Breno Suguru Costa Tominaga
Sander Söhngen Rodrigues

Sistema Multissetorial de Saúde em Nuvem

São Paulo, SP

2024

Breno Suguru Costa Tominaga
Sander Söhngen Rodrigues

Sistema Multissetorial de Saúde em Nuvem

Trabalho de conclusão de curso apresentado ao Departamento de Engenharia de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo para obtenção do Título de Engenheiro.

Universidade de São Paulo – USP

Escola Politécnica

Departamento de Engenharia de Computação e Sistemas Digitais (PCS)

Orientador: Prof. Dr. Jorge Luis Risco Becerra

São Paulo, SP

2024

Este trabalho é dedicado aos visionários que desafiam os limites do possível, e aos que, com coragem e determinação, transformam ideias em legado.

Resumo

O mundo moderno é marcado por grandes problemas de saúde e obesidade devido aos atuais hábitos de consumo. Ao longo dos últimos 50 anos, a taxa de obesidade quase que triplicou e, em 2016, cerca de 1.9 bilhão de adultos estavam em situação de sobrepeso com 34% desses, 650 milhões, em cenário de obesidade (WHO, 2021). Com base nisso e no fato de que problemas de obesidade são de caráter preventivo, torna-se evidente a necessidade de medidas de controle e soluções de mitigação. Pensando nisso, o sistema multissetorial de saúde em nuvem é uma plataforma digital que visa englobar e centralizar variados serviços de saúde através de aplicativos dedicados e integrações em nuvem. Além disso, possui o objetivo principal de nortear a jornada de uma pessoa física por meio do suporte de agendamentos, exames e orientações com todos os dados integrados. A ideia é que com a utilização desse sistema, as consultas e os tratamentos sejam facilitados por ventura das tecnologias integradas aplicadas nesse cenário, deixando o fluxo de informações de um cliente mais transparente para todas as plataformas no sistema respeitando as diretrizes do LGPD.

Dando continuidade à descrição, o sistema funcionará como SaaS, Software as a Service, aglutinando, em primeira instância, soluções tecnológicas de serviços médicos, consultas com médicos, psicólogos, nutricionistas, treinadores físicos, e comerciais, mercados, lojas, academias, soluções essas que pavimentam a jornada de um paciente para um vida mais saudável sendo elas obesas ou não.

Palavras-chave: Web, Saúde, Nuvem, Obesidade, Sobrepeso, Interoperabilidade, Plataforma Digital.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

1.1 Motivação

Problemas com sobrepeso e obesidade nem sempre estiveram tão em evidência quanto nos dias modernos. Para se ter uma noção, em uma medição de 2016, 39% dos adultos acima de 18 anos estão em situação de sobrepeso e 13% são pessoas obesas. Mais preocupante do que isso é a situação dos jovens e adolescentes: em 2020, em torno de 39 milhões de crianças com idade abaixo de 5 anos eram obesas ou estavam em situação de sobrepeso e, em 2016, mais de 340 milhões de jovens de 5 a 18 anos estavam em situação análoga (WHO, 2021).

Considerando essa situação alarmante, medidas para solucionar esse cenário se tornam necessárias e, ao longo dos anos, várias alternativas surgiram. Os hospitais, por muitas vezes, possuem seus próprios sistemas para realizar o atendimento e tratamento de pacientes nessa situação, além dos planos de saúde possuírem também outras abordagens com outras soluções. No entanto, a grande maioria dessas alternativas se apresentam na forma de sistemas isolados e não possuem uma cadeia de dados única, fazendo com que um paciente tenha dificuldades na sua jornada. Portanto, além de uma interface humano-computador amigável, uma integração fluída de dados é de suma importância na manutenção da aderência do usuário ao plano programado existindo, então, a necessidade de uma maneira mais eficiente de invocar os vários dados desses serviços (PENG; BAI, 2019).

1.2 Objetivos

Considerando o problema geral de falta de integração entre serviços de saúde e a necessidade de melhorar a experiência do paciente, o principal objetivo do projeto é o desenvolvimento de um conjunto de aplicativos e de uma arquitetura em nuvem capaz de fornecer eficiência e transparência na integração de dados. Nesse sentido, o trabalho consiste basicamente no fornecimento de serviços de saúde digital, envolvendo sistemas de agendamento, teleconsultas, registro de históricos médicos, acompanhamento de tratamentos e integração com parceiros, sempre visando facilitar a circulação de informações entre diferentes setores e profissionais qualificados. Vale observar que, embora o sistema tenha sido motivado pela necessidade de enfrentar cenários complexos no cuidado à saúde, ele não se restringe a um único tipo de problema, mantendo-se flexível aos objetivos e condições de cada usuário. Em resumo, o resultado final será uma plataforma digital que centraliza e garante escalabilidade a todos os serviços envolvidos. Exemplificando um caso de uso, um paciente poderia marcar uma consulta com um especialista através de um dos

aplicativos, realizar os exames solicitados e, no momento do retorno, ter todos os dados armazenados e analisados pelo médico, permitindo também que outros setores integrados acessem essas informações. Esse resultado reflete o objetivo do trabalho: aumentar a fluidez e a eficiência no fluxo de um plano individual por meio da integração e compartilhamento seguro de dados. Por fim, todos os fluxos de comunicação no sistema devem respeitar as diretrizes da Lei Geral de Proteção de Dados, utilizando cifração, criptografia e rigorosos controles de acesso.

1.3 Justificativa

Devido a vários desafios apresentados pelo modelo convencional de cuidados de saúde, como, por exemplo, a falta de recursos, custos elevados e eficácia limitada, tem-se observado uma transição para uma abordagem de saúde mais focada no paciente, onde esse desempenha um papel mais participativo no seu próprio tratamento. No artigo "GetHealthyHarlem.org: Developing a Web Platform for Health Promotion and Wellness Driven by and for the Harlem Community"(KHAN JESSICA S. ANCKER, 2009), os autores discutem a criação de um site comunitário desenvolvido para permitir que a comunidade de Harlem construa uma base de conhecimento compartilhada sobre saúde e bem-estar. Este trabalho destacou os benefícios significativos de uma plataforma digital centrada na comunidade, evidenciando resultados positivos na promoção da saúde e bem-estar coletivo através da participação ativa dos membros da comunidade, indicando benefícios nessa mudança de paradigma. Porém, o artigo "A literature review of current technologies on health data integration for patient-centered health management"(PENG; BAI, 2019), aponta a dificuldade da integração de dados entre diferentes serviços de saúde, principalmente devido à falta de interoperabilidade, em níveis estruturais e semânticos. Ainda assim, a receptividade e o interesse crescentes dos profissionais de saúde evidenciado no artigo "Open SOA Health Web Platform for Mobile Medical Apps"(MEYER, 2014), incluindo médicos e enfermeiros, no uso de dispositivos móveis inteligentes e aplicativos para suporte ao tratamento de pacientes apontam para um caminho promissor, pois essa tendência sublinha a necessidade de plataformas que integrem dados de saúde de um paciente entre diferentes serviços de saúde.

Portanto, uma plataforma que integre diferentes serviços de saúde e permita uma comunicação clara entre os profissionais, mas que tenha o paciente como o centro e participante ativo garante que sejam explorados os aspectos positivos apontados anteriormente ao mesmo tempo em que busca resolver a carência de interoperabilidade dos diversos serviços.

1.4 Organização do Trabalho

Com o objetivo de desenvolver gradativamente a plataforma, o trabalho será organizado em etapas bem definidas estabelecendo fundação teórica para depois implementação, execução e testes de validação. No capítulo atual explicitamos as principais motivações para o projeto que sustentaram o desenvolvimento de todas as partes da monografia. Seguindo para os dois capítulos subsequentes, é feito um esclarecimento de conceitos e métodos que serão utilizados durante as fases posteriores. Finalmente, os capítulos finais envolvem especificações, o desenvolvimento propriamente dito, que inclui implementação e testes, e considerações finais do trabalho.

2 Aspectos Conceituais

2.1 Digital Platform

Plataformas digitais são como ecossistemas virtuais que facilitam as interações e transações entre diferentes usuários, incluindo consumidores, fornecedores e a própria plataforma. Elas são geralmente caracterizadas pela sua capacidade de escalar rapidamente, oferecendo um ambiente modular onde serviços e produtos podem ser facilmente integrados ou descontinuados (RYAZANOVA, 2021). Podemos resumir essas características da seguinte maneira:

1. **Escalabilidade**: Possuem a capacidade de crescer e se adaptar rapidamente a uma demanda crescente, mantendo ou melhorando sua eficiência operacional. Isso é essencial para atender ao dinamismo e à evolução contínua do mercado digital.

2. **Modularidade**: Oferecem um ambiente onde serviços e produtos podem ser facilmente integrados ou removidos. Essa flexibilidade permite que as plataformas se ajustem às mudanças nas necessidades dos usuários ou às inovações tecnológicas sem grandes transtornos.

3. **Integração de Processos de Informação**: As plataformas digitais facilitam a integração de diferentes processos de informação, o que é crucial para a criação de novos modelos de negócios e para a melhoria da comunicação interna e externa das empresas. Isso inclui desde a gestão de recursos humanos até a logística e o gerenciamento da cadeia de suprimentos.

4. **Catalisadores de Progresso Tecnológico**: Desempenham um papel fundamental no avanço científico e tecnológico, possibilitando o desenvolvimento de novas tecnologias e a disseminação rápida de inovações. Através da colaboração e compartilhamento de informações, elas potencializam a pesquisa e o desenvolvimento em diversos setores.

5. **Formação de Novos Modelos de Negócios**: As plataformas digitais são fundamentais na criação de modelos de negócios inovadores que podem transformar setores inteiros. Elas permitem que empresas experimentem, testem e implementem novas estratégias comerciais com maior agilidade e menor risco.

Considerando uma aplicação de saúde, essas características impulsionam a eficiência nos diversos tipos de serviços que a plataforma digital viria a oferecer e com isso é possível garantir seu bom funcionamento.

2.2 Microsserviço

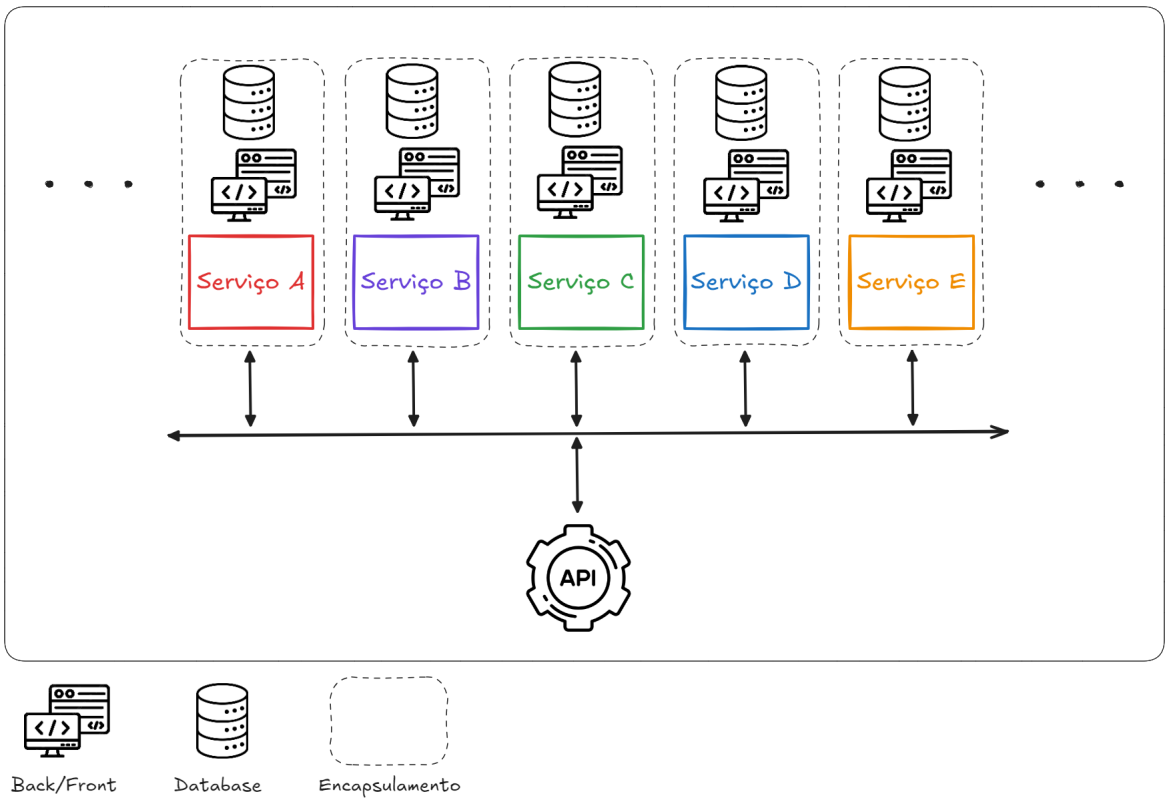
Dentro do desenvolvimento de sistemas atuais, sempre existe a discussão sobre qual será o método ou arquitetura mais eficiente dada certa aplicação. Entre esses métodos, temos a abordagem de síntese em microsserviços. A arquitetura de microsserviços é um estilo arquitetural que tem ganhado ampla adoção na engenharia de software, sendo especialmente adequada para sistemas distribuídos e aplicações baseadas em nuvem, como é o caso do sistema multissetorial de saúde proposto. Um microsserviço pode ser definido como uma unidade de software dentro de um sistema/plataforma que é independente de outras unidades, implementando uma funcionalidade de negócio específica e isolada (ALSHUQAYRAN N., 2016). Isso quer dizer que cada microsserviço é executado em seu próprio processo e se comunica com outros serviços por meio de interfaces bem definidas (BALALAIE A., 2016). Pode-se dizer que esse tipo de arquitetura contrasta com a arquitetura monolítica tradicional, em que todos os componentes do sistema são implementados em parcelas de um código-fonte unificado, dificultando a escalabilidade e a manutenção.

2.2.1 Arquitetura de Microsserviços

O que estamos chamando de arquitetura, nesse caso, é a abordagem de desenvolvimento baseado em uma série de princípios fundamentais que visam garantir a modularidade, independência e escalabilidade do sistema. Como mencionado anteriormente, cada microsserviço representa uma unidade coesa de funcionalidade de negócio, implementada de forma independente e implantada em um ambiente isolado. Esse isolamento é frequentemente garantido pelo uso de contêineres, como Docker, que encapsulam cada serviço junto com suas dependências, garantindo portabilidade entre diferentes ambientes de execução e, no caso do sistema proposto nesse trabalho, utiliza-se ferramentas de cloud para simular esse encapsulamento, como será abordado mais pra frente. Dada essa segmentação, assim, importante mencionar que a comunicação entre os microsserviços é mais comumente feita através de interfaces como REST APIs.

Tirando esses aspectos principais, existem também outros recursos/diferenciais que uma arquitetura de microsserviços pode ter. Práticas como o Design Orientado a Domínio (DDD) e o Bounded Context são frequentemente usadas para definir os limites de cada microsserviço, garantindo que cada serviço represente uma unidade coesa de lógica de negócio e tenha pouca (ou nenhuma) dependência de outros serviços. Esse isolamento pode facilitar a evolução independente de cada serviço, o que permite que diferentes equipes possam trabalhar de maneira autônoma em suas respectivas áreas (BALALAIE A., 2016).

Figura 1 – Arquitetura de microsserviços



2.2.2 Benefícios e Desafios dos Microsserviços

Existem várias maneiras de analisar quais as vantagens e desafios de uma arquitetura e microsserviços. Basicamente, podemos determinar as seguintes características:

- **Benefícios**:
  - **Escalabilidade independente**: Cada serviço pode alocar os recursos conforme a sua necessidade, não tendo dependência com os outros.
  - **Modularidade**: A arquitetura modular facilita a manutenção e evolução contínua do sistema, permitindo que diferentes serviços sejam desenvolvidos e implantados independentemente.
  - **Resiliência**: Falhas em um serviço não comprometem o funcionamento dos demais, permitindo que o sistema degrade de forma controlada
  - **Agilidade**: O ciclo de vida de desenvolvimento é mais rápido, com a possibilidade de entrega contínua e implantação frequente de novos recursos.
  - **Flexibilidade tecnológica**: Cada microsserviço pode ser implementado com diferentes tecnologias, facilitando a escolha da melhor ferramenta para cada tarefa.

- **Desafios**:
  - **Complexidade de comunicação**: A interação entre múltiplos serviços pode aumentar a latência e exigir soluções robustas para garantir a consistência dos dados.
  - **Segurança**: Cada microsserviço requer suas próprias políticas de segurança, aumentando a complexidade de autenticação e autorização.
  - **Gerenciamento de falhas**: A implementação de mecanismos de tolerância a falhas, como Circuit Breakers, adiciona complexidade à arquitetura.
  - **Monitoramento e rastreabilidade**: A observação de interações distribuídas entre serviços é desafiadora e exige ferramentas específicas para identificar gargalos e falhas.
  - **Coordenação entre equipes**: Mudanças nos contratos de serviço podem impactar outras partes do sistema, exigindo cuidado na gestão de APIs e versionamento.

2.3 Computação em Nuvem

A computação em nuvem é um paradigma de desenvolvimento definido pela disponibilização sob demanda de recursos computacionais como serviços na rede. Considerando isso, ela oferece uma série de características fundamentais que transformam o modo como as empresas e indivíduos acessam e utilizam recursos tecnológicos, especialmente em grandes empresas e setores que crescem conforme a demanda. Estas características incluem a escalabilidade, elasticidade, acesso on-demand e o pagamento conforme o uso. Como referência, segundo a National Institute of Standards and Technology (NIST), os cinco pilares da computação em nuvem incluem o autoatendimento sob demanda, o acesso amplo à rede, o agrupamento de recursos, a elasticidade rápida e a medição de serviços (TIKU, 2022). O ponto principal é que esses aspectos permitem que os usuários acessem recursos de computação, pesados ou não, sem precisar gerenciar a infraestrutura subjacente, ou seja, sem precisar construir ou montar uma estrutura física de computação no seu local. O fator interessante para o emprego desse tipo de computação, no nosso trabalho, é a obtenção de vários recursos e conveniências tecnológicas que facilitam a implementação de uma arquitetura de microsserviços. Podemos dizer que o sistema proposto apenas funcionará de maneira ótima se empregado esse tipo de tecnologia.

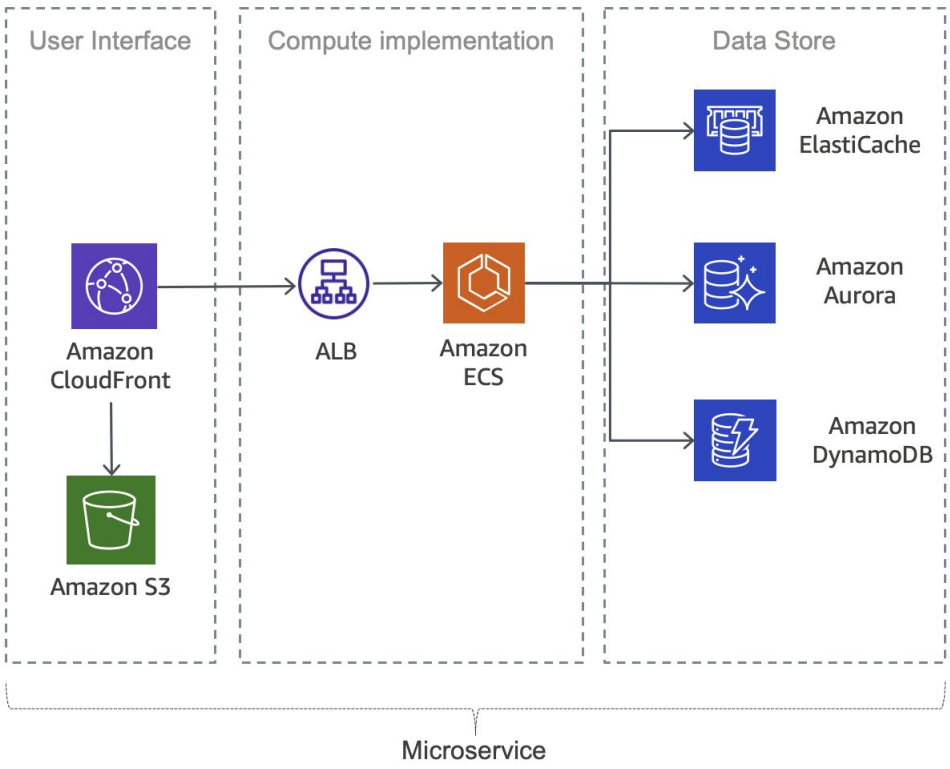
2.3.1 Amazon Web Services (AWS) e seus serviços

A Amazon Web Services, AWS, é um plataforma que provêm serviços em nuvem fornecendo uma grande quantidade de soluções de rede e de infraestrutura. Nela, as

características do que seriam pertencentes são bem exemplificadas através de seus diversos serviços. Por exemplo, o Amazon Elastic Compute Cloud (EC2) oferece escalabilidade ao permitir que outros serviços solicitantes (ou o próprio usuário) ajustem a capacidade computacional conforme a demanda de suas aplicações, pagando apenas pelo que utilizam. Outro exemplo é o Amazon Simple Storage Service (S3), que armazena dados com alta durabilidade e acessibilidade global, o que permite que empresas escalem suas operações sem grandes investimentos em hardware. Outro serviço interessante é o AWS Lambda que vai além de fornecer recursos, fornecendo computação sob demanda sem que o usuário precise provisionar servidores, ideal para automação de tarefas e no nosso caso para o provisionamento de microsserviços

Importante mencionar também que a computação em nuvem resolve desafios relacionados ao desenvolvimento de software em larga escala, como a necessidade de personalização para diferentes usuários, conforme analisado (GANG; BADARCH, 2023). E isso é um fator diferencial para plataformas como a AWS, com suas ofertas em SaaS, IaaS e PaaS. Todas essas alternativas facilitam a implementação de soluções customizadas, tornando-se essencial para empresas que precisam de soluções flexíveis e escaláveis para grandes bases de clientes.

