UAB
Universitat Autònoma de Barcelona

Dipòsit digital de documents de la UAB

This is the **published version** of the bachelor thesis:

Marchiori Soms, Hector. *Conversational AI Agent for M3 : Natural Language Interface for Electronic Component Management at Salicru S.A.*. Treball de Final de Grau (Universitat Autònoma de Barcelona), 2026 (Grau en Intel·ligència Artificial)

This version is available at https://ddd.uab.cat/record/336308

under the terms of the  license.

Conversational AI Agent for M3: Natural Language Interface for Electronic Component Management at Salicru S.A

Hector Marchiori Soms

June 27, 2026

Abstract

This paper presents M3-Agent IA, a conversational artificial intelligence agent that provides natural-language access to the Infor M3 database at Salicru S.A., a Spanish manufacturer of uninterruptible power supplies with over fifty years of industry experience. Electronics laboratory engineers previously faced a steep learning curve to retrieve component data from M3, as the system requires familiarity with non-intuitive database schemas, encoded field names, and domain-specific query conventions, introducing significant delays in procurement and design workflows. The proposed system removes this barrier by allowing engineers to pose queries in plain English or Spanish and receive grounded, structured answers in seconds. Technically, the agent is built on Google Gemini 3.5 Flash with native function calling and orchestrates four specialised tools: an M3 Structured Query Language engine with a T-SQL-to-PostgreSQL adaptation layer, a DigiKey API v4 connector for real-time external market data, and a Retrieval-Augmented Generation pipeline for datasheet consultation using Gemini Vision and `gemini-embedding-2` embeddings stored in pgvector. Persistent cross-session memory and vector-based datasheet indexing, backed by two dedicated PostgreSQL instances, allow the agent to recognise previously queried components without re-querying M3. The system is deployed on Google Kubernetes Engine at Salicru and also runs locally against a PostgreSQL replica.

Keywords: Large Language Models; natural language interfaces; Retrieval-Augmented Generation; text-to-SQL; electronic component management

1 INTRODUCTION

SALICRU S.A. is a Spanish manufacturer of uninterruptible power supplies and electrical protection equipment founded in 1966 and headquartered in Santa Maria de Palautordera, Catalonia. The company's electronics laboratory actively manages several thousand discrete components stored in an Infor M3 database hosted on Microsoft Azure SQL Server.

Retrieving component data from M3 presents a significant learning curve for laboratory engineers. The system requires familiarity with non-intuitive encoded field names, domain-specific query conventions, and a complex relational schema spanning multiple interlinked tables, introducing significant delays in procurement and design workflows.

This project proposes M3-Agent IA, a conversational AI agent designed to eliminate the barriers to accessing manufacturing knowledge. It enables engineers to ask questions in natural English or Spanish and receive accurate, grounded, and well-structured answers within seconds. The system integrates four capabilities in a single conversational flow: natural language querying of the M3 database, real-time external market data retrieval via DigiKey API v4, technical parameter extraction from manufacturer datasheets using a Retrieval-Augmented Generation pipeline, and persistent cross-session memory that retains previously queried component data across sessions.

Validation with five Salicru engineers over three weeks demonstrated consistent and measurable benefits: mean task completion times were reduced by 82% across all measured categories, with peak reductions of 94% for external equivalent search and 92% for obsolescence detection. The overall Quality of Experience score was 8.8/10, with unanimous willingness to recommend the tool, and engineers with no prior M3 experience achieved operational autonomy from the first day of use. Economic analysis shows a 957% annual return on investment for the 15-engineer team.

The main objective of this project is to design, im-

• Contact E-mail: h.marchiori02@gmail.com
• Supervised by: Javier Panadero Martínez
• Academic Year 2025/26

plement and deploy a production-ready conversational AI agent that enables Salicru's engineers to access M3 component data without requiring Structured Query Language expertise. To achieve this, five specific objectives are pursued: *(i)* develop a natural language interface for M3 using large language model function calling and schema-guided query generation; *(ii)* integrate external market data from DigiKey API v4 in a unified conversational flow; *(iii)* implement a Retrieval-Augmented Generation pipeline for datasheet consultation; *(iv)* develop a cross-session persistent memory system that retains component knowledge across user sessions; and *(v)* evaluate the system's impact on real engineering workflows through a structured user study.

The remainder of the paper is structured as follows: Section 2 presents the specific objectives; Section 3 describes the project planning and methodology; Section 4 reviews related work; Section 5 presents the system architecture; Section 6 details the implementation of each component; Section 7 presents the evaluation results; Section 8 analyses the return on investment; and Section 9 concludes with future work directions.

2 Objectives

The primary objective of this project is to design, implement and deploy a conversational AI agent that provides natural-language access to Salicru's Infor M3 database, eliminating the learning curve that currently limits self-service component data retrieval in the electronics laboratory.

To achieve this, the project is divided into five specific objectives:

- **Design and Implement a Natural Language Interface for M3:** Develop a text-to-SQL engine guided by a domain-specific schema context document, enabling engineers to query component inventory, stock levels, and manufacturer references in plain language without Structured Query Language knowledge.
- **Develop an External Market Data Integration:** Design and implement a DigiKey API v4 connector that retrieves real-time pricing, stock availability, lifecycle status, and datasheet URLs, unifying internal database data and external market information in a single conversational flow.
- **Implement a Datasheet RAG Pipeline:** Design and implement a Retrieval-Augmented Generation pipeline that processes manufacturer datasheets using Gemini Vision and stores embeddings in pgvector, enabling engineers to query technical parameters directly from PDF documentation.
- **Develop a Cross-Session Persistent Memory System:** Design and implement a memory system backed by PostgreSQL full-text search that retains component knowledge across sessions, reducing redundant queries and improving response latency for repeated lookups.
- **Conduct an Empirical Evaluation:** Assess the system's impact on real engineering workflows through a structured user study with Salicru's laboratory engineers, measuring task completion time and Quality of Experience.

3 Project Planning

3.1 Development Methodology

The project followed an Agile methodology structured around the five mandatory UAB submission dates, which served as hard deadlines anchoring each development phase. Rather than a waterfall approach, each phase was designed to deliver a functional and demonstrable component by the time of the next follow-up session. This ensured continuous validation of the system against real engineering requirements at Salicru throughout the development window.

The development spanned 16 weeks, from 10 March to 2 July 2026, was originally structured into five phases: Planning & Analysis, Internal M3 Search, DigiKey API Integration, Obsolescence Management, M3 Auto-Coding. Figure 1 shows the planned Gantt chart.

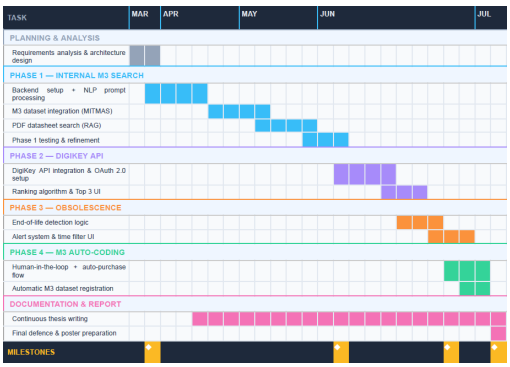


Fig. 1: Planned project timeline (10 Mar – 2 Jul 2026). Diamond markers indicate mandatory UAB submission dates.

3.2 Planned vs. Actual Execution

The actual development deviated from the initial plan in two significant ways. First, the implementation progressed faster than anticipated: a fully functional system integrating all three core tools — the M3 SQL engine, the DigiKey API v4 connector, the RAG pipeline — was deployed to Salicru's Google Kubernetes Engine infrastructure on **30 March 2026**, several weeks ahead of the milestones originally planned for each individual phase.

Once the first module was operational, it became evident that repeated queries about the same components introduced unnecessary latency and redundant API token consumption. This motivated the design and integration of a cross-session persistent memory system, backed by PostgreSQL full-text search, as an unplanned but high-impact addition to the architecture.

Second, Phase 4 (M3 Auto-Coding and Human-in-the-Loop registration module) was descoped following consultation with Salicru's engineering team. Engineers prioritised consolidating and evaluating the three existing tools over adding new write-access functionality to M3, a decision also motivated by security considerations around granting an LLM-driven system write permissions to a production ERP database.

Table 1 summarises the planned phases against actual outcomes.

