|  |
| --- |
| **ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА** |
| **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**  **ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  **«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ВОДНОГО ТРАНСПОРТА»**  **(ФГБОУ ВО «СГУВТ»)** |

Факультет: Электромеханический

Кафедра: Информационных систем

Направление подготовки /специальность: 09.03.02 Информационные системы и технологии

Направленность (профиль)/специализация: Проектирование информационных систем и их компонентов

|  |  |  |
| --- | --- | --- |
| **ДОПУЩЕН К ЗАЩИТЕ** | | |
| Заведующий кафедрой | | |
|  |  | С.В. Моторин |
| (Подпись) |  | (И.О. Фамилия) |
|  | | |
| (Дата) | | |

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

на тему:

Разработка системы сбора и обработки данных публикационной активности\_

преподавателей ФГБОУ ВО «СГУВТ». Клиентский модуль.\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил  обучающийся |  |  | ИТ-201 |  |  | |
|  | (Фамилия И.О.) |  | (группа) |  | (подпись) | |
| Руководитель |  |  | Кандидат тех. наук,  доцент |  |  | |
|  | (Фамилия И.О.) |  | (должность, ученая степень, ученое звание) |  | (подпись) | |
| Консультант |  | | |  |  | |
|  | (Фамилия И.О.) | | |  | (подпись) | |
| Консультант |  | | |  |  | |
|  | (Фамилия И.О.) | | |  | (подпись) | |
| Консультант |  | | |  |  | |
|  | (Фамилия И.О.) | | |  | (подпись) | |
| Нормоконтроль | Голышев Д.Н. | | |  |  | |
|  | (Фамилия И.О.) | | |  | | (подпись) | |

Новосибирск 2024

|  |
| --- |
| **ФЕДЕРАЛЬНОЕ АГЕНТСТВО МОРСКОГО И РЕЧНОГО ТРАНСПОРТА** |
| **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**  **ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**  **«СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ВОДНОГО ТРАНСПОРТА»**  **(ФГБОУ ВО «СГУВТ»)** |

Факультет: Электромеханический

Кафедра: Информационных систем

Направление подготовки /специальность: 09.03.02 Информационные системы и технологии

Направленность (профиль)/специализация: Проектирование информационных систем и их компонентов

|  |  |  |
| --- | --- | --- |
| **УТВЕРЖДАЮ** | | |
| Заведующий кафедрой | | |
|  |  | С.В. Моторин |
| (Подпись) |  | (И.О. Фамилия) |
|  | | |
| (Дата) | | |

**ЗАДАНИЕ**

на выпускную квалификационную работу

обучающемуся \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(фамилия, имя, отчество)

Тема работы \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Утверждена приказом ректора № \_\_\_\_\_\_\_\_ от «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Срок сдачи законченной работы «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

Исходные данные \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Содержание пояснительной записки (перечень вопросов, подлежащих разработке) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перечень графического материала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Консультанты по работе (указать относящиеся к ним разделы) \_\_\_\_\_\_\_\_\_\_\_\_

\_Безопасность жизнедеятельности\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_/ Е.А.Пахомов /\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Дата выдачи задания «\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024 г.

|  |  |  |
| --- | --- | --- |
| Руководитель |  | Обучающийся |
|  |  |  |
| (фамилия, имя, отчество) |  | (фамилия, имя, отчество) |
| Кандидат технических наук,  доцент |  | ИТ-201 |
| (учёная степень, учёное звание) |  | (группа) |
|  |  |  |
| (подпись, дата) |  | (подпись, дата) |

РЕФЕРАТ

Выпускная квалификационная работа \_\_ страниц, \_\_ рисунков, \_\_ таблиц, \_\_ источников.

ДОКУМЕНТООБОРОТ, LARAVEL, LARAVEL ELOQUENT ORM, ИМУЩЕСТВЕННЫЕ ОТНОШЕНИЯ, СИСТЕМА ДОКУМЕНТООБОРОТА.

Цель работы - разработка программного обеспечения для упрощения доступа к отчетности по вопросам водопользования и имущественных отношений.

В процессе работы проводилась исследования рынка и разработка программного обеспечения для отдела правовых и имущественных отношений.

Область применения: документооборот в сфере пользования внутренних водных путей.

В результате исследования было создано программное обеспечение облегчающие документооборот в ФБУ “Администрация Обь-Иртышводпуть”.

Данное программное обеспечение интегрируется с текущей системой документооборота и передачи документов.

СОДЕРЖАНИЕ

[РЕФЕРАТ 1](#_Toc161186422)

[ВВЕДЕНИЕ 3](#_Toc161186423)

[АНАЛИТИЧЕСКИЙ ОБЗОР 4](#_Toc161186424)

[1.1 Стейкхолдеры 4](#_Toc161186425)

[1.2 Сравнительный анализ продукта с существующими решениями 6](#_Toc161186426)

[1.3 Определение надсистемы продукта 9](#_Toc161186427)

[1.4 Выбор инструментария 9](#_Toc161186428)

[1.5 Выводы 11](#_Toc161186429)

[АРХИТЕКТУРА ПРОДУКТА 13](#_Toc161186430)

[2.1 Архитектура программного обоспечения 13](#_Toc161186431)

[2.2 Архитектура базы данных программного обеспечения 18](#_Toc161186432)

[2.3 Выводы 19](#_Toc161186433)

ВВЕДЕНИЕ

В современном информационном обществе, где бизнес-процессы становятся все более сложными и динамичными, эффективное управление документами становится одним из ключевых факторов успешной деятельности предприятий. Сфера документооборота, охватывающая создание, обработку, передачу и хранение документов, является неотъемлемой частью организационной инфраструктуры, влияющей на операционную эффективность и конкурентоспособность предприятия.

Несмотря на значительные технологические достижения последних лет, многие организации продолжают сталкиваться с проблемами устаревших систем документооборота, что приводит к снижению производительности, увеличению времени на обработку документов и повышению риска ошибок. В свете этих вызовов, разработка инновационного программного обеспечения в сфере документооборота становится актуальной задачей, направленной на оптимизацию бизнес-процессов и повышение операционной эффективности.

