Cybersecurity Risk Assessment Platform Overview

Table of Contents

1. Platform Overview
2. Technical Stack
3. System Architecture
4. User Flow
5. Execution Flow
6. Key Functions
7. Data Models
8. Error Handling
9. Security Features

Platform Overview

The Cybersecurity Risk Assessment Platform is a comprehensive solution that leverages Monte Carlo simulation to quantify and analyze cybersecurity risks. The platform combines historical data analysis, industry-specific insights, and AI-driven recommendations to provide a detailed risk assessment.

Key Features

- Quantification of cybersecurity risks
- Historical incident analysis
- AI-driven remediation strategies
- Visual risk metrics
- Dynamic question generation
- Industry-specific analysis

Technical Stack

Frontend

- React with TypeScript
- Material-UI components
- Custom charts for data visualization
- Vite for development

Backend

- FastAPI (Python)
- NumPy and Pandas for data processing
- GPT-4 Mini client for AI integration
- Matplotlib for visualization
- CSV-based data storage (UMD Cyber Events Database)

System Architecture

Frontend Structure

- `InitialInputForm.tsx`: Initial data collection
- `Summary.tsx`: Results display
- `RiskMetricsDisplay.tsx`: Risk visualization
- `DynamicQuestions.tsx`: Dynamic question handling
- `IndustryAnalysis.tsx`: Industry-specific analysis
- `HistoricalAnalysis.tsx`: Historical data analysis
- `RemediationStrategies.tsx`: Strategy recommendations

Backend Structure

- `main.py`: FastAPI application
- `risk_processor.py`: Core processing logic
- `historical_analyzer.py`: Historical data analysis
- `risk_state.py`: State management
- `gpt4_mini_client.py`: AI integration

User Flow

1. **Initial Input Collection**
   - Organization details
   - Industry information
   - Company size
   - Location data
2. **Dynamic Question Generation**
   - Security measures
   - Incident history
   - Data protection
   - Response plans
   - Training programs
   - Technical measures
3. **Industry Analysis**
   - Industry-specific threats
   - Common vulnerabilities
   - Regulatory requirements
   - Best practices
4. **Historical Analysis**
   - Similar incidents
   - Impact patterns

    - Frequency analysis
    - Trend identification
5. **Monte Carlo Simulation**
    - Risk scenarios
    - Probability distributions
    - Impact calculations
    - Confidence intervals
6. **Remediation Strategy Generation**
    - AI-driven recommendations
    - Priority-based actions
    - Resource allocation
    - Implementation timeline

Execution Flow

API Endpoints

1. `/api/initial-input`
    - Processes initial organization data
    - Generates initial risk metrics
    - Creates dynamic questions
2. `/api/dynamic-questions`
    - Updates risk metrics based on responses
    - Adjusts confidence levels
    - Refines analysis
3. `/api/industry-analysis`
    - Conducts industry-specific analysis
    - Identifies common threats
    - Generates industry insights
4. `/api/historical-analysis`
    - Analyzes historical incidents
    - Calculates similarity scores
    - Generates risk adjustments
5. `/api/simulate_risk`
    - Runs Monte Carlo simulations
    - Generates probability distributions
    - Calculates risk metrics
6. `/api/get_remediation_strategies`
    - Generates AI-driven recommendations
    - Prioritizes actions
    - Creates implementation plans

Key Functions

Risk Processor (risk_processor.py)

Core Flow Functions

- `process_initial_input`: Main entry point for initial risk assessment
- `process_dynamic_questions`: Handles dynamic question responses

- `process_industry_analysis`: Processes industry-specific analysis
- `process_historical_analysis`: Integrates historical data
- `process_simulation`: Runs Monte Carlo simulation
- `process_remediation_strategies`: Generates remediation recommendations

Support Functions

- `_calculate_initial_tef`: Calculates initial threat event frequency
- `_calculate_initial_vulnerability`: Calculates initial vulnerability
- `_adjust_metrics_for_industry`: Adjusts metrics based on industry
- `_adjust_metrics_for_historical`: Adjusts metrics based on historical data
- `_adjust_metrics_for_simulation`: Adjusts metrics for simulation
- `_adjust_metrics_for_remediation`: Adjusts metrics for remediation

Historical Analyzer (historical_analyzer.py)

Core Flow Functions

- `find_similar_incidents`: Main function to find similar historical incidents
- `calculate_risk_adjustments`: Calculates risk adjustments from historical data
- `get_historical_analysis`: Compiles complete historical analysis

Support Functions

- `_calculate_similarity_score`: Calculates similarity between incidents
- `_process_umd_incidents`: Processes UMD database incidents
- `_process_provided_incidents`: Processes provided database incidents
- `_calculate_frequency_factor`: Calculates frequency adjustment factor
- `_calculate_magnitude_factor`: Calculates magnitude adjustment factor
- `_calculate_confidence_score`: Calculates confidence in adjustments

Risk State (risk_state.py)

Core Flow Functions

- `set_user_inputs`: Stores initial user inputs
- `update_risk_metrics`: Updates risk metrics
- `get_current_state`: Returns current state
- `set_selected_scenario`: Sets selected risk scenario
- `set_dynamic_questions`: Stores dynamic questions
- `add_question_answer`: Stores question answers

Support Functions

- `_validate_metrics`: Validates risk metrics
- `_normalize_metrics`: Normalizes metric values
- `_calculate_aggregates`: Calculates aggregate metrics

Data Models

Request Models

- `InitialInputRequest`
- `DynamicQuestionRequest`
- `IndustryAnalysisRequest`
- `HistoricalAnalysisRequest`
- `SimulationRequest`
- `RemediationRequest`

Response Models

- `RiskMetricsResponse`
- `DynamicQuestionsResponse`
- `IndustryAnalysisResponse`
- `HistoricalAnalysisResponse`
- `SimulationResponse`
- `RemediationResponse`

Error Handling

Strategies

- Comprehensive logging
- Input validation
- Graceful degradation
- Error recovery
- State preservation

Logging Levels

- INFO: Normal operations
- WARNING: Potential issues
- ERROR: Critical failures
- DEBUG: Detailed information

Security Features

API Security

- CORS configuration
- Input validation
- Rate limiting
- Authentication
- Authorization

Data Protection

- Secure API endpoints
- Data encryption
- Access control
- Audit logging
- Compliance checks

Process Flow

1. Initial input → `process_initial_input`
2. Dynamic questions → `process_dynamic_questions`
3. Industry analysis → `process_industry_analysis`
4. Historical analysis → `process_historical_analysis` → `find_similar_incidents`
5. Simulation → `process_simulation`
6. Remediation → `process_remediation_strategies`

Each step uses support functions to calculate specific metrics and adjustments, with the state being maintained throughout the process.