Elissandro Caetano Júnior

Sistema Web para o Monitoramento de Estações de Tratamento de Água no Âmbito do Programa Água Doce

Belo Horizonte, Minas Gerais
2026

Elissandro Caetano Júnior

Sistema Web para o Monitoramento de Estações de Tratamento de Água no Âmbito do Programa Água Doce

Projeto de Aplicação Tecnológica

Universidade Federal de Minas Gerais
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Orientador: Thiago Ferreira de Noronha
Coorientador: Anolan Yamilé Milanés Barrientos

Belo Horizonte, Minas Gerais
2026

Resumo

A escassez de água potável no Semiárido brasileiro representa um dos maiores desafios socioambientais do país. O Programa Água Doce (PAD), instituído pelo Governo Federal em 2003, busca garantir o acesso sustentável à água de qualidade por meio da dessalinização via osmose reversa. Este trabalho apresenta o desenvolvimento de um sistema web com alertas inteligentes para o monitoramento e controle das estações de dessalinização do PAD. A solução integra microcontroladores ESP32 a uma arquitetura web moderna (Node.js no *back-end* e Angular no *front-end*), além de um aplicativo móvel que coleta as leituras dos sensores via Bluetooth e as sincroniza com o servidor. O sistema oferece *dashboards* interativos, controle de acesso multinível (administradores, gestores e operadores) e um módulo de alertas que sinaliza, de forma visual, leituras fora dos limites operacionais. Para assegurar a confiabilidade, foram desenvolvidos testes automatizados cobrindo o *back-end* e o *front-end*. Os resultados mostram a viabilidade de uma solução de baixo custo para reduzir despesas de manutenção, antecipar falhas, diminuir deslocamentos de equipes técnicas e contribuir para a gestão sustentável dos recursos hídricos, ampliando a disponibilidade de água potável às comunidades vulneráveis do Semiárido brasileiro.

Palavras-chave: Internet das Coisas; Monitoramento de Água; Dessalinização; Alertas Inteligentes; ESP32; Sistemas Web; Osmose Reversa; Gestão de Recursos Hídricos.

Abstract

Water scarcity in the Brazilian semi-arid region is one of the country's greatest socio-environmental challenges. The Freshwater Program (PAD), established by the Federal Government in 2003, aims to ensure sustainable access to quality water through reverse osmosis desalination. This work presents the development of a web system with intelligent alerts for monitoring and controlling PAD desalination stations. The solution integrates ESP32 microcontrollers with a modern web architecture (Node.js on the back-end and Angular on the front-end), together with a mobile application that collects sensor readings over Bluetooth and synchronizes them with the server. The system provides interactive dashboards, multi-level access control (administrators, managers, and operators), and an alert module that visually flags readings outside operational limits. To ensure reliability, automated tests were developed covering both the back-end and the front-end. The results demonstrate the feasibility of a low-cost solution to reduce maintenance costs, anticipate failures, minimize technical team displacement, and contribute to sustainable water resource management, increasing the availability of drinking water to vulnerable communities in the Brazilian semi-arid region.

Keywords: Internet of Things; Water Monitoring; Desalination; Intelligent Alerts; ESP32; Web Systems; Reverse Osmosis; Water Resources Management.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

A escassez de água potável no Semiárido brasileiro constitui um dos maiores desafios socioambientais do país. Embora a região disponha de recursos hídricos subterrâneos em abundância, grande parte dessas águas é salobra ou salina (Brasil. Ministério do Meio Ambiente, 2012; Brasil. Ministério da Integração e do Desenvolvimento Regional, 2025), o que inviabiliza o consumo humano direto. Nesse contexto, o Governo Federal instituiu, em 2003, o Programa Água Doce (PAD), coordenado pelo Ministério do Meio Ambiente em parceria com instituições federais, estaduais e municipais. O objetivo central é garantir o acesso sustentável à água de qualidade, por meio da dessalinização via osmose reversa e da gestão comunitária de sistemas descentralizados, atendendo especialmente localidades difusas do Semiárido (Brasil. Ministério do Meio Ambiente, 2012; Saia, 2018).

A dimensão do programa ajuda a entender o desafio envolvido. O PAD articula uma rede de cerca de 200 instituições, distribuídas pelos dez estados do Semiárido, e em 2011 passou a integrar o Plano Brasil sem Miséria, no âmbito do Programa Água para Todos (Brasil. Ministério do Meio Ambiente, 2017b). Sua meta previa a recuperação, a implantação e a gestão de cerca de 1.200 sistemas de dessalinização, com investimentos da ordem de R$ 210 milhões e o benefício de aproximadamente 480 mil pessoas, uma média de 400 pessoas por sistema (Brasil. Ministério do Meio Ambiente, 2017b). Manter tantas unidades funcionando, dispersas por uma região extensa e de difícil acesso, é justamente o ponto em que o monitoramento se torna crítico.

Figura 1 – Açude típico de regiões do semiárido brasileiro

Fonte: Queiroz (2026).

A tecnologia de osmose reversa, utilizada nos dessalinizadores do PAD, remove os sais dissolvidos da água a partir de membranas semipermeáveis (Greenlee et al., 2009; Elimelech and Phillip, 2011). No contexto desta pesquisa, essas membranas são integradas a sistemas

de monitoramento baseados em microcontroladores ESP32, permitindo acompanhar em tempo real as variáveis críticas do processo. A efetividade do tratamento depende do controle rigoroso de parâmetros como condutividade elétrica (indicativa da concentração de sólidos dissolvidos), pressão da bomba, temperatura e vazão/volume tratado (Murugan et al., 2023). Valores fora da faixa ideal podem comprometer o funcionamento do sistema, reduzir sua vida útil e elevar custos operacionais (Murugan et al., 2023; How2Electronics, 2023). Assim, mais do que a disponibilidade de água subterrânea, o fator crítico é a gestão adequada desses sistemas, assegurando sua manutenção preventiva e corretiva em tempo hábil.

Figura 2 – Sistema de dessalinizador

Figura 3 – Estação de Tratamento

Fonte: Queiroz (2026).

1.1 Aspectos Sócio-Econômicos

Dentro dessa realidade, a operação enfrenta dificuldades que vão desde o alto custo de deslocamento de equipes técnicas a campo até a falta de visibilidade em tempo real sobre o funcionamento dos equipamentos (Brasil. Ministério da Integração e do Desenvolvimento Regional, 2025). A ausência de dados confiáveis sobre falhas e consumo implica riscos de prejuízo financeiro, desperdício de recursos públicos e, sobretudo, ameaça ao abastecimento de comunidades vulneráveis. Surge, portanto, a necessidade de soluções tecnológicas que permitam monitorar remotamente os sistemas de dessalinização, garantindo maior eficiência operacional e redução de riscos.

Atualmente, o acompanhamento das estações é feito de forma predominantemente manual e local. Como ilustra a Figura 4, o controle operacional dos dessalinizadores depende de medidores analógicos (voltímetro, amperímetro, condutivímetro, manômetros e medidores de vazão), cujas leituras precisam ser observadas presencialmente pelo operador, sem registro digital ou transmissão automática. Não há, por parte do poder público, um monitoramento sistemático e em tempo real desses parâmetros; mesmo as análises de qualidade da água nem sempre ocorrem na periodicidade adequada (Queiroz, 2026). Esse cenário é agravado pela escala e pela dispersão geográfica do programa: são muitas

famílias atendidas, frequentemente em localidades de difícil acesso, o que torna onerosa e pouco eficiente uma manutenção baseada no deslocamento constante de equipes técnicas a campo.

Essa lacuna fica mais evidente quando se observa como o sistema é operado no dia a dia. Cada estação fica sob a responsabilidade de um operador local, eleito pelo grupo gestor da comunidade, a quem cabe ligar e desligar o equipamento, distribuir a água, realizar a limpeza periódica dos tanques e, sobretudo, avisar o conselho gestor quando o equipamento apresenta problemas e prestar contas da operação (Brasil. Ministério do Meio Ambiente, 2017b). Na prática, tanto a detecção de falhas quanto a prestação de contas dependem da observação atenta de uma única pessoa, sem nenhum apoio automatizado, o que reforça a necessidade de uma ferramenta que registre e comunique os dados de funcionamento de forma confiável. A Figura 5 mostra um chafariz comunitário, ponto onde a água tratada é distribuída e onde o operador acompanha boa parte da rotina.

Figura 4 – Painel de controle de uma estação de dessalinização do Programa Água Doce, com medidores analógicos de tensão, corrente, condutividade, pressão e vazão, lidos manualmente pelo operador.

Figura 5 – Chafariz comunitário de uma estação do Programa Água Doce, onde a água dessalinizada é distribuída à população.

Fonte: Queiroz (2026).

Este trabalho se insere nesse contexto, propondo o desenvolvimento de um sistema web com alertas inteligentes voltado para o monitoramento e controle das estações de dessalinização do Programa Água Doce. A iniciativa busca integrar dados operacionais críticos, prever defeitos, reduzir a necessidade de equipes em campo e contribuir para a gestão sustentável dos recursos hídricos no Semiárido brasileiro. Para viabilizar a coleta mesmo em locais com conectividade intermitente, a solução combina o sistema web a um aplicativo móvel que lê os sensores das estações via Bluetooth e sincroniza os dados com o servidor assim que há rede disponível.

1.2 Objetivo Geral

Aprimorar o sistema web existente do Programa Água Doce, integrando microcontroladores ESP32 e aprimorando os módulos de *back-end* (Node.js, Express, TypeORM e MySQL) e *front-end* (Angular, Angular Material, Chart.js e Leaflet), com suporte a Docker para implantação. O aprimoramento terá como foco a implementação de um sistema de alertas inteligentes, capaz de identificar anomalias operacionais em tempo real e notificar gestores e administradores, contribuindo para o monitoramento e controle remoto eficiente das estações de dessalinização, reduzindo custos de manutenção e garantindo maior disponibilidade de água potável no Semiárido brasileiro.