Целью данного дипломного исследования является разработка программного обеспечения, предназначенного для улучшения процессов документооборота, с целью обеспечения более эффективного управления информацией в организации. В рамках работы будут рассмотрены существующие проблемы в области документооборота, выявлены требования к программному обеспечению, разработаны соответствующие решения, а затем проведена оценка эффективности и влияния нового программного продукта на бизнес-процессы.

АНАЛИТИЧЕСКИЙ ОБЗОР

1.1 Стейкхолдеры

Из проведенного анализа можно заключить, что разработка программного обеспечения для сферы документооборота представляет собой сложный процесс, требующий вовлечения различных заинтересованных групп с разнообразными потребностями и ожиданиями. Согласно рекомендациям стандарта ГОСТ Р 57100-2016, заинтересованные стороны могут быть объединены в определенные категории.

Основными стейкхолдерами данного продукта являются отделы правовых и имущественных отношений других компаний, отдел эксплуатации гидротехнических сооружений, отдел судового хозяйства и промдеятельности, технический отдел, отдел правовых и имущественных отношений ФБУ “Администрация Обь-Иртышводпуть”.

В представленной таблице приведен функционал приложения, который является ключевыми компонентами для решения проблем и удовлетворения потребностей различных стейкхолдеров. Это способствует созданию полноценного и эффективного инструмента для пользователей программного обеспечения в области документооборота.

Таблица 1 Стейкхолдеры

| Стейкхолдеры | Проблема | Функционал |
| --- | --- | --- |
| **Отделы правовых и имущественных отношений** | Затруднения в быстром поиске и доступе к актуальным версиям документов, что может привести к ошибкам в согласовании и утверждении документов, а также потере времени на рутинные административные задачи. | Разработка системы управления электронным документооборотом с функцией автоматизации рабочих процессов, включая механизмы уведомлений о заданиях и контроля выполнения, а также возможность совместной работы над документами в режиме реального времени. |

Продолжение таблицы 1

|  |  |  |
| --- | --- | --- |
| Стейкхолдеры | Проблема | Функционал |
| **Отдел эксплуатации гидротехнических сооружений** | Ограниченный доступ к технической документации в удаленных районах, необходимость в быстром реагировании на экстренные ситуации без доступа к бумажным документам. | Создание онлайн-платформы для хранения и управления технической документацией, обеспечивающей доступ к данным из любой точки мира при наличии интернет-соединения, а также мобильного приложения для работы в условиях без доступа к сети. |
| **Отдел судового хозяйства и промышленности** | Затруднения в совместной работе над юридическими документами и их согласовании с различными участниками процесса, а также задержки в связи с трудностями в поиске и обработке информации | Разработка системы управления правовой документацией с функциями электронной подписи, отслеживания версий документов и автоматического уведомления участников процесса об изменениях, а также возможностью анализа и поиска документов по ключевым параметрам |
| **Технический отдел** | Затруднения в обмене информацией и координации работы между инженерами и техническими специалистами, а также сложности в управлении обширной технической документацией | Внедрение системы электронного хранилища для технической документации с возможностью быстрого доступа к инструкциям и руководствам, а также системы управления задачами и проектами для координации работы между сотрудниками и контроля выполнения задач |

В заключении данной главы по анализу стейкхолдеров в разработке программного обеспечения для сферы документооборота можно выделить, что успешная реализация проекта требует учета разнообразных потребностей и ожиданий заинтересованных сторон.

Создание полноценного и эффективного инструмента для управления документооборотом в описанной сфере предполагает учет множества аспектов, начиная от обеспечения быстрого доступа к актуальным версиям документов и автоматизации рабочих процессов до обеспечения совместной работы над документами в режиме реального времени. Такой подход способствует повышению эффективности бизнес-процессов и обеспечивает успешное взаимодействие всех участников процесса.

1.2 Сравнительный анализ продукта с существующими решениями

После тщательного изучения трудностей и потребностей заинтересованных сторон, включая пользователей и заказчиков, были выработаны требования, которые определяют необходимый функционал программного обеспечения.

Проводится анализ конкурентов в выбранной рыночной нише, сфокусированный на популярных информационных продуктах с аналогичными целями и характеристиками. Это исследование направлено на выявление преимуществ и недостатков конкурентов, а также определение оптимальной позиции разрабатываемого программного обеспечения на рынке.

Программное обеспечение "Контур Диадок" представляет собой комплексный онлайн-сервис для электронного документооборота, разработанный компанией "Контур". Основные характеристики и функции данного программного продукта:

1. **Электронный обмен документами:** Позволяет пользователям обмениваться различными видами документов онлайн. Это включает в себя создание, отправку, прием, хранение и обработку документов в электронном формате.
2. **Цифровая подпись:** Пользователи могут подписывать документы с помощью цифровой подписи, что обеспечивает им юридическую значимость и подтверждение подлинности.
3. **Хранение документов:** "Контур Диадок" обеспечивает безопасное и надежное хранение всех электронных документов. Пользователи могут легко получить доступ к своим документам в любое время и из любого места.
4. **Управление документами:** Позволяет управлять документами, включая их поиск, сортировку, фильтрацию и организацию в соответствии с различными критериями.
5. **Автоматизация процессов:** "Контур Диадок" предоставляет возможность автоматизировать ряд рутинных задач и процессов, связанных с обработкой документов, что улучшает эффективность работы.
6. **Интеграция с другими сервисами:** Сервис интегрируется с другими программными продуктами и информационными системами, что облегчает процессы обмена данными и совместной работы.
7. **Безопасность:** "Контур Диадок" обеспечивает высокий уровень безопасности для всех передаваемых и хранимых данных, включая применение современных методов шифрования и защиты.

