|  |
| --- |
| МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ  федеральное государственное АВТОНОМНОЕ образовательное учреждение высшего образования  «Национальный исследовательский ядерный университет «МИФИ» |
| **Технологический институт –**  филиал федерального государственного автономного образовательного учреждения высшего образования «Национальный исследовательский ядерный университет «МИФИ»  **(ТИ НИЯУ МИФИ)** |

**Кафедра информационных технологий и прикладной математики**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА**

направление подготовки 09.03.01 «Информатика и вычислительная техника»

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Тема:** | «Разработка программного модуля «Анализ выполнения плана | | | | |
|  | запуска сопроводительных паспортов, контроль изготовления | | | | |
|  | крупных узлов, прогноз выполнения контрактов» | | | | |
|  | |  | | |  |
| **Руководитель** | |  | | |  |
| Руководитель группы УИТиС | |  |  |  | И. Д. Смирнов |
| *(должность)* | |  | *(подпись)* |  | *(И.О. Фамилия)* |
| **Студент** | |  |  |  |  |
| ИВТ-40Д | |  |  |  | М.А Щелканов |
| *(группа)* | |  | *(подпись)* |  | *(И.О. Фамилия)* |
|  | |  |  |  |  |
|  | |  |  |  |  |

**ВКР допущена к защите в ГЭК**

**Заведующий кафедрой**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Н.В. Чупракова

«25» июня 2024 г.

*ВКР защищена «25» июня 2024 г.*

*Протокол ГЭК №\_\_\_\_\_ на оценку «\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_»*

*Секретарь ГЭК\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Ю.А. Порохина/*

г. Лесной – 2024 г.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc137939820)

[ГЛАВА 1 ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ 6](#_Toc137939821)

[1.1 Анализ предметной области 6](#_Toc137939822)

[1.2. Описание бизнес-процесса 12](#_Toc137939823)

[1.3 Средства разработки программного обеспечения 16](#_Toc137939824)

[ГЛАВА 2 РАЗРАБОТКА ПРИЛОЖЕНИЯ 20](#_Toc137939825)

[2.1. Построение вспомогательной информационной модели 20](#_Toc137939826)

[2.2 Проектирование базы данных 20](#_Toc137939827)

[2.3 Проектирование клиентской части 31](#_Toc137939828)

[ГЛАВА 3 ИСПОЛЬЗОВАНИЕ ПРИЛОЖЕНИЯ 37](#_Toc137939829)

[3.1 Руководство пользователя 37](#_Toc137939830)

[3.2 Планируемый эффект от внедрения 40](#_Toc137939831)

[ЗАКЛЮЧЕНИЕ 44](#_Toc137939832)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 45](#_Toc137939833)

[ПРИЛОЖЕНИЕ А 48](#_Toc137939834)

[ПРИЛОЖЕНИЕ Б 49](#_Toc137939835)

[ПРИЛОЖЕНИЕ В 50](#_Toc137939836)

[ПРИЛОЖЕНИЕ Г 52](#_Toc137939837)

[ПРИЛОЖЕНИЕ Д 56](#_Toc137939838)

[ПРИЛОЖЕНИЕ Е 65](#_Toc137939839)

# **ВВЕДЕНИЕ**

В концепцию Росатома-2030 входит увеличение доли гражданской продукции до 50%. При этом производственная система комбината «Электрохимприбор», последние пятьдесят лет направленная на выполнение гособоронзаказа, не может быть перестроена моментально. Есть множество механизмов и регуляторов, которые необходимы предприятию, чтобы пройти путь от подготовки плана до отгрузки готовой продукции. Создание программного обеспечения, независимого от выбора операционной системы, с использованием открытых технологий становится одним из приоритетов для российских разработчиков.

Следующим шагом стал запуск проекта «Матрица комплектации». Эта система, связывает воедино несколько уже существующих информационных систем, формирует новые связи, структуры и управляющие механизмы, которых до этого момента на предприятии просто не существовало.

Процесс формирования производственного состава и производственного планирования являются важными аспектами управления производством.

Формирование производственного состава:

* Определение необходимых компонентов для изготовления продукции по заказу
* Включает элементы из конструкторского и технологического составов
* Предполагает расчет процесса производства, учитывая трудоемкость операции, последовательность сборки и необходимое количество компонентов по заказу

Производственное планирование:

* Определяет целевые показатели и действия для достижения их с учетом внешних и внутренних факторов
* Включает описание технологического процесса требования к организации производственного процесса, программу производства продукции состав необходимого оборудования.
* План учитывает производственную мощность и требования заказчика.

Дополнительно вводятся страховые циклы для обеспечения точности и надежности производства. В результате формируется «Лист комплектации», который служит точной инструкцией для производства.

«Управление информационных технологий и связи» (УИТиС) ФГУП «Комбинат «Электрохимприбор» отвечает за создание и обслуживание ПО для автоматизации работы предприятия. В рамках проектов Росатома, направленных на снижение санкционных рисков и укрепление информационной безопасности, необходимо обновить систему управления предприятием, используя импортонезависимые компоненты. Использование импортонезависимых компонентов обеспечит надежную защиту критически важного оборудования от функций, которые могут содержаться в зарубежных вычислительных системах, угрожая безопасности данных.

В рамках проекта необходимо реализовать ИТ-поддержку возможностей, которые являются частью системы, функционирующей в отделе 084, с целью повышения эффективности анализа ситуации на производстве.

Объектом исследования выпускной квалификационной работы является создание информационной системы, отражающей состояние системы «Матрица комплектации», которая запущена на ФГУП «Комбинат «Электрохимприбор».

Предметом исследования является процесс отслеживания листов комплектаций на предприятии.

Целью работы является разработка веб-приложения «Анализ выполнения плана запуска сопроводительных паспортов контроль сроков изготовления ключевых узлов в закрытой локально-вычислительной сети», позволяющего пользователям диспетчерского отдела 084 осуществлять следующие функции:

* Просмотр всех листов комплектации по цеху;
* Просмотр потребности листа комплектации;
* Просмотр маршрута ДСЕ;
* Просмотр дерева входящих ПП.

Технические требования устанавливают набор технологий для проектной работы. Разработка приложения должна вестись на Python с применением фреймворков Django и DjangoRest Framework, а также HTML, CSS, JavaScript и Bootstrap5.

Для достижения цели проекта предстоит выполнить ряд задач:

* Анализировать специфику деятельности;
* Оценить текущую систему на предприятии для интеграции и разработать модель бизнес-процессов;
* Создать проект базы данных;
* Разработать пользовательский интерфейс приложения;
* Создать веб-сервис;
* Провести тестирование программного обеспечения;
* Написать руководство для пользователя;
* Оценить результаты внедрения разработки.

# **ГЛАВА 1 ПРЕДПРОЕКТНОЕ ИССЛЕДОВАНИЕ**

## **Анализ предметной области**

На предприятии в данный момент запущенно несколько программных комплексов. На основе данных этих систем программный комплекс «Анализ выполнения плана запуска сопроводительных паспортов контроль сроков изготовления ключевых узлов» будет проводить анализ. Рассмотрим подробно эти комплексы:

МКАДД (ОТ99Р) - Межцеховой Комплекс Анализа Движения Деталей.

Программа ОТ98Р “Межцеховой комплекс анализа движения деталей (МКАДД)” входит в состав комплекса задач по управлению основным производством цеха.

Информация базы данных системы «МКАДД» используется для:

* Формирования и печати сопроводительных паспортов;
* Ведения пооперационного учета выполнения операций;
* Ведения учета готовых деталей и технологических потерь;
* Формирования и печати сдаточных накладных на передачу продукции;
* Диспетчирования прохождения ДСЕ по межцеховому ТМ;
* Ведения учета движения продукции по сдаточным накладным;
* Создания сменного задания рабочему;
* Получения учетной информации для формирования нарядов;
* Получения оперативной информации о ходе производственного процесса;
* Формирования аналитических справок.

AT18V - Справочная система базы данных «Предмет производства». База данных «Предмет производства» является частью системы «Паутина» - Планирование, Анализ, Учет, Техническая подготовка Информации основного производства.

Программа AT18V «Автоматизированная система просмотра технологической базы данных «Предмет производства» предназначена для просмотра БД в рамках системы ПАУТИНА.

AT20P - «Расчет нормы расхода материала на деталь» входит в состав информационного комплекса задач подсистемы «Техническая подготовка производства» и функционирует в корпоративной сети комбината.

Задача предназначена для автоматизированного расчета нормы расхода материала на деталь с целью получения данных о количестве материала для изготовления изделия.

AT20P определяет:

* Чистый вес детали по КД с учетом конкретного материала из ВЗ;
* Норму расхода материала по ВЗ на одну деталь с учетом количества образцов на деталь;
* Коэффициент использования материала (КИМ).

AT43P - «Автоматизированная система ведения операционных трудовых нормативов на базе техпроцессов» входит в состав системы подготовки технологического обеспечения комбината «Поток». Программа AT43P предназначена для:

* Автоматизированного формирования нормокарт на основании техпроцессов;
* Распечатки утвержденных нормокарт;
* Формирования единой информационной базы данных техпроцессов и операционных трудовых нормативов;
* Формирования справочной информации по расценкам;
* Получения справочной информации об истории изменения данных системы на основе накопительного архива.

В программе для одной детали или сборочной единицы (ДСЕ) предусмотрена возможность наличия нескольких нормокарт («архивных», «действующих» и «в разработке»). Одна из «действующих» нормокарт имеет признак «основной» и используется для расчетов трудоемкости и расценки.

Программа допускает формирование блоков операций, выполняемых на разном оборудовании, внутри одной нормокарты.

МР07Р - «Автоматизированная система формирования и обработки электронных заявок на МТР» является частью автоматизированной системы управления предприятием АИС КАСКАД.

Программа МР07Р предназначена для:

* Формирования заявок на МТР от подразделений комбината в электронном виде;
* Утверждение электронных заявок ответственным за тематику;
* Обработки электронных заявок инженерами ОМТСКО;
* Контроль процесса формирования и обработки электронных заявок.

На основе этих программных комплексов составляется следующий перечень данных, по которым будет происходить:

* Сопроводительные паспорта;
* Нормокарты;
* Справочник предметов производства;
* Справочник ведомостей материалов;
* Заявки;
* План производства;
* Справочник операций;
* Система МКАДД.

Система «Анализ выполнения плана запуска сопроводительных паспортов, контроль сроков изготовления ключевых

узлов» на основе этих данных должна помогать отслеживать выполнение плана и проводить анализ ошибок планирования комбината «Электрохимприбор» и предоставлять всю необходимую функциональность:

* просмотр действующих задач отдела;
* отображение статусов ДСЕ;
* отображение заявок на материалы и их поставки;
* построение дерева ДСЕ и учет ключевых узлов;
* всестороннее аналитическое представление.

Данные системы используются в работе специалистами подразделений комбината. Система «Анализ выполнения плана запуска сопроводительных паспортов контроль сроков изготовления ключевых

узлов» физически разделена на несколько частей:

* часть, работающая в PLM-системе комбината;
* часть, работающая в корпоративной сети комбината;
* часть, работающая на защищенных локальных автоматизированных рабочих местах.

В корпоративной сети комбината обрабатывается (просматривается, рассчитывается, анализируется) информация. Данные хранятся на SQL-сервере корпоративной сети, который находится в отделе управления информационных технологий и связи. Выгрузка и загрузка информации на сервер корпоративной сети производится с рабочей станции, подключенной к ней [20].

В качестве места хранения данных была выбрана система витрин данных в PostgreSQL. Подробнее о системе витрин:

Витрины данных в базах данных – это специализированные хранилища, предназначенные для анализа и обработки больших объемов информации, связанных с определенной предметной областью приложения. Они обеспечивают быстрый доступ к данным для аналитических запросов и отчетности, не затрагивая операционные системы [4]. Основные характеристики витрин данных:

* Оптимизация производительности: витрины данных оптимизированы для чтения и анализа, что позволяет выполнять сложные запросы с высокой скоростью.
* Денормализация данных: для ускорения доступа к данным витрины часто содержат денормализованные данные, что уменьшает количество необходимых соединений таблиц.
* Историчность: витрины данных могут хранить исторические данные за длительный период, что важно для анализа трендов.
* Субъектно-ориентированный подход: данные организованы вокруг ключевых объектов предметной области.
* Интеграция данных: витрины интегрируют данные из различных источников, обеспечивая единую точку доступа к информации по предметной области.

Витрины данных играют важную роль в процессе принятия решений, предоставляя инфраструктуру для детального анализа и извлечения значимой информации о состояниях [14].

Система «Анализ выполнения плана запуска сопроводительных паспортов, контроль сроков изготовления ключевых узлов» включает в себя различные типы таблиц:

* Основные таблицы предоставляют информацию о документации, разработанной для деталей или сборочных единиц (ДСЕ), и обеспечивают удобный доступ к необходимым данным.
* Справочные таблицы содержат вспомогательные данные, собранные из внешних систем.
* Специальные таблицы предназначены для хранения данных, специфичных для определенного набора операционных трудовых стандартов. Структура как основных, так и справочных таблиц одинакова во всех аспектах. Основные таблицы системы перечислены в таблице 1.1.

Таблица 1.1 – Основные таблицы системы

| Имя таблицы | Краткое описание |
| --- | --- |
| mtx\_booking | Таблица бронирования – таблица распределения деталей по позициям по- детального плана цеха |
| mtx\_calculation\_unit | Калькуляционная единица – информационный объект управленческого учета системы управления производством, используемый для планирования и сбора прямых фактических затрат |
| mtx\_complectation\_matrix | Матрица комплектации – информационный объект автоматизированной системы управления производством |
| mtx\_demand | Потребность ПП в материалах для производства |
| mtx\_route | Таблица движения ПП по цехам и код операции |
| mtx\_strycture\_analysis | Таблица для хранения итогов функции анализа матрицы |
| mtx\_request | Таблица заказов материалов по МТР для матрицы |
| passport\_for\_matrix | Таблица для расчетов на начало года |

Справочные таблицы системы перечислены в таблице 1.2.

Таблица 1.2 – Справочные таблицы системы

| Имя таблицы | Краткое описание |
| --- | --- |
| mtr | Основной справочник МТР предприятия, содержит информацию, используемую другими информационными системами |
| field\_glossary | Справочник полей для перевода интерфейса |
| ref\_status, | Справочник статусов матрицы |
| nomenclature | Номенклатура – таблица с именем, характеристикой и кодом, а также типоразмером и типом ПП |
| Mtx\_request\_header | Заголовки заявок по матрице |

## **1.2. Описание бизнес-процесса**

Для решения бизнес-задачи необходимо разработать информационную систему, адаптированную к нуждам определенного отдела или функциональной подсистемы предприятия. Это включает в себя:

* Разработку моделей бизнес-процессов или функционирования ресурсов, которые соответствуют архитектуре предприятия и могут быть автоматизированы (полностью или частично) согласно стандартам ФГУП «Комбинат «Электрохимприбор».
* Обоснование выбора программного обеспечения и требований к его настройке или модификации, включая анализ того, насколько функционал выбранного решения соответствует задачам и предложенным моделям бизнес-процессов.
* Описание информационной поддержки бизнес-задачи, в том числе классификацию и кодирование данных, форматы документов, процесс документооборота, а также концептуальные и логические модели баз данных.
* Оценку воздействия предложенного решения на работу организации и ключевые показатели эффективности бизнес-процессов.

В процессе моделирования на предприятии используются различные типы диаграмм:

* Диаграмма цепочки добавленной стоимости (VAD) - для описания взаимодействия процессов на высоком уровне.
* Диаграмма плавательных дорожек (SLD) - для детализации процессов на среднем и нижнем уровне.
* Диаграмма расширенной цепочки процессов, управляемой событиями (eEPC) - для полного описания процессов на всех уровнях.
* Диаграмма организационной структуры (OC) - для визуализации организационных моделей.
* Диаграмма дерева функций (FT) - для представления структуры процессов и функций.
* Диаграмма типов информационных систем (ASTD) - для описания структуры информационных систем.

Чтобы автоматизировать рассматриваемый процесс, заказчик и исполнитель должны однозначно понимать структуру разработки. Модель бизнес-процесса позволяет выстроить полную схему того, как выглядит процесс, и какая его часть подвергнется автоматизации с помощью программного модуля [3].

Бизнес-процесс по функционированию системы «Анализ выполнения плана запуска сопроводительных паспортов, контроль сроков изготовления ключевых узлов» в закрытой локально-вычислительной сети планируется выстроить в нотации «swimlane».

На рисунке 1.3 представлена диаграммы внешнего уровня бизнес-процесса.

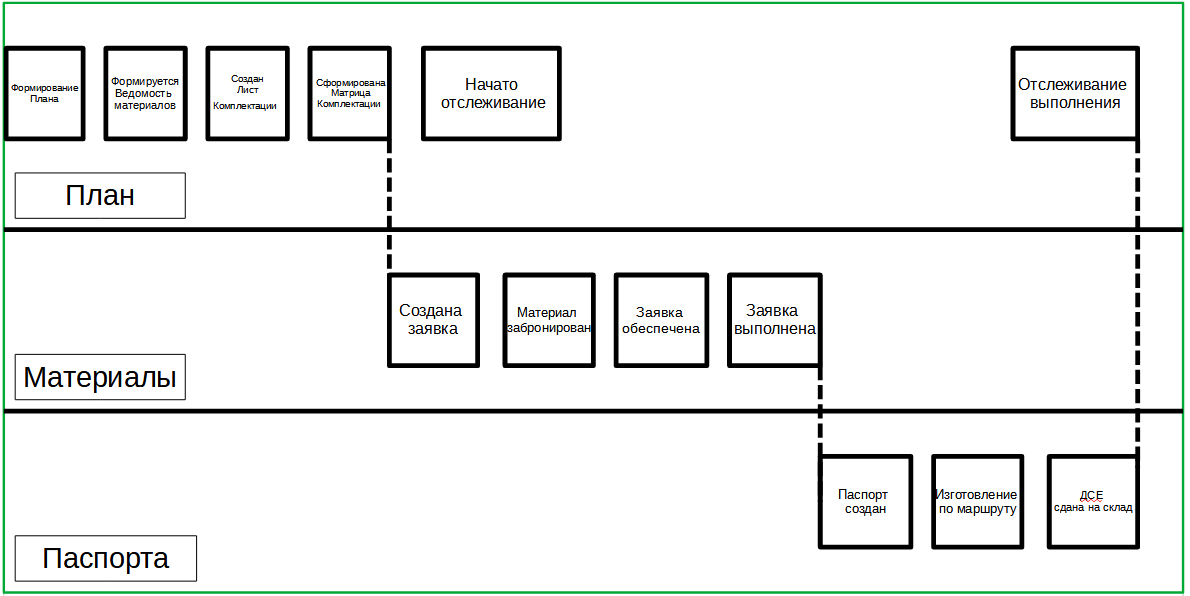
Рисунок 1.3 – Внешний уровень бизнес-процесса

Диаграмма высокого уровня бизнес-процесса представляет собой визуализацию, которая охватывает процесс в общих чертах, избегая детального изложения каждого шага. Этот вид диаграммы иллюстрирует основную структуру процесса и подчеркивает его важнейшие фазы, которые типично присутствуют в рамках процесса.

