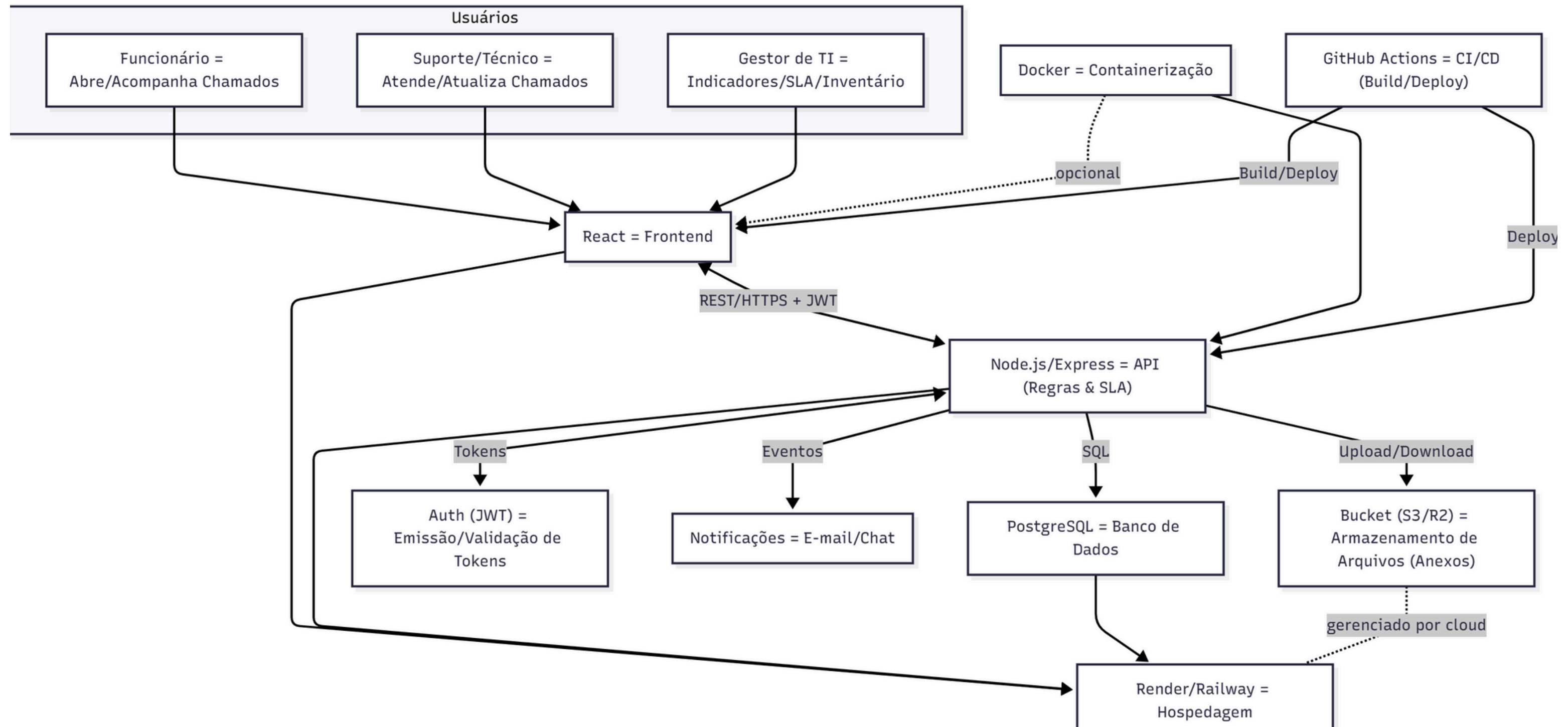


# DIAGRAMA

## EXPLICAÇÃO DA ESTRUTURA



# PEÇAS DA ESTRUTURA

01

## REACT

interface única onde usuário cria, consulta e acompanha chamados

03

## BANCO DE DADOS

Guarda dados estruturados (chamados, usuários, histórico de status, ativos).

05

## AUTH

Autentica e autoriza. Cada acesso leva um token com o perfil. Define o que pode ver e fazer.

02

## API

valida dados, controla status, checa SLA e orquestra integrações.

04

## BUCKET

armazena arquivos (prints, PDFs, logs).

06

## NOTIFICAÇÕES

envia avisos automáticos nos eventos importantes (criação, mudança de status, encerramento).

# FLUXO DE CHAMADO

01

USUÁRIO → FRONTEND  
PREENCHE/CONSULTA SEU  
CHAMADO.

02

FRONTEND ↔ API (HTTPS  
+ JWT)  
TROCA SEGURA DE DADOS  
COM O TOKEN DO  
USUÁRIO.

03

API → POSTGRESQL  
CONSULTA CHAMADO,  
HISTÓRICO, USUÁRIO E  
ATIVO.

04

API → BUCKET: SALVA  
ANEXOS E REGISTRA O LINK  
NO BANCO.  
  
API → NOTIFICAÇÕES:  
DISPARA E-MAIL/CHAT EM  
CADA EVENTO RELEVANTE.

# SLA - CONTROLE DE PRAZOS

01

O Ticket Express trabalha com prazos-alvo definidos por tipo e por prioridade de chamado, como crítico, alto e médio. Assim que um chamado é aberto, o sistema inicia a contagem do tempo de atendimento. Cada mudança de status é registrada com data e hora no histórico do banco de dados, formando uma trilha completa do atendimento. Os status típicos são aberto, em atendimento, resolvido e encerrado, e todas as transições ficam registradas para auditoria.

02

A API recalcula o SLA a cada mudança de status. O tempo conta enquanto o chamado está em atendimento e pode pausar quando estiver aguardando usuário ou terceiro, conforme regras definidas. É possível considerar apenas tempo útil, com horário comercial, fuso e feriados configuráveis. Quando o prazo se aproxima do limite, o sistema emite alerta preventivo; se o prazo estoura, o chamado é marcado fora do SLA e são enviadas notificações de estouro para os responsáveis e para a gestão.

03

Ao encerrar o chamado, a API grava os tempos finais como tempo de primeira resposta, tempo total de atendimento e tempo de resolução. Esses dados alimentam os painéis e relatórios do gestor, que mostra indicadores como tempo médio por prioridade e por categoria, distribuição por técnico, filas e gargalos. Com isso, a gestão consegue priorizar melhor, alocar recursos e melhorar continuamente o processo de suporte.

# OPERAÇÃO, ENTREGA E SEGURANÇA

01

Empacotamos a aplicação com Docker, priorizando a API, para que o sistema rode do mesmo jeito em qualquer ambiente: desenvolvimento, homologação e produção. O processo de integração e entrega contínua é feito com GitHub Actions. A cada mudança no código, a pipeline executa testes automatizados, gera os artefatos de build e publica novas versões do frontend e da API sem intervenção manual. Esse fluxo reduz erros humanos, acelera os lançamentos e permite voltar rapidamente para uma versão anterior.

02

Todo tráfego ocorre em HTTPS, garantindo criptografia de ponta a ponta. A autenticação e a autorização usam JWT como crachá digital, com identidade e permissões embutidas, e aplicamos RBAC para separar o que funcionário, técnico e gestor podem acessar. A API valida e sanitiza todas as entradas para bloquear dados inválidos e ataques comuns. Anexos como imagens, PDFs e logs ficam em bucket na nuvem com links controlados, enquanto o banco armazena apenas a referência do arquivo.