Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления

Кафедра интеллектуальных информационных технологий

Отчет по производственной (технологической) практике

Выполнил студент гр. 221703
Вечорко Д. Н.

Руководитель практики от
предприятия:
системный аналитик
Мажар А. А.

Руководитель практики
от университета:
ассистент кафедры ИИТ
Крищенович В. А.

Минск 2025

СОДЕРЖАНИЕ

Введение 4
1 Характеристика места практики 6
2 Анализ средств и методов автоматизированного формирования отчётов на основе источника данных 8
2.1 Анализ предметной области автоматизированного формирования отчётов 8
2.2 Анализ подходов к реализации модулей формирования отчётов 9
2.2.1 Использование BI-систем и отчетных инструментов . . 9
2.2.2 Шаблонизаторы отчётов 10
2.2.3 Программная реализация на языках общего назначения 10
2.2.4 Облачные и SaaS-решения 11
2.2.5 Выбор технологии: факторы и критерии 11
2.3 Анализ аналогов систем формирования отчётов 12
2.3.1 JasperReports 12
2.3.2 Microsoft Power BI 13
2.3.3 Python + Pandas 15
2.4 Вывод 16
3 Проектирование и реализация модуля формирования отчётов . . 18
3.1 Постановка задачи 18
3.1.1 Цель работы 18
3.1.2 Задачи модуля 18
3.2 Потенциальные пользователи 18
3.3 Общая архитектура модуля 19
3.4 Описание компонентов 22
3.4.1 Модуль контроллера 22
3.4.2 Сервисный слой 22
3.4.3 Интерфейс стратегии формирования отчёта 22
3.4.4 Реализации стратегий 22
3.4.5 Репозиторий получения данных 23
3.4.6 DTO (объект передачи данных) 23
3.4.7 Генерация Word-документа 23
3.4.8 Механизм фильтрации данных 23
3.4.9 Локальное хранение шаблонов отчётов 24
3.4.10Интеграция с Docker-окружением 24
3.5 Технологии 24
3.5.1 Java 17 24
3.5.2 Spring Boot 25
3.5.3 Apache POI XWPF 25

3.5.4 PostgreSQL 25
3.5.5 Hibernate / JPA 25
3.5.6 Docker 26
3.5.7 Swagger UI 26
3.5.8 Maven 26
3.6 Вывод 26
Заключение 28

ВВЕДЕНИЕ

Производственная практика была пройдена на предприятии ОАО «ЦНИИТУ» — Центральном научно-исследовательском институте организации и техники управления, которое является одним из ведущих научно-исследовательских центров Республики Беларусь в области разработки и внедрения автоматизированных систем управления (АСУ), информационных технологий и программных комплексов. Институт с момента своего основания в 1961 году играет ключевую роль в становлении и развитии информатизации промышленности, государственного управления и других важных направлений цифровой трансформации экономики страны.

За более чем пятидесятилетнюю историю ОАО «ЦНИИТУ» накопил уникальный опыт в создании методологических основ проектирования АСУ, разработке стандартов и руководящих материалов по их внедрению, а также в практическом применении современных информационных технологий для решения задач повышения эффективности управления производственными процессами. В рамках своей деятельности предприятие сотрудничает как с отечественными организациями, так и с международными компаниями, реализует проекты в интересах органов государственного управления, крупных промышленных холдингов и предприятий различных отраслей.

Практика в ОАО «ЦНИИТУ» предоставила возможность не только познакомиться с работой одного из флагманов белорусской информатизации, но и получить реальный практический опыт в сфере проектирования и внедрения автоматизированных систем управления, в том числе в части анализа данных, формирования отчетности и разработки модулей импортозамещения. Это позволило студенту приблизиться к решению задач, характерных для профессиональной деятельности специалиста в области информационных технологий и управления.

Целью производственной практики являлось ознакомление с деятельностью предприятия, структурой и организацией работы подразделений, изучение нормативно-правовой базы, регулирующей деятельность института, а также приобретение практических навыков в анализе и разработке компонентов информационных систем, связанных с обработкой и представлением данных.

Для достижения указанной цели были поставлены следующие задачи:

1. Ознакомиться со структурой предприятия, внутренним распорядком, техникой безопасности, производственными процессами и автоматизированными системами предприятия, на котором проводится практика;

2. Изучить основные технические и нормативно-правовые акты, действующие на предприятии, включая стандарты и методологии проектирования информационных систем;

3. Провести анализ существующих средств и методов автоматизиро-

ванного формирования отчётов на основе источников данных, используемых в текущих проектах института;

4. Разработать модель модуля экспорта данных и автоматизированного формирования отчётов по импортозамещающей продукции национальных предприятий, ориентируясь на современные подходы к построению информационных систем;

5. Подготовить отчёт по производственной (технологической) практике, содержащий описание выполненных работ, полученные результаты и рекомендации.

Реализация данных задач способствовала углублённому пониманию принципов функционирования научно-исследовательского института, особенностей его деятельности в условиях быстро меняющегося информационного пространства, а также формированию у студента профессиональных компетенций в области анализа данных, разработки программных решений и применения современных информационных технологий в управлении промышленными и государственными процессами.

Кроме того, прохождение практики в ОАО «ЦНИИТУ» дало возможность познакомиться с коллективом высококвалифицированных специалистов, работающих над актуальными проектами, а также принять участие в решении реальных задач, имеющих прикладное значение для экономики и развития цифровых технологий в стране.

1 ХАРАКТЕРИСТИКА МЕСТА ПРАКТИКИ

ОАО «ЦНИИТУ» — это многопрофильное предприятие, одно из ведущих научно-исследовательских организаций Республики Беларусь в области разработки и внедрения автоматизированных систем управления общего и специального назначения, информационных технологий, программного обеспечения, а также интеграции информационных ресурсов для промышленности, государственного управления и других сфер.

На предприятии создана уникальная методология проектирования и разработки крупных интегрированных информационных систем, основанная на принципах типизации, унификации и стандартизации проектных решений. В деятельности ЦНИИТУ используются современные технологии, аналогичные применяемым в мировой практике, включая Web-технологии, системы электронного документооборота, хранилища данных, аналитические платформы, средства поддержки принятия решений и другие «ключевые» направления в области информатизации.

Сильной стороной научно-технической политики ОАО «ЦНИИТУ» является системный подход к разработке автоматизированных систем, глубокое знание предметной области, а также наличие собственной нормативно-методической базы, которая соответствует действующим межгосударственным и отраслевым стандартам.

Компания успешно работает на рынке стран ближнего и дальнего зарубежья по разработке и внедрению информационных систем, программных комплексов и решений в интересах промышленности, энергетики, транспорта, социальной сферы и государственного управления. Среди реализованных проектов:

– создание и развитие информационно-вычислительных систем Министерства промышленности («ИВС-Минпром», «ИАС-Минпром»);

– автоматизация управления нефтегазовым комплексом («ИВС-Белнефтехим»);

– разработка и внедрение Web-ориентированных информационных систем («WEB-КСИ»);

– создание портальных решений и систем электронного документооборота;

– автоматизация производственных процессов на предприятиях машиностроения, приборостроения, химической промышленности и других отраслях;

– участие в проектах Парка высоких технологий;

– выполнение международных НИОКР, в том числе в рамках программы ЮНИДО.