1.3 Objetivos Específicos

- **Expandir a API RESTful**, garantindo armazenamento seguro e estruturado dos dados e incluindo suporte para a geração de alertas automáticos.
- **Aprimorar o controle de acesso e permissões**, assegurando diferentes níveis de uso (administradores, gestores e operadores) e acesso restrito a estações designadas.
- **Desenvolver *dashboards* aprimorados**, com gráficos em tempo real e visualização geográfica, destacando indicadores de alerta para gestores e administradores.
- **Implementar um sistema de alertas inteligentes**, configurando limiares de operação e regras de anomalia que gerem notificações visuais e relatórios de acompanhamento.
- **Permitir o gerenciamento avançado de estações e usuários**, possibilitando edição, atribuição de papéis e configuração personalizada de alertas.
- **Disponibilizar extração de relatórios consolidados**, incluindo registros de eventos, alertas e histórico operacional das estações, para auditoria e apoio à tomada de decisão.
- **Implementar testes automáticos** para garantir a qualidade do código, a confiabilidade das funcionalidades críticas e a detecção precoce de regressões.

2 Impactos Sociais

O desenvolvimento deste sistema web para monitoramento das estações de dessalinização do Programa Água Doce apresenta impactos sociais significativos para as comunidades do Semiárido brasileiro. A implementação de um sistema de alertas inteligentes contribui diretamente para a garantia de fornecimento contínuo de água potável, reduzindo o risco de interrupções no abastecimento que afetam milhares de famílias em situação de

vulnerabilidade social (Brasil. Ministério da Integração e do Desenvolvimento Regional, 2025).

Para além dos ganhos operacionais, relatos de moradores das comunidades atendidas evidenciam mudanças concretas na percepção e na relação com a água tratada. Em entrevistas realizadas no contexto do Programa Água Doce, observa-se que o acesso à água dessalinizada não apenas melhorou as condições de consumo, mas também ampliou a consciência das comunidades sobre a importância de uma água de qualidade (Queiroz, 2026):

> Esse Programa Água Doce ajudou muito mais pessoas do meio rural [...]. As pessoas daquela comunidade hoje têm uma conscientização grande, sabem da importância que tem aquela água para eles. Uma água realmente pura, limpa para o consumo humano, então mudou muito a consciência das pessoas hoje. (Entrevistado E18)

No mesmo sentido, um morador destaca o efeito do programa sobre a noção de qualidade da água e o cuidado com a saúde da família:

> Eu acredito que a mudança seja no conceito das pessoas. Se não em todas, em toda a comunidade, mas naquelas que têm condições de usar. Porque aquele que vai lá buscar, ele vai no intuito de que aquela água é de qualidade e que vai melhorar, com aquele uso, a saúde da sua família. (Morador M2)

Esses depoimentos reforçam que o impacto do sistema de monitoramento transcende a dimensão técnica: ao assegurar a continuidade e a confiabilidade do tratamento, a solução contribui diretamente para a saúde, a dignidade e a autonomia das comunidades do Semiárido. As Figuras 6 e 7 registram o cotidiano de coleta e transporte da água tratada nos pontos de distribuição do programa.

Figura 6 – Transporte de galões de água tratada coletada em um ponto de distribuição do Programa Água Doce.

Figura 7 – Moradores coletando água dessalinizada em ponto de distribuição comunitário, com transporte por motocicleta e tração animal.

Fonte: Queiroz (2026).

A redução dos custos operacionais e a otimização da manutenção preventiva permitem que os recursos públicos sejam utilizados de forma mais eficiente, possibilitando a expansão do programa para atender um número maior de comunidades (Saia, 2018). Além disso, o monitoramento em tempo real e a capacidade de resposta rápida a falhas evitam desperdícios de água tratada, um recurso precioso em regiões de escassez hídrica.

Para dimensionar esse alcance, a Tabela 1 apresenta as localidades atendidas pelo programa no estado de Alagoas, com o número de famílias beneficiadas e a produção semanal de água dessalinizada. Para preservar a identificação das comunidades, os povoados foram anonimizados. Apenas nesse recorte, mais de mil famílias são atendidas, com uma produção que ultrapassa 300 mil litros de água potável por semana, o que dá a medida do volume de dados operacionais que um sistema de monitoramento precisa acompanhar de forma contínua.

Tabela 1 – Povoados atendidos pelo Programa Água Doce em Alagoas, com as comunidades anonimizadas.

Município	**Povoado**	**Famílias beneficiadas**	**Produção (L/semana)**
Estrela de Alagoas	Povoado 1	250	50.000
Estrela de Alagoas	Povoado 2*	100	14.000
Igaci	Povoado 3	50	35.000
Igaci	Povoado 4	92	24.780
Palmeira dos Índios	Povoado 5	89	56.960
Palmeira dos Índios	Povoado 6	185	57.160
Palmeira dos Índios	Povoado 7	250	50.000
Palmeira dos Índios	Povoado 8	85	11.900
Santana do Ipanema	Povoado 9*	60	9.000
Total		**1.161**	**308.800**

* Unidades Demonstrativas.

Fonte: Adaptado de Brasil. Ministério do Meio Ambiente (2010).

Do ponto de vista da gestão pública, o sistema proporciona maior transparência e controle sobre o funcionamento das estações, facilitando auditorias e prestação de contas. A disponibilidade de dados históricos e relatórios consolidados permite análises que podem orientar políticas públicas mais efetivas para o setor de recursos hídricos no Semiárido.

Essa transparência dialoga diretamente com o modelo de gestão do programa. Em cada comunidade, as regras de uso e operação são definidas em um Acordo de Gestão, assinado pelas famílias beneficiadas, que estabelece direitos, deveres, a forma de prestação de contas e, inclusive, como a própria comunidade deve acompanhar o cumprimento do acordo (Brasil. Ministério do Meio Ambiente, 2017c). Ao registrar de modo confiável o funcionamento das estações e ao disponibilizar uma página pública de consulta, o sistema oferece exatamente o tipo de informação que esse acompanhamento coletivo pressupõe, fortalecendo a governança participativa prevista pelo programa.

Há ainda um impacto positivo na geração de renda e no aproveitamento ambiental dos

rejeitos. O concentrado salino resultante da dessalinização, em vez de ser simplesmente descartado, pode ser reaproveitado em sistemas produtivos, sobretudo na criação de peixes, como a tilápia rosa, e no cultivo de forragem para os animais (Figura 8) (Brasil. Ministério do Meio Ambiente, 2017a). Além de complementar a renda das famílias, esse reúso reduz o impacto ambiental do descarte e fecha o ciclo de uso da água nas comunidades. O monitoramento contínuo de variáveis como salinidade, temperatura e oxigênio é justamente o que torna essas atividades viáveis e seguras, o que reforça o valor de uma solução tecnológica integrada de acompanhamento.

Figura 8 – Tanques de contenção do concentrado salino utilizados na criação de peixes, exemplo de reúso produtivo do rejeito da dessalinização.

Fonte: Queiroz (2026).

Por fim, ao minimizar a necessidade de deslocamentos frequentes de equipes técnicas, o sistema contribui para a sustentabilidade ambiental do programa, reduzindo emissões de carbono e custos com transporte, ao mesmo tempo em que garante maior agilidade na identificação e correção de problemas operacionais. Dessa forma, os ganhos ambientais somam-se aos benefícios sociais e econômicos já descritos, reforçando o caráter sustentável e integrado da solução para a realidade do Semiárido.

3 Referencial Teórico

O Programa Água Doce (PAD) é uma política pública criada pelo Governo Federal para garantir o acesso sustentável à água potável em comunidades do Semiárido brasileiro,

utilizando sistemas de dessalinização de águas subterrâneas salobras. O processo de dessalinização é realizado, em grande parte, por meio da osmose reversa, que utiliza membranas semipermeáveis para remover sais dissolvidos e tornar a água própria para o consumo humano. A efetividade desse processo depende do controle de variáveis como condutividade elétrica, pressão, temperatura e vazão, fundamentais para garantir o funcionamento adequado dos equipamentos e evitar falhas (Brasil. Ministério do Meio Ambiente, 2012).

A própria concepção do programa já reconhece a importância de acompanhar esses parâmetros. A dessalinização separa a água em duas frações: a água potável, encaminhada ao chafariz, e o concentrado salino, rejeito do processo. Nas Unidades Demonstrativas, esse concentrado é reaproveitado em sistemas produtivos (como tanques para a criação de peixes), atividade que exige o monitoramento constante de temperatura, pH, salinidade e oxigênio (Brasil. Ministério do Meio Ambiente, 2017a). Quando descartado de forma inadequada, porém, o concentrado degrada o solo e, carregado pelas chuvas, pode atingir açudes e aquíferos, tornando a água mais salina e comprometendo o próprio sistema (Brasil. Ministério do Meio Ambiente, 2017a). Acompanhar grandezas como a condutividade e a salinidade, portanto, não interessa apenas à manutenção dos equipamentos, mas também à sustentabilidade ambiental da operação. A Figura 9 mostra os reservatórios que separam a água bruta, o concentrado e a água doce em uma estação do programa.

Figura 9 – Reservatórios de uma estação de dessalinização do Programa Água Doce: água bruta, concentrado e água doce.

Fonte: Queiroz (2026).

3.1 Internet das Coisas e Microcontroladores ESP32

As limitações de infraestrutura e conectividade em comunidades rurais tornam o monitoramento manual das estações caro, sujeito a erros e ineficiente. Nesse contexto, a aplicação da Internet das Coisas (IoT) oferece uma solução tecnológica viável e de baixo custo (Atzori et al., 2010; How2Electronics, 2023). O ESP32, microcontrolador amplamente utilizado em aplicações de IoT, destaca-se pela integração de conectividade Wi-Fi/Bluetooth e capacidade de processamento, possibilitando a coleta, transmissão e pré-processamento de dados em ambientes com acesso restrito à internet (Mitrović et al., 2023; Suárez et al., 2023; IoT Project Ideas, 2025).

