omada

WHITE PAPER

PROMISE TO MAP

A Practical Approach to Threat Modeling Artificial Intelligence and Large Language Model Systems for Digital Healthcare and Beyond

Lead Authors

William Dougherty, MSIT, CISSP, CISM, CCSP
Chief Information Security Officer

Patrick Curry, PhD
Vice President of Compliance

Lucia Savage, JD
Chief Privacy and Regulatory Officer

Contributing Authors

Terry Miller, MS Predictive Analytics
Vice President AI & Machine Learning

Michael Capizzi, MS Human Language
Staff Applied AI Engineer

Marty Lynch, MS Tech Management
Sr Manager, Applied Data Science

Introduction

In 2019 Omada Health published the **INCLUDES NO DIRT** threat model to guide security and engineering teams on how to evaluate security, privacy, and compliance risks to any system, including new applications and new vendors. New artificial intelligence (AI) capabilities, and especially large language models (LLMs) have expanded the risks that must be modeled when the system includes AI features or functions. Omada has newly developed the PROMISE TO MAP threat model to address the additional complexities that AI systems introduce. This AI threat model is intended to expand on the INCLUDES NO DIRT model to include threats that are AI-specific. The PROMISE TO MAP threat model includes directly and by reference all risks previously identified in the INCLUDES NO DIRT model.

The authors of this model were specifically focused on building AI systems to use in a healthcare organization. We believe, however, this model is applicable to most any other business; especially other regulated industries. AI systems introduce new complexities in regulated environments. Organizations used to deterministic software and human actions must account for the risk of probabilistic software with or without humans in the process. Defining the thresholds for what is considered an error, and defining acceptable error rates and new quality assurance processes must be established.

This model relies heavily on concepts from the OWASP LLM Top 10, NVIDIA's NeMo Guardrail framework, LangChain's Development Framework, NIST AI-600-1, and numerous other industry sources. Our intent is to simplify these concepts and make them immediately actionable for risk assessing teams. As with the INCLUDES NO DIRT MODEL, we've included a simplified questionnaire worksheet at the end of this paper that teams can use to begin assessing AI systems. We encourage practitioners to modify, extend, and use this model to suit your own unique needs.

Threat Model Taxonomy

The foundation of Omada's threat model approach is to build a common language between risk assessing teams and engineering teams. For this reason, most of this threat model document is devoted to providing definitions. By agreeing to a common taxonomy, threat modeling teams can quickly move to analyzing risks rather than worrying about semantics. The PROMISE TO MAP model will use the same system taxonomy as INCLUDES NO DIRT, included here for referential purposes.

SYSTEM

The thing being modeled. This can be an application, business process, network, a vendor services, etc. The defining characteristic of a "system" is that the organization desires to protect it from threats.

TRUST BOUNDARY

The places in a system where principles interact. Some models also refer to attack surfaces, which are a type of trust boundary where a threat actor can interact, but trust boundaries can exist in a system beyond the attack surface.

VULNERABILITY

A weakness in a system. Vulnerabilities are things that can be exploited.

THREAT

An actor or principle. A threat can be an employee, a malicious third party, a business process, a natural occurrence, or a piece of code.

ATTACK VECTOR

The method by which a threat exploits a vulnerability.

RISK

A resulting bad outcome when a threat exploits a vulnerability in a system.

PROBABILITY

The likelihood of a risk occurring.

IMPACT

The cost of a risk occurring.

CONTROL

A feature or mitigation in a system that reduces the probability or impact of a risk.

THREAT MODELING

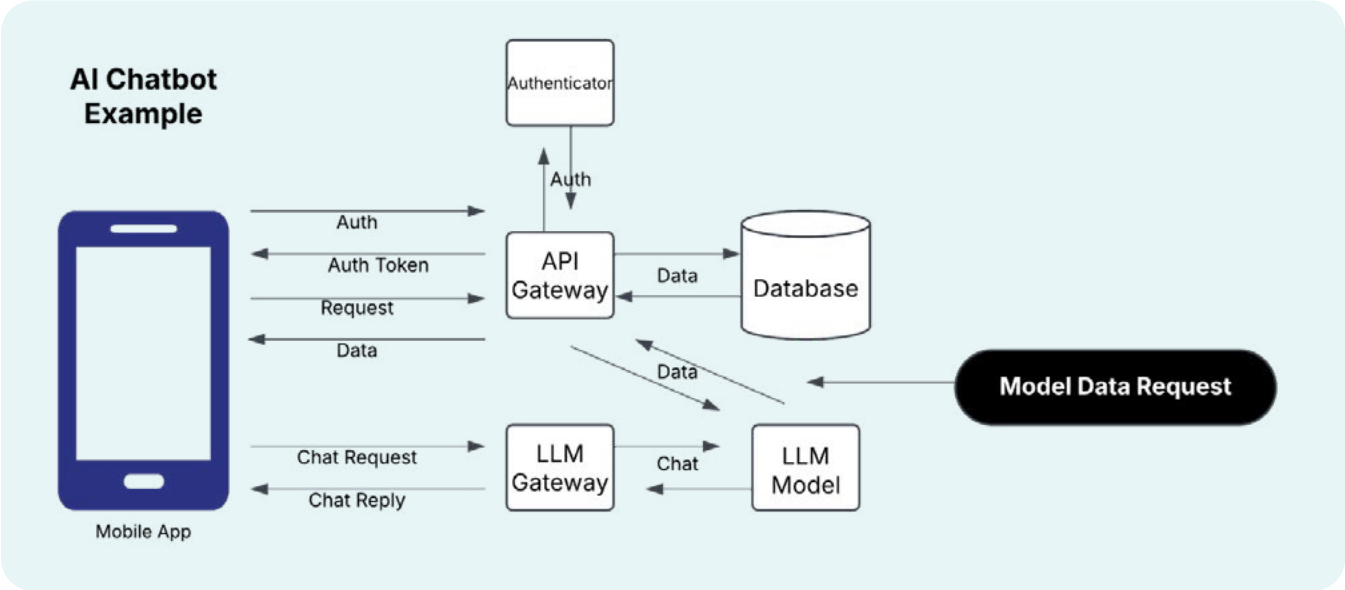
An analysis of a system's vulnerabilities, controls, and threats against a defined list of risks.

ACTION ITEMS

Tasks to be performed by the system owner or risk assessor as a result of the threat model.

System Taxonomy Example

The following is an example to illustrate the above defined terms, and how they would apply in an AI threat model. This example would cover one specific vulnerability. A complete threat model would have similar analysis for a multitude of risks.



System	AI Chatbot Application	
System Trust Boundaries	API Endpoint LLM Gateway	LLM Model Database
Threats	LLM Model	
Vulnerabilities	Unauthenticated Access to API from LLM Model	
Attack Vector	Model Data Request for data on the wrong person	
Risks	Excessive Agency, Sensitive Information Disclosure	
Probability	unknown	
Impact	High	
Controls	Access Tokens and API data access authorizations	
Action Items	Transmit Access Tokens in chat session and have LLM model use Access Tokens to authenticate data access requests from API endpoints.	

