**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ**

**ФГБОУ ВО  
«БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

**ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ**

**Кафедра «Информатика и программное обеспечение»**

**«У Т В Е Р Ж Д А Ю»**

Зав. кафедрой «И и ПО», к.т.н., доцент

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Подвесовский А.Г.

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2021 г.

**ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО КОМПЛЕКСА УПРАВЛЕНИЯ НАДЕЖНОЙ ОТПРАВКОЙ ЭЛЕКТРОННЫХ ПИСЕМ С РЕАЛИЗАЦИЕЙ ОСНОВНОГО МЕХАНИЗМА РАССЫЛКИ**

**И ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА**

**ДИПЛОМНАЯ РАБОТА**

Документы текстовые

Всего \_\_\_\_ листов в папке

**Руководитель**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. Трубаков А.О.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

**Консультанты:**

по экономической части

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. Титарев Д.В.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

по организационной части

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ст. преп. Зяблова Е.С.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

**Нормоконтролер**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ к.т.н., доц. Трубаков А.О.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

**Студент**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Сухарев Е.А.

«\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_2021 г.

**БРЯНСК 2021**

АННОТАЦИЯ

Данная дипломная работа посвящена вопросам проектирования и разработки программного комплекса, служащего для управления и контроля надежной отправки электронных писем.

На данный момент в спроектированном комплексе выполнена реализация следующих модулей:

1. интерфейсы пользовательской части для просмотра доступной информации;
2. компоненты пользовательской части, отвечающие за обработку действий пользователя и отправку запросов на сервер;
3. web-API [2] для обработки получаемых сервером запросов;
4. фоновая служба для асинхронной отправки сообщений;
5. база данных для хранения основных сущностей.

В разделе «Анализ требований» представлен обзор процесса отправки электронного сообщения, программ-аналогов, диаграммы вариантов использования, IDEF0-диаграммы. Описана диаграмма BPMN, рассмотрен бизнес-процесс от создания сообщения пользователем до отправки этого сообщения получателям.

Экономический анализ представляет собой определение организационной структуры, необходимой для создания программного комплекса, составление календарного плана и расчёт себестоимости проекта.

В рамках проектирования и разработки программного комплекса были выбраны основные инструменты разработки, разработана архитектура программного комплекса, спроектированы и разработаны основные интерфейсы. Спроектирована база данных, а особенности основных ее сущностей отражены на ER-диаграмме. Также разработаны диаграмма потоков данных и диаграмма классов для службы отправки сообщений.

Тестирование программного комплекса содержит перечень видов испытаний, их краткое описание, а также результаты проведенных испытаний.

**БРЯНСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ**

**Направление 09.03.04 – «Программная инженерия»**

**профиль «Разработка программно-информационных систем»**

**ЗАДАНИЕ**

**на дипломную работу**

студенту группы О-17-ПРИ-рпс-б

***Сухареву Евгению Александровичу***

1. Тема работы  *Проектирование программного комплекса управления надежной отправкой электронных писем с реализацией основного механизма рассылки и пользовательского интерфейса*

Утверждена приказом по БГТУ №

1. Срок предоставления законченной работы  *май 2021 г.*
2. Исходные данные

Цель работы  *проектирование программного комплекса управления рассылкой электронных сообщений с возможностью последующей интеграции в системы, требующие высокой надежности отправки*

Инструментальные средства  *Microsoft Visual Studio 2019 Community, MS SQL Server 2019 (Express Edition), Java 8, Android Studio 4.2., C#,*

Основные требования  *а) асинхронная рассылка сообщений выбранным адресам с помощью выбранных сервисов доставки; б) обеспечение повышенной надежности доставки; в) возможность получения и просмотра подробной информации о статусе доставки сообщения*

Дата выдачи задания  *25 марта*  2021г.

Заведующий кафедрой  Подвесовский А.Г.

Руководитель работы  Трубаков А.О.

Задание принял к исполнению  *25 марта*  2021г.

Студент  Сухарев Е.А.

