Bits to Banking

Master Guide — From Bits → Banking → Calypso

# Dedication and Acknowledgements

First and foremost, all thanks are due to Allah, who blessed us with knowledge, patience, and strength to pursue this project. I thank my parents, my late father who always encouraged me to seek knowledge, and my brothers and sisters who supported me throughout the years.

I extend heartfelt thanks to my family, mentors, friends, and colleagues who shaped my journey of learning and service. To my dear friend Mujahid Sufian and his wife Dr. Alefiah Mubeen, steadfast supporters of the Umicom Foundation and now contributors to this book, bringing deep expertise in Machine Learning and Data Science. To Dr. Ahmed El Banaa, physiotherapy specialist at the Military Hospital in Egypt, who worked tirelessly with Palestinian patients, and has become a close friend and brother. To Islam Mahmoud, who despite injury continued designing, documenting and supporting aid distribution in Gaza. To Mohammed Al Danaf and his family for their bravery and sacrifice in delivering aid and supporting through their business network in Gaza. To Khaled Matter, a mobile developer from Gaza, who even while injured volunteered to share knowledge through his phone. To Mohammed Soliman, always rushing to support financially and contributing his experience. To Saheesh Rafeeque, business partner and founder of Eastern Bridge, for his backing and logistics expertise.

I also extend heartfelt thanks to Tarneem Elyan, an Artificial Intelligence and Data Engineer from North Gaza, who despite the hardships of war has volunteered her time to distribute humanitarian aid in her community and to contribute her knowledge to this book. Her courage and expertise in data science and AI bring hope and practical value to our shared mission.

Finally, to all the brave men and women who risked their lives delivering humanitarian aid, many martyred or injured, and to the doctors serving Palestinians in Egypt, and to every software engineer, reviewer, and student who contributes to this vision—this book is dedicated to you.

# How to Use This Series

• ≤ 4 chapters per volume. Small, printable, and easy to study.

• Kid → Teen → Adult. We begin with everyday analogies, then level up with code and labs.

• Smooth transitions. Each volume ends with a 1-page refresh and “what’s next.”

• Code in boxes; simple diagrams. Monospace, high contrast, print-friendly.

# Roadmap at a Glance

• Vol. 1 — Foundations of Computing: bits, gates, Boolean, binary & hex, CPU, ISA (x86/ARM/RISC-V), assembly → C → C++ → Java.

• Vol. 2 — Operating Systems & Linux From Scratch: what an OS does, kernel vs userland, processes, memory, filesystems, network; build a tiny Linux.

• Vol. 3 — Databases & Data Engineering: PostgreSQL/Oracle, indexing/transactions, NoSQL, warehousing, ETL, data quality.

• Vol. 4 — Networking & Protocols: TCP/UDP, HTTP/REST, TLS; FIX, SWIFT ISO 20022, FpML; queues and messaging.

• Vol. 5 — Cybersecurity for Builders: threats, OWASP, memory bugs, secrets/KMS, secure deployment, audit/compliance.

• Vol. 6 — Programming Deep Dives: C/C++, Java (JVM, concurrency), Python, Rust, Zig, PHP; patterns, testing, packaging, web & mobile overviews.

• Vol. 7 — Finance & Treasury Primer: markets 101, asset classes, front/middle/back office, PnL, risk, settlement, clearing; history of money → digital.

• Vols. 8–12 — Calypso Series (multiple slim books): Fast-Track; Architecture & Core; Risk Compute; Position/Liquidity/KPI; Market Data/Quotes; Integration/Messaging.

• Vol. 13 — Customization & Extensions: SPIs, adapters, scheduled tasks, UI add-ons, reports; multi-language examples.

• Vol. 14 — Testing, Deployment & Ops: CATT, perf/load, observability, HA/DR, upgrades, blue/green, rollback.

• Vol. 15 — RISC-V & Advanced Computing: assembly → Linux on RISC-V, QEMU/WSL2 labs, cross-compiling, tiny compiler projects.

• Vol. 16 — Projects & Capstones: commodity exchange, privacy-first video platform, social + fundraising site, UmiCOIN (asset-backed).

# Mini-Introductions (Sample)

## Volume 1 — Foundations of Computing

What you’ll get: the “why” behind every 1 and 0. Play with binary using fingers, build AND/OR with paper switches, watch a CPU “fetch-decode-execute,” then translate simple programs from Python → C → assembly.

You’ll be able to:

1) Explain binary and overflow to a kid.

2) Write a 10-line program and peek at its assembly.

3) Connect CPU cycles to how a trade instruction is processed.

## Volume 2 — Operating Systems & Linux From Scratch

We explain processes, memory, files, users, then compile a tiny Linux. You’ll boot your own system and run basic tools.

You’ll be able to:

1) Read process lists and memory usage.

2) Understand system services (systemd) and logs.

3) Prepare a finance-friendly Linux profile.

# Video Micro-Lessons (Titles & Descriptions)

• Bits vs Switches (3 min) — A light switch becomes a 1 or 0; two switches make AND/OR.

• Binary Counting with Fingers (5 min) — Count to 31 on one hand; meet overflow.

• From CPU to Instructions (6 min) — Fetch, decode, execute; registers are sticky notes.

• Hello, Assembly (7 min) — Add two numbers with registers & memory moves.

• C to Machine Code (8 min) — Compile a tiny C program and compare assembly.

• Processes & Memory (6 min) — Stacks/heaps/syscalls; why programs crash.

• Build a Tiny Linux (8 min) — Compile + boot a minimal system; first prompt!

• Your First REST Endpoint (5 min) — Create /hello; add a nightly job.

# Lightweight Diagrams (Printable ASCII)

[CPU]--fetch-->[Instruction]--decode-->[Execute]  
 | |  
 [Registers] <-------------------- [ALU]

Process  
 ├─ Code (read-only)  
 ├─ Heap (grows up) <-- new objects  
 └─ Stack (grows down) <-- function calls

REST Control Plane  
Client -> Risk Services -> Risk Server  
 \\_\_ health/status \\_\_ MQ + compute

# On-Ramp Labs (Quick Wins)

• Build & run a one-file program in C, C++, Java, Python.

• Boot a tiny Linux VM; list processes & services.

• Create /hello in Java; schedule a nightly job; check logs & metrics.

• Create a DB table, insert a row, read it back.

# Glossary (Short & Sweet)

• Bit — Tiniest piece of info: 0 or 1.

• CPU — The ‘thinker’ that follows instructions fast.

• Register — A tiny ultra-fast box inside the CPU.

• Process — A running program with its own memory world.

• Message Queue — A mailbox for servers to pass work.

• Latency — How long something takes (track p95/p99).

# Where to Go Next

Start with Volume 1 — Foundations of Computing, then Volume 2 — Operating Systems & Linux From Scratch. When ready, enter the Calypso volumes for banking-grade platforms.