COMPUTERIZED ACCOUNTING SYSTEM v1.3.176

SOURCE CODE REVIEW DOCUMENT

Technical debt is the measure of the cost of reworking a solution caused by choosing an easy yet limited solution.

The most significant consequence of a complex technical debt is that it hinders a company's ability to compete and innovate. It robs you of resources, time, energy, and the ability to innovate, adapt, and grow.

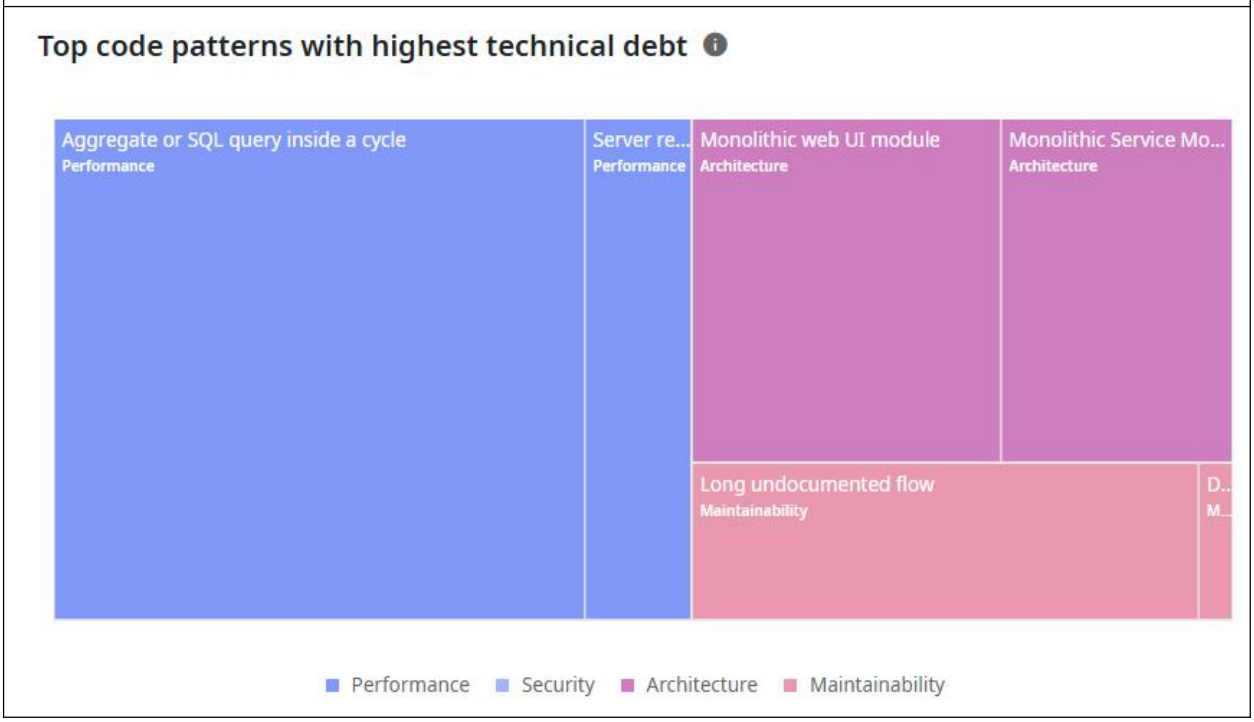
And it's one of those things that's hard to identify and manage and even harder to avoid.

COMPUTERIZED ACCOUNTING SYSTEM v1.3.176

TECHNICAL DEBT PER CATEGORY

CATEGORY	FINDINGS		TECHNICAL DEBT (%)	
	Previous SPRINT	Current SPRINT	Previous SPRINT	Current SPRINT
Performance	21	25	64%	65%
Security	0	0	0%	0%
Architecture	1	1	35%	31%
Maintainability	1	13	1%	4%
Total	23	39	**100%**	**100%**

TOP CODE PATTERN WITH HIGHEST TECHNICAL DEBT



FINDINGS		
CODE PATTERNS	**FINDINGS**	**TECHNICAL DEBT (%)**
Aggregate or SQL query inside a cycle	16	48.48%
End-User module providing services	1	3.03%
Sequence of connected Aggregates	5	15.15%
Unlimited records in Aggregate	4	12.12%
Unused Action in Module	1	3.03%
Missing descriptions on public element or parameter	6	18.18%
Duplicated Code	8	0%
TOTAL	39	100%

Remarks:

1. **Aggregate or SQL query inside a cycle**

Impact - Executing Aggregates and SQL queries inside a cycle can have severe performance impact due to database communication overhead repeated at each iteration. The impact can be greatly worsened when iterating through a list with a high number of elements or when having nested cycles.

2. **End-User module providing services**

Impact - An End-user module is not intended to provide services - usually, any reference to it reveals the existence of misplaced reusable code. References to an End-user compromise life cycle independence between applications and typically pull a lot of indirect unwanted references.

3. **Sequence of connected Aggregates**

Impact - A sequence of Aggregates that reference one another usually leads to unnecessary data fetching. Considering that each Aggregate executes a request to the database, this results in unnecessary database communication overhead.

4. **Unlimited records in Aggregate**

Impact - More records are fetched from the database than are used by the application, resulting in useless I/O and memory consumption.

5. **Unused Action in Module**

Impact - Unused actions can bloat your code base, make maintenance difficult, and increase security risks.

6. **Missing descriptions on public element or parameter**

Impact - Meaningful descriptions in modules, public elements, entities, and input/output parameters clarify their purpose and expected behavior. It's crucial when consuming closed modules, because the implemented logic isn't visible.

The technical debt of the modules (SCALE)

Lowest Technical Debt
Low Technical Debt
Medium Technical Debt
High Technical Debt
Highest Technical Debt

COMPUTERIZED ACCOUNTING SYSTEM v1.3.176 Over all technical Debt is "*MEDIUM TECHNICAL DEBT*"

	Name	Designation	Signature	Date Signed
Prepared By	Rommel Simundo	Application Analyst I		Sept 29, 2025
Checked By	Dominic Bacod	Accounting and Insurance Development Head		
Approved By	Jigger Caneo	Chief Technology Officer		