

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОГО ГОСУДАРСТВЕННОГО АВТОНОМНОГО ОБРАЗОВАТЕЛЬНОГО УЧРЕЖДЕНИЯ
ВЫСШЕГО ОБРАЗОВАНИЯ

**Национальный исследовательский технологический университет «МИСИС»
Новотроицкий филиал**

ФАКУЛЬТЕТ Металлургических Технологий

КАФЕДРА Математики и Естествознания

НАПРАВЛЕНИЕ Прикладная информатика

ДИСЦИПЛИНА Программная инженерия

КУРСОВОЙ ПРОЕКТ

на тему: **Разработка ui/ux интерфейсов для слежения за металлом**

Студент группы БПИ-21

Л. В. Алексеева

Руководитель к.п.н, доцент

Р. Р. Абдулвелеева

Новотроицк, 2025 г.

Содержание

Введение	3
1 Описание и анализ предметной области	5
1.1 Описание предметной области	5
1.2 Описание систем-аналогов	6
1.2.1 Системы мониторинга и управления производственными процессами	6
1.2.2 Уникальные особенности web-приложения для слежения за металлом	8
2 Постановка задачи и разработка требований	10
2.1 Постановка задачи	10
2.2 Формирование требований к программному обеспечению	12
2.2.1 Функциональные требования	12
2.2.2 Технические требования для разработки UI/UX интерфейсов для слежения за металлом	13
2.3 Ограничения и особенности	16
2.4 Методология разработки	18
3 Проектирование системы	22
3.1 Обоснование выбора языка программирования и средств разработки	22
3.1.1 Комплект средств разработки React	22
3.1.2 Выбор JavaScript и React для разработки веб-приложения	25
3.1.3 Преимущества использования JSON	26
3.1.4 Алгоритм работы приложения	28
3.2 Инструменты визуализации для анализа и управления процессами	32
3.2.2 Диаграмма Исикавы (Cause-and-Effect Diagram)	36
3.2.3 Диаграмма EPC (Event-Driven Process Chain)	39
3.2.4 Диаграмма IDEF0	41
4 Конструирование интерфейса web-приложения для слежения за металлом	43
4.1 Архитектура приложения	43
4.2 Принцип взаимодействия компонентов	45
4.3 Основные элементы интерфейса	45
Заключение	50
Список использованных источников	52

Введение

В современном мире цифровые технологии играют ключевую роль в оптимизации производственных процессов. Автоматизация и визуализация данных позволяют предприятиям повышать эффективность работы, улучшать контроль за производством и минимизировать риски потерь.

Одним из важных направлений является разработка интерфейсов для мониторинга материалов на производственных площадках. В данной работе рассматривается создание web-приложения для отслеживания положения листового проката в листопрокатном цехе. Приложение использует данные из JSON-файлов, содержащие информацию о заказах, марке стали, размерах и текущем положении листов.

Визуализация выполняется с помощью интерактивной мнемосхемы, что упрощает контроль за перемещением материалов и снижает вероятность их утери или повреждения [14, 15]. Разработка интерфейса web-приложения велась с использованием JavaScript и фреймворка React, что позволило создать динамичную и удобную систему взаимодействия пользователя с данными. Отображение информации осуществляется в реальном времени, а интерактивные элементы позволяют пользователю получать сведения о каждом листе нажатием кнопки.

Цель данной работы – разработка удобного и эффективного интерфейса для мониторинга положения металлических листов, что способствует повышению прозрачности и управляемости производственного процесса. В ходе исследования были рассмотрены основные этапы проектирования и реализации web-приложения, алгоритмы работы с JSON-данными, а также способы визуализации информации на мнемосхеме.

Задачи работы:

1. Разработка алгоритма обработки данных из JSON-файлов для отображения информации о листах.
2. Создание интерактивной мнемосхемы для визуализации положения листов в реальном времени.

3. Реализация пользовательского интерфейса с использованием JavaScript и фреймворка React.
4. Обеспечение интерактивности интерфейса, позволяющего пользователю получать детальную информацию о каждом листе.
5. Тестирование и оптимизация работы приложения для повышения удобства использования и точности отображения данных.

1 Описание и анализ предметной области

1.1 Описание предметной области

В современных металлургических производствах слежение за положением металлических листов играет важную роль в обеспечении эффективного управления процессами. Корректное отображение местоположения и параметров листового проката позволяет снизить риски его потери или повреждения, а также повысить общую производственную эффективность. Однако, на многих предприятиях контроль за положением металлических листов ведется вручную или с использованием устаревших систем, что приводит к неточностям, задержкам в производстве и усложнению логистических процессов. Для решения данной проблемы было разработано web-приложение, которое автоматически отображает местоположение листов в листопрокатном цехе на интерактивной мнемосхеме. Оно получает данные из JSON-файлов, содержащих информацию о номере заказа, марке стали, размерах и текущем положении листов, что позволяет в реальном времени визуализировать состояние производства.

Основные участники процесса включают:

- 1) Операторов производства, которые используют приложение для отслеживания текущего состояния и расположения листового проката;
- 2) Административный персонал, ответственный за контроль и управление процессами производства;
- 3) Программистов и технических специалистов, занимающихся поддержкой и доработкой системы. Предметная область данного web-приложения охватывает сферу автоматизированного мониторинга состояния металлических листов в производственном процессе.

Основные элементы предметной области включают:

- 1) Отслеживание и визуализация местоположения листов: приложение использует JSON-файлы для отображения текущего расположения металлических

листов на мнемосхеме. Пользователь может в реальном времени видеть расположение каждого листа и его параметры.

2) Автоматизация производственного процесса Приложение оптимизирует управление листопрокатом, предоставляя актуальную информацию о заказах, что позволяет снижать риски путаницы и потерь.

3) Информационное взаимодействие Пользователи могут получать детальные данные о каждом листе, включая его идентификатор, марку стали, размеры и заказ, к которому он относится.

4) Развитие функционала. В перспективе возможна интеграция дополнительных функций, таких как система уведомлений о перемещениях листов, прогнозирование загруженности складских зон и интеграция с ERP-системами предприятия.

Основные цели разработки приложения: автоматизация процесса контроля за положением листового проката, снижение рисков потери или повреждения металлических листов, повышение эффективности работы операторов производства и управленческого персонала, упрощение процесса анализа данных о производственном процессе.

1.2 Описание систем-аналогов

1.2.1 Системы мониторинга и управления производственными процессами

В металлургической промышленности и смежных отраслях используются различные системы для мониторинга состояния производства, отслеживания движения материалов и управления производственными процессами. Рассмотрим несколько существующих решений.

Система «MES» (Manufacturing Execution System). MES-системы применяются на промышленных предприятиях для контроля и управления производственными процессами в режиме реального времени. Они интегрируются с различными источниками данных, включая ERP-системы и датчики IoT, что позволяет

обеспечивать полный контроль над перемещением материалов, включая листовой прокат. Основные функции: отслеживание перемещения и текущего состояния материалов на производстве; автоматизированный учет заказов и их текущего статуса; контроль качества продукции; интеграция с системами планирования ресурсов предприятия (ERP).

Система «SCADA» (Supervisory Control and Data Acquisition) SCADA-системы используются для диспетчеризации и мониторинга промышленных процессов. Они позволяют собирать и анализировать данные с различных датчиков и автоматических систем, в том числе с систем отслеживания положения металлических листов на производстве. Основные функции: мониторинг параметров производства в режиме реального времени; визуализация технологических процессов на мнемосхемах; автоматизированное оповещение о неисправностях и отклонениях; управление исполнительными механизмами на производстве.

Система оперативного мониторинга и диспетчеризации «Ausferr». Эта система предназначена для мониторинга и управления производственными процессами в металлургической промышленности. Она позволяет отслеживать местоположение и состояние материалов, оптимизировать логистику производства и предотвращать потери или повреждения листового проката. Основные функции: отображение перемещения металлопроката на интерактивных схемах; фиксация времени и условий хранения продукции; автоматическое уведомление о критических изменениях параметров; гибкая настройка интерфейса для различных категорий пользователей.

Существующие системы, такие как MES, SCADA и «Ausferr», обладают широкими возможностями для мониторинга производственных процессов и управления материалами. Однако, большинство из них предназначены для комплексного управления предприятием, что делает их внедрение затратным и сложным. Разрабатываемое web-приложение ориентировано на решение узкоспециализированной задачи – визуализацию местоположения металлических листов на производстве с помощью интерактивного интерфейса. Оно обладает следующими преимуществами: простота внедрения и использования; гибкость и

масштабируемость; визуализация информации в реальном времени с использованием JSON-файлов; возможность дальнейшего развития с интеграцией в более крупные производственные системы.

Таким образом, предлагаемое web-приложение дополняет существующие системы, предоставляя удобный инструмент для мониторинга состояния листового проката в режиме реального времени.

1.2.2 Уникальные особенности web-приложения для слежения за металлом

Существующие системы мониторинга производства обладают мощным функционалом, однако разрабатываемое web-приложение имеет ряд уникальных особенностей, которые делают его более удобным и эффективным для конкретной задачи – отслеживания положения металлических листов в листопрокатном цехе. Эти особенности обусловлены спецификой металлургического производства и направлены на повышение точности, скорости и удобства работы с данными:

1. Ориентированность на производственные процессы в металлургии. Приложение разработано специально для металлургического предприятия, что позволяет учитывать все нюансы работы с листовым прокатом. В отличие от универсальных систем (SCADA, MES), оно сфокусировано на визуализации и управлении перемещением металлических листов. Это включает в себя отображение таких параметров, как номер заказа, уникальный идентификатор листа, марка стали, размеры листа и текущее положение на производственной площадке. Такой подход позволяет минимизировать ошибки, связанные с неправильной интерпретацией данных, и повысить точность контроля за производственным процессом.

2. Визуализация в реальном времени. Приложение использует данные из JSON-файлов для мгновенного отображения положения листов на интерактивной мнемосхеме. Это позволяет рабочим и диспетчерам легко отслеживать местоположение каждого листа без необходимости ручного поиска в документации. Например, приложение отображает листы разными цветами в зависимости от их статуса (в обработке, готовые к отгрузке, на хранении), что упрощает визуальное

восприятие информации. Также реализована возможность получения детальной информации о каждом листе при нажатии на соответствующий элемент мнемосхемы.

3. Простота интеграции и масштабируемость. Благодаря использованию React и JavaScript [2], приложение легко интегрируется в существующую IT-инфраструктуру предприятия. В отличие от сложных и дорогостоящих промышленных решений оно не требует значительных затрат на внедрение. Приложение может работать с различными источниками данных (JSON-файлы, базы данных, API), что делает его гибким инструментом для разных этапов производства. Кроме того, архитектура приложения позволяет легко добавлять новые функции и расширять его функционал в будущем.

