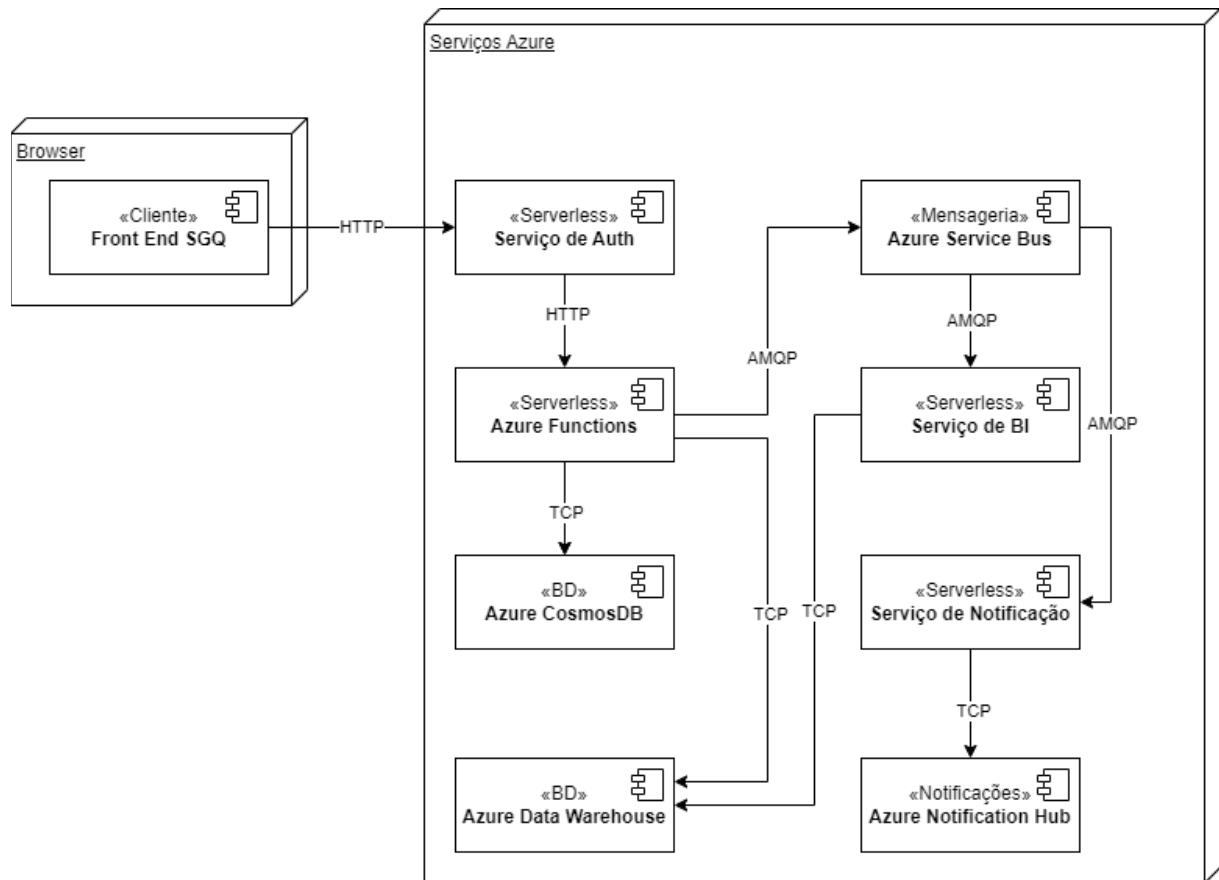


Figura 3: Diagrama de implantação.

Serviços Azure: Suíte de serviços cloud que facilitam a escalabilidade e disponibilidade do sistema. As Azure Functions são altamente escaláveis e podem responder dinamicamente às demandas de tráfego, garantindo eficiência e economia de recursos. Eles também são orientados a eventos, onde funções ou serviços são acionados por eventos específicos, como nos serviços de BI e de notificação. A natureza pay-as-you-go dos serviços serverless significa que os custos estão diretamente relacionados ao uso, tornando-os uma escolha econômica para o nosso sistema. O Azure Service Bus garante que algumas ações que poderiam virar gargalos dentro do sistema não o sejam, tornando o sistema assíncrono à certas respostas que não dependem de entrega instantânea, reduzindo pontos críticos que ameacem a disponibilidade do sistema. Utilizamos dois bancos que tem entrega rápida da informação e permitem uma alta taxa de customização das informações (Azure CosmosDB e Data Warehouse). Todas os serviços elencados acima permitem escalonamento horizontal sob demanda e redundância geográfica.

5. Prova de Conceito (POC) / protótipo arquitetural

5.1. Implementação e Implantação

O protótipo construído para validar a arquitetura proposta se baseia em um front end Angular que se comunica com serviços serverless através de chamadas Rest. Essas Azure Functions Serverless são responsáveis por diversas ações, tais como salvar informações em um banco de dados NoSQL e enviar e consumir mensagens para possibilitar o armazenamento de informações metrificadas para BI e para o envio de notificações.

- **Front end:** Angular
- **Serviços:** Azure Functions
- **Banco de dados:** Azure CosmosDB
- **Banco de dados BI:** Azure Data Warehouse
- **Monitoramento:** Azure App Insights
- **Mecanismo de autenticação/Autorização:** JWT
- **Serviço de mensageria:** Azure Service Bus
- **Serviço de notificações:** Azure Notifications Hub

O protótipo da aplicação foi desenvolvido para validar a arquitetura proposta em relação aos requisitos de negócios do Sistema de Gestão da Qualidade (SGQ). Isso foi realizado por meio da análise de requisitos não funcionais durante a execução de casos de uso específicos, que incluíram funcionalidades como registro de controle de usuários, controle de checklists e modo de alto contraste. Esse processo permite verificar a adequação da arquitetura e dos requisitos não-funcionais selecionados. Seguem abaixo os UC's que serão implementados no sistema:



Figura 6: Módulo de usuários.

- **Módulo de Usuários**

Esse requisito permite ao usuário consultar, alterar ou excluir um usuário. Após a realização da alteração de um perfil para um perfil mais restrito, por exemplo, o usuário deverá perder acessos às funcionalidades fora do seu escopo de uso.

O caso de uso abaixo demonstra as ações que serão implementadas no sistema no módulo de usuários:

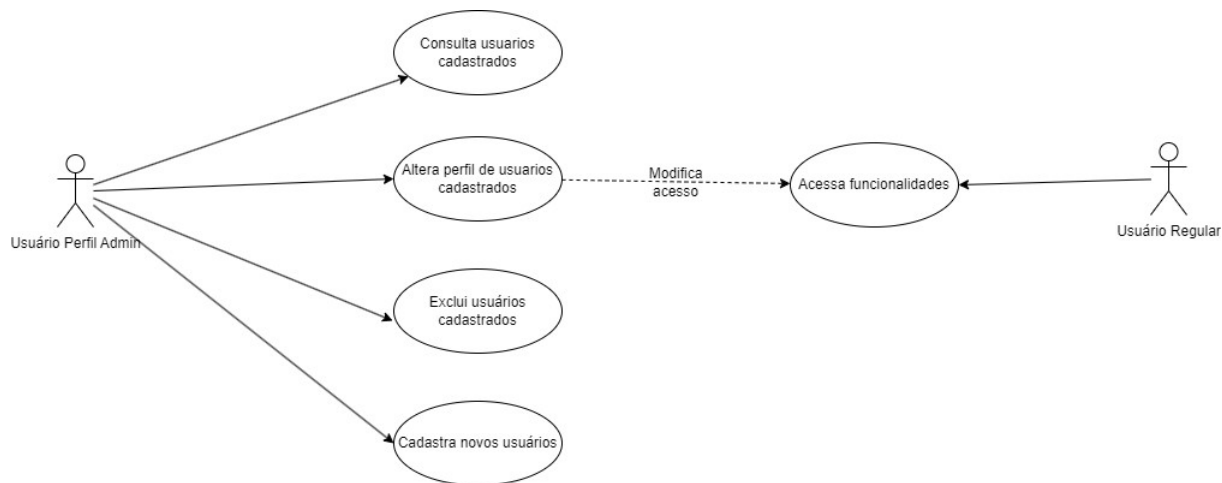


Figura 7: Diagrama de caso de uso do módulo de usuários.