Figura 10 – ESP32

Fonte: Elaborado pelo autor.

A arquitetura de comunicação em sistemas IoT pode ser implementada através de diversos protocolos, sendo o MQTT (Message Queuing Telemetry Transport) um dos mais utilizados devido à sua leveza e eficiência em redes com largura de banda limitada (Kodali and Soratkal, 2016). Além disso, o HTTP/HTTPS permanece como alternativa robusta para transmissão de dados em aplicações que priorizam a compatibilidade com infraestruturas web existentes (Jan et al., 2021).

A segurança em sistemas IoT representa uma preocupação crescente, especialmente em aplicações críticas como o monitoramento de recursos hídricos. A implementação de mecanismos de autenticação, criptografia de dados e controle de acesso é fundamental para proteger a integridade das informações transmitidas e prevenir ataques cibernéticos (Sicari et al., 2015).

3.2 Sistemas Web e Arquiteturas Modernas

O acompanhamento de processos industriais é tradicionalmente realizado por meio de ferramentas SCADA (Supervisory Control and Data Acquisition), reconhecidas pela robustez no controle e na supervisão de plantas industriais, mas também pelo alto custo de aquisição, implementação e manutenção (Raghunandan, 2022; Kvist, 2023). Esse custo elevado torna tais plataformas inviáveis para muitas aplicações de monitoramento ambiental de menor escala e motiva a adoção de plataformas IoT customizadas e de baixo custo. No caso do PAD, essa abordagem combina microcontroladores ESP32 com bibliotecas Arduino para a coleta em campo, um aplicativo móvel para o monitoramento e a sincronização das leituras, e uma API RESTful que alimenta *dashboards* web interativos, consolidando as informações dos sensores em gráficos e mapas dinâmicos. Essa combinação mostra-se mais adequada a contextos descentralizados e de recursos limitados, como o do PAD.

A arquitetura REST (Representational State Transfer), proposta por Fielding (Fielding, 2000), estabelece princípios fundamentais para o desenvolvimento de serviços web escaláveis e desacoplados. A adoção de APIs RESTful facilita a integração entre diferentes componentes do sistema e permite a expansão futura com novos módulos e funcionalidades.

Node.js emergiu como plataforma robusta para o desenvolvimento de aplicações web de alto desempenho, especialmente adequada para sistemas que demandam processamento assíncrono e comunicação em tempo real (Tilkov and Vinoski, 2010). Sua arquitetura baseada em eventos e modelo não-bloqueante permite o gerenciamento eficiente de múltiplas conexões simultâneas, característica essencial para sistemas IoT com dezenas ou centenas de dispositivos conectados.

A utilização de ORMs (Object-Relational Mapping), como o TypeORM, simplifica a interação com bancos de dados relacionais, abstraindo a complexidade das consultas SQL e proporcionando maior produtividade no desenvolvimento (Ambler, 2003). *Frameworks front-end* modernos, como Angular, oferecem estruturas robustas para o desenvolvimento de interfaces interativas e responsivas, facilitando a criação de *dashboards* complexos com visualização de dados em tempo real (Fain and Moiseev, 2018).

3.3 Sistemas de Alertas Inteligentes

Além da coleta e visualização de dados, um sistema de monitoramento hídrico precisa ser capaz de interpretar informações e acionar alertas preventivos. Pesquisas recentes apontam que sistemas IoT com alertas automáticos, registro histórico de dados e monitoramento remoto aumentam a eficiência operacional e permitem respostas mais rápidas a falhas ou condições adversas (Forhad et al., 2024). De forma complementar, estudos mostram que a detecção de alterações em parâmetros críticos, como o pH da água, seguida do envio de alertas imediatos, é fundamental para reduzir riscos e preservar a qualidade da água

distribuída (Miller, 2023).

A detecção de anomalias em sistemas de monitoramento pode ser implementada através de diversas técnicas de aprendizado de máquina. O algoritmo Local Outlier Factor (LOF) destaca-se pela capacidade de identificar padrões anômalos em conjuntos de dados multidimensionais, comparando a densidade local de cada ponto com a densidade de seus vizinhos (Breunig et al., 2000). Algoritmos baseados em árvores de decisão, como o Random Forest, também demonstram eficácia na classificação de eventos normais e anômalos em séries temporais (Chandola et al., 2009).

Em revisões sistemáticas, técnicas de detecção de anomalias, como Local Outlier Factor e Random Forest, têm se destacado por sua capacidade de identificar desvios operacionais e disparar alertas inteligentes, tornando os sistemas mais resilientes e confiáveis (Zulkifli et al., 2022). A implementação de limiares adaptativos, que se ajustam automaticamente com base no histórico de dados, pode reduzir significativamente o número de falsos positivos em sistemas de alerta (Hayes and Capretz, 2015). Nesse sentido, a convergência entre dessalinização, IoT, sistemas web e mecanismos de alerta inteligentes apresenta-se como um caminho promissor para ampliar a eficiência operacional, reduzir custos de manutenção e garantir a sustentabilidade do Programa Água Doce, assegurando o acesso contínuo à água potável em comunidades vulneráveis do Semiárido brasileiro.

4 Metodologia

O desenvolvimento deste projeto adotou princípios de metodologias ágeis, especialmente inspirados no *framework* Scrum (Schwaber and Sutherland, 2020), devido à sua eficácia em projetos complexos que exigem flexibilidade e entregas incrementais. Essa escolha metodológica justifica-se pela natureza dinâmica do projeto e pela necessidade de adaptação contínua aos requisitos identificados ao longo do desenvolvimento, alinhando-se aos valores propostos pelo Manifesto Ágil (Beck et al., 2001).

O trabalho foi organizado em ciclos curtos de desenvolvimento, permitindo entregas frequentes e avaliações constantes do progresso. Reuniões semanais foram realizadas para revisar o andamento das atividades, identificar obstáculos e ajustar prioridades. Essa abordagem colaborativa garantiu que o foco permanecesse nas funcionalidades de maior impacto para o objetivo final do sistema.

As funcionalidades foram priorizadas de acordo com seu valor para os usuários finais e para o cumprimento dos objetivos do Programa Água Doce. Inicialmente, concentrou-se no desenvolvimento da infraestrutura base do sistema, incluindo a expansão da API e a configuração dos mecanismos de autenticação. Em seguida, foram implementados os *dashboards* de visualização e o sistema de alertas inteligentes, componentes essenciais para o monitoramento operacional das estações.

Durante todo o processo, houve preocupação constante com a qualidade das entregas.

Testes foram realizados ao final de cada ciclo de desenvolvimento para validar as funcionalidades implementadas e garantir que atendessem aos requisitos estabelecidos (Pressman and Maxim, 2014). Problemas identificados foram corrigidos imediatamente, evitando o acúmulo de pendências técnicas.

A metodologia adotada também facilitou a comunicação entre os membros da equipe e os orientadores do projeto. Feedback contínuo foi incorporado ao planejamento, permitindo ajustes nas funcionalidades com base nas necessidades reais dos gestores e operadores das estações de dessalinização. Essa flexibilidade foi fundamental para o sucesso do projeto, possibilitando respostas rápidas a mudanças nos requisitos e prioridades.

4.1 Requisitos Funcionais e Não-Funcionais

A definição dos requisitos partiu da análise do sistema web existente do Programa Água Doce e das necessidades de quem opera as estações em campo. Para descrever esses requisitos de forma clara e centrada no usuário, adotamos histórias de usuário organizadas em épicos (Raharjana et al., 2021). Cada história segue o formato "como <papel>, quero <objetivo>, para <benefício>", o que mantém o foco no valor entregue a cada perfil (Sommerville, 2011). Foram considerados cinco papéis: **administrador** (configura o sistema), **gestor** (acompanha as estações sob sua responsabilidade), **operador** (coleta e envia leituras), **dispositivo/aplicativo** (ESP32 e app móvel que enviam dados automaticamente) e **visitante** (público externo, sem autenticação).

Épico 1: Autenticação e controle de acesso

- Como usuário, quero autenticar com usuário e senha, para acessar as funcionalidades de acordo com o meu papel.
- Como usuário autenticado, quero encerrar a sessão com segurança, para proteger meu acesso.
- Como administrador, quero que cada rota respeite o papel do usuário, para garantir que cada um só faça o que lhe é permitido.
- Como gestor ou operador, quero enxergar apenas as estações sob minha responsabilidade, para focar no que me cabe e preservar a confidencialidade.

Épico 2: Gestão de usuários

- Como administrador, quero cadastrar usuários definindo seus papéis, para controlar quem acessa o sistema e com qual permissão.
- Como administrador, quero editar nome, senha e papéis de um usuário, para manter dados e permissões atualizados.

- Como administrador, quero listar, criar, editar e remover usuários pela própria interface, para gerenciar a equipe sem recorrer à API.
- Como administrador, quero vincular usuários a estações como gestores ou operadores, para definir responsabilidades.

Épico 3: Cadastro e configuração de estações e sensores

- Como administrador, quero cadastrar estações com identificador e localização, para organizar os pontos de monitoramento.
- Como administrador, quero editar uma estação e seus sensores e responsáveis, para manter a configuração correta.
- Como administrador, quero arquivar uma estação sem apagar seus dados históricos, para preservar a integridade e poder auditar ou recuperar depois.
- Como administrador, quero cadastrar e configurar sensores com unidade, modelo, precisão e limites mínimo e máximo, para permitir leituras consistentes e alertas.

