Инженерный вестник Дона, №2 (2026)
ivdon.ru/ru/magazine/archive/n2y2026/10720

Разработка прототипа системы управления техническими средствами физической защиты объекта на базе операционной системы Astra Linux

В.Р. Скрипова

Уральский федеральный университет

Аннотация: Проектирование автоматизированных систем управления физической защитой объектов является одним из востребованных направлений в сфере разработки отечественных программных продуктов. В статье представлены архитектура программно-аппаратной системы, оценка средств разработки, необходимых для реализации веб-приложения на базе операционной системы Astra Linux, и описание эксперимента по созданию прототипа системы. Для построения системы использовались следующие инструменты: фреймворк Angular – для клиентского слоя, фреймворк FastAPI, библиотека SQLAlchemy, протокол WebSocket – для серверного слоя, объектно-реляционная система управления базами данных PostgreSQL – для организации хранения данных. Результатом работы стала система управления техническими средствами, демонстрирующая взаимодействие с устройствами и базой данных. Реализованный прототип послужит основой для разработки программно-аппаратного комплекса физической защиты объекта.
Ключевые слова: отечественная операционная система, веб-приложение, средства разработки, система управления, база данных, датчик, мониторинг.

Введение

Востребованность исследований в области разработки программных продуктов для комплексов физической защиты объектов обусловлено следующими факторами. Во-первых, устаревание программного обеспечения (ПО), задействованного в ранее реализованных системах физической защиты. Системы требуют пересмотра, реструктуризации и оптимизации. Во-вторых, согласно Указу Президента РФ №166 от 30.03.2022 «О мерах по обеспечению технологической независимости и безопасности критической информационной инфраструктуры Российской Федерации», необходимо осуществить переход на импортозамещающие программные продукты. В-третьих, наличие требования обязательной сертификации программного обеспечения в органах Федеральной службы по техническому и экспортному контролю (ФСТЭК), выполнение которого обеспечит защиту информации. Главными составляющими комплекса защиты объекта являются совокупность технических средств и автоматизированная распределенная

информационная система управления. Одной из главных функциональных задач системы является организация процесса мониторинга технических средств (отслеживание текущего состояния), который включает в себя три основные стадии: опрос устройств (сбор данных), обработка полученных данных, передача данных в пользовательский интерфейс в режиме реального времени [1].

В данной статье рассматриваются веб-технологии для реализации процессов сбора, обработки и передачи данных. Цель исследования – выявить наиболее эффективные инструменты для разработки программного продукта и создать прототип системы управления техническими средствами. Главными критериями оценки выступают требования к импортозамещению и сертификации программного обеспечения, а также перспектива выполнения требований к масштабируемости, производительности, отказоустойчивости и быстродействию системы. Основными методами исследования являются оценка и эксперимент. Соответственно, можно выделить ряд задач исследования: сделать краткий обзор и оценку эффективности средств разработки веб-приложения, описать эксперимент по созданию прототипа системы управления, подвести итоги результатов исследования.

Оценка средств разработки системы

Автоматизированные системы управления физической защитой объекта, функционирующие на операционной системе (ОС) Windows, потеряли свою актуальность. В условиях импортозамещения, перед разработчиками ставится задача о создании новых программных продуктов – систем управления, реализованных на базе отечественной операционной системы. Средствами разработки выступают программные инструменты, поддерживаемые отечественной операционной системой. В дополнение к основной задаче, согласно установленным государством требованиям на применение информационных технологий в государственных

информационных системах и системах обработки персональных данных, для разработчиков систем вводится условие обязательной сертификации программного обеспечения. Сертификация программного обеспечения для защиты информации в России проводится ФСТЭК России. В ходе сертификации оцениваются как технические, так и организационные меры защиты информации, выявляются уязвимости и проверяется соответствие установленным нормам и стандартам безопасности Российской Федерации.

Учитывая основные требования, в качестве основной операционной системы была выбрана отечественная Astra Linux Special Edition 1.8 (Astra Linux SE). Благодаря наличию сертификатов Министерства Обороны, Федеральной службы безопасности и ФСТЭК России, Astra Linux SE позволяет работать с информацией вплоть до уровня государственной тайны.

По мере развития области разработок программных продуктов, клиент-серверная архитектура системы, объектно-ориентированный подход в программировании, многопоточные решения и надежные системы управления базами данных (СУБД) стали стандартами для создания масштабируемых и гибких приложений. Для соответствия системы современным тенденциям был осуществлен подбор инструментов разработки: языка программирования, фреймворков для создания клиентской и серверной части приложения, реляционной СУБД и веб-сервера для передачи данных на базе протокола WebSocket.