Phase	Planned	Actual	Status
Planning	Mar 10–16	Mar 10–16	Done
M3 Search	Mar–May	Mar 10–30	Done
DigiKey API	May–Jun	Mar 10–30	Done
Obsolescence	Jun	Mar–Apr	Done
Memory	—	Mar–Apr	Done
Auto-Coding	Jun	—	Descoped
Evaluation	—	May–Jun	Done
Documentation	Apr–Jul	Apr–Jul	Done

TABLE 1: Planned vs. actual phase execution.

3.3 Risk Management

The main technical risk identified during planning was dependency on external APIs with usage limits and potential downtime: Google Gemini API and DigiKey API v4. This was mitigated by implementing automatic retry logic with exponential backoff for Gemini 429 rate-limit errors, OAuth 2.0 token caching with proactive renewal for DigiKey.

A second risk identified was the potential for the LLM to generate queries accessing sensitive data beyond the scope of the system's intended purpose. This was mitigated through two complementary mechanisms: the schema context document explicitly instructs the agent to restrict its operations strictly to the five permitted tables and to never attempt to infer, access, or expose data outside those boundaries; and the two-stage SQL security validation described in Section 6.1.2 enforces this constraint at the execution level, rejecting any query that references unauthorised tables regardless of how the request was formulated.

Together, these controls ensure that the agent cannot be used to access confidential corporate data that an AI system should not be exposed to, preserving both data privacy and the integrity of Salicru's information systems.

4 State of the Art

4.1 Large Language Models and Agentic Reasoning

The emergence of large language models (LLMs) with instruction-following capabilities has enabled a shift from passive text generation toward *agentic* systems that plan, act, incorporate external observations into a reasoning chain. The ReAct framework [1] formalises this as an interleaved sequence of *Thought*, *Action*, *Observation* steps, allowing models to call external tools and update their beliefs iteratively. Google Gemini [2] extends this with native parallel function calling, allowing multiple tools to be invoked in a single model turn, which is critical for latency-sensitive applications such as the one presented here.

4.2 Text-to-SQL

Text-to-SQL translates natural language into executable SQL queries against a relational schema. The Spider benchmark [3] established a standard evaluation protocol across 200 databases and 138 domains; state-of-the-art systems such as DIN-SQL [4] decompose the problem into schema linking, query classification, generation sub-tasks, achieving execution accuracy above 80% on Spider.

However, these systems are designed for well-documented, publicly available databases with descriptive field names. Corporate databases such as M3 present fundamentally different challenges: field names are proprietary and non-intuitive (`MMITNO` for item number, `MBSTQT` for stock quantity, `MPLVDT` for reference expiry date), values follow installation-specific conventions (e.g. `IDSTAT = '10'` for active suppliers in Salicru's instance, contrary to the documented default), correct queries require implicit domain knowledge such as mandatory filters and specific join patterns that are not inferrable from the schema alone. No publicly available text-to-SQL system is trained or evaluated on databases of this type, making direct application of existing systems infeasible without domain-specific adaptation. This motivates the *schema prompting* approach adopted in this work [5, 12].

4.3 Retrieval-Augmented Generation

RAG [6] augments a generative model with a retrieval stage that fetches relevant passages from an external corpus before generating an answer, reducing hallucination and enabling grounding in authoritative sources. Dense retrieval using bi-encoder models [7] maps both queries and documents to a shared embedding space; approximate nearest-neighbour search over vector stores such as pgvector [8] enables sub-second retrieval at scale.

Applied to technical documentation, RAG has been used to extract structured parameters from semiconductor datasheets. Recent work extends RAG to multimodal settings where PDFs contain tables and figures not amenable to plain text extraction: vision-language models process each page as an image, producing structured text representations that preserve numerical values and conditional columns [9].

4.4 Persistent Memory in Conversational Agents

Standard LLM-based agents are stateless by design: each conversation starts with no knowledge of previous interactions. Several approaches have been proposed to address this limitation. Retrieval-augmented memory systems store past interactions in a vector database and retrieve relevant context at query time [6]. Key-value memory stores such as those used in MemGPT [10] allow agents to explicitly write and read structured facts across sessions. Full-text search systems using inverted indexes offer an alternative that prioritises exact keyword matching over semantic similarity, which is advantageous when retrieving specific component codes or manufacturer references.

In this work, PostgreSQL `tsvector` full-text search with a GIN index was chosen over vector-based memory for two reasons: *(i)* component codes and manufacturer references (`020AB000000`, `BC327-25`) are exact strings for which keyword matching outperforms semantic similarity; and *(ii)* the same PostgreSQL instance used for M3 data replication can host the memory store, avoiding an additional infrastructure dependency. Each memory record is typed and scored by importance, enabling the agent to pri-

oritise critical facts such as M3 codes over secondary characteristics when injecting context into the system prompt.

5 System Architecture

5.1 System Overview

The agent operates as a conversational orchestrator: given a user query in natural language, Gemini 3.5 Flash determines which tool or combination of tools is needed, executes them, observes the results, and synthesises a grounded response returned through the Chainlit web interface. Figure 2 illustrates the overall system.

5.2 Technology Stack

- **Orchestrator:** Google Gemini 3.5 Flash with native function calling, temperature 0.1, maximum 8 consecutive tool calls per turn.
- **Interface:** Chainlit web application, HTTP 8000, supports file upload for PDF datasheets.
- **M3 database:** Microsoft Azure SQL Server hosting Salicru's corporate database, accessed read-only via ODBC Driver 18.
- **Memory and RAG:** Two PostgreSQL 15 instances deployed in GKE: `m3_memory_db` (cross-session memory, tsvector + GIN index) and `m3_rag_db` (vector embeddings, pgvector 0.8.2).
- **External APIs:** DigiKey API v4 for component search, pricing, datasheet URLs; `gemini-embedding-2` for dense text embeddings.
- **Production:** Google Kubernetes Engine (GKE).

5.3 Data Layer

The agent queries five tables from Salicru's corporate Azure SQL Server database in read-only mode. Table 2 summarises the tables and their role in the system.

Table	Rows	Content
MITMAS	181,576	Article master
MITBAL	327,091	Stock by warehouse
MITPOP	138,538	Manufacturer references
CIDMAS	7,420	Supplier master
MITWHL	48	Warehouse catalogue
Total	654,673	

TABLE 2: M3 tables queried by the agent in production.

A domain-specific challenge encountered during development was that `CIDMAS.IDSTAT` uses value `'10'` for active suppliers in Salicru's instance, contrary to the standard M3 documented value of `'20'`. Filtering by `IDSTAT` would silently exclude all real manufacturers, a rule encoded explicitly in the schema context document to prevent the LLM from introducing this error during SQL generation. Similarly, `MITPOP.MPPOPN` stores a full descriptive name rather than a bare manufacturer part number (e.g. `TRANS BIP SEÑAL BC327-25` instead of `BC327-25`), requiring the agent to extract the significant portion of the string when presenting results to the user.

5.4 Production Deployment

The system runs on Google Kubernetes Engine (GKE) in Salicru's Google Cloud project `salicru-sa`, cluster `salicru-test`, namespace `dev`. Two pods handle all production traffic:

- **m3-agent:** A stateless Chainlit application container built from a custom Docker image pushed to `eu.gcr.io/salicru-sa/m3-agent`. Handles all user interactions, LLM orchestration, tool dispatching. Connects to Azure SQL Server over an encrypted ODBC connection for M3 queries.
- **m3-agent-db-rag:** A stateful `pgvector/pgvector:pg15` container providing both `m3_memory_db` and `m3_rag_db`. Backed by a PersistentVolumeClaim ensuring memory and vector index persistence across pod restarts.

The CI/CD pipeline is managed by Jenkins: on each commit to the `master` branch, the Docker image is rebuilt, pushed to the Google Container Registry, the deployment is updated via `kubectl patch`. The `m3-agent` pod is stateless and can be restarted or scaled horizontally without data loss, as all persistent state resides in the database pod.

A local development environment mirrors production using a PostgreSQL replica of the five M3 tables, populated from CSV exports of the Azure SQL Server database. A single environment variable (`USE_SYNTHETIC_DB`) switches the agent between the live Azure SQL Server connection and the local replica without code changes, enabling full offline development and testing.