Продукция ОАО «ЦНИИТУ» в последнее время включает в себя:

– программные комплексы и автоматизированные системы для управ-

ления промышленными и нефтехимическими предприятиями;

– системы безналичных расчетов, учета и контроля на транспорте и в торговле;

– решения для электронного документооборота с использованием электронной цифровой подписи;

– информационно-аналитические системы для мониторинга, прогнозирования и поддержки управленческих решений;

– технологии интеграции информационных ресурсов, включая централизованные и распределенные базы данных, хранилища данных, информационные кластеры.

Предприятие активно сотрудничает с множеством компаний и организаций, такими как:

– Государственный комитет по науке и технологиям Республики Беларусь;

– Министерство промышленности Республики Беларусь;

– Концерн «Белнефтехим»;

– Парк высоких технологий;

– Высшие учебные заведения: БГУ, БГУИР, БНТУ и др.;

– Международные организации и научные центры стран СЭВ.

ОАО «ЦНИИТУ» обладает значительным интеллектуальным потенциалом, накопленным за десятилетия работы. Здесь трудились и продолжают работать кандидаты и доктора наук, лауреаты государственных премий, авторы сотен публикаций и изобретений. Многие выпускники и сотрудники ЦНИИТУ стали известными специалистами в области информационных технологий, экономики и управления.

Таким образом, ОАО «ЦНИИТУ» остается флагманом информатизации, сохраняя статус головного научного центра в области автоматизации управления и информационных технологий. Его опыт, компетенции и технологии остаются актуальными в условиях стремительно меняющегося цифрового мира.

2 АНАЛИЗ СРЕДСТВ И МЕТОДОВ АВТОМАТИЗИРОВАННОГО ФОРМИРОВАНИЯ ОТЧЁТОВ НА ОСНОВЕ ИСТОЧНИКА ДАННЫХ

2.1 Анализ предметной области автоматизированного формирования отчётов

Автоматизированное формирование отчётов — это процесс сбора, обработки и представления информации в виде документа, удобного для анализа и принятия решений. Такие отчёты могут быть как регулярными (ежедневными, еженедельными, ежемесячными), так и разовыми, создаваемыми по запросу пользователя. Они широко используются в различных сферах деятельности: бизнес-аналитике, государственном управлении, образовании, здравоохранении, промышленности и других областях.

Формирование отчётов становится особенно актуальным при работе с большими объемами данных, требующих точности, скорости и стандартизации. Автоматизация этого процесса позволяет:

– минимизировать человеческие ошибки;

– повысить скорость подготовки отчётов;

– упростить доступ к данным;

– обеспечить гибкость в выборке и фильтрации информации;

– сократить время на подготовку документов для руководства и внешних заказчиков.

Преимущества автоматизированного формирования отчётов:

– **Скорость получения информации**: данные собираются и обрабатываются программно, без участия человека, что значительно ускоряет процесс.

– **Точность и достоверность**: исключается вероятность ошибок при ручном вводе или переносе данных.

– **Стандартизация форматов**: отчёты создаются в заранее определённых шаблонах, соответствующих установленным стандартам.

– **Гибкость в настройке**: пользователь может задать параметры фильтрации, группировки, временные периоды и другие условия.

– **Интеграция с различными системами**: возможность работы с базами данных, API, файлами и другими источниками данных.

В отличие от универсальных решений, ориентированных на широкий круг пользователей, специализированные системы формирования отчётов должны учитывать особенности конкретной предметной области, организационной структуры и используемых информационных систем. Это особенно важно при разработке решений, предназначенных для взаимодействия с

государственными органами, промышленными предприятиями или научно-исследовательскими организациями.

Целью создания модуля автоматизированного формирования отчётов является повышение эффективности работы с данными за счёт программной реализации механизма экспорта, обработки и оформления информации. Такая система должна обеспечивать:

– интеграцию с источниками данных;
– возможность динамической фильтрации и агрегации информации;
– экспорт в различные форматы (в том числе Excel, PDF, DOCX);
– поддержку шаблонов и конфигураций;
– простоту внедрения и масштабируемость решения.

На практике формирование отчётов часто сталкивается с рядом трудностей:

– сложность подключения к разнородным источникам данных;
– необходимость обработки больших объёмов информации;
– требования к точности и своевременности предоставления отчётов;
– необходимость соответствия внутренним и внешним нормативам оформления;
– ограниченная гибкость готовых решений.

Для достижения высокой степени автоматизации и гибкости в рамках проекта выбрано решение, основанное на использовании языка программирования Java и библиотеки Apache POI. Такой подход позволяет полностью контролировать логику формирования отчётов, работать с различными типами данных и обеспечивает независимость от стороннего программного обеспечения.

2.2 Анализ подходов к реализации модулей формирования отчётов

Разработка модуля автоматизированного формирования отчётов предполагает выбор технологии, которая будет использоваться для извлечения, обработки и экспорта данных. Существует несколько подходов к реализации таких модулей, каждый из которых имеет свои преимущества и ограничения.

2.2.1 Использование BI-систем и отчетных инструментов

BI-платформы, такие как Power BI, QlikView, Tableau, предоставляют мощные средства для визуализации и формирования отчётов. Их основные характеристики:

– встроенные шаблоны и визуализации;
– возможность подключения к различным источникам данных;

– интерактивный интерфейс для анализа данных;
– автоматическое обновление отчётов;
– возможность публикации и совместного использования.

Однако использование BI-систем связано с рядом ограничений:

– зависимость от лицензий и платформы;
– сложности при работе с нестандартными форматами;
– ограниченная гибкость в настройке логики обработки данных;
– необходимость обучения пользователей работе с платформой.

Эти системы особенно эффективны в средах, где важна наглядность представления информации и интерактивный анализ данных, но могут быть менее удобны при необходимости строгого соответствия внешним стандартам оформления отчётов или при необходимости глубокой кастомизации логики обработки данных.

2.2.2 Шаблонизаторы отчётов

Инструменты, такие как JasperReports, BIRT, iReport, позволяют создавать отчёты на основе SQL-запросов и шаблонов. Они обеспечивают:

– гибкую настройку форматов вывода;
– работу с различными источниками данных;
– поддержку нескольких форматов экспорта (PDF, Excel, HTML).

К недостаткам можно отнести:

– сложность настройки и разработки шаблонов;
– необходимость знаний специализированных языков и форматов;
– относительно высокую стоимость внедрения и сопровождения.

Такие решения находят применение в корпоративной среде, где требуется регулярное формирование стандартизированных отчётов по заданным шаблонам. Однако они требуют определённого уровня квалификации от разработчиков и администраторов.

2.2.3 Программная реализация на языках общего назначения

При использовании языков программирования, таких как Java, Python или C#, разрабатывается собственная логика формирования отчётов. Основные преимущества:

– полный контроль над логикой обработки и формирования отчётов;
– возможность работы с любыми источниками данных;
– гибкость в настройке форматов и структуры отчётов;
– отсутствие зависимости от сторонних инструментов.

Недостатки:

– необходимость разработки всех компонентов с нуля;

– более высокие трудозатраты на начальном этапе;

– необходимость глубокого понимания структуры данных и форматов.

Данный подход позволяет создать максимально адаптированное решение, однако требует значительных ресурсов на проектирование и реализацию. Он может быть целесообразен в случаях, когда ни одно из готовых решений не удовлетворяет всем требованиям заказчика.

