ALFIYA ABDIMUTALIPOVA

Enhancing Refracted Security Platform with AI Capabilities

2025

Table of contents

Company Overview

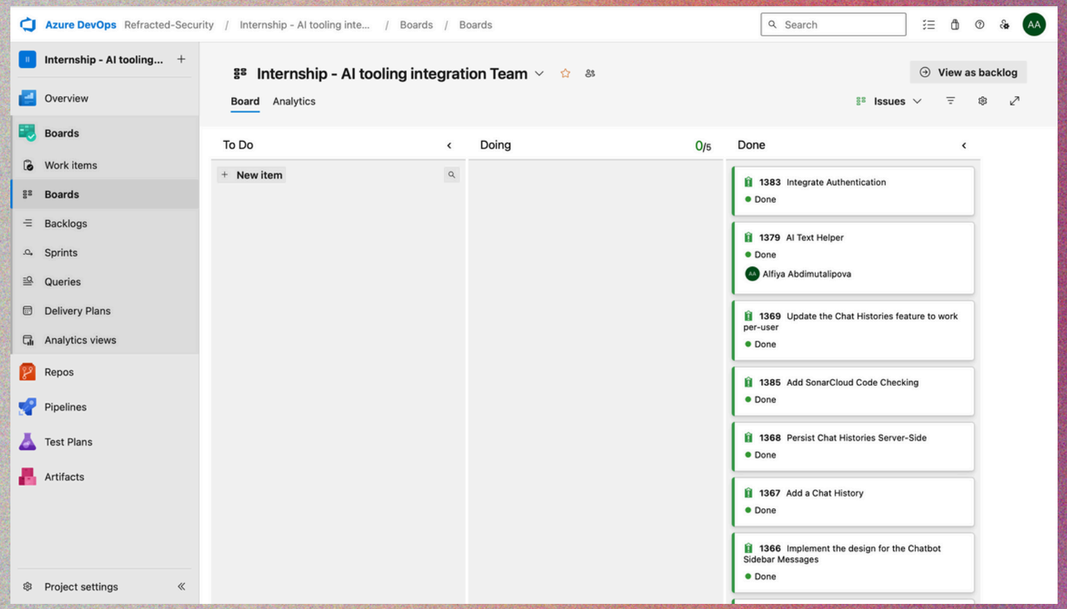
Refracted Security

Specializes in vulnerability management for organizations.

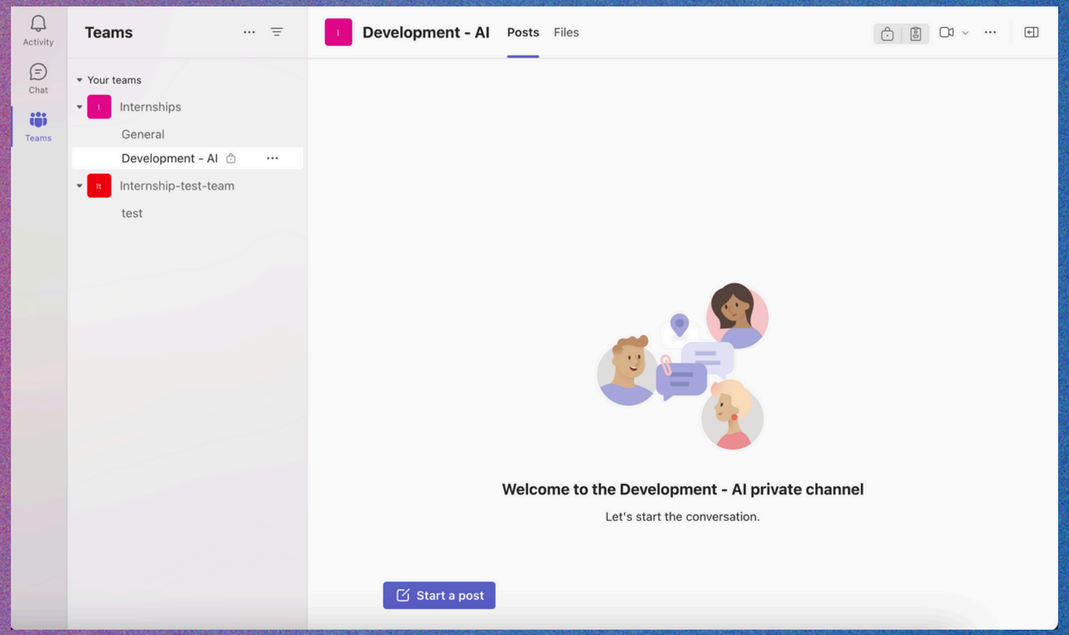
Mission: Provide effective tools for assessing, addressing, and mitigating vulnerabilities.

REFRACTED security

Collaboration Tools



Azure DevOps



Microsoft Teams

Key Deliverables

Overall Goal: Develop AI-based features to enhance the platform’s efficiency and user experience.

