RFC: HospiCast

Título: Previsão Inteligente de Demandas Hospitalares

Data: 01/07/2025

Aluno: Mateus de Faria da Silva

Curso: Engenharia de Software

1. Introdução

O HospiCast é um projeto de TCC em desenvolvimento que visa criar uma plataforma preditiva para antecipar a demanda por atendimentos hospitalares (emergência e internações) no Brasil, com foco na realidade do Sistema Único de Saúde (SUS). Utilizando o modelo de séries temporais Prophet, o sistema analisa dados históricos do DATASUS, fatores climáticos (ex.: chuvas) e sazonalidades (ex.: feriados como Carnaval) para prever o volume de atendimentos com 30 dias de antecedência. A solução oferece um dashboard interativo para gestores hospitalares e uma API REST para integração externa, otimizando a alocação de leitos, a gestão de escalas médicas e a redução de filas, que atualmente ultrapassam 3 horas em média (CNES/ANS, 2024). O projeto será validado com dados fictícios e testes piloto, com entrega do MVP em setembro de 2025.

2. Motivação e Justificativa

A superlotação hospitalar é um desafio crítico no Brasil, com ocupação média de UTIs acima de 80% e atrasos no pronto-socorro que elevam em 15% as complicações em emergências (Ministério da Saúde, 2024). O HospiCast auxilia no planejamento operacional de hospitais, especificamente na:

- **Alocação de leitos**: Antecipação de picos de demanda para evitar sobrecarga.
- **Gestão de escalas médicas**: Ajuste de turnos com base em previsões.
- **Distribuição de recursos**: Otimização de insumos como medicamentos e equipamentos.
- **Políticas públicas**: Dados para secretarias de saúde planejarem investimentos.

Comparação com Benchmarks:

- **BedDemand (NHS)**: Previsão de ocupação de leitos no Reino Unido, focada em hospitais de grande porte, sem adaptação a sazonalidades brasileiras (ex.: Carnaval) ou dados climáticos.
- **Google COVID-19 Forecast**: Previu casos de COVID-19 globalmente, mas com foco em pandemias, sem aplicação à rotina hospitalar do SUS.
- **HospiCast**: Adaptado ao SUS, integra dados climáticos (ex.: chuvas no Sudeste) e feriados locais, com interface acessível para hospitais de pequeno e médio porte.

3. Objetivos

Objetivo Principal: Desenvolver, até 30/09/2025, uma plataforma que preveja o volume diário de atendimentos hospitalares (emergência e internações) para 30 dias, com precisão de MAPE ≤15%, utilizando dados históricos e climáticos, no contexto do SUS.

Objetivos Secundários (SMART):

- **Essenciais**:
    - Estruturar, até 15/07/2025, um dataset com dados diários de 5 hospitais (DATASUS), limpo e validado, contendo pelo menos 2 anos de histórico.
    - Implementar, até 15/08/2025, o modelo Prophet com MAPE ≤10% (ou ≤15% em cenários complexos) para previsões de 30 dias.
    - Desenvolver, até 15/08/2025, uma API REST documentada via Swagger, com latência média <200ms para 50 requisições/segundo.
    - Configurar, até 31/08/2025, autenticação JWT para proteger API e dashboard.
    - Garantir, até 15/09/2025, cobertura de testes automatizados >80%, medida via pytest/coverage.py.
- **Desejáveis**:
    - Minimizar custos, até 30/09/2025, utilizando ferramentas de código aberto (ex.: Python, PostgreSQL) e buscando parcerias com hospitais para validação sem ônus financeiro.
    - Implementar, até 30/09/2025, exportação de relatórios em PDF com previsões detalhadas.

4. Escopo

O projeto HospiCast, abrange:

- **Incluso**:
    - Previsão do volume diário de atendimentos (emergência e internações) para 30 dias, com base em dados históricos (DATASUS), climáticos e sazonais.
    - Dashboard interativo com filtros por cidade e período.
    - API REST para consulta de previsões.
    - Autenticação segura via JWT.
- **Excluído**:
    - Previsão de patologias específicas ou diagnósticos clínicos.
    - Integração com sistemas hospitalares legados (ex.: prontuários eletrônicos).
    - Previsões com granularidade horária.
- **Expansão Futura**:
    - Integração com APIs climáticas externas (ex.: INMET).
    - Previsão por especialidade médica (ex.: cardiologia, pediatria).

