David Yates

Platform Rescue and Re-platforming Contractor: Node.js, TypeScript, PHP, Terraform, Kubernetes

Crewe, Cheshire | david@oneforthecode.com | 07867 697095

I am most often brought in when a platform already exists and is in trouble: re-platforming Sky Casino, decoupling Missguided's storefront from a legacy Magento stack, picking up Low6's NHL platform mid-flight and consolidating single sign-on across every game on it. Nearly always regulated or high-traffic, across fintech, betting and gaming, telehealth, insurance and e-commerce. 15+ years, with Node.js and TypeScript at the centre, React on most engagements, Laravel where the stack is PHP, and enough Terraform, Docker and Kubernetes to own the delivery pipeline rather than wait on someone else's. A pattern across all of it: I end up building the tooling the team was missing, the local development environments, the form and component harnesses, the mock servers, the CI gates, because the fastest way to fix delivery is usually to fix what the developers are working with. I contract through my own company, Built By Up, and via Apex Enterprise Consulting, bringing in and mentoring developers when an engagement needs the capacity.

LinkedIn https://www.linkedin.com/in/dave-yates-0064a1136/ GitHub https://github.com/s3y

SKILLS

Core competencies	Platform rescue, Re-platforming, Legacy migration, Full-stack development, API design, Front-end architecture, Micro-frontends, Design systems, Component libraries, UI development, UX development, Responsive websites, Accessibility, Mobile application development, iOS development, Android development, E-commerce development, Payments, Identity and access, Distributed systems, Event-driven architecture, Infrastructure as code, CI/CD, Self-hosting and homelab, Relational databases, NoSQL databases, Data migration, Software planning, Test-driven development, Agile, Scrum, Best practice and standards, Mentoring and tutoring, Technical leadership, Hardware support, Network and server management, Machine learning platforms
Ways of working	Communication, from an early career in sales, Leadership and prioritising team workload, Problem solving under pressure, Knowledge sharing, Planning and organising, Turning a written concept into a working prototype, Pair programming, Code review, Working with BAs, testers and product owners, Remote and distributed teams, Working across time zones
Platforms	Linux, UNIX, macOS, Windows, Android, iOS, AWS, Azure, Google Cloud, Cloudflare, Proxmox, Kubernetes, OpenShift
Languages, commercial	TypeScript, JavaScript, PHP, C#, Java, SQL, Bash
Languages, own projects	Python, Swift, Rust, Go
Backend and APIs	Node.js, Bun, Express, Laravel, GraphQL, Apollo GraphQL, REST, tRPC, Swagger, SOAP, WebSockets, Microservices, Domain Driven Design
Frontend	React, Next.js, Vue, Angular, AngularJS, React Native, Expo, Redux, RxJS, SWR, Tailwind, Sass, LESS, Bootstrap, Material Design, Storybook, Vite, npm workspaces, Lerna, Gatsby, Inertia, Micro-frontends
Data	PostgreSQL, MS SQL, MySQL, Oracle, MongoDB, InfluxDB, Redis, Prisma, Drizzle, Entity Framework, X12 EDI, XML feeds
Messaging and eventing	Azure Service Bus, Kafka, NATS, RabbitMQ, Pusher, Dead letter queues, Event schemas, Correlation IDs
Identity and security	Single sign-on, Keycloak, JWT, RBAC, CSRF, Webhook signature verification, GDPR, PII
Platform and DevOps	Docker, Docker Compose, Kubernetes, OpenShift, kubectl, Terraform, OpenTofu, Ansible, GitOps, Jenkins, GitHub Actions, GitLab CI, Azure DevOps, AWS, Azure, AWS Lambda, AWS S3, AWS EC2, AWS RDS, AWS IAM, AWS CloudFront, AWS API Gateway, Cloudflare, PM2, Linux, Proxmox, Caddy, Tailscale
Observability and performance	Prometheus, Grafana, Datadog, New Relic, Kibana, Telemetry, Heap profiling, Load testing, Artillery, Optimizely, Hotjar
Testing and quality	Jest, Vitest, Playwright, Cypress, Cucumber, Test-driven development, Mocking, CI gates, Static analysis, Code review
Commerce, CMS and integrations	Magento, WordPress, Shopify, Sitecore, Custom CMS, Fredhopper, Salesforce, Zendesk, ShipStation, Coinflow, Airtable, Alexa API, OCR, LLM integration

Linux, shell and self-hosting	Linux, Terminal-first, Neovim, Emacs, Bash, sed, awk, grep, systemd, cron and timers, SSH, Proxmox, Caddy, nginx, Tailscale, Authentik, Pi-hole, Unbound, DNS, Reverse proxies, SOPS, Backups, Networking
Documentation and diagramming	Mermaid, PlantUML, draw.io, Excalidraw, Architecture diagrams, Auth flow diagrams, Runbooks

EXPERIENCE

Continuously in work since 2009.

Naturecan · Software Engineer via Apex Enterprise Consulting **May 2026 to Present**

Brought in to work on a prescription and telehealth platform behind a supplements and wellness brand, along with its localised multi-currency storefronts. The work spans the payment path, the compliance around it, and the integrations the operation runs on, with a good deal of GDPR and PII alongside. Presided over a migration from Bun to Node, and worked with the stakeholders daily, including on managing their AI budgets and getting AI workflows working well for the team. The storefronts ship to more than 40 countries.