Такая диаграмма обычно содержит главные этапы процесса, демонстрируя основные действия, необходимые для его завершения. К примеру, это могут быть этапы, связанные с собиранием информации, ее анализом, созданием стратегий и планов, а также выпуском продукции или услуг.

Кроме того, диаграмма высокого уровня может показывать взаимодействие между разными участниками процесса и указывать на ресурсы, необходимые для его успешного исполнения. Часто такая диаграмма служит для краткого обзора процесса и выделения его ключевых особенностей.

Диаграмма на макроуровне бизнес-процесса служит для представления общего облика и ключевых компонентов процесса, включая основные этапы, участников, необходимые ресурсы и их взаимодействие. Этот инструмент является важным для базового понимания структуры и основных аспектов процесса, однако он не предназначен для раскрытия детальной информации и нюансов, которые могут быть подробно изучены на более детализированных уровнях представления процесса. Так, диаграмма на макроуровне дает возможность оценить процесс в целом, выделяя его этапы, участников, ресурсы и взаимосвязи, но не вдается в глубину деталей, доступных на более глубоких уровнях анализа [5].

Диаграмма детализированного уровня бизнес-процесса представляет собой инструмент, который обеспечивает глубокое понимание процесса путем визуализации каждого аспекта и шага в более мелких деталях (Приложение А) в сравнении с диаграммой общего уровня. Этот тип диаграммы раскрывает все этапы процесса, их последовательность и взаимодействия между участниками.

Включая точные описания действий на каждом этапе, диаграмма детализированного уровня может также содержать иллюстрации, текстовые описания или схемы для наглядного представления компонентов процесса. Эта диаграмма используется для тщательного анализа процесса, выявления его уязвимых мест и разработки стратегий улучшения [7].

На диаграмме детализированного уровня могут быть изображены такие элементы как документооборот, перечни задач, методические указания, роли участников процесса и их вклад в его выполнение, ограничения, сроки выполнения и другие важные аспекты [5].

Следовательно, диаграмма внутреннего уровня бизнес-процесса обеспечивает тщательный и детализированный взгляд на процесс, его составляющие и их взаимодействие. Этот инструмент может быть применен для глубокого анализа и усовершенствования процесса внутри организации, что способствует повышению эффективности и удовлетворенности клиентов. Применение нотации «swimlane» предоставляет следующие преимущества:

1. Содействует организации и картографированию бизнес-процессов с учетом потока информации и/или действий между участниками.
2. Дает возможность четко определить ответственность каждого участника за определенный этап процесса, способствуя более эффективному распределению ролей и задач.
3. Облегчает коммуникацию о процессе с коллегами и партнерами благодаря наглядности и понятности визуального представления.
4. Упрощает идентификацию проблемных областей и промежуточных результатов на каждом этапе [6].
5. В сочетании с другими графическими нотациями улучшает понимание связей между различными процессами, делая использование более удобным для пользователя. Таким образом, нотация «swimlane» способствует лучшему осмыслению бизнес-процессов и предоставляет полное представление о последовательности действий и ответственности участников [17].

## **1.3 Средства разработки программного обеспечения**

Технические требования к разработке веб-приложения предписывают использование определенного набора веб-технологий, включая HTML, CSS, JavaScript и Python, а также другое программное обеспечение, не зависящее от импорта.

Серверная сторона:

* Python — это язык программирования высокого уровня с динамической типизацией и автоматическим управлением памяти, предназначенный для широкого спектра задач.
* Django — фреймворк для создания веб-приложений на Python, который следует паттерну проектирования MVC [2].
* MVC (Model-View-Controller) — это архитектурный шаблон, разделяющий данные приложения, пользовательский интерфейс и управляющую логику на три отдельные сущности, каждая из которых может изменяться независимо. В Django этот шаблон реализован как Model-View-Template (MVT), где Model управляет данными, View отвечает за логику отображения пользовательских действий, а Template предназначен для генерации HTML-вывода [15].

DjangoRest Framework (DRF) — библиотека для работы с моделями Django, позволяющая создавать мощные API [9].

API (Application Programming Interface) — набор определений и протоколов для создания и интеграции программного обеспечения.

PostgreSQL — мощная объектно-реляционная система управления базами данных.

Клиентская сторона:

* HTML (HyperTextMarkup Language) — язык разметки для создания веб-страниц, который интерпретируется браузерами [15].
* CSS (Cascading Style Sheets) — язык стилей, используемый для определения внешнего вида документов, написанных на языке разметки.
* JavaScript — гибкий язык программирования, применяемый для создания интерактивных веб-страниц [11].
* Bootstrap — комплект инструментов с открытым исходным кодом для разработки с использованием HTML, CSS и JS, включающий шаблоны дизайна для различных компонентов интерфейса и расширений на JavaScript.

Для этого стека был выбран редактор кода PyCharm. PyCharm позволяет легко редактировать код на разных языках. С его помощью можно быстро создать проект, он подсвечивает синтаксис кода и помогает автоматически править ошибки, занимает достаточно мало места на жестком диске, не потребляет лишние ресурсы системы. Редактор кода представлен на рисунке 1.4.

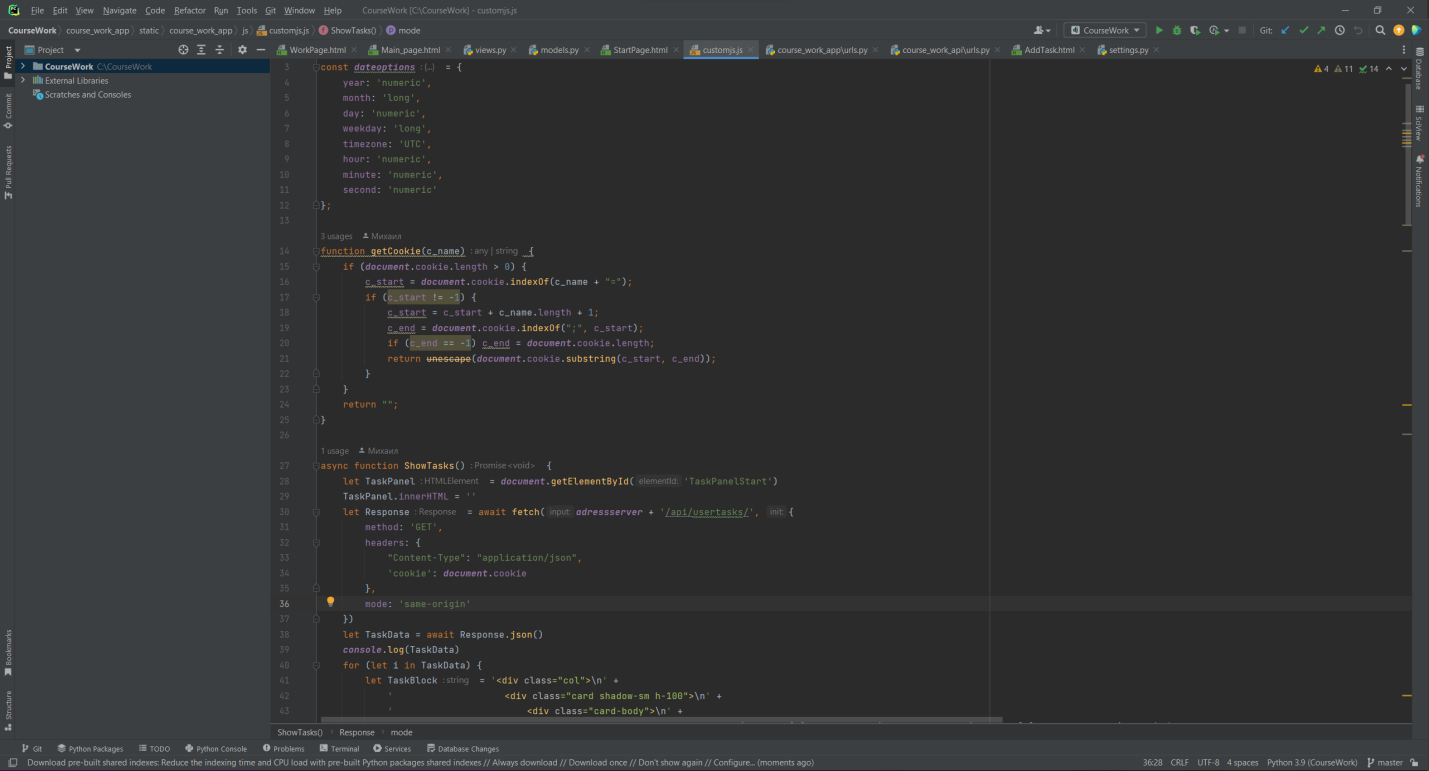


Рисунок 1.4 – Редактор кода PyCharm

Для создания базы данных, использующейся в приложении, была использована стандартная для предприятия и указанная в техническом задании СУБД PostgreSQL. Управление СУБД осуществляется через графическую оболочку PgAdmin 4, которая представлена на рисунке 1.5.

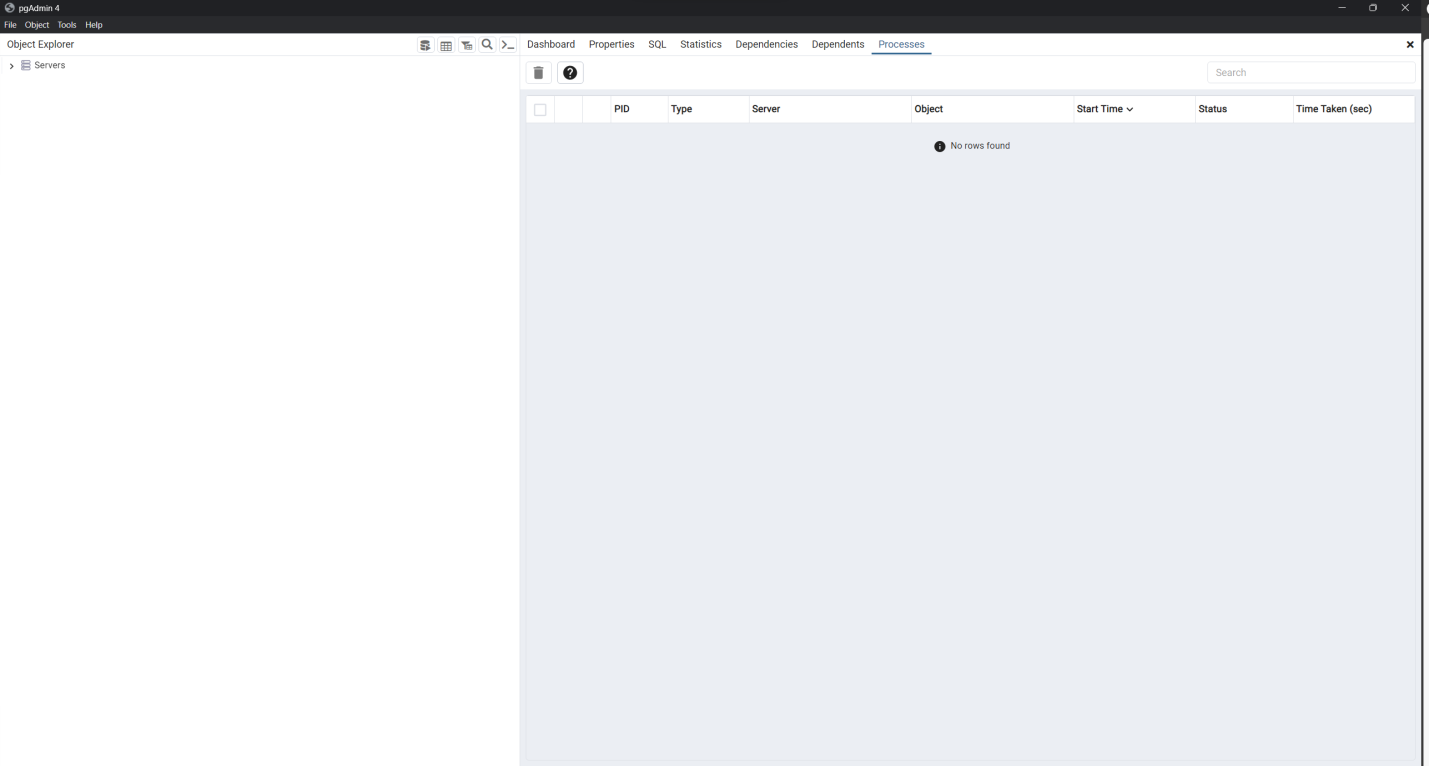


Рисунок 1.5 – PgAdmin 4

Вывод: в соответствии с техническим заданием на разработку программного продукта в данной главе:

1. Проведен анализ предметной области;
2. Построены модели бизнес-процесса;
3. Описаны инструментальные средства.

# **ГЛАВА 2 РАЗРАБОТКА ПРИЛОЖЕНИЯ**

## **2.1** **Построение вспомогательной информационной модели**

На предприятии развернута информационная система отображения актуальной информации «Витрины». Эта система играет важную роль в предоставлении сотрудникам и клиентам актуальных данных о продукции, услугах, новостях и других событиях. Однако для более эффективного управления и принятия решений необходимо разработать аналитическую модель, которая будет использовать данные из этой системы что и будет являться вспомогательной информационной системой.

Рассмотрим концепцию системы «Витрины»:

Витрина данных — это подмножество (срез) хранилища данных, представляющее собой массив тематической, узконаправленной информации, ориентированной, например, на пользователей одной рабочей группы или подразделения [13].

Основные цели выделения витрин:

1. Приближение данных к конечным пользователям;
2. Ограничение только необходимыми данными;
3. Так как витрина представляет специализированную однородную информацию, ее может настроить один специалист в предметной области.
4. Витрина обеспечивает быстрый доступ к необходимым пользователю сведениям.
5. Благодаря относительно небольшому объему данных сокращается время анализа информации на витрине.

Основой аналитической модели послужат следующие объекты базы данных:

Материализованное представление – это дублированная таблица данных, созданная путем объединения данных из нескольких существующих таблиц для ускорения извлечения данных [16].

Преимуществами использования материализованных представлений служат:

1. Скорость: запросы на чтение могут сразу обращаться к материализованному представлению без необходимости каждый раз вычислять новую информацию. Это особенно полезно для сложных запросов.
2. Простота хранения данных: материализованные представления объединяют сложную логику запросов в одной таблице, что упрощает обслуживание кода для разработчиков.
3. Согласованность: материализованные представления обеспечивают согласованное представление данных, собранных в определенный момент. Они также позволяют управлять доступом к данным.

Хранимые процедуры SQL — это мощный инструмент в разработке баз данных. Они позволяют повысить производительность, централизовать бизнес-логику и повторно использовать код [14]. Преимуществами использования материализованных представлений служат:

1. Централизация бизнес-логики: Хранимые процедуры позволяют централизовать бизнес-логику приложения, что упрощает ее понимание и поддержку. Концентрация бизнес-правил в хранимых процедурах обеспечивает единообразное применение изменений в рамках всей организации и помогает избежать расхождений в данных [19].
2. Производительность: Вызов хранимых процедур выполняется быстро и эффективно. Они компилируются один раз и сохраняются в исполняемой форме, что ускоряет выполнение. Исполняемый код также автоматически кэшируется, снижая требования к памяти [18].
3. Упрощение управления кодом: Хранимые процедуры представляют собой набор инструкций, которые выполняются как единое целое. Это упрощает управление кодом, особенно при изменениях в бизнес-логике [10].

Модель «сущность-связь» — это подход к моделированию данных, который позволяет создавать концептуальные схемы для описания предметных областей. Эта модель широко применяется в процессе концептуального проектирования баз данных. “Сущность-связь” представляет собой не просто метод обработки данных, но и способ описания взаимосвязей между элементами системы, известными как “сущности” [16].

В ER-модели информация изображается в форме графической диаграммы, состоящей из трех ключевых элементов: сущностей, атрибутов и связей.

* Сущность –это объект или понятие, которое относится к определенной категории объектов;
* Атрибут – это свойство или характеристика сущности;
* Связь – это отношение, устанавливающее связь между двумя или более сущностями.

Использование ER-модели позволяет разрабатывать сложные информационные системы и определять структуру связей между сущностями. Это облегчает задачу разработчикам при создании эффективных и точных баз данных различной степени сложности. ER-модель также используется для проектирования логической структуры базы данных. На рисунке 2.1 изображена информационная модель «сущность-связь» и находится в Приложении Б.

Рисунок 2.1 – Физическая модель таблицы complectation\_matrix\_analysis

## **2.2 Проектирование и создание вспомогательной информационной системы**

Создание аналитической функции для вспомогательной информационной системы началось с создания таблицы для хранения результатов выполнения функции анализа выполнения плана.

