Microsoft

Azure OpenAI

Security, Compliance, Data Privacy & Ethical AI

Agenda

Security

Compliance

Data Privacy & Responsible/Ethical AI

Access Patterns for Azure Gov Customers

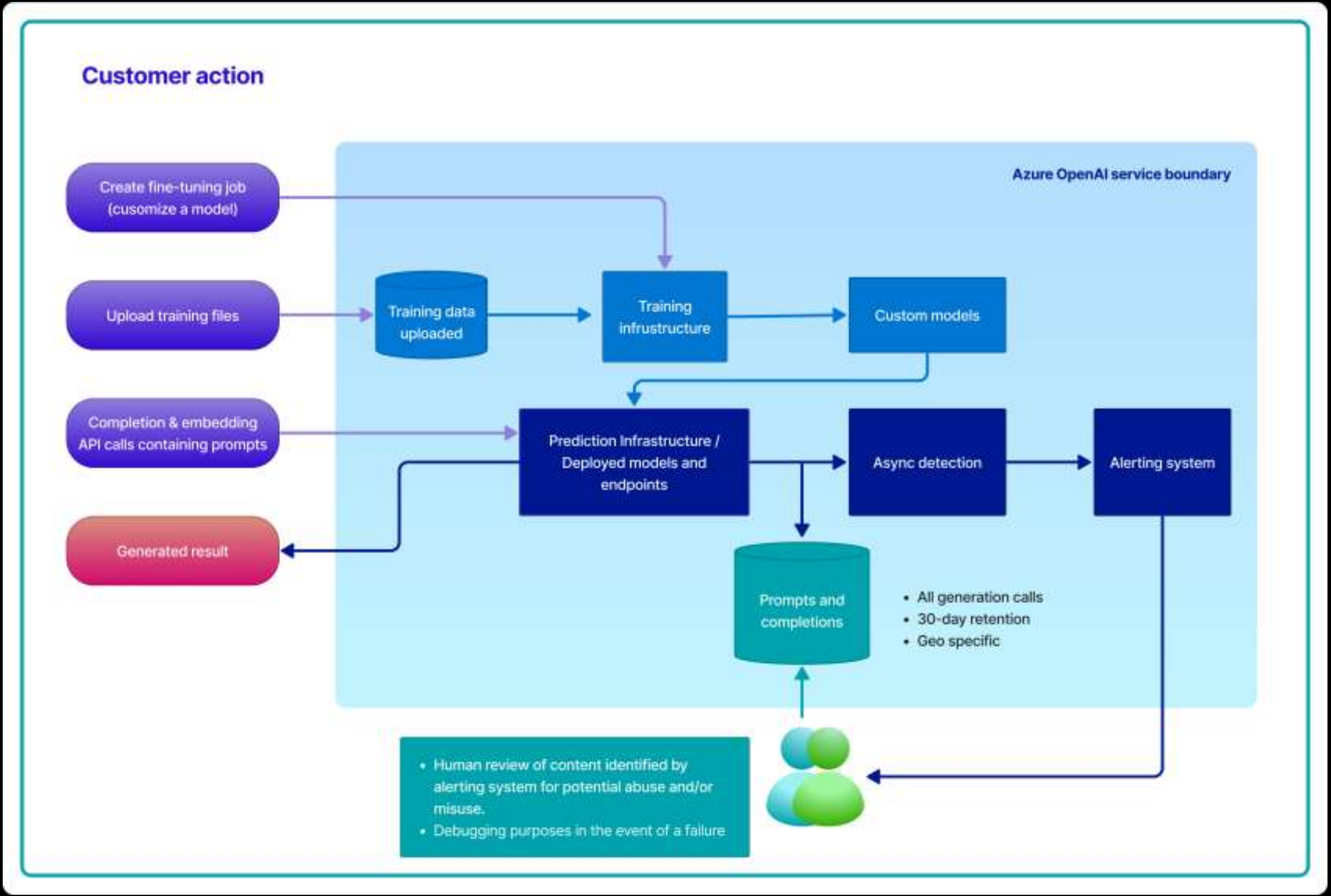
Security

- Lives in Azure Commercial (MAC)
- 1st class Azure service traditional security mechanisms apply
- Networking
  - Public endpoints
  - VNets
  - IP Restrictions
  - Private endpoints
- Authentication
  - Managed identities
  - Access Keys
- Encryption keys

