Proposta de Solução SaaS de Manutenção Preditiva Industrial Baseada em IA na AWS

1. Introdução

1.1. Problemática

A indústria moderna enfrenta desafios significativos relacionados à manutenção de equipamentos críticos. Falhas inesperadas em componentes essenciais da linha de produção resultam frequentemente em paradas não planejadas, acarretando perdas substanciais de produtividade, aumento dos custos de manutenção corretiva e, em alguns casos, comprometendo a segurança operacional.[1] As estratégias tradicionais de manutenção, como a manutenção reativa (corrigir após a falha) e a manutenção preventiva (baseada em cronogramas fixos), mostram-se frequentemente ineficientes. A manutenção reativa leva a interrupções custosas e não planejadas, enquanto a manutenção preventiva pode resultar na substituição prematura de componentes ainda funcionais ou, inversamente, não evitar falhas que ocorrem antes do intervalo programado.[2] Este cenário evidencia a necessidade premente de abordagens mais inteligentes e proativas para a gestão da manutenção industrial, conforme o problema enfrentado pela empresa industrial descrita na challenge.

1.2. Solução Proposta: Plataforma SaaS de Manutenção Preditiva Baseada em IA

Para endereçar as limitações das abordagens tradicionais, propõe-se o desenvolvimento de uma plataforma de Manutenção Preditiva (PdM) baseada em Inteligência Artificial (IA) e Machine Learning (ML), entregue no modelo Software-as-a-Service (SaaS). A PdM utiliza dados coletados em tempo real de sensores acoplados aos equipamentos, juntamente com dados históricos, para prever potenciais falhas antes que elas ocorram.[1] O objetivo principal é otimizar os cronogramas de manutenção, intervir apenas quando necessário, reduzir significativamente o tempo de inatividade não planejado e prolongar a vida útil dos ativos industriais.[2]

A solução será uma plataforma SaaS multi-tenant (multilocatário), permitindo que múltiplos clientes industriais utilizem o serviço de forma segura e isolada, compartilhando a infraestrutura subjacente para otimizar custos e eficiência operacional.[7] A plataforma integrará coleta de dados de sensores IoT, processamento e armazenamento em nuvem, modelos avançados de IA/ML para análise preditiva (estimativa de Vida Útil Remanescente - RUL e classificação de falhas), dashboards interativos para visualização de dados e alertas automatizados.[1]

1.3. Contexto e Objetivos do Projeto

Este projeto é desenvolvido no âmbito do curso de graduação em Inteligência Artificial da FIAP, respondendo a challenge proposta. A Challenge simula um cenário onde a Hermes Reply, empresa especializada em soluções para a Indústria 4.0 com expertise em IoT, IA e ML (Transformação Digital e Gerenciamento de Processos e Negócios), desenvolve esta plataforma SaaS.

Os objetivos centrais do projeto, alinhados aos objetivos da challenge, são:

1. Desenvolver uma solução de software SaaS para análise preditiva de manutenção de equipamentos industriais.
2. Utilizar dados de sensores em tempo real (coletados via ESP32).
3. Implementar uma arquitetura SaaS multi-tenant, garantindo segurança e isolamento de dados.
4. Processar dados continuamente para detecção de anomalias e previsão de falhas.
5. Aplicar técnicas avançadas de análise preditiva e ML (regressão, séries temporais, classificação, redes neurais).
6. Criar dashboards interativos para visualização em tempo real do status dos equipamentos, previsões e recomendações.
7. Implementar alertas automatizados (e-mail, SMS, etc.) sobre falhas iminentes.
8. Demonstrar a integração com dispositivos IoT (ESP32) para coleta de dados.
9. Gerar relatórios inteligentes e automatizados com insights e recomendações operacionais.

O projeto será documentado em um repositório GitHub privado e culminará em um protótipo funcional da plataforma SaaS, incluindo o software, dashboard e relatórios (“Entregáveis”).

2. Entrega 1: Design Fundamental - Metodologia, Tecnologias e Conceito de Pipeline

2.1. Metodologia Adotada: CRISP-DM

Para estruturar o desenvolvimento deste projeto de Machine Learning, adota-se a metodologia CRISP-DM (Cross-Industry Standard Process for Data Mining).[10] Esta é uma metodologia amplamente reconhecida e robusta para projetos de ciência de dados e ML. O CRISP-DM organiza o projeto em seis fases principais:

1. **Business Understanding (Entendimento do Negócio):** Foco nos objetivos do

projeto sob a perspectiva do negócio, definição de critérios de sucesso e tradução desses objetivos em um problema de mineração de dados.
2. **Data Understanding (Entendimento dos Dados):** Coleta inicial de dados, familiarização com os dados, identificação de problemas de qualidade e descoberta de insights iniciais.
3. **Data Preparation (Preparação dos Dados):** Seleção, limpeza, construção de features, integração e formatação dos dados brutos para torná-los adequados para as ferramentas de modelagem.
4. **Modeling (Modelagem):** Seleção e aplicação de técnicas de modelagem (algoritmos de ML), calibração de parâmetros e construção de modelos preditivos.
5. **Evaluation (Avaliação):** Avaliação dos modelos construídos sob a ótica dos objetivos de negócio e critérios de sucesso definidos. Determinação dos próximos passos.
6. **Deployment (Implantação):** Organização e apresentação dos resultados da modelagem de forma útil para o cliente/usuário final, incluindo o planejamento de monitoramento e manutenção.

Embora as fases sejam apresentadas sequencialmente, o CRISP-DM é fundamentalmente um processo iterativo.[10] É comum e necessário retornar a fases anteriores à medida que novos conhecimentos são adquiridos. Por exemplo, problemas descobertos na fase de Modelagem podem exigir um retorno à Preparação dos Dados, ou a avaliação dos resultados pode levar a um reajuste dos objetivos de negócio.[10] Esta natureza iterativa é particularmente adequada para projetos de ML como este, que envolvem exploração de dados, experimentação com modelos e refinamento contínuo com base no desempenho e feedback.[10] As "Entregas" da challenge podem ser mapeadas para estas fases: a **entrega 1** foca no **Entendimento do Negócio e dos Dados (inicial)**; a **entrega 2** aprofunda o **Entendimento e Preparação dos Dados**; a **entrega 3** detalha aspectos da **Preparação de Dados (modelagem de BD)** e Modelagem; e a **entrega 4** engloba **Modelagem, Avaliação e Implantação**.

2.2. Stack Tecnológico Principal

A seleção das tecnologias é fundamental para o sucesso do projeto, alinhando-se às preferências indicadas na Entrega 1 e às necessidades da solução.

- **Linguagem de Programação: Python** (versão 3.x) será a linguagem principal. Sua vasta gama de bibliotecas para análise de dados (Pandas, NumPy), Machine

Learning (Scikit-learn, TensorFlow, Keras) e integração com serviços em nuvem (AWS SDK - Boto3) a torna ideal para este projeto.[12]

- **Bibliotecas de Análise de Dados:**
  - **Pandas:** Para manipulação eficiente de dados tabulares (DataFrames), limpeza e transformações.[12] Essencial para a fase de Preparação dos Dados (Entrega 2).
  - **NumPy:** Para operações numéricas eficientes, especialmente com arrays multidimensionais, fundamental para cálculos matemáticos e manipulação de dados de sensores.[12]
  - **Matplotlib / Seaborn:** Para visualização de dados durante a Análise Exploratória de Dados (EDA), permitindo a criação de gráficos (histogramas, scatter plots, etc.) para entender padrões e anomalias.[12] Relevante para a Entrega 2.
- **Banco de Dados:** Inicialmente, um banco de dados **SQL relacional**, como **PostgreSQL**, será utilizado, preferencialmente hospedado no **Amazon RDS** na nuvem. A escolha se baseia na natureza estruturada de parte dos dados (metadados de equipamentos, usuários, tenants) e na familiaridade com SQL. Para lidar com o grande volume e a natureza temporal dos dados de sensores, estratégias de particionamento como **Range ou Interval Partitioning** baseadas em timestamp serão cruciais para garantir performance em consultas e manutenibilidade.[15] Embora bancos NoSQL como DynamoDB [17] possam ser considerados para dados de séries temporais em alta escala, a abordagem inicial seguirá a preferência por SQL.
- **Plataforma de Nuvem: AWS** será a plataforma de nuvem utilizada. A AWS oferece um ecossistema completo e maduro de serviços que atendem diretamente às necessidades do projeto, incluindo AWS IoT Core, processamento de dados (Lambda, Kinesis, Glue), armazenamento (S3, RDS), Machine Learning (SageMaker), serverless (Lambda, API Gateway) e monitoramento (CloudWatch).[9]
- **Bibliotecas de Machine Learning:**
  - **Scikit-learn:** Para implementação de modelos de baseline (como Regressão Logística, SVM), pré-processamento de dados (escalonamento, encoding) e avaliação de modelos.[12]
  - **TensorFlow / Keras:** Frameworks poderosos para Deep Learning, essenciais para construir modelos como LSTMs, que são adequados para a análise de séries temporais e estimativa de RUL.[12] Keras, rodando sobre TensorFlow, oferece uma API de alto nível que facilita a prototipagem rápida.[24] Relevantes para a Entrega 4.

Tabela 1: Resumo do Stack Tecnológico

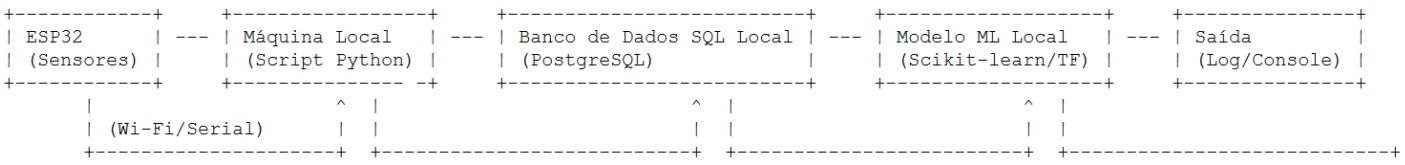
Categoria	Tecnologia Escolhida	Justificativa
Linguagem	Python 3.x	Amplo ecossistema de bibliotecas para dados/ML, forte suporte comunitário, integração com AWS.[12] Preferência da Entrega 1..
Análise de Dados	Pandas, NumPy, Matplotlib/Seaborn	Padrão da indústria para manipulação (Pandas), computação numérica (NumPy) e visualização (Matplotlib/Seaborn).[12] Requisito.
Banco de Dados	PostgreSQL (AWS RDS)	Banco relacional robusto, suporte a SQL, particionamento avançado.[15] Preferência por SQL.
Plataforma de Nuvem	AWS (Amazon Web Services)	Serviços abrangentes para IoT, dados, ML, serverless; escalabilidade e confiabilidade.[9] Requisito.
Machine Learning	Scikit-learn, TensorFlow/Keras	Scikit-learn para baselines/pré-proc.; TF/Keras para Deep Learning (LSTMs).[12] Requisito.
Hardware IoT	ESP32	Microcontrolador de baixo custo com Wi-Fi/Bluetooth integrados, adequado para coleta de dados de sensores.[25] Requisito.

Esta tabela consolida as escolhas tecnológicas, fornecendo uma justificativa clara para cada componente, alinhada aos requisitos da challenge e às necessidades técnicas do projeto.

2.3. Design Inicial do Pipeline de Dados (MVP Local vs. Alvo na Nuvem)

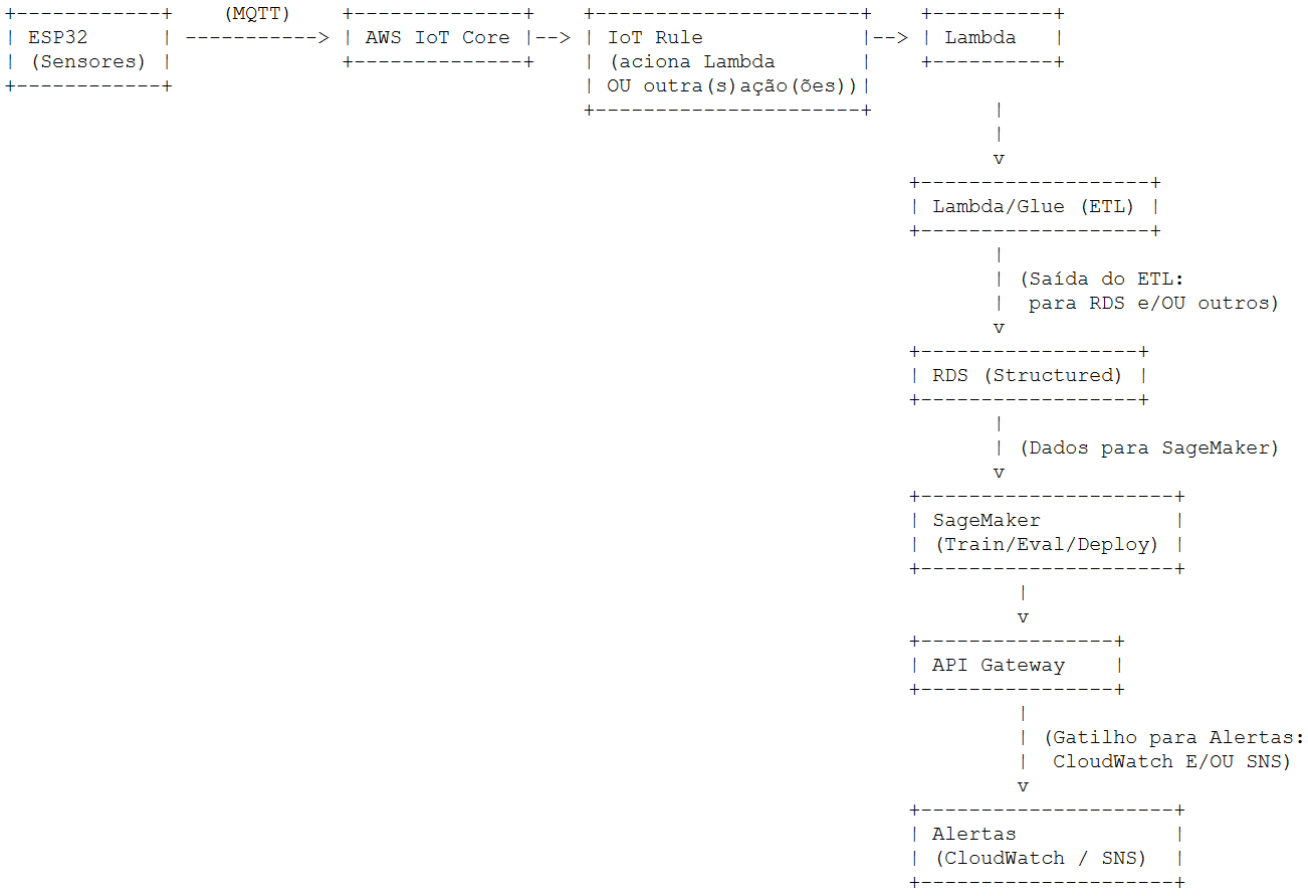
O projeto prevê uma evolução desde um Minimum Viable Product (MVP) rodando localmente até a arquitetura final na nuvem AWS, conforme sugerido como entrega.

- **Pipeline do MVP Local:**



  - *Descrição:* Nesta fase inicial, os dados dos sensores conectados ao ESP32 (MPU6050, DHT22, etc.) são enviados para um script Python rodando em uma máquina local. Este script realiza um processamento mínimo e armazena os dados em um banco de dados SQL local (como PostgreSQL). Um modelo de ML, também treinado e executado localmente usando Scikit-learn ou TensorFlow/Keras, processa esses dados para gerar previsões básicas, exibidas em logs ou no console. Esta abordagem permite validar a coleta de dados e a lógica básica do modelo com menor complexidade de infraestrutura.

- **Pipeline Alvo na Nuvem (Conceitual):**



  - *Descrição:* A arquitetura final na AWS é significativamente mais robusta e escalável. Os dados do ESP32 são enviados via MQTT para o AWS IoT Core, que atua como gateway seguro. Uma Regra IoT direciona os dados para processamento: pode ser para o Kinesis Data Streams para lidar com alto volume em tempo real, ou diretamente para uma função Lambda para processamento inicial. Funções Lambda ou jobs do AWS Glue realizam o ETL (Extração, Transformação, Carga), limpando, transformando e enriquecendo os dados. Os dados processados são armazenados de forma otimizada: dados brutos e intermediários em S3, dados estruturados e agregados no Amazon RDS (PostgreSQL). O Amazon SageMaker é utilizado para todo o ciclo de vida do ML: treinamento de modelos (LSTM, Random Forest), avaliação e implantação dos modelos como endpoints. O API Gateway expõe APIs RESTful para que o frontend (dashboard) possa consumir os dados e as previsões. Dashboards interativos (QuickSight ou Grafana) visualizam o estado dos equipamentos e as previsões. O Amazon CloudWatch monitora todo o sistema, coletando logs e métricas, e aciona alertas via SNS em caso de anomalias ou previsões críticas.