Python – это универсальный язык программирования, который становится все более популярным в научных исследованиях [2]. Поскольку язык программирования Python установлен по умолчанию в ОС Astra Linux SE, поддерживает объектно-ориентированное программирование, а также имеет множество библиотек и фреймворков, которые упрощают разработку и увеличивают производительность, то, несомненно, Python занял место лидирующего языка программирования в проектировании системы.

В реализации пользовательского интерфейса системы было принято решение использовать Angular. Angular – это популярный JavaScript фреймворк для создания одностраничных приложений. В его основе лежит принцип разделения представления, модели и логики. Одностраничные приложения – это веб-приложения, основанные на идее использования одной веб-страницы как каркаса, в то время как ее содержимое динамически изменяется средствами JavaScript. Такие приложения имеют преимущества в скорости загрузки, в первую очередь потому, что данные не подгружаются каждый раз при совершении каких-либо действий [3]. Angular позволяет в сжатые сороки создавать программный код, тестировать отдельные части разработанных приложений, проводить быструю компиляцию, имеется поддержка различных браузеров, HTTP (HyperText Transfer Protocol), WebSockets [4].

Для разработки серверного слоя был выбран фреймворк FastAPI. FastAPI – это современный веб-фреймворк для Python, который благодаря своей скорости и удобству использования становится всё более популярным среди разработчиков. Быстрая обработка запросов, поддержка асинхронного программирования и интеграция с различными библиотеками делают его оптимальным выбором для построения высоконагруженных и масштабируемых сервисов [5]. Универсальные возможности фреймворка в полной мере проявляются при реализации модульной функциональности. Благодаря способности к быстрой разработке интерфейса программирования приложения (Application Programming Interface – API), FastAPI позволяет разработчикам создавать независимые модули, которые могут быть легко интегрированы в общую алгоритмическую структуру. Эта интеграция происходит через стандартизированные интерфейсы, гарантируя, что каждый модуль следует предопределённому набору шаблонов ввода-вывода. Как следствие, когда для конкретного модуля требуются обновления или

оптимизация, ядро алгоритма остаётся незатронутым, что способствует эффективному сопровождению и расширяемости [6]. Для организации работы с базой данных (БД) FastAPI был дополнен библиотекой SQLAlchemy. Библиотека, написанная на языке программирования Python, предназначена для работы с системами управления реляционными базами данных MySQL, PostgreSQL, SQLite, Oracle и т.д. [7]. SQLAlchemy включает в себя технологию объектно-реляционного отображения (Object-Relational Mapping – ORM), которая связывает объектно-ориентированные модели с реляционными базами данных, позволяя разработчикам абстрагироваться от прямого написания SQL-запросов (Structured Query Language – SQL) [8].

При выборе СУБД предпочтение было отдано PostgreSQL. PostgreSQL является самой продвинутой СУБД с открытым исходным кодом, предлагающей мощные функции для обеспечения целостности данных и надёжности хранения данных [9]. PostgreSQL – это объектно-реляционная система управления базами данных, поддерживающая транзакции, репликацию, триггеры и хранимые процедуры. Поэтому PostgreSQL оптимальна для использования в сложных приложениях, где предъявляются высокие требования к расширяемости, производительности, надежности и гибкости. Само программное обеспечение PostgreSQL входит в состав дистрибутива ОС Astra Linux SE, а также является сертифицированным ФСТЭК средством защиты информации.

В вопросе подбора веб-сервера наиболее оптимальным вариантом оказался веб-сервер Apache HTTP Server (Apache). Веб-сервер входит в состав основного репозитория ОС Astra Linux SE, является сертифицированным средством, поддерживает работу протокола WebSocket, а также обеспечивает надежность, гибкость и высокую производительность.

Эксперимент по реализации прототипа системы управления

При проведении данного этапа исследования были использованы: сервер с установленной ОС Astra Linux Special Edition 1.8, инсталлированные на сервер пакеты средств разработки, испытательный стенд, включающий техническое средство охраны и устройство идентификации. Основной целью опытно-экспериментального этапа исследования является создание на базе выбранных инструментов прототипа системы, демонстрирующего взаимодействие с устройствами физической защиты и работу с базой данных.

