Developer Onboarding - Amadeus Genesis Hack

🔷 GENERAL INFORMATION

Network Access

Amadeus L1 blockchain is **live on mainnet**.
Developers can interact with the network through:

- **RPC Endpoints**
- **Chain Specifications**
- **Wallet Setup Guide**
- **Block Explorer**
- **Testnet Faucet** (will be provided soon)

➡️ See the **Tech Docs** for full details:
https://docs.ama.one

Why Build on Amadeus?

1. Consensus-Level Agent Training

- uPoW embeds *real compute* (training + inference) directly into consensus.
- Agents can **self-evolve on-chain**, with weight updates cryptographically verified.
- Future support for **privacy-preserving training proofs**. This creates the world's first blockchain where **agent intelligence is a consensus primitive**.

2. Deterministic Agent Runtime

- WASM-based runtime ensures **predictable execution** across all nodes.
- Deterministic computation enables:

    - Safe agent orchestration
    - Verifiable reasoning
    - Reproducible outputs across the entire network
- All agent state transitions are **persistent and globally verifiable**.

3. High-Performance Layer 1 (Built for Agents)

- ~0.5s finality (real-time agent execution)
- Rust/Elixir architecture
- BLS12-381 aggregated signatures
- Parallelized networking for agent-heavy workloads

Amadeus is *the* blockchain for onchain intelligence.

Is Useful Proof of Work (uPoW) live on mainnet yet?

Yes - partially.
uPoW is live on mainnet today in its initial form, supporting MatMul-based Useful Compute.

What's *not* live yet:

- End-to-end agent training via uPoW
- Consensus-level validation of agent weight updates
- Training proof settlement

These are upcoming features on the Nova Compiler and Agent Runtime roadmap.

For this hackathon:

- **Hard Hack**
    - Uses real MatMul workloads, aligned with the live uPoW pipeline.

    - Benchmarks reflect the compute that miners run on Amadeus mainnet today (not simulations).

- **Soft Hack**
    - You can assume uPoW exists as a compute layer today, but cannot rely on full agent training integration yet.
    - Designs may incorporate uPoW as a future training + inference engine.

Do submissions have to be open source?

No.
You may submit:

- Fully open source
- Partially open source
- Closed source with proper documentation

Open source submissions receive higher consideration where applicable.

Can I use external models or datasets?

Yes.
You may use:

- Pre-trained models
- External datasets
- Public weights
- Your own custom data

🔷 HARD HACK — RISC-V BENCHMARKING COMPETITION

The Hard Hack focuses on low-level performance engineering using RISC-V workloads and upcoming AMA compute primitives.

Environment Setup

Which architecture are we targeting?

The benchmarking infrastructure will be executed on:

- **RISC-V chips (TensTorrent-class hardware)**

Exact microarchitecture and specs will be released before Day 1.

What is the expected input/output format?

Two workload types:

1. Matrix Multiplication (MatMul)

- Fixed matrix sizes (to be provided)
- Required precision (fp32/fp16/int8)
- Expected output: execution metrics + result hash

2. AMA Workloads (Task-Specific)

These may include:

- Convolution kernels
- Attention-style workloads

- Small model inference microbenchmarks

Documentation for each workload will include:

- Input schema
- Output schema
- Time/memory expectations

Reference Docs

Validator setup & node info:
https://docs.ama.one/validator/running-a-node

API Reference

Endpoints for submitting benchmark results, fetching workloads, pulling validation results will be released prior to start.

What do submissions include?

Every submission **must include**:

- Raw metrics (latency, throughput, ops/sec)
- Correctness proof / output hash
- Docker container for reproducibility
- Source code or compiled binary
- Benchmark metadata (compiler flags, libraries used, etc.)

Environment & Constraints

- **Hardware:** RISC-V chips (TensTorrent) or GPU-based simulation
- **Data types:** Provided with workload
- **Time limits:** Strict (per workload)
- **Memory limits:** Enforced
- **Caching:** Allowed (in this iteration)

Are caching or precomputation allowed?

Yes, caching is allowed **as long as workload input is not modified**.

Number of submissions per day?

Unlimited.

Do I have to containerize my submission?

Optional, but recommended for full reproducibility.

Submission Workflow

1. **Request API Key**
2. **Receive Workload Spec**
3. **Run Locally / Optimize**
4. **Submit via JSON or Upload Container**

5. **Receive Score**
6. **Optional Validation Run**
7. **Score Locked to Leaderboard**

Evaluation & Scoring

Criteria

- **Latency** (primary)
- **Throughput**
- **Correctness**
- **Resource usage** (optional depending on workload)

Miner competition scoring is based on:
✔ **valid-sols / second**

ZK-style tasks may include:
✔ **novelty + correctness weighting**

Scoring Formula

Released with workloads. Likely:

```
score = weighted(latency, throughput, correctness)
```

Tie-Break Rules

1. Lowest latency
2. Lowest memory usage
3. Earliest submission timestamp

Leaderboard

A real-time leaderboard will be available.

Appeals

If your score seems incorrect, you may request:

- **Manual review**
- **Re-run on reference hardware**

Fairness & Anti-Cheat Rules

Is caching allowed?

Yes for this event.

Are hardcoded model parameters allowed?

Only to optimize compute — not to circumvent workload rules.

Can I modify workload shapes or data?

❌ **No.**
This results in instant disqualification.

Are optimized libraries allowed?

Yes:

- BLIS
- OpenBLAS
- TVM
- Custom kernels

Random seeds?

If randomness affects output correctness, seed must be consistent.

🔷 SOFT HACK — IDEATHON TRACK

The Soft Hack is a **design + architecture challenge** focusing on what could be built on Amadeus.

What is technically possible today?

You can build using:

- Token minting / transfers
- State proofs
- TX / receipt proofs
- WASM VM Contracts
- MCP integration for agents

Should we build for today or the future?

- **Build** for *today's* features
- **Ideate** for upcoming uPoW & Nova Runtime

What NOT to propose

- Web2 apps with a forced AMA label
- CEX tokens or memecoins
- Irrelevant NFTs
- Pure frontends with no agent logic
- Forks of existing DeFi apps with no innovation

Preferred Verticals

We especially welcome:

- **DeFi trading agents**
- **Sensor/perception agents**
- **Risk evaluation agents**
- **Compliance/KYC logic agents**
- **Onchain AI marketplaces**
- **Swarm coordination frameworks**

Submission Requirements

Your submission must include:

✔ Concept Deck

Clear overview, problem/solution, use cases.

✔ Architecture Diagram

System components + data flow + agent roles.

✔ Prototype / Mockups (optional)

UI, diagrams, logic flows, or code.

✔ How Amadeus Is Used

Explain integration points:

- uPoW (future)
- WASM runtime
- State proofs
- Agent identity/memory
- Oracle streams
- Swarm coordination

✔ Monetization Path (optional)

How the application sustains itself.

✔ Tradeoffs & Feasibility

What works today vs requires future support.

Evaluation Criteria

1. Creativity & Novelty

- Originality of idea
- Differentiation from existing apps

2. Technical Feasibility

- Can it be built on Amadeus today?
- If future features required, are assumptions reasonable?

3. Implementation Specificity

- Clarity of architecture
- Defined data flows
- Realistic build plan

4. Contribution to the Ecosystem

- Expands agent use cases
- Helps tooling, infra, or adoption
- Potential for ecosystem impact

5. Documentation Quality

- Clear, structured, complete
- Easy for judges to understand the design

6. Real-World Usefulness

- Solves an actual problem
- Has a plausible path to usage