A transição do MVP local para a arquitetura completa na nuvem representa um salto considerável em complexidade. Reconhecer essa diferença é crucial para o planejamento. A migração não será um passo único, mas sim um processo faseado. Inicialmente, pode-se migrar a ingestão de dados para o IoT Core, mantendo o processamento e BD localmente. Em seguida, o banco de dados pode ser movido para o RDS. Posteriormente, o processamento de dados pode ser transferido para Lambda/Glue e, finalmente, o treinamento e a implantação de ML para o SageMaker. Esta abordagem incremental permite gerenciar a complexidade, testar cada componente na nuvem e garantir uma transição mais suave, o que é mais realista para um projeto acadêmico que visa demonstrar competência em ambas as configurações.

2.4. Cronograma de Implementação de Alto Nível

Um cronograma preliminar, sujeito a ajustes, delineado com base nas entregas:

Fase	Entrega	Principais Atividades	Alinhamento CRISP-DM
Entrega 1	Fase 3 do Curso	Definição do problema, pesquisa de tecnologias, design inicial da arquitetura (local e nuvem), seleção da metodologia (CRISP-DM), cronograma detalhado, análise de riscos.	Business Understanding, Data Understanding
Entrega 2	Fase 4 do Curso	Montagem do circuito ESP32, desenvolvimento do código de coleta de dados dos sensores, implementação da ingestão inicial (local), início da coleta de dados, definição de User Stories, implementação de rotinas de limpeza de dados e EDA (Análise Exploratória de Dados).	Data Understanding, Data Preparation
Entrega 3	Fase 5 do Curso	Modelagem do banco de dados (SQL Developer Data Modeler), implementação do schema no BD local/RDS, desenvolvimento de scripts para carga de dados, refinamento da estratégia de particionamento e multi-tenancy no BD.	Data Preparation
Entrega 4	Fase 6 do Curso	Implementação da arquitetura AWS completa (IoT Core, Lambda/Glue, S3, RDS, SageMaker, API Gateway), treinamento e avaliação dos modelos de ML (LSTM, RF) no SageMaker, implantação do modelo (MME/MCE - Modelo como Endpoint / Modelo como Container Endpoint), desenvolvimento do dashboard (QuickSight/Grafana), configuração de logs/alertas (CloudWatch), implementação da geração de relatórios inteligentes, finalização do MVP (Produto Mínimo Viável) e documentação.	Modeling, Evaluation, Deployment

Este cronograma aloca mais tempo para a Entrega 4 devido à sua complexidade, envolvendo a integração de múltiplos serviços AWS e o ciclo completo de MLOps.

2.5. Análise de Pontos Fortes e Riscos

- **Pontos Fortes:**
  - **Expertise Relevante:** O projeto alinha-se com a expertise da Hermes Reply em Indústria 4.0 e soluções de IA/IoT.
  - **Tecnologias Padrão de Mercado:** Utilização de Python e AWS, que são amplamente adotados na indústria, aumentando a relevância do projeto.
  - **Alto Potencial de Impacto:** A manutenção preditiva pode gerar economias significativas e aumentar a eficiência operacional para clientes industriais.
  - **Escalabilidade SaaS:** O modelo SaaS permite atender múltiplos clientes com uma infraestrutura gerenciável.
  - **Riqueza de Dados:** Dados de múltiplos sensores (vibração, temperatura, etc.) podem fornecer insights profundos sobre a saúde do equipamento.
- **Riscos e Estratégias de Mitigação:**

Risco Identificado	Estratégias de Mitigação
Qualidade dos Dados dos Sensores: Dados ruidosos, inconsistentes ou com falhas de sensores podem comprometer a precisão do modelo.	Implementar validação rigorosa na ingestão, técnicas robustas de limpeza de dados e monitoramento da saúde dos sensores.
Precisão e Generalização do Modelo ML: Modelos podem não generalizar bem para novos equipamentos ou condições operacionais. Overfitting é um risco.	Avaliação rigorosa (cross-validation), comparação com baselines, ajuste de hiperparâmetros, monitoramento contínuo do desempenho em produção e retreinamento periódico (MLOps).
Complexidade da Arquitetura Multi-Tenant na Nuvem: Integrar múltiplos serviços AWS de forma segura e eficiente para múltiplos tenants é complexo.	Planejamento cuidadoso da arquitetura, uso de Infrastructure as Code (IaC) como Terraform ou CloudFormation, testes extensivos de isolamento e performance, seguir as melhores práticas do AWS Well-Architected Framework (SaaS Lens).8
Segurança (IoT, Nuvem, Dados): Vulnerabilidades em dispositivos IoT (senhas fracas, atualizações inseguras 34), configurações incorretas na nuvem ou falhas na isolação de dados podem levar a brechas.	Seguir as melhores práticas de segurança da AWS (IAM, VPC, criptografia 36), proteger dispositivos IoT (autenticação forte, updates seguros), realizar auditorias de segurança regulares.
Conformidade com LGPD: Falha em cumprir os requisitos da LGPD pode resultar em multas pesadas e danos à reputação.38	Implementar uma estratégia clara de conformidade (DPAs, gestão de direitos, segurança), definir papéis de controlador/processador, garantir transparência e base legal para processamento, nomear um DPO (Data Protection Officer) se necessário.
"Noisy Neighbor" em Recursos Compartilhados: Em modelos Pool ou Bridge, um tenant com uso intensivo pode impactar a performance de outros.42	Monitoramento cuidadoso do uso de recursos por tenant, implementação de quotas ou throttling (limitação de taxa de processamento, se aplicável na arquitetura), e potencialmente migrar tenants problemáticos para recursos mais isolados.
Vendor Lock-in (AWS): Dependência excessiva dos serviços proprietários da AWS pode dificultar futuras migrações.	Utilizar padrões abertos sempre que possível (e.g., SQL padrão, containers Docker), projetar a arquitetura de forma modular.

2.6. Plano de Desenvolvimento e Divisão de Responsabilidades

Entrega	Descrição da Entrega	Coordenador	RM	Contribuição
1	Design Fundamental - Metodologia, Tecnologias e Conceito de Pipeline	Omar	561375	TODOS
2	Aquisição e Preparação de Dados	Paulo	564262	TODOS
3	Armazenamento e Estrutura de Dados	Deivisson	565095	TODOS
4	Implementação na Nuvem, Integração de IA e Implantação do MVP	Renan	566175	TODOS

3. Entrega 2: Aquisição e Preparação de Dados

3.1. Integração de Sensores ESP32

A coleta de dados confiáveis é a base da manutenção preditiva. Utilizaremos o microcontrolador ESP32 devido ao seu baixo custo, conectividade Wi-Fi/Bluetooth integrada e capacidade de interfacear com diversos sensores.[25] Os sensores especificados (MPU6050, DHT22, GAS, PIR, LDR - Entrega 2) fornecem um conjunto rico de dados sobre as condições operacionais e ambientais do equipamento.

- **Design do Circuito**

Sensor / Componente	Grandeza(s) Medida(s) / Tipo	Interface com ESP32	Pinos ESP32 (Exemplos)	Dados Fornecidos / Finalidade Principal	Considerações Técnicas / Requisitos Específicos
MPU6050	Acelerômetro (vibração x,y,z), Giroscópio (velocidade angular x,y,z), Temperatura (do chip)	I2C (SDA, SCL)	GPIO 21 (SDA), GPIO 22 (SCL)	Dados de vibração e orientação para detecção de problemas mecânicos (desalinhamento, desgaste). Temperatura do chip.	-
DHT22	Temperatura, Umidade (ambiental)	Digital (1 pino de dados - DATA)	GPIO 4	Monitoramento de condições ambientais que afetam desempenho/degradação do equipamento.	Requer resistor pull-up (geralmente 10kΩ) entre DATA e VCC. Atenção à tensão de operação (3.3V vs 5V); se alimentado com 5V, pode necessitar de divisor de tensão.
Sensor de Gás (MQ-XXX, e.g., MQ-135 ou MQ-2)	Concentração de Gás (PPM), Alerta de Limiar de Gás	Analógica (AOUT), Digital (DOUT)	AOUT: ADC (e.g., GPIO 34, 35, 36, 39 - ADC1). DOUT: Pino digital.	Leitura da concentração do gás. Saída digital pode ser usada para um alerta simples de limiar.	VCC geralmente 5V. Calibração é essencial para leituras significativas (R0 em ar limpo, uso de curvas de resposta do datasheet). Disponibilidade de bibliotecas MicroPython robustas pode ser um desafio; adaptação de código Arduino ou desenvolvimento customizado pode ser necessário.
PIR (Sensor de Movimento Infravermelho Passivo - HC-SR501)	Detecção de Presença/Movimento	Digital (Pino OUT)	GPIO 13 ou GPIO 18	Detecta presença/movimento (útil para saber se o equipamento está em operação ou se há atividade humana próxima).	Possui ajustes de sensibilidade e tempo de delay no próprio sensor.
LDR (Resistor Dependente de Luz)	Intensidade da Luz Ambiente	Analógica (via divisor de tensão)	Pino ADC	Mede a intensidade da luz ambiente.	Usado em um circuito divisor de tensão com um resistor fixo (e.g., 10kΩ). O ponto central do divisor é conectado a um pino ADC.

    - **Diagramas de Fiação Detalhados**: Serão fornecidos no repositório do projeto, especificando todas as conexões VCC, GND e pinos de dados/I2C/ADC para cada sensor.
- **Código de Coleta de Dados (MicroPython/Arduino):**
    - **Linguagem:** MicroPython é preferível pela facilidade de uso com Python, mas Arduino C++ é uma alternativa robusta se bibliotecas MicroPython não estiverem disponíveis ou maduras para todos os sensores (especialmente MQ-XXX).
    - **Bibliotecas:** Utilizar bibliotecas específicas para cada sensor (e.g., dht para DHT22 [26], Adafruit_MPU6050 [25] - adaptada ou via wrapper se necessário, bibliotecas I2C genéricas). Para sensores analógicos (MQ, LDR), usar a classe machine.ADC em MicroPython [48] ou analogRead() em Arduino.[46]
    - **Lógica de Leitura:** O código principal em um loop lerá periodicamente cada sensor.
        - MPU6050: Ler aceleração (x,y,z), giroscópio (x,y,z), temperatura.[25]
        - DHT22: Ler temperatura e umidade.[26]
        - MQ-XXX: Ler valor ADC, converter para tensão e, após calibração, estimar PPM.[45]
        - PIR: Ler estado digital (0 ou 1).[49]
        - LDR: Ler valor ADC, converter para tensão/nível de luz.
    - **Formato de Dados:** Empacotar as leituras de todos os sensores (com timestamp e ID do dispositivo/sensor) em um formato estruturado, como JSON, para fácil transmissão e processamento. Exemplo JSON: {"device_id": "ESP32_01", "timestamp": "2024-12-18T10:00:00Z", "mpu6050": {"acc_x": 0.1,...}, "dht22": {"temp": 25.5, "hum": 60.2},...}
    - **Transmissão:** Enviar os dados via MQTT para o AWS IoT Core (na arquitetura final) ou via HTTP/Serial para o servidor local (no MVP). A frequência de transmissão deve ser configurável (e.g., a cada 10 segundos, 1 minuto), balanceando a granularidade dos dados com o consumo de rede/energia.
- **Desafios e Considerações:** A principal dificuldade reside na obtenção e validação de leituras precisas dos sensores MQ, que exigem pré-aquecimento e calibração cuidadosa.[45] A compatibilidade e robustez das bibliotecas MicroPython para todos os sensores listados deve ser verificada; caso contrário, o desenvolvimento em Arduino C++ pode ser mais pragmático para a coleta de dados no ESP32.

3.2. Limpeza de Dados e Análise Exploratória de Dados (EDA)

Após a ingestão, os dados brutos dos sensores raramente estão prontos para modelagem. A fase de Limpeza e EDA (parte do Entendimento e Preparação dos Dados do CRISP-DM) é crucial para garantir a qualidade dos dados e extrair informações iniciais.

- **Metodologia:**
  - *Carregamento e Inspeção Inicial:* Carregar os dados (CSV, JSON, ou de banco de dados) em DataFrames do Pandas. Usar .info(), .describe(), .head() para ter uma visão geral da estrutura, tipos de dados, valores ausentes e estatísticas descritivas.
  - *Tratamento de Valores Ausentes:* Identificar colunas com valores ausentes (.isnull().sum()). Estratégias incluem:
    - **Imputação:** Preencher valores ausentes com a média, mediana (robusta a outliers) ou moda (para dados categóricos) da coluna.[1] Para séries temporais, pode-se usar interpolação linear ou preenchimento forward/backward fill.
    - **Remoção:** Excluir linhas ou colunas com alta porcentagem de valores ausentes, mas com cautela para não perder informação valiosa.
  - *Detecção e Tratamento de Outliers:* Outliers são valores extremos que podem distorcer a análise e os modelos.
    - **Visualização:** Box plots e histogramas ajudam a identificar outliers visualmente.
    - **Métodos Estatísticos:** Z-score (assume distribuição normal) ou IQR (Interquartile Range, mais robusto).
    - **Tratamento:** Remover, ajustar (capping/winsorization) ou transformar os dados (e.g., logaritmo).[5] A decisão depende da natureza do outlier (erro de medição vs. evento real extremo).
  - *Verificação de Tipos de Dados:* Garantir que cada coluna tenha o tipo de dado correto (numérico, datetime, categórico). Converter se necessário (e.g., string de timestamp para objeto datetime).
  - *Normalização/Escalonamento:* Muitos algoritmos de ML (e.g., SVM, redes neurais) performam melhor com dados escalonados. Usar StandardScaler (média 0, desvio padrão 1) ou MinMaxScaler (escala para) do Scikit-learn.[1] Aplicar após a divisão treino/teste para evitar data leakage.
  - *Análise Exploratória de Dados (EDA):*
    - **Análise Univariada:** Examinar a distribuição de cada variável individualmente (histogramas, curvas de densidade, box plots).

        - **Análise Bivariada:** Explorar relações entre pares de variáveis (scatter plots para numérico vs. numérico, box plots para numérico vs. categórico). Calcular correlações (e.g., matriz de correlação de Pearson) para identificar relações lineares.[12]
        - **Análise de Séries Temporais:** Plotar os dados dos sensores ao longo do tempo para visualizar tendências, sazonalidade e anomalias.
- **Ferramentas:** Python com as bibliotecas Pandas, NumPy, Matplotlib e Seaborn.[12]
- **Processo Iterativo:** A limpeza e a EDA são interligadas. Visualizações podem revelar a necessidade de limpeza adicional (e.g., outliers), e a limpeza pode alterar as distribuições que são então reexaminadas. Este ciclo se repete até que os dados sejam considerados de qualidade suficiente para a modelagem.
- **Ideias Iniciais de Features:** A EDA pode sugerir features importantes. Por exemplo, uma forte correlação entre aumento de temperatura e vibração pode indicar um modo de falha específico. Picos consistentes em leituras de gás podem sinalizar vazamentos. A análise das séries temporais pode revelar a taxa de degradação de um componente.

3.3. User Stories (Histórias de Usuário)

User stories são descrições curtas e simples de uma funcionalidade contada da perspectiva do usuário final, focando no valor que essa funcionalidade entrega.[52] Elas são fundamentais na metodologia Agile (e compatíveis com o espírito iterativo do CRISP-DM) para guiar o desenvolvimento e garantir que a solução atenda às necessidades reais. O formato padrão é: "Como um(a) [persona], eu quero [realizar uma ação], para que [eu possa alcançar um benefício]".[52] Cada user story deve ter critérios de aceitação claros que definem quando a história está "concluída".

- **Personas Principais:**
    - **Engenheiro(a) de Manutenção:** Focado(a) na saúde de equipamentos específicos, diagnóstico de falhas, alertas e execução de reparos.
    - **Gerente de Planta:** Visão geral da produção, OEE (Overall Equipment Effectiveness), custos de manutenção, planejamento estratégico de recursos.
    - **Administrador(a) da Plataforma SaaS (Hermes Reply):** Gerenciamento de tenants, monitoramento do sistema, segurança, faturamento.
- **Exemplos de User Stories e Critérios de Aceitação:**
    - **Engenheiro de Manutenção:**
        1. **História:** "Como um Engenheiro de Manutenção, eu quero visualizar em tempo real os níveis de vibração (RMS) e temperatura do Motor Principal da Linha 3, para que eu possa monitorar sua condição operacional

instantaneamente."
    - **Critérios de Aceitação:**
        - O dashboard exibe um gráfico de linha atualizado a cada 10 segundos com o RMS da vibração do motor selecionado.
        - O dashboard exibe um gauge ou gráfico de linha atualizado a cada 10 segundos com a temperatura do motor selecionado.
        - Limites operacionais normais (pré-configurados) são visualmente indicados nos gráficos.
        - O usuário pode selecionar o Motor Principal da Linha 3 em uma lista de equipamentos monitorados.
2. **História:** "Como um Engenheiro de Manutenção, eu quero receber um alerta por e-mail e no dashboard quando a probabilidade de falha prevista para a Bomba Hidráulica B exceder 75%, para que eu possa agendar uma inspeção preventiva."
    - **Critérios de Aceitação:**
        - O sistema calcula a probabilidade de falha para a Bomba Hidráulica B a cada hora.
        - Se a probabilidade > 75%, um alerta é gerado no dashboard (com destaque visual).
        - Se a probabilidade > 75%, um e-mail é enviado para o engenheiro responsável pela Bomba B contendo ID do equipamento, probabilidade de falha e timestamp.
        - O alerta no dashboard permite reconhecimento (acknowledgement).
3. **História:** "Como um Engenheiro de Manutenção, eu quero ver a estimativa de Vida Útil Remanescente (RUL) para o Rolamento da Extrusora Z, juntamente com um intervalo de confiança, para que eu possa planejar a substituição da peça com antecedência."
    - **Critérios de Aceitação:**
        - O dashboard exibe o RUL previsto (em dias/horas/ciclos) para o rolamento selecionado.
        - O RUL é acompanhado por um intervalo de confiança (e.g., 95%).
        - O histórico de previsões de RUL pode ser visualizado em um gráfico.
        - O modelo de RUL é executado diariamente para atualizar a previsão.