Программное обеспечение 1С:Документооборот – это комплексное решение для электронного документооборота, разработанное компанией 1С. Основные характеристики и функции этого программного продукта:

1. **Электронный обмен документами:** 1С:Документооборот позволяет пользователям обмениваться различными типами документов, включая счета, договоры, акты и другие.
2. **Цифровая подпись:** Система поддерживает использование электронной цифровой подписи для обеспечения подлинности и юридической значимости документов.
3. **Управление документами:** Пользователи могут создавать, редактировать, хранить, отправлять и получать документы в единой системе управления, обеспечивая эффективное управление документооборотом.
4. **Хранение документов:** Программное обеспечение обеспечивает надежное и безопасное хранение всех электронных документов, что позволяет легко находить и получать доступ к ним при необходимости.
5. **Автоматизация процессов:** 1С:Документооборот предлагает ряд инструментов для автоматизации рутинных задач, связанных с обработкой документов, что повышает эффективность работы и сокращает временные затраты.
6. **Интеграция с другими системами:** Система может интегрироваться с другими программными продуктами и информационными системами, такими как ERP-системы, что обеспечивает более удобный и эффективный обмен данными.
7. **Отчетность и аналитика:** Пользователям доступны инструменты для создания отчетов и анализа данных, что позволяет оценить эффективность документооборота и принимать обоснованные управленческие решения.
8. **Безопасность:** Программное обеспечение обеспечивает высокий уровень безопасности для передаваемых и хранимых данных, включая механизмы шифрования и защиты информации от несанкционированного доступа.

1С:Документооборот широко используется в различных организациях для автоматизации процессов управления документами и повышения эффективности работы с ними.

После проведения анализа рынка в данной области были выявлены следующие ключевые функциональные решения, представленные в продуктах:

1. Электронный обмен документами
2. Цифровая подпись и шифрование
3. Управление рабочими процессами
4. Интеграция с другими системами
5. Отчетность и аналитика

В итоге, проведенный анализ показал, что ключевые функциональные решения, представленные в "Контур Диадок" и "1С:Документооборот", являются необходимыми и достаточными для удовлетворения потребностей пользователей. Эти решения включают в себя электронный обмен документами, использование цифровой подписи, автоматизацию рабочих процессов, интеграцию с другими системами, а также инструменты для отчетности и аналитики.

1.3 Определение надсистемы продукта

В результате проведенного анализа потребностей заинтересованных сторон, веб-браузеры могут служить надсистемой для данного продукта. Они обеспечивают доступ к функциональным возможностям программного обеспечения для управления документооборотом, предоставляя пользователю гибкость работы из любого места с доступом к Интернету. Браузеры позволяют легко обмениваться документами, осуществлять электронную подпись, управлять рабочими процессами, анализировать данные и многое другое, что делает их идеальной платформой для реализации функционала программного обеспечения в сфере документооборота.

Дополнительно к учету браузеров в качестве надсистемы продукта, необходимо обеспечить однородность визуального представления программного обеспечения на всех браузерах. Это означает, что пользовательский интерфейс должен быть адаптирован и совместим с различными браузерами, чтобы обеспечить единый и последовательный пользовательский опыт независимо от того, какой браузер используется. Такой подход поможет избежать различий в отображении интерфейса и обеспечит удобство использования продукта для всех пользователей, что важно для обеспечения качественного функционирования программного обеспечения в сфере документооборота.

1.4 Выбор инструментария

Для успешной реализации выявленных функциональных требований, выделенных в предыдущих разделах исследования, необходимо тщательно подобрать набор инструментов. Этот набор представляет собой комплекс технических ресурсов и средств, включающих программные платформы, фреймворки, языки программирования, инструменты для управления версиями и средства разработки.

Для обеспечения эффективной работы с данными пользователей и документами, было решено использовать базу данных MySQL. MySQL - это реляционная система управления базами данных, широко применяемая веб-разработчиками благодаря своей надежности, производительности и гибкости. Она предоставляет широкий набор функций для хранения, обработки и управления данными, а также обладает отличной поддержкой для множества языков программирования и фреймворков.

Использование MySQL позволяет эффективно организовывать структуру данных, обеспечивать быстрый доступ к информации, обеспечивать целостность и безопасность данных. Кроме того, она обладает возможностью масштабирования, что позволяет удовлетворить потребности в росте и развитии продукта со временем.

Выбор MySQL в качестве базы данных обусловлен его широким распространением, отличной документацией и поддержкой сообщества разработчиков, что обеспечивает стабильность и надежность в работе с данными, необходимыми для функционирования программного обеспечения в сфере документооборта.

Для разработки программного обеспечения выбран язык программирования PHP вместе с фреймворком Laravel, который предоставляет удобный и эффективный инструментарий для создания веб-приложений. Одним из преимуществ Laravel является встроенный ORM (Object-Relational Mapping), который облегчает работу с базами данных, позволяя взаимодействовать с ними на уровне объектов, а не SQL-запросов. Это упрощает процесс работы с данными и делает код более читаемым и поддерживаемым.

Использование ORM в Laravel позволяет разработчикам сосредоточиться на бизнес-логике приложения, а не на деталях работы с базой данных. Они могут легко определять и работать с моделями данных, устанавливать отношения между ними и выполнять запросы к базе данных с помощью высокоуровневого API.

Таким образом, выбор PHP в сочетании с Laravel и возможностью использовать ORM обусловлен желанием создать эффективное и масштабируемое программное обеспечение в сфере документооборта, обеспечивая при этом удобство и производительность веб-разработки.

Для реализации интерфейса пользователя веб-приложения был выбран язык JavaScript, который широко используется для создания интерактивных элементов на веб-страницах. JavaScript был интегрирован с фреймворком Laravel, используя Blade шаблоны. Blade - движок шаблонов, встроенный в Laravel, который облегчает создание и поддержку пользовательского интерфейса.

Использование JavaScript в сочетании с Blade шаблонами позволяет создавать динамические и интерактивные элементы на веб-страницах, такие как формы, таблицы, диаграммы и многое другое. Это позволяет улучшить пользовательский опыт и добавить функциональность, такую как валидация форм, динамическая загрузка данных и обновление страниц без перезагрузки.

Таким образом, комбинация JavaScript и Blade шаблонов фреймворка Laravel обеспечивает эффективное взаимодействие между серверной и клиентской частями приложения, а также обеспечивает удобство в разработке и поддержке пользовательского интерфейса.

