|  |  |
| --- | --- |
|  | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Системы обработки информации и управления

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

***К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ***

***НА ТЕМУ:***

**Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки**

Студент \_\_ИУ-5 82\_\_ **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_А.М.Ким\_\_**

(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель ВКР **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_А.А.Максаков\_\_**

(Подпись, дата) (И.О.Фамилия)

Консультант **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_М.В.Черненький\_\_**

(Подпись, дата) (И.О.Фамилия)

Нормоконтролер **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_Ю.Н.Кротов \_\_**

(Подпись, дата) (И.О.Фамилия)

2024 г.

АННОТАЦИЯ

Расчётно-пояснительная записка квалификационной работы бакалавра содержит 68 страниц. С приложениями объем составляет 90 страниц. Работа включает в себя 33 таблицы и 26 иллюстраций. В процессе выполнения было использовано 17 источников.

Объектом разработки является система прогнозирования нагрузки СХД.

Хххххххххххххххххххххххххххххххххххх ххххххххххххххххххххххх ххх ххххххххххххххххххххххххххх хххххххххххххххххх ххххххххххххххххххх хххх ххххххххххххххххххххххххххххххххххххх хххххххххххххх хххххххххх хх ххххххххххххххххх ххххххххххххххххх хххххххххххххх ххххххххххххх ххх хххххххххххх.

Цель работы заключается хххххххххххх хххххххххххх ххххххххх хх хххххххх ххххххххххххххх ххх хххххххххххх ххххххххххххххх ххххх ххххххххххххх хххххххххххххх хххххххххх хх ххххххххххххххххх хххххх ххххххххххх хххххххххххххх ххххххххххххх ххх хххххххххххх.

В процессе выполнения квалификационной работы бакалавра хххххххххххх хххххххххххх ххххххххх хх хххххххх ххххххххххххххх ххх хххххххххххх ххххххххххххххх ххххх ххххххххххххх хххххххххххххх хххххххххх хх ххххххххххххххххх хххххх ххххххххххх хххххххххххххх ххххххххххххх ххх хххххххххххх.

Пояснительная записка содержит 3 приложения.

СОДЕРЖАНИЕ

[АННОТАЦИЯ 2](#_Toc168277935)

[СОДЕРЖАНИЕ 3](#_Toc168277936)

[СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ 4](#_Toc168277937)

[ВВЕДЕНИЕ 5](#_Toc168277938)

[1 ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ 6](#_Toc168277939)

[1.1 Общетехническое обоснование разработки 6](#_Toc168277940)

[1.1.1 Постановка задачи проектирования 6](#_Toc168277941)

[1.1.2 Описание предметной области 7](#_Toc168277942)

[1.1.3 Выбор критериев качества 8](#_Toc168277943)

[1.1.4 Анализ прототипов и аналогов 9](#_Toc168277944)

[1.1.5 Требования предъявляемые к разрабатываемой системе 19](#_Toc168277945)

[2 ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ 21](#_Toc168277946)

[2.1 Исследование моделей прогнозирования 21](#_Toc168277947)

[3 КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 30](#_Toc168277948)

[3.1 Конструкторская часть 30](#_Toc168277949)

[3.1.2 Технологии бэкенда 32](#_Toc168277950)

[3.1.3 СУБД 34](#_Toc168277951)

[3.1.4 Модель прогнозирования 38](#_Toc168277952)

[3.1.5 Архитектура системы 39](#_Toc168277953)

[3.1.6 Пользовательский интерфейс 41](#_Toc168277954)

[ЗАКЛЮЧЕНИЕ 46](#_Toc168277955)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 47](#_Toc168277956)

[ПРИЛОЖЕНИЕ А ГРАФИЧЕСКАЯ ЧАСТЬ 49](#_Toc168277957)

[ПРИЛОЖЕНИЕ В ТЕХНИЧЕСКОЕ ЗАДАНИЕ 51](#_Toc168277958)

[ПРИЛОЖЕНИЕ В ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ 57](#_Toc168277959)

# СПИСОК ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

1. **СХД** – система хранения данных
2. **ПО** – програмное обеспечение
3. **Python** – высокоуровневый язык программирования
4. **JS** – JavaScript высокоуровневый язык программирования
5. **DOM** – Document Object Model
6. **AJAX** – Asynchronous JavaScript and XML
7. **React** – фреймворк для веб-разработки
8. **СУБД** – система управления базами данных
9. **БД** – база данных
10. **PostgreSQL** – база данных
11. **ORM** – object-relational mapping
12. **HTTP** – hypertext transfer protocol
13. **TCP** – transmission control protocol
14. **REST API** – representational state transfer application programming interface
15. **SVM** – support vector machine

ВВЕДЕНИЕ

Актуальность данной работы обусловлена стремительным ростом объемов данных в современном мире. С каждым годом количество генерируемой информации увеличивается, что связано с рядом факторов. Во-первых, увеличение числа интернет-пользователей в мире способствует генерации большего объема данных. Во-вторых, рост популярности контента высокого качества, требует больших мощностей для их хранения и обработки.

Системы хранения данных (СХД) играют ключевую роль в управлении этими объемами информации. Они не только обеспечивают хранение данных, но и гарантируют их доступность и безопасность, что является критически важным для бизнес-процессов современных компаний. Перегрузка таких систем может привести к серьезным сбоям, вплоть до полной потери данных, что недопустимо в бизнес-среде.

В связи с этим возникает острая необходимость в прогнозировании нагрузок на системы хранения данных. Эффективное прогнозирование позволяет не только оптимизировать использование ресурсов, но и предотвратить длительные простои, минимизируя тем самым потери данных и финансовые издержки. Для решения этой задачи активно используются современные методы искусственного интеллекта и машинного обучения, которые позволяют анализировать большие объемы информации и делать точные прогнозы в реальном времени.

1 ПОСТАНОВКА ЗАДАЧ РАЗРАБОТКИ

1.1 Общетехническое обоснование разработки

1.1.1 Постановка задачи проектирования

Для организации современного взаимодействия между компонентами системы прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки необходимо разработать программное обеспечение.

Система должна предоставлять возможность пользователям выбирать параметры СХД, таблицы из базы данных, изменять уровни прогнозирования, выбирать режимы скользящего окна, указывать интервалы и их количество, а также выбирать уровень предсказания. Также предусмотрена возможность нахождения глобального минимума и построения облака точек.

Важной частью системы является возможность изменения уровня прогнозирования через интерфейс приложения. Пользователи смогут задавать параметры и SQL-запросы для изменения уровня прогнозирования, что обеспечит гибкость и адаптивность системы. Дополнительно, система будет поддерживать режим скользящего окна, который можно будет настроить как автоматически, так и вручную, указывая необходимые интервалы.

Помимо этого, пользователи должны иметь возможность визуализировать результаты прогнозирования на графике. Это включает отображение исторических данных, прогнозируемых уровней нагрузки и облака точек, если оно выбрано. Визуализация данных позволит пользователям быстро оценивать текущую ситуацию и принимать обоснованные решения на основе полученных прогнозов.

Таким образом, задача проектирования может быть сформулирована следующим образом:

* разработать интерфейс для указания параметров СХД и выбора таблиц из базы данных;
* реализовать возможность изменения уровня прогнозирования;
* обеспечить выбор режима скользящего окна: автоматический и ручной;
* включить опции для определения уровня прогнозирования;
* визуализация графиков, отображающие исторические данные, прогнозируемые уровни, облако точек и прогноз;
* реализовать возможность развертывания системы на любой платформе.

1.1.2 Описание предметной области

Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки представляет собой высокотехнологичное решение, предназначенное для анализа и предсказания нагрузки на СХД. В условиях стремительного роста объемов данных, обусловленного развитием информационных технологий и цифровой трансформацией, вопросы эффективного управления и прогнозирования нагрузки на СХД приобретают особую значимость. Современные СХД являются критически важными компонентами ИТ-инфраструктуры организаций, обеспечивая хранение, обработку и доступ к огромным массивам данных.

Основная цель системы прогнозирования нагрузки СХД заключается в предотвращении перегрузок и сбоев, которые могут привести к потере данных, снижению производительности и нарушению бизнес-процессов. Для достижения этой цели разрабатываемая система должна обеспечивать точное прогнозирование нагрузки на основе анализа исторических данных и текущих параметров работы СХД. Прогнозирование позволяет заранее выявлять потенциальные проблемы и принимать превентивные меры для их предотвращения, что способствует повышению надежности и эффективности работы СХД.

Система будет состоять из нескольких ключевых компонентов, включая фронтенд, бэкенд и базу данных. Фронтенд представляет собой пользовательский интерфейс, через который пользователи будут взаимодействовать с системой, настраивать параметры прогнозирования и просматривать результаты анализа. Бэкенд отвечает за обработку данных, выполнение алгоритмов прогнозирования и взаимодействие с базой данных. База данных будет хранить исторические данные о нагрузке на СХД.

На рисунке 1 показано взаимодействие оператора (пользователя) и СХД с системой.

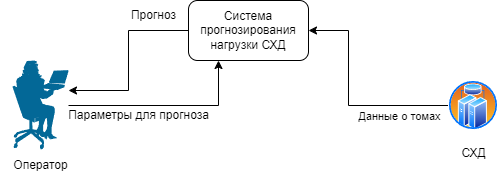


Рисунок 1 - Предметная область

**1.1.3 Выбор критериев качества**

Для разрабатываемого программного изделия приоритетными являются следующие критерии качества:

1. точность прогнозирования;
2. производительность;
3. интеграция и совместимость;
4. отказоустойчивость;
5. интуитивно понятный интерфейс (оценка).

Назначим весовые коэффициенты обозначенным критериям. Результаты приведены в таблице 1.

Таблица 1 - Проранжированные критерии качества

| №  п/п | Название критерия | Весовой коэффициент |
| --- | --- | --- |
| 1. | Точность прогнозирования | 5 α |
| 2. | Производительность | 3 α |
| 3. | Интеграция и совместимость | 3 α |
| 4. | Отказоустойчивость | 2 α |
| 5. | Интуитивно понятный интерфейс (оценка) | α |

1.1.4 Анализ прототипов и аналогов

Для управления объёмом данных на хранилищах и предотвращения их переполнения используются системы уведомлений о пороговых значениях заполнения. В анализе и сравнении находится ключевая особенность актуальности моей системы – аналогов практически нет. Рассмотрим похожие системы для дальнейшего сравнения.

#### 1.1.4.1 HPE System Reporter

HPE System Reporter является инструментом для мониторинга и отчётности, который используется в управлении системами хранения данных, предоставляемых Hewlett Packard Enterprise (HPE). Этот инструмент позволяет операторам систем получать подробные данные о производительности и использовании ресурсов в их хранилищах данных.

Основные преимущества HPE System Reporter включают:

1. Мониторинг производительности: System Reporter предоставляет данные о производительности системы хранения, что позволяет операторам оптимизировать работу системы и своевременно выявлять потенциальные проблемы.
2. Управление пороговыми значениями и оповещениями: Инструмент

позволяет настроить пороговые значения для различных параметров системы хранения, таких как использование дискового пространства или производительность I/O операций. При достижении этих порогов система может автоматически отправлять оповещения, что помогает предотвратить перегрузку системы или её сбой.

1. Отчётность: System Reporter предоставляет обширные возможности для создания отчётов, которые могут быть настроены для отображения различной информации о состоянии системы хранения. Эти отчёты могут помочь в анализе тенденций использования и планировании будущих расширений системы хранения.

HPE System Reporter представляет собой мощный инструмент для операторов данных, который помогает обеспечивать высокую производительность и доступность систем хранения данных, а также помогает в принятии обоснованных решений на основе данных о текущем состоянии инфраструктуры хранения.

На рисунке 1 представлен интерфейс системы.