- **Gerente de Planta:**

1. **História:** "Como Gerente de Planta, eu quero um painel de visão geral que mostre a contagem de equipamentos em estado crítico (RUL baixo ou alta probabilidade de falha) por linha de produção, para que eu possa alocar equipes de manutenção de forma eficaz."
    - **Critérios de Aceitação:**
        - O dashboard exibe um gráfico de barras ou tabela mostrando o número de equipamentos críticos (RUL < X dias OU Prob. Falha > Y%) por linha de produção.
        - Os dados são atualizados pelo menos diariamente.
        - É possível clicar em uma linha de produção para ver a lista detalhada dos equipamentos críticos.
2. **História:** "Como Gerente de Planta, eu quero acessar um relatório mensal que compare o tempo de inatividade não planejado deste mês com os meses anteriores e com a meta, para que eu possa avaliar o impacto da plataforma de manutenção preditiva."
    - **Critérios de Aceitação:**
        - Um relatório em PDF/CSV é gerado automaticamente no dia 1º de cada mês.
        - O relatório contém um gráfico de barras comparando o downtime não planejado (em horas) dos últimos 6 meses.
        - O relatório inclui o valor do downtime do mês atual e a meta definida.
        - O relatório está disponível para download na plataforma.

- **Administrador(a) da Plataforma SaaS:**
    1. **História:** "Como Administrador da Plataforma, eu quero ter uma interface para cadastrar um novo tenant (cliente industrial), definindo seu nome, plano de assinatura e criando um administrador inicial para ele, para que o novo cliente possa começar a usar a plataforma."
        - **Critérios de Aceitação:**
            - Interface web de administração permite inserir dados do novo tenant.
            - Um novo registro de tenant é criado no banco de dados com um TenantID único.
            - Um usuário administrador inicial é criado e associado ao novo tenant.
            - As credenciais do administrador inicial são geradas e exibidas/enviadas de forma segura.

2. **História:** "Como Administrador da Plataforma, eu quero monitorar o consumo de recursos (e.g., chamadas de API, uso de armazenamento S3/RDS, horas de computação SageMaker) por tenant, para que eu possa garantir a alocação justa de recursos e realizar o faturamento correto."
    - **Critérios de Aceitação:**
        - Um dashboard de administração exibe métricas de consumo agregadas por tenant (requer tagging adequado dos recursos AWS).
        - Os dados de consumo podem ser filtrados por período (dia, semana, mês).
        - Alertas podem ser configurados se um tenant exceder limites pré-definidos (associados ao plano).

Estas user stories, com seus critérios de aceitação, servem como requisitos funcionais claros e testáveis, guiando o desenvolvimento das funcionalidades da plataforma, especialmente o dashboard e a lógica de negócios subjacente, garantindo que as necessidades dos diferentes usuários sejam atendidas.

4. Entrega 3: Armazenamento e Estrutura de Dados

4.1. Modelagem do Banco de Dados

Uma modelagem de banco de dados bem estruturada é essencial para armazenar e recuperar eficientemente os dados coletados e gerados pela plataforma. Utilizaremos a abordagem padrão de modelagem em três níveis (Conceitual, Lógico, Físico) e a ferramenta Oracle SQL Developer Data Modeler, conforme requisito 2.4.1.

- **Modelo Conceitual:** Define as principais entidades de negócio e seus relacionamentos de alto nível.
  - ***Entidades****:* Tenant (Cliente), Usuário, Equipamento, Sensor, LeituraSensor, PrediçãoML, Alerta, OrdemServiço (potencial integração futura).
    - ***Tenant (Cliente)***

      `1..1` Tenant `possui múltiplos (1..N)` Usuários

      `1..1` Tenant `possui múltiplos (1..N)` Equipamentos
    - ***Usuário***

      `1..N` Usuários `pertencem a (N..1)` Tenant
    - **Equipamento**

      `1..N` Equipamentos `pertencem a (N..1)` Tenant

      `1..1` Equipamento `possui múltiplos (1..N)` Sensores

      `1..1` Equipamento `pode ter múltiplas (0..N)` PrediçõesML

      `1..1` Equipamento `pode ter múltiplos (0..N)` Alertas
    - **Sensor**

      `1..N` Sensores `pertencem a (N..1)` Equipamento

      `1..1` Sensor `gera múltiplas (1..N)` LeiturasSensor
    - **LeituraSensor**

      `1..N` LeiturasSensor `são geradas por (N..1)` Sensor

    - **PrediçãoML**

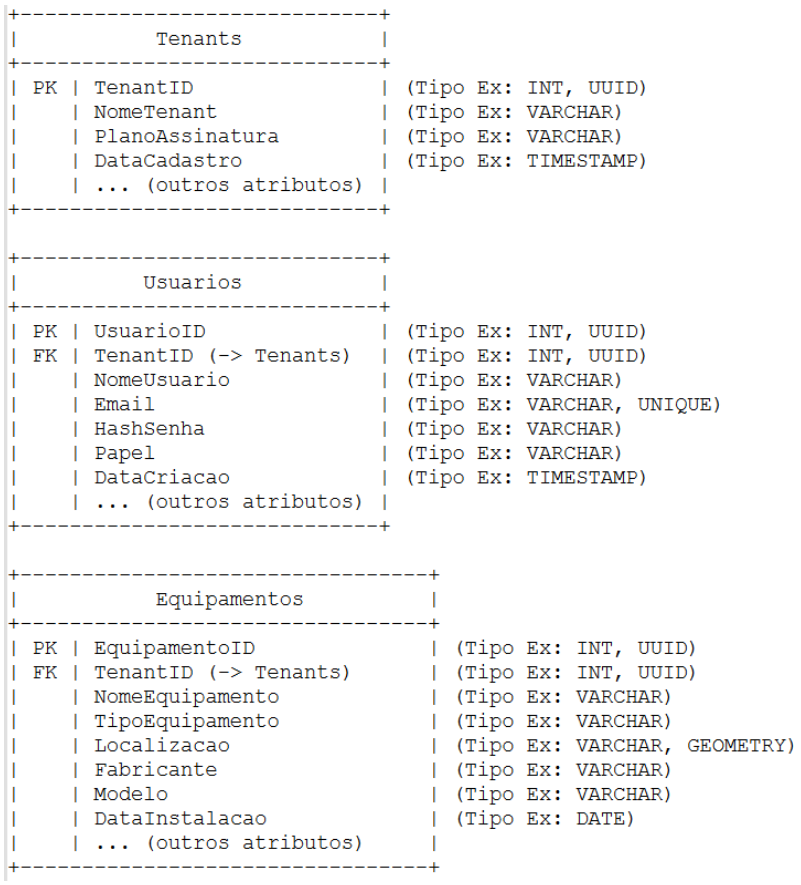
      `0..N` PrediçõesML `são associadas a (N..1)` Equipamento
    - **Alerta**

      `0..N` Alertas `são associados a (N..1)` Equipamento

      `1..1` Alerta `pode gerar uma (0..1 ou 0..N)` OrdemServiço (potencial integração futura)
    - **OrdemServiço** (Entidade para potencial integração futura)

      `0..N` OrdensServiço `podem ser geradas por (N..1)` Alerta

- **Modelo Lógico (Schema ERD):** Este modelo detalha a estrutura das tabelas, colunas, tipos de dados e os relacionamentos através de chaves primárias (PK) e estrangeiras (FK).



```
+---------------------------------+
|            Sensores             |
+---------------------------------+
| PK | SensorID                   | (Tipo Ex: INT, UUID)
| FK | EquipamentoID (->Equipam.) | (Tipo Ex: INT, UUID)
| FK | TenantID (-> Tenants)      | (Tipo Ex: INT, UUID) *Denormalização ou para regras de acesso direto*
|    | TipoSensor                 | (Tipo Ex: VARCHAR, e.g., 'MPU6050', 'DHT22')
|    | UnidadeMedida              | (Tipo Ex: VARCHAR)
|    | FaixaOperacional           | (Tipo Ex: VARCHAR)
|    | LocalInstalacaoSensor      | (Tipo Ex: VARCHAR)
|    | ... (outros atributos)     |
+---------------------------------+

+-----------------------------------------+
|             LeiturasSensores            |
+-----------------------------------------+
| PK | LeituraID                          | (Tipo Ex: BIGINT, UUID)
| FK | SensorID (-> Sensores)             | (Tipo Ex: INT, UUID)
| FK | TenantID (-> Tenants)              | (Tipo Ex: INT, UUID) *Denormalização ou para particionamento/queries*
|    | TimestampLeitura                   | TIMESTAMP WITH TIME ZONE
|    | ValorLeitura                       | JSONB (flexível) OU Colunas Específicas:
|    |   temperatura                      | FLOAT (opcional)
|    |   umidade                          | FLOAT (opcional)
|    |   acc_x                            | FLOAT (opcional)
|    |   ... (outras colunas específicas) |
|    | *Nota: JSONB no PostgreSQL é flexível. Colunas específicas podem ter melhor performance para queries analíticas.*
+-----------------------------------------+

+---------------------------------+
|            ModelosML            |
+---------------------------------+
| PK | ModeloID                   | (Tipo Ex: INT, UUID)
|    | NomeModelo                 | (Tipo Ex: VARCHAR)
|    | VersaoModelo               | (Tipo Ex: VARCHAR)
|    | TipoModelo                 | (Tipo Ex: VARCHAR, e.g., 'LSTM_RUL', 'RF_Class')
|    | ArquivoArtefatoS3          | (Tipo Ex: VARCHAR, URL/URI)
|    | DataTreinamento            | (Tipo Ex: TIMESTAMP)
|    | MetricasAvaliacao          | JSONB
|    | ... (outros atributos)     |
+---------------------------------+

+---------------------------------+
|           PredicoesML           |
+---------------------------------+
| PK | PredicaoID                 | (Tipo Ex: BIGINT, UUID)
| FK | EquipamentoID (->Equipam.) | (Tipo Ex: INT, UUID)
| FK | TenantID (-> Tenants)      | (Tipo Ex: INT, UUID)
| FK | ModeloID (-> ModelosML)    | (Tipo Ex: INT, UUID)
|    | TimestampPredicao          | TIMESTAMP WITH TIME ZONE
|    | ValorRUL                   | FLOAT (Resultado da Predição de Vida Útil Remanescente)
|    | ProbabilidadeFalha         | FLOAT
|    | IntervaloConfianca         | JSONB
|    | ... (outros atributos)     |
+---------------------------------+

+--------------------------------------+
|               Alertas                |
+--------------------------------------+
| PK | AlertaID                        | (Tipo Ex: BIGINT, UUID)
| FK | EquipamentoID (->Equipamentos)  | (Tipo Ex: INT, UUID)
| FK | TenantID (-> Tenants)           | (Tipo Ex: INT, UUID)
| FK | UsuarioReconhecimento (->Usuarios) (opcional) | (Tipo Ex: INT, UUID)
|    | TimestampAlerta                 | TIMESTAMP WITH TIME ZONE
|    | TipoAlerta                      | (Tipo Ex: VARCHAR, e.g., 'RUL_Baixo', 'Prob_Alta', 'Anomalia')
|    | Severidade                      | (Tipo Ex: VARCHAR, e.g., 'Baixa', 'Média', 'Alta', 'Crítica')
|    | Descricao                       | (Tipo Ex: TEXT)
|    | Status                          | (Tipo Ex: VARCHAR, e.g., 'Novo', 'Reconhecido', 'Em Análise', 'Resolvido', 'Fechado')
|    | ... (outros atributos)          |
+--------------------------------------+
```

- *Diagrama ERD:* Será gerado usando o Oracle SQL Developer Data Modeler e incluído no repositório.

- **Modelo Físico (Considerações para PostgreSQL no RDS):**
  - **Tipos de Dados:** Otimizar tipos de dados (e.g., usar FLOAT ou DOUBLE PRECISION para leituras numéricas, TIMESTAMP WITH TIME ZONE para timestamps, VARCHAR com tamanhos apropriados, INTEGER ou BIGINT para IDs).
  - **Indexação:** Criar índices B-tree em chaves primárias e estrangeiras. Criar índices adicionais em colunas frequentemente usadas em cláusulas WHERE ou JOIN, especialmente TimestampLeitura, TenantID e EquipamentoID nas tabelas de leituras, predições e alertas. Índices compostos podem ser úteis (e.g., em (TenantID, EquipamentoID, TimestampLeitura)).
  - **Particionamento:** Implementar a estratégia de particionamento definida na seção 4.2 na tabela LeiturasSensores (e potencialmente PredicoesML, Alertas).
  - **Constraints:** Definir constraints NOT NULL, UNIQUE e CHECK onde apropriado para garantir a integridade dos dados.

4.2. Tratamento de Dados de Séries Temporais: Particionamento

Dados de sensores IoT são inerentemente séries temporais e podem acumular volumes massivos rapidamente, tornando tabelas únicas impraticáveis para consultas e gerenciamento. O particionamento de tabelas é uma técnica essencial para lidar com essa escala.[16]

- **Estratégia:** Utilizar **Range** ou **Interval Partitioning** no PostgreSQL, particionando a tabela LeiturasSensores (e possivelmente outras tabelas de alto volume como PredicoesML e Alertas) com base na coluna TimestampLeitura.[15]
  - **Range Partitioning:** Define explicitamente os limites de cada partição (e.g., uma partição para cada mês de 2024). Requer criação manual de novas partições para dados futuros.
  - **Interval Partitioning (Preferível):** Uma extensão do Range Partitioning. Define-se um intervalo (e.g., '1 month' ou '1 week') e o banco de dados cria automaticamente novas partições conforme dados com timestamps fora dos ranges existentes são inseridos.[15] Isso simplifica enormemente a administração, pois não é preciso criar partições manualmente para o futuro. É necessário definir pelo menos uma partição de range inicial.
- **Benefícios:**
  - **Partition Pruning:** O otimizador de consultas do PostgreSQL pode ignorar partições que não contêm os dados solicitados por um filtro de tempo (e.g., WHERE TimestampLeitura >= '2024-12-01' AND TimestampLeitura < '2025-

01-01'), lendo apenas a partição de Dezembro de 2024. Isso acelera drasticamente as consultas.[15]
    - **Gerenciamento de Dados:** Facilita tarefas como backup/restauração de partições individuais, reconstrução de índices por partição e, crucialmente, a remoção ou arquivamento de dados antigos (e.g., DROP PARTITION ou DETACH PARTITION para mover dados com mais de 2 anos para um armazenamento mais barato).[15]
- **Implementação (Conceitual PostgreSQL):**

```
SQL
CREATE TABLE LeiturasSensores (
    LeituraID BIGSERIAL,
    SensorID INT NOT NULL,
    TenantID INT NOT NULL,
    TimestampLeitura TIMESTAMPTZ NOT NULL,
    ValorLeitura JSONB,
    PRIMARY KEY (LeituraID, TimestampLeitura) -- Chave primária deve incluir a chave de partição
) PARTITION BY RANGE (TimestampLeitura);

-- Criar partição inicial (exemplo)
CREATE TABLE LeiturasSensores_2024_01 PARTITION OF LeiturasSensores
    FOR VALUES FROM ('2024-01-01 00:00:00+00') TO ('2024-02-01 00:00:00+00');

-- Para Interval Partitioning (sintaxe pode variar ligeiramente ou requerer extensões como pg_partman):
-- A ideia é definir um intervalo para criação automática após a partição inicial.
```

- **Consideração Adicional (Multi-Tenancy):** Em um ambiente multi-tenant com banco de dados compartilhado (Pool ou Bridge), pode ser vantajoso usar **particionamento composto**. Por exemplo, particionar primeiro por TenantID (usando LIST ou HASH) e depois subparticionar cada partição de tenant por TimestampLeitura (usando RANGE ou INTERVAL).[16] Isso otimizaria consultas que filtram tanto por tenant quanto por tempo, garantindo que apenas os dados relevantes para um tenant específico em um período específico sejam acessados. A escolha entre particionar primeiro por tenant ou por tempo depende dos padrões de consulta mais frequentes.

