**Атакующий веб-сервиса (Red Team)**

**Команда: Агапитов, Иксанов**

Содержание

[1 Аналитика по реализации выбранного направления в России и мире 2](#_Toc187950584)

[2 Предназначение сервиса 3](#_Toc187950585)

[3 Функциональные возможности сервиса 4](#_Toc187950586)

[4 Архитектура сервиса с учетом интеграции с ИИ 5](#_Toc187950587)

[5 Тех. Стек 6](#_Toc187950588)

[6 Требования кибербезопасности сервиса 7](#_Toc187950589)

[7 Бэклог для получения MVP сервиса 8](#_Toc187950590)

[8 Дальнейшие перспективы развития проекта 13](#_Toc187950591)

[9 SAST-сканирование 14](#_Toc187950592)

[10 Git-secrets-сканирование 15](#_Toc187950593)

[11 SCA-сканирование 16](#_Toc187950594)

Ссылка на публичный репозиторий с описываемым проектом: <https://github.com/SeegmaEpsilon/KaliCashka>.

# 1 Аналитика по реализации выбранного направления в России и мире

**Мировой рынок**: в мире активно развиваются решения для кибербезопасности с использованием ИИ. Особенно важное место занимают платформы, которые используют методы машинного обучения и ИИ для анализа угроз и разработки стратегии защиты. Соревнования в формате CTF, такие как DEFCON и Pwn2Own, способствуют развитию технологий в области offensive security (взлом и защита веб-приложений).

**Рынок в России**: в России CTF-соревнования также популярны, организуются на различных уровнях — от университетских до крупных компаний (например, Positive Hack Days и OmCTF). Применение ИИ для анализа и автоматизации взлома веб-сервисов находится на ранней стадии, что предоставляет значительные возможности для создания инновационных решений.

# 2 Предназначение сервиса

Сервис предназначен для автоматизации анализа и взлома веб-приложений в рамках соревнований CTF с использованием искусственного интеллекта (ИИ) и методов машинного обучения. Основная цель сервиса — помощь атакующей команде (Red Team) в поиске уязвимостей веб-сервисов, таких как SQL-инъекции, XSS, CSRF и другие, с последующим предоставлением рекомендаций по их эксплуатации.

# 3 Функциональные возможности сервиса

**1. Анализ веб-приложений**: сервис автоматически сканирует веб-приложения на наличие уязвимостей, используя ИИ для выявления скрытых проблем.

***Что сделано*:** для следования Манифесту исследователя кибербезопасности (CTF-участника, «WHITE HAT») сервис вместо автоматического реализован автоматизированным, чтобы иметь больше контроля над поведением сервиса.

**2. Рекомендации по эксплуатации уязвимостей**: на основе анализа сервис предлагает возможные атаки и их последствия.

***Что сделано*:** Сервис на основе результатов выполненных команд предлагает возможные атаки и их последствия.

**3. Автоматизация атак**: сервис может запускать автоматические атаки на обнаруженные уязвимости, используя методы pentesting.

***Что сделано*:** для следования Манифесту исследователя кибербезопасности (CTF-участника, «WHITE HAT») сервис вместо автоматического реализован автоматизированным, чтобы иметь больше контроля над поведением сервиса.

**4. Отчеты об уязвимостях**: сервис предоставляет детализированные отчеты с описанием уязвимостей, степенью риска и предложениями по их устранению.

***Что сделано*:** формирование отчета запланировано в дальнейших обновлениях по платной подписке.

**5. Интеграция с ИИ**: использование ChatGPT 4o для анализа полученных данных и генерации инструкций по дальнейшему проведению атак.

***Что сделано*:** в качестве ИИ консультанта выбран GigaChat.

# 4 Архитектура сервиса с учетом интеграции с ИИ

**Frontend**: веб-интерфейс для взаимодействия пользователя с системой, разработанный на основе React.js.

***Что сделано*:** использован React.js.

**Backend**: микросервисная архитектура, построенная на основе Python (FastAPI или Flask, Django).

***Что сделано*:** использован FastAPI.

**ИИ-компонент**: ChatGPT 4o или GigaChat для анализа данных, обнаружения уязвимостей и предоставления инструкций по их эксплуатации.

***Что сделано*:** использован GigaChat.

**База данных**: использование PostgreSQL или MongoDB для хранения информации об анализируемых веб-приложениях и результатах атак.

***Что сделано*:** использован SQLAlchemy.

**Интеграция с инструментами для тестирования безопасности**: интеграция с инструментами, такими как OWASP ZAP, для улучшения функциональности анализа и атаки.