- Built the payment functionality the finance side depends on, including gap-free sequential invoice numbering and failed-payment handling.
- Hardened the payment path with webhook signature verification across three providers, and put role-based access control across the payment features.
- Integrated Zendesk for support queries and ShipStation for fulfilment, both webhook-driven, with telemetry behind them.
- Migrated the platform runtime from Bun to Node.

Node.js, Bun, TypeScript, React, Docker, Payments, RBAC, GDPR, PII, Zendesk, ShipStation, Telemetry, Git, GitHub, JIRA, Mermaid

Low6 · Full Stack Engineer via Apex Enterprise Consulting, US hours **Sep 2025 to Jul 2026**

Brought in to build Goal Chase, a game on the NHL GameZone platform, working American hours alongside a day contract because the clients are US-based, against a four to six week deadline for what was realistically a two to three month build from scratch. Stayed to attack the reason every game started from scratch: around 400 repositories, a repo per service, nothing reusable and no testing strategy at all. Pitched and led GameZone V2, sold as a re-platform but really a fix for the architecture, where the front end, the back end, the auth service and every individual game each minted their own tokens and held their own user data. Built the custom admin tooling the games are run from, which meant a great deal of time zone work, and found answers for the gaps in the sports feeds: janky XML, missing player data, missing jerseys.

- Consolidated single sign-on across the platform and every game in under a week, replacing four separate token-minting implementations, and closed the hardcoded SSO state behind it as a CSRF fix.
- Recovered a bingo scoring defect that had mis-scored roughly 70,000 players, saving the commercial relationship.
- Production incidents down from daily to none by the time I left, having set the definition of done and the non-functional requirements the company had been working without.
- Shipped and supported Goal Chase, NxtBet, Brackets, Bingo, StatsStreak and Lines, with the quest system and unified admin behind them.
- Moved the estate onto Service Bus with defined event schemas, dead letter queue monitoring and correlation IDs through every request and log.
- Migrated the data from MSSQL to PostgreSQL, including the quest system off a hyperscale database and the duplicate-user cleanup that came with it.
- Built the Azure infrastructure as Terraform modules: Container Apps, Key Vault secrets, Static Web Apps and Functions, with OIDC federation so pipelines held no credentials.
- Integrated Coinflow for payments on NxtBet, and a managed auth provider alongside it.
- Brought around 400 repositories, one service per repo, under standard practice and a first testing strategy.
- Azure estate where the NHL platform alone ran about £10k a month, worked continuously for cost reduction.
- Delivered against a fixed tournament deadline, with traffic peaking in the tens of thousands of concurrent users during match windows.
- Load tested the platform with Artillery ahead of every path to live.
- Made the platform white label, demonstrating bookmaker, NFL and NBA builds alongside the NHL one to show how far brands, assets and games could be swapped.
- Built the alerting and monitoring they had none of, Slack handlers on CI runs and errors, so incidents surfaced internally rather than arriving from the client.
- Productised the offering into consumable SDKs, with custom admin and leaderboard functionality built to be reused rather than rebuilt per game.
- Ran the Terraform and OpenTofu orchestration on Azure day to day, and handled production incidents, PII and GDPR.
- Migrated the runtime from Bun to Node.

Node.js, Bun, TypeScript, React, Azure, Terraform, OpenTofu, Docker, PostgreSQL, MS SQL, Redis, Drizzle, Service Bus, Single sign-on, White label, Payments, XML feeds, CI/CD, Playwright, GDPR, PII, Git, JIRA, Artillery, Load testing, Mermaid, draw.io, Ansible

CDP · Node Engineer via Apex Enterprise Consulting Feb 2025 to Feb 2026

Joined CDP's environmental disclosure platform as a Node engineer, owning the copy-forward and questionnaire setup that every reporting company passes through before it can file for the year. It presents as a form and is nothing like one underneath: a wizard carrying a great deal of conditional logic, built in React over Node services with Prisma over MS SQL, and integrated with the disclosure management service. Load is seasonal rather than steady, arriving as a burst when the reporting window opens. Nobody asked for a local development environment because there wasn't one. Onboarding was poor enough that I wrote my own scripts to do the job, then tidied them up and opened a PR: Bash tying together the Docker Compose files, service wiring and seed data so most of the platform came up on a laptop, which mattered because the services lean on data held elsewhere in the platform and bugs were otherwise very hard to reproduce. Worked the Azure DevOps pipelines throughout. The programme takes disclosures from around 25,000 companies, roughly two-thirds of global market capitalisation.

- Ran submissions through to scoring, including batches such as 115 organisations sitting in amendment state.
- Built out the guidance journey: a server-rendered guidance page, select-all on guidance prompts, and PDF generation carrying visible tags, options lists, mandatory questions and transposed matrix tables.
- Wrote the local development environment nobody had asked for, then handed it to the team as a pull request.