Основные поля таблицы:

* cma\_id– первичный ключ результата анализа;
* mtx\_id – внешний ключ матрицы комплектации;
* diff\_start – разница запуска детали с планируемой датой запуска (включает в себя разницу запуска входящих деталей в случае отсутствия запуска детали);
* prev\_diff\_start – разница запуска входящей детали (наиболее опаздывающая);
* diff\_release – разница выпуска детали с планируемой датой выпуска (включает в себя разницу выпуска входящей детали в случае отсутствия запуска детали);
* prev\_diff\_release – разница выпуска входящей детали (наиболее опаздывающая);
* ts\_chg – дата выполнения расчетов.

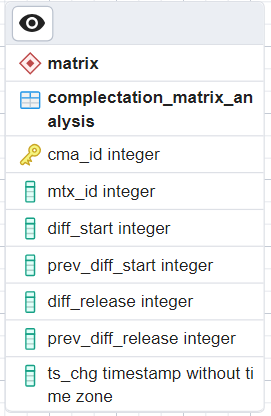


Рисунок 2.1 – Физическая модель таблицы complectation\_matrix\_analysis

Также для осуществления функции анализа выполнения плана требуется создать несколько материализованных представлений.

Материализованное представление pas\_min\_max\_view содержит в себе выборку запуска-выпуска сопроводительных паспортов на детали. Выборка заключатся в том, что из всех дат паспортов выбираются наименьшие по срокам даты запуска и проверки паспорта (в случае отсутствия даты контроля используется дата печати паспорта) [12].

Основные поля материализованного представления:

* mtx\_id– первичный ключ матрицы комплектации;
* min\_pas\_date\_print– наименьшая дата печати/проверки паспорта;
* max\_pas\_date\_control–наибольшая дата печати/проверки паспорта;

Листинг 2.1 – Материализованное представление pas\_min\_max\_view

create materialized view matrix.pas\_min\_max\_view as

SELECT mb.mtx\_id,

min(COALESCE(pfm.pas\_date\_print, pfm.pas\_date\_control::timestamp without time zone)) AS min\_pas\_date\_print,

max(pfm.pas\_date\_control) AS max\_pas\_date\_control

FROM sc\_mtx.mtx\_booking mb

JOIN sc\_mkd.passport\_for\_matrixpfm ON mb.pas\_id = pfm.pas\_id

GROUP BY mb.mtx\_id;

Материализованное представление mtx\_booking\_sum\_view содержит сумму выпущенных и запущенных деталей по всем паспортам. Необходима для прогноза выполнения плана и контроля запуска сопроводительных паспортов.

Основные поля материализованного представления:

* mtx\_id– первичный ключ матрицы комплектации;
* booking\_quanity\_start– количество всех запущенных паспортов;
* booking\_quanity\_release–количество выпущенных паспортов;

Листинг 2.2 – Материализованное представление mtx\_booking\_sum\_view

create materialized view sc\_mtx.mtx\_booking\_sum\_view as

SELECT mtx\_booking.mtx\_id,

sum(mtx\_booking.mtb\_quantity\_start) AS booking\_quanity\_start,

sum(mtx\_booking.mtb\_quantity\_release) AS booking\_quanity\_release

FROM sc\_mtx.mtx\_booking

GROUP BY mtx\_booking.mtx\_id;

Для функционирования системы «Анализ выполнения плана запуска сопроводительных паспортов, контроль сроков изготовления ключевых узлов» была создана функция анализа матрицы комплектации и системы МКАДД, основанная на построении дерева входимости, которая основана на дереве комплектации. Рассмотрим основы расчетов этой функции.

В данной функции используется структура данных, известная как дерево, которое представляет собой связный и ациклический граф. Это означает, что между любыми двумя узлами существует путь, но при этом нет замкнутых циклов. Из этих свойств следует, что в дереве всегда будет на одно ребро меньше, чем узлов, и только один уникальный путь соединяет любую пару узлов [8].

Чтобы пройти через все узлы дерева, был разработан специальный метод, основанный на рекурсивном обходе в прямом порядке. Этот метод подразумевает движение от корня к листьям, при котором после посещения узла происходит переход к его потомкам и так продолжается до тех пор, пока не будут исследованы все ветви дерева. Во время обхода дерева рассчитывается разница между планируемой датами запуска-выпускаи датами запуска-проверки паспортов на деталь [3].

Функция анализа далее записывает данные в complectation\_matrix\_analysisпри отсутствии собственного запуска паспорта детали функция записывает в запуск детали – выпуск наиболее позднего по выпуску входящего.

Листинг 2.3 – Функция analysiscm

BEGIN

FOR current\_id IN

SELECT mcb.mtx\_id

from sc\_mtx.mtx\_bookingmcb

inner join matrix.pas\_min\_max\_viewpmmv on mcb.mtx\_id = pmmv.mtx\_id

group by mcb.mtx\_id

LOOP

SELECT CASE

WHEN (pmmv.min\_pas\_date\_print IS NOT NULL AND mcm.mtx\_start\_date IS NOT NULL)

THEN DATE(pmmv.min\_pas\_date\_print) - mcm.mtx\_start\_date

ELSE 0 END,

CASE

WHEN (pmmv.max\_pas\_date\_control IS NOT NULL AND mcm.mtx\_release\_date IS NOT NULL)

THEN DATE(pmmv.max\_pas\_date\_control) - mcm.mtx\_release\_date

ELSE 0 END

INTO current\_start\_diff,current\_release\_diff

from sc\_mtx.mtx\_complectation\_matrix mcm

inner join matrix.pas\_min\_max\_viewpmmv on mcm.mtx\_id = pmmv.mtx\_id

where mcm.mtx\_id = current\_id

GROUP BY mcm.mtx\_id, min\_pas\_date\_print, max\_pas\_date\_control, mcm.mtx\_start\_date, mcm.mtx\_release\_date;

WITH RECURSIVE tree AS (select mls.cls\_id,

mls.cls\_parent\_cls\_id,

current\_start\_diff AS diff\_s,

current\_release\_diff AS diff\_r

from

sc\_mtx.mtx\_complectation\_matrix mcm

INNER join sc\_mtx.mtx\_complectation\_list\_structuremls on mls.cls\_id = mcm.cls\_id

where mcm.mtx\_id = current\_id

UNION ALL

select mls.cls\_id,

mls.cls\_parent\_cls\_id,

CASE

WHEN (pmmv.min\_pas\_date\_print IS NOT NULL AND mcm.mtx\_start\_date IS NOT NULL)

THEN DATE(pmmv.min\_pas\_date\_print) - mcm.mtx\_start\_date

ELSE 0 END,

CASE

WHEN (pmmv.max\_pas\_date\_control IS NOT NULL AND

mcm.mtx\_release\_date IS NOT NULL)

THEN DATE(pmmv.max\_pas\_date\_control) - mcm.mtx\_release\_date

ELSE 0 END

from sc\_mtx.mtx\_complectation\_matrix mcm

inner join matrix.pas\_min\_max\_viewpmmv on mcm.mtx\_id = pmmv.mtx\_id

join sc\_mtx.mtx\_complectation\_list\_structuremls on mls.cls\_id = mcm.cls\_id

join tree t ont.cls\_id = mls.cls\_parent\_cls\_id)

select MIN(diff\_r),diff\_s

INTO release\_diff, start\_diff

FROM tree GROUP BY diff\_s;

После создания функции было разработано представление в серверной части приложения, которое формирует из таблицы JSON дерево.

Листинг 2.4 –представление MatrixTreeSearch

class MatrixTreeSearch(APIView):

page\_query\_param = 'page'

def get(self, request, mtx\_id):

with connections['cscd'].cursor() as cursor:

cursor.callproc('matrix.get\_matrix\_tree', [mtx\_id])

nodes = self.dictfetchall(cursor)

def build\_tree(node, nodes):

children = [child for child in nodes if child['cls\_parent\_cls\_id'] == node['cls\_id']]

node['children'] = [build\_tree(child, nodes) for child in children]

node['data'] = {}

return node

roots = [node for node in nodes if node['cls\_parent\_cls\_id'] == nodes[0]['cls\_id']]

forest = [build\_tree(root, nodes) for root in roots]

return Response(forest)

def dictfetchall(self, cursor):

columns = [col[0] for col in cursor.description]

return [dict(zip(columns, row)) for row in cursor.fetchall()]

Для работы серверной части приложения были настроены файлы settings.py и были включены разработанные приложения в список включенных приложений [1]. Содержимое конфигурационного файла представлено на рисунке 2.2.

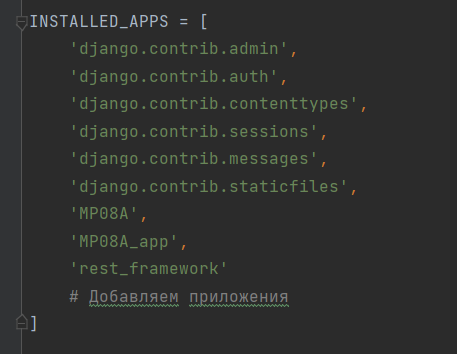


Рисунок 2.2 – Список приложений

Настройка подключения к базе данных устанавливается в settings.py [2] путем выбора двигателя связи с баззой данных и настройкой порта и адресса подключения (рисунок 2.3).

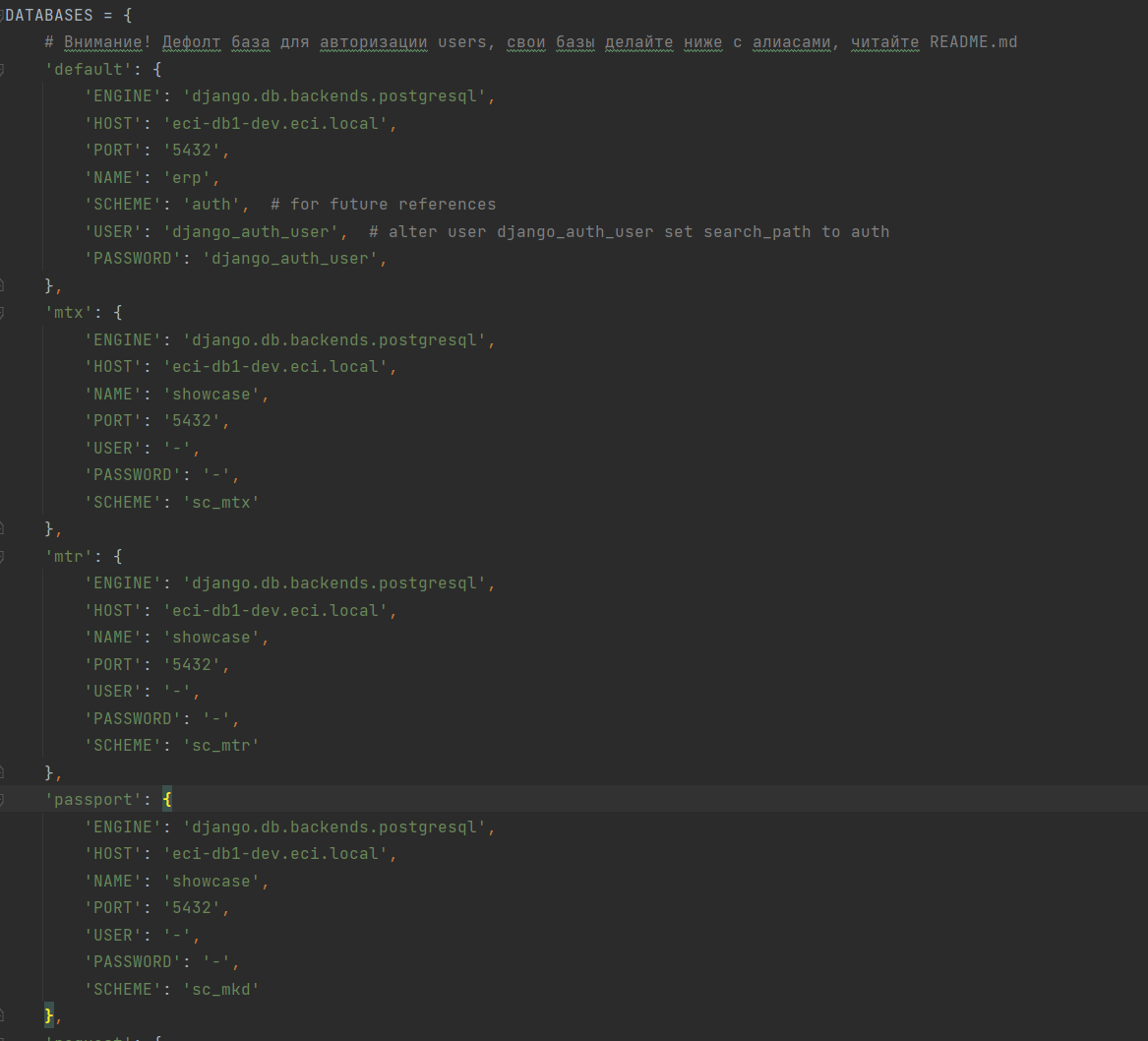


Рисунок 2.3 – Связь с базой данных

Модели описываются в файле в файле models.py. На рисунке 2.4 представлен пример описания моделей для справочников [14].

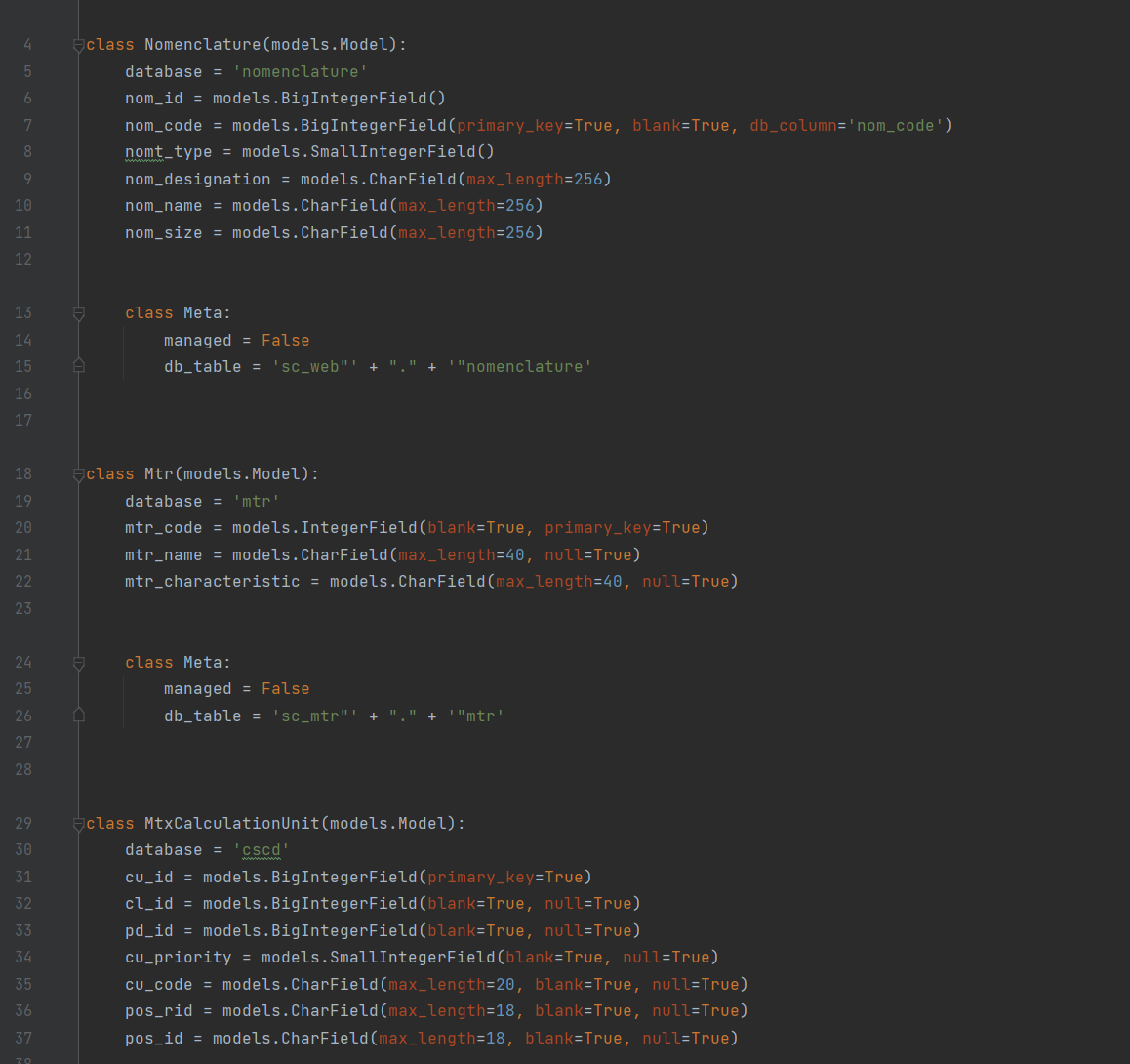


Рисунок 2.4 – Пример описания моделей для справочника

## **2.3 Проектирование клиентской части**

После разработки базы данных был создан прототип интерфейса веб-приложения. На рисунке 2.5 демонстрируется экран входа в систему. Данный экран используется исключительно в контексте выпускной квалификационно работы. В случае интеграции проекта в корпоративную информационную систему будет использоваться стандартная политика авторизации, установленная на предприятии [11].