• Módulo de Checklists

Esse módulo permite ao usuário gestor ou admin criar seus próprios checklists personalizados. Será implementada a *visualização* de um checklist como exemplo.

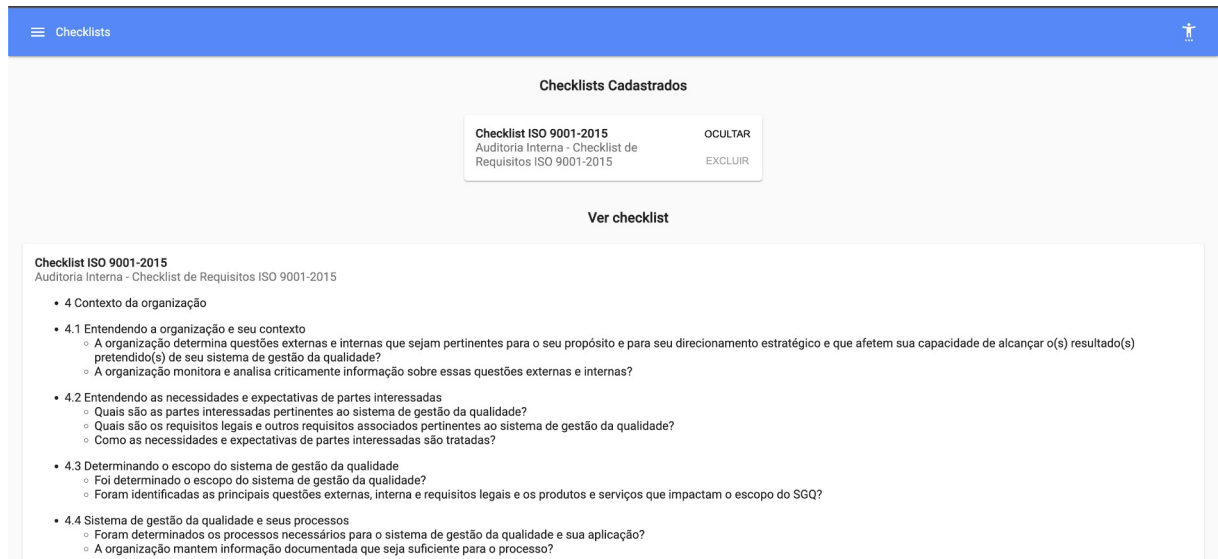


Figura 8: Módulo de checklist

O caso de uso abaixo demonstra o módulo de checklists:

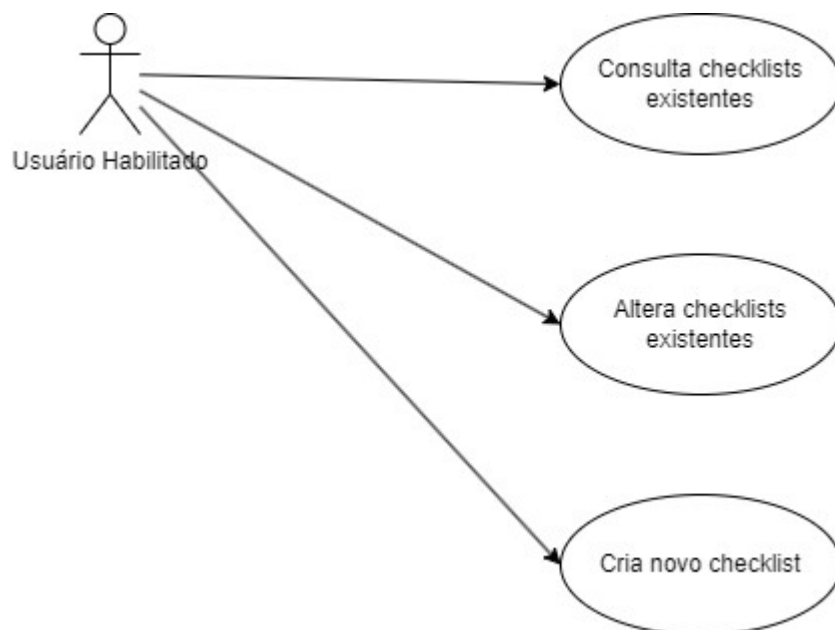


Figura 9: Diagrama de caso de uso do módulo de checklists

- **Modo de alto contraste**

Esse módulo permite ao usuário alterar seu esquema de cores para um esquema que seja visível por pessoas com deficiências visuais.

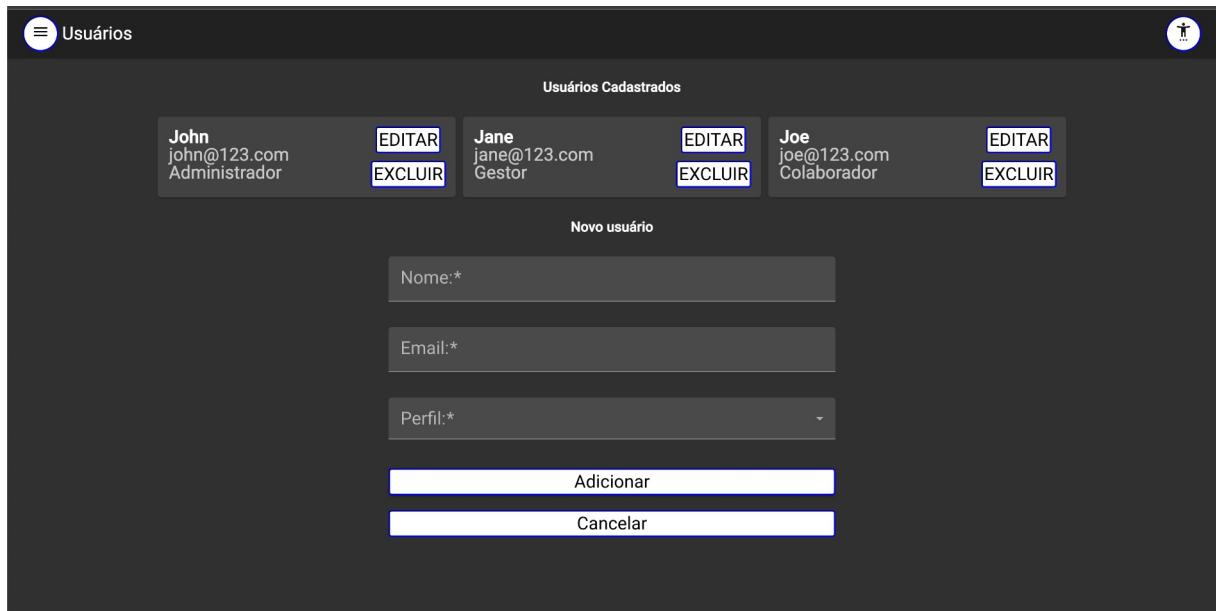


Figura 10: Módulo de acessibilidade

O caso de uso abaixo demonstra as ações que serão implementadas no sistema no modo de alto contraste:

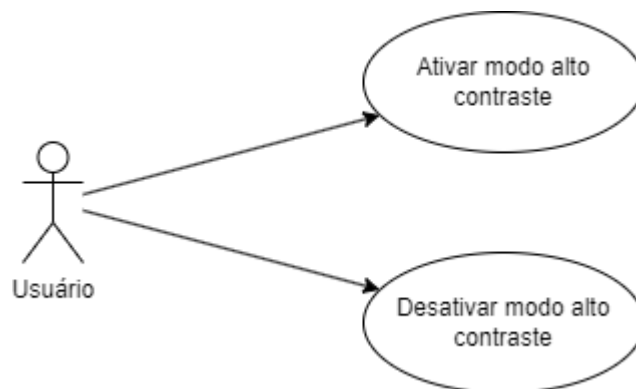


Figura 11: Diagrama de casos de uso do módulo de acessibilidade

Os requisitos *não funcionais* analisados durante a execução de tais requisitos funcionais são os seguintes:

- **Segurança – Bloqueio de acessos que não são autenticados/autorizados**

Esse requisito foi escolhido para garantir que os usuários que não possuam acesso a determinadas áreas do sistema não possam acessá-las, e garantir que usuários que não tenham acesso ao sistema não consigam utilizá-lo.