2.2.4 Облачные и SaaS-решения

С развитием облачных технологий появились сервисы, позволяющие формировать отчёты через веб-интерфейсы без установки программного обеспечения. Примеры: Google Data Studio, Microsoft Power BI Service, Looker. Преимущества:

– минимальные усилия на настройку и запуск;

– доступ к данным через браузер;

– мгновенное обновление и масштабируемость;

– интеграция с другими облачными сервисами.

Ограничения:

– зависимость от интернет-соединения;

– ограничения по безопасности и конфиденциальности данных;

– ограниченная возможность настройки логики обработки;

– возможная привязка к конкретному провайдеру.

Эти инструменты популярны среди малого бизнеса и проектов, где важна скорость внедрения и простота использования, но могут быть неприемлемы в условиях, где требуется работа с конфиденциальной информацией или строгое соответствие внутренним нормативам.

2.2.5 Выбор технологии: факторы и критерии

На основе проведённого анализа можно выделить следующие ключевые критерии выбора подхода к реализации модуля формирования отчётов:

1. **Формат выходных документов**: важнейший параметр, так как во многих организациях предпочтение отдается формату Excel или PDF.

2. **Источники данных**: необходимо учитывать типы источников (локальные базы данных, облачные хранилища, API), а также их доступность и производительность.

3. **Гибкость настройки**: возможность изменения логики фильтрации, группировки и агрегации данных.

4. **Сложность внедрения**: время, необходимое на интеграцию и настройку выбранного решения.

5. **Зависимость от стороннего ПО**: наличие лицензионных ограничений и требований к серверной инфраструктуре.

6. **Безопасность и конфиденциальность**: особенно актуально при работе с государственными или коммерческими данными.

7. **Масштабируемость**: способность решения работать с увеличением объёмов данных и количества пользователей.

Каждый из рассмотренных подходов имеет свои сильные стороны и ограничения, что делает его более или менее подходящим в зависимости от условий и требований конкретного проекта. В следующем подразделе будет проведён сравнительный анализ этих подходов и даны рекомендации по их применению в различных сценариях.

2.3 Анализ аналогов систем формирования отчётов

Анализ существующих решений, применяемых для автоматизированного формирования отчётов, позволяет выделить ключевые особенности, преимущества и недостатки различных подходов. В данном подразделе рассматриваются три наиболее распространённых инструмента: JasperReports, Microsoft Power BI и программная реализация на Python с использованием библиотеки Pandas.

2.3.1 JasperReports

JasperReports[1] — это один из самых популярных open-source инструментов для создания отчётов, разработанный компанией TIBCO Software Inc. Он предназначен для генерации отчётов в различных форматах (PDF, Excel, HTML, CSV и др.) и активно используется в корпоративной среде.

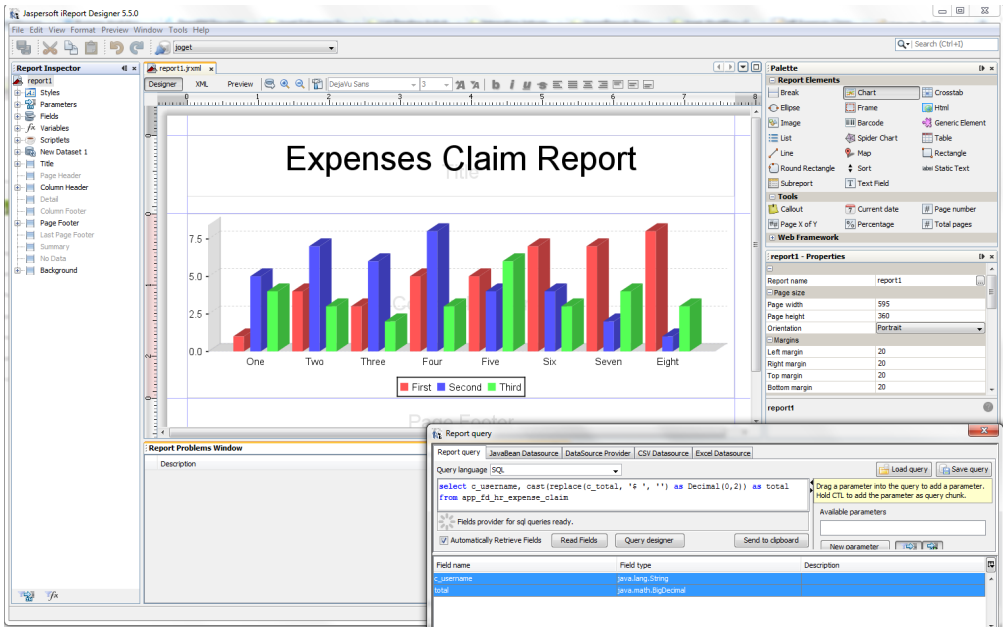

Рисунок 2.1 – Интерфейс JasperReports Studio

Преимущества JasperReports:

1. *Поддержка множества форматов вывода* — позволяет экспортировать отчёты в PDF, XLSX, DOCX, HTML, RTF, CSV.

2. *Гибкая система шаблонов* — предоставляет мощный редактор для создания сложных макетов отчётов.

3. *Интеграция с базами данных* — возможность подключения к SQL, Hibernate, XML, JSON и другим источникам данных.

4. *Работа в составе приложений* — может быть встроен в Java-приложения через API.

5. *Открытый исходный код* — бесплатное использование и поддержка сообществом.

Недостатки JasperReports:

1. *Сложность освоения* — требует знания специализированных форматов и языков (JRXML).

2. *Ограниченная поддержка динамических данных* — сложности при работе с изменяющимися структурами данных.

3. *Необходимость дополнительных библиотек* — для полной функциональности требуется установка дополнительных модулей (например, iText для PDF).

4. *Малая производительность на больших объёмах данных* — медленная обработка при работе с большими выборками.

5. *Требуется серверная инфраструктура* — для запуска веб-сервисов необходимо разворачивать сервер (JasperServer).

2.3.2 Microsoft Power BI

Power BI[2] — это аналитическая платформа от Microsoft, позволяющая не только формировать отчёты, но и создавать интерактивные дашборды, визуализации и KPI-панели. Используется как в облачном, так и в локальном варианте.

