



**SMART MANUFACTURING EXECUTION SYSTEM:
REVOLUTIONIZING INDUSTRY WITH IOT AND AI
(MODULE: CHATBOT)**

UNDERGRADUATE THESIS

Submitted as one of the requirements to obtain

Sarjana Komputer

By:

Daniel Maheswara Wicaksono

001202200055

FACULTY OF COMPUTER SCIENCE

INFORMATICS STUDY PROGRAM

CIKARANG

August, 2025

SIMILARITY INDEX REPORT



Daniel Maheswara Wicaksono

latest_individual_report

Thesis Computing

Document Details

Submission ID
trn:oid::3618:112749907

Submission Date
Sep 17, 2025, 7:02 PM GMT+7

Download Date
Sep 17, 2025, 7:25 PM GMT+7

File Name
latest_individual_report.pdf

File Size
5.0 MB

102 Pages

10,750 Words

66,186 Characters



4% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Filtered from the Report

- Bibliography
- Quoted Text

Match Groups

-  **42 Not Cited or Quoted** 4%
Matches with neither in-text citation nor quotation marks
-  **1 Missing Quotations** 0%
Matches that are still very similar to source material
-  **0 Missing Citation** 0%
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted** 0%
Matches with in-text citation present, but no quotation marks

Top Sources

- 2%  Internet sources
- 2%  Publications
- 3%  Submitted works (Student Papers)

ChatGPT Zero Results

Daniel Maheswara Wicaksono

latest_individual_report

 Thesis Computing

Document Details

Submission ID
trn:oid::3618:112749907

Submission Date
Sep 17, 2025, 7:02 PM GMT+7

Download Date
Sep 19, 2025, 1:16 PM GMT+7

File Name
latest_individual_report.pdf

File Size
5.0 MB

102 Pages

10,750 Words

66,186 Characters

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer

Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (i.e., our AI models may produce either false positive results or false negative results), so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?

The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?

Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.



ABSTRACT

Manufacturing Execution Systems (MES) are essential for real-time process monitoring and control in Industry 4.0 environments. Despite advanced IoT integration, maintenance technicians still depend on manual document searches through operator manuals, maintenance logs, and error-code tables leading to prolonged troubleshooting, inconsistent results, and expensive downtime. To address these challenges, we present a fully on-premise, Retrieval-Augmented Generation (RAG) chatbot module that delivers contextually relevant procedural guidance directly within the MES. Our system preprocesses multimodal technical documents including PDFs, tables, and schematics using PyMuPDF, Camelot, Tesseract OCR, and BLIP image captioning, and partitions the content into semantically coherent, overlapping text chunks. A hybrid retrieval engine fuses BM25 keyword matching with dense-vector similarity search (via FAISS), and the top candidates are passed to a locally hosted Qwen3:1.7b language model through LangChain for answer generation. Importantly, all components run entirely on-premise without any external API calls, preserving data privacy and eliminating subscription costs. Evaluation on real-world machine manuals demonstrates sub-second average response latency and over 90 % retrieval accuracy, significantly reducing reliance on printed manuals and accelerating maintenance tasks. Future work includes support for multilingual manuals and session-memory for follow-up queries.

Keywords: *Retrieval-Augmented Generation, Hybrid Search, Ollama, LangChain, Chatbot, Knowledge Ingestion, RAG, Manufacturing, Industry*

TABLE OF CONTENTS

Smart Manufacturing Execution System: Revolutionizing Industry with IoT and AI (MODULE: CHATBOT).....	1
Copyright by Daniel Maheswara Wicaksono 2025.....	2
Smart Manufacturing Execution System: Revolutionizing Industry with IoT and AI (Module: Chatbot)	2
STATEMENT OF ORIGINALITY	Error! Bookmark not defined.
SCIENTIFIC PUBLICATION APPROVAL FOR ACADEMIC INTEREST ...	Error! Bookmark not defined.
ADVISOR APPROVAL FOR PUBLICATION.....	iv
SIMILARITY INDEX REPORT	v
ChatGPT Zero Results.....	vii
ABSTRACT	ix
DEDICATION	x
ACKNOWLEDGEMENTS	xi
TABLE OF CONTENTS.....	xii
TABLE OF FIGURES.....	xv
TABLE OF TABLE	xviii
CHAPTER I INTRODUCTION	16
1.1 Background of the Problem.....	16
1.2 Problem Statement	18
1.3 Objectives	19
1.4 Problem from a Customer Perspective.....	20
CHAPTER II SPECIFICATION.....	21
2.1 Existing Business Process	21
2.2 Proposed Business Process.....	22