- Garantir que usuários não autenticados não possuam acesso ao sistema
- Garantir que usuários não autorizados não possuam acesso às áreas que não são de seu escopo
- Garantir que, após a exclusão ou alteração do tipo de perfil de acesso ao sistema o usuário perca o acesso às áreas que não foram autorizadas.

- **Portabilidade – Visual adaptado automaticamente para dispositivos móveis**

Esse requisito foi escolhido para garantir que os usuários possuam uma boa experiência de uso mesmo em dispositivos móveis. Isso é especialmente importante nas funcionalidades acessadas possivelmente a partir de smartphones. Os critérios de aceite para a análise deste requisito são:

- A aplicação deve ter interfaces que se ajustem automaticamente às várias resoluções dos dispositivos de acesso, reorganizando e redimensionando seus elementos de maneira otimizada.
- As interfaces devem ser amigáveis e fáceis de usar, mesmo quando adaptadas para resoluções menores, mantendo a consistência visual.
- A identidade visual da aplicação deve ser mantida em todas as resoluções para proporcionar uma experiência de usuário uniforme.
- A aplicação deve ser acessível através dos principais navegadores web modernos, como Chrome, Firefox, Edge e Safari, além de oferecer suporte a navegadores em dispositivos móveis.

- **Acessibilidade – Permitir o uso do sistema por pessoas com dificuldades visuais**

Esse requisito foi escolhido para garantir que pessoas que tem dificuldade com o contraste das cores, tal como o daltonismo e seus diversos subtipos, pessoas com fotosensibilidade ou baixa visão consigam utilizar o sistema.

- A aplicação deve possuir um modo de alto contraste facilmente identificável e ativável
- O usuário deve perceber a alteração do esquema de cores para um com alto contraste em todas as áreas do sistema, incluindo imagens

5.2 Código

Repositório do projeto: <https://github.com/fbalbinoribeiro/sgq>

Foi utilizada a stack principal do Angular com RxJS para a observabilidade e NgRX Store para o controle de estado do front com NgRX Effects para controle dos efeitos colaterais dos estados.

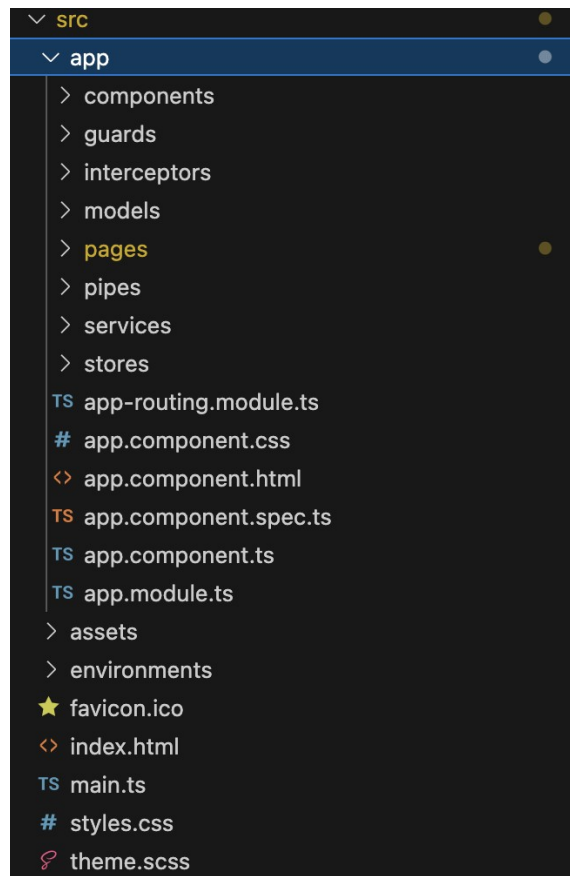


Figura 12: Estrutura de código do front-end

Para o back end, foi utilizado Azure Functions com conexões ao CosmosDB e Azure Service Bus.

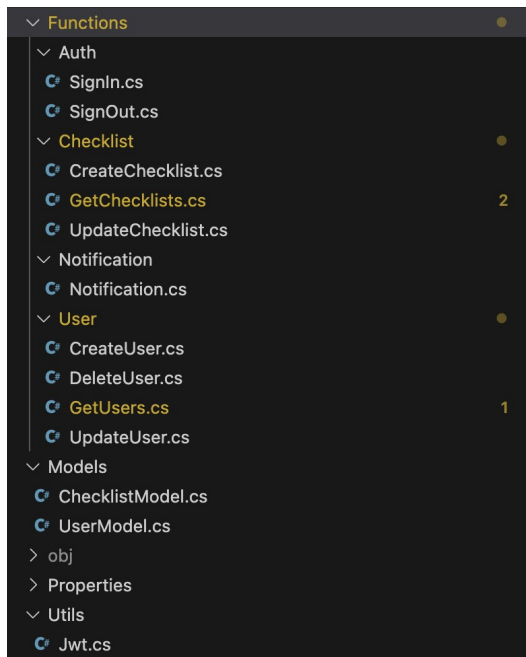


Figura 13: Estrutura de código do back-end

6. Avaliação da Arquitetura

6.1. Análise das abordagens arquiteturais

A utilização da arquitetura serverless é interessante por sua capacidade de simplificar o desenvolvimento e operação de aplicações, permitindo que os desenvolvedores se concentrem apenas no código de suas funções e serviços, enquanto a infraestrutura subjacente é gerenciada automaticamente pelo provedor de nuvem. Isso resulta em maior agilidade no desenvolvimento, escalabilidade elástica para lidar com variações na demanda, e economia de custos, pois os recursos são alocados sob demanda, sem a necessidade de manter servidores constantemente ativos. Além disso, a arquitetura serverless promove uma abordagem mais modular e orientada a serviços, facilitando a construção de sistemas mais robustos e de fácil manutenção.

Embora o modelo serverless ofereça diversas vantagens, como escalabilidade automática e redução de complexidade operacional, também apresenta algumas desvantagens a considerar. Uma delas é a potencial falta de controle direto sobre a infraestrutura, o que pode limitar a personalização e configurabilidade para certos casos de uso complexos. Além disso, os custos podem ser pouco previsíveis, uma vez que as cobranças são baseadas no consumo de recursos, podendo se tornar mais caros em cenários de alta demanda. O tempo de

inicialização das funções serverless, em alguns casos, pode causar latências indesejadas em aplicativos que requerem respostas instantâneas, como em sistemas de tempo real. Portanto, ao adotar a arquitetura serverless, é importante ponderar essas potenciais desvantagens em relação aos benefícios oferecidos.

6.2. Cenários

Cenário 1: Ao acessar a aplicação em um dispositivo móvel com tela de dimensões reduzidas, os elementos de interface do frontend devem se adaptar de maneira inteligente para preservar a experiência do usuário. Isso inclui redimensionamento de proporções, reajuste de posicionamento e outros ajustes necessários, enquanto a identidade visual da aplicação deve ser mantida nos layouts adaptados.

Cenário 2: Ao acessar a aplicação e utilizar o modo de alto contraste, o sistema deverá fazer as adaptações necessárias, tais como alterar o esquema de cores para um esquema que permita uma fácil identificação dos botões, exibindo cores acessíveis, aumentando o contraste das imagens e aumentando o tamanho e legibilidade das fontes ao longo de todo o sistema.