Extended AI Threat Model Summary

The INCLUDES NO DIRT original threat model was based on a list of risks, their corresponding property or goal, and the primary realm that is most concerned with that risk. The original model is repeated below for completeness. Below the original model is an extended model that is specific to AI systems. This model will be extended over time as Omada learns and grows. Each of these risks and goals are defined below after the summary.

ORIGINAL INCLUDES NO DIRT MODEL

	Risk	Property/Goal	Realm
I	**Identifiability**	Anonymity	Privacy
N	**Non-Repudiation**	Plausible Deniability	Privacy
C	**Clinical Error**	Correct Application of Clinical Standards	Compliance
L	**Linkability**	Unlinkability	Privacy
U	**Unlicensed Activity**	Proper Credentials or Licensure	Compliance
D	**Denial of Service**	Availability	Security
E	**Elevation of Privilege**	Authorization	Security
S	**Spoofing**	Authentication	Security
N	**Non-Compliant to Policy or Obligations**	Policy or Contractual Adherence	Compliance
O	**Overuse**	Minimum Necessary	Compliance
D	**Data Error**	Integrity	Security
I	**Information Disclosure**	Confidentiality	Security
R	**Repudiation**	Non-Repudiation	Security
T	**Tampering**	Integrity	Security

Extended Model - PROMISE TO MAP

AI Systems have some unique properties that are worthy of modeling separately from other systems. The risks below have been derived from the OWASP LLM Top 10, internal Omada experience, and other sources. There is some overlap to the original INCLUDES NO DIRT model. Where overlap exists, it has been included here intentionally because AI systems may include both AI and non-AI components.

PROMISE TO MAP

	Risk	Property/Goal	Realm
P	**Prompt Injection**	Validated Input	Security
R	**RAG and Embedding Weaknesses**	Verified Embeddings	Compliance
O	**Overreliance**	Validated Output	Security
M	**Misinformation**	Accurate Information	Compliance
I	**Improper Output**	Validated Output	Compliance
S	**Sensitive Information Disclosure**	Confidentiality	Privacy
E	**Excessive Agency**	Agent Authorization	Security
T	**Training Data Poisoning**	Verified Data Sets	Security
O	**Overconsumption**	Constrained Consumption	Security
M	**Model Theft**	Model Confidentiality	Security
A	**AI Supply Chain**	Verified Models and Data Sets	Security
P	**Prompt Leakage**	Prompt Confidentiality	Security

Risk Definitions

INCLUDES NO DIRT

The INCLUDES NO DIRT model included definitions for each risk, which are included here for easy reference.

INDENTIFIABILITY

Identifiability is the property of a system that lets activities be traced to a specific user. Some systems, such as an application for reporting fraud or abuse, may require an option to act anonymously. If anonymity is required, then the risk assessor must identify what controls are in place to ensure it.

NON-REPUDIATION

Non-Repudiation is the process by which it can be proven that a user performed an action. Like anonymity, some systems may require plausible deniability. The difference between anonymity and plausible deniability is subtle but important. An application that allows for anonymity may still record IP addresses, machine IDs or other metadata that can be traced back to the user. For some systems, such as a Whistleblower application, it may be required to ensure plausible deniability. In those cases, it's important to analyze not only the user features but also the metadata being recorded.

CLINICAL ERROR

In healthcare, and especially within digital health, accuracy in management, transformation, and interpretation of data is critical. This is especially true for data that concerns the health status, condition, or participation of an individual. Clinically relevant errors can occur if the system does not enforce agreed-upon clinical standards, or does not preserve information fidelity. Errors in clinical data can result in real harm for patients, and sometimes permanent harm up to and including death.

LINKABILITY

Linkability is the ability to relate two or more pieces of information. Linkability can be a risk to both anonymity and plausible deniability. In healthcare, linkability most often comes up in the context of de-identification of protected health information (PHI).

UNLICENSED ACTIVITY

Many activities in healthcare require specific licensure or certification to be legally performed, either at the entity or the person level. Failing to track licenses or the scope of practice they authorize can create a significant legal risk for a company when those rules apply. In digital health, where care is often provided nationwide, a company may need to track compliance with multi-faceted licensing requirements across different jurisdictions. For example, there may be a need to track licenses of professionals, or credentials required by customers.

DENIAL OF SERVICE

Denial of Service is any activity that impacts the Availability of a system. Availability means that the system is able to perform its tasks when required by the business.

ELEVATION OF PRIVILEGE

Elevation of Privilege occurs when a user is able to perform a function that exceeds his or her authorization. A system may apply authorizations directly to a user or to a group (role) of which the user is a member. In healthcare, a weak authorization scheme can threaten the security of the system, its compliance with the minimum necessary standard, and potentially incorrect or inappropriate provision of services. For example, a billing clerk for a lab who is supposed to only see payments due may inappropriately have access to an individual's lab test results.

SPOOFING

Spoofing is the ability for a user to pretend to be someone else. It is a risk for systems that have weak or non-existent authentication mechanisms. The required strength of an authentication mechanism depends on the system being protected and the types of data it houses. A system that stores or processes protected health information (PHI) may have different requirements than a company Intranet. 45 CFR § 164.312 of the HIPAA Security Rule establishes technical safeguards including authentication that must be implemented to protect PHI. The NIST 800-63b Digital Identity Guidelines is a good reference to authentication methods and levels.

NON-COMPLIANT TO POLICY OR OBLIGATIONS

All organizations have a wide variety of rules, regulations, internal policies, and contractual obligations to which they must adhere. As a threat assessor, it is important to identify the specific policies and obligations that apply to the system being modeled, and then review how those obligations are enforced, monitored, and audited. HIPAA requires Covered Entities and their business associates to have written policies and procedures for handling PHI, and the absence of such written policies is a threat in its own right, on top of the threats posed by lack of consistent process. In addition, there may be state-specific rules on the confidentiality of health care data or the privacy of consumer data collected through internet-based transactions). As applied to AI systems, the policies, regulations, license agreements, and contracts are rapidly evolving and new systems may require updates to policies, customer contracts, and terms of use.

OVERUSE

Overuse is a risk prominent in healthcare, although other industries may have similar obligations. 45 CFR § 164.502 of the HIPAA Privacy Rule restricts covered entities and business associates from the use or disclosure of protected health information (PHI) to the "minimum necessary"to accomplish its intended purpose except for use by or sharing with a physician or other health care professional who is providing treatment to an individual. If a system stores, processes or accesses PHI and does not have mechanisms in place to limit the use of the data to the minimum necessary when that limit is required, there may be a risk of overuse. Additionally, in cases where people provide specific consent for the use of their information regardless of regulation, it is important to understand if the risk of use outside of consent exists.

DATA ERROR

Data error is any risk to the integrity of data in the system, due to weak controls, user error, software bugs or faulty logic. Data error is generally unintentional or accidental, as opposed to intentional tampering. Systems must be evaluated on their controls to verify data integrity and correct any error identified. Consideration in testing should be paid to data transformation and the movement of data between systems.