***Что сделано*:** интеграция с инструментами SAST и DAST сканирования осуществлена путем установки соответствующих пакетов на базе виртуальной машины Kali Linux.

# 5 Тех. Стек

**Frontend**: React.js для создания интерфейса.

***Что сделано*:** использован React.js.

**Backend**: Python (FastAPI или Flask, Django) для микросервисной архитектуры.

***Что сделано*:** использован FastAPI.

**ИИ**: использование модели ChatGPT 4o или GigaChat для анализа и рекомендаций.

***Что сделано*:** использован GigaChat.

**База данных**: PostgreSQL или MongoDB для хранения данных о веб-приложениях и уязвимостях.

***Что сделано*:** использован SQLAlchemy.

**Инструменты для тестирования безопасности**: OWASP ZAP, Metasploit для автоматизированного анализа уязвимостей.

***Что сделано*:** интеграция с инструментами SAST и DAST сканирования осуществлена путем установки соответствующих пакетов на базе виртуальной машины Kali Linux.

# 6 Требования кибербезопасности сервиса

**Аутентификация и авторизация**: встроенные механизмы контроля доступа (хэш) для обеспечения безопасного использования сервиса.

***Что сделано*:** все данные пользователей хэшируются bcrypt.

**Шифрование данных**: все данные должны быть зашифрованы (TLS, AES-256) как при передаче, так и при хранении.

***Что сделано*:** использован TLS.

**Логирование и аудит**: ведение журналов всех действий пользователей и операций ИИ для дальнейшего анализа.

***Что сделано*:** реализован журнал действий пользователя и ответов ИИ-консультанта.

# 7 Бэклог для получения MVP сервиса

1. **Реализация анализа веб-приложений**: базовая реализация сканирования веб-приложений с использованием ИИ для обнаружения уязвимостей.

***Что сделано*:** реализован сканирование веб-приложений с помощью виртуальной машины на базе Kali Linux и сервиса-помощника с ИИ.

1. **Генерация рекомендаций по атакам**: базовая функциональность по предоставлению рекомендаций для эксплуатации уязвимостей.

***Что сделано*:** Сервис на основе результатов выполненных команд предлагает возможные атаки и их последствия.

1. **Создание интерфейса для пользователей**: веб-интерфейс для загрузки данных о веб-приложениях, просмотра отчетов и рекомендаций.

***Что сделано*:** веб-интерфейс реализован с помощью React.js и чатом между ИИ-консультантом и полем ввода команд.

1. **Интеграция с ИИ**: интеграция ChatGPT 4o для анализа полученных данных и генерации инструкций.

***Что сделано*:** интеграция осуществлена с ИИ GigaChat.

1. **Отчеты о состоянии веб-приложений**: визуализация обнаруженных уязвимостей и генерация отчетов о них.

***Что сделано*:** визуализация обнаруженных уязвимостей осуществляется через чат общения с ИИ-консультантом, формирование отчета запланировано в дальнейших обновлениях по платной подписке.

1. **Интеграция с инструментами тестирования безопасности**: подключение к таким инструментам, как OWASP ZAP, для автоматизации анализа уязвимостей.

***Что сделано*:** интеграция с инструментами SAST и DAST сканирования осуществлена путем установки соответствующих пакетов на базе виртуальной машины Kali Linux.