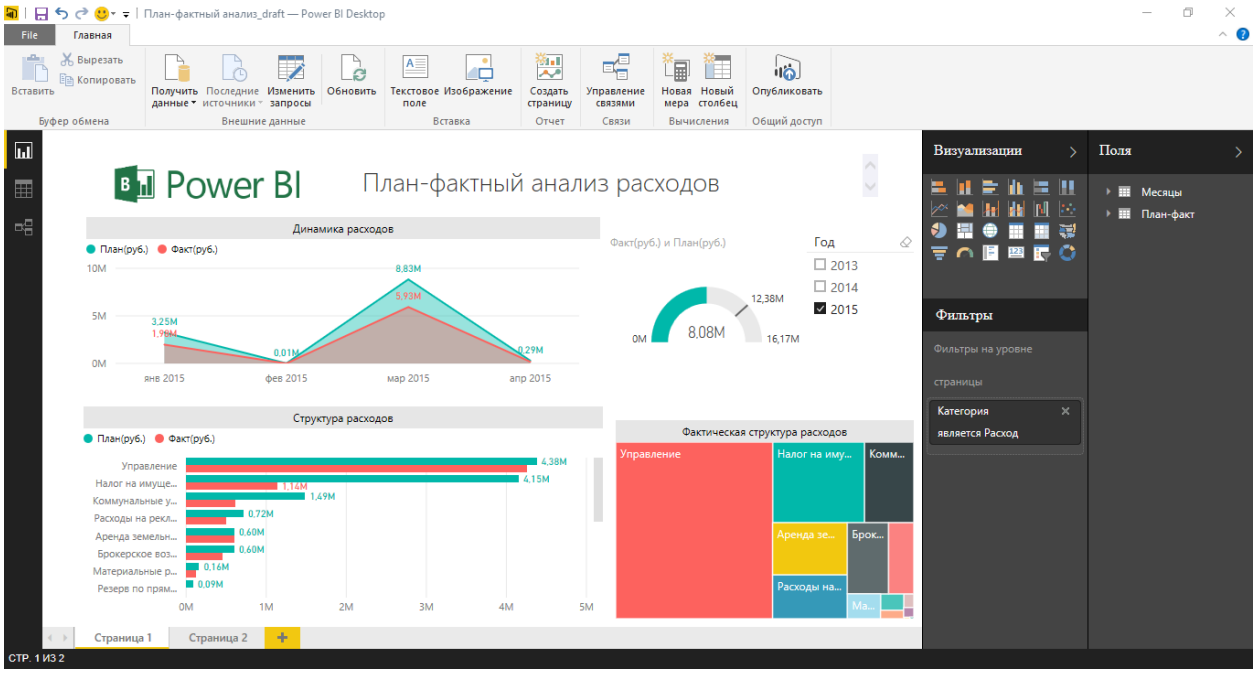

Рисунок 2.2 – Интерфейс Microsoft Power BI

Преимущества Microsoft Power BI:

1. *Интерактивная визуализация* — поддерживает построение графиков, диаграмм, сводных таблиц и дашбордов.
2. *Облачная и локальная версия* — возможность работы как в облаке, так и на локальных серверах.
3. *Поддержка множества источников данных* — доступ к данным из Excel, SQL Server, Oracle, SharePoint, REST API и других источников.
4. *Автоматизация обновления отчётов* — настраивается расписание регулярного обновления данных.
5. *Удобный пользовательский интерфейс* — интуитивно понятный конструктор отчётов и дашбордов.

Недостатки Microsoft Power BI:

1. *Зависимость от Windows-платформы* — хотя есть клиенты под macOS и мобильные устройства, основная функциональность ориентирована на Windows.
2. *Платное лицензирование* — для полноценного использования необходимы подписки Pro или Premium.
3. *Сложности с кастомизацией отчётов* — ограниченная возможность написания собственной логики формирования.
4. *Высокие требования к ресурсам* — требует мощного оборудования для работы с большими объёмами данных.
5. *Ограничения в свободе экспорта* — некоторые форматы экспорта ограничены или требуют доработки.

2.3.3 Python + Pandas

Python вместе с библиотекой Pandas[3] представляет собой гибкое программное решение для работы с данными и формирования отчётов. Это подход, который позволяет разрабатывать полностью кастомизированные решения без привязки к конкретной платформе.

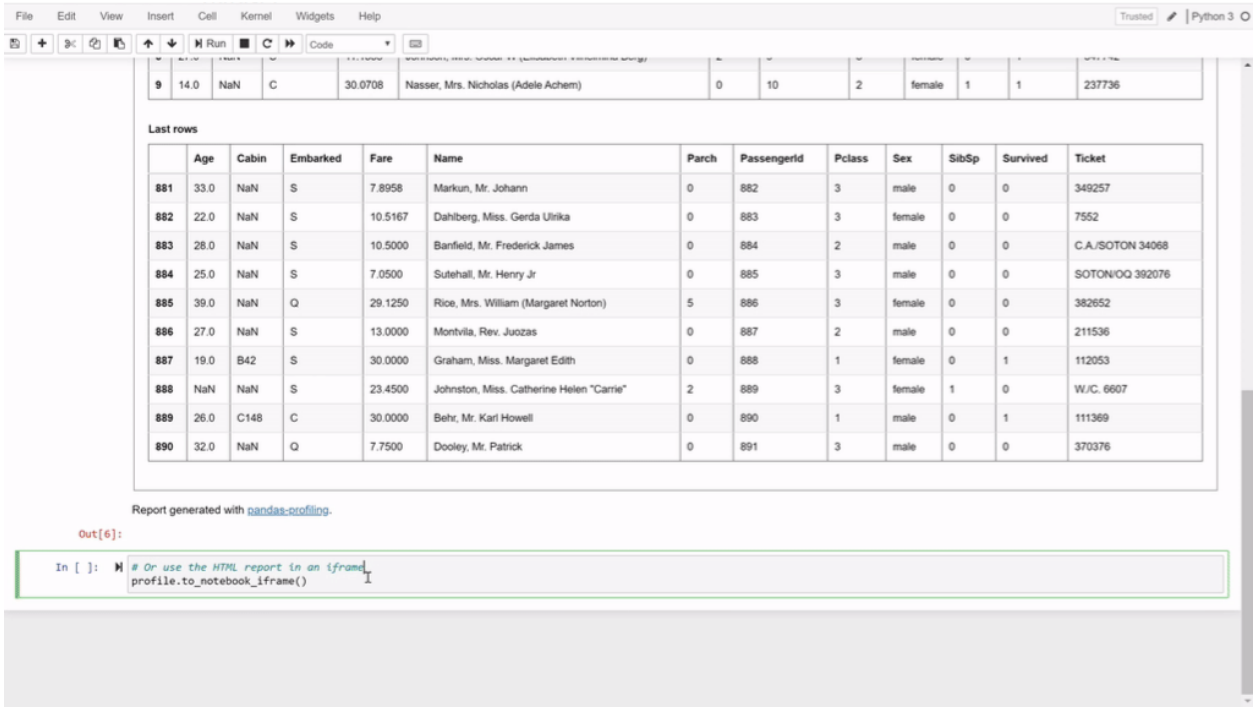

Рисунок 2.3 – Пример использования Pandas и Python для формирования отчётов

Преимущества Python + Pandas:

1. *Полный контроль над процессом формирования отчётов* — возможность реализовать любую логику обработки и фильтрации данных.
2. *Большое количество библиотек* — поддержка matplotlib, seaborn, openpyxl, pdfkit и других для визуализации и экспорта.
3. *Открытый исходный код* — бесплатное использование и широкое сообщество разработчиков.
4. *Кроссплатформенность* — работает на Windows, Linux, macOS.
5. *Легко интегрируется с другими технологиями* — может быть частью web-приложений, ETL-процессов, скриптовых задач и т.д.

Недостатки Python + Pandas:

1. *Требует программирования* — необходимо знание Python и библиотек.
2. *Сравнительно низкая скорость обработки больших наборов данных* — особенно при использовании циклов вместо векторизации.
3. *Нужны дополнительные компоненты для оформления отчётов* — экспорт в Excel, PDF и другие форматы требует подключения сторонних

библиотек.

4. *Не имеет встроенного UI* — отсутствие готового интерфейса для пользователей.

5. *Сложности масштабирования* — без правильной архитектуры сложно использовать в крупных проектах.

2.4 Вывод

В ходе выполнения данного раздела был проведён комплексный анализ задачи автоматизированного формирования отчётов на основе источника данных. Были рассмотрены ключевые аспекты предметной области, включая цели, функции и проблемы формирования отчётов, изучены существующие подходы к реализации таких систем, проанализированы аналоги и выявлены их основные ограничения.

