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

Altamir Dias Cassiano
Guilherme da Silva Lima

OPERADORA DE PLANO DE SAÚDE

Belo Horizonte
2022

Altamir Dias Cassiano
Guilherme da Silva Lima

OPERADORA DE PLANO DE SAÚDE

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

Orientador(a): Prof. Dr. Pedro A. Oliveira

Belo Horizonte
2022

Dedicamos esse trabalho a nossa família por dividirem conosco esse sonho.

AGRADECIMENTOS

Agradecemos primeiramente a Deus, a nossas esposas por todo apoio desde o início desse projeto, agradecemos também a PUC Minas pela estrutura disponibilizada para realizarmos o curso e aos professores obrigados por todo ensinamento durante tal jornada.

RESUMO

Este projeto aborda a arquitetura de solução para um sistema de Operadora de Plano de Saúde, nele apresentaremos tecnologias e arquiteturas modernas apresentadas na área de tecnologia da informação.

Com os recentes eventos que ocorreram na saúde Mundial, principalmente com a pandemia do Covid-19, onde levou várias pessoas a óbito, existe a devida preocupação com a facilidade de acesso e procedimentos no campo da saúde. Mais que nunca, os investimentos em tecnologias onde oferece aos usuários segurança para realizar procedimentos de dentro de suas casas, fazendo com que o mesmo não se exponha a filas e a riscos eminentes a locomoção até o ponto de atendimento. Para chegar a esse resultado, será abordado a solução arquitetônica com tecnologias modernas apresentadas no curso de formação, trabalhando em comunicação, segurança entre outros.

Palavras-chave: arquitetura de software, saúde, segurança, tecnologia, informação.

SUMÁRIO

1. Apresentação

A Boa Saúde é uma grande operadora de saúde que possui mais de 10.000 colaboradores cadastrados e mais de um 1.000.000 de associados ativos, nos diversos municípios onde atua, fornecendo assim planos odontológico e de saúde.

Na data de 21/01/2022 os dados fornecidos pelo Sistema de Saúde do Governo (https://covid.saude.gov.br) eram de 622.563 óbitos acumulados tendo como causa as complicações do Covid-19 apenas no Brasil.

“A epidemiologia do SARS-CoV-2 indica que a maioria das infecções se espalha por contato próximo (menos de 1 metro), principalmente por meio de gotículas respiratórias. Não há evidência de transmissão eficiente para pessoas em distâncias maiores ou que entram em um espaço horas depois que uma pessoa infectada esteve lá.”[Dados de: https://www.gov.br/saude/pt-br/coronavirus/como-e-transmitido {acesso em 22/01/2022}]

Com o grande número de casos de infectados pelo novo Covid-19, as operadoras privadas que complementam o Sistema Único de Saúde (SUS) no atendimento à população, necessitaram ainda mais de investimento na área de tecnologia. Essa ação se fez necessário devido a necessidade ainda maior de manter os pacientes em locais seguros e longe de filas ou áreas de risco, como postos de saúdes públicos ou pontos de atendimentos privados. Para chegar a esse resultado, será abordado solução tecnológica e arquitetônica de módulos de gestão de atendimentos, controle de pacientes, localização de pontos de atendimentos, inteligência de negócio entre outros.

1.1 Problema

O catálogo tecnológico atual da Boa Saúde Assistência Médica é composto por sistemas e soluções de diversas áreas de aplicação, em diferentes plataformas computacionais, tendo assim um grande legado tecnologia na empresa. Possui Sistemas Administrativo- Financeiro, Sistema de Gestão de Produtos e Serviços, Sistema de Atenção à Saúde, entre outros.

Outra dificuldade que o time da Boa Saúde tem é manter o legado, devido as diversas tecnologias utilizadas, além da manutenção o legado dificulta o crescimento da empresa para modelos mais modernos e que atendam as novas necessidades do negócio.

Além do mais, os sistemas legados, possuem dificuldade de interoperabilidade com soluções internas e externas da própria empresa ou de mercado.

1.2 Objetivo do trabalho

Com a intenção de melhorar as tecnologias de interoperabilidade, garantindo a eficiência operacional, será apresentado a descrição do projeto de Arquitetura do Sistema de Gestão da Saúde do Associado (no qual ganha a sigla GISA).

Aqui detalharemos os requisitos e tecnologias a serem adotadas durante o período de construção.

Os objetivos são apresentar uma solução para Operadora de Plano de Saúde, onde aumente o controle e monitoria, para prover dados suficientes ou mitigar cenários de risco para o negócio.

Os objetivos específicos são:

I. **Módulo de Informações Cadastrais:** trata-se de um módulo cujo escopo consiste em obter e manter informações de todos os prestadores e associados, dentre as quais se destacam: informações pessoais, de localização, de formação, de saúde e outras necessárias ao negócio da empresa. Essas informações têm como fonte os próprios prestadores e associados, dados existentes em sistemas médicos e registros de consultas e exames, etc.

II. **Módulo de Gestão e Estratégia**: tem como escopo prover a gestão estratégica de todos os projetos, produtos e serviços da empresa, com indicadores do andamento individual e global dos projetos e serviços, na forma de um cockpit. Para este módulo será utilizada uma ferramenta de gestão corporativa adquirida no mercado;

III. **Módulo de Serviços ao Associado**: esta parte do sistema é baseada numa solução de workflow, com o uso de Business Process Management – BPM. Por meio deste módulo é possível desenhar, analisar e acompanhar todos os processos existentes na empresa - tanto os já existentes quanto os que ainda serão implantados, desta forma melhorando o desempenho e a eficiência desses processos;

1,3 Definições e Abreviaturas

- **GISA**: Gestão Integral da Saúde do Associado;
- **API**: Application Programming Interface;
- **RNF**: Requisito - Funcional;
- **RNF**: Requisito Não - Funcional;
- **POC**: Proof of Concept – Prova de conceito;
- **UC**: Use Case ou Caso de Uso;
- **MIC**: Módulo de informações cadastrais;
- **MGE**: Módulo de Gestão e Estratégia;
- **MSA**: Módulo de Serviço ao Associado;
- **SQS**: Amazon Simple Queue Service;
- **SAAS**: Software as a Service – Software como serviço;

2. Especificação da Solução

Esta seção descreve os requisitos comtemplados neste projeto arquitetural, dividido em dois grupos: Funcionais e Não Funcionais. As informações que estão contidas nesta sessão, são informações advindas da documentação previamente fornecida.

2.1 Requisitos Funcionais

ID	Descrição Resumida	Dificuldade (B/M/A)*	Prioridade (B/M/A)*
RF01	**Módulo de informações cadastrais:** Manter e obter informações. O sistema deve ser capaz de obter e manter informações como: informações pessoais, localização, saúde e outras necessárias ao negócio.	M	A
RF02	**Módulo de Gestão e Estratégia:** O sistema deve ser capaz de disponibilizar indicadores individuais e globais, tendo como fim a gestão de projetos e serviços. Deve ser apresentado em modo de cockpit para melhor visualização.	A	A
RF03	**Módulo de Serviços ao Associado:** Por meio desse módulo o usuário associado terá a possibilidade de recuperar os processos previamente cadastrados.	M	A

*B=Baixa, M=Média, A=Alta.

2.2 Requisitos Não Funcionais

ID	Descrição	Prioridade B/M/A
RNF01	O sistema deve ser acessível nas plataformas *web* e móvel	A
RNF02	O sistema deve ser hospedado em nuvem híbrida, sendo a forma de hospedagem documentada;	A
RNF03	O sistema deve ser modular e implantável por módulos, de	A

	acordo com as prioridades e necessidades da empresa;	
RNF04	O sistema deve apresentar interface responsiva	A
RNF05	O sistema deve apresentar bom desempenho, não ultrapassando 3 segundos em suas operações	M
RNF06	O sistema deve apresentar boa manutenibilidade	M
RNF07	O sistema deve ser testável em todas as suas funcionalidades	
RNF08	O sistema deve ser recuperável (resiliente) no caso da ocorrência de erro	A
RNF09	O sistema deve utilizar tecnologias de APIs para possibilitar suas integrações	M
RNF10	Deve ser desenvolvido utilizando recurso de configuração, com integração contínua	A

2.3 Restrições Arquiteturais

- A solução deve fornecer o menor custo possível para empresa;
- O sistema deve ser modular com a finalidade de fácil implantação;
- O sistema deve possibilitar a implantação em nuvem hibrida;
- As APIs devem respeitar o padrão RESTful;

2.4 Mecanismos Arquiteturais

Análise	*Design*	Implementação
Persistência	Serviço do Google Cloud para armazenamento	Firebase Storage
Front end	Single Page Application	React
Back end	Processamento de regras de negócio	C#, Dot Net Core
Integração	Publicação automatizada e continua	Azure DevOps
Auditoria	Auditória baseada em logs do sistema;	Serilog com sinks console e arquivo
Deploy	Processo de compilação e publicação	Azure DevOps
Versionamento e Gestão	Manutenibilidade	GitHub, Azure DevOps

de ciclo de Vida do Fonte		
Hospedagem	Hospedagem em Cloud	AWS - Amazon Web Service
Event Bus	Interface com legado	AWS - SQS

3. Modelagem Arquitetural

Esta seção apresenta a modelagem arquitetural da solução proposta, de forma a permitir seu completo entendimento visando à implementação da prova de conceito.

