MICROFLOW OPTIMIZATION

The Key to a Faster Mendix Application

Anti-Patterns & Traps to Avoid

Bonus: Quick Implementation Guide

golden earth

WHAT DO WE MEAN BY MICROFLOW OPTIMIZATION?

Here's why:

- ✕ Each commit operation **adds overhead** to the database.
- ✕ Committing inside loops **multiplies this overhead**, making processing exponentially slower.
- ✕ Especially problematic for high-volume processing.
- ✕ **Fix it:** Batch updates and commit once at the end of the operation.

3

MORE ANTI-PATTERNS TO AVOID

- ✕ Retrieving too many records without without filtering them first.
- ✕ Multiple sequential retrieves instead of combining queries.
- ✕ Calling sub-microflows with commits unnecessarily
- ✕ Repeated calculations instead of caching the result
- ✕ Monolithic microflows that do too much

golden earth

4

ACTIONABLE IMPLEMENTATION PLAN

Let’s Put It Into Action

✓ Time to review your microflows & start optimizing!

golden earth

1. REVIEW YOUR EXISTING MICROFLOW

- ✓ Identify your most frequently executed microflows.
- ✓ Check for performance bottlenecks in large data processing flows.
- ✓ Identify loops that could be optimized or removed.
- ✓ Look for unnecessary retrieves that could be improved.
- ✓ Check if your microflows are handling **too many responsibilities** split logic where necessary.

2. OPTIMIZE DATABASE OPERATIONS

- ✓ Use **XPath/OQL** instead of retrieving full tables and filtering in-memory.
- ✓ Retrieve **only required attributes** instead of entire records.
- ✓ Ensure indexing is applied to frequently queried attributes.

3. OPTIMIZE MICROFLOW EXECUTION

- ✓ Replace **loops with list operations** where possible.
- ✓ Use **batch commits** instead of committing inside loops.
- ✓ Implement **asynchronous processing** for long-running operations.
- ✓ Add **performance logging** to track execution time for key microflows.

4. TEST & MEASURE PERFORMANCE

- ✓ Establish a **baseline** by measuring current execution times.
- ✓ Run **performance tests** under realistic data loads.
- ✓ Optimize **high-impact microflows first** for maximum performance gain.
- ✓ Set up **alerts for slow-performing microflows.**

TOOLS & RESOURCES FOR DEVELOPERS

- **Mendix Runtime Dashboard** – Monitor app execution & performance
- **APM (Application Performance Management) Tools** – Analyze real-time performance data.
- **Database Query Analyzers** – Optimize database retrieval times.
- **Microflow Profiling Tools** – Track execution time per microflow step.
- **Performance Testing Frameworks** – Load-test your application under high user traffic.

10

DEVELOPER-FRIENDLY CHECKLIST FOR DAILY US

- ✓ Are all database interactions optimized? (No redundant retrieves, proper indexing)
- ✓ Are you avoiding commits inside loops?
- ✓ Are large datasets being batch processed?
- ✓ Are you using built-in list operations instead of manual loops?
- ✓ Are expensive operations being done asynchronously when needed?
- ✓ Are you logging performance-critical operations?
- ✓ Are you monitoring execution times and setting baselines?
- ✓ Are frequently used microflows optimized for scalability?
- ✓ Is your error handling structured and consistent?
- ✓ Is your microflow structure modular, readable, and easy to maintain?

golden earth

SAVE THIS CHECKLIST. SHARE IT WITH YOUR TEAM. START OPTIMIZING.

What are your top tips for building high performance microflows?

Drop them in the comments let's build better together

hello@goldenearth.io

www.goldenearth.io

golden earth