// TMG SECURITY RESEARCH — CHECKLIST

AI/LLM Application Security Checklist

A practical, pre-deployment checklist for AI and LLM applications — what to verify across inputs, retrieval, tools, output, and high-impact actions before a feature ships.

TMG Security Research Team

Version 1.0

Published 08 September 2026

Condensed from TMG's full AI/LLM security analysis

What this checklist is

Most reported LLM security incidents are not model failures — they are ordinary application security failures in a system that happens to contain a model. This checklist turns that finding into fourteen checks across seven categories, covering the request path from input through retrieval, tools, output, and logging.

> This is TMG Security's own condensed, practical reading of the checklist published in TMG's full AI/LLM security analysis. See References for the primary TMG source and the official external frameworks.

What this checklist is not

- Not an official OWASP, NIST, or MITRE checklist. TMG does not claim their endorsement, affiliation, or authorship.
- Not a certification — completing it does not constitute a certified or industry-standard assessment.
- Not a multi-agent security framework, an autonomous-agent risk classification framework, or a memory-poisoning framework. See TMG-ARC for that.
- Not a source of new statistics — this checklist contains no new CWE/CVE figures or external data.

AI/LLM application security checklist

Categories and check wording are condensed from TMG's full AI/LLM security analysis. The "why it matters" column is TMG's own condensed reasoning, not a quotation from any external standard.

Category	Check	Why it matters
Input & Context	Map every input the model can read, including retrieved and machine-generated content	Every hop in the request path is a trust transition — you cannot secure what you have not mapped.
Input & Context	Label which of those sources are attacker-influenceable	Where anyone could have written the content and the model can call tools after reading it, that's an untrusted-input-to-privileged-action path.
Retrieval / RAG	Confirm retrieval is filtered by the requesting user's entitlements at query time	Retrieval that can't answer "which documents can this user match against?" before the query runs isn't authorization. It's search.
System Instructions	Confirm system instructions contain no credentials, internal endpoints, or business rules that would be damaging if disclosed	Anything placed in reachable context that a user isn't entitled to see, the model will summarise and surface — including a system prompt.
System Instructions	Confirm error paths don't surface stack traces or internal identifiers through the model's reply	The model doesn't distinguish an internal error from any other content it can read.
Tool Authorization	Inventory every tool the model can invoke, in every conversation state	A model can only do what its tools let it do — an incomplete inventory is an incomplete threat model.
Tool Authorization	Review each tool's permissions independently of the model	An over-broad tool is a finding the model's own behaviour will never reveal.
Tool Authorization	Test authorization at the tool and API layer directly, bypassing the model	The model should not be the authorization layer — authorization must be enforced, and tested, in the tool layer.
Tool Authorization	Apply least privilege to the tool layer, then re-test with a low-privilege account	Tools are APIs; the same object-level authorization discipline applies, proven only by re-testing under low privilege.
Output Handling	Validate and encode model output before it is rendered or executed	Model output is attacker-influenceable content, not trusted markup.
High-Impact Actions	Require explicit confirmation for irreversible or high-value actions	Confirmation only works when the user can evaluate the specific action and its parameters.
High-Impact Actions	Confirm tool responses return only the fields the model needs, not full objects	Over-returning data hands more than the request required to the model and anything reading its output.
High-Impact Actions	Confirm conversation memory cannot persist one user's data into another user's context	Memory crossing user boundaries turns a single-session mistake into a standing disclosure.
Logging & Identity	Log tool invocations with the requesting identity, not the service identity	Logging the service identity instead of the user's makes incident reconstruction, and accountability, impossible.

Rows 11–14 (System Instructions and two High-Impact Action rows) are TMG's reorganization of disclosure causes and controls already set out in TMG's underlying analysis — not new findings or external facts.

How to use this checklist

The model influences behaviour. The application determines consequence. Testing an LLM feature in isolation produces findings about the model; testing the system produces findings about the business. Run each row against the actual system, not only through the chat interface — the interesting findings are behind the tool boundary, and reaching them requires reading the integration code, not just talking to the model.

Practical testing guidance

Assessments that only exercise the chat surface tend to under-report. Exercise the tool and API layer directly, with a low-privilege account, and confirm authorization and data-handling behaviour independently of what the model says or refuses to do. Where a check depends on identity, confirm which identity actually reaches the downstream system.

Scope and TMG-ARC

> This checklist covers LLM applications generally, including those with tool or function-calling. It is not a multi-agent security framework, an autonomous-agent risk classification framework, a memory-poisoning framework, a formal certification checklist, or an official OWASP, NIST, or MITRE publication.
>
> For deeper agentic and autonomous-agent risk classification, see TMG's Agentic AI Risk Classification Framework (TMG-ARC). This checklist and TMG-ARC are independent; neither depends on the other.

Attribution and limitations

- This is TMG Security's own practical checklist, condensed from TMG's published AI/LLM security analysis — not an official OWASP, NIST, or MITRE checklist.
- Not a certification. Does not cover authentication, model/API configuration, rate limiting, or third-party integration risk.
- Memory is covered only by one row (High-Impact Actions). Memory poisoning, autonomous planning, and multi-agent communication are out of scope — see TMG-ARC.
- Contains no new statistics, CVE/CWE figures, or external data beyond TMG's underlying analysis.

References

TMG SOURCE

- tmgsec.com/resources/ai-llm-security-vulnerabilities/ (full analysis)

OFFICIAL EXTERNAL SOURCES

- OWASP Top 10 for Large Language Model Applications
- OWASP GenAI Security Project
- MITRE ATLAS
- NIST AI Risk Management Framework

Relevant TMG Security links

- AI / LLM Security Testing — tmgsec.com/ai-llm-security/
- API Security Testing — tmgsec.com/api-security/
- Threat Modeling — tmgsec.com/threat-modeling/

> **This is TMG Security's own practical checklist, not an OWASP, NIST, or MITRE publication.** TMG does not create, govern, or claim authorship of those frameworks, and is not affiliated with or endorsed by them.