3.1 Macroarquitetura

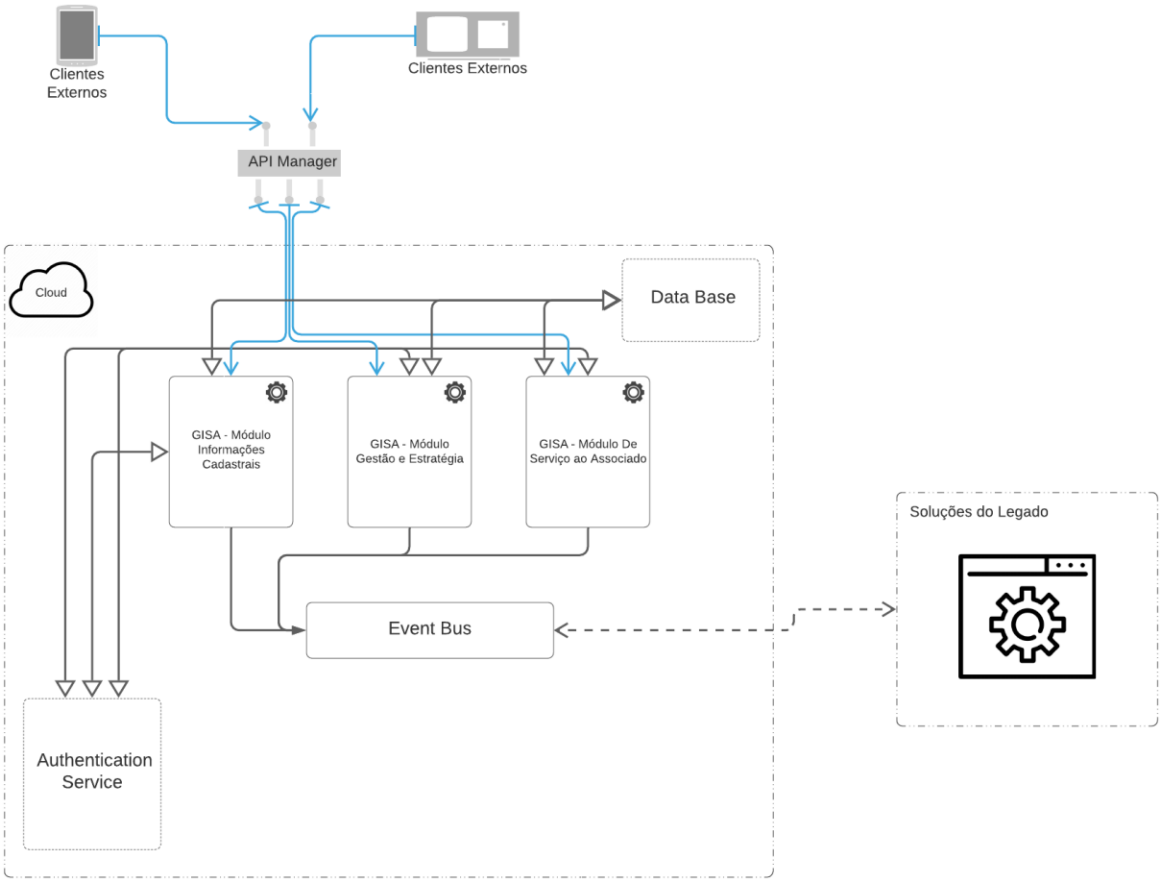

Figura 1 - Visão Geral da Solução. Criado com https://lucid.app/.

A figura 1 mostra a especificação o diagrama geral da solução proposta, com todos os módulos que serão apresentados na POC e suas interfaces com as soluções do legado.

3.2 Descrição Resumida dos Casos de Uso / Histórias de Usuário

UC01 – Cadastro de Prestador na aplicação	
Descrição	Com dispositivos móveis ou via Browser Web, deve possibilitar o cadastro de Prestador.
Atores	Usuário
Prioridade	A
Requisitos associados	RF01
Fluxo Principal	O usuário deve poder realizar seu o cadastro através desta funcionalidade, de forma autônoma e segura, tanto utilizando uma solução móvel como Smartphones e tablets ou ainda por um Desktop utilizando Web Browser.

UC02 – Edição de Prestador previamente cadastrado	
Descrição	Com dispositivos móveis ou via Browser Web, deve possibilitar a edição e de um Prestador previamente cadastrado no sistema.
Atores	Usuário
Prioridade	A
Requisitos associados	RF01
Fluxo Principal	O usuário deve poder realizar a atualização do cadastro através desta funcionalidade, de forma autônoma e segura, tanto utilizando uma solução móvel como Smartphones e tablets ou ainda por um Desktop utilizando Web Browser.

UC03 – Visualização Estratégica de Prestador	
Descrição	Com dispositivos móveis ou via Browser Web, deve possibilitar a visão estratégica de Prestador em forma de cockpit com suas devidas informações.
Atores	Usuário
Prioridade	A
Requisitos associados	RF02
Fluxo Principal	O usuário deve poder realizar o acesso de forma simples e intuitiva à visualização estratégica, nela o usuário tem a possibilidade de filtrar o Prestador.

3.3 Visão Lógica

Esta seção mostra a especificação dos diagramas da solução proposta, com todos os seus componentes, propriedades e interfaces. Nas subseçoes a seguir são apresentados os diagramas de Classes, Componentes e Implantação.

3.3.1 Diagrama de Classes

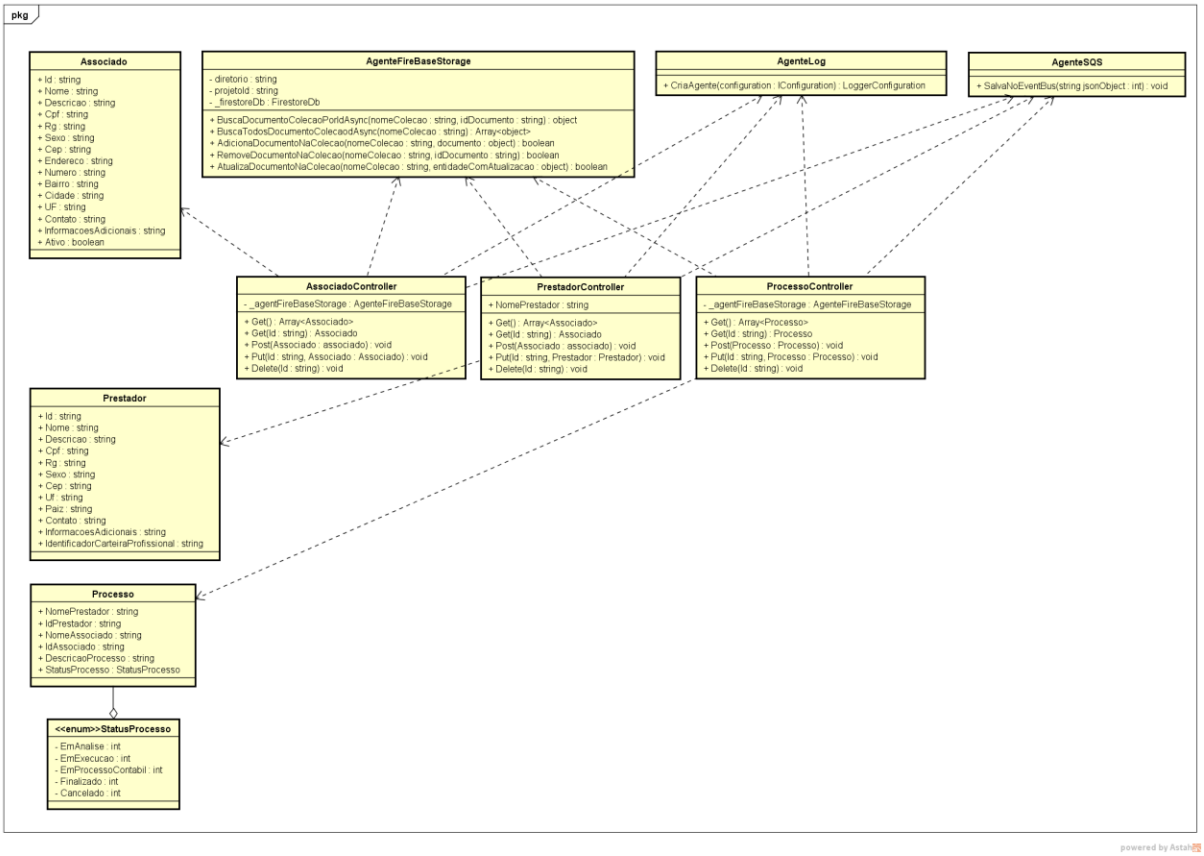

Figura 2 – Diagrama de classes. Criado com Astah Community.

A figura 2 apresenta as classes da aplicação representada pelo diagrama de classes e seus relacionamentos.

3.3.2 Diagrama de Componentes

Abaixo o diagrama de componentes da solução a GISA, nela pode se notar visão sucinta dos componentes e suas interfaces, logo abaixo o texto sobre cada componente.

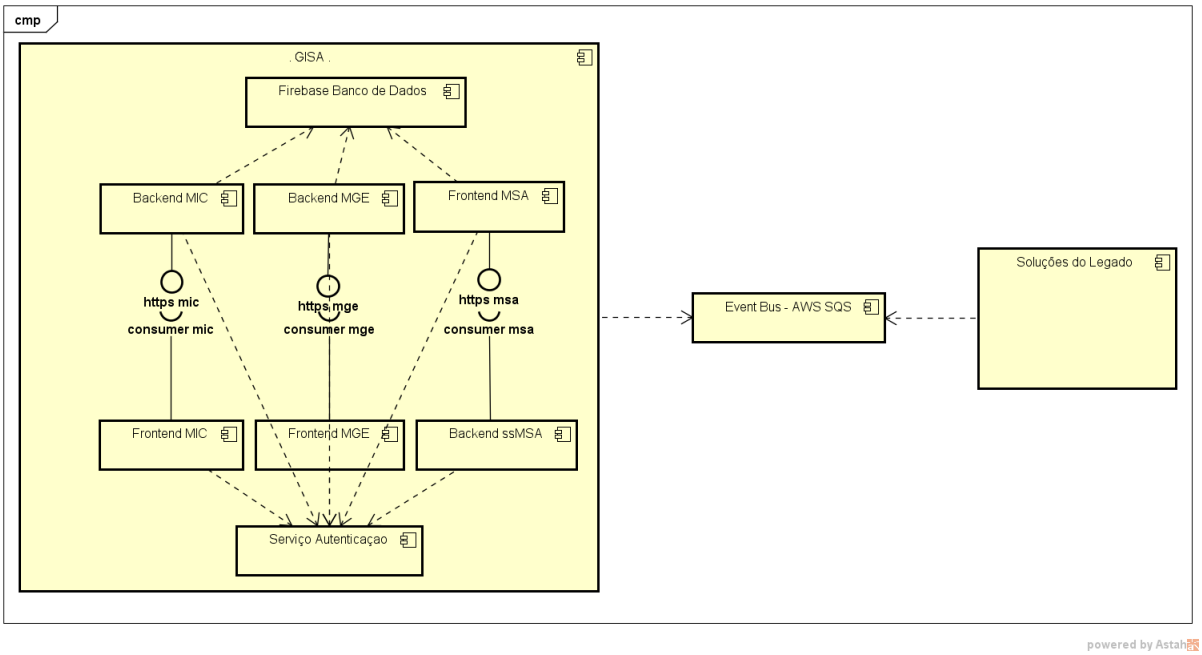

Figura 2 – Diagrama de Componentes. Criado com Astah Community.