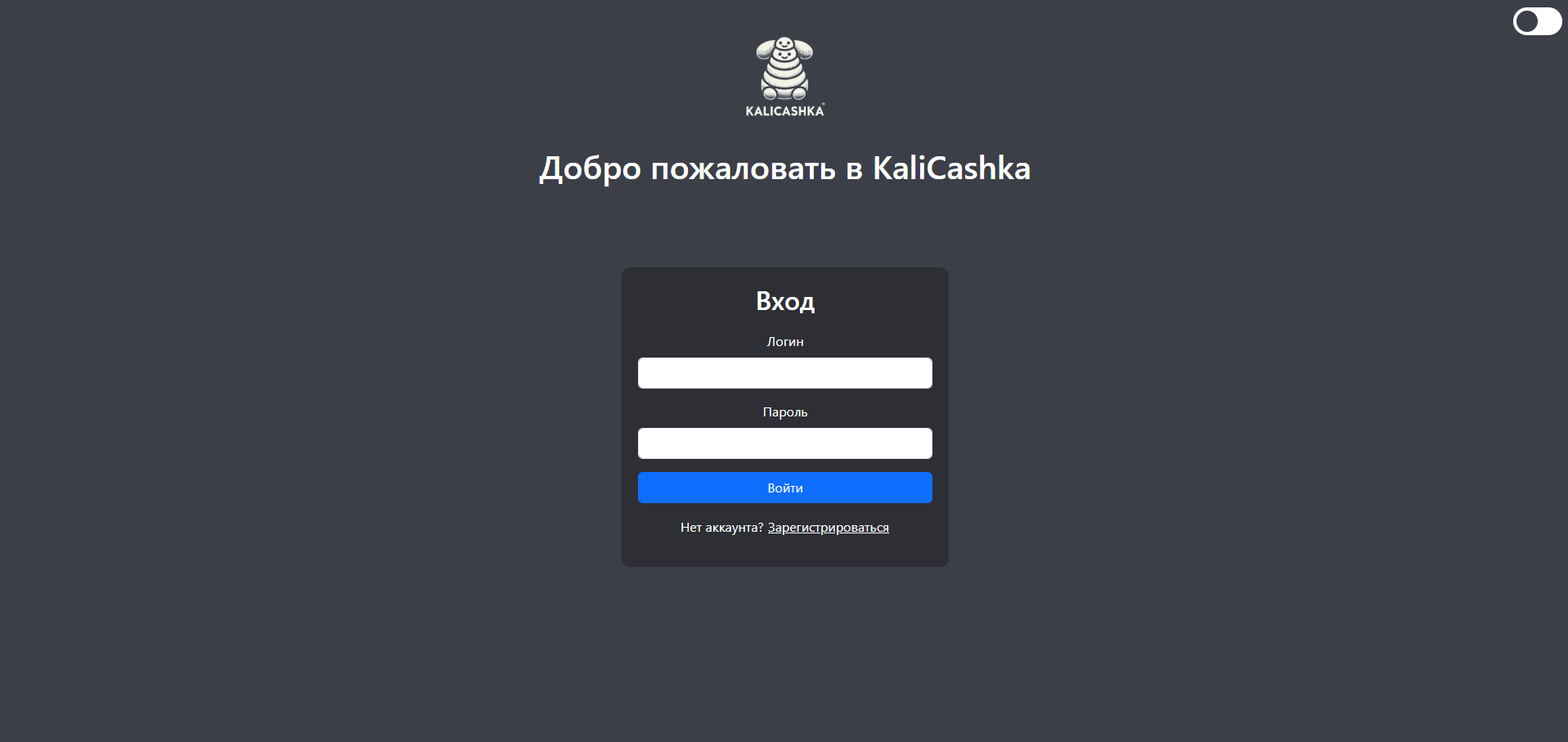


Рисунок 1 – Окно регистрации/авторизации пользователей

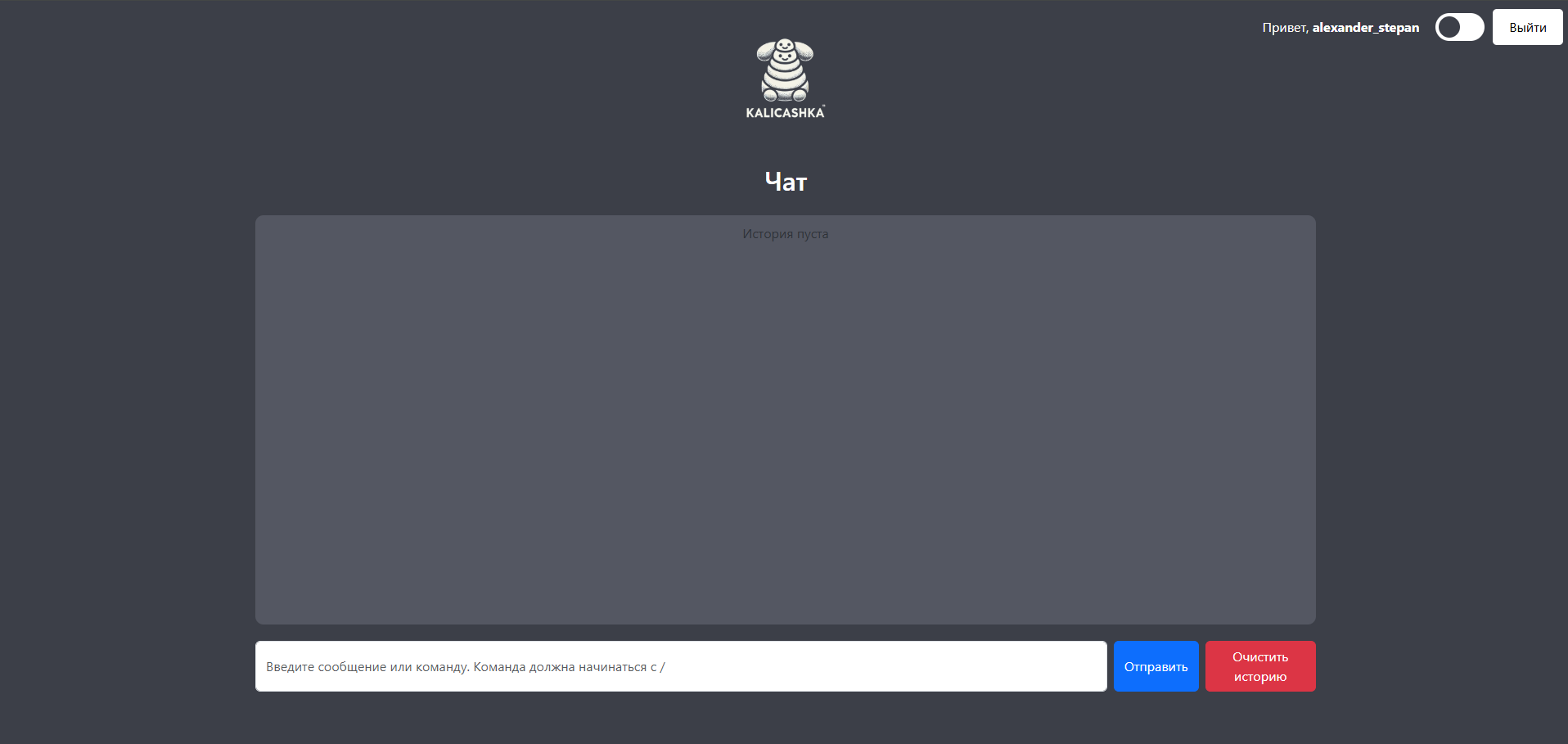


Рисунок 2 – Окно взаимодействия пользователя с сервисом

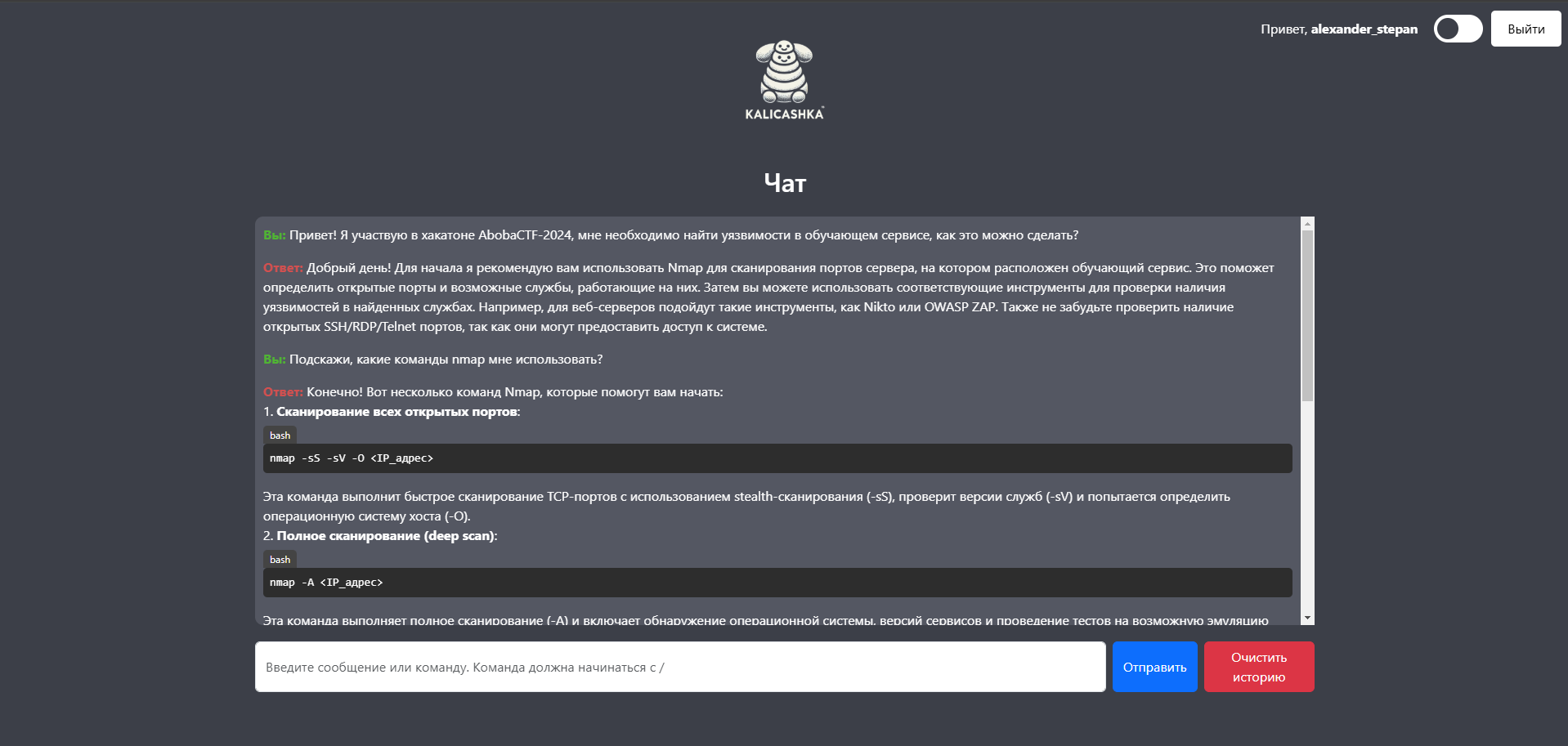


Рисунок 3 – Окно взаимодействия пользователя с сервисом в работе

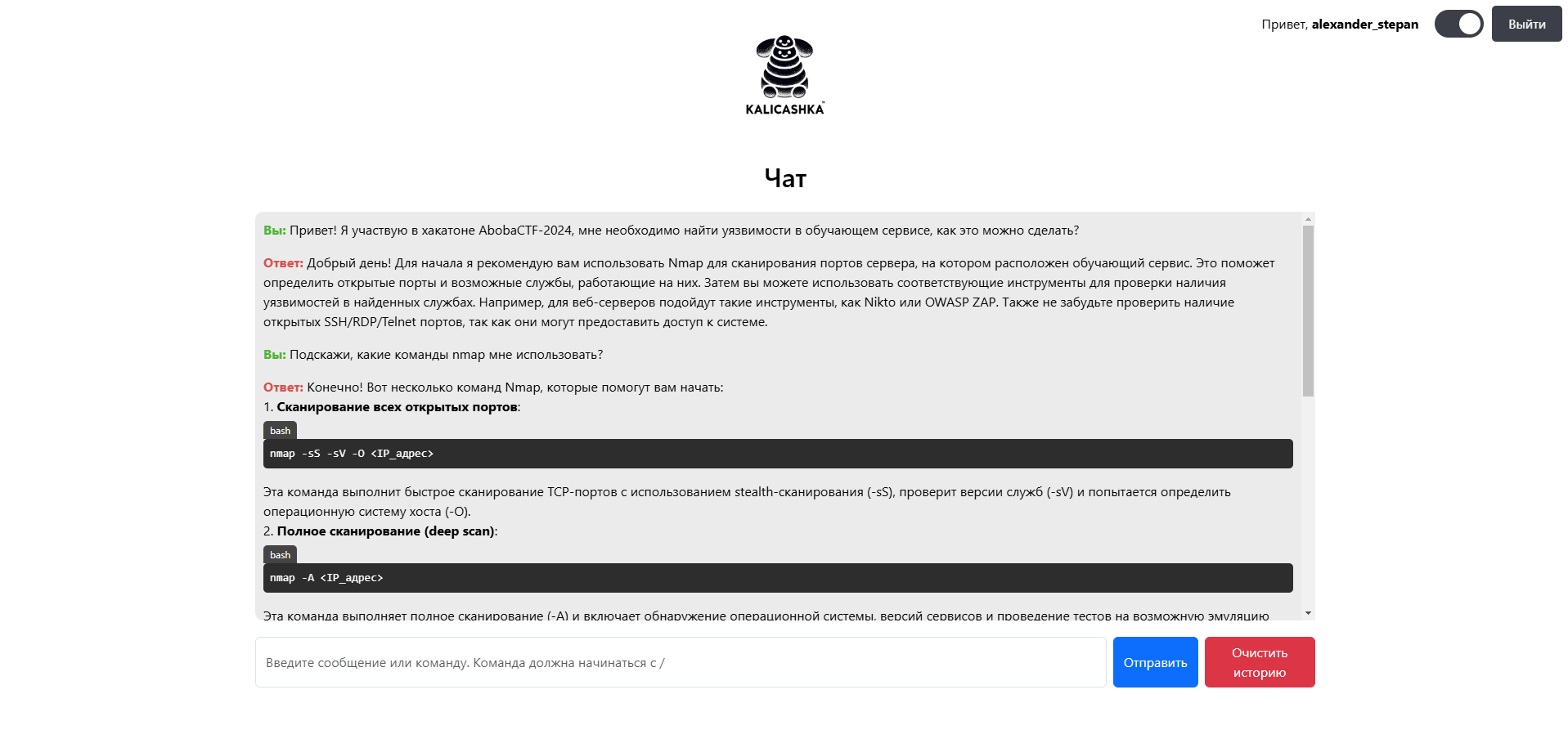


Рисунок 4 – Окно взаимодействия пользователя с сервисом в работе, белая тема

