

MD SAMIUL ALAM SUMEL

Web Developer — Business Systems & Workflow Automation

Mongla, Bangladesh | sa.sumel91@gmail.com | +880 1312 312 512 | sasumel.pages.dev | github.com/samiulAsumel | linkedin.com/in/samiul-alam-sumel

PROFESSIONAL SUMMARY

Web developer who builds business systems and internal tools, with 12+ years of hands-on experience in cargo billing, vehicle and cargo tracking, revenue reporting and operational workflows at Mongla Port Authority (Traffic Department). Replaces manual and Excel processes with working web systems: three self-built tools (Port Billing, CarView and OT Bill) are in daily use at the billing desk and in the Traffic Department. Builds with AI-assisted development (Claude Code); requirements, business rules, testing and validation stay with me. Open to international and remote roles in web development for business systems and internal tools, process automation, operations technology or product building.

TECHNICAL SKILLS

Domain & Port Operations: Port billing, wharf-rent / VAT / levy calculation, advance billing estimates, cargo and vehicle tracking, container tallying and statistics, revenue reporting, C&F agent workflows, customs-related reporting

Proven in live projects: HTML5, CSS3, JavaScript (ES6+), PWA / offline-first, Cloudflare Pages & Workers, Firebase Auth & Firestore, Chart.js & Excel export, automated tests, Git & GitHub, React & Next.js, AI-assisted development (Claude Code)

Working knowledge: TypeScript, Tailwind CSS & Vite, Hono & Zod APIs, SQL on Cloudflare D1, database design, Vitest & CI/CD (used in OpsFlow, in development)

Currently developing: Node.js & Express depth, automated testing depth, TypeScript depth

Self-directed practice: Linux / Ubuntu, Bash scripting, systemd & cron / timers, monitoring scripts (public repos on GitHub)

PROFESSIONAL EXPERIENCE

Senior Outdoor Assistant (Revenue Branch) — Traffic Department

Nov 2017 – Present

Mongla Port Authority — Bangladesh's second-largest international seaport

Core Port Operations

- Billing: wharf-rent billing (slab-wise charges, VAT and levy, inside/outside cargo split), vehicle, general-cargo and stuffing, delivery and shipping-related billing, demurrage, advance billing estimates and billing records, applying the port's published tariff.
- Reporting: daily and monthly revenue reports, cumulative income statements and monthly Traffic Department reporting for port authority management; container, cargo, vehicle and jetty statistics; Excel-based data preparation and record checking.
- Operations records: container tallying, vehicle delivery and position tracking across warehouse, shed and yard, customs-related vehicle (auction) reporting, and the bank-deposit (challan) workflow with monthly statements.
- Coordination: C&F agents (paperwork, billing queries, advance payment planning), shipping agents and stevedoring stakeholders.

From manual work to software

- Moved from manual work to Excel automation, then to internal tools, web apps and business systems: Port Billing System, CarView and OT Bill Automation, each built around the real workflow after talking to the people who use it, and each in daily use.
- Builds with AI-assisted development as a productivity tool; the requirements, business rules, review and testing stay with me.
- Maintains an active public GitHub portfolio (github.com/samiulAsumel) and a self-directed practice in Linux and Bash.

Junior Outdoor Assistant — Port & Terminal Operations

Nov 2013 – Nov 2017

Mongla Port Authority

- Assisted in port inspections, vessel operations, and cargo handling under senior supervision; monitored cargo dwell time and vehicle positioning across terminal locations.
- Built foundational expertise in wharfrent billing, maritime logistics, and customs clearance that underpins all later self-initiated digital work.

FEATURED PROJECTS

Port Billing System — [Live](#) | [Case study](#)

Problem: C&F agents, importers and shipping agents kept asking what a bill would be. The port's billing system makes permanent entries, so they could not check wharf rent before committing, and revenue staff worked each estimate out by hand or in Excel, causing billing disputes, longer cargo dwell time and repeat counter visits.

Advance bill calculator that applies the port's rules in a fixed order: slab-based wharf rent, river dues, landing, removal, weighment and hoisting added to the base, VAT on the base, then levy; inside/outside storage totalled separately. Saved bills with search, a dashboard and a print-ready A4 invoice. The rules run in the browser, so estimates work offline; slab, VAT and rounding rules are covered by automated tests and were compared against real bills. Used daily at the billing counter.

Tech: JavaScript, HTML/CSS, Cloudflare Pages & Worker, PWA, Automated Tests, A4 Print

Vehicle Tracking / CarView — [Case study](#)

Problem: Vehicle positions across warehouse, shed and yard were recorded in one person's Excel file, invisible to everyone else. When that person was unavailable, the whole team lost visibility.