Figura 2 – Exemplo de uma implementação de microsserviços com serviços da AWS (Simple microservices architecture on AWS)



2.4 Lei Geral de Proteção de Dados (LGPD)

A Lei Geral de Proteção de Dados, conhecida como LGPD, é uma norma que foi implementada no Brasil com a Lei nº 13.709/2018 para regulamentar o uso de dados pessoais no contexto digital e físico, devido ao avanço do ambiente virtual na vida das pessoas. Esta lei surge em resposta à crescente digitalização e coleta de dados, garantindo ao titular dos dados maior controle sobre seu uso e promovendo a autodeterminação informativa, que nada mais que é que o direito de cada indivíduo decidir como suas informações são coletadas e utilizadas (seja em qualquer ambiente digital que se tenha no Brasil). A LGPD estrutura-se em princípios como transparência, segurança e necessidade, com o consentimento do usuário sendo um pilar central para o processamento de dados pessoais. Podemos concluir que o objetivo é criar uma estrutura onde o uso de dados pessoais seja seguro e respeite os direitos fundamentais dos indivíduos, incluindo sua privacidade e liberdade.

Tabela 1 – Informações Gerais sobre a Lei Geral de Proteção de Dados (LGPD)

Aspecto	Descrição
Fundamentos	A LGPD é baseada em direitos fundamentais, como privacidade, liberdade e autodeterminação informativa.
Principais Artigos	Artigos que tratam dos princípios de proteção, direitos dos titulares, responsabilidades dos controladores e operadores de dados, e sanções aplicáveis.
Direitos dos Titulares	Inclui o direito de acesso, retificação, exclusão, portabilidade dos dados, entre outros.
Consentimento	O tratamento de dados exige consentimento expresso e informado do titular, salvo exceções específicas da lei.
Sanções	Penalidades em caso de infração incluem advertências, multas de até 2% do faturamento (limitado a R$50 milhões por infração), e bloqueio de dados.
Agente Regulador	A Autoridade Nacional de Proteção de Dados (ANPD) é responsável pela fiscalização e orientação sobre a LGPD.

Fonte: Produzido pelos autores com base na Lei nº 13.709/2018

Anotações: A ANPD é o órgão responsável por monitorar e garantir o cumprimento da lei. É válido observar que, no caso do sistema proposto, haverá uma prova de conceito de que essa lei é respeitada.

2.4.1 Impacto da LGPD em Sistemas de Saúde

A implementação da LGPD nos sistemas de saúde é particularmente crítica, pois envolve a proteção de dados sensíveis, como informações de saúde que podem expor detalhes pessoais ou vulnerabilidades dos indivíduos. No nosso caso, além das informações básicas que cada usuário deve preencher, seja eles pacientes ou profissionais, vão existir dados de diagnósticos e problemas particulares que precisam ser tratados no sistema para

garantir uma integridade e privacidade ao usuário. Considerando que esses dados são fundamentais para o funcionamento dos serviços de saúde propostos, torna-se evidente que seu uso requer uma atenção especial devido ao risco de violação de privacidade e possíveis consequências negativas para o paciente. A única maneira de garantir que esses princípios não sejam violados é construir um sistema que obedeça a LGPD que, por sua vez, obriga instituições a adotarem mecanismos de consentimento e proteção de dados rigorosos para evitar acessos não autorizados e garantir que o uso dessas informações seja feito de forma ética e segura, respeitando a confidencialidade e privacidade dos pacientes (LUGATI; ALMEIDA, 2020).

2.4.2 Criptografia e Segurança de Dados Sensíveis

Dada a quantidade de informações e fluxos que o sistema vai residir, é evidente a necessidade de uma infraestrutura criptográfica por trás dessas informações. A proteção de dados sensíveis em conformidade com a LGPD implica no uso de tecnologias com criptografia para garantir que informações pessoais sejam protegidas de acessos não autorizados e mais imporante para impedir exposição de pacientes. A criptografia é utilizada como uma ferramenta de desenvolvimento para que seja possível a construção de microsserviços já pensados na integração desses dados de maneira categórica. Isso quer dizer que é possível garantir o cumprimento da LGPD ao utilizar diversos serviços de computação em nuvem já pensados nessa cifração dos dados. Com isso, de maneira prática, o sistema de saúde deve assegurar não somente a criptografia desses dados como também o seu transporte seguro e o seu uso adequado, sempre respeitando as leis propostas. As tecnologias em si vão ser discutidas mais pra frente.

3 Método do Trabalho

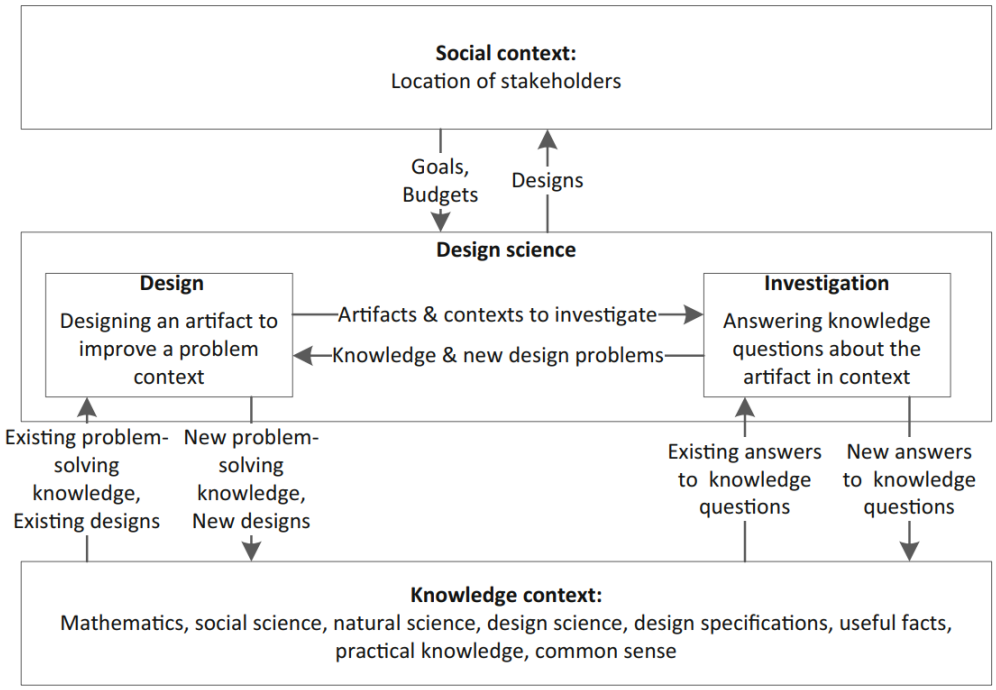
3.1 Metologia de Desenvolvimento

3.1.1 Abordagem Design Science

Para modelar e implementar o sistema, a metologia seguida será a proposta pelo Design Science (WIERINGA, 2014). Essa metologia enfatiza a investigação e o estudo de artefatos em contextos específicos a fim de destacar a importância da compreensão tanto do objeto de estudo quanto das atividades principais, como a modelagem e a investigação. Em resumo, será necessário achar um artefato, uma solução, que concorde com o nosso problema através da investigação.

No nosso caso, para entender o problema do design, o contexto da obesidade e da falta de integração nos sistemas de saúde atuais nos levará a desenvolver um artefato adequado para esse cenário. Vale observar que existe distinção, nessa metodologia, entre esses problemas de design e questões de conhecimento, ressaltando a necessidade, assim, de uma maior investigação.

Figura 3 – Exemplo de framework de Design Science



Fonte: (WIERINGA, 2014)

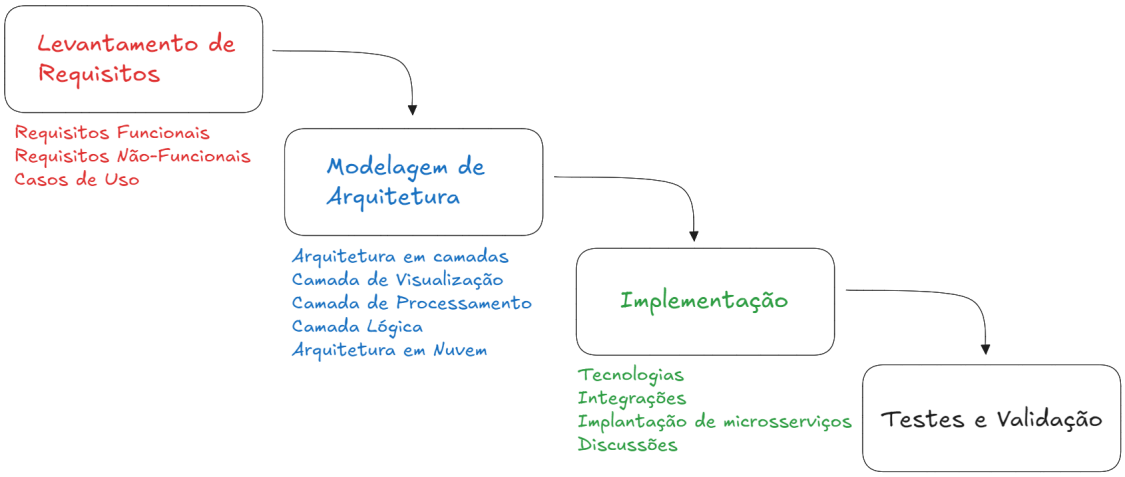
3.1.2 Justificativa da Escolha Metodológica

De maneira objetiva, a escolha do Design Science como metodologia se justifica por suas diversas vantagens na construção de sistemas tecnológicos complexos, como o proposto nesse trabalho. Essa abordagem permite uma análise profunda tanto do objeto de estudo quanto de seu contexto, promovendo um entendimento holístico que é essencial para soluções que envolvem múltiplas camadas e integração tecnológica. Justamente por isso que se alinha de maneira eficaz ao desenvolvimento de um sistema que incorpora tecnologias em nuvem e arquitetura de microsserviços, enfatizando a criação de artefatos que não só resolvem problemas específicos, mas também atendem a requisitos de escalabilidade, interoperabilidade e adaptabilidade necessários para a arquitetura. Além disso, temos o respaldo do nosso professor orientador que auxiliou no estudo e nas adequadas referências aos métodos propostos.

3.2 Fases de Desenvolvimento

O desenvolvimento do sistema seguirá uma abordagem estruturada, composta por quatro fases principais, cada uma com objetivos específicos para assegurar a qualidade e a robustez do sistema final. Isso é necessário para garantir o correto desenvolvimento das arquiteturas e visões propostas no trabalho

Figura 4 – Esquema das fases de desenvolvimento do trabalho



3.2.1 Fase de Levantamento de Requisitos

Na primeira fase, o foco será na coleta e análise detalhada dos requisitos funcionais e não funcionais do sistema. Para a obtenção dessas informações, além de revisões de sistemas similares e necessidades do mercado para obter uma visão completa do que o

sistema deve atender, deve haver um levantamento de características do sistema que servirá de base para a modelagem e implementação subsequente. Dessa elaboração, considerando as características propostas de microsserviços e tecnologias em nuvem, é possível formular os requisitos a partir dos diversos casos de uso que o sistema de negócio pode ter.

3.2.2 Fase de Modelagem da Arquitetura

Após a definição dos requisitos, a segunda fase basicamente envolve a modelagem da arquitetura do sistema, onde será delineada a separação do sistema em camadas estruturais para enfim ser feita uma arquitetura em nuvem. Nessa fase é feita a escolha de tecnologias, a definição de padrões de comunicação e a criação de diagramas arquiteturais que garantam que o sistema seja escalável, modular e capaz de atender às demandas de desempenho e segurança. Importante mencionar que a arquitetura desenvolvida para o sistema deve considerar a LGPD e permitir que os dados fluam de maneira criptografada.

3.2.3 Fase de Implementação

Na fase de implementação, o desenvolvimento do sistema será iniciado seguindo a arquitetura e os requisitos estabelecidos. A implementação seguirá práticas de programação para facilitar a integração de componentes e a escalabilidade futura. Cada microsserviço será desenvolvido obedecendo os príncipios de encapsulamento e com isso todo o sistema deverá ter fluxos de integração. É nessa fase também que será registrado em documento discussões e questões a respeito da implementação.

3.2.4 Fase de Testes e Validação

Finalmente, a fase de testes e validação garantirá que o sistema implementado em nuvem até então funcione conforme o esperado. Os testes vão abranger desde questões de funcionalidade até verificações de quanto o sistema respeita os requisitos e as arquiteturas estabelecidos anteriormente. Dentre todos os requisitos, um que ganha destaque é a escalabilidade e por isso é importante avaliar o quanto o sistema em si consegue integrar outros serviços.

4 Especificação

4.1 Serviços da plataforma

Os seguintes serviços foram descobertos analisando as necessidades dos usuários e identificando lacunas nos sistemas atuais.

Tabela 2 – Lista de Serviços Disponíveis

Nº	Serviço	Descrição
1	Serviço de Gestão de Usuários	Gerencia o cadastro, autenticação, autorização e perfis dos usuários da plataforma.
2	Serviço de Agendamento de Consultas e Exames	Permite o agendamento de consultas médicas e associados.
3	Serviço de Realização de Consultas (Telemedicina)	Facilita consultas online por meio de videochamadas e troca de mensagens seguras entre pacientes e profissionais.
4	Serviço de Gestão de Documentos	Armazena, criptografa e gerencia o acesso aos documentos.
5	Serviço de Avaliação e Acompanhamento do Paciente	Profissionais registram avaliações, planos de tratamento e acompanham o progresso dos pacientes ao longo do tempo.
6	Serviço de Compras	Gerencia compras realizadas na plataforma, incluindo pedidos a mercados, farmácias, lojas esportivas e academias.
7	Serviço de Pagamentos e Transações	Processa pagamentos, gerencia transações financeiras e integra métodos de pagamento como cartão de crédito e Pix.
8	Serviço de Integração com Hospitais	Hospitais podem se cadastrar, gerenciar disponibilidade e receber solicitações de exames.
9	Serviço de Integração com Mercados	Mercados se cadastram, gerenciam seus produtos e recebem pedidos de compras dos usuários.
10	Serviço de Integração com Farmácias	Farmácias gerenciam seu catálogo de medicamentos e processam pedidos de medicamentos dos pacientes.
11	Serviço de Integração com Lojas Esportivas	Lojas esportivas gerenciam produtos e recebem pedidos de equipamentos e vestuário esportivo.
12	Serviço de Integração com Academias	Academias gerenciam planos, serviços e recebem matrículas de pacientes interessados.
13	Serviço de Notificações	Envia notificações em tempo real aos usuários sobre operações do sistema.
14	Serviço de Comunicação em Tempo Real	Gerencia a comunicação em tempo real entre os usuários e profissionais, facilitando a comunicação em vídeo entre os usuários do sistema.
15	Serviço de Autenticação e Autorização	Garante a segurança do sistema, gerenciando tokens de acesso, permissões dos usuários e políticas de segurança.
16	Serviço de Gateway de API	Atua como ponto de entrada unificado para serviços.

Fonte: Elaborado com base nos serviços disponíveis no sistema

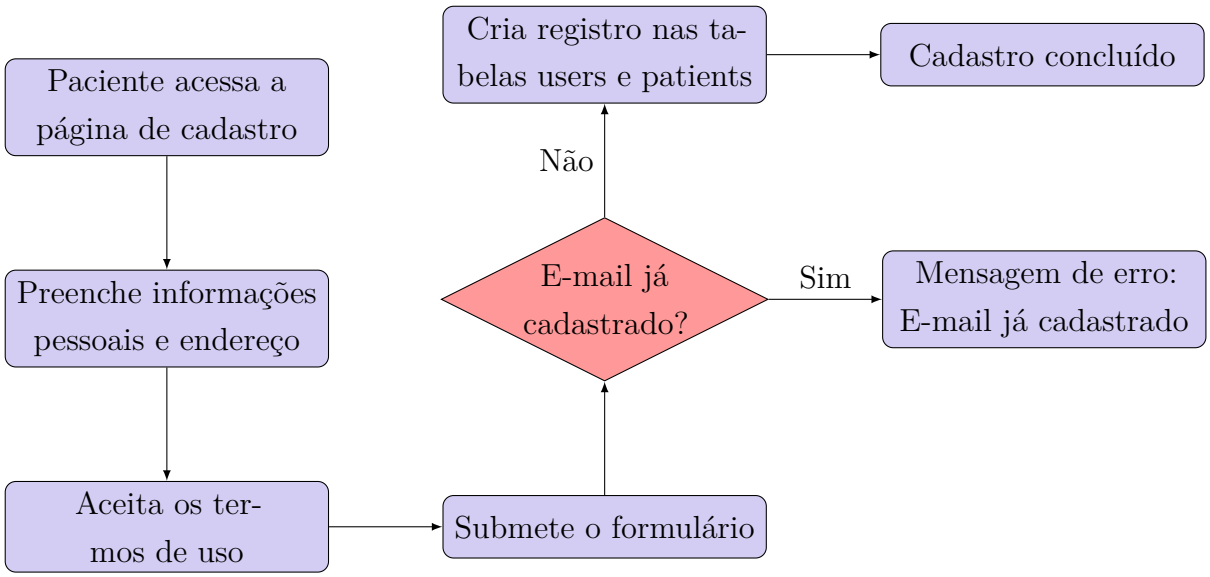
4.2 Casos de Uso

Como forma de visualizar a jornada de um usuário no sistema proposto, a definição de casos de uso é útil ao estabelecer uma jornada clara de como o sistema sendo um SaaS, *Software as a Service*, na visão de negócio mostrada anteriormente é utilizado na prática. Considerando isso, podemos documentar casos de uso, primeiro para estabelecer uma visão mais concreta do negócio e segundo para fixar pontos de partida visando a definição dos requisitos, da arquitetura e da implementação. Assim, vamos mostrar a seguir apenas alguns casos de uso fundamentais para o sistema.

4.2.1 Cadastro e Gerenciamento de Usuários

Parte principal do sistema é o cadastro e gerenciamento dos usuários. Para essas funcionalidades fundamentais, existem os seguintes casos:

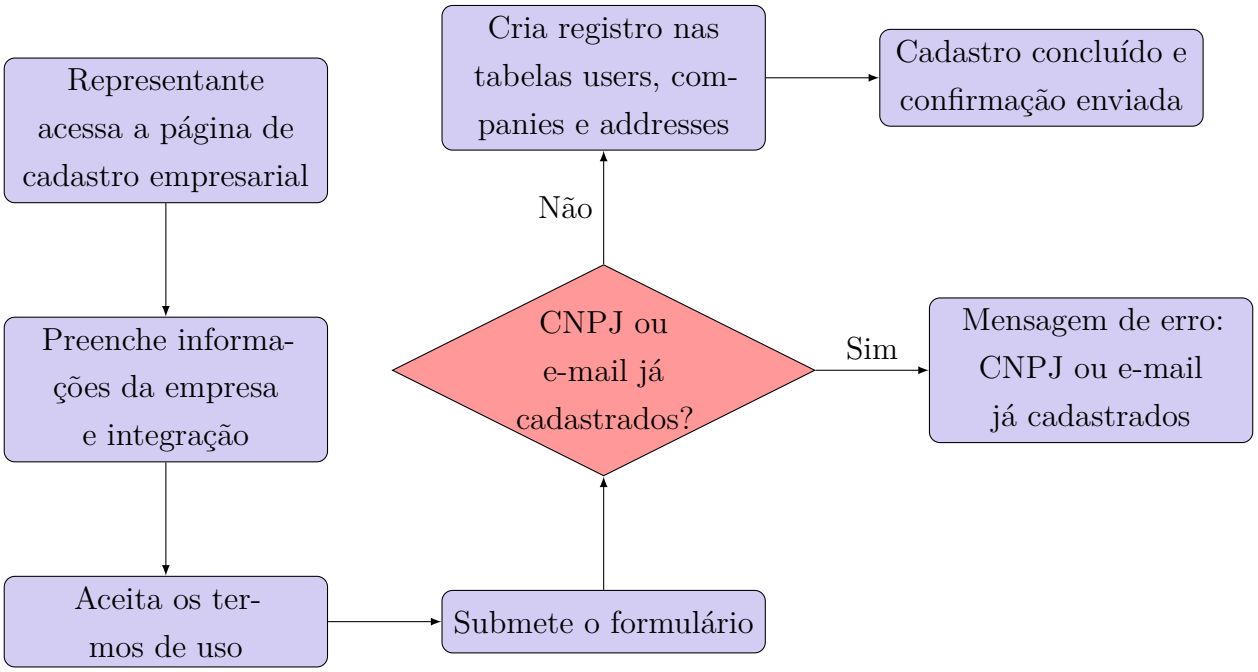
- **Cadastro de Paciente**
  - **Ator Primário**: Paciente
  - **Descrição**: O paciente acessa a página de cadastro da plataforma. Ele deve preencher informações pessoais, como nome, e-mail, senha, data de nascimento, peso, altura, gênero e restrições alimentares. Além disso, fornece um endereço residencial completo. Após isso, é necessário aceitar os termos de uso e a política de privacidade para submeter o formulário. O sistema verifica se o e-mail já está cadastrado. Caso não esteja, um novo registro é criado nas tabelas *users* e *patients*.
  - **Pré-condições**: O paciente não deve ter um cadastro prévio com o mesmo e-mail.
  - **Pós-condições**: O paciente está registrado na plataforma e pode fazer login.



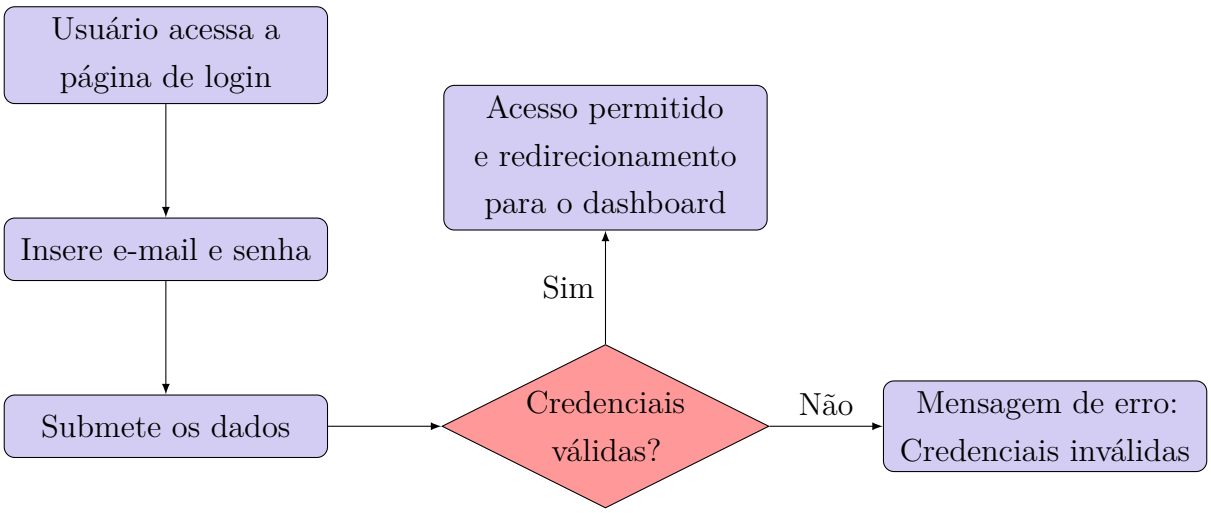
- **Cadastro de Profissional de Saúde**
  - **Ator Primário**: Profissional de Saúde
  - **Descrição**: O profissional de saúde acessa a página de cadastro na plataforma. Ele deve preencher informações pessoais e profissionais, como nome, e-mail, senha, profissão, credenciais (CRM, CRP, etc.), preço das consultas, biografia e especialidades. Após isso, submete o formulário para validação. O sistema verifica se o e-mail já está cadastrado. Caso não esteja, um novo registro é criado nas tabelas *users* e *professionals*. O profissional recebe uma confirmação de cadastro por e-mail.
  - **Pré-condições**: O profissional deve fornecer credenciais válidas.
  - **Pós-condições**: O profissional está registrado na plataforma, mas não está ativo até que seja validado.