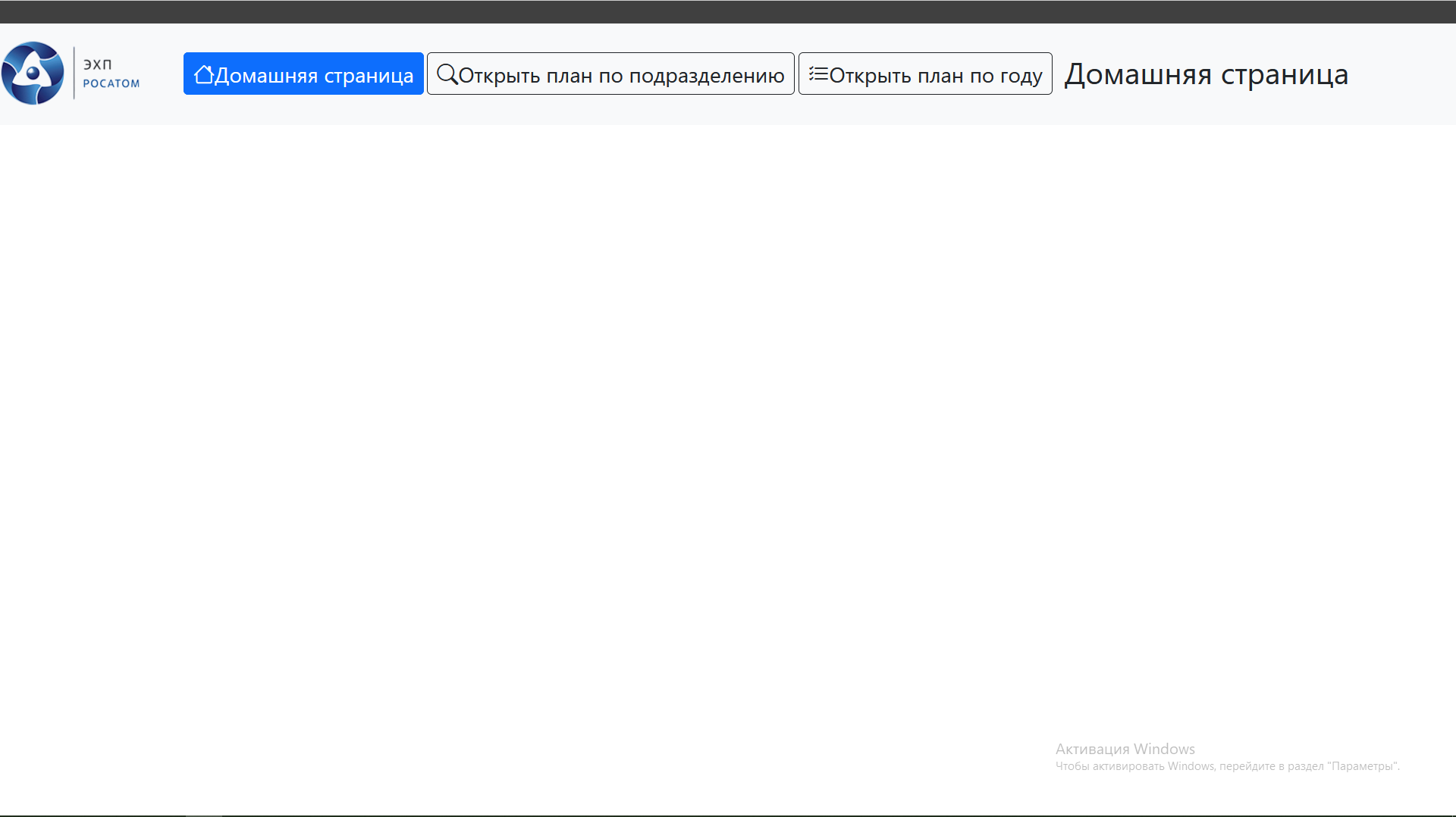


Рисунок 2.5 – Страница интерфейса

На странице плана по году представлена таблица плана по выбранному году с отображением готовности в виде кружочка с цветовой индикацией и возможности поиска для открытия подетального плана (рисунок 2.6 , 2.7).

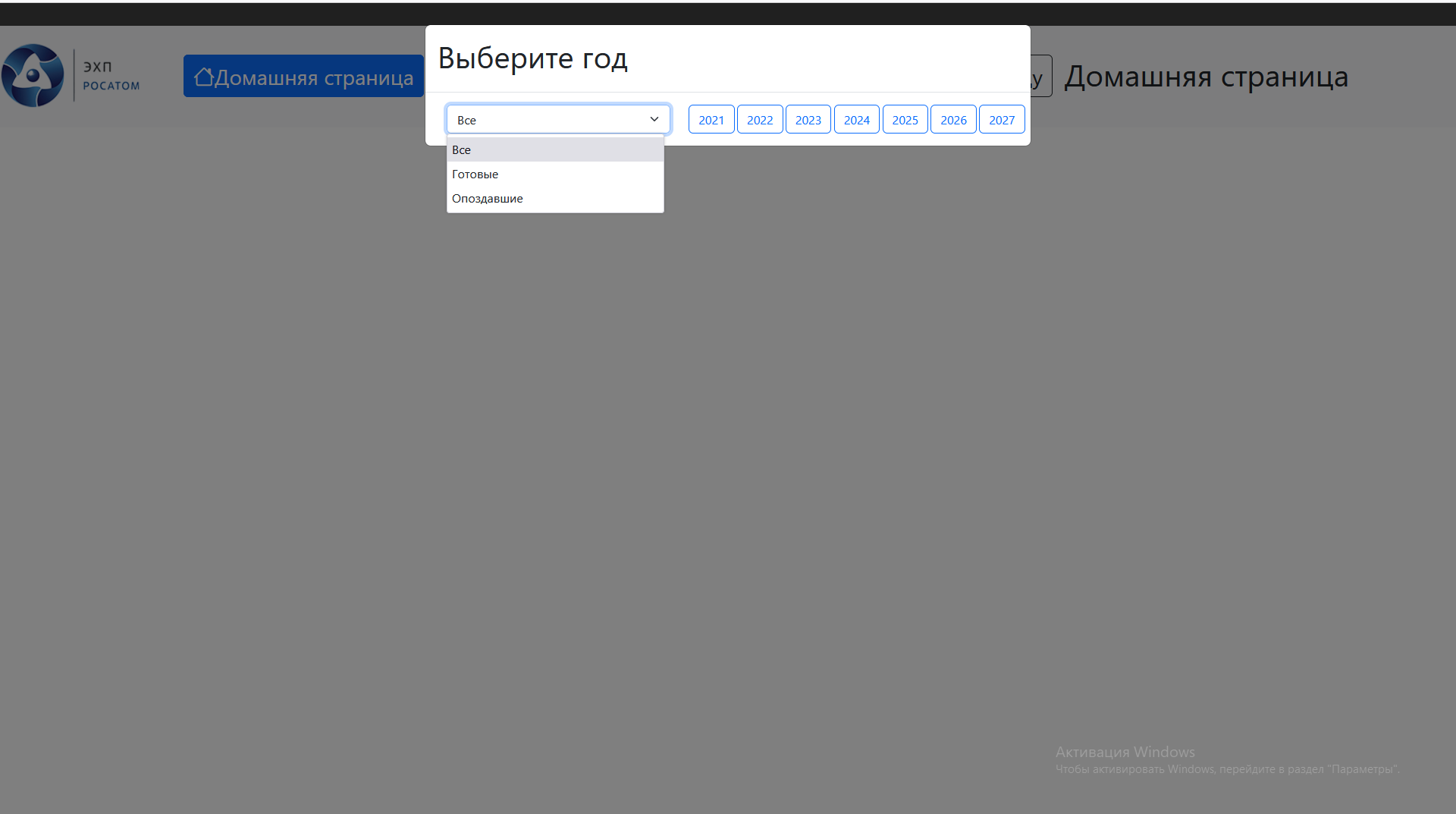


Рисунок 2.6 – Выбор года просмотра плана

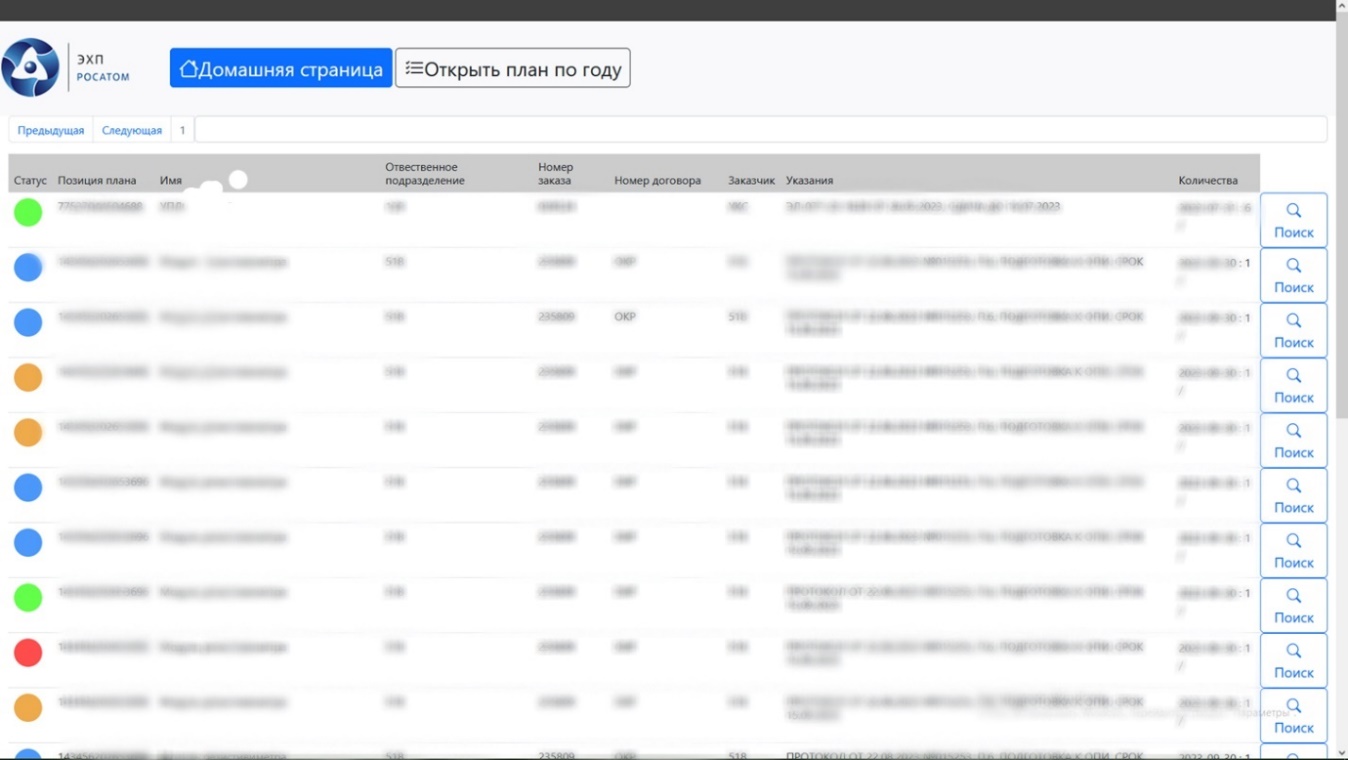


Рисунок 2.6 – Страница плана по году

Выполнив поиск по объекту плана, открывается страница подетального плана с цветовой индикацией готовности детали. Реализация представлена на рисунке 2.7.

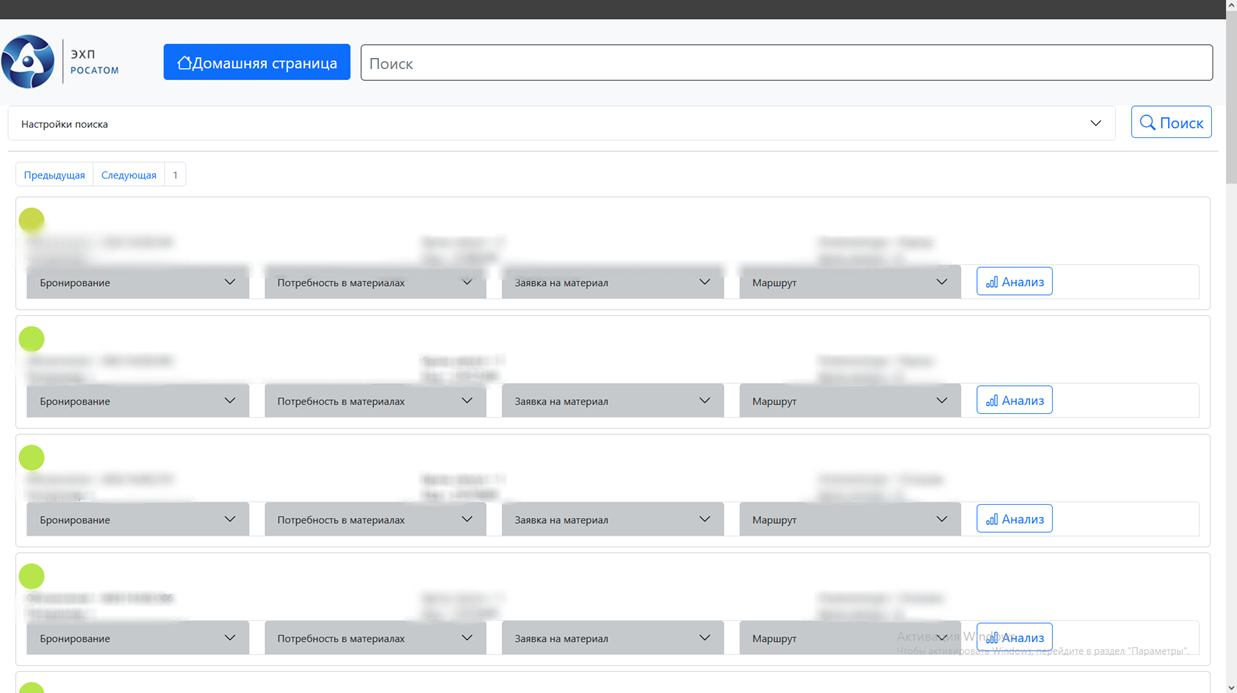


Рисунок 2.7 – Страница подетального плана

На рисунке 2.8 представлена реализация настройки вывода информации (Отключение полей, фильтр по ключевым узлам) подетального плана на страницу (Приложение Е).

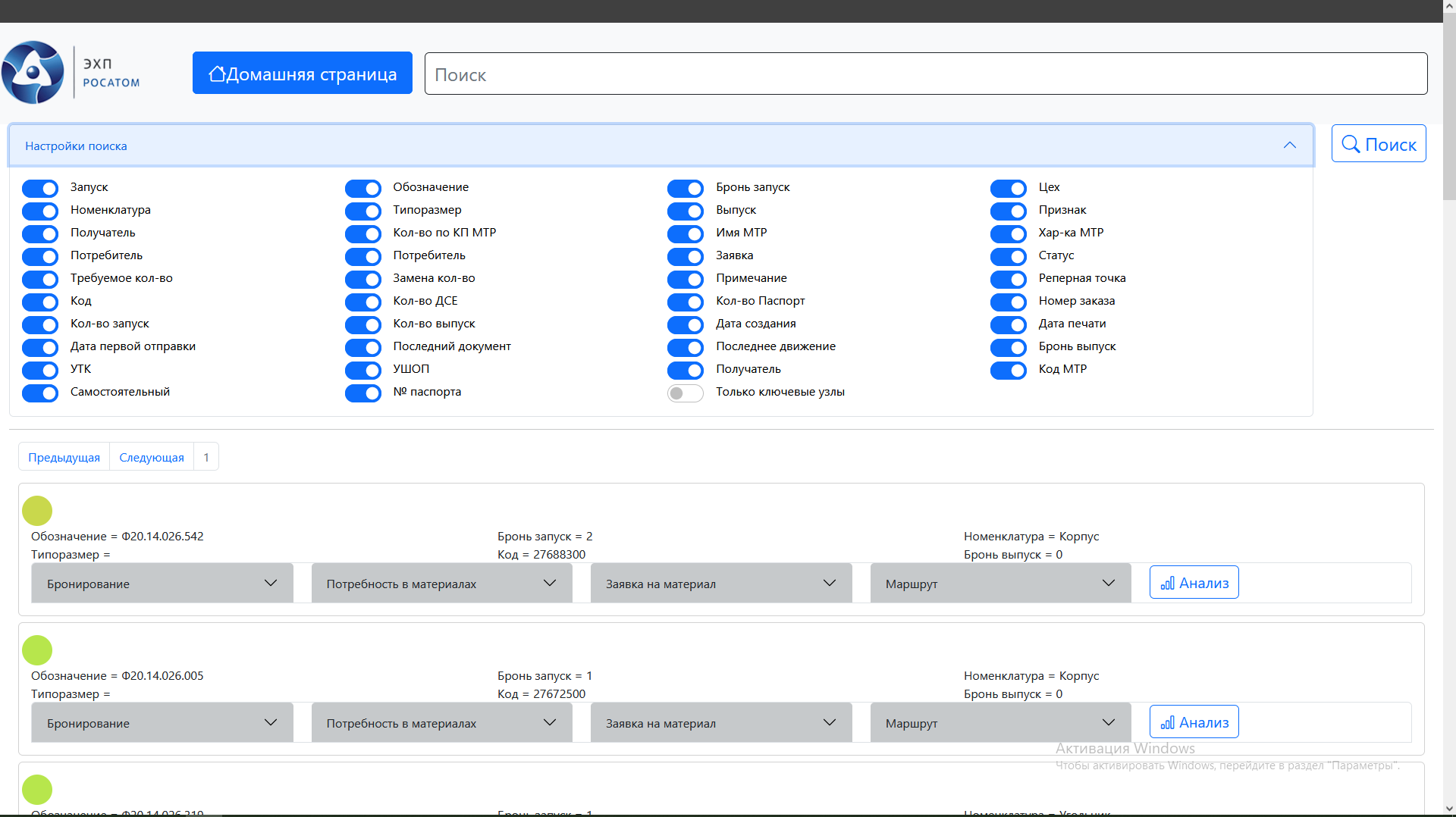


Рисунок 2.8 – Фильтрация подетального плана

На рисунке 2.9 представлена реализация отображения подробной информации по детали: бронирование – список паспортов, входящих в деталь; потребность в материалах – список необходимых материалов; заявка на материалы – статус заказов на материалы для детали; маршрут – движение детали по цехам с УШОП.