5.5 Novelty of this Work

Existing text-to-SQL systems target well-documented public databases; existing conversational agents for corporate data rely on proprietary, vendor-specific solutions tightly coupled to a single platform [11]; and existing Retrieval-Augmented Generation systems for technical documentation do not integrate with live inventory or supplier data. To our knowledge, no published system combines: *(i)* natural language querying of a proprietary corporate database with installation-specific naming conventions; *(ii)* real-time external market data retrieval from a distributor API; *(iii)* multimodal Retrieval-Augmented Generation over manufacturer datasheets; and *(iv)* cross-session persistent memory in a unified conversational agent deployed in production. This combination constitutes the primary novelty of M3-Agent IA.

6 Implementation

6.1 M3 SQL Query Engine

6.1.1 Schema Prompting Strategy

The schema context document is the central guide for the orchestrator — it is the compass that orients Gemini when generating SQL queries against Salicru's corporate database. Without it, the model would have no knowledge of the proprietary field names, domain conventions, or implicit constraints that make M3 queries correct.

The document, approximately 3,000 tokens in length, encodes five categories of information. First, *table defini-*

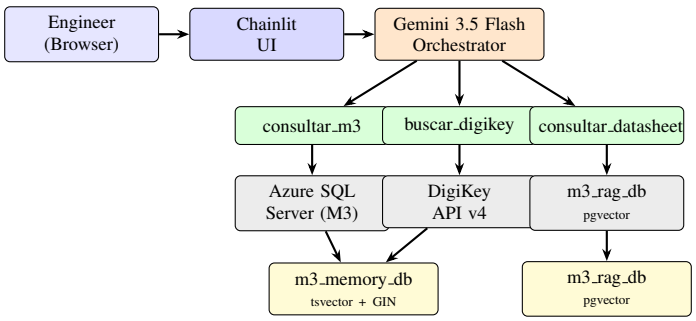


Fig. 2: M3-Agent IA system architecture. Gemini 3.5 Flash orchestrates three tools: an M3 SQL engine querying Azure SQL Server, a DigiKey API v4 connector, and a RAG pipeline backed by pgvector.

tions: for each of the five permitted tables, every relevant field is described with its type, semantic meaning, example values. For instance, `MMITDS` is annotated as storing abbreviated component names (e.g. `R F CARB AX` for "resistencia film carbon axial"), not full words — a critical distinction that prevents the model from generating `LIKE '%resistencia%'` queries that would return no results. Second, *domain conventions*: explicit documentation of Salicru-specific values such as `IDSTAT = '10'` for active suppliers and `CONO = 444` as the company identifier. Third, *canonical join patterns*: the exact join syntax required to link tables correctly, including the `LTRIM(RTRIM())` wrapping needed to handle trailing spaces in varchar fields. Fourth, *mandatory filters*: rules that must always be applied, such as `MMSTAT = '20'` to exclude obsolete articles. Fifth, *fifteen explicit SQL generation rules* developed iteratively through observation of failure modes during development.

Key rules include: *(i)* always filter `MMSTAT = '20'` for active items only; *(ii)* never apply `LIMIT` to manufacturer reference queries, as partial results cause the agent to report incomplete data to the user; *(iii)* never filter `CIDMAS` by `IDSTAT`, since Salicru's active suppliers carry value `'10'`, not the standard `'20'`, which would otherwise silently exclude all real manufacturers; and *(iv)* use a two-step retrieval pattern: first obtain the M3 code from `MITMAS` and `MITPOP` without joining `CIDMAS`, then resolve manufacturer names in a second query using the known code as a precise filter.

6.1.2 Security Controls

All LLM-generated SQL passes through two validation stages before execution:

- **Stage 1 — Syntactic validation:** Any statement not beginning with `SELECT`, or containing DML/DDL keywords (`INSERT`, `UPDATE`, `DELETE`, `DROP`), is rejected immediately before reaching the database.
- **Stage 2 — Semantic validation:** All table references are extracted from the query using a regular expression and verified against a whitelist of exactly five permitted tables (`MITMAS`, `MITBAL`, `MITPOP`, `CIDMAS`, `MITWHL`). Any query referencing tables outside this set raises an explicit error, regardless of whether the request originated from the LLM or from a user attempting prompt injection through the chat interface.

6.1.3 SQL Adaptation Layer

In production, the agent queries Azure SQL Server directly using T-SQL. During local development, a PostgreSQL replica of the five M3 tables is used instead. A transparent *SQL adaptation layer* in `core/db.py` bridges the two dialects at runtime when `USE_SYNTHETIC_DB=true`, applying the following transformations to every query before execution:

- `MVXJDTA.TABLE` → `"MVXJDTA"."TABLE"` (PostgreSQL requires double-quoted identifiers for case-sensitive schemas)
- `TOP` N → `LIMIT` N (different syntax for limiting result rows)
- `LTRIM(RTRIM(`x`))` → `TRIM(`x`)` (simplified whitespace trimming in PostgreSQL)
- `GETDATE()` → `CURRENT_DATE` (different current date function)
- `TRY_CAST` → `CAST` (PostgreSQL does not support `TRY_CAST`)

When `USE_SYNTHETIC_DB=false`, the layer is bypassed entirely and the original T-SQL query is sent directly to Azure SQL Server. The same prompt and schema context generate correct SQL for both environments without any code changes.

6.2 DigiKey API Integration

6.2.1 Component Search

The `buscar_digikey` tool queries the DigiKey API v4 keyword search endpoint. Each request applies the following filters and parameters: locale ES and currency € for Spanish market pricing; `SortByBestMatch` to rank results by relevance; `ExcludeMarketPlaceProducts: true` to return only stocked distributor inventory; `RequestedQuantity: 1` to retrieve unit pricing; and an optional manufacturer filter that restricts results to a specific supplier when the manufacturer is known from M3.

In normal operation, results are additionally filtered to retain only products with `ProductStatus = Active`, excluding `Discontinued` and `EndOfLife` items. When the search returns no active results, the filter is relaxed and all statuses are returned. For each result the tool returns: manufacturer part number, DigiKey part number, stock quantity, unit price, lifecycle status, up to 12 technical parameters, a direct datasheet URL.

Authentication uses the OAuth 2.0 client-credentials flow. The access token is cached in memory and renewed automatically 120 seconds before expiry. A `threading.Lock` prevents token-refresh race conditions when the obsolescence tool issues parallel requests.

6.2.2 Obsolescence Detection

Electronic components have a finite commercial lifecycle. Manufacturers typically support a component for 10 to 20 years before issuing an End-of-Life notice, after which production ceases and distributor stock gradually depletes. In the power electronics sector, where Salicru operates, this lifecycle is generally longer than in consumer electronics, but obsolescence remains a critical risk: a single discontinued component can block the production of an entire UPS model if no validated alternative is identified in advance. Proactive obsolescence management is therefore essential to ensure supply chain continuity.

The obsolescence detector cross-references internal lifecycle fields with real-time DigiKey data in two stages:

- **Stage 1 — Internal M3 scan:** Queries the database for components meeting either of two conditions: `MPLVDT > 0` (manufacturer reference with a non-zero end-of-validity date, indicating the supplier has communicated an expiry date for that reference) or `MMSTAT = '50'` (article explicitly marked obsolete in the database). Internally manufactured components (`MMMABU = 1`) are skipped as they have no external market equivalent.
- **Stage 2 — Real-time market validation:** Submits parallel DigiKey keyword searches using a `ThreadPoolExecutor(max_workers=5)`, enriching each flagged component with real-time lifecycle status from the distributor. It is important to note that DigiKey results are based on keyword matching and may return equivalent or alternative components with available stock rather than the exact part number registered in M3. The lifecycle status returned should therefore be interpreted as an approximation of market availability rather than a definitive obsolescence confirmation for the specific reference.

Table 3 illustrates the detection criteria.

Condition	M3 Field	Value
Expired reference	`MPLVDT`	> 0
Obsolete article	`MMSTAT`	`'50'`
Skip (internal mfr)	`MMMABU`	`= 1`

TABLE 3: Obsolescence detection criteria.

6.3 RAG Pipeline for Datasheet Consultation

6.3.1 Motivation

Manufacturer datasheets are the authoritative source for electrical parameters such as breakdown voltages, gain ranges, thermal resistance, and package pinouts that are not stored in the corporate database. A RAG pipeline allows engineers to query these parameters in natural language rather than opening a PDF and scanning tables manually.

6.3.2 Indexing Pipeline

When the agent needs datasheet information, it retrieves the PDF URL from the DigiKey result and downloads it. The document is processed page by page: each page is rendered as an image and passed to Gemini 3.5 Flash Vision with a structured extraction prompt that outputs a standardised text representation preserving table structure, parameter names, values, measurement conditions.

