**FIAP - Faculdade de Informática e Administração Paulista**

**Grupo 16:**

Christopher Svoboda

Renato Silveira dos Santos

**HACKATON**

CRMED

Porto Alegre

2024

**Repositório do Projeto MVP:**

**https://github.com/svobodachris/hackatonfiap**

# Diagrama de ComponentesDiagrama Descrição gerada automaticamente

# Diagrama de Sequência

**Diagrama

Descrição gerada automaticamente**

# AS-IS e TO-BE

**Diagrama

Descrição gerada automaticamente**

Todos os diagramas e imagens apresentados a seguir podem ser encontrados com melhor qualidade no seguinte caminho:

**https://github.com/svobodachris/hackatonfiap/tree/main/docs**

# Blueprint da Arquitetura de Solução do Sistema CRMED

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

# Arquitetura da Solução Cloud do Sistema CRMED

**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

# Arquitetura do Pipeline CI/CD do Sistema CRMED

**Diagrama

Descrição gerada automaticamente**

# CI/CD:

# 

# Azure API:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

# Azure APP:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

# Azure Grupo de Recursos:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

# Entregável MVP:

Como entregável de MVP criamos a API com os end-points implementados em Node.js.

**Repositório:** https://github.com/svobodachris/hackatonfiap

API CRMED com as operações do MVP:Aplicativo

Descrição gerada automaticamente com confiança média

Endpoint para Consulta de Médicos:Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Endpoint para Agendamento de Consulta:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Endpoint para Cancelamento de Agendamento de Consulta:

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

# Arquitetura Escolhida: Arquitetura Clean

Escolhemos essa arquitetura por ser a mais adequada para o CRMED por várias razões:

1. **Desacoplamento e Flexibilidade:** O CRMED envolve integração com serviços externos (como o Google Meet para a videochamada e a Azure Blob para armazenamento de documentos) e a necessidade de um núcleo de aplicação robusto que deve ser flexível para futuras alterações. A arquitetura Clean permite uma clara separação entre a lógica de negócios e os detalhes de implementação externa, facilitando alterações e integrações.
2. **Facilidade de Manutenção e Escalabilidade:** Com a Arquitetura Clean, é mais fácil manter e escalar o sistema, uma vez que as responsabilidades estão bem definidas e isoladas. Isso é crucial para o sistema de telemedicina CRMED possa evoluir com novas funcionalidades e requisitos ao longo do tempo.
3. **Testabilidade:** A Arquitetura Clean facilita o teste da lógica de negócios de forma independente dos componentes externos, garantindo que o núcleo do sistema funcione conforme esperado.
4. **Independência de Tecnologias:** O sistema pode evoluir tecnologicamente sem afetar a lógica de negócios central, uma vantagem importante para a integração com novos serviços ou tecnologias.

**A Arquitetura de Microserviços** pode ser considerada no futuro, especialmente para escalabilidade e desenvolvimento independente de funcionalidades, mas para um MVP e considerando a complexidade inicial, a Arquitetura Clean oferece um equilíbrio ideal entre flexibilidade, manutenção e escalabilidade.

# Componentes Principais:

* **Front-end (Interface do Usuário)**
  + **Paciente**
    - Tela de Login
    - Tela de Busca de Médicos
    - Tela de Agendamento de Consultas
    - Tela de Acesso ao Prontuário Eletrônico
  + **Médico**
    - Tela de Login
    - Tela de Cadastro/Edição de Horários
    - Tela de Aceite/Recusa de Consultas
    - Tela de Gerenciamento de Consultas
* **Back-end (Serviços e APIs)**
  + **Autenticação e Autorização**
    - Serviço de Autenticação (login usando CRM/CPF e senha)
    - Gerenciamento de Sessões e Tokens
  + **Gerenciamento de Consultas**
    - API para Cadastro/Edição de Horários Disponíveis
    - API para Aceite/Recusa de Consultas
    - API para Agendamento e Cancelamento de Consultas
  + **Tele Consulta**
    - Serviço de Geração de Links de Reunião que será pelo Google Meet
  + **Prontuário Eletrônico**
    - Serviço de Upload e Acesso a Documentos
    - Serviço de Compartilhamento de Documentos
* **Banco de Dados**
  + **Base de Dados de Usuários**
    - Informações de Médicos e Pacientes
  + **Base de Dados de Consultas**
    - Horários Disponíveis
    - Consultas Agendadas
  + **Base de Dados do Prontuário Eletrônico**
    - Documentos e Arquivos do Paciente
* **Infraestrutura**
  + **Serviços de Nuvem (Azure)**
    - Servidores para Front-end e Back-end
    - Armazenamento de Dados
    - Balanceadores de Carga
  + **CI/CD (Integração e Entrega Contínua)**
    - Pipeline de Deploy Automatizado
    - Ferramentas de Teste e Validação
* **Segurança**
  + **Camadas de Segurança**
    - Criptografia de Dados
    - Monitoramento de Segurança
    - Proteção contra-ataques (firewalls, DDoS e restrição do banco de dados na sub-rede da aplicação)