- **Cadastro de Empresa Parceira**
  - **Ator Primário**: Representante da Empresa
  - **Descrição**: O representante de uma empresa parceira acessa a página de cadastro empresarial na plataforma. Ele preenche informações da empresa, como nome, e-mail, senha, CNPJ, tipo de empresa e endereço comercial. Também fornece detalhes para integração, como API Key e endpoints para produtos e pedidos. Após aceitar os termos de uso e a política de privacidade, submete o formulário. O sistema verifica se o CNPJ ou e-mail já estão cadastrados. Caso não estejam, cria registros nas tabelas *users*, *companies* e *addresses*. A empresa recebe uma confirmação de cadastro por e-mail.
  - **Pré-condições**: A empresa deve ser válida e não estar previamente cadastrada.

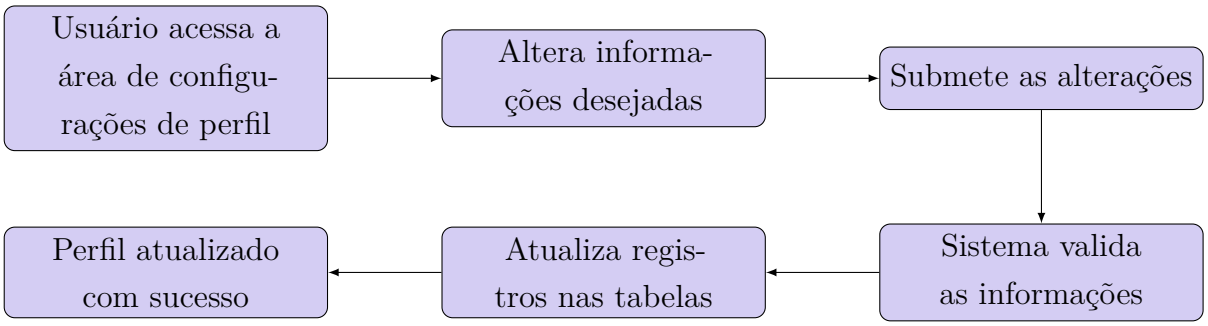
– **Pós-condições**: A empresa está registrada na plataforma e pode integrar-se via API.



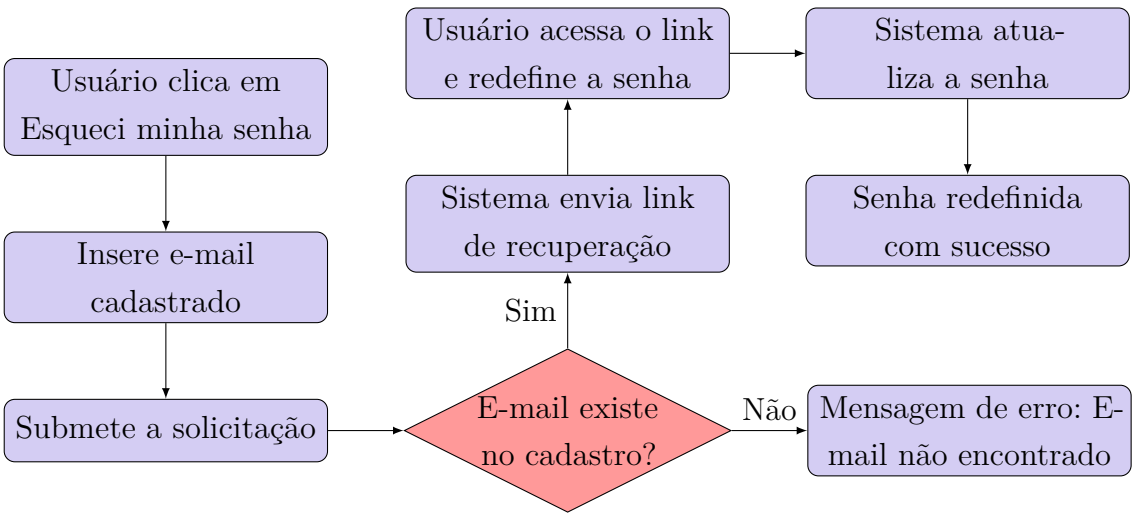
- **Login de Usuário**
  - **Ator Primário**: Todos os Usuários
  - **Descrição**: O usuário acessa a página de login da plataforma. Ele insere seu e-mail e senha e submete os dados para autenticação. O sistema verifica se as credenciais fornecidas são válidas. Caso sejam, o usuário é autenticado e redirecionado para seu dashboard específico. Caso contrário, uma mensagem de erro é exibida informando que as credenciais são inválidas.
  - **Pré-condições**: O usuário deve estar previamente cadastrado na plataforma.
  - **Pós-condições**: O usuário tem acesso às funcionalidades correspondentes ao seu perfil.



- **Atualização de Perfil**
  - **Ator Primário**: Todos os Usuários
  - **Descrição**: O usuário acessa a área de configurações de perfil da plataforma. Ele pode alterar as informações desejadas, como dados pessoais, endereço e outras preferências. Após realizar as alterações, o usuário submete os dados. O sistema valida as novas informações e atualiza os registros correspondentes nas tabelas de usuários.
  - **Pré-condições**: O usuário deve estar autenticado e logado na plataforma.
  - **Pós-condições**: As informações do perfil do usuário são atualizadas com sucesso.



- **Recuperação de Senha**
  - **Ator Primário**: Todos os Usuários
  - **Descrição**: O usuário clica na opção "Esqueci minha senha"na página de login. Em seguida, insere seu e-mail cadastrado e submete a solicitação. O sistema verifica se o e-mail existe no cadastro. Se existir, um link de recuperação de senha é enviado para o e-mail do usuário. O usuário acessa o link, redefine a senha e o sistema atualiza o registro com a nova senha.
  - **Pré-condições**: O usuário deve ter um e-mail válido cadastrado no sistema.
  - **Pós-condições**: O usuário pode acessar o sistema com a nova senha.



Esses casos fundamentais básicos do sistema podem ser resumidos na seguinte tabela:

Tabela 3 – Casos de Uso para Cadastro e Gerenciamento de Usuários

Caso de Uso	**Ator Primário**	**Descrição Breve**
Cadastro de Paciente	Paciente	O paciente se cadastra na plataforma fornecendo informações pessoais, endereço e aceita os termos de uso. O sistema verifica a existência do e-mail antes de criar o registro.
Cadastro de Profissional de Saúde	Profissional de Saúde	O profissional de saúde cadastra-se na plataforma, inserindo informações pessoais e profissionais, como credenciais e especialidades. A verificação de e-mail é feita antes do cadastro ser confirmado.
Cadastro de Empresa Parceira	Representante da Empresa	Um representante cadastra uma empresa parceira, inserindo informações da empresa, CNPJ, e dados de integração. Após validação, registros são criados nas tabelas de usuários e empresas.
Login de Usuário	Todos os Usuários	O usuário faz login na plataforma fornecendo e-mail e senha. O sistema valida as credenciais e redireciona para o dashboard.
Atualização de Perfil	Todos os Usuários	O usuário acessa as configurações para atualizar suas informações de perfil. As alterações são submetidas e registradas no sistema.
Recuperação de Senha	Todos os Usuários	O usuário solicita recuperação de senha. O sistema envia um link por e-mail para redefinir a senha, que é atualizada após a confirmação.

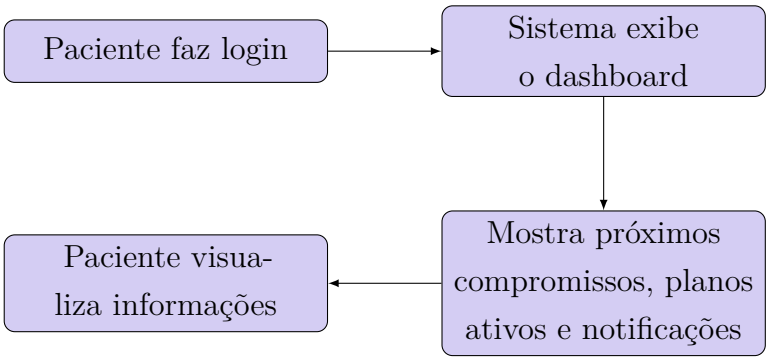
Fonte: Elaborado com base nos requisitos do sistema de gerenciamento de usuários

4.2.2 Jornada do paciente

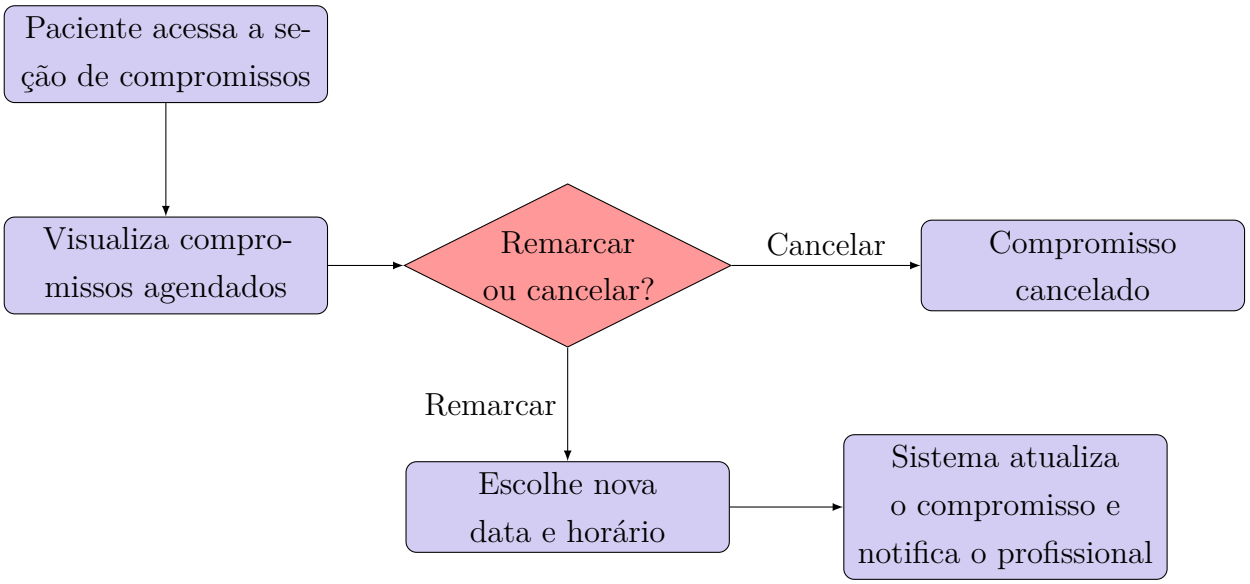
A jornada do paciente nada mais é que os passos necessários para o paciente completar uma atividade dentro da plataforma. Pode-se dizer que que essa jornada em específico engloba praticamente todos os microsserviços a serem implementados de forma a evidenciar a importância da integração entre todos os serviços. Com isso, julgamos importante, entre todas as outras possibilidades de casos de usos, listar a do paciente em específico pois ele é o público-alvo principal no qual o sistema vai atender.

- **Visualizar Dashboard**
  - **Ator Primário**: Paciente

  - **Descrição**: O paciente faz login na plataforma. O sistema exibe o dashboard com informações como próximos compromissos, planos de tratamento ativos e notificações recentes.
  - **Pré-condições**: O paciente deve estar logado.
  - **Pós-condições**: O paciente tem uma visão geral de suas informações.

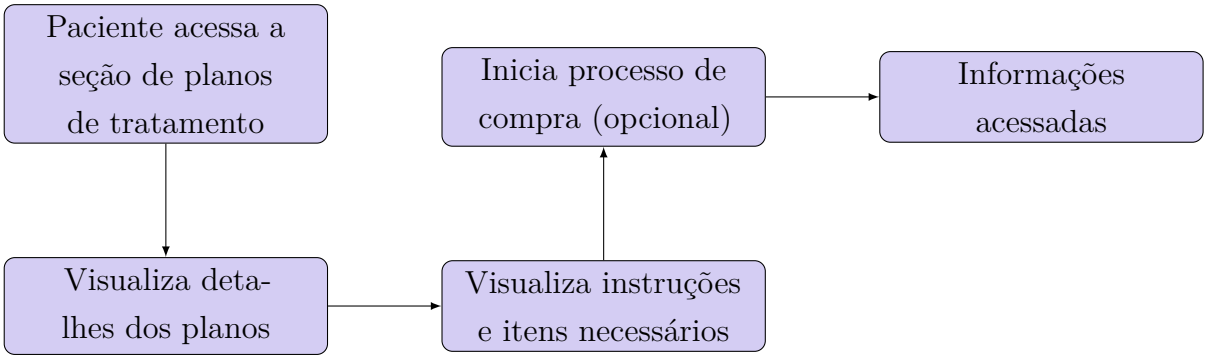


- **Gerenciar Compromissos**
  - **Ator Primário**: Paciente
  - **Descrição**: O paciente acessa a seção de compromissos e visualiza consultas e exames agendados. Pode optar por remarcar ou cancelar um compromisso. Se remarcar, escolhe uma nova data e horário disponíveis. O sistema atualiza o agendamento e notifica o profissional.
  - **Pré-condições**: O paciente deve ter compromissos agendados.
  - **Pós-condições**: O compromisso é atualizado conforme a ação do paciente.

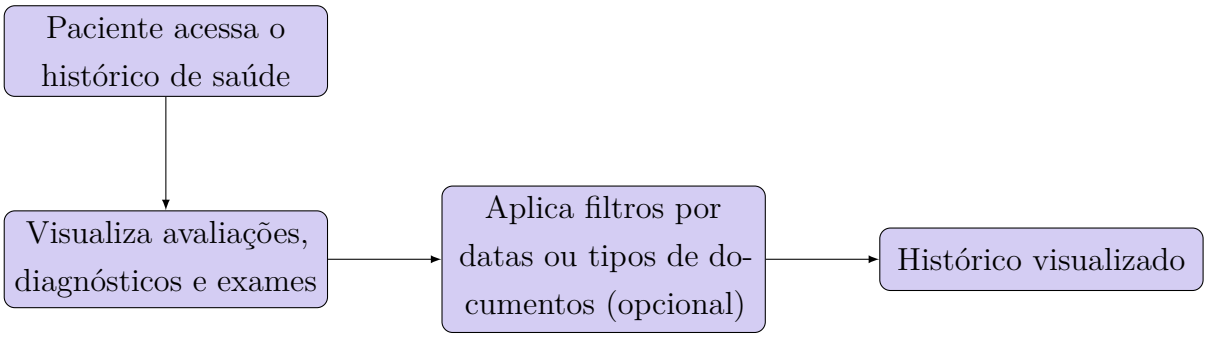


- **Acessar Planos de Tratamento**
  - **Ator Primário**: Paciente

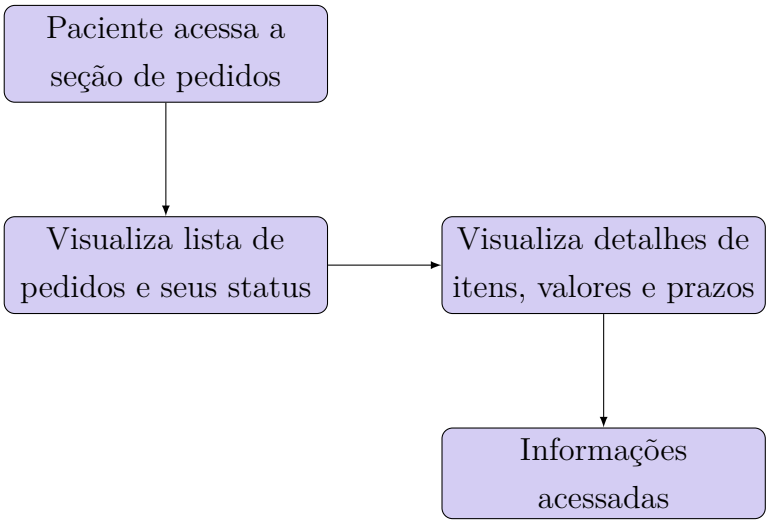
  - **Descrição**: O paciente acessa a seção de planos de tratamento e visualiza detalhes dos planos ativos e concluídos. Pode visualizar instruções e itens necessários para o tratamento, além de iniciar o processo de compra dos itens, se aplicável.
  - **Pré-condições**: O paciente deve ter planos de tratamento cadastrados.
  - **Pós-condições**: O paciente obtém informações para seguir seu tratamento.



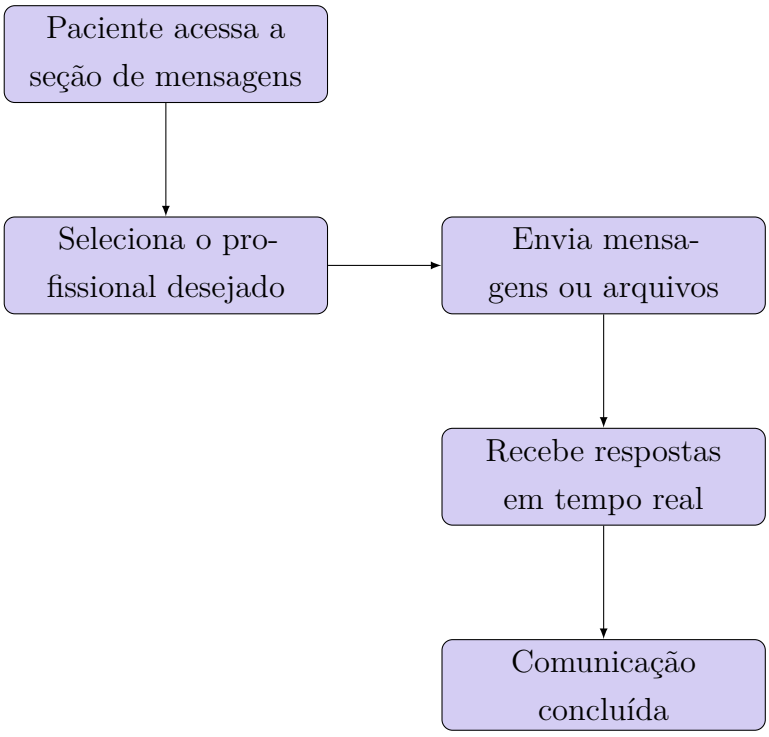
- **Visualizar Histórico de Saúde**
  - **Ator Primário**: Paciente
  - **Descrição**: O paciente acessa o histórico de saúde para visualizar avaliações, diagnósticos e resultados de exames. Pode aplicar filtros por datas ou tipos de documentos para encontrar informações específicas.
  - **Pré-condições**: O paciente deve ter registros de saúde armazenados.
  - **Pós-condições**: O paciente tem acesso completo ao seu histórico médico.



- **Acompanhar Pedidos**
  - **Ator Primário**: Paciente
  - **Descrição**: O paciente acessa a seção de pedidos e visualiza a lista de pedidos realizados, com detalhes como status, itens, valores e prazos de entrega.
  - **Pré-condições**: O paciente deve ter realizado pedidos.
  - **Pós-condições**: O paciente está informado sobre o andamento de suas compras.



- **Comunicar-se com Profissionais**
  - **Ator Primário**: Paciente
  - **Descrição**: O paciente acessa a seção de mensagens, seleciona o profissional com quem deseja se comunicar, envia mensagens ou arquivos e pode receber respostas em tempo real.
  - **Pré-condições**: O paciente deve ter tido um atendimento recente com o profissional.
  - **Pós-condições**: A comunicação é estabelecida para esclarecimentos ou orientações.



Diante desses casos do paciente, assim, podemos sintetizar o seguinte resumo:

Tabela 4 – Casos de Uso para o Dashboard do Paciente

Caso de Uso	Ator Primário	Descrição Breve
Visualizar Dashboard	Paciente	O paciente faz login na plataforma. O sistema exibe o dashboard com informações como próximos compromissos, planos de tratamento ativos e notificações recentes.
Gerenciar Compromissos	Paciente	O paciente acessa a seção de compromissos, visualiza consultas e exames agendados e pode optar por remarcar ou cancelar um compromisso. O sistema atualiza o agendamento e notifica o profissional.
Acessar Planos de Tratamento	Paciente	O paciente acessa a seção de planos de tratamento, visualiza detalhes dos planos ativos e concluídos, e pode iniciar o processo de compra dos itens necessários para o tratamento.
Visualizar Histórico de Saúde	Paciente	O paciente acessa o histórico de saúde para visualizar avaliações, diagnósticos e resultados de exames, podendo aplicar filtros por datas ou tipos de documentos.
Acompanhar Pedidos	Paciente	O paciente acessa a seção de pedidos e visualiza a lista de pedidos realizados, incluindo detalhes como status, itens, valores e prazos de entrega.
Comunicar-se com Profissionais	Paciente	O paciente acessa a seção de mensagens, seleciona o profissional com quem deseja se comunicar, envia mensagens ou arquivos e pode receber respostas em tempo real.