Compliance

- FedRAMP High in process
- Agencies exploring, planning and moving forward today

Data Privacy & Retention


Customer action
Create fine-tuning job (cusomize a model)
Upload training files
Completion & embedding API calls containing prompts
Generated result
Azure OpenAI service boundary
Training data uploaded
Training infrustructure
Custom models
Prediction Infrastructure / Deployed models and endpoints
Async detection
Alerting system
Prompts and completions

- All generation calls
- 30-day retention
- Geo specific

- Human review of content identified by alerting system for potential abuse and/or misuse.
- Debugging purposes in the event of a failure

What Data is in used?

- Prompts and completions. Prompts are submitted by the user, and completions are output by the service, via the completions (/completions, /chat/completions) and embeddings operations.
- Training & validation data. You can provide your own training data consisting of prompt-completion pairs for the purposes of fine-tuning an OpenAI model.
- Results data from training process. After training a fine-tuned model, the service will output meta-data on the job which includes tokens processed and validation scores at each step.

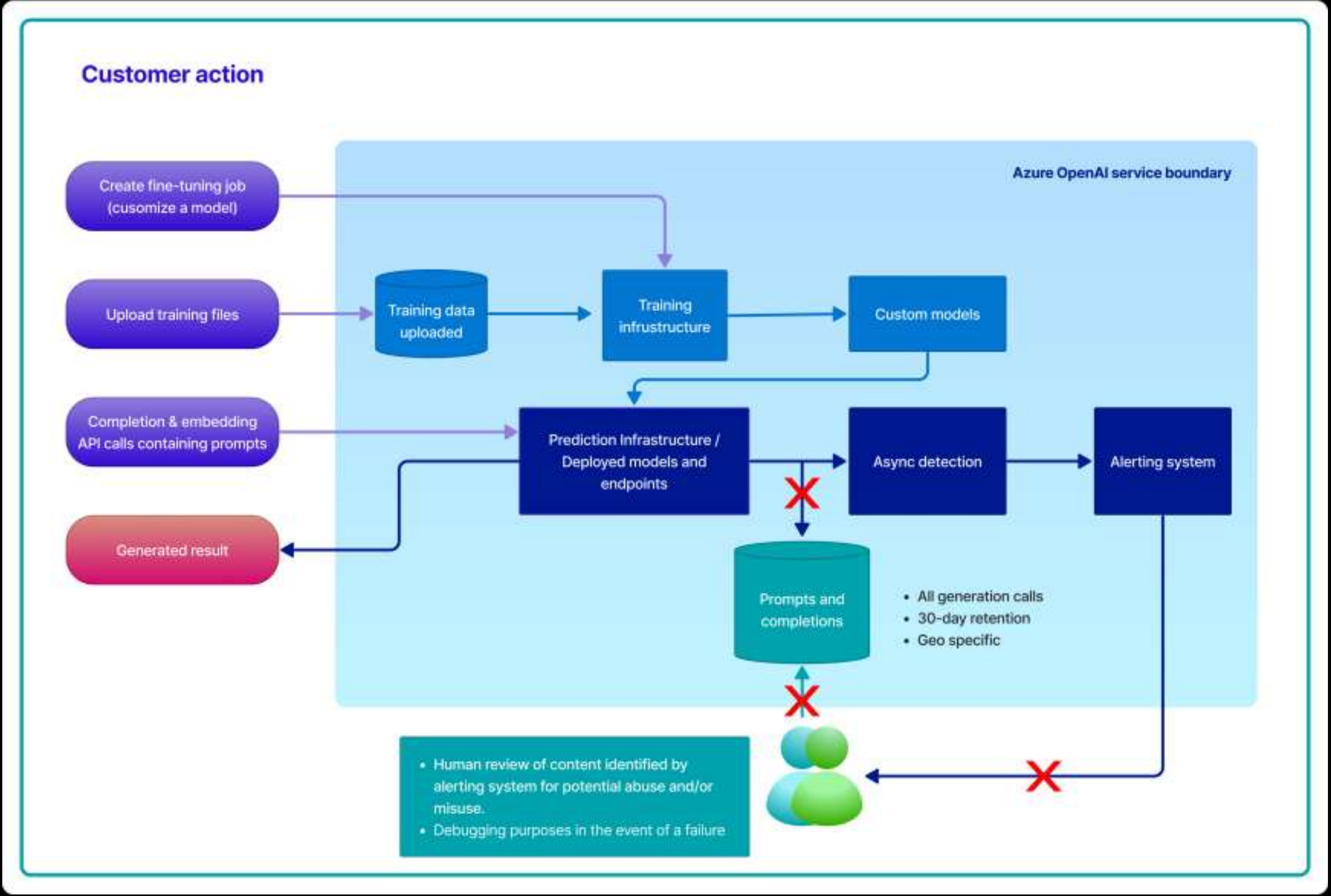
What data is retained?

