МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«Национальный исследовательский ядерный университет «МИФИ»

**Обнинский институт атомной энергетики —**

филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ» **(ИАТЭ НИЯУ МИФИ)**

Отделение интеллектуальных кибернетических систем

Кафедра Компьютерных Систем, Сетей и Технологий

**Научно-исследовательская работа**

по направлению подготовки: 09.03.01. «Информатика и вычислительная техника»

**«Разработка системы мониторинга и анализа движения вагонного парка»**

Выполнил: Марков К.А.  
студент группы ИВТ-Б21 (подпись, дата)

Руководитель образовательной программы 09.03.01   
«Информатика и вычислительная техника» Старков С.О.  
профессор ОИКС, доктор физ.-мат. наук (подпись, дата)

Обнинск, 2025

**РЕФЕРАТ**

Отчет 20 стр., 1 табл., 0 рис., 4 ист.

GOLANG, SUPERSET, POSTGRESQL, DOCKER, HTTP, БАЗА ДАННЫХ, СЕРВЕР, КЛИЕНТ, JSON  
Предмет исследования – процесс разработки системы мониторинга и анализа движения вагонного парка на основе веб-технологий.  
Цель работы – создание и описание технологического стека и архитектуры решения, обеспечивающего эффективное управление движения вагонного парка.

В работе использованы такие технологии, как Golang для разработки бизнес-логики, Superset для отчетности и визуализации данных, PostgreSQL в качестве основной СУБД, Docker для контейнеризации, JSON для конфигурации и обмена данными между сервисами.

Теоретическая значимость – работа направлена на исследование границ применимости, а также на разработку альтернативного от нынешнего способа управления вагонным парком.   
Практическая значимость – система управления и анализа для автоматизации рабочих процессов компании оператору ж/д подвижного состава ООО “Универсальные решения”.

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 5](#_Toc197713847)

[1 Технологический стек решения 7](#_Toc197713848)

[1.1 Анализ требований к технологическому стеку 7](#_Toc197713849)

[1.2 Подробное описание выбранных технологий 7](#_Toc197713850)

[1.3 Вывод по главе 10](#_Toc197713851)

[2. Архитектура предлагаемого решения 11](#_Toc197713852)

[2.1 Основные требования к архитектуре 11](#_Toc197713853)

[2.2 Сравнительный анализ современных подходов к архитектуре 12](#_Toc197713854)

[2.3 Общая схема архитектуры решения. 13](#_Toc197713855)

[2.4 Подробное описание компонентов системы. 13](#_Toc197713856)

[2.5 Структура и механизмы конфигурации 14](#_Toc197713857)

[2.6 Обеспечение надежности и производительности 14](#_Toc197713858)

[2.7 Вывод по главе 14](#_Toc197713859)

[3. Рекомендации по внедрению и оценка предлагаемого решения 15](#_Toc197713860)

[3.1 Рекомендации по практическому внедрению решения 15](#_Toc197713861)

[3.2 Анализ потенциальных рисков и способы их минимизации 15](#_Toc197713862)

[3.3 Сравнительный анализ с альтернативными подходами 15](#_Toc197713863)

[3.4 Технико-экономическая оценка предлагаемого подхода 17](#_Toc197713864)

[3.5 Перспективы дальнейшего развития решения 17](#_Toc197713865)

[3.6 Вывод по главе 18](#_Toc197713866)

[ЗАКЛЮЧЕНИЕ 19](#_Toc197713867)

[ИСТОЧНИКИ 20](#_Toc197713868)

**СОКРАЩЕНИЯ И ОБОЗНАЧЕНИЯ**

* **СУБД** - Система Управления Базами Данных
* **БД** – База Данных
* **BI** - Business Intelligence
* **API** - Application Programming Interface
* **JSON** - JavaScript Object Notation
* **CI/CD** – Continuous Integration, Continuous Delivery
* **CAC** - Customer Acquisition Cost

# ВВЕДЕНИЕ

В современном мире, где логистика и транспорт играют ключевую роль в экономике, эффективное управление вагонным парком становится критически важным для компаний, занимающихся грузоперевозками железнодорожным транспортом. Внедрение системы мониторинга и анализа движения вагонного парка является не просто желательным, а необходимым шагом для обеспечения конкурентоспособности, оптимизации затрат и повышения эффективности бизнеса.