# Descrição de Como os Requisitos Não Funcionais Serão Atendidos

1. **Alta Disponibilidade**
   * Utilização de serviços da nuvem Azure com alta disponibilidade e balanceadores de carga para garantir que o sistema esteja disponível 24/7.
   * Implementação de backups regulares para prevenir perda de dados e garantir continuidade.
2. **Escalabilidade**
   * Serviços conteinerizados em Azure Kubernetes (AKS) que permite escalar instâncias (pods) dos componentes individualmente conforme a demanda.
   * Uso de serviços de nuvem que suportam escalabilidade horizontal para lidar com até 20.000 usuários simultâneos.
3. **Segurança**
   * Implementação de criptografia para proteger as informações dos pacientes.
   * Monitoramento contínuo e práticas de segurança como firewalls e proteção contra DDoS para garantir a integridade e segurança do sistema.

# Atendimento aos Requisitos Não Funcionais

**1. Alta Disponibilidade**

**Estratégias e Tecnologias:**

* **Infraestrutura de Nuvem:** Utilizamos serviços de nuvem Microsoft Azure, que oferece alta disponibilidade e redundância em sua infraestrutura. Este provedor garante que o sistema esteja operacional 24/7, com suporte a failover automático (que é quando um servidor assume o lugar de outro que ficou indisponível) e backup.
* **Balanceamento de Carga:** Implementação de balanceadores de carga (load balancers) para distribuir o tráfego entre múltiplas instâncias do servidor, garantindo que nenhuma instância fique sobrecarregada e que o serviço permaneça disponível mesmo em caso de falha de uma instância.
* **Criptografia de Dados:** Utilização de criptografia de dados sensíveis e confidenciais, em conformidade com as regras de LGPD (Lei Geral de Proteção de Dados).
* **Monitoramento e Alertas:** Implementação de sistemas de monitoramento disponibilizados pela própria Azure para detectar e alertar sobre falhas ou degradações no desempenho, permitindo uma rápida intervenção e manutenção proativa.

**2. Escalabilidade**

**Estratégias e Tecnologias:**

* **Conteinerização:** O sistema é projetado utilizando contêiners de serviços que possibilita escalarmos em mais pods individuais conforme necessário, sem afetar todo o sistema. Isso facilita a escalabilidade vertical e horizontal.
* **Escalabilidade Horizontal:** Uso de contêineres e orquestradores como o Kubernetes (AKS) para permitir a escalabilidade horizontal, onde novos contêineres podem ser adicionados para lidar com picos de demanda. A Azure permite setarmos as métricas de uso e escalar conforme a demanda.
* **Banco de Dados Escalável:** Optamos pela utilização de bancos de dados que suportem escalabilidade horizontal e vertical. Ainda, facilitando futuras melhorias após o MVP, definimos a escolha pelo uso do banco de dados NoSQL MongoDB, que proporciona escalabilidade fácil e flexível.
* **Cache e CDN:** Como vamos lidar com pico de 20.000 usuários simultâneos, teremos a Implementação de caching para reduzir a carga no banco de dados e melhorar o tempo de resposta. Uso de redes de entrega de conteúdo (CDN) para distribuir o conteúdo estático e melhorar o desempenho geral.