На основе анализа предметной области установлено, что автоматизация формирования отчётов позволяет значительно повысить точность, скорость и стандартизацию представления информации. Однако эта задача осложняется разнородностью источников данных, необходимостью соответствия внешним и внутренним стандартам, высокими требованиями к достоверности и безопасностью обработки конфиденциальной информации.

Исследование существующих подходов показало, что:

BI-системы (например, Power BI) обеспечивают удобство визуализации и интерактивности, но имеют ограниченную гибкость при работе с нестандартными форматами. Шаблонизаторы отчётов (например, JasperReports) предоставляют мощные средства формирования структурированных документов, однако требуют значительных трудозатрат на настройку и имеют низкую производительность на больших объёмах данных. Программная реализация на языках общего назначения (например, Python + Pandas) обеспечивает максимальную гибкость и контроль над логикой, но требует глубокого технического понимания и дополнительных усилий для реализации пользовательского интерфейса и экспорта в нужные форматы.

Анализ аналогов позволил выявить типичные недостатки современных решений:

– *Ограниченная гибкость*: большинство готовых решений не позволяют легко адаптировать логику фильтрации, группировки и оформления отчётов под конкретные бизнес-требования.

– *Проблемы с интеграцией*: многие системы плохо взаимодействуют с разнородными источниками данных или требуют предварительного преобразования форматов.

– *Зависимость от платформы и лицензий*: использование коммерческих решений связано с затратами на подписку, привязкой к операционной системе и ограничениями масштабируемости.

– *Недостаточная производительность*: при работе с большими объёмами данных наблюдается снижение скорости обработки и увеличение времени формирования отчётов.

– *Сложности с обеспечением безопасности*: особенно у облачных решений, где данные передаются на серверы третьих лиц, что делает их неприемлемыми при работе с конфиденциальной информацией.

– *Отсутствие полной автономности*: некоторые решения требуют постоянного интернет-соединения, что ограничивает их применимость в условиях ограниченного доступа к сети.

Таким образом, ни одно из существующих решений не является универсальным и полностью удовлетворяющим всем возможным требованиям. Каждое из них имеет определённые компромиссы между простотой использования, гибкостью, производительностью и уровнем безопасности.

Выявленные недостатки существующих решений подтверждают актуальность разработки собственного программного модуля, который позволит:

– полностью контролировать логику обработки и формирования отчётов;

– работать с любыми источниками данных;

– обеспечивать высокую степень защиты информации;

– использовать открытые технологии и библиотеки;

– минимизировать зависимость от стороннего программного обеспечения.

Таким образом, выявленные недостатки существующих решений подтверждают актуальность разработки собственного программного модуля для автоматизированного формирования отчётов. Такой подход позволит создать систему, которая обеспечит максимальную гибкость в настройке логики обработки данных, поддержку различных источников информации, высокую степень защиты и независимость от сторонних платформ и лицензий. Это требует поиска эффективной архитектуры и технологической базы, способной минимизировать компромиссы между производительностью, простотой сопровождения и функциональными возможностями, а также нивелировать ключевые ограничения уже рассмотренных аналогов.

3 ПРОЕКТИРОВАНИЕ И РЕАЛИЗАЦИЯ МОДУЛЯ ФОРМИРОВАНИЯ ОТЧЁТОВ

3.1 Постановка задачи

3.1.1 Цель работы

Целью разработки модуля является реализация функционала формирования отчётов в формате DOCX на основе данных о продукции предприятия, с возможностью выбора пользователем одного из нескольких доступных шаблонов и фильтрации по следующим параметрам:

– Наименование продукта
– Тип продукции
– Признак импортозамещения

Модуль интегрируется в существующее Spring Boot приложение, использует данные из PostgreSQL и предоставляет REST API для последующей интеграции с фронтендом. Все отчёты формируются динамически на основе актуальных данных из БД.

3.1.2 Задачи модуля

Для достижения цели были определены следующие задачи:

– Предоставлять REST API методы для генерации отчётов
– Поддерживать несколько шаблонов формирования документов
– Использовать фильтры, переданные через URL-параметры
– Формировать документы в формате .docx с таблицами и заголовками
– Обеспечивать скачивание файла клиентом после формирования
– Работать в Docker-контейнере и взаимодействовать с локальной БД
– Возвращать корректные HTTP-статусы и сообщения об ошибках

3.2 Потенциальные пользователи

Модуль предназначен для использования сотрудниками предприятий, занимающихся анализом и отчетностью по выпускаемой продукции. Основные категории пользователей:

– **Инженеры и специалисты по качеству.** Используют отчёты для анализа выпускаемых изделий;

– **Финансовые и планово-экономические отделы.** Формируют документы для внутреннего использования;

– **Руководители и менеджеры проектов.** Получают структурированную информацию для принятия решений;

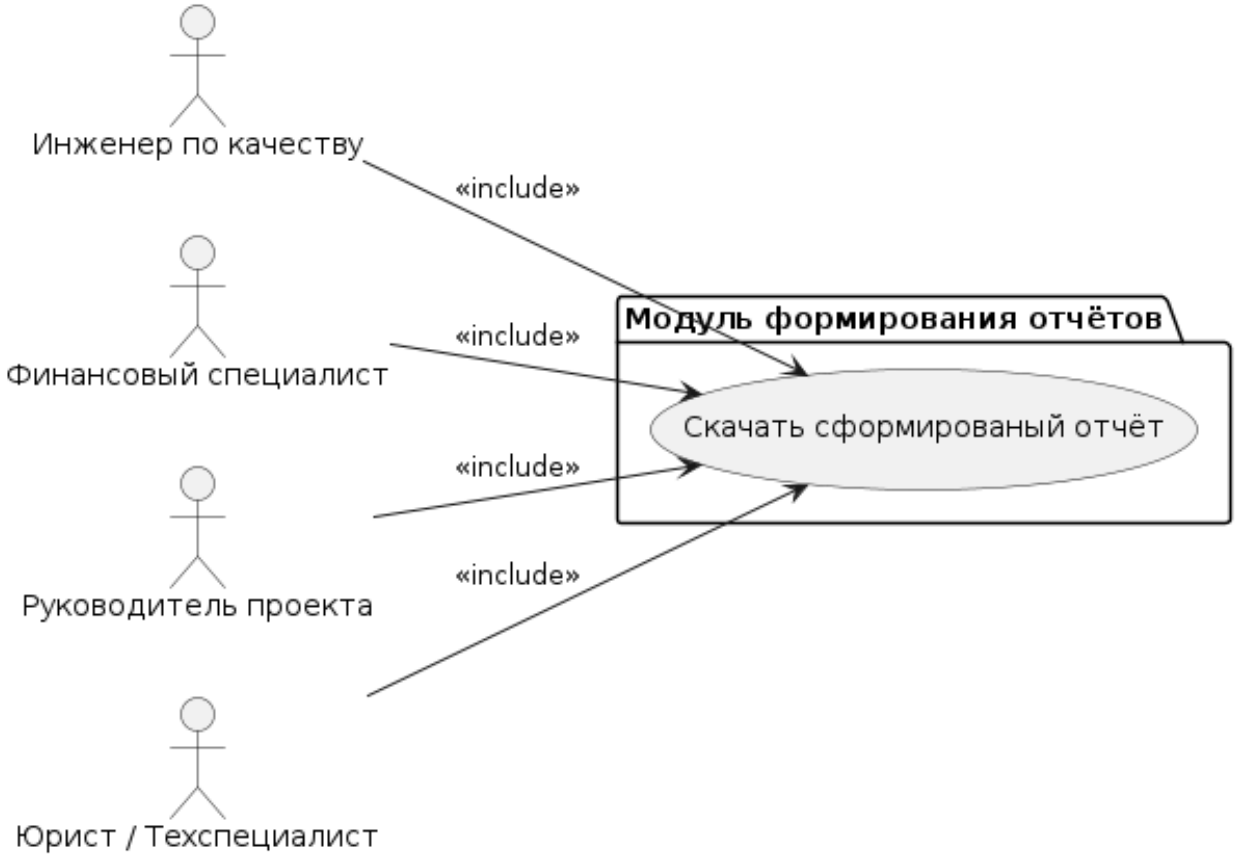

Рисунок 3.1 – Use-Case диаграмма модуля формирования отчётов

– **Юристы и технические специалисты.** Используют документы для отчётности и сертификации.
Модуль обеспечивает удобство подготовки официальных документов без необходимости ручного формирования таблиц и заголовков.