1. ***Повышение эффективности использования вагонного парка:***
   1. *Сокращение времени простоя:* Система позволяет отслеживать местоположение вагонов в режиме реального времени, выявлять задержки и простои, что позволяет оперативно принимать меры для их устранения. Это приводит к увеличению оборачиваемости вагонов и, как следствие, к повышению эффективности использования активов.
   2. *Улучшение планирования:* Система предоставляет информацию о текущем местоположении и состоянии вагонов, что позволяет более точно планировать погрузку, разгрузку и ремонт, избегая задержек и простоев.
2. ***Повышение безопасности:***
   1. *Соблюдение нормативных требований:* Система помогает компаниям соблюдать требования законодательства в области безопасности железнодорожных перевозок.
3. ***Улучшение качества обслуживания клиентов:***
   1. *Предоставление информации о местоположении груза:* Клиенты могут в режиме реального времени отслеживать местоположение своего груза, что повышает их удовлетворенность и лояльность.
4. ***Принятие обоснованных управленческих решений:***
   1. *Анализ данных:* Система предоставляет широкий спектр данных о движении вагонного парка, которые можно использовать для анализа эффективности работы, выявления проблемных зон и принятия обоснованных управленческих решений.
   2. *Оценка эффективности инвестиций:* Система позволяет оценить эффективность инвестиций в обновление вагонного парка, внедрение новых технологий и оптимизацию логистических процессов.
5. ***Соответствие современным требованиям рынка:***
   1. *Облачные решения:* Многие системы мониторинга предлагаются в виде облачных решений, что снижает затраты на инфраструктуру и обеспечивает гибкость и масштабируемость.

Внедрение системы мониторинга и анализа движения вагонного парка является стратегически важным шагом для компаний, стремящихся к повышению эффективности, снижению затрат, повышению безопасности и улучшению качества обслуживания клиентов. В условиях растущей конкуренции и ужесточения требований рынка, наличие такой системы является необходимым условием для успешного развития бизнеса.

*Целью данной работы* является создание и описание технологического стека и архитектуры решения, обеспечивающего эффективное управление движения вагонного парка.

*Теоретическая значимость* исследования заключается в поиске границ применимости технологического стека в задаче, связанной с мониторингом вагонного парка, а также в разработке альтернативного от нынешнего способа управления вагонным парком.

*Практическая значимость* заключается в построении решения для управления, анализа и автоматизации рабочих процессов компании оператору ж/д подвижного состава ООО “Универсальные решения”.

# 1 Технологический стек решения

В работе использованы такие технологии, как Golang для разработки бизнес-логики, Superset для отчетности и визуализации данных, PostgreSQL в качестве основной СУБД, Docker для контейнеризации, JSON для конфигурации и обмена данными между сервисами.

### 1.1 Анализ требований к технологическому стеку

При разработке системы мониторинга вагонного парка необходимо учесть следующие требования к технологическому стеку: хранение большого объема данных (данные будут храниться за несколько прошлых лет), целостность данных, возможность построения аналитической отчетности (создание дашбордов, генерация отчетности для клиентов), визуализация данных, должна присутствовать поддержка различных форматов данных (JSON, XML, CSV,) для обмена информацией с другими системами, а также масштабируемость, отказоустойчивость, безопасный доступ к данным и функциональности системы с использованием ролей и разрешений.

### 1.2 Подробное описание выбранных технологий

*Golang (Go) для разработки бизнес-логики*

Golang отлично подходит для разработки серверной части системы, особенно для микросервисной архитектуры. Он справится с задачами обработки данных, мониторинга событий, интеграции с другими системами и предоставления API.

Go становится все более популярным языком для разработки бэкенда, особенно в области DevOps и облачных вычислений. Его преимущества:

* Высокая производительность: Go компилируется в машинный код, что обеспечивает высокую скорость работы и низкое потребление ресурсов. Это критически важно для системы, которая должна обрабатывать большие объемы данных в режиме реального времени.
* Параллелизм: Go имеет встроенную поддержку конкурентности (goroutines и channels), что позволяет эффективно использовать многоядерные процессоры и обрабатывать множество запросов одновременно.
* Статическая типизация: Статическая типизация позволяет выявлять ошибки на этапе компиляции, что повышает надежность системы.
* Простота: Go имеет простой и лаконичный синтаксис, что облегчает разработку, поддержку и масштабирование кода.
* Экосистема: Go имеет развитую экосистему библиотек и инструментов, особенно для работы с сетями, базами данных и облачными сервисами.