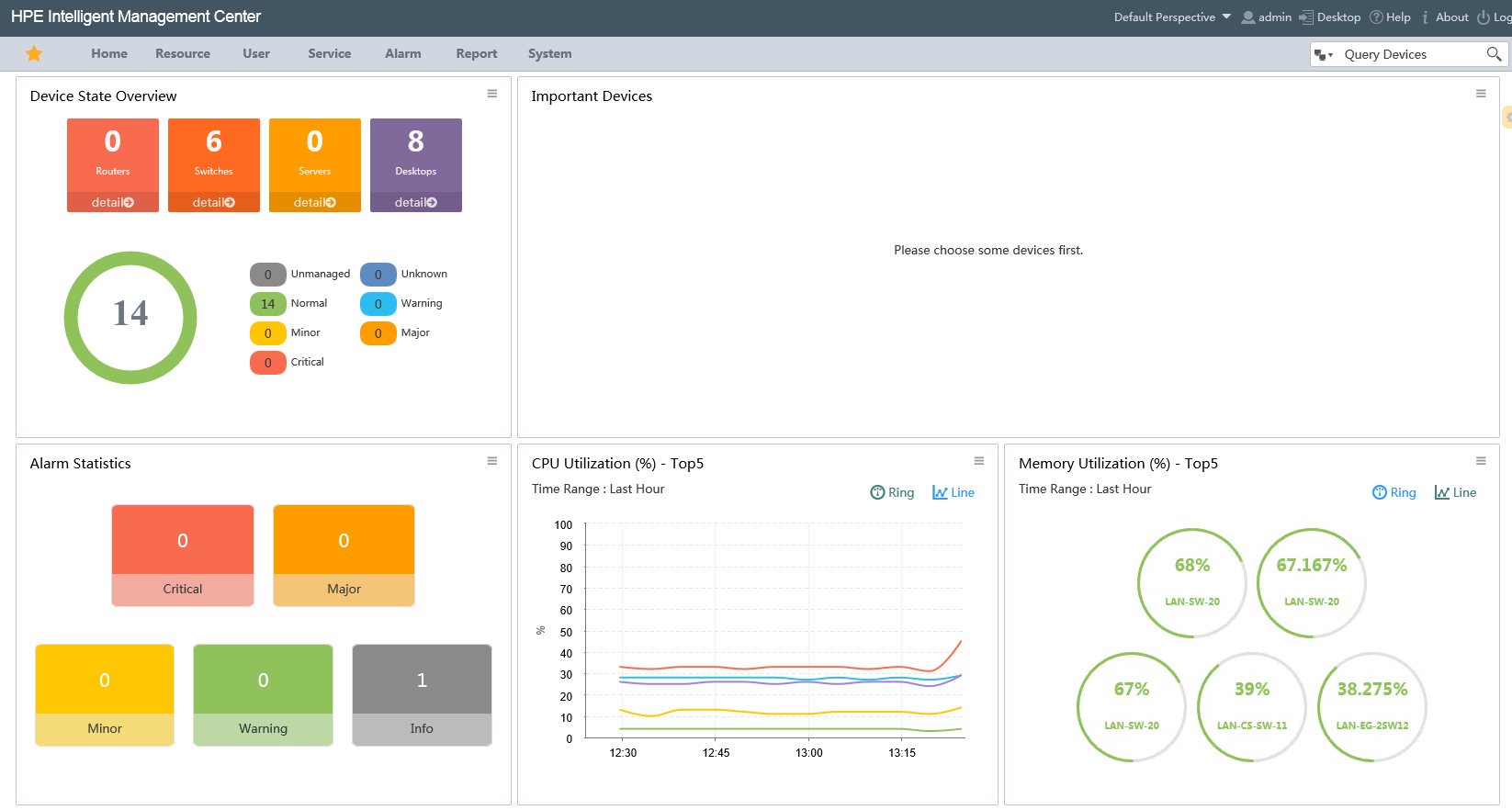


Рисунок 2 – Интерфейс HPE system reporter

#### 1.1.4.2 SolarWinds Storage Resource Monitor

SolarWinds Storage Resource Monitor — это инструмент для мониторинга и управления системами хранения данных, разработанный компанией SolarWinds. Он предназначен для системных администраторов и операторов, предоставляя подробные данные о производительности и использовании ресурсов в их хранилищах данных.

Основные преимущества SolarWinds Storage Resource Monitor включают:

1. Мониторинг производительности: Storage Resource Monitor

обеспечивает мониторинг в режиме реального времени для различных аспектов систем хранения данных, включая использование емкости, производительность ввода/вывода и доступность ресурсов. Это позволяет операторам выявлять узкие места и оптимизировать работу системы для повышения эффективности.

1. Управление пороговыми значениями и оповещениями: Инструмент

позволяет настраивать пороговые значения для ключевых параметров системы хранения. При достижении этих порогов система автоматически отправляет оповещения, что помогает операторам быстро реагировать на потенциальные проблемы и предотвращать сбои системы.

1. Отчётность и аналитика: Storage Resource Monitor предоставляет

мощные возможности для создания и настройки отчётов. Эти отчёты позволяют анализировать текущие тенденции использования ресурсов и планировать будущие расширения системы хранения. Инструмент также включает визуальные панели и графики для лучшего понимания данных.

1. Прогнозирование использования: Система использует машинное

обучение для прогнозирования будущих потребностей в емкости на основе исторических данных. Это помогает операторам заранее планировать обновления и предотвращать нехватку ресурсов.

SolarWinds Storage Resource Monitor является мощным инструментом для обеспечения высокой производительности и доступности систем хранения данных, а также для принятия обоснованных решений на основе детализированных данных о состоянии инфраструктуры.

На рисунке 3 представлен интерфейс SolarWinds Storage Resource Monitor.

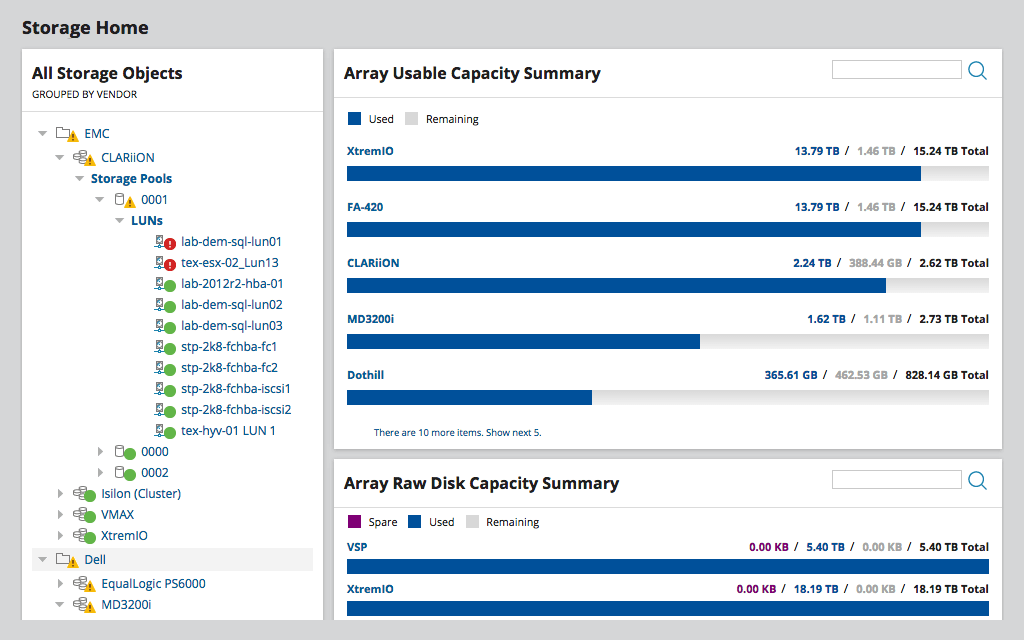


Рисунок 3 – Интерфейс SolarWinds Storage Resource

#### 1.1.4.3 Сравнение систем

На основе представленных критериев в таблице 1 проведем сравнение систем для обоснования надобности реализации системы прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки. В таблице 2 представлены критерии с их обозначениями и единицами измерения.

Таблица 2 - Критерии качества

| Критерий | Обозначение | Единица измерения |
| --- | --- | --- |
| Точность прогнозирования | K1 | (количественный критерий) |
| Производительность | K2 | (качественный критерий) |
| Интеграция и совместимость | K3 | (качественный критерий) |
| Отказоустойчивость | K4 | (качественный критерий) |
| Интуитивно понятный интерфейс (оценка) | K5 | (количественный критерий) |

Оценим системы на основе критериев представленных выше. В таблицах 3-5 показан результат оценки.

Таблица 3 – Результат оценки HPE System Reporter

| Критерий | Оценка |
| --- | --- |
| Точность прогнозирования | 9 |
| Производительность | Хорошая |
| Интеграция и совместимость | Отсутствует |
| Отказоустойчивость | Средняя |
| Интуитивно понятный интерфейс (оценка) | 8 |

Таблица 4 – Результат оценки SolarWinds Storage Resource Monitor

| Критерий | Оценка |
| --- | --- |
| Точность прогнозирования | 95 |
| Производительность | Средняя |
| Интеграция и совместимость | Средняя |
| Отказоустойчивость | Высокая |
| Интуитивно понятный интерфейс (оценка) | 4 |

Таблица 5 – Результат оценки системы прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки

| Критерий | Оценка |
| --- | --- |
| Точность прогнозирования | 93 |
| Производительность | Высокая |
| Интеграция и совместимость | Полная |
| Отказоустойчивость | Высокая |
| Интуитивно понятный интерфейс (оценка) | 7 |

После оценки систем обозначим варианты в таблице 6.

Таблица 6 – Варианты систем

| Критерий | Оценка |
| --- | --- |
| HPE System Reporter | В1 |
| SolarWinds Storage Resource Monitor | В2 |
| Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки | В3 |

Далее необходимо представить качественные критерии в виде количественных для корректного сравнения. Их представления оформлены в виде таблиц 7-11.

Таблица 7 - Вербально-числовая шкала для К2 (производительность)

| Качественная оценка | Количественная оценка |
| --- | --- |
| Высокая | 1 |
| Хорошая | 0,7 |
| Средняя | 0,3 |
| Низкая | 0 |

Таблица 8 – Вербально-числовая шкала для К3 (интеграция и совместимость)

| Качественная оценка | Количественная оценка |
| --- | --- |
| Полная | 1 |
| Средняя | 0,5 |
| Отсутствует | 0 |

Таблица 9 – Вербально-числовая шкала для К4 (отказоустойчивость)

| Качественная оценка | Количественная оценка |
| --- | --- |
| Высокая | 1 |
| Средняя | 0,7 |
| Низкая | 0,4 |
| Отсутствует | 0 |

Проведем нормализацию критериев по методу полной нормализации формула (1).

(1)

| Код критерия | Критерий | Варианты систем | | |
| --- | --- | --- | --- | --- |
| В1 | В2 | В3 |
| К1 | Точность прогнозирования | 0 | 1 | 0,3 |
| K5 | Интуитивно понятный интерфейс (оценка) | 1 | 0 | 0,75 |

Распределим оценки по вариантам систем, что представлено в таблице 10.

Таблица 10 – Количественные оценки по вариантам систем

| Код критерия | Критерий | Варианты систем | | |
| --- | --- | --- | --- | --- |
| В1 | В2 | В3 |
| К1 | Точность прогнозирования | 0 | 1 | 0,3 |
| К2 | Производительность | 0,7 | 0,3 | 1 |
| К3 | Интеграция и совместимость | 0 | 0,5 | 1 |
| К4 | Отказоустойчивость | 0,7 | 1 | 1 |
| К5 | Интуитивно понятный интерфейс (оценка) | 1 | 0 | 0,75 |

При помощи Парето-оптимальности найдем варианты систем, которые не являются Парето-оптимальными, результат представлен в таблице 11.

Таблица 11 - Сравнение вариантов на Парето-оптимальность систем

| Вариант системы | Вариант системы | | |
| --- | --- | --- | --- |
| В1 | В2 | В3 |
| В1 | 0 | 0 | 0 |
| В2 | 0 | 0 | 0 |
| В3 | 0 | 0 | 0 |
| Результат сравнения | 0 | 0 | 0 |
| Парето-оптимальность | Да | Да | Да |

Определим весовые коэффициенты критериев, используя метод балльной оценки. Для этого определим величину α, рассчитав значения весовых коэффициентов по формуле 2.