INFORMATION DISCLOSURE

Information disclosure is any unauthorized, non-permitted, or unintended publication, leak, or loss of data that threatens the confidentiality of data held by the organization. While HIPAA allows many disclosures without requiring an individual to consent (45 CFR 164.506) and other disclosure with an individual's written "authorization," (45 CFR 164.508), for PROMISETOMAP, we are modeling the threat of unauthorized Information Disclosure. In addition to authentication and authorization controls, systems must have strong encryption, data locality, and physical security to protect confidentiality. 45 CFR § 164.312 of the HIPAA Security Rule establishes technical safeguards including data encryption at rest and in transit that may be used to protect the confidentiality of PHI.

REPUDIATION

Most systems require non-repudiation, or the ability to prove a specific user performed a specific action. Note that this is the mirror risk to a system that requires anonymity. Authentication, authorization, system logging, accurate timestamps and digital signatures can all be used to assure non-repudiation. When assessing non-repudiation controls, the assessor should also look at how long logs or other evidence are retained to make sure they match policy and obligations.

TAMPERING

Tampering is the intentional modification of the system or its data with an intent to do harm. In healthcare, tampering can impact confidentiality, integrity, availability, and clinical accuracy. Anti-tampering controls may include network security, physical security, chain of custody, change management, code review, and vulnerability assessments. The HHS Office of Civil Rights (OCR) has stated that tampering which renders PHI unavailable, such as by ransomware, is a reportable Breach.

Risk Definitions

PROMISE TO MAP

The PROMISE TO MAP extension of INCLUDES NO DIRT definitions are specific to AI systems. Because there are some overlaps to the original model, where applicable, those overlaps are included in the definition.

PROMPT INJECTION

Prompt injection occurs when user input alters the behavior or output of the AI system in unintended ways. Prompt injections can occur intentionally, such as attempts to jailbreak or bypass the model controls, or unintentionally when user input differs from what is expected. Prompt injection risks can lead to Clinical Error, Denial of Service, Elevation of Privilege, Data Error, Information Disclosure, and Tampering risks.

RAG AND EMBEDDING WEAKNESSES

Retrieval Augmented Generation (RAG) is the process of combining pre-trained models with external knowledge sources. Weaknesses in how RAG sources, embeddings, and vectors are generated, stored, or retrieved can allow for the injection of harmful content, manipulate model outputs, disclose sensitive information, and generate unintended outputs. As an example, a model providing recommendations to a clinician that is using outdated or conflicting clinical standards via RAG could produce recommendations that do not meet company standards.

OVERRELIANCE

Overreliance occurs when an AI model produces an output that is not properly verified before use, either by humans or automated systems. Overreliance is especially impactful when chaining models together, where the output of one model becomes the input of the next. Overreliance can also occur when a model is providing direct feedback to the user, and the user is unable to distinguish between proper and improper results. Overreliance impacts Clinical Error, Elevation of Privilege, Data Error and Tampering, as well as Misinformation.

MISINFORMATION

Misinformation occurs when the AI system produces false or misleading information that appears credible, such as model hallucinations. Misinformation combined with Overreliance can produce harmful effects. Misinformation can be a result of RAG and Embedding Weaknesses, Training Data Poisoning, AI Supply Chain issues, and can directly impact Clinical Error and Data Error.

IMPROPER OUTPUT

Improper output occurs when the output of an AI system differs from the intended output. Although related to Misinformation, improper output may be outputs that simply do not match what was expected, such as an AI application that produces text in a csv format, where JSON was expected, or it may be outputs that are correct but undesirable. Outputs that create unlawful bias or other unwanted content such as profanity, outputs that are not clinically appropriate to the member, or outputs that would violate regulatory requirements such as creating a medical device would be other examples of Improper Outputs. Improper Outputs are directly related to Clinical Error, Unlicensed Activity, and Non-compliant to Policy or Obligations.

SENSITIVE INFORMATION DISCLOSURE

Sensitive Information Disclosure occurs when the output of an AI system includes information that includes PHI, PII, or other confidential information that exceeds what the user is entitled to, or exceeds the expectations of the system. As an example, an AI system that discloses the PHI of one member to a different member. Sensitive Information Disclosure can result from data being improperly included in training data or RAG sources, weak authentication controls, or Prompt Injection. Sensitive Information Disclosure is related to Elevation of Privilege, Spoofing, Overuse, and Information Disclosure. When it involves PHI, it may create a reportable Breach.

EXCESSIVE AGENCY

Excessive Agency occurs when an AI system has the ability to exceed its intended functionality, permissions, or autonomy. As an example, an LLM that is intended to have read access to a member's data exceeds its purpose when it modifies the data instead of reading it. Excessive Agency is a concern with most AI systems, but should be a top consideration for any system that includes autonomous AI agents. Excessive Agency is related to Unlicensed Activity, Elevation of Privilege, Clinical Error, Non-Compliance to Policy or Obligations, Data Error, and Repudiation.

TRAINING DATA POISONING

Training Data Poisoning occurs when pre-training, fine-tuning, or embedding data is manipulated or modified to introduce vulnerabilities, errors, or biases. It can impact security, compliance and privacy. Training Data Poisoning is of top concern when user inputs and activities are fed directly to a model for continuous improvement, without steps to sanitize the data. Training Data Poisoning is related to Data Error and Tampering.

OVERCONSUMPTION

Overconsumption occurs when an AI system exceeds the compute or financial resources of the system. Overconsumption is directly related to Denial of Service risks, but can also occur from Prompt Injection, Overreliance, and Excessive Agency.

MODEL THEFT

Model Theft is the unauthorized acquisition, copying, or extraction of proprietary AI models. Model Theft can occur via malicious insiders accessing back-end model training and storage systems, or through Prompt Injection that results in model extraction. Model Theft is a form of Information Disclosure.

AI SUPPLY CHAIN

AI Supply Chain risks are related to the integrity of AI models, datasets, and vendors. AI Supply Chain risks include software licensing risks, poorly designed or developed models, datasets that contain false information or malicious data, and AI vendors that have weak security, compliance, and privacy controls. Open source AI models and dataset repositories such as Hugging Face create a risk if the models and datasets are not properly evaluated and vetted before use. New AI vendors create additional third-party risks, especially when the use of the vendor involves PHI, PII, or Customer Data. AI Supply Chain risks are related to most INCLUDES NO DIRT risks, depending on the context.

PROMPT LEAKAGE

Prompt Leakage occurs when system prompts contain sensitive information or proprietary methods, and the prompts are discoverable through Prompt Injection or Improper Output. Prompt Leakage may allow a malicious actor to infer the functionality of the AI system, providing guidance for how to attack the system.

Controls Definitions

AI systems exist as a component of larger software systems. As an example, a chatbot interface could exist inside a mobile application with non-AI features. As such, a large number of the security, compliance, and privacy controls related to AI systems are the same as those for other applications. The unique nature of AI systems has led to the development of AI specific controls. A control is a feature or mitigation in a system that reduces the probability or impact of a risk. Generally, there are three categories of controls:

PREVENTATIVE CONTROLS

Controls that are proactive in nature, designed to anticipate and block attack vectors and reduce risk in advance. Patching a system, implementing a firewall, or prohibiting a role in a system from doing an action, are examples of preventative controls.