To balance accuracy with response latency, the pipeline processes a maximum of 8 pages per datasheet regardless of document length. This limit was established empirically to keep first-access indexing time under 68 seconds while covering the electrical characteristics sections present in the first pages of most component datasheets. If the user uploads a PDF manually, the system first validates that it contains electronics-related content before indexing.

The extracted text for each page is embedded using `gemini-embedding-2` (768-dimensional vectors) and stored as a chunk in `m3_rag_db` using pgvector with an HNSW index for approximate nearest-neighbour search. Indexed documents are keyed by MD5 hash of the PDF content, so a second query for the same component is served from the existing index without re-downloading or re-processing the PDF, reducing latency from 45–68 s (first access) to 2–3 s (cached).

6.3.3 Retrieval and Synthesis

Given a user question, the query is enriched with domain-specific technical synonyms (e.g. "gain" expands to include "hFE current gain beta classification groups") before embedding. The top-k ($k = 3$) most similar chunks are retrieved by cosine similarity and passed to Gemini 3.5 Flash for answer synthesis. Responses include page-level source references enabling engineers to verify values in the original document.

6.4 Persistent Memory System

The persistent memory system eliminates redundant M3 queries for previously searched components and reduces response latency for repeated lookups. It is implemented across two PostgreSQL 15 databases with different storage strategies optimised for their respective retrieval patterns.

6.4.1 Cross-Session Factual Memory

`m3_memory_db` stores structured facts about components retrieved during previous sessions. Each record contains: a content string (the fact itself), a semantic type, an importance score, a namespace identifier, a keyword tag list, a content hash for deduplication, creation and access timestamps.

Four semantic types are defined, each with a fixed importance score: `codigo_m3` (score 9) stores the mapping between a component name and its M3 article code; `ref_fabricante` (score 8) stores active manufacturer references and their associated supplier; `caracteristicas` (score 7) stores key technical characteristics extracted from the M3 description; and `digikey_equivalente` (score 9) stores confirmed external market equivalents.

Retrieval uses PostgreSQL `tsvector` full-text search with a GIN index. The user query is tokenised, stopwords are removed, the remaining tokens form an OR-query executed against a pre-computed `search_vector` column that weights content tokens higher than tag tokens. The top-5 results by importance and relevance score are injected into the system prompt before the model processes the question. Deduplication is enforced via MD5 hash of the content string, preventing the same fact from being stored multiple times across sessions.

Stock levels, prices, availability data are explicitly excluded from memory, as these values change frequently and cached data would lead to incorrect answers.

6.4.2 Vector Memory for Datasheet Content

`m3_rag_db` stores the indexed content of manufacturer datasheets as dense vector embeddings. Each record contains a document hash, component name, M3 code, manufacturer part number, source URL, page number, chunk text, a 768-dimensional embedding vector generated by `gemini-embedding-2`.

Retrieval uses cosine similarity search via pgvector's HNSW index, which provides approximate nearest-neighbour search in sub-second time. Unlike `m3_memory_db`, which uses keyword matching optimised for exact component codes, `m3_rag_db` uses semantic similarity optimised for technical parameter queries where the exact terminology may differ between the user's question and the datasheet text.

7 Evaluation

7.1 Experimental Setup

The system was evaluated by five engineers from Salicru's electronics laboratory over a three-week period (May–June 2026). Each engineer independently completed a set of standardised tasks both using the native M3 interface (control condition) and using M3-Agent IA (experimental condition), recording completion times and qualitative feedback in a structured evaluation form. The completed evaluation forms are included in the appendix of this report as primary evidence. Tasks covered six categories: component lookup by reference or description, manufacturer reference retrieval, stock query by warehouse, obsolescence check, external equivalent search via DigiKey, technical parameter retrieval from datasheet. Subjective Quality of Experience (QoE) was collected using a structured form with ten dimensions rated 1–10, completed after at least two weeks of daily agent use.

7.2 Response Time

Table 4 reports mean task completion times averaged across all five engineer evaluation forms, both before and after agent deployment. Agent times include all LLM API calls, database queries, and network latency from query submission to final response displayed in the Chainlit UI.

The largest absolute time savings are observed in external equivalent search (GAP = 5.96 min) and component lookup (GAP = 4.92 min). Notably, component lookup and manufacturer reference retrieval, while showing moderate relative reductions (86% and 84%), represent the highest-frequency tasks in daily laboratory work. Their combined impact on the team's productivity is therefore disproportionately large relative to the percentage figures alone.

Task	Without (min)	With (min)	GAP (min)	Red. (%)
Comp. lookup	5.70	0.78	4.92	86
Mfr. refs.	3.25	0.53	2.72	84
Stock/warehouse	1.50	0.48	1.02	68
Obsolescence	4.67	0.37	4.30	92
DigiKey equiv.	6.33	0.37	5.96	94
Datasheet param	2.50	0.80	1.70	68
Mean	**4.00**	**0.56**	**3.44**	**82**

TABLE 4: Mean task completion times averaged across five engineer evaluation forms (May–June 2026). GAP = time saved per task. Agent times include all LLM and database latency.

7.3 Quality of Experience

Table 5 presents QoE scores collected after at least two weeks of daily agent use, averaged across the five evaluation forms. The overall mean score was 8.8/10. The highest-rated dimension was net promoter intent (all five engineers rated it 10/10), reflecting unanimous willingness to recommend the tool to colleagues. The lowest-rated dimension was response speed (mean 7.6/10), consistently identified as the primary area for improvement across three of five engineers in open feedback.

Dimension	Mean (1–10)
Response accuracy	8.4
Response speed	7.6
Ease of use	9.2
NL understanding	8.8
DigiKey data utility	8.8
Obsolescence detection	8.8
Datasheet (RAG)	8.8
Data trustworthiness	8.6
Workload reduction	9.0
Net promoter intent	10.0
Overall mean	**8.8**

TABLE 5: Quality of Experience scores averaged across five engineer evaluation forms. Scale: 1 = very poor, 10 = excellent.

7.4 Learning Curve

Figure 3 shows the autonomous task completion rate over time for three engineers who reported no prior M3 experience, drawn from their evaluation forms. With the agent, all three reached 65–100% autonomous completion within the first day of use, compared to the several weeks typically

required to reach equivalent autonomy through direct M3 training.

The flat trajectories of Engineer A (65%) and Engineer B (90%) do not indicate stagnation but rather reflect that both engineers reached a stable and sufficient level of autonomy from day one and maintained it consistently throughout the evaluation period. Their roles — focused on hardware development and PCB design respectively — involve a limited and well-defined set of M3 queries (component lookup and stock consultation), which the agent handles correctly and consistently from the first interaction. There was therefore no room or need for further progression: the agent solved their M3-related tasks completely from day one. Engineer C shows a different profile, reaching 50% on day one and 100% from day two onward, reflecting a broader range of query types that required one additional session to cover fully. In all three cases, the agent enabled a level of autonomy that would have taken several weeks to achieve through direct M3 training, as illustrated by the dashed baseline. One engineer noted: *"Although my work is centred on PCB design and I do not use M3 habitually, the agent has allowed me to significantly speed up daily tasks."*

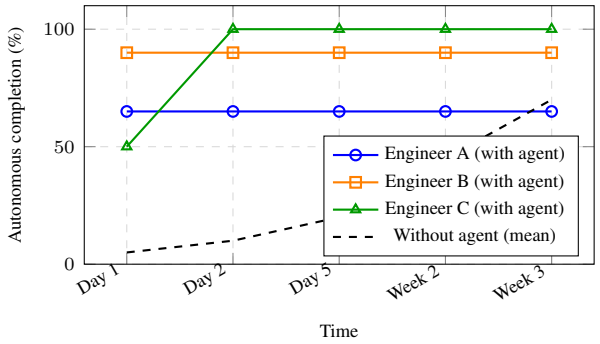


Fig. 3: Autonomous M3 task completion rate over time for three engineers with no prior M3 experience. The dashed line represents the mean progression without the agent based on engineer estimates.

7.5 Representative Use Cases

Table 6 presents representative queries submitted by engineers during the evaluation period, illustrating the diversity of tasks handled correctly in production.

Query	Tool	Correct
"Find compatible models for 023AE000376"	M3	Yes
"Is MT40A1G16TB active or obsolete?"	DigiKey	Yes
"Find flat cable and connector for 010AG000006"	M3	Yes
"Find stock alternative for 014JL000001"	M3	Yes
"Find alternative for NCV2931ADT5.0RKG"	DigiKey	Yes
"Search IC equivalents in the market"	DigiKey	Yes
"Stock of component by value only"	M3+DK	Yes

TABLE 6: Representative engineer queries during evaluation (May–Jun 2026). M3+DK indicates both tools were used.