Épico 4: Coleta e envio de leituras

- Como dispositivo IoT ou aplicativo, quero enviar lotes de leituras autenticando por chave de API, para registrar dados sem depender de *login* de usuário.
- Como aplicativo, quero que estações e sensores ausentes sejam criados automaticamente ao enviar leituras, para não depender de cadastro prévio em campo.
- Como operador em campo, quero coletar leituras via Bluetooth e armazená-las localmente para sincronizar depois, para trabalhar mesmo sem internet no momento da coleta.

Épico 5: Visualização e análise

- Como gestor ou administrador, quero uma página inicial com o panorama do sistema, para entender rapidamente o estado geral.
- Como gestor, quero abrir o *dashboard* de uma estação e ver apenas os dados dela, para monitorá-la individualmente.
- Como gestor, quero filtrar as leituras por intervalo de datas e por sensor, para analisar períodos e grandezas específicas.
- Como gestor ou administrador, quero que o sistema identifique automaticamente as leituras anômalas no momento em que chegam, para ser avisado de problemas sem inspecionar os dados manualmente.

- Como gestor, quero que leituras fora dos limites do sensor sejam destacadas, para identificar problemas rapidamente.
- Como gestor, quero exportar as leituras de uma estação em CSV, para gerar relatórios e analisar os dados em outras ferramentas.

Épico 6: Transparência pública

- Como visitante, quero ver as estações da rede sem precisar de *login*, para acompanhar o monitoramento de forma transparente.

Épico 7: Integração do aplicativo móvel com o sistema

O aplicativo móvel responsável pela coleta em campo (leitura dos sensores via Bluetooth, armazenamento local e visualização no dispositivo) foi desenvolvido em trabalho anterior. A contribuição deste trabalho concentrou-se em **integrar esse aplicativo ao restante do sistema**, conectando-o ao *back-end* e ao fluxo de dados consumido pelo *front-end*, conforme as histórias a seguir.

- Como operador, quero que as leituras coletadas pelo aplicativo sejam enviadas ao servidor, autenticando por chave de API, para centralizar os dados no sistema e disponibilizá-los aos *dashboards*.
- Como operador ou desenvolvedor, quero alternar o aplicativo entre o servidor local e o de produção, para testar e operar a integração sem alterar o código.
- Como desenvolvedor, quero gerar leituras de teste no aplicativo, para validar o envio e a exibição dos dados sem depender de um dispositivo físico.

Épico 8: Operação e observabilidade

- Como administrador, quero consultar e limpar os erros registrados pela API, para acompanhar a saúde do sistema e diagnosticar problemas.
- Como desenvolvedor integrador, quero uma documentação navegável das rotas da API, para integrar com o sistema sem precisar ler o código-fonte.

Além dos requisitos funcionais, foram definidos requisitos não-funcionais que orientaram decisões de arquitetura e qualidade, resumidos na Tabela 2.

Tabela 2 – Requisitos não-funcionais do sistema

Categoria	**Descrição**
Segurança	Autenticação por JWT em cookie *httpOnly* e senhas com hash bcrypt; autorização por papel e por vínculo usuário–estação; dispositivos e aplicativo autenticam por chave de API distinta do *login* de usuário; princípio do menor privilégio no auto-registro.
Confiabilidade	Remoção de estações por *soft delete*, preservando o histórico; registro estruturado de erros da API para acompanhamento operacional.
Testabilidade	Suíte de testes automatizados no *back-end* e no *front-end*, executada a cada ciclo, com banco de testes isolado em memória.
Desempenho	Consultas com filtros e *endpoint* de leituras agregadas (média, mínimo e máximo por hora ou dia) para séries longas.
Escalabilidade	Arquitetura REST desacoplada e modular, permitindo adicionar estações, sensores e módulos sem reescrever o núcleo (Chung et al., 2012).
Usabilidade	Interfaces responsivas com Angular Material, indicadores visuais de alerta e gráficos interativos.
Portabilidade	Containerização com Docker, garantindo execução consistente entre ambientes de desenvolvimento e produção.

Fonte: Elaborado pelo autor.

4.2 Principais Funcionalidades

As funcionalidades descritas a seguir foram efetivamente implementadas e cobrem o ciclo completo do monitoramento, da coleta em campo à visualização e à gestão.

Coleta e sincronização. A coleta em campo é realizada por um aplicativo móvel em Flutter, desenvolvido em trabalho anterior, que se comunica com o ESP32 via Bluetooth e armazena as leituras localmente. A contribuição deste trabalho foi a integração desse aplicativo ao sistema: uma rotina de sincronização passou a enviar as leituras pendentes ao servidor por meio de uma rota autenticada por chave de API. Estações e sensores ainda não cadastrados são criados automaticamente a partir do primeiro envio, o que evita a dependência de cadastro prévio em campo.

Visão geral do sistema. A página inicial apresenta um panorama com indicadores de estações, sensores, leituras e alertas ativos, um painel com os alertas de todas as estações e a lista de estações com acesso direto ao *dashboard* de cada uma.

Dashboard por estação. Cada estação possui um painel com suas últimas leituras, seus alertas, sua localização em mapa (Agafonkin, 2015) e um gráfico (Nelli, 2018) que permite filtrar as leituras por sensor e por período. Para séries longas, o servidor disponibiliza leituras agregadas por hora ou por dia.

Alertas. Cada sensor possui limites mínimo e máximo configuráveis. Na ingestão das leituras, valores fora desses limites são marcados e passam a aparecer destacados na interface e nos painéis de alerta, permitindo identificar anomalias rapidamente. Sensores sem limite definido não geram alertas falsos.

Gestão de estações e usuários. Administradores cadastram e editam estações

(nome, coordenadas, sensores e responsáveis) e gerenciam usuários pela própria interface, inclusive o vínculo de gestores e operadores a estações. A remoção de uma estação é feita por *soft delete*: a estação deixa de aparecer nas listagens, mas seus dados históricos são preservados no banco.

Relatórios e transparência. As leituras de uma estação podem ser exportadas em CSV, com filtros de período e sensor, para análise em outras ferramentas. Há ainda uma página pública, acessível sem *login*, que exibe as estações da rede em um mapa, atendendo à transparência e à prestação de contas do programa.

O acesso a essas funcionalidades respeita o papel do usuário: administradores têm visão global e poder de configuração; gestores e operadores enxergam apenas as estações sob sua responsabilidade. Os principais parâmetros acompanhados são condutividade elétrica, pressão, temperatura e vazão/volume.

4.3 Rotas da API

A API RESTful do sistema disponibiliza *endpoints* organizados em módulos: autenticação, gerenciamento de usuários, sensores, estações (incluindo leituras, alertas, exportação e acesso público) e operação. A Tabela 3 apresenta as principais rotas implementadas, seus métodos HTTP correspondentes e suas funcionalidades.

Tabela 3 – Rotas da API RESTful do sistema de monitoramento

Método	Rota	Descrição
Autenticação e Autorização		
POST	`/auth/login`	Autenticar e gerar *token* JWT
POST	`/auth/register`	Auto-registro (cria apenas OPERATOR)
POST	`/auth/logout`	Encerrar sessão
GET	`/auth/me`	Obter dados do usuário autenticado
Gerenciamento de Usuários		
GET	`/users`	Listar usuários
POST	`/users`	Criar usuário com papel (ADMIN)
PUT	`/users/{id}`	Editar usuário (ADMIN)
DELETE	`/users/{id}`	Remover usuário (ADMIN)
GET	`/users/{id}/stations`	Listar estações vinculadas
PUT	`/users/{id}/stations`	Atualizar vínculos usuário–estação
Gerenciamento de Sensores		
POST	`/sensors`	Cadastrar sensor
GET	`/sensors/{identifier}`	Obter detalhes de um sensor
PUT	`/sensors/{identifier}`	Atualizar configurações e limites
DELETE	`/sensors/{identifier}`	Remover sensor
Estações		
GET	`/stations`	Listar estações (conforme o papel)
POST	`/stations`	Cadastrar estação
GET	`/stations/summary`	Indicadores gerais (visão geral)
GET	`/stations/public`	Listar estações (acesso público)
GET	`/stations/public/{identifier}`	Detalhe público de uma estação
GET	`/stations/export/csv`	Exportar leituras em CSV
GET	`/stations/{identifier}`	Obter detalhes de uma estação
PUT	`/stations/{id}`	Atualizar estação e vínculos
DELETE	`/stations/{id}`	Arquivar estação (*soft delete*)
GET	`/stations/{identifier}/readings`	Leituras com filtros de período/sensor
GET	`/stations/{identifier}/readings/aggregated`	Leituras agregadas (hora/dia)
GET	`/stations/{identifier}/alerts`	Alertas da estação
POST	`/stations/{identifier}/device-readings`	Envio de leituras pelo dispositivo (chave de API)
Operação e Logs		
GET	`/logs`	Listar *logs* de erro (ADMIN)
DELETE	`/logs`	Limpar *logs* (ADMIN)

Fonte: Elaborado pelo autor.

A maior parte das rotas exige um *token* JWT válido. As exceções são as rotas de autenticação (`/auth/login` e `/auth/register`) e as rotas públicas de transparência (`/stations/public...`). O envio de leituras pelo dispositivo (`device-readings`) usa um mecanismo de autenticação próprio, baseado em chave de API, independente do *login* de usuário.

4.3.1 Cadastro Estações

Nesta tela, o administrador cadastra novas estações informando nome, identificador único e localização geográfica (latitude e longitude). Após a validação, a estação passa a integrar a rede de monitoramento e fica disponível para o vínculo de sensores e de responsáveis.

Figura 11 – Tela de Cadastro de Estações

Fonte: Elaborado pelo autor.

4.3.2 Cadastro Sensores

O cadastro de sensores associa cada sensor a uma estação, definindo a grandeza medida, a unidade, o modelo e os limites operacionais (mínimo e máximo). Esses limites são utilizados posteriormente para classificar as leituras e disparar alertas quando os valores ficam fora da faixa esperada.