DETECTIVE CONTROLS

Controls that are reactive in nature, designed to identify attacks and vulnerabilities when they occur. Event logging, monitoring, or report reviews on a periodic basis, are examples of detective controls.

CORRECTIVE CONTROLS

Controls that result after an event that help minimize ongoing damage. Incident response procedures , feature kill switches, and actions taken as a result of a detective report control, are examples of corrective controls.

Generally there are also multiple ways that preventative, detective, and corrective controls can be implemented.

PHYSICAL CONTROLS

Physical controls such as barriers, locks, cameras, and fire control systems that reduce physical risks.

ADMINISTRATIVE CONTROLS

Controls that provide governance and guidance, such as policies, procedures, training, testing, and auditing. Administrative controls often (but not always) depend on humans to use them or to implement them properly.

TECHNICAL CONTROLS

Controls that implement and enforce administrative controls such as firewalls, patch systems, access control, and backups.

Key Controls[1] Related to Development and Operations of AI/LLM Systems

With regards to the design, build, operations, and ongoing support of AI/LLM systems, the goal is to create a layered approach that uses preventative, detective, and corrective controls, through both administrative and technical approaches, and balances risk with user experience to create the desired outcome. When developing AI systems, preventative controls can be deployed during the design and development of the system, or at runtime. Likewise detective controls can exist in production during runtime, or as a regular out-of-band monitoring process. These controls will expand over time as companies learn and grow their AI capabilities. While many controls are technological, others are human. AI system key controls include:

DOCUMENTATION

Documentation in the form of policies, requirements documents, technical designs, threat models, testing methodologies, subject matter expert (SME) reviews, executive sign offs, and logs of activities all form a strong foundational set of administrative controls.

RECORD REPOSITORIES

Omada has multiple record repositories, or stores, including both member health records and Omada business records. These stores have built-in technical controls to protect against data error and tampering, including version control, data backup, and retention features. The development and operation of new AI systems can account for some AI risks by adopting existing record stores. Records for a given business process or transaction type may be stored in multiple repositories; for example, a tech spec or a product requirements doc may exist on a file server and also be included with related code in the code repository.

AUTHENTICATION AND AUTHORIZATION

Because the AI system exists within larger software systems, they can inherit and utilize many different general controls. One key control that the AI systems can use to reduce the risks of Sensitive Information Disclosure is the use of existing authentication and authorization mechanisms. AI systems that utilize the authentication tokens when retrieving information from or writing information to other systems can inherit the restrictions enforced by those authentication tokens.

USER INTERFACE

The user interface (UI) can itself provide technical and administrative controls in an AI system. As an example, a requirement to be transparent to the end user when they are dealing with an AI agent instead of a human can be surfaced within the UI.

OPERATIONAL TEAM WORKFLOWS

Existing human-based workflows, sometimes referred to as "human in the loop," can provide administrative, detective and corrective controls in relation to an AI system. As an example of a detective control at Omada, the transcript of an interaction between a member and an AI system can be provided to the Care Team. If the Care Team reviews the transcript, they may detect errors or mistakes made by the AI system. The Care Team then has the opportunity to both correct the mistake with the member and provide feedback to the AI engineering team to adjust the AI system.

SOURCE GROUPS

Some internally-developed AI models need access to copies of production data sets. To be most useful, these copies of production data need to include some PHI, but the minimum necessary rule may apply to that use. The training of these models includes the risks of Overuse, Sensitive Information Disclosure, Training Data Poisoning, and AI Supply Chain risks. Omada has developed the concept of

1 Certain regulatory systems, e.g. Sarbanes-Oxley in the US, may identify certain controls on certain business processes as "key controls." Here, we use "key controls" specifically about the subject (AI) and not any connected business process.

Source Groups which are curated subsets of copies of production data where the data copied is limited to what Omada deems necessary, and so the content of source groups has built-in mitigations against Overuse and Sensitive Information Disclosure. When AI engineering teams want to develop or train a model using production datasets, they must specify the needed Source Groups, which allows them to adhere to the principle of minimum necessary. The people who can access and use Source Groups are limited, which is an Administrative Control and a Technical Control that mitigates the risk of Training Data Poisoning.

AI/LLM MODEL & SYSTEM DEVELOPMENT

The creation and tuning of AI models and systems contains several activities that represent preventative controls, including fine-tuning, retrieval-augmented generation (RAG), and prompt engineering. At each of these steps, engineers can influence and constrain the output of the system to produce desired results and reduce Misinformation, Improper Output, Unlicensed Activity, Sensitive Information Disclosure, and Prompt Leakage. The use of subject matter experts (SMEs) such as clinical, legal, and security SMEs to work with the engineering teams at appropriate times provides an additional layer of preventative administrative control that is recommended as a best-practice.

FINE-TUNING

The act of refining a pre-trained model with specialized data and targeted learning to influence the output of the model. Fine-tuning specializes the knowledge of a model, improves the performance and cost of a model, and allows the output to be constrained, creating a specialized version of the model.

RETRIEVAL-AUGMENTED GENERATION (RAG)

RAG enhances the capabilities of an AI model by connecting it to external knowledge sources. The RAG queries its knowledge base at runtime for relevant information and adds it to the context of a prompt, allowing the model to create a better response. RAG improves accuracy and reduces hallucinations, and allows for the use of near-real-time context. RAG is faster to create and modify than Fine-tuning but at a cost in runtime processing time.

PROMPT ENGINEERING

The process of refining the inputs to an LLM system to influence and guide outputs. Prompt Engineering can be used as a preventative control by telling the model what types of output Omada does and does not want.

PROMPT ENGINEERING WITH SME

A human subject matter expert (SME) works directly with the LLM system and with the prompt engineer to tune the prompts toward a desired output. Having subject matter experts involved in the prompt engineering stage is a preventative control against risks related to the expertise, such as ensuring the system output is clinically accurate.

JUDGES

Judges are used during model development, data labeling, data set curation, prompt engineering, and ongoing operations to evaluate the output of AI models. Judges can either be human or LLM, and can be used to check outputs for things such as accuracy, bias, and relevance. LLM development platforms such as Langsmith provide the ability to systematically create and use judges to measure AI features and their improvement over time.

HUMAN SME AS JUDGE

A human subject matter expert (SME) reviews the output of an LLM and makes a judgement as to whether the output matches the desired output. The human can also provide notes as to their reasoning for the judgment. This preventative control helps the model trainers improve the system. As an example, if the model was supposed to produce a healthy recipe, the Judge would review recipes output by the model and make a binary decision yes or no.

LLM AS JUDGE

The output of an LLM is fed to an LLM Judge to make a judgement as to whether the output matches the desired output, and provides its reasoning. The process is the same as Human as a Judge, except that it is both faster and cheaper, but may lack expertise.

HUMAN SME AS EVALUATOR OF AN LLM JUDGE

The judgement of an LLM Judge is reviewed by a human to determine whether the Judge made the correct judgement. In this detective control, the human makes a binary decision as to the correctness of the LLM Judge's original decision and provides their reasoning.

The key difference between these methods is that human and LLM as a Judge allows the team to improve the accuracy of the primary model. Human as Evaluator of an LLM Judge allows the team to improve the accuracy of the LLM judge. AI teams may employ all three methods when designing and operating LLM AI systems. In each case, the judgement of these methods is used to improve the Fine-tuning, RAG, and Prompt Engineering, in an iterative manner, until the desired levels of accuracy are achieved.

