CyberSaga: Immersive Cybersecurity Education Platform

Introduction

CyberSaga is an innovative, immersive cybersecurity education platform designed to transform abstract security concepts into interactive, personalized learning adventures. The application leverages AI-driven scenario generation to create realistic cybersecurity challenges that adapt to each user's industry, role, and skill level. Through a gamified approach, users navigate through various security scenarios where their decisions have meaningful consequences, providing practical experience in a risk-free environment.

This documentation provides a comprehensive overview of the CyberSaga platform, detailing its architecture, components, functionality, and implementation. It serves as both a technical reference and a guide to understanding the educational methodology behind the application.

Project Overview

Vision and Mission

CyberSaga was developed with the vision of making cybersecurity education accessible, engaging, and effective for individuals across various industries and experience levels. The platform's mission is to:

1. Demystify complex cybersecurity concepts through interactive storytelling
2. Provide personalized learning experiences that adapt to each user's context
3. Build practical security skills through decision-based scenarios
4. Track progress and identify knowledge gaps for targeted improvement
5. Foster a security-conscious mindset applicable to real-world situations

Target Audience

The platform is designed for:

- Professionals across various industries (healthcare, finance, education, technology, etc.)
- Individuals with different roles (executives, managers, IT professionals, administrative staff)
- Users with varying levels of cybersecurity experience (beginner, intermediate, advanced)
- Organizations seeking to improve their security awareness training programs

Key Features

- Personalized Scenarios: AI-generated cybersecurity challenges tailored to the user's industry, role, and experience level
- Decision-Based Learning: Interactive scenarios where users make security decisions and receive immediate feedback
- Knowledge Assessment: Post-scenario quizzes to evaluate understanding of key security concepts
- Learning Moments: Contextual explanations of security principles triggered by user decisions
- Progress Tracking: Comprehensive dashboard to monitor skill development and identify areas for improvement

- Certification: Downloadable certificates upon successful completion of scenarios

Technical Architecture

Tech Stack

CyberSaga is built using the following technologies:

- Frontend Framework: Streamlit (Python-based web application framework)
- Backend Language: Python 3.8+
- AI Integration: Groq LLM API (llama-3.3-70b-versatile model)
- Data Storage: Local JSON files (for user profiles and scenario data)
- Image Processing: PIL (Python Imaging Library) for certificate generation
- Styling: Custom CSS with responsive design principles
- Environment Management: dotenv for configuration

System Requirements

- Python 3.8 or higher
- Internet connection (for AI API calls)
- Modern web browser
- Minimum 4GB RAM
- 100MB disk space

Dependencies

The application relies on several Python packages:

- streamlit
- agno
- PIL (Pillow)
- python-dotenv
- uuid
- datetime
- json
- base64
- typing

Core Components

The CyberSaga platform consists of several interconnected components, each responsible for specific functionality:

1. Main Application (app.py)

The central component that orchestrates the entire application flow. It initializes the Streamlit interface, manages session state, and coordinates interactions between other components. Key responsibilities include:

- Setting up the Streamlit page configuration
- Initializing session state variables
- Managing navigation between different application views
- Rendering UI components and handling user interactions
- Coordinating scenario execution and assessment

2. AI Agent (agent.py)

The intelligence behind the platform, responsible for generating personalized content and analyzing user decisions. The `SecurityGuideAgent` class leverages the Groq LLM API to:

- Generate cybersecurity scenarios based on user profiles
- Create decision points with multiple options
- Analyze user decisions and provide feedback
- Generate learning moments related to security principles
- Create knowledge assessment questions
- Provide personalized recommendations

3. Scenario System (scenarios.py)

Defines the structure and types of cybersecurity scenarios available in the platform. It includes:

- Base `Scenario` class with common properties and methods
- Specialized scenario classes for different security domains:
  - `PhishingScenario`
  - `RansomwareScenario`
  - `SocialEngineeringScenario`
  - `DataProtectionScenario`
  - `NetworkSecurityScenario`
- `DecisionPoint` and `LearningMoment` data classes
- Factory function for creating scenarios of different types

4. User Profile Management (user_profile.py)

Handles user data, progress tracking, and personalization. The `UserProfile` class manages:

- User personal information (name, email, industry, role)
- Progress tracking (completed scenarios, points, skill levels)
- User preferences (difficulty, focus areas)
- Persistent storage of profile data
- Recommendation algorithms based on past performance

5. Certificate Generator (certificate_generator.py)

Creates visually appealing certificates upon scenario completion. Features include:

- Generation of high-resolution certificate images
- Dynamic text placement based on content length
- Visual styling with borders, colors, and decorative elements
- Base64 encoding for web display
- Download functionality for saving certificates

6. Prompts System (prompts.py)

Contains the structured prompts used to guide the AI agent in generating content. These include:

- System prompt defining the agent's role
- Scenario generation prompts
- Decision point generation prompts
- Decision analysis prompts
- Learning moment prompts
- Assessment generation prompts
- Recommendation prompts

User Flow

The CyberSaga platform follows a structured user flow designed to provide a seamless and educational experience:

1. Welcome and Onboarding

- User arrives at the welcome page with an introduction to CyberSaga
- Onboarding form collects personal information:
    - Name

    - Email
    - Industry (healthcare, finance, education, etc.)
    - Role (executive, manager, IT professional, etc.)
    - Cybersecurity experience level (beginner, intermediate, advanced)
- Information is used to initialize the user profile and personalize future content
- Option to skip onboarding for demo mode

2. Scenario Selection

- User is presented with available cybersecurity scenarios
- Each scenario displays:
    - Title
    - Domain (phishing, ransomware, social engineering, etc.)
    - Brief description
    - Difficulty level
    - Estimated completion time
- User can select number of assessment questions (3-7)
- Scenarios are presented in a card-like format with visual indicators
- User selects a scenario to begin

3. Scenario Execution

- Selected scenario is loaded with a detailed description
- User progresses through a series of decision points:
    - Each decision point presents a question with multiple options
    - User selects their response
    - System provides immediate feedback on the decision
    - Feedback includes explanation of why the decision was correct or incorrect
    - Learning moments are presented to reinforce key security concepts
- Progress is tracked throughout the scenario
- User's decisions are recorded for later analysis

4. Knowledge Assessment

- Upon completing all decision points, user proceeds to knowledge assessment
- Assessment contains multiple-choice questions testing understanding of key concepts
- Questions are generated based on the scenario domain and user's profile
- User submits answers and receives immediate feedback
- System calculates overall score based on decision accuracy and assessment performance

5. Scenario Summary

- Comprehensive summary of the scenario experience is presented
- Summary includes:
    - Decision history with correctness indicators
    - Learning moments encountered
    - Assessment results with explanations
    - Overall performance score
- User can review their decisions and the associated learning points

6. Certificate Generation

- Certificate of completion is generated based on user's performance
- Certificate includes:
    - User's name
    - Scenario title
    - Completion date
    - Overall score
- User can download the certificate as a PNG file
- Navigation options to choose another scenario or view progress dashboard

7. Progress Dashboard

- Dashboard displays overall progress across all completed scenarios
- Visualizations show:
    - Skill levels in different security domains
    - Completed scenarios with performance metrics
    - Knowledge gaps and strengths
    - Recommendations for improvement
- User can navigate back to scenario selection or choose recommended scenarios

Scenario System

The scenario system is the core educational component of CyberSaga, designed to provide realistic cybersecurity challenges that adapt to each user's context.

Scenario Structure

Each scenario consists of:

1. Basic Information:
    - Title
    - Description
    - Security domain (phishing, ransomware, etc.)
    - Difficulty level
    - Industry context
2. Decision Points:
    - Series of questions with multiple options
    - Each option has associated consequences
    - One option is marked as correct (best security practice)
    - Feedback for each option explains security implications
3. Learning Moments:
    - Triggered by user decisions
    - Explain key security principles relevant to the scenario

- Provide practical tips and best practices
- Reinforce learning through contextual explanations

Scenario Types

The platform includes specialized scenario types for different security domains:

1. Phishing Scenarios:
    - Focus on email and message-based social engineering
    - Teach identification of suspicious communications
    - Cover various phishing types (spear phishing, whaling, etc.)
2. Ransomware Scenarios:
    - Simulate ransomware attack situations
    - Focus on prevention, detection, and response
    - Include decision points on ransom payment considerations
3. Social Engineering Scenarios:
    - Cover in-person and phone-based manipulation
    - Teach verification procedures and security protocols
    - Include various attack vectors (pretexting, baiting, etc.)
4. Data Protection Scenarios:
    - Focus on handling sensitive information
    - Cover compliance requirements and best practices
    - Include scenarios on data breaches and response
5. Network Security Scenarios:
    - Address wireless and remote access security
    - Cover secure configuration and monitoring
    - Include scenarios on unusual network activity