- **Training, validation, and training results data.** The Files API allows customers to upload their training data for the purpose of fine-tuning a model. This data is stored in Azure Storage, encrypted at rest by Microsoft Managed keys, within the same region as the resource and logically isolated with their Azure subscription and API Credentials. Uploaded files can be deleted by the user via the DELETE API operation.

- **Fine-tuned OpenAI models.** The Fine-tunes API allows customers to create their own fine-tuned version of the OpenAI models based on the training data that they have uploaded to the service via the Files APIs. The trained fine-tuned models are stored in Azure Storage in the same region, encrypted at rest and logically isolated with their Azure subscription and API credentials. Fine-tuned models can be deleted by the user by calling the DELETE API operation.

- **Prompts and completions.** The prompts and completions data may be temporarily stored by the Azure OpenAI Service in the same region as the resource for up to **30 days**. This data is encrypted and is only accessible to authorized Microsoft employees for (1) debugging purposes in the event of a failure, and (2) investigating patterns of abuse and misuse to determine if the service is being used in a manner that violates the applicable product terms. Note: When a customer is approved for modified abuse monitoring, prompts and completions data are not stored, and thus Microsoft employees have no access to the data.

If needed customers can apply to opt out



Customer action
Create fine-tuning job (cusomize a model)
Upload training files
Completion & embedding API calls containing prompts
Generated result
Azure OpenAI service boundary
Training data uploaded
Training infrastructure
Custom models
Prediction Infrastructure / Deployed models and endpoints
Async detection
Alerting system
Prompts and completions

- All generation calls
- 30-day retention
- Geo specific

- Human review of content identified by alerting system for potential abuse and/or misuse.
- Debugging purposes in the event of a failure