3.3 Общая архитектура модуля

Модуль формирования отчётов представляет собой модульную архитектуру (рис. 3.2) и включающую в себя:

– **Контроллер формирования отчёта**. Обеспечивает обработку запросов пользователя на генерацию отчётов. Принимает параметры фильтрации и шаблона, вызывает соответствующую реализацию сервиса и возвращает клиенту готовый к скачиванию Word-документ.

– **Сервисный слой (ReportService)**. Содержит бизнес-логику формирования отчёта. Выполняет обращение к данным, заполнение документа, создание таблицы и структурирование информации. Сервис использует DTO для передачи данных между слоями и обеспечивает унифицированный интерфейс для всех типов отчётов.

– **Репозиторий получения данных**. Отвечает за выборку продукции из базы данных PostgreSQL. Поддерживает динамическую фильтрацию

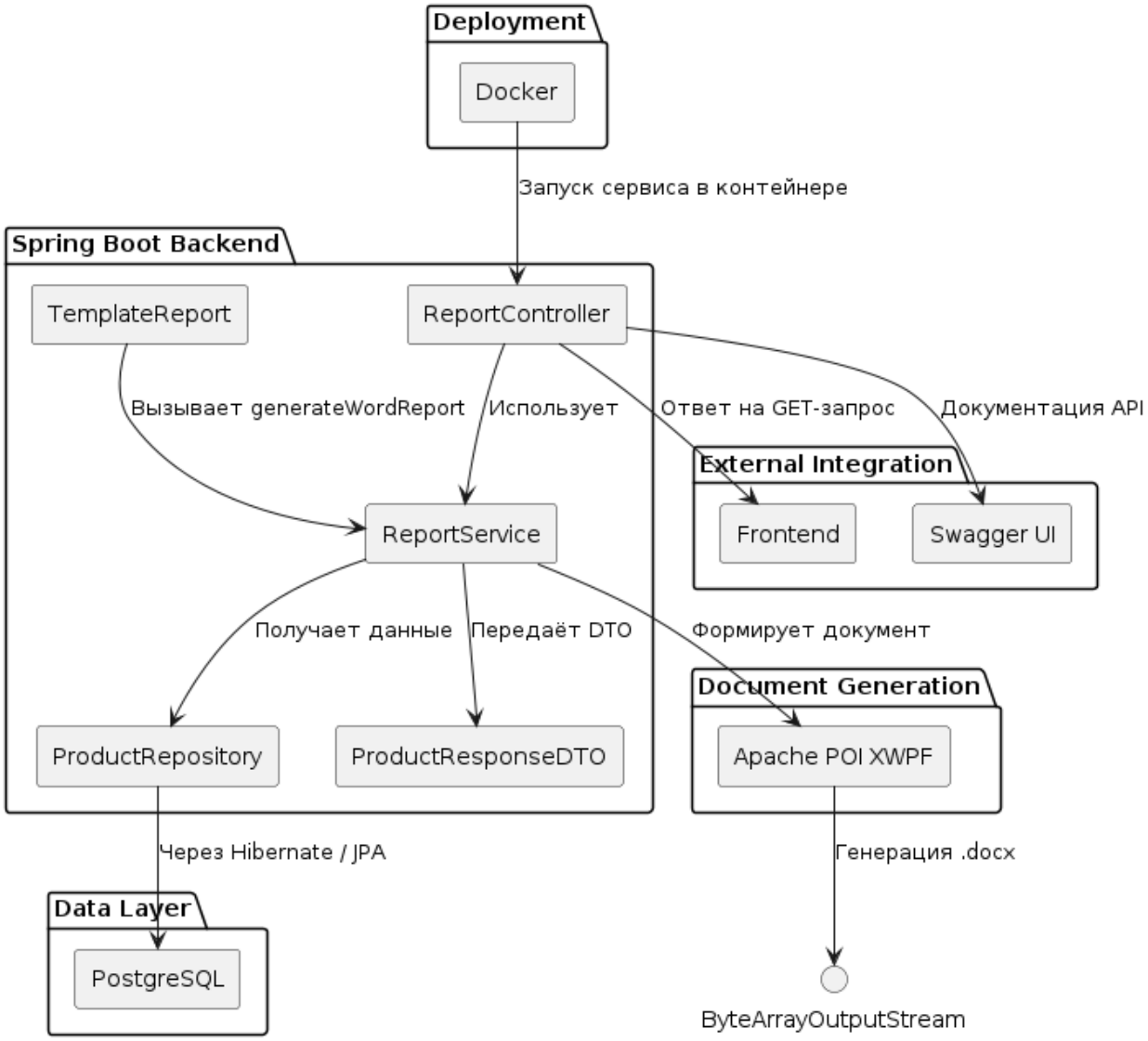

Рисунок 3.2 – Архитектура модуля формирования отчётов

по наименованию, типу и признаку импортозамещения. Интегрирован через JPA/Hibernate.

– **Шаблонный движок (TemplateReport)**. Представляет собой реализации стратегий формирования отчётов. Каждый шаблон содержит уникальные особенности оформления или структуры, но использует общие методы сервиса для генерации документа. Позволяет добавлять новые шаблоны без изменения контроллера или основной логики сервиса.

– **DTO (ProductResponseDTO)**. Объект передачи данных, который используется для хранения и передачи информации о продуктах между уровнями системы.

– **База данных (PostgreSQL)**. Используется для хранения информации о продукции, предприятиях и справочных данных (ОКРБ, ТН ВЭД). Доступ к данным осуществляется через Hibernate / JPA.

– **Генерация документа (Apache POI XWPF)**. Библиотека Apache POI применяется для создания и заполнения документов формата DOCX. Через неё реализуются операции над текстом, таблицами и стилями в отчёте.

– **Интеграция с фронтендом**. Модуль предоставляет точку взаимодействия для фронтенда, через которую фронтенд может отправить запрос на формирование отчёта с указанием фильтров и шаблона.

– **Документация API (Swagger UI)**. Для описания и тестирования функционала модуля используется Swagger UI, интегрированный в Spring Boot приложение. Это позволяет разработчикам и тестировщикам проверять работу эндпоинтов без сторонних инструментов.

– **Запуск в контейнере (Docker)**. Модуль упакован в Docker-образ и предназначен для запуска в контейнере. Он взаимодействует с локальной базой данных через специальное имя host.docker.internal, что позволяет использовать уже запущенную на хостовой машине PostgreSQL.