Scenario Generation

Scenarios are generated dynamically using the AI agent, which:

1. Creates a narrative based on the user's industry and role

2. Develops decision points of appropriate difficulty
3. Ensures educational value through learning moments
4. Maintains narrative coherence throughout the scenario
5. Balances realism with educational objectives

AI Integration

CyberSaga leverages advanced AI capabilities to create personalized, adaptive learning experiences.

AI Agent Architecture

The `SecurityGuideAgent` class serves as the interface to the AI capabilities, using:

- Groq LLM API with the llama-3.3-70b-versatile model
- Structured prompts to guide content generation
- JSON parsing for structured data extraction
- Error handling for robust operation

Key AI Functions

Scenario Generation:

```
def generate_scenario(self, security_domain, threat_type, industry,
role, experience_level)
```

1. Creates a narrative scenario tailored to the user's context, formatted as HTML for display.

Decision Point Generation:

```
def generate_decision_points(self, scenario_title, scenario_domain,
user_industry, user_role, experience_level)
```

2. Creates a series of decision points with multiple options, marking the correct option based on security best practices.

Decision Analysis:

```
def analyze_decision(self, scenario_description, user_decision,
is_correct)
```

3. Provides feedback on the user's decision, explaining why it was correct or incorrect from a security perspective.

Learning Moment Generation:

```
def generate_learning_moment(self, scenario_description,
security_domain)
```

4. Creates educational content explaining key security principles relevant to the scenario.

Knowledge Assessment:

```
def generate_knowledge_assessment(self, scenario_title,
scenario_domain, user_industry, user_role, experience_level,
num_questions)
```

5. Creates assessment questions to test the user's understanding of security concepts.

Prompt Engineering

The AI's behavior is guided by carefully crafted prompts that:

1. Define the agent's role and objectives
2. Specify the format and structure of generated content
3. Provide context about the user and scenario
4. Set parameters for difficulty and complexity
5. Ensure educational value and accuracy

Error Handling

The AI integration includes robust error handling:

- JSON parsing with exception handling
- Content validation to ensure proper structure
- Fallback mechanisms for failed generation attempts
- Logging for debugging and improvement

Certificate Generation

The certificate system provides tangible recognition of user achievements and reinforces learning accomplishments.

Certificate Design

Certificates are designed to be visually appealing and professional, with:

- High-resolution (2000x1400 pixels) for quality printing
- Decorative borders and styling elements
- Dynamic text placement based on content length
- Color scheme with green accents for visual appeal
- Clear hierarchy of information

Certificate Content

Each certificate includes:

- CyberSaga branding and title
- User's name prominently displayed
- Completed scenario title
- Overall score percentage
- Completion date
- Decorative elements and styling

Technical Implementation

The certificate generation process:

1. Creates a blank image using PIL
2. Adds decorative borders and styling elements
3. Places text elements with appropriate fonts and sizes
4. Handles long scenario titles by breaking into multiple lines
5. Converts the image to base64 for web display
6. Provides download functionality for saving as PNG

Font Handling

The system includes robust font handling:

- Attempts to use system fonts (Arial, DejaVuSans)
- Falls back to alternative fonts if primary choices unavailable
- Adjusts font sizes for readability and aesthetics
- Handles text measurement for proper placement

Certificate Display

The certificate is displayed in the Streamlit interface with:

- Responsive sizing for different screen dimensions
- Subtle border and shadow effects
- Download button for saving the certificate
- Navigation options to continue using the platform

User Profile Management

The user profile system manages personalization, progress tracking, and data persistence.