**3. Segurança**

**Estratégias e Tecnologias:**

* **Criptografia:** Implementação de criptografia AES (Advanced Encryption Standard) para dados em repouso e TLS (Transport Layer Security) para dados em trânsito. Isso garante a proteção dos dados sensíveis dos pacientes contra acessos não autorizados.
* **Criptografia do Banco de Dados:** Uso de criptografia do banco de dados diretamente no Azure Cosmo DB que é gerenciado pela Azure.
* **Autenticação e Autorização:** Uso de protocolos robustos de autenticação, como OAuth 2.0 e JWT (JSON Web Tokens), para garantir que apenas usuários autenticados e autorizados possam acessar determinadas funcionalidades e dados.
* **Proteção contra-ataques:** Implementação de firewalls e sistemas de proteção contra DDoS (Distributed Denial of Service) para defender o sistema contra ataques externos. Ferramentas de monitoramento e análise contínua são usadas para identificar e mitigar vulnerabilidades e atividades suspeitas.
* **Controle de Acesso e Permissões:** Definição de políticas rigorosas de controle de acesso e gerenciamento de permissões, garantindo que apenas usuários autorizados possam acessar e manipular dados específicos.
* **Desenvolvimento:** Implementação no pipeline de desenvolvimento a utilização do SonarQube e do CheckMarx para garantia da qualidade de software, segurança e para validação e tratamento do Top 10 das vulnerabilidades do Owasp.

Estas abordagens garantem que o sistema CRMED esteja preparado para operar de forma eficiente e segura, atendendo a todos os requisitos não funcionais estabelecidos e, principalmente, em conformidade com a LGPD (Lei Geral de Proteção de Dados) e com todas as legislações que regulam o tema Proteção de Dados em relação a Área da Saúde.

# Justificativas das Escolhas Técnicas

**1. Frontend (Interface do Usuário)**

**Escolha: Framework React.js**

* **Justificativa:**
  + **Desempenho:** React.js porque permite a construção de interfaces de usuário responsivas e performáticas.
  + **Componentização:** Facilita a criação de componentes reutilizáveis, o que melhora a manutenção e escalabilidade da aplicação.
  + **Comunidade e Suporte:** Possui uma grande comunidade consolidada, que também disponibiliza muitos recursos e bibliotecas.

**2. Back-end (Serviços e APIs)**

**Escolha: Node.js com Express**

* **Justificativa:**
  + **Desempenho e Escalabilidade:** Node.js tem como principal vantagem a sua alta performance e capacidade de lidar com grandes volumes de conexões simultâneas, o que é essencial para suportar até 20.000 usuários simultâneos acesso e utilizando o CRMED.
  + **Desenvolvimento Rápido:** Express.js que oferece um framework minimalista e flexível e que acelera o desenvolvimento de APIs.
  + **Integração com Front-end:** A combinação de Node.js com React.js facilita a criação da integração entre o back-end e front-end, também encurtando o tempo de codificação.

**3. Banco de Dados**

**Escolha: MongoDB**

* **Justificativa:**
  + **Modelo de Dados Flexível:** Visando a evolução do sistema CRMED, foi definido uso de MongoDB pela facilidade de armazenar dados heterogêneos em documentos BSON, pois ele não requer um esquema fixo.
  + **Escalabilidade e Flexibilidade:** Por ser um sistema que precisa escalar horizontalmente, MongoDB foi a escolha definida, pois permite alta flexibilidade na modelagem de dados.

**4. Infraestrutura**

**Microsoft Azure**

* **Justificativa:**
  + **Alta Disponibilidade e Escalabilidade:** O serviço na nuvem Azure oferece serviços robustos para alta disponibilidade, escalabilidade e balanceamento de carga.
  + **Serviços Gerenciados:** Oferece serviços gerenciados que simplificam a administração de infraestrutura, como Azure Cosmos DB para bancos de dados e Azure Blob Storage para armazenamento de arquivos.
  + **Segurança:** Possui camadas de segurança avançadas e certificações que atendem a normas rigorosas de proteção de dados.