• **BackEnd** - **MIC**: BackEnd do Módulo de informações cadastrais, backend baseado em micro-backend com a responsabilidade de receber requisições do componente FrontEnd – MIC, realizar devidos processamentos e entregar para a fila correspondente no Event Bus;

• **FrontEnd** - **MIC**: FrontEnd do Módulo de informações cadastrais, frontend baseado em SPA com a responsabilidade de enviar requisições para o componente BackEnd – MIC;

• **BackEnd** - **MGE**: BackEnd do Módulo de Gestão e Estratégia, backend baseado em micro-backend com a responsabilidade de receber requisições do componente FrontEnd – MGE, realizar devidos processamentos e entregar para a fila correspondente no Event Bus;

• **FrontEnd** - **MGE**: FrontEnd do Módulo de Gestão e Estratégia, frontend baseado em SPA com a responsabilidade de enviar requisições para o componente BackEnd – MGE;

• **BackEnd** - **MSA**: BackEnd do Módulo de Serviço ao Associado, backend baseado em micro-backend com a responsabilidade de receber requisições do componente FrontEnd – MSA, realizar devidos processamentos e entregar para a fila correspondente no Event Bus;

• **FrontEnd** - **MSA**: FrontEnd do Módulo de Serviço ao Associado, frontend baseado em SPA com a responsabilidade de enviar requisições para o componente BackEnd – MSA;

3.3.3 Diagrama de Implantação

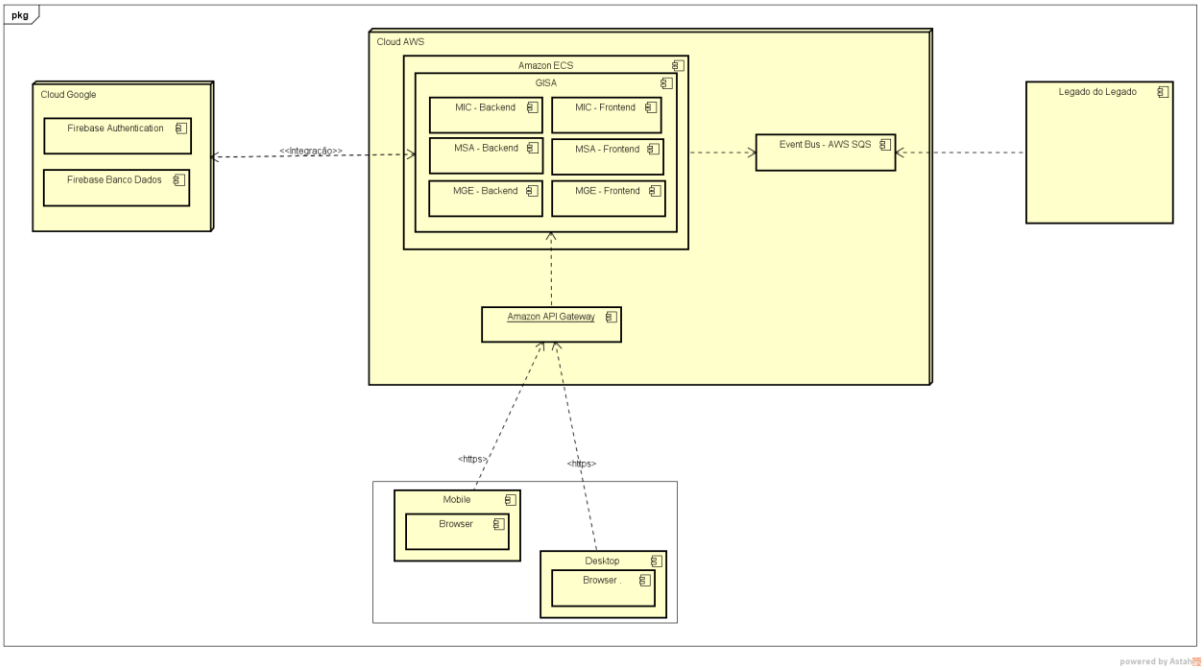

Figura 4 – Diagrama de Implantação. Criado com Astah Community.

O Diagrama de Implantação acima demonstra o ambiente de deploy defendido para suportar a arquitetura do projeto GISA com seus módulos MIC, MSA, MGE. Nele podemos notar a arquitetura multicloud com AWS e Google Cloud.

4. Prova de Conceito (POC) e Protótipo Arquitetural

A prova de conceito desse projeto visa atender as necessidades críticas do MIC. A seguir, será apresentado as evidências dos testes sobre a POC tendo como objetivo atender os requisitos não funcionais de confiabilidade, resiliência e compatibilidade.

Os artefatos utilizados nessa POC estão no link https://github.com/altamirdiascassiano/TCC-PUC.

4.1. Implementação

Para a prova de conceito foi utilizados as tecnologias de mercado de ponta, como Amazon SQS, Google Cloud Firebase Storage para suportar os serviços de base de dados e mensageria. Estas soluções são acessadas pelo projeto GISA. Por sua vez o GISA, está versionado no GitHub e tendo a Integração Contínua do Azure DevOps (vide item 4.1.1).

Abaixo é demonstrado visão do projeto arquitetural em nível do código fonte, apresentado assim a solução criada pelo Visual Studio 2019.

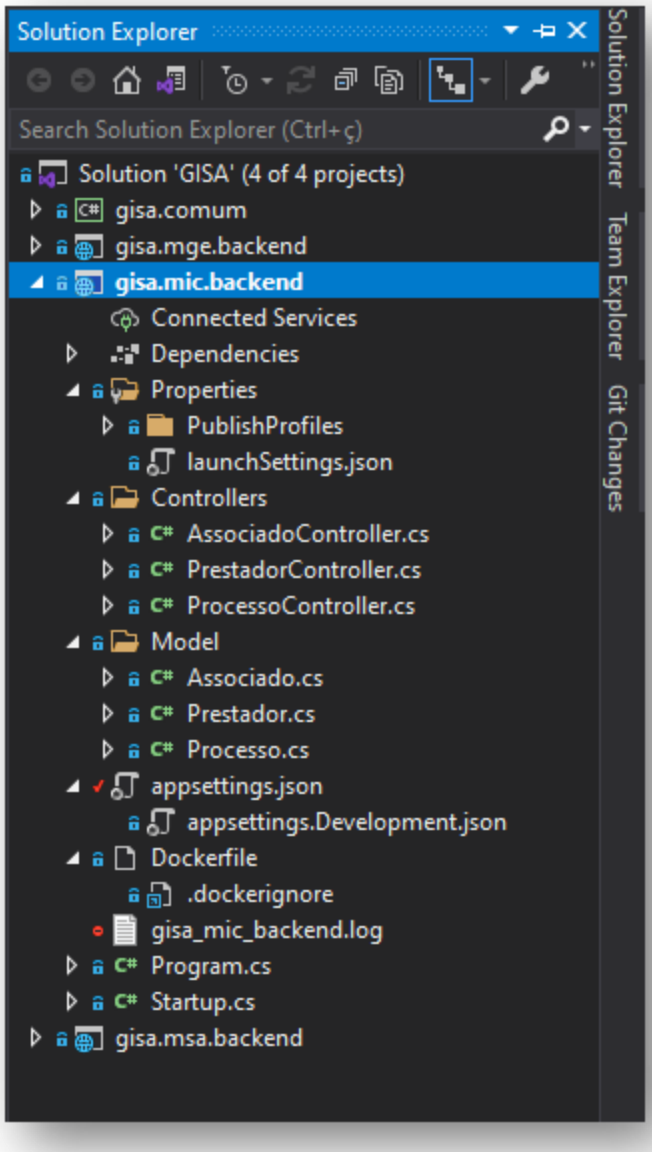

Figura 5 – Solução no Visual Studio 2019.

4.1.1 Integração Continua

Para integração continua está sendo utilizado o Microsoft Azure DevOps (SAAS) como ferramenta de build e deploy automatizado.

A solução é plugada ao repositório de fontes do GISA, que a cada alteração é disparada a execução do pipeline, garantindo a integração contínua conforme imagens abaixo:

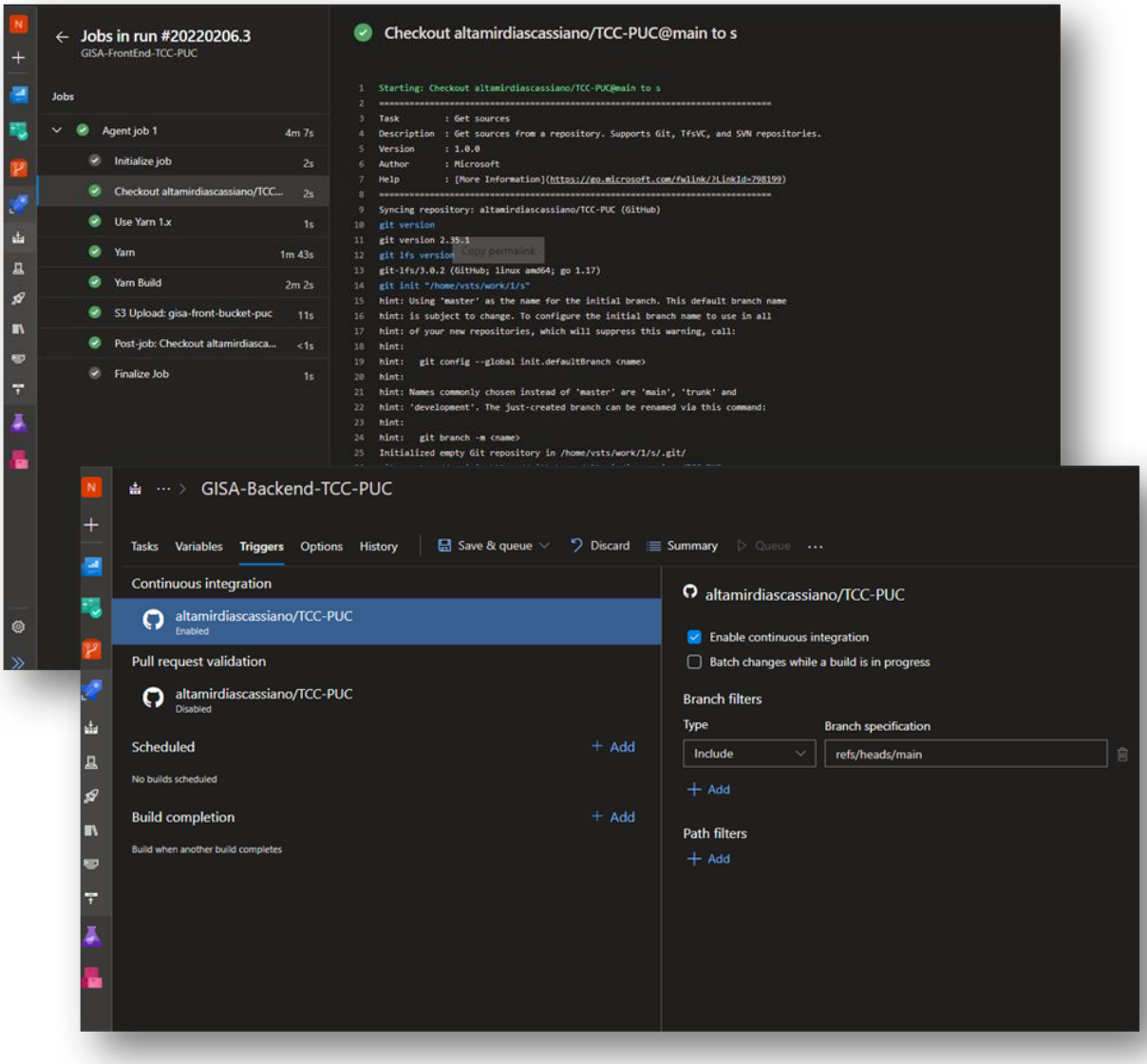

Figura 6 – Integração Continua.