Figura 12 – Tela de Cadastro de Sensores

Fonte: Elaborado pelo autor.

4.3.3 Cadastro Usuário

A tela de cadastro de usuários permite criar contas e atribuir o papel correspondente: administrador, gestor ou operador, definindo o nível de acesso de cada um às funcionalidades

do sistema.

Figura 13 – Cadastro Usuário

Fonte: Elaborado pelo autor.

4.3.4 Dashboards e Alertas

O painel de controle consolida o estado da rede e centraliza os alertas gerados automaticamente a partir das leituras anômalas, destacando as estações e os sensores que requerem atenção e apoiando a resposta rápida a situações críticas.

Figura 14 – Dashboard de Controle e Sistema de Alertas

Fonte: Elaborado pelo autor.

4.3.5 Visão Geral do Sistema

A página inicial reúne os principais indicadores da rede, o painel de alertas e a lista de estações, oferecendo uma visão consolidada do sistema (Figura 15). O menu lateral (Figura 16) dá acesso às demais áreas da aplicação.

Figura 15 – Página inicial com a visão geral do sistema: indicadores, painel de alertas e lista de estações.

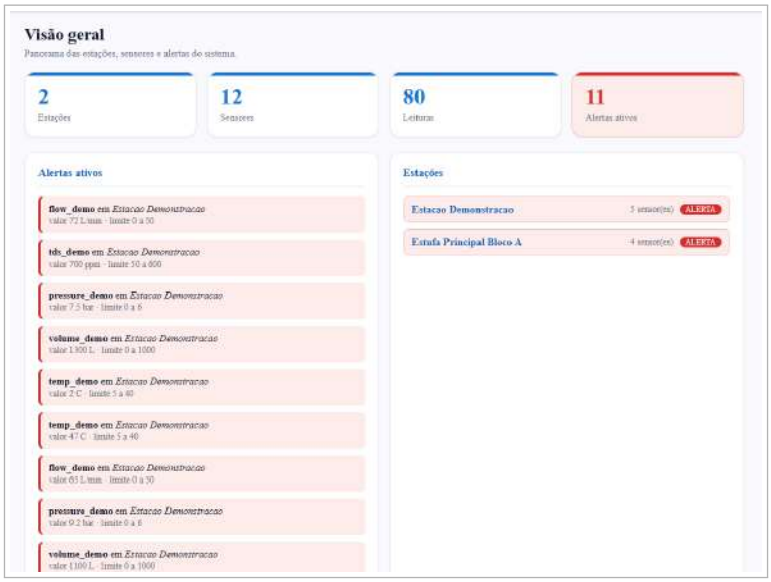


Figura 16 – Menu lateral de navegação do sistema.

Fonte: Elaborado pelo autor.

4.3.6 Dashboard por Estação

O *dashboard* de cada estação detalha suas últimas leituras e alertas, com gráficos filtráveis por sensor e por período e a localização da estação no mapa, apoiando a análise da evolução dos dados ao longo do tempo.

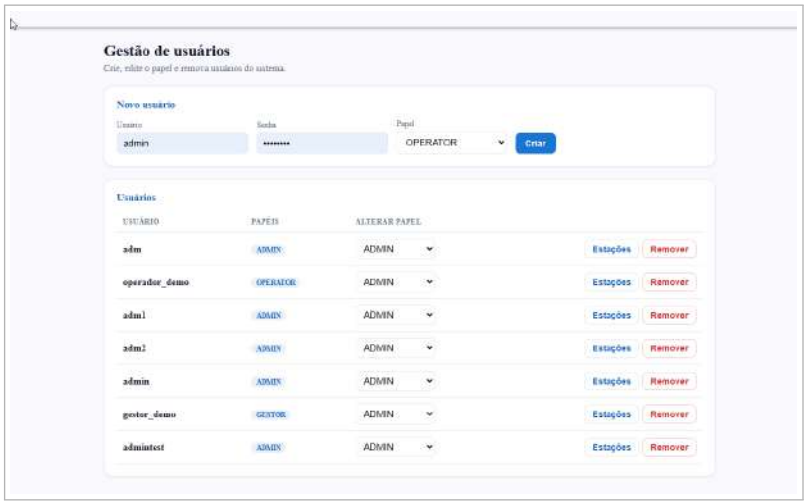
Figura 17 – Dashboard de uma estação, com últimas leituras, alertas, gráfico filtrável por sensor/período e localização no mapa.

Fonte: Elaborado pelo autor.

4.3.7 Gestão de Usuários

Esta tela permite a gestão completa de usuários: criação de contas, edição de papéis e vínculo de gestores e operadores às estações sob sua responsabilidade, controlando quais dados cada usuário pode visualizar ou operar.

Figura 18 – Tela de gestão de usuários, com criação, edição de papéis e vínculo de gestores e operadores a estações.

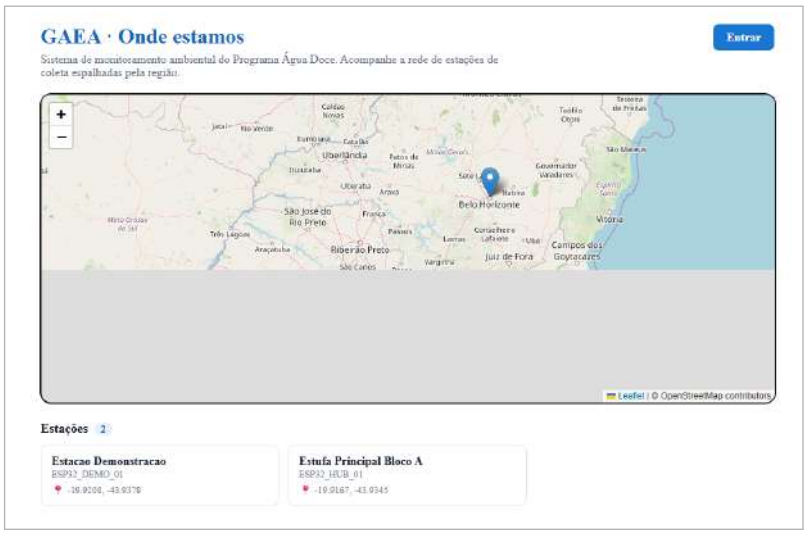


Fonte: Elaborado pelo autor.

4.3.8 Página Pública de Transparência

A página pública é acessível sem autenticação e apresenta o mapa das estações da rede, atendendo ao requisito de transparência ao disponibilizar informações ao público geral.

Figura 19 – Página pública, acessível sem autenticação, com o mapa das estações da rede.



Fonte: Elaborado pelo autor.

4.3.9 Edição de Estação

O formulário de edição reúne, em uma única tela, os dados cadastrais da estação, seus sensores e os responsáveis vinculados, permitindo atualizá-los de forma centralizada. As alterações respeitam as permissões por papel, de modo que apenas perfis autorizados podem efetivá-las.

Figura 20 – Formulário de edição de uma estação, com seus sensores e responsáveis.

Fonte: Elaborado pelo autor.

4.3.10 Aplicativo Móvel

O aplicativo móvel é utilizado na coleta de dados em campo. A partir de sua tela de sincronização (Figura 22), as leituras coletadas são enviadas ao servidor e passam a ficar disponíveis na *dashboard* web, integração validada neste trabalho.

Figura 21 – Tela inicial do aplicativo de coleta utilizado em campo.

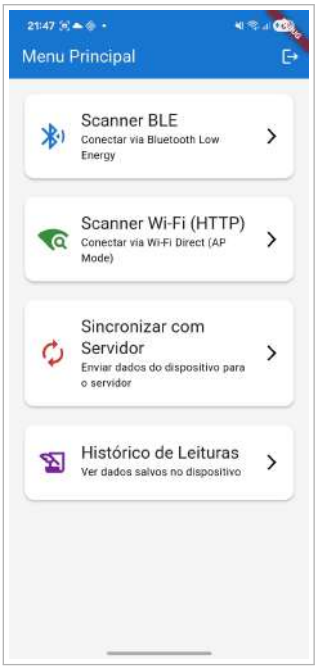


Figura 22 – Tela de sincronização do aplicativo, que envia as leituras coletadas ao servidor (parte integrada neste trabalho).

Fonte: Elaborado pelo autor.

4.4 Tecnologias Utilizadas

A arquitetura do sistema foi desenvolvida utilizando tecnologias modernas e consolidadas no mercado. No *back-end*, foi utilizado Node.js com o *framework* Express para construção da API RESTful. O TypeORM foi escolhido como ORM (Object-Relational Mapping) para facilitar a interação com o banco de dados MySQL, que armazena todos os dados operacionais, registros de alertas e informações de usuários (Ramakrishnan and Gehrke, 2003).