**5. CI/CD / Infra**

**Escolha: GitHub**

* **Justificativa:**
  + **Controle de Versões:** O GitHub permite que desenvolvedores colaborem e façam mudanças em projetos compartilhados enquanto mantêm um registro detalhado do progresso.
  + **Colaboração:** Facilita a colaboração entre membros de equipe, permitindo que trabalhem juntos em projetos.
  + **Gerenciamento Preciso de Versões:** Ajuda a controlar versões e alterações em aplicativos.
  + **Recursos de IA:** Oferece sugestões em códigos e busca por erros ou falhas de segurança.

**Escolha: GitHub Actions**

* **Justificativa:**
  + **Automatização e Eficiência:** Ferramenta de CI/CD que automatiza o processo de build, testes e deploy, reduzindo erros manuais e acelerando o ciclo de desenvolvimento.
  + **Integração com Repositórios:** O GitHub Actions integra-se com o próprio controle de versão do GitHub, a plataforma de gerenciamento de código escolhida.

**Escolha: Infraestrutura – Terraform**

* **Justificativa:**

O Terraform simplifica a criação e o gerenciamento de recursos de infraestrutura, tornando o provisionamento mais ágil e consistente. É uma ferramenta essencial para equipes ágeis. Algumas das razões pela utilização do Terraform:

* **Provisionamento de Infraestrutura:** O Terraform é útil para criar e configurar recursos como servidores, redes, bancos de dados e balanceadores de carga.
* **Orquestração de Recursos em Nuvem:** Ele facilita a gestão de recursos em várias plataformas de nuvem, como a Microsoft Azure.
* **Gestão de Infraestrutura em Múltiplos Ambientes:** O Terraform permite gerenciar infraestrutura em ambientes híbridos ou multinuvem.
* **Automação e Orquestração de Implantações:** Com o Terraform, é possível automatizar a implantação e orquestrar mudanças na infraestrutura.
* **Gerenciamento de Estado de Infraestrutura:** Ele mantém um registro do estado atual da infraestrutura, facilitando atualizações e manutenção.

**Escolha: Qualidade de Código – SonarQube**

* **Justificativa:**

O SonarQube é uma plataforma de código aberto que permite a inspeção contínua da qualidade do código e integra-se facilmente aos pipelines de CI/CD.

Ele auxilia os desenvolvedores a detectar e corrigir problemas como code smells, erros e vulnerabilidades de segurança em suas bases de código. Além disso, o SonarQube realiza análises de cobertura, examinando os testes unitários do projeto e novos testes para entradas de código adicionais.

Essa ferramenta é essencial para manter o código limpo, fácil de ler e seguro, contribuindo para a qualidade contínua do software.

Também está relacionado com o OWASP (Open Web Application Security Project):

* **Cobertura de Vulnerabilidades OWASP:** O SonarQube oferece uma visão detalhada das vulnerabilidades de segurança, incluindo os 10 principais riscos do OWASP. Ele ajuda a identificar problemas como code smells, erros e vulnerabilidades em seu código.
* **Análise Contínua:** O SonarQube monitora constantemente o código, garantindo que ele esteja em conformidade com os padrões de segurança, incluindo o OWASP Top 10.
* **Relatórios de Segurança:** Ele fornece relatórios dedicados para rastrear a segurança do projeto em relação aos padrões OWASP e CWE Top 25.
* **Análise SAST:** O SonarQube identifica padrões no código-fonte que podem levar a problemas de controle de acesso, como verificações de autenticação ausentes ou configuração inadequada de controles baseados em funções.
* **Taint Analysis:** O SonarQube rastreia dados não confiáveis ​​no fluxo de execução, ajudando a evitar comprometimentos de segurança.

**Escolha: Segurança de Código – CheckMarx**

* **Justificativa:**

O Checkmarx é uma ferramenta essencial para garantir a qualidade e segurança do código. Ele oferece análise estática, dinâmica e de composição de software, identificando vulnerabilidades e fornecendo orientações de correção. Além disso, integra-se facilmente aos pipelines de CI/CD, promovendo a segurança contínua durante o desenvolvimento.

