**Атакующий веб-сервиса (red team)**

**Команда: Агапитов, Иксанов**

Содержание

[1 Аналитика по реализации выбранного направления в России и мире 2](#_Toc185517326)

[2 Предназначение сервиса 2](#_Toc185517327)

[3 Функциональные возможности сервиса 3](#_Toc185517328)

[4 Архитектура сервиса с учетом интеграции с ИИ 4](#_Toc185517329)

[5 Тех. Стек 4](#_Toc185517330)

[6 Требования кибербезопасности сервиса 5](#_Toc185517331)

[7 Бэклог для получения MVP сервиса 5](#_Toc185517332)

# 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*:** запланировано в дальнейших обновлениях.

# 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*:** запланировано в дальнейших обновлениях.

# 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*:** запланировано в дальнейших обновлениях.

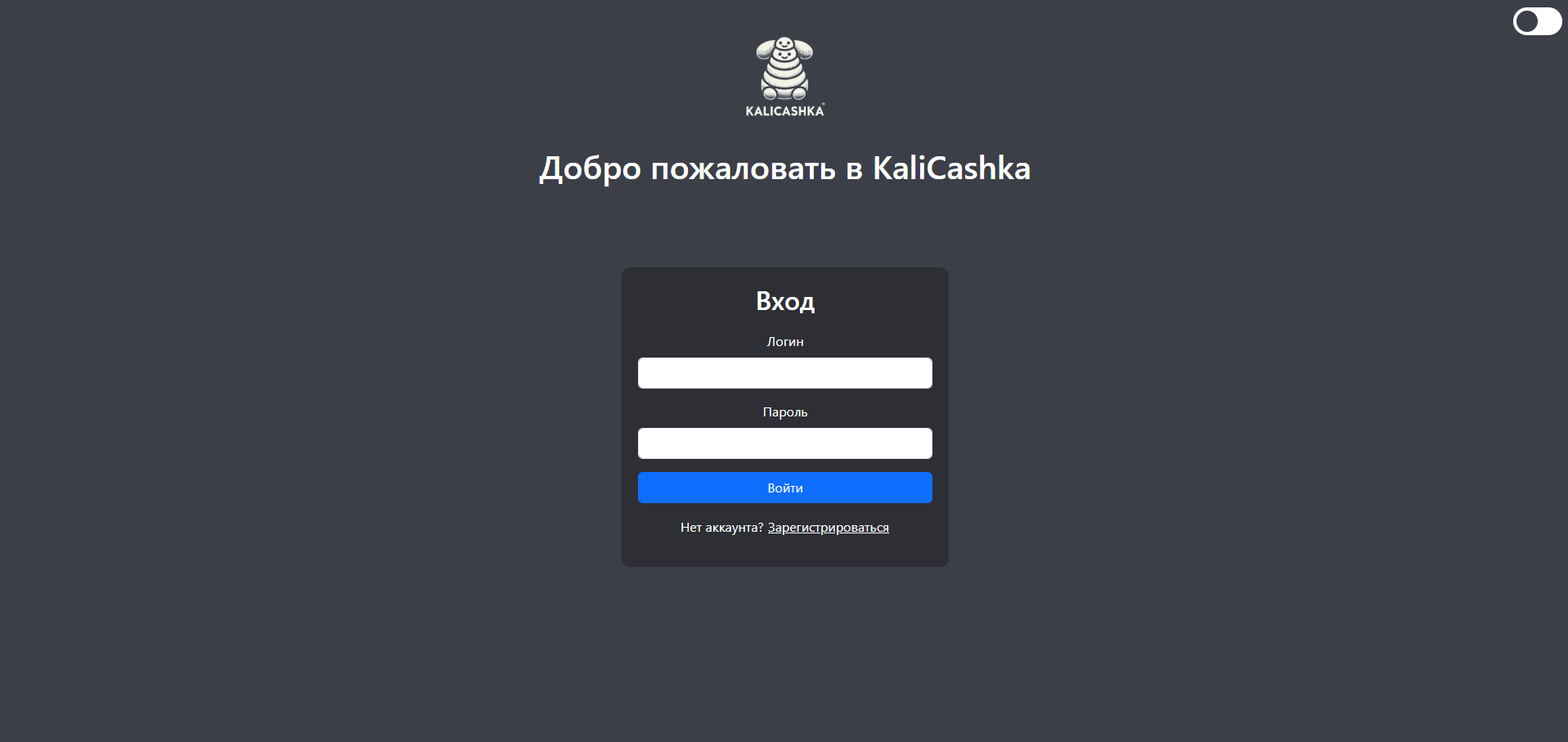


Рисунок 1 – Окно регистрации/авторизации пользователей

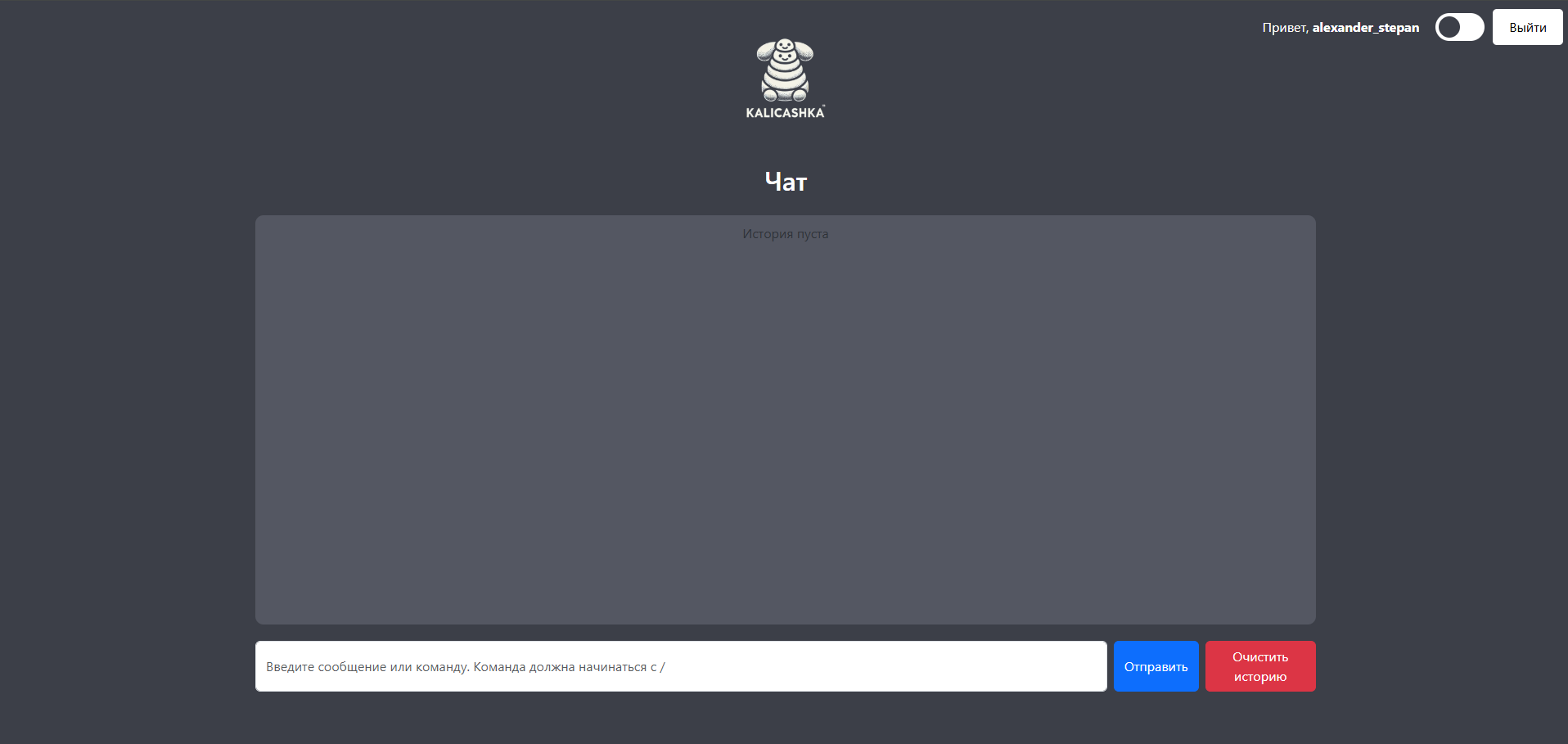