СОДЕРЖАНИЕ

[АННОТАЦИЯ 2](#_Toc74611470)

[ВВЕДЕНИЕ 6](#_Toc74611471)

[1. АНАЛИЗ ТРЕБОВАНИЙ 8](#_Toc74611472)

[1.1. Обзор процессов отправки и доставки сообщения 8](#_Toc74611473)

[1.1.1. Общие сведения 8](#_Toc74611474)

[1.1.2. Основные понятия 8](#_Toc74611475)

[1.1.3. Процесс отправки сообщения 10](#_Toc74611476)

[1.2. Обзор программ-аналогов 12](#_Toc74611477)

[1.2.1. Amazon SES 12](#_Toc74611478)

[1.2.2. Sendpulse 13](#_Toc74611479)

[1.2.3. Tin-cat email queue 14](#_Toc74611480)

[1.2.4. Сравнение 15](#_Toc74611481)

[1.3. Диаграмма вариантов использования 15](#_Toc74611482)

[1.3.1. Оператор 15](#_Toc74611483)

[1.3.2. Администратор 16](#_Toc74611484)

[1.4. IDEF0-Диаграмма 18](#_Toc74611485)

[1.5. Модель бизнес-процесса отправки сообщения 20](#_Toc74611486)

[1.6. Выводы 22](#_Toc74611487)

[2. ТЕХНИЧЕСКОЕ ЗАДАНИЕ 23](#_Toc74611488)

[2.1. Основание для разработки 23](#_Toc74611489)

[2.2. Назначение и область применения 23](#_Toc74611490)

[2.3. Требования к программному комплексу 23](#_Toc74611491)

[2.3.1. Требования к функциональным характеристикам 23](#_Toc74611492)

[2.3.2. Требования к надежности 26](#_Toc74611493)

[2.4. Условия эксплуатации 27](#_Toc74611494)

[2.4.1. Климатические условия эксплуатации 27](#_Toc74611495)

[2.4.2. Требования к квалификации и численности персонала 27](#_Toc74611496)

[2.4.3. Требования к составу и параметрам технических средств 27](#_Toc74611497)

[2.4.4. Требования к программной совместимости 28](#_Toc74611498)

[2.5. Программная документация 28](#_Toc74611499)

[2.5.1. Предварительный состав программной документации 28](#_Toc74611500)

[2.6. Стадии и этапы разработки 28](#_Toc74611501)

[2.6.1. Стадии разработки 28](#_Toc74611502)

[2.6.2. Содержание работ по этапам 29](#_Toc74611503)

[2.7. Порядок контроля и приемки 29](#_Toc74611504)

[2.7.1. Виды испытаний 29](#_Toc74611505)

[2.7.2. Общие требования к приемке работы 30](#_Toc74611506)

[3. ЭКОНОМИЧЕСКИЙ АНАЛИЗ 32](#_Toc74611507)

[4. ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО КОМПЛЕКСА 34](#_Toc74611508)

[4.1. Клиентская часть 34](#_Toc74611509)

[4.2. Серверная часть 34](#_Toc74611510)

[4.3. Модель данных 36](#_Toc74611511)

[4.4. Низкоуровневое проектирование 38](#_Toc74611512)

[4.4.1. ER-диаграмма 38](#_Toc74611513)

[4.4.2. Диаграмма потоков данных 42](#_Toc74611514)

[4.4.3. Диаграмма классов для службы отправки сообщений 44](#_Toc74611515)

[4.5. Проектирование интерфейса 45](#_Toc74611516)

[4.6. Проектирование системы балансировки нагрузки 49](#_Toc74611517)

[5. ТЕСТИРОВАНИЕ ПРОГРАММНОГО КОМПЛЕКСА 51](#_Toc74611518)

[5.1. Тестирование Web API методом черного ящика 51](#_Toc74611519)

[5.2. Модульное тестирование 54](#_Toc74611520)

[5.3. Кроссбраузерное тестирование интерфейсов 54](#_Toc74611521)

[6. ОРГАНИЗАЦИОННАЯ ЧАСТЬ 58](#_Toc74611522)

[ЗАКЛЮЧЕНИЕ 60](#_Toc74611523)

[СПИСОК ЛИТЕРАТУРЫ 61](#_Toc74611524)

[ПРИЛОЖЕНИЯ 63](#_Toc74611525)

[Приложение А 63](#_Toc74611526)

[Приложение Б 72](#_Toc74611527)

[Приложение В 82](#_Toc74611528)

[Приложение Г 93](#_Toc74611529)

# ВВЕДЕНИЕ

Информационные технологии получают всё большее развитие в современном мире. Для быстрого обмена информацией существует множество различных способов, и электронная почта до сих пор занимает лидирующие позиции в сфере коммуникации.

Электронное сообщение представляет из себя информацию, переданную или полученную пользователем сети Интернет. Электронным документом признается электронное сообщение, подписанное цифровой подписью.