Node.js, TypeScript, React, Prisma, MS SQL, Docker, Docker Compose, Bash, Azure DevOps, CI/CD, Git, Mermaid, Ansible

Hargreaves Lansdown · Senior Engineer via Built By Up Mar 2024 to Feb 2025

Brought in to modernise the cash ISA transfer process, a React form over their legacy PHP stack, inside an FCA-regulated release process where changes go to committee. That meant working out a large estate of Java microservices, and writing custom auth layers to bridge the legacy PHP stack and AWS so the forms would load past the access control layer at all. The second project digitised the movement of assets between holdings, taking sort code and account number and integrating the services that pull the information down: a React front end hosted in S3, serverless behind it posting onto a queue their Java transfers engine picks up, with Terraform and GitLab pipelines as the bread and butter. Contracts across Salesforce and the downstream services were Swagger-defined, so a lot of the work was holding my own boundaries rather than assuming everyone had read them the same way. The hard part was the estate rather than the code: half on-premise and half moving to AWS, WAF and ACL limits, users to be passed between services, and everything to be walked through development and staging before it saw production. They also wanted a repeatable way to produce React forms, so I built one: npm workspaces, having started in Nx, with a Vite dev server that injected development variables into the window object so every form project bootstrapped from a single page and the form under test could be swapped without a restart. It is documented and still in the project. Set up the pipelines and working practices for both projects, and implemented the Flare component library after the design team was dismantled three months in. When the testers went as well, I covered it with my own end-to-end tests and stricter unit tests. A FTSE 100 platform, over 1.8m retail investors and more than £150bn under administration.

- Cash ISA transfer forms are live in production.
- Built the repeatable React form tooling: npm workspaces with a Vite dev server injecting development variables into the window, so every form project bootstrapped from one page and the form under test swapped without a restart. Documented and still in the project.
- Wrote the custom auth layers bridging the legacy PHP stack and AWS, so forms served from S3 got past the access control layer.
- Implemented the Flare component library after the design team was dismantled, and covered the lost test team with end-to-end and stricter unit tests.

React, PHP, Java, AWS, AWS S3, AWS Serverless, Terraform, GitLab CI, Salesforce, Swagger, Storybook, Vite, npm workspaces, Playwright, Bash, Git, JIRA, draw.io

Stylus Education · Software Engineer via Built By Up Feb 2024 to Mar 2024

Joined an education startup building an automatic marker for exam papers sent out to schools, with the results feeding the next set of papers a student is given. The founder is an ex-teacher who had the idea long before the tooling existed and was running it on Airtable, so the first work was structural: lock the schema down and get the data off a spreadsheet pretending to be a database. Marking with a language model is defensible with the right guardrails, and I have always argued for deterministic results wherever one will do, so the guardrails were the job. The real obstacle was not the model but the paper: scans came in at poor quality and many questions carried images the model had to read. Solved it at the input rather than the prompt, with strong OCR and image pre-processing so submissions were legible enough to be inferred from correctly.

- Made poor-quality scanned exam papers legible enough to mark reliably, by fixing the input with OCR and pre-processing rather than by prompting harder.
- Marking papers for 20 to 30 Key Stage 2 schools, and every pupil sitting them.

Node.js, TypeScript, React, OCR, LLM integration, Prompt guardrails, Airtable, Git, JIRA

PepsiCo · Software Engineer via Built By Up Jan 2024 to Feb 2024

Brought in to rescue a React Native build that was already late. A developer I had mentored had been hired to deliver it and the project had got away from them: God classes, and AI thrown at problems in place of understanding them. Took it apart and got it into shape against the deadline. The app itself was an internal social feed for PepsiCo staff, posts and engagement rewarded with points on a daily timer, React Native against an API with Cloudflare in front. Most of the value was in insisting on fundamentals over whichever tool was to hand that week.

- Rescued a late React Native build and delivered it inside the deadline it had already missed.
- Internal audience of more than 2,000 staff.

React Native, TypeScript, Node.js, Cloudflare, Git, JIRA

BBC · Senior Engineer via Built By Up Feb 2023 to Dec 2023

Joined as the sole developer embedded in a BBC squad, working the international sign-on project. Three services sat behind it, a session service in Node whose job was swapping tokens, plus a front-end and a back-end auth service, so most of the work was middleware, written test-first throughout. Built the form changes for the new sign-up flows, which meant working around geo-caching, and integrated Optimizely to run the experiments: bucketing users, reading what came back and reporting the results to the business, including the several occasions where the honest answer was that the existing flow was already the better one. Deployments and feature toggles both ran through their own AWS-based pipeline. Accessibility carried real weight there, and traffic at that scale was handled with far less machinery than you would expect, Node and PM2 scaled up and down as demand moved. Audience in the tens of millions.

- Built the sign-up form changes for the new international auth flows, working around geo-caching.
- Integrated Optimizely and ran the experiments end to end, bucketing users and reporting results back to the business.

Node.js, React, Redux, React Context, Optimizely, PM2, AWS, Jest, Test-driven development, Docker, Bash, Git, GitHub, JIRA, Excalidraw

General System · Senior Full Stack Engineer Feb 2022 to Feb 2023

Brought in to help take a Ruby on Rails monolith to Node microservices on Kubernetes, on a platform that ties a government operations centre together. First work was the interop handshakes between operations centres over the Matrix protocol, so one command room could work with another, a capability they had neither on the platform nor much of in practice. Verification was a short authentication string, three five-letter words drawn from a Wordle dictionary that the joining room read back to the room that issued them. Then the asset ingestion pipeline behind the vault where an operation's documents are processed and held, where I argued for the builder pattern and friends over the procedural approach in front of me. Also worked on Keycloak, including shipping a custom Java build for it, and on Postgres, where the schema was expressed almost entirely through views. Pushed the move from JavaScript to TypeScript and brought the architect round to it. Later work was the React Native app: a better mapping overlay, and boundaries drawn on a map that install triggers, so a target crossing into or out of an area raises an alert. Each feature was mine to deliver end to end, the BA keeping them separable enough for that to work. The local development environment was Bash over a stack of Docker services, and everything shipped through self-hosted GitLab. The product is built to be installed in air-gapped environments.