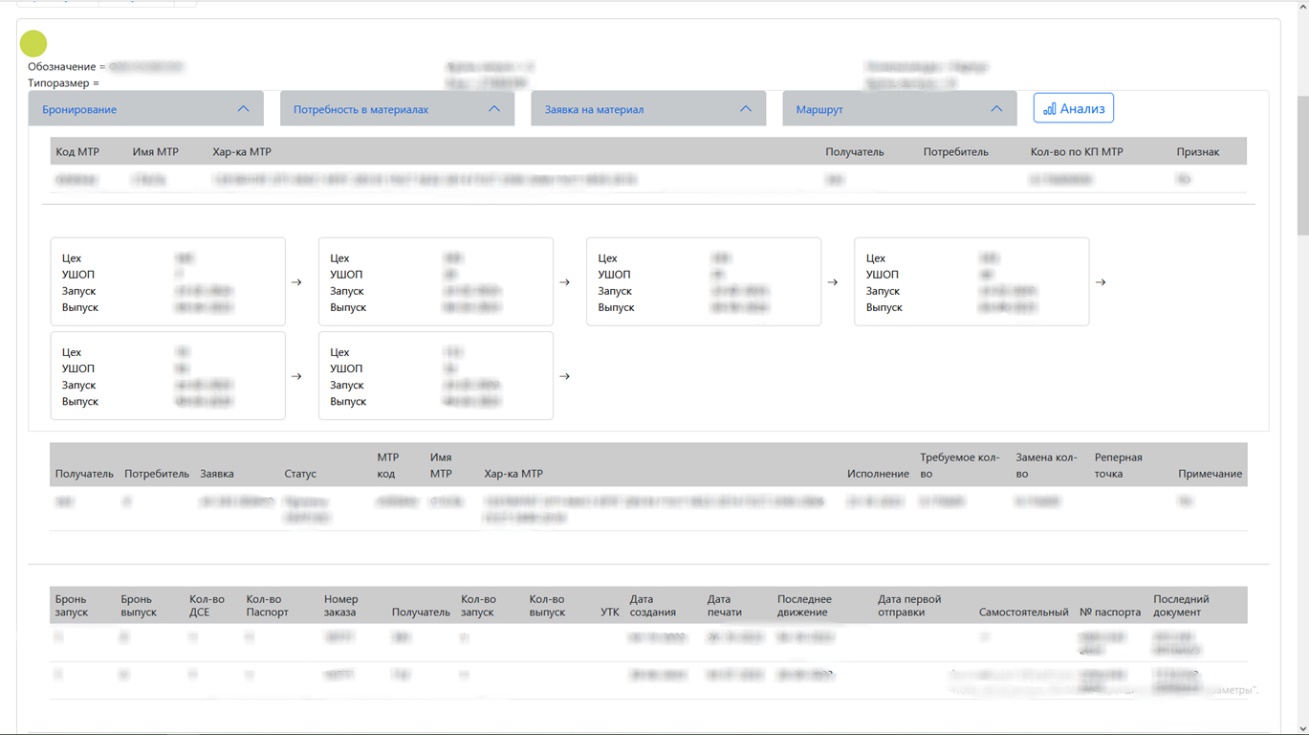


Рисунок 2.9 – Подробнее о детали

На рисунке 2.10 представлена страница анализа матрицы комплектации с построенным деревом комплектации и анализом готовности.

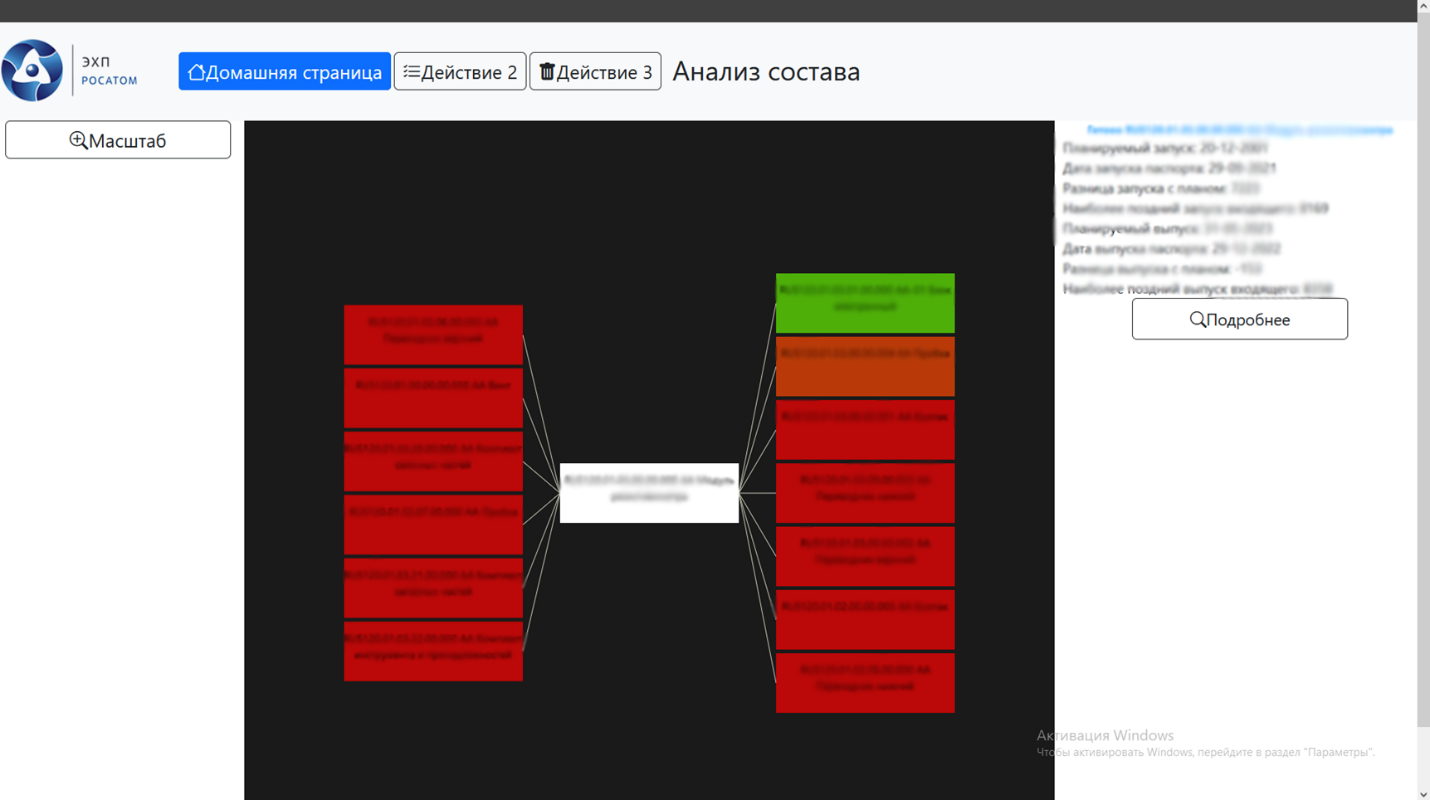


Рисунок 2.10 – Справочник операций

Вывод: в соответствии с техническим заданием на разработку программного продукта в данной главе:

1. Построена информационная модель «сущность-связь»;
2. Спроектирована база данных;
3. Разработана клиентская часть приложения.

# **ГЛАВА 3 ИСПОЛЬЗОВАНИЕ ПРИЛОЖЕНИЯ**

## **3.1 Руководство пользователя**

Программное обеспечение для управления операционными трудовыми стандартами в защищенной локальной вычислительной сети является приложением, которое включает в себя клиентскую часть, работающую как веб-сервис непосредственно в интернет-браузере пользователя [11].

Для корректной работы веб-сервиса «Анализ выполнения плана запуска сопроводительных паспортов, контроль сроков изготовления ключевых узлов» необходимо следующее минимальное оборудование:

Процессор: от Pentium II;

ОЗУ: минимум 4 Гб;

Свободное место на жестком диске: не менее 1 Гб;

Операционная система: Astra Linux.

Клиентская часть программы не требует установки, так как она доступна через веб-сервис. Для запуска программы достаточно перейти по ссылке, предоставленной системным администратором.

Основная страница: после входа в приложение откроется страница с выбором режима отображения. Внешний вид страницы представлен на рисунке 3.1

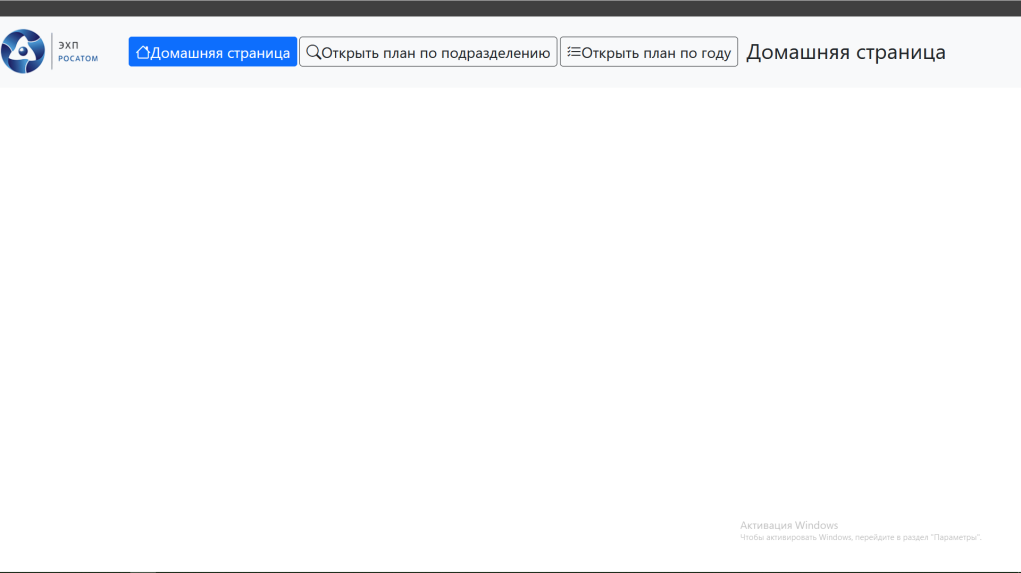


Рисунок 3.1 – Основная страница

На странице присутствуют следующие варианты отображения (рисунок 3.3):

* Вывод плана по году все (рисунок 3.2);
* Вывод плана по году готовые (рисунок 3.2);
* Вывод плана по году неготовые (рисунок 3.2);
* Вывод подетального плана по подразделению.

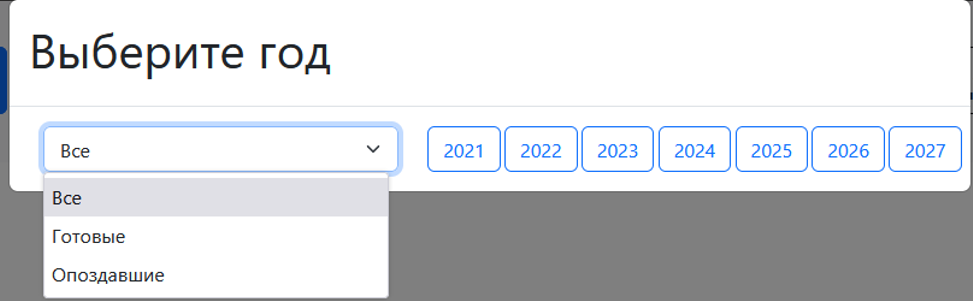


Рисунок 3.2 – Основные функции

При выборе года и фильтра плана открывается страница с планом (рисунок 3.3). На странице присутствует следующий функционал:

* Отображение готовности позиции плана;
* Отображение подробной информации по позиции плана;
* Поиск позиции плана в подетальном плане;

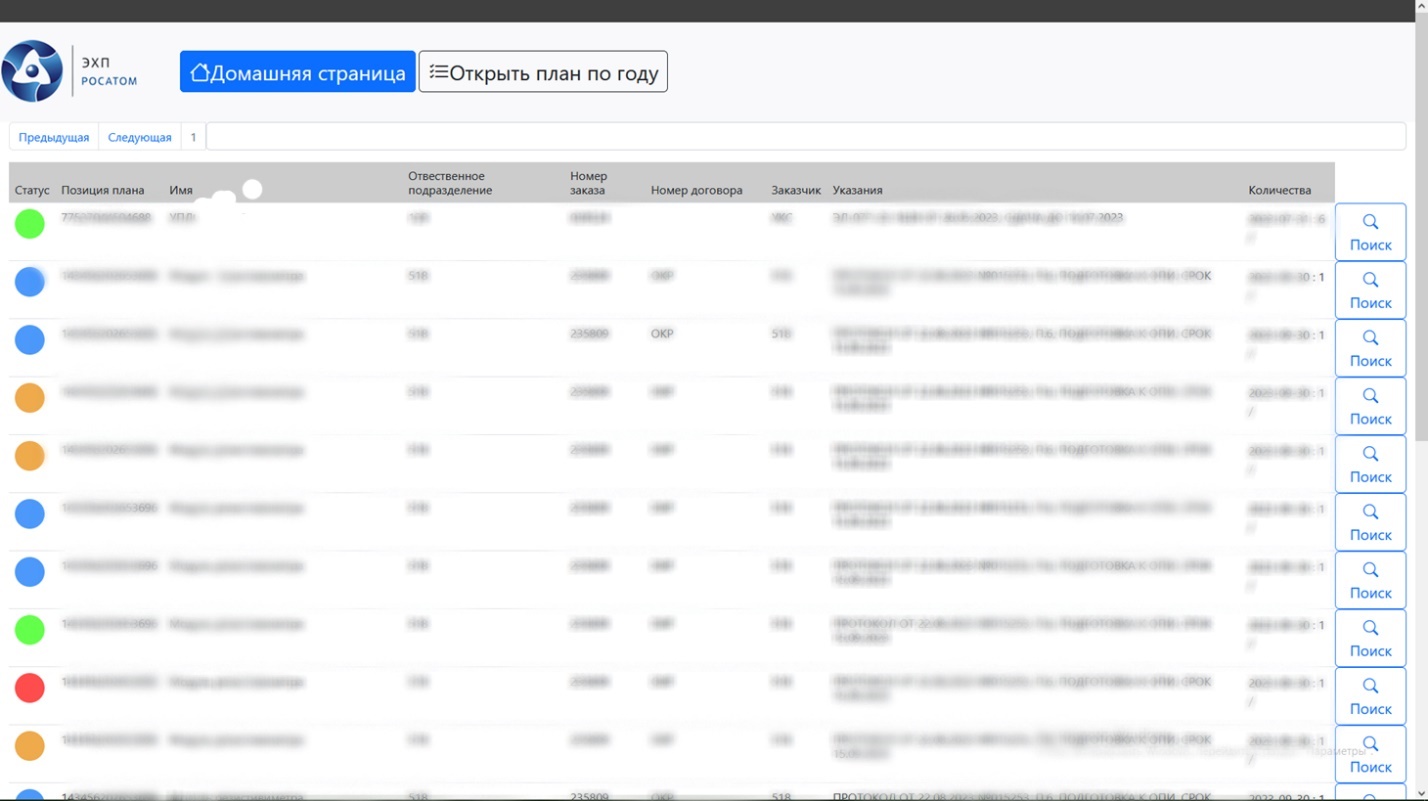


Рисунок 3.3 – Страница плана

После поиска в плане открывается страница с подетальным планом. На странице присутствует следующий функционал:

* Отображение готовности позиции подетального плана;
* Отображение подробной информации по позиции подетального плана;
* Поиск позиции подетального плана
* Настройка отображения
* Поиск подробной связанной информации: бронирование, потребность в материалах, заявки на материалы, маршрут.
* Проведение анализа комплектации на готовность.