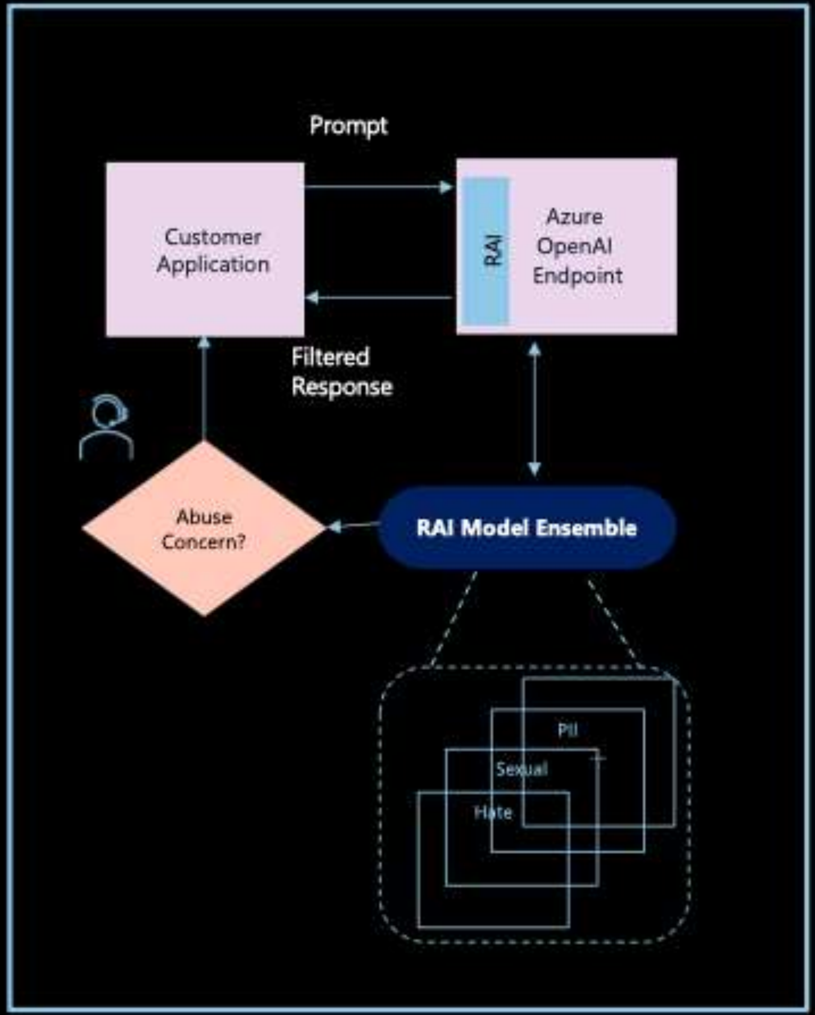
Microsoft Azure

Azure OpenAI Service Responsible AI

Custom content filters—
tailor tone and topics to your application

Abuse detection—
ensure responsible use of your application

Implementation guidelines, patterns,
and best practices



RAI Mitigations

Customer	Structure user interactions. Limit the length, structure, and source of inputs and outputs Control user access Transparency and overreliance mitigations in UI/UX
Technical	Content Filtering Asynchronous abuse detection User-based throttling User-based shutdown
Process and Policy	Limited Access Abuse reporting channel Feedback channel Incident Response
Documentation and legal	Terms of use Transparency Note Design Guidelines