, (2)

где *п* – общее количество весовых коэффициентов;

*ai* – весовые коэффициенты.x

Подсчитаем значения локальных критериев для каждой программы с помощью метода взвешенной суммы, используя формулу 3 и представим результат в таблице 12.

, (3)

где *kij* –коэффициент нормализации;

*ai* – весовые коэффициенты.

Таблица 12 - Значения локальных критериев методом взвешенной суммы

| Код критерия | Коэффициент важности локального критерия () | Значения локальных критериев | | |
| --- | --- | --- | --- | --- |
| В1 | В2 | В3 |
| K1 | 0,35 | 0 | 0,35 | 0,105 |
| K2 | 0,22 | 0,154 | 0,066 | 0,22 |
| K3 | 0,22 | 0 | 0,11 | 0,22 |
| K4 | 0,14 | 0,098 | 0,14 | 0,14 |
| K5 | 0,07 | 0,07 | 0 | 0,0525 |
|  | | 0,322 | 0,666 | 0,7375 |

Получив результат, можно сделать вывод, что В3 (система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки) является самым оптимальным вариантом. Поэтому эта система является актуальной и имеет преимущества перед аналогами.

1.1.5 Требования предъявляемые к разрабатываемой системе

#### 1.1.5.1 Функциональные требования

Так как разрабатываемая система основном предназначена для операторов, следящих за нагрузкой томов СХД, то функциональные требования к системе такие:

* указание параметров СХД;
* возможность выбирать таблицу из БД во frontend;
* возможность изменение уровня прогнозирования из БД во frontend;
* выбор режима скользящего окна (автоматический или ручной);
* при ручном вводе указывать интервал и его количество;
* выбор уровня предсказания;
* выбор альтернативного варианта прогнозирования;
* выбор нахождения глобального минимума, для предотвращения ошибок, при необходимости;
* выбор построения облака точек по необходимости;
* визуализация графика.

#### 1.1.5.2 Требования к надежности

При возникновении критических ошибок программа должна сохранять свою работоспособность, а также рабочие файлы для последующего восстановления.

#### 1.1.5.3 Требования к составу технических средств

Операционная система, которая поддерживает интерпретатор Python, NodeJS и postrgres: Linux, macOS, или Windows. Интерпретатор языка программирования Python: Python 3.10. Установленные библиотеки numpy, pandas, tenacity, requests, matplotlib, plotly, scipy, scikit-learn, fastapi; также нужен NodeJS с установленным фреймворком React и библиотеками: plotly, axios, styled-component, react-spinners; и БД postgres.

2 ИССЛЕДОВАТЕЛЬСКАЯ ЧАСТЬ

### 2.1 Исследование моделей прогнозирования

#### 2.1.1 Постановка задачи

Заказчик заранее выдвинул 3 модели для прогнозирования временных рядов: Random Forest, SVM и линейная регрессия. Задача состояла в том, чтобы исследовать какая модель без особой настройки больше подходит для системы. Также заказчиком были выданы исторические данные, чтобы провести исследование. В пункте 3.1.3 показаны примеры данных и описаны параметры.

#### 2.1.2 Random Forest

Random Forest – это мощный ансамблевый метод машинного обучения, используемый для задач классификации и регрессии. Он состоит из множества отдельных деревьев решений, которые работают совместно, чтобы улучшить общую производительность модели. Основная идея метода случайных лесов заключается в построении большого числа деревьев решений на случайных подмножествах данных и признаков, а затем усреднении (или голосовании) результатов этих деревьев для получения окончательного прогноза.

Преимущества метода случайных лесов:

* высокая точность: Благодаря использованию множества деревьев,

случайные леса часто достигают высокой точности на тестовых данных;

* стабильность и устойчивость к переобучению: Путем усреднения

результатов множества деревьев метод случайных лесов снижает риск переобучения, присущий отдельным деревьям решений;

* работа с большими наборами данных и большим числом признаков:

Метод случайных лесов хорошо масштабируется и может обрабатывать большие объемы данных с многочисленными признаками;

* интерпретируемость: Хотя отдельные деревья могут быть

сложными для интерпретации, важность признаков можно оценить на основе их вклада в разделение узлов.

Ограничения метода случайных лесов:

* временная и вычислительная сложность: Обучение множества

деревьев и выполнение прогнозов могут быть вычислительно затратными, особенно для очень больших наборов данных;

* память: метод требует значительных объемов памяти для хранения

множества деревьев.

#### 2.1.3 Support vector machine (SVM)

Метод опорных векторов (Support Vector Machine, SVM) — это мощный алгоритм машинного обучения, который широко используется для задач классификации и регрессии. SVM стремится найти оптимальную гиперплоскость, которая наиболее эффективно разделяет данные различных классов в многомерном пространстве признаков.

Преимущества метода опорных векторов:

* высокая точность: SVM часто достигает высокой точности

классификации, особенно в случаях, когда данные линейно разделимы или могут быть разделены в более высокоразмерном пространстве;

* эффективность на данных с большим числом признаков: SVM

хорошо работает с данными, содержащими большое количество признаков;

* гибкость в выборе ядра: Возможность выбора и настройки

различных ядер позволяет SVM справляться с различными задачами классификации;

* устойчивость к переобучению: Путем настройки параметра

регуляризации SVM может избегать переобучения и хорошо обобщать данные.

Ограничения метода опорных векторов:

* выбор ядра и параметров: Эффективность SVM сильно зависит от

правильного выбора ядра и его параметров, что требует экспериментов и настройки;

* вычислительная сложность на больших наборах данных: SVM

может быть вычислительно затратным при работе с очень большими наборами данных, так как обучение требует значительных ресурсов;

* интерпретируемость: в некоторых случаях модели SVM могут быть

трудными для интерпретации, особенно при использовании нелинейных ядер.

#### 3.1.4 Линейная регрессия

Линейная регрессия — это один из наиболее простых и широко используемых методов статистического анализа и машинного обучения, применяемый для прогнозирования и установления зависимости между переменными. Она основывается на предположении, что существует линейная связь между зависимой переменной (целевая переменная) и одной или несколькими независимыми переменными (признаки).

Преимущества линейной регрессии:

* простота и интерпретируемость: Линейная регрессия проста в

понимании и интерпретации, что делает её доступной для широкого круга пользователей;

* быстрота вычислений: Модель линейной регрессии легко и быстро

обучается, даже на больших наборах данных;

* эффективность на линейных зависимостях: Линейная регрессия

эффективно работает, если существует линейная зависимость между переменными.

Ограничения линейной регрессии:

* линейность: Линейная регрессия предполагает линейную связь

между переменными, что не всегда соответствует реальным данным;

* чувствительность к выбросам: Модель может быть чувствительна к

выбросам, которые могут значительно повлиять на результаты;

* мультиколлинеарность: Высокая корреляция между независимыми

переменными может привести к нестабильным оценкам коэффициентов модели.

#### 2.1.5 Метрики оценки

Для оценки качества моделей машинного обучения и их прогнозов используют различные метрики. В данном контексте рассмотрим три основные метрики: средняя абсолютная ошибка (MAE), среднеквадратичная ошибка (MSE) и коэффициент детерминации (R²). Каждая из этих метрик имеет свои особенности и преимущества для анализа производительности моделей.

#### 2.1.5.1 MAE

Средняя абсолютная ошибка (Mean Absolute Error, MAE) измеряет среднюю величину ошибок между предсказанными и фактическими значениями. Она рассчитывается как среднее арифметическое абсолютных разностей между предсказанными и истинными значениями.

Преимущества MAE:

* прозрачность и легкость интерпретации;
* линейность: каждое отклонение имеет одинаковое влияние на итоговую

метрику, что позволяет легко понимать и интерпретировать ошибки.

Недостатки MAE:

* не учитывает направление ошибок (перепредсказание или

недопредсказание).

#### 2.1.5.2 MSE

Среднеквадратичная ошибка (Mean Squared Error, MSE) измеряет среднее квадратическое отклонение предсказанных значений от фактических. Она более чувствительна к большим ошибкам по сравнению с MAE, так как квадратичная функция значительно увеличивает вес больших ошибок.

Преимущества MSE:

* высокая чувствительность к большим ошибкам, что помогает выявить

модели с крупными отклонениями;

* полезна при настройке моделей, где необходимо минимизировать

большие ошибки.

Недостатки MSE:

* чувствительность к выбросам может исказить общую оценку модели;
* единицы измерения MSE отличаются от единиц исходных данных, что может затруднить интерпретацию.

#### 2.1.5.3 Коэффициент детерминации

Коэффициент детерминации (R²) оценивает долю дисперсии зависимой переменной, объясняемой моделью. Он показывает, насколько хорошо предсказанные значения совпадают с фактическими, и варьируется от 0 до 1. Значение R² близкое к 1 указывает на высокое качество модели.

Преимущества R²:

* Интуитивно понятная интерпретация: показывает долю объясненной

дисперсии.

* Позволяет сравнивать качество различных моделей на одном наборе

данных.

Недостатки R²:

* Может быть завышен для моделей с большим количеством параметров.
* Не всегда указывает на улучшение модели, особенно если модель

переобучена.

#### 2.1.6 Исследование

После написания кода были проведены эксперименты на исторических данных, предоставленные заказчиком. Далее рассмотрим результаты каждой модели по отдельности.

#### 2.1.6.1 Результаты эксперимента Random Forest

На рисунках 4-5 представлены результаты эксперимента Random Forest.

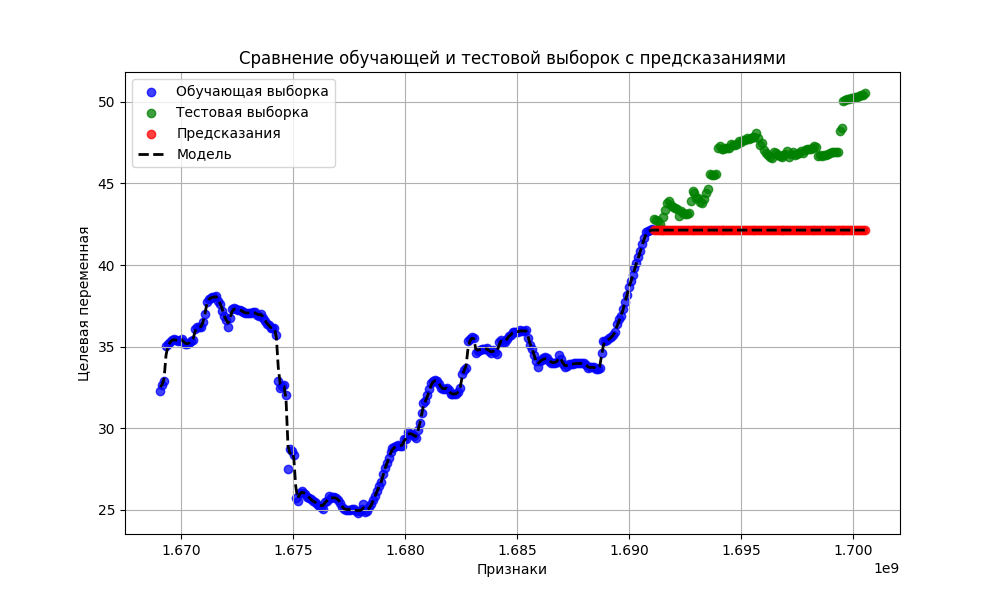


Рисунок 4 – Визуализация прогнозирования модели Random Forest

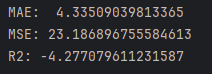


Рисунок 5 – Результаты оценивания Random Forest по метрикам

На основе полученных результатов можно сделать вывод, что модель имеет плохую производительность, то есть плохо предсказывает данные на тестовой выборке.

#### 2.1.6.2 Результаты эксперимента SVM

На рисунках 6-7 представлены результаты эксперимента SVM.