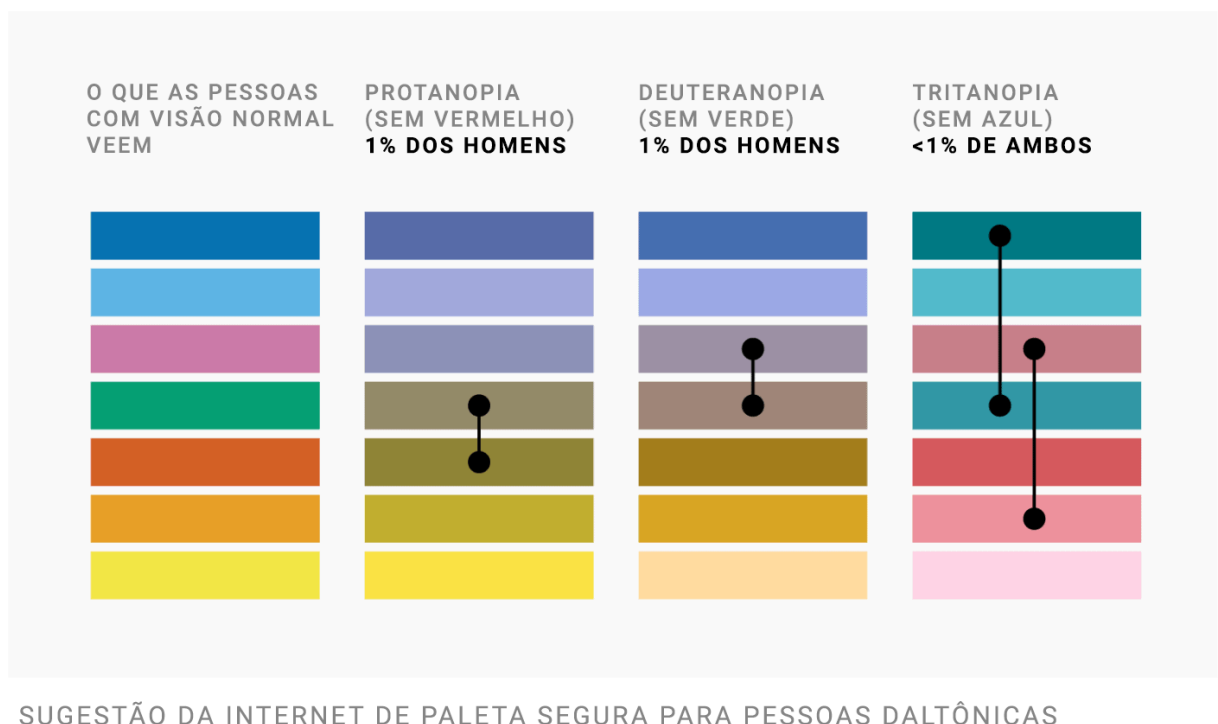


Figura 14: Demonstração das cores de acordo com os diferentes tipos de daltonismo

Cenário 3: Ao acessar a aplicação e utilizar o módulo de controle de usuário, ao alterar o perfil de acesso de algum usuário, o mesmo deve ser redirecionado para a tela de login após qualquer ação dentro do sistema e deverá ser necessário reinserir as credenciais de acesso.

Cenário 4: Ao acessar a aplicação e utilizar o módulo de controle de usuário, ao excluir o perfil de acesso de algum usuário, o mesmo deve ser redirecionado para a tela de login após qualquer ação dentro do sistema e não deverá mais conseguir acessar o sistema.

Cenário 5: Ao acessar a aplicação, o usuário deverá perceber uma experiência fluida de utilização do sistema, sem travamentos ou interrupções. Do login até a visualização de um checklist, o usuário deverá conseguir realizar as operações necessárias em até um minuto.

6.3. Avaliação

● Cenário 1

Atributo de qualidade	Portabilidade
Requisito de qualidade	A aplicação deve se ajustar automaticamente para diversos tipos de layout, sejam web ou mobile
Preocupação	Fornecer uma experiência de uso não satisfatória
Cenário	Cenário 1
Ambiente	Funcionamento com carga normal.
Estímulo	Usuário acessando as principais áreas do sistema
Mecanismo	Utilizar uma tecnologia responsiva
Medida de resposta	A aplicação deve se ajustar a dispositivos com diversos tamanhos de tela
Riscos	Uma experiência não satisfatória pode resultar em dificuldades ao utilizar o sistema
Pontos sensíveis	N/A
Trade-offs	N/A

Evidências

A aplicação apresentou o comportamento esperado, ajustando apropriadamente quando acessada tanto a partir de um celular, quanto a partir de um computador.