2.3 Global Description Of the Product	23
2.3.1 Main Functionality	23
2.3.2 User Characteristics.....	24
2.3.3 Constraints	24
2.3.4 Product Development Environment	25
2.3.5 Product Operational Environment	25
2.3. REQUIREMENT ANALYSIS.....	26
CHAPTER III DESIGN & METHODOLOGY	27
3.1 Data Collection and Preprocessing	27
3.2 Knowledge Indexing.....	27
3.3 Hybrid Retrieval Pipeline	28
3.4 Response Generation.....	30
CHAPTER IV SYSTEM ARCHITECTURE	31
4.1 System Overview.....	31
4.2 Document Preprocessing.....	31
4.3 Structured Knowledge & Indexing	31
4.4 Hybrid Retrieval Engine.....	32
4.5 Query Processing & Response	32
4.6 Real-Time Index Management	33
CHAPTER V DESIGN IMPLEMENTATION	34
5.1 User Interface Of the Chabot.....	34
5.2 User Manual Guide	36
5.2.1 How to Create the Chabot	36
5.2.2 How to use Chatbot in the MES System	41
5.3 Full Code Implementation.....	44
5.3.1 chatbot_get_knowledge.py:	44

5.3.2 chatbot_app.py:	54
CHAPTER VI TESTING & EVALUATION	68
6.1 Testing Setup	68
6.2 Results Testing	87
6.3 Analysis Of The Problem	90
6.4 Causes of Inconsistent Responses.....	90
6.5 Impact of Format Stability	91
6.6 Functional Testing	92
6.6.1 Testing Results of Every Function in the Specification	92
6.7 Testing Other Specifications	93
CHAPTER VI CONCLUSION.....	94
REFERENCES	96

TABLE OF FIGURES

Figure 2.1 Flow Of Current Bussiness Process.....	22
Figure 2.2 Flow Of The Proposed Solution	23
Figure 2.3 Chatbot Functional requirement	26
Figure 2.4 MES System Functional Requirements	26
Figure 3.1 Formula of The Rangking Function of Okapi BM25.....	27
Figure 3.2 Dataflow Diagram Of The Architecture.....	29
Figure 5.1 Chat Page UI For Admin With Manage Button.....	34
Figure 5.2 Manage Page UI For Admin Only.....	35
Figure 5.3 Chat Page UI For Non-Admin Without Manage Button.....	35
Figure 5.4 Proof of Ollama Installation Successful.....	37
Figure 5.5 Ollama Serve Running Succesfully.....	37
Figure 5.6 Pulling Selected Ollama Model.....	38
Figure 5.7 Pulling The Embedding Model.....	38
Figure 5.8 List Of All Pulled Model.....	39
Figure 5.9 How To Place The PDF File.....	40
Figure 5.10 Result After Indexing The Knowledge.....	41
Figure 5.11 Upload and Deleting PDF code.....	42
Figure 5.12 Calling indexing functionality Code.....	43
Figure 5.13 Module Imports and Logging Setup.....	45
Figure 5.14 Global Configuration and NLP Setup.....	45
Figure 5.15 Class Definition: BM25 (Initialization and Fitting)	46
Figure 5.16 BM25 Scoring and Searching.....	47
Figure 5.17 HybridSearcher for Initialization & Search.....	47

Figure 5.18 HybridSearcher for Pickle Serialization Hooks.....	48
Figure 5.19 build_index for Setup & Timer.....	48
Figure 5.20 build_index for spaCy & BLIP Initialization.....	49
Figure 5.21 build_index for PDF Discovery & Extraction.....	50
Figure 5.22 build_index for Chunking Documents.....	50
Figure 5.23 build_index for BM25 Fitting & Persistence.....	51
Figure 5.24 build_index for Embedding & FAISS Build.....	51
Figure 5.25 build_index for HybridSearcher & Atomic Swap.....	52
Figure 5.26 Module imports and environment setup.....	53
Figure 5.27 Secondary imports and thread primitives.....	53
Figure 5.28 Global paths and random seed configuration.....	54
Figure 5.29 Logging setup and NLTK download.....	54
Figure 5.30 Application configuration constants.....	55
Figure 5.31 Chat log bootstrap and helper to record exchanges.....	55
Figure 5.32 Function normalize_vectors for Faiss index.....	56
Figure 5.33 Function warm_up_model for embeddings and LLM.....	56
Figure 5.34 Simple text helpers get_synonyms and preprocess_text.....	57
Figure 5.35 Function reload_searcher to hot-swap indexes.....	57
Figure 5.36 Class BM25 with tokenization and scoring.....	58
Figure 5.37 Class HybridSearcher merged from BM25 and vector.....	59
Figure 5.38 Loading embeddings and FAISS index at startup.....	60
Figure 5.39 Prompt template and LLM chain setup.....	61
Figure 5.40 Function preprocess_answer to clean output.....	61
Figure 5.41 Memory persistence and Flask app creation.....	61
Figure 5.42 Core RAG runner function.....	62

Figure 5.43 Debug retrieval endpoint for score inspection.....	62
Figure 5.44 Chat user interface endpoints.....	63
Figure 5.45 Chat API endpoint with logging.....	64
Figure 5.46 Management endpoints for PDF and memory.....	64
Figure 5.47 Rebuild reload and cancel indexing endpoints.....	65
Figure 5.48 Index status endpoint and main guard.....	66

TABLE OF TABLE

Table 2.1 List Of The Role Capabilities	24
Table 6.1 Overview Results Testing in file evaluation.xlsx.....	67
Table 6.2 The Metrics Results.....	86
Table 6.3 Testing Result.....	90
Table 6.4 Testing Non Functionalites.....	91