5. Especificação Técnica

O HospiCast utiliza um pipeline técnico que processa dados históricos e climáticos, gera previsões com o modelo Prophet e entrega resultados via API e dashboard. O desenvolvimento prioriza robustez, usabilidade e conformidade com padrões acadêmicos.

Dataset Mínimo: Dados diários de atendimentos de 5 hospitais (DATASUS), cobrindo 2 anos, tratados no Sprint 1 para remover nulos (interpolação linear) e outliers (método IQR).

- ### 5.1 Requisitos Funcionais

ID	Requisito	Descrição	Exemplo de Uso	Prioridade
RF 01	Cadastro de hospitais	Permitir o cadastro de hospitais com informações como nome, cidade, estado e capacidade de leitos, armazenando os dados no banco de dados.	Gestor cadastra "Hospital Municipal, São Paulo, SP, 100 leitos" no dashboard.	Crítica
RF 02	Importação de dados históricos	Importar arquivos CSV com dados diários de atendimentos do DATASUS, validando e armazenando no banco de dados para uso no modelo preditivo.	Gestor importa arquivo CSV com atendimentos de 2023-2024.	Crítica
RF 03	Integração de dados climáticos	Obter dados climáticos (ex.: chuva, temperatura) via API pública (ex.: INMET) para incorporar ao modelo de previsão.	Sistema obtém previsão de chuva para São Paulo em 15/08/2025.	Crítica
RF	Geração de	Produzir previsões do volume	Sistema prevê 150 ± 10	Crítica

04	previsões	diário de atendimentos (emergência e internações) para os próximos 30 dias usando o modelo Prophet.	atendimentos para 15/08/2025.	
RF 05	Filtros no dashboard	Permitir que gestores filtrem previsões por cidade e período no dashboard interativo.	Gestor filtra previsões para São Paulo, 01/08 a 30/08/2025.	Desejável
RF 06	Atualização de dados	Permitir que gestores autorizados atualizem manualmente dados históricos de atendimentos no sistema.	Gestor atualiza dados de atendimentos de julho/2025.	Desejável

5.2 Requisitos Não-Funcionais

- **RNF01 - Rapidez**: O sistema deve responder em menos de 1 segundo para consultas típicas (ex.: previsão para um hospital), medido por testes Locust com 50 requisições/segundo.
- **RNF02 - Disponibilidade**: O sistema deve estar disponível 24 horas, 7 dias da semana, com tempo de atividade ≥99,9%, monitorado via Grafana.
- **RNF03 - Segurança**: Todas as comunicações devem usar HTTPS, e o acesso à API/dashboard exige autenticação JWT com tokens de 24 horas.
- **RNF04 - Usabilidade**: O dashboard deve ser fácil de usar, com interface clara para gestores não técnicos, testada com 5 usuários fictícios.
- **RNF05 - Conformidade**: Dados hospitalares devem ser anonimizados, respeitando a LGPD (Lei nº 13.709/2018).
- **RNF06 - Responsividade**: O frontend deve se adaptar a telas de 320px a 1920px (ex.: celular, desktop).
- **RNF07 - Exportação**: Relatórios em PDF devem ser gerados em menos de 5 segundos, com layout claro e legível.
- **RNF08 - Manutenibilidade**: O código deve ser modular, com documentação em docstrings, facilitando futuras atualizações.

5.3 Requisitos de IA

- Prophet modelará sazonalidade semanal/ anual, gerando previsões 30 dias à frente.
- Métricas: MAPE, RMSE, sMAPE.
- Gráfico exemplo será incluído para ilustrar sazonalidade.

5.4 Design, Resiliência e Observabilidade

- Arquitetura MVC em containers: React, FastAPI, Prophet, PostgreSQL.
- Retries e circuit-breaker HTTPX. Alertas no dashboard indicarão picos e necessidade de

retreinamento.
- Retreinamento mensal ou se MAPE >15%.
- Modelos versionados em `models/` (MLflow futuro). Logs estruturados, traces e CI/CD via GitHub Actions.

5.5 Funcionamento do Modelo de IA (Prophet)

O núcleo de previsão do HospiCast será um modelo de séries temporais desenvolvido com o **Facebook Prophet**, uma biblioteca especializada para prever valores futuros com base em dados históricos, ajustando sazonalidades e tendências.

- O Prophet irá prever o **volume diário de atendimentos hospitalares** para os próximos 30 dias, permitindo antecipar superlotações, falta de leitos e necessidade de ajuste nas equipes.