7.6 Discussion

The evaluation results demonstrate consistent and significant task completion time reductions across all measured categories, with an overall mean reduction of 82%. The two most impactful improvements are in component lookup (86% reduction) and manufacturer reference retrieval (84% reduction), as these are the highest-frequency tasks in the laboratory workflow. Even a moderate time reduction in these categories compounds into substantial daily savings across the full engineering team.

The QoE results reveal an important asymmetry: ease of use (9.2/10) and net promoter intent (10/10) are the highest-rated dimensions, confirming that the natural language interface successfully removes the learning curve barrier identified in the introduction. In contrast, response speed (7.6/10) is the only dimension below 8/10, reflecting the inherent latency of external LLM API calls. This limitation is structural rather than architectural — the system is already optimised with persistent memory (reducing repeated queries from 13–15 s to under 6 s) and functional indexes (eliminating full-table scans on 327,091 MITBAL rows) — and cannot be fully resolved without a self-hosted model deployment.

The learning curve results are particularly significant. Engineers with no prior M3 experience achieved 65–100% autonomous task completion from the first day of use, compared to several weeks of training required without the agent. This suggests the agent does not merely accelerate existing workflows — it fundamentally changes the accessibility of the system, enabling engineers whose primary role does not involve M3 (such as PCB designers) to independently retrieve component data without specialist support.

The most common failure mode reported by engineers was occasional hallucination in extended conversations, consistent with known limitations of context-window saturation in long sessions. This affected a minority of interactions and did not prevent correct answers from being retrieved in subsequent queries. A second failure mode specific to the RAG module was high first-access latency (45–68 s) for datasheet parameter queries, noted by engineers as high relative to direct PDF consultation, though second-access latency (2–3 s from cache) was rated positively.

Overall, the evaluation confirms that M3-Agent IA meets its primary objective: enabling engineers to access component data without SQL expertise, with measurable time savings, high user acceptance, a demonstrable reduction in the M3 learning curve.

8 Return on Investment

8.1 API Cost Analysis

The operational cost of M3-Agent IA was measured directly from billing data during the evaluation period. The team of 15 engineers incurs an observed cost of **10 EUR/day** using Gemini 3.5 Flash ($1.50/$9.00 per million input/output tokens). At an exchange rate of approximately $1.15/EUR,

this corresponds to $11.50 USD/day.

Assuming standard M3 query usage — prompts that include the schema context, conversation history, and M3 results (1,500 input tokens) with structured responses (500 output tokens) at a cost of $0.0121 USD/query — the daily budget supports approximately **940 queries/day** across the team. This corresponds to 63 queries/engineer/day or approximately 8 queries/hour in a standard 8-hour working day. This figure reflects the multi-step nature of each user interaction: a single engineer question typically triggers 3–5 internal API calls (SQL generation, tool dispatching, response synthesis), each consuming tokens independently.

8.2 Cost-Benefit Analysis

To quantify the economic impact of M3-Agent IA, a cost-benefit analysis was conducted comparing the annual cost of M3-related workflows before and after agent deployment. The analysis considers two cost components: personnel time spent on M3-related tasks, valued at the mean electronics engineer salary in Spain [13], and the operational cost of the Gemini API. Personnel time was derived from the mean task completion times measured in Section 7.2, applied across the full 15-engineer team over 220 working days per year.

Table 7 summarises the annual cost breakdown.

Cost component	Without agent	With agent
Personnel (M3 tasks)	25,465 EUR	3,481 EUR
API cost (Gemini)	—	2,200 EUR
Total annual cost	**25,465 EUR**	**5,681 EUR**

TABLE 7: Annual cost of M3-related workflows (15 engineers, 220 working days, 17.16 EUR/h).

Personnel cost without the agent is calculated from the mean task time of 28.5 min/day per engineer measured in the evaluation. With the agent, this reduces to 3.9 min/day, with the remaining cost representing the residual time for query formulation and result review. The API cost of 2,200 EUR/year corresponds to the observed 10 EUR/day operational cost over 220 working days.

8.3 ROI Calculation

Return on Investment (ROI) was selected as the primary metric for this analysis because it directly relates the net economic benefit of the system to its operational cost, providing a clear and intuitive measure of value that is independent of team size or absolute cost figures. A high ROI in this context indicates that the time savings generated by the agent far exceed the cost of the API subscription, making the investment economically justified even at small team scales.

The standard ROI formula is applied as follows:

$$ROI = \frac{\text{Net benefit}}{\text{Investment cost}} \times 100\% \qquad (1)$$

Where the net benefit is the annual personnel time saving (23,265 EUR) and the investment cost is the annual API cost (2,200 EUR):

$$ROI = \frac{23{,}265 - 2{,}200}{2{,}200} \times 100 = \mathbf{957}\% \qquad (2)$$

The system delivers a **957% annual ROI**, with a net annual saving of **21,065 EUR** and full API cost recovery in under two working days. The 77.7% reduction in total workflow cost demonstrates that the primary value of the agent is not merely convenience but a measurable and quantifiable economic impact on engineering operations.

It is important to note that this analysis considers only the direct time savings in M3-related tasks. Additional economic value not quantified here includes reduced onboarding time for new engineers (Section 7.4) and faster procurement decisions enabled by real-time DigiKey market data in the same conversational flow.

9 Conclusions and Future Work

9.1 Conclusions

This paper has presented M3-Agent IA, a production-deployed conversational AI agent that provides natural-language access to Salicru's Infor M3 database. The system demonstrates that LLM function calling, combined with careful schema prompting and domain-specific tool design, can effectively bridge the gap between domain experts and complex enterprise databases, removing the learning curve that previously limited self-service access to component data.

Evaluation by five Salicru engineers over three weeks shows consistent and significant reductions in task completion time across all measured categories, with a mean reduction of 82% and peak reductions of 94% for external equivalent search and 92% for obsolescence detection. The overall Quality of Experience score of 8.8/10 and unanimous net promoter intent (10/10) indicate strong user acceptance. One engineer summarised the impact as follows: *"It is a tool I already use every day. It allows me to search for any component and finding equivalents can save me a lot of time."*

Economic analysis across the 15-engineer production team shows a 957% annual ROI, with a net saving of 21,065 €/year and full cost recovery in under two working days. The system is already in daily production use at Salicru, having been deployed to GKE on 30 March 2026.

The three main technical contributions of this work are: *(i)* a multi-tool architecture unifying internal ERP queries, external market data, datasheet RAG in a single conversational flow; *(ii)* a T-SQL-to-PostgreSQL adaptation layer enabling identical prompt logic across Azure SQL Server production and PostgreSQL development environments; and *(iii)* a schema prompting strategy with fifteen domain-specific SQL generation rules addressing the unique challenges of ERP database access, including encoded field names, non-standard status codes, implicit join constraints.

9.2 Future Work

Three concrete directions follow directly from the limitations identified during evaluation.

Response latency reduction. Response speed was the lowest-rated QoE dimension (7.6/10) and the most fre-

quently cited area for improvement. The current architecture depends entirely on the Google Gemini external API, introducing variable latency beyond the system's control. A natural next step is to evaluate self-hosted open-weight models such as Llama 3 [14] or Mistral [15] deployed within Salicru's GKE infrastructure, trading some response quality for deterministic sub-second latency and elimination of per-token API costs.

Schema expansion. The current whitelist of five permitted M3 tables covers component inventory and manufacturer references but excludes procurement orders, bill-of-materials data, supplier pricing history. Extending the schema context document and security whitelist to cover these tables would enable the agent to assist with full procurement workflows, from component selection through to purchase order generation.

RAG retrieval accuracy. First-access datasheet query latency (45–68 s) was noted as high relative to direct PDF consultation. Two improvements are planned: chunk-level sub-indexing, which splits large pages into parameter-level chunks to improve cosine similarity precision for specific parameter queries; and pre-indexing of the most frequently consulted datasheets at system startup, eliminating cold-start latency for common components.

Multi-department access control. One engineer suggested *"personalised access by department to different documents."* A role-based access layer would allow different teams (laboratory, procurement, engineering) to query different subsets of M3 tables and document collections, improving both security and relevance of responses.