TRACES EVALUATION WITH JUDGE

The same judges that are used to develop the System can also be used as detective controls to monitor the System for precision and drift. These can be any combination of Human SME as Judge, LLM as Judge, or Human SME as Evaluator. Real world user inputs and system outputs can be fed back into the LLM platform as traces for evaluation by a judge, and deviations from expected precision can be evaluated by the team and used to further train the model.

SAMPLE SIZE

Determining the appropriate sample size for a given system or control activity is an important control point in the development and evaluation of AI systems. AI teams need to determine and document the appropriate level of testing needed for a given system to generate confidence in the control framework. As an example, Human SME as Judge and Human SME as Evaluator are expensive and time consuming processes. Teams must determine how many samples the SME must evaluate are appropriate. Likewise, Traces Evaluation with Judge requires computational resources. For a given system, the team must determine what a statistically significant number of trace samples would be, or if all traces must be evaluated, to satisfy the risks based on the functions of the system. Determining the appropriate sample size should be done in consultation with the SMEs and documented as a part of the system control framework.

ASYNCHRONOUS ALERTS

Monitoring tools such as Traces Evaluation with a Judge can generate alerts to the engineering team or the operations team as detective or corrective controls. The response to these alerts can be used to adjust the operations team's workflows. As an example, a traces evaluation with a judge might identify a member chat transcript indicating a potential for self-harm. An alert could be sent to the coach inbox instructing them to review the transcript and take appropriate action.

GUARDRAILS

In LLM AI systems, guardrails are methods to validate and control inputs and outputs to the AI system. There are different frameworks of LLM guardrails, such as the NVIDIA NeMo guardrails framework. Guardrails can be applied to user input, AI system retrieval and execution tasks, and AI system outputs. Foundation Models often have guardrails built in during their training process to influence model output. Guardrails when properly implemented can act as a significant mitigation to Prompt Injection, Overreliance, Misinformation, and Improper Output, which also in turn prevent other risks such as Clinical Error or Unlicensed Activity. Most LLM guardrails are implemented as prompts to either base or fine-tuned models to identify specific cases. As an example an output guardrail on a nutrition agent response might take the system output and prompt a fine-tuned clinical nutrition model with a question such as "does this answer match Omada's clinical nutrition guidance for a member with these characteristics." Guardrails can run serially in the data flow (blocking) or parallel to the data flow (non-blocking). Because each guardrail increases computational work and can impact latency, the proper use of guardrails is an important consideration when balancing user experience, cost, and risk.

FOUNDATION MODEL GUARDRAILS

Guardrails built into the base model. Different models have different levels of guardrails built into them, such

as guardrails against racist or profane output. The foundation model guardrails may change over time with newer versions of the foundation model. As such, they may be relied upon but require testing during the system build process to evaluate the capabilities and performance of the model. Newer versions of the model should be evaluated before deployment to ensure changes to the model including its guardrails don't degrade the expected controls. As an example, teams could evaluate multiple foundational models or versions of the same model using a red team dataset and LLM as a judge to compare the guardrail capabilities of multiple foundation models against each other. When relying on foundation model guardrails, engineers should document the specific functions the system depends on.

INPUT GUARDRAILS

Input guardrails evaluate the input from the user. They can execute fully before allowing the input to flow to the next step in the data flow, blocking the input if it is determined to violate the guardrail, or they can execute in parallel to the data flow, but block the output of the next step if it is determined the input violates the guardrail. The decision to implement a serial (blocking) or parallel (non-blocking) input guardrail is a balance between risk, latency, compute cost, and user experience. Generally input guardrails should block when the next step is an execution agent, but should not block when the next step is a generative agent. Input guardrails are an important control against prompt engineering risks.

RETRIEVAL AND EXECUTION GUARDRAILS

When the AI system chains multiple models and execution agents together, and the output of one step in the data flow becomes an input to the next step, a retrieval or execution guardrail can protect against overreliance risks. Like input guardrails, retrieval and execution guardrails can be implemented as blocking or non-blocking to balance risk with system performance.

OUTPUT GUARDRAILS

Output guardrails evaluate the output of the model or system. They can be used to control against many risks, but also add latency to the system. While output guardrails can run in parallel to other processes, the final determination of an output guardrail necessarily runs last and the output of the system is blocked when it violates the guardrail.

AGENTIC ACTION NODES

AI systems can include LLM models, guardrails, and AI agents. Whereas LLM models generate an output, and guardrails produce an evaluation, agentic nodes are designed to allow the AI to execute an action. The actions or blocked actions of an agentic node can themselves be preventative controls in a system.

DATASETS

The knowledge sources and datasets used during finetuning, RAG, prompt engineering, guardrail development, and judge evaluation can be preventative and detective controls.

SME EVALUATED DATASET

Datasets selected or reviewed by a SME to determine its appropriateness for the function of a system. As an example, if Omada were finetuning a model that does nutritional evaluation, a clinical nutrition SME should be involved in the selection of the nutritional dataset used to finetune the model.

RED TEAM DATASET

The development of AI systems, models, and guardrails requires systematic methods of testing the system. A red team dataset can be used to test performance of a system's controls. Red team datasets are designed to get the system to produce an output that violates its purpose. As an example, a red team dataset of malicious prompts can test whether the system will produce a profane output, such as a dirty joke, or an output that is unsafe, such as a recipe that includes harmful ingredients. Red team datasets can enhance testing during the development process to reduce the risks of Prompt Injection, Overreliance, Misinformation, Improper Output, Sensitive Information Disclosure, Excessive Agency, Overconsumption, Model Theft, and Prompt Leakage. Redteam data sets may need human SME input or review, depending on the risk being tested.

RED TEAM AGENTS

One method of automating the testing of AI systems is the development of hostile or red team AI agents.

Red team agents differ from red team datasets in that they can be given autonomy and goals to explore new methods and techniques. This gives Omada the ability to iterate and improve AI systems over time, and provides a mechanism for testing new threats.

PROCUREMENT PROCESS

Omada uses procurement requests to process all requests for vendors and applications, including open source licenses and tools. Procurement requests initiate the security and legal review of vendors, contracts, and licenses. New AI vendors, new AI model licenses, and new AI services must go through Omada's normal process. An appropriate procurement process can ensure that, for example, use of a licensed foundation model does not result in Information Disclosure back to the model's author. This is one mitigation to AI Supply Chain Risks and risks to PHI.

CLIP REVIEWS

Compliance, Legal, Information Security, and Privacy must review new AI systems, applications, and features. The AI PRD process and Threat Model provides CLIP teams the opportunity to assess the risks of new AI activities, and to perform AI threat modeling. CLIP staff are available to answer questions and brainstorm methods before submitting for an official CLIP review.