4. Гибкость и настройка под нужды производства. Приложение позволяет изменять параметры отображения, добавлять новые виды данных и адаптировать интерфейс под конкретные требования предприятия. Например, можно настроить цветовую схему для разных марок стали или добавить новые параметры для отображения (например, толщину листа или сроки выполнения заказа). Это делает приложение удобным инструментом для разных категорий пользователей – от операторов на производственной линии до руководителей цеха.

5. Интерактивность и удобство взаимодействия. Приложение предоставляет интерактивные элементы управления, такие как кнопки для отображения информации о конкретных листах или участках цеха. Это позволяет пользователям быстро получать необходимую информацию без необходимости сложных манипуляций с интерфейсом. Например, при нажатии на кнопку "json 15" отображается участок мнемосхемы с листами, соответствующими данному заказу, что упрощает навигацию и поиск.

Таким образом, разрабатываемое web-приложение представляет собой узкоспециализированное, но эффективное решение для мониторинга и управления листовым прокатом. Его уникальность заключается в сочетании простоты использования, гибкости и ориентации на конкретные задачи металлургического производства, что делает его более удобным и практичным в сравнении с универсальными системами.

2 Постановка задачи и разработка требований

2.1 Постановка задачи

Перед разработчиками стоит задача создания web-приложения для отслеживания положения металлических листов в листопрокатном цехе. Целью данного проекта является повышение эффективности контроля за передвижением листового проката, сокращение рисков его потери или повреждения, а также оптимизация производственного процесса за счет автоматизированной визуализации данных [3].

Разработанное web-приложение предназначено для отслеживания положения металлических листов в листопрокатном цехе [6]. Основной целью приложения стало повышение эффективности контроля за передвижением листового проката, сокращение рисков его потери или повреждения, а также оптимизация производственного процесса за счет автоматизированной визуализации данных. В процессе разработки были успешно решены следующие ключевые задачи:

1. Реализация удобного пользовательского интерфейса. Пользовательский интерфейс приложения разработан с использованием фреймворка React, что обеспечило его динамичность и интерактивность. Интерфейс позволяет сотрудникам предприятия легко отслеживать текущее положение листов в цехе, включая информацию о номере заказа, марке стали, размерах и уникальном идентификаторе каждого листа. Например, при нажатии на элемент мнемосхемы пользователь получает детальную информацию о конкретном листе, что упрощает процесс контроля.

2. Динамическое обновление данных из JSON-файлов. Приложение автоматически загружает и обрабатывает данные из JSON-файлов, которые содержат информацию о листах (номер заказа, марка стали, размеры, текущее положение и уникальный идентификатор). Это обеспечивает актуальность отображаемой информации и поддержку работы системы в реальном времени. JSON-файлы

используются как основной источник данных, что упрощает интеграцию с существующими системами и обеспечивает легкость обновления информации.

3. Интуитивная визуализация данных с использованием мнемосхемы. Для наглядного отображения положения листов в цехе реализована интерактивная мнемосхема. Каждый лист обозначен на схеме в зависимости от его текущего состояния (например, в обработке, готов к отгрузке, на хранении) и местоположения. Цветовая дифференциация листов по статусам и заказам упрощает визуальное восприятие информации. Мнемосхема динамически обновляется в зависимости от данных, полученных из JSON-файлов.

4. Стабильная работа в условиях промышленного предприятия. Приложение успешно интегрировано в существующую IT-инфраструктуру предприятия и поддерживает работу с различными источниками данных (JSON-файлы, базы данных, API) [10]. Оно устойчиво к нагрузкам и обеспечивает стабильную работу даже в условиях большого объема данных. Это позволяет использовать приложение в реальных производственных условиях без сбоев и задержек.

5. Интерактивные элементы управления. В интерфейсе приложения реализованы интерактивные элементы, такие как кнопки для отображения информации о конкретных листах или участках цеха. Например, при нажатии на кнопку "json 15" отображается участок мнемосхемы с листами, соответствующими данному заказу. Это упрощает навигацию и поиск необходимой информации [11].

6. Гибкость и настройка под нужды производства. Приложение обладает гибкостью и может быть адаптировано под конкретные требования предприятия. Это включает возможность изменения параметров отображения, добавления новых видов данных и настройки интерфейса под нужды разных категорий пользователей (операторы, диспетчеры, руководители). Например, можно настроить цветовую схему для разных марок стали или добавить новые параметры для отображения (например, толщину листа или сроки выполнения заказа).

Разработанное web-приложение стало надежным инструментом для оптимизации управления листовым прокатом, обеспечивая наглядность и удобство работы с данными в режиме реального времени. Его уникальность заключается в

сочетании простоты использования, гибкости и ориентации на конкретные задачи металлургического производства, что делает его более удобным и практичным в сравнении с универсальными системами. Приложение уже успешно используется на предприятии, демонстрируя высокую эффективность в решении поставленных задач.

2.2 Формирование требований к программному обеспечению

2.2.1 Функциональные требования

В приложении должна быть возможность управления меню и заказами со стороны административного персонала. Отображение информации о листах – web-приложение должно предоставлять пользователям возможность просматривать актуальные данные о металлических листах, включая номер заказа, марку стали, размеры, уникальный идентификатор и текущее положение на мнемосхеме.

Разработанное web-приложение для отслеживания положения металлических листов в листопрокатном цехе включает в себя ряд ключевых функций, которые обеспечивают эффективное управление данными и удобство взаимодействия для пользователей. Основные функциональные возможности приложения включают:

1. Приложение предоставляет пользователям возможность просматривать актуальные данные о металлических листах, включая: номер заказа, марку стали, размеры листа, уникальный идентификатор, текущее положение на мнемосхеме. Вся информация отображается в удобном и наглядном формате, что упрощает процесс контроля и поиска данных.

2. Загрузка и обработка JSON-файлов: система автоматически загружает JSON-файлы с производственными данными, анализирует их содержимое и динамически обновляет информацию в приложении. Это обеспечивает актуальность данных и позволяет работать с информацией в режиме реального времени. JSON-файлы содержат все необходимые параметры, такие как номер заказа, марка стали, размеры и текущее положение листов.

3. Интерактивное управление интерфейсом: пользователь может взаимодействовать с интерфейсом приложения, используя следующие возможности: выбор листов на схеме для получения детальной информации, фильтрация листов по заданным параметрам (например, по номеру заказа, марке стали или статусу), получение детальной информации по каждому объекту при нажатии на соответствующий элемент мнемосхемы. Эти функции делают работу с приложением интуитивно понятной и удобной.

4. Цветовая индикация статуса листов: приложение использует систему цветовой маркировки для визуального различения заказов по их состоянию. Например: листы в обработке могут отображаться синим цветом, готовые к отгрузке – зеленым, на хранении – серым. Цветовая индикация ускоряет процесс идентификации и контроля, позволяя быстро оценить текущее состояние производства.

5. Гибкость и масштабируемость: система поддерживает возможность добавления новых параметров и адаптации под изменяющиеся условия работы производства. Это включает: добавление новых полей данных (например, толщина листа, сроки выполнения заказа), настройку интерфейса под нужды разных категорий пользователей (операторы, диспетчеры, руководители), интеграцию с другими системами управления производством. Гибкость приложения позволяет легко масштабировать его функционал в будущем.

2.2.2 Технические требования для разработки UI/UX интерфейсов для слежения за металлом

Разработанное web-приложение для отслеживания положения металлических листов в листопрокатном цехе учитывает современные стандарты проектирования пользовательских интерфейсов [12]. Для обеспечения удобства и эффективности работы с системой были реализованы следующие технические требования к UI/UX:

1. Интуитивно понятный интерфейс

Интерфейс приложения разработан с учетом принципов логичной навигации, что позволяет пользователям быстро находить необходимую информацию без специальной подготовки. Основные элементы управления, такие как кнопки, фильтры и мнемосхема, расположены в соответствии с пользовательскими ожиданиями. Например:

- 1) На главной странице отображается мнемосхема цеха с цветовой индикацией листов;
- 2) Кнопки для фильтрации данных (по номеру заказа, марке стали, статусу) расположены в верхней части экрана;
- 3) Детальная информация о листе отображается при нажатии на соответствующий элемент мнемосхемы.

2. Адаптивный дизайн

Интерфейс приложения корректно отображается на различных устройствах и экранах, включая стационарные компьютеры, планшеты и мобильные устройства. Это достигается за счет использования адаптивной верстки и фреймворка React, который обеспечивает гибкость и масштабируемость интерфейса. Например:

- 1) На мобильных устройствах мнемосхема автоматически масштабируется для удобства просмотра;
- 2) Элементы управления (кнопки, фильтры) адаптируются под размер экрана, сохраняя свою функциональность.

3. Визуальная доступность данных

Информация о положении листов представлена в удобочитаемом формате с использованием цветовой индикации, схем и графиков. Это позволяет пользователям быстро воспринимать информацию и принимать решения. Например:

- 1) Листы на мнемосхеме отображаются разными цветами в зависимости от их статуса (в обработке, готов к отгрузке, на хранении);
- 2) Для отображения данных используются интерактивные элементы, такие как всплывающие подсказки и таблицы с детальной информацией.

4. Минимизация когнитивной нагрузки

Интерфейс приложения максимально прост и понятен, что позволяет пользователям сосредоточиться на решении производственных задач без отвлечения на сложные элементы управления. Например:

- 1) На главной странице отображается только необходимая информация (мнемосхема, кнопки управления, фильтры);
- 2) Дополнительные параметры (например, настройки отображения) скрыты в выпадающих меню, чтобы не перегружать интерфейс.

5. Эргономичное расположение элементов

Кнопки управления, фильтры и другие элементы интерфейса расположены таким образом, чтобы ими можно было удобно пользоваться на различных устройствах. Например:

- 1) Основные кнопки управления (например, "Обновить данные", "Фильтры") расположены в верхней части экрана;
- 2) Элементы мнемосхемы (листы, участки цеха) имеют достаточный размер для удобного взаимодействия на сенсорных устройствах;
- 3) Фильтры и настройки отображения доступны в одном месте, что упрощает их использование.

6. Цветовая индикация и визуальные подсказки

Для повышения удобства работы с приложением используется система цветовой индикации и визуальных подсказок. Например:

- 1) Листы на мнемосхеме выделены цветами в зависимости от их статуса (например, синий – в обработке, зеленый – готов к отгрузке);
- 2) При наведении на элемент мнемосхемы появляется всплывающая подсказка с детальной информацией о листе (номер заказа, марка стали, размеры).

7. Поддержка взаимодействия в реальном времени

Приложение обеспечивает динамическое обновление данных в режиме реального времени, что позволяет пользователям всегда работать с актуальной информацией. Например:

1) При изменении положения листа на производственной площадке мнемосхема автоматически обновляется;

2) Пользователи могут отслеживать перемещение листов без необходимости ручного обновления страницы.

Реализованные технические требования к UI/UX интерфейсу обеспечивают высокую эффективность работы с приложением и удобство взаимодействия для всех категорий пользователей.