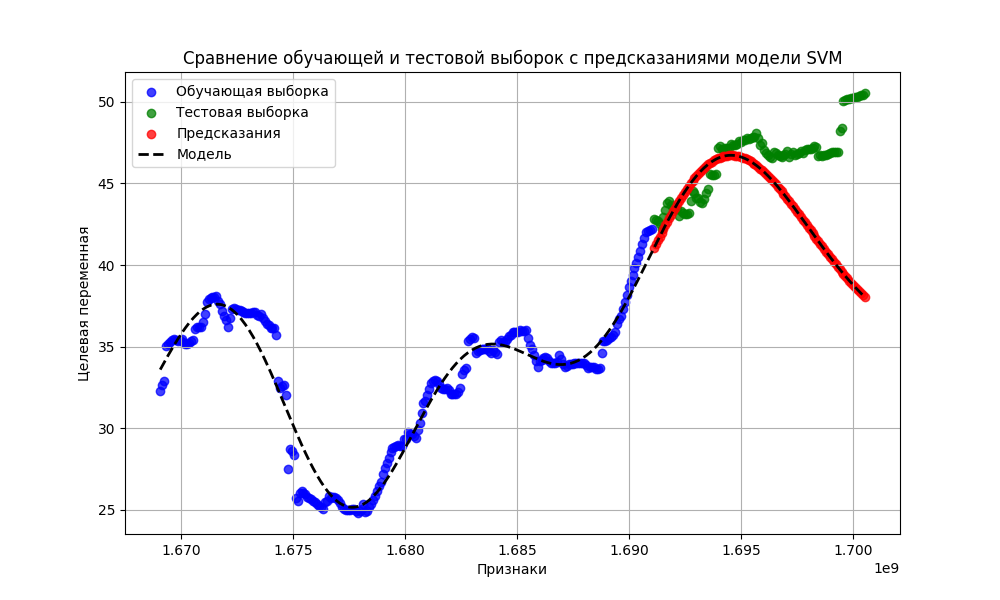


Рисунок 6 – Визуализация прогнозирования модели SVM

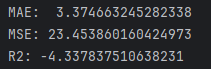


Рисунок 7 – Результаты оценивания SVM по метрикам

На основе полученных результатов можно сделать вывод, что модель справляется немного лучше, чем Random Forest, но все также имеет огромные ошибки, что видно на графике визуализации прогнозирования модели.

#### 2.1.6.3 Результаты эксперимента линейной регрессии

На рисунках 8-9 представлены результаты эксперимента линейной регрессии.

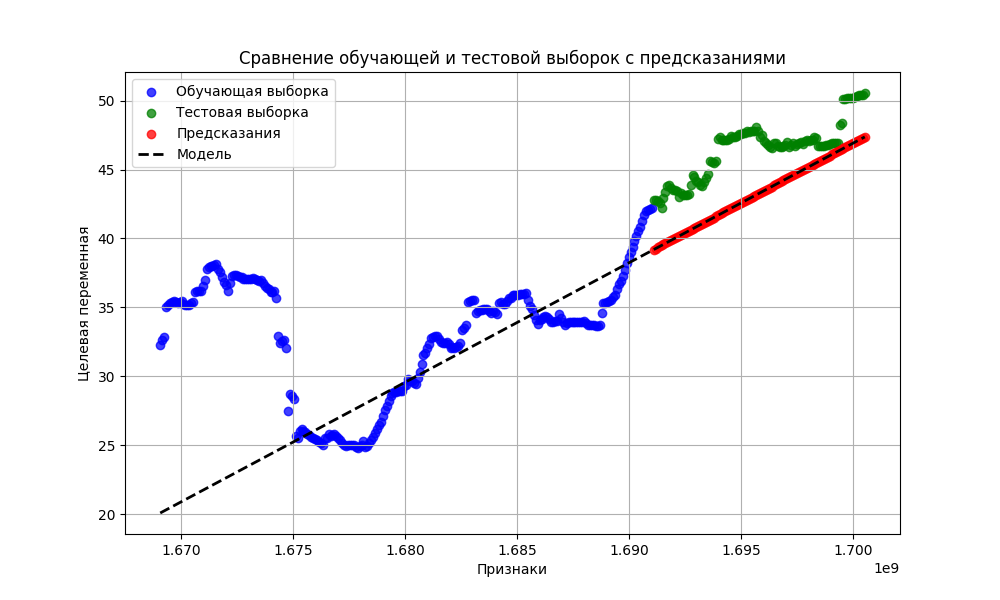


Рисунок 8 – Визуализация прогнозирования линейной регрессии

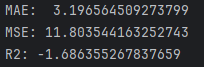


Рисунок 9 – Результаты оценивания линейной регрессии по метрикам

#### 2.1.7 Сравнение результатов

В таблице 13 мы сравним результаты по метрикам.

Пусть:

* В1 – Random Forest;
* В2 – SVM;
* В3 – линейная регрессия.

Таблица 13 – Сравнение результатов эксперимента

| Метрика | Вариант модели | | |
| --- | --- | --- | --- |
| В1 | В2 | В3 |
| MAE | 4.3 | 23.2 | -4.3 |
| MSE | 3.4 | 23.5 | -4.3 |
| R2 | 3.2 | 11.8 | -1.7 |

При помощи Парето-оптимальности найдем варианты моделей, которые не являются Парето-оптимальными, результат представлен в таблице 14. Учтем, что чем меньше метрика – тем лучше.

Таблица 14 - Сравнение вариантов на Парето-оптимальность моделей

| Вариант системы | Вариант модели | | |
| --- | --- | --- | --- |
| В1 | В2 | В3 |
| В1 | 0 | 0 | 0 |
| В2 | 0 | 0 | 0 |
| В3 | 1 | 1 | 0 |
| Результат сравнения | 1 | 1 | 0 |
| Парето-оптимальность | Нет | Нет | Да |

Получив результаты, можно сделать вывод, что линейная регрессия справляется лучше, чем Random Forest и SVM, однако также, как и они, совершает ошибки, поэтому в дальнейшем нужно производить препроцессинг данных.

3 КОНСТРУКТОРСКО-ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

3.1 Конструкторская часть

**3.1.1 Технологии фронтенда**

#### 3.1.1.1 Язык программирования JavaScript

JavaScript — это ключевой язык программирования для создания динамических веб-приложений. С момента своего создания в 1995 году он значительно эволюционировал и стал важнейшим инструментом в арсенале веб-разработчиков. JS является высокоуровневый, интерпретируемый языком программирования, предназначенный для добавления интерактивных элементов на веб-страницы. В отличие от статических языков разметки, таких как HTML и CSS, JS позволяет внедрять динамику и реагировать на действия пользователя, что делает веб-страницы более живыми и функциональными.

Основные возможности JS включают манипуляции с DOM и использование AJAX. DOM (Document Object Model) представляет собой программный интерфейс для HTML и XML-документов. JS предоставляет средства для динамического изменения структуры, стиля и содержания веб-страниц. С его помощью можно добавлять, удалять и изменять элементы, делая страницы более интерактивными и адаптивными к действиям пользователя.

Еще одна важная функция JS — это AJAX (Asynchronous JavaScript and XML), техника, позволяющая загружать данные с сервера без перезагрузки страницы. JavaScript позволяет отправлять асинхронные запросы к серверу и обрабатывать ответы, что обеспечивает динамическое обновление содержимого страницы и значительно улучшает пользовательский опыт.

#### 3.1.1.2 Фреймворк React

React — это библиотека для создания пользовательских интерфейсов, разработанная и поддерживаемая компанией Facebook. Она была впервые представлена в 2013 году и с тех пор стала одной из самых популярных технологий в веб-разработке. React позволяет создавать интерактивные и динамичные веб-приложения с использованием компонентного подхода.

React занял важное место в экосистеме фронтенд-разработки благодаря своей гибкости и производительности. Он предоставляет разработчикам мощные инструменты для создания сложных пользовательских интерфейсов с высокой степенью повторного использования кода. React широко используется в крупных проектах, таких как Facebook, Instagram и WhatsApp, что подтверждает его эффективность и надежность.

React имеет ряд преимуществ:

1. Компонентный подход – каждый компонент представляет собой

независимую и повторно используемую часть пользовательского интерфейса. Это позволяет разработчикам разбивать приложения на небольшие, управляемые блоки, которые можно легко комбинировать и тестировать;

1. Виртуальный DOM – представляет собой легковесную копию реального

DOM, которая обновляется в памяти перед тем, как изменения будут применены к реальному DOM. Этот процесс позволяет React эффективно управлять обновлениями интерфейса, уменьшая количество дорогостоящих операций с реальным DOM и обеспечивая высокую производительность и отзывчивость приложений;

1. Однонаправленный поток данных – он делает управление состоянием

приложения более предсказуемым и удобным. Данные передаются от родительских компонентов к дочерним через свойства (props), что облегчает отслеживание и отладку состояния приложения. Это также способствует лучшей организации и структуре кода;

1. Огромное сообщество и богатая экосистема: Существует множество

библиотек и инструментов, которые расширяют возможности React и облегчают разработку. Например, такие библиотека, как Redux, помогают управлять состоянием приложения. А активное сообщество разработчиков постоянно создает и поддерживает новые инструменты и библиотеки, что делает React еще более мощным и гибким.

### 3.1.2 Технологии бэкенда

#### 3.1.2.1 Язык программирования Python

Python – это высокоуровневый, интерпретируемый язык программирования общего назначения, известный своей простой и понятной синтаксисом. Он был создан Гвидо ван Россумом и впервые выпущен в 1991 году. Python поддерживает множество парадигм программирования, включая процедурное, объектно-ориентированное и функциональное программирование.

Основные преимущества Python:

1. Удобство, гибкости и скорости разработки – все это достигается

благодаря простому синтаксису и богатой стандартной библиотекой;

1. Активное и большое сообщество – python обладает одним из самых

крупных и поддерживающих сообществ разработчиков, что обеспечивает доступ к большим возможностям и понятной документации;

1. Обширная экосистема – существует огромное множество библиотек и

фрейморков для python, что позволяет охватить практически все сферы разработки.

Примеры использования Python:

— веб-разработка;

— научные исследования;

— аналитика данных;

— машинное обучение;

— игровая разработка.

#### 3.1.2.2 Фреймворк Django

Django – это высокоуровневый веб-фреймворк на языке Python, который позволяет быстро создавать надежные и масштабируемые веб-приложения. Он был разработан для упрощения процесса создания сложных и многофункциональных сайтов. Django следует философии "Не повторяй себя" (DRY, Don’t Repeat Yourself) и "Согласованность важнее конфигурации" (Convention over Configuration).

Основные характеристики Django:

— полный стек;

— быстрая разработка;

— безопасность;

— гибкость;

— активное сообщество.

Преимущества фреймворка:

1. Быстрая разработка и развертывание: Django позволяет быстро

создавать веб-приложения благодаря своему полному стеку инструментов и шаблонам, что особенно важно для стартапов и проектов с ограниченными сроками.

1. Удобство и простота: Благодаря четкой документации и интуитивно

понятному интерфейсу, Django легко изучить и использовать, что снижает порог входа для новых разработчиков.

1. Автоматизация задач: Django включает административную панель,

которая автоматически генерируется на основе моделей данных. Это упрощает управление контентом и данными, позволяя разработчикам сосредоточиться на более важных задачах.

1. Совместимость с различными базами данных: Django поддерживает

работу с различными базами данных, включая PostgreSQL, MySQL, SQLite и Oracle, что делает его гибким инструментом для разработки.

1. Расширяемость: Django легко расширять и интегрировать с другими

библиотеками и фреймворками. Существуют многочисленные плагины и расширения, которые могут быть добавлены для расширения функциональности приложения.

**3.1.3 СУБД**

PostgreSQL (часто сокращаемая как Postgres) — это мощная, открытая реляционная система управления базами данных (СУБД), которая подчеркивает расширяемость и соответствие стандартам SQL. Она была разработана для управления большими объемами данных и поддержки сложных запросов, обеспечивая высокую надежность и производительность.

Основные характеристики PostgreSQL:

1. Расширяемость: PostgreSQL позволяет разработчикам добавлять новые

функции, такие как типы данных, функции, операторы, агрегаты, индексы и языки процедур.