- Around 15 Node microservices on Docker and Kubernetes, designed to run in air-gapped environments.
- Built the interop handshake letting one operations centre work with another over the Matrix protocol, verified by a short authentication string read back between rooms.
- Wrote the asset ingestion pipeline behind the document vault.
- Delivered the mapping overlay and map-drawn boundaries in the React Native app, solving the point-in-polygon test so that crossing a boundary raises an alert.
- Moved the codebase from JavaScript to TypeScript, having made the case to the architect.
- Began consolidating state management into one coherent library, React Context and Redux both being used heavily across the front end and the combination having grown unwieldy.

Node.js, TypeScript, Kubernetes, Docker, Postgres, Matrix protocol, Keycloak, Java, Ruby on Rails, React, React Native, Redux, State management, React Context, Bash, GitLab CI, Jest, Prometheus, Grafana, Git, JIRA, PlantUML, Ansible

Sky Betting & Gaming · Software Engineer via Built By Up, Remote Nov 2021 to Feb 2022

Returned for a three month engagement on cross-bookmaker account consolidation, letting a customer who holds accounts at more than one bookmaker bring them into one. Built the front end for it: asking whether other accounts exist, taking consent to share information between them, and starting the matching from there. Behind it a Kafka queue fed a Java service doing the cross-account matching, and the work reached into PHP because Sky's original authentication layer was written entirely in it. Left it ready to deploy at the end of the contract, release timing sitting with Sky rather than with us.

- Came back in to carry the authentication migration as the platform consolidated, working directly with the platform team.

React, Java, PHP, Kafka, Docker, Git, JIRA

AirNow · React Developer Contract, remote Aug 2021 to Nov 2021

Shipped features on an app market intelligence platform, the sort that tells a publisher what its competitors are earning: download and revenue estimates, keyword and store performance, competitor tracking. React with Apollo and GraphQL over an AWS backend, so most of it was presenting large, slow-moving datasets in a way somebody could act on. The whole team shared one back-end development environment that every local stack pointed at, so a change by anyone could stop everyone. Rather than argue about it, I built the alternative and showed them: working test first against mocks, with a mock server shipped alongside the local stack so each developer could work in isolation.

- Built and demonstrated a mock server alongside the local stack, in place of the single shared back-end environment the whole team depended on and which took everyone down when it fell over.

React, Apollo GraphQL, GraphQL, TypeScript, AWS, Data visualisation, Test-driven development, Mocking, Node.js, Git, JIRA

GoDaddy · Lead Node Developer Jun 2021 to Aug 2021

Walked in as the sole owner of the core commerce APIs, orders, checkout, basket and payments, after the previous team had left en masse. The brief was the expensive contractor's brief: hold the platform up for three months while two permanent hires were recruited, and have them able to run it by the time I left. The services were written in a hard functional style, Ramda throughout, Fluture in place of promises and Flow in place of TypeScript, with the type coverage broken and very little test cover behind any of it. Downtime on orders or checkout cost around £70,000 an hour, and the usual cause was an upstream team changing a contract the downstream services took as given, the estate being service to service and linear rather than event driven. Spent much of it in kubectl, shepherding pods and replicas on OpenShift with Jenkins in front. Put in the practice that was missing: pull requests and review, test-driven development, and CI worth the name. Made the case for a rewrite, and moved the services towards TypeScript and tests as far as a live estate allows.

- Held up the core commerce estate for a customer base of over 20m across roughly 12 stateless Node services, alone.
- Recruited into a team that had walked out, and left two permanent hires able to run it.
- Downtime on a core commerce service cost around £70,000 an hour.
- Cut Docker build times from six or seven minutes to around forty-five seconds using build stages, which mattered because every incident was lengthened by how long a deploy took.
- Traced the memory leaks bringing services down, working from heap snapshots.

Node.js, RamdaJS, Fluture, Flow, TypeScript, Functional programming, Kubernetes, OpenShift, kubectl, Docker, Jenkins, Kibana, Domain Driven Design, Test-driven development, Git, GitHub, JIRA, Heap profiling, Memory leaks

Good Things Foundation · Senior Software Engineer via Built By Up, Remote Mar 2021 to Jun 2021

Joined as the only senior on the platform behind the UK digital inclusion charity, a learning management system used by educators for community learning, English language teaching among it. A small team of juniors had built it on AWS in Angular and done a genuinely good job of it: the framework's opinions had kept them out of trouble, it was TypeScript throughout and the tests held up. The gap was everything around the code. There was no Terraform at all, so I built it from the ground up and put the platform behind automated CI, and by the time I left they could raise infrastructure from a change in GitHub, which was the GitOps model they wanted for running the services behind the platform. Also ran audits across the estate to see that it was doing what it ought to.

- Led three to four developers technically, as the only senior on the engagement, reviewing their work and setting delivery standards.
- Built the AWS estate as Terraform from nothing, so infrastructure changes went through review rather than by hand.