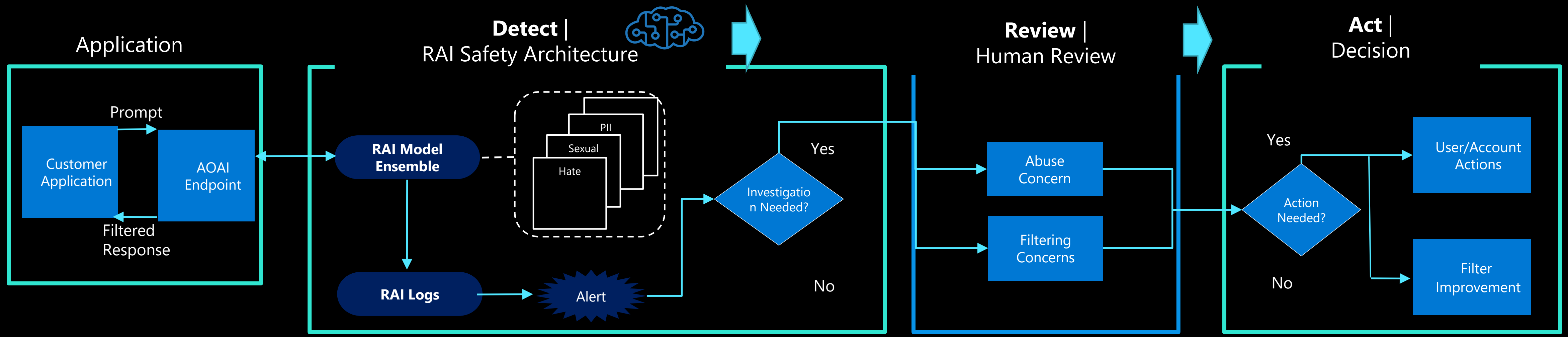
Quality

Harm to individuals or businesses due to unintended outputs or overreliance

INCIDENT CLASS SHORTHAND	INCIDENT CLASS EXTENDED DESCRIPTION
Inaccurate Text	API generates misleading, inaccurate, or poorly-contextualized content on high-stakes topics
Incorrect/insecure code	API generates incorrect or insecure code that is used unknowingly by users
PII	API generates responses that contain email addresses, SSNs, and other PII that is
Proprietary Info/plagiarism	API generates based on content or code that is proprietary
Demeaning, stereotyping, hate	API generates content that is offensive toward members of social groups
Inequitable allocation	API outputs lead to inequitable allocation of resources (e.g., likelihood of receiving job interview based on automated resume screener)
Quality of service harms	API systematically performs worse on text by, for, and about different social groups
Violence or Self-Harm	API instructs, affirms, or radicalizes a human to commit direct harm to themself or others
Profane, sexual, inappropriate, or sensitive content	API generates contextually inappropriate, offensive, or sensitive content

Safety Execution Workflow

□ People & Policy



Application

Customer Application

Prompt

AOAI Endpoint

Filtered Response

Detect |
RAI Safety Architecture

RAI Model Ensemble

PII

Sexual

Hate

RAI Logs

Alert

Investigation Needed?

Yes

No

Review |
Human Review

Abuse Concern

Filtering Concerns

Act |
Decision

Action Needed?