Средой разработки является Visual Studio Code. Visual Studio Code (VS Code) - это среда разработки, которая предоставляет разработчикам широкий набор инструментов для написания, отладки и управления кодом. Она отличается легкостью использования, мощными функциями и расширяемостью. VS Code поддерживает множество языков программирования, интегрируется с различными технологиями и фреймворками, что делает его популярным выбором среди разработчиков.

Использование S3 сервера для хранения файлов обеспечивает значительные преимущества. S3 предлагает масштабируемость, позволяя хранить неограниченное количество данных, что важно для систем документооборота. Безопасность данных обеспечивается через шифрование и управление доступом с помощью политик IAM, а также логирование и мониторинг. Экономичность обусловлена системой оплаты только за использованные ресурсы, что делает S3 выгодным для хранения больших объемов данных.

В ходе анализа доступных вариантов инструментария для разработки программного обеспечения становится ясным, что определение конкретных технологий и средств представляет собой важный этап, имеющий существенное воздействие на успешное завершение проекта.

1.5 Выводы

1. **Анализ стейкхолдеров**: Проект по разработке программного обеспечения для сферы документооборота представляет собой сложный процесс, требующий учета потребностей различных заинтересованных сторон. Успешная реализация проекта зависит от удовлетворения этих потребностей и ожиданий.
2. **Сравнительный анализ продукта с конкурентами**: Программное обеспечение "Контур Диадок" и 1С:Документооборот предоставляют схожий функционал, включая электронный обмен документами, цифровую подпись, управление документами, автоматизацию процессов, интеграцию с другими системами, отчетность и аналитику. Оба продукта широко используются для управления документооборотом в организациях.
3. **Определение надсистемы продукта и выбор инструментария**: Для управления документооборотом веб-браузеры используются в качестве надсистемы. Ключевые инструменты разработки включают в себя базу данных MySQL для хранения данных, язык программирования PHP с фреймворком Laravel для бизнес-логики и веб-интерфейса, JavaScript с Blade шаблонами для интерактивности пользовательского интерфейса, и Visual Studio Code в качестве среды разработки.

Учет потребностей стейкхолдеров, сравнение с конкурентами, выбор подходящих инструментов разработки - все эти шаги необходимы для успешного завершения проекта по разработке программного обеспечения для документооборота. Тщательный анализ и выбор правильных решений в каждом этапе разработки являются ключевыми для достижения целей проекта.

2 АРХИТЕКТУРА ПРОДУКТА

2.1 Архитектура программного обеспечения

Архитектура продукта играет важную роль в обеспечении эффективной работы и достижении целей. Она определяет основные принципы взаимодействия между компонентами приложения, формируя его структуру и организацию.

Гибкость и масштабируемость приложения обеспечиваются благодаря его архитектуре. Это позволяет легко вносить изменения, добавлять новые функции и улучшать продукт без нарушения его базовой структуры. Такой подход оказывает существенное влияние на долгосрочное развитие и поддержку продукта.

Кроме того, архитектура позволяет эффективно использовать технические ресурсы приложения. Оптимизированная структура снижает нагрузку на систему, способствуя ее стабильной работе и снижению вероятности возникновения ошибок и сбоев.

Выбор архитектуры проекта в пользу MVC (рисунок 1) был обусловлен соответствием этой модели большинству выдвинутых требований, изложенных в предшествующем разделе. Ее эффективность в данном контексте проявляется через ясное разделение ролей и обязанностей между компонентами. Это гарантирует устойчивость и надежность архитектуры, обеспечивая удобство в поддержке и дальнейшем развитии системы.



Рисунок 1 – Схема взаимодействия между компонентами MVC

Клиентское приложение может быть представлено веб-интерфейсом, мобильным приложением или любым другим типом пользовательского интерфейса. Оно взаимодействует с серверной частью через RESTful API, предоставляемое микросервисами на Laravel. Пользовательские запросы, такие как загрузка документов, поиск и фильтрация, отправляются на сервер и обрабатываются соответствующими микросервисами.

Архитектура микросервисов (рисунок 2) предполагает разделение приложения на независимые сервисы, каждый из которых отвечает за определенный функционал.

Модуль управления пользователями: Отвечает за регистрацию, аутентификацию, управление правами доступа и профилями пользователей.

Модуль загрузки и хранения документов: Обрабатывает операции по загрузке, хранению и извлечению документов.

Модуль поиска и фильтрации документов: Предоставляет возможности для поиска документов по различным критериям и фильтрации результатов. Й Модуль отчетности и аналитики: Обрабатывает запросы на создание отчетов и проведение аналитики по документам.

Каждый микросервис реализован на фреймворке Laravel и взаимодействует с остальными микросервисами через API, что обеспечивает независимость и возможность автономного масштабирования каждого сервиса.



Рисунок 2 – Архитектура проекта

S3 сервер используется для хранения документов и данных. Его основные функции включают:

* Надежное хранение: Данные хранятся в нескольких географически разнесенных дата-центрах, что обеспечивает их доступность и безопасность.
* Масштабируемость: Возможность хранения любого объема данных без необходимости управления физическими ресурсами.
* Безопасность: Поддержка шифрования данных и управления доступом, что обеспечивает защиту информации.

Микросервисы взаимодействуют с S3 сервером с помощью AWS SDK, что позволяет загружать, извлекать и управлять файлами.

Redis используется для кэширования данных и управления сессиями, что помогает ускорить обработку запросов и снизить нагрузку на базу данных. Его основные функции включают:

* Кэширование: Хранение часто запрашиваемых данных для быстрого доступа, что снижает время отклика системы.
* Управление сессиями: Обеспечение надежного хранения данных сессий пользователей.
* Очереди задач: Управление очередями задач для асинхронной обработки данных, что повышает производительность системы.

2.2 Взаимодействие компонентов

Запрос клиента: Клиентское приложение отправляет запрос на микросервисный API, используя HTTPS.

Обработка запроса: Запрос попадает в соответствующий микросервис на Laravel. Контроллеры Laravel обрабатывают запрос, вызывают соответствующие сервисы и репозитории.

Если данные необходимы из базы данных, микросервис обращается к базе данных напрямую. Если данные часто запрашиваются и находятся в кэше, микросервис проверяет Redis и, если данные найдены, возвращает их. Если данных в кэше нет, они извлекаются из базы данных и сохраняются в Redis для будущих запросов.