ACKNOWLEDGMENTS

The author thanks Javier Panadero Martínez for his academic supervision and guidance throughout this project, the engineering team at Salicru S.A. for their time, feedback and access to production systems during the evaluation phase.

Above all, the author wishes to express his deepest gratitude to his family, whose unconditional support made this work possible: his mother Iolanda Soms Rodriguez, his father Michele Marchiori, his brother Joan Marchiori Soms, his second father Jordi Pera Fabregà. Thank you for everything, without you, nothing of what I've done and what I will do would be and will be reality.

The author also thanks his friends Isaac Correa Pacheco, Marina Funoll Cruz, Iván Jiménez Almagro, Víctor Sánchez Requena and Héctor Salgado de Oñate for their encouragement and company throughout these years.

Finally, the author thanks Nut, his dog and study companion, for his unconditional presence during long working sessions and for enforcing daily walks at precisely the moments they were most needed.

REFERENCES

[1] S. Yao et al., "ReAct: Synergizing Reasoning and Acting in Language Models," *Proc. ICLR*, 2023.

[2] G. Team et al., "Gemini: A Family of Highly Capable Multimodal Models," *arXiv:2312.11805*, 2023.

[3] T. Yu et al., "Spider: A Large-Scale Human-Labeled Dataset for Complex and Cross-Domain Semantic Parsing and Text-to-SQL," *Proc. EMNLP*, 2018.

[4] M. Pourreza and D. Rafiei, "DIN-SQL: Decomposed In-Context Learning of Text-to-SQL with Self-Correction," *Proc. NeurIPS*, 2023.

[5] I. Trummer, "From BERT to GPT-3: Revisiting Text-to-SQL Systems," *VLDB*, 2022.

[6] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," *Proc. NeurIPS*, 2020.

[7] V. Karpukhin et al., "Dense Passage Retrieval for Open-Domain Question Answering," *Proc. EMNLP*, 2020.

[8] A. Kane, "pgvector: Open-Source Vector Similarity Search for PostgreSQL," GitHub, 2023.

[9] M. Mathew et al., "DocVQA: A Dataset for VQA on Document Images," *Proc. WACV*, 2021.

[10] C. Packer et al., "MemGPT: Towards LLMs as Operating Systems," *arXiv:2310.08560*, 2023.

[11] SAP, "SAP Joule: AI Copilot Embedded Across the SAP Portfolio," SAP Technical Documentation, 2024.

[12] X. Qiu et al., "A Survey on Text-to-SQL," *arXiv:2303.12626*, 2023.

[13] Adecco, "Guía del Mercado Laboral 2026," Adecco Group Spain, 2026.

[14] Meta AI, "Llama 3: Open Foundation and Fine-Tuned Chat Models," *arXiv:2407.21783*, 2024.

[15] A. Jiang et al., "Mistral 7B," *arXiv:2310.06825*, 2023.

APPENDIX

Appendix Index

Appendix A ..... Engineer Evaluation Forms
Appendix B ..... System Demo Video
Appendix C ..... Source Code Repository

Appendix A — Engineer Evaluation Forms

The following pages contain the five anonymised evaluation forms completed by Salicru's engineers during the evaluation period (May–June 2026). The level of detail varies across forms and this variability was accounted for in the averaged results reported in Section 7, where each metric was computed only over the engineers who provided a valid measurement for that specific task.

M3-Agent IA: Evaluación de Impacto

Salicru S.A. · Departamento de I + D

TFG UAB 2026 — Hèctor Marchiori Soms

Nombre del ingeniero	[redacted]
Departamento	
Fecha inicio evaluación	
Fecha fin evaluación	
Versión del agente	V1.0 - Salicru Cloud

Este documento recoge la evaluación del impacto del agente M3-Agent-Electronics en las tareas cotidianas del departamento de I + D de Salicru. Por favor, rellena cada sección con la mayor precisión posible. Los datos son anónimos y se utilizarán exclusivamente para mejorar el sistema.

1. Medición de Tiempos por Tarea

Registra el tiempo (en minutos) que tardas en completar cada tarea, antes y después de usar el agente. Indica las observaciones relevantes y firma con tus iniciales.

Tarea	Tiempo sin agente (min)	Tiempo con agente (min)	Mejora (%)	Observaciones	Firmado
Localizar código M3 de un componente					
Buscar referencias de fabricante activas					
Consultar stock por almacén					
Detectar componentes obsoletos					
Buscar equivalente externo (DigiKey)					
Consultar datasheet de un componente					
Responder a pregunta técnica de un proveedor					
Preparar lista de componentes para pedido					

Tiempo total antes del agente (suma): _____ min

Tiempo total con el agente (suma): _____ min

Mejora global: _____ %

2. Quality of Experience (QoE)

Evalúa cada dimensión marcando la casilla correspondiente. 1 = muy deficiente, 10 = excelente.

Rellena esta sección después de al menos 2 semanas de uso del agente.

Dimensión evaluada	1	2	3	4	5	6	7	8	9	10
Precisión de las respuestas	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Velocidad de respuesta	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Facilidad de uso (interfaz Chainlit)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Comprensión de preguntas en lenguaje natural	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Utilidad de la información de DigiKey	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Calidad de la detección de obsolescencias	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Información de datasheets (RAG)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Confianza en los datos devueltos	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Reducción de carga de trabajo percibida	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Recomendarías el agente a un compañero	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Puntuación media total: _____ / 10

3. Curva de Aprendizaje

Esta sección está pensada para ingenieros que se incorporan al laboratorio sin experiencia previa en M3. Registra semanalmente qué porcentaje de tareas M3 eres capaz de completar de forma autónoma (sin ayuda de un compañero), en dos escenarios: sin el agente y con el agente.

Si ya conoces M3, rellena la columna 'Sin agente' con estimaciones retroactivas de tu primer mes.

Semana	Sin agente — tareas completadas autónomamente (%)	Con agente — tareas completadas autónomamente (%)	Comentarios / Incidencias
Dia 1			
Dia 2			
Dia 5			
Semana 2			
Semana 3			

Tiempo estimado para autonomía completa SIN agente: _____ semanas

CON agente: _____ semanas

4. Feedback Cualitativo

Responde brevemente a las siguientes preguntas abiertas. No hay respuestas correctas o incorrectas.

Pregunta	Respuesta abierta
¿Qué tarea hace el agente mejor de lo esperado?	
¿Qué tarea hace peor o debería mejorar?	
¿Hubo alguna respuesta incorrecta destacable?	
¿El agente cambió tu flujo de trabajo habitual?	
¿Qué funcionalidad añadirías?	
Comentarios adicionales	

5. Casos de Uso Destacados

Describe brevemente 2-3 situaciones concretas en las que el agente haya sido especialmente útil o haya fallado de forma notable. Incluye la consulta realizada y el resultado obtenido.

Caso 1

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 2

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 3

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

6. Firma y Fecha

Firma del ingeniero	Fecha de entrega
[redacted]	

M3-Agent IA: Evaluación de Impacto

Salicru S.A. · Departamento de I + D

TFG UAB 2026 — Hèctor Marchiori Soms

Nombre del ingeniero	[redacted]
Departamento	Desenvolupament
Fecha inicio evaluación	
Fecha fin evaluación	
Versión del agente	V1.0 - Salicru Cloud

Este documento recoge la evaluación del impacto del agente M3-Agent-Electronics en las tareas cotidianas del departamento de I + D de Salicru. Por favor, rellena cada sección con la mayor precisión posible. Los datos son anónimos y se utilizarán exclusivamente para mejorar el sistema.

1. Medición de Tiempos por Tarea

Registra el tiempo (en minutos) que tardas en completar cada tarea, antes y después de usar el agente. Indica las observaciones relevantes y firma con tus iniciales.

Tarea	Tiempo sin agente (min)	Tiempo con agente (min)	Mejora (%)	Observaciones	Firmado
Localizar código M3 de un componente	5/10	1		Depende del componente	
Buscar referencias de fabricante activas					
Consultar stock por almacén	3	1	66%		
Detectar componentes obsoletos					
Buscar equivalente externo (DigiKey)					
Consultar datasheet de un componente	5	2	60%		
Responder a pregunta técnica de un proveedor					
Preparar lista de componentes para pedido					

Tiempo total antes del agente (suma): _____ min

Tiempo total con el agente (suma): _____ min

Mejora global: _____ %

2. Quality of Experience (QoE)

Evalúa cada dimensión marcando la casilla correspondiente. 1 = muy deficiente, 10 = excelente.