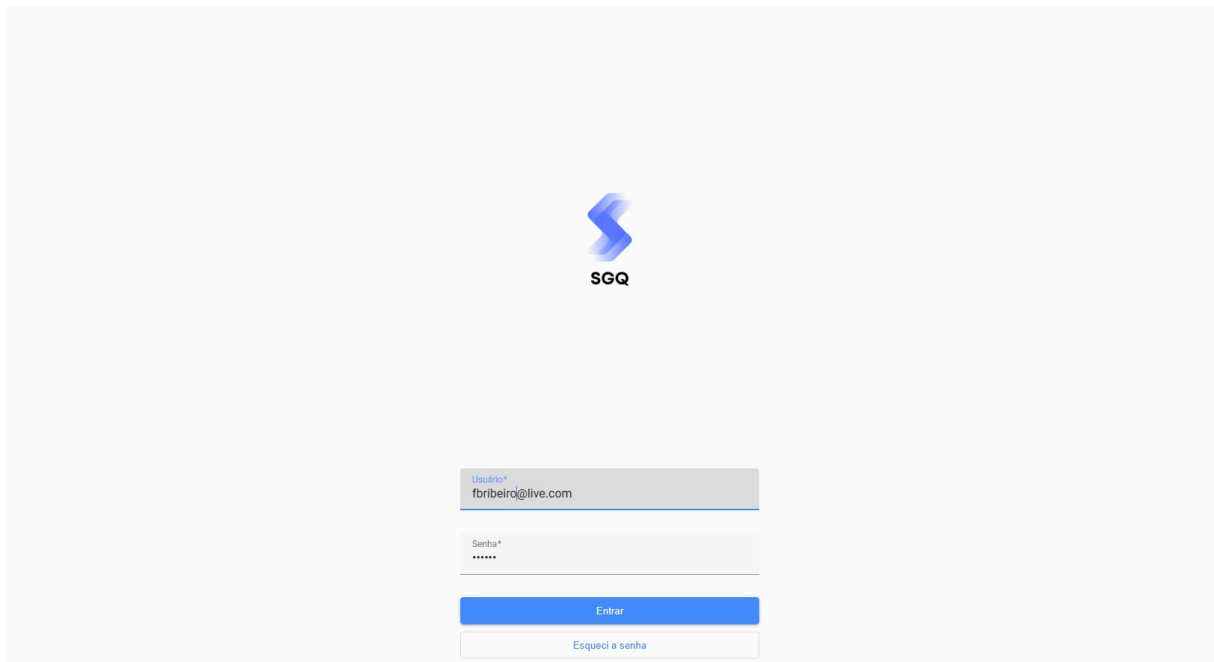


Figura 15: Utilização a partir de um computador – Pagina de login

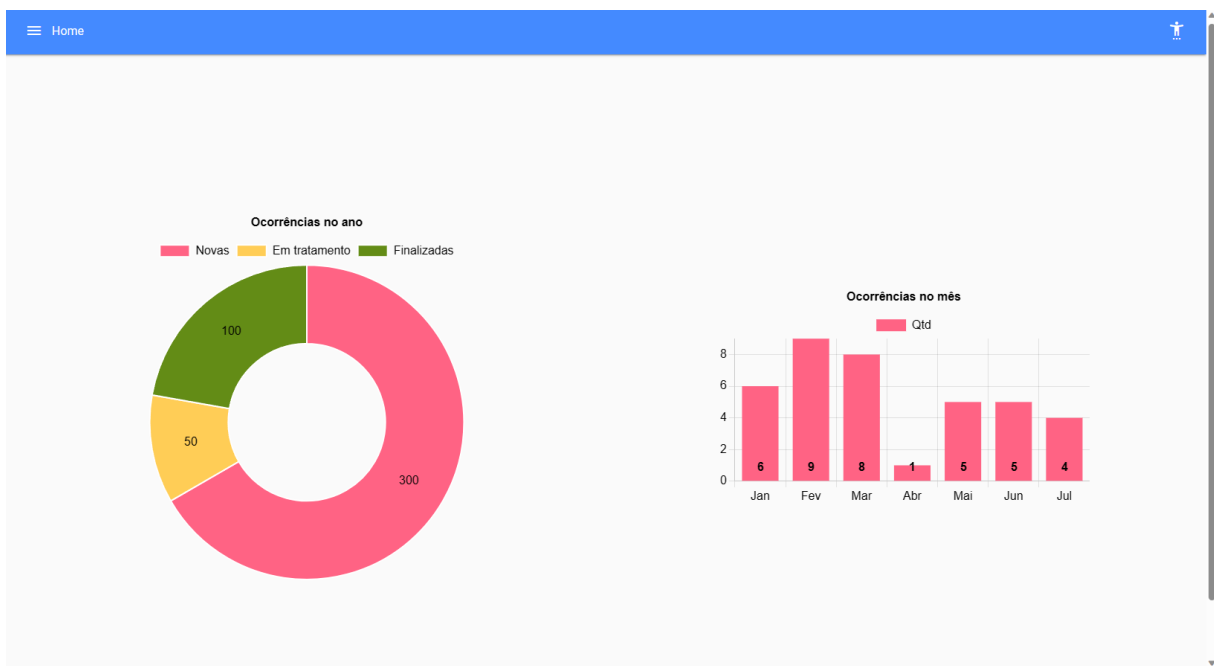
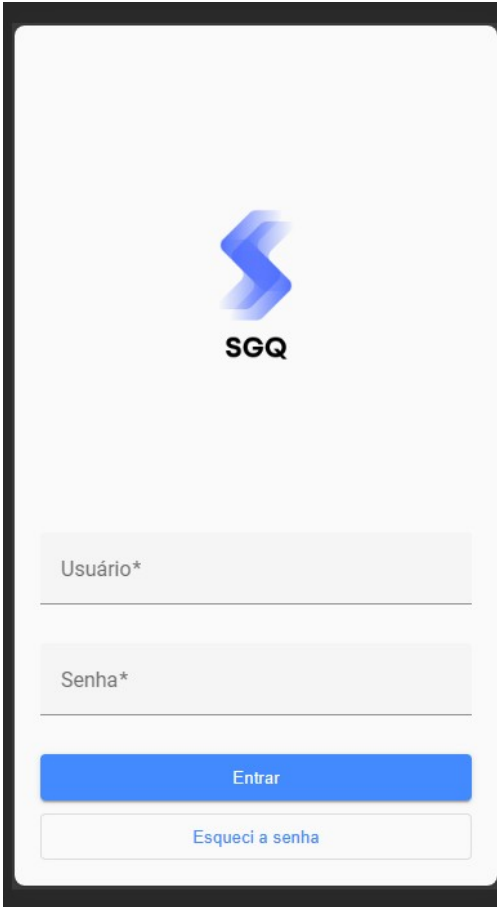


Figura 16: Utilização a partir de um computador – Home

Figura 17: Utilização a partir de um computador – Usuarios

Figura 18: Utilização a partir de um computador – Checklist



The image shows a mobile application login screen. At the top center is the SGQ logo, which consists of a blue stylized 'S' shape above the letters 'SGQ'. Below the logo are two input fields: the first is labeled 'Usuário*' and the second is labeled 'Senha*'. Below these fields is a blue button labeled 'Entrar'. At the bottom, there is a link labeled 'Esqueci a senha'.

Figura 19: Utilização a partir de um celular – Login



Figura 20: Utilização a partir de um celular – Home

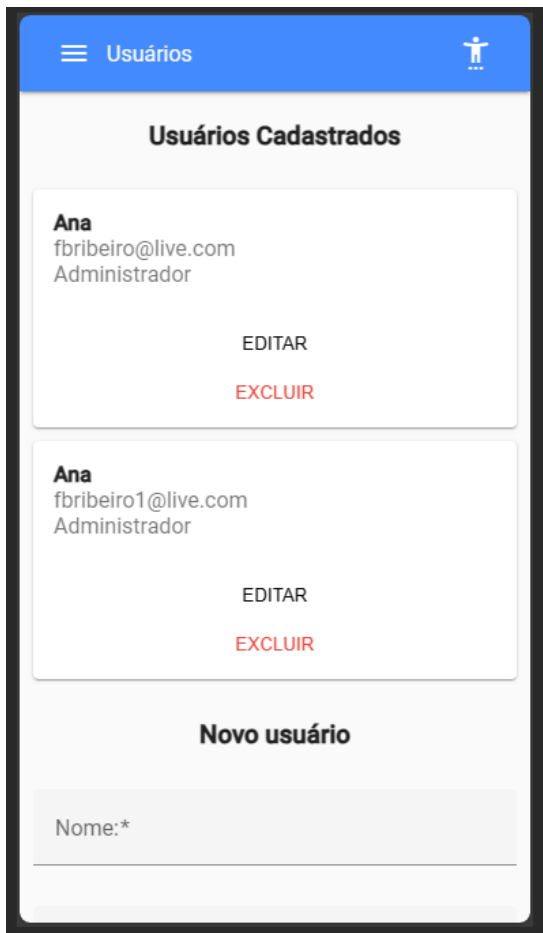


Figura 21: Utilização a partir de um celular – Usuarios

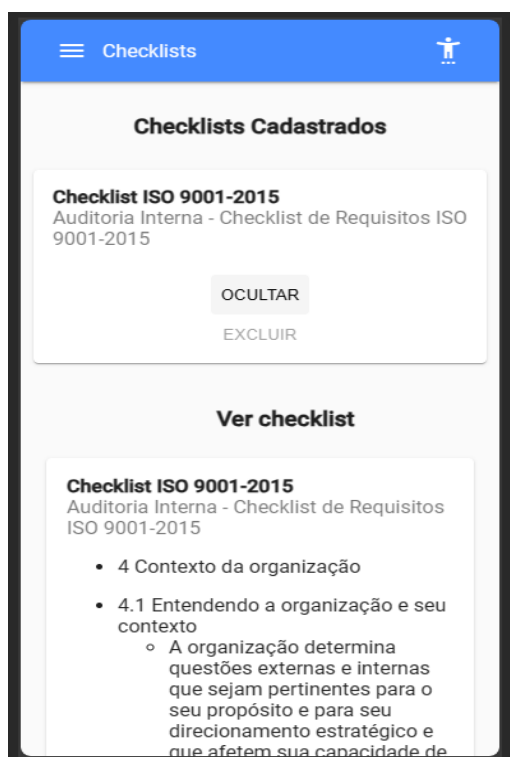


Figura 22: Utilização a partir de um celular – Checklists

● **Cenário 2**

Atributo de qualidade	Acessibilidade
Requisito de qualidade	A aplicação deve permitir que o sistema seja utilizado por pessoas com deficiências visuais que limitem a visualização de cores, utilizando um modo de alto contraste
Preocupação	Fornecer uma experiência de uso não satisfatória para pessoas com deficiências visuais
Cenário	Cenário 2
Ambiente	Funcionamento com carga normal.
Estímulo	Usuário acessando as principais áreas do sistema
Mecanismo	Modo de alto contraste
Medida de resposta	A aplicação deve se ajustar corretamente ao modo de alto contraste, alterando fontes e cores
Riscos	Uma experiência não satisfatória para pessoas com deficiências visuais pode resultar em dificuldades ao utilizar o sistema
Pontos sensíveis	N/A
Trade-offs	N/A

Evidências

A aplicação apresentou o comportamento esperado, ajustando-se apropriadamente quando em modo alto contraste.