Angular, TypeScript, AWS, Terraform, GitOps, CI/CD, Docker, Git, GitHub

Sky Betting & Gaming · Full Stack Web Developer via Built By Up Sep 2020 to Mar 2021

Joined Sky to re-platform the Sky Casino front end. It was Next.js already but on a very early version, and I made the case for upgrading in place; they wanted a fresh build, so it became a re-platform onto a newer Next with Apollo and a good deal of GraphQL in front of it, and Terraform injecting the configuration. Took a turn on the PHP upgrade as well, which meant Docker images built and pushed out onto boxes, plenty of SSH and command line. The other half of the job was the permanent team, who needed bringing up on TypeScript, linting, test-driven development and reviews worth the name. Deployment was Jenkins jobs starting Jenkins jobs, at a scale comparable to the BBC work later on. It landed, though my engagement ended once the build was done and only the release remained.

- Re-platformed the front end of Sky Casino, one of the UK's largest online casino products, serving a customer base in the millions.

Next.js, React, TypeScript, Apollo GraphQL, GraphQL, MySQL, Docker, Terraform, Jest, Cypress, PHP, Jenkins, Bash, Git, GitLab, JIRA

Radius Payment Solutions · Head of Front End Permanent, Crewe Jul 2020 to Sep 2020

Took the front-end lead at a fleet and fuel payments business, the brief split between the marketing and micro-site estate on the payments side and feature work on Live Maps, their telematics product, Python behind a React front end on AWS. The strategy I set was to ship micro-sites from configuration rather than build each one by hand, with a shared component library underneath, so a new site became a config change rather than a project. Wrote the reference implementation for it, the iCompario site: Next with a GraphQL layer over Apollo, Tailwind, SWR and typed form handling, carrying its own unit and Cypress coverage plus the image and sitemap optimisation an SEO-led estate lives on. Reported to the head of UX in a heavily matrixed organisation, and left at the end of probation to go back to contract work.

- Shared component library adopted across more than 200 micro-sites.
- Set the strategy to ship micro-sites from configuration rather than build each by hand, on a shared component library.
- Built the iCompario reference implementation: Next with an Apollo GraphQL layer, Tailwind, SWR, typed forms, unit and Cypress coverage, plus image and sitemap optimisation.

Next.js, React, GraphQL, Apollo, Tailwind, SWR, Sass, Jest, Cypress, Python, PHP, AWS RDS, AWS EC2, AWS IAM, AWS S3, AWS CloudFront, AWS CloudWatch, AWS API Gateway, Git, GitLab, JIRA

Built By Up · Freelance Engineer Own company Mar 2020 to Jul 2020

Freelanced through my own company across the first lockdown, on engagements for smaller firms. The substantial one was a Shopify integration in Laravel for a client whose product data arrived as X12 EDI, the ANSI standard that predates most of the web, so I wrote a PHP library to parse it and convert the output into the shipping manifests the integration needed.

- Wrote an X12 EDI parser library in PHP so an archaic product-data feed could drive a Shopify integration.

Laravel, PHP, Shopify, X12 EDI, MySQL, Docker, Git

Missguided · Full Stack Engineer via Upnorth, Manchester Dec 2019 to Mar 2020

Joined to shore up a stretched team through a digital transformation, decoupling the storefront from a legacy Magento stack and shipping React in its place, which went well and carried on throughout. Fredhopper sat behind search and merchandising. The company had chosen to add contractors rather than wait on permanent hires, And Digital alongside us, so day-to-day direction came from me and the other senior contractors rather than from above. The engagement ended in March 2020 when covid shut the country down. Turnover was £202m in the financial year the engagement ran through, to 29 March 2020.

- Decoupled the front end from Magento during a scale-up period at one of the UK's largest fast-fashion e-tailers.

React, Redux, TypeScript, GraphQL, Magento, PHP, Kubernetes, Docker, New Relic, Fredhopper, AWS Lambda, Jenkins, Jest, Cypress, Git, GitLab, JIRA

Assurant · UX and Front End Tech Lead via Upnorth, Crewe Jul 2019 to Nov 2019

Brought in to lead the front end and the UX across the customer support tools and a full UX audit of their Barclays MPI site. The audit produced the usual assets, personas and competitor research, but the useful part was a set of deliverables that let them hand work to third parties and still keep control of the code that came back: a design system, a web style guide and a React component library that could be demoed to the business and emit HTML an agency could actually use. Wrote the wireframes and the high-fidelity mockups, then started the front-end reskin myself to get the delivery moving. On the claims side, through the period they took on the Monzo contract, the mobile claim journey was the bulk of it: a device lookup was needed and the commercial API cost £20,000, so I showed another developer how to scrape the data and stand up our own, covering 90 to 95 per cent of devices in use. Got the claim down to three steps, put analytics behind it so decisions came from data, and managed the contributions of three or four external agencies, one in Brazil.

- Device lookup API built in-house from scraped data, covering 90 to 95 per cent of devices in use, in place of a £20,000 commercial one.