1. Поддержка стандартов: PostgreSQL соответствует многим стандартам

SQL, что делает его совместимым с большинством других реляционных СУБД.

1. Открытый исходный код: PostgreSQL распространяется под лицензией

PostgreSQL, которая является свободной и открытой, что позволяет использовать, изменять и распространять программное обеспечение без ограничений.

#### 3.1.3.1 Описание таблицы data

Рассмотрим таблицы, хранящиеся в БД. На рисунке 10 изображен пример таблицы data.

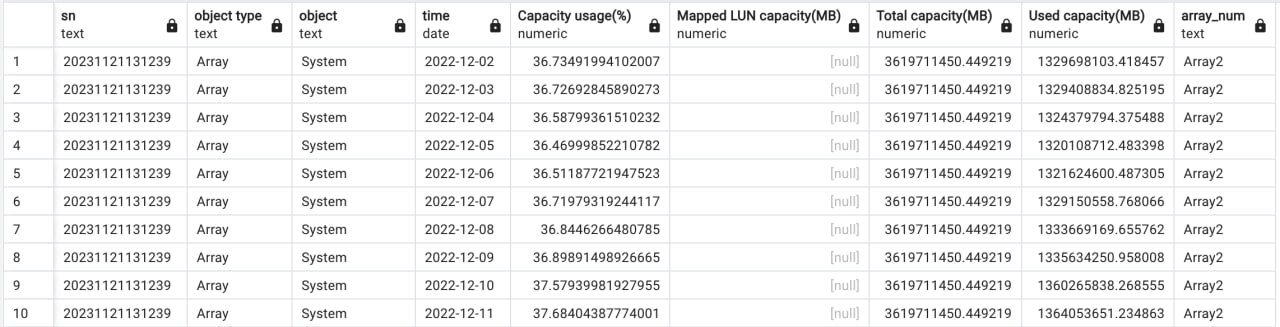


Рисунок 10 – Пример таблицы data

В таблице data хранятся исторические данные нагрузки томов СХД, распишем каждый параметр:

* sn – серийный номер хранилища данных;
* object type – тип объекта;
* object – название объекта;
* time – время записи данных;
* Array\_num – название системы хранения.

Опишем пропущенные параметры. В СХД Huawei Dorado выгружаются метрики заполнения дискового пространства в виде временных рядов в БД.

Объекты типа Array обозначают отдельную систему хранения. Предполагается, что в одной кластере хранения могут быть объединены несколько систем хранения. Уникальным идентификатором объектов этого типа является поле Object, и оно содержит название объекта, которое задается вручную администратором системы хранения данных. Внутри одной СХД не может быть 2 объекта типа Array с одинаковым значением поля Object. В двух разных СХД могут быть 2 объекта типа Array с одинаковым значением поля Object.

Объекты типа Array имеют 3 поля с параметрами:

1. Total Capacity - совокупная емкость всех созданных StoragePool в

данной системе хранения. Измеряется в Mb. Предполагается, что этот параметр обозначает все доступное дисковое пространство соответствующей СХД. Значение этого параметра может быть увеличено путем физического добавления новых накопителей к СХД (array upgrade). Значение этого параметра не может быть уменьшено.

1. Used Capacity - совокупная емкость всех данных, физически записанных

на накопители в рамках этой системы хранения. Измеряется в Mb. Значение этого параметра складывается из физического объема данных, записанных в LUN, FS и мгновенных снимках этих LUN и FS.  
Значение этого параметра не может превышать Total Capacity. Значение этого параметра может увеличиваться или уменьшаться. Если Used Capacity по значению приближается к Total Capacity, то это означает о скором окончании свободного дискового пространства. При полном соответствии значение Total и

1. Used Capacity наступает критическая неисправность СХД – нехватка

свободного дискового пространства, которая приводит к простою всех сервисов и может привести к недоступности данных, уже записанных на СХД.

1. Capacity Usage - процентное соотношение Used Capacity к Total

Capacity.

Объекты типа Storage Pool обозначают дисковые пулы, на базе которых создаются блочные тома (LUN), файловые системы (FS) и их служебные данные (мета, мгновенные снимки). Объект типа Array может содержать более одного Storage Pool либо не содержать ни одного. Объект Storage Pool всегда существует в рамках одного Array, и не может существовать без привязки к Array.

Уникальным идентификатором объектов этого типа является поле Object, и оно содержит название объекта, которое задается вручную администратором системы хранения данных. Внутри одной СХД не может быть 2 объекта типа Array с одинаковым значением поля Object. В двух разных СХД могут быть 2 объекта типа Array с одинаковым значением поля Object.

Объекты типа Storage Pool имеют 4 поля с параметрами:

1. Total Capacity - полезная емкость данного дискового пула. Измеряется

в Mb. Вычисляется из количества дисков с учетом коэффициента RAID и количества хранимых метаданных. Значение этого параметра может быть увеличено путем добавления новых накопителей (storage pool expand) из числа неиспользуемых. Значение этого параметра не может быть уменьшено.

1. Used Capacity - совокупная емкость всех данных, физически записанных

на накопители в рамках этого дискового пула. Измеряется в Mb. Значение этого параметра складывается из физического объема данных, записанных в LUN, FS и их служебных данных (мгновенных снимках и метаданных).  
Значение этого параметра не может превышать Total Capacity. Значение этого параметра может увеличиваться или уменьшаться. Если Used Capacity по значению приближается к Total Capacity, то это означает о скором окончании свободного дискового пространства в Storage Pool’е. При полном соответствии значение Total и Used Capacity все созданные блочные тома и файловые системы блокируются для записи, но данные на них доступны для чтения.

1. Capacity Usage - процентное соотношение Used Capacity к Total

Capacity.

1. Mapped LUN capacity - совокупная емкость всех блочных томов,

которые были презентованы хостам. Измеряется в Мб.

В СХД Huawei Dorado используются тонкие тома (Thin LUN), которые занимают дисковое пространство только по мере записи физических данных. Пример: в момент создания блочный том размером 100Тб занимает 0 Мб дискового пространства. После его презентации хосту и записи на него 10 Гб он начинает занимать соответствующие 10 Гб дискового пространства.  
Значение Mapped LUN capacity может увеличиваться или уменьшаться. Значение этого параметра может превышать значение Total Capacity.

#### 3.1.3.2 Описание таблицы Level

Пример данных, хранящихся в таблице Level, представлен на рисунке 11.

Объясним каждый параметр:

* LEVEL0 – уровень прогнозирования для 0 уровня;
* LEVEL1 – уровень прогнозирования для 1 уровня;
* LEVEL2 – уровень прогнозирования для 2 уровня;
* object – название системы хранения для каждого уровня прогнозирования.

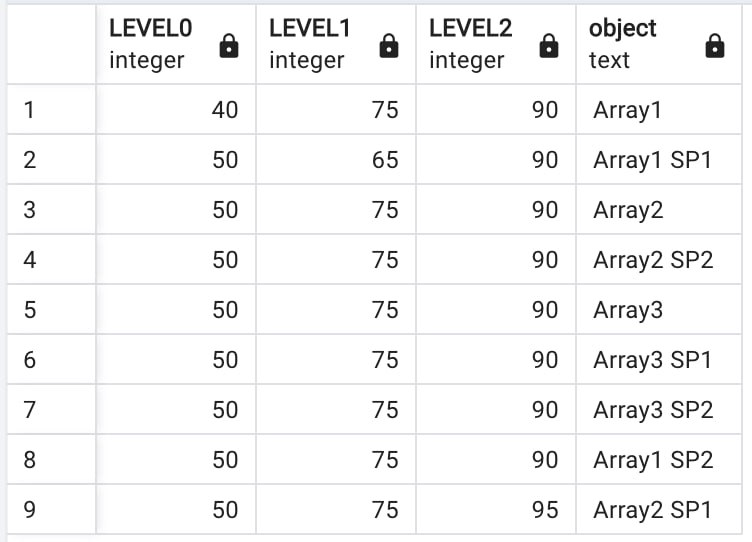


Рисунок 11 – Пример таблицы Level

### 3.1.4 Модель прогнозирования

Линейная регрессия – это метод статистического анализа, который используется для моделирования и анализа зависимостей между переменными. В линейной регрессии предполагается, что зависимость между зависимой переменной (или целевой переменной) и одной или несколькими независимыми переменными (предикторами) можно описать линейной функцией.

Основная цель линейной регрессии — найти такие параметры линейной функции (коэффициенты), которые минимизируют разницу между предсказанными значениями и фактическими значениями зависимой переменной.

Для построения модели линейной регрессии для предсказания временных рядов нужно учитывать, что временные ряды имеют временную зависимость, то есть текущее значение ряда зависит от предыдущих значений. Один из распространённых подходов заключается в использовании метода автокорреляции и лагов. Однако в нашем случае использовались скользящие окна, так как нужно было строить облака точек. Скользящее окно – это метод, используемый для анализа временных рядов и обработки данных, при котором анализируются подмножества данных (окна) определенной длины.

В системе мы можем выбрать один из двух режимом скользящего окна:

— автоматический;

— ручной (продвинутый).

Они отличаются тем, что в автоматическом режиме при помощи метода пиков, размер скользящего окна подбирается самостоятельно и постоянно меняется. В ручном (продвинутом) же мы задаем размер окна вручную, указывая интервал и его количество.

### 3.1.5 Архитектура системы

Архитектура системы прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки состоит из фронтенд части, реализованная на React JS, бэкенд на Python Django и БД на Postgres. Рассмотрим диаграмму развертывания на рисунке 12.

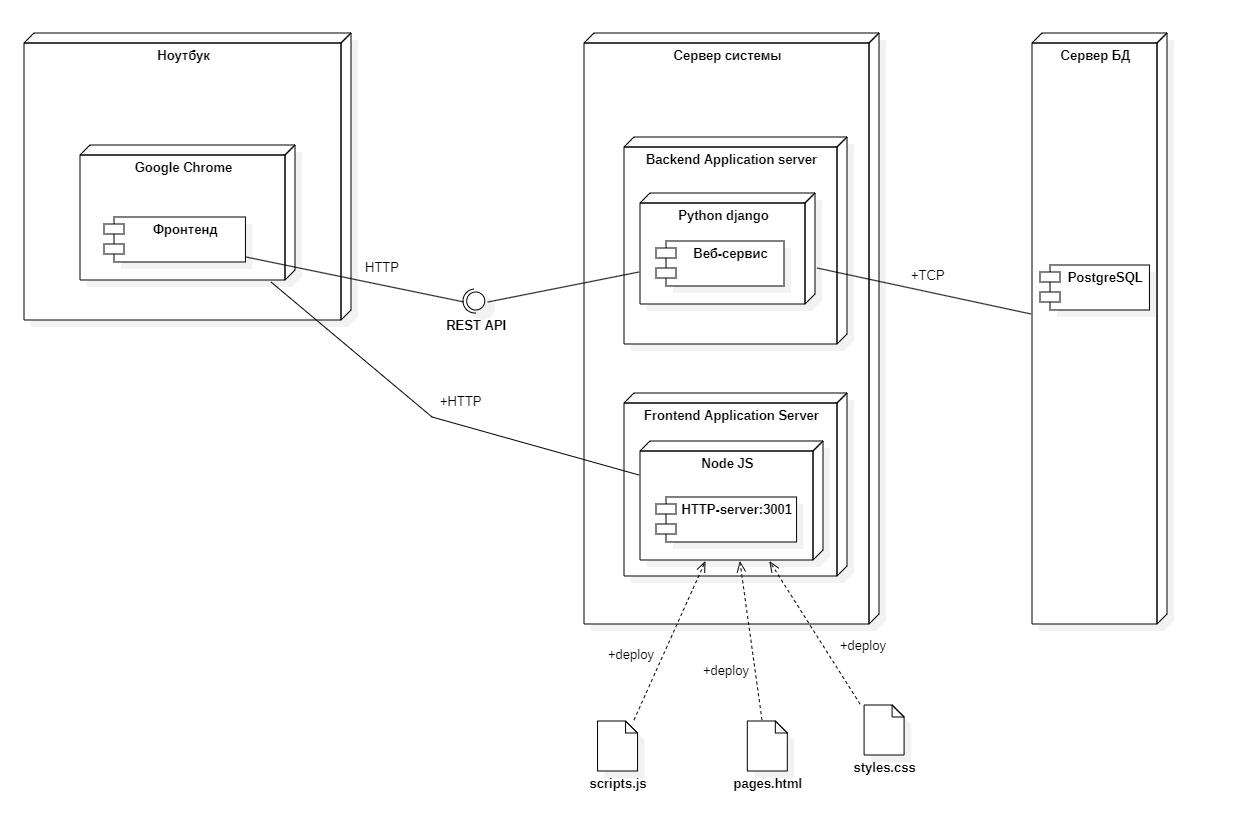


Рисунок 12 – Диаграмма развертывания

Диаграмма развёртывания изображает взаимодействие компонентов веб-приложения, включающего фронтенд, бэкенд и базу данных. Вот её описание и взаимодействие компонентов:

1. Ноутбук (Клиентская сторона):

— на клиентской стороне запущен веб-браузер (Google Chrome), в котором работает фронтенд-приложение;

— фронтенд-приложение обращается к REST API, используя HTTP-запросы.