**6. Segurança**

**Escolha: Criptografia AES e TLS / Implementação de Firewall e Proteção contra DDoS**

* **Justificativa:**
  + **Criptografia:** AES (Advanced Encryption Standard) para criptografia de dados em repouso e TLS (Transport Layer Security) para criptografia de dados em trânsito garantem a proteção das informações sensíveis dos pacientes.
  + **Proteção contra-ataques:** Firewalls e medidas de proteção contra DDoS ajudam a proteger a aplicação contra acessos não autorizados e ataques de negação de serviço.

**Escolha: Key Vault**

* **Justificativa:**

O Azure Key Vault é um serviço na nuvem que permite armazenar e gerenciar chaves criptográficas e outros segredos de maneira segura. Ele é usado para proteger dados confidenciais, como senhas e certificados SSL/TLS, e oferece recursos como HSMs validados pelo padrão FIPS 140-2, escalabilidade global e auditoria de uso.

**Escolha: Azure Security Center**

* **Justificativa:**

O Azure Security Center é uma ferramenta de gerenciamento de segurança que oferece insights sobre o estado de segurança de suas cargas de trabalho em nuvem híbrida. Ele ajuda a reduzir a exposição a ataques e a responder rapidamente a ameaças detectadas.

**7. Logs**

**Escolha: Azure Monitor**

* **Justificativa:**

O Azure Monitor é uma solução abrangente de monitoramento da Microsoft que coleta, analisa e responde a dados de monitoramento em ambientes de nuvem e locais. Ele oferece ferramentas para monitorar aplicativos, infraestrutura e rede, ajudando a transformar negócios, melhorar o desempenho e fortalecer a segurança. Com o Azure Monitor, você pode:

* **Monitorar em qualquer lugar:** Obtenha insights em ambientes híbridos e de nuvem.
* **Solucionar problemas sem esforço:** Identifique e resolva problemas rapidamente.
* **Impulsionar a inovação:** Use dados para otimizar recursos e tomar decisões informadas.
* **Habilitar a preparação empresarial:** Esteja preparado para situações críticas com alertas e respostas automatizadas.

**Escolha: Audit Logs**

* **Justificativa:**

Os Audit Logs no contexto da Azure são registros que fornecem informações detalhadas sobre eventos e atividades em serviços e recursos hospedados na plataforma. Esses logs ajudam a monitorar e auditar alterações, identificar problemas de segurança e rastrear ações realizadas por usuários e sistemas. Alguns tipos de logs disponíveis na Azure incluem:

* **Activity logs:** Registram eventos de controle e gerenciamento relacionados às operações CREATE, UPDATE e DELETE no Azure Resource Manager. Eles oferecem insights sobre as ações executadas em seus recursos na assinatura.
* **Azure Resource logs:** Fornecem dados frequentes sobre o funcionamento dos recursos gerenciados pelo Azure Resource Manager. Isso inclui informações sobre operações realizadas pelo próprio recurso.
* **Microsoft Entra ID reporting logs:** Relatam atividades de entrada de usuários e informações sobre gerenciamento de grupos. Esses logs são úteis para rastrear alterações em aplicativos, grupos, usuários, licenças e atributos de segurança personalizados.
* **Virtual machines and cloud services logs:** Capturam dados do sistema e logs em máquinas virtuais, como o serviço de log de eventos do Windows e o Syslog no Linux. Esses dados podem ser transferidos para uma conta de armazenamento de sua escolha.
* **Azure Storage Analytics:** Fornece métricas e informações de rastreamento para contas de armazenamento. Isso ajuda a analisar solicitações, identificar tendências de uso e diagnosticar problemas com sua conta de armazenamento.
* **Network security group (NSG) flow logs:** Exibem fluxos de entrada e saída com base em regras específicas, mostrando informações sobre tráfego IP através de um Grupo de Segurança de Rede.

**Escolha: Log Analytics**

* **Justificativa:**

O Log Analytics é uma ferramenta no portal do Azure que permite editar e executar consultas de log em dados armazenados no Azure Monitor. Com ele, você pode analisar registros, filtrar informações e visualizar resultados em gráficos. É útil para monitorar recursos, identificar tendências e criar alertas baseados em logs.