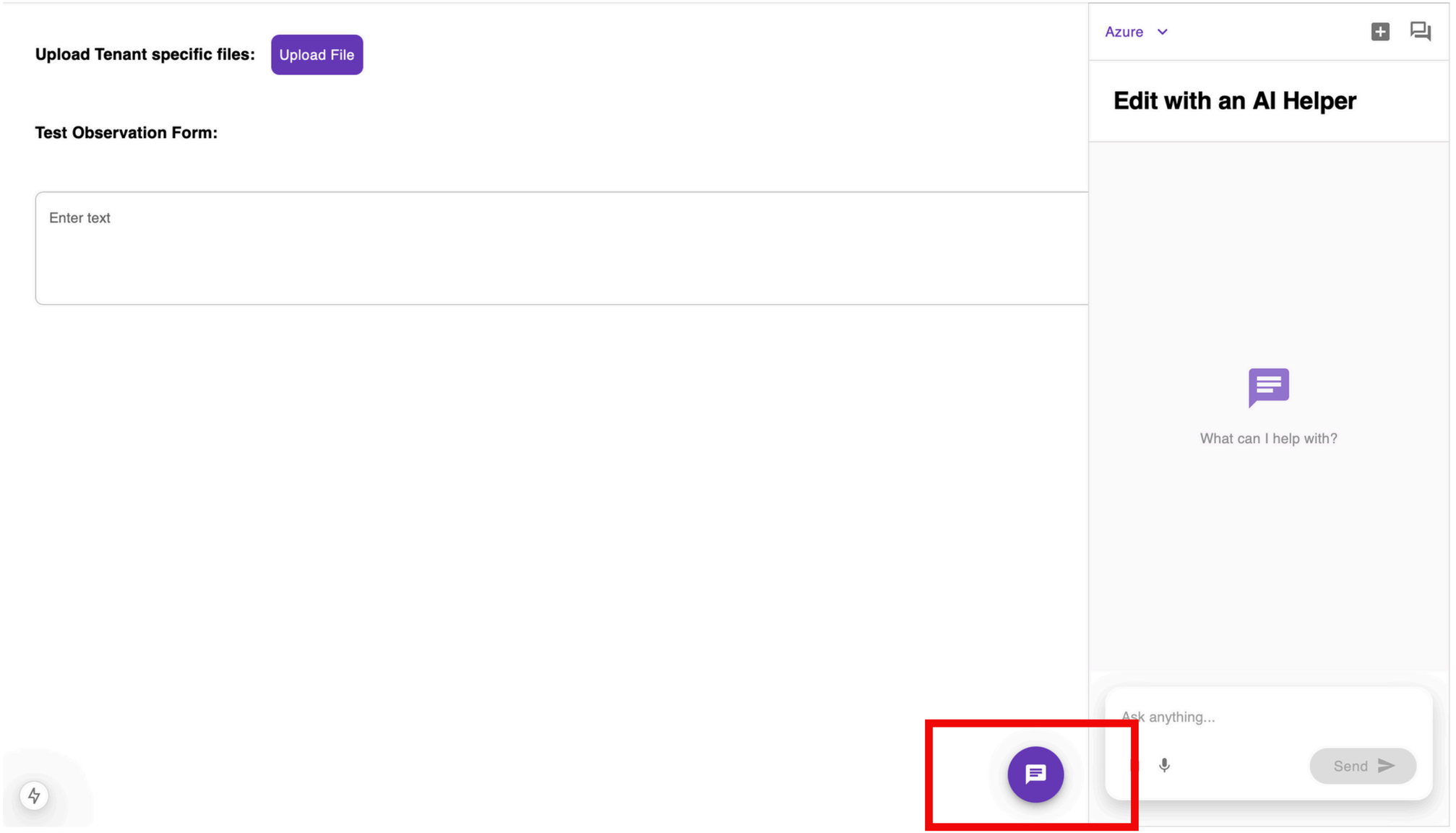
1. **Functional Sidebar Chat Helper**

2. **Observations Improvement Feature**

3. **Automated data gathering**

Functional Sidebar Chat Helper:

Floating button that expands into a sidebar chat interface.

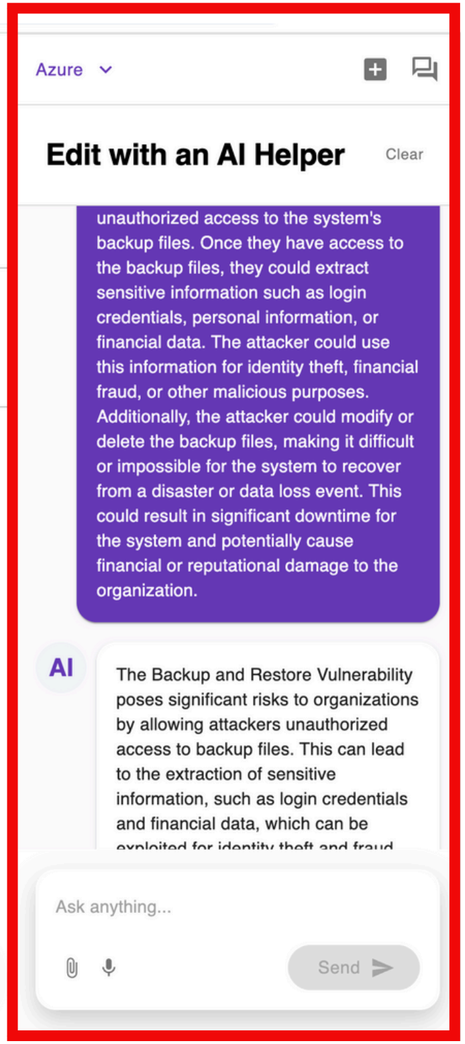


Responsive UI using Material UI and React components.