Figura 23 – Diagrama do Banco de Dados

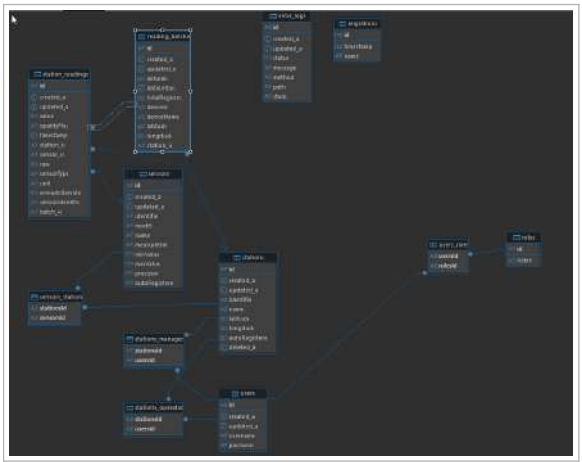
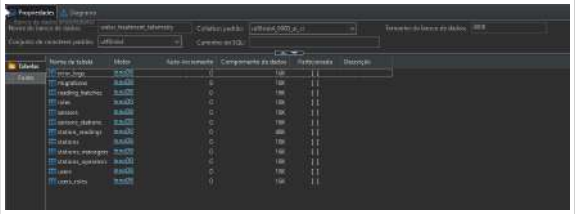


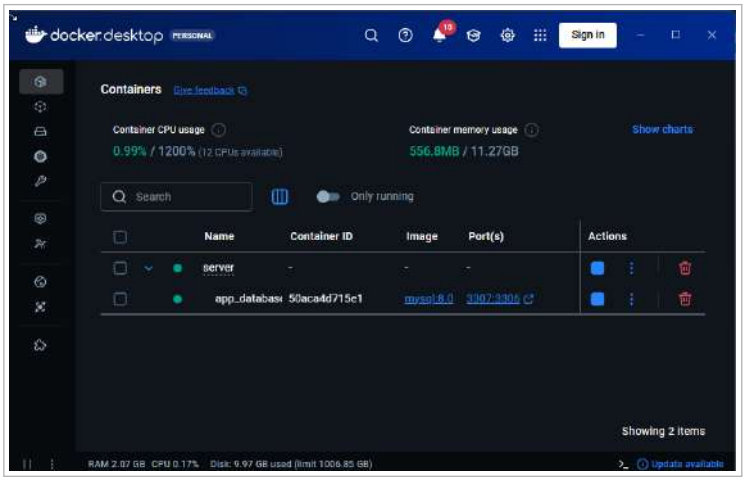
Figura 24 – Tabelas do Banco de Dados



Fonte: Elaborado pelo autor.

Para containerização e facilitar a implantação, foi adotado o Docker (Merkel, 2014), permitindo que o sistema seja executado de forma consistente em diferentes ambientes. Os microcontroladores ESP32 são responsáveis pela coleta de dados dos sensores nas estações de dessalinização, transmitindo as informações para o servidor através de conexões Wi-Fi.

Figura 25 – Container da aplicação em Docker.



Fonte: Elaborado pelo autor.

No *front-end*, além do Angular, Angular Material, Chart.js e Leaflet já mencionados,

foram utilizadas bibliotecas complementares para garantir responsividade e compatibilidade com diferentes dispositivos. A comunicação entre *front-end* e *back-end* ocorre exclusivamente através da API RESTful, seguindo padrões REST e garantindo segurança através de *tokens* de autenticação baseados em JWT (JSON Web *Tokens*)(Jones et al., 2015).

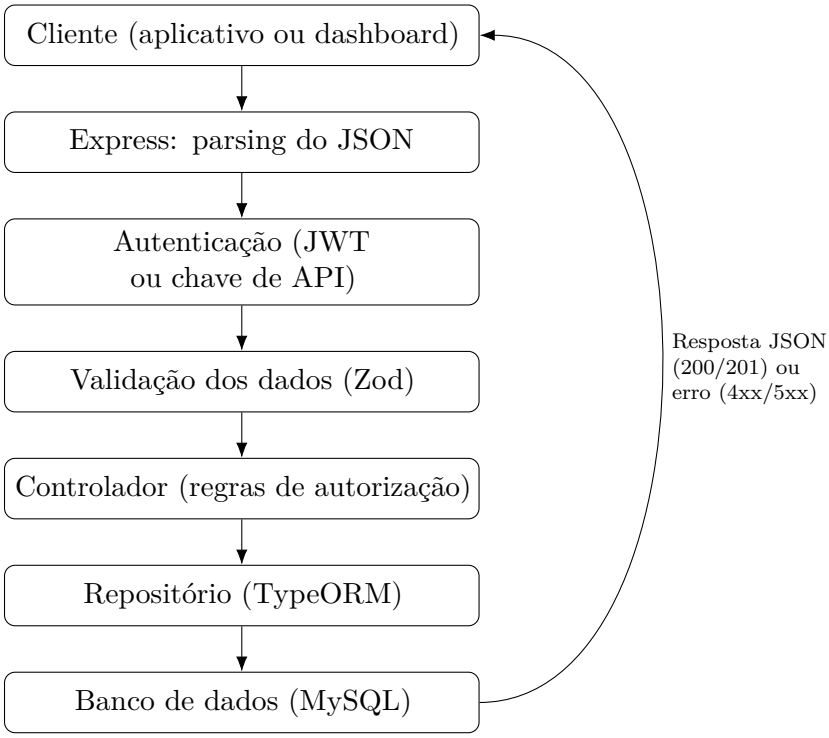
4.5 Arquitetura em Camadas

O *back-end* foi organizado segundo uma arquitetura em camadas, com responsabilidades bem separadas. As rotas (`routes/`) associam cada *endpoint* ao seu controlador; os controladores (`controllers/`) interpretam a requisição e orquestram a resposta; os serviços (`services/`) concentram regras de negócio reutilizáveis, como o `AuthService`, responsável pela autenticação, e o `AccessControlService`, que centraliza a verificação de papéis e do vínculo entre usuário e estação; os repositórios (`repositories/`) encapsulam todo o acesso ao banco por meio do TypeORM; e as entidades (`entities/`) representam as tabelas e seus relacionamentos. Essa separação favorece a manutenção, o reúso de componentes e a testabilidade do código.

Entre as camadas atuam *middlewares* reutilizáveis: o `AuthMiddleware` valida o *token* JWT; o `ApiKeyMiddleware` autentica por chave de API, mecanismo usado na integração com o aplicativo de coleta; o `RequireRoleMiddleware` controla o acesso por papel; o `ValidationMiddleware` verifica os dados de entrada e saída com a biblioteca Zod; e o `ErrorMiddleware` faz o tratamento centralizado de erros, registrando as falhas de forma estruturada. Os mesmos *schemas* Zod usados na validação alimentam a geração automática da documentação OpenAPI, mantendo código e documentação sincronizados.

O fluxo de uma requisição percorre essas camadas de forma previsível (Figura 26). Tomando como exemplo o envio das leituras coletadas em campo (`POST /stations/:identifier/device-readings`): o corpo da requisição é convertido em objeto, autenticado pela chave de API, validado pelos *schemas* Zod e encaminhado ao controlador, que aplica as regras de autorização e aciona o repositório; este persiste os dados no MySQL por meio do TypeORM, e o controlador devolve uma resposta em JSON com o código HTTP adequado. Eventuais erros são capturados de forma centralizada e registrados para acompanhamento operacional.

Figura 26 – Fluxo de uma requisição pelas camadas do servidor (autenticação, validação, controlador, repositório e banco).



Fonte: Elaborado pelo autor.

4.6 Modelo de Dados

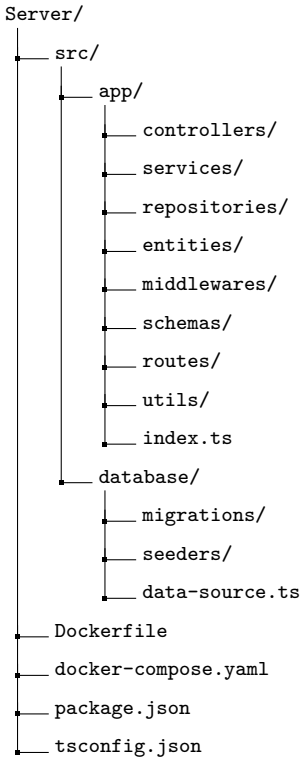
O armazenamento foi modelado em um banco relacional MySQL, acessado pelo TypeORM. As principais entidades são `Station` (estação), `Sensor`, `StationReading` (leitura individual) e `ReadingBatch` (lote que agrupa as leituras de uma mesma sincronização), além de `User` e `Role` para usuários e papéis, e `ErrorLog` para o registro de erros da API. As estações relacionam-se com sensores e com seus responsáveis (gestores e operadores), e cada leitura é vinculada à sua estação, ao sensor e ao instante de coleta. As Figuras 23 e 24 apresentam, respectivamente, o modelo entidade–relacionamento e as tabelas resultantes.

A evolução do esquema foi controlada por migrações versionadas, que registram cada alteração aplicada ao banco: entre elas, a criação das tabelas de leituras e lotes, a carga inicial de papéis e do usuário administrador, a inclusão de indicadores de auto-cadastro de estações e sensores, a criação da tabela de *logs* e a adição do campo de exclusão lógica. A remoção de estações é feita por *soft delete*: em vez de apagar o registro, marca-se a data de exclusão (`deleted_at`), de modo que a estação deixa de aparecer nas consultas, mas seus dados históricos permanecem preservados para auditoria e eventual recuperação.

4.7 Organização do Projeto

A solução é composta por três repositórios independentes: o servidor (API REST), a aplicação web e o aplicativo móvel, sendo que a participação do aplicativo neste trabalho restringe-se à sua integração com o servidor. No servidor, o código-fonte está organizado em `src/app/`, com pastas dedicadas a controladores, serviços, repositórios, entidades, *schemas*, *middlewares*, rotas e utilitários, e em `src/database/`, que reúne a configuração da conexão, as migrações e os *seeders* de dados iniciais (Figura 27).

Figura 27 – Organização das pastas do servidor (camadas e diretório de banco de dados).

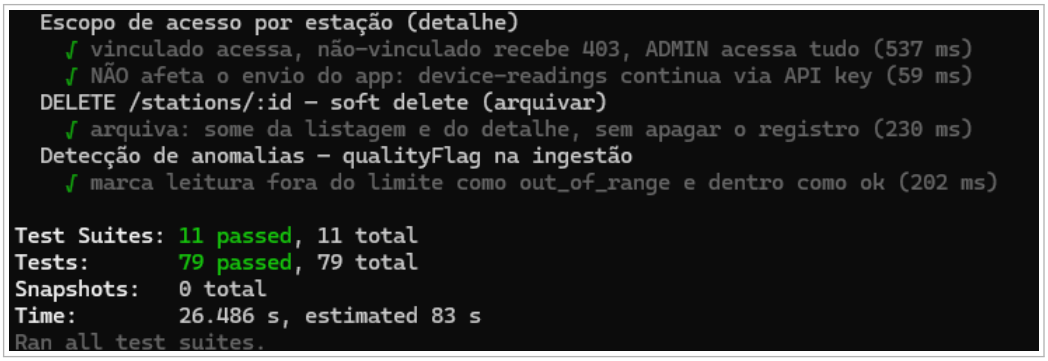


Fonte: Elaborado pelo autor.