Если запрос требует загрузки или извлечения файлов, микросервис взаимодействует с S3, используя AWS SDK. Файлы загружаются в S3 или извлекаются оттуда, предоставляя URL для доступа к ним.

После обработки запроса и выполнения всех необходимых операций, микросервис отправляет ответ клиентскому приложению, содержащий данные или подтверждение выполнения операции.

2.3 Архитектура базы данных программного обеспечения

Архитектура базы данных в программном обеспечении для документооборта имеет ключевое значение для хранения, управления и обработки документов различных типов. Она включает в себя структуру данных (рисунок 3), определение связей между документами и другими сущностями, индексирование и поиск, обеспечение безопасности и доступа, а также аудит и отчетность. Кроме того, архитектура базы данных должна обеспечивать масштабируемость, производительность и интеграцию с другими системами. Основные таблицы базы данных для системы документооборота включают:

Таблица 2 Departments

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | bigint(20) unsigned | Уникальный идентификатор отдела (Primary Key) |
| name | varchar(255) | Название отдела |
| description | varchar(255) | Описание отдела |
| deleted\_at | timestamp | Метка времени удаления записи (Soft Delete) |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |

Таблица 3 Users

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | bigint(20) unsigned | Уникальный идентификатор пользователя (Primary Key) |
| name | varchar(255) | Имя пользователя |
| email | varchar(255) | Email пользователя |
| email\_verified\_at | timestamp | Дата и время подтверждения email |
| password | varchar(255) | Пароль пользователя |
| remember\_token | varchar(100) | Токен для запоминания пользователя |
| department\_id | bigint(20) unsigned | Идентификатор отдела (Foreign Key к departments) |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |
| deleted\_at | timestamp | Метка времени удаления записи (Soft Delete) |

Таблица 4 Documents

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | bigint(20) unsigned | Уникальный идентификатор документа (Primary Key) |
| title | varchar(255) | Название документа |
| content | varchar(255) | Содержание документа |
| version | varchar(255) | Версия документа |
| user\_id | bigint(20) unsigned | Идентификатор пользователя (Foreign Key к users) |
| state | varchar(255) | Состояние документа |
| document\_link | varchar(255) | Ссылка на документ |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |
| deleted\_at | timestamp | Метка времени удаления записи (Soft Delete) |

Таблица 5 Document\_versions

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | bigint(20) unsigned | Уникальный идентификатор версии документа (Primary Key) |
| document\_id | bigint(20) unsigned | Идентификатор документа (Foreign Key к documents) |
| content | varchar(255) | Содержание версии документа |
| version | varchar(255) | Версия документа |
| user\_id | bigint(20) unsigned | Идентификатор пользователя (Foreign Key к users) |
| document\_link | varchar(255) | Ссылка на версию документа |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |
| deleted\_at | timestamp | Метка времени удаления записи (Soft Delete) |

Таблица 6 Roles

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| id | bigint(20) unsigned | Уникальный идентификатор роли (Primary Key) |
| name | varchar(255) | Название роли |
| guard\_name | varchar(255) | Название охранника (guard name) |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |

Таблица 7 Permissions

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| id | bigint(20) unsigned | Уникальный идентификатор разрешения (Primary Key) |
| name | varchar(255) | Название разрешения |
| guard\_name | varchar(255) | Название охранника (guard name) |
| created\_at | timestamp | Дата и время создания записи |
| updated\_at | timestamp | Дата и время последнего обновления записи |

Таблица 8 Model\_has\_roles

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| role\_id | bigint(20) unsigned | Идентификатор роли (Foreign Key к roles) |
| model\_type | varchar(255) | Тип модели |
| model\_id | bigint(20) unsigned | Идентификатор модели |

Таблица 9 Model\_has\_permissions

|  |  |  |
| --- | --- | --- |
| **Поле** | **Тип данных** | **Описание** |
| permission\_id | bigint(20) unsigned | Идентификатор разрешения (Foreign Key к permissions) |
| model\_type | varchar(255) | Тип модели |
| model\_id | bigint(20) unsigned | Идентификатор модели |

Таблица 10 Role\_has\_permissions

| **Поле** | **Тип данных** | **Описание** |
| --- | --- | --- |
| permission\_id | bigint(20) unsigned | Идентификатор разрешения (Foreign Key к permissions) |
| role\_id | bigint(20) unsigned | Идентификатор роли (Foreign Key к roles) |

Описание связей:

Departments иusers: Один отдел может иметь много пользователей, установлена связь "один ко многим" через поле department\_id в таблице users.

Usersи documents: Один пользователь может создавать много документов, связь "один ко многим" через поле user\_id в таблице documents.

Documents **и** document\_versions: Один документ может иметь много версий, связь "один ко многим" через поле document\_id в таблице document\_versions.

Roles, permissions и users: Используются таблицы model\_has\_roles, model\_has\_permissions и role\_has\_permissions для реализации отношений "многие ко многим" между пользователями, ролями и разрешениями.

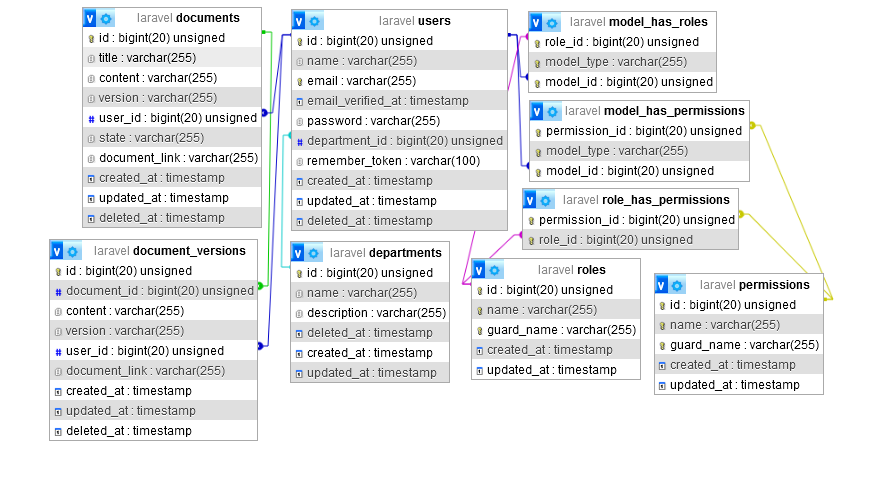


Рисунок 3 – Схема базы данных

Эти связи обеспечивают целостность данных и позволяют легко управлять документооборотом в системе.