4.3. Estratégia de Dados Multi-Tenancy

A arquitetura SaaS deve garantir que os dados de um tenant (cliente) sejam estritamente isolados dos dados de outros tenants (Requisito 1.3.2). A escolha do modelo de isolamento de dados no banco de dados (RDS) tem implicações significativas em custo, complexidade, escalabilidade e nível de isolamento.[53]

- **Modelos de Isolamento no RDS:**
  1. **Silo (Banco de Dados por Tenant):**
     - *Descrição:* Cada tenant possui sua própria instância de banco de dados RDS separada.[53]
     - *Prós:* Isolamento máximo de dados e performance (sem "noisy neighbors"), mais fácil de atender a requisitos de conformidade estritos, custos de BD claramente atribuíveis por tenant.[7]
     - *Contras:* Custo mais elevado (instância por tenant), maior complexidade de gerenciamento (provisionamento, backups, atualizações por tenant), limites de conta AWS no número de instâncias RDS.[7]
  2. **Bridge (Schema por Tenant):**
     - *Descrição:* Tenants compartilham a mesma instância RDS, mas cada um tem seu próprio schema (conjunto de tabelas) dentro do banco de dados.[42]
     - *Prós:* Bom nível de isolamento lógico, custo e complexidade moderados, permite variações de schema por tenant (embora possa complicar migrações).[42]
     - *Contras:* Recursos da instância RDS (CPU, memória, I/O) são compartilhados, risco de "noisy neighbor" ainda existe, limites do SGBD no número de schemas/tabelas por instância.[53]
  3. **Pool (Banco de Dados Compartilhado, Schema Compartilhado):**
     - *Descrição:* Todos os tenants usam a mesma instância RDS e o mesmo schema. As tabelas contêm dados de múltiplos tenants, diferenciados por uma coluna TenantID.[42]
     - *Prós:* Menor custo por tenant, menor complexidade de gerenciamento da infraestrutura de BD, mais fácil de agregar dados entre tenants (se necessário).[53]
     - *Contras:* Isolamento mais fraco (depende da lógica da aplicação e/ou Row-Level Security - RLS no BD), maior risco de "noisy neighbor", mais propenso a atingir limites da instância única (storage, conexões, I/O), complexidade na implementação de RLS e consultas.[7]

Tabela 2: Comparação de Modelos de Dados Multi-Tenancy (RDS)

Característica	Silo (BD por Tenant)	Bridge (Schema por Tenant)	Pool (BD/Schema Compartilhado)
Isolamento Dados	Alto (Físico)	Médio (Lógico - Schema)	Baixo (Lógico - TenantID/RLS)
Custo por Tenant	Alto	Médio	Baixo
Complexidade Gestão	Alta (Múltiplas Instâncias)	Média (Schemas em 1 Instância)	Baixa (Instância Única)
Escalabilidade	Limitada por nº instâncias/conta	Risco "Noisy Neighbor", limites BD	Risco "Noisy Neighbor", limites BD
Flexibilidade Schema	Alta (por instância)	Alta (por schema)	Baixa (Schema único)
Adequado Para	Poucos tenants, alta conformidade	Centenas de tenants, mod. isol.	Milhares de tenants, custo sensível
Chave AWS/BD	Instância RDS dedicada	Schemas PostgreSQL/MySQL	Coluna TenantID, Row-Level Security

Esta tabela resume as principais trocas entre os modelos [7], auxiliando na escolha da estratégia mais adequada para o contexto do projeto.

- **Estratégia Escolhida e Justificativa:**
  - Para o **MVP** deste projeto acadêmico, o modelo **Pool (Banco de Dados/Schema Compartilhado)** é recomendado.
  - **Justificativa:**
    - *Custo:* É a opção mais econômica, crucial para um ambiente de desenvolvimento/MVP com orçamento limitado. Evita o custo de múltiplas instâncias RDS.
    - *Complexidade de Infraestrutura:* Simplifica o gerenciamento da

infraestrutura de banco de dados, permitindo focar mais no desenvolvimento da aplicação e dos modelos de ML.
        - *Escopo do MVP:* Para um número limitado de tenants simulados no MVP, os riscos de "noisy neighbor" e limites de instância são gerenciáveis.
    - **Implicações e Mitigações:**
        - A escolha do modelo Pool exige que a **lógica da aplicação** seja cuidadosamente implementada para sempre filtrar os dados pelo TenantID correto em todas as consultas SQL.
        - Para um isolamento mais robusto, a implementação de **Row-Level Security (RLS)** no PostgreSQL deve ser considerada, embora adicione complexidade às queries e à gestão de permissões.
        - A **indexação eficiente** na coluna TenantID (e em combinação com TimestampLeitura) é fundamental para o desempenho das consultas.
        - Deve-se reconhecer que, para uma **escala de produção real** com muitos tenants ou requisitos de isolamento estritos, uma migração futura para o modelo Bridge ou Silo pode ser necessária. A arquitetura da aplicação deve ser projetada com essa potencial evolução em mente.

5. Entrega 4: Implementação na Nuvem, Integração de IA e Implantação do MVP

Esta fase representa a culminação do projeto, integrando todos os componentes na arquitetura AWS alvo, implementando os modelos de IA/ML e entregando o protótipo funcional (MVP).

5.1. Arquitetura Detalhada na Nuvem AWS

A arquitetura final na AWS orquestra diversos serviços para fornecer uma solução SaaS de manutenção preditiva escalável, segura e eficiente.

- **Diagrama de Arquitetura Detalhado:** (Um diagrama visual usando ícones AWS padrão será incluído no repositório, mostrando o fluxo descrito abaixo).
- **Ingestão de Dados:**
    1. **Dispositivos IoT (ESP32):** Coletam dados dos sensores (MPU6050, DHT22, MQ-XXX, PIR, LDR).
    2. **Conexão Segura:** Utilizam certificados X.509 para autenticar e estabelecer uma conexão segura via protocolo **MQTT** com o **AWS IoT Core**.[17] O IoT Core atua como um message broker gerenciado e escalável, capaz de lidar com milhões de dispositivos.

3. **Regra IoT:** Uma regra configurada no AWS IoT Core é acionada quando mensagens chegam em um tópico MQTT específico (e.g., tenant/{tenantId}/device/{deviceId}/data). Esta regra é o ponto de partida para o processamento na nuvem.[27]
4. **Roteamento Inicial:** A Regra IoT pode ter múltiplas ações. Uma abordagem robusta é:
    - Enviar os dados brutos para um bucket **Amazon S3** (e.g., s3://raw-sensor-data/{tenantId}/...) para arquivamento, auditoria e potencial reprocessamento.[19]
    - Enviar os dados para **Amazon Kinesis Data Streams**.[19] Kinesis é ideal para lidar com fluxos de dados de alta velocidade e permite que múltiplos consumidores processem os dados em paralelo e em tempo real (e.g., uma função Lambda para processamento imediato e talvez Kinesis Data Analytics para agregações em tempo real no futuro).

- **ETL (Extração, Transformação, Carga):**
    1. **Consumo do Stream:** Uma função **AWS Lambda** [22] é configurada como consumidor do Kinesis Data Stream (ou acionada diretamente pela Regra IoT se Kinesis não for usado inicialmente). Lambda é adequado para processamento event-driven e serverless. Para ETLs mais complexos ou que exigem um ambiente Spark/Python gerenciado, **AWS Glue** [28] poderia ser usado, especialmente na preparação de dados para treinamento em batch no SageMaker. Para o fluxo em tempo real do MVP, Lambda é uma escolha eficiente.
    2. **Lógica de Transformação (Lambda):** A função Lambda executa transformações essenciais:
        - Validação do schema dos dados recebidos.
        - Parse do JSON.
        - Conversão de tipos de dados (e.g., string para float, timestamp).
        - Enriquecimento: Adição consistente do TenantID (extraído do tópico MQTT ou de metadados do dispositivo).
        - Cálculo de features simples (e.g., médias móveis curtas, se necessário para o dashboard em tempo real).
        - Tratamento de erros e envio de mensagens malformadas para uma Dead-Letter Queue (DLQ) SQS.
    3. **Saída do ETL:** A função Lambda escreve os dados processados e estruturados:
        - Para o banco de dados **Amazon RDS (PostgreSQL)**, na tabela

LeiturasSensores (e outras tabelas relevantes).[29]
        - Opcionalmente, para outro bucket **S3** (e.g., s3://processed-sensor-data/{tenantId}/...) em formato otimizado (e.g., Parquet) para análise e treinamento de modelos ML.[19]
- **Armazenamento de Dados:**
    - **Amazon RDS (PostgreSQL):** Armazena dados estruturados: metadados de tenants, usuários, equipamentos, sensores, leituras processadas recentes (para dashboards), predições e alertas. Configurado com Multi-AZ para alta disponibilidade e backups automáticos. Utiliza particionamento por tempo (e talvez TenantID) na tabela LeiturasSensores.[15]
    - **Amazon S3:** Armazena dados brutos (arquivamento), dados processados para ML, artefatos de modelos treinados pelo SageMaker e relatórios gerados. Utiliza classes de armazenamento (Standard, IA, Glacier) e políticas de ciclo de vida para otimizar custos.[60] A estrutura de pastas/prefixos garante o isolamento dos dados dos tenants.[55]
- **Machine Learning (Ciclo de Vida):**
    1. **Treinamento: Amazon SageMaker** é usado para treinar os modelos LSTM (RUL) e Random Forest (Classificação).[9] Jobs de treinamento são iniciados (manualmente ou via SageMaker Pipelines), lendo dados processados do S3, utilizando instâncias de computação apropriadas (CPU para RF, GPU para LSTM), e salvando os artefatos do modelo treinado de volta no S3.[62]
    2. **Avaliação:** Após o treinamento, um job de avaliação (SageMaker Processing ou script customizado) calcula métricas de desempenho no conjunto de teste.
    3. **Registro:** Modelos aprovados são versionados e registrados no **SageMaker Model Registry**.[63]
    4. **Implantação:** Modelos registrados são implantados como **SageMaker Endpoints** (Real-time, MME/MCE conforme estratégia definida) ou usados em **SageMaker Batch Transform** jobs.[65]
- **Exposição e Consumo (API & Frontend):**
    1. **API Backend:** Funções **AWS Lambda** implementam a lógica de negócios da API (e.g., buscar dados do RDS para o dashboard, invocar endpoints SageMaker para previsões sob demanda, gerenciar usuários/tenants).
    2. **API Gateway:** Atua como o frontend da API.[23] Recebe requisições HTTP do dashboard/aplicação cliente, autentica/autoriza (usando **Amazon Cognito** para gerenciamento de usuários ou Lambda Authorizers para lógica customizada de tenant), roteia as requisições para as funções Lambda

apropriadas, e retorna as respostas. Gerencia throttling e logging.
3. **Dashboard:** Aplicação web (desenvolvida separadamente ou usando **AWS QuickSight** [30] ou **Amazon Managed Grafana** [31]) que consome as APIs via API Gateway para exibir dados e previsões aos usuários (Engenheiros, Gerentes).

- **Monitoramento e Alertas:**
  1. **Logging: CloudWatch Logs** centraliza logs de todos os serviços (IoT Core, Lambda, API Gateway, SageMaker, RDS).[32]
  2. **Métricas: CloudWatch Metrics** coleta métricas de performance e utilização dos serviços AWS. Métricas customizadas (e.g., RUL previsto, probabilidade de falha) podem ser publicadas pelo Lambda ou SageMaker.
  3. **Alarmes: CloudWatch Alarms** são configurados sobre métricas ou logs para detectar condições anômalas (erros, latência alta, previsões críticas).[32]
  4. **Notificações:** Alarmes acionam o **Amazon SNS (Simple Notification Service)** para enviar notificações (e-mail, SMS) aos responsáveis.[32]
- **Implementação Multi-Tenancy:**
  - **Isolamento de Dados:** Conforme definido na Entrega 3 (Pool model com TenantID no RDS, prefixos/buckets separados no S3 com políticas IAM/Bucket [55]).
  - **Contexto do Tenant:** O TenantID é passado como parâmetro ou extraído de tokens de autenticação (JWT via Cognito/Authorizer) no API Gateway e propagado para Lambdas e consultas ao banco de dados. Para SageMaker MME/MCE, o TenantID (ou um mapeamento dele) é usado nos parâmetros TargetModel ou TargetContainerHostname.[66]
  - **Isolamento de Recursos:** Uso de **IAM Roles** específicas por tenant para acessos granulares a S3 e potencialmente SageMaker (se usando modelos customizados por tenant e MME/MCE).[36] Pipelines SageMaker e Lambdas podem assumir essas roles.[71]

5.2. Implementação do Modelo AI/ML

Esta seção detalha a implementação prática dos modelos de ML escolhidos dentro do ambiente SageMaker.

- **Justificativa da Seleção de Modelos:**
  - **RUL (LSTM):** Redes LSTM são escolhidas pela sua capacidade comprovada de modelar sequências temporais longas e capturar a dinâmica de degradação presente nos dados de sensores ao longo do tempo, o que é essencial para estimar a vida útil remanescente.[1] Arquiteturas como CNN-LSTM podem ser exploradas para extração automática de features

espaciais/temporais dos dados brutos dos sensores antes da camada LSTM.[74]
    - **Classificação de Falhas (Random Forest):** Random Forest é selecionado por sua robustez, bom desempenho em dados tabulares (após feature engineering), capacidade de lidar com um grande número de features e fornecer métricas de importância de features, o que pode ajudar a entender os drivers das falhas.[1]
    - **Modelos Baseline:** Regressão Logística ou SVM para classificação, e modelos de regressão mais simples (Linear, Ridge) ou ARIMA para RUL, serão usados como benchmarks para garantir que os modelos mais complexos oferecem valor agregado.[1]
- **Engenharia de Features Detalhada:**
    - **Dados de Entrada:** Séries temporais multivariadas dos sensores (temperatura, umidade, aceleração x/y/z, giroscópio x/y/z, gás PPM, PIR on/off, nível LDR) e potencialmente dados operacionais (velocidade, torque, se disponíveis).
    - **Técnicas Aplicadas:**
        - **Domínio do Tempo:** Cálculo de estatísticas em janelas deslizantes (rolling windows) de diferentes tamanhos (e.g., 1 min, 10 min, 1 hora): média, desvio padrão, variância, RMS (Root Mean Square - especialmente para vibração), mínimo, máximo, mediana, skewness, kurtosis.[84] Taxa de variação (derivada). Contagem de eventos (e.g., picos de vibração acima de um limiar).
        - **Domínio da Frequência (para Vibração - MPU6050):** Aplicar FFT (Fast Fourier Transform) em janelas de dados de aceleração para extrair features como: energia em bandas de frequência específicas (associadas a falhas conhecidas como desbalanceamento, desalinhamento, problemas de rolamento), frequência dominante, amplitude de pico.[84]
        - **Features de Interação:** Combinações de sensores (e.g., Temperatura * Umidade, Vibração_RMS * Velocidade) que podem ter significado físico ou empírico.[84]
        - **Features Lag:** Valores passados de um sensor ou feature derivada como preditores do estado atual/futuro.[84]
    - **Seleção de Features:** Utilizar a importância de features do Random Forest [81] ou técnicas como análise de correlação, Recursive Feature Elimination (RFE) para selecionar o subconjunto mais preditivo de features, reduzindo a dimensionalidade e potencialmente melhorando a performance e interpretabilidade do modelo.

- **Tratamento de Dados Desbalanceados (Classificação):**
  - Dado que falhas são eventos raros, o dataset de classificação será provavelmente desbalanceado.
  - **Estratégia:** Aplicar **SMOTE** para gerar amostras sintéticas da classe minoritária (falha) no conjunto de treinamento para balancear as classes.[85] Alternativamente ou complementarmente, explorar **Cost-Sensitive Learning** ajustando os pesos das classes no algoritmo Random Forest (muitas implementações, como a do Scikit-learn, suportam class_weight='balanced' ou pesos customizados) para penalizar mais fortemente a classificação incorreta de falhas.[85] A escolha final dependerá da performance em métricas relevantes como Precision, Recall e F1-Score para a classe minoritária.
- **Pipeline de Treinamento SageMaker:**
  - Utilizar **Amazon SageMaker Pipelines** para orquestrar o fluxo de trabalho de ML de ponta a ponta.[9]
  - **Passos Detalhados:**
    1. ProcessingStep (Pré-processamento e Feature Engineering):
       - Input: Dados processados do S3 (saída do ETL Lambda/Glue).
       - Script: Python usando Scikit-learn e Pandas rodando em um container SageMaker Processing (e.g., Scikit-learn container).
       - Ações: Aplicar engenharia de features (rolling stats, FFT, lag features), escalonamento final (StandardScaler/MinMaxScaler fit no treino, transform no treino/validação/teste), tratamento de desbalanceamento (SMOTE aplicado *apenas* no conjunto de treino).
       - Output: Conjuntos de Treino, Validação e Teste com features engenheiradas, salvos no S3.
    2. TrainingStep (Treinamento do Modelo):
       - Input: Dados de treino/validação do S3.
       - Algoritmo: Container SageMaker pré-construído para Scikit-learn (para RF) ou TensorFlow/Keras (para LSTM), ou um container customizado.
       - Hiperparâmetros: Definidos como parâmetros do pipeline (e.g., número de árvores para RF, arquitetura LSTM, taxa de aprendizado).
       - Instância: CPU (e.g., ml.m5.xlarge) para RF, GPU (e.g., ml.g4dn.xlarge) pode ser necessária para LSTM.[88] Considerar Spot Instances para economizar custos.[60]
       - Output: Artefatos do modelo treinado (e.g., model.joblib para RF, SavedModel para TF) salvos no S3.