Архитектура спроектирована таким образом, чтобы обеспечить:

– **Чёткое разделение уровней**: каждый компонент выполняет свою роль, не нарушая принцип единственной ответственности.

– **Поддержка нескольких шаблонов**: за счёт паттерна «Стратегия» можно легко добавлять новые виды отчётов.

– **Формирование документа на основе фильтрованных данных**: отчёт всегда создаётся на основе актуального состояния системы.

– **Интеграция с внешними системами**: через REST API и внедрённые библиотеки.

– **Простота масштабирования**: добавление новых шаблонов, форматов или источников данных не требует переписывания всего модуля.

3.4 Описание компонентов

3.4.1 Модуль контроллера

Контроллер обеспечивает обработку HTTP-запросов на формирование отчётов. Он принимает параметры фильтрации и идентификатор шаблона, использует внедрённые стратегии формирования отчётов и возвращает сформированный Word-документ клиенту. Метод контроллера поддерживает скачивание файла через браузер с применением корректных заголовков. Контроллер интегрирован в REST API и автоматически документируется через Swagger UI, что позволяет тестировать работу эндпоинта без использования сторонних инструментов.

3.4.2 Сервисный слой

Сервисный слой реализует основную бизнес-логику формирования отчётов. Он отвечает за:

– Получение данных из репозитория;

– Создание нового Word-документа;

– Добавление заголовков, таблиц и других элементов оформления;

– Заполнение таблицы актуальными данными о продукции и предприятиях;

– Возврат документа в виде потока байтов для дальнейшей передачи клиенту.

Все операции выполняются в памяти, что исключает необходимость сохранения временных файлов и ускоряет процесс генерации отчёта.

3.4.3 Интерфейс стратегии формирования отчёта

Для поддержки нескольких шаблонов отчётов используется интерфейс, задающий контракт для всех реализаций формирования документов. Этот интерфейс определяет метод, который должен быть реализован каждой конкретной стратегией. Применение интерфейса позволяет системе динамически выбирать нужный способ формирования отчёта на основе параметра запроса, не изменяя при этом сам контроллер или сервис.

3.4.4 Реализации стратегий

Каждая реализация стратегии представляет собой отдельный класс, соответствующий одному шаблону отчёта. Они используют общие методы из сервисного слоя, но могут отличаться порядком заполнения, структурой документа или дополнительным оформлением. Добавление новых шаблонов сводится к созданию новой реализации без изменения существующего кода,

что полностью соответствует принципу открытости/закрытости объектно-ориентированного проектирования.

3.4.5 Репозиторий получения данных

Репозиторий обеспечивает выборку данных о продукции из БД с учётом заданных пользователем фильтров. Репозиторий реализован как JPA-интерфейс, использующий JPQL-запросы. Если параметр фильтра отсутствует, условие игнорируется, и данные возвращаются без ограничений по данному полю.

3.4.6 DTO (объект передачи данных)

DTO-объект служит универсальным контейнером данных, используемых при формировании отчётов. В него входят следующие поля:

– Идентификатор продукта;
– Наименование продукта;
– Тип продукции;
– Название предприятия;
– Адрес предприятия;
– Признак импортозамещения.

Такой подход позволяет отделить логику формирования отчёта от структуры базы данных и сделать сервис более устойчивым к изменениям в модели данных.

3.4.7 Генерация Word-документа

Формирование отчёта осуществляется с использованием библиотеки Apache POI XWPF, которая предоставляет возможность программного создания и манипуляций с документами формата DOCX.

Процесс генерации включает:

– Создание нового документа;
– Добавление заголовков и подзаголовков;
– Формирование таблицы с заданными колонками;
– Заполнение таблицы данными о продукции.

Сгенерированный отчёт возвращается в виде массива байтов, что позволяет клиенту сразу начать его скачивание без промежуточного сохранения на сервере.

3.4.8 Механизм фильтрации данных

Фильтрация данных реализуется на уровне репозитория и учитывает следующие параметры:

– Наименование продукта;
– Тип продукции;
– Признак импортозамещения.

Пользовательские фильтры передаются в запросе и применяются только при наличии значения.

Такой подход делает выборку гибкой и эффективной, позволяя формировать отчёты как по всем данным, так и по уточнённым условиям.

3.4.9 Локальное хранение шаблонов отчётов

Шаблоны отчётов могут быть реализованы как в виде Java-классов, так и в будущем — как готовые DOCX-файлы, хранящиеся в директории ресурсов.

Это позволит упростить настройку внешнего вида отчёта и исключит необходимость правки кода при изменении оформления.

На данном этапе шаблоны реализуются через вызовы соответствующих методов в сервисе, которые различаются лишь способом заполнения документа, но используют один и тот же источник данных.

3.4.10 Интеграция с Docker-окружением

Модуль формирования отчётов спроектирован с учётом современных подходов к контейнеризации и может быть успешно запущен внутри Docker-контейнера.

Для взаимодействия с базой данных используется специальный DNS-адрес host.docker.internal, который указывает на хостовую машину из контейнера. Это позволяет сервису подключаться к БД запущенной в другом контейнере.

Docker-образ создаётся через стандартный Dockerfile и docker-compose.yml, что делает развёртывание сервиса простым и воспроизводимым. Все зависимости упаковываются вместе с приложением, что гарантирует стабильность работы вне зависимости от окружения.

3.5 Технологии

3.5.1 Java 17

Java[4] — один из самых популярных языков программирования, ориентированный на создание надежного, переносимого и масштабируемого кода. Разработка модуля велась на версии Java 17, которая обеспечивает улучшенную производительность, безопасность и совместимость с современными библиотеками. Язык предоставляет богатый набор возможностей для

работы с объектами, обработки исключений, управления памятью и многопоточности, что делает его идеальным выбором для реализации сложных сервисов в корпоративной среде.

3.5.2 Spring Boot

Spring Boot[5] — это фреймворк с открытым исходным кодом, позволяющий быстро создавать автономные и готовые к использованию микросервисы на базе Spring. Он предоставляет удобную настройку REST API, работу с базами данных через JPA, внедрение зависимостей, а также автоматическое конфигурирование компонентов. С помощью Spring Boot были созданы контроллеры, сервисы и репозитории, обеспечивающие чёткое разделение уровней приложения и простоту масштабирования.

3.5.3 Apache POI XWPF

Apache POI[6] — это библиотека с открытым исходным кодом, предназначенная для работы с офисными документами формата Microsoft Office Open XML. Модуль XWPF этой библиотеки используется для создания и манипуляций с файлами DOCX. Она позволяет динамически строить документы с заголовками, абзацами, таблицами и стилями, что обеспечило формирование отчётов с необходимым оформлением без использования внешних шаблонов.

3.5.4 PostgreSQL

PostgreSQL[7] — это мощная система управления реляционными базами данных с открытым исходным кодом, поддерживающая широкий спектр функций, включая транзакции, триггеры, пользовательские типы данных и JSON-поля. Данная СУБД была выбрана как основное хранилище данных системы, в котором хранятся информация о продукции, предприятиях и справочных классификаторах.

3.5.5 Hibernate / JPA

Hibernate[8] — реализация Java Persistence API (JPA), предоставляющая высокоуровневый интерфейс для работы с реляционными данными в объектно-ориентированном виде. Эта библиотека обеспечивает маппинг между Java-объектами и таблицами в БД, автоматическую генерацию SQL-запросов и управление транзакциями. Интеграция с PostgreSQL через Hibernate позволила эффективно работать с данными и минимизировать написание SQL-запросов вручную.