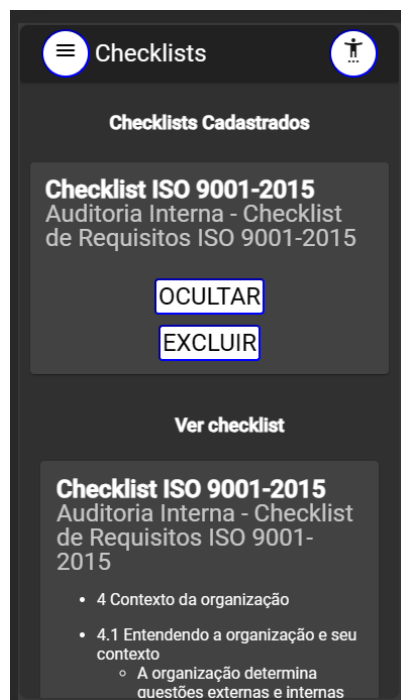


Figura 23: Modo de alto contraste

● Cenário 3

Atributo de qualidade	Segurança
Requisito de qualidade	A aplicação deve impedir que usuários não autorizados acessem ao sistema após alteração de seu perfil
Preocupação	Fornecer segurança de uso e proteção dos dados do sistema
Cenário	Cenário 3
Ambiente	Funcionamento com carga normal.
Estímulo	Usuário acessando as principais áreas do sistema
Mecanismo	JWT
Medida de resposta	A aplicação deve impedir acesso não autorizado
Pontos sensíveis	N/A
Trade-offs	N/A

Evidências

A aplicação apresentou o comportamento esperado, impedindo acesso à áreas não autorizadas do sistema.

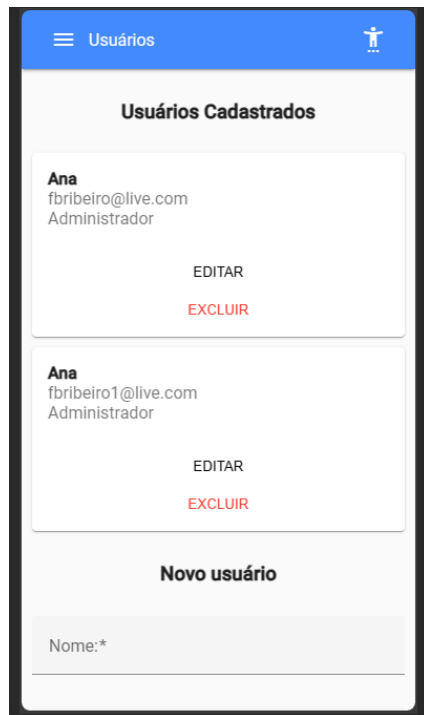


Figura 24: Lista de usuários cadastrados

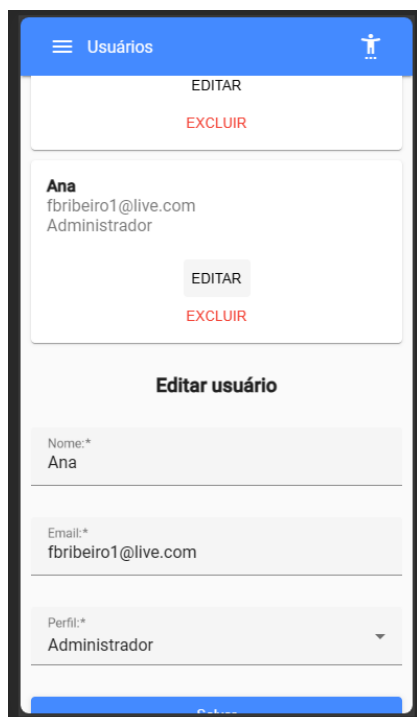
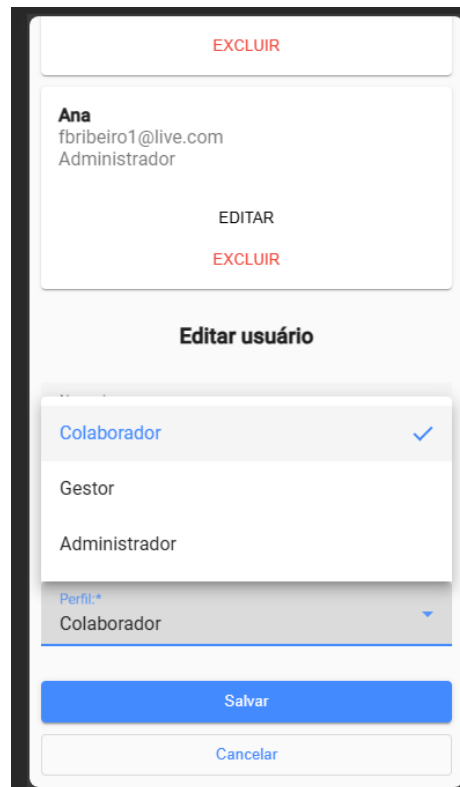
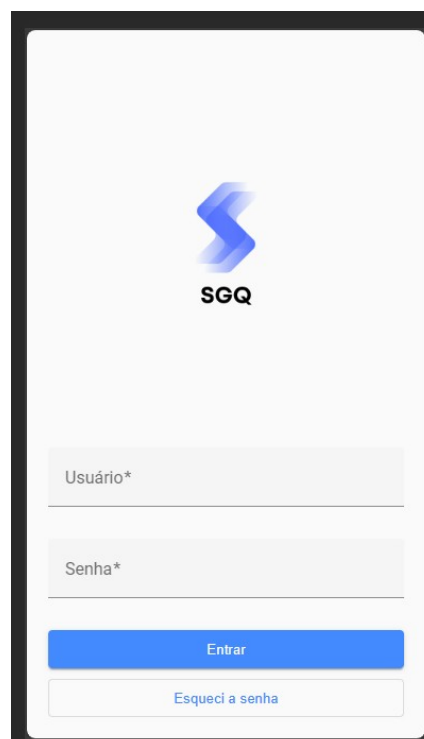


Figura 25: Perfil de uso usuário fbribeiro1@live.com



The image shows a mobile application interface for editing a user profile. At the top, there is a red button labeled "EXCLUIR". Below this, a user card displays the name "Ana", email "fbribeiro1@live.com", and role "Administrador". Under the card are buttons for "EDITAR" and another "EXCLUIR". The main section is titled "Editar usuário". It features a dropdown menu for selecting a profile, with options "Colaborador" (selected with a blue checkmark), "Gestor", and "Administrador". Below the dropdown is a label "Perfil: *" and the selected value "Colaborador". At the bottom are two buttons: a blue "Salvar" button and a white "Cancelar" button.

Figura 26: Alteração de perfil para colaborador



The image shows a mobile application login screen. At the top center is the SGQ logo, which consists of a blue stylized 'S' shape above the text "SGQ". Below the logo are two input fields: "Usuário*" and "Senha*". Under these fields is a blue "Entrar" button. At the bottom is a white button with the text "Esqueci a senha" in blue.

Figura 27: Usuário deslogado automaticamente

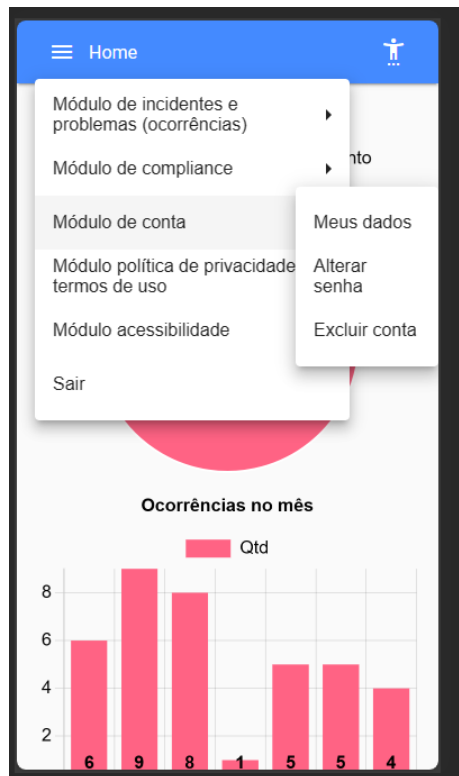


Figura 28: Usuário logado com perfil de acesso correspondente

● Cenário 4

Atributo de qualidade	Segurança
Requisito de qualidade	A aplicação deve impedir que usuários não autorizados acessem ao sistema após exclusão de seu perfil
Preocupação	Fornecer segurança de uso e proteção dos dados do sistema
Cenário	Cenário 4
Ambiente	Funcionamento com carga normal.
Estímulo	Usuário acessando as principais áreas do sistema
Mecanismo	JWT
Medida de resposta	A aplicação deve impedir acesso não autorizado

Pontos sensíveis	N/A
Trade-offs	N/A

Evidências

A aplicação apresentou o comportamento esperado, impedindo acesso ao sistema.

