ARUM MOHEE

AI Engineer | Enterprise Transformation, Agent Engineering & Product Delivery

Manchester, UK • moheearum7@gmail.com • Portfolio available online

PROFILE

I help enterprises and agencies turn AI ambition into an operating capability.

My work spans workflow discovery, context and agent architecture, software engineering, multimodal production, evaluation and adoption. I also work as an embedded or subcontracted technical partner for consultancies adding AI capability to their existing services.

01 / Core mandate

Enterprise AI transformation
Enter the operation, audit workflows, identify leverage and define a credible operating model and delivery roadmap.

Agent & context engineering
Design role boundaries, memory, retrieval, state, tools, evidence gates and human approval.

Product engineering
Build native, web and realtime interfaces with TypeScript, React, Next.js, Node.js, Swift and Electron.

Partnership & adoption
Embed with internal teams or partner agencies, test failure modes, train users and improve capability through live use.

02 / Current professional practice

AI Systems & Operations Engineer | iScalable CURRENT

Own the design and implementation of an agent-supported operating layer for marketing delivery across research, creative, approval, build, reporting and organisational memory.

- Mapped recurring campaign delivery into six explicit stages with state, ownership, evidence requirements and escalation paths.
- Designed specialist delivery, growth and research agents connected to Slack, Notion, Figma and Klaviyo context.
- Built role prompts, scoped memory, decision logs, correction memory, scheduled processes and human-review controls.
- Disabled early proactive behaviour when testing showed unsupported check-ins; added source-backed evidence gates before consequential actions.
- Use Claude Code and OpenAI Codex as engineering environments across planning, implementation, debugging, review and visual QA.

Commercial context: work spanning 50+ brands across agency, ecommerce and campaign operations.

03 / Selected systems and engineering evidence

Marker & Mara | Realtime meeting and sales intelligence DEPLOYED WEB / DESKTOP BUILT

A meeting environment where captions, transcript memory, live coaching, shared-screen annotation and private AI support operate inside the room rather than after the call.

- Built realtime rooms with LiveKit, Deepgram Nova-3 captions, speaker-labelled transcripts, talk-share analysis and Electron screen annotation.
- Mara reasons over the current call using Gemini or Claude, with local deterministic fallback logic when model credentials are unavailable.
- Sales mode detects objection language, commitments and buying signals; surfaces discreet response guidance, next steps and structured post-call notes.
- Stack: Next.js 16, React 19, TypeScript, LiveKit, Deepgram, Anthropic, Gemini and Electron.

Gobby | Private local dictation for macOS SHIPPED UTILITY

Hold the left Option key, speak and release. Audio is processed locally and the transcript is inserted at the active cursor.

- Engineered a resident whisper.cpp server to avoid model reload and achieve documented warm inference of approximately 0.05-0.07 seconds.
- Implemented waveform feedback, silence guards, personal dictionary correction, clipboard-safe insertion and stable macOS permissions.
- Packaged for Apple Silicon and Intel using Swift and AppKit.

SnapMessage | View-once media inside iMessage FIELD-TESTED ON TWO IPHONES

A native iMessage extension for taking or selecting a photo, adding a caption and sending a time-limited snap within the conversation.

- Built the in-extension camera and compose flow with SwiftUI and AVFoundation.
- Implemented client-side payload handling with CryptoKit/AES-GCM and tested the complete send/open flow across two physical devices.
- Project test evidence reports 21 passing unit tests across payload, image codec and store behaviour.

AppForge & Product Listing Studio | AI-assisted build and media operations WORKING INTERNAL WORKFLOWS

Two examples of AI used as production infrastructure rather than a standalone prompt.

- AppForge structures research, planning, build execution, logs and completion state around isolated software ideas; SnapMessage originated through this pipeline before manual iteration.
- Product Listing Studio routes ordinary product references through isolation, background reconstruction, relighting, scene generation, product-accuracy review and export.
- Creative workflow combines Higgsfield, GPT Image, Nano Banana, Seedance, Kling, Veo evaluation, Photoshop and DaVinci Resolve.

04 / Technical capability

Agentic engineering

Claude Code, OpenAI Codex, tool use, structured outputs, role/state design, RAG, scoped memory, decision logs, scheduled processes, evaluation gates

LLMs & voice

Anthropic, OpenAI APIs, Gemini, Gemini Live, Deepgram Nova-3, whisper.cpp, transcript-grounded reasoning, deterministic fallbacks

Software

TypeScript, Node.js, React, Next.js, Python, Swift, SwiftUI, AppKit, Electron, REST APIs, webhooks, Vercel

Realtime & security

LiveKit, AES-GCM/CryptoKit, short-lived credentials, expiring objects, local-first processing, device and regression testing

Agency & commerce

Slack, Notion, Figma, Klaviyo, Shopify, TikTok Shop, Amazon FBA, campaign state, creative hand-off, reporting

Generative production

Higgsfield Cinema Studio, Seedance, Kling, Veo, GPT Image, Nano Banana, reference systems, continuity, product locking, DaVinci Resolve, Adobe

05 / Education and operating background

BSc Computer Science - 2:1 | **Staffordshire University** GRADUATED 2024

Software design, databases, security, development methods and the discipline to explain and test technical decisions.

Business & Computer Science | **College** COMPLETED

Business qualification D*D* and A-Level Computer Science grade C; early combination of commercial and technical thinking.

Ecommerce & campaign operations | **Independent and agency work** 2021-PRESENT

Hands-on work across Amazon FBA, Shopify, TikTok Shop, product imagery, listings, storefronts, retention, creative testing and client delivery.

06 / How I work

- Start with the workflow, commercial constraint and accountable human - not with a model looking for a use case.
- Use models for interpretation and generation; use deterministic software for rules, state, permissions and validation.
- Make system state visible, keep consequential actions reviewable and treat failed assumptions as architecture feedback.
- Stay current through continuous model, interface and competitor research, then test new capability against real work before adopting it.

References and further technical evidence available on request.