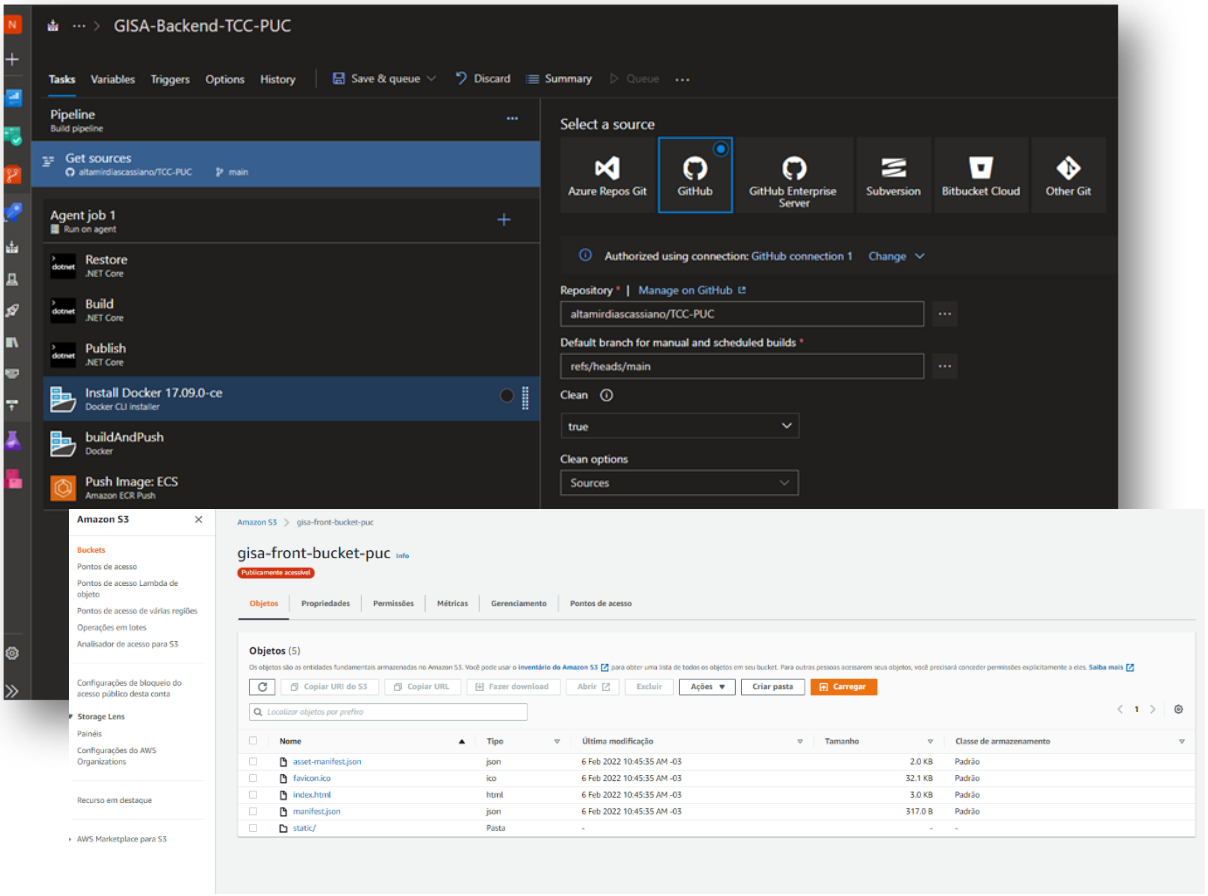

Figura 7 – Resultado Integração Continua.

Na solução implementada buscamos otimizar recursos e gastos com infraestrutura em nuvem, diante disso usamos o conceito multicloud, que significa ter acesso a diferentes recursos oferecidos por diferentes provedores cloud.

Neste exemplo o repositório de fontes da aplicação está no GitHub (SAAS), o pipeline de CI no Azure DevOps (SAAS) e as publicações sendo feitas nos serviços da Amazon AWS.

4.2 Interfaces e APIs

Abaixo são apresentadas as interfaces de comunicação com aplicação desenvolvida. Seguindo boas práticas de mercado o desenvolvimento das APIs foi baseado em requisições HTTP com especificação OpenAPI.

O Swagger abaixo demonstra e as devidas operações:

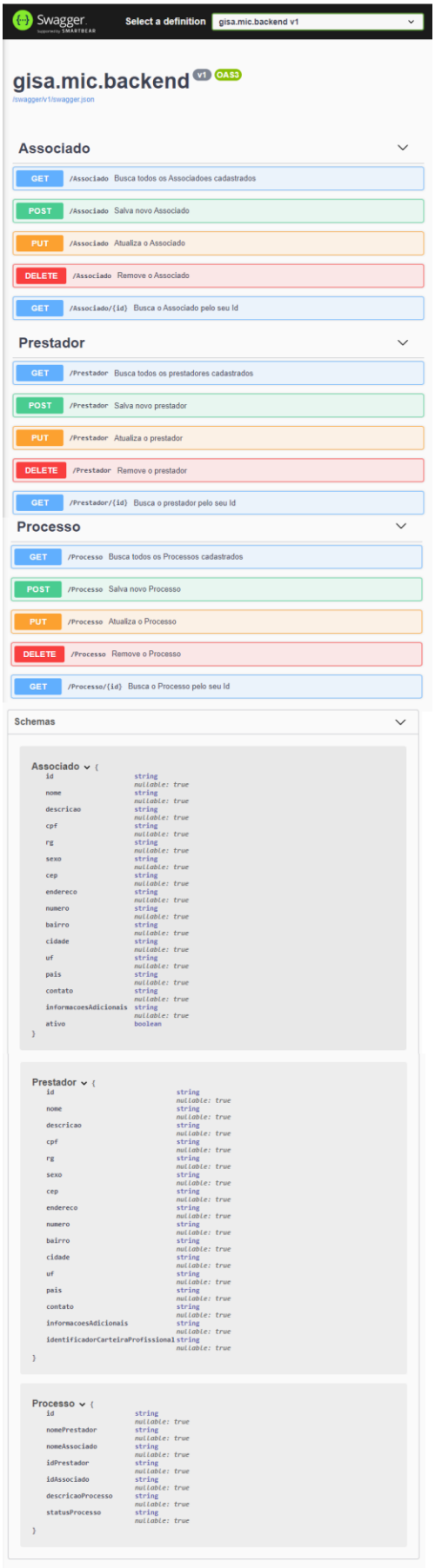

Figura 8 – Swagger API Total.

4.2.1 Associado

A API de Associado (https://localhost:5001/Associado), permite as operações de Busca, cadastro, atualização e remoção de Associado, respeitando assim o verbos HTTP abaixo e documento no Swagger.

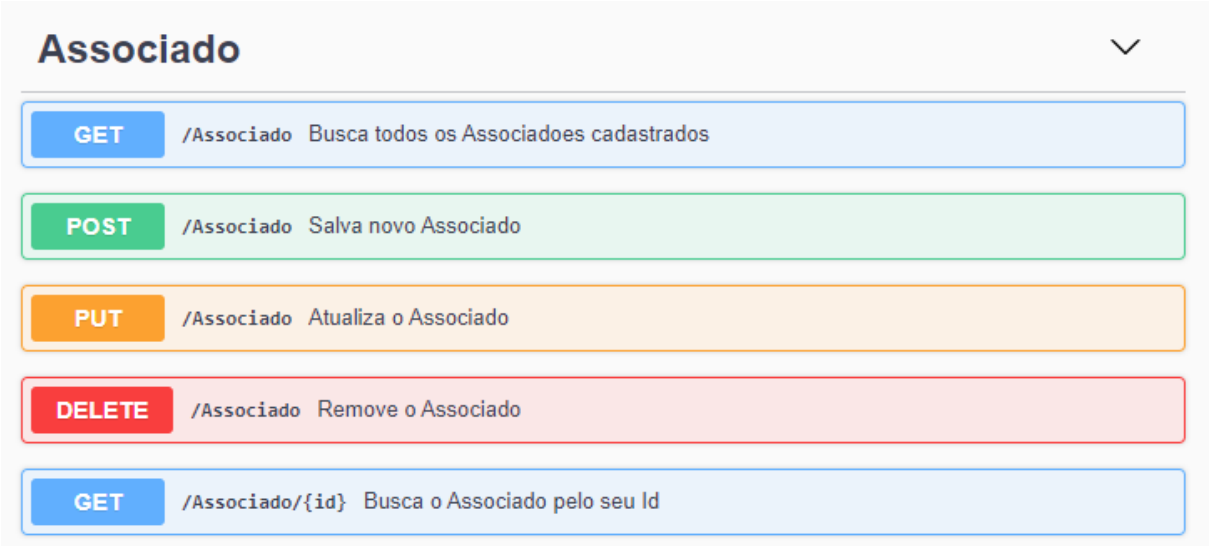

Figura 9 – Swagger API Associado.

4.2.2 Prestador

A API de Associado (https://localhost:5001/Prestador), permite as operações de Busca, cadastro, atualização e remoção de Prestador, respeitando assim o verbos HTTP abaixo e documento no Swagger.