2.4 Безопасность и производительность

Обеспечение безопасности данных является ключевым аспектом архитектуры. Включает следующие меры:

Шифрование данных: Использование SSL/TLS для защиты данных при передаче, шифрование данных в S3 для защиты при хранении.

* Аутентификация и авторизация: Использование JWT (JSON Web Tokens) для аутентификации пользователей и управление доступом с помощью ролей и политик.
* Логирование и мониторинг: Ведение журналов доступа и операций для аудита и мониторинга безопасности системы.

Оптимизация производительности достигается за счет использования кэширования в Redis, асинхронной обработки задач и масштабируемой инфраструктуры S3. Эти меры позволяют снизить время отклика системы и обеспечить ее устойчивую работу под высокой нагрузкой.

Для обеспечения качества и стабильности системы используются различные методы тестирования:

* Функциональное тестирование: Проверка корректности работы всехфункциональных модулей.
* Нагрузочное тестирование: Оценка производительности системы под высокой нагрузкой.
* Обработка и исправление ошибок: Использование инструментов для мониторинга ошибок и их оперативного устранения.

2.3 Выводы

Архитектура программного обеспечения играет ключевую роль в обеспечении эффективности работы и достижении целей. Она задает основные принципы взаимодействия между компонентами, формируя структуру и организацию системы.

MVC (Model-View-Controller) была выбрана в качестве архитектурной модели благодаря её соответствию требованиям проекта. Она обеспечивает ясное разделение ролей между компонентами, что упрощает поддержку и развитие системы.

Клиентское приложение взаимодействует с серверной частью через RESTful API, предоставляемое микросервисами на фреймворке Laravel. Архитектура микросервисов разделяет приложение на независимые сервисы, каждый из которых отвечает за определённый функционал, такие как управление пользователями, загрузка и хранение документов, поиск и фильтрация, отчетность и аналитика.

Использование S3 сервера для хранения данных обеспечивает надёжное, масштабируемое и безопасное хранение данных. Микросервисы взаимодействуют с S3 через AWS SDK для управления файлами. Redis используется для кэширования данных и управления сессиями, что ускоряет обработку запросов и снижает нагрузку на базу данных.

Таким образом, выбранная архитектура обеспечивает надёжность, гибкость и масштабируемость системы, оптимизируя использование технических ресурсов и поддерживая стабильную работу продукта.

3 Реализация проекта

3.1 Описание процесса разработки

Процесс разработки программного обеспечения является основополагающим для создания качественного и надежного продукта. В данной главе рассматривается процесс разработки системы документооборота, основанной на микросервисной архитектуре с использованием фреймворка Laravel, S3 сервера и Redis. Основные этапы включают планирование, проектирование, разработку, тестирование и развёртывание системы.

На этапе планирования определяются цели и требования к системе, разрабатывается общий план проекта, устанавливаются сроки и распределяются ресурсы. Этот этап включает анализ требований пользователей, определение функциональных и нефункциональных характеристик системы, разработку плана проекта с ключевыми этапами и созданием графика работ, а также оценку необходимых ресурсов, включая людские ресурсы, оборудование и программные средства.

Проектирование системы включает создание архитектуры, разработку схем базы данных и интерфейсов. Определяется общая архитектура системы с выбором микросервисного подхода, разрабатывается схема базы данных с определением таблиц и связей между ними, а также создаются макеты пользовательского интерфейса и определяются основные компоненты и их взаимодействие.

На этапе разработки происходит непосредственное создание системы. Создаются отдельные микросервисы с использованием фреймворка Laravel, каждый из которых отвечает за конкретный функционал, такие как управление пользователями или хранение документов. Внедряется взаимодействие с S3 сервером для хранения документов и Redis для кэширования данных и управления сессиями.

Тестирование системы обеспечивает выявление и устранение ошибок. Проверяется корректность работы всех функциональных модулей, оценивается производительность под высокой нагрузкой и проверяется система на уязвимости для обеспечения защиты данных.

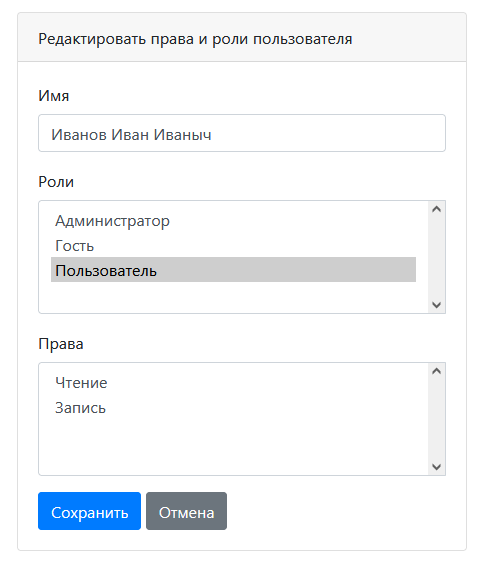
После завершения разработки и тестирования система готова к развёртыванию и эксплуатации. На этом этапе происходит установка и настройка системы на сервере, миграция базы данных и настройка окружения. Впоследствии осуществляется непрерывный мониторинг работы системы, оперативное устранение возникших проблем и регулярные обновления и улучшения.