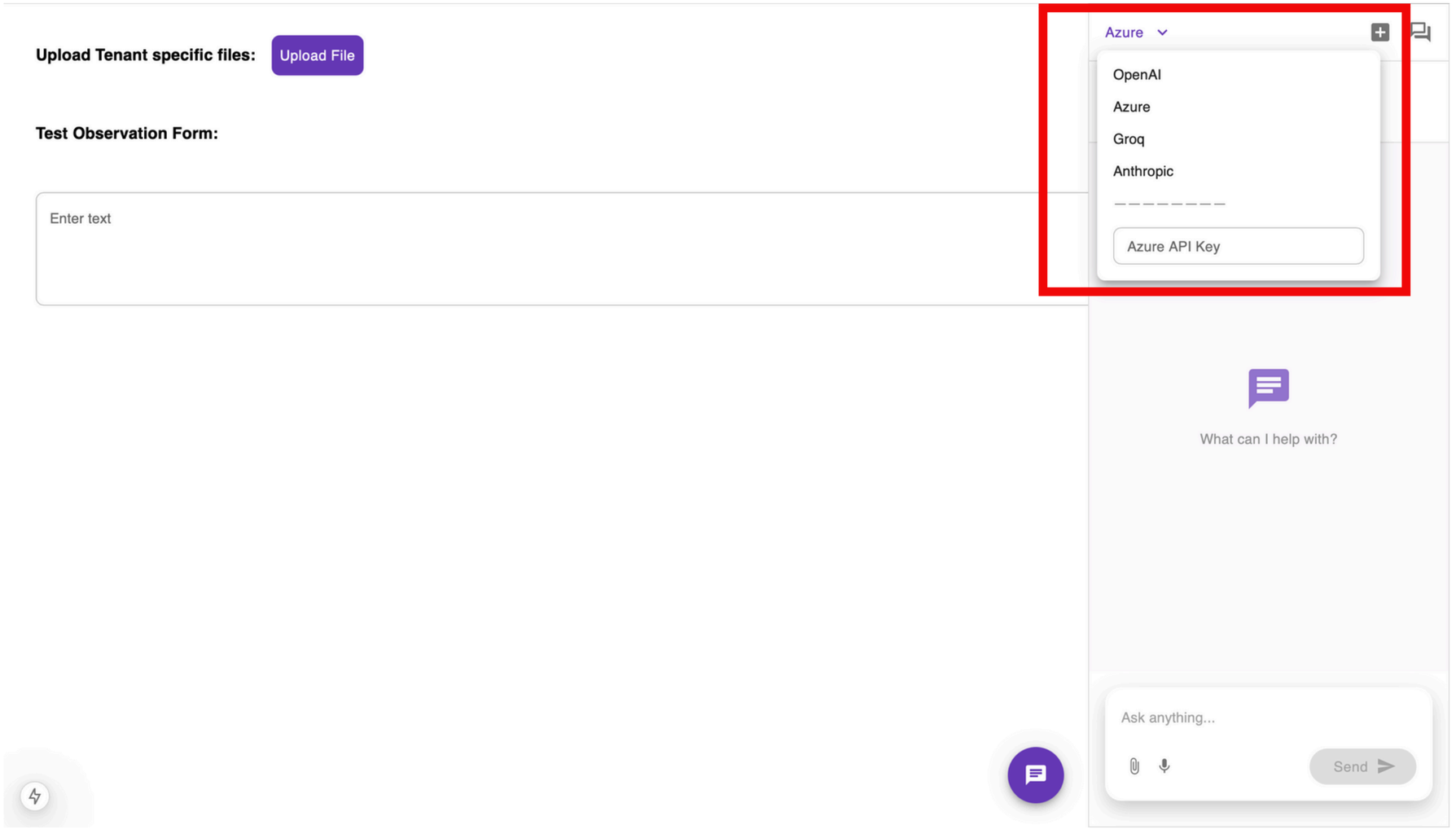
Upload Tenant specific files: Upload File

Test Observation Form:

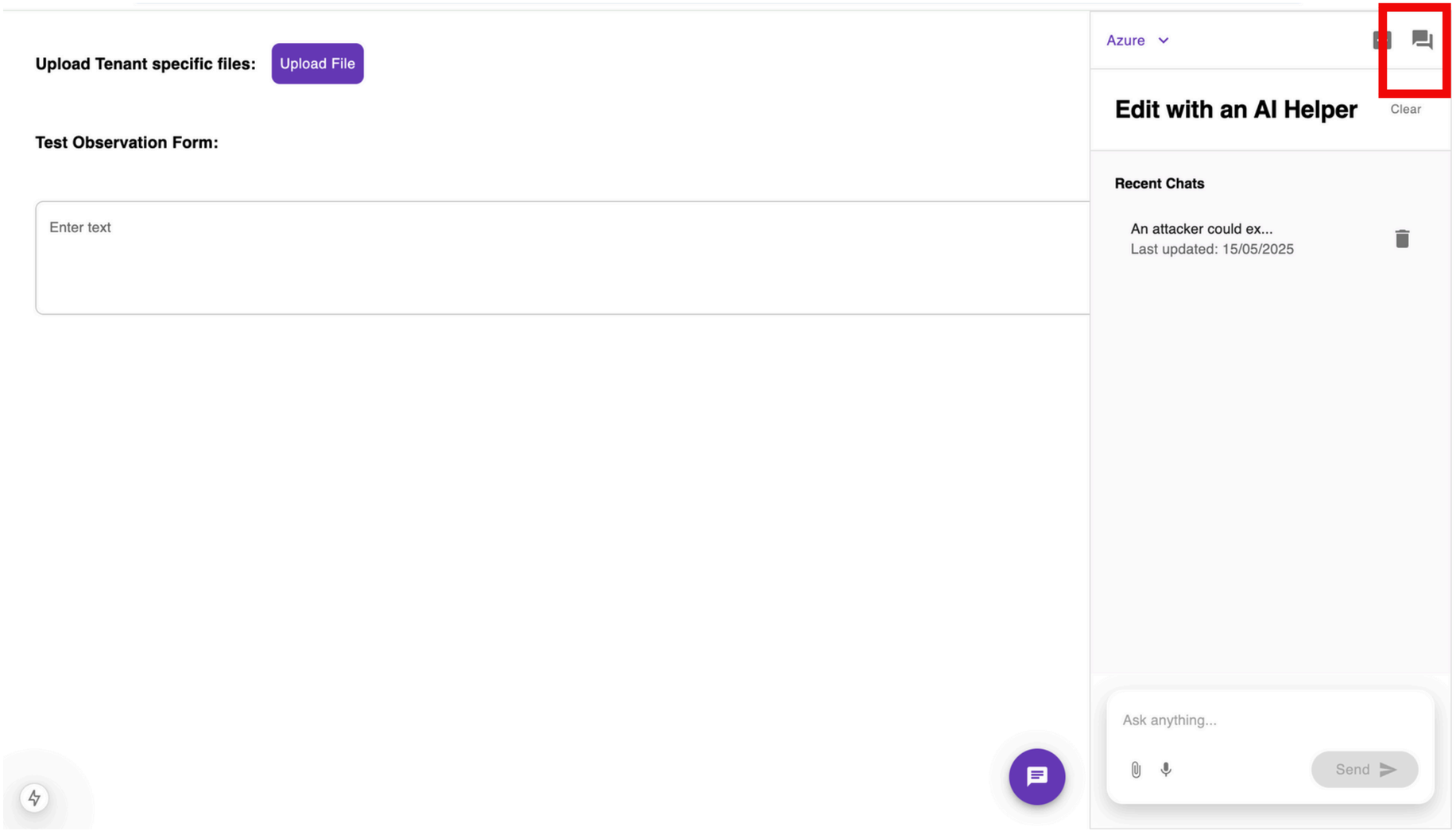
Enter text



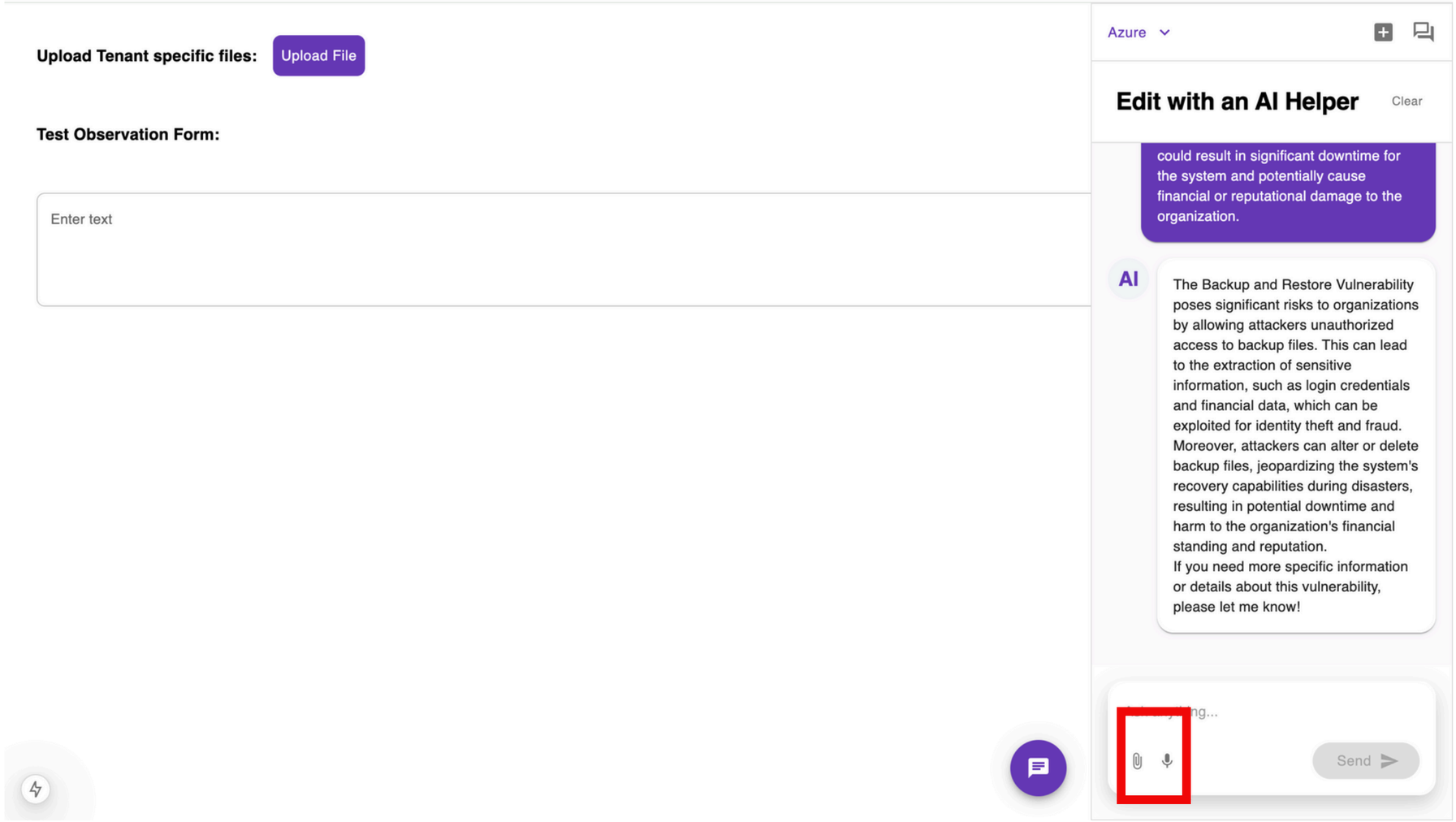
Supports multiple AI models (e.g., GPT variants via Azure OpenAI)



Persistent chat history across sessions.