# 8 Дальнейшие перспективы развития проекта

Текущий проект может быть улучшен и расширен следующим функционалом:

* + - 1. применение полностью автоматической атаки на сервис вместо автоматизированной при соглашении пользователя с положениями об ответственности. Таким образом, в некотором приближении формируется отчет о результатах автоматического применения распространенных атак;
      2. переработка и улучшение UX/UI (например, в случае автоматизированной атаки, копирование предлагаемых моделью команд прямо в строку чата);
      3. предварительное сканирование атакуемого сервиса с целью построения некоторого графа/дерева предполагаемых атак в интерактивной графической форме;
      4. постоянный мониторинг наиболее популярных баз данных уязвимостей типа OWASP, CWE, CVE и CAPEC;
      5. возможное отображение команд, выполняемых в виртуальной машине Kali Linux, в сыром виде для дополнительных данных, не обработанных ИИ в случае сбоя.

# 9 SAST-сканирование

С помощью инструмента Semgrep был произведено SAST-сканирование исходного кода проекта. Результат сканирования представлен на рисунке 5.

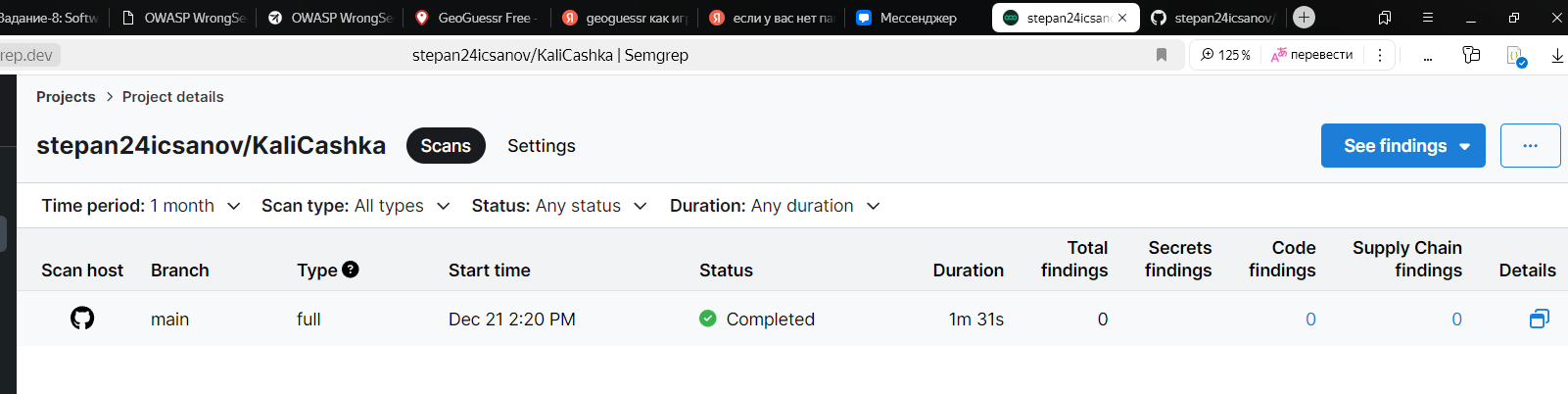


Рисунок 5 – Результат SAST-сканирования

Из рисунка 5 видно, что в результате SAST-сканирования не было найдено уязвимостей в проекте.

# 10 Git-secrets-сканирование

С помощью инструмента git-secrets был произведен поиск секретов в исходном коде проекта. Результат сканирования представлен на рисунке 6.