Como o Prophet funciona:

Matematicamente, o Prophet decompõe a série em três principais componentes:

$y(t)=g(t)+s(t)+h(t)+\varepsilon_t$y(t) = g(t) + s(t) + h(t) + \varepsilon_ty(t)=g(t)+s(t)+h(t)+εt

- **g(t)**: tendência de longo prazo (ex.: aumento geral no fluxo de pacientes)
- **s(t)**: efeitos sazonais (ex.: ciclos semanais e anuais)
- **h(t)**: efeitos de feriados ou eventos especiais (ex.: campanhas de vacinação)
- ε_t: erro aleatório

O modelo detecta automaticamente **changepoints** — pontos em que o padrão de crescimento ou declínio muda significativamente.

Periodicidade e granularidade:

- O modelo usará dados com **granularidade diária** por hospital, abrangendo ao menos **2 anos** no dataset inicial.
- Permitirá previsões para **30 dias futuros**.

Pipeline Técnico

Etapa	Tecnologia	Descrição
ETL	pandas, numpy	Ler dados do DATASUS / CNES, limpar nulos, tratar outliers.
Modelagem	Prophet	Ajustar dados históricos, calibrar sazonalidade semanal/ anual.
Previsão	Prophet	Gerar previsões com yhat, yhat_lower, yhat_upper.
Validação	scikit-learn	Calcular MAPE, RMSE, sMAPE.

Export	PostgreSQL / JSON	Salvar previsões para API e dashboard.

Retreinamento e robustez

- O modelo será **re-treinado mensalmente** ou sempre que o erro MAPE ultrapassar 15%, conforme definido pela política de governança.
- Possui mecanismos para lidar com outliers e incluir feriados no cálculo, mantendo previsões confiáveis.

API e dashboard

- Endpoint /forecast no FastAPI executará o pipeline Prophet e retornará as previsões em JSON.
- O dashboard (React) consumirá essas previsões para visualização gráfica.

Métricas de Avaliação

MAPE: erro percentual médio, ex.: 'erro 8%'.
RMSE: penaliza grandes desvios.
sMAPE: percentual simétrico, mais estável.

Fórmulas aproximadas:
MAPE = (1/n) Σ| (y - y_hat)/y |
RMSE = sqrt((1/n) Σ(y - y_hat)²)
sMAPE = (1/n) Σ| y - y_hat | / ((|y|+|y_hat|)/2)

5.6 Caso de Uso: Cadastrar Hospital

Ator Primário: Funcionário administrativo
Pré-condição: Usuário autenticado

Fluxo principal:
1. Funcionário acessa 'Cadastro de Hospital'.
2. Sistema exibe formulário: Nome, Cidade, Estado, Capacidade.
3. Funcionário preenche e clica 'Salvar'.
4. Sistema valida campos obrigatórios, exibe erros se houver.
5. Se válido, grava no banco e mostra 'Hospital cadastrado com sucesso'.

Pós-condição: Hospital pronto para associar previsões futuras.
Regras: Não permite duplicar Nome+Cidade; Nome, Cidade e Estado são obrigatórios.

6. Stack Tecnológica

- Python (Prophet, pandas, numpy, scikit-learn)
- FastAPI
- React

- PostgreSQL
- Docker, Swagger, Grafana

7. Segurança

- JWT protege API e painel. Endpoints exigem `Authorization: Bearer <token>`.

8. MVP Prioritário

- **Funcionalidades**:
    - Previsão de 30 dias com Prophet.
    - Dashboard básico com gráficos.
    - API REST (/forecast) com JWT.
- **Pós-MVP**:
    - Integração com APIs climáticas.
    - Relatórios em PDF.

9. Métricas de Sucesso

- **Técnicas**:
    - MAPE ≤10% (ou ≤15%), via scikit-learn.
    - Latência API <200ms para 50 requisições/segundo (Locust).
    - Cobertura de testes >80% (pytest).
- **Usabilidade**:
    - Satisfação ≥80% em testes com 5 gestores fictícios.
    - Adoção: ≥3 hospitais fictícios testando o MVP.

10. Cronograma

Sprint	Período	Entregáveis
1	01/07 - 15/07	Dataset limpo (CSV), script ETL, modelo inicial.
2	16/07 - 31/07	Prophet ajustado, dashboard inicial.
3	01/08 - 15/08	API Swagger, CI/CD (GitHub Actions).

4	16/08 - 31/08	Frontend completo, JWT.
5	01/09 - 15/09	Testes de carga (Locust), hardening, deploy.
6	16/09 - 30/09	Testes piloto com 3 hospitais fictícios, ajustes finais.