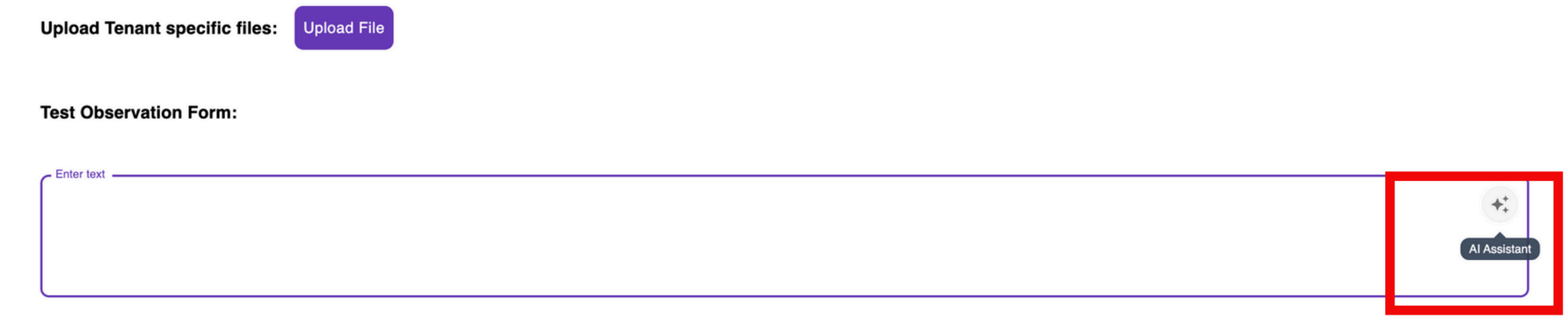


File Upload and Voice Input Possibilities

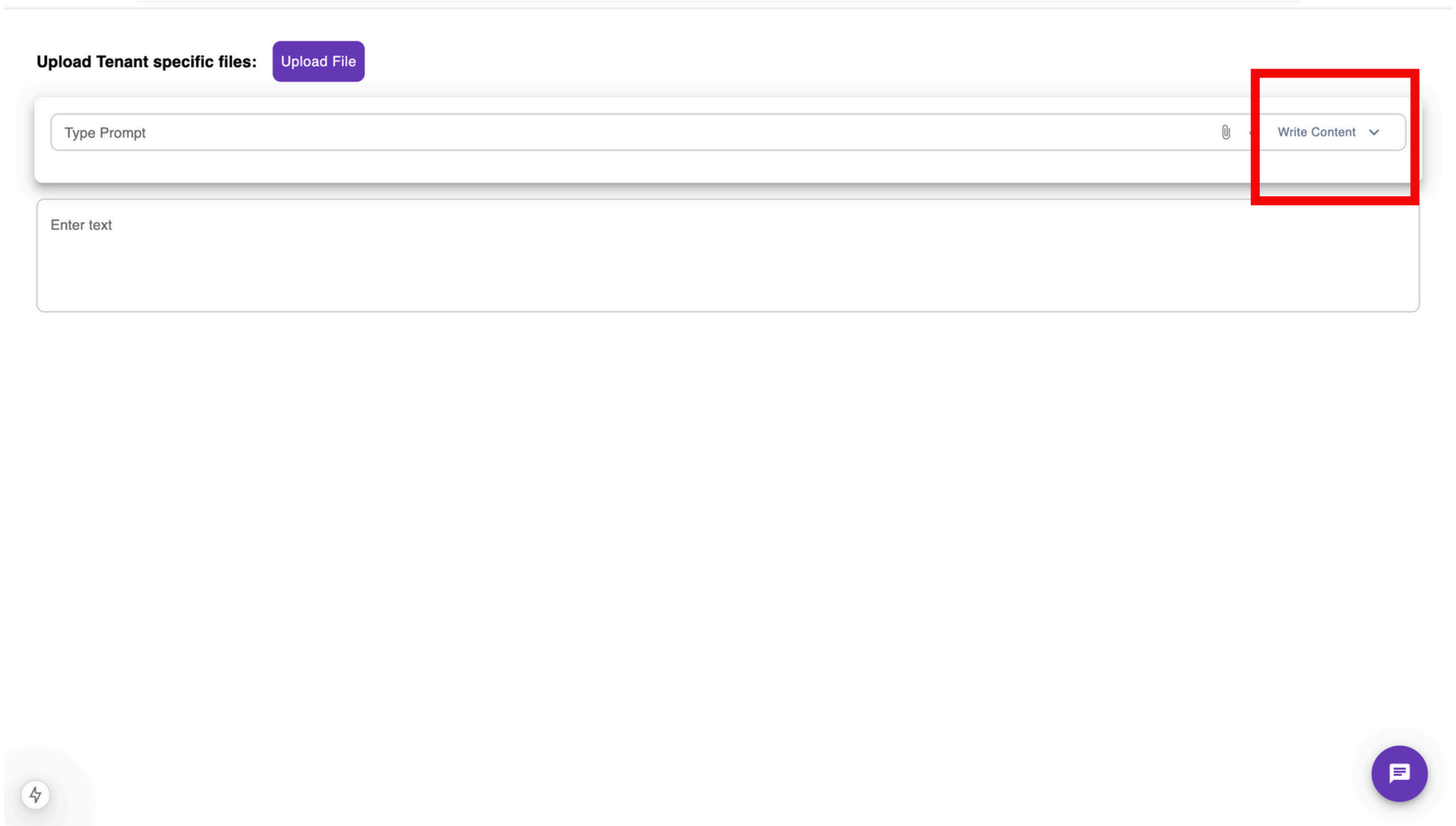


Observations Improvement Feature

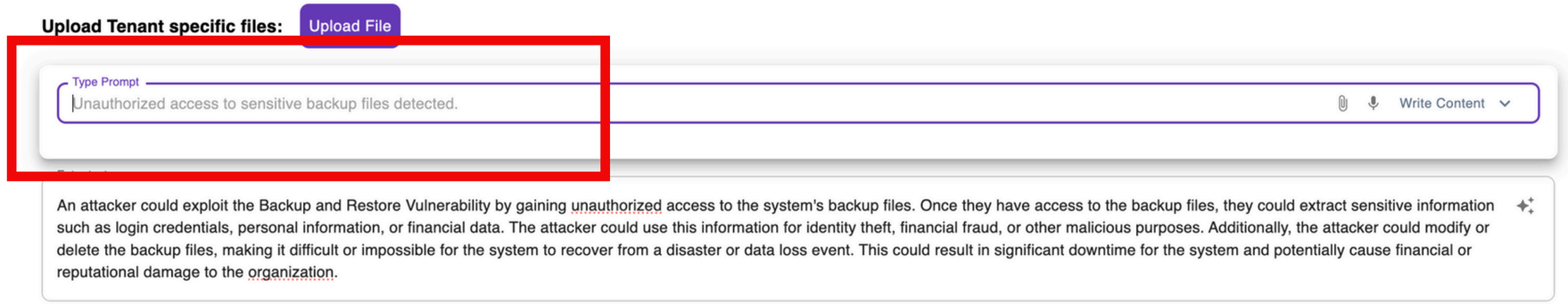
Dialog-based UI triggered by an AI button dynamically placed at the end of editable fields.