*Superset для отчетности и визуализации данных*

Superset идеально подходит для создания дашбордов и отчетов, отображающих ключевые показатели работы вагонного парка. Он позволяет визуализировать данные из различных источников (в том числе PostgreSQL) и предоставлять пользователям интерактивные инструменты для анализа.

Superset является современной платформой бизнес-аналитики (BI) с открытым исходным кодом, которая набирает популярность. Его преимущества:

* Интерактивные дашборды: Superset позволяет создавать интерактивные дашборды с различными типами графиков и диаграмм (линейные графики, столбчатые диаграммы, карты, круговые диаграммы и т.д.).
* Простота использования: Superset имеет интуитивно понятный интерфейс, который позволяет пользователям без специальных знаний создавать свои собственные отчеты и дашборды.
* Поддержка различных источников данных: Superset поддерживает широкий спектр источников данных, включая PostgreSQL, MySQL, Oracle, SQL Server, MongoDB и другие.
* Масштабируемость: Superset может быть масштабирован для обработки больших объемов данных и одновременной работы большого количества пользователей.
* Открытый исходный код: Superset имеет открытый исходный код, что позволяет адаптировать его к специфическим потребностям компании.

*PostgreSQL в качестве основной СУБД*

PostgreSQL - это надежная и масштабируемая СУБД, которая отлично подходит для хранения данных о движении вагонного парка, техническом состоянии вагонов, грузах и других связанных данных.

PostgreSQL является одной из самых популярных СУБД с открытым исходным кодом, широко используемой в различных отраслях. Его преимущества:

* Надежность: PostgreSQL известна своей надежностью и стабильностью.
* Масштабируемость: PostgreSQL может быть масштабирована для обработки больших объемов данных и одновременной работы большого количества пользователей.
* Расширяемость: PostgreSQL поддерживает широкий спектр расширений, которые позволяют добавлять новые функциональные возможности.
* Поддержка ACID-транзакций: PostgreSQL обеспечивает поддержку ACID-транзакций, что гарантирует целостность данных.
* Соответствие стандартам: PostgreSQL соответствует стандартам SQL и поддерживает широкий спектр функций SQL.
* Геопространственные данные: PostgreSQL с расширением PostGIS отлично подходит для хранения и обработки геопространственных данных, что необходимо для отслеживания местоположения вагонов.

*Docker для контейнеризации*

Docker позволяет упаковывать приложения (Golang сервисы, Superset) и их зависимости в контейнеры, что обеспечивает их переносимость и упрощает развертывание и управление.

Docker является стандартом де-факто для контейнеризации приложений. Его преимущества:

* + Изоляция: Контейнеры изолируют приложения друг от друга, что предотвращает конфликты зависимостей и повышает безопасность.
  + Переносимость: Контейнеры могут быть запущены на любой платформе, поддерживающей Docker.
  + Масштабируемость: Контейнеры легко масштабируются, что позволяет быстро увеличивать или уменьшать ресурсы в зависимости от нагрузки.
  + Автоматизация: Docker позволяет автоматизировать процессы сборки, развертывания и управления приложениями.
  + Упрощение разработки: Docker упрощает процесс разработки, позволяя разработчикам создавать и тестировать приложения в изолированной среде.

*JSON для конфигурации и обмена данными между сервисами*

JSON является удобным и широко распространенным форматом для хранения конфигурационных файлов и обмена данными между микросервисами.

JSON является стандартом де-факто для обмена данными в веб-приложениях и микросервисных архитектурах. Его преимущества:

* + Простота: JSON имеет простой и понятный синтаксис.
  + Легкость: JSON является легким форматом, что обеспечивает высокую скорость обработки.
  + Поддержка различными языками программирования: JSON поддерживается большинством современных языков программирования.
  + Читаемость: JSON является легко читаемым форматом, что упрощает отладку и поддержку.

### 1.3 Вывод по главе

Текущий технологический стек является современным, актуальным и хорошо подходит для разработки системы мониторинга вагонного парка. Он обеспечивает высокую производительность, масштабируемость, надежность и удобство разработки и управления.