Profile Structure

Each user profile contains:

1. Basic Information:
    - User ID
    - Creation and update timestamps
    - Personal information (name, email, industry, role)
2. Progress Data:
    - Completed scenarios with performance metrics
    - Total points accumulated
    - Scenarios started and completed
    - Skill levels in different security domains
3. Preferences:
    - Difficulty preference (adaptive, beginner, etc.)
    - Focus areas for learning

Data Persistence

User profiles are stored as JSON files:

- Located in the "profiles" directory
- Named according to user ID
- Updated whenever profile changes occur
- Loaded on application startup

Progress Tracking

The system tracks user progress through:

1. Scenario Completion Records:
    - Scenario ID and title
    - Completion date
    - Points earned
    - Decision accuracy metrics
    - Assessment scores
2. Skill Level Tracking:
    - Domain-specific skill ratings
    - Updated based on performance
    - Visualized in the progress dashboard

Recommendation Engine

The profile system includes a recommendation engine that:

1. Identifies completed scenarios to avoid repetition
2. Analyzes past performance to identify weak areas
3. Prioritizes scenarios in domains with knowledge gaps
4. Provides personalized scenario recommendations

Code Documentation

app.py

The main application file orchestrating the entire platform:

```
"""
CyberSaga: An immersive cybersecurity education platform
Main Streamlit application file
"""
```

Key components:

1. Imports and Setup:
   - Imports necessary libraries and custom modules
   - Loads environment variables
   - Configures Streamlit page settings
2. Session State Initialization:
   - Initializes user profile
   - Creates security agent instance
   - Sets up scenario and progress tracking variables
3. Helper Functions:
   - `reset_scenario()`: Resets the current scenario state
   - `save_decision()`: Records user decisions for a scenario
   - `save_learning_moment()`: Stores learning moments for reference
   - `create_sample_scenarios()`: Generates demonstration scenarios
4. UI Components:
   - `load_css()`: Applies custom styling to the application
   - `show_welcome()`: Displays welcome page and onboarding form
   - `show_scenario_selection()`: Presents available scenarios
   - `show_scenario()`: Handles scenario execution and decision points
   - `show_scenario_summary()`: Displays completion summary and assessment
   - `show_progress_dashboard()`: Visualizes user progress and skills
5. Main Application Logic:
   - `main()`: Controls application flow based on current step
   - Conditional rendering of different application views

agent.py

The AI agent module handling content generation:

"""

```
AI Agent module for CyberSaga application.
This module contains the SecurityGuideAgent class that handles scenario generation
and user interaction.
"""
```

Key components:

1. SecurityGuideAgent Class:
   - Initializes with Groq LLM model
   - Maintains user profile for personalization
   - Provides methods for content generation
2. Content Generation Methods:
   - `generate_scenario()`: Creates narrative scenarios
   - `generate_decision_points()`: Creates decision challenges
   - `generate_decision_point()`: Creates individual decision points
   - `analyze_decision()`: Provides feedback on user choices
   - `generate_learning_moment()`: Creates educational content
   - `generate_knowledge_assessment()`: Creates assessment questions
3. Helper Methods:
   - `update_user_profile()`: Updates profile information
   - `_extract_json_from_response()`: Parses JSON from AI responses
   - `_clean_html()`: Sanitizes HTML content

scenarios.py

Defines the scenario system structure:

```
"""
Scenarios module for CyberSaga application.
This module contains classes for different types of cybersecurity scenarios.
"""
```

Key components:

1. Data Classes:
   - `DecisionPoint`: Represents a decision challenge
   - `LearningMoment`: Represents educational content
2. Base Scenario Class:
   - Common properties for all scenarios
   - Methods for adding decision points and learning moments
   - Completion tracking
3. Specialized Scenario Classes:
   - `PhishingScenario`: Email and message-based threats
   - `RansomwareScenario`: Ransomware attack situations
   - `SocialEngineeringScenario`: In-person manipulation
   - `DataProtectionScenario`: Data handling and privacy
   - `NetworkSecurityScenario`: Network-based threats
4. Factory Function:
   - `create_scenario()`: Creates appropriate scenario type

user_profile.py

Manages user data and progress:

```
"""
User profile management for CyberSaga application.
This module handles user profiles, progress tracking, and skill assessment.
"""
```

Key components:

1. UserProfile Class:
   - Initializes with default profile structure
   - Methods for loading and saving profile data
   - Functions for updating personal information
   - Progress tracking and scenario completion recording

2. Recommendation System:
    - `get_recommended_scenarios()`: Suggests scenarios based on past performance
    - Identifies weak areas and prioritizes relevant scenarios

certificate_generator.py

Handles certificate creation and display:

```
# Certificate generation module for CyberSaga
```