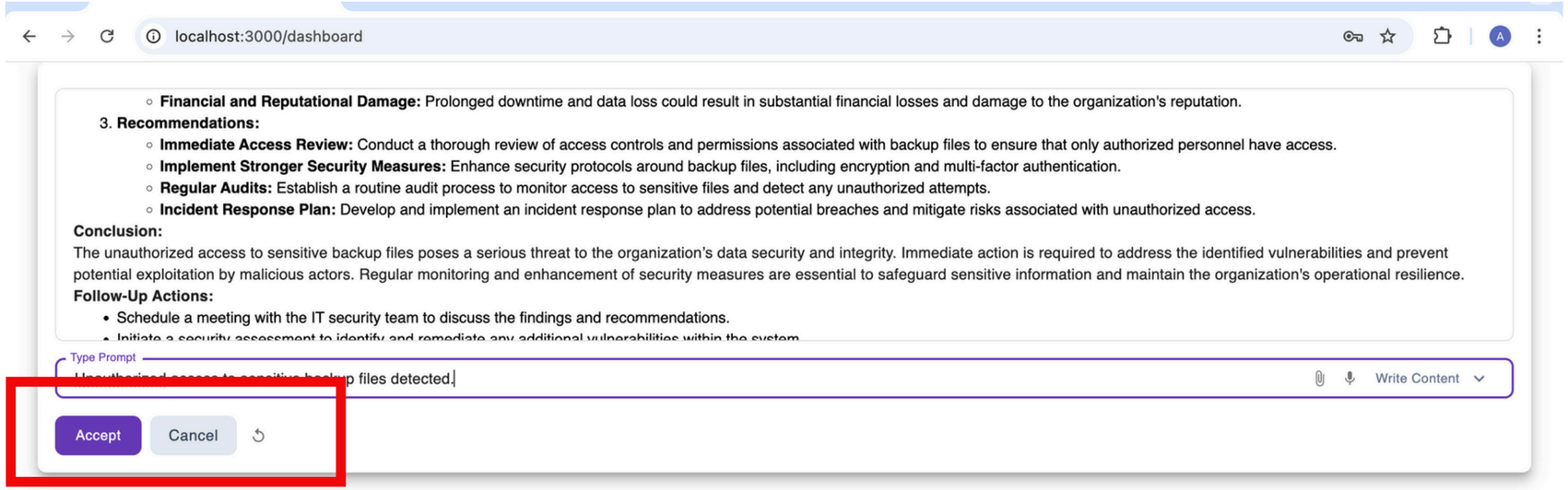
Supports multiple actions: summarize, rewrite, generate new content



AI-placeholder suggestion



Users can accept or reject AI-generated suggestions.



Security Levels

Overall Goal: Develop Multi-Tenant Security Architecture

1. **Public Sources**
2. **Tenant Specific Files**
3. **User Temporary Files**

Security Levels

Overall Goal: Develop Multi-Tenant Security Architecture

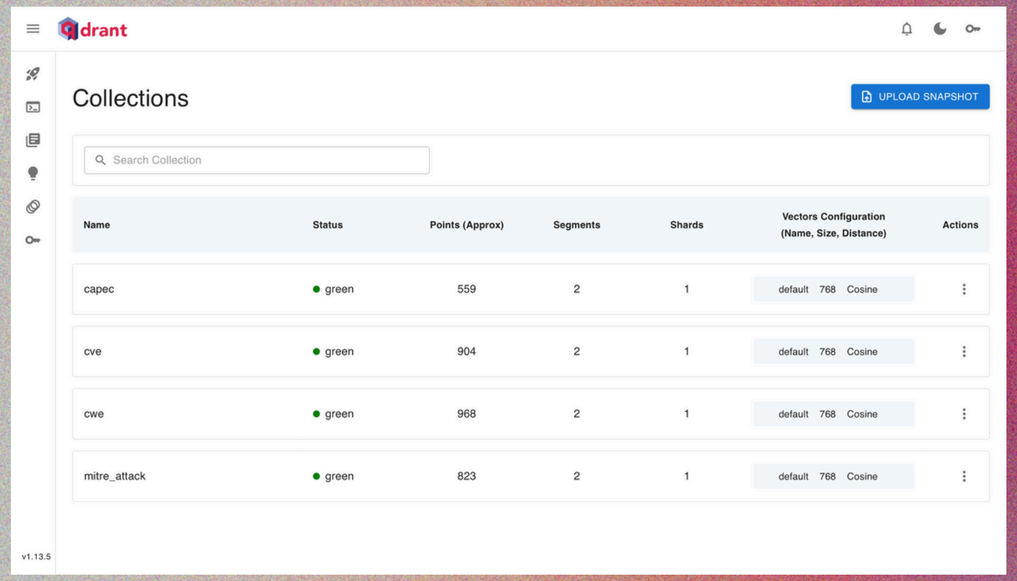
1. **Public Sources**
2. Tenant Specific Files
3. User Temporary Files