3. EvaluationStep (Avaliação do Modelo):
    - Input: Modelo treinado do S3, conjunto de teste do S3.
    - Script: Python usando Scikit-learn/TensorFlow rodando em um container SageMaker Processing.
    - Ações: Carregar o modelo e o conjunto de teste. Fazer predições. Calcular métricas de avaliação:
        - Para RUL (Regressão): RMSE (Root Mean Squared Error), MAE (Mean Absolute Error), R^2 Score. Métricas específicas de prognóstico como Scoring Function da PHM Society, Prognostic Horizon (PH).[89]
        - Para Classificação (Falha): Accuracy, Precision, Recall, F1-Score (especialmente para a classe de falha), AUC-ROC, AUC-PR (Precision-Recall curve, melhor para desbalanceados), Confusion Matrix.[85] Considerar métricas baseadas em custo.[86]
    - Output: Relatório de avaliação (JSON com métricas) salvo no S3.
4. ConditionStep (Verificação de Qualidade):
    - Input: Relatório de avaliação do S3.
    - Condição: Comparar uma métrica chave (e.g., F1-score da classe de falha, RMSE do RUL) com um limiar pré-definido ou com a performance do modelo baseline/anterior. Exemplo: evaluation_metrics['f1_failure'] > 0.7.
    - Output: Decide se o pipeline continua para o registro do modelo.
5. RegisterModelStep (Registro do Modelo):
    - Input: Artefatos do modelo treinado do S3, relatório de avaliação.
    - Ação: Se a ConditionStep for aprovada, registra a nova versão do modelo no **SageMaker Model Registry**, associando as métricas de avaliação e outros metadados (e.g., link para o job de treinamento).[63]

- **Rastreamento de Experimentos (MLflow):**
    - Integrar chamadas da API MLflow (mlflow.start_run, mlflow.log_param, mlflow.log_metric, mlflow.sklearn.log_model ou mlflow.tensorflow.log_model, mlflow.register_model) dentro dos scripts de treinamento e avaliação executados nos steps do SageMaker Pipeline.[64]
    - Configurar o SageMaker para usar um servidor de rastreamento MLflow (pode ser hospedado separadamente em EC2/ECS/Fargate com um BD RDS/Aurora para metadados e S3 para artefatos [64]).
    - Isso permite visualizar e comparar todas as execuções do pipeline, hiperparâmetros testados e performance dos modelos de forma centralizada

no UI do MLflow, facilitando a colaboração e a seleção do melhor modelo.[64]

5.3. Estratégia de Implantação do Modelo (SageMaker)

A implantação eficaz dos modelos treinados é crucial para que a plataforma forneça previsões em tempo real ou em batch.

- **Opções de Hosting SageMaker:**
  - **Real-Time Endpoint:** Ideal para inferências de baixa latência (< 60s timeout, < 6MB payload) necessárias para alertas imediatos e dashboards dinâmicos.[65] Mantém instâncias provisionadas.
  - **Batch Transform:** Adequado para processar grandes volumes de dados offline (e.g., gerar previsões diárias para todos os equipamentos para relatórios).[65] Não mantém instâncias persistentes, paga-se pelo tempo de processamento.
  - **Serverless Inference:** Bom para tráfego esporádico ou imprevisível, escala automaticamente (inclusive para zero), mas pode ter latência de cold start.[72]
  - **Asynchronous Endpoint:** Para payloads grandes (até 1GB) ou tempos de inferência longos (até 15 min), com notificações via SNS.[65] Escala para zero.
  - *Escolha:* Para este caso, uma combinação de **Real-Time Endpoint** (para o dashboard e alertas) e **Batch Transform** (para relatórios periódicos) parece a mais adequada.
- **Estratégias de Hosting Multi-Tenant:** Como a plataforma é SaaS, é preciso decidir como lidar com modelos que podem ser genéricos ou específicos por tenant.

Tabela 3: Comparação de Estratégias de Implantação de Modelos (Multi-Tenant)

Estratégia	Modelo Custo	Escalabilidade	Complexidade Gestão	Isolamento Tenant	Flexibilidade Framework	Adequação
Modelo Geral (1 Endpt)	Compartilhado/Baixo	Endpoint único	Baixa	Nível Aplicação	Único	Casos simples, baixa personalização

Endpoint Dedicado/Tenant	Por Tenant/Alto	Por Tenant	Alta	Alta (Endpoint)	Qualquer	Alta necessidade de isolamento, poucos tenants
SageMaker MME	Compartilhado/Méd-Bx	Endpoint Compartilhado	Média	Via Carregamento Modelo	**Único por MME**	**Muitos modelos similares (custo-eficaz)**
SageMaker MCE	Compartilhado/Médio	Endpoint Compartilhado	Média-Alta	Via Invocação Container	**Múltiplos**	Muitos modelos diversos (frameworks)
EC2 Dedicado	Por Instância/Var.	Manual/ASG	Muito Alta	Alta (Instância)/ Manual	Qualquer	Necessidade de controle total, gestão própria

Referência: [7, 65, 66, 72, 91, 92, 93, 94, 95, 96]

- **Estratégia Escolhida e Justificativa:**
  - **Recomendação:** Utilizar **Amazon SageMaker Multi-Model Endpoints (MME)**.[66]
  - **Justificativa:**
    - *Custo-Efetividade:* MMEs permitem hospedar milhares de modelos (potencialmente um por tenant ou por tipo de equipamento/tenant) em um conjunto compartilhado de instâncias (CPU ou GPU), reduzindo drasticamente os custos em comparação com endpoints dedicados por tenant.[66] Zendesk reportou 90% de economia.[93]
    - *Escalabilidade Gerenciada:* SageMaker gerencia o carregamento dinâmico de modelos do S3 para a memória da instância sob demanda, baseado nas invocações.[66] Modelos frequentemente usados permanecem em cache para baixa latência. O endpoint em si pode usar auto-scaling.[93]
    - *Adequação ao Caso de Uso:* Assumindo que os modelos por tenant serão variações do mesmo algoritmo base (e.g., LSTM treinado com dados específicos do tenant, ou RF treinado com dados específicos), eles usarão o mesmo framework (TensorFlow ou Scikit-learn), o que é um requisito do MME.[72]
    - *Gerenciamento:* Embora haja menos controle por modelo individual comparado a endpoints dedicados [93], a complexidade geral de gerenciar

a infraestrutura é significativamente menor do que gerenciar instâncias EC2 manualmente.[93]

- **Implementação com MME:**
  - Armazenar os artefatos de modelo de cada tenant em S3 com uma estrutura de chave clara (e.g., s3://tenant-models-bucket/{tenantId}/model.tar.gz).
  - Criar um SageMaker Model apontando para o diretório raiz (s3://tenant-models-bucket/) e configurar o modo MultiModel.
  - Criar um Endpoint Configuration e um Endpoint usando este SageMaker Model.
  - Na lógica da aplicação (Lambda por trás do API Gateway), ao receber uma requisição de inferência para um tenant específico, invocar o endpoint SageMaker usando invoke_endpoint, passando o caminho S3 completo do modelo do tenant no parâmetro TargetModel (e.g., TargetModel='{tenantId}/model.tar.gz').[66] SageMaker carregará o modelo correto para a instância e executará a inferência.

5.4. Implantação do MVP

- **Features do MVP:** O MVP final incluirá:
  - Ingestão de dados de pelo menos um conjunto de sensores ESP32 via AWS IoT Core.
  - Pipeline de ETL básico (Lambda) armazenando dados no RDS.
  - Treinamento e implantação de pelo menos um modelo ML (e.g., Random Forest para classificação) via SageMaker (MME).
  - API Gateway expondo endpoints para buscar dados recentes e obter previsões.
  - Dashboard básico (QuickSight/Grafana) exibindo leituras de sensores e previsões para um tenant de teste.
  - Configuração básica de logs no CloudWatch.
  - Isolamento de dados funcional para pelo menos dois tenants simulados.
- **Links de Implantação:**
  - **Link para Software SaaS (se aplicável/público):** [Inserir Link de Acesso ao Protótipo Web]
  - **Link para Repositório GitHub (Privado):** [Inserir Link para GitHub contendo código e documentação completa - conforme 1.4.1]

5.5. Implementação do Dashboard

Um dashboard interativo é essencial para que os usuários (Engenheiros, Gerentes) visualizem o status dos equipamentos e as previsões geradas.

- **Escolha da Tecnologia:**
  - **AWS QuickSight:** [30]
    - *Prós:* Serviço totalmente gerenciado pela AWS, fácil integração com fontes de dados AWS (RDS, S3, Redshift), modelo de precificação pay-per-session pode ser vantajoso, bom suporte a multi-tenancy com namespaces e RLS, capacidade de embedding em aplicações SaaS.
    - *Contras:* Menos flexibilidade de customização visual comparado a Grafana/Dash, pode exigir SPICE (cache in-memory) para melhor performance, o que adiciona custo.
  - **Amazon Managed Grafana / Grafana Auto-hospedado:** [31]
    - *Prós:* Excelente para visualização de séries temporais, altamente customizável (plugins, visualizações), integra-se bem com CloudWatch, Prometheus e bancos de dados SQL/NoSQL. Open source (se auto-hospedado).
    - *Contras:* Pode exigir mais configuração e gerenciamento (especialmente se auto-hospedado). Gerenciamento de multi-tenancy via "Organizations" pode ser menos direto que namespaces QuickSight para alguns casos de uso SaaS.
  - **Plotly Dash (Auto-hospedado):**
    - *Prós:* Controle total sobre a aplicação (framework Python), integração nativa com pipeline de dados Python/ML, visualizações interativas e customizáveis.
    - *Contras:* Requer desenvolvimento web, hospedagem (EC2, ECS, App Runner) e gerenciamento da infraestrutura.
  - **Recomendação: AWS QuickSight** é recomendado para o MVP devido à sua integração nativa com AWS, facilidade de uso para BI e forte suporte a multi-tenancy e embedding, alinhando-se bem com a arquitetura SaaS na AWS.[68]
- **Visualizações Chave (Baseadas nas User Stories - Seção 3.3):**
  - *Visão do Engenheiro:* Gráficos de linha em tempo real (ou near real-time via Direct Query ou refresh SPICE) para vibração, temperatura; Gauges para RUL e Probabilidade de Falha; Tabelas de alertas ativos; Históricos de sensores e previsões.
  - *Visão do Gerente:* Gráficos de barras/pizza resumindo equipamentos por status de risco/linha de produção; KPIs de downtime; Tabelas comparativas de performance.

    - *Interatividade:* Filtros por tenant, equipamento, linha de produção, período de tempo. Drill-down de visões gerais para detalhes de equipamentos.
- **Considerações Multi-Tenant (QuickSight):**
    - Utilizar **Namespaces** em QuickSight para isolar completamente os recursos (datasets, análises, dashboards) de cada tenant.[68]
    - Criar datasets parametrizados ou usar **Row-Level Security (RLS)** para garantir que usuários de um tenant vejam apenas seus próprios dados, mesmo que usem análises/dashboards compartilhados.[68] RLS requer um dataset de permissões mapeando usuários a TenantIDs/Equipamentos permitidos.
    - Utilizar a API de embedding do QuickSight para gerar URLs seguras e específicas do tenant para integrar os dashboards na aplicação SaaS.[68]
- **Atualização em Tempo Real:** QuickSight pode usar Direct Query para buscar dados do RDS em tempo real (com impacto na performance do BD) ou usar SPICE (cache in-memory) com agendamentos de refresh (e.g., a cada 15 minutos, 1 hora) para melhor performance do dashboard.[68] A escolha depende do requisito de latência versus custo/impacto no BD.

5.6. Logging e Alertas

Um sistema robusto de logging e alertas é vital para a operacionalização e monitoramento da plataforma.

- **Serviço de Logging: AWS CloudWatch Logs** será o repositório centralizado.[32]
    - *Fontes de Log:* AWS IoT Core (logs de conexão, regras), AWS Lambda (logs de execução, print statements), API Gateway (logs de acesso, execução), SageMaker (logs de treinamento, endpoint/invocação), RDS (logs do motor, slow query log), Logs de Aplicação (se houver componentes em EC2/ECS).
    - *Estrutura:* Organizar logs em Log Groups (e.g., /aws/lambda/my-etl-function, /aws/sagemaker/Endpoints/my-mme-endpoint, AWSIotLogsV2 [32]) e Log Streams. Utilizar formato JSON estruturado para logs sempre que possível para facilitar a consulta e análise.
- **Serviço de Alertas: Amazon CloudWatch Alarms**.[32]
- **Mecanismos de Alerta:**
    - **Alarmes Baseados em Métricas:**
        - *Operacionais:* Monitorar métricas padrão da AWS: Erros Lambda (Errors), Latência API Gateway (Latency), CPU/Memória RDS (CPUUtilization), Utilização/Latência Endpoint SageMaker (GPUUtilization, ModelLatency), Mensagens IoT não processadas. Definir limiares e criar alarmes para

desvios.
        - *Preditivos:* Publicar as saídas dos modelos ML (RUL, Probabilidade de Falha) como Métricas Customizadas do CloudWatch. Criar alarmes quando RUL < threshold_RUL ou ProbabilidadeFalha > threshold_Prob.
    - **Alarmes Baseados em Logs:** Criar Filtros Métricos no CloudWatch Logs para procurar por padrões específicos (e.g., "ERROR", "FATAL", "Timeout") e criar alarmes baseados na contagem desses padrões.
    - **Alertas Diretos da Aplicação:** A lógica de negócio (e.g., em uma Lambda pós-predição) pode diretamente publicar uma mensagem no SNS se uma condição crítica for detectada (e.g., falha iminente prevista).
- **Notificação:** Utilizar **Amazon SNS** para distribuir os alertas.[32]
    - Criar tópicos SNS (e.g., CriticalAlerts, WarningAlerts).
    - Configurar CloudWatch Alarms (ou a lógica da aplicação) para publicar mensagens nesses tópicos.
    - Configurar assinaturas para os tópicos SNS, direcionando as notificações para os canais apropriados:
        - E-mail para engenheiros de manutenção/gerentes de planta (listas de distribuição).
        - SMS para alertas críticos.
        - Endpoint HTTPS para integração com sistemas de ticketing ou PagerDuty.
        - Fila SQS para processamento automatizado adicional.

5.7. Relatórios Inteligentes

Além do dashboard em tempo real, a plataforma deve gerar relatórios periódicos com insights mais profundos e recomendações acionáveis.

- **Geração Automatizada:**
    - *Mecanismo:* Agendar uma função **AWS Lambda** ou um job **SageMaker Batch Transform** para executar periodicamente (e.g., semanalmente, mensalmente).
    - *Processo:* O job/função irá:
        1. Consultar dados históricos relevantes do RDS e/ou S3 (leituras de sensores, predições passadas, alertas gerados, registros de manutenção se disponíveis).
        2. Executar análises agregadas e potencialmente rodar modelos ML (Batch Transform) para gerar previsões em lote.
        3. Compilar os resultados em um formato de relatório (e.g., PDF, CSV).
        4. Salvar o relatório no S3 e/ou notificar os usuários (e.g., link por e-mail via

SNS).