Для достижения поставленной цели был сформулирован ряд задач:

1. Спроектировать программно-аппаратную архитектуру системы.
2. Создать пользовательский интерфейс системы.
3. Разработать сервис по работе с базой данных.
4. Реализовать сервисы для мониторинга устройств.
5. Обеспечить обработку событий устройства идентификации.

В первую очередь была спроектирована программно-аппаратная архитектура, отражающая поток данных между элементами системы (рис. 1).

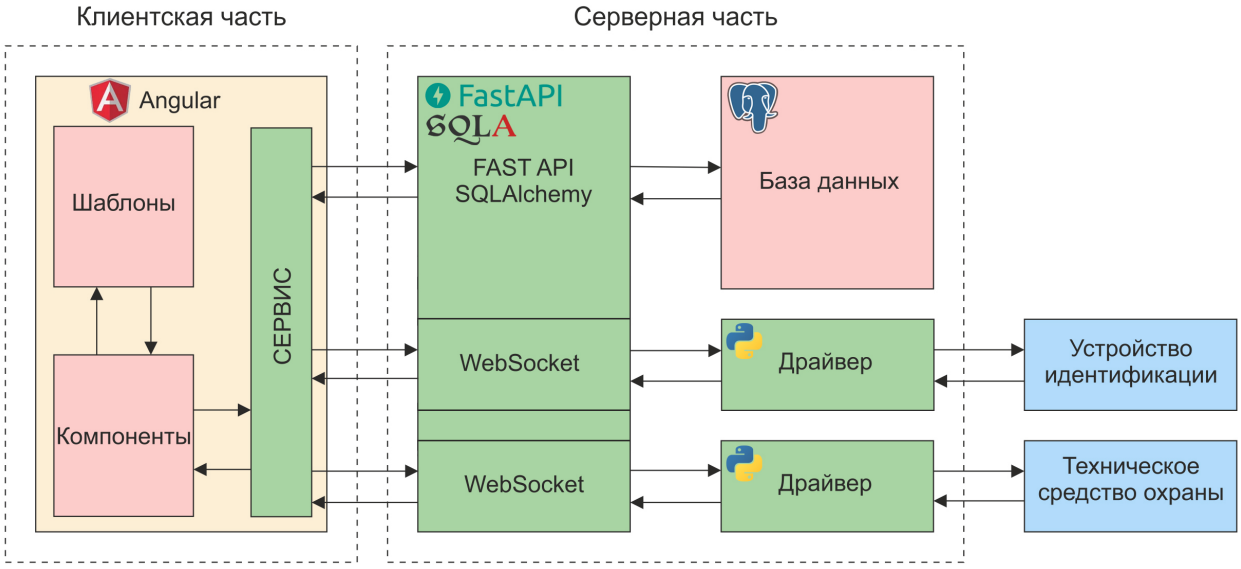

Рис. 1. – Программно-аппаратная архитектура системы. Иллюстрация автора.

Клиентская часть. Фреймворк Angular использовался для создания пользовательского интерфейса. В процессе разработки приложения было создано семь компонентов, включающих методы работы с входными и выходными данными. Каждый компонент имеет собственный шаблон, который отвечает за отображение данных и реализацию необходимых функций в пользовательском интерфейсе. В проекте были созданы компоненты следующих разделов веб-приложения: «БД абонентов», «Регистрация абонента», «Детальный просмотр», «Редактирование информации абонента», «Биометрия», «Мониторинг ГЕОР», «Мониторинг ТП-4М». С целью организации взаимодействия с серверной частью был реализован сервис. В разработанном сервисе использовался класс *HttpClient*, который предоставляет ряд методов для отправки HTTP-запросов типов: получение (GET), отправление (POST), размещение (PUT), удаление (DELETE). В целях поддержки функционирования протокола WebSocket в сервис была подключена библиотека RxJS – инструмент для работы с асинхронными потоками данных. Все ответы от серверной части поступали в сервис клиентской части. Полученные данные распределялись по компонентам для обработки и передачи в шаблоны.

Перед запуском разработанного приложения клиентской части на сервере необходимо было осуществить компиляцию проекта с помощью команды *ng build,* которая собирает исходный код приложения в набор файлов для развертывания. Полученный набор файлов был скопирован в папку для хранения файлов клиентской части на сервере. Автоматический запуск веб-сервера Apache в ОС Astra Linux SE был настроен с помощью команды: *sudo systemctl enable apache2*. При старте веб-сервера приложение становилось активным и доступным для компьютеров в общей сети. Данная возможность была реализована через настройку файла конфигурации веб-сервера. Для доступа к разработанному пользовательскому интерфейсу

достаточно в адресной строке браузера ввести статический IP-адрес (Internet Protocol – IP) сервера.