2.3 Ограничения и особенности

Разработанная система для мониторинга и слежения за металлическими листами в листопрокатном цехе имеет ряд ограничений и особенностей, которые обусловлены спецификой производственного процесса и техническими требованиями. Эти аспекты важно учитывать при использовании системы:

1. Ограничение по территории использования

Система функционирует исключительно в пределах производственной площадки завода и предназначена для мониторинга перемещения металлических листов в рамках листопрокатного цеха. Она не предусматривает возможность отслеживания материалов за пределами завода или интеграции с внешними системами, не связанными с производственным процессом.

2. Отсутствие возможности изменения данных пользователем

Функциональность системы ограничена мониторингом и визуализацией данных. Пользователи не могут вносить изменения в информацию о листах (например, корректировать номер заказа, марку стали или положение). Все данные обновляются автоматически через интеграцию с внешними информационными системами, такими как SCADA или MES, которые предоставляют актуальную информацию в формате JSON-файлов.

3. Зависимость от графика работы цеха

Система функционирует в соответствии с установленным графиком работы цеха и не предоставляет доступа к данным в режиме 24/7. Это означает, что в периоды

вне рабочих часов (например, ночью или в выходные дни) доступ к системе может быть ограничен. Данное ограничение связано с тем, что обновление информации происходит только в активные часы работы производства.

4. Зависимость от внешних источников данных

Система получает данные исключительно из внешних информационных систем (например, JSON-файлов, SCADA, MES). Это означает, что корректность и актуальность информации в приложении напрямую зависят от качества и своевременности предоставления данных этими системами. В случае сбоев или задержек в работе внешних источников данные в приложении могут быть не обновлены.

5. Ограниченная функциональность для административного персонала

Административный персонал может управлять некоторыми аспектами интерфейса (например, настройками отображения или фильтрами), но не имеет возможности изменять производственные данные. Все изменения в данных происходят только через автоматическое обновление из внешних систем.

6. Отсутствие автономной работы

Система не может функционировать автономно без подключения к внешним источникам данных. Это означает, что в случае отключения или сбоя в работе внешних систем приложение не сможет обновлять информацию о положении листов.

7. Ограничения по масштабируемости

Хотя система поддерживает возможность добавления новых параметров и адаптации под изменяющиеся условия производства, ее функциональность ограничена текущими техническими возможностями и интеграцией с существующими системами. Расширение функционала может потребовать дополнительных ресурсов и доработок.

Преимущества и компенсация ограничений

Несмотря на указанные ограничения, система успешно решает поставленные задачи за счет:

- 1) Автоматизации процессов – данные обновляются автоматически, что минимизирует необходимость ручного вмешательства;
- 2) Интеграции с существующими системами – использование JSON-файлов и других источников данных обеспечивает актуальность информации;
- 3) Удобства использования – интуитивный интерфейс и визуализация данных упрощают процесс мониторинга.

2.4 Методология разработки

Для реализации проекта по разработке интерфейсов для слежения за металлом будет использована методология итеративной и инкрементальной разработки. Эта модель позволяет гибко адаптировать процесс разработки, оперативно внедрять изменения и улучшения на ранних этапах, а также эффективно учитывать потребности пользователей. Проект ориентирован на создание web-приложения для отображения положения листов металла в реальном времени, что требует быстрой адаптации и улучшений интерфейса по мере работы. Методология итеративной и инкрементальной разработки включает шесть фаз:

1. Анализ и планирование

На этом этапе был проведен сбор требований к приложению, анализ потребностей пользователей и планирование структуры и функционала системы. Основные задачи включали:

- 1) Изучение специфики данных о металлических листах (номер заказа, марка стали, размеры, текущее положение);
- 2) Определение ключевых функций приложения (визуализация данных, интерактивное управление, цветовая индикация);
- 3) Планирование архитектуры приложения и выбор технологий (React, JavaScript, JSON).

Результатом этапа стало техническое задание, включающее основные требования и ожидания от системы.

2. Проектирование [4].

На этапе проектирования был разработан дизайн пользовательского интерфейса (UI) и создана архитектура приложения. Основные задачи включали:

- 1) Разработка прототипов интерфейсов, демонстрирующих основные взаимодействия с системой (мнемосхема, кнопки управления, фильтры);
- 2) Определение структуры данных и формата JSON-файлов;
- 3) Создание макетов интерфейса с учетом требований к адаптивности и удобству использования.

Результатом этапа стали готовые прототипы интерфейсов и техническая документация.

3. Реализация и тестирование.

На этапе реализации началась разработка приложения с интеграцией базового функционала. Основные задачи включали:

- 1) Реализация отображения данных о листах металла на мнемосхеме;
- 2) Интеграция системы с JSON-файлами для загрузки и обработки данных;
- 3) Разработка интерактивных элементов управления (кнопки, фильтры, всплывающие подсказки).

После каждого этапа разработки проводилось тестирование для выявления и устранения возможных ошибок. Это включало:

- 1) Функциональное тестирование (проверка корректности отображения данных);
- 2) Юзабилити-тестирование (оценка удобства использования интерфейса);
- 3) Тестирование на различных устройствах (проверка адаптивности).

4. Оценка итерации.

После завершения каждой итерации проводилась оценка достигнутых результатов. Основные задачи включали:

- 1) Сбор обратной связи от тестировщиков и пользователей;
- 2) Анализ выявленных проблем и предложений по улучшению;

3) Корректировка плана разработки для улучшения качества интерфейса и функционала.

Результатом этапа стало внесение изменений в приложение на основе обратной связи.

5. Итерации и инкременты.

Разработка продолжалась по принципу итераций, каждая из которых добавляла новый функционал или улучшения в интерфейс [7]. Основные задачи включали:

- 1) Добавление цветовой индикации для отображения статуса листов;
- 2) Реализация фильтрации данных по различным параметрам (номер заказа, марка стали, статус);
- 3) Улучшение адаптивности интерфейса для мобильных устройств.

Этот подход позволил гибко реагировать на изменения в требованиях и потребности пользователей.

6. Завершение проекта.

После нескольких итераций приложение было готово к финальному запуску. Основные задачи включали:

- 1) Проведение финального тестирования всех функций приложения;
- 2) Подготовка документации и инструкций для пользователей;
- 3) Запуск системы и ее интеграция в рабочие процессы предприятия.

Результатом этапа стало полностью готовое и протестированное приложение, доступное для использования сотрудниками предприятия.

Преимущества выбранной методологии.

Использование методологии итеративной и инкрементальной разработки позволило:

- 1) Гибко адаптироваться к изменениям – вносить корректировки на основе обратной связи от пользователей и тестирования;
- 2) Оперативно внедрять улучшения – каждая итерация добавляла новый функционал или улучшения, что ускоряло процесс разработки;

3) Минимизировать риски – тестирование на каждом этапе позволяло выявлять и устранять ошибки на ранних стадиях;

4) Учитывать потребности пользователей – обратная связь от тестирующих и сотрудников предприятия помогала улучшать интерфейс и функционал.

Сравнение с другими методологиями.

Waterfall – в отличие от жесткого следования этапам, итеративная модель позволила гибко адаптировать продукт под изменяющиеся требования;

Agile – методология "Итеративная и инкрементальная" оказалась более управляемой и предсказуемой, что важно для проектов с ограниченными ресурсами или жесткими сроками.

Данная методология обеспечила успешную разработку и внедрение web-приложения для слежения за металлическими листами, отвечающего всем требованиям пользователей и производственного процесса.

3 Проектирование системы

3.1 Обоснование выбора языка программирования и средств разработки

3.1.1 Комплект средств разработки React

React – это популярная библиотека JavaScript, разработанная компанией Facebook, для создания динамических и интерактивных пользовательских интерфейсов. React позволяет разрабатывать веб-приложения с высокой производительностью и удобством для разработчиков. Основным языком программирования для React – JavaScript, который является одним из самых распространенных языков для веб-разработки. JavaScript поддерживает асинхронное программирование, что делает его идеальным для работы с сетевыми запросами и динамическим обновлением данных. Рассмотрим преимущества React перед другими библиотеками и фреймворками [1].

Компонентный подход: react основан на компонентной архитектуре, что позволяет разбивать интерфейс на независимые, переиспользуемые компоненты. Это упрощает разработку, тестирование и поддержку кода.

Виртуальный DOM: react использует виртуальный DOM, который оптимизирует процесс обновления интерфейса. Это обеспечивает высокую производительность даже при работе с большими объемами данных.

Кроссплатформенность: react позволяет разрабатывать как веб-приложения, так и мобильные приложения с использованием React Native. Это дает возможность использовать одну кодовую базу для разных платформ.

Богатая экосистема: react имеет огромное сообщество разработчиков и множество дополнительных библиотек и инструментов, таких как Redux для управления состоянием приложения, React Router для маршрутизации и другие.

Поддержка JSON: react легко интегрируется с JSON-форматом, что делает его идеальным выбором для приложений, работающих с данными в формате JSON. Это

особенно важно для приложений, таких как разрабатываемое, где данные о листах металла поступают в виде JSON-файлов.

React был выбран в качестве основной библиотеки для разработки интерфейса web-приложения благодаря своим преимуществам, таким как компонентный подход, высокая производительность, кроссплатформенность и удобство работы с JSON-данными. Рассмотрим, как именно React использовался в проекте:

1. Компонентный подход.

React позволил разбить интерфейс приложения на независимые и переиспользуемые компоненты, что упростило разработку и поддержку кода. В проекте были созданы следующие основные компоненты:

1) Мнемосхема – компонент для отображения положения листов в цехе. Каждый элемент мнемосхемы (например, лист или участок цеха) был реализован как отдельный компонент.

2) Информационная панель – компонент для отображения детальной информации о выбранном листе (номер заказа, марка стали, размеры, текущее положение).

3) Фильтры и кнопки управления – компоненты для фильтрации данных и управления отображением (например, кнопки для выбора конкретного заказа или участка цеха).

Компонентный подход позволил легко тестировать и модифицировать отдельные части интерфейса без влияния на остальную систему.

2. Виртуальный DOM.

Использование виртуального DOM в React обеспечило высокую производительность приложения, особенно при работе с большими объемами данных. Например:

1) При обновлении данных из JSON-файлов (например, изменение положения листа) React эффективно обновлял только те части интерфейса, которые изменились, что минимизировало нагрузку на браузер и улучшило отзывчивость приложения.

2) Виртуальный DOM также позволил реализовать динамическое обновление мнемосхемы в реальном времени без необходимости полной перерисовки страницы.

3. Кроссплатформенность.

Хотя в данном проекте основным фокусом была разработка веб-приложения, использование React открывает возможности для создания мобильной версии приложения с использованием React Native. Это может быть полезно для расширения функционала в будущем, например, для предоставления доступа к системе через мобильные устройства.

4. Поддержка JSON.

React легко интегрировался с JSON-форматом, что сделало его идеальным выбором для данного проекта. Основные задачи, связанные с JSON, включали:

- 1) Загрузка данных из JSON-файлов с информацией о листах (номер заказа, марка стали, размеры, текущее положение);
- 2) Парсинг и обработка данных для отображения на мнемосхеме;
- 3) Динамическое обновление интерфейса при изменении данных в JSON-файлах.