- **Insights Orientados por IA:** Os relatórios devem ir além da simples apresentação de dados.
  - **Análise de Tendências:** Identificar equipamentos com tendências de degradação acelerada (RUL diminuindo rapidamente) ou aumento na frequência de anomalias.
  - **Identificação de Padrões de Falha:** Usar técnicas de agregação e análise (possivelmente clustering ou análise de regras de associação em dados históricos de falha/alerta) para identificar padrões comuns que precedem falhas em certos tipos de equipamento.
  - **Importância de Features:** Incluir insights derivados da importância de features dos modelos de classificação (e.g., "Para falhas em bombas do tipo X, a vibração na frequência Y e a temperatura Z foram os indicadores mais fortes no último período").[81]
  - **Comparação entre Equipamentos/Linhas:** Comparar a saúde e o risco previsto entre equipamentos similares ou linhas de produção para identificar outliers ou áreas de maior preocupação.
- **Recomendações Acionáveis:** O ponto crucial é traduzir as previsões e insights em ações concretas para a equipe de manutenção.[3]
  - **Baseadas em RUL:** "Equipamento A: RUL previsto de 15 dias (Intervalo de Confiança: 10-20 dias). **Recomendação:** Agendar inspeção detalhada e possível substituição do componente Y na próxima janela de manutenção planejada (dentro de 10 dias)." [98]
  - **Baseadas em Probabilidade de Falha:** "Equipamento B: Probabilidade de falha nas próximas 24h estimada em 90%. **Recomendação:** Parada imediata para inspeção corretiva. Foco: Verificar [feature importante 1] e [feature importante 2]." [98]
  - **Baseadas em Anomalias:** "Equipamento C: Detectada anomalia persistente no sensor de pressão (desvio de 3 desvios padrão da média histórica). **Recomendação:** Verificar sensor de pressão quanto a falhas ou drift. Inspecionar sistema relacionado a possíveis vazamentos." [98]
  - **Otimização de PM:** "Equipamento D: Manutenção preventiva agendada para próxima semana, mas RUL previsto > 100 dias e probabilidade de falha < 5%. **Recomendação:** Considerar adiar a manutenção preventiva programada para otimizar recursos." [98]
- **Integração com CMMS (Computerized Maintenance Management System):** Para maximizar o valor, a plataforma pode ser projetada para integrar-se com

sistemas CMMS existentes.[4] As recomendações acionáveis poderiam acionar automaticamente a criação de ordens de serviço (Work Orders) no CMMS, pré-preenchidas com informações do equipamento, previsão, e recomendação, agilizando o fluxo de trabalho da manutenção.[4] Isso normalmente envolveria o uso de APIs fornecidas pelo CMMS.

5.8. Refinamento Iterativo do Modelo (MLOps)

A performance dos modelos de ML pode degradar ao longo do tempo devido a mudanças nas condições operacionais ou no próprio equipamento (model drift, concept drift). Um processo iterativo de refinamento é essencial.

- **Estratégia MLOps:** Implementar práticas de MLOps para automação e gerenciamento do ciclo de vida do ML.
- **Monitoramento Contínuo:**
  - *Monitoramento Técnico:* Usar CloudWatch para monitorar métricas dos endpoints SageMaker (latência, erros 4xx/5xx, utilização de CPU/GPU/Memória).[100]
  - *Monitoramento de Performance do Modelo:* Implementar **SageMaker Model Monitor** ou lógica customizada para monitorar:
    - *Data Drift:* Detectar mudanças estatísticas nos dados de entrada que chegam ao endpoint em comparação com os dados de treinamento.
    - *Model Quality Drift:* Monitorar a acurácia/performance do modelo em produção (requer coleta de ground truth - dados reais de falha/vida útil - o que pode ter um delay significativo). Comparar previsões com resultados reais quando disponíveis.
    - *Bias Drift:* Monitorar se o modelo desenvolve vieses em suas previsões para diferentes grupos (se aplicável).
- **Retreinamento Automatizado:**
  - *Gatilhos:* Agendar retreinamentos periódicos (e.g., mensalmente) usando **SageMaker Pipelines** e **Amazon EventBridge Scheduler**. Adicionalmente, configurar gatilhos baseados em alertas do Model Monitor (e.g., retreinar se o drift de dados exceder um limiar).
  - *Processo:* O pipeline de retreinamento usará os dados mais recentes acumulados no S3/RDS para treinar uma nova versão do modelo, avaliá-la e registrá-la no Model Registry.[100]
- **Teste A/B (Variantes de Produção):**
  - *Mecanismo:* Usar **SageMaker Production Variants** para implantar a nova versão do modelo (candidata) ao lado da versão atualmente em produção no

mesmo endpoint.[101]
    - *Teste:* Direcionar uma pequena porcentagem do tráfego de inferência real para o modelo candidato (e.g., 10%). Coletar métricas de performance técnica (latência, erros) e, se possível, de negócio (comparar a utilidade das previsões, feedback dos usuários) para ambas as variantes.[101]
    - *Decisão:* Se o modelo candidato mostrar performance superior ou igual, gradualmente aumentar o tráfego para ele e eventualmente desativar a variante antiga. Se não, reverter e analisar o motivo.[101]
- **Ciclo de Feedback Humano:** Estabelecer um mecanismo para que engenheiros de manutenção forneçam feedback sobre a precisão e utilidade das previsões e alertas. Esse feedback (e.g., confirmação de falha, causa raiz real) é valioso para melhorar a rotulagem dos dados e refinar os modelos em futuras iterações de retreinamento.

6. Segurança e Conformidade

A segurança e a conformidade regulatória são aspectos críticos para uma plataforma SaaS, especialmente ao lidar com dados industriais potencialmente sensíveis e operar no Brasil sob a LGPD.

6.1. Estratégia de Conformidade com a LGPD (Brasil)

A Lei Geral de Proteção de Dados Pessoais (Lei nº 13.709/2018) estabelece regras sobre o tratamento de dados pessoais no Brasil.[41] Embora os dados de sensores de máquinas não sejam inerentemente "dados pessoais", eles podem se tornar pessoais se vinculados a operadores identificáveis ou se a análise permitir inferir informações sobre indivíduos. Além disso, dados de usuários da plataforma (engenheiros, gerentes) são dados pessoais. Portanto, a conformidade com a LGPD é mandatória.[37]

- **Aplicabilidade e Escopo Extraterritorial:** A LGPD se aplica a qualquer operação de tratamento realizada no Brasil, ou que vise oferecer bens/serviços a indivíduos no Brasil, ou que envolva dados coletados no Brasil, independentemente de onde a empresa esteja sediada.[37]
- **Princípios Fundamentais:** A plataforma deve aderir aos princípios da LGPD [38]:
    - *Finalidade:* O tratamento de dados (sensores, usuários) deve ter propósitos legítimos, específicos, explícitos e informados (e.g., fornecer o serviço de manutenção preditiva contratado).
    - *Adequação:* O tratamento deve ser compatível com as finalidades informadas.
    - *Necessidade:* Coletar e processar apenas os dados mínimos necessários para

atingir a finalidade (data minimization). Aplicável aos dados de sensores e de usuários.
    - *Livre Acesso:* Garantir aos titulares (usuários da plataforma) acesso fácil e gratuito aos seus dados.
    - *Qualidade dos Dados:* Garantir que os dados dos usuários sejam exatos e atualizados.
    - *Transparência:* Fornecer informações claras e acessíveis aos titulares sobre o tratamento de seus dados.
    - *Segurança:* Adotar medidas técnicas e administrativas para proteger os dados (detalhado na Seção 6.2).
    - *Prevenção:* Adotar medidas para prevenir danos decorrentes do tratamento de dados.
    - *Não Discriminação:* Não utilizar os dados (incluindo resultados de IA/ML) para fins discriminatórios ilícitos ou abusivos.[39]
    - *Responsabilização e Prestação de Contas (Accountability):* Ser capaz de demonstrar a conformidade com a LGPD.
- **Bases Legais:** O tratamento de dados de usuários e, potencialmente, dados operacionais vinculados, deve se basear em uma das bases legais do Art. 7º da LGPD. As mais prováveis são:
    - *Execução de Contrato:* O tratamento é necessário para executar o contrato de serviço SaaS com o cliente (tenant).[39]
    - *Legítimo Interesse:* O tratamento pode ser baseado no legítimo interesse do controlador (cliente) ou de terceiros (Hermes Reply), desde que não infrinja os direitos fundamentais do titular e após avaliação de impacto (LIA). Pode aplicar-se ao processamento de dados operacionais para melhoria do serviço.
    - *Consentimento:* Necessário se outras bases legais não se aplicarem, especialmente para dados sensíveis (improvável neste contexto) ou usos secundários dos dados. O consentimento deve ser livre, informado e inequívoco.[38]
- **Papéis de Controlador e Processador:** É crucial definir os papéis [39]:
    - **Controlador:** A empresa cliente (tenant) que contrata o serviço SaaS é, via de regra, a controladora dos seus dados operacionais e dos dados dos seus funcionários que usam a plataforma. Ela determina as finalidades e os meios essenciais do tratamento.
    - **Processador (Operador):** A Hermes Reply, como provedora da plataforma SaaS, atua como processadora (operadora) dos dados em nome e sob as

instruções do cliente (controlador).

- **Tabela 4: Responsabilidades Controlador vs. Processador (LGPD - Contexto SaaS PdM)**

Área de Responsabilidade	**Controlador (Cliente Industrial)**	**Processador (Hermes Reply)**	**Referência LGPD (Exemplos)**
Definição da Base Legal	**Primária**	Deve seguir instruções	Art. 7º, 11º
Gestão Direitos dos Titulares	**Primária**	**Auxilia** o controlador	Art. 18º
Implementação Medidas de Segurança	Define requisitos	**Implementa** (técnicas/administrativas)	Art. 46º, 50º
Nomeação do Encarregado (DPO)	**Obrigatório** (regra geral)	**Obrigatório** (em certos casos)	Art. 41º
Notificação de Incidente de Segurança	**Notifica ANPD/Titulares**	**Notifica Controlador** sem demora	Art. 48º
Contrato de Tratamento (DPA)	**Garante existência/adequação**	**Assina e cumpre**	Art. 39º (implícito)
Manutenção Registros de Operações	**Mantém seus registros**	**Mantém seus registros**	Art. 37º
Avaliação de Impacto (RIPD/DPIA)	**Realiza** (se necessário)	**Auxilia** o controlador	Art. 38º

- **Contratos de Tratamento de Dados (DPAs):** Um DPA claro e detalhado é essencial entre Hermes Reply (Processador) e cada cliente (Controlador). Deve especificar: objeto, duração, natureza e finalidade do tratamento; tipo de dados

pessoais e categorias de titulares; obrigações e direitos do controlador; obrigações do processador (incluindo seguir instruções, garantir confidencialidade, implementar segurança, auxiliar o controlador, regras para subcontratação, exclusão/devolução de dados ao término).[37]

- **Direitos dos Titulares:** A plataforma deve ter mecanismos para que Hermes Reply possa auxiliar os clientes (controladores) a responder às solicitações dos titulares (seus funcionários usuários) quanto a acesso, correção, eliminação, portabilidade, informação sobre compartilhamento, revogação de consentimento e revisão de decisões automatizadas.[38]
- **Medidas de Segurança:** Implementar medidas técnicas e administrativas robustas (detalhadas na Seção 6.2) para proteger os dados contra acessos não autorizados e incidentes.[38]
- **Notificação de Incidentes de Segurança:** Hermes Reply (processador) deve notificar o cliente (controlador) relevante sobre qualquer incidente de segurança que possa acarretar risco ou dano relevante, sem demora injustificada. O controlador é responsável por notificar a ANPD (Autoridade Nacional de Proteção de Dados) e os titulares, se necessário.[38] Um plano de resposta a incidentes deve estar em vigor.
- **Encarregado (DPO):** Tanto o controlador quanto o processador podem precisar nomear um Encarregado pelo Tratamento de Dados Pessoais (DPO). O DPO atua como canal de comunicação com titulares e a ANPD.[38]
- **Transferência Internacional:** Se houver transferência de dados para fora do Brasil (e.g., servidores AWS em outra região), garantir que seja feita em conformidade com as regras da LGPD (países com nível adequado de proteção, cláusulas contratuais padrão, normas corporativas globais, etc.).[39]
- **Decisões Automatizadas (IA/ML):** O Art. 20 da LGPD concede ao titular o direito de solicitar a revisão de decisões tomadas unicamente com base em tratamento automatizado de dados pessoais que afetem seus interesses (e.g., uma previsão de falha que leve a uma ação de manutenção impactante).[110] O controlador deve fornecer informações claras sobre os critérios e procedimentos usados na decisão automatizada, respeitados os segredos comercial e industrial.[110] É crucial garantir que os modelos de IA/ML não resultem em discriminação.[39]
- **Dados de IoT:** Aplicar os princípios da LGPD à coleta via ESP32: coletar apenas dados necessários (Necessidade), para a finalidade específica de PdM (Finalidade), garantir a segurança da transmissão e armazenamento (Segurança), e ser transparente com o cliente sobre quais dados são coletados e como são

usados (Transparência).[38]

6.2. Melhores Práticas de Segurança na AWS

A segurança na nuvem é uma responsabilidade compartilhada. A AWS cuida da segurança *da* nuvem (infraestrutura física), enquanto o cliente (Hermes Reply) é responsável pela segurança *na* nuvem (configuração de serviços, dados, acesso).

- **IAM (Identity and Access Management):**
  - **Princípio do Menor Privilégio:** Conceder apenas as permissões estritamente necessárias para cada usuário, grupo ou role (função IAM). Evitar usar o usuário root da conta.
  - **Roles para Serviços:** Usar IAM Roles para permitir que serviços AWS (Lambda, EC2, SageMaker) acessem outros serviços AWS (S3, RDS) de forma segura, sem armazenar credenciais de longo prazo.[36]
  - **Políticas Granulares:** Criar políticas IAM customizadas que restrinjam o acesso a recursos específicos (e.g., buckets S3 específicos do tenant, endpoints SageMaker) com base em condições como tags (TenantID), prefixos S3 ou atributos do solicitante.[36]
  - **MFA (Multi-Factor Authentication):** Habilitar MFA para usuários IAM com privilégios elevados (administradores).[37]
- **Segurança de Dados (S3 e RDS):**
  - **Isolamento S3:** Usar buckets S3 dedicados por tenant ou prefixos dentro de um bucket compartilhado. Aplicar políticas de bucket e/ou políticas IAM para garantir que um tenant só possa acessar seus próprios dados/prefixos.[55]
  - **Bloqueio de Acesso Público S3:** Garantir que o acesso público esteja bloqueado para todos os buckets que contêm dados de tenants.
  - **Criptografia S3:** Habilitar criptografia no lado do servidor (SSE-S3 ou SSE-KMS com chaves gerenciadas pelo cliente ou pela AWS) para dados em repouso.[36] Exigir criptografia em trânsito (HTTPS/TLS) para uploads/downloads.
  - **Propriedade de Objetos S3:** Usar a configuração "Bucket owner enforced" para simplificar o gerenciamento de permissões, desabilitando ACLs.[112]
  - **Segurança RDS:** Colocar instâncias RDS em sub-redes privadas dentro de uma VPC. Usar Security Groups para permitir acesso apenas de fontes autorizadas (e.g., Lambdas da aplicação, instâncias EC2 de administração). Habilitar criptografia em repouso para a instância RDS e seus backups. Usar SSL/TLS para conexões com o banco de dados. Gerenciar credenciais de banco de dados de forma segura (e.g., AWS Secrets Manager).

- **Segurança de Rede (VPC):**
  - **Isolamento com VPC:** Projetar uma Virtual Private Cloud (VPC) com sub-redes públicas (para recursos que precisam de acesso à internet, como NAT Gateways ou Bastion Hosts) e privadas (para RDS, Lambdas, instâncias EC2 backend, endpoints SageMaker).[36]
  - **Security Groups:** Atuam como firewalls stateful no nível da instância/recurso, permitindo tráfego apenas em portas e protocolos específicos de fontes definidas (outros security groups ou IPs).
  - **Network ACLs (NACLs):** Atuam como firewalls stateless no nível da sub-rede, fornecendo uma camada adicional de controle de tráfego.
  - **VPC Endpoints:** Usar VPC Endpoints (Interface ou Gateway) para permitir que recursos dentro da VPC acessem serviços AWS (S3, Kinesis, SageMaker, DynamoDB, etc.) sem atravessar a internet pública, melhorando a segurança e potencialmente a performance.[36]
- **Criptografia:**
  - **Em Repouso:** Criptografar todos os dados sensíveis armazenados no S3, RDS (incluindo snapshots), volumes EBS anexados a EC2/SageMaker Notebooks. Usar AWS Key Management Service (KMS) para gerenciar as chaves de criptografia, considerando o uso de Customer Managed Keys (CMKs) para maior controle.[36]
  - **Em Trânsito:** Usar TLS/SSL para todas as comunicações: entre cliente e API Gateway, entre API Gateway e Lambda, entre Lambda e RDS/S3/SageMaker, entre ESP32 e IoT Core.[36] Configurar API Gateway e outros serviços para exigir conexões HTTPS.
- **Segurança SageMaker:**
  - **Notebook Instances:** Lançar notebooks em VPCs privadas, restringir acesso via IAM, criptografar volumes de armazenamento.
  - **Training Jobs:** Executar jobs em VPCs privadas, usar roles IAM com permissões mínimas, criptografar dados de entrada/saída e artefatos do modelo.[36]
  - **Endpoints:** Implantar endpoints em VPCs privadas, usar roles IAM para acesso ao modelo, exigir autenticação/autorização IAM para invocação, criptografar tráfego com TLS.[36] Considerar PrivateLink para acesso privado ao endpoint.
- **Logging e Monitoramento (Segurança):**
  - **AWS CloudTrail:** Habilitar CloudTrail para registrar todas as chamadas de API feitas na conta AWS, essencial para auditoria e investigação de

incidentes.[36]
    - **Amazon GuardDuty:** Habilitar GuardDuty para detecção inteligente de ameaças, monitorando logs (CloudTrail, VPC Flow Logs, DNS logs) em busca de atividades maliciosas ou não autorizadas.[36]
    - **AWS Security Hub:** Usar Security Hub para obter uma visão centralizada dos alertas de segurança e do status de conformidade em toda a conta AWS.
    - **CloudWatch Logs/Alarms:** Monitorar logs em busca de eventos de segurança (e.g., tentativas de login falhadas, acesso negado) e configurar alarmes (conforme Seção 5.6).