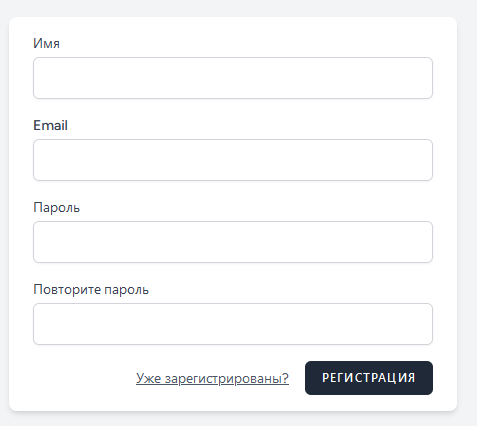
Таким образом, процесс разработки программного обеспечения включает несколько ключевых этапов, от планирования до развертывания и поддержки системы. Каждый этап имеет свои особенности и задачи, которые необходимо выполнить для создания качественного продукта. Применение микросервисной архитектуры, использование фреймворка Laravel, S3 сервера и Redis обеспечивает гибкость, масштабируемость и надежность системы документооборота.

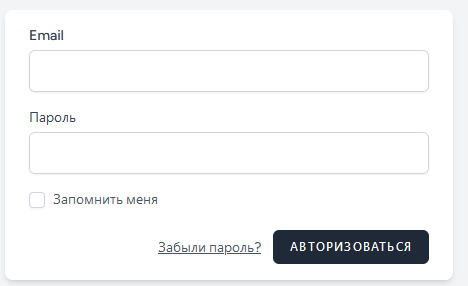
3.2. Разработка основных модулей системы

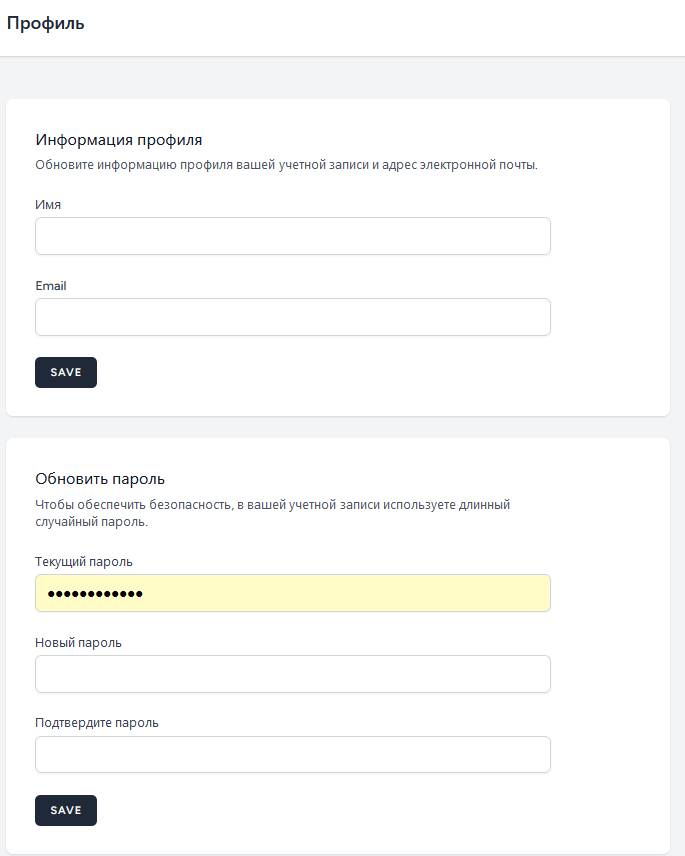
Разработка основных модулей системы является центральной частью процесса создания программного обеспечения Основными модулями системы являются модуль управления пользователями, модуль загрузки и хранения документов, модуль поиска и фильтрации документов, а также модуль отчетности и аналитики.

**Модуль управления пользователями** отвечает за регистрацию, аутентификацию, управление правами доступа и профилями пользователей. На этапе разработки этого модуля реализуется функциональность регистрации новых пользователей с валидацией данных, аутентификация с использованием JWT (JSON Web Tokens) для обеспечения безопасности, а также управление ролями и разрешениями пользователей с использованием встроенных возможностей Laravel.