React позволял легко работать с JSON-данными, что упростило интеграцию приложения с внешними системами.

5. Асинхронное программирование.

JavaScript, как основной язык программирования для React, поддерживает асинхронное программирование, что было использовано для:

- 1) Загрузки данных из JSON-файлов без блокировки интерфейса;
- 2) Реализации динамического обновления данных в реальном времени;
- 3) Обработки сетевых запросов и ошибок (например, при отсутствии доступа к JSON-файлам).

6. Адаптивный дизайн.

React позволил реализовать адаптивный интерфейс, который корректно отображается на различных устройствах (стационарные компьютеры, планшеты, мобильные устройства). Это было достигнуто за счет:

- 1) Использования CSS-фреймворков (например, Bootstrap или Material-UI);
- 2) Создания компонентов, которые адаптируются под размер экрана;
- 3) Тестирования интерфейса на различных устройствах.

Использование React в проекте обеспечило следующие преимущества:

- 1) Упрощение разработки – компонентный подход и богатая экосистема ускорили процесс создания интерфейса;
- 2) Высокая производительность – виртуальный DOM и асинхронное программирование обеспечили быструю работу приложения даже с большими объемами данных;
- 3) Гибкость и масштабируемость – возможность добавления новых функций и адаптации под изменяющиеся требования;
- 4) Удобство работы с JSON – легкая интеграция с JSON-форматом упростила обработку данных о листах.

3.1.2 Выбор JavaScript и React для разработки веб-приложения

JavaScript был выбран в качестве основного языка программирования для разработки веб-приложения, так как он является стандартом для веб-разработки и поддерживается всеми современными браузерами. React, в свою очередь, был выбран как наиболее подходящий инструмент для создания интерактивных интерфейсов благодаря своей производительности, гибкости и простоте интеграции с JSON-данными. В разрабатываемом приложении React используется для создания динамических мнемосхем, которые отображают положение листов металла в реальном времени. Данные о листах поступают в формате JSON, что позволяет легко их обрабатывать и отображать на интерфейсе. Компонентный подход React позволяет создавать переиспользуемые элементы интерфейса, такие как кнопки, таблицы и графические элементы, что упрощает разработку и поддержку приложения.

3.1.3 Преимущества использования JSON

JSON (JavaScript Object Notation) был выбран в качестве основного формата для передачи данных о металлических листах в разработанном web-приложении. Этот формат обеспечивает легкость чтения, редактирования и обработки данных как человеком, так и машиной. Рассмотрим, как именно JSON использовался в проекте и как данные интегрировались в приложение.

1. Структура JSON-файла [5] содержит всю необходимую информацию о листе металла, которая используется в приложении, обозначенном на рисунке 1:

```
{
  "0": {
    "id_slab": "Z0213387017",
    "order": "4802380238",
    "melting": "Z13387",
    "batch": "2912",
    "steel_grade": "K56-2",
    "size": [
      "10",
      "2216",
      "12870"
    ],
    "position": "00",
    "moving_to": "00"
  }
}
```

Рисунок 1 – Пример JSON-файла

1. id_slab – уникальный идентификатор листа;
2. order – номер заказа;
3. melting – номер плавки;
4. batch – номер партии;
5. steel_grade – марка стали;

6. size – размеры листа (толщина, ширина, длина);
7. position – текущее положение листа в цехе;
8. moving_to – направление перемещения листа.

2. Загрузка и обработка JSON-файлов.

В приложении JSON-файлы загружаются и обрабатываются с использованием JavaScript. Основные шаги включают:

3. Отображение данных на мнемосхеме.

Данные из JSON-файла используются для отображения положения листов на интерактивной мнемосхеме. Например:

- 1) Поле position определяет текущее положение листа на схеме;
- 2) Поле moving_to указывает направление перемещения, что позволяет динамически обновлять мнемосхему;
- 3) Поля size и steel_grade используются для отображения дополнительной информации о листе.

4. Взаимодействие с данными.

Приложение позволяет пользователям взаимодействовать с данными, например:

- 1) При нажатии на элемент мнемосхемы отображается детальная информация о листе (используются поля id_slab, order, steel_grade, size);
 - 2) Фильтрация листов по номеру заказа (order) или марке стали (steel_grade).
- ### 5. Динамическое обновление данных.

JSON-файлы могут обновляться в реальном времени, что позволяет приложению динамически обновлять интерфейс. Например:

- 1) При изменении поля position или moving_to мнемосхема автоматически обновляется;
- 2) Новые данные загружаются и отображаются без необходимости перезагрузки страницы.

6. Пример использования данных из JSON.

На основе вашего примера JSON-файла приложение может отображать следующую информацию:

- 1) Уникальный идентификатор листа: Z0213387017;
- 2) Номер заказа: 4802380238;
- 3) Марка стали: K56-2;
- 4) Размеры листа: толщина – 10, ширина – 2216, длина – 12870;
- 5) Текущее положение: 00 (например, участок цеха);
- 6) Направление перемещения: 00 (например, следующий участок цеха).

Использование JSON в проекте обеспечило следующие преимущества:

- 1) Легкость чтения и редактирования – JSON-файлы легко читаются как разработчиками, так и сотрудниками предприятия;
- 2) Быстрая обработка данных – JSON легко интегрируется с JavaScript, что обеспечивает высокую скорость обработки данных;
- 3) Гибкость – JSON позволяет добавлять новые поля и изменять структуру данных без значительных изменений в коде приложения;

Совместимость – JSON поддерживается большинством современных языков программирования и систем, что упрощает интеграцию с внешними источниками данных.

JSON стал идеальным выбором для передачи и обработки данных в разработанном web-приложении, обеспечивая удобство и эффективность работы с информацией о металлических листах.

3.1.4 Алгоритм работы приложения

Разработанное web-приложение для отслеживания положения металлических листов в листопрокатном цехе работает по четкому алгоритму, который включает несколько ключевых этапов. На основе предоставленных файлов проекта, рассмотрим, как каждый этап реализован в коде.

Первый этап - Загрузка JSON-файлов.

Приложение загружает JSON-файлы, содержащие информацию о листах металла. В проекте это реализовано через импорт JSON-файлов в компоненты. Например, в файле. json1js на рисунке 1:

```
import data from "../jsons/sheets2.json";
```

Рисунок 2 – Импорт JSON-файла

На основе полученных данных приложение отображает мнемосхему, где каждый лист представлен в виде прямоугольника с соответствующими параметрами. В проекте это реализовано через компоненты Table6 и Table7, которые отображают данные в табличном виде. Например, в файле json1.js на рисунке 3:

```
import {useState} from 'react';
import Table6 from './table1';
import Table7 from './table2';
import Legend from './legend1';
import './json1.css';

function Box() {
  return (
    <div>
      <div class="container">
        <div >
          <Table7 />
        </div>
        <div>
          <Table6/>
        </div>
        <div class='legend'>
          <Legend/>
        </div>
      </div>
    </div>
  );
}

export default function App() {
  const [isShown, setIsShown] = useState(false);
```

Рисунок 4 – Отображение мнемосхемы

Здесь компоненты Table6 и Table7 отвечают за отображение данных о листах, а компонент Legend отображает легенду с дополнительной информацией.

Второй этап - взаимодействие с интерфейсом.

Пользователь может взаимодействовать с интерфейсом, нажимая на кнопки для отображения дополнительной информации или изменения отображаемых данных. В проекте это реализовано через состояние (useState) и условный рендеринг. Например, в файле App.js на рисунке 5:

```
export default function App() {  
  const [isShown, setIsShown] = useState(false);  
  
  const handleClick = event => {  
    setIsShown(current => !current);  
  };  
  
  return (  
    <div>  
      <button onClick={handleClick}>json1</button>  
      {isShown && <Box />}  
    </div>  
  );  
}
```

Рисунок 5 – Взаимодействие с пользователем

Здесь при нажатии на кнопку "json1" отображается компонент Box, который содержит мнемосхему и таблицы с данными.

Третий этап - отображение дополнительной информации.

При взаимодействии с элементами интерфейса (например, при нажатии на кнопку) отображается дополнительная информация о листах. В проекте это реализовано в компоненте tableLeg (файл legend1.js на рисунке 6):

```

import React from "react";
import "../legend1.css";
import data from "../jsons/sheets2.json";

var datalist1 = data["0"];

class tableLeg extends React.Component {
  render() {
    return (
      <div>
        <table class="legend" border={2} height={130}>
          <tbody>
            <tr>
              <td><img src={require('./imgOrPov22.png')} width={20} height={130}/></td>
              <td height={125} width={300}>
                <ul>
                  <li> номер заказа: {datalist1["order"]}</li>
                  <li> номер плавки: {datalist1["melting"]}</li>
                  <li> марка стали: {datalist1["steel_grade"]}</li>
                  <li> размер листа: {datalist1["size"][0]} {datalist1["size"][1]} {datalist1["size"][2]}</li>
                </ul>
              </td>
            </tr>
          </tbody>
        </table>
      </div>
    );
  }
}

export default tableLeg;

```

Рисунок 6 – Взаимодействие с компонентами

Здесь отображается таблица с детальной информацией о листе, включая номер заказа, номер плавки, марку стали и размеры.

Итоговый алгоритм работы приложения:

- 1) Загрузка JSON-файлов: данные о листах металла загружаются из JSON-файлов (например, sheets2.json).
- 2) Анализ данных: данные из JSON-файлов анализируются и преобразуются в удобные структуры (например, datalist1).
- 3) Отображение мнемосхемы: данные отображаются в виде таблиц (Table6, Table7) и легенды (Legend).
- 4) Взаимодействие с интерфейсом: пользователь может нажимать на кнопки (например, "json1") для отображения дополнительной информации.
- 5) Отображение дополнительной информации: при взаимодействии с интерфейсом отображается детальная информация о листах (например, в компоненте tableLeg).

б) Динамическое обновление данных: интерфейс обновляется в реальном времени при изменении состояния (например, при нажатии на кнопку).

Таким образом, приложение обеспечивает удобное и эффективное взаимодействие с данными о металлических листах, что делает его надежным инструментом для контроля за производственным процессом.

3.2 Инструменты визуализации для анализа и управления процессами

Визуализация играет ключевую роль в анализе, моделировании и управлении сложными процессами, позволяя представить информацию в структурированной и легко воспринимаемой форме. Диаграммы Исикавы, EPC, IDEF0 и диаграмма Ганта относятся к числу таких инструментов и используются для различных аспектов управления проектами и бизнес-процессами. Несмотря на различие в форме и назначении, эти диаграммы тесно взаимосвязаны: каждая из них фокусируется на определённой стороне процесса — от причинно-следственного анализа (диаграмма Исикавы) и логики выполнения действий (EPC), до функциональной декомпозиции (IDEF0) и календарного планирования (диаграмма Ганта). В совокупности они обеспечивают всестороннее представление о проекте, что способствует более точному планированию, выявлению узких мест, повышению эффективности и достижению поставленных целей.