Key components:

1. Certificate Generation:
    - `generate_certificate()`: Creates certificate images
    - Font handling and fallback mechanisms
    - Dynamic text placement and styling
    - Base64 encoding for web display
2. Certificate Display:
    - `show_certificate_page()`: Renders certificate in the UI
    - Provides download functionality
    - Calculates scores for certificate display
    - Handles navigation options

prompts.py

Contains AI guidance prompts:

```
"""
Prompts for the CyberSaga Streamlit application.
These prompts are used to guide the AI agent in generating personalized
cybersecurity scenarios.
"""
```

Key components:

1. System Prompt:
   - Defines the AI agent's role and objectives
2. Content Generation Prompts:
   - `SCENARIO_GENERATION_PROMPT`: Creates narrative scenarios
   - `DECISION_POINTS_PROMPT`: Creates multiple decision points
   - `DECISION_POINT_PROMPT`: Creates individual decision points
   - `DECISION_ANALYSIS_PROMPT`: Analyzes user decisions
   - `LEARNING_MOMENT_PROMPT`: Creates educational content
   - `ASSESSMENT_PROMPT`: Creates general assessment questions
   - `KNOWLEDGE_ASSESSMENT_PROMPT`: Creates specific assessment questions

Deployment Guidelines

Local Development Setup

Clone the Repository:

```
git clone [repository-url]
cd CyberSaga_Streamlit
```

1.

Create Virtual Environment:

```
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate
```

2.

Install Dependencies:

```
pip install -r requirements.txt
```

3.
4. Environment Configuration:

Create a `.env` file with necessary API keys:

```
GROQ_API_KEY=your_api_key_here
```

-

Run the Application:

```
streamlit run app.py
```

5.

Production Deployment

For production deployment, consider:

1. Streamlit Cloud:
    - Connect your GitHub repository
    - Configure secrets for API keys
    - Set up requirements.txt for dependencies
2. Docker Deployment:
    - Create a Dockerfile for containerization
    - Include all dependencies and environment setup
    - Expose the appropriate port for the Streamlit server
3. Server Requirements:
    - Minimum 2GB RAM
    - 1GB disk space
    - Modern CPU with at least 2 cores
    - Stable internet connection for API calls
4. Security Considerations:
    - Secure API keys using environment variables
    - Implement user authentication for production use
    - Consider data encryption for stored profiles
    - Implement rate limiting for API calls

Future Enhancements

The CyberSaga platform has significant potential for expansion and improvement:

Content Enhancements

1. Expanded Scenario Library:
    - Additional security domains (IoT security, cloud security, etc.)
    - Industry-specific scenario collections
    - Role-based scenario paths
2. Interactive Elements:
    - Simulated phishing emails for identification practice
    - Network diagram interactions for security configuration
    - Code snippet analysis for secure coding practices
3. Media Integration:
    - Video demonstrations of security concepts
    - Audio narration for accessibility
    - Interactive infographics for complex topics

Technical Enhancements

1. Advanced AI Features:
    - Conversational agent for answering security questions
    - Adaptive difficulty based on user performance
    - Natural language processing for free-text responses
2. User Management:
    - Multi-user support with authentication
    - Team-based learning and competition
    - Administrator dashboard for monitoring progress
3. Analytics and Reporting:
    - Detailed performance analytics
    - Exportable progress reports
    - Comparative benchmarking against industry averages

4. Integration Capabilities:
    - LMS (Learning Management System) integration
    - SSO (Single Sign-On) support
    - API for extending functionality

Educational Enhancements

1. Learning Paths:
    - Structured curriculum with progressive difficulty
    - Certification tracks for different security domains
    - Prerequisite relationships between scenarios
2. Collaborative Learning:
    - Team-based scenario solving
    - Discussion forums for security topics
    - Peer review of security decisions
3. Gamification Elements:
    - Achievement badges for milestone completion
    - Leaderboards for friendly competition
    - Experience points and level progression

This documentation represents a comprehensive overview of the CyberSaga platform as of March 2025. The system continues to evolve with new features and improvements to enhance the cybersecurity education experience.

For questions, feedback, or contributions, please contact the development team.

CyberSaga: Where Security Education Becomes an Adventure