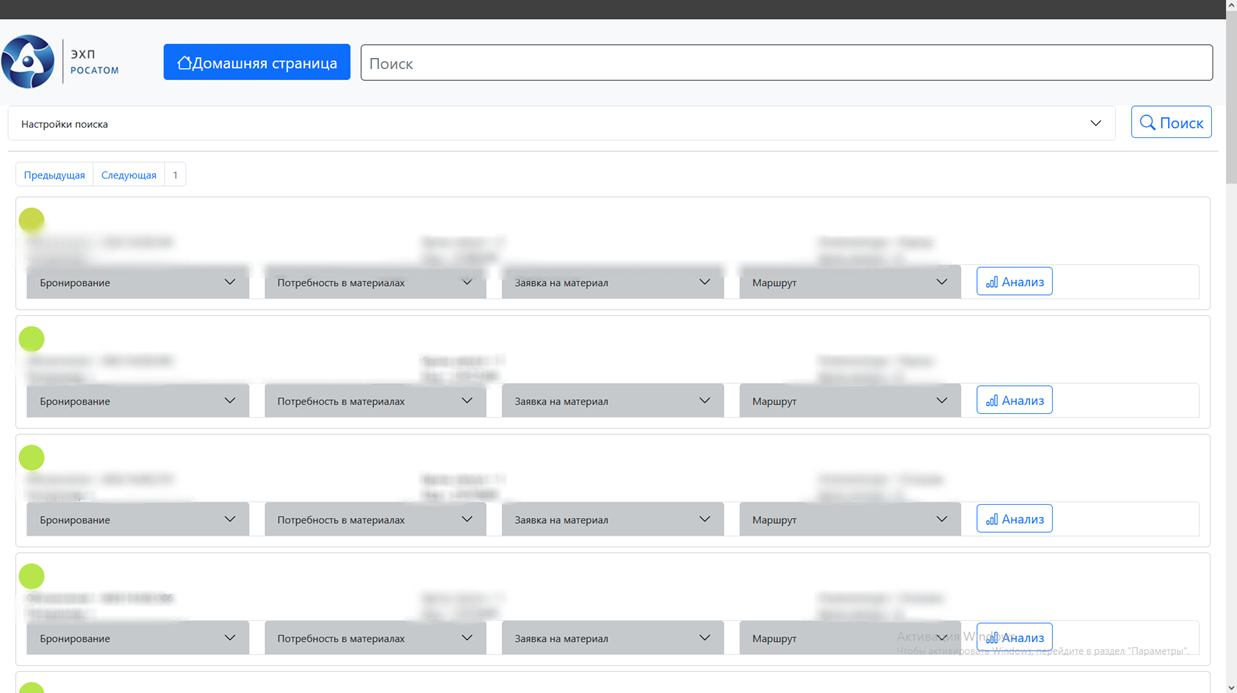


Рисунок 3.4 – Страница подетального плана

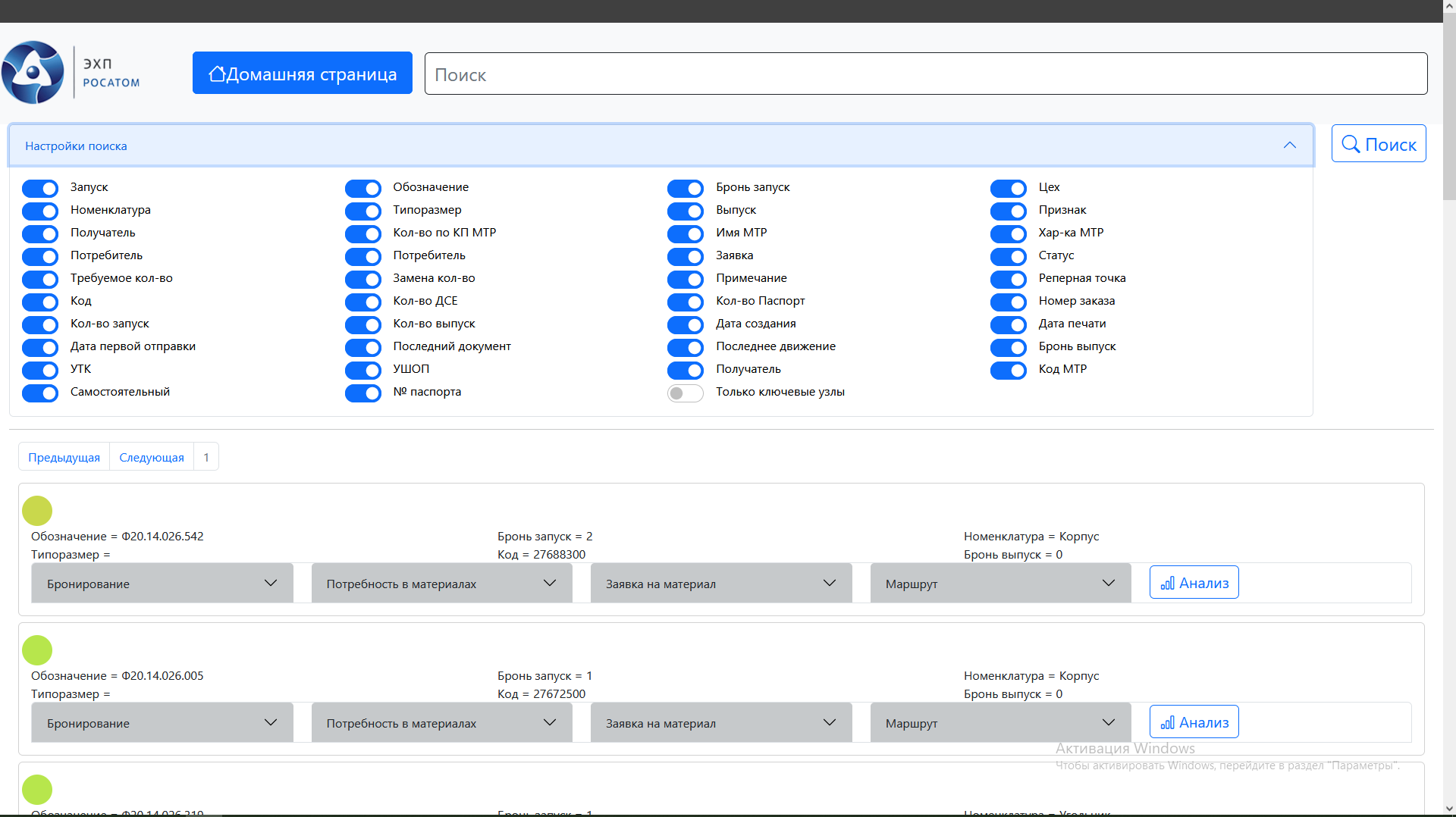


Рисунок 3.5 – Настройка отображения подетального плана и его поиск

Открытие детальных разделов отображает связанную с деталью информацию (рисунок 3.6).

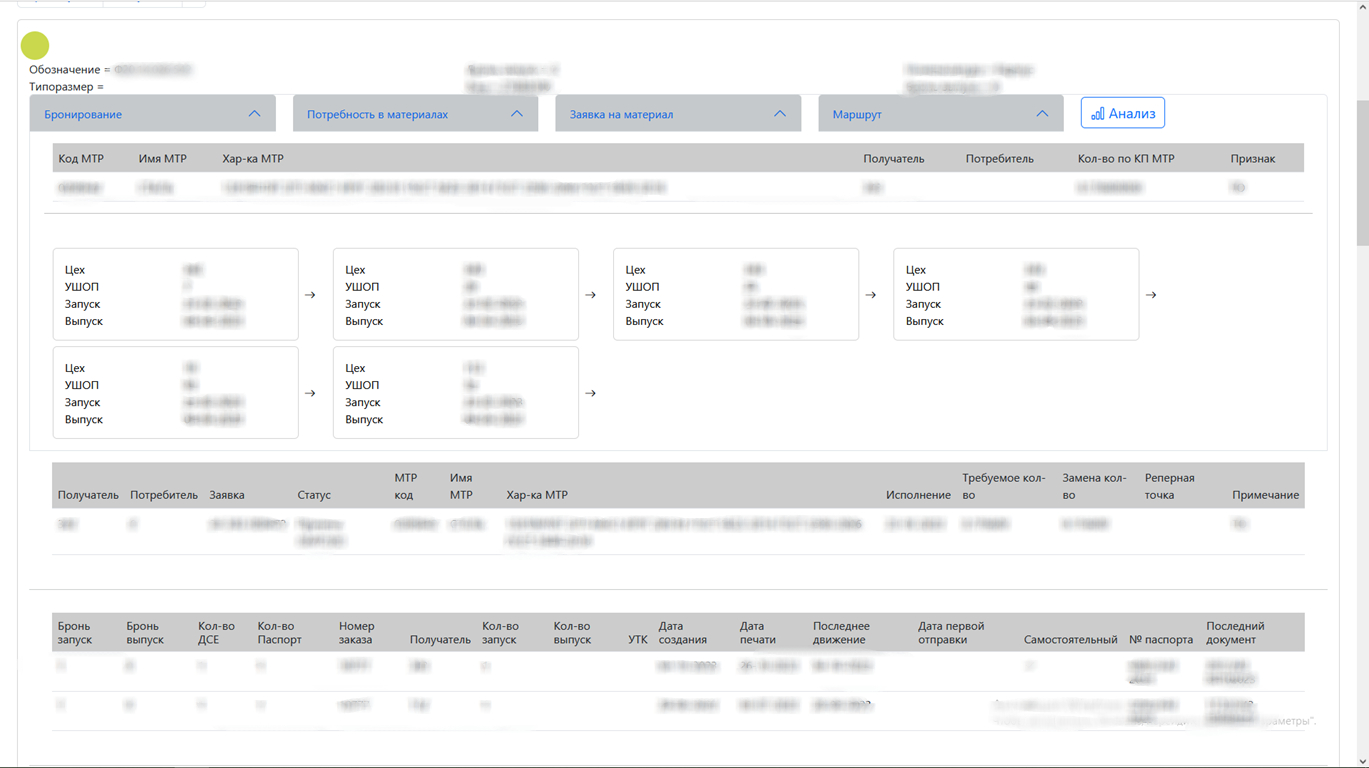


Рисунок 3.6 – Подробная информация по записи в подетальном плане

Кнопка «Анализ» вызывает страницу с деревом комплектации и информацией о входящих деталях с отображением статуса готовности.

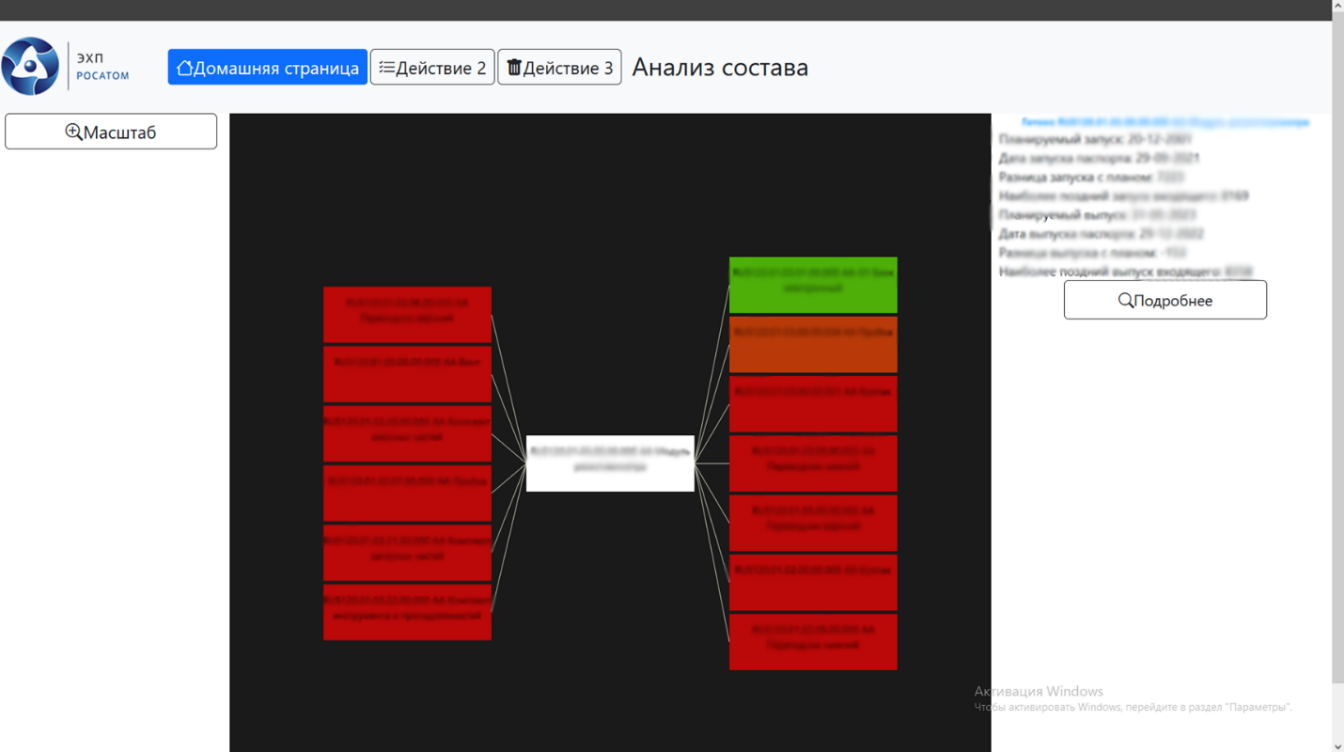


Рисунок 3.7 – Меню справочники

Остальные функции программы не отличаются по процедурам добавления, изменения и также интуитивно понятны.

## **3.2 Планируемый эффект от внедрения**

Внедрение автоматизированных систем управления бизнес-процессами может оказать значительное влияние на экономику компании. Основное преимущество такой автоматизации — это уменьшение расходов за счет сокращения необходимости в ручной работе. Это позволяет ускорить выполнение рутинных и трудоемких задач, снижая при этом риск ошибок и несоответствий [12].

Автоматизация способствует повышению производительности и эффективности, ускоряя процессы и улучшая качество работы, что может привести к увеличению объемов производства и сокращению издержек.

Автоматизация также помогает компаниям упрощать и оптимизировать свои операции, улучшая управление процессами, уменьшая сложность и увеличивая гибкость. Это может привести к сокращению числа сотрудников и, как следствие, к экономии на зарплате и социальных выплатах.

Хотя автоматизированные системы не являются непосредственным источником дохода, они могут служить инструментом для увеличения прибыли или снижения затрат. Основная выгода от автоматизации заключается в улучшении экономических показателей компании, включая экономию трудовых и финансовых ресурсов благодаря:

* уменьшению времени, необходимого для выполнения расчетов;
* сокращению времени на поиск и подготовку документации;
* экономии на расходных материалах, таких как бумага, носители информации и картриджи;
* сокращению числа административных работников.

Стоит подчеркнуть, что снижение трудозатрат достигается за счет автоматизации документооборота и уменьшения времени на поиск информации.

Таким образом, автоматизация бизнес-процессов предоставляет компаниям возможность не только сократить затраты, но и повысить общую эффективность работы, что способствует увеличению прибыли.

Применение этого инструмента автоматизации может потребовать изменений в бизнес-процессах, поскольку задачи будут выполняться быстрее. Это позволяет пользователям эффективнее обрабатывать большие объемы данных в течение рабочего дня, что может привести к сокращению расходов на персонал или к ускоренному росту бизнеса без увеличения числа сотрудников, занятых обработкой информации.

Заключение: согласно техническому заданию на создание программного продукта, в данном разделе:

1. Создано руководство для пользователей;
2. Произведена оценка ожидаемого эффекта от его внедрения.

# **ЗАКЛЮЧЕНИЕ**

В ходе разработки проекта был проведен анализ специализированной области, ключевых терминов, инструментов и целевых пользователей. Произошло ознакомление с предприятием ФГУП «Комбинат «Электрохимприбор» и его ключевыми бизнес-процессами. Были созданы модели бизнес-процессов и информационная модель типа «сущность-связь», что подтвердило необходимость разработки приложения для управления задачами пользователей системы.

Для функционирования приложения была создана база данных, а также необходимые хранимые процедуры и клиентская часть, включающая HTML-документы. В результате работы над выпускной квалификационной работой было создано прикладное приложение, независимое от импорта и основанное на веб-технологиях, обладающее всеми требуемыми бизнес-функциями.

Информационную систему под названием «Анализ выполнения плана запуска сопроводительных паспортов и контроль сроков изготовления ключевых компонентов» предстоит интегрировать в PLM-систему ФГУП «Комбинат «Электрохимприбор».