1. Сервер системы:

Содержит два сервера приложений: Backend Application Server и Frontend Application Server.

Backend Application Server:

* использует Python Django для обработки веб-сервисов;

— обрабатывает запросы от фронтенда через REST API;

— взаимодействует с базой данных PostgreSQL по протоколу TCP.

Frontend Application Server:

— использует Node.js;

— запускает HTTP-сервер на порту 3001;

— деплой статических файлов (scripts.js, pages.html, styles.css) происходит на этот сервер.

1. Сервер базы данных (Сервер БД):

— хранит данные в PostgreSQL;

— получает запросы от Backend Application Server по протоколу TCP.

Взаимодействие компонентов:

— фронтенд-приложение в браузере на ноутбуке отправляет HTTP-запросы к REST API на Backend Application Server;

— Backend Application Server обрабатывает эти запросы, выполняет логику приложения и взаимодействует с базой данных PostgreSQL для получения данных при помощи Django ORM;

— Frontend Application Server обслуживает статические файлы (скрипты, HTML-страницы и стили CSS), которые могут использоваться фронтенд-приложением для обновления контента или интерфейса.

Таким образом, все компоненты работают совместно, обеспечивая функциональность веб-приложения, включая взаимодействие между клиентом, сервером приложений и базой данных.

Термины, которые стоит разобрать:

1. HTTP-запрос – это строка с информацией, которую отправляет клиент на

сервер, для выполнения определенного действия, такого как получение веб-страницы или отправка данных на сервер.

1. TCP – это один из основных протоколов интернета, относящийся к

транспортному уровню. TCP обеспечивает надёжную, упорядоченную и проверенную доставку потока данных между приложениями, работающими на узлах сети.

1. REST API – это архитектурный стиль, интерфейс, позволяющий

приложениям взаимодействовать с сервером через стандартные HTTP-запросы

1. Django ORM – это инструмент, встроенный в Django, которые позволяет

разработчикам работать с БД, используя ООП, разработчики могут взаимодействовать с базой данных с помощью Python-объектов и методов, вместо написания сырых SQL-запросов.

### 3.1.6 Пользовательский интерфейс

Начальный экран представлен на рисунке 13.

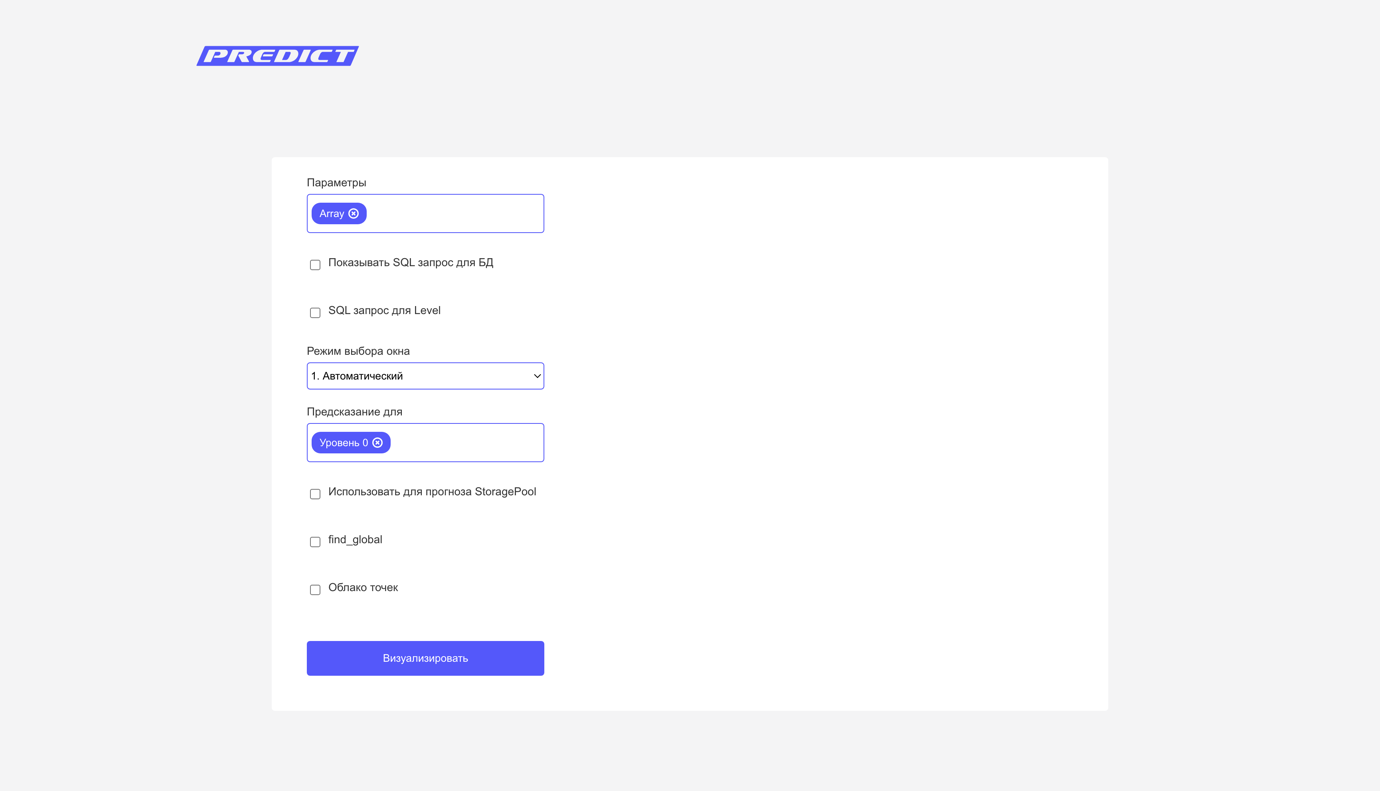


Рисунок 13 – Начальный экран

На рисунке 14 представлено поле ввода параметров. Параметры – указываются array, storagepool1 и storagepool2. Где array – это система хранения, а storagepool – это дисковые пулы в array.



Рисунок 14 – поле ввода параметров

На рисунке 15 показан чекбокс «Показать SQL запрос для БД», при нажатии на который появляется поле ввода SQL запроса для БД.

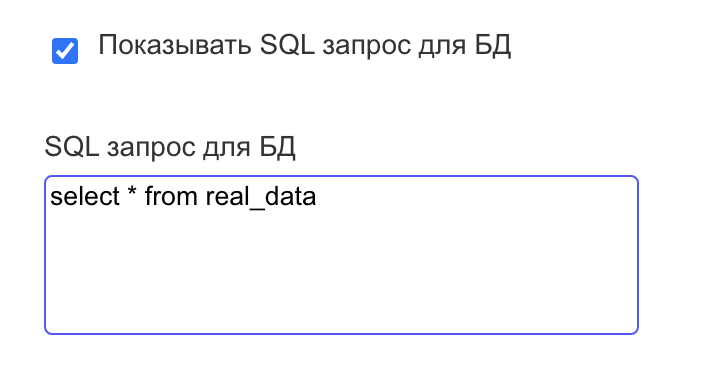


Рисунок 15 – поле ввода SQL запроса для БД

На рисунке 16 показан чекбокс «SQL запрос для Level», при нажатии на который появляется окно ввода SQL запрос для таблицы Level.

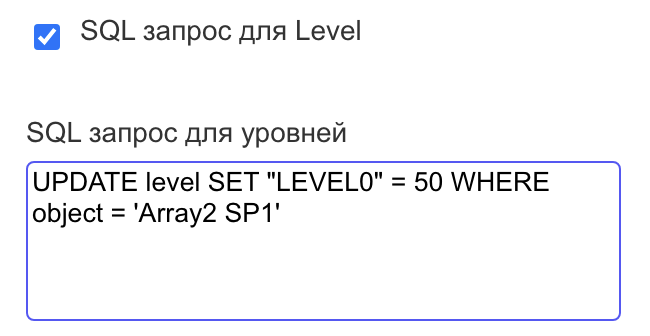


Рисунок 16 – поле ввода SQL запроса для таблицы Level

Далее на рисунке 17 представлен режим выбора окна. Как было написано ранее, скользящее окно имеет 2 режима: автоматический и ручной.

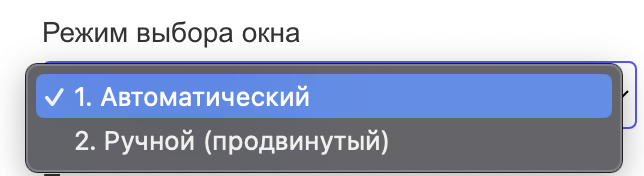


Рисунок 17 – Режим выбора окна

На рисунке 18 изображен select, в котором мы можем выбрать уровень прогнозирования.

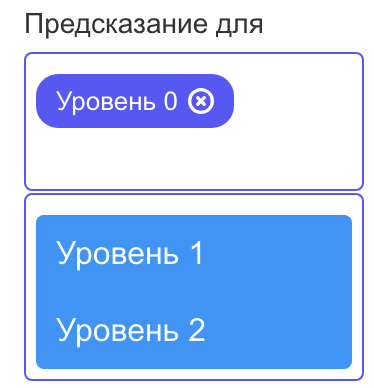


Рисунок 18 – выбор уровня прогнозирования

На рисунке 19 изображены чекбоксы: «Использовать для прогноза StoragePool», «find\_global» и «Облако точек». Чекбокс «Использовать для прогноза StoragePool» был добавлен по запросу заказчика. Эта опция позволяет учесть в прогнозе Array нагрузку на 100%, если хотя бы один дисковый пул заполнен. Чекбокс «find\_global» становится доступным только в автоматическом режиме скользящего окна. Этот метод напрямую влияет на размер окна, поскольку он определяет глобальный минимум, от которого строится прогноз. Необходимость данного метода возникает, когда исторические данные демонстрируют значительное падение на всем участке. Чекбокс «Облако точек» предназначен для построения облака точек, отображающего все предсказания при использовании скользящего окна.

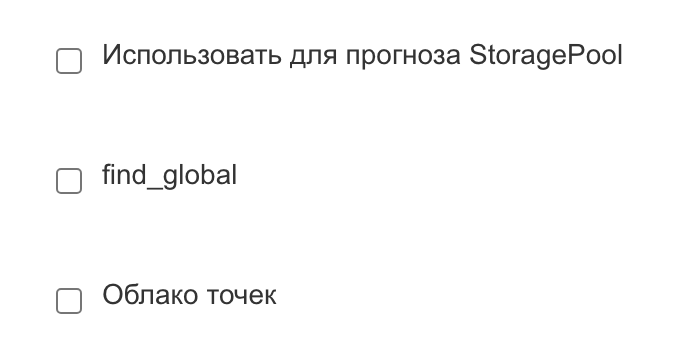


Рисунок 19 – Чекбоксы «Использовать для прогноза StoragePool», «find\_global» и «Облако точек»

Далее при нажатии на кнопку «визуализировать» появляется график с историческими данными, прогнозом, уровнями прогнозирования и возможно облако точек при нажатии на соответствующий чекбокс. На рисунке 20 представлен вариант, когда без изменений параметров в начальном окне нажата кнопка визуализации.