Para a implantação, a aplicação é containerizada com Docker, garantindo execução consistente entre os ambientes de desenvolvimento e de produção (Figura 25); as configurações sensíveis são parametrizadas por variáveis de ambiente. A qualidade do código é apoiada pelo uso de TypeScript, pela verificação estática com ESLint e por uma suíte de testes automatizados com Jest, executada sobre um banco de dados isolado em memória, o que permite validar o sistema sem depender da infraestrutura de produção (Figura 28). Além dos testes automatizados, a integração com o aplicativo móvel de coleta foi validada de ponta a ponta — do envio das leituras pelo aplicativo à sua persistência no servidor e exibição na *dashboard* web —, conforme registrado no *Pull Request* de integração[1].

[1] `https://github.com/GAEA-UFMG/pad-mobile/pull/1`

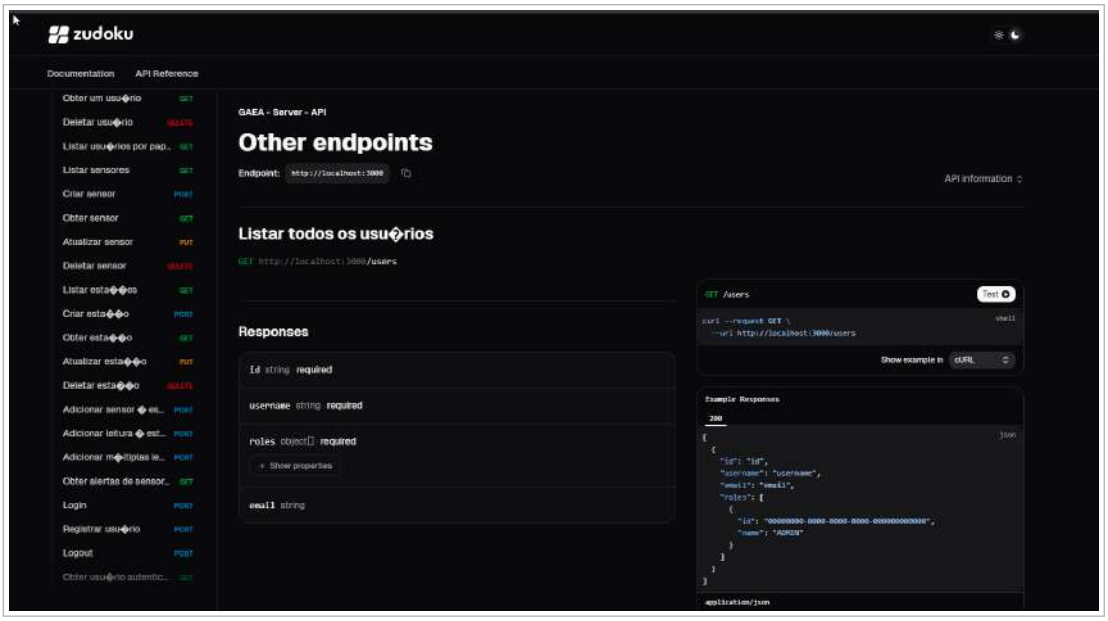
Figura 28 – Execução da suíte de testes automatizados do back-end.



Fonte: Elaborado pelo autor.

4.8 Documentação das Rotas da API com Zudoku

Figura 29 – Documentação rotas da API - Zudoku



Fonte: Elaborado pelo autor.

5 Resultados e Discussões

Ao final do trabalho, todos os objetivos específicos definidos foram atendidos. Os resultados a seguir descrevem o que foi entregue e validado.

- **API RESTful expandida:** a API passou a armazenar e organizar os dados operacionais das estações e ganhou novos *endpoints* de consulta (leituras com filtros, leituras agregadas, alertas, resumo, exportação e acesso público), além de uma rota dedicada ao envio de dados pelo dispositivo, autenticada por chave de API, com auto-cadastro de estações e sensores.

- **Controle de acesso diferenciado:** os papéis de administrador, gestor e operador foram modelados com verificação por papel e por vínculo usuário–estação; gestores e operadores só acessam as estações sob sua responsabilidade, e o auto-registro público cria apenas o perfil de menor privilégio.
- **Dashboards interativos:** foram entregues uma visão geral do sistema, com indicadores e painel de alertas, e um *dashboard* por estação com últimas leituras, alertas, mapa e gráfico filtrável por sensor e período, todos consumindo dados reais.
- **Sistema de alertas:** limiares mínimo e máximo por sensor passaram a ser verificados na ingestão das leituras; valores fora da faixa são marcados e destacados nos painéis e nas listagens, e os erros da API são registrados para acompanhamento operacional.
- **Gerenciamento avançado:** a interface permite editar estações e gerenciar usuários e seus vínculos com estações. A remoção de estações é feita por *soft delete*, preservando todo o histórico.
- **Extração de relatórios:** as leituras de uma estação podem ser exportadas em CSV, com filtros de período e sensor, apoiando análises externas e auditorias.
- **Testes automatizados:** foram desenvolvidos testes cobrindo o *back-end* (autenticação, papéis, estações, sensores, usuários, leituras, exportação, acesso público, *soft delete* e *logs*) e o *front-end* (serviços e componentes principais), executados com banco de testes isolado em memória, sem dependência de infraestrutura externa.

Para validar as funcionalidades e produzir as telas apresentadas, o sistema foi populado com uma estação de demonstração contendo cinco grandezas (vazão, temperatura, pressão, sólidos dissolvidos e volume), com leituras dentro e fora dos limites operacionais, além de um gestor e um operador vinculados. Esse cenário permitiu exercitar tanto os gráficos e filtros quanto o disparo dos alertas.

Espera-se que o sistema contribua para a redução de custos operacionais, a antecipação de falhas, a diminuição de deslocamentos de equipes técnicas e, consequentemente, para maior disponibilidade e continuidade no fornecimento de água potável às comunidades atendidas pelo Programa Água Doce no Semiárido brasileiro. Embora a confirmação plena desses benefícios dependa da operação em escala real, os testes automatizados e os dados de demonstração já evidenciam a viabilidade técnica da proposta.

6 Conclusão

Este trabalho apresentou o desenvolvimento de um sistema web com alertas inteligentes para o monitoramento de estações de tratamento de água do Programa Água Doce, integrando tecnologias de Internet das Coisas através de microcontroladores ESP32 e

uma arquitetura web moderna baseada em Node.js e Angular. A proposta buscou endereçar desafios críticos enfrentados pela gestão das estações de dessalinização no Semiárido brasileiro, onde a ausência de monitoramento em tempo real e a dificuldade de acesso às comunidades remotas comprometem a eficiência operacional e a continuidade do fornecimento de água potável.

A implementação do sistema demonstrou a viabilidade de uma solução tecnológica de baixo custo e alta eficiência para o monitoramento remoto de parâmetros críticos como condutividade elétrica, pressão, temperatura e vazão. A expansão da API RESTful permitiu o armazenamento estruturado e seguro dos dados operacionais, enquanto o sistema de controle de acesso diferenciado garantiu que administradores, gestores e operadores pudessem interagir com o sistema de acordo com suas responsabilidades específicas. A coleta em campo, realizada por um aplicativo móvel já existente, foi integrada ao sistema por meio de uma rotina de sincronização que envia as leituras ao servidor, e a confiabilidade das entregas foi sustentada por uma arquitetura modular em camadas e por uma suíte de testes automatizados, construídas ao longo do desenvolvimento.

Os *dashboards* interativos desenvolvidos proporcionam visualização clara e intuitiva dos dados, facilitando a tomada de decisão e permitindo a identificação rápida de anomalias operacionais. O módulo de alertas inteligentes configura-se como um componente fundamental para a manutenção preventiva, disparando notificações visuais quando os parâmetros monitorados ultrapassam os limiares estabelecidos, reduzindo significativamente o tempo de resposta a situações críticas.

Do ponto de vista social, o sistema contribui diretamente para a garantia de acesso contínuo à água potável em comunidades vulneráveis do Semiárido, reduzindo riscos de interrupções no abastecimento e otimizando a utilização de recursos públicos. A capacidade de extração de relatórios consolidados apoia processos de auditoria e análises históricas e, ao disponibilizar uma página pública de consulta, a solução fortalece a transparência e a prestação de contas previstas no modelo de gestão comunitária do programa, apoiando o acompanhamento coletivo definido nos acordos de gestão.

O acompanhamento de grandezas como salinidade e temperatura conecta-se ainda à dimensão ambiental e produtiva do Programa Água Doce, na qual o concentrado da dessalinização é reaproveitado em atividades como a criação de peixes, agregando renda às famílias e reduzindo o impacto do descarte. Em conjunto, esses resultados indicam que a solução desenvolvida pode contribuir não apenas para a eficiência operacional, mas também para a sustentabilidade ambiental e a governança participativa da gestão hídrica no Semiárido brasileiro.

7 Trabalhos Futuros

Como continuidade natural deste projeto, identificam-se etapas para consolidar a aplicação prática do sistema e ampliar seu alcance.