# 2. Архитектура предлагаемого решения

### 2.1 Основные требования к архитектуре

Прежде чем приступить к проектированию системы, необходимо определить ключевые требования к ее архитектуре. Эти требования обозначат создание решения, способного эффективно обрабатывать большие объемы данных и адаптироваться к изменяющимся потребностям бизнеса. Ниже перечислены основные требования, которые будут учтены при разработке архитектуры системы мониторинга и анализа движения вагонного парка:

* *Масштабируемость*. Система должна справляться с ростом объема данных без существенного снижения производительности.
* *Отказоустойчивость*. Система должна продолжать функционировать даже при отказе отдельных компонентов. Необходимо предусмотреть механизмы обработки ошибок.
* *Модульность*. Система должна быть разбита на независимые модули (микросервисы), что упрощает разработку, тестирование, развертывание и внесение изменений.
* *Интегрируемость*. Система должна иметь возможность интеграции с другими системами (например, с системами учета) через API.
* *Безопасность*. Система должна обеспечивать защиту данных от несанкционированного доступа. Необходимо реализовать механизмы администрирования пользователей.
* *Производительность*. Система должна обеспечивать быструю обработку данных и отклики на запросы пользователей.

### 2.2 Сравнительный анализ современных подходов к архитектуре

|  |  |  |  |
| --- | --- | --- | --- |
| Подход | Преимущества | Недостатки | Применимость к НИР |
| Монолитная архитектура | Простота разработки и развертывания | Сложно масштабировать, низкая отказоустойчивость, сложно вносить изменения | Не подходит по требованиям |
| Микросервисная архитектура | Высокая масштабируемость, отказоустойчивость, модульность, гибкость | Сложность разработки и управления, повышенные требования к инфраструктуре | Удовлетворяет требования |
| Серверная архитектура | Простота разработки, масштабируемость за счет облачных провайдеров | Зависимость от облачного провайдера, возможны ограничения по функциональности | Можно рассмотреть для отдельных компонентов |

Таблица 1. Сравнительный анализ подхода к архитектуре.

Микросервисная архитектура наилучшим образом отвечает требованиям.

### 2.3 Общая схема архитектуры решения.



### 2.4 Подробное описание компонентов системы.

Архитектура системы мониторинга и анализа движения вагонного парка базируется на микросервисной архитектуре, где каждый микросервис отвечает за определенную функцию. Взаимодействие между сервисами осуществляется посредством API Gateway, который выступает в качестве единой точки входа для всех клиентских запросов. Данный подход обеспечивает модульность, масштабируемость и отказоустойчивость.

*Сервис агрегации данных*: отвечает за прием, обработку и сохранение данных о движении вагонного парка из различных источников. Он выполняет важную функцию унификации данных, приводя их к единому формату, используя соответствия наименований столбцов в БД. Сервис обрабатывает несоответствия во входных форматах данных: в случае обнаружения несоответствия формату, система оповещает пользователя и предлагает вручную добавить новые соответствия наименований для корректной обработки данных.

*Сервис формирования отчетности*: на основе данных, агрегированных и сохраненных в БД, формирует отчеты о движении вагонного парка, используя предопределенные шаблоны и правила.

*Сервис взаимодействия с API почты*: инкапсулирует логику взаимодействия с внешним API почтового сервиса.  
 *Фронтенд сервис*: отвечает за визуальное отображение данных и взаимодействие с пользователем. Он предоставляет интерфейс для простого использования функций сервиса, также для администрирования сервиса.

*Сервис администрирования*: предоставляет функциональность для администрирования системы, включая управление пользователями, настройку прав доступа, получение данных из БД для отображение актуальной информации о пользователях.

*Сервис визуализации данных (Superset):* Этот сервис отвечает за визуализацию данных, собранных и обработанных другими сервисами системы. Superset предоставляет веб-интерфейс для создания интерактивных дашбордов, графиков и отчетов на основе данных из базы данных. Он позволяет пользователям самостоятельно настраивать параметры и фильтры для построения визуализаций.

### 2.5 Структура и механизмы конфигурации

Конфигурация системы будет храниться в Github-репозитории. Для управления конфигурациями настроен Github Actions. Данный инструмент CI/CD обеспечивает запуск тестов, сборку Docker-образов и их развертывание на удаленные серверы при каждом изменении кода.