CLINICAL AND CARE TEAM REVIEWS

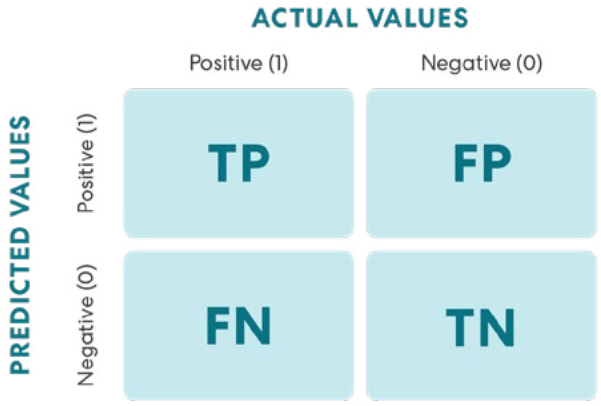
Clinical and Care Teams must review new AI systems, applications and features. The AI PRD process and Threat Model process provides Clinical and Care Teams the opportunity to assess the efficacy and impact of new AI systems, ensuring that Omada's products are guided by science. Clinical and Care Teams can also be used as Human SME Judges or other relevant SMEs to assist with System development and tuning.

AUDITS

Internal and external auditors can review documentation, system statistics, and system output traces to evaluate system performance as a detective control. Engineers may choose to incorporate similar reviews in QA testing or periodic system maintenance or review.

CONFUSION MATRICES, PRECISION, AND SYSTEM EVALUATION

A Confusion Matrix table is a method of evaluating the accuracy of an AI model or system. In data science terms, the output of a model can produce four possible outcomes: True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN). Precision is the number of True Positive outputs the model creates, and is a measure of how trustworthy the model is for the desired outcome.



Confusion Matrices can be used during the risk evaluation of an AI model, but get more complicated when evaluating the System as a whole. Let's take a simple example of a model meant to produce a healthy recipe, with an LLM Judge, and an Output Guardrail. During the development process, a combination of fine-tuning, RAG, and prompt engineering were used in an iterative process with an LLM judge that made binary judgements of healthy or not healthy. During runtime, the output of the model is evaluated by a Guardrail that is making a real-time judgement of healthy or not healthy, and blocks the outputs it considers not healthy. The model, the guardrail, and the judge each have their own Confusion Matrix to measure correctness. The model is correct when it creates a healthy recipe. The guardrail is correct when it blocks an unhealthy recipe. When chained together, we can assess the overall performance of the system from both a user experience and a risk perspective.

In an LLM system chain like this, the **System Output Matrix** might look something like the table below.

When using an Output Matrix like this to evaluate the risk and strength of the controls of an AI System, the level of risk that can be accepted depends on the functions of the System and the trade-offs between performance, precision, and risk. From a risk perspective in this example, only a False Positive is a failure of controls, but both a False Negative and a True Negative impact user experience.

System Output Matrix for Healthy Recipe Recommender System		Model Output	
		Healthy Recipe	*Not Healthy Recipe*
Guardrail Action	*Allow*	*True Positive* **System Works**	*False Positive* **Tune Model AND Guardrail**
	Block	*False Negative* **Tune Guardrail**	*True Negative* **Tune Model**

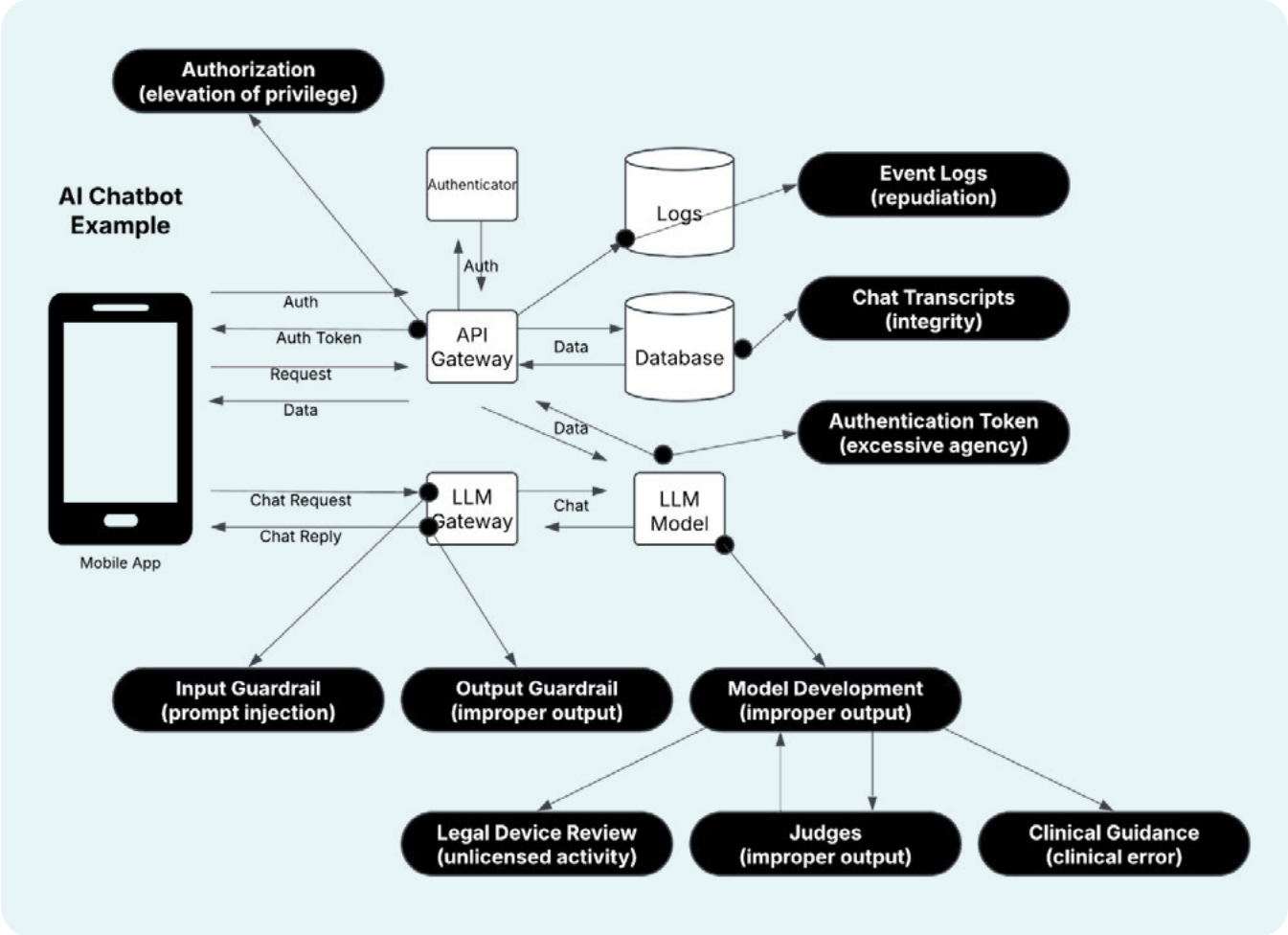
Defense in Depth Control Matrix for AI systems

The above controls represent a menu of options that can be applied to an AI system to reduce the risk of a given system for a given purpose. During the threat modeling process, teams should evaluate and document which controls will be used to mitigate which risks to build trust in the overall system. The overall control matrix for an AI system is as follows:

AI/LLM Control Matrix	Administrative	Technical
Preventative	• Documentation • Source Groups • Sample Size • Procurement Process • CLIP reviews • Clinical & Care Team Reviews	• Record Repositories • Authentication & Authorization • Fine Tuning • RAG • Prompt Engineering • Prompt Engineering with SME • Human SME as Judge • LLM as Judge • Human SME as Evaluator • Foundation Model Guardrails • Input Guardrails • Retrieval & Execution Guardrails • Output Guardrails • Agentic Action Nodes • Source Group • SME Evaluated Dataset • Red Team Dataset • Red Team Agents • System Output Matrix

AI/LLM Control Matrix	Administrative	Technical
Detective	• Operational Team Workflows • Sample Size • Audit	• Traces Evaluation with Judge • Asynchronous Alerts • SME Evaluated Dataset • Red Team Dataset • System Output Matrix
Corrective	• Operational Team Workflows	• Asynchronous Alerts • Agentic Action Nodes • System Output Matrix

A well designed system should have a multitude of controls including preventative and detective controls to address the applicable risks, based on the intended function of the system.



The control matrix can be applied to the AI system risks in numerous ways, but certain controls naturally map to certain risks. The below risk/control matrix is intended to make it easier for teams to determine the best controls for a given system:

AI/LLM Risk Matrix	Administrative	Technical
Prompt Injection	• Documentation	• Foundation Model Guardrails • Input Guardrails • Retrieval & Execution Guardrails • Output Guardrails • Red Team Dataset • Red Team Agents • User Interface
RAG and Embedding Weaknesses	• Documentation • Procurement Process • CLIP reviews • Clinical & Care Team Reviews	• Prompt Engineering with SME • Human SME as Judge • LLM as Judge • Traces Evaluation with Judge
Overreliance	• Documentation • CLIP reviews	• Retrieval & Execution Guardrails • Output Guardrails • Red Team Agents • Traces Evaluation with Judge • System Output Matrix • User Interface
Misinformation	• Care Team Workflows	• Human SME as Judge • LLM as Judge • Human SME as Evaluator • Foundation Model Guardrails • Input Guardrails • Output Guardrails • Red Team Dataset • Red Team Agents • Traces Evaluation with Judge • System Output Matrix

AI/LLM Risk Matrix	Administrative	Technical
Improper Output	• Documentation • CLIP Reviews • Clinical & Care Team Reviews • Care Team Workflows • Audit	• Human SME as Judge • LLM as Judge • Human SME as Evaluator • Foundation Model Guardrails • Input Guardrails • Output Guardrails • Red Team Dataset • Red Team Agents • Traces Evaluation with Judge • System Output Matrix
Sensitive Information Disclosure	• Documentation • CLIP Reviews • Care Team Workflows • Audit	• Authentication & Authorization • Prompt Engineering • Prompt Engineering with SME • Red Team Dataset • Red Team Agents • Traces Evaluation with Judge • System Output Matrix
Excessive Agency	• Documentation • CLIP Reviews • Audit	• Authentication & Authorization • Foundation Model Guardrails • Input Guardrails • Retrieval & Execution Guardrails • Output Guardrails • Agentic Action Nodes • Red Team Agents • System Output Matrix
Training Data Poisoning	• Documentation • Source Groups • CLIP reviews • Clinical & Care Team Reviews	• Fine Tuning • RAG • Prompt Engineering with SME • Human SME as Judge • Source Groups • Red Team Dataset • Red Team Agents • Traces Evaluation with Judge • System Output Matrix

AI/LLM Risk Matrix	Administrative	Technical
Overconsumption	• Documentation • Procurement Process • CLIP reviews • Audit	• Input Guardrails • Prompt Engineering • Asynchronous Alerts
Model Theft	• Documentation • CLIP reviews • Audit	• Authentication & Authorization
AI Supply Chain	• Documentation • Procurement Process • CLIP reviews • Audit	• Fine Tuning • RAG • Prompt Engineering • Prompt Engineering with SME • Human SME as Judge • LLM as Judge • Human SME as Evaluator • SME Evaluated Dataset • System Output Matrix • Traces Evaluation with Judge • System Output Matrix
Prompt Leakage	• Documentation	• Prompt Engineering • Foundation Model Guardrails • Input Guardrails • Output Guardrails • Red Team Dataset • Red Team Agents

Conclusion

When we created the original INCLUDES NO DIRT model, our primary focus was to simplify the process of evaluating complex systems to establish a fast, repeatable, consistent approach. This approach worked well for us over the past six years in evaluating new features, new vendors, and new systems, but was lacking when we tried to apply it to AI/LLM systems. AI introduces a spectrum of new risks, as well as an entirely new vocabulary to risk teams. Early in the process, we realized that our AI engineers and our risk assessors were talking past each other because they lacked a common language. As an example, risk assessors didn't understand what was meant by fine tuning, and engineers didn't understand how to apply concepts like minimum necessary to the training of a model. The PROMISE TO MAP threat model is our attempt to build a common ground. Prior to publishing this model to the world, we've used it internally on multiple projects and it has increased the velocity of our teams. We hope the users of the PROMISE TO MAP model are able to adapt it to their companies to quickly and safely build and deploy new AI and LLM systems.

Appendix 1 - PROMISE TO MAP Threat Model Worksheet

This SAMPLE worksheet is provided as an example for how to use the PROMISE TO MAP threat model to assess new AI systems at a healthcare company. It organizes the PROMISE TO MAP risks into three stages: System and Model Development, System Functionality, and Ongoing System Operations and Tuning. It also includes specific risks from the INCLUDES NO DIRT model that are likely to need additional controls in an AI system. The purpose of this worksheet is to guide conversations with the team to determine in-scope requirements, identify the controls matrix that will address those requirements, and establish action items for the team. Companies attempting to adopt the PROMISE TO MAP model should modify this workbook to match your unique business concerns.

1.0 - SYSTEM DESCRIPTION

Describe the AI System being analyzed and its function. Links to product documentation such as PRDs, technical specs, and other documentation should also be included. The system description should include the goals/intent of the system and dataflow diagrams that describe how it will function. Include here links to PRDs, technical specifications, and other documentation that cover the specific sources of data used by the system. In some cases, it may be necessary to provide a complete list of data down to the individual column names to allow for a complete analysis.

2.0 - SYSTEM AND MODEL DEVELOPMENT

System and Model Development risks present themselves in the design phase, and should be dealt with by teams from the beginning, although they will persist throughout the life of the system.

2.1 - AI SUPPLY CHAIN

List the models, datasets, and vendors required by the system, and how each will be used. If the system requires new vendors or new third party models, identify them and the corresponding procurement requests.

2.2 - OVERUSE

Does the system development require training, fine-tuning, or RAG using PHI? If so, identify the Source Groups or other PHI not in a Source Group that constitute the minimum necessary dataset for the purpose of creating the system. (“Minimum necessary” means the least amount of PHI you would need to do the proposed use.

2.3 - RAG AND EMBEDDING WEAKNESSES

Does the system require RAG or embeddings with external knowledge sources? If so, describe the data sources and the process to generate, store, and retrieve the data. How are the knowledge sources identified, curated and versioned, and retired if necessary, to keep them consistent and current, and avoid conflicts? Is the company authorized, by, e.g. a license, to use that source that way?