3.2.1 Диаграмма Ганта

Диаграмма Ганта — это мощный инструмент в области управления проектами и планирования задач. Основная идея диаграммы Ганта заключается в том, чтобы визуализировать ход выполнения проекта во времени с использованием горизонтальных столбцов, представляющих задачи, и стрелок, показывающих длительность каждой задачи. Вот основные характеристики диаграммы Ганта:

1. Хронологическое представление: диаграмма Ганта отображает задачи и события в хронологическом порядке. Это позволяет легко определить последовательность действий и зависимости между ними.

2. Горизонтальные столбцы: каждая задача представляется в виде горизонтального столбца, чаще всего с указанием названия задачи.

3. Длительность задачи: длина столбца соответствует времени, необходимому для выполнения задачи. Это помогает определить, какие задачи требуют больше времени, а какие — меньше.

4. Зависимости между задачами: с помощью стрелок или линий связи можно показать зависимости между задачами, указав, что выполнение одной задачи зависит от завершения другой.

5. Идентификация критических путей: диаграмма Ганта позволяет определить критические пути в проекте - последовательность задач, которая определяет минимальное время выполнения всего проекта.

Диаграмма Ганта позволяет четко увидеть, какие этапы проекта следуют друг за другом, и какова общая продолжительность работы. Визуализация временных рамок помогает избегать планирования слишком кратких или длительных периодов, что может привести к проблемам в ходе выполнения работы. Диаграмма Ганта также помогает в управлении рисками и адаптации к изменениям. Когда четко видно зависимости между задачами и сроки, можно быстро реагировать на изменения в плане. Если одна задача задерживается, можно пересмотреть график и определить, как это повлияет на остальные задачи. Это помогает минимизировать риски и обеспечивать гибкость в планировании.

Для управления сроками разработки была составлена диаграмма Ганта, которая наглядно отображает этапы работ, их продолжительность и взаимосвязи. В контексте данного проекта диаграмма включает следующие ключевые этапы:

1. Подготовительные работы (15.01.2024 – 15.03.2024, 60 дней).

1. Анализ производственных процессов цеха.

2. Сбор требований от операторов и технологов.

2. Анализ требований (10.03.2024 – 10.04.2024, 31 день).

1. Формализация функциональных и технических требований к интерфейсу.
2. Определение структуры JSON-файлов для передачи данных.
3. Проектирование интерфейса (16.03.2024 – 20.05.2024, 65 дней).
 1. Создание макетов мнемосхемы с цветовой индикацией листов.
 2. Проработка компонентов React (кнопки, таблицы, легенда).
4. Разработка прототипа (21.05.2024 – 31.08.2024, 102 дня).
 1. Реализация базового интерфейса с отображением листов (как в коде: Json1, Json2 и т. д.).
 2. Интеграция статических JSON-данных для тестирования.
5. Интеграция с JSON (01.09.2024 – 01.10.2024, 30 дней).
 1. Настройка загрузки и парсинга JSON-файлов с данными о листах.
 2. Динамическое обновление мнемосхемы (аналогично RectangleGrid).
6. Разработка функционала (01.08.2024 – 01.10.2024, 61 день).
 1. Добавление фильтрации по заказам (кнопки "json1", "json2").
 2. Реализация всплывающих подсказок с деталями о листах.
7. Тестирование интерфейса (31.08.2024 – 31.10.2024, 61 день).
 1. Проверка корректности отображения данных на разных устройствах.
 2. Юзабилити-тесты с сотрудниками цеха.
8. Оптимизация производительности (31.08.2024 – 31.10.2024, 61 день).
 1. Ускорение загрузки JSON-файлов.
 2. Улучшение отзывчивости интерфейса при большом количестве листов.
9. Документирование (01.11.2024 – 31.01.2025, 91 день).
 1. Написание руководства для пользователей (операторов).
 2. Подготовка технической документации для разработчиков.
10. Внедрение и обучение (01.11.2024 – 01.12.2024, 30 дней).
 1. Установка системы на рабочие станции цеха.
 2. Проведение тренингов для сотрудников.
11. Поддержка и обновления (02.10.2024 – 03.11.2024, 32 дня).
 1. Исправление багов по обратной связи.

2. Добавление новых функций (например, уведомлений о перемещениях).

Диаграмма Ганта представлена на таблице 1.

Таблица 1 – Диаграмма Ганта по проекту разработка ui/ux интерфейсов для слежения за металлом

№ п/п	Наименование работ	Начало работ	Окончание работ	Продолжительность работ, дни
1	Подготовительные работы	15.01.2024	15.03.2024	60
2	Анализ требований	10.03.2024	10.04.2024	31
3	Проектирование интерфейса	16.03.2024	20.05.2024	65
4	Разработка прототипа	21.05.2024	31.08.2024	102
5	Интеграция с JSON	01.09.2024	01.10.2024	30
6	Разработка функционала	01.08.2024	01.10.2024	61
7	Тестирование интерфейса	31.08.2024	31.10.2024	61
8	Оптимизация производительности	31.08.2024	31.10.2024	61
9	Документирование	01.11.2024	31.01.2025	91
10	Внедрение и обучение	01.11.2024	01.12.2024	30
11	Поддержка и обновления	02.10.2024	03.11.2024	32

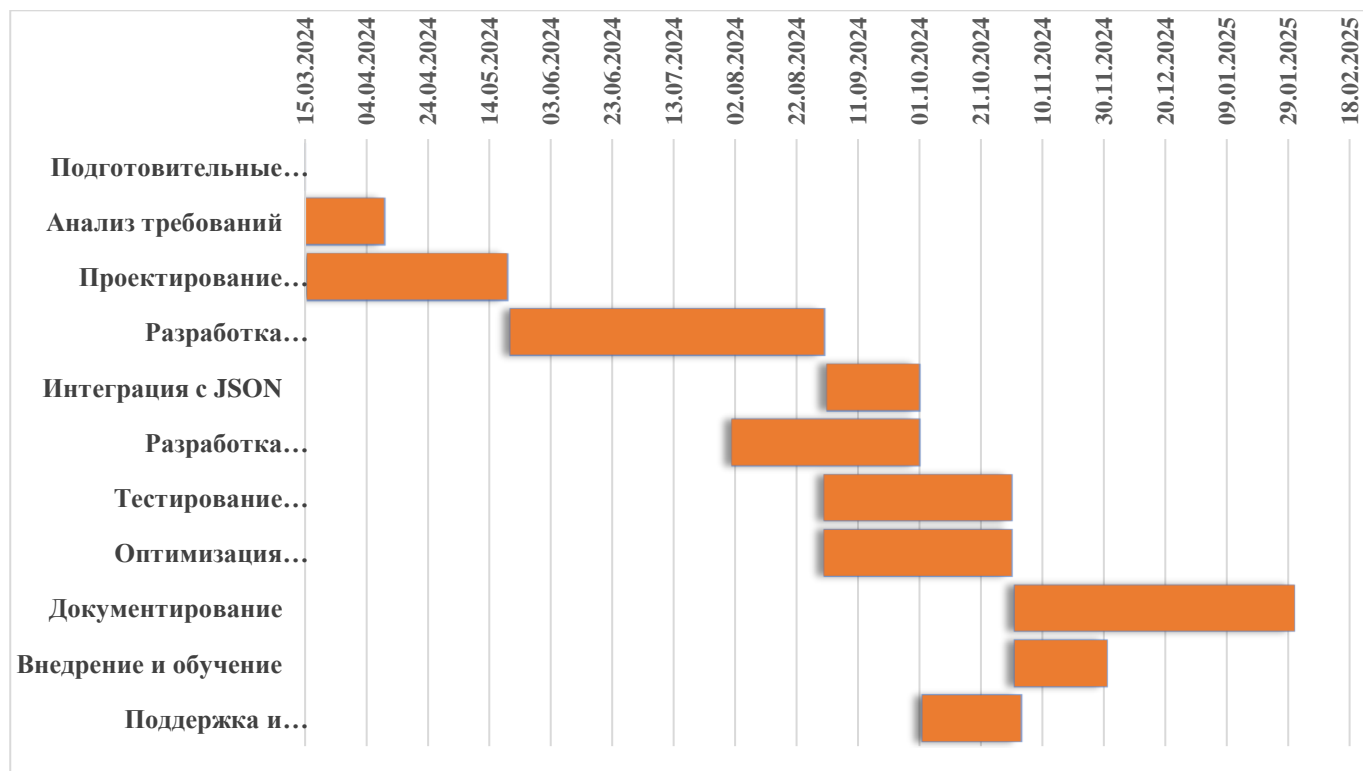


Рисунок 7 – Диаграмма Ганта

3.2.2 Диаграмма Исикавы (Cause-and-Effect Diagram)

Диаграмма Исикавы, также известная как диаграмма "рыбьей кости", используется для анализа причинно-следственных связей в системе. Она помогает выявить основные факторы, влияющие на проблему или процесс. В контексте разработки системы для слежения за металлом диаграмма Исикавы может быть использована для анализа факторов, влияющих на эффективность отслеживания листов металла.

Цель диаграммы Исикавы

1. Выявить ключевые причины возникновения проблемы.
2. Сгруппировать факторы по категориям (оборудование, данные, персонал и т. д.).

3. Определить точки улучшения для оптимизации системы.

1. Проблема (голова рыбы- неэффективное отслеживание положения металлических листов) проявляется в:

1. Потере листов на производстве.
2. Задержках в обновлении данных.
3. Ошибках визуализации на мнемосхеме.

2. Основные категории (кости рыбы)

Разработка ПО:

1. Неточная спецификация требований
2. Сложности в UI/UX-проектировании
3. Ошибки в алгоритме обработки JSON
4. Недостаточное тестирование

Технологии:

1. Ограничения React/JavaScript
2. Ограничения React/JavaScript
3. Проблемы с CSS-адаптацией

Данные:

1. Неточные координаты листов
2. Задержки в обновлении JSON
3. Ошибки парсинга данных

Оборудование:

1. Сбои датчиков позиционирования
2. Проблемы с серверами хранения JSON

Персонал:

1. Ошибки операторов при вводе данных
2. Недостаточное обучение работе с системой

Внешние факторы:

1. Влияние температуры/влажности на датчики
2. Перебои электропитания

1) Проблема (голова рыбы): основная проблема, которую необходимо решить (например, "Потеря листов металла в производственном процессе").

2) Основные категории (кости рыбы): основные группы факторов, влияющих на проблему. Проблемы с датчиками, сбои в работе оборудования. Персонал: ошибки операторов, недостаток квалификации. Процессы: неэффективные процессы отслеживания, отсутствие автоматизации. данные: неточность данных, задержки в обновлении информации. Внешние факторы: влияние внешних условий (например, температура, влажность). Диаграмма данного web-приложения показана на рисунке 8.

Проблемы и их реализация

1. Разработка ПО

1. Проблема: Неточности в требованиях приводят к некорректной визуализации.