Рисунок 2 – Окно взаимодействия пользователя с сервисом

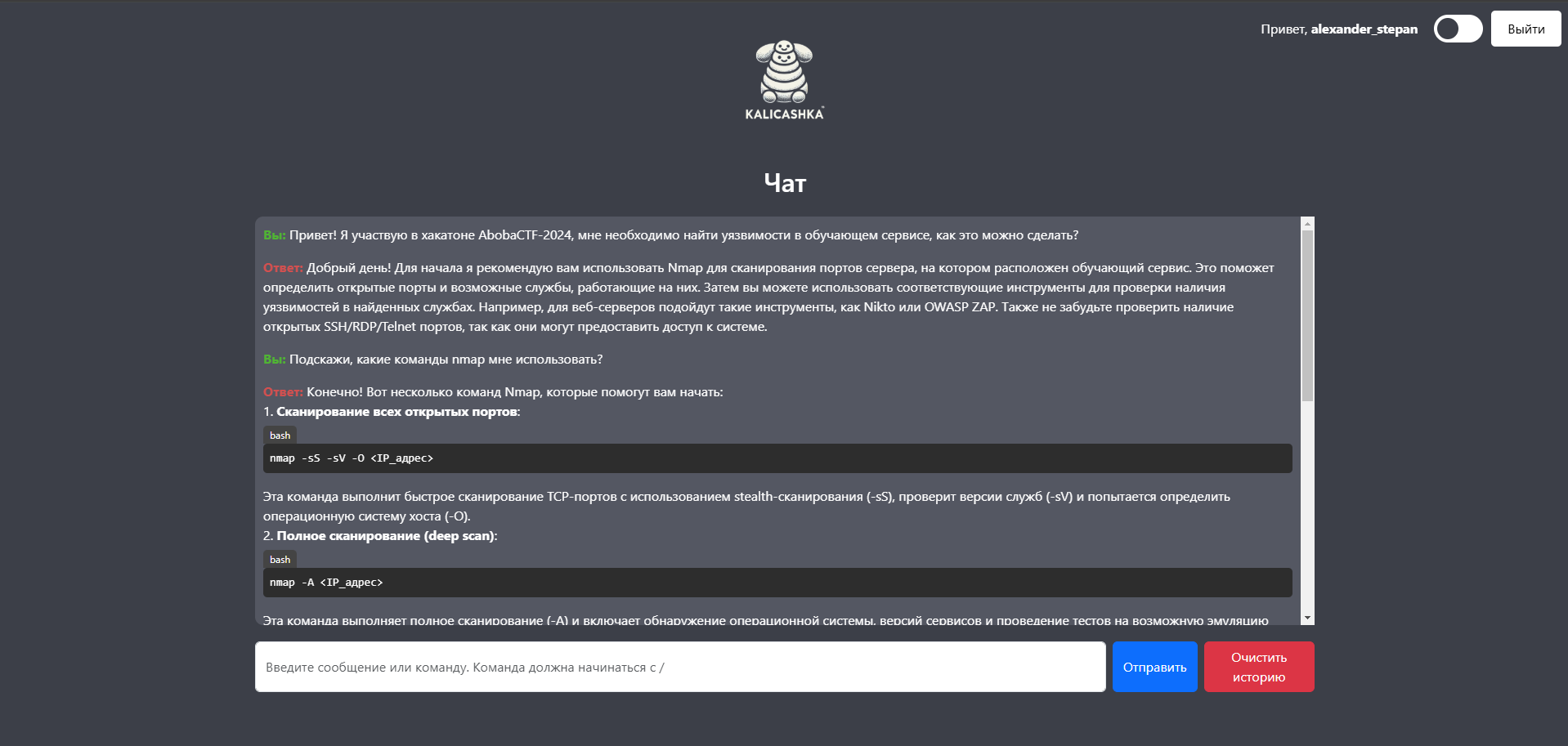


Рисунок 3 – Окно взаимодействия пользователя с сервисом в работе

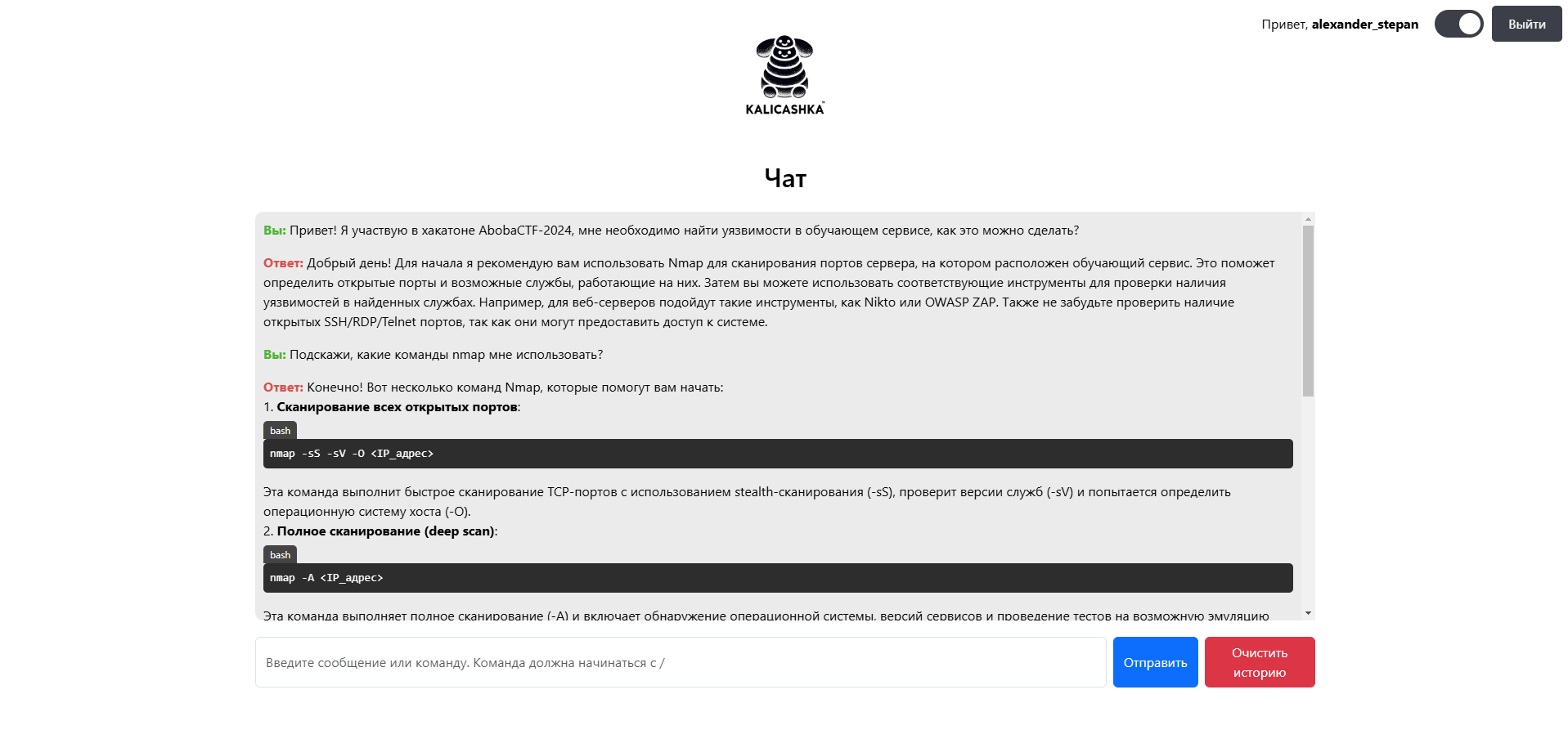


Рисунок 4 – Окно взаимодействия пользователя с сервисом в работе, белая тема