Серверная часть. С помощью графического клиента pgAdmin, подключенного к локально установленному серверу базы данных PostgreSQL, была создана база данных. В каждой таблице базы данных был определен набор полей и типы хранимых данных. Между таблицами были установлены связи. Исполнение команды *systemctl enable postgresql* в ходе установки СУБД обеспечила автоматический запуск сервера PostgreSQL в ОС Astra Linux SE.

В целях обеспечения взаимодействия с базой данных был реализован сервис на базе фреймфорка FastAPI и библиотеки SQLAlchemy. Прежде всего необходимо было настроить подключение к базе данных. Для решения данной задачи был разработан файл *database.py*, в котором были прописаны настройки подключения, механизм формирования сессий и базовый класс моделей. На следующем шаге потребовалось разработать файл *models.py*, содержащий модели данных, которые описывали структуру базы данных посредством Python-классов. В главном файле сервиса *main.py* было создано API для работы с объявленной базой данных, где с помощью методов запросов SQLAlchemy были реализованы операции добавления, чтения, обновления и удаления данных. В процессе обработки поступающих HTTP-запросов, разработанный сервис генерировал ответы в текстовом формате обмена данными JSON (JavaScript Object Notation – JSON).

Процесс для мониторинга устройств был основан на реализации протокола WebSocket и использовании пакета asyncio, предназначенного для организации асинхронного программирования с использованием цикла событий сопрограмм. Концепция асинхронного программирования подразумевает явное установление особых точек при выполнении вычислений (процессов). Эти точки позволяют отдельным вычислениям

избегать ожидания завершения всех остальных вычислений, как это происходит в случае последовательного программирования [10]. Все действия цикла опроса устройств и обработки событий были прописаны внутри асинхронной функции, объявленной с помощью ключевого слова *async*. Сообщения о текущем состоянии технических средств физической защиты, передавались из серверной части в сервис клиентской части, распределялись по компонентам для публикации в заданных областях шаблонов: «Охранное табло», в области которого с помощью графических элементов интерфейса была реализована визуализация состояния устройства, «Журнал событий» и «Проходы через кабину».

Обработка событий устройства идентификации выполнялась следующим образом: событие об успешном опознании абонента устройством содержало идентификационный признак, наличие которого отслеживалось при обработке данных в сервисе клиентской части, что, в свою очередь, запускало функцию отправки HTTP-запроса на поиск в базе данных персональной информации абонента, обладающим соответствующим идентификационным признаком. Зафиксированное устройством время поступления события с идентификационным признаком и полученные персональные данные публиковались в области «Проходы через кабину».

Инициализация работы сервисной части осуществлялась путем использования веб-сервера для Python – uvicorn. При запуске через uvicorn файл сервиса *main.py* работает как приложение, которое постоянно находится в оперативной памяти сервера и обслуживает запросы. Команда для запуска: *uvicorn main:app --host IP-адрес сервера --port 8000*.

Результаты исследования

В ходе исследования был сформирован оптимальный набор средств разработки веб-приложения на базе ОС Astra Linux Special Edition 1.8, отвечающих требованиям к импортозамещению и сертификации

программного обеспечения, масштабируемости, производительности, отказоустойчивости и быстродействию.

Во второй части исследования была проведена опытно-экспериментальная работа по созданию прототипа системы управления техническими средствами физической защиты. Реализованная система демонстрирует взаимодействие с компонентами программно-аппаратного комплекса в пользовательском интерфейсе приложения. На рис. 2 представлен вывод данных абонентов из базы данных. Все персональные данные абонентов в эксперименте – вымышленные, и используются только для тестирования работы прототипа системы.

