Zwe Khant Aung (meowveloper)

> TypeScript, Zig, Nix, Linux
>
> Full Stack Developer & Systems Enthusiast
>
> Bangkok, Thailand | +66822918935 | meowveloper.dev@gmail.com
>
> Github: meowveloper

About Me

Full Stack Engineer with a Systems Mindset I build high-level web applications with **NuxtJS** and **TypeScript**, while exploring low-level performance with Zig. I do not just use frameworks; I care about how software interacts with the OS. Whether I'm optimizing SEO for an EdTech marketplace or managing memory in a custom CLI tool, my goal is always the same: speed, type-safety, and absolute reproducibility. Even though I am currently focusing on system programming, I can also handle wherever the typescript touches.

Professional Experience

Frontend Consultant (Arsari)

(April 2024 - Present)

- Frontend Architecture: Leading the development of the core user interface using **NuxtJS** and **VueJS**, focusing on scalability and user experience.
- Product Iteration: Translating high-level business goals into functional technical requirements in an agile, fast-paced environment.
- Scalability at best, type safe components, custom hooks, composables and improve SEO
- Centralized state management with NuxtJS's built-in composables
- Improve maintainability with **functional programming** such as **pure functions** and **immutable data**
- Improve performance by **SSR** and NuxtJS's best practice caching
- Internationalization (i18n): Developed a seamless multi-language toggle (English/Burmese) using `@nuxtjs/i18n`.
- Official website link: https://arsari.app

Projects

Password Strength Auditor (released v1.1.1) (psa)

- PSA is a high-performance, cross-platform command-line tool designed for auditing password strength. Written in Zig(0.15.2), it leverages **manual memory management** and **zero-allocation optimization** to deliver maximum speed.
- Features:
  - **Dictionary** Attack: High-speed wordlist checking with optimized file I/O.
  - **Brute-Force** Attack: Recursive character combination generation with interactive configuration.
  - Audit Mode: Batch processing of multiple hashes with automated security scoring and reporting.
  - **Cross-Platform**: Compiles natively for Linux, Windows, and macOS.
  - Zero Dependencies: Uses Zig's standard library for all cryptography (MD5) and I/O.
- github link: https://github.com/meowveloper/psa.git

meowmux (released v1.1.1) (meowmux)

- Meowmux is a lightweight, TUI-based CLI project manager written in Zig (0.15.2). It simplifies workflow by allowing you to quickly manage and switch between projects, automatically launching them in tmux sessions.
- Features:
  - Interactive **TUI**: Navigate your project list with ease using **Vim-style** keys (j/k) or arrow keys implemented with **self-written** terminal cleaning and UI logic **without any external libraries**.
  - Project Management: Add, Edit, and Delete projects directly from the interface.
  - Tmux Integration: **Automatically creates or attaches** to a named tmux session for the selected project.
  - Path Autocomplete: Supports tab-completion for file paths when adding or editing projects with **self-written** file system iterating and search logic **without any external libraries**.
  - Config persistence: Safely stores your project list in `~/.config/meowmux/projects.json`.
- github link: https://github.com/meowveloper/meowmux.git

meowkey (still in progress)

- Meowkey is a lightweight background process for Linux, written in **Zig (0.15.2)**, designed to generate mechanical keyboard sounds in real-time as you type.
- Core Objectives:
  - **Zero Latency:** Minimize the delay between physical key presses and audio feedback.
  - **Global Input Capture:** Utilize the Linux Input Subsystem (`/dev/input/event*`) to intercept keystrokes regardless of which window is active.
  - **Low Resource Usage:** Leverage Zig's efficiency to ensure the process remains unnoticeable in the background.

    - **ALSA Integration:** Direct interfacing with the Advanced Linux Sound Architecture for high-performance audio playback.
- Technical Architecture:
    1. **Input Listener**: The application will read from one or more `/dev/input/event*` devices. It filters for `EV_KEY` events to detect `KEY_DOWN` transitions.
    2. **Audio Engine**: A low-latency buffer management system using ALSA.

DevOps & Developer Experience

- **Reproducible Environments:** Expert in **Nix/NixOS** (daily driver) and **Flakes** for creating zero-latency development shells (`nix develop`).
- **Environment Parity:** Replaced bloated Docker-based local workflows with native, deterministic Nix environments, reducing build times by 30% and local resource overhead.
- **Infrastructure as Code:** Declaratively manage system-wide configurations, ensuring consistent setups across multiple machines.

> markdown version of this CV can be found here.