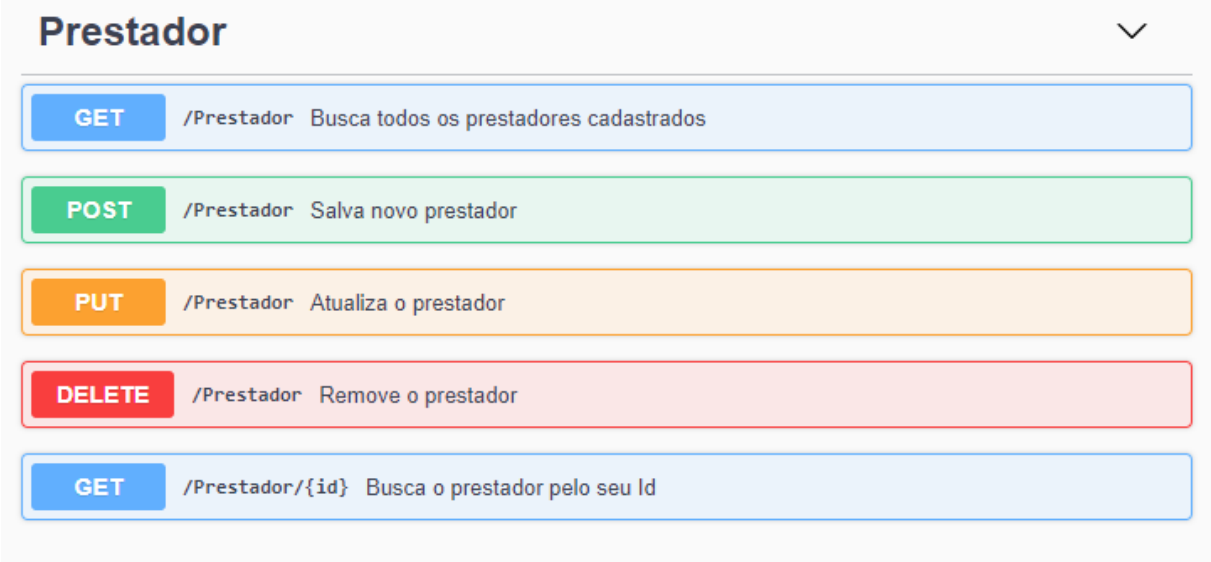

Figura 10 – Swagger API Prestador.

4.2.3 Processo

A API de Associado (https://localhost:5001/Processo), permite as operações de Busca, cadastro, atualização e remoção de Processo, respeitando assim os verbos HTTP abaixo e documento no Swagger.

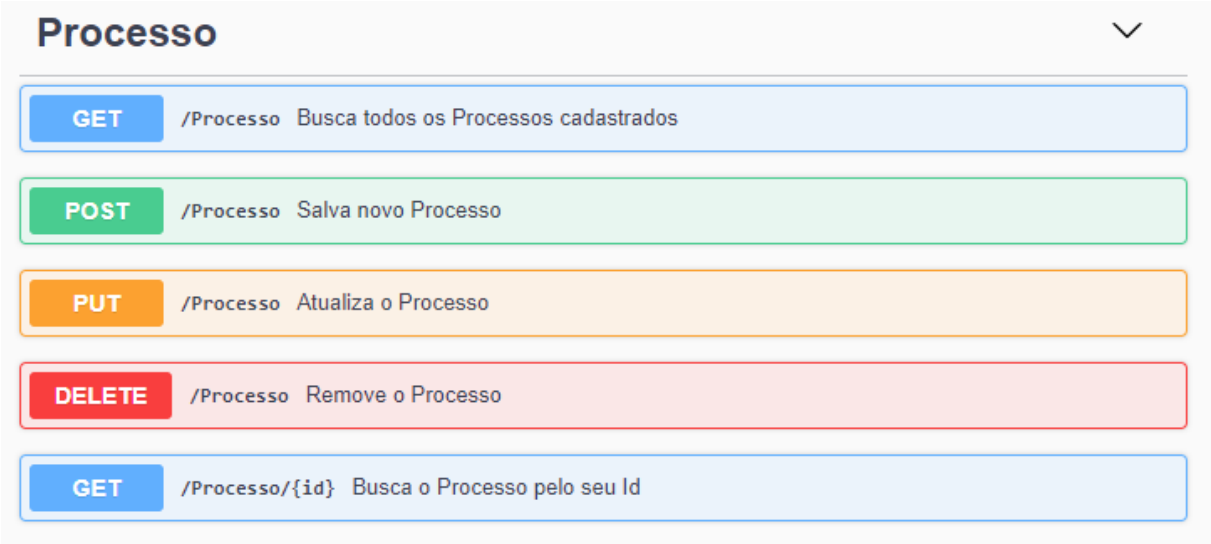

Figura 11 – Swagger API Processo.

5. Avaliação da Arquitetura

5.1. Análise das abordagens arquiteturais

Neste tópico será apresentado as abordagens arquiteturais relevantes neste projeto, devidamente com sua importância e complexidade. Os casos apresentados a seguir foram escolhidos de acordo com sua relevância para o projeto.

Abaixo a apresentação dos módulos de construção sobre a responsabilidade do time de tecnologia da Boa Saúde.

Atributos de Quailidade	Cenários	Importância	Complexidade
Interoperabilidade	Cenário 1: O sistema deve se comunicar com sistemas de	A	M

	outras tecnologias.		
Tolerância a falhas	Cenário 2: O sistema deve permanecer em funcionamento em caso de falha.	A	B
Auditabilidade	Cenário 3: Em casos de falhas, o sistema deve manter os dados da falha afim de ser auditável.	A	B
Usabilidade	Cenário 4: O sistema deve prover boa usabilidade independentemente da plataforma de acesso que o usuário estiver utilizando.	M	M
Manutenibilidade	Cenário 5: O sistema deve ter a manutenção facilitada.	M	M

Legenda: **A** para Alta, **M** para Média e **B** para Baixa.

Para os módulos adquiridos no mercado são considerados que os mesmos devem conter pelo menos as seguintes características: Segurança, Interoperabilidade, Manutenibilidade e Auditabilidade.

5.2. Cenários

- **Cenário 1 - Interoperabilidade**: O sistema quando receber requisição HTTP nas API deverá retornar o retorno de acordo com as definições RESTFull. Por exemplo: se um módulo externo requisitar a url https://GISA/MIC/Prestador/1243 com método GET, caso exista o Prestador com Id 1243, então esse Prestador será recuperado.
- **Cenário 2 - Tolerância a Falha:** Caso algum módulo sofra falha no processamento de seus dados, esse cenário de falha não pode ocasionar falha no sistema. Por exemplo: durante o processamento o módulo de backend perde a conexão com o front end.
- **Cenário 3 - Auditabilidade:** O time de tecnologia da 'Boa Saúde' deve ter a possibilidade analítica sobre os cenários de falhas que possam ocorrer durante a produção dos módulos. Por exemplo: acesso as informações de erro caso ocorra uma tentativa de acesso com usuário ou senha inválidos.
- **Cenário 4 - Usabilidade**: Os usuários precisam de forma simples e rápida realizar operações no sistema, sendo o acesso feito por Desktop ou em Browser por equipamentos mobile, exemplo Smartphones e tablets.

- **Cenário 5 - Manutenibilidade:** A arquitetura baseada em aproveitamento de componentes como é o exemplo dos componentes que estão no pacote gisa.comum aumenta assim a manutenibilidade da solução.

5.3. Evidências da Avaliação

Cenário 1 - Interoperabilidade:

Atributo de Qualidade:	Interoperabilidade
Requisito de Qualidade:	O sistema deve se comunicar com outras tecnologias.
Preocupação: Interface de integração com legado ou futuras aplicações.	
O sistema deve ter como resposta a uma requisição uma saída de fácil leitura por outro componente. Tendo como exemplo uma requisição HTTPS na API de Prestador será retornado o prestador.	
Cenário(s):	
Cenário 1	
Ambiente:	
Sistema em operação normal	
Estímulo:	
É simulado o envio de uma requisição para o serviço REST do módulo MIC.	
Mecanismo:	
Criar um serviço REST para atender às requisições dos sistemas que queira se comunicar.	
Medida de resposta:	
Retornar os dados requisitados no formato JSON.	
Considerações sobre a arquitetura:	
Riscos:	Alguma instabilidade na rede pode deixar a conexão lenta ou mesmo a perda de pacotes.
Pontos de Sensibilidade:	Não há
Tradeoff:	Não há

Abaixo foi demonstrado o cenário acima citado, para esse cenário foi utilizado o Postman, ferramenta amplamente utilizado no mercado de desenvolvimento de software. Pode se notar que com a requisição foi solicitado a busca do Prestador

1243, com isso foi retornado o médico Marcos Luis França e seus devidos dados de cadastro.

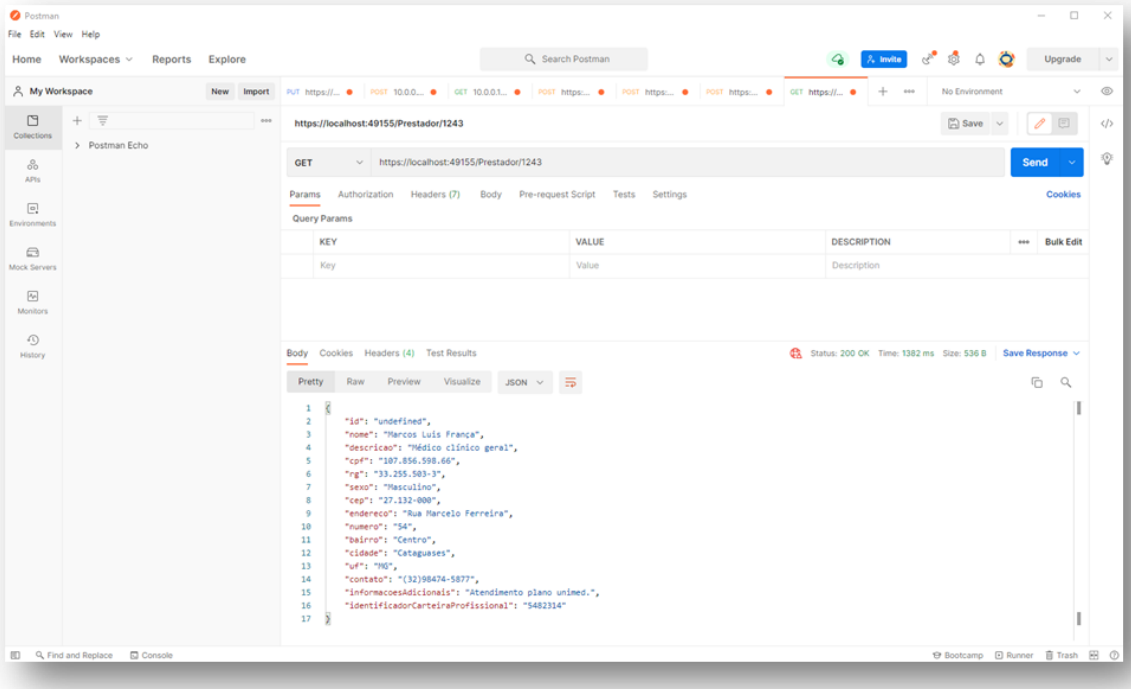

Figura 12 – Postmam fazendo Get API Prestados.

Cenário 2 - Tolerância a Falha

Atributo de Qualidade:	Tolerância a Falha
Requisito de Qualidade:	Em caso de uma tentativa de operação sem sucesso o sistema não pode ficar inoperante para o usuário.
Preocupação: Continuidade na utilização do sistema.	
Caso ocorra alguma falha interna na aplicação a mesma não pode se tornar inoperante.	
Cenário(s):	
Cenário 2	
Ambiente:	
Sistema em operação normal	
Estímulo:	
O usuário envia efetua uma ação no GISA do módulo MIC e este está inacessível ou inoperante.	