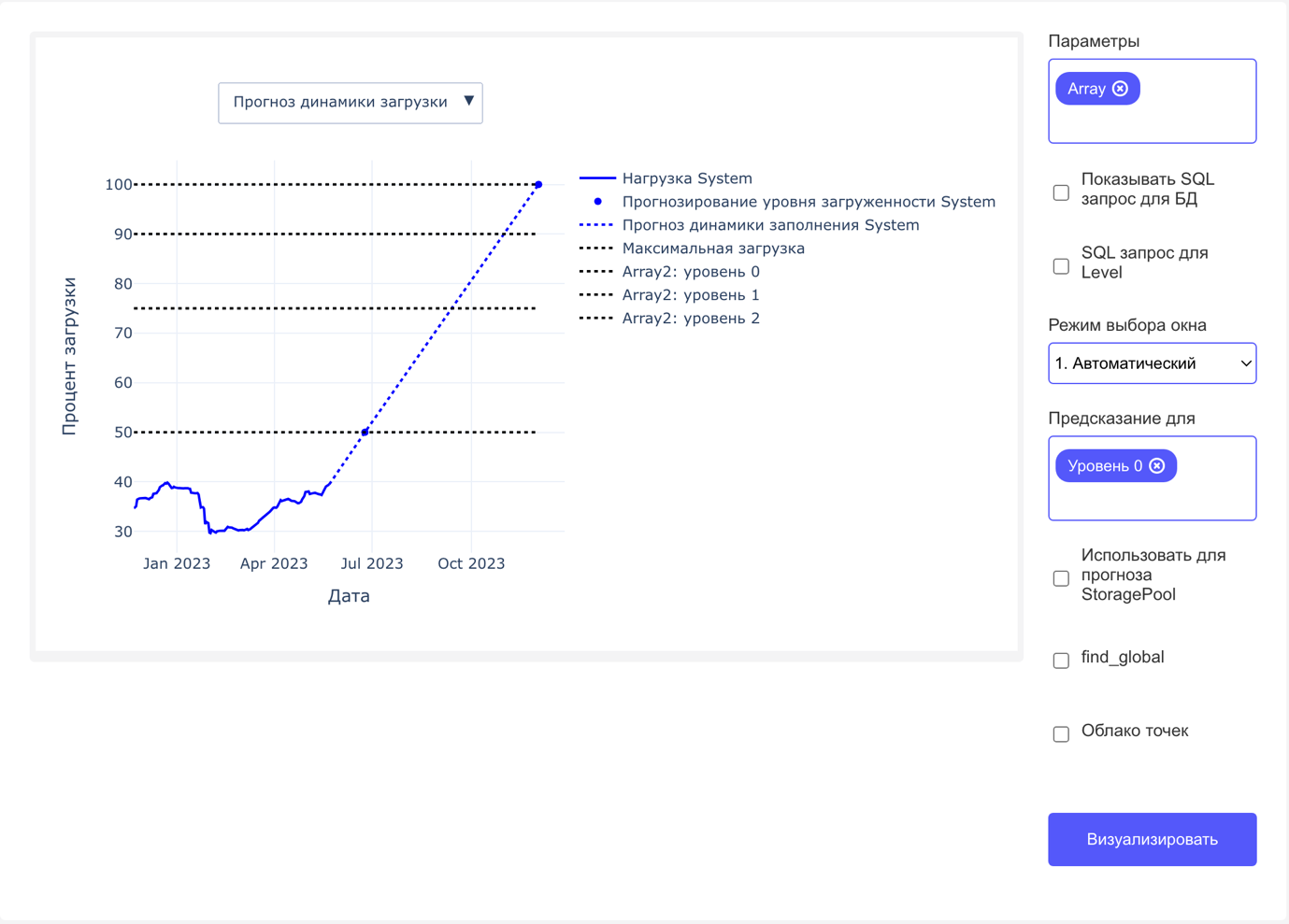


Рисунок 20 – Экран после нажатия на кнопку «визуализировать»

В приложении А находятся рисунки, на которых изображены результаты изменения параметров.

ЗАКЛЮЧЕНИЕ

При разработке системы прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки были получены следующие результаты:

* Была изучена предметная область, связанная с системами хранения

данных и методами прогнозирования их нагрузки. Проведен анализ существующих решений и технологий, включая использование алгоритмов машинного обучения для прогнозирования и оптимизации нагрузки.

* Были сформулированы требования к разрабатываемой системе,

включающие необходимость высокой точности прогнозирования, производительности, интеграции с существующими системами, отказоустойчивости и удобства использования.

* Спроектирована архитектура системы, включающая фронтенд на React,

бэкенд на Python/Django и базу данных на PostgreSQL. Определены ключевые компоненты системы, такие как модули сбора данных, модуль прогнозирования и интеллектуальная подсистема настройки.

* Разработана система прогнозирования нагрузки СХД, включающая

алгоритмы машинного обучения для анализа данных и прогнозирования, а также интерфейс для настройки параметров и визуализации результатов. Проведены тестирования, подтвердившие высокую точность и надежность системы.

Полученное информационно-программное изделие обладает возможностями расширения за счет интеграции дополнительных платформ и алгоритмов. Это позволит системе адаптироваться к новым условиям и сохранять конкурентоспособность на рынке.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Конституция Российской Федерации: офиц. Текст.- М.: ЭКСМО, 2019. – 36с.
2. Федеральный закон «О полиции» от 07.02.2011 N 3-ФЗ (ред. от 03.05.2024), [Электронный ресурс].–URL:http://www.consultant.ru, Дата обращения 03.06.2024.
3. Григорьев Ю. А., Ревунков Г. И. Банки данных: Учеб. для вузов. – М.: Изд-во МГТУ им. Н. Э. Баумана, 2002. – 320 с.
4. Евсеев А. В., Мышенков К. С. Проектирование информационных систем: Учебное пособие. – М.: Изд. комплекс МГУПП, 2006. – 190 с.
5. Липаев В. В. Системное проектирование сложных программных средств для информационных систем. – М.: СИНТЕГ, 2002. - 224 с.
6. Официальный сайт СУБД PostgreSQL [Электронный ресурс] // postgresql.org URL: https://www.postgresql.org (дата обращения: 10.04.2019).
7. Электронная библиотека. [Электронный ресурс] – URL: http://www.zipsistes.ru/ (дата обращения: 02.06.2024).
8. Методические рекомендации по подготовке и защите выпускной квалификационной работы бакалавра. / Кротов Ю.Н. [Электронный ресурс] – URL: https://drive.google.com/file/d/1pEcfTr3xDdJ81Hxz2F6GcbtNV1n3dan6/view. (дата обращения: 25.01.2023).
9. Официальный сайт System Reporter overwiew [Электронный ресурс] // HPE

URL: <https://www.hpe.com/psnow/resources/ebooks/a00114824en_us_v2/sr_overview.html> (дата обращения 04.05.2024).

1. System Reporter threshold alerts conscept [Электронный ресурс] // techhub

URL: <https://techhub.hpe.com/eginfolib/storage/docs/3PARStorStoreServStorage/3PARStorServManagementConsole/SSMC3_7_OLH/_help_start.html#general/toc-s-system-reporter-alert-concepts.html> (дата обращения 04.05.2024).

1. СХД (Система хранения данных) / itglobal [Электронный ресурс] – URL:

<https://itglobal.com/ru-ru/company/glossary/shd-sistema-hraneniya-dannyh/> (Дата обращения 04.05.2024).

1. Сеть хранения данных / Wikipedia [Электронный ресурс] – URL:

<https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D1%8C_%D1%85%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85> (дата обращения 05.05.2024).

Порядок оформления списка использованных источников

|  |  |
| --- | --- |
| **Вид источника** | **Форма описания** |
| **Журнальные статьи** | Автор. Статья / Авторы // Журнал. – Год. – Номер. – Страницы размещения статьи.  Если над статьей работало более 4 человек, то в заглавии один из них не упоминается. |
| **Монографии** | Автор. Название. / Авторы – Номер. – Город и издательство, год выпуска. – Страницы, на которых размещена работа.  Разрешается не использовать знаки тире при оформлении данного описания, а обходиться лишь точками для разделения отдельных частей.  Если при написании использовались труды других авторов, то их можно упомянуть в общем перечислении, либо дописать в квадратных скобках в качестве отдельной части. |
| **Авторефераты** | Автор. Название работы: (регалии автора). – Город, год издания. – Количество страниц. |
| **Диссертации** | Автор. Название: (после двоеточия можно указать статус работы и регалии автора). – Город, год издательства. – Страницы, на которых размещена работа или общее количество страницы. |
| **Обзоры (аналитика)** | Название / Автор. – Город: Издательство, год выпуска. – Количество страниц. |
| **Патенты** | Патент РФ Номер, дата выпуска |
| Авторы. Название // Патент России Номер, год. Номер бюллетеня. |
| **Материалы конференций** | Название. Тема конференции, Город, год выпуска. Количество страниц. |
| Автор. Название // Тема конференции (Место и дата проведения) – Город, год выпуска. – Страницы, на которых напечатана работа, либо их количество. |
| **Интернет-документы** | URL, дата обращения к ресурсу. |
| Название работы / Автор. URL (дата обращения по ссылке). |
| **Учебники** | Автор. Название / Авторы. – Город: Издательство, год выпуска. – Количество страниц.  При авторстве 4-х и более человек оформление производится аналогично журнальным статьям. |
| **Учебные пособия** | Название / (Авторы работ) // Редактор. – Город: Издательство, год выпуска. – Количество страниц. |
| **Словари** | Автор. Название / Авторы. – Город: Издательство, год выпуска. – Количество страниц. |

ПРИЛОЖЕНИЕ А ГРАФИЧЕСКАЯ ЧАСТЬ

В графическую часть выпускной квалификационной работы входят:

А.1. Общая архитектура изделия.

А.2. Схема предметной области.

А.3. Талица сравнения с аналогами.

А.4. Архитектура системы.

А.5. Инфологическая модель.