3.5.6 Docker

Docker[9] — платформа контейнеризации, которая позволяет упаковывать приложения и их зависимости в изолированные контейнеры. Технология активно применялась для тестирования и запуска сервиса в различных окружениях: локально, на сервере или в CI/CD пайплайне. Модуль успешно функционирует внутри контейнера, взаимодействуя как с локальной БД, так и с БД, запущенной в другом контейнере.

3.5.7 Swagger UI

Swagger UI[10] — это веб-интерфейс, предназначенный для документирования и тестирования REST API. Он был интегрирован в Spring Boot приложение через зависимость springdoc-openapi-starter-webmvc-ui, что дало возможность просматривать доступные эндпоинты, тестировать запросы и автоматически генерировать документацию. Это упростило работу команды и ускорило процесс интеграции с фронтендом.

3.5.8 Maven

Maven[11] — это популярная система управления проектами с открытым исходным кодом, разработанная Apache. Она предоставляет стандартный способ организации проекта, управления зависимостями и автоматизации сборки Java-приложений. Maven основана на декларативном подходе к описанию проекта через файл pom.xml, что делает его особенно удобным для командной разработки и интеграции в CI/CD-процессы. Она использовалась для управления зависимостями, автоматической сборки проекта и интеграции с Docker-образами.

3.6 Вывод

Проектирование и разработка модуля формирования отчётов позволили реализовать гибкий и расширяемый компонент системы. Модуль обеспечивает автоматическое создание Word-документов на основе данных, соответствующих заданным критериям фильтрации, с возможностью выбора одного из нескольких шаблонов.

В процессе проектирования был применён паттерн проектирования «Стратегия», который обеспечил:

– Расширяемость функционала без изменения существующего кода;
– Поддержку нескольких шаблонов отчётов;
– Простоту внедрения новых видов отчётов в будущем.

В ходе реализации были выполнены следующие задачи:

– Разработана логика формирования отчётов в формате .docx с использованием библиотеки Apache POI.

– Реализовано динамическое управление шаблонами через интерфейс WordReportStrategy.

– Организована работа с данными из БД через JPA/Hibernate с возможностью фильтрации

– Обеспечена совместимость модуля с PostgreSQL и Docker-окружением.

– Выполнено внедрение в существующее Spring Boot приложение без изменения структуры базы данных или бизнес-логики других модулей.
Особенностью реализации стал подход к формированию отчёта — вся информация передаётся через DTO-объекты, что обеспечило отделение логики формирования документа от модели данных системы.

Таким образом, разработанный модуль представляет собой законченное решение для формирования отчётов в формате Word, которое может быть легко интегрировано в текущую информационную систему. Модуль сочетает в себе современные методики разработки, расширяемость и практичность реализации. Его можно использовать как основу для дальнейшего развития с поддержкой новых форматов, шаблонов и источников данных.

ЗАКЛЮЧЕНИЕ

За время производственной практики были проанализированы современные подходы и технологии разработки программного обеспечения, связанные с автоматизацией формирования отчётов на основе фильтрованных данных. В ходе практики была реализована backend-часть модуля, позволяющая пользователям создавать Word-документы на основе информации о продукции предприятия с поддержкой нескольких шаблонов и возможностью фильтрации по необходимым параметрам.

Модуль был разработан в рамках Spring Boot приложения, использующего PostgreSQL[7] в качестве основного хранилища данных. Для работы с документами применялась библиотека Apache POI XWPF[6], которая позволила генерировать файлы формата DOCX динамически на стороне сервера. Ключевым архитектурным решением стало применение паттерна проектирования «Стратегия», обеспечивающее расширяемость функционала без изменения существующих компонентов. Это сделало возможным добавление новых шаблонов отчётов независимо от уже реализованных, что соответствует принципу открытости/закрытости объектно-ориентированного проектирования.

Особое внимание было уделено совместимости модуля с окружением Docker[9], что обеспечило возможность тестирования и запуска сервиса в изолированной среде. Модуль показал стабильную работу как с локальной базой данных, так и с БД, запущенной внутри контейнера. Благодаря унифицированному REST API эндпоинту, модуль может быть интегрирован в любые системы, взаимодействующие с информационной системой учёта продукции.

Практическая значимость заключается в том, что разработанный модуль позволяет автоматизировать процесс подготовки отчётных документов, исключая ручное формирование таблиц и заголовков. Его можно использовать для анализа выпускаемой продукции, поиска субконтракторов, внутренней отчётности и других задач, где важна актуальность и точность данных.

Таким образом, в рамках производственной практики были изучены и применены современные методики backend-разработки, включая использование Java[4], Spring Boot[5], Hibernate[8] и Apache POI[6]. Разработанный модуль демонстрирует высокую степень расширяемости и готов к использованию в реальных условиях. Он может быть адаптирован для работы с другими форматами отчётов, такими как Excel или PDF, а также использоваться в микросервисной архитектуре. Полученный опыт стал основой для дальнейшего развития в области создания масштабируемых и надёжных решений на базе Spring Boot.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] JasperReports — библиотека генерации отчётов с поддержкой PDF, Excel и других форматов [Электронный ресурс]. — Режим доступа: https://community.jaspersoft.com/project/jasperreports-library. – Date of access: 25.06.2025.

[2] Microsoft Power BI — платформа для визуализации и анализа данных [Электронный ресурс]. — Режим доступа: https://powerbi.microsoft.com/. – Date of access: 25.06.2025.

[3] Pandas — библиотека анализа и обработки данных на Python [Электронный ресурс]. — Режим доступа: https://pandas.pydata.org/. – Date of access: 25.06.2025.

[4] Java Platform, Standard Edition Documentation [Электронный ресурс]. — Режим доступа: https://docs.oracle.com/en/java/javase/17/. – Date of access: 25.06.2025.

[5] Spring Boot — Framework для создания автономных приложений на Java [Электронный ресурс]. — Режим доступа: https://spring.io/projects/spring-boot. – Date of access: 25.06.2025.

[6] Apache POI — Java библиотека для работы с документами Microsoft Office [Электронный ресурс]. — Режим доступа: https://poi.apache.org/. – Date of access: 25.06.2025.

[7] PostgreSQL — самая продвинутая open-source реляционная СУБД [Электронный ресурс]. — Режим доступа: https://www.postgresql.org/. – Date of access: 25.06.2025.

[8] Hibernate ORM — реализация Java Persistence API [Электронный ресурс]. — Режим доступа: https://hibernate.org/orm/. – Date of access: 25.06.2025.

[9] Docker — платформа контейнеризации приложений [Электронный ресурс]. — Режим доступа: https://www.docker.com/. – Date of access: 25.06.2025.

[10] Swagger UI — интерфейс для документирования REST API [Электронный ресурс]. — Режим доступа: https://swagger.io/tools/swagger-ui/. – Date of access: 25.06.2025.

[11] Apache Maven — система управления проектом и автоматизации сборки [Электронный ресурс]. — Режим доступа: https://maven.apache.org/. – Date of access: 25.06.2025.

[12] PlantUML — инструмент для генерации UML-диаграмм из текстового описания [Электронный ресурс]. — Режим доступа: https://plantuml.com/. – Date of access: 25.06.2025.

[13] Project Lombok — автоматизация boilerplate кода в Java [Электронный ресурс]. — Режим доступа: https://projectlombok.org/. – Date of access: 25.06.2025.