# Module XX – Mock Runtime Execution Layer

Version: 4.1 | Classification: CTO Mandatory (Executable Simulation Layer)

Scope: Capsule evaluation harness, containerized test runtime, CLI sandbox interface

## 0. Purpose & Simulation Objective

This module provides a capsule-compliant simulation runtime for MaxOneOpen v4.1. It allows engineering teams, auditors, or governance analysts to validate capsule behavior without a full implementation. The runtime mimics capsule chain execution, fork reactions, and alert emissions using containerized mocks and test CLI endpoints.

## 1. Architecture Overview

- Built on Docker or WASM sandbox engine  
- CLI-driven simulation controller (`capsule-runner`)  
- Local volume-backed ledger trace  
- Signature bypass for simulation only (marked as mock)  
- Configurable capsule injector and audit signal emitter

## 2. Capsule Execution Hooks

Supported mock capsule types:  
- Manifest Capsule (inject & bind)  
- Execution Capsule (with policy scope)  
- Replay Capsule (triggers RAC & freeze)  
- Alert Capsule (emits TVC, PCC, RAC)  
- Federation Capsule (mock FRC, FCC)  
- Recovery Capsule (simulate GRCC → REC)

## 3. CLI Simulation Interface

Examples:  
- `capsule-runner inject manifest --file m01.mock.json`  
- `capsule-runner simulate fork --input ./path/fork-01/`  
- `capsule-runner replay capsule --id CPS-X994`  
- `capsule-runner alert trace --zone audit-core`

## 4. Container Configuration & Runtime Assets

- Image: `maxoneopen/mock-runtime:latest`  
- Volumes: `/ledger`, `/capsules`, `/logs`  
- Mount configuration: YAML-based scenario config  
- Optional: `capsule-replay-viewer` for trace inspection (UI-ready)

## 5. Certification Disclaimer

This runtime is not part of a production deployment. It is a capsule logic simulator for validation and onboarding. All emitted capsules are marked as `mock`, `unsigned`, and `trace-only`. For certified capsule chains, use validated runtime replay as described in Module 17.