

IBEANUSI DAVID UCHECHUKWU

Full-Stack Web Developer

✉ davidibeanusi203@gmail.com ☎ 09049519049 📍 Lagos, Nigeria 🔗 stacklorecode.vercel.app
🌐 linkedin.com/in/david-ibeanusi

SUMMARY

Full-stack developer with over 4 years of experience building and shipping applications using React, Node.js, and Express, along with 2+ years of experience developing .NET applications. I'm comfortable learning new technologies when a project requires them, but I believe in choosing the right tool for the job rather than adding technologies just to fill a résumé. Every technology I use is a deliberate choice, and I can explain the reasoning and trade-offs behind it.

WORK EXPERIENCE

Backend Engineer, Orchestrator System <i>AVI Wavy (currently closed)</i>	06/2025 – Present Lagos State, Nigeria
- Contributed to a microservice orchestration layer, implementing service-to-service communication logic and API contracts for a subset of the platform's backend services. - Worked within a small backend team, writing and reviewing code to keep service integrations consistent with the platform's data-exchange standards.	
Full-Stack Developer Intern (SIWES) <i>Rygma Solutions</i>	07/2025 – 09/2025 Lagos State, Nigeria
- Built backend features for an internal Central Customer Management System (Node.js/TypeScript, PostgreSQL) that aggregates customer records from three internal SaaS products into a single admin-facing view, working from a written functional spec. - Implemented authentication and role-based authorization for the API, and applied PostgreSQL row level security policies to enforce access control at the database layer. - Containerized the application with Docker and Docker Compose, and resolved inter-container networking and port-exposure issues to get the frontend, API, and database communicating correctly. - Wrote unit tests for backend endpoints with Jest and applied layered-architecture and SOLID principles to the project structure.	
Full-Stack Developer <i>Open Source & Freelance</i>	03/2025 – Present Lagos State, Nigeria
- Shipped responsive, production frontends (Vite, React, Tailwind CSS) for client and open-source projects, including a live e-commerce storefront (Koyah Brand) wired to REST APIs for checkout and real-time product updates. - Built a RESTful e-commerce API (Node.js, Express, PostgreSQL) with role-based auth and a normalized schema, then integrated it with a frontend mockup to demonstrate full request-to-render flow. - Delivered CRUD task-management tooling with a focus on reusable components and documented, maintainable code, reviewed as part of a small open-source team.	

EDUCATION

B.Sc. Computer Science <i>Federal University Oye-Ekiti</i>	2026
Certifications: AWS Practical Training Nextwork (2026); The Complete 2024 Web Development Bootcamp at Udemy	

CORE SKILLS

Languages & Frameworks

TypeScript, JavaScript, C#, React, Node.js, Express.js, Fastify, .NET / ASP.NET Core

Data & Cloud

PostgreSQL, MongoDB/Mongoose, AWS (EC2, S3, IAM, Lambda), Git, Docker

Practices

Dependency injection, event-driven architecture, AST-based static analysis, REST API design

Soft Skills

Communication, problem-solving, teamwork, adaptability, attention to detail

PROJECTS

SmartApiAnalyzer(Automated API Security & Performance Scanner)

06/2026 – Present

.NET (C#), ASP.NET Core, React/Vite | Personal Project

- Designed a multi-project .NET solution that ingests a target API, validates and discovers its routes, then runs a configurable, priority-ordered pipeline of agents (security, latency, cost, credentials) and halts automatically the moment a threat check fails.
- Chose C# for the analysis engine because the pipeline performed computationally intensive analysis before feeding results into machine-learning scoring. C# and the .NET runtime provided the execution efficiency needed to process these workloads without making the analysis stage a bottleneck.
- Used parallel processing across independent analysis agents to improve overall throughput by allowing multiple agents to process data concurrently rather than sequentially. This significantly reduced processing time, but introduced a memory trade-off, as running multiple agents simultaneously required more memory to maintain their workloads and intermediate results.
- Built an event-driven architecture (in-memory event bus/queue, background worker) that decouples ingestion from agent execution, so each scan runs asynchronously in its own scoped session instead of blocking the API.
- Paired the backend with a React/Vite frontend so results from each agent stream back to the user as the pipeline runs, rather than only after the full scan completes.

Log Complexity (AI Code Complexity Analyzer)

01/2026 – Present

Full-stack TypeScript (React 19, Vite, Fastify 5) | Personal Project

- Built a full-stack tool that parses source code into an AST and scores its complexity using a locally hosted LLM (node-llama-cpp), removing the need for a paid external API and keeping analysis on device.
- Used TypeScript end-to-end and ts-morph for AST parsing specifically because JavaScript/TypeScript already has a mature, battle-tested parsing ecosystem (Tree-sitter/ts morph) letting me ship a working analyzer instead of writing a parser from scratch for a language I don't know well.
- Designed the backend with a DI container and NodeNext module resolution, supporting batch uploads of up to 300 files (200KB each) through Fastify's multipart plugin.
- Delivered a working JavaScript/TypeScript analysis pipeline end-to-end, with the compiler layer architected so additional language compilers (C++, Java, Rust) can be plugged in without touching the core engine.

ACADEMIC PROJECTS

Academic Project: RESTful API for E-Commerce Platform

02/2024 – 06/2024

Node.js, Express, PostgreSQL | Team Project, Federal University Oye-Ekiti

Ekiti State, Nigeria

- Built a RESTful API simulating core e-commerce functionality product catalog management, user authentication, and order processing as part of a university team project.
- Implemented secured endpoints with basic authentication and role-based access logic.
- Designed a normalized relational database schema with optimized queries for efficient data retrieval.
- Integrated the API with a frontend mockup to demonstrate end-to-end client-server interaction.