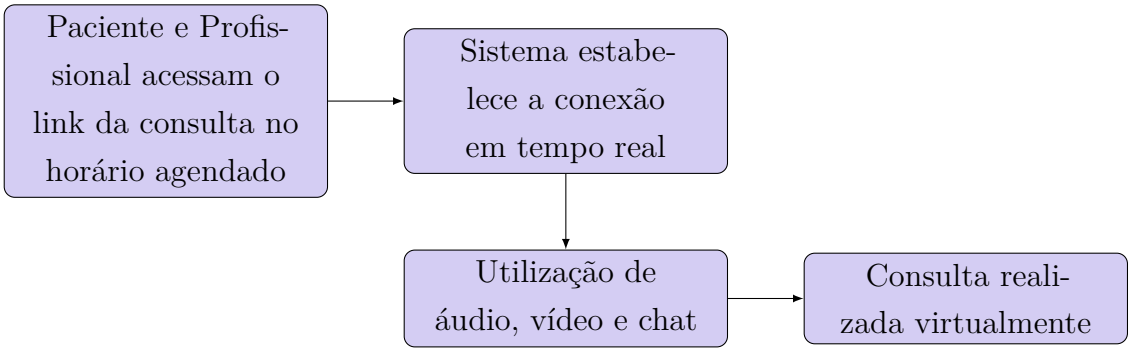
Fonte: Elaborado com base nos requisitos do sistema de dashboard do paciente

4.2.3 Telemedicina

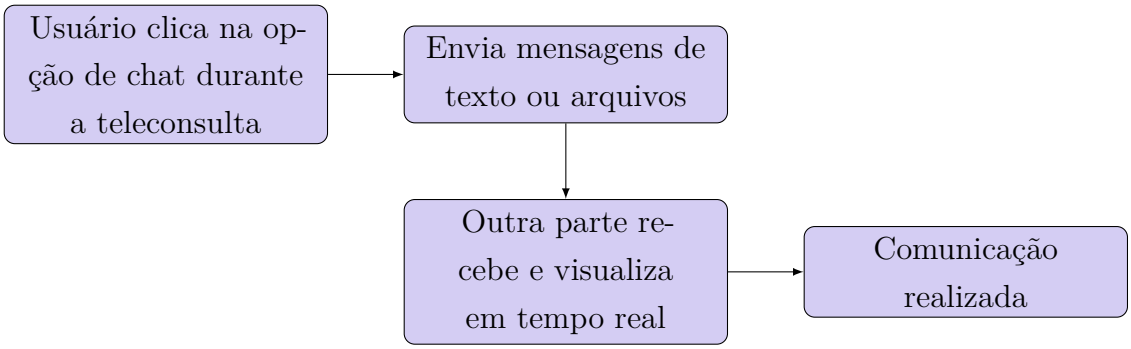
Como um exemplo final de caso de uso possível para o sistema, a telemedicina é uma das features primárias do sistema em que será possível a consulta com o profissional de saúde de maneira integrada, por meio de comunicação online. Ela é caracterizada principalmente pela troca de informação entre paciente e o profissional de maneira a simular uma consulta médica presencial.

- **Iniciar Teleconsulta**
  - **Ator Primário**: Paciente e Profissional
  - **Descrição**: No horário agendado, tanto o paciente quanto o profissional acessam o link da consulta. O sistema estabelece uma conexão em tempo real, permitindo que ambos utilizem áudio, vídeo e chat durante a sessão.
  - **Pré-condições**: Deve haver suporte técnico adequado e conexão estável.

– **Pós-condições**: A consulta é realizada virtualmente com qualidade.



- **Enviar Mensagens e Arquivos Durante a Consulta**
  - **Ator Primário**: Paciente e Profissional
  - **Descrição**: Durante a teleconsulta, o usuário clica na opção de chat, podendo enviar mensagens de texto ou arquivos, como exames e prescrições. A outra parte recebe e visualiza o conteúdo em tempo real.
  - **Pré-condições**: A teleconsulta deve estar em andamento.
  - **Pós-condições**: A comunicação é enriquecida com informações adicionais.



Finalmente, de maneira análoga aos outros casos, podemos sintetizar a seguinte tabela:

Tabela 5 – Casos de Uso para Consultas Online (Telemedicina)

Caso de Uso	**Ator Primário**	**Descrição Breve**
Iniciar Teleconsulta	Paciente e Profissional	No horário agendado, ambos acessam o link da consulta, e o sistema estabelece a conexão em tempo real, permitindo o uso de áudio, vídeo e chat.
Enviar Mensagens e Arquivos Durante a Consulta	Paciente e Profissional	Durante a teleconsulta, o usuário clica na opção de chat, enviando mensagens ou arquivos, como exames, com visualização em tempo real pela outra parte.

Fonte: Elaborado com base nos requisitos do sistema de telemedicina

4.3 Requisitos Funcionais (RF)

Os requisitos funcionais de um sistema descrevem as funcionalidades específicas que ele deve oferecer para atender às necessidades dos usuários. A partir da análise dos casos de uso, identificamos um conjunto de requisitos funcionais e, para cada um deles, associamos os serviços necessários à sua implementação. Essa associação foi feita consultando a Tabela 2, que lista todos os serviços disponíveis, suas finalidades e responsabilidades. Na prática, cada requisito funcional é associado ao serviço que melhor o atende.

4.3.1 Cadastro e Gerenciamento de Usuários

Tabela 6 – Requisitos Funcionais do Cadastro e Gerenciamento de usuários

Requisito Funcional	**Serviços Afetados**
Cadastro de pacientes, profissionais de saúde e empresas parceiras	1, 8, 9, 10, 11, 12
Coleta de informações pessoais dos pacientes	1, 4
Associação de endereço residencial ao perfil do paciente	1
Acesso ao histórico médico, planos de tratamento e avaliações	4, 5
Coleta de informações profissionais dos profissionais de saúde	1, 4
Verificação e validação das credenciais dos profissionais	1, 15
Gerenciamento de disponibilidade para agendamentos	1, 2
Acesso a avaliações detalhadas e simplificadas dos pacientes	5, 4
Cadastro de informações empresariais das empresas parceiras	1, 8, 9, 10, 11, 12
Associação de endereço comercial ao perfil da empresa	1, 8, 9, 10, 11, 12

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.2 Dashboards

Tabela 7 – Requisitos Funcionais do Dashboard do Paciente

Paciente	
Requisito Funcional	**Serviços Afetados**
Dashboard com visão consolidada de compromissos, planos e notificações	1, 2, 4, 5, 13
Gerenciamento de pedidos e acompanhamento de status	1, 6, 7, 9, 10, 11, 12, 13
Comunicação direta com profissionais via chat	1, 14, 15

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

Tabela 8 – Requisitos Funcionais do Dashboard dos Profissionais

Profissionais	
Requisito Funcional	**Serviços Afetados**
Dashboard para gerenciamento de agenda	1, 2, 13
Acesso ao histórico de atendimentos, avaliações e planos dos pacientes	1, 4, 5
Realização de teleconsultas e emissão de prescrições e documentos	1, 3, 4, 13

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

Tabela 9 – Requisitos Funcionais do Dashboard das Empresas

Empresas	
Requisito Funcional	**Serviços Afetados**
Dashboard para gerenciamento de pedidos recebidos	1, 6, 9, 10, 11, 12, 13
Acesso a informações de integração via API	1, 15, 16
Atualização de informações empresariais e catálogo de produtos	1, 4, 9, 10, 11, 12

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.3 Agendamento de Consultas e Exames

Tabela 10 – Requisitos Funcionais de Agendamento de Consultas e Exames

Requisito Funcional	**Serviços Afetados**
Visualização da disponibilidade dos profissionais de saúde	1, 2
Agendamento de consultas médicas, nutricionais, psicológicas e de educação física	1, 2, 13
Agendamento de exames médicos	1, 2, 8, 13
Calendário interativo para gerenciamento de compromissos	1, 2
Notificações de lembrete para consultas e exames	1, 2, 13
Remarcação e cancelamento de consultas	1, 2, 13
Integração com hospitais e laboratórios parceiros	1, 2, 8
Recebimento e armazenamento de resultados de exames	1, 4, 8

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.4 Consultas Online (Telemedicina)

Um fluxo essencial do sistema é o da consulta online. Representa funcionalidades de comunicação online.

Tabela 11 – Requisitos Funcionais de Consultas Online (Telemedicina)

Requisito Funcional	**Serviços Afetados**
Realização de consultas online	1, 3, 14, 15
Chat em tempo real para troca de mensagens e arquivos	1, 3, 4, 14, 15
Atendimento às normas de telemedicina e sigilo médico	1, 3, 4, 15

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.5 Integração com Parceiros e Facilitação de Compras

Funcionalidades de comunicação via API. Também engloba features de recomendações de produtos.

Tabela 12 – Requisitos Funcionais de Integração com Parceiros e Facilitação de Compras

Requisito Funcional	**Serviços Afetados**
Consulta às APIs dos parceiros	1, 6, 7, 9, 10, 11, 12, 16
Criação de opções de pedido	1, 6, 9, 10, 11, 12
Visualização e comparação de opções de pedido	1, 6, 9, 10, 11, 12
Finalização simplificada de compras	1, 6, 7
Integração com diversos tipos de estabelecimentos	1, 6, 9, 10, 11, 12, 16

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.6 Planos de Tratamento e Acompanhamento

Aqui evidencia-se, praticamente, as funcionalidades de criação e manipulação de bancos de dados com planos de tratamento e acompanhamentos.

Tabela 13 – Requisitos Funcionais de Planos de Tratamento e Acompanhamento

Requisito Funcional	**Serviços Afetados**
Criação de planos de tratamento personalizados	1, 4, 5
Registro detalhado de itens necessários para cada plano	1, 4, 5, 6
Geração de avaliações detalhadas e simplificadas	1, 4, 5
Controle de compartilhamento de avaliações e planos	1, 4, 5, 15

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.7 Comunicação e Mensageria

Basicamente todo e qualquer operação de mensageria presente na plataforma digital.

Tabela 14 – Requisitos Funcionais de Comunicação e Mensageria

Requisito Funcional	**Serviços Afetados**
Chat integrado para comunicação	1, 14, 15
Envio de mensagens e arquivos	1, 4, 14, 15

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.3.8 Pagamentos e Transações Financeiras

Funcionalidades de integração de processos/fluxos de pagamento no sistema de saúde, seja ele na hora que realizar consultas ou compras ocorrentes dentro do ecossistema da plataforma.

Tabela 15 – Requisitos Funcionais de Pagamentos e Transações Financeiras

Requisito Funcional	**Serviços Afetados**
Processamento seguro de pagamentos	1, 7, 15
Registro de detalhes dos pedidos e pagamentos	1, 6, 7
Acompanhamento do status dos pedidos	1, 6, 7, 13
Acesso ao histórico de transações e emissão de recibos	1, 4, 7

Fonte: Elaborado com base nos requisitos do sistema e nos serviços disponíveis

4.4 Requisitos Não Funcionais

Para finalizar a parte de especificação de requisitos, os requisitos não-funcionais são aqueles que representam as características intrínsecas ao ambiente de produção do sistema, englobando aspectos que não se referem ao funcionamento prático do sistema. Podemos dizer que são "features"auxiliares e necessárias para o bom funcionamento do produto ao garantir as apropriadas capacidades de computação necessárias.

Tabela 16 – Requisitos Não Funcionais

Requisito Não Funcional	**Descrição**
Desempenho	Tempo de resposta inferior a 3 segundos, Suporte a 10.000 usuários simultâneos, Otimização de consultas às APIs de parceiros
Segurança	Implementação de criptografia de dados, Controle de acesso com autenticação segura, Conformidade com a LGPD, Uso de protocolos seguros (HTTPS), Armazenamento criptografado de senhas, Logs de auditoria para monitoramento
Usabilidade	Interface intuitiva e fácil de navegar, Design responsivo
Escalabilidade	Ajuste dinâmico de recursos conforme a carga, suporte a aumento de usuários sem impacto no desempenho, arquitetura modular permitindo adicionar ou remover serviços, bem como integrar novos componentes em nuvem para escalabilidade horizontal e vertical sob demanda
Manutenibilidade	Boas práticas de programação e padrões de codificação, Documentação abrangente, Atualizações e correções sem interrupção prolongada
Portabilidade	Compatibilidade com diferentes sistemas operacionais e navegadores
Legal e Regulatório	Conformidade com LGPD e legislações de saúde
Interoperabilidade	Integração com outros sistemas de saúde, APIs seguindo padrões RESTful e formato JSON
Experiência do Usuário	Feedback claro e imediato às ações do usuário, Minimização de entradas repetitivas

Fonte: Elaborado com base nos requisitos não funcionais do sistema

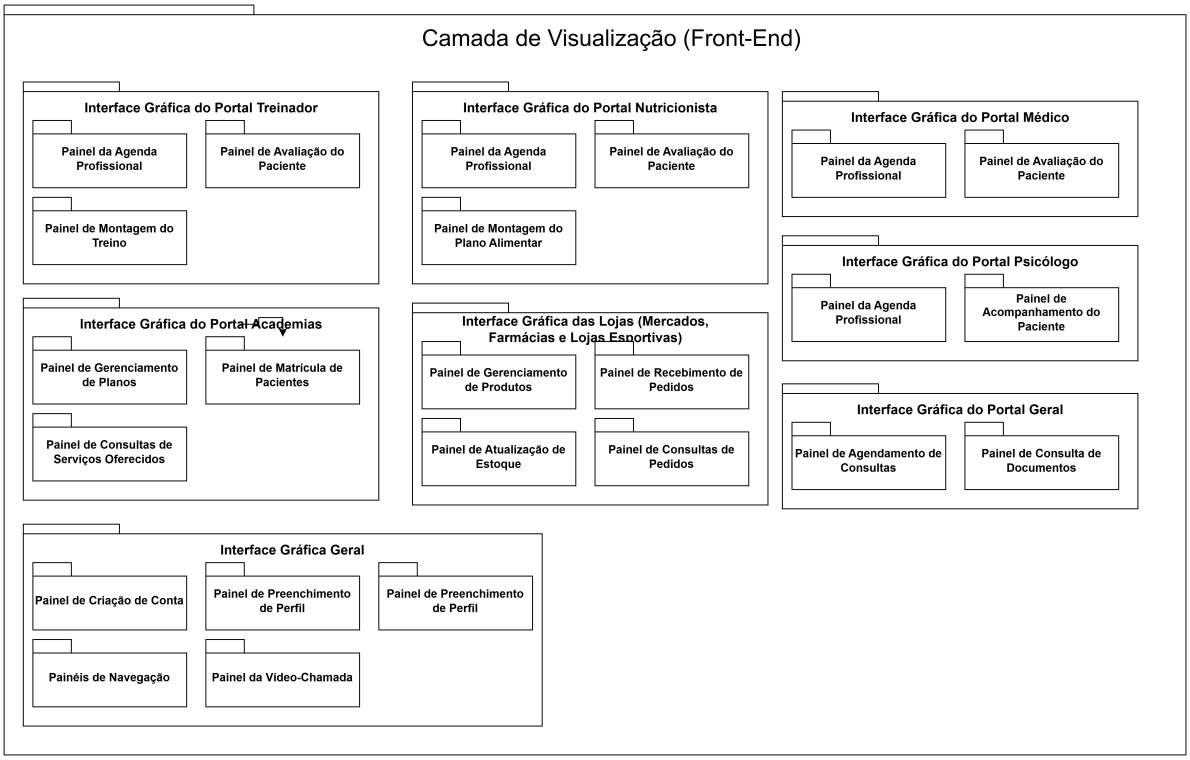
4.5 Arquitetura

Visando explorar as capacidades de uma estrutura não-monolítica, a arquitetura proposta para a plataforma de saúde é modular e baseada em microsserviços, organizada em camadas interconectadas que incluem a Camada de Apresentação, a Camada de Lógica de Negócios, a Camada de Persistência de Dados e a Camada de Comunicação. Para organizar os fluxos de informação no sistema, a infraestrutura em nuvem suporta todas essas camadas, executando-as em contêineres para alta disponibilidade e escalabilidade.

A construção dessa arquitetura partiu das especificações e serviços estabelecidos nos estágios iniciais do projeto, onde foram identificados os requisitos funcionais e não funcionais, bem como as necessidades de cada tipo de usuário (pacientes, profissionais, empresas parceiras). Com base nesses requisitos, definimos os microsserviços correspondentes a cada funcionalidade, atribuindo responsabilidades claras a cada um. Em seguida, as camadas foram organizadas de modo a assegurar que as interfaces gráficas, a lógica de negócios, o armazenamento de dados e a comunicação entre componentes pudessem ser tratadas de forma independente, garantindo escalabilidade, segurança, flexibilidade e manutenção facilitada à medida que o sistema evolui.

4.5.1 Camada de Visualização

Figura 5 – Esquema de serviços da camada de visualização



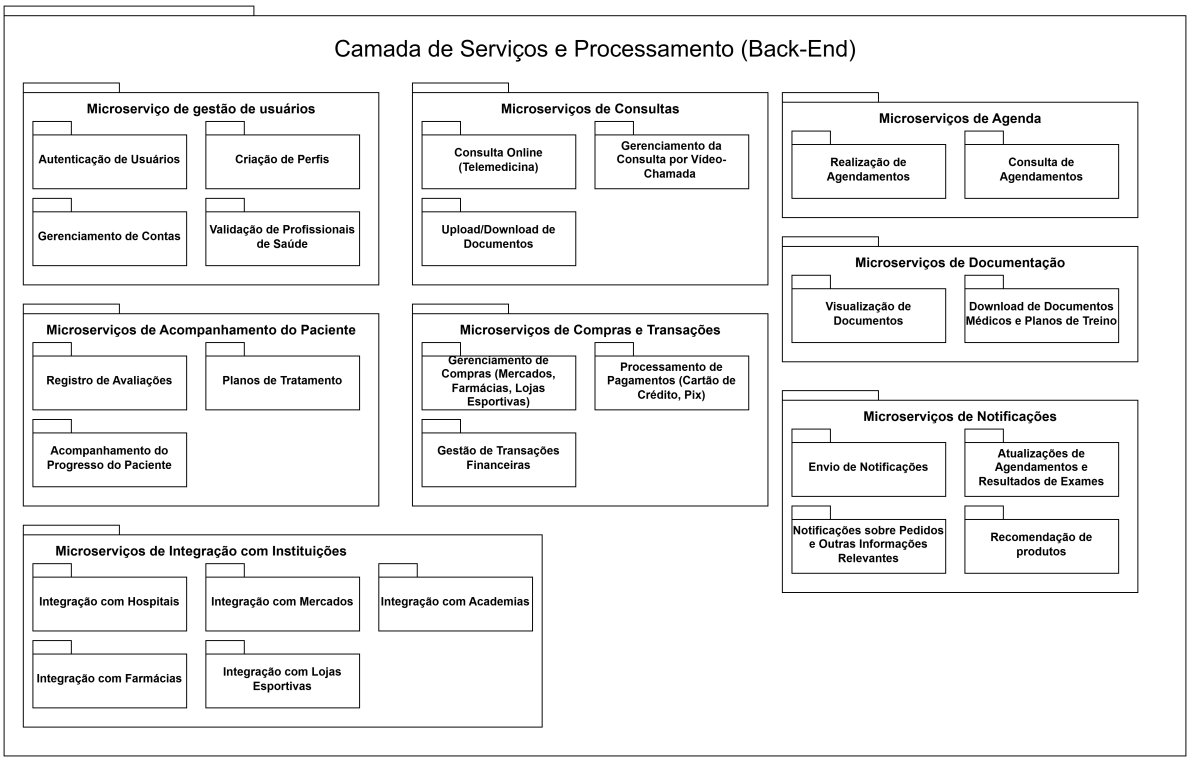
Como pode-se ver, esta camada interage diretamente com os usuários finais e inclui todos os aspectos da interface do usuário (UI) e da experiência do usuário (UX). Podemos resumir nos seguintes produtos:

- Interfaces gráficas que permitem aos usuários interagir com os serviços da plataforma, como perfis de usuário, agendamento de consultas e exames, acesso a resultados de exames bem como realizar compras e consultas.
- Portais especializados para diferentes perfis de usuários (médicos, nutricionistas, treinadores e psicólogos) com funcionalidades adaptadas às suas necessidades.

Vale observar que as interfaces gráficas do sistema são completamente independentes dos outros componentes do sistema sendo esses, responsáveis por alimentar as interfaces de informações por meio de comunicação na cloud.

4.5.2 Camada de Serviços de Processamento

Figura 6 – Esquema de serviços da camada de processamento



Seguindo para as funções complementares das visuais, a camada de processamento é a seção responsável pela lógica de negócios e execução das operações do sistema como descrito na imagem acima. É nessa camada que teremos a organização em microsserviços das funções do sistema e com isso é possível concluir que ela terá as principais informações,

dos usuários do sistema e afins, que serão veiculadas através da rede para as interfaces visuais da plataforma.

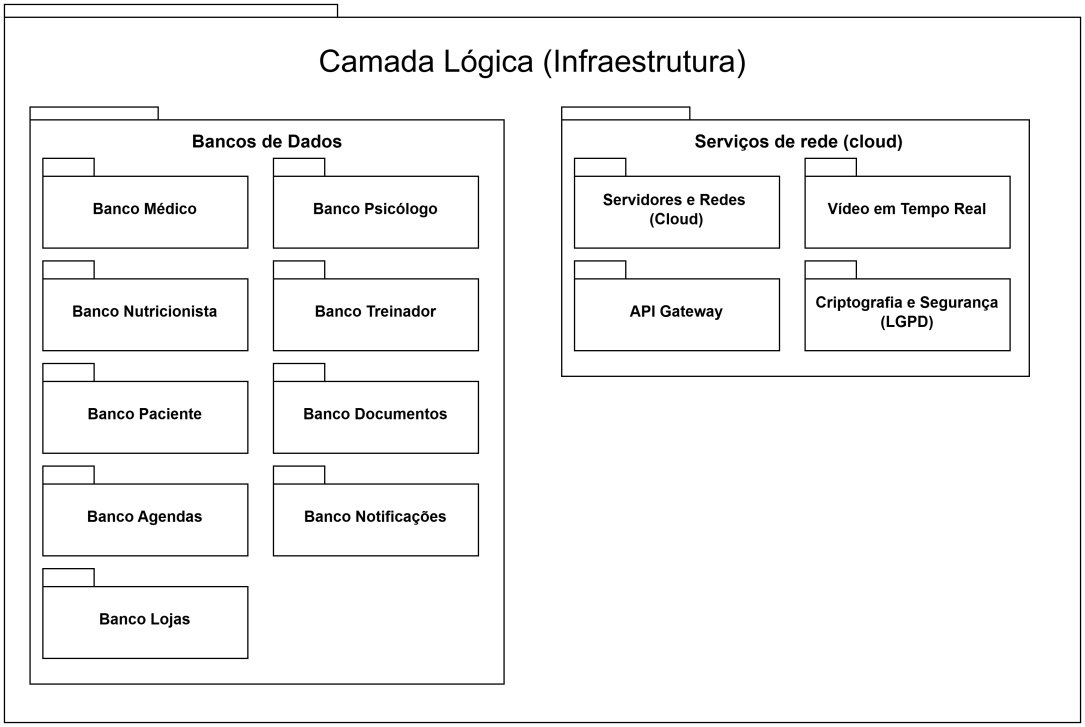
- Gestão de Usuários: Serviços relacionados à autenticação, criação e gerenciamento de perfis e validação de profissionais.
- Consultas: Funcionalidades para telemedicina, gerenciamento de comunicação em tempo real e compartilhamento de documentos.
- Agendamento: Gerenciamento de agendamentos e consultas programadas.
- Documentação: Visualização e download de documentos médicos e planos associados.
- Acompanhamento de Pacientes: Registro, monitoramento de progresso e definição de planos de tratamento.
- Compras e Transações: Integração com compras e pagamentos, além de gestão financeira.
- Notificações: Envio de alertas, atualizações e recomendações personalizadas.
- Integração com Instituições: Conexões com sistemas externos como hospitais, farmácias e academias.