### 2.6 Обеспечение надежности и производительности

В системе предусмотрены механизмы горизонтального масштабирования и балансировщика нагрузок, что значительно повышает отказоустойчивость, а также производительность.

### 2.7 Вывод по главе

В данной главе предложена архитектура системы мониторинга и анализа движения вагонного парка, основанная на микросервисной архитектуре и принципах горизонтального масштабирования. Данная архитектура отвечает требованиям масштабируемости, отказоустойчивости, модульности и производительности, что позволяет эффективно решать поставленные задачи. Разделение на микросервисы обеспечивает гибкость и возможность независимого развития отдельных компонентов системы. Использование сервиса управления конфигурацией упрощает администрирование и позволяет динамически настраивать систему. Наличие балансировщика нагрузки позволяет распределять нагрузку между несколькими экземплярами сервисов, повышая отказоустойчивость и производительность системы. Предложенные механизмы обеспечения надежности гарантируют стабильную и эффективную работу сервиса.

# 3. Рекомендации по внедрению и оценка предлагаемого решения

### 3.1 Рекомендации по практическому внедрению решения

Внедрение системы рекомендуется начать с развертывания системы на ограниченном количестве данных в течение определенного периода. Это позволит протестировать функциональность системы в реальных условиях, выявить потенциальные проблемы и доработать необходимые компоненты.

После успешного завершения “пилотного проекта” постепенно увеличивать количество вводных данных и расширять функциональность системы. Мониторить производительность и стабильность системы необходимо на каждом этапе.  
 Необходимо протестировать сервис при максимальных нагрузках, провести имитирование пика нагрузок (100 человек и 1000 файлов) для тестирования отказоустойчивости при сбоях отдельных микросервисов. не подводя риску рабочий процесс организации.  
 После завершения тестирования сервиса, необходимо провести обучение персонала по работе с системой, включая использование интерфейса, функции администрирования для уполномоченных работников, реагирование на аварийные ситуации.

Произвести миграцию данных с предыдущей системы управления и учета на новую.

### 3.2 Анализ потенциальных рисков и способы их минимизации

*Несанкционированный доступ к данным*. Необходимо предусмотреть и обезопасить сервис от несанкционированного доступа к БД. Избегать в коде программ незащищенных от SQL-injecting методов, избегать конкатенации строк и тд.

*Потеря данных из БД*. Для минимизации рисков возникновения данной проблемы необходимо создавать резервные копии, использовать репликацию базы данных. Также рекомендуется ограничить доступ к базе данных, используя принцип наименьших привилегий.

### 3.3 Сравнительный анализ с альтернативными подходами

В качестве альтернативных подходов к разработке системы мониторинга и анализа движения вагонного парка можно рассмотреть:

1. Использование готового решения от крупных вендоров (например, Oracle OTM или SAP TM):

* системы уже готовы и не требуют разработки с нуля,
* имеют большой функционал,
* обеспечивают техническую поддержку и обновление системы.

Однако у данного подхода есть ряд недостатков:

* стоимость лицензий и внедрения очень высока,
* компания становится зависимой от вендора и его ценовой политики,
* настройка и доработка функционала готовых решений может быть сложной и дорогостоящей. Монолитная архитектура затрудняет масштабирование и внедрение новых технологий.

2. Разработка на базе PaaS платформы (например, AWS IoT Core + AWS Lambda + Amazon RDS):

* облачные платформы позволяют легко масштабировать систему в зависимости от нагрузки,
* PaaS платформы предоставляют большой выбор сервисов, которые можно комбинировать для построения системы с нужным функционалом,
* нет необходимости в собственной серверной инфраструктур.

Недостатки:

* компания становится зависимой от выбранного облачного провайдера,
* необходимо уделять особое внимание безопасности данных в облаке,
* стоимость использования облачных сервисов может быть сложной для прогнозирования и контроля.