6.3. Visão Geral de Frameworks Relevantes

Adotar frameworks de segurança e conformidade reconhecidos demonstra maturidade e ajuda a construir uma postura de segurança robusta, essencial para a confiança dos clientes SaaS.

- **ISO/IEC 27001:**
    - *Descrição:* Padrão internacional para estabelecer, implementar, manter e melhorar continuamente um Sistema de Gestão de Segurança da Informação (SGSI).[113]
    - *Relevância:* Fornece uma abordagem holística e baseada em risco para a segurança da informação. A certificação ISO 27001 é frequentemente um requisito ou um diferencial importante para clientes B2B, especialmente em setores regulados, pois demonstra um compromisso formal com as melhores práticas de segurança.[114] Cobre controles técnicos, processos e gestão.
- **NIST Cybersecurity Framework (CSF):**
    - *Descrição:* Framework desenvolvido pelo National Institute of Standards and Technology (EUA), organizado em torno de cinco funções principais: Identificar, Proteger, Detectar, Responder, Recuperar (a versão 2.0 adiciona Governar).[113]
    - *Relevância:* Oferece uma linguagem comum e uma estrutura flexível e baseada em risco para gerenciar a segurança cibernética. É amplamente adotado e pode ser adaptado para diferentes tipos de organização e setores, incluindo ambientes SaaS na nuvem.[114] Ajuda a alinhar atividades de segurança com os objetivos de negócio.
- **SOC 2 (System and Organization Controls 2):**
    - *Descrição:* Framework de relatório desenvolvido pelo AICPA (American Institute of Certified Public Accountants) focado nos controles de uma organização de serviços relevantes para segurança, disponibilidade,

integridade de processamento, confidencialidade e privacidade dos dados dos clientes (Trust Services Criteria).[115]
    - *Relevância:* Extremamente relevante para provedores de SaaS. Um relatório SOC 2 (Tipo 1 ou Tipo 2) emitido por um auditor independente fornece aos clientes e stakeholders garantias sobre a eficácia dos controles implementados pela organização de serviços para proteger seus dados e garantir a disponibilidade do serviço. Frequentemente solicitado por clientes empresariais.
- **AWS Well-Architected Framework (SaaS Lens):**
    - *Descrição:* Conjunto de melhores práticas da AWS para projetar e operar sistemas na nuvem AWS de forma segura, de alto desempenho, resiliente e eficiente.[8] O SaaS Lens adiciona perguntas e orientações específicas para arquiteturas multi-tenant.[8]
    - *Relevância:* Fornece orientação prática e específica da AWS para construir a plataforma SaaS, cobrindo todos os pilares (Excelência Operacional, Segurança, Confiabilidade, Eficiência de Performance, Otimização de Custos, Sustentabilidade) no contexto multi-tenant.[8] Utilizar este framework durante o design e a revisão da arquitetura ajuda a evitar armadilhas comuns e a construir uma solução robusta na AWS.

A consideração destes frameworks desde as fases iniciais do projeto ajudará a incorporar as melhores práticas de segurança e conformidade na arquitetura e nos processos operacionais da plataforma SaaS de manutenção preditiva.

7. Conclusão

7.1. Resumo do Projeto e Entregáveis

Este projeto propôs o desenvolvimento de uma plataforma SaaS (Software-as-a-Service) multi-tenant para manutenção preditiva industrial, utilizando Inteligência Artificial e Machine Learning sobre dados coletados em tempo real por sensores IoT (ESP32). A solução visa superar as limitações das abordagens de manutenção tradicionais, permitindo a previsão de falhas de equipamentos, a otimização de cronogramas de manutenção e a redução de custos operacionais para clientes industriais.

O desenvolvimento seguiu a metodologia CRISP-DM de forma iterativa, abrangendo as quatro entregas especificadas na challenge acadêmica:

1. **Design Fundamental:** Definição da metodologia, stack tecnológico (Python,

AWS, PostgreSQL, Scikit-learn, TensorFlow/Keras), arquitetura de pipeline conceitual (local vs. nuvem) e análise inicial de riscos.

2. **Aquisição e Preparação de Dados:** Detalhamento da integração dos sensores ESP32 (MPU6050, DHT22, MQ-XXX, PIR, LDR), estabelecimento de processos de limpeza de dados e Análise Exploratória de Dados (EDA), e definição de User Stories para guiar o desenvolvimento.
3. **Armazenamento e Estrutura de Dados:** Modelagem do banco de dados relacional (usando Oracle SQL Developer Data Modeler), implementação de estratégias de particionamento para dados de séries temporais e definição de uma estratégia de isolamento de dados multi-tenant (modelo Pool recomendado para MVP).
4. **Implementação na Nuvem, IA e MVP:** Detalhamento da arquitetura AWS completa (IoT Core, Kinesis/Lambda, Glue, S3, RDS, SageMaker, API Gateway, CloudWatch, SNS), implementação dos modelos de ML (LSTM para RUL, Random Forest para classificação) incluindo feature engineering e MLOps (SageMaker Pipelines, MLflow), estratégia de implantação (SageMaker MME), desenvolvimento do dashboard (QuickSight), configuração de logging/alertas e geração de relatórios inteligentes com recomendações acionáveis.

O entregável final consiste nesta documentação detalhada e no código-fonte/protótipo funcional (MVP) disponibilizado em um repositório GitHub privado, incluindo o software SaaS acessível (se aplicável), o dashboard de monitoramento e exemplos de relatórios.

7.2. Reflexão sobre o Cumprimento dos Requisitos da Challenge

A solução proposta aborda sistematicamente todos os objetivos e requisitos delineados na "CHALLENGE REPLY":

- **Solução SaaS para Análise Preditiva:** O núcleo do projeto é a plataforma SaaS que utiliza IA/ML para prever falhas (1.3.1).
- **Dados de Sensores em Tempo Real:** A integração com ESP32 e AWS IoT Core/Kinesis garante a coleta e processamento de dados em tempo real (1.3.2.1, 1.3.2.5, 2.3.1).
- **Arquitetura Multi-Tenant:** A estratégia de isolamento de dados (Pool model no RDS, prefixos/políticas no S3, IAM) e o gerenciamento de contexto de tenant atendem ao requisito de multi-tenancy e isolamento (1.3.2).
- **Análise Preditiva e ML:** Foram detalhadas técnicas avançadas (LSTM, Random Forest), feature engineering, tratamento de desbalanceamento e o uso do

SageMaker para o ciclo de vida completo do ML (1.3.2.2, 2.2.1.5, 2.5.1).
- **Dashboard Interativo:** A implementação com QuickSight (ou alternativa) atende à necessidade de visualização em tempo real (1.3.2.3, 1.4.2.2, 2.5.1).
- **Alertas Automatizados:** O uso de CloudWatch Alarms e SNS cumpre o requisito de alertas inteligentes (1.3.2.4, 2.5.1).
- **Relatórios Inteligentes:** A geração automatizada de relatórios com insights e recomendações acionáveis foi detalhada (1.3.2.6, 1.4.2.3, 2.5.1).
- **Tecnologias Especificadas:** Foram incorporadas as tecnologias preferenciais (Python, Análise de Dados, BD, Nuvem AWS, ML, ESP32, Scikit-learn/Keras/TF) e o pipeline solicitado (2.2.1, 2.2.1.6).
- **Entregas Específicas:** Cada entrega (Metodologia/Tecnologias, ESP32/Limpeza/EDA/User Stories, Modelagem BD, Implementação Nuvem/IA/MVP) foi abordada nas seções correspondentes do relatório (2.2.1, 2.3.1, 2.4.1, 2.5.1).
- **Conformidade LGPD:** A estratégia de conformidade, incluindo papéis de controlador/processador e DPAs, foi delineada.
- **Documentação e Protótipo:** O relatório e o link para o repositório/protótipo constituem os entregáveis documentais e de software (1.4.1, 1.4.2).

O projeto, conforme descrito, atende de forma abrangente aos requisitos da challenge, com ênfase nos aspectos técnicos de IA/ML e arquitetura de nuvem solicitados.

7.3. Potenciais Melhorias Futuras

Embora o MVP entregue seja funcional e cumpra os requisitos essenciais, existem diversas avenidas para aprimoramento futuro da plataforma:

- **Modelagem ML Avançada:** Explorar arquiteturas de Deep Learning mais complexas (Transformers para séries temporais, Graph Neural Networks se a relação entre equipamentos for relevante), modelos híbridos, ou técnicas de AutoML para otimizar a seleção e o tuning de modelos.
- **Integração com CMMS:** Desenvolver conectores para sistemas CMMS populares (SAP, Maximo, etc.) para automatizar a criação de ordens de serviço a partir das recomendações preditivas, fechando o ciclo da manutenção.[4]
- **Edge Computing:** Para casos de uso que exigem latência ultrabaixa ou operação offline, implantar partes da lógica de pré-processamento e inferência de ML diretamente no edge (próximo aos equipamentos) usando AWS IoT Greengrass.[19]
- **Análise de Causa Raiz (RCA):** Incorporar técnicas de IA (e.g., explicabilidade de

modelos - SHAP, LIME) para não apenas prever falhas, mas também ajudar a diagnosticar a causa raiz provável.
- **Otimização de Manutenção:** Utilizar técnicas de otimização (e.g., programação linear, reinforcement learning) para recomendar o cronograma ótimo de manutenção considerando múltiplos equipamentos, restrições de recursos (peças, mão de obra) e custos.
- **Personalização Avançada por Tenant:** Permitir maior customização dos dashboards, relatórios e regras de alerta por tenant. Implementar modelos ML verdadeiramente específicos por tenant (requer volume de dados suficiente por tenant).
- **Segurança e Conformidade Aprimoradas:** Buscar certificações formais como ISO 27001 ou SOC 2 para aumentar a confiança dos clientes.[113] Implementar controles de segurança ainda mais granulares.
- **Interface do Usuário:** Desenvolver uma interface web mais rica e intuitiva para a plataforma SaaS.

Este projeto estabelece uma base sólida para uma solução de manutenção preditiva industrial eficaz e escalável, com amplo potencial de evolução e agregação de valor para a indústria.

Referências citadas

1. A Machine Learning Implementation to Predictive Maintenance and Monitoring of Industrial Compressors - MDPI, acessado em maio 8, 2025, https://www.mdpi.com/1424-8220/25/4/1006
2. www.sciencexcel.com, acessado em maio 8, 2025, https://www.sciencexcel.com/articles/tA2EXaU930ap7WTSMfqZLet9YmNHyfDXoDIfywIZ.pdf
3. RUL forecasting for wind turbine predictive maintenance based on deep learning - PMC, acessado em maio 8, 2025, https://pmc.ncbi.nlm.nih.gov/articles/PMC11639378/
4. How to Reduce Downtime with a Predictive Maintenance CMMS, acessado em maio 8, 2025, https://llumin.com/how-to-reduce-downtime-with-a-predictive-maintenance-cmms/
5. AI-Powered Predictive Maintenance: How It Works - LLumin CMMS, acessado em maio 8, 2025, https://llumin.com/ai-powered-predictive-maintenance-how-it-works/
6. AI-Powered Predictive Maintenance Ultimate Guide 2024 | Boost Efficiency, acessado em maio 8, 2025, https://www.rapidinnovation.io/post/ai-for-predictive-maintenance
7. Designing Multi-tenant SaaS Architecture on AWS: Complete Guide - Clickittech,

acessado em maio 8, 2025, https://www.clickittech.com/software-development/multi-tenant-architecture/
8. Pillars of the Well-Architected Framework - SaaS Lens, acessado em maio 8, 2025, https://docs.aws.amazon.com/wellarchitected/latest/saas-lens/the-pillars-of-the-well-architected-framework.html
9. Predictive Maintenance module - Connected Mobility Solution on AWS, acessado em maio 8, 2025, https://docs.aws.amazon.com/solutions/latest/connected-mobility-solution-on-aws/predictive-maintenance-module.html
10. What is CRISP DM? - Data Science PM, acessado em maio 8, 2025, https://www.datascience-pm.com/crisp-dm-2/
11. CRISP-DM - Machine Learning Bookcamp, acessado em maio 8, 2025, https://mlbookcamp.com/article/crisp-dm
12. Top 15 Python Libraries for Data Analytics [2025 updated] | GeeksforGeeks, acessado em maio 8, 2025, https://www.geeksforgeeks.org/python-libraries-for-data-analytics/
13. Top 10 Python Libraries for Data Analytics - Noble Desktop, acessado em maio 8, 2025, https://www.nobledesktop.com/classes-near-me/blog/top-python-libraries-for-data-analytics
14. 9 of the Best Python Machine Learning Tools - Anaconda, acessado em maio 8, 2025, https://www.anaconda.com/guides/python-machine-learning-tools
15. Recommendations for Choosing a Partitioning Strategy, acessado em maio 8, 2025, https://docs.oracle.com/en/database/oracle/oracle-database/19/vldbg/recommendations-partition-strategy.html
16. 4 Partitions, Views, and Other Schema Objects - Oracle Help Center, acessado em maio 8, 2025, https://docs.oracle.com/en/database/oracle/oracle-database/18/cncpt/partitions-views-and-other-schema-objects.html
17. ESP32 IOT Development: A Comprehensive Guide to AWS Cloud and Mobile Integration, acessado em maio 8, 2025, https://www.researchgate.net/publication/389664155_ESP32_IOT_Development_A_Comprehensive_Guide_to_AWS_Cloud_and_Mobile_Integration
18. Deploy a solution that uses Amazon SageMaker to help automate the detection of potential equipment failures and provide recommended actions to take - Predictive Maintenance Using Machine Learning, acessado em maio 8, 2025, https://docs.aws.amazon.com/solutions/latest/predictive-maintenance-using-machine-learning/welcome.html
19. Using AWS IoT for Predictive Maintenance | The Internet of Things ..., acessado em maio 8, 2025, https://aws.amazon.com/blogs/iot/using-aws-iot-for-predictive-maintenance/
20. Building an AWS IoT Core device using AWS Serverless and an ESP32, acessado em maio 8, 2025, https://aws.amazon.com/blogs/compute/building-an-aws-iot-core-device-using-aws-serverless-and-an-esp32/
21. Real-Time Data Streaming with AWS Kinesis - cloudyuga.guru, acessado em maio 8, 2025, https://cloudyuga.guru/blogs/real-time-data-streaming-with-aws-

kinesis/
22. Unlock the Power of Serverless Data Processing with AWS Lambda - Algoscale, acessado em maio 8, 2025, https://algoscale.com/blog/unlock-the-power-of-serverless-data-processing-with-aws-lambda/
23. API Management - Amazon API Gateway - AWS, acessado em maio 8, 2025, https://aws.amazon.com/api-gateway/
24. Top Python AI and Machine Learning Libraries - Codewave, acessado em maio 8, 2025, https://codewave.com/insights/python-ai-machine-learning-libraries/
25. ESP32 MPU-6050 Accelerometer and Gyroscope (Arduino) - Random Nerd Tutorials, acessado em maio 8, 2025, https://randomnerdtutorials.com/esp32-mpu-6050-accelerometer-gyroscope-arduino/
26. How to Use a DHT22 Sensor Module with ESP32 for Temperature and Humidity Monitoring, acessado em maio 8, 2025, https://www.samgalope.dev/2025/01/04/how-to-use-a-dht22-sensor-module-with-esp32-for-temperature-and-humidity-monitoring/
27. Using AWS Lambda with AWS IoT - AWS Documentation, acessado em maio 8, 2025, https://docs.aws.amazon.com/lambda/latest/dg/services-iot.html
28. AWS Glue for Data Integration: Streamlining Cloud-Based ETL Workflows - CloudOptimo, acessado em maio 8, 2025, https://www.cloudoptimo.com/blog/aws-glue-for-data-integration-streamlining-cloud-based-etl-workflows/
29. Multitenant best practices for Amazon RDS Custom for Oracle | AWS ..., acessado em maio 8, 2025, https://aws.amazon.com/blogs/database/multitenant-best-practices-for-amazon-rds-custom-for-oracle/
30. Near real-time dashboards from a source database to a cloud data warehouse on AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/publicsector/near-real-time-dashboards-from-a-source-database-to-a-cloud-data-warehouse-on-aws/
31. Plotly panel - Amazon Managed Grafana - AWS Documentation, acessado em maio 8, 2025, https://docs.aws.amazon.com/grafana/latest/userguide/plotly-panel.html
32. Monitor AWS IoT using CloudWatch Logs, acessado em maio 8, 2025, https://docs.aws.amazon.com/iot/latest/developerguide/cloud-watch-logs.html
33. Architecting SaaS on AWS - Ibexlabs, acessado em maio 8, 2025, https://www.ibexlabs.com/architecting-saas-on-aws/
34. The Top 8 IT/OT/IoT Security Challenges and How to Solve Them | Balbix, acessado em maio 8, 2025, https://www.balbix.com/insights/addressing-iot-security-challenges/
35. Top 10 Vulnerabilities that Make IoT Devices Insecure - CyberArk, acessado em maio 8, 2025, https://www.cyberark.com/resources/blog/top-10-vulnerabilities-that-make-iot-devices-insecure
36. Securing Machine Learning Pipelines: Best Practices in Amazon SageMaker,