Mecanismo:	
Tratativas de conexão e exceções.	
Medida de resposta:	
A interface do usuário não “quebra” com exceções e sim mostra uma mensagem ao usuário.	
Considerações sobre a arquitetura:	
Riscos:	Caso o back end não tenha alta disponibilidade para atender todas as integrações e utilização do front end em casos de gargalo podem sofre com baixo desempenho.
Pontos de Sensibilidade:	É necessário alta escalabilidade do back end para atender volume extremamente grandes de requisições.
Tradeoff:	Não há

Abaixo foi demonstrado o cenário acima citado, para esse cenário foi desligado o back end, simulando assim indisponibilidade do serviço. Pode se notar que ao usuário realizar tentativa de operação é apresentado a mensagem abaixo. Permitindo uma nova tentativa de operação ou retorno a algum outro menu da

aplicação.

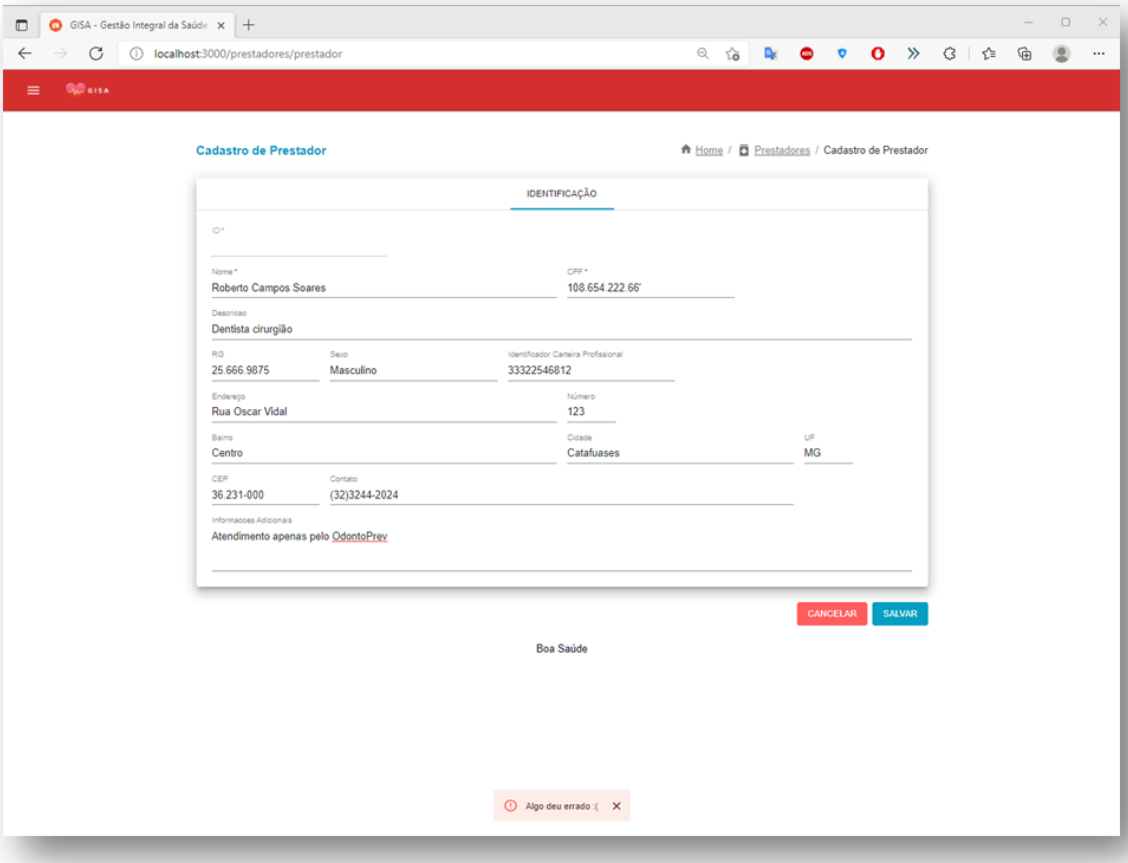

Figura 13 – GISA MIC – Tolerância a Falha.

Cenário 3 - Auditabilidade

Atributo de Qualidade:	Observabilidade
Requisito de Qualidade:	Em caso de uma falha interna na aplicação o time técnico deve ter a possibilidade de rastrear a mesma.
Preocupação: Saúde do ambiente.	
Caso ocorra alguma falha interna na aplicação a mesma deve guardar tais eventos afim de auditoria.	
Cenário(s):	
Cenário 3	
Ambiente:	
Sistema em operação normal	
Estímulo:	

Token de autenticação a base de dados e ao Event Bus inválido ou expirado.	
Mecanismo:	
Tratativa de eventos de logs.	
Medida de resposta:	
Identificação nos logs o erro informado.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Não há.
Tradeoff:	Atualmente é indicado utilização de ferramentas mais poderosas para gestão de e Observabilidade, como ELK, Gray Log, DataDog. Com isso esse é um ponto de melhoria na arquitetura.

Abaixo foi demonstrado o cenário acima citado, para esse cenário foi alterado a autenticação nos serviços de nuvem. Com a requisição do Swagger é possível notar o código correto do HTTP Response (Erro 500), e na imagem 12 é visto o dado de auditoria no log com o erro.

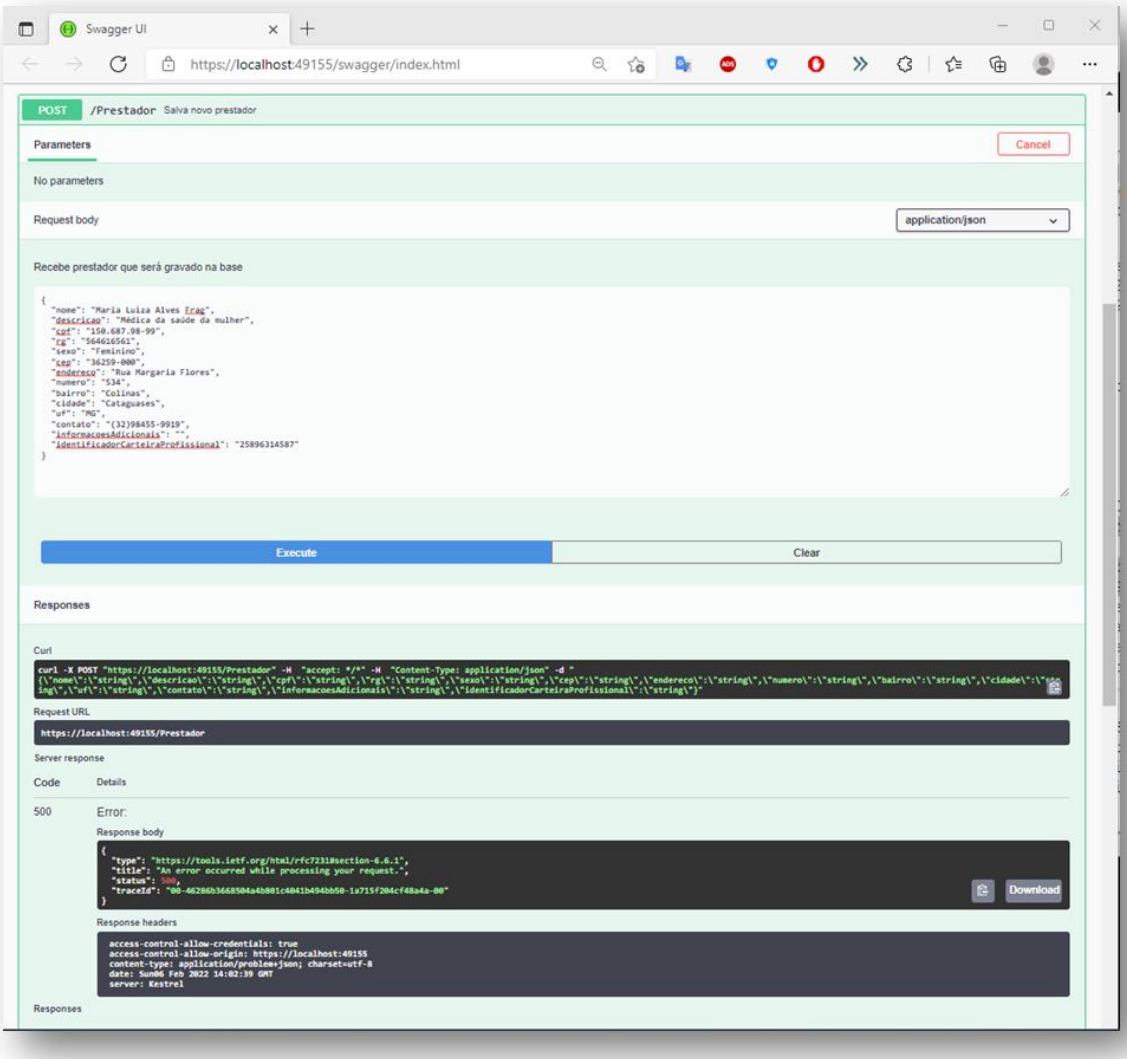

Figura 14 – Swagger Prestador - Tolerância a Falha.