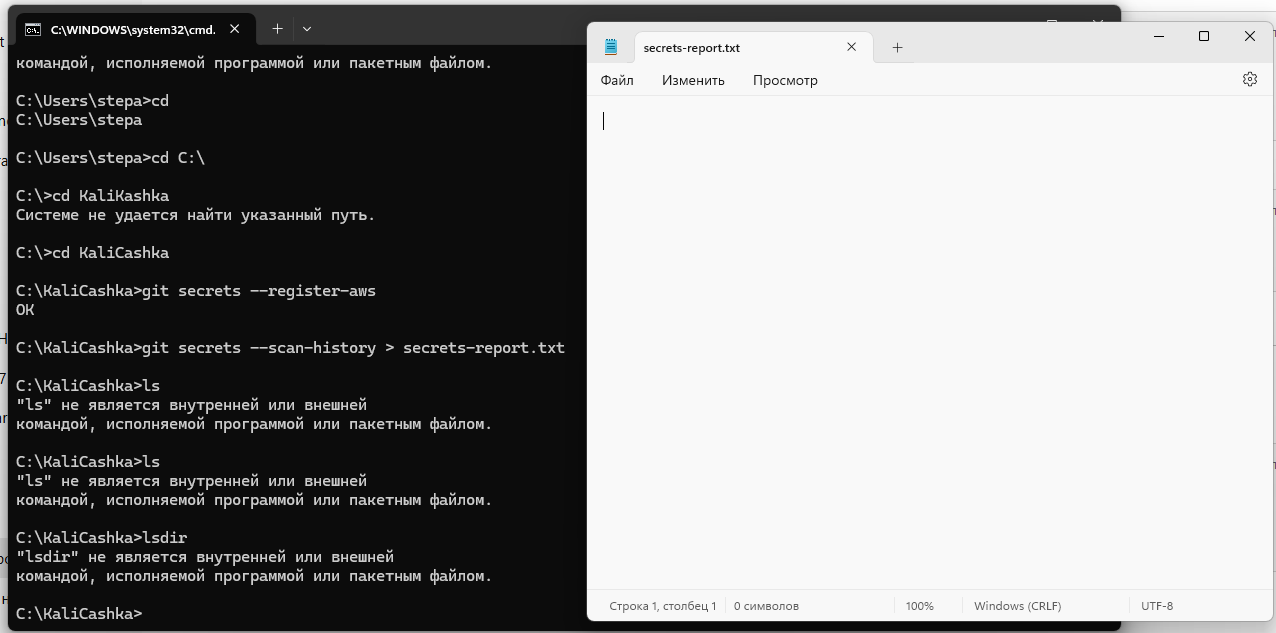


Рисунок 6 – Результат поиска секретов с помощью инструмента

git-secrets

Из рисунка 6 видно, что секретов в исходном коде проекта не обнаружено.

# 11 SCA-сканирование

С помощью инструмента trivy было проведено SCA-сканирование. В результате сканирования были найдены следующие уязвимости зависимостей:

**1. Backend (Python)**

PyJWT

Уязвимость: CVE-2024-53861

Описание: Некорректное сравнение строк в поле iss, что может позволить частичное совпадение.

Текущая версия: 2.10.0

Рекомендуемая версия: 2.10.1

Серьезность: Низкая.

Решение: Обновите PyJWT до версии 2.10.1.

Python-Jose

Уязвимость 1: CVE-2024-33663

Описание: Уязвимость алгоритмической путаницы с ключами OpenSSH ECDSA.

Текущая версия: 3.3.0

Серьезность: Критическая.

Решение: Обновите библиотеку или замените её.

Уязвимость 2: CVE-2024-33664

Описание: Возможность DoS-атаки через специально созданные JWE токены (JWT-бомба).

Серьезность: Средняя.

Решение: Обновите библиотеку до последней версии или ограничьте обработку подозрительных данных.

Python-Multipart

Уязвимость: CVE-2024-53981

Описание: Возможность DoS через неправильную обработку данных multipart/form-data.

Текущая версия: 0.0.17

Рекомендуемая версия: 0.0.18

Серьезность: Высокая.

Решение: Обновите Python-Multipart до версии 0.0.18.

**2. Frontend (Node.js)**

Nanoid

Уязвимость: CVE-2024-55565

Описание: Некорректная обработка нецелых значений.

Текущая версия: 3.3.7

Рекомендуемая версия: 3.3.8 или 5.0.9

Серьезность: Средняя.

Решение: Обновите Nanoid.

Nth-Check

Уязвимость: CVE-2021-3803

Описание: Уязвимость сложной регулярной экспрессии (ReDoS).

Текущая версия: 1.0.2

Рекомендуемая версия: 2.0.1

Серьезность: Высокая.

Решение: Обновите Nth-Check до версии 2.0.1.

Path-To-RegExp

Уязвимость: CVE-2024-52798

Описание: Уязвимость регулярных выражений (ReDoS).

Текущая версия: 0.1.10

Рекомендуемая версия: 0.1.12

Серьезность: Средняя.