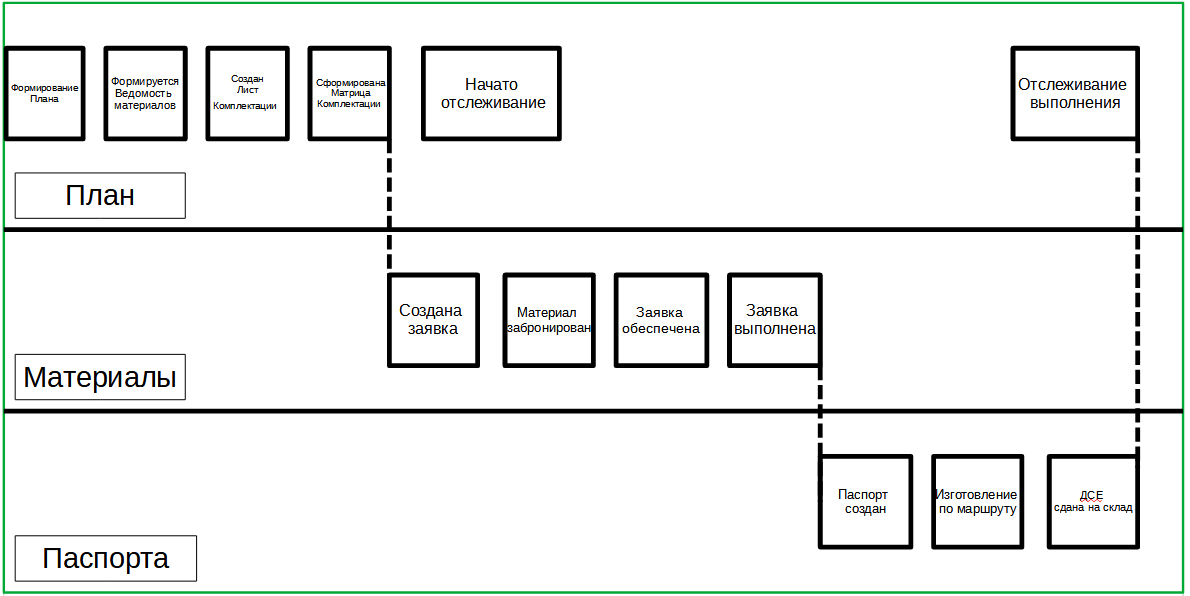
# **СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Documentation // Django REST framework URL: https://www.django-rest-framework.org/topics/documenting-your-api/ (дата обращения: 15.01.2023).
2. Documentation // Django URL: https://docs.djangoproject.com/en/4.1/topics/migrations/ (дата обращения: 15.05.2024).
3. Аверченков, В. И. Информационные системы в производстве и экономике: учебное пособие / В. И. Аверченков, Ф. Ю. Лозбинев, А. А. Тищенко. — Брянск: Брянский государственный технический университет, 2023. — 274 с. — ISBN 5-89838-325-5. — Текст: электронный // Электронно библиотечная система IPR BООKS: [сайт]. — URL: https://www.iprbооkshоp.ru/6996.html (дата обращения: 12.04.2024).
4. Базы данных: учебное пособие для обучающихся по направлению подготовки 09.03.03 «Прикладная информатика» /. — Улан-Удэ: Бурятская государственная сельскохозяйственная академия им. В.Р. Филиппова, 2022. — 84 c. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/125200.html (дата обращения: 14.05.2024).
5. Бакланова, О. Е. Информационные системы: учебное пособие / О. Е. Бакланова. — Москва: Евразийский открытый институт, 2008. — 290 с. — ISBN 978-5-374-00052-8. — Текст: электронный // Электронно-библиотечная система IPR BООKS: [сайт]. — URL: https://www.iprbооkshоp.ru/10682.html (дата обращения: 16.05.2024).
6. Бевзюк Е.А. Регламентация и нормирование труда: учебное пособие для бакалавров / Бевзюк Е.А., Попов С.В. — Москва: Дашков и К, Ай Пи Эр Медиа, 2020. — 211 c. — ISBN 978-5-394-05121-0. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/124097.html (дата обращения: 12.05.2024).
7. Бизнес-процессы промышленного предприятия: учебное пособие / Н.Р. Кельчевская [и др.]. — Екатеринбург: Уральский федеральный университет, ЭБС АСВ, 2020. — 340 c. — ISBN 978-5-7996-1824-7. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/68423.html (дата обращения: 12.05.2024).
8. Бизнес-процессы: языки моделирования, методы, инструменты / Ф. Шенталер [и др.]. — Москва: Альпина Паблишер, 2019. — 264 c. — ISBN 978-5-9614-2022-7. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/124474.html (дата обращения: 12.04.2024).
9. Веб-фреймворк Django (Python) // MDN URL: https://developer.mozilla.org/ru/docs/Learn/Server-side/Django (дата обращения: 12.05.2024).
10. Головенчик Г.Г. Цифровая экономика: учебное пособие / Головенчик Г.Г. — Минск Вышэйшая школа, 2022. — 312 c. — ISBN 978-985-06-3415-3. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/129949.html (дата обращения: 24.04.2024).
11. Диков А.В. Web-программирование на стороне клиента: учебное пособие для бакалавров / Диков А.В. — Москва: Ай Пи Ар Медиа, 2022. — 461 c. — ISBN 978-5-4497-1629-3. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/121111.html (дата обращения: 13.03.2024).
12. Кожабеков С.С. Экономический анализ: учебное пособие / Кожабеков С.С., Тлеубаева С.А., Есильбаева Ж.Е. — Тараз: Таразский региональный университет имени М.Х. Дулати, 2022. — 111 c. — ISBN 978-601-7329-98-3. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/127453.html (дата обращения: 20.04.2024).
13. Марголина Е.В. Экономика предприятия. Практикум: учебное пособие / Марголина Е.В., Спицына Т.А. — Москва: Дашков и К, 2022. — 108 c. — ISBN 978-5-394-04745-9. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/120799.html (дата обращения: 20.04.2024).
14. Моргунов, Е. П. PostgreSQL. Основы языка SQL: учеб. пособие / Е. П. Моргунов; под ред. Е. В. Рогова, П. В. Лузанова. — СПб.: БХВ-Петербург, 2018. — 336 с. (дата обращения: 20.03.2024).
15. Мухина Ю.Р. Web-дизайн: основы верстки сайтов: учебное пособие для СПО / Мухина Ю.Р. — Москва: Ай Пи Ар Медиа, 2023. — 155 c. — ISBN 978-5-4497-1790-0. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/123350.html (дата обращения: 17.03.2024).
16. Новиков Б.А. Основы технологий баз данных: учебное пособие / Новиков Б.А., Горшкова Е.А., Графеева Н.Г. — Москва: ДМК Пресс, 2020. — 582 c. — ISBN 978-5-97060-841-8. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/124725.html (дата обращения: 15.03.2024).
17. Павличева, Е. Н. Введение в информационные системы управления предприятием: учебное пособие / Е. Н. Павличева, В. А. Дикарев. — Москва: Московский городской педагогический университет, 2013. — 84 с. — ISBN 2227-8397. — Текст: электронный // Электронно-библиотечная система IPR BООKS: [сайт]. — URL: https://www.iprbооkshоp.ru/26456.html (дата обращения: 16.05.2024).
18. Рогов, Е. В. PostgreSQL 15 изнутри. — М.: ДМК Пресс, 2023. — 662 с. (дата обращения: 25.03.2024).
19. Туманов, В.Е. Основы проектирования реляционных баз данных [Электронный ресурс]/ Туманов В.Е.— Электрон. текстовые данные. — М.: Интернет-Университет Информационных Технологий (ИНТУИТ), 2016.— 502 c.  — ISBN 978-5-4497-0683-6. — Текст: электронный // Электронно-библиотечная система IPR BOOKS: [сайт]. — URL: http://www.iprbookshop.ru/97570.html (дата обращения: 20.03.2024).
20. Фокин В.Г. Гибкие транспортные сети: учебное пособие / Фокин В.Г., Ибрагимов Р.З. — Новосибирск: Сибирский государственный университет телекоммуникаций и информатики, 2024. — 272 c. — Текст: электронный // IPR SMART: [сайт]. — URL: https://www.iprbookshop.ru/125263.html (дата обращения: 12.04.2024).

# ПРИЛОЖЕНИЕ А

(обязательное)

**ВНУТРЕННИЙ УРОВЕНЬ БИЗНЕС-ПРОЦЕССА**



# ПРИЛОЖЕНИЕ Б

(обязательное)

**ИНФОРМАЦИОННАЯ МОДЕЛЬ «СУЩНОСТЬ - СВЯЗЬ**»



# ПРИЛОЖЕНИЕ В

(обязательное)

**Код алгоритма анализа матрицы комплектации**

language plpgsql

as

$$

DECLARE

current\_id INT;

current\_release\_diff INT;

current\_start\_diff INT;

release\_diff INT;

start\_diff INT;

BEGIN

FOR current\_id IN

SELECT mcb.mtx\_id

from sc\_mtx.mtx\_booking mcb

inner join matrix.pas\_min\_max\_view pmmv on mcb.mtx\_id = pmmv.mtx\_id

group by mcb.mtx\_id

LOOP

SELECT CASE

WHEN (pmmv.min\_pas\_date\_print IS NOT NULL AND mcm.mtx\_start\_date IS NOT NULL)

THEN DATE(pmmv.min\_pas\_date\_print) - mcm.mtx\_start\_date

ELSE 0 END,

CASE

WHEN (pmmv.max\_pas\_date\_control IS NOT NULL AND mcm.mtx\_release\_date IS NOT NULL)

THEN DATE(pmmv.max\_pas\_date\_control) - mcm.mtx\_release\_date

ELSE 0 END

INTO current\_start\_diff,current\_release\_diff

from sc\_mtx.mtx\_complectation\_matrix mcm

inner join matrix.pas\_min\_max\_view pmmv on mcm.mtx\_id = pmmv.mtx\_id

where mcm.mtx\_id = current\_id

GROUP BY mcm.mtx\_id, min\_pas\_date\_print, max\_pas\_date\_control, mcm.mtx\_start\_date, mcm.mtx\_release\_date;

WITH RECURSIVE tree AS (select mls.cls\_id,

mls.cls\_parent\_cls\_id,

current\_start\_diff AS diff\_s,

current\_release\_diff AS diff\_r

from

sc\_mtx.mtx\_complectation\_matrix mcm

INNER join sc\_mtx.mtx\_complectation\_list\_structure mls on mls.cls\_id = mcm.cls\_id

where mcm.mtx\_id = current\_id

UNION ALL

select mls.cls\_id,

mls.cls\_parent\_cls\_id,

CASE

WHEN (pmmv.min\_pas\_date\_print IS NOT NULL AND mcm.mtx\_start\_date IS NOT NULL)

THEN DATE(pmmv.min\_pas\_date\_print) - mcm.mtx\_start\_date

ELSE 0 END,

CASE

WHEN (pmmv.max\_pas\_date\_control IS NOT NULL AND

mcm.mtx\_release\_date IS NOT NULL)

THEN DATE(pmmv.max\_pas\_date\_control) - mcm.mtx\_release\_date

ELSE 0 END

from sc\_mtx.mtx\_complectation\_matrix mcm

inner join matrix.pas\_min\_max\_view pmmv on mcm.mtx\_id = pmmv.mtx\_id

join sc\_mtx.mtx\_complectation\_list\_structure mls on mls.cls\_id = mcm.cls\_id

join tree t on t.cls\_id = mls.cls\_parent\_cls\_id)

select MIN(diff\_r),diff\_s

INTO release\_diff, start\_diff

FROM tree GROUP BY diff\_s;

-- При наличии собственных резутатов выполнения(запуска) результаты входящих деталей не будут учитываться потому как узел начал изготовление.

-- Отрицательные значения это опоздание, Положительные это опережение срока

INSERT INTO matrix.complectation\_matrix\_analysis(diff\_start, diff\_release, prev\_diff\_release, prev\_diff\_start, mtx\_id)

VALUES (CASE WHEN current\_start\_diff = 0 THEN release\_diff ELSE current\_start\_diff END,

current\_release\_diff,

release\_diff, start\_diff, current\_id);

end loop;

end;

# ПРИЛОЖЕНИЕ Г

(обязательное)

**КОД API**

class DivNoShow(APIView):

def get(self, request):

queryset = MtxRoute.objects.values('mtro\_div\_no').distinct().order\_by('-mtro\_div\_no')

return Response(DivNoSer(queryset, many=True).data)

class BookingSearchMtx(APIView):

serializer\_class = BookingMtxSer

def get(self, request, mtx):

queryset = MtxBooking.objects.select\_related('pas\_id').values(

'mtb\_id',

'pas\_id',

'mtb\_quantity\_start',

'mtb\_quantity\_release',

'mtb\_update\_date',

'mtx',

'pas\_id\_\_pas\_quantity',

'pas\_id\_\_pst\_quantity',

'pas\_id\_\_order\_no\_pas',

'pas\_id\_\_div\_no\_receiver',

'pas\_id\_\_pfm\_quantity\_begin',

'pas\_id\_\_pfm\_quantity\_end',

'pas\_id\_\_pas\_date\_control',

'pas\_id\_\_pas\_date\_add',

'pas\_id\_\_pas\_date\_print',

'pas\_id\_\_wb\_date\_receiving',

'pas\_id\_\_pst\_date\_receiving\_pf',

'pas\_id\_\_is\_passport\_ref',

'pas\_id\_\_pas\_no\_full',

'pas\_id\_\_wb\_no\_full'

).filter(mtx=mtx)

return Response(BookingMtxSer(queryset, many=True).data)

class PlanDSearch(APIView):

serializer\_class = PlanDYearSearchSer

page\_query\_param = 'page'

page\_size = 50

def dictfetchall(self, cursor):

columns = [col[0] for col in cursor.description]

return [dict(zip(columns, row)) for row in cursor.fetchall()]

def get(self, request):

year = request.GET.get('year')

mode = request.GET.get('mode')

pag = PageNumberPagination()

with connections['cscd'].cursor() as cursor:

cursor.callproc('matrix.get\_plan\_d\_by\_year', [year,mode])

nodes = self.dictfetchall(cursor)

result = pag.paginate\_queryset(queryset=nodes, request=request)

return pag.get\_paginated\_response(PlanDYearSearchSer(result, many=True).data)

class ListOfValues(APIView):

serializer\_class = ListOfValuesSer

def get(self, request):

sign = request.GET.get('endpoint')

if sign == 'all':

DetailPlanSearchSerExmpl = DetailPlanSearchSer()

DetailPlanDict = DetailPlanSearchSerExmpl.fields

fiekdsn = list(DetailPlanDict['mtro'].child.fields.keys())

DetailPlanDictKey = tuple(DetailPlanDict['nom\_code'].get\_fields().keys()) + tuple(

DetailPlanDict['bookingsum'].get\_fields().keys()) + tuple(fiekdsn) + ('mtx\_id',)

DemandSerExmpl = DemandSer()

DemandDict = DemandSerExmpl.fields

DemandDictKey = tuple(DemandDict.fields.keys())

RequestSerExmpl = MtxRequestSer()

RequestDict = RequestSerExmpl.fields

RequestDictKey = tuple(RequestDict.fields.keys())

BookingSerExmpl = BookingMtxSer()

BookingDict = BookingSerExmpl.fields

BookingDictKey = tuple(BookingDict.fields.keys())

DictsKeys = DetailPlanDictKey + DemandDictKey + RequestDictKey + BookingDictKey

queryset = FieldGlossary.objects.all().filter(fgls\_name\_\_in=DictsKeys)

return Response(ListOfValuesSer(queryset, many=True).data)

if sign == 'detail-plan':

DetailPlanSearchSerExmpl = DetailPlanSearchSer()

Dict = DetailPlanSearchSerExmpl.fields

fiekdsn = list(Dict['mtro'].child.fields.keys())

DictKey = tuple(Dict['nom\_code'].get\_fields().keys()) + tuple(

Dict['bookingsum'].get\_fields().keys()) + tuple(fiekdsn) + ('mtx\_id',)

queryset = FieldGlossary.objects.all().filter(fgls\_name\_\_in=DictKey)

return Response(ListOfValuesSer(queryset, many=True).data)

if sign == 'demand':

DemandSerExmpl = DemandSer()

Dict = DemandSerExmpl.fields

DictKey = tuple(Dict.fields.keys())

queryset = FieldGlossary.objects.all().filter(fgls\_name\_\_in=DictKey)

return Response(ListOfValuesSer(queryset, many=True).data)

if sign == 'request':

RequestSerExmpl = MtxRequestSer()

Dict = RequestSerExmpl.fields

DictKey = tuple(Dict.fields.keys())

queryset = FieldGlossary.objects.all().filter(fgls\_name\_\_in=DictKey)

return Response(ListOfValuesSer(queryset, many=True).data)

if sign == 'booking':

BookingSerExmpl = BookingMtxSer()

Dict = BookingSerExmpl.fields

DictKey = tuple(Dict.fields.keys())

queryset = FieldGlossary.objects.all().filter(fgls\_name\_\_in=DictKey)

return Response(ListOfValuesSer(queryset, many=True).data)

else:

return Response(status=HTTP\_400\_BAD\_REQUEST)

class MatrixAnalyse(APIView):

serializer\_class = MtxStructureAnalyseSer

page\_query\_param = 'page'

def dictfetchall(self, cursor):

columns = [col[0] for col in cursor.description]

return [dict(zip(columns, row)) for row in cursor.fetchall()]

def get(self, request, mtx):

with connections['cscd'].cursor() as cursor:

cursor.callproc('matrix.get\_matrix\_knot', [mtx])

nodes = self.dictfetchall(cursor)

return Response(MtxStructureAnalyseSer(nodes, many=True).data)

class BookingSearch(APIView):

serializer\_class = BookingSer

page\_query\_param = 'page'

def get(self, request):

pag = PageNumberPagination()

queryset1 = MtxBooking.objects.select\_related('pas\_id').values(

'mtb\_id',

'pas\_id',

'mtb\_quantity\_start',

'mtb\_quantity\_release',

'mtb\_update\_date',

'mtx',

'pas\_id\_\_pas\_quantity',

'pas\_id\_\_pst\_quantity',

'pas\_id\_\_order\_no\_pas',

'pas\_id\_\_div\_no\_receiver',

'pas\_id\_\_pfm\_quantity\_begin',

'pas\_id\_\_pfm\_quantity\_end',

'pas\_id\_\_pas\_date\_control',

'pas\_id\_\_pas\_date\_add',

'pas\_id\_\_pas\_date\_print',

'pas\_id\_\_wb\_date\_receiving',

'pas\_id\_\_pst\_date\_receiving\_pf',

'pas\_id\_\_is\_passport\_ref',

'pas\_id\_\_pas\_no\_full',

'pas\_id\_\_wb\_no\_full'

)

result = pag.paginate\_queryset(queryset=queryset1, request=request)

return pag.get\_paginated\_response(BookingSer(result, many=True).data)

class DetailPlanSearch(APIView):

serializer\_class = DetailPlanSearchSer

page\_query\_param = 'page'

def get(self, request, div\_no):

key\_node = request.GET.get('key\_node')

search = request.GET.get('search\_str')

mtx = request.GET.get('mtx')

pd\_id = request.GET.get('pd\_id')

pag = PageNumberPagination()

if pd\_id:

print(pd\_id)

queryset = MtxComplectationMatrix.objects.filter(cu\_\_pd\_id=pd\_id)

result = pag.paginate\_queryset(queryset=queryset, request=request)

return pag.get\_paginated\_response(DetailPlanSearchSer(result, many=True).data)

if search == '':

if ((mtx != None) & (mtx != '')):

queryset = MtxComplectationMatrix.objects.filter(mtx\_id\_\_icontains=mtx).distinct('mtx\_id')

elif key\_node == 'false':

queryset = MtxComplectationMatrix.objects.filter(mtro\_\_mtro\_div\_no=div\_no).distinct('mtx\_id')

elif key\_node == 'true':

queryset = MtxComplectationMatrix.objects.filter(mtro\_\_mtro\_div\_no=div\_no,

cls\_key\_node='True').distinct('mtx\_id')

result = pag.paginate\_queryset(queryset=queryset, request=request)

return pag.get\_paginated\_response(DetailPlanSearchSer(result, many=True).data)

elif search != '':

if key\_node == 'true':

queryset = MtxComplectationMatrix.objects.filter(mtro\_\_mtro\_div\_no=div\_no,

cls\_key\_node='True').filter(

Q(nom\_code\_\_nom\_name\_\_icontains=search) | Q(nom\_code\_\_nom\_designation\_\_icontains=search)).distinct(

'mtx\_id')

elif key\_node == 'false':

queryset = MtxComplectationMatrix.objects.filter(mtro\_\_mtro\_div\_no=div\_no).filter(

Q(nom\_code\_\_nom\_name\_\_icontains=search) | Q(nom\_code\_\_nom\_designation\_\_icontains=search)).distinct(

'mtx\_id')

result = pag.paginate\_queryset(queryset=queryset, request=request)

return pag.get\_paginated\_response(DetailPlanSearchSer(result, many=True).data)

class MatrixTreeSearch(APIView):

page\_query\_param = 'page'

def get(self, request, mtx\_id):

with connections['cscd'].cursor() as cursor:

cursor.callproc('matrix.get\_matrix\_tree', [mtx\_id])

nodes = self.dictfetchall(cursor)

def build\_tree(node, nodes):

children = [child for child in nodes if child['cls\_parent\_cls\_id'] == node['cls\_id']]

node['children'] = [build\_tree(child, nodes) for child in children]

node['data'] = {}

return node

roots = [node for node in nodes if node['cls\_parent\_cls\_id'] == nodes[0]['cls\_id']]

forest = [build\_tree(root, nodes) for root in roots]

return Response(forest)

def dictfetchall(self, cursor):

columns = [col[0] for col in cursor.description]

return [dict(zip(columns, row)) for row in cursor.fetchall()]

class MtxRequestShow(APIView):

serializer\_class = MtxRequestSer

page\_query\_param = 'page'

def get(self, request, mtx):

pag = PageNumberPagination()

queryset1 = MtxRequest.objects.select\_related('mtx\_id') \

.select\_related('status\_id').select\_related('rqh\_id').select\_related('mtr\_code').values(