React, C#, Node.js, SQL, RabbitMQ, Design systems, Component libraries, UX audit, Wireframing, Adobe XD, Hotjar, Git, GitLab, JIRA

LateRooms · Principal Software Engineer via Upnorth, Manchester Apr 2019 to Jul 2019

Came in as principal on a white-label travel product, React over Node with Node the interface to a C# microservices estate, and the job turned out to be as much about the team as the code: 12 to 15 developers at every level. Moved them from trunk-based branching to Git flow, which gave the delivery process accountability and made deployments something everyone could follow. Spent a lot of it pair programming with the junior and mid developers, not just on their tickets but on what to do with their mastery time, and ran JS guilds so people had somewhere to pitch ideas and catch each other up. With the seniors I unified the design patterns a non-greenfield codebase had accumulated, moving observables onto redux-thunk. Chose Lerna for the monorepo over Yarn workspaces, and ran the static generation against server rendering question as a week's spike rather than a decree, the developer who favoured Gatsby arriving at his own conclusion, which is the only kind that sticks. Swapped New Relic for Datadog with the back-end principal and improved the Docker pipelines behind CircleCI and Rancher. Everything was test driven, and coverage sat around 92%, which prompted the more useful conversation: that a percentage is not evidence the tests are any good.

- Led a team of 12 to 15 developers across all levels, through a move from trunk-based branching to Git flow.

React, Redux, redux-thunk, RxJS, Node.js, C#, Microservices, Lerna, Yarn, Gatsby, Next.js, Rancher, CircleCI, Datadog, New Relic, Docker, Jest, Cypress, Test-driven development, Git, GitHub, JIRA

Vitality Group Inc · Lead Front End Developer via Upnorth, Bournemouth Sep 2018 to May 2019

Joined a project Vitality were keeping quiet, an investigation into a new insurance line away from the life and health business they are known for, held up in part waiting on a third party's API. C# developers had bootstrapped the front end in AngularJS because that is what shipped with MVC, and I moved it onto modern Angular. The hard part was getting a modern front end to live inside Sitecore: I wrote an adapter that serialised the CMS data and injected it into the window for the front end to pick up, and exported components individually so Sitecore could call one by name and populate it. A shared library plus an unusual way of bootstrapping the application, which is the same shape Sitecore's own JSS later shipped. Micro-frontends before the word was in circulation. Also built an automated pipeline off their Bitbucket repos, and optional switching of data sources while developing. When the third party missed the API and the line was paused I merged into the wider team, found every team building in its own silo, and pitched React as the single front-end framework so that hiring and upskilling became tractable, then led the migration onto a React and Redux architecture and brought the company's own UI and UX library into sync with it. Built a component explorer for it, a custom Storybook in effect, where a component hydrates with real data and its props can be changed on the page, and brought across the work a CSS specialist had done in Jekyll, teaching him React on the way.

- Wrote the adapter that serialised Sitecore data into the window and exported components individually, so the CMS could call one by name and populate it. A micro-frontend in 2018.
- Built a component explorer, a custom Storybook in effect, hydrating components with real data and letting props be changed on the page.
- Pitched React as the single front-end framework across teams that had each been building in their own silo.

Angular, AngularJS, React, Redux, Sitecore, JSS, Micro-frontends, C#, SQL, Bitbucket, Jenkins, Jekyll, Git, JIRA

HSBC · Software Engineer Leeds May 2018 to Sep 2018

Joined to build features on the bank's mobile app in React and React Native, at a point where competitors were moving faster and the app needed to catch up. Part of the work was moving consumption onto modern APIs rather than the legacy services sitting behind them.

React, React Native, REST APIs, C#, JavaScript, Git, JIRA

Inprofit Traders · **Senior Applications Developer** Own product, between contracts **Mar 2018 to May 2018**

Built a Betfair trading platform full time between contracts, my own product rather than a client's. Vue on the front, Laravel and MySQL behind it, InfluxDB holding the tick data, and Redis and WebSockets carrying prices through to the screen. Used machine learning to model and predict price data, with both classification and non-classification methods over the historical set, alongside scraping and the Betfair API itself. The problems were the ones that make trading software hard: latency, the volume coming off the exchange, and pricing that is correct rather than approximately correct.

Vue, Laravel, PHP, MySQL, InfluxDB, Redis, WebSockets, Python, Machine learning, Web scraping, Betfair API, Docker, Git

CoinSchedule · **Software Developer** **Nov 2017 to Feb 2018**

Joined an ICO listings platform through the 2017 crypto boom to bootstrap the development cycle for a new version of the site: Angular 6 with Universal for server rendering, over a Laravel API. Built the service worker, the unit tests and the server rendering, and got into implementing smart contracts on Ethereum. Also did the architectural work on how the infrastructure should roll out through the next phase, which included the competitor review and the case that a business living on being crawled should stay server-rendered rather than move to a single-page app.

- Tracked ICOs and token sales across the market, surfacing crowdsale and exchange data as the 2017 cryptocurrency boom peaked.
- Ran the competitor analysis and made the case that a business dependent on being crawled should stay server-rendered rather than move to a single-page app.

Angular, Angular Universal, Server-side rendering, Laravel, PHP, Service workers, Ethereum, Smart contracts, Blockchain, SEO, Git

ERPaaS · **Software Developer** Permanent, remote **Aug 2017 to Nov 2017**

Inherited an AngularJS project that had to render form inputs dynamically from whatever the server returned, over a Java Spring backend, for a business selling into clients who each wanted a different relational database underneath. Built a JSON proxy that consumed the various back-end services and gave the front end one consistent shape to work against, and moved the data layer from stored procedures onto Hibernate and JPA. Much of it was extending the Oracle platform into something the company could sell as a data platform in its own right, with functionality those products do not ship with. Jenkins for deploys, BEM for Sass, Google Material for the UI. Local development ran through a browser extension that hijacked the HTTP calls, the only way to see what a custom plugin was actually doing.