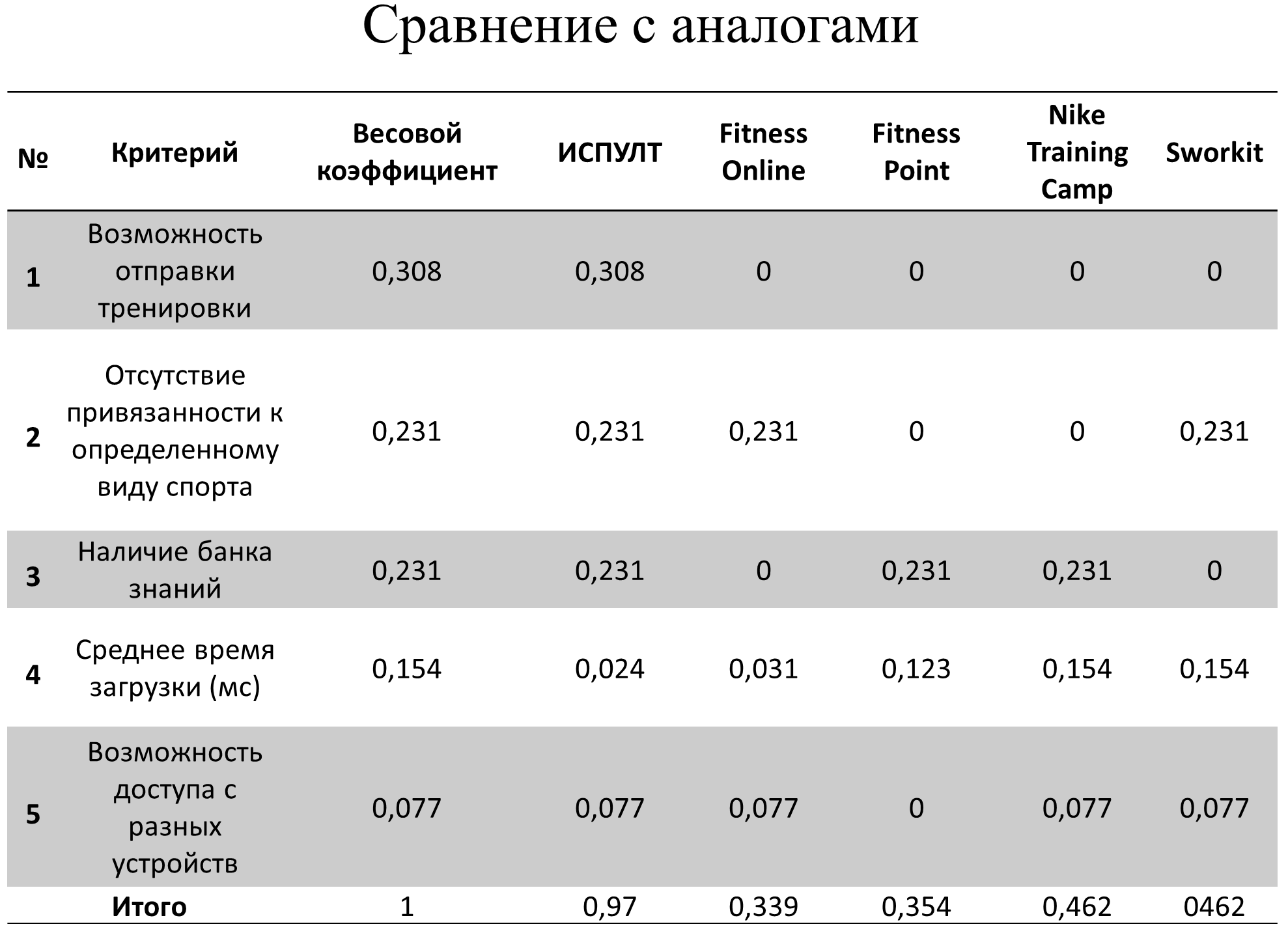
А.6. Даталогическая модель.

А.7. Пользовательские формы отображения информации.

А.8. Пользовательские формы ввода.

А.9. Экранные формы системы.

А.3 Таблица сравнения с аналогами

****

ПРИЛОЖЕНИЕ В ТЕХНИЧЕСКОЕ ЗАДАНИЕ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

|  |  |  |
| --- | --- | --- |
| Утверждаю  Заведующий кафедрой ИУ-5 |  | Согласовано  научный руководитель |
| \_\_\_\_\_\_\_\_\_\_\_В.И.Терехов  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |  | \_\_\_\_\_\_\_\_\_\_А.А. Максаков  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |

**Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки**

Техническое задание

(вид документа)

писчая бумага

(вид носителя)

5

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Ким Алексей Максимович |
| "\_\_"\_\_\_\_\_\_\_\_\_\_\_2024 г. |  |

Москва – 2024

1. **Ведение**

Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки. Будет создано 3 контейнера для работы системы – Frontend, BackEnd и БД postrgres.

1. **Основание для разработки**

Основанием для разработки является задание на выпускную квалификационную работу, подписанное руководителем выпускной работы и утверждённое заведующим кафедрой ИУ5 МГТУ им. Н.Э. Баумана 15 декабря 2023 года.

1. **Назначение и цель разработки**

Разрабатываемая система предназначена для прогнозирования перегрузки СХД (система хранилища данных), для предотвращения ошибок и повышение эффективности работы СХД.

1. **Требования к программе или программному изделию**

4.1 Требования к функциональным характеристикам:

Программа должна выполнять следующие функции:

4.1.1 Указание параметров СХД

4.1.2 Возможность выбирать таблицу из БД во frontend

4.1.3 Возможность изменение уровня прогнозирования из БД во frontend

4.1.4 Выбор режима скользящего окна (автоматический или ручной)

4.1.5 При ручном вводе указывать интервал и его количество

4.1.6 Выбор уровня предсказания

4.1.7 Выбор альтернативного варианта прогнозирования

4.1.8 Выбор нахождения глобального минимума, для предотвращения ошибок, при необходимости

* + 1. Выбор построения облака точек по необходимости
    2. Визуализировать график

4.2 Требования к надежности

При возникновении критических ошибок программа должна сохранять свою работоспособность, а также рабочие файлы для последующего восстановления.

4.3 Требования к составу технических средств

Операционная система, которая поддерживает интерпретатор Python , NodeJS и postrgres: Linux, macOS, или Windows. Интерпретатор языка программирования Python: Python 3.10. Установленные библиотеки numpy, pandas, tenacity, requests, matplotlib, plotly, scipy, scikit-learn, fastapi; также нужен NodeJS с установленным фреймворком React и библиотеками: plotly, axios, styled-component, react-spinners; и БД postgres.

**5. Требования к программной документации**

Для представления заказчику разрабатываются следующие документы:

1. Техническое задание.

2. Программа и методика испытаний.

3. Руководство пользователя – c описанием всех действий, которые пользователь может произвести, и реакцию системы на эти действия; порядок действий пользователя при зависании или сбое программы.

4. Расчётно-пояснительная записка.

**6. Технико-экономические показатели**

Требования к данному разделу не предъявляются

**7. Стадии и этапы разработки**

7.1 График приведен в соответствии с учебными неделями МГТУ им. Н.Э. Баумана 4 курса 8 семестра.

|  |  |  |
| --- | --- | --- |
| **№ п/п** | **Наименование этапов выпускной квалификационной работы** | **Сроки выполнения этапов** |
| 1. | Задание на выполнение работы. Формулирование проблемы, цели и задач работы. | *26.01.2024*  *Планируемая дата* |
| 2. | Разработка и формулирование ТЗ | *20.03.2024*  *Планируемая дата* |
| 3. | Разработка программы и методики испытания | *20.04.2024*  *Планируемая дата* |
| 4. | Разработка второй части РПЗ “Конструкторско-технологическая часть” | *27.04.2024*  *Планируемая дата* |
| 5. | Защита макета программы (АСОиУ) | *12.05.2024*  *Планируемая дата* |
| 6. | Разработка заключения, приложений, оформление работы | *23.05.2024*  *Планируемая дата* |
| 7. | Подготовка доклада и презентации. Предзащита | *25.05.2024*  *Планируемая дата* |
| 8. | Получение заключения научного руководителя | *28.05.2024*  *Планируемая дата* |
| 9. | Допуск работы к защите на ГЭК (нормоконтроль) | *30.05.2024*  *Планируемая дата* |
| 10. | Защита работы на ГЭК | *01.06.2024*  *Планируемая дата* |

**8. Порядок контроля и приёмки**

Приём программного изделия в виде тестовых испытаний осуществляется в ходе «Защиты макетов программ – предварительной защиты ВКРБ» в период с 15 по 24 мая 2024 года в соответствие с разработанной программой и методикой испытаний.

На испытание представляются: рабочая программа, результаты выполнения и код данной программы.

ПРИЛОЖЕНИЕ В ПРОГРАММА И МЕТОДИКА ИСПЫТАНИЙ

МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

им. Н.Э. Баумана

Кафедра «Системы обработки информации и управления»

|  |  |  |  |
| --- | --- | --- | --- |
|  | Утверждаю  Заведующий кафедрой ИУ-5 |  | Согласовано  научный руководитель |
|  | \_\_\_\_\_\_\_\_\_\_\_ В.И.Терехов  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |  | \_\_\_\_\_\_\_\_\_\_ А.А. Максаков  "\_\_"\_\_\_\_\_\_\_\_\_\_2024 г. |

**Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки**

Программа и методика испытаний

(вид документа)

писчая бумага

(вид носителя)

5

(количество листов)

ИСПОЛНИТЕЛЬ: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Ким Алексей Максимович

"\_\_"\_\_\_\_\_\_\_\_\_\_\_2024 г.

Москва - 2024

**Аннотация**

В данном документе описываются последовательность и методы проведения испытаний при тестировании программного изделия, состав и структура технических и программных средств, необходимых для проведения испытаний, а также приводятся требования к предъявляемой документации, характеристикам программы применительно к условиям эксплуатации и требования к информационной и программной совместимости. Описывается тестовый пример и реакция системы на него.

1. **Объект испытаний**

Система прогнозирования нагрузки СХД с интеллектуальной подсистемой настройки Цель испытаний

Цель испытания – проверка функционирования всех указанных в техническом задании функций программы.

1. **Состав предъявляемой документации**

На испытания программного изделия предъявляются следующие документы:

1) Техническое задание.

2) Программа и методика испытаний.

1. **Технические требования**

* 1. **Требования к программной документации**

Комплектность программной документации должна удовлетворять разделу данного документа "Состав предъявляемой документации".

* 1. **Требования к техническим характеристикам**
     1. **Требования к составу аппаратного обеспечения**

Система будет выполнятся на ноутбуке с операционной системой Mac OS Ventura, с установленным Docker container.

* + 1. **Требование к составу программного обеспечения**

Для работы программного изделия требований к составу программного обеспечения не предъявляется.

1. **Порядок проведения испытаний**

Испытания данного программного изделие будут проводиться в следующем порядке:

1) Запуск системы.

2) Тестирование функционала системы.

* 1. **Требования к составу аппаратного обеспечения**

Требования к составу аппаратного обеспечения учитываются согласно пункту 4.2.1.

* 1. **Требования к составу программного обеспечения**

Требования к составу программного обеспечения учитываются согласно пункту 4.2.2.

1. **Методы испытаний**

|  |  |  |  |
| --- | --- | --- | --- |
| **N** | **N пункта ТЗ** | **Выполняемые действия** | **Результат** |
| 1 | 4.1.1  Указание параметра | Пользователь выбирает параметр один из трех СХД, который он хочет прогнозировать. Можно выбрать несколько | Прогнозируется по выбранному параметру |
| 2 | 4.1.2  Возможность выбрать таблицу из БД | Пользователь выбирает любую таблицу с данными. | Выбирается определенная таблица из БД с историческими данными СХД |
| 3 | 4.1.3  Возможность изменение уровня прогнозирования из БД во frontend | Пользователь вводит в специальное поле SQL запрос для изменения уровня прогнозирования | Уровень прогнозирования в БД, как отдельная таблица, меняет параметр на прописанные в SQL запросе |
| 4 | 4.1.4  Выбор режима скользящего окна | Пользователь:   1. Выбирает автоматический 2. Выбирает ручной    1. Вводит интервал    2. Вводит размер интервала | 1. Происходит выбор автоматического окна, то есть скользящее окно будет подстраиваться и находить оптимальный размер окна 2. Появляется дополнительные 2 поля ввода – интервал и размер интервала |
| 5 | 4.1.5  При ручном вводе указывать интервал и его количество | Пользователь вводит интервал и количество интервалов | После ввода интервала и его размера на всех исторических данных один размер окна |
| 6 | 4.1.6  Выбор уровня предсказания | Пользователь в меню выбирает уровень предсказания (можно выбрать несколько) | Происходит предсказание на определенный уровень загрузки в %, выбранные пользователем |
| 7 | 4.1.7  Выбор альтернативного варианта прогнозирования | Пользователь нажимает на чекбокс | Происходит смена варианта прогнозирования, то есть воспользоваться прогнозированием на основе StoragePool |
| 8 | 4.1.8  Выбор нахождения глобального минимума | Пользователь может выбрать нахождение глобального минимума, если у нас автоматическое скользящее окно | Находится глобальный минимум временных рядов, после чего автоматическое окно начинается с него, что помогает избежать ошибки out of bounds, которая возникает из-за пологих участков графика |
| 9 | 4.1.9  Выбор построения облако точек | Пользователь нажимает на чекбокс с построением облака точек | На графике будет построено облако точек |
| 10 | 4.1.10  Визуализировать график | Пользователь нажимает на кнопку визуализировать | Происходит визуализация графика с историческими данными, прогнозом, уровнями и облако точек, если оно выбрано |