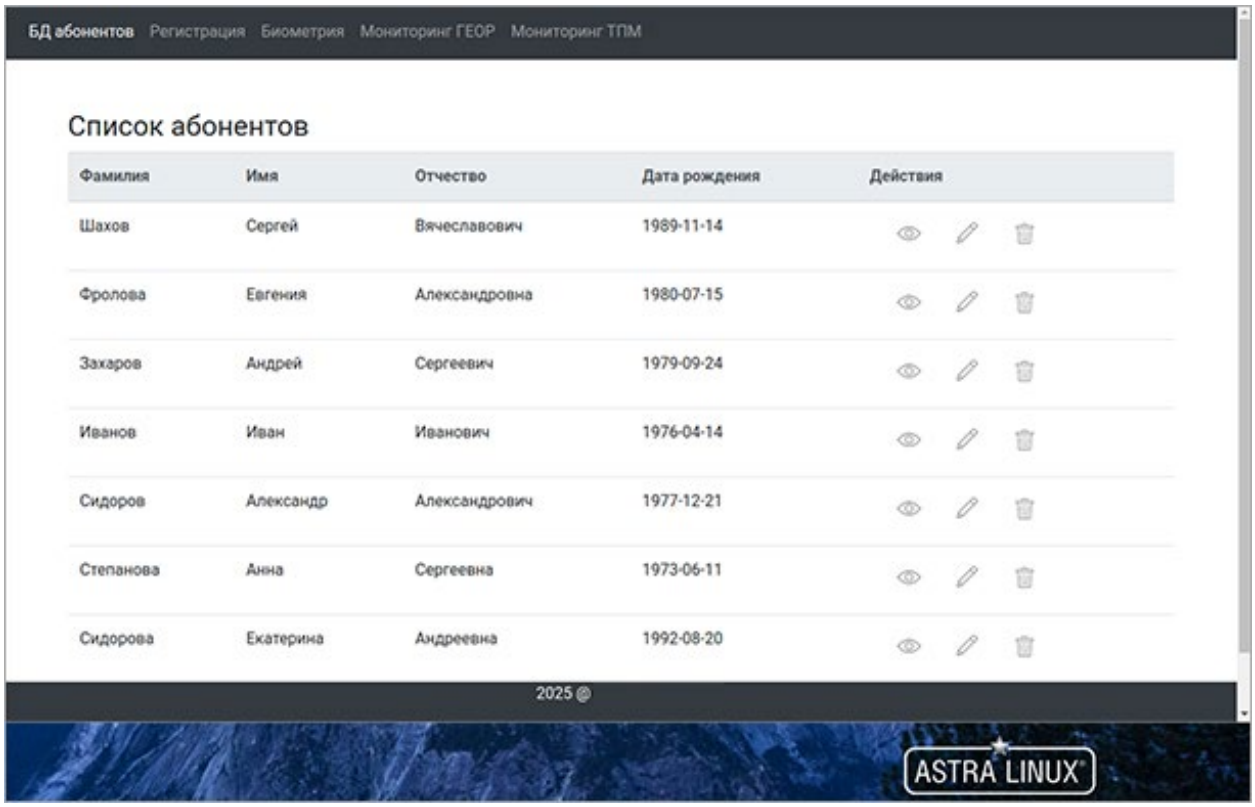

Рис. 2. – Вывод информации о зарегистрированных абонентах в пользовательском интерфейсе. Иллюстрация автора.

На рис. 3 и рис. 4 показан процесс мониторинга технического средства охраны, реализованный с помощью протокола WebSocket, передающего данные в пользовательский интерфейс в режиме реального времени.