2. Решение:

1. Детальный анализ производственных процессов перед разработкой.
2. Прототипирование интерфейса с участием операторов цеха.

2. Данные (JSON-обработка)

1. Проблема: Задержки в обновлении координат листов.

2. Решение:

1. Оптимизация парсинга JSON (кеширование, инкрементальное обновление).

2. Валидация данных перед отображением.

3. Оборудование

1. Проблема: Датчики не всегда передают актуальные данные.

2. Решение:

1. Резервные источники данных (например, ручной ввод при сбоях).

2. Мониторинг состояния датчиков.

4. Персонал

1. Проблема: Операторы не всегда корректно используют интерфейс.

2. Решение:

1. Проведение обучающих тренингов.

2. Упрощение интерфейса (подсказки, цветовая индикация).

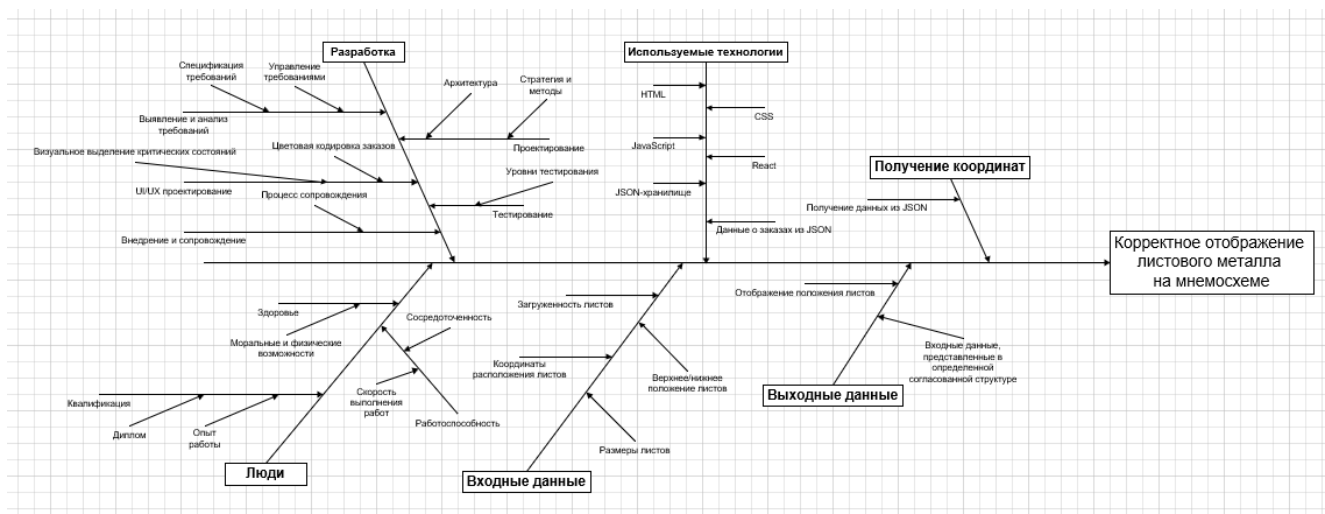


Рисунок 8 - Диаграмма Исикавы

Диаграмма Исикавы позволила провести структурированный анализ ключевых факторов, влияющих на эффективность системы слежения за металлическими листами. На основе проведенного исследования можно сделать следующие выводы:

1. Основные проблемы сосредоточены в нескольких областях: технические ограничения (задержки обработки JSON, сбои датчиков), человеческий

фактор (ошибки операторов, недостаточное обучение), процесс разработки (неточности в требованиях, недостаточное тестирование).

2. Критические точки улучшения: оптимизация загрузки и парсинга JSON-данных, повышение надежности оборудования (датчики, серверы), упрощение интерфейса и обучение персонала.

3. Рекомендации для реализации: внедрить кеширование данных для уменьшения задержек, добавить валидацию входных данных для исключения ошибок, провести дополнительные тесты UI/UX с участием операторов.

Диаграмма наглядно показала, что проблема носит комплексный характер, и ее решение требует системного подхода, включающего как технические доработки, так и организационные меры.

Применение диаграммы Исикавы помогла выявить корневые причины неэффективности системы и наметить конкретные шаги для ее оптимизации.

3.2.3 Диаграмма EPC (Event-Driven Process Chain)

Диаграмма EPC используется для моделирования бизнес-процессов и отображения последовательности событий и функций. Она позволяет визуализировать поток процессов, участников и ресурсов. В контексте системы слежения за металлом диаграмма EPC может быть использована для описания процесса отслеживания листов металла. Основные элементы диаграммы EPC:

1) События (Events): Состояния системы, которые инициируют или завершают процесс (например, "Поступление нового заказа").

2) Функции (Functions): Действия, которые выполняются в процессе (например, "Обновление данных о листах").

3) Участники (Actors): Роли или системы, которые выполняют функции (например, "Оператор", "Система отслеживания").

4) Потоки данных (Data Flows): Передача данных между функциями и событиями.

Данная диаграмма показана на рисунке 9.

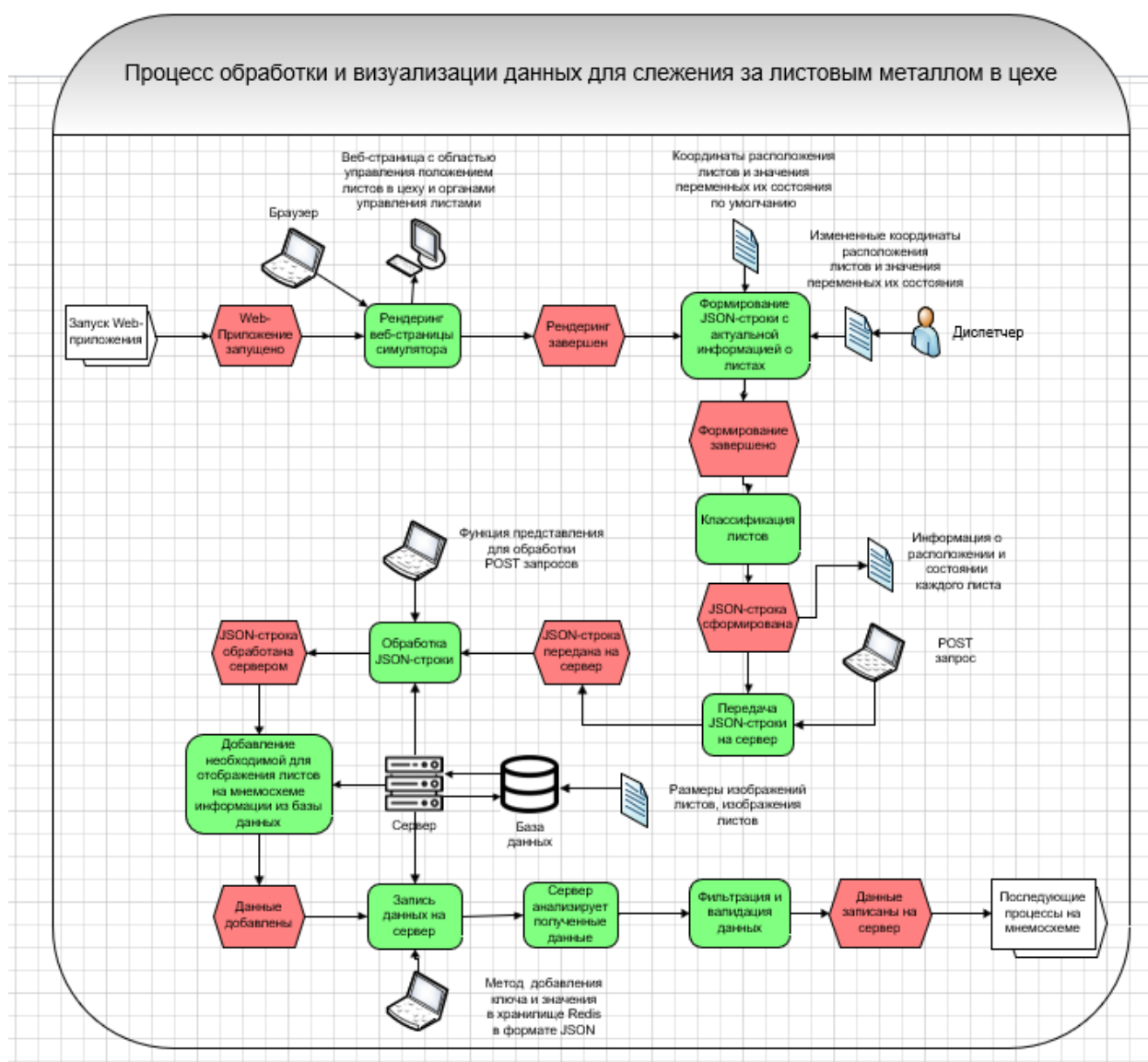


Рисунок 9 - Диаграмма EPC процесс обработки и визуализации данных для слежения за листовым металлом в цехе

Опишем основные этапы процесса.

Запуск Web-приложения: пользователь (диспетчер) открывает браузер, в котором загружается web-приложение, производится рендеринг веб-интерфейса с возможностью управления положением листов в цехе, завершается процесс рендеринга, и интерфейс готов к работе.

Формирование данных о листах: приложение собирает актуальную информацию о листах (координаты, текущее состояние, номер заказа и т. д.),

создается JSON-строка с этими данными, данные классифицируются и подготавливаются к передаче, передача данных на сервер, сформированная JSON-строка, отправляется на сервер, анализирует данные, фильтрует и выполняет валидацию.

Данные записываются в базу данных и сохраняются. Используется хранилище Redis для ускоренного доступа к данным. Обновление отображения на мнемосхеме. После обработки данные о листах обновляются в web-приложении.

Положение и состояние листов отображаются на мнемосхеме в реальном времени.

Диспетчер может изменять координаты и отслеживать статус каждого листа.

При необходимости отправляются новые POST-запросы для обновления информации.

3.2.4 Диаграмма IDEF0

Диаграмма IDEF0 используется для функционального моделирования системы. Она позволяет описать процессы, входные и выходные данные, механизмы и управляющие воздействия. В контексте системы слежения за металлом диаграмма IDEF0 может быть использована для описания процесса обработки данных о листах металла. Основные элементы диаграммы IDEF0:

- 1) **Функции (Activities):** Основные процессы системы (например, "Обработка данных о листах").
- 2) **Входные данные (Inputs):** Данные, которые поступают в систему (например, "JSON-файлы с информацией о листах").
- 3) **Выходные данные (Outputs):** Результаты выполнения функции (например, "Обновленная мнемосхема").
- 4) **Механизмы (Mechanisms):** Ресурсы, необходимые для выполнения функции (например, "Сервер", "База данных").
- 5) **Управляющие воздействия (Controls):** Правила или ограничения, которые управляют процессом (например, "Протоколы обработки данных").

Данная диаграмма указана на рисунке 10.