Yes

No

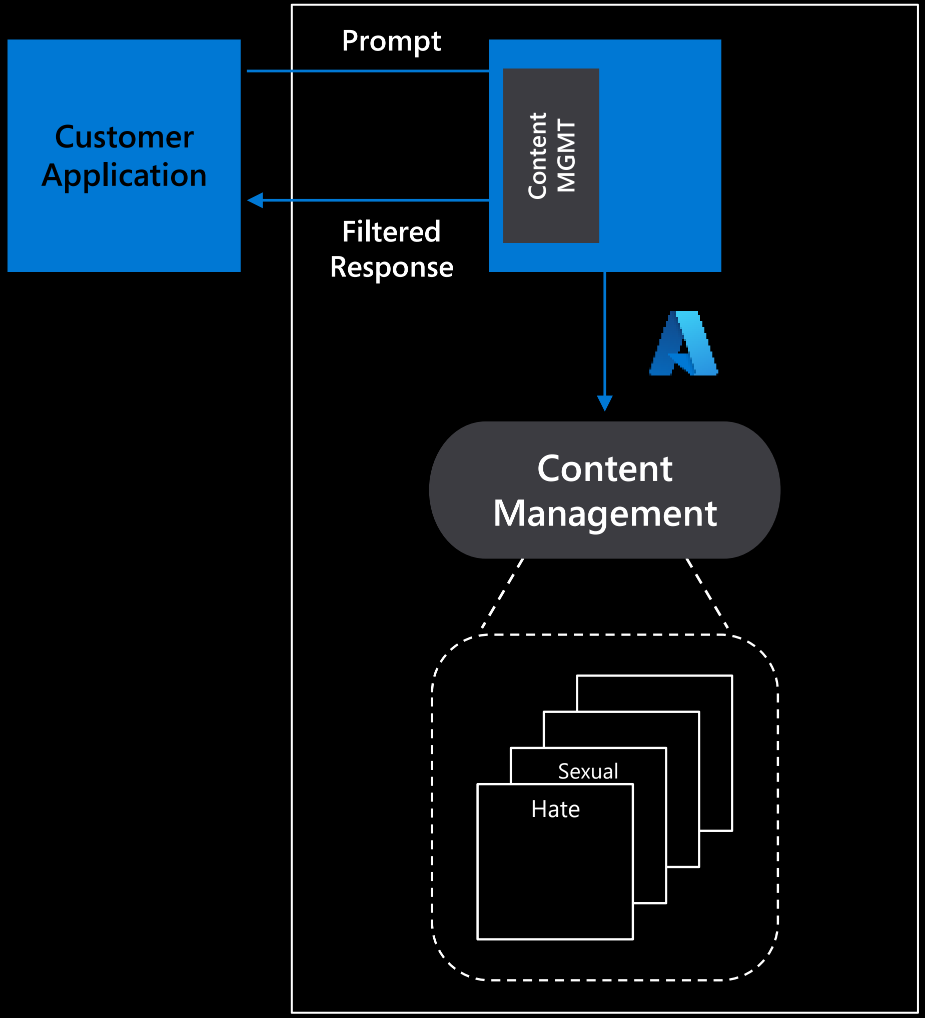
User/Account Actions

Filter Improvement

Azure OpenAI Service Responsible AI

Content filtering—
can filter out abuse and misuse

Implementation guidelines, patterns, and best practices

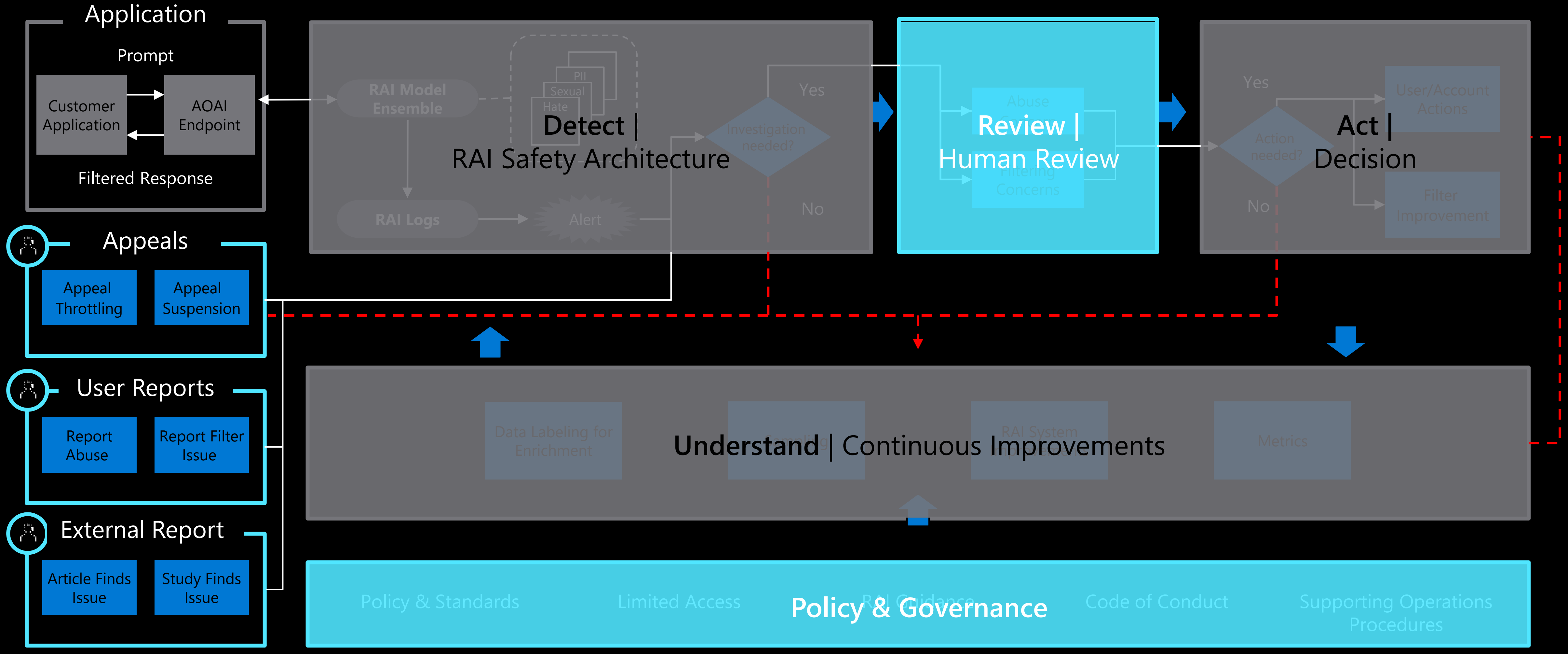


Safety Execution Flow

System/Tech Improvement Loop

Tech

People & Policy



Application

Prompt

Customer Application

AOAI Endpoint

Filtered Response

Appeals

Appeal Throttling

Appeal Suspension

User Reports

Report Abuse

Report Filter Issue

External Report

Article Finds Issue

Study Finds Issue

Detect | RAI Safety Architecture

RAI Model Ensemble

PII

Sexual

Hate

RAI Logs

Alert

Investigation needed?

Yes

No

Review | Human Review

Abuse

Filtering Concerns

Act | Decision

Action needed?