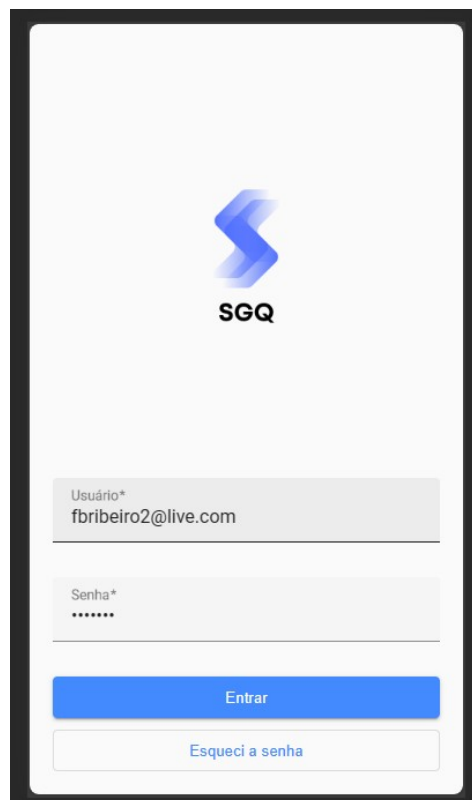


Figura 29: Usuário logando como fbribeiro@live.com

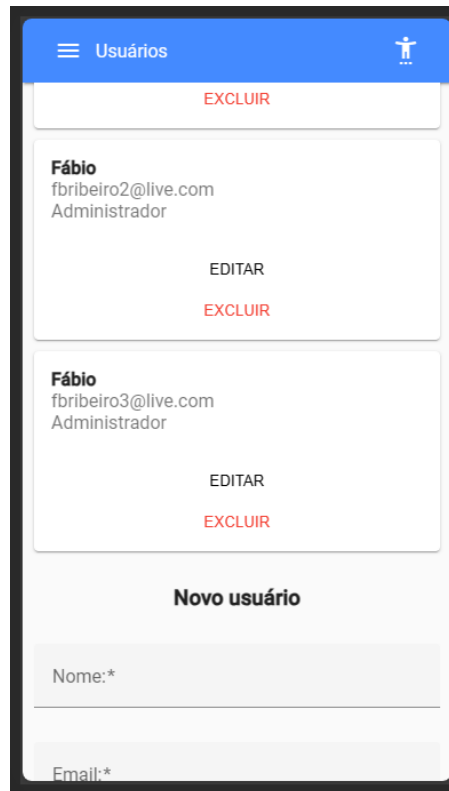


Figura 30: Excluindo o próprio usuário

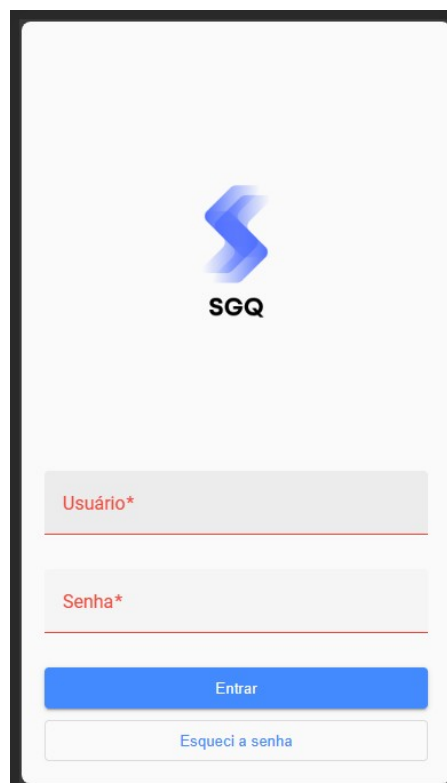


Figura 31: Retornando para login

● **Cenário 5**

Atributo de qualidade	Usabilidade
Requisito de qualidade	A aplicação deve ser fluida e simples de utilizar, de modo que seja possível ir do login até a visualização do checklist em menos de um minuto
Preocupação	Experiência de usuário
Cenário	Cenário 4
Ambiente	Funcionamento com carga normal.
Estímulo	Usuário acessando módulo de checklist
Mecanismo	Menus
Medida de resposta	A aplicação deve permitir que o acesso ocorra em até um minuto
Pontos sensíveis	N/A
Trade-offs	N/A

Evidências

A aplicação apresentou o comportamento esperado, demonstrando fluidez desde o login até a visualização do checklist

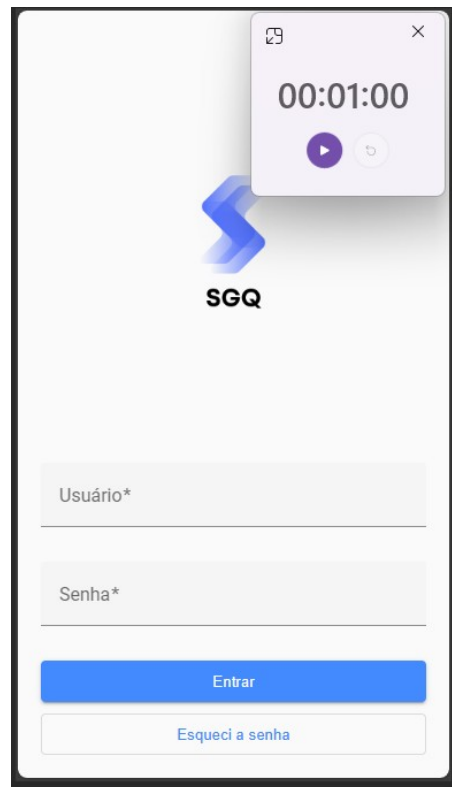


Figura 32: Tela de login

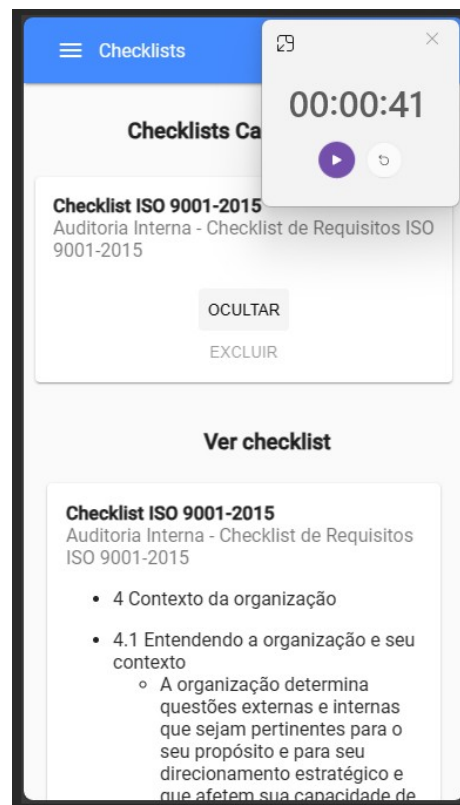


Figura 33: Tela de checklists

6.4. Resultado

Com base na arquitetura proposta e após a prova de conceito, foi possível identificar alguns pontos fortes e fracos e oportunidades de melhoria. Abaixo, alguns desses pontos:

Pontos fortes:

1. A arquitetura serverless se mostrou funcional e rápida de ser desenvolvida;
2. Apesar de gerar complexidade por contra do escopo restrito de cada função, a extensibilidade das funções é interessante, e é fácil criar novas funções a partir de qualquer ponto do sistema;
3. É simples gerenciar infraestrutura serverless, já que o escalonamento é autogerenciável (apesar de configurável);
4. O modo de alto contraste foi relativamente mais simples de ser implementado do que pensado originalmente. Apenas tive que usar sobrecarga de temas no Angular;
5. Utilizando a suíte de cloud da Azure de ponta a ponta, a integração entre os serviços é quase automática, dependendo de pouquíssima configuração;

Pontos fracos:

1. A previsibilidade de custos é mais complexa do que quando utilizando outras técnicas, tais como microsserviços tradicionais;
2. Ao utilizarmos a suite de cloud da Azure, geramos acoplamento tecnológico dentro da própria Azure, e se quisermos migrar para outras soluções tais qual AWS, precisaremos alterar razoavelmente o código;
3. Serverless pode ter um alto custo quando ocorre escalonamento;

Pontos de melhoria

1. Poderia ter linkado o Azure Cosmos DB diretamente como input/output ou triggers, a partir das funções. Delegamos responsabilidades para as funções que poderiam ser um pouco mais simplificadas;

7. Conclusão

Acredito que os objetivos foram atingidos de maneira satisfatória. Com a prova de conceito, é possível demonstrar a pipeline de funções, salvando e recuperando informações do banco, enviando e consumindo mensagens das filas. Se esse projeto fosse continuado e implementado, consigo enxergá-lo como funcional e possível com a base que foi consolidada nesse documento.