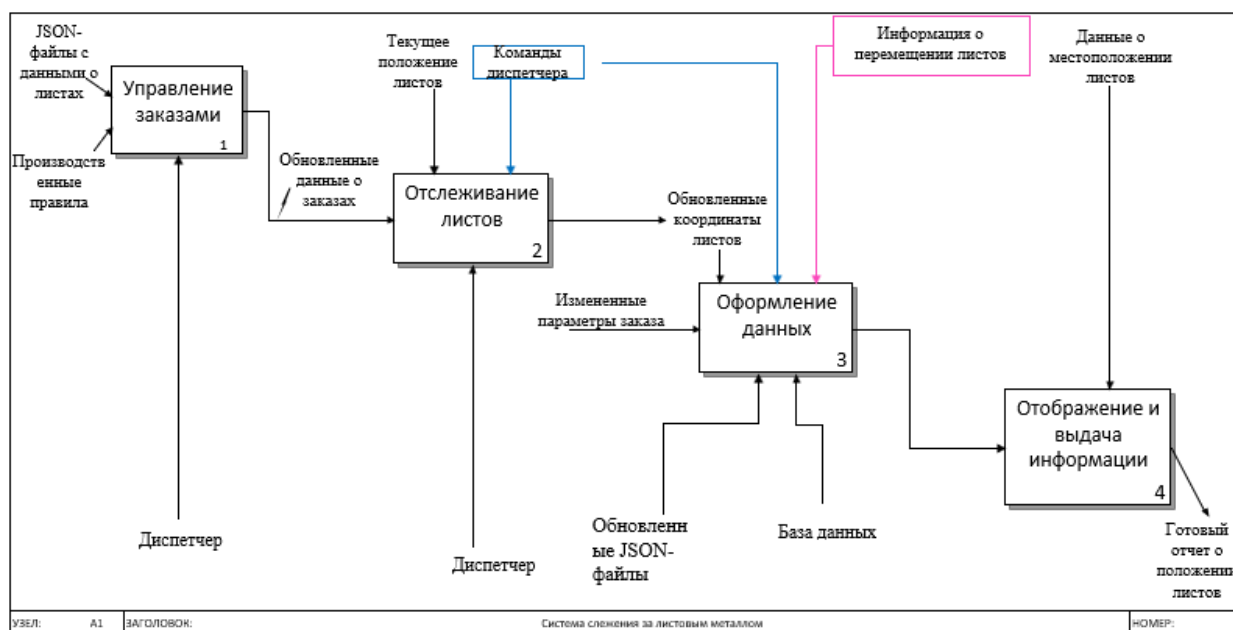


Рисунок 10 - Диаграмма IDEF0

Основные этапы процесса

Управление заказами:

- 1) Диспетчер работает с JSON-файлами, содержащими данные о листах (номер заказа, размеры, положение, марка стали и др.).
- 2) Производственные правила определяют порядок обработки листов.
- 3) Формируются обновленные данные о заказах, которые передаются в систему отслеживания.
- 4) Отслеживание листов

Система получает текущее положение листов и отслеживает их перемещение. Диспетчер может вносить изменения, отправляя команды для корректировки параметров заказа. Информация о перемещениях передается в модуль оформления данных.

Оформление данных: вносятся измененные параметры заказов, обновляются координаты листов и формируются JSON-файлы с актуальными данными, данные записываются в базу данных для последующего использования, отображение и выдача информации: данные о местоположении листов передаются в систему

визуализации. Генерируется готовый отчет о положении листов. Web-приложение отображает актуальное состояние листового проката.

4 Конструирование интерфейса web-приложения для слежения за металлом

4.1 Архитектура приложения

Разработанное web-приложение предназначено для отслеживания местоположения листового металла в производственном цехе. Основная цель системы – обеспечение наглядного контроля за текущим состоянием листов, минимизация потерь и оптимизация логистических процессов. Архитектура приложения построена по клиент-серверной модели и включает следующие ключевые компоненты:

1. Frontend (Клиентская часть).

Клиентская часть приложения реализована с использованием React.js, что обеспечивает высокую интерактивность и плавную работу интерфейса. Основные функции и особенности:

- 1) Динамическое обновление интерфейса: React позволяет обновлять только те части интерфейса, которые изменились, что повышает производительность.
- 2) Мнемосхема: Визуализация данных о листах металла представлена в виде интерактивной мнемосхемы, где каждый лист отображается в виде прямоугольника с цветовой индикацией его статуса [8, 14].

Компонентный подход: Интерфейс разбит на независимые компоненты, такие как: Table6, Table7 – для отображения данных в табличном виде; Legend – для отображения легенды с детальной информацией о листах; Box – контейнер для мнемосхемы и таблиц.

Интерактивные элементы: Пользователь может взаимодействовать с интерфейсом, нажимая на кнопки (например, "json1") для отображения дополнительной информации. Пример показан на рисунке 11.

```

class tableLeg extends React.Component {
  render() {
    return (
      <div>
        <table class="legend" border={2} height={130}>
          <tbody>
            <tr>
              <td><img src={require('./imOrPov33.png')} width={20} height={130}/></td>
              <td height={125} width={300}>
                <ul>
                  <li>номер заказа: {datalist0["order"]}</li>
                  <li>номер плавки: {datalist0["melting"]}</li>
                  <li>марка стали: {datalist0["steel_grade"]}</li>
                  <li>размер листа: {datalist0["size"][0]} {datalist1["size"][1]} {datalist1["size"][2]}</li>
                </ul>
              </td>
            </tr>
            <tr>
              <td><img src={require('./imOrPov22.png')} width={20} height={130}/></td>
              <td height={125} width={300}>
                <ul>
                  <li>номер заказа: {datalist1["order"]}</li>
                  <li>номер плавки: {datalist1["melting"]}</li>
                  <li>марка стали: {datalist1["steel_grade"]}</li>
                  <li>размер листа: {datalist1["size"][0]} {datalist1["size"][1]} {datalist1["size"][2]}</li>
                </ul>
              </td>
            </tr>
          </tbody>
        </table>
      </div>
    );
  }
}

export default tableLeg;

```

Рисунок 11 – Клиентская часть приложения

2. Backend (Серверная часть).

Серверная часть отвечает за обработку данных из JSON-файлов и взаимодействие с клиентской частью.

Основные функции:

- 1) Загрузка и обработка данных: JSON-файлы загружаются и анализируются для дальнейшего использования в интерфейсе.
- 2) Фильтрация и преобразование данных: Данные фильтруются и преобразуются в удобные для отображения структуры.

Расширяемость: В будущем серверная часть может быть расширена для работы с базами данных (например, SQLite, PostgreSQL) или API промышленных систем.

Пример обработки данных:

```
import data from './jsons/sheets2.json'.
```

```
var datalist1 = data["0"];
```

3. Взаимодействие между компонентами.

Frontend взаимодействует с Backend для получения данных из JSON-файлов. Backend обрабатывает данные и передает их в Frontend для отображения на мнемосхеме и в таблицах. Хранилище данных (JSON-файлы) предоставляет актуальную информацию о листах металла.

4. Пример работы системы.

Загрузка данных: Приложение загружает JSON-файл с данными о листах металла. Обработка данных: Данные анализируются и преобразуются в удобные для отображения структуры. Отображение данных: Данные отображаются на мнемосхеме и в таблицах.

Взаимодействие с пользователем: Пользователь может нажимать на кнопки для отображения дополнительной информации или фильтрации данных.

4.2 Принцип взаимодействия компонентов

Пользователь загружает веб-страницу приложения. Приложение запрашивает список доступных JSON-файлов. Пользователь выбирает нужный файл – данные загружаются и обрабатываются. Интерфейс отображает актуальное состояние листов на мнемосхеме. Пользователь может взаимодействовать с листами (просматривать характеристики, менять фильтры).

4.3 Основные элементы интерфейса

Панель выбора JSON-файла: представлена в виде набора кнопок, каждая из которых соответствует конкретному файлу (например, json1, json2, json15). При нажатии на кнопку загружаются данные из выбранного файла.

Мнемосхема расположения листов: отображает визуальную карту цеха. Листы металла представлены в виде прямоугольников с уникальными идентификаторами. Расположение объектов на схеме соответствует их реальному положению в производственном цехе.

Блок информации о листах: после выбора конкретного листа пользователю предоставляется детальная информация: номер заказа, номер плавки, марка стали, размер листа (длина, ширина, толщина), текущее состояние (ожидание обработки, в процессе, готов к отгрузке)

Цветовое кодирование состояния листов: желтый – новые листы, только поступившие в цех. Зеленый – листы в процессе обработки. Синий – листы, готовые к отгрузке. Оранжевый – листы с особыми характеристиками. При загрузке приложения отображается список доступных JSON-файлов. После выбора файла отображается мнемосхема с актуальными данными. Клик по листу открывает дополнительное окно с детальной информацией.

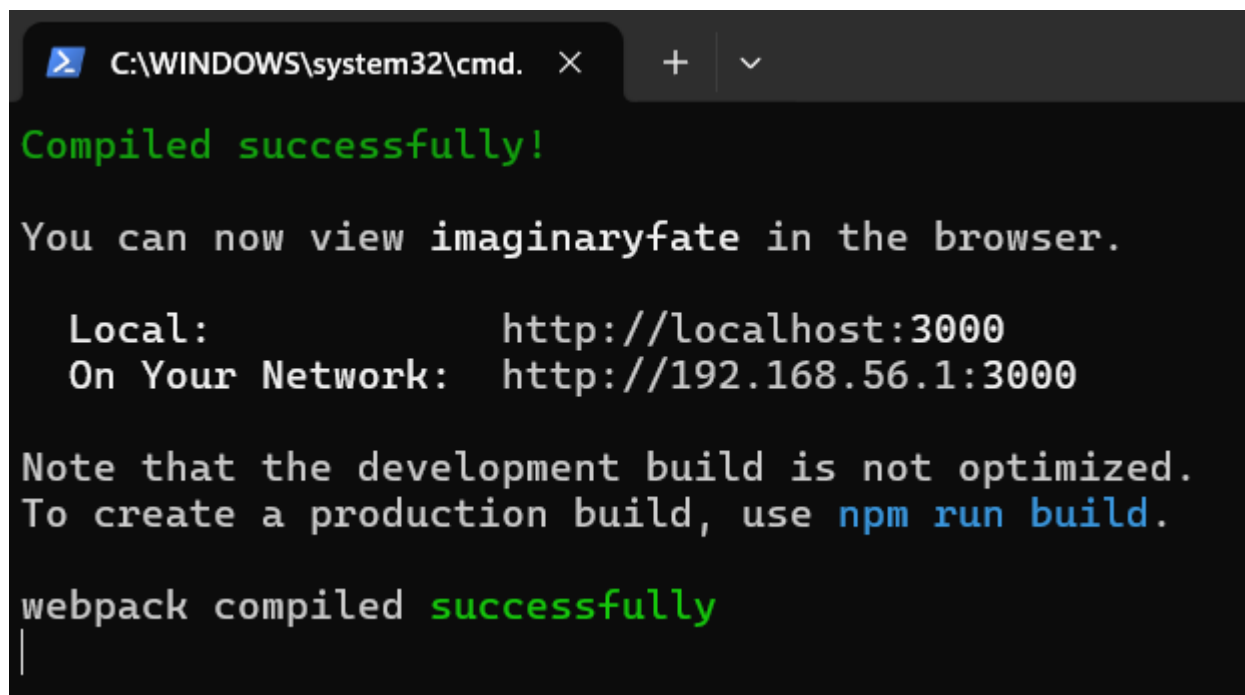
Алгоритм работы:

- 1) Получение списка доступных JSON-файлов.
- 2) Выбор пользователем нужного файла.
- 3) Разбор содержимого файла и его отображение на интерфейсе.
- 4) Формирование мнемосхемы с учетом расположения листов.
- 5) Отображение детальной информации по запросу пользователя.