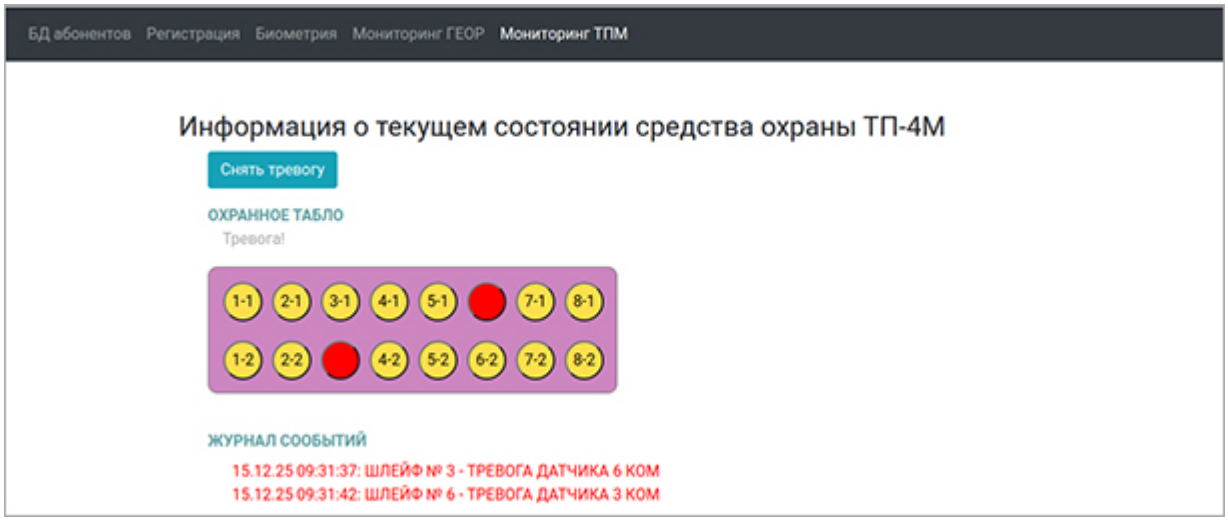

Рис. 3. – Вывод журнала событий и визуального отображения датчиков технического средства охраны в состоянии «Тревога». Иллюстрация автора.

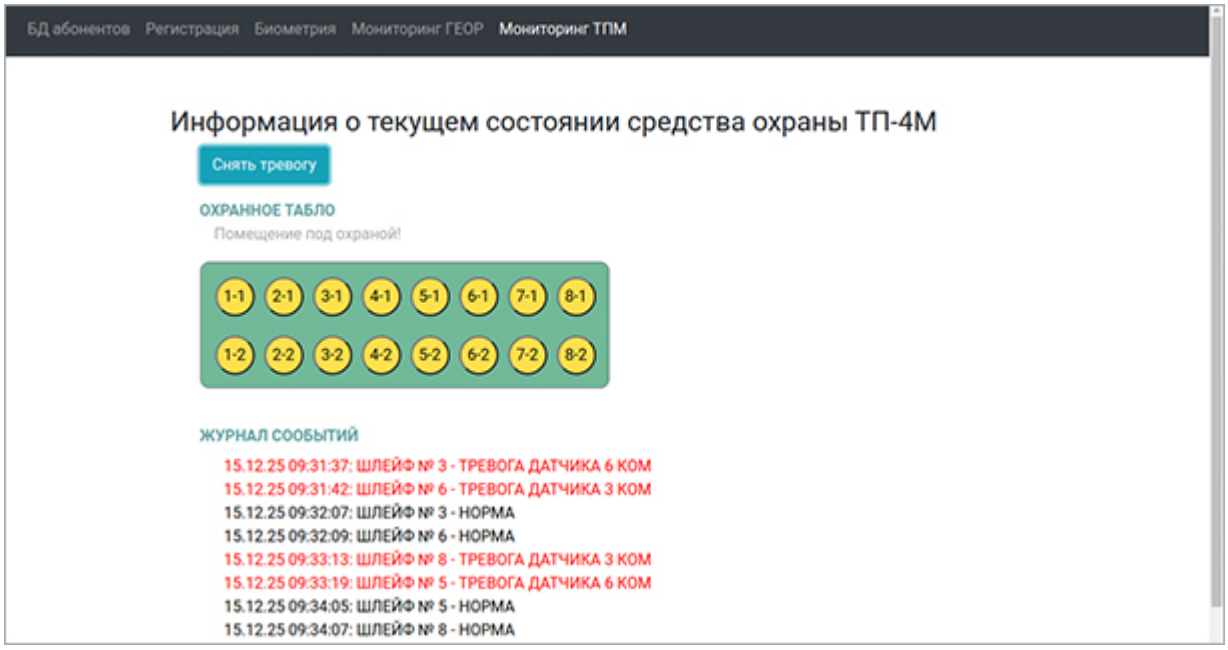

Рис. 4. – Вывод журнала событий состояния датчиков технического средства охраны. Иллюстрация автора.

Демонстрация обработки данных, полученных от устройства идентификации в режиме реального времени, приведена на рис. 5.