Do the external knowledge sources involve clinical standards or practices? If so, describe the methods used to validate them by the clinical team. Document the source of any content and why we believe it is clinically or scientifically current and sound, and where a relevant clinician has signed off on that.

2.4 - EXCESSIVE AGENCY AND ELEVATION OF PRIVILEGE

Describe the mechanisms to constrain the agency of the system, including any detective controls that identify that changes to agency or privilege have unexpectedly occurred. How are authentication and authorization implemented in the system?

2.5 - TAMPERING

What aspects of the system should create a durable record? How are inputs and outputs of the system recorded?

3.0 - SYSTEM FUNCTIONALITY

System functionality risks are related to the core use of the system and require ongoing controls, including continual testing, tuning, and oversight. They directly impact the user experience and include many clinical, compliance, privacy, and security risks.

3.1 - PROMPT INJECTION

Does the system accept inputs from the user? If so, describe the specific types of inputs that are expected.

What mechanisms, including guardrails, will the system use to validate input to prevent prompt injections? Describe where they will be implemented, how they will be tested, and how they will persist over time.

How will the system handle user inputs that differ from the expected input?

3.2 - OVERRELIANCE

Does the system chain models together, so that the output of one model becomes the input of the next? If so, what mechanisms, including guardrails, will the system use to validate output at each step to prevent overreliance? Describe where they will be implemented, how they will be tested, and how they will persist over time.

3.3 - MISINFORMATION

How does the system training, testing, and operation prevent the production of false or misleading information such as hallucinations? Describe the testing methodologies, judges, red team agents, and guardrails to be used.

3.4 - IMPROPER OUTPUT

How does the system training, testing, and operation prevent improper output? Describe the testing methodologies, judges, red team agents, and guardrails to be used? What considerations are in place to ensure that guardrails persist over time?

How does the system prevent unlawful bias, if applicable?

How does the system prevent offensive output?

How does the system account for clinical accuracy, clinical safety, and timeliness?

Based on the purpose of the system, what are the high risk clinical concerns of the system?

Does the system monitor for, and take action if it detects indications that the member intends to commit harm to self or others?

Does the system involve making calculations or recommendations? If so, describe the methodologies, judges (human and machine), red team agents, and guardrails to be used to validate the calculations or recommendations. Note that certain calculations may trigger a deeper FDA analysis. If that is completed already, link it here.

How does the system prevent model drift?

3.5 - UNLICENSED ACTIVITY

Does the system involve activities that could be construed as a scope of practice requiring licensure if performed by a human? If so, how does the system training, testing and operation prevent the activities of the system crossing a licensure line?

Does the system involve activities that could be construed as being a regulated medical device? Activities where the system output is a diagnosis or a treatment plan, are two examples, but there are other outputs that may only issue from medical devices. If so, how does the system training, testing and operation prevent the activities of the system becoming a medical device? Indicate if the Legal department has performed a device analysis of the system and what the conclusion was. If the legal department completed its analysis AND concluded it was not a device, please still describe the system training, testing and operation controls that prevent the model from evolving into a device.

Does the system create any output that would modify the actions or behaviors of the care team? If so, describe the process for testing and validating the output of the system by the care team. Please consider any cases where care team use of AI tools may become so swift as to be inadvertently but effectively "automated."

Does the system have any function or create any output that would require an update to Omada's Terms of Use or internal policies?

3.6 - SENSITIVE INFORMATION DISCLOSURE

How does the system prevent system output from inappropriately disclosing sensitive information, such as disclosing member PHI to the wrong member? How does the system prevent usage of the PHI of one person in responding to inputs about a different person? Describe the testing methodologies, judges, red team agents, and guardrails to be used?

3.7 - PROMPT LEAKAGE

How does the system prevent the leakage of sensitive information and methods contained within prompts? Describe the testing methodologies, judges, red team agents, and guardrails to be used?

4.0 - ONGOING SYSTEM OPERATIONS AND TUNING

System operations risks are related to the long-term operation of the system, but generally do not impact user experience.

4.1 - OVERCONSUMPTION AND DENIAL OF SERVICE

How does the system prevent overconsumption and denial of service, in the development, testing, training, and ongoing use? Describe the mechanisms to rate limit inputs, prevent overconsumption of compute, and monitor use and billing of third party services.

4.2 - MODEL THEFT AND INFORMATION DISCLOSURE

How does the system prevent model theft and information disclosure?

4.3 - REPUDIATION

How are system inputs and outputs logged and recorded at each step in the data flow? Describe the mechanisms to ensure non-repudiation.

4.4 - TRAINING DATA POISONING

Does the operation of the system include continuous training, fine-tuning, or RAG, based on user inputs or previous system outputs? If so, describe the mechanisms to sanitize data to prevent training data poisoning.

5.0 - ACTION ITEMS

The results of a threat model should be a list of actions the team intends to take to establish the proper controls, based on the function of the system. This can be as simple as a list of action items and priorities, or as complicated as necessary. Omada has developed a repeatable worksheet that includes easy to consume requirements and maps the requirements to the subject matter experts (who is responsible for the requirement), the associated threat model risks, and the applicable controls. As teams evaluate new systems, they add to the list. Threat modelers should develop their own system for capturing the output of the threat model to ensure all requirements are captured, and to use to evaluate systems before launch and on an ongoing basis.

AI System Name

#	ID	Applicable Yes/No	Requirements	Requirement SME	Threat Model Risks	Preventative Controls	Detective Controls	Corrective Controls	Output Guardrails Required?	Action Item Needed?
30			DO encrypt all member data in transit and at rest. DO ensure all databases encrypt member data at the column level per Omada database encryption standards	Security	Sensitive Informati... Tampering				☐	☐
31			DO validate all input from the member to the system before any agent acts or any output is returned	Security	Prompt Injection				☐	☐
32			DO validate the output of any agent or model before it is accepted by the next agent or model when chaining multiple agents or models together	Security	Overuse				☐	☐
33			DO restrict the system from outputting confidential information about the system prompts or the model	Security	Sensitive Informati...				☐	☐
34			DO implement rate limiting, monitoring, and alerting to measure and prevent the overconsumption of resources.	Security	Overconsumption Denial of Service				☐	☐
35			DO restrict access to system models and data sources to authorized Omadans	Security	Model Theft Sensitive Informati...				☐	☐
36			DO ensure that all components of the system are backed up according to Omada's backup policy	Security	Denial of Service				☐	☐
37			DO ensure that any training data derived from production is reviewed prior to reprocessing by the system	Security	Training Data Pois...				☐	☐
38			DO ensure all changes to the system are logged	Security Compliance	Repudiation				☐	☐
39			DO monitor, review, and audit confusions matrix statistics, system output matrix statistics or any other system product statistics deemed relevant for the particular function to ensure system is functioning as intended	Security Compliance	Improper Output				☐	☐
40			DO restrict the system to only output PHI that is for the logged-in member	Security Privacy	Sensitive Informati...				☐	☐
41									☐	☐
42									☐	☐
43									☐	☐