Evolução das funcionalidades: entre as melhorias previstas estão a notificação ativa dos alertas por e-mail ou *push* (hoje a sinalização é visual, na própria aplicação) e o suporte a sensores capazes de medir múltiplas grandezas em um único módulo, o que exige uma evolução do modelo de dados. Esses pontos foram deliberadamente deixados para uma etapa posterior por dependerem de infraestrutura adicional de envio ou de mudanças estruturais no domínio.

Validação e aceitação das funcionalidades: antes da adoção em larga escala, pretende-se avaliar a aceitação do sistema junto a seus usuários, em duas etapas. Na primeira, a equipe de Engenharia Sanitária (que atua como intermediária entre o desenvolvimento e a operação em campo) validaria as funcionalidades quanto à aderência aos requisitos técnicos e às necessidades de gestão do programa. Na segunda, já com o sistema em uso, os próprios operadores em campo avaliariam a usabilidade e a utilidade das funcionalidades no cotidiano, fornecendo retorno para ajustes e para a definição de novas prioridades. Essa validação em camadas ajuda a assegurar que a solução atenda tanto aos critérios técnicos quanto à realidade operacional das comunidades.

Homologação em ambiente de produção: A primeira etapa dos trabalhos futuros consiste na implantação do sistema em ambiente de produção real, com a instalação dos microcontroladores ESP32 nas estações de dessalinização selecionadas como piloto. Este processo envolverá testes de integração com a infraestrutura existente, validação da conectividade em condições reais de operação e ajustes necessários para garantir a estabilidade e confiabilidade do sistema em campo. A homologação permitirá identificar eventuais desafios técnicos não previstos em ambiente de desenvolvimento e realizar os refinamentos necessários antes da expansão para outras estações.

Treinamento de operadores e gestores: Para garantir a efetiva utilização do sistema, será fundamental desenvolver e executar um programa de capacitação direcionado aos diferentes perfis de usuários. Os operadores locais receberão treinamento prático sobre o monitoramento básico das estações através das interfaces web, interpretação dos indicadores visuais e procedimentos de resposta a alertas. Os gestores serão capacitados nas funcionalidades avançadas do sistema, incluindo análise de relatórios consolidados, configuração de limiares de alerta e utilização dos dados históricos para planejamento e tomada de decisão. Este processo de capacitação é essencial para a apropriação da tecnologia pelos usuários finais e para maximizar os benefícios do sistema.

Coleta e análise de dados operacionais reais: Com o sistema em operação, iniciar-se-á um período de coleta sistemática de dados operacionais das estações monitoradas. Esses dados permitirão validar os modelos de detecção de anomalias, ajustar os limiares

de alerta com base em padrões reais de funcionamento e construir uma base histórica robusta para análises futuras. A coleta de dados em escala real também possibilitará a identificação de padrões sazonais, tendências de degradação de equipamentos e correlações entre variáveis que não seriam observáveis em ambiente simulado.

Metrificação dos benefícios gerados: Uma etapa crucial será a mensuração objetiva dos benefícios proporcionados pelo sistema. Serão estabelecidas métricas quantitativas e qualitativas para avaliar o impacto da solução, incluindo: redução no tempo médio de resposta a falhas, diminuição do número de interrupções no fornecimento de água, economia em custos de deslocamento de equipes técnicas, aumento na disponibilidade operacional das estações, e melhoria nos indicadores de qualidade da água tratada. A comparação desses indicadores com dados históricos anteriores à implementação do sistema permitirá documentar de forma objetiva o retorno sobre o investimento e justificar a expansão da solução para outras estações do programa.

As etapas descritas complementam-se entre si, da validação inicial à mensuração objetiva de impacto. A execução desses trabalhos futuros não apenas consolidará a aplicação prática do sistema desenvolvido, mas também fornecerá evidências concretas de seu valor para a gestão sustentável dos recursos hídricos no Semiárido brasileiro, potencialmente servindo como modelo para iniciativas similares em outras regiões do país e do mundo.

Referências

Agafonkin, V. (2015). Leaflet: an open-source javascript library for mobile-friendly interactive maps. Acesso em: 12 set. 2025.

Ambler, S. W. (2003). *Agile Database Techniques: Effective Strategies for the Agile Software Developer*. John Wiley & Sons.

Atzori, L., Iera, A., and Morabito, G. (2010). The internet of things: A survey. *Computer Networks*, 54(15):2787–2805.

Beck, K. et al. (2001). Manifesto for agile software development. Acesso em: 03 out. 2025.

Brasil. Ministério do Meio Ambiente (2010). Programa água doce: localidades atendidas em alagoas.

Brasil. Ministério do Meio Ambiente (2017a). Caminho da sustentabilidade: componente de sustentabilidade ambiental do programa água doce. Cartilha.

Brasil. Ministério do Meio Ambiente (2017b). Cuidando da nossa água: componente de sustentabilidade ambiental do programa água doce. Cartilha.

Brasil. Ministério do Meio Ambiente (2017c). Vamos fazer um acordo: componente de sustentabilidade ambiental do programa água doce. Cartilha.

Brasil. Ministério da Integração e do Desenvolvimento Regional (2025). Programa Água doce. Acesso em: 21 abr. 2026.

Brasil. Ministério do Meio Ambiente (2012). Programa Água doce: documento base. Technical report, MMA, Brasília.

Breunig, M. M. et al. (2000). Lof: identifying density-based local outliers. In *ACM SIGMOD Record*, volume 29, pages 93–104.

Chandola, V., Banerjee, A., and Kumar, V. (2009). Anomaly detection: A survey. *ACM Computing Surveys*, 41(3):1–58.

Chung, L., Nixon, B. A., Yu, E., and Mylopoulos, J. (2012). *Non-functional Requirements in Software Engineering*. Springer Science & Business Media.

Elimelech, M. and Phillip, W. A. (2011). The future of seawater desalination: energy, technology, and the environment. *Science*, 333(6043):712–717.

Fain, Y. and Moiseev, A. (2018). *Angular Development with TypeScript*. Manning Publications, 2 edition.

Fielding, R. T. (2000). *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine.

Forhad, H. M., Uddin, M. R., Chakrovorty, R. S., et al. (2024). An iot-based water quality monitoring system with real-time alert mechanism, historical data logging, and remote supervision. *Heliyon*, 10(23):e40746.

Greenlee, L. F. et al. (2009). Reverse osmosis desalination: Water sources, technology, and today's challenges. *Water Research*, 43(9):2317–2348.

Hayes, M. A. and Capretz, M. A. (2015). Contextual anomaly detection framework for big sensor data. *Journal of Big Data*, 2(1):1–22.

How2Electronics (2023). Iot based drinking water quality monitoring system with esp32. Acesso em: 18 nov. 2025.

IoT Project Ideas (2025). Iot water quality monitoring with tds sensor & esp32. Acesso em: 09 mar. 2026.

Jan, F., Min-Allah, N., and Düştegör, D. (2021). Iot based smart water quality monitoring: Recent techniques, trends and challenges for domestic applications. *Water*, 13(13):1729.

Jones, M., Bradley, J., and Sakimura, N. (2015). Json web token (jwt). Technical Report RFC 7519, Internet Engineering Task Force.

Kodali, R. K. and Soratkal, S. R. (2016). Mqtt based home automation system using esp8266. In *2016 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, pages 1–5. IEEE.

Kvist, M. (2023). A general scada application for water and wastewater. Technical report.

Merkel, D. (2014). Docker: lightweight linux containers for consistent development and deployment. *Linux Journal*, 2014(239):2.

Miller, M. (2023). Internet of things (iot) for water quality monitoring: a review. *Sensors*, 23(2):4729.

Mitrović, N. et al. (2023). Implementation and testing of websocket protocol in esp32 based iot systems. *Facta Universitatis, Series: Electronics and Energetics*, 36(2):267–284.

Murugan, T. M. et al. (2023). Monitoring and controlling the desalination plant using iot. *Measurement: Sensors*, 27:100720.

Nelli, F. (2018). *Python Data Analytics: With Pandas, NumPy, and Matplotlib*. Apress, 2 edition.

Pressman, R. S. and Maxim, B. R. (2014). *Software Engineering: A Practitioner's Approach*. McGraw-Hill Education, 8 edition.

Queiroz, D. A. S. E. (2026). *Saneamento 5.0: integração tecnológica da gestão de sistemas de separação por membranas no contexto do Programa Água Doce*. Tese (doutorado em saneamento, meio ambiente e recursos hídricos), Universidade Federal de Minas Gerais, Escola de Engenharia, Belo Horizonte.

Raghunandan, K. (2022). Supervisory control and data acquisition (scada). In *Introduction to Wireless Communications and Networks: A Practical Perspective*, pages 321–337. Springer.

Raharjana, I. K., Siahaan, D., and Fatichah, C. (2021). User stories and natural language processing: A systematic literature review. *IEEE Access*, 9:53811–53826.

Ramakrishnan, R. and Gehrke, J. (2003). *Database Management Systems*. McGraw-Hill, 3 edition.

Saia, A. (2018). O acordo de gestão do programa Água doce: importância da gestão compartilhada dos bens comuns para o desenvolvimento local. Acesso em: 15 ago. 2025.

Schwaber, K. and Sutherland, J. (2020). The scrum guide: The definitive guide to scrum. Acesso em: 28 jan. 2026.

Sicari, S. et al. (2015). Security, privacy and trust in internet of things: The road ahead. *Computer Networks*, 76:146–164.

Sommerville, I. (2011). *Software Engineering*. Pearson Education, 9 edition.

Suárez, L. J. S. et al. (2023). Aplicación en flutter para gestión de equipo iot. Master's thesis, Universidad de Granada.

Tilkov, S. and Vinoski, S. (2010). Node.js: Using javascript to build high-performance network programs. *IEEE Internet Computing*, 14(6):80–83.

Zulkifli, C. Z. et al. (2022). A systematic review of iot-based water quality monitoring systems: applications, challenges and future directions. *Water*, 14(22):3621.