APÊNDICES

URL do repositório: <https://github.com/fbalbinoribeiro/sgq>

URL do vídeo: <https://youtu.be/ixuJUKc5t4M>

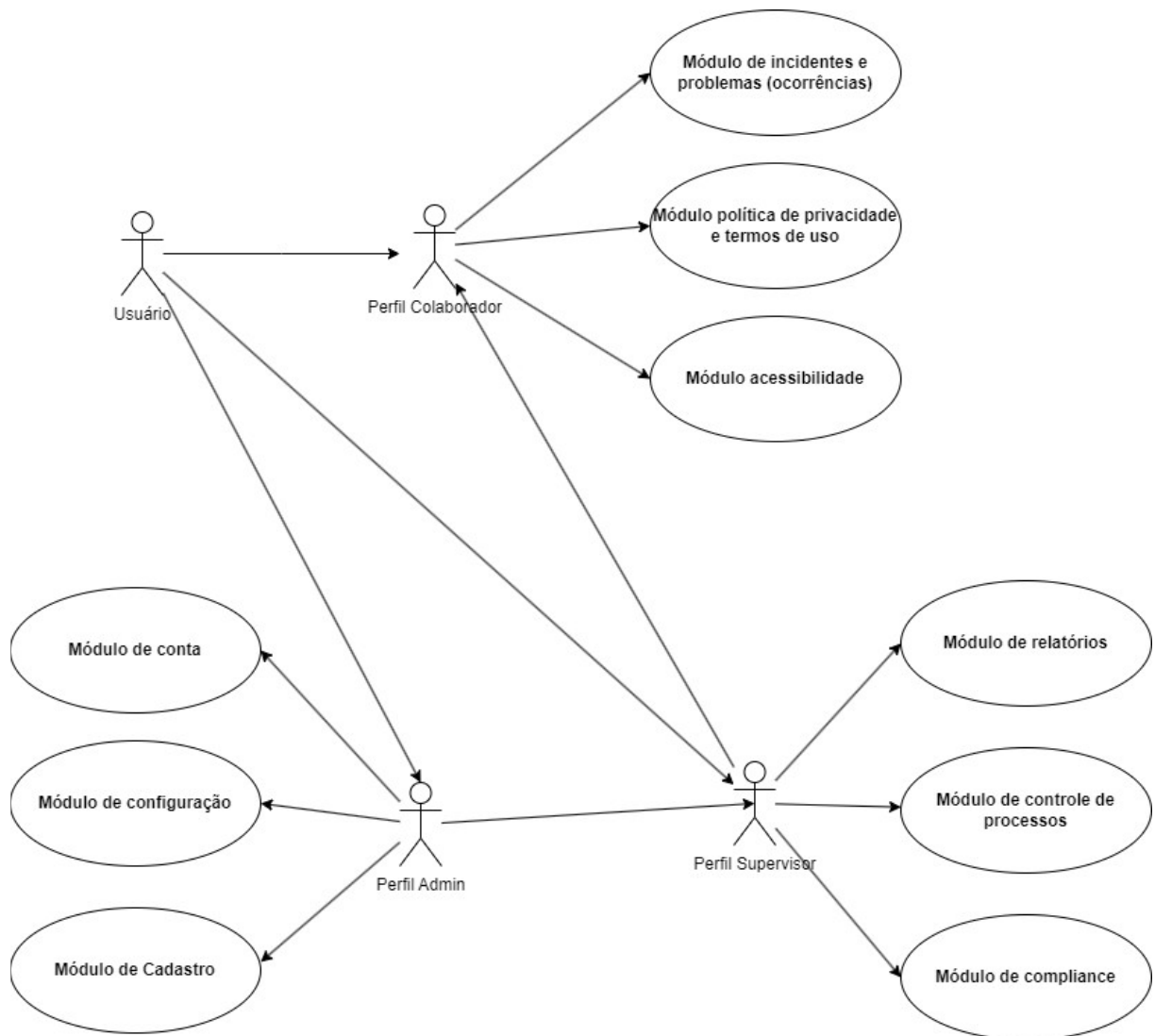


Figura 34: Diagrama de casos de uso dos requisitos funcionais

CHECKLIST PARA VALIDAÇÃO DOS ITENS E ARTEFATOS DO TRABALHO

Nº	Item a ser cumprido	Sim	Não	Não se aplica
Completeza do documento				
	Todos os elementos iniciais do documento (capa, contracapa, resumo, sumário...) foram definidos?	X		
	Os objetivos do trabalho (objetivos gerais e pelo menos três específicos) foram especificados?	X		
	Os requisitos funcionais foram listados e priorizados?	X		
	Os requisitos não funcionais foram listados identificados usando o estilo estímulo-resposta?	X		
	As restrições arquiteturais foram definidas?	X		
	Os mecanismos arquiteturais foram identificados?	X		
	O diagrama de caso de uso foi apresentado junto com uma breve descrição de cada caso de uso?	X		
	O modelo de componente e uma breve descrição de cada componente foi apresentada?	X		
	O modelo de implantação e uma breve descrição de cada elemento de hardware foi apresentada?	X		
	Prova de conceito: uma descrição da implementação foi feita?	X		
	Prova de conceito: as tecnologias usadas foram listadas?	X		
	Prova de conceito: os casos de uso e os requisitos não funcionais usados para validar a arquitetura foram listados?	X		
	Prova de conceito: os detalhes da implementação dos casos de uso (telas, características, etc) foram apresentadas?	X		
	Prova de conceito: foi feita a implantação da aplicação e indicado como foi feita e onde está disponível?	X		
	As interfaces e/ou APIs foram descritas de acordo com um modelo padrão?	X		
	Avaliação da arquitetura: foi feita uma breve descrição das características das abordagens da proposta arquitetural?	X		

	Avaliação da arquitetura: Os atributos de qualidade e os cenários onde eles seriam validados foram apresentados?	X		
	Avaliação da arquitetura: a avaliação com as evidências dos testes foi apresentada?	X		
	Os resultados e a conclusão foi apresentada?	X		
	As referências bibliográficas foram listadas?			X
	As URLs com os códigos e com o vídeo da apresentação da POC foram listadas?	X		

Nº	Item a ser cumprido	Sim	Não	Não se aplica
Consistência dos itens do documento				
	Todos os requisitos funcionais foram mapeados para casos de uso?	X		
	Todos os casos de uso estão contemplados na lista de requisitos funcionais?	X		
	Os requisitos não funcionais, mecanismos arquiteturais e restrições c arquiteturais estão coerentes com os modelos de componentes e implantação?	X		
	Os modelos de componentes e implantação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?	X		
	As tecnologias listadas na implementação estão coerentes com os requisitos não funcionais, mecanismos arquiteturais e restrições arquiteturais?	X		
	Os casos de uso e os requisitos não funcionais listados na implementação estão coerentes com o que foi listado nas seções anteriores?	X		
	Os atributos de qualidade usados na avaliação estão coerentes com os requisitos não funcionais na cessão três?	X		
	Os cenários definidos estão no contexto dos casos de uso implementados?	X		
	O apresentado no item resultado está coerente com o que foi mostrado no item avaliação?	X		