Rellena esta sección después de al menos 2 semanas de uso del agente.

Dimensión evaluada	1	2	3	4	5	6	7	8	9	10
Precisión de las respuestas	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Velocidad de respuesta	☐	☐	☐	☐	☐	☐	☒	☐	☐	☐
Facilidad de uso (interfaz Chainlit)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒
Comprensión de preguntas en lenguaje natural	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐
Utilidad de la información de DigiKey	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐
Calidad de la detección de obsolescencias	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐
Información de datasheets (RAG)	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐
Confianza en los datos devueltos	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Reducción de carga de trabajo percibida	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒
Recomendarías el agente a un compañero	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒

Puntuación media total: 8,9 / 10

3. Curva de Aprendizaje

Esta sección está pensada para ingenieros que se incorporan al laboratorio sin experiencia previa en M3. Registra semanalmente qué porcentaje de tareas M3 eres capaz de completar de forma autónoma (sin ayuda de un compañero), en dos escenarios: sin el agente y con el agente.

Si ya conoces M3, rellena la columna 'Sin agente' con estimaciones retroactivas de tu primer mes.

Semana	Sin agente — tareas completadas autónomamente (%)	Con agente — tareas completadas autónomamente (%)	Comentarios / Incidencias
Dia 1	10	90	
Dia 2	15	90	
Dia 5	25	90	
Semana 2	50	90	
Semana 3	75	90	

Tiempo estimado para autonomía completa SIN agente: 3/4 semanas

CON agente: 1 semanas

4. Feedback Cualitativo

Responde brevemente a las siguientes preguntas abiertas. No hay respuestas correctas o incorrectas.

Pregunta	Respuesta abierta
¿Qué tarea hace el agente mejor de lo esperado?	Ver información de datasheet o stock
¿Qué tarea hace peor o debería mejorar?	Mejorar la rapidez de las respuestas
¿Hubo alguna respuesta incorrecta destacable?	
¿El agente cambió tu flujo de trabajo habitual?	Me agiliza a la hora de buscar información sobre un componente, por lo que si
¿Qué funcionalidad añadirías?	Le añadiría que según la persona que lo utilice, tuviera acceso o no a diferentes documentos. Que fuera mas personalizado por departamentos
Comentarios adicionales	Aunque mi trabajo está mas centrado en el diseño de PCB y no utilizo M3 de forma habitual, si que me ha permitido agilizar significativamente varias de mis tareas diarias

5. Casos de Uso Destacados

Describe brevemente 2-3 situaciones concretas en las que el agente haya sido especialmente útil o haya fallado de forma notable. Incluye la consulta realizada y el resultado obtenido.

Caso 1

Consulta realizada	Busqueda de alternativas de un componente por falta de stock
Resultado del agente	Me ofrecio diferentes alternativas, priorizando por la orden que yo le daba
¿Correcto?	☒ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	9

Caso 2

Consulta realizada	Necesitaba saber información sobre un ic
Resultado del agente	Me dio los datos que necesitaba y seguidamente, me resolvió dudas que tenia sobre ello
¿Correcto?	☒ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	9

Caso 3

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

6. Firma y Fecha

Firma del ingeniero	Fecha de entrega

M3-Agent IA: Evaluación de Impacto

Salicru S.A. · Departamento de I + D

TFG UAB 2026 — Hèctor Marchiori Soms

Nombre del ingeniero	[redacted]
Departamento	
Fecha inicio evaluación	
Fecha fin evaluación	
Versión del agente	V1.0 - Salicru Cloud

Este documento recoge la evaluación del impacto del agente M3-Agent-Electronics en las tareas cotidianas del departamento de I + D de Salicru. Por favor, rellena cada sección con la mayor precisión posible. Los datos son anónimos y se utilizarán exclusivamente para mejorar el sistema.

1. Medición de Tiempos por Tarea

Registra el tiempo (en minutos) que tardas en completar cada tarea, antes y después de usar el agente. Indica las observaciones relevantes y firma con tus iniciales.

Tarea	Tiempo sin agente (min)	Tiempo con agente (min)	Mejora (%)	Observaciones	Firmado
Localizar código M3 de un componente					
Buscar referencias de fabricante activas					
Consultar stock por almacén					
Detectar componentes obsoletos					
Buscar equivalente externo (DigiKey)					
Consultar datasheet de un componente					
Responder a pregunta técnica de un proveedor					
Preparar lista de componentes para pedido					

Tiempo total antes del agente (suma): _____ min

Tiempo total con el agente (suma): _____ min

Mejora global: _____ %

2. Quality of Experience (QoE)

Evalúa cada dimensión marcando la casilla correspondiente. 1 = muy deficiente, 10 = excelente.

Rellena esta sección después de al menos 2 semanas de uso del agente.

Dimensión evaluada	1	2	3	4	5	6	7	8	9	10
Precisión de las respuestas	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Velocidad de respuesta	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Facilidad de uso (interfaz Chainlit)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Comprensión de preguntas en lenguaje natural	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Utilidad de la información de DigiKey	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Calidad de la detección de obsolescencias	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Información de datasheets (RAG)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Confianza en los datos devueltos	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Reducción de carga de trabajo percibida	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Recomendarías el agente a un compañero	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Puntuación media total: _____ / 10

3. Curva de Aprendizaje

Esta sección está pensada para ingenieros que se incorporan al laboratorio sin experiencia previa en M3. Registra semanalmente qué porcentaje de tareas M3 eres capaz de completar de forma autónoma (sin ayuda de un compañero), en dos escenarios: sin el agente y con el agente.

Si ya conoces M3, rellena la columna 'Sin agente' con estimaciones retroactivas de tu primer mes.

Semana	Sin agente — tareas completadas autónomamente (%)	Con agente — tareas completadas autónomamente (%)	Comentarios / Incidencias
Dia 1			
Dia 2			
Dia 5			
Semana 2			
Semana 3			

Tiempo estimado para autonomía completa SIN agente: _____ semanas

CON agente: _____ semanas

4. Feedback Cualitativo

Responde brevemente a las siguientes preguntas abiertas. No hay respuestas correctas o incorrectas.

Pregunta	Respuesta abierta
¿Qué tarea hace el agente mejor de lo esperado?	
¿Qué tarea hace peor o debería mejorar?	
¿Hubo alguna respuesta incorrecta destacable?	
¿El agente cambió tu flujo de trabajo habitual?	
¿Qué funcionalidad añadirías?	
Comentarios adicionales	

5. Casos de Uso Destacados

Describe brevemente 2-3 situaciones concretas en las que el agente haya sido especialmente útil o haya fallado de forma notable. Incluye la consulta realizada y el resultado obtenido.

Caso 1

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 2

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 3

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

6. Firma y Fecha

Firma del ingeniero	Fecha de entrega
[redacted]	

M3-Agent IA: Evaluación de Impacto

Salicru S.A. · Departamento de I + D

TFG UAB 2026 — Hèctor Marchiori Soms

Nombre del ingeniero	[redacted]
Departamento	Desenvolupament → Hardware
Fecha inicio evaluación	22/09/2026
Fecha fin evaluación	16/06/2026
Versión del agente	V1.0 - Salicru Cloud

Este documento recoge la evaluación del impacto del agente M3-Agent-Electronics en las tareas cotidianas del departamento de I + D de Salicru. Por favor, rellena cada sección con la mayor precisión posible. Los datos son anónimos y se utilizarán exclusivamente para mejorar el sistema.

1. Medición de Tiempos por Tarea

Registra el tiempo (en minutos) que tardas en completar cada tarea, antes y después de usar el agente. Indica las observaciones relevantes y firma con tus iniciales.

Tarea	Tiempo sin agente (min)	Tiempo con agente (min)	Mejora (%)	Observaciones	Firmado
Localizar código M3 de un componente	10 min	2 min	$(1-\frac{10}{2})\cdot 100$		R
Buscar referencias de fabricante activas	No utilitzat				
Consultar stock por almacén	2 min	1 min	$(1-\frac{1}{2})\cdot 100$		R
Detectar componentes obsoletos	No utilitzat				
Buscar equivalente externo (DigiKey)	No utilitzat				
Consultar datasheet de un componente	No utilitzat				
Responder a pregunta técnica de un proveedor	No utilitzat				
Preparar lista de componentes para pedido	No utilitzat				

Tiempo total antes del agente (suma): _____ min

Tiempo total con el agente (suma): _____ min

Mejora global: _____ %

2. Quality of Experience (QoE)

Evalúa cada dimensión marcando la casilla correspondiente. 1 = muy deficiente, 10 = excelente.