Мнемосхема расположения листов.

Мнемосхема отображает визуальную карту цеха. Листы металла представлены в виде прямоугольников с уникальными идентификаторами. Расположение объектов на схеме соответствует их реальному положению в производственном цехе. Реализация: каждый лист отображается в виде прямоугольника с цветовой индикацией его состояния. Положение прямоугольников на схеме определяется данными из JSON-файла (поле position).

Интерфейс работает и запускается следующим образом: папка с проектом открывается в терминале и прописывается следующая команда: `npm start`, после чего запускается web-приложение. Если в программе нет ошибок, то в терминале выводится такая надпись, показанная на рисунке 12.



```
C:\WINDOWS\system32\cmd. X + v
Compiled successfully!
You can now view imaginaryfate in the browser.

Local:      http://localhost:3000
On Your Network:  http://192.168.56.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
|
```

Рисунок 12 – Запуск программы

Далее выводится следующий интерфейс, показанный на рисунке 13:



Рисунок 13 – Презентация интерфейса

На сайте расположены кнопки json1, json2 и тд, при нажатии на каждую кнопку выводится мнемосхема и сверху листы, в соответствии с информацией, которая хранится в json файлах, которые находятся в проекте. Также справа указана таблица, где находится ключевая информация о листах, которая содержится в json-файлах.

Мнемосхема расположения листов

json1 | json2 | json3 | json4 | json5 | json6 | json7 | json8 | json9 | json10 | json11 | json12 | json13

Информация о листах

• номер заказа: 4803270327 • номер плавки: Z14704 • марка стали: K56-2 • размер листа: 10 1618 12559
• номер заказа: 4802380238 • номер плавки: Z13387 • марка стали: K56-2 • размер листа: 10 1618 12559
• номер заказа: 4804680468 • номер плавки: V20135 • марка стали: K56-2 • размер листа: 10 1618 12559

Рисунок 14 - Пользовательский интерфейс

Мнемосхема расположения листов

json1 | json2 | json3 | json4 | json5 | json6 | json7 | json8 | json9 | json10 | json11 | json12 | json13 | json14 | json15

Информация о листах

• номер заказа: 4803270327 • номер плавки: Z14704 • марка стали: K56-2 • размер листа: 10 2528 12870
• номер заказа: 4802380238 • номер плавки: Z13387 • марка стали: K56-2 • размер листа: 10 2528 12870
• номер заказа: 4804680468 • номер плавки: V20135 • марка стали: K56-2 • размер листа: 10 2528 12870
• номер заказа: 4804700470 • номер плавки: Z20609 • марка стали: K56-2 • размер листа: 10 2528 12870

Рисунок 15 - Отображение листов в соответствии с данными, полученными из json-файла

Разработанное web-приложение для отслеживания металлических листов успешно решает поставленные задачи, предоставляя операторам цеха удобный и наглядный инструмент для контроля за перемещением листового проката.

Ключевые преимущества системы.

1. Интуитивный интерфейс: панель выбора JSON-файлов с кнопочной навигацией (json1, json2, ..., json15), мнемосхема с точным отображением положения

листов в реальном времени, детальная информация о каждом листе (номер заказа, марка стали, размеры, статус).

2. Эффективная визуализация: цветовое кодирование состояний (желтый – новый, зеленый – в обработке, синий – готов к отгрузке), динамическое обновление данных при загрузке новых JSON-файлов.

3. Простота использования: запуск приложения одной командой (npm start), адаптивный дизайн, работающий на разных устройствах.

Приложение значительно упрощает мониторинг листового проката, снижая риски потерь и повышая эффективность управления производственным процессом. Дальнейшая доработка системы позволит расширить ее функционал и интегрировать с другими промышленными решениями.

Заключение

В данной работе была проведена разработка и анализ web-приложения для мониторинга положения металлических листов в листопрокатном цехе. Основные этапы работы включали описание предметной области, постановку задачи, разработку функциональных и технических требований, проектирование системы и выбор инструментов разработки [9]. Подведем итоги по каждому из разделов.

В первом разделе была рассмотрена предметная область, охватывающая сферу автоматизированного мониторинга состояния металлических листов в производственном процессе. Были выделены основные участники процесса, такие как операторы производства, административный персонал и технические специалисты. Также были проанализированы существующие системы-аналоги, такие как MES, SCADA и «Ausferr», что позволило выявить их преимущества и недостатки.

Разрабатываемое web-приложение было позиционировано как узкоспециализированное решение, ориентированное на визуализацию положения листового проката в реальном времени, что делает его уникальным в сравнении с универсальными системами.

Во втором разделе были сформулированы основные задачи проекта, включая разработку удобного пользовательского интерфейса, динамическое обновление данных из JSON-файлов, визуализацию данных на мнемосхеме и обеспечение стабильной работы приложения в условиях промышленного предприятия. Были определены функциональные, эргономические и технические требования к приложению, а также выбрана методология разработки – итеративная и инкрементальная модель. Это позволило гибко управлять процессом разработки, оперативно вносить изменения и учитывать обратную связь от пользователей.

В третьем разделе было обосновано использование языка программирования JavaScript и библиотеки React для разработки web-приложения. React был выбран благодаря своей компонентной архитектуре, высокой производительности и удобству работы с JSON-данными. Также были рассмотрены преимущества использования JSON для передачи данных о листах металла, что обеспечило простоту интеграции и

быструю обработку информации. Алгоритм работы приложения включает загрузку JSON-файлов, анализ данных, отображение мнемосхемы и взаимодействие с пользователем.

В заключение можно сделать следующий вывод. В ходе выполнения работы была успешно разработана концепция web-приложения для мониторинга положения металлических листов в листопрокатном цехе. Приложение позволяет автоматизировать процесс контроля за перемещением листового проката, снизить риски потери или повреждения материалов и повысить эффективность работы операторов и управленческого персонала. Использование современных технологий, таких как React и JSON, обеспечило высокую производительность и удобство взаимодействия с интерфейсом. Разработанное web-приложение имеет потенциал для дальнейшего развития, включая интеграцию с ERP-системами предприятия, добавление функций уведомлений и прогнозирования загруженности складских зон. Внедрение данного решения на производстве позволит повысить прозрачность и управляемость производственного процесса, что в итоге приведет к снижению издержек и повышению общей эффективности работы предприятия.

По данной работе был получен специальный приз в которой присутствовала часть реализации, которая описана в данной работе (см. приложение А).

Список использованных источников

1. Андреев, А.В. Разработка веб-приложений с использованием React / А. В. Андреев. – Москва: ДМК Пресс, 2021. – 320 с. – ISBN 978-5-97060-987-6.
2. Белов, С.И. JavaScript: полное руководство для разработчиков / С. И. Белов. – Санкт-Петербург: Питер, 2020. – 480 с. – ISBN 978-5-4461-1234-2.
3. Васильев, Д.А. Современные методы визуализации данных в веб-приложениях / Д. А. Васильев. – Москва: Наука, 2019. – 256 с. – ISBN 978-5-02-041673-4.
4. Григорьев, И.Н. Проектирование пользовательских интерфейсов: от теории к практике / И. Н. Григорьев. – Санкт-Петербург: БХВ-Петербург, 2022. – 384 с. – ISBN 978-5-9775-5012-9.
5. Дубов, А.С. JSON: обработка и использование данных в веб-приложениях / А. С. Дубов. – Москва: МЦНМО, 2021. – 208 с. – ISBN 978-5-4439-1200-2.
6. Егоров, В.П. Методологии разработки программного обеспечения: Agile, Scrum, Waterfall / В. П. Егоров. – Москва: Академия, 2020. – 288 с. – ISBN 978-5-4461-0932-8.
7. Иванов, К.Л. React и Redux: разработка современных веб-приложений / К. Л. Иванов. – Санкт-Петербург: Питер, 2021. – 416 с. – ISBN 978-5-4461-1736-0.
8. Козлов, М.А. Визуализация данных в промышленных системах / М. А. Козлов. – Москва: Издательский дом "Вильямс", 2018. – 352 с. – ISBN 978-5-8459-1338-0.
9. Ларин, П.В. Проектирование и разработка веб-интерфейсов / П. В. Ларин. – Санкт-Петербург: БХВ-Петербург, 2020. – 368 с. – ISBN 978-5-9775-5011-2.
10. Морозов, А.Н. Базы данных и их интеграция с веб-приложениями / А. Н. Морозов. – Москва: ДМК Пресс, 2019. – 304 с. – ISBN 978-5-97060-986-9.
11. Петров, В.С. Интерактивные мнемосхемы в промышленных системах / В. С. Петров. – Москва: Наука, 2020. – 272 с. – ISBN 978-5-02-041674-1.
12. Сидоров, А.А. Современные подходы к разработке веб-приложений / А. А. Сидоров. – Санкт-Петербург: Питер, 2021. – 448 с. – ISBN 978-5-4461-1737-7.
13. сравнительный анализ методов и средств визуализации перемещения кранов электросталеплавильного цеха АО "Уральская сталь" для использования в веб-

приложении Корсаков В.А., Абдулвелеева Р.Р. В сборнике: Цифровые системы и модели: теория и практика проектирования, разработки и применения. Материалы национальной (с международным участием) научно-практической конференции. Казань, 2024. С. 270-274. 0 4.

14. Аналитика уязвимости веб-приложения мнемосхемы электросталеплавильного цеха с системой визуализации движения кранов и сталь-ковшей Кравченко В.А., Абдулвелеева Р.Р. В сборнике: Цифровые системы и модели: теория и практика проектирования, разработки и применения. Материалы национальной (с международным участием) научно-практической конференции. Казань, 2024. С. 275-279.

15. R. R. Abdulveleeva, V. A. Korsakov, and I. R. Abdulveleev, “Crane movement modulator for web application of electric steel melting shop mnemonic scheme with crane movement visualisation system,” 2024 International Ural Conference on Electrical Power Engineering (UralCon), Magnitogorsk, Russian Federation, 2024, pp. 812–816, doi: 10.1109/UralCon62137.2024.10718976

ПРИЛОЖЕНИЕ А

Специальный приз, полученный за работу, в которой присутствовала часть реализации, которая описана в данной работе.



Рисунок 1 – Специальный приз