11. Riscos e Mitigação

Risco	Prob.	Impacto	Mitigação
Dados indisponíveis (DATASUS)	70%	Alto	Dados sintéticos via Faker; validação com hospitais reais.
MAPE <10% não atingido	50%	Médio	Testar SARIMAX/XGBoost; ajustar hiperparâmetros.
Dashboard confuso	40%	Médio	Testes de usabilidade com 5 personas.
Baixa adesão por hospitais	50%	Médio	Workshops e documentação amigável.
Custos excedem verba	30%	Médio	Uso de ferramentas open-source; parcerias com hospitais.

12. Referências

- https://facebook.github.io/prophet/
- https://fastapi.tiangolo.com/
- https://docs.microsoft.com/aspnet/core/blazor/
- https://scikit-learn.org/
- https://datasus.saude.gov.br/

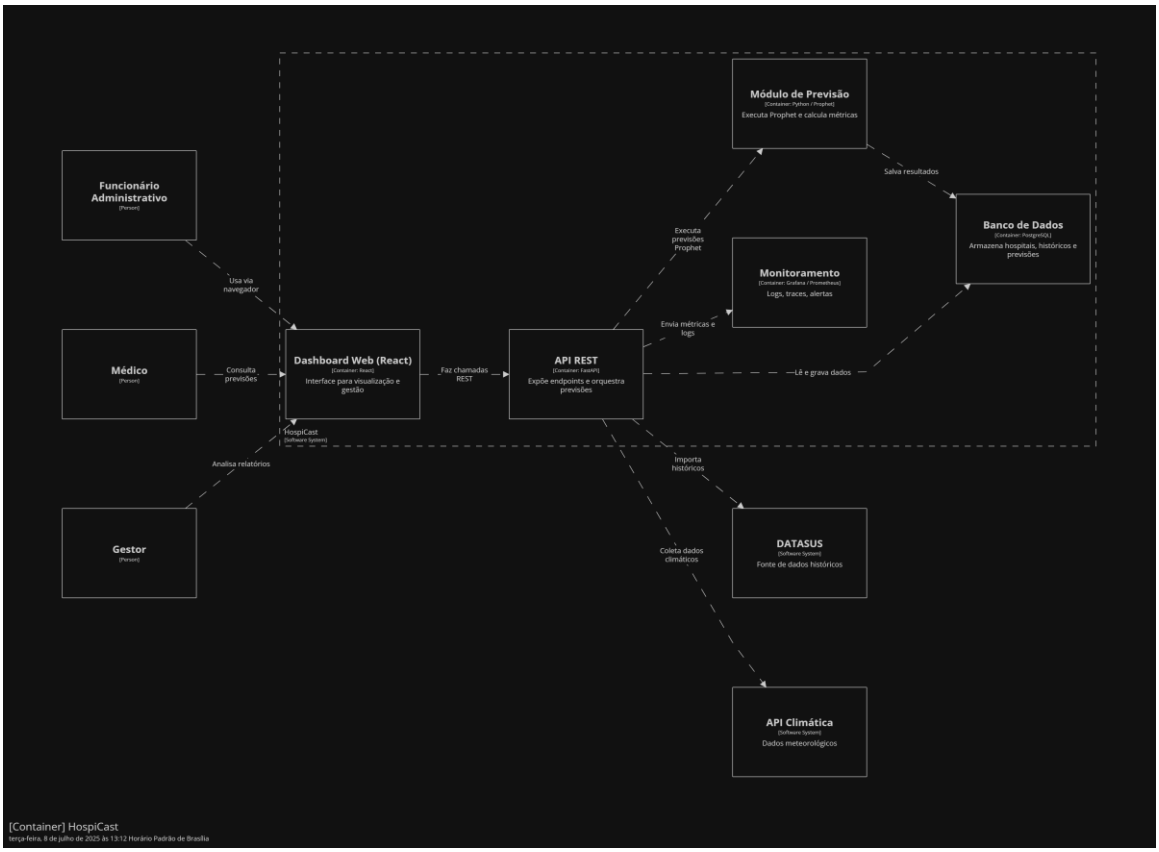
- https://draw.io

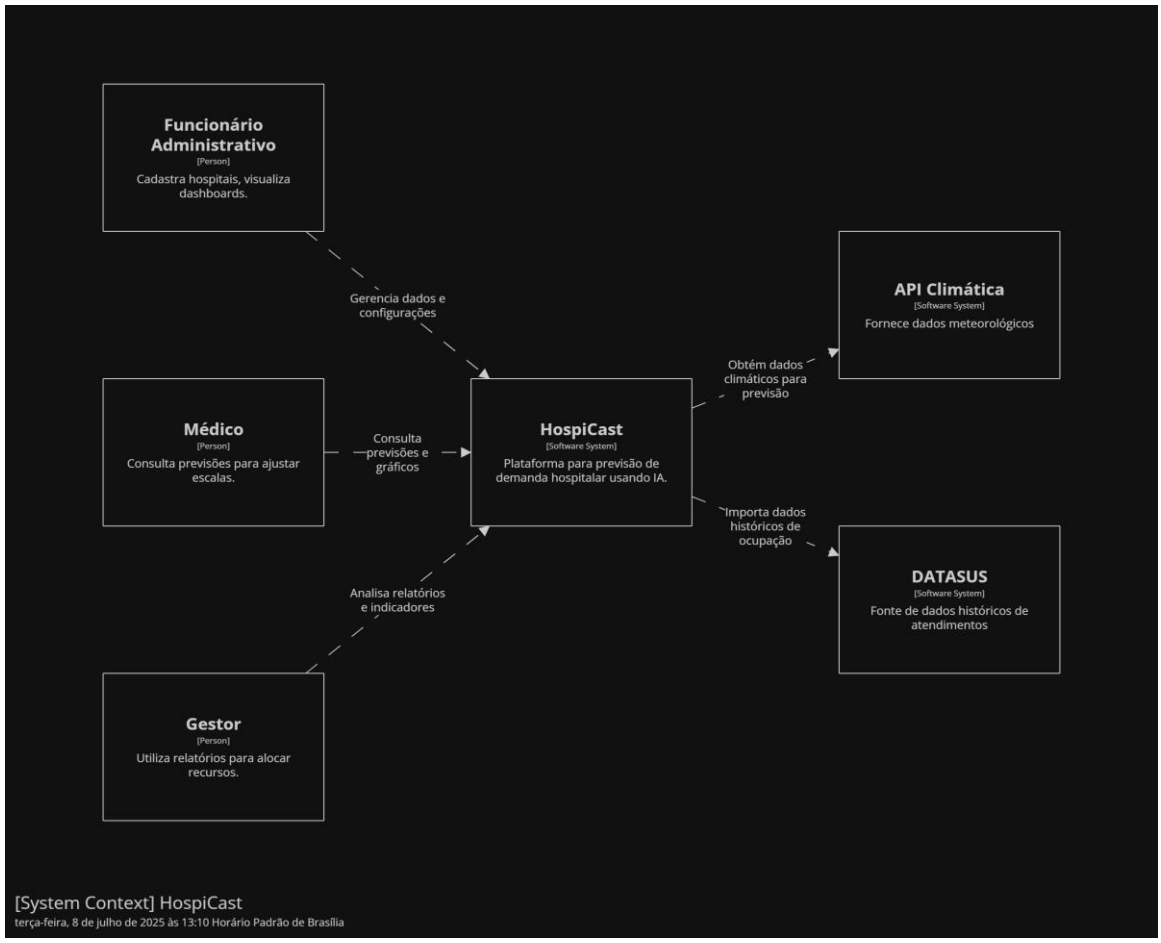
13. Metodologia