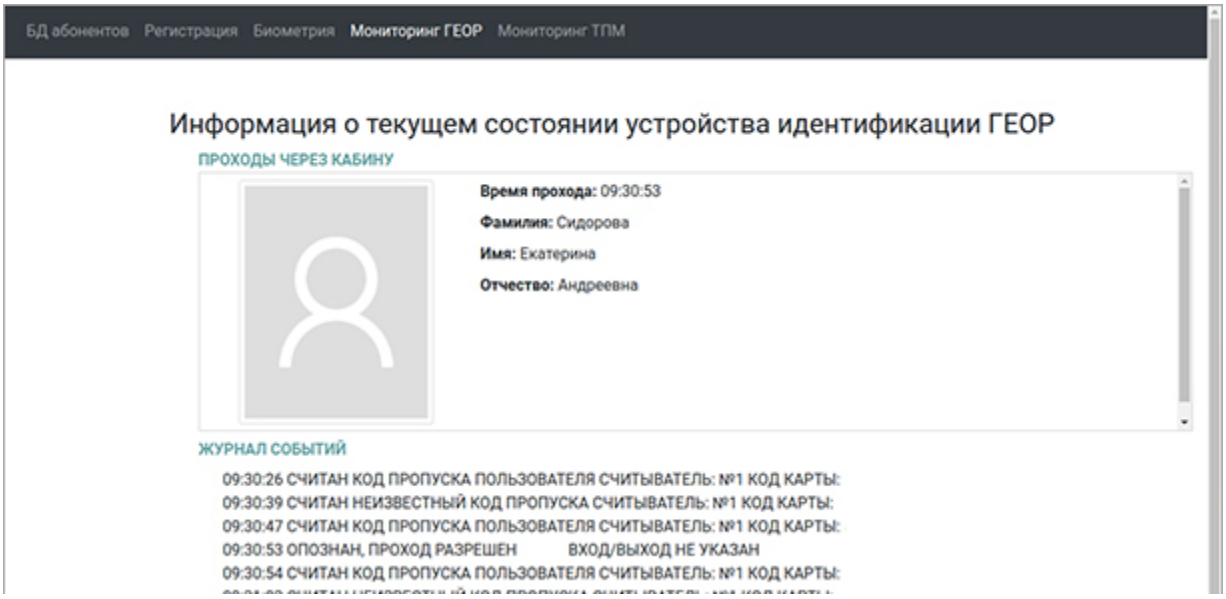

Рис. 5. – Демонстрация процесса мониторинга устройства идентификации. Иллюстрация автора.

На основании полученных результатов работы можно сделать вывод о том, что цель исследования была достигнута, все поставленные задачи выполнены. Реализованный прототип системы включает в себя базовые операции необходимого функционала системы управления, что послужит основой для дальнейшей разработки более масштабного программно-аппаратного комплекса физической защиты объекта.

Заключение

В данной статье была проведена оценка средств разработки веб-приложения на базе ОС Astra Linux Special Edition 1.8 с учетом требований к импортозамещению и сертификации программного обеспечения. На базе выбранных инструментов веб-разработки был реализован прототип системы управления комплексом физической защиты объекта, демонстрирующий взаимодействие с техническими средствами и хранилищем данных. В перспективе планируется применить мультиагентный подход и методы искусственного интеллекта [11] для детального проектирования системы и провести исследования по разработке микросервисной архитектуры, брокера

сообщений и менеджера устройств для обеспечения выполнения требований по масштабируемости, производительности, отказоустойчивости и быстродействию системы. Учитывая тот факт, что рассмотренные в статье средства разработки и структура клиент-серверного веб-приложения на базе отечественной операционной системы могут быть применены при построении автоматизированных систем управления в различных сферах, изложенный в статье материал будет иметь интерес для исследователей в других областях.

Литература

1. Скрипова В.Р., Аксенов К.А. Обзор решений для оптимизации системы управления комплексом защиты объекта // Инженерный вестник Дона, 2025, №3. URL: ivdon.ru/ru/magazine/archive/n3y2025/9884.

2. Rayhan Abu, Gross David. The Rise of Python: A Survey of Recent Research // Lab: CBECL's R&D Lab, 2023, September. URL: researchgate.net/publication/373633075.

3. Гек Д.К., Тихоненко Д.В. Angular как средство создания нативного приложения // Актуальные проблемы авиации и космонавтики, 2021, Том 2, С. 486-487. URL: cyberleninka.ru/article/n/angular-kak-sredstvo-sozdaniya-nativnogo-prilozheniya.

4. Алешина Т.А., Ткаченко А.Л., Широкова Е.В. Анализ нововведений в работе Angular/AngularJS, обновление 8.0 и его значение для web-разработчиков // Дневник науки, 2021, №12. URL: dnevniknauki.ru/images/publications/2021/12/physics/Alyoshina_Tkachenko_Shirokova.pdf.

5. Сермягин К.А., Пылькин А.Н. Построение эффективной архитектуры для высоконагруженных и масштабируемых сервисов на FastAPI: принципы и лучшие практики // Вестник РГРТУ, 2025, № 92. URL: vestnik.rsreu.ru/ru/archive/2025/vypusk-92/1662-1995-4565-2025-92-49-56.