Yes

No

User/Account Actions

Filter Improvement

Understand | Continuous Improvements

Data Labeling for Enrichment

RAI System

Metrics

Policy & Governance

Policy & Standards

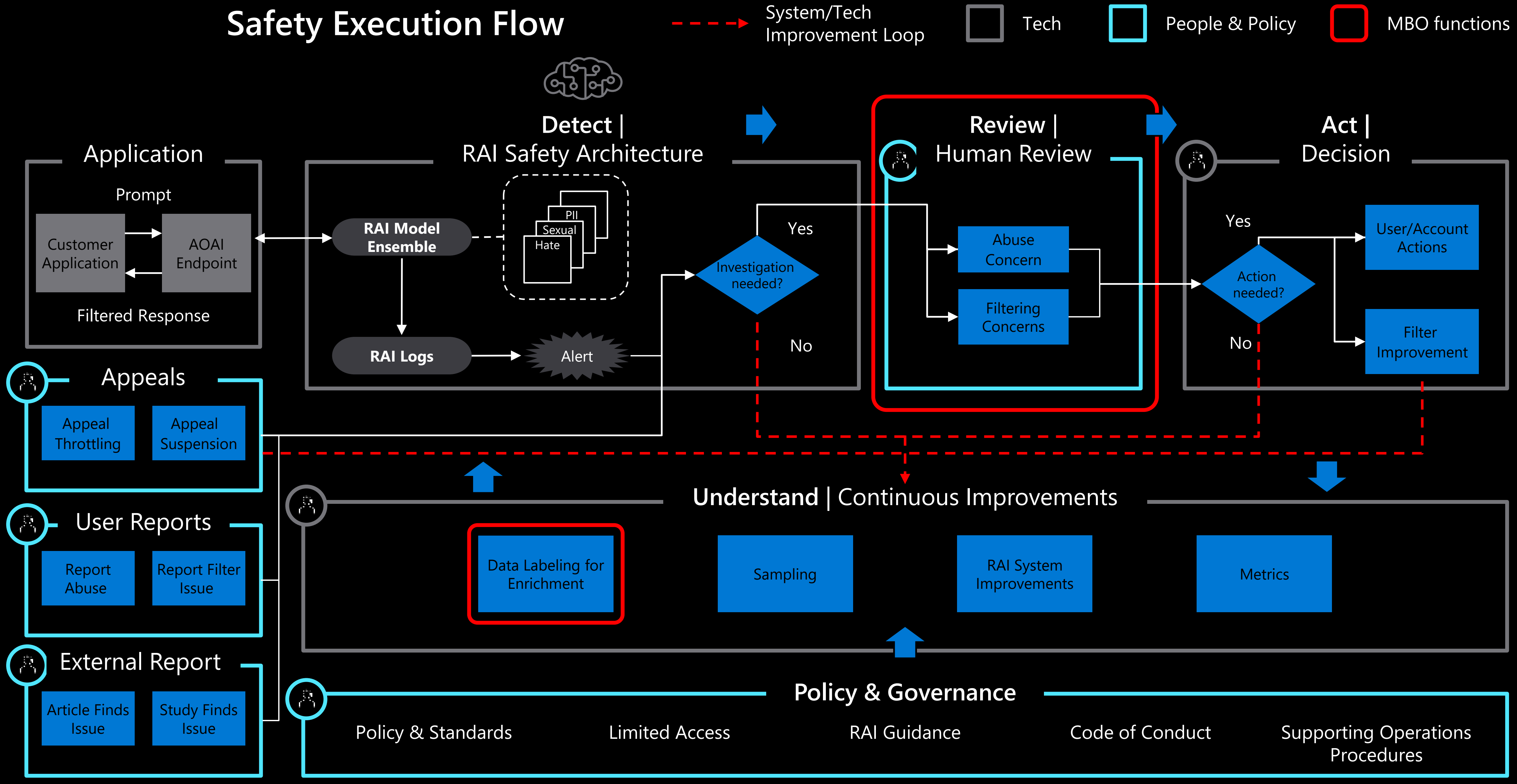
Limited Access

RAI Guidance

Code of Conduct

Supporting Operations Procedures

Safety Execution Flow



Responsible AI resources

Content filtering:
https://learn.microsoft.com/en-us/azure/cognitive-services/openai/concepts/content-filter

Responsible AI resources (aka.ms/RAIResources)

Human and AI Interaction Toolkit
(https://www.microsoft.com/en-us/haxtoolkit/workbook/)

Evaluating and integrating Azure OpenAI for your use

Practices for responsible use

Ensure Human Oversight

- Let people edit generated outputs.
- Highlight potential inaccuracies in generated outputs.
- Remind uses that they are accountable for final decisions and/or final content.
- Limit how people can automate your product or service.

Implement technical limits on inputs and outputs

- Limit the length of inputs and outputs.
- Structure inputs to limit open-ended responses and to give users more refined control.
- Return outputs from validated, reliable source materials.
- Implement blocklists and content moderation.
- Put rate limits in place.

Authenticate Users

- To make misuse more difficult, consider requiring that customers sign in and, if appropriate, link a valid payment method.
- Consider working only with known, trusted customers in the early stages of development.
- Applications that do not authenticate users may require other, stricter mitigations to ensure the application cannot be used beyond its intended purpose.

Test your application thoroughly

- Conduct adversarial testing where trusted testers attempt to find system failures, poor performance, or undesirable behaviors.
- Understand risks and consider appropriate mitigations.
- Communicate the capabilities and limitations to stakeholders.

Establish Feedback Channels for users and impacted groups

- Build feedback features into the user experience.
- Publish an easy-to-remember email address for public feedback.

Evaluating and integrating Azure OpenAI for your use

Scenario-specific practices

If your application powers chatbots or other conversational AI systems

Follow the Microsoft guidelines for responsible development of conversational AI systems

If you are developing an application in a high-stakes domain or industry

In healthcare, human resources, education, or the legal field, thoroughly assess how well the application works in your scenario, implement strong human oversight, thoroughly evaluate how well users understand the limitations of the application, and comply with all relevant laws.

Consider additional mitigations based on your scenario.

Learn more here on our website

Azure Gov Customers Access Patterns

- Direct use of MAC endpoints
- MAG->MAC routing
- On-prem hair pinning (not recommended)

Access Patterns Summary (MAG->MAC)

- For customers primarily in MAG
  - Put the majority of the workload in MAG
  - Limit data transmitted to the MAC AOAI endpoint
  - If any data being retained in MAC is deal breaker
    - Apply to opt out of review process
    - Avoid fine tuning/custom models

Microsoft

Thank you