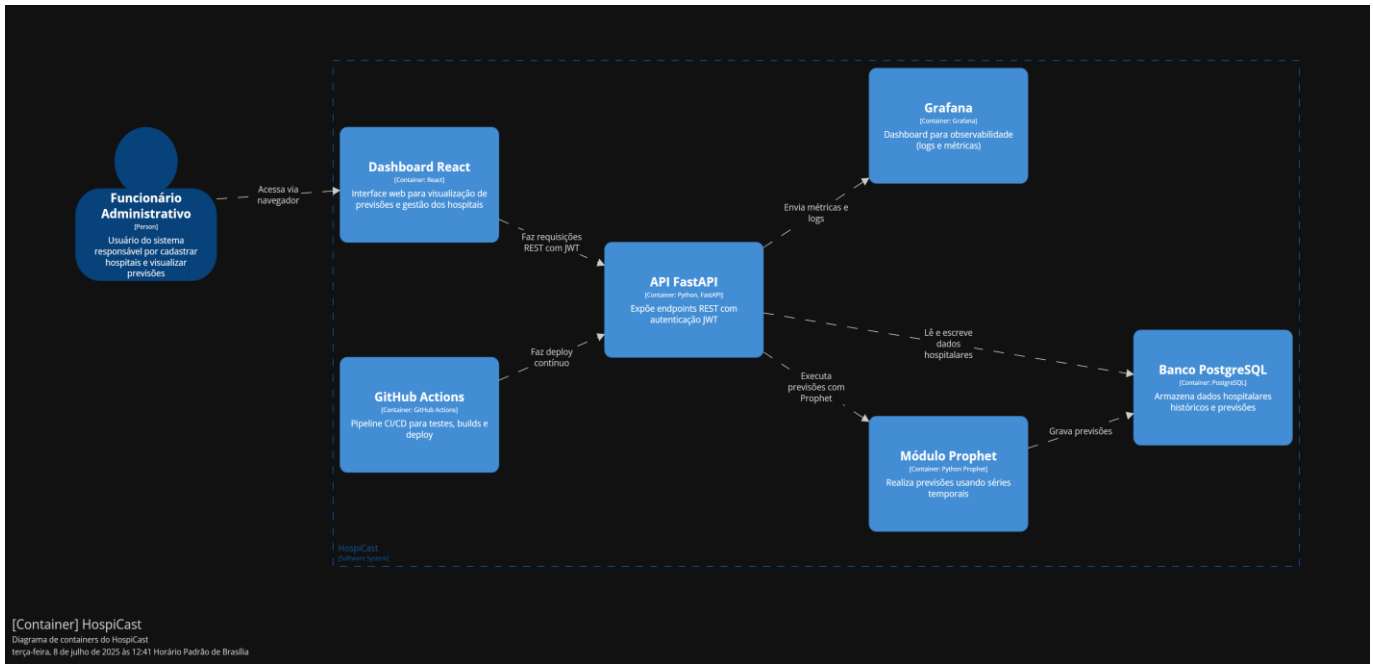
O projeto adota a metodologia ágil com sprints de 15 dias:

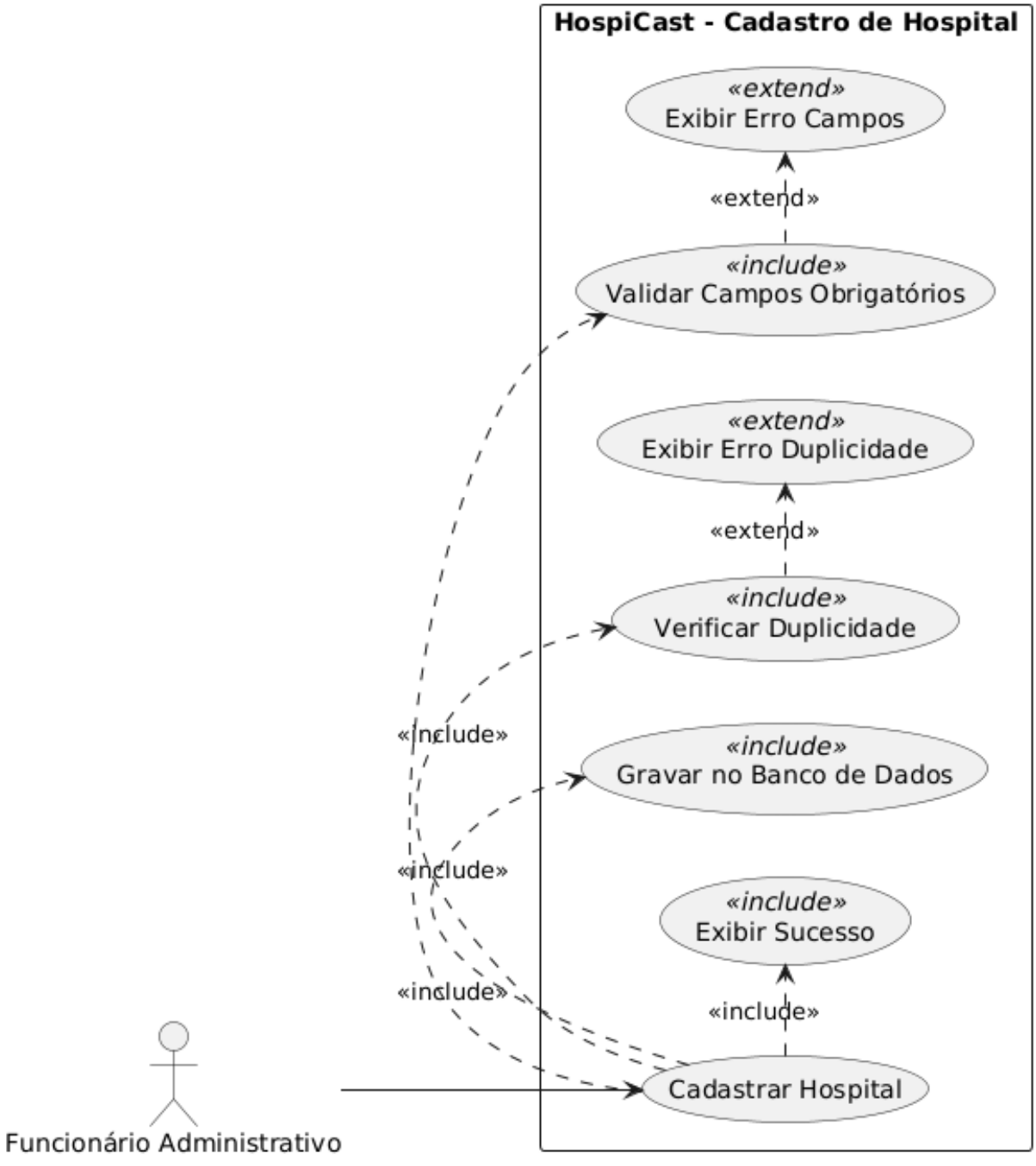
- **Coleta de Dados**: Dados reais (DATASUS) e sintéticos (Faker).
- **Prototipagem**: MVP funcional até Sprint 5.
- **Validação**: Testes piloto com 3 hospitais fictícios; feedback de gestores.
- **Iterações**: Ajustes baseados em métricas e usabilidade.