acessado em maio 8, 2025, https://tutorialsdojo.com/securing-machine-learning-pipelines-best-practices-in-amazon-sagemaker/
37. Brazil Data Privacy - Amazon Web Services (AWS), acessado em maio 8, 2025, https://aws.amazon.com/compliance/brazil-data-privacy/
38. LGPD Compliance Checklist: The Ultimate Guide for 2025 - Captain ..., acessado em maio 8, 2025, https://captaincompliance.com/education/lgpd-compliance-checklist/
39. The Ultimate Guide to the LGPD | Resource | DataGuidance, acessado em maio 8, 2025, https://www.dataguidance.com/resource/ultimate-guide-lgpd
40. LGPD- Compliance guide | ManageEngine EventLog Analyzer, acessado em maio 8, 2025, https://www.manageengine.com/products/eventlog/compliance/lgpd.html
41. Understanding LGPD: Brazil's Comprehensive Data Protection Framework, acessado em maio 8, 2025, https://www.compliancehub.wiki/understanding-lgpd-brazil-privacy-laws/
42. Building a Multi-Tenant Enterprise SaaS Application on AWS ..., acessado em maio 8, 2025, https://frontegg.com/guides/building-a-multi-tenant-enterprise-saas-application-on-aws
43. MPU6050 tilt demonstration with LEDs - EzloPi, acessado em maio 8, 2025, https://pi.ezlo.com/tutorials-and-kits/mpu6050-tilt-demonstration-with-leds/
44. Send data measured by DHT22 sensor connected to ESP32 via Bluetooth - Robotique Site, acessado em maio 8, 2025, https://www.robotique.site/tutorial/send-data-measured-by-dht22-sensor-connected-to-esp32-via-bluetooth/
45. ESP32 MQ135-Gas Sensor(Air Quality Index) - Group 9 : 8 Steps - Instructables, acessado em maio 8, 2025, https://www.instructables.com/ESP32-MQ135-Gas-SensorAir-Quality-Index/
46. ESP32 With An MQ-135 Gas Sensor - Iotwebplanet.com, acessado em maio 8, 2025, https://www.iotwebplanet.com/esp32-with-an-mq-135-gas-sensor/
47. HOW to Use MQ-2 Gas Sensor with ESP32 - Keyestudio, acessado em maio 8, 2025, https://www.keyestudio.com/blog/how-to-use-mq-2-gas-sensor-with-esp32-299
48. Lesson 04: Gas Sensor Module (MQ-2) - SunFounder's Documentations!, acessado em maio 8, 2025, https://docs.sunfounder.com/projects/umsk/en/latest/04_pi_pico/pico_lesson04_mq2.html
49. Lesson 12: PIR Motion Module (HC-SR501) - SunFounder's Documentations!, acessado em maio 8, 2025, https://docs.sunfounder.com/projects/umsk/en/latest/04_pi_pico/pico_lesson12_pir_motion.html
50. PIR Sensors with ESP32 - Sam Galope, acessado em maio 8, 2025, https://www.samgalope.dev/2025/01/13/pir-sensors-with-esp32/
51. ESP32/ESP8266 Analog Readings with MicroPython - Random Nerd Tutorials,

acessado em maio 8, 2025, https://randomnerdtutorials.com/esp32-esp8266-analog-readings-micropython/
52. User Stories | Examples and Template - Atlassian, acessado em maio 8, 2025, https://www.atlassian.com/agile/project-management/user-stories
53. Multitenancy on Amazon RDS - SaaS Storage Strategies, acessado em maio 8, 2025, https://docs.aws.amazon.com/whitepapers/latest/multi-tenant-saas-storage-strategies/multitenancy-on-rds.html
54. Guidance for Multi-Tenant Architectures on AWS, acessado em maio 8, 2025, https://aws.amazon.com/solutions/guidance/multi-tenant-architectures-on-aws/
55. Partitioning and Isolating Multi-Tenant SaaS Data with Amazon S3 - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/apn/partitioning-and-isolating-multi-tenant-saas-data-with-amazon-s3/
56. Process Streaming Data with Kinesis Analytics - AWS, acessado em maio 8, 2025, https://aws.amazon.com/awstv/watch/dd4e9e0d092/
57. Understanding serverless data processing - AWS Documentation, acessado em maio 8, 2025, https://docs.aws.amazon.com/serverless/latest/devguide/serverless-usecases.html
58. Enhancing Serverless Data Pipelines: Leveraging AWS Lambda, Step Functions, and Kinesis for Real-Time Analytics - ResearchGate, acessado em maio 8, 2025, https://www.researchgate.net/publication/390947002_Enhancing_Serverless_Data_Pipelines_Leveraging_AWS_Lambda_Step_Functions_and_Kinesis_for_Real-Time_Analytics
59. Harnessing AWS Glue for Real-Time Data Processing and Analytics - DataTerrain, acessado em maio 8, 2025, https://dataterrain.com/aws-glue-real-time-data-processing-analytics
60. Cost Optimization Strategies for Amazon SageMaker - CloudThat Resources, acessado em maio 8, 2025, https://www.cloudthat.com/resources/blog/cost-optimization-strategies-for-amazon-sagemaker
61. Design patterns for multi-tenant access control on Amazon S3 | AWS Storage Blog, acessado em maio 8, 2025, https://aws.amazon.com/blogs/storage/design-patterns-for-multi-tenant-access-control-on-amazon-s3/
62. Machine Predictive Maintenance with MLOps - Deployed on AWS - Analytics Vidhya, acessado em maio 8, 2025, https://www.analyticsvidhya.com/blog/2025/02/machine-predictive-maintenance-with-mlops/
63. Centrally track and manage your model versions in Amazon SageMaker - YouTube, acessado em maio 8, 2025, https://www.youtube.com/watch?v=KzwSVsZ4nlc
64. Managing your machine learning lifecycle with MLflow and Amazon ..., acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/managing-your-machine-learning-lifecycle-with-mlflow-and-amazon-sagemaker/
65. Deploying multi-tenant machine learning models on AWS | Chaos Gears,

acessado em maio 8, 2025, https://chaosgears.com/blog/deploying-machine-learning-models-in-saas-applications-on-aws
66. Run multiple deep learning models on GPU with Amazon ... - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/save-on-inference-costs-by-using-amazon-sagemaker-multi-model-endpoints/
67. Amazon API Gateway - AWS Documentation, acessado em maio 8, 2025, https://docs.aws.amazon.com/apigateway/latest/developerguide/welcome.html
68. Building Dashboards for Multi-tenanted SaaS Products using AWS Quicksight, acessado em maio 8, 2025, https://numinolabs.com/design/building-dashboards-for-multi-tenanted-saas-products-using-aws-quicksight/
69. Support multi-tenant applications for SaaS environments using Amazon QuickSight - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/business-intelligence/support-multi-tenant-applications-for-saas-environments-using-amazon-quicksight/
70. Logging and Monitoring - AWS IoT Core, acessado em maio 8, 2025, https://docs.aws.amazon.com/iot/latest/developerguide/security-logging.html
71. Implementing a Multi-Tenant MLaaS Build Environment with ... - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/apn/implementing-a-multi-tenant-mlaas-build-environment-with-amazon-sagemaker-pipelines/
72. SageMaker Multi-Model vs Multi-Container Endpoints | Towards Data Science, acessado em maio 8, 2025, https://towardsdatascience.com/sagemaker-multi-model-vs-multi-container-endpoints-304f4c151540/
73. Example IAM identity-based policies - AWS Identity and Access ..., acessado em maio 8, 2025, https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_examples.html
74. CNN-LSTM Hybrid Deep Learning Model for Remaining Useful Life Estimation - arXiv, acessado em maio 8, 2025, https://arxiv.org/pdf/2412.15998?
75. Predict Remaining Useful Life (RUL) - MATLAB & Simulink - MathWorks, acessado em maio 8, 2025, https://www.mathworks.com/help/predmaint/predict-remaining-useful-life.html
76. Prediction of the Remaining Useful Life of Bearings Through CNN-Bi-LSTM-Based Domain Adaptation Model - MDPI, acessado em maio 8, 2025, https://www.mdpi.com/1424-8220/24/21/6906
77. (PDF) Attention-Based LSTM Network for Rotatory Machine Remaining Useful Life Prediction - ResearchGate, acessado em maio 8, 2025, https://www.researchgate.net/publication/343035060_Attention-Based_LSTM_Network_for_Rotatory_Machine_Remaining_Useful_Life_Prediction
78. arxiv.org, acessado em maio 8, 2025, https://arxiv.org/pdf/2412.15998
79. A Guide to Random Forest in Machine Learning | EJable, acessado em maio 8, 2025, https://www.ejable.com/tech-corner/ai-machine-learning-and-deep-learning/random-forest-in-machine-learning/
80. Random Survival Forests - randomForestSRC, acessado em maio 8, 2025,

https://www.randomforestsrc.org/articles/survival.html
81. Journal of Artificial Intelligence, Machine Learning and Data Science - urfjournals.org — Virtualmin, acessado em maio 8, 2025, https://urfjournals.org/open-access/predictive-maintenance-testing-in-machine-learning-combining-manual-insights-java-programming-and-data-science-for-automation.pdf
82. USING RANDOM FORESTS AND INTERNET OF THINGS SENSORS TO OPTIMIZE PREDICTIVE MAINTENANCE IN SMART FACTORIES - Computer Research and Development, acessado em maio 8, 2025, https://journalcrd.org/wp-content/uploads/6-CRD2408.pdf
83. Applied Predictive Maintenance Part 4 of 6: Machine Learning Workflows with Sci-Kit Learn to Build Predictive Maintenance Models - Patterson Consulting, acessado em maio 8, 2025, https://pattersonconsultingtn.com/blog/predictive_maintenance_w_snowflake_ml_part_4_modeling.html
84. How to use AI for Predictive Maintenance - Mercity AI, acessado em maio 8, 2025, https://www.mercity.ai/blog-post/use-ai-for-predictive-maintenance
85. Handling Imbalanced Data: 7 Innovative Techniques for Success, acessado em maio 8, 2025, https://datasciencedojo.com/blog/techniques-to-handle-imbalanced-data/
86. Cost-Sensitive Learning for Imbalanced Classification - MachineLearningMastery.com, acessado em maio 8, 2025, https://machinelearningmastery.com/cost-sensitive-learning-for-imbalanced-classification/
87. Workflows for Machine Learning - Amazon SageMaker Pipelines, acessado em maio 8, 2025, https://aws.amazon.com/sagemaker/pipelines/
88. Optimizing costs for machine learning with Amazon SageMaker - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/optimizing-costs-for-machine-learning-with-amazon-sagemaker/
89. Metrics for evaluating performance of prognostic techniques - ResearchGate, acessado em maio 8, 2025, https://www.researchgate.net/publication/251867162_Metrics_for_evaluating_performance_of_prognostic_techniques
90. Predictive Model Evaluation for PHM - CiteSeerX, acessado em maio 8, 2025, https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=79f48f105ea55e3e3c674d98fe43e9caa7a5f7f0
91. Model hosting patterns in Amazon SageMaker, Part 1: Common design patterns for building ML applications on Amazon SageMaker | AWS Machine Learning Blog, acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/model-hosting-patterns-in-amazon-sagemaker-part-1-common-design-patterns-for-building-ml-applications-on-amazon-sagemaker/
92. Run ML inference on unplanned and spiky traffic using Amazon SageMaker multi-model endpoints - HKU SPACE AI Hub, acessado em maio 8, 2025,

https://aihub.hkuspace.hku.hk/2024/02/20/run-ml-inference-on-unplanned-and-spiky-traffic-using-amazon-sagemaker-multi-model-endpoints/
93. How to scale machine learning inference for multi-tenant SaaS use ..., acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/how-to-scale-machine-learning-inference-for-multi-tenant-saas-use-cases/
94. Multi-Container Endpoints with Hugging Face Transformers and Amazon SageMaker, acessado em maio 8, 2025, https://www.philschmid.de/sagemaker-huggingface-multi-container-endpoint
95. LLM model inference deployment models and pricing comparison | AWS re:Post, acessado em maio 8, 2025, https://repost.aws/questions/QUTRgMcljMTpWQDqhKOoiKWA/llm-model-inference-deployment-models-and-pricing-comparison
96. AWS Sagemaker or AWS EC2 for llm model training - Reddit, acessado em maio 8, 2025, https://www.reddit.com/r/aws/comments/1c9q1gx/aws_sagemaker_or_aws_ec2_for_llm_model_training/
97. AI-Driven Predictive Analytics: A Futurism Guide, acessado em maio 8, 2025, https://www.futurismtechnologies.com/guides/ai-driven-predictive-analytics-a-futurism-guide/
98. Using predictive model output data - IBM, acessado em maio 8, 2025, https://www.ibm.com/docs/en/mapms/3_cloud?topic=data-using-predictive-model-output
99. Smart CMMS Software Solutions / Maintenance App - CheckProof, acessado em maio 8, 2025, https://www.checkproof.com/maintenance/cmms/
100. Deploy Models with SageMaker: A Complete Guide - BytePlus, acessado em maio 8, 2025, https://www.byteplus.com/en/topic/536443
101. A/B Testing ML models in production using Amazon SageMaker - AWS, acessado em maio 8, 2025, https://aws.amazon.com/blogs/machine-learning/a-b-testing-ml-models-in-production-using-amazon-sagemaker/
102. What is the LGPD in Brazil and how can companies be compliant? - Didomi, acessado em maio 8, 2025, https://www.didomi.io/blog/what-is-the-lgpd-brazil-and-how-can-companies-be-compliant
103. Data protection laws in Brazil, acessado em maio 8, 2025, https://www.dlapiperdataprotection.com/index.html?t=law&c=BR
104. LGPD Compliance: Checklist & Best Practices - Mandatly, acessado em maio 8, 2025, https://mandatly.com/lgpd-compliance/lgpd-compliance-checklist-best-practices
105. What is Brazil's LGPD Compliance? - Securiti, acessado em maio 8, 2025, https://securiti.ai/what-is-lgpd/
106. acessado em dezembro 31, 1969, https://www.gov.br/anpd/pt-br/documentos-e-publicacoes/guia-orientativo-para-definicoes-dos-agentes-de-tratamento-de-dados-pessoais-e-do-encarregado.pdf
107. Data Processing Agreements under the LGPD - first privacy, acessado em maio

8, 2025, https://www.first-privacy.com/special-markets/brazil/data-processing-agreements-under-the-lgpd
108. The Ultimate Guide to LGPD Compliance | Blog | OneTrust, acessado em maio 8, 2025, https://www.onetrust.com/blog/the-ultimate-guide-to-lgpd-compliance/
109. LGPD Brazil Compliance Solutions - Ground Labs, acessado em maio 8, 2025, https://www.groundlabs.com/compliance/lgpd/
110. Brazilian legal framework on automated decision-making | International Bar Association, acessado em maio 8, 2025, https://www.ibanet.org/Brazilian-legal-framework-on-automated-decision-making
111. An LGPD Compliance Inspection Checklist to Assess IoT Solutions (FSE 2024 - Industry Papers), acessado em maio 8, 2025, https://2024.esec-fse.org/details/fse-2024-industry/32/An-LGPD-Compliance-Inspection-Checklist-to-Assess-IoT-Solutions
112. AWS re:Inforce 2023 - Amazon S3 encryption and access control best practices (DAP306), acessado em maio 8, 2025, https://www.youtube.com/watch?v=ukk3R5DrdSs
113. 6 IT Security Frameworks for Cybersecurity, acessado em maio 8, 2025, https://www.legitsecurity.com/aspm-knowledge-base/top-it-security-frameworks
114. Integrating SaaS Solutions into Cybersecurity Frameworks - Cyber Management Alliance, acessado em maio 8, 2025, https://www.cm-alliance.com/cybersecurity-blog/integrating-saas-solutions-into-cybersecurity-frameworks
115. acessado em dezembro 31, 1969, https.www.aicpa-cima.com/resources/download/soc-2-reporting-on-an-examination-of-controls-at-a-service-organization-relevant-to-security-availability-processing-integrity-confidentiality-or-privacy