4.5.3 Infraestrutura

Finalmente, último segmento da plataforma digital proposta é a camada lógica do sistema. Aqui tem-se serviços de infraestrutura que incluem recursos de integração e comunicação entre os diferentes microsserviços disponíveis. Dada essa característica, essa camada é composta por serviços de computação em nuvem responsáveis por organizar o fluxo de informações do sistema. Vale observar que isso também incluem todas as diretrizes de segurança necessárias para a manipulação de dados sensíveis de usuários.

- Bancos de Dados: Sistemas de armazenamento especializados para diferentes áreas, como médicos, psicólogos, nutricionistas, pacientes, agendas, lojas, documentos e notificações.
- Servidores e Redes: Recursos baseados em nuvem para suportar os microsserviços e garantir escalabilidade e estabilidade.
- API Gateway: Gerenciamento centralizado de APIs, permitindo integração segura entre os serviços.
- Vídeo em Tempo Real: Comunicação em tempo real utilizando vídeo-chamada integrada.

Figura 7 – Esquema de serviços da infraestrutura



- Criptografia e Segurança: Diretrizes e ferramentas de segurança em conformidade com a LGPD para proteção de dados sensíveis.

5 Desenvolvimento do Trabalho

Todas as seções anteriores que definem os aspectos e conhecimentos necessários para o desenvolvimento tanto da adequada metodologia escolhida quanto dos requisitos do projeto proposto são de extrema importância para a correta implementação do sistema proposto por esse trabalho. Nesse sentido, vamos mostrar neste capítulo todos os fluxos e detalhes do desenvolvimento do sistema multissetorial de saúde em nuvem.

Partindo das bases teóricas e dos requisitos discutidos, portanto, explicitaremos os aspectos práticos da implementação, como a arquitetura baseada em microsserviços, as integrações com APIs e o uso de tecnologias de computação em nuvem. Além disso, serão discutidos os desafios enfrentados e as soluções propostas para garantir que o sistema atenda às necessidades dos usuários, respeitando as diretrizes da LGPD.

5.1 Tecnologias

Considerando que o objetivo central deste capítulo é proporcionar uma visão clara e detalhada de como os componentes do sistema foram concebidos e integrados para entregar uma experiência fluida, vamos primeiro definir as tecnologias escolhidas para a implementação do sistema para depois dissertar sobre como os componentes foram implementados.

5.1.1 Serviços da AWS

A integração dos serviços de computação em nuvem é fundamental para a implementação do nosso sistema. Utilizamos diversos serviços da AWS para arquitetar a infraestrutura em nuvem. A seguir, explicamos cada serviço, descrevendo sua funcionalidade e como é utilizado em nosso projeto.

- **AWS Cognito**
  - *O que é*: O AWS Cognito é um serviço de autenticação, autorização e gerenciamento de usuários para aplicativos web e móveis. Ele permite adicionar registro de usuários, login e controle de acesso às aplicações de forma segura.
  - *Como utilizamos*: Utilizamos o Cognito para gerenciar a autenticação dos usuários no sistema. Quando um usuário é registrado, ele começa com o e-mail não verificado. Uma função Lambda altera seu status para e-mail verificado após a confirmação.

- **AWS API Gateway**
  - *O que é*: O AWS API Gateway é um serviço que permite criar, publicar, manter, monitorar e proteger APIs em qualquer escala. Ele atua como uma porta de entrada para aplicativos que acessam dados, lógica de negócios ou funcionalidades de serviços de back-end.
  - *Como utilizamos*: O API Gateway é usado para criar endpoints que acionam funções Lambda, servindo como a maneira tradicional de comunicação entre o front-end e o back-end.
- **AWS Lambda**
  - *O que é*: O AWS Lambda permite executar código sem provisionar ou gerenciar servidores. Você paga apenas pelo tempo de computação consumido.
  - *Como utilizamos*: Utilizamos funções Lambda para simular microsserviços, com cada função realizando uma tarefa específica. Isso promove uma arquitetura altamente modularizada.
- **AWS EventBridge**
  - *O que é*: O AWS EventBridge é um barramento de eventos *serverless* que facilita a criação de aplicações orientadas a eventos usando dados de suas próprias aplicações, aplicativos integrados SaaS e serviços da AWS.
  - *Como utilizamos*: Utilizamos o EventBridge para agendar *triggers* de funções Lambda. Por exemplo, às 0h de cada dia, uma Lambda é acionada para criar *slots* de disponibilidade para os próximos 90 dias.
- **Amazon SQS (Simple Queue Service)**
  - *O que é*: O Amazon SQS é um serviço de enfileiramento de mensagens totalmente gerenciado que permite desacoplar e escalar microsserviços, sistemas distribuídos e aplicações *serverless*.
  - *Como utilizamos*: Utilizamos o SQS para acionar Lambdas que criam as disponibilidades dos profissionais ao criar a conta, garantindo que essas tarefas sejam processadas de forma assíncrona e resiliente.
- **AWS Location Service**
  - *O que é*: O AWS Location Service facilita para os desenvolvedores adicionar funcionalidades de localização aos seus aplicativos sem sacrificar a segurança dos dados do usuário.

  - *Como utilizamos*: Implementamos o AWS Location para funcionalidades que requerem dados geográficos, como mostrar a localização de empresas ou calcular distâncias entre usuários e prestadores de serviço.

- **AWS RDS (Relational Database Service)**
  - *O que é*: O AWS RDS facilita a configuração, operação e escalabilidade de bancos de dados relacionais na nuvem.
  - *Como utilizamos*: Utilizamos o RDS para gerenciar o armazenamento de dados relacionais do sistema, como informações de usuários, agendamentos e transações.

- **AWS VPC (Virtual Private Cloud)**
  - *O que é*: O AWS VPC permite provisionar uma seção isolada logicamente da Nuvem AWS onde você pode lançar recursos da AWS em uma rede virtual que você define.
  - *Como utilizamos*: Configuramos uma VPC para isolar nossa infraestrutura de rede, aumentando a segurança e permitindo o controle sobre o ambiente de rede, incluindo seleção de endereços IP, sub-redes e configuração de tabelas de rotas.

- **Amazon S3 (Simple Storage Service)**
  - *O que é*: O Amazon S3 é um serviço de armazenamento de objetos que oferece escalabilidade, disponibilidade de dados, segurança e performance líderes do setor.
  - *Como utilizamos*: Usamos o S3 para armazenar arquivos estáticos e ativos do aplicativo, como imagens de perfil, documentos e outros arquivos necessários.

- **AWS KMS (Key Management Service)**
  - *O que é*: O AWS KMS é um serviço gerenciado que facilita a criação e o controle de chaves de criptografia usadas para criptografar dados.
  - *Como utilizamos*: Utilizamos o KMS para gerenciar chaves de criptografia que protegem dados sensíveis armazenados em nossos serviços principalmente informações de usuários, como dados do RDS e objetos no S3.

- **Amazon Route 53**
  - *O que é*: O Amazon Route 53 é um serviço web de DNS altamente disponível e escalável.
  - *Como utilizamos*: Usamos o Route 53 para gerenciar o roteamento de tráfego para nosso aplicativo, incluindo a configuração de domínios personalizados e roteamento de solicitações para o CloudFront ou API Gateway.

- **Amazon CloudFront**
  - *O que é*: O Amazon CloudFront é um serviço de rede de entrega de conteúdo (CDN) que distribui dados, vídeos, aplicativos e APIs para clientes globalmente com baixa latência e altas velocidades de transferência.
  - *Como utilizamos*: O CloudFront é usado para distribuir nosso aplicativo front-end React e conteúdos estáticos, garantindo rápida entrega de conteúdo aos usuários em todo o mundo. A intenção aqui também é utilizar esse serviço para garantir protocolos de criptografia e segurança adequados para o desenvolvimento web padrão
- **AWS CloudWatch**
  - *O que é*: O AWS CloudWatch é um serviço de monitoramento que fornece dados e insights acionáveis para monitorar aplicações, responder a alterações de performance e otimizar a utilização de recursos.
  - *Como utilizamos*: Usamos o CloudWatch para monitorar nossos recursos AWS e aplicativos, coletando e rastreando métricas, coletando e monitorando arquivos de log, e configurando alarmes.

Dessa forma, a integração desses serviços AWS permite construir uma infraestrutura escalável, segura e eficiente, alinhada com os requisitos do nosso sistema e os princípios de arquitetura de microsserviços.

5.1.2 Linguagens de programação

Dada a definição dos serviços em nuvem, uma outra discussão significativa foi realizada para definir as linguagens que seriam a base do sistema. Inicialmente, um protótipo do sistema foi desenvolvido utilizando o framework Django. No entanto, decidimos transicionar esse estilo de código para melhor representar a arquitetura de microsserviços proposta. Com isso, optamos pelas seguintes tecnologias:

- **Python**:
  - **Integração com AWS Lambda**: Como vamos utilizar funções Lambda da AWS, é necessário empregar *layers* para incluir bibliotecas que não estão instaladas por padrão no ambiente Lambda. O Python oferece um ecossistema vasto de bibliotecas e *layers* já implementadas, o que facilita esse processo. Considerando isso, podemos dizer que vamos manter a mesma linguagem que o Django, mas não utilizar o framework para realizar o desenvolvimento

  - **Modularização e Desempenho**: Tem-se várias Lambdas bem modularizadas, cada uma realizando apenas uma tarefa específica. Com tempos de execução inferiores a 100ms, a facilidade de implementação torna-se uma prioridade maior do que a velocidade de processamento da linguagem.

- **React**:

  - **Adequação ao Frontend**: O Django não é um framework especializado para frontend; é voltado para backend ou desenvolvimento *full-stack*. Utilizá-lo limitaria nossas opções de design ao Bootstrap, não atendendo às necessidades de um frontend moderno e responsivo.
  - **Arquitetura de Microsserviços**: Para o backend, o uso do Django implicaria no desenvolvimento da API de cada serviço como um bloco monolítico, exigindo hospedagem em containers ou instâncias EC2. Isso contraria a abordagem de microsserviços que adotamos.
  - **Popularidade e Comunidade**: O React é um dos frameworks mais utilizados para frontend, oferecendo uma vasta quantidade de componentes já desenvolvidos e uma comunidade ativa.
  - **Bibliotecas e Ferramentas**: A disponibilidade de bibliotecas como o Chakra UI e o Amazon Chime SDK para React acelera o desenvolvimento e garante consistência na interface do usuário. Além disso, React pode request diretamente pro API Gateway que por consequência triggar as lambdas e isso simplifica muito a implementação.

Dessa forma, a combinação de Python para o backend, através de funções Lambda altamente modularizadas, e React para o frontend nos permite construir um sistema eficiente, escalável e de fácil manutenção, e, nesse caso, também alinhado com os princípios da arquitetura de microsserviços.

5.2 Bases de Dados

5.2.1 Segmentação e Independência Estrutural

Na arquitetura de microsserviços, é ideal que cada serviço possua seu próprio banco de dados, garantindo segmentação e independência estrutural. Essa abordagem promove o isolamento entre serviços, facilitando a manutenção, a escalabilidade e reduzindo a possibilidade de efeitos colaterais causados por alterações em outros serviços. No entanto, essa prática pode implicar em custos adicionais significativos que seriam válidos para uma implantação em ambiente produtivo, mas não para a prova de conceito do nosso trabalho proposto.

Assim, devido às restrições de custo, especialmente considerando que o AWS RDS oferece apenas 750 horas gratuitas por mês (o que equivale a aproximadamente um mês de uso para uma instância), optou-se por utilizar um único banco de dados para todos os serviços. Para manter a segurança e a independência lógica dos dados, foram implementados mecanismos que impedem que endpoints acessem todo o banco de dados indiscriminadamente. Quando um serviço necessita de dados de outro, utilizam-se endpoints específicos ou o Amazon SQS para mediar essa comunicação.

Por exemplo, o serviço de consultas não tem acesso direto ao banco de dados de profissionais nem de pacientes. Para exibir os dados necessários, é utilizado um endpoint que retorna as informações com base no ID do profissional ou paciente, que é uma coluna na tabela de consultas. Gostaríamos de destacar aqui, portanto, que apesar de optarmos por modificar um pouco a estrutura, **essa abordagem ainda sim mantém a separação lógica dos dados e respeita os princípios de encapsulamento de uma arquitetura de microsserviços.**

5.2.2 Relacional vs. Não-Relacional

A escolha entre bancos de dados relacionais e não-relacionais depende das necessidades específicas do sistema. Bancos de dados relacionais, como os oferecidos pelo AWS RDS, são ideais para aplicações que requerem consistência e integridade referencial, suportando transações complexas e relacionamentos estruturados entre tabelas. Eles são especialmente adequados quando é necessário manter transações ACID (Atomicidade, Consistência, Isolamento e Durabilidade), garantindo que todas as operações no banco de dados sejam executadas de forma confiável.

Por outro lado, bancos de dados não-relacionais (NoSQL) oferecem maior flexibilidade na modelagem de dados e são projetados para lidar com grandes volumes de dados não estruturados ou semi-estruturados. Eles são frequentemente utilizados em aplicações que exigem alta escalabilidade horizontal e desempenho, como sistemas de *big data*, análises em tempo real e aplicações que lidam com dados de formatos variados.

Considerando isso, optou-se por utilizar um **banco de dados relacional** devido à natureza dos dados e operações do sistema, que requerem transações confiáveis, integridade dos dados e relações bem definidas entre diferentes entidades, como usuários, profissionais, agendamentos e pagamentos. Por curiosidade, se fossemos optar pela outra alternativa, teríamos que utilizar o DynamoDB da AWS, necessitando de algumas configurações adicionais.

Tabela 17 – Comparação entre Bancos de Dados Relacionais e Não-Relacionais

Características	**Banco Relacional**	**Banco Não-Relacional**
Modelo de Dados	Baseado em tabelas com linhas e colunas	Flexível, incluindo documentos, chave-valor, colunas e grafos
Linguagem de Consulta	SQL (Structured Query Language)	Diversas APIs e linguagens específicas
Esquema	Esquema rígido, definido previamente	Esquema flexível ou ausente
Transações	Suporte a transações ACID	Suporte eventual a ACID, frequentemente BASE (Basicamente Disponível, Estado Suave, Consistência Eventual)
Escalabilidade	Escalabilidade vertical (aumento de recursos do servidor)	Escalabilidade horizontal (adição de mais servidores)
Consistência	Forte consistência e integridade referencial	Consistência eventual ou definida pela aplicação
Tipos de Dados	Adequado para dados estruturados	Adequado para dados semi-estruturados ou não estruturados
Aplicações	Sistemas com relações complexas e necessidade de transações confiáveis	Aplicações com grande volume de dados e necessidade de alta escalabilidade

Fonte: Adaptado de estudos sobre bancos de dados relacionais e não-relacionais

5.2.3 Organização da Base de Dados na Prática

A organização do banco de dados foi planejada para refletir as necessidades operacionais do sistema, mantendo a coerência e eficiência nas operações. A seguir, são apresentadas as principais tabelas do banco de dados com uma breve descrição:

- **Usuários**: armazena informações básicas de autenticação e autorização, como `user_id` e `user_type` (paciente, profissional ou empresa).
- **Pacientes**: contém detalhes pessoais dos pacientes, como nome, data de nascimento, gênero e informações de saúde relevantes.
- **Profissionais**: registra dados dos profissionais de saúde, incluindo especialidades, qualificações e horários de atendimento.
- **Empresas**: guarda informações sobre empresas ou clínicas de saúde, como razão social, CNPJ e endereços.
- **Consultas**: gerencia o agendamento e o histórico de consultas entre pacientes e profissionais, armazenando datas, horários e status.
- **Disponibilidades**: mantém os horários disponíveis dos profissionais para agendamento de consultas.

As representações visuais do modelo entidade-relacionamento (ER) dessas tabelas são apresentadas a seguir. Os detalhes sobre as manipulações serão discutidos posteriormente.

- **Entidade-Relacionamento dos Agendamentos**:

  A tabela de **Agendamentos** (`appointments`) gerencia o agendamento de consultas entre **Pacientes** e **Profissionais**. Cada agendamento possui um identificador único (`appointment_id`), referências ao `slot_id` e `patient_id`, além de campos para o **status** da consulta (`status_enum`) e timestamps de criação e atualização. A relação entre **Agendamentos**, **Slots**, **Pacientes** e **Profissionais** é de um-para-muitos, onde um paciente pode ter múltiplos agendamentos e um profissional pode ter múltiplos agendamentos associados a ele.

- **Entidade-Relacionamento das Avaliações**:

  A tabela de **Avaliações** (`assessments`) permite que os **Pacientes** avaliem os **Profissionais** após as consultas. Cada avaliação contém um identificador único (`assessment_id`), referência ao `appointment_id`, `patient_id`, `professional_id`, além de campos para indicar se é uma avaliação simplificada (`is_simplified`) e o conteúdo da avaliação. A relação é de um-para-um entre **Agendamentos** e **Avaliações**, garantindo que cada consulta possa receber uma avaliação específica.

- **Entidade-Relacionamento das Empresas**:

  A tabela de **Empresas** (`companies`) armazena informações sobre as empresas parceiras ou contratantes do sistema. Cada empresa possui um identificador único (`company_id`), referências ao `user_id` na tabela **Usuários**, além de campos como `cnpj`, `company_type`, `api_key`, `is_active`, `products_endpoint`, `orders_endpoint` e `address_id` que referencia a tabela **Endereços**. A relação entre **Empresas** e **Profissionais** é de um-para-muitos, permitindo que uma empresa gerencie múltiplos profissionais de saúde.

- **Entidade-Relacionamento dos Documentos**:

  A tabela de **Documentos** (`documents`) gerencia os arquivos associados aos **Usuários**, **Pacientes**, **Profissionais** e **Agendamentos**. Cada documento possui um identificador único (`document_id`), referências aos `patient_id`, `professional_id` e `appointment_id`, além de campos para o tipo de documento, nome, data de upload e a chave do objeto no S3 (`s3_object_key`). A relação é de muitos-para-um, onde múltiplos documentos podem estar associados a um único usuário, paciente, profissional ou agendamento.

- **Entidade-Relacionamento das Notificações**:

A tabela de **Notificações** (`notifications`) gerencia as mensagens enviadas aos **Usuários**. Cada notificação possui um identificador único (`notification_id`), referência ao `user_id`, tipo de notificação (`notification_type_enum`), conteúdo da mensagem, status de leitura (`is_read`) e timestamp de criação. A relação é de um-para-muitos, permitindo que um usuário receba múltiplas notificações.

- **Entidade-Relacionamento das Opções de Pedido**:

  A tabela de **Opções de Pedido** (`order_options`) lista os diferentes serviços e produtos disponíveis no sistema. Cada opção possui um identificador único (`order_option_id`), referências ao `plan_id` e `company_id`, além de campos para o preço total (`total_price`) e timestamp de criação. A relação entre **Opções de Pedido**, **Planos** e **Empresas** é de muitos-para-muitos, intermediada pela tabela **Item_Options** (`item_options`), que registra quais itens de plano foram incluídos em cada opção de pedido.

- **Entidade-Relacionamento dos Pedidos**:

  A tabela de **Pedidos** (`orders`) detalha as solicitações de serviços ou produtos efetuadas pelos **Pacientes**. Cada pedido possui um identificador único (`order_id`), referências ao `patient_id`, `order_option_id`, além de campos para o valor total (`total_amount`), data do pedido (`order_date`), status (`status_enum`), método de pagamento (`payment_method_enum`), identificador da transação (`transaction_id`) e referência ao endereço de envio (`shipping_address_id`). A relação com **Opções de Pedido** é de muitos-para-muitos, permitindo que um pedido inclua múltiplas opções, intermediado pela tabela **Item_Options**.

- **Entidade-Relacionamento dos Pacientes**:

  A tabela de **Pacientes** (`patients`) organiza as informações dos pacientes cadastrados no sistema. Cada paciente possui um identificador único (`patient_id`), referências ao `user_id` na tabela **Usuários**, além de campos para data de nascimento (`birthday`), peso (`weight`), altura (`height`), gênero (`gender`), restrições alimentares (`food_restrictions`) e referência ao endereço (`address_id`). A relação entre **Pacientes** e **Agendamentos** é de um-para-muitos, permitindo que um paciente tenha múltiplas consultas agendadas.

- **Entidade-Relacionamento dos Planos**:

  A tabela de **Planos** (`plans`) estrutura os planos de assinatura ou pacotes de serviços oferecidos aos **Pacientes** e **Profissionais**. Cada plano possui um identificador único (`plan_id`), referências ao `patient_id` e `professional_id`, além de campos para o tipo de plano (`plan_type_enum`), descrição (`description`) e timestamp de criação (`created_at`). A relação entre **Planos** e **Itens de Plano** (`plan_items`) é de um-para-muitos, permitindo que um plano inclua múltiplos itens de plano.

- **Entidade-Relacionamento dos Profissionais**:

  A tabela de **Profissionais** (`professionals`) detalha as informações dos profissionais de saúde disponíveis na plataforma. Cada profissional possui um identificador único (`professional_id`), referências ao `user_id` na tabela **Usuários**, além de campos para a profissão (`profession_enum`), credenciais (`credentials`), preço por consulta (`price`), status de verificação (`is_verified`), bio (`bio`) e especialidades (`specialties`). A relação entre **Profissionais** e **Agendamentos** é de um-para-muitos, permitindo que um profissional atenda a múltiplos agendamentos.

