

Danillo Vieira

Fullstack Engineer

danillovieira.com

Lisbon, Portugal

Mobile: +351 967 804 318 | **Email:** danillovieira7@gmail.com | **LinkedIn:** linkedin.com/in/danvieira-dev | **github:** github.com/Danvieira7

Full-Stack Engineer with 7+ years architecting and shipping cross-platform applications (React, React Native, Next.js, TypeScript) across web, mobile, and desktop, including production blockchain transaction platforms, forensic-report generation pipelines, and internal developer tooling adopted across engineering teams. Bilingual (Portuguese/English), dual citizen (Brazil/US), EU work permit.

Technical Skills

Languages: TypeScript · JavaScript · Python · Go

Frameworks: React.js · React Native · Next.js · Redux · Node.js · Express.js · Storybook · Quarto · Tailwind CSS

Tools: Vite · Firebase · Vercel · Supabase · python-docx · lxml · REST APIs · Heroku · HTML5 · CSS3 · Styled-components · SCSS · ESLint · Prettier

CI/CD: GitHub Actions · Jenkins

Testing: Jest · Cypress · React Testing Library

Databases: MongoDB · PostgreSQL · MySQL

Experience

Full-Stack Engineer - Token Recovery

Lisbon, PT | Apr – Sep 2026

- Designed a two-mode Python pipeline (Quarto, python-docx, lxml) that compiles 21 modular JSON section files (43 KB) into a publication-ready .docx in under 300ms — supporting first-run generation from raw API context and on-demand regeneration from analyst-edited section files.
- Built a multi-stage normalization layer handling type coercion, block merging, field defaulting, and activation filtering across 100+ content blocks; serialized the final hierarchically numbered document model to JSON before render.
- Engineered a two-pass regex footnote engine scanning 37,000+ characters per run — with unicode superscript mapping and keyword deduplication — auto-injecting Quarto inline footnote syntax throughout the compiled document.
- Implemented a 7-pass python-docx post-processing pipeline producing output Quarto can't render natively: full-bleed tables, TOC field injection, cover styling, dynamic page breaks, and Page X of Y footers — delivering a 663 KB formatted report.

Full-Stack Engineer - Teranode Group

Lisbon, PT | Nov 2020 – Apr 2026

- Spearheaded end-to-end development of two production applications built from scratch within a Turborepo monorepo (10 shared packages):
 - Teranode Console – blockchain transaction management platform with real-time analytics, developer tools, and access to Token Studio & Teranode Wallet.
 - Teranode Admin Dashboard – internal monitoring tool enabling employees to track app performance, assist customers, and manage services in real time.
- Architected the shared UI system across the platform using 10 internal packages — including a UI library (168 components and icons), a shared hooks package (9 custom hooks), typed themes, utilities, and a wallet client — enabling consistent patterns and zero duplication across products.
- Implemented and documented the design system in Storybook (50+ components), featuring interactive demos, prop controls, visual guidelines, and theme customization—empowering other developers to easily adopt, extend, and maintain consistent UI across all apps.
- Engineered state management with Redux and secure session handling — covering authentication flows, cookie lifecycle, and role-based access across the Console and Admin Dashboard.
- Translated raw blockchain service data — transaction volumes, wallet distributions, and chain analytics — into real-time graphs and monitoring dashboards consumed by end users and internal support teams.
- Collaborated with backend teams to build and consume RESTful APIs (Go, [Node.js](#)).
- Developed React Native and universal React applications targeting 3 platforms (web, iOS, and Android).

Freelance Full-Stack Developer

Los Angeles, USA | Mar 2019 – Nov 2020

- Built a full-stack app for Six Sweet Under from scratch — React SPA, Node.js/Express MVC backend, MongoDB, REST API integration, and Heroku deployment.
- Rebuilt the Freck Beauty e-commerce storefront (Next.js, WooCommerce API) — handling full payment and product DB integration end-to-end, contributing to a 12% increase in sales.
- Overhauled the purchase flow by replacing disruptive cart-page redirects with a nonblocking checkout prompt, and converted full-page product pagination to async WooCommerce API calls — keeping users in the browsing flow and eliminating unnecessary reloads.

Projects

Abbey - Stem Player

Lisbon, PT | 2026

abbeyplayer.com

A real-time multi-track stem player for musicians (React 18, TypeScript, Web Audio API): per-instrument mute, solo, and volume control designed for live rehearsal and performance, not casual playback.

- Architected a pnpm monorepo spanning a React/Vite web app, an Electron desktop build, and a shared player-core package: enforcing a strict service-layer boundary so the same audio engine and UI ran unmodified in both a browser tab and an Electron renderer, eliminating a duplicate implementation of the mixing engine.
- Engineered a real-time Web Audio API mixing engine: independent per-stem gain/mute/solo nodes, a BS.1770-approximated LUFS/peak meter, and multi-channel output routing (setSinkId) that sends the music mix and the click/count-in cues to separate physical outputs, designed for live stage use, not just headphone playback.
- Integrated an AI stem-separation pipeline (Replicate/Demucs) as a resumable background job that survives page navigation and app restarts, streaming results into Supabase Storage and syncing live progress across band members via Firestore.
- Isolated a third-party API credential from the client across three different runtimes (a Vite dev proxy, a Vercel Edge Function, and an Electron main-process IPC handler) by designing one client-side service contract all three satisfy identically, so the security model held as the product grew from web-only to a signed cross-platform desktop app (macOS/Windows, via electron-builder).

Chico Romero - Band Platform

Lisbon, PT | 2026

chicoromelo.com

A full-stack platform for Chico Romero, a working Portuguese band (Next.js 14, React 18, Supabase/PostgreSQL): a public band website paired with a private, role-gated admin system the band uses to run real day-to-day operations.

- Optimized for performance and SEO using getStaticProps with Incremental Static Regeneration (24h revalidation), pre-rendering public pages at build time for fast, crawlable, search-indexable delivery without hitting the database on every visit; locked down admin routes with Next.js Edge Middleware enforcing Supabase session auth and an email whitelist.
- Designed a tiered RBAC system (5 permission levels) enforced independently at two layers: Next.js Edge Middleware, gating every request before it reaches a page, and per-route API

handlers re-validating the caller's token and tier, so a bypassed or forged client request could never reach data outside its permission scope.

- Shipped a production admin platform actively used to run a real organization's operations (event/gear/staff scheduling, inventory, and a custom band-performance evaluation system), backed by a PostgreSQL schema evolved through 20+ sequential migrations rather than ad hoc changes.
- Built an idempotent, externally-triggered SMS reminder pipeline (Twilio) after determining Vercel's built-in cron couldn't meet the product's timing requirements, logging every send to guarantee exactly-once delivery even when scheduler runs overlap; added field-level AES-256-GCM encryption for stored credentials with a backward-compatible decrypt path, shipping the security upgrade with zero downtime or data migration.

Education

General Assembly

Santa Monica, USA | 2019

Software Engineering | Data Structures and Algorithms