**8. Load Balancer**

**Escolha: Azure Load Balancer**

* **Justificativa:**

O Azure Load Balancer é uma ferramenta que distribui eficientemente o tráfego de rede de entrada entre um grupo de servidores ou recursos de backend. Ele opera na camada 4 do modelo OSI (Open Systems Interconnection) e serve como o ponto único de contato para os clientes. Aqui estão alguns pontos-chave sobre o Azure Load Balancer:

* **Balanceamento de carga:** O Azure Load Balancer distribui fluxos de entrada que chegam à frente do balanceador de carga para as instâncias do pool de backend. Isso é feito de acordo com regras de balanceamento de carga configuradas e sondas de integridade.
* **Tipos de balanceadores de carga:**
  + **Balanceador de carga público:** Fornece conexões de saída para máquinas virtuais dentro da sua rede virtual, traduzindo seus endereços IP privados para endereços IP públicos. É usado para equilibrar o tráfego da Internet para suas VMs.
  + **Balanceador de carga interno (privado):** Usado quando IPs privados são necessários apenas no frontend. Ele equilibra o tráfego dentro de uma rede virtual.
* **Cenários de uso:**
  + Balanceamento de tráfego interno e externo para máquinas virtuais do Azure.
  + Baixa latência e alto throughput.
  + Aumento da disponibilidade distribuindo recursos dentro e entre zonas.
  + Configuração de conectividade de saída para máquinas virtuais do Azure.
  + Monitoramento de recursos balanceados usando sondas de integridade.
  + Encaminhamento de portas para acessar máquinas virtuais por meio de endereços IP públicos e portas.
  + Suporte para balanceamento de carga IPv6.
* **Métricas e insights:** O Azure Load Balancer fornece métricas multidimensionais por meio do Azure Monitor, permitindo análises detalhadas de desempenho e saúde do serviço.

O Azure Load Balancer ajuda a escalar aplicativos e criar serviços altamente disponíveis, oferecendo baixa latência e alta escalabilidade, que será necessário para nossa aplicação em momentos de pico.

**9. Comunicação**

**Escolha: Azure Communication Services**

* **Justificativa:**

O Azure Communication Services oferece APIs de comunicação multicanal para adicionar voz, vídeo, chat, mensagens de texto/SMS, e-mail e muito mais a todas nossas aplicações. Com essas APIs, podemos criar experiências de comunicação personalizadas. Utilizaremos esse serviço para realizar a comunicação das marcações de consultas e seus aceites e cancelamentos.

**10. Imagem e versionamento de imagem**

**Escolha: Docker**

* **Justificativa:**

Docker é uma ferramenta essencial para desenvolvedores que desejam acelerar o desenvolvimento de aplicativos. Com o Docker, você pode criar, compartilhar, executar e verificar aplicativos em qualquer lugar, sem a necessidade de configurações de ambiente tediosas. Aqui estão alguns pontos-chave:

* **Contêineres:** O Docker usa contêineres para empacotar aplicativos e suas dependências. Isso oferece consistência e portabilidade, permitindo que os aplicativos sejam executados em qualquer ambiente, desde Kubernetes local até serviços em nuvem como Azure ACI.
* **Facilidade de uso:** O Docker simplifica o processo de criação de ambientes de desenvolvimento. Você pode criar imagens personalizadas e executar vários contêineres usando o Docker Compose.
* **Integração com ferramentas existentes:** O Docker funciona bem com outras ferramentas de desenvolvimento, como o Visual Studio Code, GitHub e GitHub Actions.

Dessa forma o Docker é uma ferramenta importante para nossa aplicação que rodará na nuvem e terá seu pipeline de CI/CD automatizado.

**Escolha: Docker Hub**

* **Justificativa:**

O Docker Hub é o destino definitivo para encontrar e compartilhar imagens e aplicativos de contêiner. Nele, encontramos conteúdo confiável de parceiros e da própria Docker. Será o repositório onde subiremos as imagines de nossas aplicações para aplicarmos nos nossos ambientes na Azure.