Предлагаемое в данной работе решение, основанное на микросервисной архитектуре, превосходит альтернативные подходы по следующим параметрам:

* микросервисная архитектура позволяет легко масштабировать систему в соответствии с растущими потребностями
* обеспечивает гибкость в развитии и внедрении нового функционала
* разделение на независимые микросервисы обеспечивает отказоустойчивость системы
* простота в поддержке работоспособности, легче обновлять, чем в монолитных системах

### 3.4 Технико-экономическая оценка предлагаемого подхода

Экономическая польза для компании от внедрения системы мониторинга вагонного парка состоит в следующем:

1. Увеличение скорости обработки рутинных задач сотрудниками компании:

* *Сокращение времени на обработку и мониторинг состояния вагонного парка.*
* *Сокращение времени на формирование отчетности для клиентов.*

1. Сокращения доли простоя вагонного парка:

* *Аналитическая отчетность будет направлена на отражение текущего состояния вагонного парка.* Это может помочь руководителю компании принимать решение, опираясь на аналитические данных.

1. Уменьшение доли упущенных продаж:

* *Рост скорости обработки входящей потребности от потенциальных клиентов.* Отсюда расширение воронки из потенциальных клиентов в новых клиентов, что влечет за собой рост оборота компании.

1. Уменьшение стоимости привлечения новых клиентов (CAC)

* *Система мониторинга и аналитическая отчетность даст более полное представление о распределении спроса относительно направления перевозок.* На основе этих данных можно более точечно целиться в клиентов.

### 3.5 Перспективы дальнейшего развития решения

В ближайшем будущем можно рассмотреть развитие аналитического модуля, микросервис мониторинга состояния вагонов, интеграция с системами других участников перевозочного процесса.

Расширение функционала аналитического модуля для более глубокого анализа данных о движении вагонов, включая выявление закономерностей, аномалий и тенденций. Это позволит оптимизировать работу вагонного парка и принимать более обоснованные управленческие решения.

Микросервис мониторинга технического состояния вагонов может содержать факт и длительность простоя вагонного парка, общий и порожний пробег, местоположение вагонного парка на момент последнего обновления данных.

Обмен данными с системами грузоотправителей, грузополучателей и других организаций, участвующих в перевозочном процессе, для повышения эффективности и прозрачности взаимодействия.

### 3.6 Вывод по главе

В данной главе представлены рекомендации по практическому внедрению разработанной системы мониторинга и анализа движения вагонного парка, проведен анализ потенциальных рисков и предложены способы их минимизации. Проведен сравнительный анализ с альтернативными подходами, а также технико-экономическая оценка предлагаемого решения. Определены перспективы дальнейшего развития системы. Внедрение предлагаемой системы позволит повысить эффективность управления вагонным парком. Результаты проведенной технико-экономической оценки подтверждают целесообразность внедрения предлагаемого решения. Перспективы дальнейшего развития системы указывают на потенциал для дальнейшего повышения эффективности и ценности системы в будущем.

# ЗАКЛЮЧЕНИЕ

В ходе исследования было создано решение, базирующееся на мощном, но гибком технологическом фундаменте. Golang для реализации бизнес-логики, Superset для отображения данных, PostgreSQL как надежный инструмент хранения информации, Docker для обеспечения стабильности и масштабируемости, и JSON как универсального “языка общения” между компонентами – это выбор, направленный на создание оптимального решения.

Микросервисная архитектура системы позволяет ей расти и развиваться, что дает возможность гибко реагировать на изменения, добавлять новые функции и улучшать отдельные компоненты без риска поломки всей системы.

Предложенная система даст возможность решить часть проблем компании ООО “Универсальные решения”, путем оперативного принятия решений, основанных на реальных данных, оптимизации существующих процессов и, как следствие, увеличения прибыли.

Проведен анализ рисков, оценка экономической эффективности и намечены перспективы развития. Учитывая постоянно меняющуюся среду, был заложен фундамент для дальнейшего роста и совершенствования системы, что добавило ей потенциал для интеграции с другими системами и глубокого анализа данных.

# ИСТОЧНИКИ

* Go. The Go Programming Language, доступ: <https://go.dev/> (дата обращения 20.04.2025г.)
* Apache Superset, доступ: <https://superset.apache.org/> (дата обращения 20.04.2025г.)
* PostgreSQL: The world’s most advanced open source database, доступ: <https://www.postgresql.org/> (дата обращения 20.04.2025г.)
* Docker: Build, Share, Run, доступ: <https://www.docker.com/> (дата обращения 20.04.2025г.)