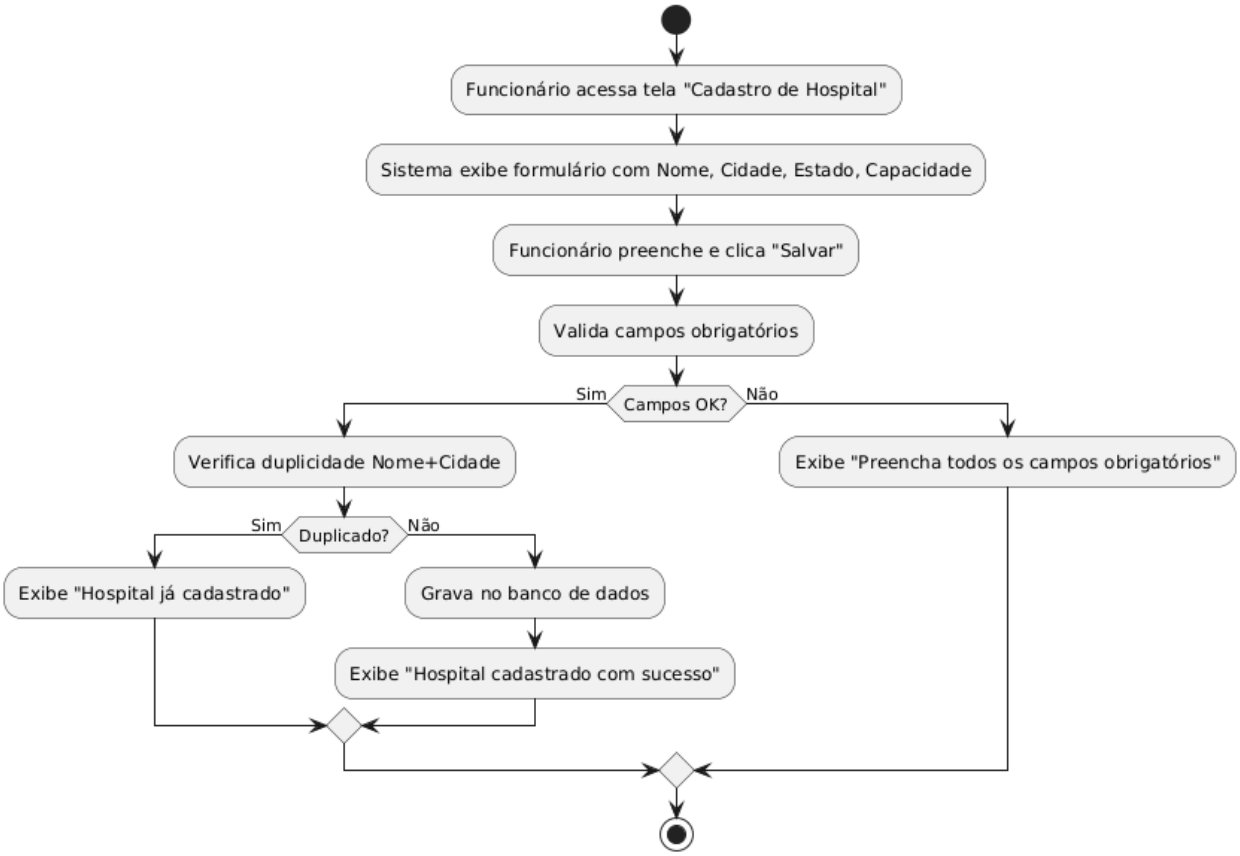
14. Diagramas C4 e Casos de Uso


Módulo de Previsão
[Container: Python / Prophet]
Executa Prophet e calcula métricas
Salva resultados
Banco de Dados
[Container: PostgreSQL]
Armazena hospitais, históricos e previsões
Funcionário Administrativo
[Person]
Usa via navegador
Executa previsões Prophet
Monitoramento
[Container: Grafana / Prometheus]
Logs, traces, alertas
Envia métricas e logs
Médico
[Person]
Consulta previsões
Dashboard Web (React)
[Container: React]
Interface para visualização e gestão
Faz chamadas REST
API REST
[Container: FastAPI]
Expõe endpoints e orquestra previsões
Lê e grava dados
HospiCast
[Software System]
Analisa relatórios
Importa históricos
Gestor
[Person]
Coleta dados climáticos
DATASUS
[Software System]
Fonte de dados históricos
API Climática
[Software System]
Dados meteorológicos
[Container] HospiCast
terça-feira, 8 de julho de 2025 às 13:12 Horário Padrão de Brasília


Funcionário Administrativo
[Person]
Cadastra hospitais, visualiza dashboards.
Gerencia dados e configurações
API Climática
[Software System]
Fornece dados meteorológicos
Obtém dados climáticos para previsão
Médico
[Person]
Consulta previsões para ajustar escalas.
Consulta previsões e gráficos
HospiCast
[Software System]
Plataforma para previsão de demanda hospitalar usando IA.
Importa dados históricos de ocupação
DATASUS
[Software System]
Fonte de dados históricos de atendimentos
Analisa relatórios e indicadores
Gestor
[Person]
Utiliza relatórios para alocar recursos.
[System Context] HospiCast
terça-feira, 8 de julho de 2025 às 13:10 Horário Padrão de Brasília


Funcionário Administrativo
