Where to go from here

This course stopped on purpose before a few topics. Each one has a book.

Unsafe Rust and calling into C are the Rustonomicon. The shop API, the CLI, and the WebAssembly page are safe Rust. You can ship those without `unsafe`.

Macros start with declarative `macro_rules`. The Book's macros chapter teaches that form first. Procedural macros come after it. You have already used derive macros without writing one.

The course did not teach sockets. `std::net` is the standard library's next page, and rustls is the TLS library people reach for.

Embedded Rust assumes a microcontroller. The Embedded Book is the introduction. This course assumes a normal operating system.

The deploy this course teaches is Running in Docker: one image for the API, and Postgres beside it. Past that image, a release build uses the release profile. Registries and orchestrators are outside this course.

criterion is the benchmarking harness for a hot function. Bringing Rust to work asked you to measure a whole program on the same input. criterion is the next step when one function is the question. Testing tools already showed a `cargo bench` sketch.

Four references stay useful after the course:

- The Book
- The Async Book
- Rust by Example
- This Week in Rust

Two books that fit after this course: Zero To Production in Rust (Axum and API-focused, the natural sequel to Building an API), and Rust for Rustaceans (intermediate habits once the basics stick).

You came in writing PHP. You can read Rust, and you can put one small tool next to an application you already run. That is the course. The next tool is yours.

> **Watch for:** Opening with `unsafe` because a post did. Stay in safe Rust until a measured problem needs a boundary.

> **Idiomatic Rust:** Pick the book for the topic you actually need next. Declarative macros before procedural ones. A release profile before a new orchestrator.

Side by side

Topic this course skipped	Where to read it
Unsafe and FFI	The Rustonomicon
Macros	The Book, declarative first
Networking and TLS	`std::net` , rustls
Embedded	The Embedded Book
Benchmarks	criterion

Key takeaways

- Unsafe, macros, networking, embedded, and benchmarks each have a named next book.
- The Docker lesson is the deploy this course taught.
- The next project is a small tool on your own code.