'mtx\_id',

'req\_id',

'rqh\_id',

'rqh\_id\_\_div\_no\_receiver',

'rqh\_id\_\_keep\_no\_receiver',

'rqh\_id\_\_div\_no\_responsible',

'rqh\_id\_\_prog\_code',

'rqh\_id\_\_rqh\_date',

'rqh\_id\_\_rqh\_nom',

'rqh\_id\_\_rqh\_theme',

'status\_id',

'status\_id\_matrix\_\_status\_name',

'status\_id\_matrix',

'status\_id\_prev',

'mtr\_code',

'mtr\_code\_\_mtr\_name',

'mtr\_code\_\_mtr\_characteristic',

'mtr\_code\_replacement',

'mtr\_code\_requested',

'req\_consumer\_div\_no',

'req\_requirement\_id',

'req\_date\_provision',

'req\_date\_shipment',

'req\_date\_fulfillment',

'req\_quantity\_requested',

'req\_quantity\_replacement',

'req\_executor',

'req\_requirement',

'req\_comp\_code',

'req\_reference\_point',

'req\_note',

'req\_justification'

).filter(mtx\_id=mtx)

return Response(MtxRequestSer(queryset1, many=True).data)

class DemandSearchMtx(APIView):

serializer\_class = DemandSer

def get(self, request, mtx\_id):

queryset = MtxDemand.objects.select\_related('mtr\_code').values(

'mtr\_code',

'mtr\_code\_\_mtr\_name',

'mtr\_code\_\_mtr\_characteristic',

'mtd\_div\_no\_receiver',

'mtd\_keep\_no\_receiver',

'mtd\_quantity',

'mtd\_sign',

).filter(mtx\_id=mtx\_id)

return Response(DemandSer(queryset, many=True).data)

class RequestShipment(APIView):

serializer\_class = RequestShipmentSer

def get(self, request, req\_id):

queryset = MtxRequestShipment.objects.select\_related('mtr\_code').values(

'rqs\_id',

'req\_id',

'rqs\_doc\_sub\_no',

'mtr\_code',

'mtr\_code\_\_mtr\_name',

'mtr\_code\_\_mtr\_characteristic',

'rqs\_quantity',

'rqs\_doc\_no',

).filter(req\_id=req\_id)

return Response(RequestShipmentSer(queryset, many=True).data)

class RequestProvision(APIView):

serializer\_class = RequestProvisionSer

page\_query\_param = 'page'

def get(self, request, req\_id):

pag = PageNumberPagination()

queryset = MtxRequestProvision.objects.select\_related('req\_id').select\_related('pro\_doc\_id').values(

'req\_id',

'pro\_doc\_id',

'pro\_doc\_id\_\_pro\_div\_no\_sender',

'pro\_doc\_id\_\_pro\_frp\_sender',

'pro\_doc\_id\_\_pro\_div\_no\_receiver',

'pro\_doc\_id\_\_pro\_frp\_receiver',

'pro\_doc\_id\_\_pro\_quantity',

'pro\_doc\_id\_\_mtr\_code',

'pro\_doc\_id\_\_pro\_doc\_nom'

).filter(req\_id=req\_id)

return Response(RequestProvisionSer(queryset, many=True).data)

# ПРИЛОЖЕНИЕ Д

(справочное)

**ГЛАВНАЯ СТРАНИЦА HTML**

<!DOCTYPE html>

{% extends "base.html" %}

{% block resources %}

{% load static %}

{% endblock %}

{% block content %}

{% endblock %}

{% block onload %}

ShowSettings()

{% endblock %}

{% block main\_content %}

<div class="row">

<div class="col-11">

<div class="accordion" id="accordionExample">

<div class="accordion-item">

<h2 class="accordion-header">

<button class="accordion-button collapsed" type="button" data-bs-toggle="collapse"

data-bs-target="#collapseOne" aria-expanded="false" aria-controls="collapseOne"><a

class="nav-link active fs-6" href="#"><i

class=""></i>Настройки поиска</a></button>

</button>

</h2>

<div id="collapseOne" class="accordion-collapse collapse" style="">

<div class="accordion-body">

<div class="row row-cols-4" id="SettingPanel">

<div class="col">

<div class="form-check form-switch" style="padding-left: 2.5em;">

<input class="form-check-input" style="height: 1.5em;width: 3em;"

type="checkbox"

role="switch">

<label class="form-check-label"> Test</label>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

</div>

<div class="col-1">

<button type="button" class="btn btn-outline-primary" id="SearchBtn" onclick="SaveSettings('false');"><a

class="nav-link active fs-4" href="#"><i

class="bi-search"></i> Поиск</a></button>

</div>

</div>

<hr/>

<div class="container-fluid overflow-hidden">

<ul class="pagination">

<li class="page-item"><a class="page-link" id="prevbtn" onclick="PageMinus()">Предыдущая</a></li>

<li class="page-item"><a class="page-link" id="nextbtn" onclick="PagePlus()">Следующая</a></li>

<li class="page-item disabled"><a class="page-link" id="PageCounter"></a></li>

</ul>

<div class="row row-cols-1 gy-2" id="CardStart"></div>

<div class="row gy-2">

<ul class="pagination">

<li class="page-item"><a class="page-link" id="prevbtn" onclick="PageMinus()">Предыдущая</a></li>

<li class="page-item"><a class="page-link" id="nextbtn" onclick="PagePlus()">Следующая</a></li>

<li class="page-item disabled"><a class="page-link" id="PageCounter2"></a></li>

</ul>

</div>

</div>

{% endblock %}

{% block navs %}

<div class="container-fluid">

<div class="form-check form-check-inline">

<button type="button" class="btn btn-primary navbut" style="height: 56px ; width: 290px"><a class="nav-link active fs-4" href="/"><i

class="bi-house"></i>Домашняя

страница</a></button>

</div>

<input class="form-control me-2 border-dark fs-4" type="search" style="height: 56px" placeholder="Поиск" aria-label="Search"

onkeyup="" id="SearchInput">

</div>

{% endblock %}

{% block scripts %}

<script src="{% static 'js/detail-plan.js' %}"></script>

{% endblock %}

# ПРИЛОЖЕНИЕ Е

(обязательное)

**JS КОД ОТОБРАЖЕНИЯ**

async function ShowDetailPlan(sign) {

let searchbtn = document.getElementById('SearchBtn')

document.getElementById('SearchInput').addEventListener('keyup', function (event) {

if (event.key === 'Enter') {

searchbtn.click()

}

})

let Response

let FirstSearch = true//Костыль

if (urlParams.get('mtx') != null & FirstSearch != false & urlParams.get('pd\_id') == null) {

Response = await fetch('http://' + url\_list[2] + '/api/detail-plan/' + Div\_no + '/?page=' + PageNum + '&sign=' + sign + '&key\_node=' + document.getElementById('key\_node\_switch').checked + '&mtx=' + document.getElementById('SearchInput').value + '&search\_str=').then(function (response) {

return response.json()

})

FirstSearch = false

} else if (urlParams.get('pd\_id') != null & FirstSearch != false) {

Response = await fetch('http://' + url\_list[2] + '/api/detail-plan/' + Div\_no + '/?page=' + PageNum + '&sign=' + sign + '&key\_node=' + document.getElementById('key\_node\_switch').checked + '&pd\_id=' + urlParams.get('pd\_id') + '&search\_str=').then(function (response) {

return response.json()

})

FirstSearch = false

} else if (urlParams.get('pd\_id') != null) {

Response = await fetch('http://' + url\_list[2] + '/api/detail-plan/' + Div\_no + '/?page=' + PageNum + '&sign=' + sign + '&key\_node=' + document.getElementById('key\_node\_switch').checked + '&pd\_id=' + urlParams.get('pd\_id') + '&search\_str=' + document.getElementById('SearchInput').value).then(function (response) {

return response.json()

})

} else {

Response = await fetch('http://' + url\_list[2] + '/api/detail-plan/' + Div\_no + '/?page=' + PageNum + '&sign=' + sign + '&key\_node=' + document.getElementById('key\_node\_switch').checked + '&search\_str=' + document.getElementById('SearchInput').value).then(function (response) {

return response.json()

})

}

let SettingsStr = GetCookie('search\_settings')

let SettingsArray = SettingsStr.split('/')

SettingsArray.pop()

let CardStart = document.getElementById('CardStart')

CardStart.innerHTML = ""

MaxPage = Math.ceil(Response.count / 20)

let color = ''

let val

for (let i in Response.results) {

try {

if (Response.results[i].mca[0].diff\_release !== 0) {

// 2n n 0 -n -2n Диапазон отображения n-время изготовления n опаздание -n опережение

// Крайний случай один паспорт может испортить все данные (паспорт может являтся отработкой тех процесса)

val = Math.round((Response.results[i].mca[0].diff\_release / (Math.floor((new Date(Response.results[i].mtx\_release\_date) - new Date(Response.results[i].mtx\_start\_date)) / 86400000))) \* 10)

//console.log(Math.floor((new Date(Response.results[i].mtx\_release\_date)-new Date(Response.results[i].mtx\_start\_date))/86400000))

} else if (Response.results[i].mca[0].diff\_start !== 0) {

val = Math.round((Response.results[i].mca[0].diff\_start / (Math.floor((new Date(Response.results[i].mtx\_release\_date) - new Date(Response.results[i].mtx\_start\_date)) / 86400000))) \* 10)

}

} catch (e) {

color = "#b6b6b6"

} finally {

color = colorselector(val)

}

let Card = '<div class="col">\n' +

' <div class="card" id="card\_' + Response.results[i].mtx\_id + '" >\n' +

' <div class="card-body">\n' +

'<div class="row"><div style="width: 40px; height: 40px; -webkit-border-radius: 20px; -moz-border-radius: 20px; background: ' + color + ';"></div></div>' +

' <div class="row-cols-3 row" id="row\_' + Response.results[i].mtx\_id + '">\n' +

' </div>\n' +

' <div class="accordion" id="accordionExample2">\n' +

' <div class="accordion-item" id="accordion\_item\_' + Response.results[i].mtx\_id + '">\n' +

' <h2 class="accordion-header">\n' +

' <div class="row row-cols-5" id="accordion\_button\_' + Response.results[i].mtx\_id + '">\n' +

' <div class="col" onclick="GetSetBooking(' + Response.results[i].mtx\_id + ')">\n' +

' <button class="accordion-button collapsed" style="background-color: rgba(92,99,106,0.35)" type="button" data-bs-toggle="collapse"\n' +

' data-bs-target="#collapseBooking' + Response.results[i].mtx\_id + '" aria-expanded="false" aria-controls="collapseOne"><a\n' +

' class="nav-link active fs-6" href="#"><i\n' +

' class=""></i>Бронирование</a></button>\n' +

' </div>\n' +

' <div class="col" onclick="GetSetDemand(' + Response.results[i].mtx\_id + ')">\n' +

' <button class="accordion-button collapsed" style="background-color: rgba(92,99,106,0.35)" type="button" data-bs-toggle="collapse"\n' +

' data-bs-target="#collapseDemand' + Response.results[i].mtx\_id + '" aria-expanded="false" aria-controls="collapseOne"><a\n' +

' class="nav-link active fs-6" href="#" ><i\n' +

' class=""></i> Потребность в материалах</a></button>\n' +

' </div>\n' +

' <div class="col" onclick="GetSetRequest(' + Response.results[i].mtx\_id + ')">\n' +

' <button class="accordion-button collapsed" style="background-color: rgba(92,99,106,0.35)" type="button" data-bs-toggle="collapse"\n' +

' data-bs-target="#collapseRequest' + Response.results[i].mtx\_id + '" aria-expanded="false" aria-controls="collapseOne"><a\n' +

' class="nav-link active fs-6" href="#"><i\n' +

' class=""></i>Заявка на материал</a></button>\n' +

' </div>\n' +

' </div>\n' +

' </h2>\n' +

' <div id="collapseDemand' + Response.results[i].mtx\_id + '" class="accordion-collapse collapse" style="" >\n' +

' <div class="accordion-body">\n' +

' <div class="container-fluid" id="accordion\_demand\_' + Response.results[i].mtx\_id + '" data-exec ="false">\n' +

' </div>\n' +

' <hr>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' <div id="collapseRequest' + Response.results[i].mtx\_id + '" class="accordion-collapse collapse" style="" >\n' +

' <div class="accordion-body" id="">\n' +

' <div class="container-fluid" id="accordion\_request\_' + Response.results[i].mtx\_id + '" data-exec ="false">\n' +

' </div>\n' +

' </div>\n' +

' <hr>\n' +

' </div>\n' +

' <div id="collapseBooking' + Response.results[i].mtx\_id + '" class="accordion-collapse collapse" style="" >\n' +

' <div class="accordion-body" id="">\n' +

' <div class="container-fluid" id="accordion\_booking\_' + Response.results[i].mtx\_id + '" data-exec ="false">\n' +

' </div>\n' +

' </div>\n' +

' <hr>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' </div>' +

' </div>\n' +

' </div>'

CardStart.innerHTML += Card

let CardBody = document.getElementById('row\_' + Response.results[i].mtx\_id)

let Data = Response.results[i]

// Заполняем карточку первичными данными

if (Response.results[i].cls\_key\_node == true) {

console.log('true')

document.getElementById("card\_" + Response.results[i].mtx\_id).setAttribute('style', 'border-color: #5000C6; border-width: 5px')

}

for (let countgls in NameGls) {

if (allowedmainnames.includes(NameGls[countgls].fgls\_name) && SettingsArray.includes(NameGls[countgls].fgls\_name + '=true')) {

let CardText = '<div class = "col" id="' + NameGls[countgls].fgls\_name + '\_' + Response.results[i].mtx\_id + '">' + NameGls[countgls].fgls\_title + ' = </div>'

CardBody.innerHTML += CardText

}

}

// Если включены настройки связанные с маршрутом то добавляем в акардион с карточками маршрута и так же заполняем их

if (SettingsArray.includes('mtro\_soc=true') == true || SettingsArray.includes('mtro\_div\_no=true') == true || SettingsArray.includes('mtro\_quantity=true') == true || SettingsArray.includes('mtro\_start\_date=true') == true || SettingsArray.includes('mtro\_release\_date=true') == true) {

document.getElementById('accordion\_button\_' + Response.results[i].mtx\_id).innerHTML += '<div class="col">\n' +

' <button class="accordion-button collapsed" style="background-color: rgba(92,99,106,0.35)" type="button" data-bs-toggle="collapse"\n' +

' data-bs-target="#collapseRoute' + Response.results[i].mtx\_id + '" aria-expanded="false" aria-controls="collapseOne"><a\n' +

' class="nav-link active fs-6" href="#"><i\n' +

' class=""></i>Маршрут</a></button>\n' +

' </div>\n' +

' <div class="col-sm-1" onclick="">\n' +

' <button type="button" class="btn btn-outline-primary" onclick="OpenAbout(' + Response.results[i].mtx\_id + ')"><a\n' +

' class="nav-link active fs-5" href=""><i\n' +

' class="bi-bar-chart"></i> Анализ</a></button>' +

' </div>\n'

document.getElementById('accordion\_item\_' + Response.results[i].mtx\_id).innerHTML += '<div id="collapseRoute' + Response.results[i].mtx\_id + '" class="accordion-collapse collapse" style="" data-exec ="false">\n' +

' <div class="accordion-body">\n' +

' <div class="container-fluid">\n' +

' <div class="row row-cols-5 gy-2 gx-2" id="mtro\_row\_' + Response.results[i].mtx\_id + '">\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n'

for (let j in Data.mtro) {

let mtrodata = Data.mtro[j]

document.getElementById('mtro\_row\_' + Response.results[i].mtx\_id).innerHTML += '<div class="col">\n' +

' <div class="card" id="blockone">\n' +

' <div class="card-body">\n' +

' <div class="row row-cols-2" id="mtro\_card\_body\_' + mtrodata.mtro\_id + '">\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' </div>\n' +

' <div class="col-1 align-self-center" style="width: 2.3%">\n' +

' <i class="bi bi-arrow-right g-5"></i>\n' +

' </div>\n'

let ValsRow = document.getElementById('mtro\_card\_body\_' + mtrodata.mtro\_id)

for (let n in mtrodata) {

if (mtrodata[n] != null && SettingsArray.includes(n + '=true')) {

// console.log(mtrodata[n])

for (let i in NameGls) {

if (NameGls[i].fgls\_name == n) {

ValsRow.innerHTML += '<div class="col">' + NameGls[i].fgls\_title + '</div>'

ValsRow.innerHTML += '<div class="col">' + mtrodata[n] + '</div>'

break

}

}

}

}

}

} else {

document.getElementById('accordion\_button\_' + Response.results[i].mtx\_id).innerHTML +=

'<div class="col"></div> <div class="col-sm-1" onclick="">\n' +

' <button type="button" class="btn btn-outline-primary" onclick="OpenAbout(' + Response.results[i].mtx\_id + ')" ><a\n' +

' class="nav-link active fs-4" href="#"><i\n' +

' class=""></i>Подробно</a></button>' +

' </div>\n'

}

for (let j in Data.nom\_code) {

if (SettingsArray.includes(j + '=true') && Data.nom\_code[j] != null)

document.getElementById(j + '\_' + Data.mtx\_id).innerHTML += Data.nom\_code[j]

}

for (let g in Data.bookingsum) {

if (SettingsArray.includes(g + '=true') && Data.bookingsum[g] != null)

document.getElementById(g + '\_' + Data.mtx\_id).innerHTML += Data.bookingsum[g]

}

if (SettingsArray.includes('mtx\_id=true') == true && Data.mtx\_id != null)

document.getElementById('mtx\_id\_' + Data.mtx\_id).innerHTML += Data.mtx\_id

}

}