- Shipped features across the full stack of an ERP platform, from AngularJS views through to Oracle.

AngularJS, JavaScript, Java, Spring, Oracle, Hibernate, JPA, SQL, Jenkins, Sass, BEM, Material Design, Git

CF Woodford Investment Fund · **Web Developer** via Upnorth, Oxford **Jan 2017 to Aug 2017**

Built the internal dashboard at a fund run on radical transparency: what was being traded, what was held, who was paid, all of it on screens beside the trading monitors and open to everyone in the company. We were the dashboard team, alongside the data and trading teams. A Laravel REST API behind an AngularJS front end, with Pusher and WebSockets pushing straight into components so that nothing polled. Wrote a lot of the artisan commands, ran Homestead locally and deployed through Docker and Jenkins. The hard part was a SOAP service returning payloads around 800MB, which the browser had been left to download in a single call; I moved that back behind a service, chunked it, and stored it in MySQL rather than paying for the SOAP round trip each time. We looked at Redis to make it faster and decided MySQL was enough, which it was. On the front end I argued for components owning their own state and making their own calls, rather than a page hydrating everything at once, so each could be built in isolation and show its own skeleton while it loaded instead of the whole page waiting on the slowest response. On one of the developer days we built the CEO what he had been asking for, his dashboard read aloud and summarised through the Alexa API, which in 2017 was as near to that idea as anyone was getting.

- Moved an 800MB SOAP payload off the browser and behind a service, batched, queued and cached, having found the dashboard downloading it in a single call.
- Argued through the shift to components owning their own state and HTTP calls, so each loaded independently behind its own skeleton instead of the page waiting on the slowest response.
- Built the CEO his dashboard read aloud and summarised through the Alexa API, on a developer day, years before an LLM could have done it.
- Implemented fuzzy search from scratch for the dashboard's on-page multi-select, the firm having a policy against off-the-shelf libraries.

AngularJS, Laravel, PHP, C#, SOAP, SoapUI, Pusher, WebSockets, MySQL, Docker, Jenkins, Vagrant, Sass, BEM, Gulp, Node.js, Alexa API, Ionic, Git, Bitbucket, JIRA

Upnorth Creative Media · **Freelance Web Developer** Own company, Orkney **Feb 2016 to Jan 2017**

Returned to freelancing for a year after Toriga, building for businesses across Orkney: tour operators, a textile firm, and a motor dealership whose site managed its vehicle inventory, so a fair amount of it was custom software rather than brochure sites. WordPress with custom plugins for much of it, and Laravel REST APIs with AngularJS front ends against them where a client needed something built properly. Wrote scrapers in Node and PhantomJS, a React and Redux dashboard pulling stock data and the day's news, and released two iOS games written in Swift. Test-driven throughout, PHPUnit behind and Jasmine with Karma in front. The year ended with the decision to take a first contract rather than keep running an agency.

- Built vehicle inventory management for a motor dealership on custom WordPress plugins.
- Released two iOS games written in Swift.

PHP, Laravel, AngularJS, React, Redux, Node.js, PhantomJS, Swift, iOS, WordPress, Magento, PHPUnit, Jasmine, Karma, Sass, CSS, MySQL, Git

Toriga Energy · Senior Front End and iOS Developer Leeds Mar 2014 to Feb 2016

Led the front end for two years on software built for the green energy grant boom, ground source and air source heat pumps: installers photographing jobs on site, feeding a back office, and quotes going in against the grants. Users at both ends, installers in the field and back-office staff, and the customers behind them, the software going out through a partnership with Travis Perkins. C# MVC with AngularJS, Breeze and Entity Framework, the C# well built and the front end not, so I rebuilt it and took the dashboard onto Google Material, there being no designer to hand. Wrote the micro-sites and the calculators alongside it, heat loss and square footage, from the algorithms the business gave me. When an installer out on a job hit a memory limit in the Cordova WebView handling full-size site photographs, I made the case for going native and delivered an iOS app in Swift 2 before I left, carrying across what the Angular front end already did. Spent the last of it bringing two developers up on front-end practice.

- Delivered a native iOS app in Swift 2 after an installer hit a memory limit in the Cordova WebView on site, carrying across what the Angular front end already did.
- Rebuilt the front end and took the dashboard onto Google Material, there being no designer.
- Moved the API calls onto a background thread after the UI locked up, working it out from almost no documentation at the time, and chased down the memory leaks behind it.

AngularJS, C# MVC, Breeze, Entity Framework, HTML, LESS, Bootstrap, Material Design, Cordova, iOS, Swift, Cucumber, SQL, MongoDB, Git, JIRA, Concurrency, Threading

Creative Emporium · Web Developer Leeds Jul 2013 to Mar 2014

Took the first role of my career at a Leeds agency and inherited two to three hundred client sites from the front-end developer who was leaving as I arrived, along with responsibility for shipping a new CMS. Crunch from the first day. WordPress and Magento under most of the estate, and a CMS and e-commerce framework the team had written from scratch. A good deal of the job turned out to be dragging practice forward: SASS after I took a course on it, and migrations after a Rails course, the team having passed database files around by hand until that point. Built a product ingester as a custom Magento plugin. Work came straight from account managers rather than through any process, and I taught myself Linux as a daily driver by putting it on the work machine over a weekend.