6. Chen Junqiao. Model Algorithm Research based on Python Fast API // Frontiers in Science and Engineering, 2023, Volume 3, Number 9. URL: researchgate.net/publication/374103666.

7. Неустроев А.В. Сохранить JSON данные в базу данных SQLALchemy // Проблемы науки, 2016, № 12 (13), С. 44-45. URL: cyberleninka.ru/article/n/sohranit-json-dannye-v-bazu-dannyh-sqlalchemy.

8. Руднева С.В., Быков В.В. Сравнение различных ORM в Python: Анализ эффективности и применимости в современной разработке // Научный лидер, 2025, №11 (212). URL: scilead.ru/article/8320-sravnenie-razlichnikh-orm-v-python-analiz-eff.

9. Ермаков С.Г., Хомоненко А.Д., Шефнер А., Ляпунов В.Е., Кот Н.Д., Ахмедов Х.А. О разработке безопасных приложений на основе интеграции языка программирования Rust и СУБД PostgreSQL // Инженерный вестник Дона, 2024, №12. URL: ivdon.ru/ru/magazine/archive/n12y2024/9701.

10. Савостин П.А., Ефремова Н.Э. Практическое применение асинхронного программирования на языке Python при помощи пакета Asyncio // Программные системы и вычислительные методы, 2018, № 2, С. 11-16. URL: nbpublish.com/library_read_article.php?id=25851.

11. Aksyonov K.A., Sun Lina, Kalinin I.A. Developing AI Models Using a Code-Free Platform // Proceedings – 2025 International Russian Smart Industry Conference (SmartIndustryCon 2025): book. Institute of Electrical and Electronics Engineers Inc., Russia, Sochi, 24-28.03.2025, pp. 235-239. URL: ieeexplore.ieee.org/document/10986074.

References

1. Skripova V.R., Aksenov K.A. Inzhenernyj vestnik Dona, 2025, №3. URL: ivdon.ru/ru/magazine/archive/n3y2025/9884.

2. Rayhan Abu, Gross David. Lab: CBECL's R&D Lab, 2023, September. URL: researchgate.net/publication/373633075.

3. Gek D.K., Tikhonenko D.V. Aktual′nye problemy aviatsii i kosmonavtiki, 2021, Volume 2, pp. 486-487. URL: cyberleninka.ru/article/n/angular-kak-sredstvo-sozdaniya-nativnogo-prilozheniya.

4. Alyoshina T.A., Tkachenko A.L., Shirokova E.V. Dnevnik nauki, 2021, №12. URL: dnevniknauki.ru/images/publications/2021/12/physics/Alyoshina_Tkache.pdf.

5. Sermyagin K.A., Pylkin A.N. Vestnik RGRTU, 2025, № 92. URL: vestnik.rsreu.ru/ru/archive/2025/vypusk-92/1662-1995-4565-2025-92-49-56.

6. Chen Junqiao. Frontiers in Science and Engineering, 2023, Volume 3, Number 9. URL: researchgate.net/publication/374103666.

7. Neustroev A. V. Problemy nauki, 2016, № 12 (13), pp. 44-45. URL: cyberleninka.ru/article/n/sohranit-json-dannye-v-bazu-dannyh-sqlalchemy.

8. Rudneva S.V., Bykov V.V. Nauchnyj lider, 2025, №11 (212). URL: scilead.ru/article/8320-sravnenie-razlichnikh-orm-v-python-analiz-eff.

9. Ermakov S.G., Khomonenko A.D., Shefner A., Lyapunov V.E., Kot N.D., Akhmedov Kh.A. Inzhenernyj vestnik Dona, 2024, №12. URL: ivdon.ru/ru/magazine/archive/n12y2024/9701.

10. Savostin P.A., Efremova N.E. Programmnye sistemy i vychislitel′nye metody, 2018, № 2, pp. 11-16. URL: nbpublish.com/library_read_article.php?id=25851.

11. Aksyonov K.A., Sun Lina, Kalinin I.A. Proceedings – 2025 International Russian Smart Industry Conference (SmartIndustryCon 2025): book. Institute of Electrical and Electronics Engineers Inc., Russia, Sochi, 24-28.03.2025, pp. 235-239. URL: ieeexplore.ieee.org/document/10986074.

Дата поступления: 11.12.2025

Дата публикации: 24.01.2026