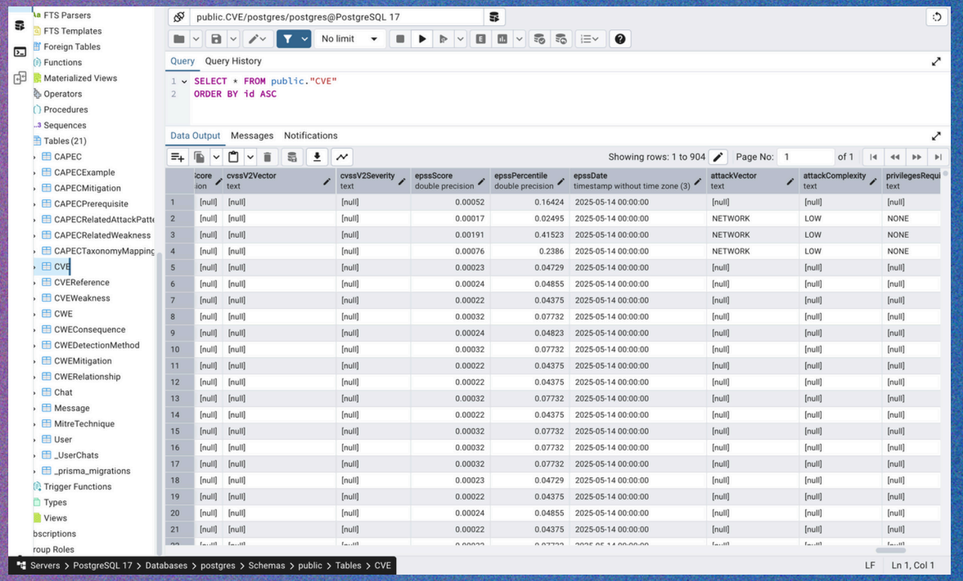
Data Sources Integrated

1. CVEs - Retrieved from APIs (limited to last 7 days)
2. EPSS Scores – Stored together, but could be updated separately
3. CAPEC & CWE - XML and JSON formats, stored with metadata
4. MITRE ATT&CK - Accessed via public API