**11. Segurança de subrede**

**Escolha: Grupos de Segurança**

* **Justificativa:**

Os Grupos de Segurança de Rede do Azure são filtros de segurança que podem ser aplicados a recursos como máquinas virtuais e sub-redes. Funcionam como um “firewall virtual”, permitindo ou bloqueando o tráfego de rede com base em regras definidas. Cada regra especifica origem, destino, porta e protocolo. Podemos criar regras personalizadas para controlar o tráfego entre nossos recursos no Azure.

**Escolha: Subrede**

* **Justificativa:**

Uma sub-rede no Azure é uma divisão lógica de um espaço de endereço IP dentro de uma rede virtual. Todos os recursos do Azure implantados em uma rede virtual são organizados em sub-redes. Essas sub-redes permitem segmentar e gerenciar o tráfego de rede entre os recursos, conforme iremos necessitar para os componentes de nossa aplicação.

**12. Azure Blob**

**Escolha: Azure Blob**

* **Justificativa:**

O Azure Blob é um serviço em nuvem que oferece armazenamento para dados não estruturados, data lakes, computação de alto desempenho e aprendizado de máquina. Ele permite criar data lakes para análise e fornece armazenamento para aplicativos móveis e nativos de nuvem. Com várias camadas de armazenamento e gerenciamento automatizado de ciclo de vida, você pode armazenar grandes quantidades de dados de maneira econômica. Além disso, o Azure Blob dá suporte a estruturas de desenvolvimento populares, como Node.js que utilizaremos nos componentes de nossa aplicação. Necessitaremos desse serviço para o armazenamento de arquivos, como exames e laudo médicos, assim como para o controle de permissão para acesso e temporalidade.

**13. Serviços de Autenticação e Autorização**

**Escolha: Azure Policy**

* **Justificativa:** 
  + O Azure Policy é um serviço no Azure que permite criar, atribuir e gerenciar definições de política em seu ambiente. Ele ajuda a impor padrões organizacionais e avaliar a conformidade em escala.
  + Com o Azure Policy, você pode:
    - Garantir que sua equipe implante recursos do Azure apenas em regiões permitidas.
    - Aplicar consistentemente tags taxonômicas aos recursos.
    - Exigir que os recursos enviem logs de diagnóstico para um espaço de trabalho do Log Analytics.
  + As definições de política são escritas em formato JSON e podem ser agrupadas em iniciativas para simplificar a gestão.
  + O Azure Policy avalia recursos com base nas regras definidas e ajuda a manter a conformidade.

**Escolha: Microsoft Entra ID**

* **Justificativa:** 
  + O Microsoft Entra ID é uma solução de gerenciamento de identidades e acesso na nuvem.
  + Ele conecta funcionários, clientes e parceiros a aplicativos, dispositivos e dados.
  + Recursos incluem:
    - Acesso adaptável seguro: Autenticação forte e políticas de acesso adaptáveis com base em risco.
    - Experiência de usuário uniforme: Entrada fácil e rápida em ambientes de várias nuvens.
    - Gerenciamento de identidade unificado: Controle centralizado de identidades e acesso a aplicativos.
    - Integrações de aplicativos e SSO: Conecte sua equipe a aplicativos em qualquer lugar com logon único.

# Como os Requisitos Não Funcionais São Atendidos

1. **Alta Disponibilidade**
   * Utilização de serviços de nuvem com SLA (Service Level Agreement) alto e redundância geográfica para garantir disponibilidade contínua (configurável na Azure).
   * Implementação de estratégias de failover e recuperação de desastres para minimizar o tempo de inatividade.
2. **Escalabilidade**
   * Conteinerização e escalabilidade horizontal para suportar muitos usuários simultâneos.
   * Uso de load balancer na Azure que permite ajustar recursos automaticamente com base na demanda.
3. **Segurança**
   * Aplicação de melhores práticas de segurança, incluindo criptografia de dados, autenticação robusta e controle de acesso.
   * Monitoramento contínuo e análises de segurança para identificar e mitigar vulnerabilidades.

Essas justificativas ajudam a explicar as escolhas técnicas e como elas atendem aos requisitos do CRMED, ao mesmo tempo em que garantem a aplicação das melhores práticas de qualidade e segurança e escalabilidade.