# Training Day-28 report

**24 July, 2025**

Today's session focused on exploring the **internal workings of Node.js** and gaining hands-on understanding of building **server-side applications without the use of frameworks** like Express. The goal was to deepen knowledge of Node.js fundamentals and low-level server handling.

**Work Undertaken:**

1. **Understanding Node.js Internal Architecture:**
   • Studied the **execution flow of a Node.js program**, which follows this order:
   – Execution of **top-level code**, including require statements and event registrations.
   – Offloading of **CPU-intensive operations** to the **libuv-managed thread pool** to prevent blocking the event loop.
   – Initiation of the **event loop**, which continues running until all tasks are completed.

• Learned the key **phases of the event loop**:
– **Timer callbacks** (executed after their delay expires)
– **I/O polling phase** (checks for readiness of IO operations)
– **Immediate callbacks** (scheduled with setImmediate)
• Understood that this cycle continues until the **call stack, event queue, and internal resources are all clear**.

2. **Building a Server Without Express:**
   • Practiced creating a **Node.js HTTP server** from scratch using the core http and url modules.
   • Manually handled:
   – **Setting response headers**
   – **Reading and streaming files**
   – **Parsing incoming URL strings** and converting them into objects using url.parse()
   – **Extracting query parameters** for routing and custom response logic.

**Key-Learnings:**
• Gained a deeper understanding of the **Node.js event loop** and how it manages asynchronous operations.efficiently.
• Learned how **libuv** delegates blocking tasks to background threads for performance optimization.
• Improved foundational skills in **writing raw server-side code**, building a better appreciation for what frameworks like Express abstract away.

**Conclusion:**
Today's session provided valuable insight into **low-level Node.js behavior** and helped reinforce core concepts like event-driven architecture and manual server creation. These concepts are crucial for debugging, optimization, and understanding the internals of higher-level frameworks.