Rellena esta sección después de al menos 2 semanas de uso del agente.

Dimensión evaluada	1	2	3	4	5	6	7	8	9	10
Precisión de las respuestas	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Velocidad de respuesta	☐	☐	☐	☐	☒	☐	☐	☐	☐	☐
Facilidad de uso (interfaz Chainlit)	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Comprensión de preguntas en lenguaje natural	☐	☐	☐	☐	☐	☐	☐	☐	☒	☐
Utilidad de la información de DigiKey	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Calidad de la detección de obsolescencias	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Información de datasheets (RAG)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Confianza en los datos devueltos	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Reducción de carga de trabajo percibida	☐	☐	☐	☐	☐	☐	☐	☒	☐	☐
Recomendarías el agente a un compañero	☐	☐	☐	☐	☐	☐	☐	☐	☐	☒

Puntuación media total: _____ / 10

3. Curva de Aprendizaje

Esta sección está pensada para ingenieros que se incorporan al laboratorio sin experiencia previa en M3. Registra semanalmente qué porcentaje de tareas M3 eres capaz de completar de forma autónoma (sin ayuda de un compañero), en dos escenarios: sin el agente y con el agente.

Si ya conoces M3, rellena la columna 'Sin agente' con estimaciones retroactivas de tu primer mes.

Semana	Sin agente — tareas completadas autónomamente (%)	Con agente — tareas completadas autónomamente (%)	Comentarios / Incidencias
Dia 1	0	65%	
Dia 2	10%	65%	
Dia 5	20%	65%	
Semana 2	40%	65%	
Semana 3	40%	65%	

Tiempo estimado para autonomía completa SIN agente: _____ semanas

CON agente: _____ semanas

4. Feedback Cualitativo

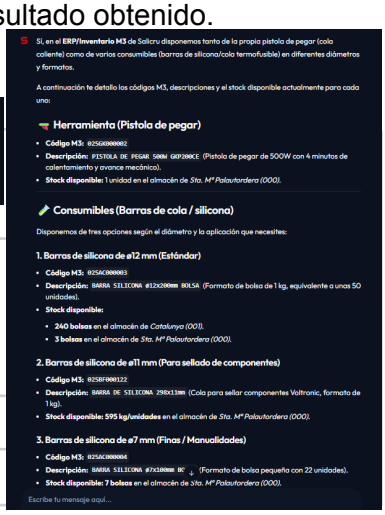
Responde brevemente a las siguientes preguntas abiertas. No hay respuestas correctas o incorrectas.

Pregunta	Respuesta abierta
¿Qué tarea hace el agente mejor de lo esperado?	Buscar codigos a partir de una descripcion
¿Qué tarea hace peor o debería mejorar?	Quan es donen molts parametres de un component i es diu que busqui un que tingui con a minim X valor.
¿Hubo alguna respuesta incorrecta destacable?	Si, quan hi ha un mal entes en el que demanes i el que busca
¿El agente cambió tu flujo de trabajo habitual?	Una mica
¿Qué funcionalidad añadirías?	Integrarlo amb mes automatitzacions de l'empresa
Comentarios adicionales	

5. Casos de Uso Destacados

Describe brevemente 2-3 situaciones concretas en las que el agente haya sido especialmente útil o haya fallado de forma notable. Incluye la consulta realizada y el resultado obtenido.

Caso 1

Consulta realizada	necesito que me digas si hay algun consumible de la cola de la tipica pistola de cola caliente, y su respectivo codigo
Resultado del agente	
¿Correcto?	☑ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	8

Caso 2

Consulta realizada	necesito saber quin es el codi dels topes que es fixen en el carril din perque els elements que hi ha en el carril din no es desplacin lateralment
Resultado del agente	
¿Correcto?	☑ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	8

Caso 3

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

6. Firma y Fecha

Firma del ingeniero	Fecha de entrega
	16/06/2026

M3-Agent IA: Evaluación de Impacto

Salicru S.A. · Departamento de I + D

TFG UAB 2026 — Hèctor Marchiori Soms

Nombre del ingeniero	[redacted]
Departamento	
Fecha inicio evaluación	
Fecha fin evaluación	
Versión del agente	V1.0 - Salicru Cloud

Este documento recoge la evaluación del impacto del agente M3-Agent-Electronics en las tareas cotidianas del departamento de I + D de Salicru. Por favor, rellena cada sección con la mayor precisión posible. Los datos son anónimos y se utilizarán exclusivamente para mejorar el sistema.

1. Medición de Tiempos por Tarea

Registra el tiempo (en minutos) que tardas en completar cada tarea, antes y después de usar el agente. Indica las observaciones relevantes y firma con tus iniciales.

Tarea	Tiempo sin agente (min)	Tiempo con agente (min)	Mejora (%)	Observaciones	Firmado
Localizar código M3 de un componente					
Buscar referencias de fabricante activas					
Consultar stock por almacén					
Detectar componentes obsoletos					
Buscar equivalente externo (DigiKey)					
Consultar datasheet de un componente					
Responder a pregunta técnica de un proveedor					
Preparar lista de componentes para pedido					

Tiempo total antes del agente (suma): _____ min

Tiempo total con el agente (suma): _____ min

Mejora global: _____ %

2. Quality of Experience (QoE)

Evalúa cada dimensión marcando la casilla correspondiente. 1 = muy deficiente, 10 = excelente.

Rellena esta sección después de al menos 2 semanas de uso del agente.

Dimensión evaluada	1	2	3	4	5	6	7	8	9	10
Precisión de las respuestas	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Velocidad de respuesta	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Facilidad de uso (interfaz Chainlit)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Comprensión de preguntas en lenguaje natural	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Utilidad de la información de DigiKey	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Calidad de la detección de obsolescencias	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Información de datasheets (RAG)	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Confianza en los datos devueltos	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Reducción de carga de trabajo percibida	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐
Recomendarías el agente a un compañero	☐	☐	☐	☐	☐	☐	☐	☐	☐	☐

Puntuación media total: _____ / 10

3. Curva de Aprendizaje

Esta sección está pensada para ingenieros que se incorporan al laboratorio sin experiencia previa en M3. Registra semanalmente qué porcentaje de tareas M3 eres capaz de completar de forma autónoma (sin ayuda de un compañero), en dos escenarios: sin el agente y con el agente.

Si ya conoces M3, rellena la columna 'Sin agente' con estimaciones retroactivas de tu primer mes.

Semana	Sin agente — tareas completadas autónomamente (%)	Con agente — tareas completadas autónomamente (%)	Comentarios / Incidencias
Dia 1			
Dia 2			
Dia 5			
Semana 2			
Semana 3			

Tiempo estimado para autonomía completa SIN agente: _____ semanas

CON agente: _____ semanas

4. Feedback Cualitativo

Responde brevemente a las siguientes preguntas abiertas. No hay respuestas correctas o incorrectas.

Pregunta	Respuesta abierta
¿Qué tarea hace el agente mejor de lo esperado?	
¿Qué tarea hace peor o debería mejorar?	
¿Hubo alguna respuesta incorrecta destacable?	
¿El agente cambió tu flujo de trabajo habitual?	
¿Qué funcionalidad añadirías?	
Comentarios adicionales	

5. Casos de Uso Destacados

Describe brevemente 2-3 situaciones concretas en las que el agente haya sido especialmente útil o haya fallado de forma notable. Incluye la consulta realizada y el resultado obtenido.

Caso 1

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 2

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

Caso 3

Consulta realizada	
Resultado del agente	
¿Correcto?	☐ Sí ☐ Parcialmente ☐ No
Valoración (1-10)	

6. Firma y Fecha

Firma del ingeniero	Fecha de entrega
[redacted]	

Appendix B — System Demo Video

A recorded demonstration of M3-Agent IA in production is available at the following URL:

`https://youtu.be/axEpd5NYz5A?si=NXNT596nxIPWsPCj`

The video shows the agent responding to six representative queries covering component lookup, manufacturer reference retrieval, stock consultation, obsolescence detection, DigiKey market search, and datasheet parameter extraction.

Appendix C — Source Code Repository

The full source code of M3-Agent IA, including all tool implementations, the schema context document, deployment configuration, and the local PostgreSQL replica setup scripts, is publicly available at:

`https://github.com/SomsJr/TFG-M3-Agent`