- **Entidade-Relacionamento dos Slots**:

  A tabela de **Slots** (`slots`) ilustra a disponibilidade de horários dos **Profissionais** para agendamentos. Cada slot possui um identificador único (`slot_id`), referências ao `availability_id` na tabela **Disponibilidades**, data e hora do slot (`slot_datetime`), além de campos para indicar se o slot está reservado (`is_reserved`) ou bloqueado (`is_blocked`). A relação entre **Slots** e **Disponibilidades** é de muitos-para-um, permitindo que múltiplos slots sejam gerenciados por uma única disponibilidade.

- **Entidade-Relacionamento dos Usuários**:

  A tabela de **Usuários** (`users`) apresenta a estrutura básica dos usuários do sistema, diferenciando-se em três tipos: **Pacientes**, **Profissionais** e **Empresas**, definidos pelo campo `user_type_enum`. Cada usuário possui um identificador único (`user_id`), `email`, `password_hash`, `name`, tipo de usuário (`user_type`), além de timestamps de criação (`created_at`) e atualização (`updated_at`). As relações principais são:

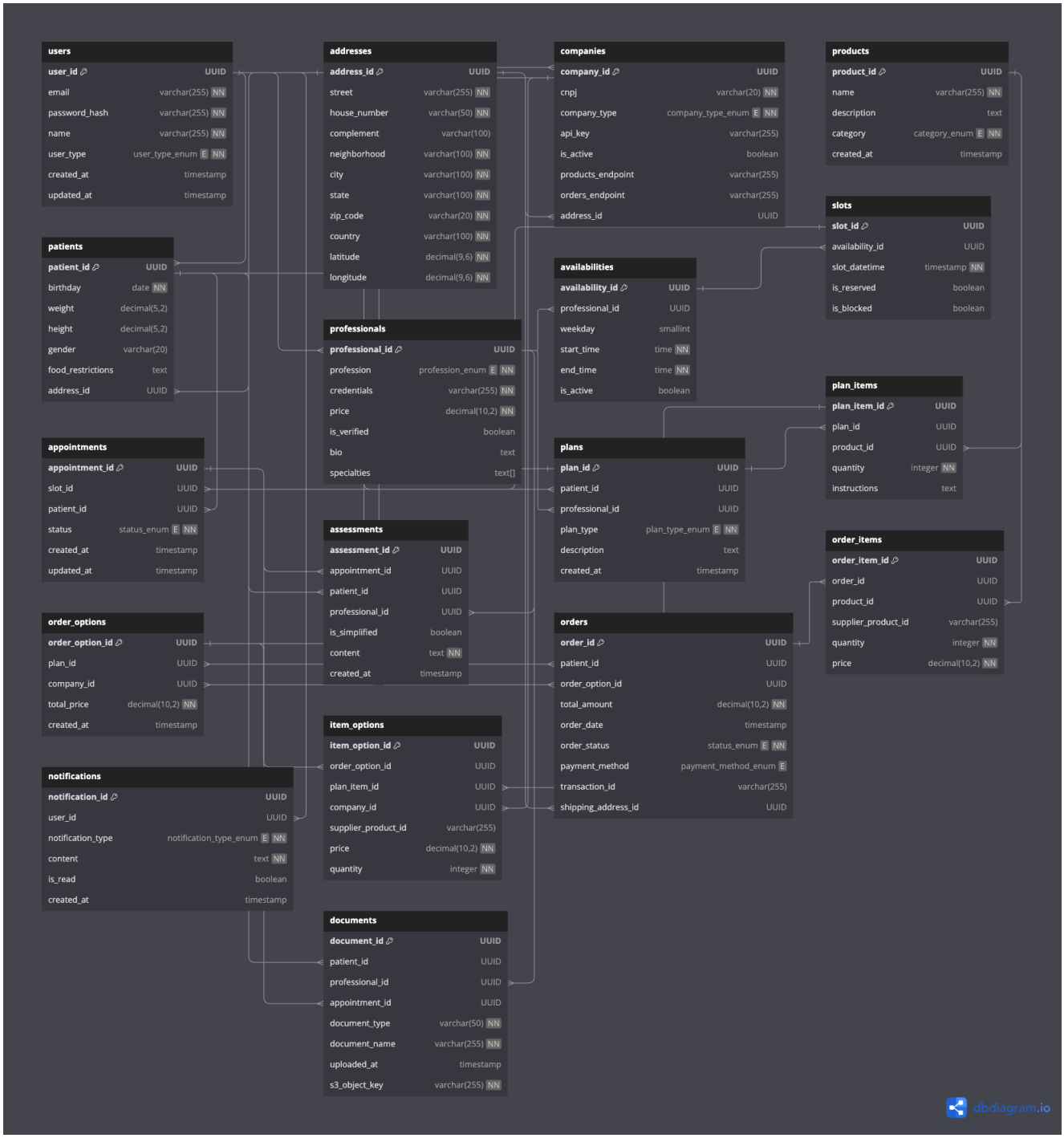
  - **Usuários Pacientes**: Relacionados à tabela **Pacientes** (`patients`) para armazenar informações específicas dos pacientes.
  - **Usuários Profissionais**: Relacionados à tabela **Profissionais** (`professionals`) para armazenar informações específicas dos profissionais de saúde.
  - **Usuários Empresas**: Relacionados à tabela **Empresas** (`companies`) para armazenar informações sobre as organizações de saúde.

Cada uma dessas relações e entidades detalham as interações entre os agentes do sistema e elas podem ser integradas em uma unica base de dados integrada.

Por conta disso, a organização nas tabelas está disposta da seguinte maneira:

Isso fornece uma visão geral do banco de dados completo e as interações entre as diferentes tabelas. Podemos resumir, finalmente na seguinte tabela:

Figura 8 – Entidade-Relacionamento Geral do Banco de Dados



5.3 Implementação dos Microsserviços

A adoção de microsserviços é fundamental para aumentar a escalabilidade, flexibilidade e facilidade de manutenção do sistema. Ao dividir a aplicação em componentes menores e independentes, podemos desenvolver, implantar e escalar cada serviço separadamente, permitindo uma resposta mais ágil às mudanças de requisitos e demandas de

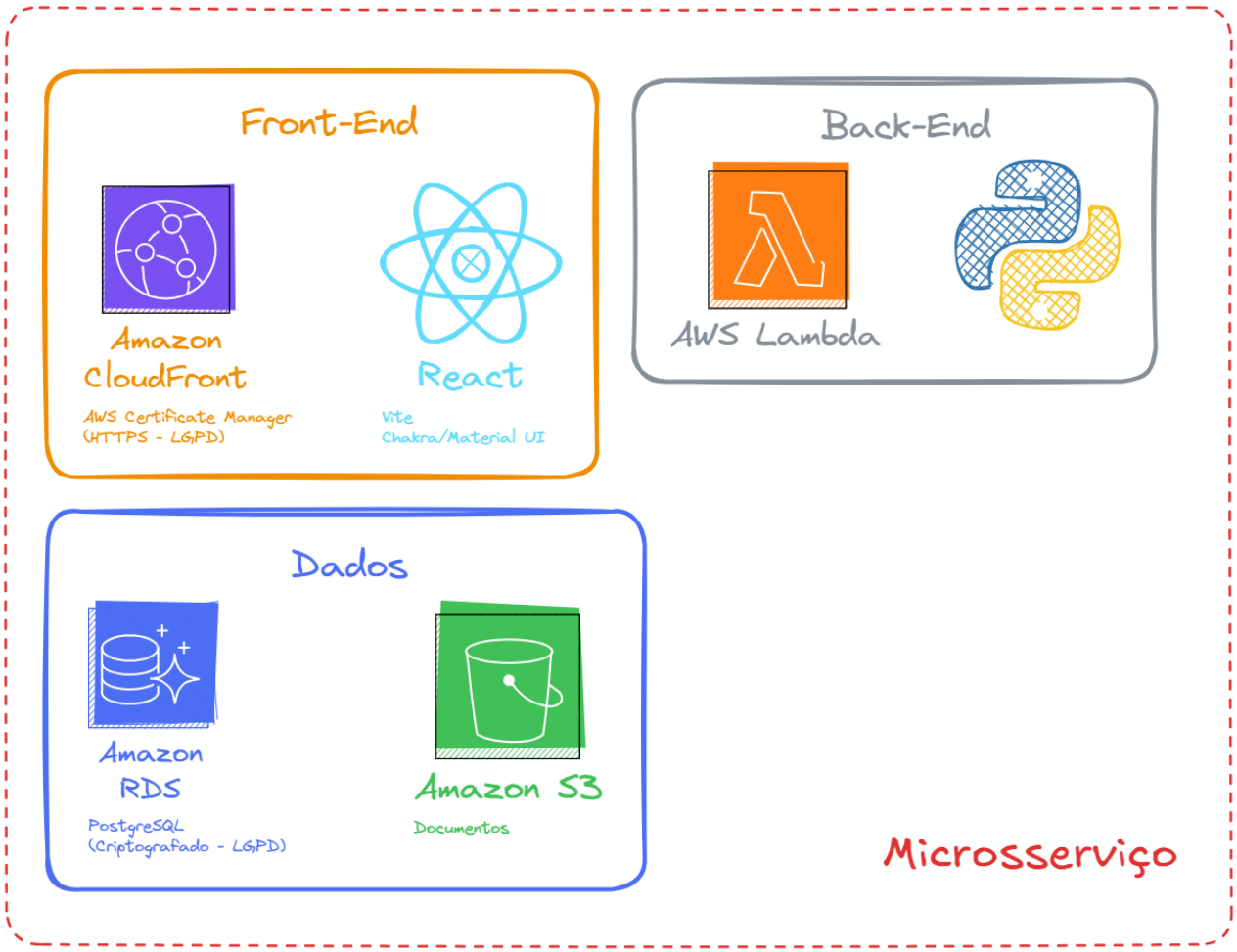
carga. Essa abordagem também facilita a integração contínua e a implementação de novas funcionalidades sem impactar todo o sistema.

Para implementar os microsserviços, utilizamos os serviços da AWS, como o AWS Lambda para execução de código sem servidor, o Amazon API Gateway para gerenciamento de APIs, o Amazon SQS para filas de mensagens e o Amazon RDS para armazenamento de dados como mencionados anteriormente. Em suma, visamos garantir que exista possibilidade de expansão com uma arquitetura bem estruturada para isso.

5.3.1 Microsserviços e Arquitetura

Podemos resumir cada unidade de microsserviços da seguinte maneira:

Figura 9 – Unidade lógica de um microsserviço



Assim, considerando a junção de microsserviços e respeitando a arquitetura em camadas definida na especificação de requisitos, podemos resumir a arquitetura em nuvem proposta pelo sistema nas imagens a seguir. Note que existe preocupação com a manipulação de dados sensíveis, adotando-se diferentes abordagens conforme o tipo de informação,

como criptografia de documentos, controle de acesso a registros e monitoramento constante das interações entre os serviços. Dessa forma, a estrutura final combina segurança, escalabilidade e modularidade, garantindo que cada componente desempenhe seu papel de forma independente, porém integrada. Isso faz com que todo o sistema fique de maneira modular

Figura 10 – Arquitetura Geral do Sistema

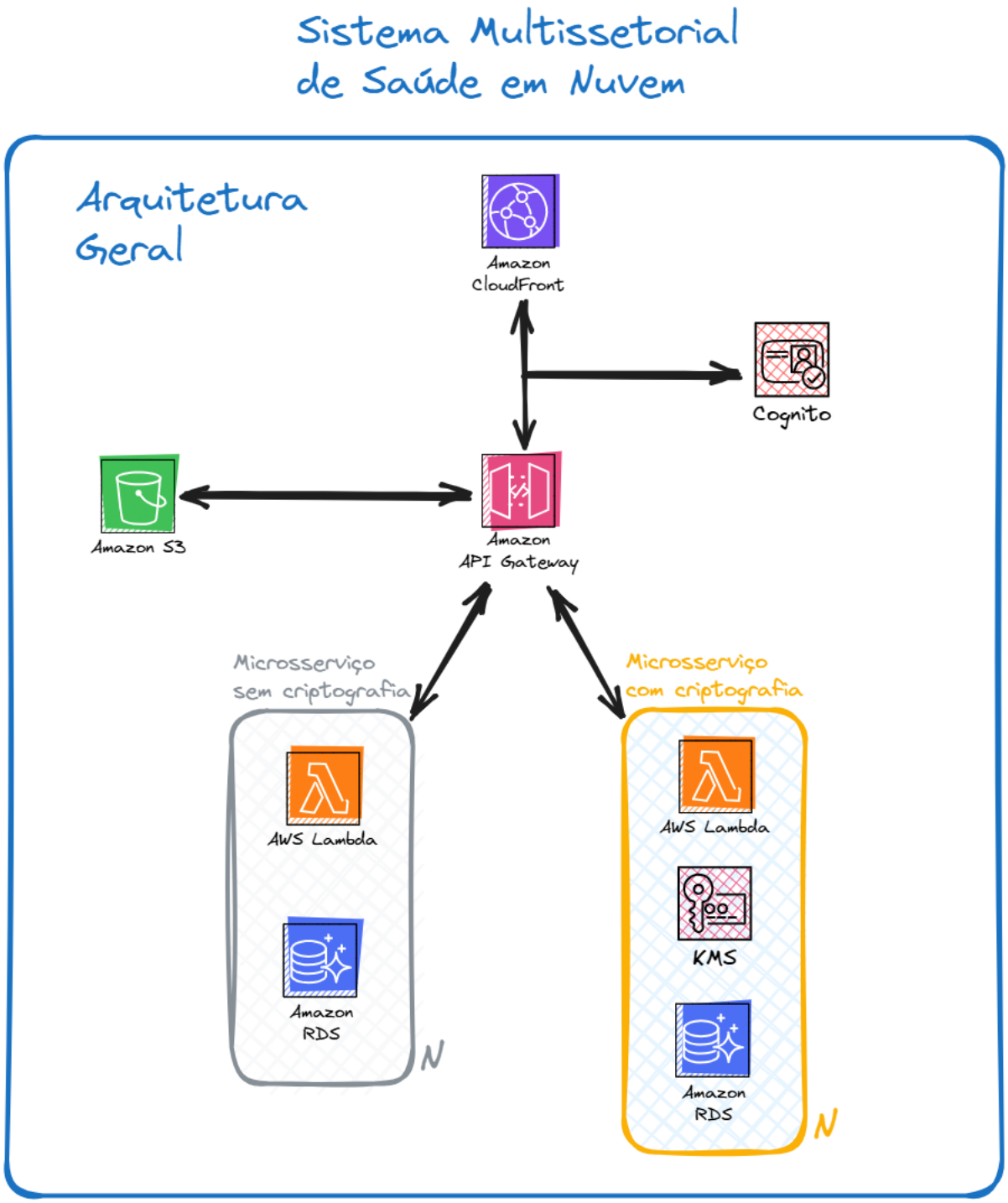


Figura 11 – Arquitetura dos signup

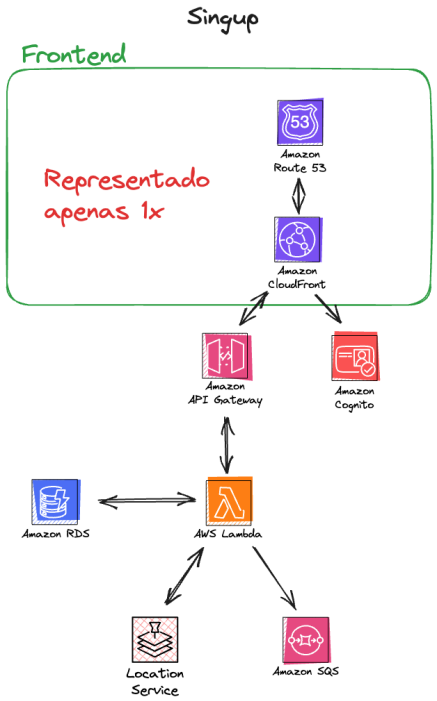


Figura 12 – Arquitetura das Páginas

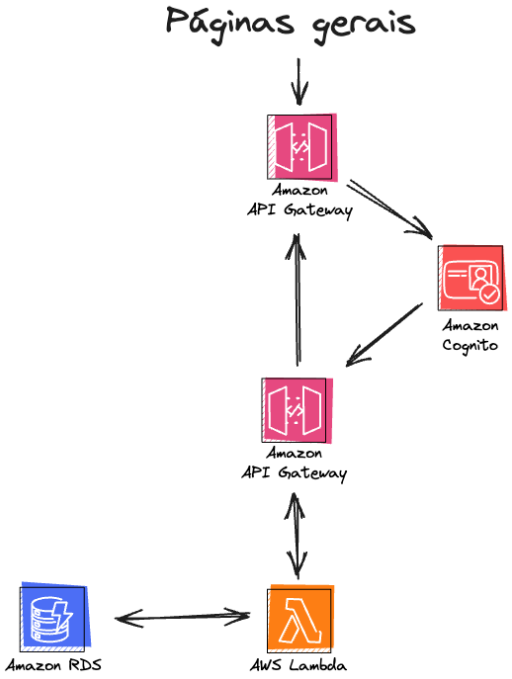


Figura 13 – Arquitetura de Enriquecimento de Dados

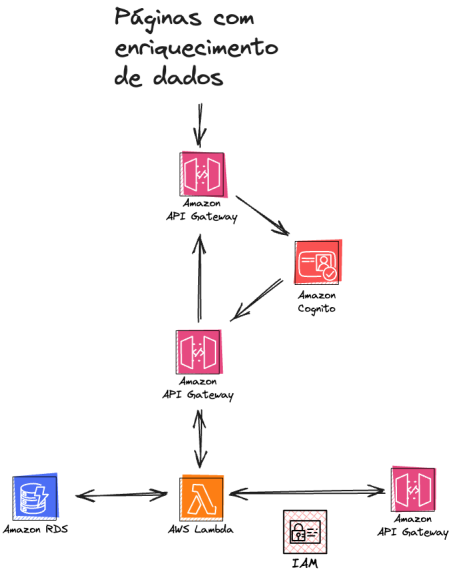
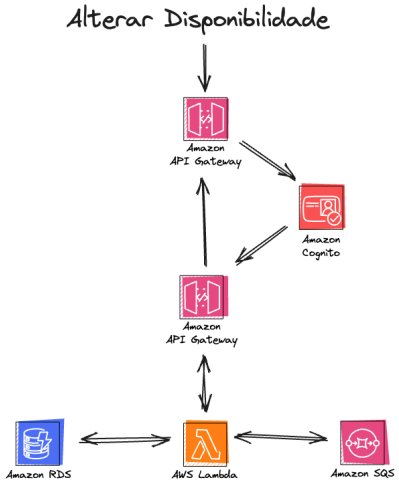


Figura 14 – Arquitetura da Disponibilidade



Essas arquiteturas não apenas organizam o fluxo de informações na nuvem, mas também fornecem uma estrutura replicável e escalável que pode ser aplicada a diversas aplicações. Ao definir claramente responsabilidades, camadas e microsserviços especializados, a solução mostra como é possível adaptar princípios similares a outros cenários tecnológicos. Esse exemplo oferece um caminho concreto para implementar abordagens semelhantes em produtos futuros, mantendo a flexibilidade e a segurança necessárias para acompanhar demandas em constante evolução.

Figura 15 – Arquitetura das opções de compra

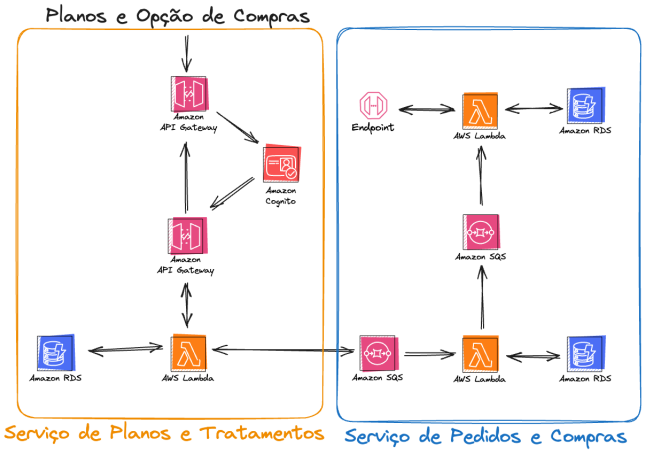


Figura 16 – Arquitetura dos slots

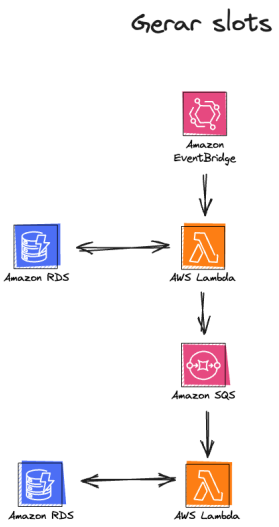


Figura 17 – Arquitetura das prescrições

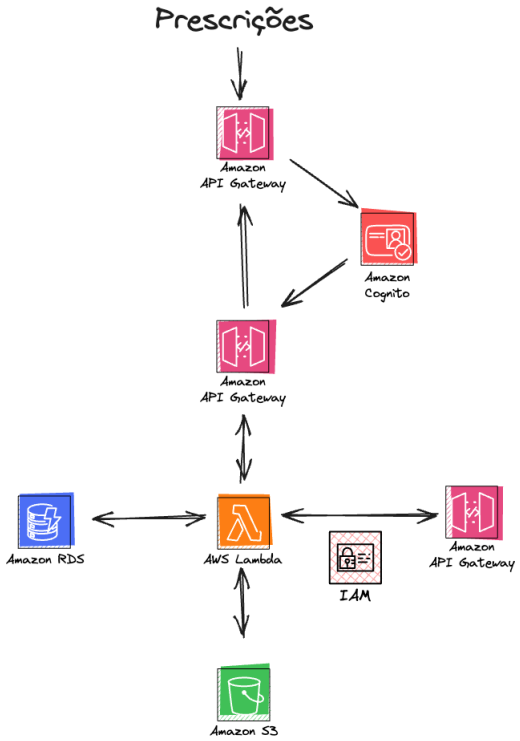
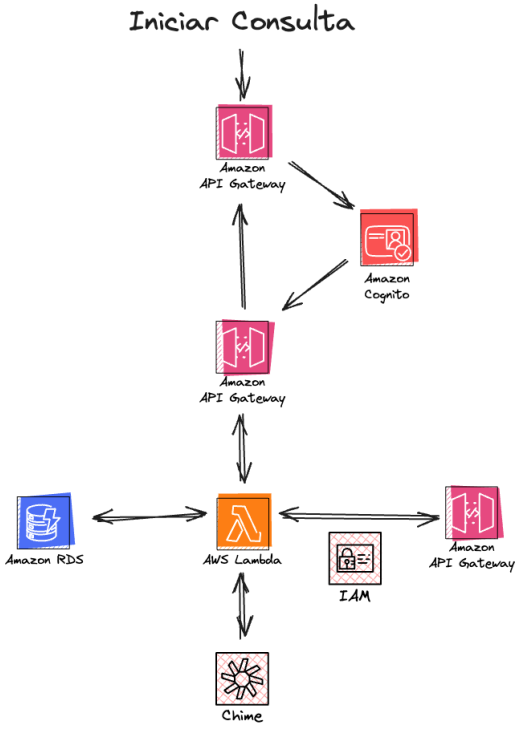


Figura 18 – Arquitetura das Consultas



5.3.2 Features da Plataforma Digital

Dando seguimento, apresentamos as principais funcionalidades da plataforma, organizadas para evidenciar os recursos disponíveis para cada tipo de usuário. As funcionalidades foram desenvolvidas para atender às necessidades específicas de pacientes, profissionais de saúde e empresas parceiras, garantindo uma experiência integrada e eficiente.