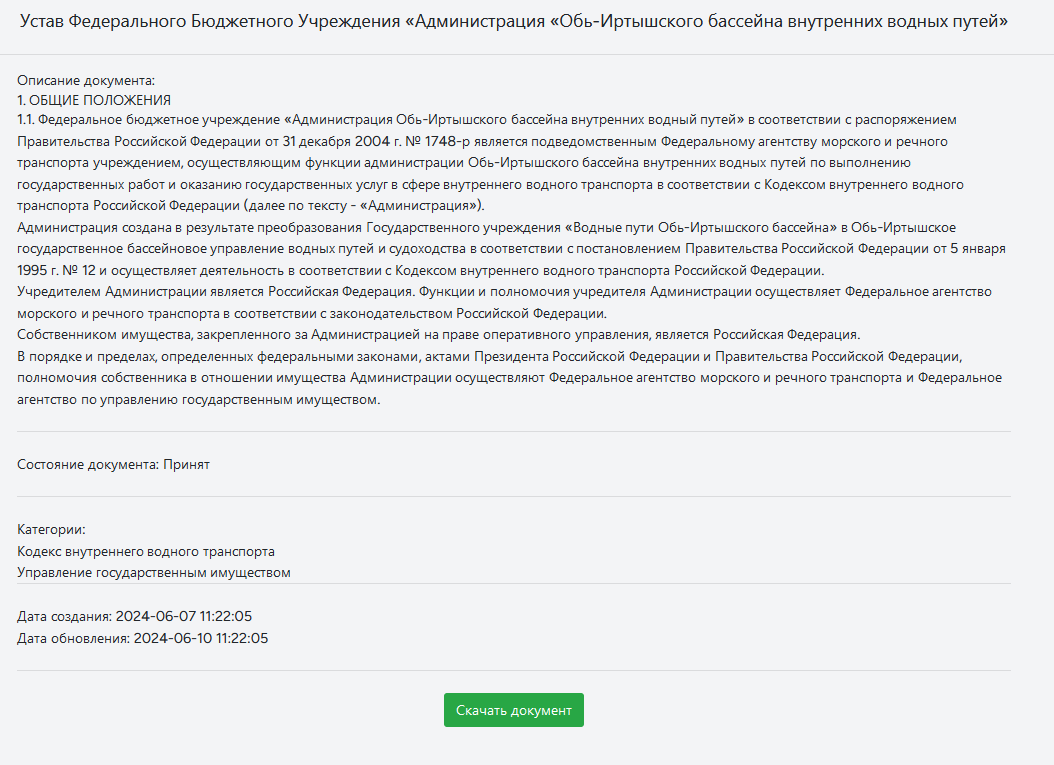




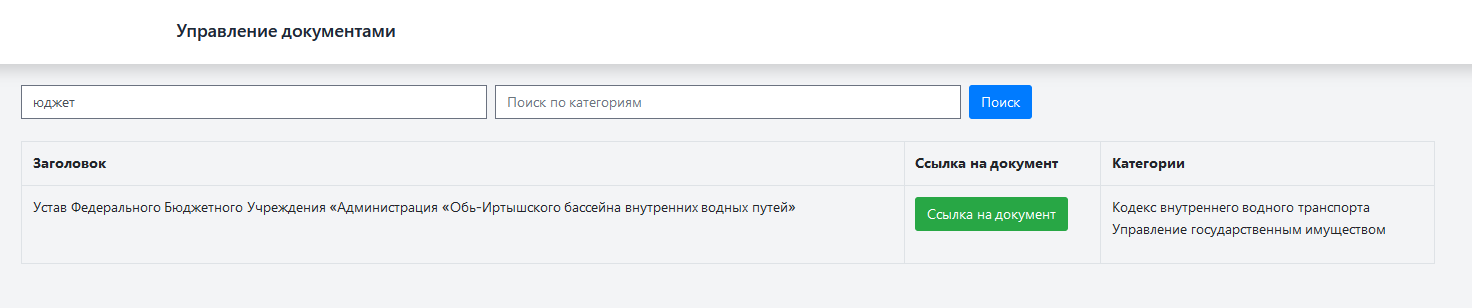




**Модуль загрузки и хранения документов** обрабатывает операции по загрузке, хранению и извлечению документов. Для реализации этого модуля используется AWS SDK, который обеспечивает взаимодействие с S3 сервером. Пользователь может загружать документы через веб-интерфейс или API, после чего документы сохраняются на S3 сервере. Также реализуется функциональность для извлечения и удаления документов, управления версиями и ссылками на документы.



**Модуль поиска и фильтрации документов** предоставляет возможности для поиска документов по различным критериям и фильтрации результатов. Этот модуль интегрируется с базой данных и Redis для ускорения поиска и обработки запросов. Пользователь может искать документы по заголовку, содержанию, автору, дате создания и другим метаданным. Результаты поиска кешируются в Redis для уменьшения нагрузки на базу данных и ускорения последующих запросов.



**Модуль отчетности и аналитики** обрабатывает запросы на создание отчетов и проведение аналитики по документам. Для разработки этого модуля используются возможности Laravel для генерации отчетов в различных форматах (PDF, Excel и т.д.). Реализуется функциональность для сбора статистики по использованию системы, анализа данных и создания различных аналитических отчетов, которые могут быть полезны для управления и оптимизации бизнес-процессов.

Каждый из этих модулей реализуется как отдельный микросервис на фреймворке Laravel, что обеспечивает независимость и возможность автономного масштабирования каждого модуля. Все микросервисы взаимодействуют между собой через RESTful API, что позволяет легко интегрировать новые модули и расширять функциональность системы.

Таким образом, разработка основных модулей системы включает в себя создание ключевых компонентов, обеспечивающих функциональность системы документооборота. Применение микросервисной архитектуры, использование фреймворка Laravel, S3 сервера и Redis обеспечивает гибкость, масштабируемость и надежность каждого модуля, что в совокупности способствует созданию качественного и устойчивого продукта.

3.3. Интеграция и развертывание системы

Процесс подготовки к развертыванию включает несколько этапов:

* Настройка сервера: Установка и настройка необходимого программного обеспечения, такого как PHP, MySQL и Nginx. Настройка безопасности сервера, включая брандмауэры и антивирусное ПО.
* Настройка окружения: Конфигурация переменных окружения, таких как параметры подключения к базе данных, настройки кеширования и другие конфигурационные параметры, специфичные для производственной среды. Использование файла .env для управления конфигурацией.
* Установка зависимостей: Установка всех необходимых зависимостей с помощью Composer. Проверка совместимости версий и выполнение миграций базы данных.

Процесс настройки окружения включает:

* Настройка базы данных: Создание и настройка базы данных, выполнение миграций для создания необходимых таблиц и структуры данных. Оптимизация производительности базы данных с использованием индексов и других методов.
* Настройка веб-сервера: Конфигурация Nginx для обслуживания приложения Laravel. Настройка правил маршрутизации, сжатия данных и кэширования.
* Настройка системного окружения: Установка и настройка инструментов для работы с очередями (например, Redis), кешированием и другими компонентами, необходимыми для производственной среды.

Система может интегрирована с внешними системами для расширения функциональности. Системы управления контентом (CMS): Интеграция с существующими CMS для публикации документов и управления контентом на веб-сайтах.

3.4. Тестирование и отладка

Процесс обработки и исправления ошибок включает следующие этапы:

* **Логирование ошибок**: Использование встроенных средств Laravel для логирования всех ошибок и исключений. Логи записываются в файлы или базы данных для последующего анализа. Настройка различных уровней логирования (например, debug, info, warning, error) позволяет упростить поиск и устранение ошибок.
* **Мониторинг системы**: Использование инструментов мониторинга, таких как Laravel Telescope и внешние сервисы мониторинга (например, New Relic, Sentry), для отслеживания состояния системы в реальном времени. Эти инструменты помогают обнаружить ошибки, сбои и другие проблемы на ранних стадиях.
* **Анализ ошибок**: Анализ логов и данных мониторинга для выявления причин ошибок и проблем. Применение методов трассировки стека и анализа контекста для детального изучения каждой проблемы.
* **Проведение тестов**: После исправления ошибок проводится повторное тестирование (регрессионное тестирование) для проверки корректности исправлений и удостоверения в отсутствии новых ошибок. Это включает повторное выполнение модульных, интеграционных и функциональных тестов.

Эти методы и подходы позволяют обеспечить высокое качество и надежность системы документооборота, своевременно обнаруживать и устранять ошибки, а также поддерживать высокий уровень производительности и безопасности.