- Built a product ingester as a custom Magento plugin.
- Introduced SASS and database migrations to a team that had been passing database files around by hand.

PHP, Magento, WordPress, Custom CMS, AngularJS, SASS, HTML, MySQL, Linux, LAMP, Git, Bitbucket, JIRA

Upnorth Creative Media · Web Developer and Founder Own company, Kirkwall Jan 2009 to Jul 2013

Founded it to earn a living and get real experience while I studied, building a web presence for local clubs and businesses across Orkney, every client by word of mouth. PHP, HTML, CSS and jQuery over MySQL, with e-commerce on Magento and WordPress where a client needed to take orders. Because I knew the hardware as well as the code, the work ran through to repairing whatever people carried in. Four years of it, around a degree and other work on the island, wound down in 2013 to go and learn the job properly in a professional team, though I kept the name for contracting later.

PHP, HTML, CSS, JavaScript, jQuery, MySQL, Magento, WordPress, Linux, Git

EDUCATION

BSc Computer Science, 2:1

2012 to 2013

University of the Highlands and Islands

Mobile Application Development, Developing Web Based Applications, Advanced Databases, Team Project, project leader on Android, Network Server Management, Multi User Operating Systems, Computer Forensics

HND Computing

2010 to 2011

University of the Highlands and Islands

HNC Computing

2009 to 2010

University of the Highlands and Islands

Ongoing Computer science fundamentals and applied engineering

2013 to Present

Self-directed

Structure and Interpretation of Computer Programs, The Ruby on Rails Tutorial, Michael Hartl, Stephen Grider's course catalogue, Laracasts, Frontend Masters, The Teach Yourself CS curriculum

PROJECTS

Built outside client engagements, in evenings and weekends.

X12 EDI parser

A PHP library for parsing X12 EDI, the ANSI standard that predates most of the web and still carries purchase orders, invoices and shipping manifests between businesses. Written while delivering a Shopify integration whose product data arrived in that format, and kept aside as its own library once the client feature was delivered.

PHP, X12 EDI, Parsing, Laravel, Shopify

FlipPricer

On-site pricing engine for a used-car flipper, a SwiftUI and SwiftData iOS app that values a vehicle at the point of viewing rather than back at a desk.

Swift, SwiftUI, SwiftData, iOS

Development toolchain

The tooling I run my own work through and keep sharpening: a prose linter that refuses machine-sounding review comments and commit messages before they reach GitHub, a spec workflow that turns a written specification into dispatched work and back into a record, and a pre-push gate that will not let lint, tests or an unreviewed diff past. Written in Python and Bash, driven from the shell, and the same instinct I bring to a client's estate.

Python, Bash, Git hooks, CI gates, Linters, Automation

Mentoring and tutoring

Have taken on developers outside the day contracts and taught them properly: structured study blocks, assignments set and marked, and pair programming through the parts that do not land from reading. Three so far, alongside the juniors mentored inside engagements, one of whom later brought me in to rescue a project of his own.

Mentoring, Tutoring, Pair programming, Curriculum

Preflight

A code review gate that runs before anything reaches GitHub. Two independent model reviewers read the branch diff, findings are merged and applied, the diff is re-reviewed, and the push is refused while blockers remain. Written in Python, installed on PATH, and used on every branch I push.

Python, Static analysis, CI gates, LLM orchestration

HAL, Holistic Automation Layer

A Rust automation layer for the homelab, written to work in the language properly rather than read about it, alongside a data structures and algorithms workbook in the same.

Rust, Automation, Data structures, Algorithms

SBG UK BJJ

Site and membership platform for a Brazilian jiu-jitsu academy, Laravel with Inertia, and a React Native app alongside it.

Laravel, Inertia, React, React Native, PHP

GymTribe

Product for gyms and their members, built and run end to end.

React Native, Expo, TypeScript, Node.js

Bayline

Product built and run end to end.

TypeScript, React, Node.js

Anvil

Codebase health analysis, transformation, and enforcement CLI. Reports on the state of a codebase, applies codemods, and enforces the standards it finds so drift gets caught in CI rather than in review.

TypeScript, Node.js, AST, CLI, Codemods

Homelab and GitOps infrastructure

Self-hosted infrastructure managed declaratively with Ansible and Terraform: Proxmox, Docker, Kubernetes, Caddy, Tailscale, Authentik single sign-on, and Pi-hole with Unbound. Everything is administered from the shell, backed up, and changed through version control rather than by hand, and it has been running and evolving continuously since 2023.

Linux, Ansible, Terraform, Proxmox, Kubernetes, Docker, Caddy, Tailscale, Authentik, systemd, Bash, DNS, Backups

Private multimodal LLM stack

Self-hosted private multimodal chat stack running entirely on local hardware, so no prompt or image ever leaves the house. Containerised Python services behind a web front end.

Python, Docker, LLM, Self-hosted

Username finder

A Python script that worked through the space of three-letter usernames, found the ones nobody held, and let me pick the one that fitted. It is why my GitHub handle is what it is.

Python, Scripting, APIs