1. **Cadastro e Gerenciamento de Usuários**

Descrição Geral: A plataforma permite o registro e gerenciamento de três tipos de usuários: pacientes, profissionais de saúde e empresas parceiras. Cada tipo de usuário possui informações e funcionalidades específicas.

Detalhes: Os pacientes podem registrar informações pessoais como nome, e-mail, senha, data de nascimento, peso, altura, gênero e restrições alimentares, além de associar um endereço residencial completo. Profissionais de saúde registram informações profissionais como credenciais (CRM, CRP, CRN, etc.), preço das consultas, biografia e especialidades, e gerenciam suas disponibilidades para agendamentos. Empresas parceiras registram informações empresariais como CNPJ, tipo de empresa, chaves de API para integração, endpoints para produtos e pedidos, e associam um endereço comercial.

Referências ao BD

Tabelas `users`, `patients`, `professionals`, `companies`, `addresses`, `availabilities`, `slots`, `assessments`, `products`.

2. **Dashboards Personalizados**

Descrição Geral: Cada tipo de usuário possui um dashboard que centraliza as funcionalidades disponíveis, proporcionando uma experiência personalizada.

Detalhes: Pacientes têm acesso a resumo de compromissos, agendamento de atendimentos, gerenciamento de consultas, opções de compras, histórico de saúde, mensagens e pagamentos. Profissionais podem gerenciar agenda, planos de tratamento, avaliações, pacientes atendidos, mensagens e prescrições. Empresas acompanham pedidos, gerenciam integração via API, atualizam perfil e gerenciam usuários associados.

Referências ao BD

Tabelas `appointments`, `plans`, `order_options`, `item_options`, `orders`, `assessments`, `documents`, `notifications`, `plan_items`.

3. **Agendamento de Consultas e Exames**

Descrição Geral: Facilita o agendamento entre pacientes e profissionais, com visualização de disponibilidades, gerenciamento de compromissos e integração com empresas de saúde para exames.

Detalhes: Permite a visualização das disponibilidades dos profissionais em um calendário interativo, agendamento de consultas médicas e exames, notificações de lembrete e opções para remarcação e cancelamento de consultas. Integração com hospitais e laboratórios parceiros para agendamento de exames e recebimento de resultados diretamente na plataforma.

Referências ao BD

Tabelas `availabilities`, `slots`, `appointments`, `notifications`, `companies`, `documents`.

4. **Planos de Tratamento e Acompanhamento**

   Descrição Geral: Permite que profissionais elaborem planos de tratamento personalizados e acompanhem o progresso dos pacientes, incluindo avaliações detalhadas e simplificadas, com controle de compartilhamento.

   Detalhes: Profissionais criam planos de nutrição, exercícios, terapia ou medicação, registram itens necessários com quantidades e instruções detalhadas, e compartilham avaliações e planos com controle sobre quem pode acessá-los, garantindo confidencialidade e conformidade com a LGPD.

   Referências ao BD

   Tabelas `plans`, `plan_items`, `assessments`, `appointments`.

5. **Integração com Parceiros e Facilitação de Compras**

   Descrição Geral: Facilita a aquisição de produtos e serviços necessários para os planos de tratamento, integrando a plataforma com parceiros como farmácias, mercados, lojas esportivas, hospitais e academias.

   Detalhes: A plataforma consulta as APIs dos parceiros para verificar disponibilidade e preços dos itens necessários, criando opções de pedidos que o paciente pode comparar e adquirir de forma simplificada. Inclui comparação de preços, detalhes dos itens e processo de compra com um clique. Tipos de parceiros incluem farmácias (compra de medicamentos), mercados (compra de alimentos), lojas esportivas (compra de equipamentos), hospitais (agendamento de exames) e academias (inscrição em planos).

   Referências ao BD

   Tabelas `companies`, `order_options`, `item_options`, `orders`, `products`.

6. **Comunicação e Mensageria**

   Descrição Geral: Fornece canais de comunicação seguros entre pacientes e profissionais, incluindo chat integrado e notificações para troca de informações relevantes.

   Detalhes: Permite troca de mensagens instantâneas, envio de arquivos e central de notificações para mensagens novas e atualizações, garantindo a confidencialidade das informações e conformidade com a LGPD.

   Referências ao BD

   Tabela `notifications`.

7. **Pagamentos e Transações Financeiras**

   Descrição Geral: Gerencia transações financeiras de forma segura, suportando métodos de pagamento como cartão de crédito e Pix, com histórico financeiro e emissão de recibos.

   Detalhes: Processamento seguro de pagamentos, registro detalhado de pedidos e transações, acompanhamento do status dos pedidos e acesso ao histórico financeiro com emissão de recibos. Integração com gateways de pagamento confiáveis.

Referências ao BD
Tabela `orders` (campos `payment_method`, `transaction_id`).

8. **Segurança e Privacidade de Dados**

   Descrição Geral: Implementa medidas robustas de segurança, incluindo criptografia de dados e controle de acesso, garantindo conformidade com a LGPD.

   Detalhes: Dados em trânsito protegidos via SSL/TLS, dados sensíveis criptografados no banco de dados, autenticação segura com validação de credenciais e políticas de conformidade com a LGPD, incluindo controle de acesso e confidencialidade das informações.

Referências ao BD
Tabela `users` (armazenamento seguro de senhas, controle de acesso).

9. **Integração com Sistemas Externos e APIs**

   Descrição Geral: Permite que empresas parceiras integrem seus sistemas à plataforma, facilitando a consulta de disponibilidade de produtos e serviços necessários aos pacientes.

   Detalhes: Fornecimento de chaves de API para autenticação segura, endpoints para produtos e pedidos fornecidos pelas empresas, e consulta de disponibilidade e preços em tempo real. Documentação e suporte para integração são disponibilizados às empresas.

Referências ao BD
Tabela `companies` (campos `api_key`, `products_endpoint`, `orders_endpoint`).

10. **Gestão de Documentos e Arquivos**

    Descrição Geral: Centraliza o armazenamento e gerenciamento de documentos médicos, garantindo segurança e acessibilidade para pacientes e profissionais.

    Detalhes: Upload e armazenamento seguro de documentos como exames, receitas e atestados. Compartilhamento controlado, onde profissionais com consultas recentes (menos de 3 meses desde a última ou 1 semana até a próxima) têm acesso aos documentos, garantindo conformidade com a LGPD.

Referências ao BD
Tabela `documents` (campos como `document_type`, `document_name`, `s3_object_key`).

11. **Backup e Recuperação de Dados**

    Descrição Geral: Assegura a integridade e disponibilidade dos dados por meio de backups automáticos e procedimentos de recuperação eficientes.

    Detalhes: Execução de backups periódicos do banco de dados e arquivos, armazenamento seguro em locais redundantes e procedimentos estabelecidos para restauração rápida em caso de falhas.

12. **Compatibilidade Multi-Plataforma**

    Descrição Geral: Garante acessibilidade em diversos dispositivos, proporcionando uma experiência consistente em computadores, tablets e smartphones.

    Detalhes: Versão web responsiva com design adaptável a diferentes tamanhos de tela, interface otimizada para usabilidade e desempenho, assegurando que todas as funcionalidades estejam disponíveis independentemente do dispositivo utilizado.

5.3.3 Endpoints

Os endpoints da API foram cuidadosamente projetados para atender às necessidades funcionais da plataforma, garantindo uma experiência segura, consistente e escalável para os usuários. A implementação da autenticação via JWT (JSON Web Token) assegura que apenas usuários autorizados possam acessar e manipular os recursos disponíveis, protegendo assim dados sensíveis e prevenindo acessos indevidos. A organização dos endpoints por tipo de serviço facilita a navegação e a manutenção da API, permitindo que desenvolvedores identifiquem rapidamente os recursos necessários para cada módulo do sistema. Além disso, a padronização dos métodos HTTP (como GET, POST e PUT) e das estruturas de URLs promove uma interface intuitiva e previsível, simplificando a integração e o desenvolvimento por parte dos usuários da API. A estrutura modular adotada possibilita a expansão futura da API com a adição de novos endpoints sem comprometer a organização existente, garantindo que a plataforma possa evoluir e se adaptar às demandas crescentes de maneira eficiente. A documentação detalhada, acompanhada da tabela a seguir, fornece uma referência clara e acessível para desenvolvedores e stakeholders, facilitando a compreensão e o uso eficaz dos recursos disponibilizados pela API. Resumimos os endpoints na seguinte tabela:

Tabela 18 – Lista de Endpoints por Serviço

Tipo de Serviço	Método HTTP e URL	Descrição Básica
Serviço de Agendamentos	POST /appointments	Cria uma nova consulta
	GET /appointments	Recupera uma lista de consultas
	POST /appointments/create-meeting/{appointment_id}	Atualiza detalhes de uma consulta específica
	PUT /appointments/{appointment_id}	Atualiza o status de uma consulta
	GET /appointments/{appointment_id}	Obtém detalhes de uma consulta específica
Serviço de Avaliações	POST /assessments	Cria uma nova avaliação
	GET /assessments/details/{assessment_id}	Obtém detalhes de uma avaliação específica
	GET /assessments/me	Lista as avaliações do usuário logado
	GET /assessments/patients/{patient_id}	Lista as avaliações de um paciente específico
Serviço de Disponibilidade	GET /availabilities	Recupera as disponibilidades atuais
	POST /availabilities	Define novas disponibilidades
Serviço de Documentos	POST /documents	Faz upload de um novo documento
	GET /documents/{document_id}	Obtém um documento específico
	GET /documents	Lista todos os documentos do usuário
Serviço de Pacientes	GET /patient/{patient_id}	Obtém informações de um paciente específico
Serviço de Planos	POST /plans	Cria um novo plano de tratamento
	GET /plans/me	Lista os planos do usuário logado
	GET /plans/patients/{patient_id}	Lista os planos de um paciente específico
Serviço de Produtos	POST /products	Adiciona um novo produto
	POST /products/fetch	Recupera informações de produtos
Serviço de Profissionais	GET /professional/{professional_id}	Obtém informações de um profissional específico
Serviço de Slots	GET /slots	Recupera os slots disponíveis
	PUT /slots/toggle/{slot_id}	Alterna o status de um slot específico
	GET /slots/me	Pega os slots do usuário
	GET /slots/professional/{professional_id}	Pega os slots do profissional
Serviço de Tipos	GET /typed/me	Obtém informações tipadas do usuário logado
Serviço de Usuários	POST /users	Cria um novo usuário
	POST /users/batch/patients	Cria múltiplos pacientes em lote
	POST /users/batch/professionals	Cria múltiplos profissionais em lote
	GET /users/professionals	Obtém informações do profissional logado
	GET /users/me	Obtém informações do usuário logado

Fonte: Elaborado com base nos endpoints disponíveis na API

5.3.4 Interfaces Gráficas da Plataforma

Como toda aplicação prática, é necessário um meio de comunicação entre os usuários e o sistema. As interfaces gráficas da plataforma foram desenvolvidas com foco na usabilidade e na experiência do usuário, proporcionando uma navegação intuitiva e eficiente para os diferentes perfis de usuários: Pacientes, Profissionais de Saúde e Empresas Parceiras. Cada tipo de usuário possui acesso a um conjunto específico de telas e funcionalidades, alinhadas às suas necessidades e objetivos dentro do sistema.

- **Paciente**

    1. **Início**: Apresenta um resumo dos próximos compromissos, acesso rápido aos planos de tratamento ativos e notificações recentes.
    2. **Agendar Atendimento**: Permite que o paciente agende consultas, filtrando por tipo de profissional e data, visualizando as opções disponíveis.
    3. **Meus Atendimentos**: Lista de consultas e exames agendados, com opções para visualizar detalhes, remarcar ou cancelar.
    4. **Opções de Compras**: Exibe planos de tratamento com opções de compra disponíveis, facilitando a aquisição de serviços e produtos.
    5. **Minhas Compras**: Permite o acompanhamento do status dos pedidos realizados e acesso ao histórico de transações.
    6. **Meu Histórico de Saúde**: Centraliza informações como avaliações recebidas, planos de tratamento e documentos médicos.
    7. **Mensagens**: Acesso à video-chamada para comunicação direta com profissionais de saúde.
    8. **Pagamentos**: Tela de checkout para finalizar compras e visualizar o histórico financeiro.

- **Profissional**

    1. **Minha Agenda**: Gerenciamento da disponibilidade, visualização de atendimentos agendados e configuração de horários.
    2. **Planos de Tratamento**: Ferramenta para criar e gerenciar planos de tratamento para os pacientes.
    3. **Avaliações**: Possibilita a criação e o gerenciamento de avaliações dos pacientes, tanto detalhadas quanto simplificadas.
    4. **Pacientes Atendidos**: Lista de pacientes já atendidos, com acesso ao histórico de consultas, avaliações e planos de tratamento.
    5. **Mensagens**: Acesso à video-chamada para comunicação direta com os pacientes.

6. **Prescrições**: Ferramenta para criar e emitir prescrições médicas.

- **Empresa**

1. **Pedidos**: Visualização e gerenciamento dos pedidos recebidos através da plataforma, com opções para atualizar o status.
2. **Integração API**: Informações e gerenciamento da integração via API com a plataforma, incluindo chaves de acesso e configuração de endpoints.
3. **Perfil da Empresa**: Permite a atualização dos dados da empresa e o gerenciamento de usuários associados.

5.4 Observações da Implementação: Segurança e Detalhes Práticos

5.4.1 Configuração de VPC e NAT Gateway

Para garantir a segurança e o isolamento dos recursos na nuvem, é recomendável que os serviços sejam implementados dentro de uma *Virtual Private Cloud* (VPC). Com isso, os recursos ficam protegidos dentro de uma rede virtual privada, e quando há necessidade de comunicação com serviços externos, utiliza-se um *NAT Gateway* para controlar o tráfego de saída.

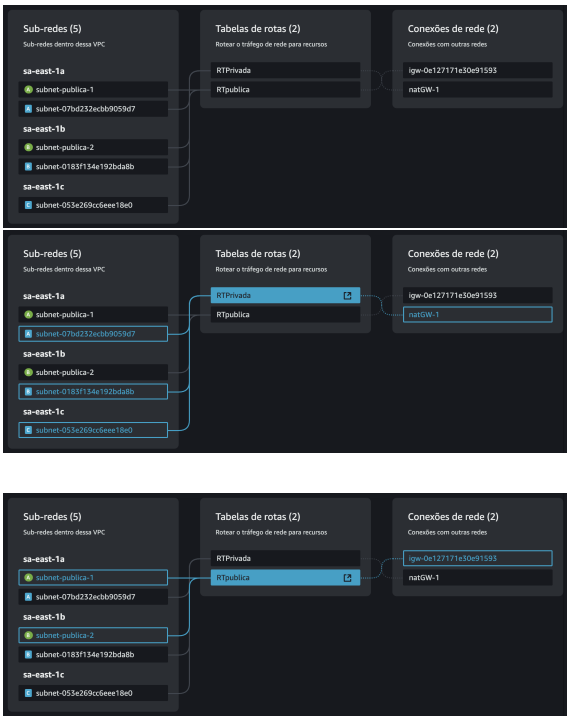
- **Configuração Ideal**

  Na configuração ideal, todos os recursos, como funções Lambda e bancos de dados, estariam dentro de uma VPC. O *NAT Gateway* permitiria que os recursos na VPC se comunicassem com a internet, quando necessário, como ao acessar o AWS Location Service ou endpoints de parceiros externos. Isso garantiria um nível adicional de segurança, isolando os recursos em sub-redes privadas e controlando rigorosamente o tráfego de entrada e saída. Além disso, a configuração de rotas apropriadas e grupos de segurança permitiria definir regras específicas para comunicação, minimizando a exposição a possíveis ameaças externas. Essa abordagem também facilitaria a conformidade com políticas de segurança e regulamentos, fornecendo uma arquitetura mais robusta e segura para a aplicação.

- **Configuração Atual**

  Devido a restrições e considerações práticas, **optou-se por manter as funções Lambda fora da VPC**. Isso permite que as Lambdas tenham acesso direto à internet sem a necessidade de um *NAT Gateway*, **simplificando a arquitetura e reduzindo custos**. No entanto, o Amazon RDS permanece dentro da VPC para garantir o controle de acesso e a segurança dos dados do banco.

Figura 19 – Configuração ideal da estrutura de Sub-redes na VPC - Redes privadas e públicas



5.4.2 Uso de Lambda e Step Functions

As AWS Step Functions fornecem uma forma de orquestrar serviços AWS, permitindo a definição de fluxos de trabalho através de máquinas de estados. Elas podem coordenar a execução de funções Lambda e outros serviços, adicionando lógica condicional e tratamento de erros. Idealmente, precisamos fazer o uso desse serviço.

- **Considerações sobre Pacientes e Empresas**

  Ao registrar pacientes e empresas, é necessário processar endereços utilizando o AWS Location Service para validar e enriquecer os dados. Atualmente, essa validação é feita dentro da própria função Lambda que cria o usuário, o que aumenta a latência (cerca de 2 segundos). Com o uso de Step Functions, **seria possível desacoplar essa etapa**, permitindo que a função de criação de usuário retorne mais rapidamente e que o processamento de endereço seja realizado de forma assíncrona por outra função Lambda.

- **Disponibilidade de Profissionais**

  Para profissionais, há a necessidade de criar disponibilidades (slots) que pertencem a outro microsserviço. Atualmente, ao criar um profissional, uma mensagem é enviada para uma fila Amazon SQS, que aciona uma função Lambda responsável por criar as disponibilidades. Embora essa abordagem funcione, as funções Lambda precisam

estar cientes da existência das filas SQS. Com as Step Functions, **seria possível orquestrar essas chamadas de forma mais transparente e eficiente**.

- **Vantagens das Step Functions**

  As Step Functions permitem que funções Lambda sejam executadas sequencialmente ou em paralelo, e que os resultados sejam passados entre elas. Isso pode **reduzir o tempo de execução total e os custos**, já que cada função Lambda é executada apenas pelo tempo necessário, sem aguardar o processamento de outras funções. Além disso, **facilita o tratamento de erros e a implementação de lógica complexa** sem sobrecarregar as funções individuais.

5.4.3 Segurança e Proteção de Dados

A segurança dos dados e a conformidade com a LGPD (Lei Geral de Proteção de Dados) são aspectos críticos do sistema. Foram implementadas várias medidas para proteger os dados dos usuários, incluindo autenticação robusta, criptografia e uso de protocolos seguros.

- **Autenticação JWT e AWS Cognito**

  O AWS Cognito é o serviço responsável por autenticar os usuários e proteger as nossas APIs. Quando um usuário é criado, seus dados como nome, e-mail e senha são passados para o Cognito, juntamente com *claims* personalizadas como `user_type` (tipo de usuário) e `user_id` (identificador único do usuário). Com essa abordagem, **não é necessário salvar a senha nem o e-mail no banco de dados**, já que a verificação das credenciais é feita pelo Cognito. Além disso, **eliminar o armazenamento da senha no banco de dados melhora a segurança e reduz o tempo de processamento durante o cadastro**, pois evita o cálculo do hash da senha.

  Ao efetuar o login, o Cognito verifica se o e-mail e a senha correspondem e retorna três tokens JWT (JSON Web Tokens), além das *claims*. O *ID Token* é um token criptografado pelo Cognito que contém as *claims* do usuário e é utilizado como *Bearer Token* nas requisições às APIs protegidas.

  No *front-end*, esses dados são armazenados no *localStorage* do navegador para que o usuário não precise realizar login novamente caso feche a janela. Também são mantidos em um objeto `user` dentro do nosso *AuthContext*, permitindo que as informações do usuário sejam acessadas pelos componentes internos, determinando o que ele pode visualizar e facilitando o envio de requisições autenticadas.