Automated data gathering

Hybrid Search





Vector Database

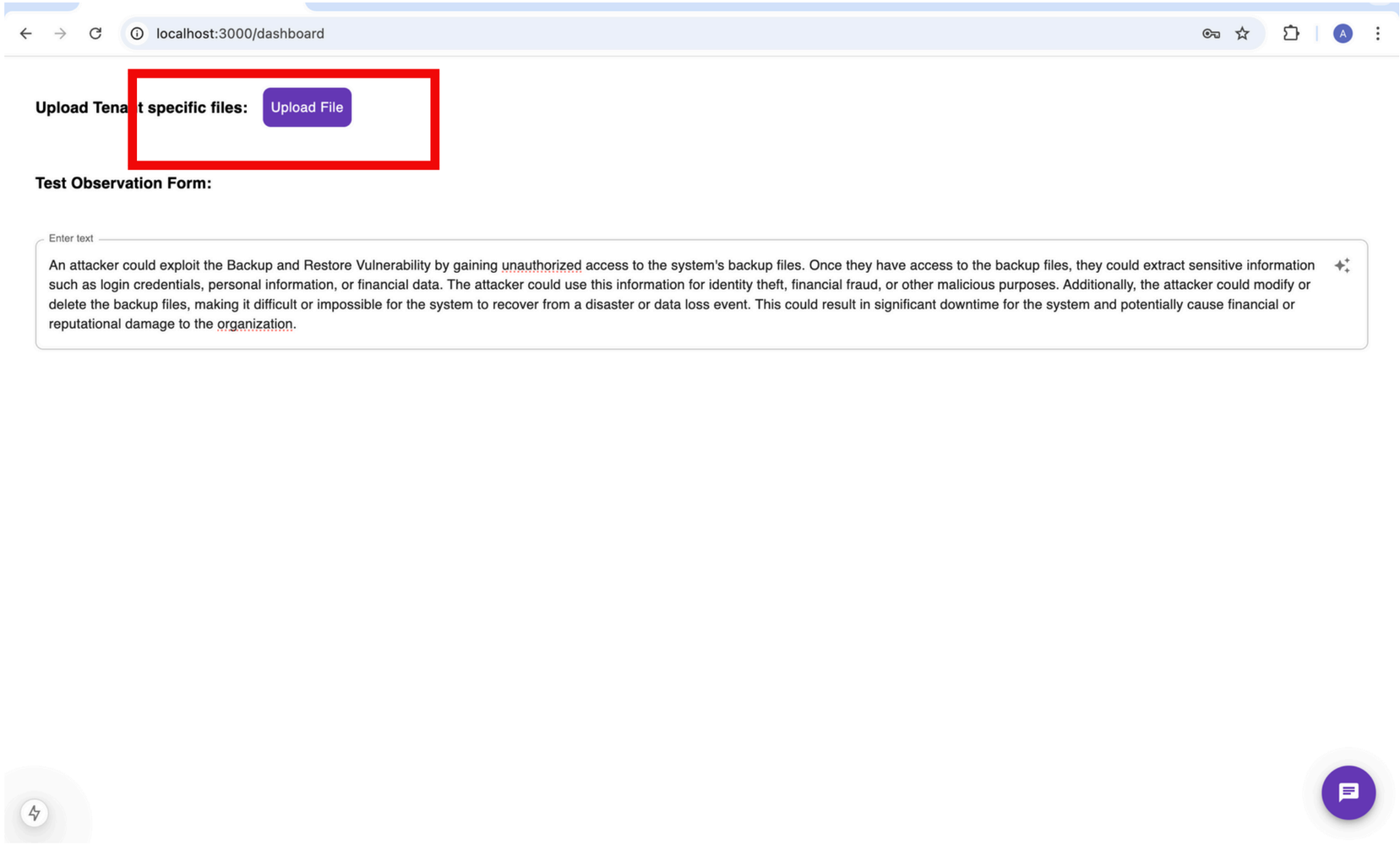
PostgreSQL

Security Levels

Overall Goal: Develop Multi-Tenant Security Architecture

1. Public Sources
2. **Tenant Specific Files**
3. User Temporary Files

Client-specific data and configurations

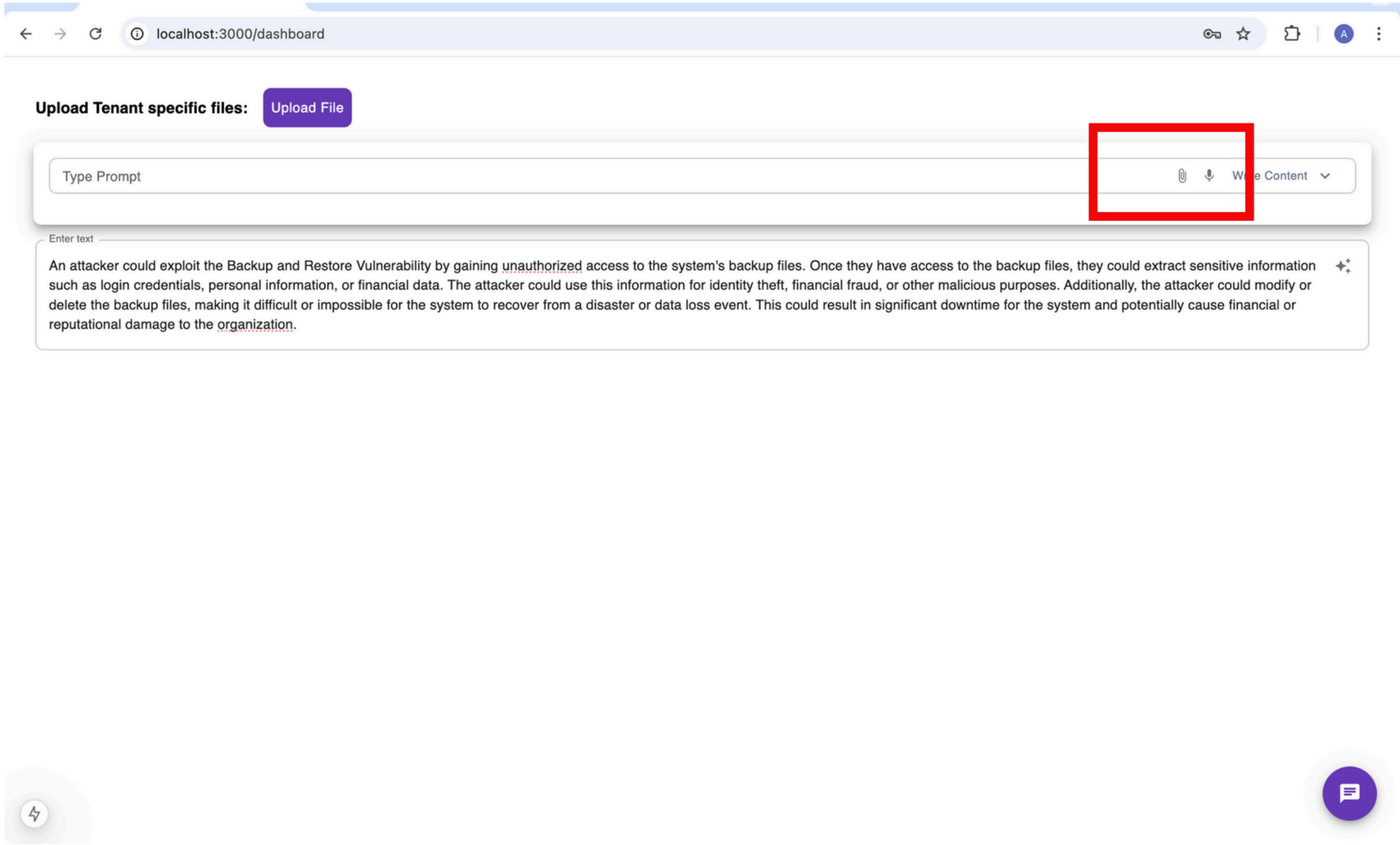


Security Levels

Overall Goal: Develop Multi-Tenant Security Architecture

1. Public Sources
2. Tenant Specific Files
3. **User Temporary Files**

Temporary session-based files (cleared after use)



Challenges

1. No Standardized Update Schedule for Public Datasets

2. Unstable Public API Endpoints

3. Qdrant Delete Request Limitation

Requirements for Integration

Self-Hosting Large Language Model (LLM)

Hosting an Embedding Model

Project Planning

March - Mid-April

Finalize advanced conversation logic in the sidebar chat.

Integrate RAG for enhanced contextual responses.

Mid-April - Early May

Implement Multi-Tenant Security Layers

Fetch and Embed Public Datasets

Early May - Internship End

Implement automated retraining and data fetching.

Conduct performance evaluation and create documentation.

Technology Stack

LAYER	TECHNOLOGY	PURPOSE
Frontend	React, Next.js, Material UI	UI components, routing, responsive design
Backend	Node.js, TypeScript, Prisma ORM, PostgreSQL	Server logic, API routes, data persistence
AI & RAG	LangChain, Azure OpenAI, ONNX Runtime, Qdrant	Prompt chaining, LLM interaction, embedding generation, vector search
Data Handling	JSON/XML/OWL parsers, multi-retriever chains	Normalizing and retrieving public datasets
Security	Multi-level access control, secure API layer	Preventing data leakage, tenant isolation
Dev Tools	Git, GitHub, SonarCloud, VS Code	Code management, quality checks, development

Thank you