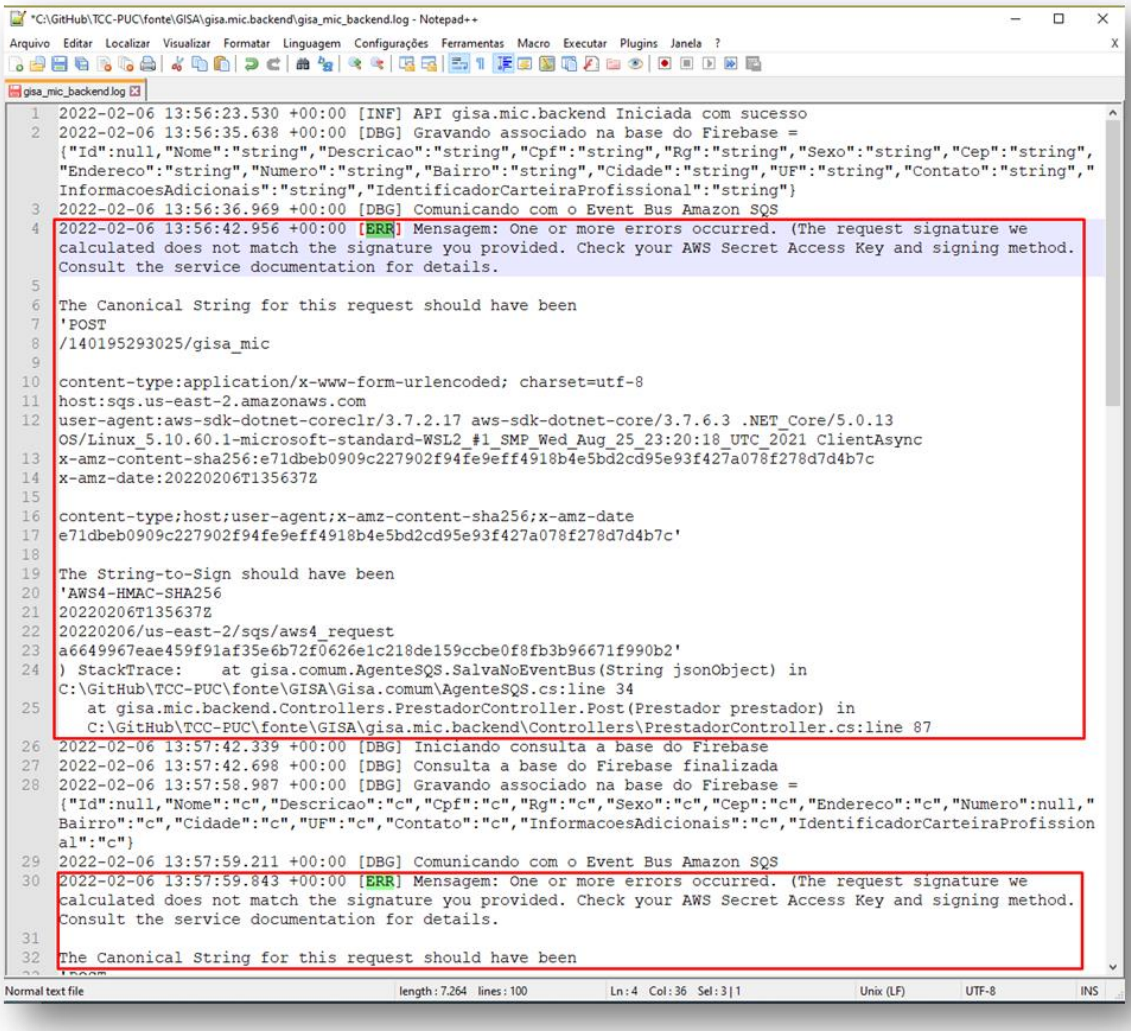

Figura 15 – Log de Auditoria.

Cenário 4 – Usabilidade

Atributo de Qualidade:	Usabilidade
Requisito de Qualidade:	O sistema deve permitir o acesso por equipamento móvel ou desktop.
Preocupação: Experiência do Usuário.	
Caso o usuário acesse por equipamento móvel ou desktop o sistema deve ser adaptar para garantir o funcionamento.	
Cenário(s):	
Cenário 4	
Ambiente:	
Sistema em operação normal	

Estímulo:	
Cadastro de Prestador por interface móvel ou desktop.	
Mecanismo:	
Template de frontend com interfaces responsivas.	
Medida de resposta:	
Capacidade de utilizar o sistema em mais de um device.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Não há.
Tradeoff:	Não há.

Abaixo foi demonstrado o cenário acima citado, para esse cenário foi feito a simulação pelo navegador com dois devices sendo simulados. Sendo assim Browser Desktop e iPhone 12 Pro.

Desktop

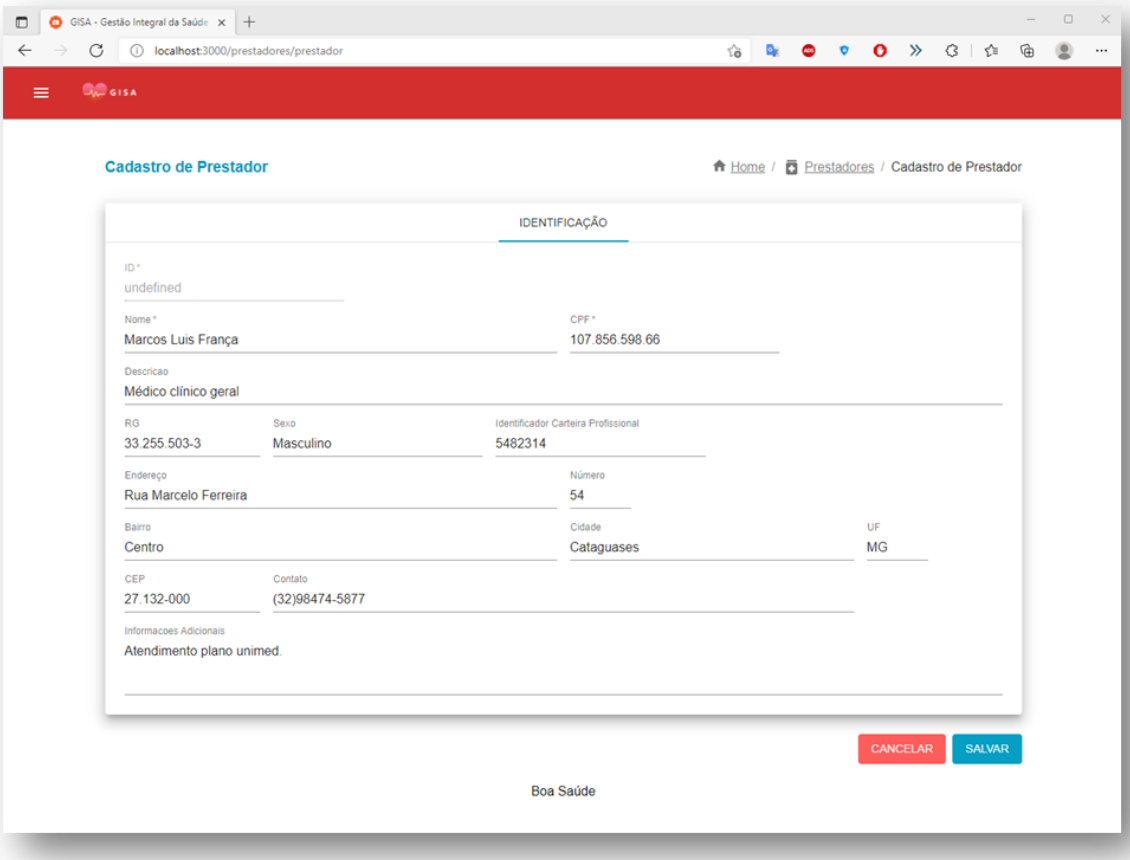

Figura 16 – GISA MIC – Acesso com Desktop.

iPhone 12 Pro

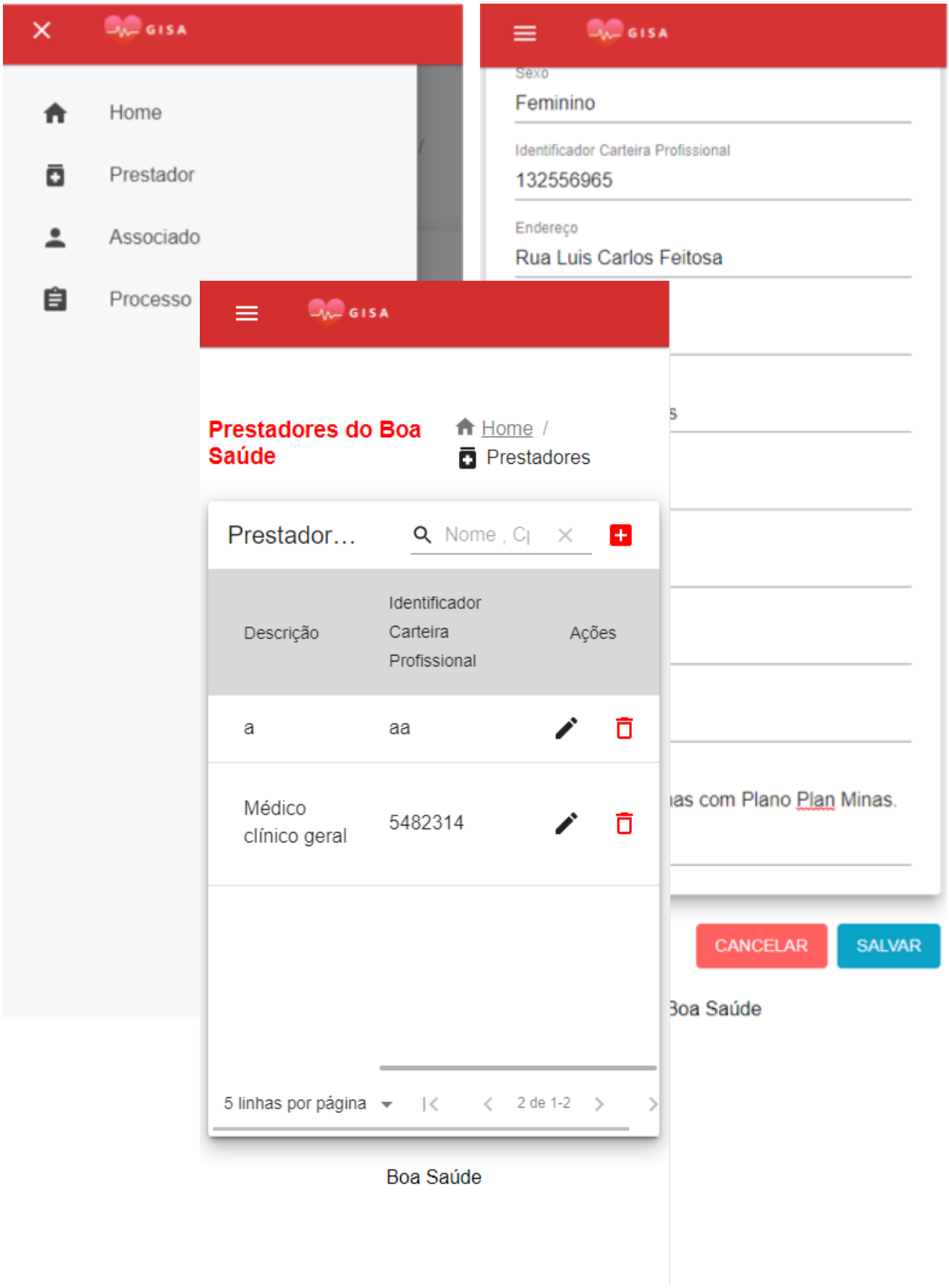

Figura 17 – GISA MIC – Acesso com iPhone 12 Pro.