  O *AuthContext Provider* gerencia o estado de autenticação no React:

```
import { createContext, useContext, useState } from 'react';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState(() => {
    const storedUser = localStorage.getItem('user');
    return storedUser ? JSON.parse(storedUser) : null;
  });

  const login = (authData) => {
    const { tokens, userDetails } = authData;

    localStorage.setItem('user', JSON.stringify({ tokens, userDetails
        }));
    setUser({ tokens, userDetails });
  };

  const logout = () => {
    localStorage.removeItem('user');
    setUser(null);
  };

  const value = {
    isLoggedIn: !!user,
    user: user ? user.userDetails : null,
    tokens: user ? user.tokens : null,
    login,
    logout,
  };

  return <AuthContext.Provider
      value={value}>{children}</AuthContext.Provider>;
};

export const useAuth = () => useContext(AuthContext);
```

Listing 5.1 – AuthContext Provider

As rotas protegidas utilizam esse contexto para verificar a autenticação e o tipo de usuário antes de permitir o acesso. O roteador que protege as páginas internas está configurado da seguinte forma:

```
import { Navigate, Outlet } from 'react-router-dom';
import { useAuth } from '../context/AuthContext';

const ProtectedRoute = ({ allowedUserType }) => {
  const { user, isLoggedIn } = useAuth();

  if (!isLoggedIn || user.user_type !== allowedUserType) {
    return <Navigate to="/" replace />;
  }

  return <Outlet />;
};
```

```
13
14      export default ProtectedRoute;
```

Listing 5.2 – Roteador Protegendo Páginas Internas

E o roteador definindo as páginas protegidas é configurado da seguinte maneira:

```
1   import { createBrowserRouter, createRoutesFromElements, Route } from
        'react-router-dom';
2   import MainLayout from '../layouts/MainLayout';
3   import Home from '../pages/Home';
4   import ProtectedRoute from './ProtectedRoute';
5   import ErrorPage from '../pages/ErrorPage';
6   import PatientRoutes from './patient/PatientRoutes';
7   import ProfessionalRoutes from './professional/ProfessionalRoutes';
8   import CompanyRoutes from './company/CompanyRoutes';
9
10  const routes = createRoutesFromElements(
11    <Route path="" element={<MainLayout />}>
12      <Route path="/" element={<Home />} />
13      <Route element={<ProtectedRoute allowedUserType="patient" />}>
14        {PatientRoutes()}
15      </Route>
16      <Route element={<ProtectedRoute allowedUserType="professional" />}>
17        {ProfessionalRoutes()}
18      </Route>
19      <Route element={<ProtectedRoute allowedUserType="company" />}>
20        {CompanyRoutes()}
21      </Route>
22      <Route path="*" element={<ErrorPage />} />
23    </Route>
24  );
25
26  const router = createBrowserRouter(routes);
27
28  export default router;
```

Listing 5.3 – Definição de Rotas Protegidas

Um exemplo de componente que utiliza o *AuthContext* para obter os dados do usuário e fazer requisições é a página inicial do paciente:

```
1   import { Box, Heading, Text } from '@chakra-ui/react';
2   import { useAuth } from '../../context/AuthContext';
3   import { useFetchTypedUserData } from '../../hooks/useAccount';
4   import Loading from '../../components/Loading/Loading';
5
6   export default function Home() {
7     const { user, tokens } = useAuth();
8     const fetchTypedUserData = useFetchTypedUserData(
9       user?.user_type,
10      tokens?.idToken
11    );
12
13    if (fetchTypedUserData.isPending || !fetchTypedUserData.data) return
          <Loading />;
```

```
14
15        const data = fetchTypedUserData.data;
16
17        const {
18          name,
19          email,
20          birthday,
21          gender,
22          height,
23          weight,
24          food_restrictions,
25        } = data;
26        const address = {
27          street: data.street,
28          house_number: data.house_number,
29          neighborhood: data.neighborhood,
30          city: data.city,
31          state: data.state,
32          zip_code: data.zip_code,
33          country: data.country,
34        }
35        const genderMapping = {
36          male: 'Masculino',
37          female: 'Feminino',
38          other: 'Outro',
39        };
40
41        return (
42          <Box maxW="container.md" mx="auto" mt={8} p={6} borderWidth="1px"
                borderRadius="lg" shadow="md">
43            <Heading size="lg" mb={4}>
44              Perfil do Paciente
45            </Heading>
46            <Text><strong>Nome:</strong> {name}</Text>
47            <Text><strong>Email:</strong> {email}</Text>
48            <Text><strong>Data de Nascimento:</strong> {birthday}</Text>
49            <Text><strong>G nero:</strong> {genderMapping[gender] ||
                  gender}</Text>
50            <Text><strong>Altura:</strong> {height} cm</Text>
51            <Text><strong>Peso:</strong> {weight} kg</Text>
52            <Text><strong>Restri  es Alimentares:</strong>
                  {food_restrictions || 'Nenhuma'}</Text>
53            {address && (
54              <>
55                <Text mt={4}><strong>Endere o:</strong></Text>
56                <Text>Rua: {address.street}, N mero:
                      {address.house_number}</Text>
57                <Text>Bairro: {address.neighborhood}, Cidade:
                      {address.city}</Text>
58                <Text>Estado: {address.state}, CEP: {address.zip_code}</Text>
59                <Text>Pa s: {address.country}</Text>
60              </>
61            )}
62          </Box>
63        );
64      }
```

Listing 5.4 – Página Inicial do Paciente

No *back-end*, os endpoints protegidos pelo Cognito recebem o JWT já decodificado, garantindo a autenticidade do `user_type` e `user_id`. Alterar o token enviado invalidaria o login, pois o token é assinado pelo Cognito. O código a seguir mostra como extrair as *claims* do evento recebido pela função Lambda:

```
    def lambda_handler(event, context):
        claims = get_claims(event)
        user_id = claims.get("custom:user_id")
        user_type = claims.get("custom:user_type")
        # L gica da fun   o

    def get_claims(event):
        request_context = event.get('requestContext', {})
        authorizer = request_context.get('authorizer', {})
        jwt = authorizer.get('jwt', {})
        claims = jwt.get('claims', {})
        return claims
```

Listing 5.5 – Extração das Claims no Lambda

Os tokens JWT possuem um prazo de validade; após esse período, é necessário que o usuário realize o login novamente para obter novos tokens. Isso adiciona uma camada de segurança adicional, garantindo que o acesso seja revogado após o tempo especificado.

Essa integração com o AWS Cognito e o uso de tokens JWT asseguram que não só apenas usuários autenticados possam acessar os recursos do sistema, mas também que não possa se fingir ser outro tipo de usuário ou outro usuário do mesmo tipo, respeitando as políticas de segurança e conformidade com a LGPD (Lei Geral de Proteção de Dados).

- **Segurança dos Dados e Conformidade com a LGPD**

  Além da autenticação e autorização, a segurança dos dados é reforçada através da criptografia em trânsito e em repouso. O uso de HTTPS é implementado tanto no front-end quanto no back-end. No front-end, o Amazon Route 53 e o Amazon CloudFront são configurados para fornecer certificados SSL e distribuir o conteúdo de forma segura. No back-end, as funções Lambda e o API Gateway garantem que as requisições sejam criptografadas.

  Os dados armazenados no Amazon RDS são criptografados em repouso, utilizando o AWS KMS (Key Management Service). Isso assegura que dados sensíveis, como informações pessoais dos usuários, estejam protegidos mesmo em caso de acesso não autorizado ao banco de dados.

Por fim, é importante mencionar que, embora medidas técnicas sejam implementadas para proteger os dados e garantir a segurança, é necessário também atender aos requisitos legais da LGPD, o que inclui a elaboração de termos de uso e políticas de privacidade claras, informando aos usuários como seus dados serão utilizados e protegidos.

5.4.4 Videochamadas com WebRTC e AWS Chime

Para implementar o recurso de videochamadas entre pacientes e profissionais, foi utilizado o Amazon Chime SDK, que facilita a criação de aplicações com recursos de comunicação em tempo real.

Ao iniciar uma consulta, uma função Lambda é acionada para criar uma reunião no Amazon Chime utilizando o ID da consulta e adicionar o usuário como participante. Essa função retorna os dados necessários para que o front-end possa ingressar na reunião. No entanto, esse processo pode introduzir uma latência perceptível (cerca de 2 segundos) e, se o paciente ou profissional tentar ingressar na consulta após um período (por exemplo, 15 minutos), poderá ser criada uma nova reunião, impedindo a comunicação entre eles.

Uma solução proposta é automatizar a criação das reuniões utilizando o Amazon EventBridge, agendando a criação antecipada das reuniões e armazenando os detalhes no banco de dados. Dessa forma, ao ingressar na consulta, os usuários obteriam os dados pré-existentes, reduzindo a latência e garantindo que ambos acessem a mesma reunião.

É importante notar que a função Lambda responsável pela integração com o Amazon Chime precisa ser implantada na região `us-east-1`, pois esse serviço não está disponível na região `sa-east-1`.

5.4.5 Considerações da Implementação

As escolhas de implementação refletem um equilíbrio entre **segurança, performance e custos**. A utilização dos serviços AWS e a adoção de boas práticas de desenvolvimento garantem que o sistema seja **robusto, seguro e escalável**, atendendo aos requisitos funcionais e não funcionais estabelecidos.

5.5 Resultados Práticos do Projeto

Com o final do desenvolvimento do sistema multissetorial de saúde em nuvem, o produto final foi uma plataforma digital que funciona corretamente de acordo com os requisitos estabelecidos anteriormente. Tem-se uma aplicação na internet que integra diferentes serviços de saúde em um ambiente único em que é possível vários usuários ao mesmo tempo. Os resultados práticos obtidos ao final do desenvolvimento reforçam

a adequação das escolhas arquiteturais, tecnológicas e estratégicas realizadas durante o projeto. A aplicação de princípios de microsserviços, a utilização de serviços em nuvem e o foco na segurança e conformidade com a LGPD resultaram em uma solução sólida, capaz de lidar com as demandas do setor de saúde de maneira escalável e flexível.

Para ilustrar como essas decisões se traduziram em resultados tangíveis, destacamos alguns pontos:

- **Integração Eficiente de Serviços**: A arquitetura em microsserviços, aliada ao uso de AWS Lambda, S3, EC2 e RDS, permitiu a comunicação fluida entre componentes, assegurando alta disponibilidade e reduzindo complexidades na integração.
- **Escalabilidade e Flexibilidade**: O provisionamento dinâmico de recursos na nuvem garante que o sistema possa lidar com flutuações na demanda, ajustando-se rapidamente e mantendo o desempenho, independentemente do número de usuários ou operações simultâneas. Isso se deve majoritariamente por conta do serviço de AWS Lambda que escala automaticamente o provisionamento de mais computação conforme o uso do sistema.
- **Experiência do Usuário Aprimorada**: O uso do React no front-end assegurou uma interface intuitiva, responsiva e acessível em diferentes dispositivos, fortalecendo a adesão dos usuários e facilitando interações com serviços de agendamento, telemedicina e acompanhamento de tratamentos. O principal diferencial aqui é que não se tem limitação de dispositivo, podendo ser utilizado por qualquer um em qualquer lugar pela internet.
- **Segurança e Conformidade**: A implementação de protocolos de autenticação e autorização, o uso de criptografia e o atendimento aos requisitos da LGPD proporcionam um ambiente confiável, preservando dados sensíveis e conquistando a confiança de pacientes, profissionais de saúde e parceiros.
- **Adaptação ao Setor de Saúde**: A plataforma final demonstra capacidade de absorver novas funcionalidades, integrar serviços adicionais e se adaptar às constantes evoluções tecnológicas e regulatórias do setor, assegurando longevidade e relevância do produto. Achamos que vale destacar aqui o fato de que a transparência foi perfeitamente implementada e é possível testar na prática no sistema desenvolvido.

6 Conclusão

Como parte final do projeto proposto, o desenvolvimento do sistema multissetorial de saúde em nuvem representou um esforço significativo para integrar diferentes serviços de saúde em uma única plataforma, visando melhorar a experiência do paciente e a eficiência dos profissionais de saúde. Ao longo deste projeto, diversos aprendizados e insights foram obtidos, contribuindo para a evolução tecnológica no setor de saúde. Finalizamos o trabalho aqui com perspectivas e observações importantes.

6.1 Conclusões do Projeto

Com base nos resultados do projeto, as seguintes conclusões foram reunidas pelos desenvolvedores:

1. **Features devem ser alinhadas de acordo com os custos.**
   Ao longo do desenvolvimento, foi evidente que a implementação de funcionalidades deve ser cuidadosamente balanceada com os custos associados. A adoção de tecnologias em nuvem e serviços como AWS Lambda e S3 proporcionou escalabilidade e flexibilidade, mas também exigiu um planejamento financeiro detalhado para garantir a sustentabilidade do projeto. Priorizar features que agregam maior valor aos usuários, enquanto se gerencia eficientemente os recursos, foi fundamental para o sucesso do sistema.

2. **Escalabilidade é crucial para o crescimento sustentável.**
   A capacidade de escalar o sistema para atender a um número crescente de usuários sem comprometer o desempenho foi um dos principais objetivos alcançados. A arquitetura baseada em microsserviços permitiu que componentes individuais fossem dimensionados de forma independente, garantindo a responsividade e disponibilidade do sistema mesmo sob alta demanda.

3. **Segurança e Conformidade são indispensáveis na área da saúde.**
   A proteção dos dados sensíveis dos pacientes e a conformidade com a LGPD foram prioridades desde o início. Implementações robustas de criptografia, autenticação e autorização, utilizando ferramentas como AWS Cognito e KMS, asseguraram que o sistema atendesse aos requisitos legais e mantivesse a confiança dos usuários.

4. **Integração eficaz entre os stakeholders melhora a eficiência.**
   Facilitar a comunicação e a troca de informações entre pacientes, profissionais de saúde e empresas parceiras mostrou-se vital para otimizar processos e melhorar

a qualidade dos serviços prestados. A plataforma atuou como um hub central, permitindo interações mais fluidas e colaborativas.

5. **Desafios na comunicação entre microsserviços requerem soluções robustas.** A complexidade inerente à comunicação entre múltiplos microsserviços destacou a necessidade de mecanismos eficientes de orquestração e monitoramento. Investimentos em tecnologias de gestão de APIs e filas de mensagens foram essenciais para garantir a consistência e a integridade dos dados.

6. **Experiência do usuário é determinante para a adoção da plataforma.** Uma interface amigável e intuitiva foi crucial para engajar os usuários e facilitar a adoção do sistema. O uso de frameworks modernos como React no front-end contribuiu para uma experiência de usuário satisfatória, impactando positivamente na percepção e utilização da plataforma.

7. **Adaptabilidade às mudanças é essencial no setor de saúde.** O ambiente de saúde é dinâmico e sujeito a constantes evoluções tecnológicas e regulatórias. A arquitetura flexível do sistema permite a incorporação rápida de novas funcionalidades e adaptações necessárias, garantindo a relevância contínua da plataforma.

6.2 Contribuições

O sistema multissetorial de saúde em nuvem apresentou contribuições significativas para a área da saúde digital:

6.3 Perspectivas de Continuidade

O potencial para expandir e aprimorar o sistema é vasto, com diversas oportunidades para agregar valor adicional aos usuários e parceiros.

6.3.1 Expansão do Sistema com Novos Microsserviços

Planeja-se desenvolver novos microsserviços que possam, por exemplo:

- **Telemedicina Avançada**: Incorporar recursos como consultas por realidade virtual ou aumentada.
- **Gerenciamento de Saúde Populacional**: Analisar dados agregados para identificar tendências e auxiliar em políticas de saúde.

- **Marketplace de Serviços de Saúde**: Expandir as opções de serviços e produtos disponíveis, conectando usuários a uma rede ainda maior de parceiros.

6.3.2 Melhoria da Segurança e Conformidade

Continuar investindo na segurança e conformidade é essencial:

- **Implementação de Blockchain**: Explorar o uso de blockchain para aumentar a segurança e a transparência nas transações e no compartilhamento de dados.
- **Certificações Internacionais**: Buscar certificações como ISO 27001 para fortalecer a credibilidade do sistema globalmente.
- **Atualizações Regulatórias**: Manter-se atualizado com mudanças nas legislações, garantindo a contínua conformidade e proteção dos usuários.

6.4 Observações Adicionais

Durante o desenvolvimento do sistema multissetorial de saúde em nuvem, consideramos os insights dos estudos de (GUTTMAN et al., 2018) e (INTERNATIONAL, 2018), que destacam a importância crucial da comunicação eficaz para garantir a segurança do paciente e a qualidade do atendimento na área da saúde. Esses trabalhos identificam barreiras de comunicação, como as comportamentais, cognitivas, linguísticas, ambientais e tecnológicas, e propõem estratégias baseadas em evidências para superá-las, incluindo a implementação de protocolos padronizados e o uso de técnicas de comunicação eficazes.

Especificamente, (GUTTMAN et al., 2018) aborda as barreiras na comunicação entre profissionais de saúde, enquanto (INTERNATIONAL, 2018) explora os desafios na comunicação entre profissionais e pacientes, como transferências inadequadas de cuidado, deficiências linguísticas e culturais. Ambos os estudos oferecem soluções práticas, como o uso de intérpretes médicos, materiais educativos adaptados e protocolos de comunicação padronizados, que informaram o design e a funcionalidade do nosso sistema.

6.4.1 Conectando os Temas ao Trabalho Proposto

Ambos os trabalhos mencionados ressaltam que falhas na comunicação podem levar a eventos adversos graves, aumentando os riscos para a segurança do paciente e os custos operacionais. Ao entender as barreiras identificadas nesses estudos, implementamos soluções práticas dentro da plataforma:

- **Barreiras Comportamentais e Cognitivas**: Ao promover uma cultura de comunicação aberta e utilizar protocolos padronizados, como o SBAR (*Situation,*

Background, Assessment, Recommendation), nosso sistema facilita a troca de informações críticas entre profissionais de saúde, reduzindo a relutância em expressar preocupações e minimizando distrações.

- **Barreiras Linguísticas e Culturais**: A plataforma oferece suporte multilíngue e materiais educativos adaptados, atendendo às necessidades de pacientes com diferentes níveis de alfabetização em saúde. Isso está alinhado com as recomendações de (INTERNATIONAL, 2018) para acomodar necessidades de linguagem e superar barreiras culturais. Na prática, isso quer dizer que qualquer pessoa pode utilizar a plataforma.
- **Barreiras Ambientais e Tecnológicas**: Com a integração de recursos como telemedicina e comunicação assíncrona, mitigamos as limitações físicas e tecnológicas que podem dificultar a interação entre pacientes e profissionais. O uso de tecnologia segura e centrada no usuário atende às estratégias sugeridas por (GUTTMAN et al., 2018) para superar barreiras ambientais e tecnológicas.

Ao abordar diretamente as barreiras de comunicação destacadas na literatura, nosso sistema não apenas melhora a eficiência operacional, mas também contribui significativamente para a segurança do paciente e a qualidade do cuidado. A adoção de práticas baseadas em evidências fortalece a plataforma como uma solução robusta e confiável no setor de saúde.

6.5 Considerações Finais

Este projeto evidenciou a capacidade transformadora da tecnologia quando aplicada de forma estratégica e centrada no usuário. A criação de um sistema multissetorial de saúde em nuvem não apenas atende a uma necessidade atual do mercado, mas também estabelece as bases para inovações futuras que podem revolucionar a forma como os serviços de saúde são prestados e gerenciados. A continuidade deste trabalho promete contribuir ainda mais para o bem-estar da sociedade e para a evolução do setor de saúde como um todo.

Referências

ALSHUQAYRAN N., A. N. . E. R. *A Systematic Mapping Study in Microservice Architecture*. 2016. IEEE. Retrieved from IEEE Xplore. Disponível em: <https://doi.org/10.1109/SOCA.2016.15>. Acesso em: 21 oct 2024. Citado na página 13.

BALALAIE A., H. A. . J. P. *Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture*. 2016. IEEE. Retrieved from IEEE Xplore. Disponível em: <https://doi.org/10.1109/MS.2016.64>. Acesso em: 21 oct 2024. Citado na página 13.

GANG, L.; BADARCH, T. *Research on Characteristics and Technologies of Cloud Computing*. 2023. American Journal of Computer Science and Technology. Retrieved from Typeset.io. Disponível em: <https://doi.org/10.11648/j.ajcst.20230601.15>. Acesso em: 24 oct 2024. Citado na página 16.

GUTTMAN, O. T. et al. *Dissecting Communication Barriers in Healthcare: A Path to Enhancing Communication Resiliency, Reliability, and Patient Safety*. 2018. Journal Article. Published in Journal of Patient Safety. Disponível em: <https://doi.org/10.1097/PTS.0000000000000541>. Acesso em: 02 Dec 2024. Citado 2 vezes nas páginas 73 e 74.

INTERNATIONAL, J. C. *Communicating Clearly and Effectively to Patients: How to Overcome Common Communication Challenges in Health Care*. 2018. White Paper. Sponsored by Cambridge Boxhill Language Assessment. Disponível em: <https://store.jointcommissioninternational.org/assets/3/7/jci-wp-communicating-clearly-final_(1).pdf>. Acesso em: 02 Dec 2024. Citado 2 vezes nas páginas 73 e 74.

KHAN JESSICA S. ANCKER, J. L. D. K. C. H. A. C. R. K. S. A. *Developing a Web Platform for Health Promotion and Wellness Driven by and for the Harlem Community*. 2009. National Library of Medicine. Disponível em: <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2815482/>. Acesso em: 04 fev 2024. Citado na página 10.

LUGATI, L. N.; ALMEIDA, J. E. d. *Da evolução das legislações sobre proteção de dados: a necessidade de reavaliação do papel do consentimento como garantidor da autodeterminação informativa*. 2020. Journal Article. Retrieved from Typeset.io. Disponível em: <https://doi.org/10.32361/2020120210597>. Acesso em: 28 oct 2024. Citado na página 18.

MEYER, J.-U. *Open SOA Health Web Platform for Mobile Medical Apps*. 2014. IEEE. Disponível em: <https://doi.org/10.1109/ETFA.2014.7005347>. Acesso em: 04 fev 2024. Citado na página 10.

PENG, P. G. C.; BAI, G. *A literature review of current technologies on health data integration for patient-centered health management*. 2019. Sage Journals. Disponível em: <https://doi.org/10.1177/1460458219892387>. Acesso em: 04 fev 2024. Citado 2 vezes nas páginas 9 e 10.

RYAZANOVA, A. A. *Properties and Examples of the Digital Platforms*. 2021. Institute for Scientific & Technical Information of the RAS, Research Center for Cryptocurrencies and Digital Assets, Moscow, Russia. Retrieved from IEEE Xplore. Disponível em: <https://doi.org/10.1109/ElConRus51938.2021.9396485>. Acesso em: 21 apr 2024. Citado na página 12.

TIKU, S. *Financial Regulation and Technology*. 2022. Book Chapter, Chapter 3, Cloud Computing. Retrieved from Typeset.io. Disponível em: <https://doi.org/10.4337/9781802205411.00013>. Acesso em: 24 oct 2024. Citado na página 15.

WHO. *Obesity and Overweight*. 2021. Site da WHO. Disponível em: <https://www.who.int/news-room/fact-sheets/detail/obesity-and-overweight>. Acesso em: 04 fev 2024. Citado 2 vezes nas páginas 3 e 9.

WIERINGA, R. J. *Design Science Methodology for Information Systems and Software Engineering*. 1. ed. Heidelberg: Springer-Verlag Berlin Heidelberg, 2014. Citado na página 19.