Offline-first installable web app tracking eight storage locations, with daily receipts, deliveries, transfers and running balances; a Cloudflare Worker writes to a private GitHub repo so every save is a version-controlled commit. Seven charts, thirteen report sections and Excel export. Built for the Traffic Department and used daily. It records quantities and movements; it is not GPS tracking.

Tech: PWA, Cloudflare Worker, Chart.js, GitHub-backed data, Excel Export

OT Bill Management / Automation — [Case study](#)

Problem: Overtime billing was a repetitive manual calculation: find the employee's pay, work out an hourly rate, add up the hours day by day, then produce the bill in the required format. Errors were easy to make and hard to spot.

Staff enter an employee profile once; the system prepares the overtime bill from daily hours, applies the rate rule automatically (basic pay divided by 104, rounded up) and prints a bill on A4, with an all-bills view and a dashboard. It is overtime billing, not payroll.

Tech: JavaScript, Employee records, Auto-calculation, A4 Print, Cloudflare Pages

OpsFlow (in development) — [Early build](#) | [Case study](#)

Problem: Transport and logistics firms run orders, trips, invoices and payments across Excel, WhatsApp and manual records.

Multi-tenant operations software, built in stages. Implemented: tenant-isolated accounts and roles, customer, route and rate-card data, fleet records and quotations. In development: orders and trips. Planned: invoices, payments, outstanding balances and reports. Private repository; an early build, not a live product.

Tech: TypeScript, React, Vite, Hono, Zod, Cloudflare Workers + D1, Vitest, GitHub Actions

Client Intake Form — [Live](#) | [GitHub](#)

Problem: Collecting project requirements from a client over email or chat is unstructured: details get missed, and without a written scope, what was agreed becomes a dispute later.

Project requirement intake form: 18 structured sections that adapt to the project type, a serverless submission API, and a password-protected admin dashboard that tracks each lead from New to Won/Lost, with the client's own email app as a fallback.

Tech: JavaScript, HTML/CSS, Cloudflare Pages Functions, GitHub API

SalahSync — [Live](#) | [GitHub](#)

Problem: A printed daily-routine sheet couldn't adapt when Fajr time or office hours shifted, or record whether a prayer was made in jamaah, alone, or missed.

Offline-first installable PWA: set Fajr time and office hours once and the day's schedule recalculates, with a prayer tracker, streaks and offline solar prayer-time calculation (no API). Personal daily-use tool, bilingual UI.

Tech: PWA, Offline-First, Vanilla JS, i18n

World Kitchen Atlas — [Live](#) | [GitHub](#)

Problem: Most recipe content online is thin, and states uncertain food history as settled fact.

Next.js static-export culinary encyclopedia on Cloudflare Pages: continent -> country -> dish browsing, client-side search, and a per-dish confidence rating. A Cloudflare Worker proxies a private GitHub data repo, with admin CRUD and a Durable-Object-backed visit counter. 295 dish pages across 7 Asian countries; other continents planned.

Tech: Next.js, TypeScript, Cloudflare Worker, Durable Objects, Static Export

English Mastery — [Live](#) | [GitHub](#)

Problem: English-reading material online has no level structure, so a learner can't tell where they are or what to read next.

Content-driven learning platform: 88 reading lessons across nine skill levels with 87 paired writing tasks and 87 speaking drills. A Node build script validates every Markdown lesson against a schema and generates the static site, so adding a lesson is a one-file change. GitHub Actions CI, auto-deployed to Cloudflare Pages.

Tech: Node.js, Vanilla JS, Static Site Generation, GitHub Actions, Cloudflare Pages

Outreach Copilot — [Live](#) | [GitHub](#)

Single-user AI-drafted outreach tool for a hand-picked lead list: Cloudflare Pages + Workers API, D1-backed leads and analytics, Workers AI drafts; every send stays a manual, deliberate step.

Anchora Commerce (source only, portfolio build, not a live shop) — [GitHub](#)

Full-stack commerce platform: storefront with cart and checkout (stock validated in a transaction) and a role-gated admin dashboard. Next.js 15, TypeScript, Prisma, PostgreSQL, Docker, Vitest and Playwright tests, GitHub Actions.

JARVIS (source available, no public demo) — [GitHub](#)

Self-hosted personal AI assistant with multi-model routing, voice I/O and persistent memory.

SELF-DIRECTED STUDY

The Odin Project & Full Stack Open

Aug 2026 – Present

Self-Directed — In Progress

Self-paced study for code comprehension, not a certificate: The Odin Project, then Full Stack Open (University of Helsinki) for JavaScript, Node.js/Express depth, testing and Git workflow.

EDUCATION & LANGUAGES

Higher Secondary Certificate — Business Studies: Khulna Public College, 2009 – 2011

Secondary School Certificate — Business Studies: Mongla Bandar Secondary School, 1999 – 2009

Languages: Bengali — Native | English — Professional