[Person]
Usuário do sistema responsável por cadastrar hospitais e visualizar previsões
Acessa via navegador
Dashboard React
[Container: React]
Interface web para visualização de previsões e gestão dos hospitais
Faz requisições REST com JWT
Grafana
[Container: Grafana]
Dashboard para observabilidade (logs e métricas)
Envia métricas e logs
API FastAPI
[Container: Python, FastAPI]
Expõe endpoints REST com autenticação JWT
Lê e escreve dados hospitalares
Banco PostgreSQL
[Container: PostgreSQL]
Armazena dados hospitalares históricos e previsões
Executa previsões com Prophet
Módulo Prophet
[Container: Python Prophet]
Realiza previsões usando séries temporais
Grava previsões
Faz deploy contínuo
GitHub Actions
[Container: GitHub Actions]
Pipeline CI/CD para testes, builds e deploy
HospiCast
[Software System]
[Container] HospiCast
Diagrama de containers do HospiCast
terça-feira, 8 de julho de 2025 às 12:41 Horário Padrão de Brasília


HospiCast - Cadastro de Hospital
«extend»
Exibir Erro Campos
«extend»
«include»
Validar Campos Obrigatórios
«extend»
Exibir Erro Duplicidade
«extend»
«include»
Verificar Duplicidade
«include»
«include»
Gravar no Banco de Dados
«include»
«include»
Exibir Sucesso
«include»
«include»
Cadastrar Hospital
Funcionário Administrativo


Funcionário acessa tela "Cadastro de Hospital"
Sistema exibe formulário com Nome, Cidade, Estado, Capacidade
Funcionário preenche e clica "Salvar"
Valida campos obrigatórios
Campos OK?
Sim
Não
Verifica duplicidade Nome+Cidade
Exibe "Preencha todos os campos obrigatórios"
Duplicado?
Sim
Não
Exibe "Hospital já cadastrado"
Grava no banco de dados
Exibe "Hospital cadastrado com sucesso"