Cenário 5 - Manutenibilidade

Atributo de Qualidade:	Manutenibilidade
Requisito de Qualidade:	O sistema deve permitir facilidade em manutenção.
Preocupação: Ciclo de vida da aplicação impactando diretamente em custo de manutenção.	
Os desenvolvedores devem ter facilidade em entendimento e também possibilidade de manutenção da solução GISA.	
Cenário(s):	
Cenário 5	
Ambiente:	
Análise estática do código fonte.	
Estímulo:	
Visão do código fonte.	
Mecanismo:	
Arquitetura baseado em reaproveitamento de código fonte.	
Medida de resposta:	
Entendimento do código fonte.	
Considerações sobre a arquitetura:	
Riscos:	Não há.
Pontos de Sensibilidade:	Conhecimento básicos em net core e C#.
Tradeoff:	Não há.

Para essa dimensão precisamos ter uma visão holística do projeto, com isso vale revisar os itens deste documento 3.3.1, 3.3.2 e 3.3.3. Abaixo tem o exemplo da classe AgenteFireBaseStorage.cs. Com essa única classe é possível realizar todas operações nas base do Firebase, independente de qual identidade que está sendo solicitada em uma gravação, o poder do “generic“ apoia na manutenibilidade da solução, uma vez que temos uma classe central para atender as demandas de banco.

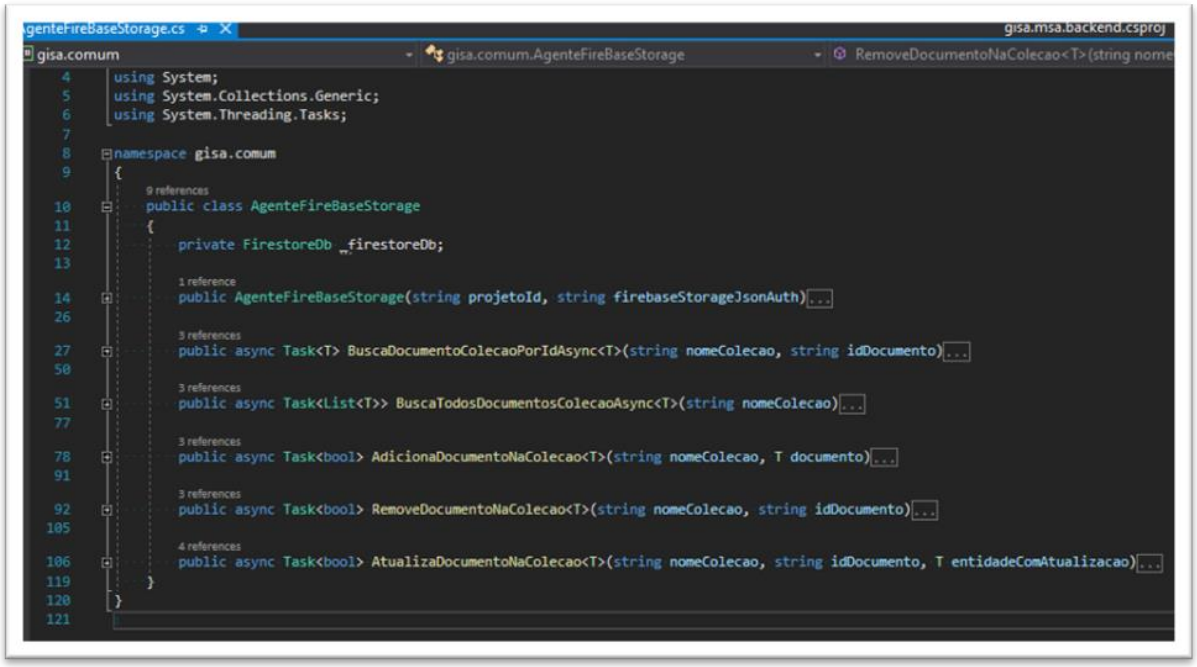

Figura 18 – Classe Acesso a base Firebase.

5.4. Resultados

O objetivo da avaliação do ponto de vista de arquitetura foi analisar os atributos do projeto. É de comum acordo para o time técnico que o projeto, atingiu os objetivos da POC, além disso o projeto foi pensando e arquitetado para evolução futura. Em linha geral o projeto contemplou de forma coerente o conhecimento apresentado durante o período de formação.

Os seguintes requisitos foram considerados nesta avaliação.

ID	Descrição	Testado	Homologado
RNF01	O sistema deve ser acessível nas plataformas *web* e móvel	SIM	SIM
RNF02	O sistema deve ser hospedado em nuvem híbrida, sendo a forma de hospedagem documentada;	SIM	SIM
RNF03	O sistema deve ser modular e implantável por módulos, de acordo com as prioridades e	SIM	SIM

	necessidades da empresa;		
RNF04	O sistema deve apresentar interface responsiva	SIM	SIM
RNF05	O sistema deve apresentar bom desempenho, não ultrapassando 3 segundos em suas operações	SIM	SIM
RNF06	O sistema deve apresentar boa manutenibilidade	SIM	SIM
RNF07	O sistema deve ser testável em todas as suas funcionalidades	NÃO	NÃO
RNF08	O sistema deve ser recuperável (resiliente) no caso da ocorrência de erro	SIM	SIM
RNF09	O sistema deve utilizar tecnologias de APIs para possibilitar suas integrações	SIM	SIM
RNF10	Deve ser desenvolvido utilizando recurso de configuração, com integração contínua	SIM	SIM

6. Conclusão

Nesta sessão será apresentado a avaliação geral do trabalho, apresentando assim os objetivos e desafios para o mesmo. Essa avaliação foi feita tendo em vista a área como Área de Arquitetura de Software.

Neste trabalho foi apresentado o caso da empresa fictícia ‘Boa Saúde’, que por ventura, vai em direção a realidade de muitas empresas atualmente. Os desafios conforme de costume foi o entendimento dos requisitos do projeto, além da preocupação de interface com o grande legado operante na empresa. As tecnologias escolhidas para a elaboração do projeto foram em direção ao conhecimento do time técnico além de serem ferramentas de mercado e com grande utilização atualmente. Essas características apoiam na redução do custo para o ciclo de vida da aplicação e desenvolvida do projeto. Na etapa de entrega de valor, foram entregues o projeto arquitetural e a POC com o código fonte devidamente versionado.

É de comum acordo para o time técnico que o projeto, atingiu os objetivos da POC, além disso o projeto foi pensando e arquitetado para evolução futura. Em linha geral o projeto contemplou de forma coerente o conhecimento apresentado durante o período de formação.

REFERÊNCIAS

• Documentação Swagger API. Disponível em: <https://swagger.io/specification/ > . Acesso em: 29 de Dezembro de 2021.

• Documentação .Net Core V5. Disponível em: <https://docs.microsoft.com/pt-br/aspnet/core/?view=aspnetcore-5.0/ > . Acesso em: 02 de Janeiro de 2022.

• Documentação Serilog. Disponível em: <https://serilog.net/>. Acesso em: 02 de Janeiro de 2022.

• Documentação Firebase Storage. Disponível em: <https://firebase.flutter.dev/docs/storage/overview/ > . Acesso em: 20 de Janeiro de 2022.

• Documentação React. Disponível em: <https://pt-br.reactjs.org/ > . Acesso em: 20 de Janeiro de 2022.

• Documentação Amazon SQS. Disponível em: < https://docs.aws.amazon.com/sqs/>. Acesso em: 03 de Fevereiro de 2022.

• Documentação Azure DevOps. Disponível em: < https://docs.microsoft.com/pt-br/azure/devops/?view=azure-devops/> . Acesso em: 05 de janeiro de 2022.

• Download Astah Community. Disponível em: < https://pt-br.reactjs.org/ > . Acesso em: 01 de Fevereiro de 2022.

• Download Postman. Disponível em: <https://www.postman.com/downloads/> . Acesso em: 20 de Dezembro de 2021.

APÊNDICES

• **Link do repositório:** https://github.com/altamirdiascassiano/TCC-PUC

• **Link Gravação da POC:** https://github.com/altamirdiascassiano/TCC-PUC/tree/main/arquivos/poc/videos

• **Link Imagens:** https://github.com/altamirdiascassiano/TCC-PUC/tree/main/arquivos/Imagens

• **Link dos Diagramas:** https://github.com/altamirdiascassiano/TCC-PUC/tree/main/arquivos/Diagramas

CHECKLIST PARA VALIDAÇÃO DOS ITENS E ARTEFATOS DO TRABALHO

Nº	Item a ser cumprido	Sim	Não	Não se aplica
Completeza do documento				
1	Todos os elementos iniciais do documento (capa, contracapa, resumo, sumário...) foram definidos?	X		
2	Os objetivos do trabalho (objetivos gerais e pelo menos três específicos) foram especificados?	X		
3	Os requisitos funcionais foram listados e priorizados?			
4	Os requisitos não funcionais foram listados e identificados usando o estilo estímulo-resposta?	X		
5	As restrições arquiteturais foram definidas?	X		
6	Os mecanismos arquiteturais foram identificados?	X		
7	Um diagrama de caso de uso foi apresentado junto com uma breve descrição de cada caso de uso?	X		
8	Um modelo de componentes e uma breve descrição de cada componente foi apresentada?	X		
9	Um modelo de implantação e uma breve descrição de cada elemento de hardware foi apresentada?			X
10	Prova de conceito: uma descrição da implementação foi feita?	X		
11	Prova de conceito: as tecnologias usadas foram listadas?	X		
12	Prova de conceito: os casos de uso e os requisitos não funcionais usados para validar a arquitetura foram listados?	X		
13	Prova de conceito: os detalhes da implementação dos casos de uso (telas, características, etc) foram apresentadas?	X		
14	Prova de conceito: foi feita a implantação da aplicação e indicado como foi feita e onde está disponível?	X		
15	As interfaces e/ou APIs foram descritas de acordo com um modelo padrão?	X		
16	Avaliação da arquitetura: foi feita uma breve descrição das características das abordagens da proposta arquitetural?	X		
17	Avaliação da arquitetura: Os atributos de qualidade e os cenários onde eles seriam validados foram apresentados?	X		
18	Avaliação da arquitetura: uma avaliação com as evidências dos testes foi apresentada?	X		
19	Os resultados e a conclusão foram apresentados?	X		
20	As referências bibliográficas foram listadas?	X		
21	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				
1	Todos os requisitos funcionais foram mapeados para casos de uso?	X		
2	Todos os casos de uso estão contemplados na lista de requisitos funcionais?	X		
3	Os requisitos não funcionais, mecanismos arquiteturais e restrições c arquiteturais estão coerentes com os modelos de componentes e implantação?	X		
4	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		
5	Os casos de uso e os requisitos não funcionais listados na implementação estão coerentes com o que foi listado nas seções anteriores?	X		
6	Os atributos de qualidade usados na avaliação estão coerentes com os requisitos não funcionais na sessão 3?	X		
7	Os cenários definidos estão no contexto dos casos de uso implementados?	X		
8	O apresentado no item resultado está coerente com o que foi mostrado no item avaliação?	X		