Решение: Обновите Path-To-RegExp.

PostCSS

Уязвимость: CVE-2023-44270

Описание: Неправильная валидация входных данных.

Текущая версия: 7.0.39

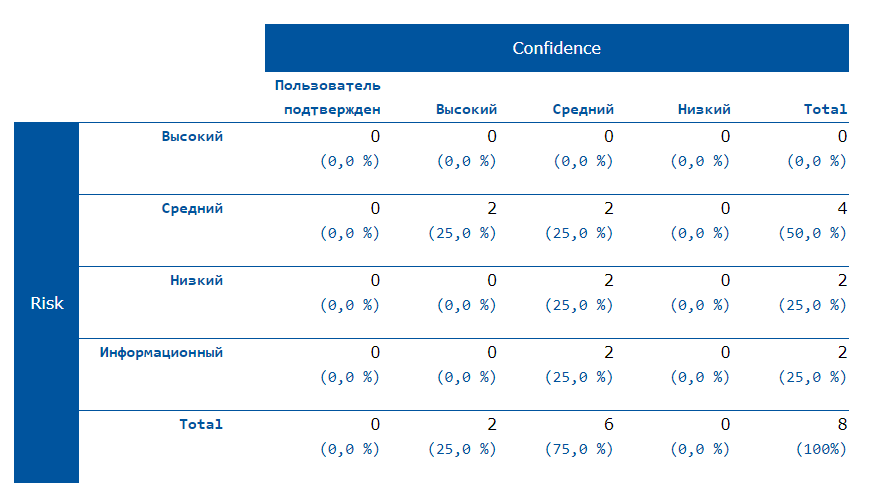
Рекомендуемая версия: 8.4.31

Серьезность: Средняя.

Решение: Обновите PostCSS.

# 12 DAST-сканирование

С помощью утилиты OWASP ZAP было произведено DAST-сканирование разработанного веб-сервиса. Результат сканирования представлен на рисунке 7.



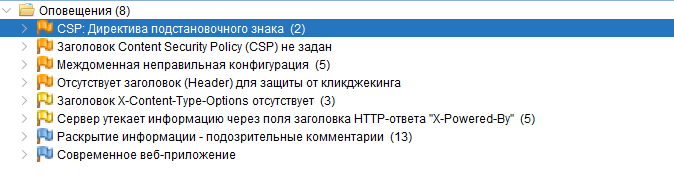


Рисунок 7 – Результат DAST-сканирования

В результате DAST-сканирования с использованием OWASP ZAP были выявлены следующие оповещения безопасности.

Директива подстановочного знака в Content Security Policy (CSP) (2 случая, риск средний) может привести к тому, что злоумышленник сможет использовать уязвимые директивы CSP для выполнения атак, таких как межсайтовый скриптинг (XSS). Рекомендуется заменить директивы подстановочного знака на более специфичные источники.

Отсутствие заголовка Content Security Policy (1 случай, риск средний) делает приложение уязвимым к атакам XSS и внедрению кода. Для усиления защиты необходимо настроить корректный заголовок CSP, ограничивающий выполнение скриптов из ненадежных источников.

Междоменная неправильная конфигурация (5 случаев, риск средний) указывает на проблемы в настройке политики доступа к ресурсам между разными доменами. Это может быть использовано для кражи данных или выполнения атак. Требуется пересмотреть настройки CORS (Cross-Origin Resource Sharing).

Отсутствие заголовка для защиты от кликджекинга (1 случай, риск средний) делает приложение уязвимым к атакам кликджекинга. Добавление заголовка X-Frame-Options или его эквивалента защитит пользователей от таких атак.

Отсутствие заголовка X-Content-Type-Options (3 случая, риск низкий) может позволить браузеру интерпретировать файлы неверного типа, что потенциально приводит к выполнению атак. Рекомендуется добавить заголовок X-Content-Type-Options: nosniff.

Утечка информации через заголовок HTTP-ответа "X-Powered-By" (5 случаев, риск низкий) раскрывает используемые серверные технологии, что облегчает поиск уязвимостей. Необходимо удалить или замаскировать этот заголовок.

Раскрытие информации через подозрительные комментарии (13 случаев, информационный) указывает на комментарии в коде, которые могут содержать чувствительные данные или отладочную информацию. Рекомендуется провести ревизию комментариев и удалить избыточную информацию.

Современное веб-приложение (1 случай, информационный) указывает на использование современных технологий, что само по себе не является угрозой, но может потребовать дополнительной проверки на соответствие лучшим практикам безопасности.