Многие крупные компании используют электронную почту для обмена документами. В таких ситуациях крайне важно, чтобы сообщение было успешно доставлено и прочитано получателем.

Для отправки и принятия электронных сообщений существует ряд протоколов электронной почты: POP3, IMAP и SMTP.

Как правило, крупные развлекательные сервисы сами не реализуют механизм отправки сообщения через протокол SMTP, а пользуются услугами готовых сервисов рассылки сообщений. К сожалению, большинство почтовых клиентов не предоставляют информации о том, доставлено ли и прочитано ли сообщение клиентов, а также не всегда предоставляют необходимый уровень надежности.

Тема работы: «Проектирование программного комплекса управления надежной отправкой электронных писем с реализацией основного механизма рассылки и пользовательского интерфейса».

Тема дипломной работы актуальна в виду того, что существующие сервисы рассылки электронных сообщений не лишены разноплановых недостатков, поэтому выбор подходящего сервиса становится нетривиальной задачей.

**Целью** дипломной работы является проектирование программного комплекса управления рассылкой электронных сообщений с возможностью последующей интеграции в системы, требующие высокой надежности отправки.

Поставленная цель достигается путем решения следующих основных **задач**:

1. анализ процессов отправки и доставки электронного письма;
2. сравнительный анализ уже имеющихся систем для доставки писем;
3. разработка и анализ требований;
4. проектирование программного комплекса;
5. программная реализация базы данных, API, серверной части, WEB-интерфейса;
6. тестирование разработанного программного комплекса;
7. интеграция с информационной системой ПланФакт.

**Объектом исследования** является процесс рассылки электронных сообщений.

**Предметом исследования** в дипломной работе являются методы и средства улучшения надежности и контроля рассылки электронных сообщений.

# АНАЛИЗ ТРЕБОВАНИЙ

Разработка программного комплекса начинается со сбора информации о деловых процессах, подлежащих автоматизации, среде их функционирования и т.п. Цель этого процесса заключается в том, чтобы все заинтересованные стороны договорились относительно общего толкования конечного результата проекта.

Анализ требований включает в себя данные, полученные в результате сбора требований к ПО, их систематизации, документирования, анализа, выявления неполноты и разрешения конфликтов.

## Обзор процессов отправки и доставки сообщения

### Общие сведения

Обмен сообщениями является неотъемлемой частью любого взаимодействия. Если важна скорость взаимодействия, чаще всего стороны используют мессенджеры. Электронная почта же из-за своей специфики обязывает вкладывать достаточно смысла в каждое сообщение. Более того, электронная почта является важным звеном в регистрации и защите аккаунтов.

При взаимодействии крупных компаний, передающих важные документы посредством электронной почты, надежность доставки является одним из ключевых параметров. В таких случаях цена недоставленного сообщения может быть крайне высока.

Процесс отправки электронного сообщения разрабатываемым программным комплексом заключается в выборе подходящих сервисов доставки, передаче сообщения этим сервисам и последующей проверке того, доставлено и прочитано сообщение или нет.

### Основные понятия

Ниже представлены определения основным понятиям, необходимым в дальнейшей работе.

*Сообщение –* определенная информация, которую необходимо передать.

*Сервис доставки* – стороннее ПО, предоставляющее функционал для рассылки сообщений. Чаще всего использование таких сервисов бесплатно до определенного объема трафика.

*Отправленное письмо –* письмо, переданное сервису доставки.

*Получатель* – лицо, которому адресовано отправляемое сообщение.

*Доставленное письмо* – отправленное письмо, полученное получателем и не помеченное как спам.

*Отправитель* – лицо, запрашивающее отправку сообщения.

*Контакт* – сохраненный получатель пользователя.

*Статус доставки –* информация о результате доставки сообщения получателю.

*Фоновая служба доставки* – процесс, запущенный на сервере и функционирующий на протяжении всего времени работы программного комплекса. Отвечает за получение сообщения из очереди и передачу этого сообщения доступным сервисам доставки.

*Авторизация* – предоставление пользователю прав на выполнение определенных действий.

*Авторизованный пользователь* – пользователь, который успешно прошел этап авторизации и получил доступ к программному комплексу.

*Web-интерфейс* – веб-страница или несколько веб-страниц, позволяющие взаимодействовать с нужным сайтом, сервисом[1].

*API (Application programming interface* – набор функций, описывающих способы взаимодействия с серверной частью разрабатываемого программного комплекса*)* [2]

*Электронная цифровая подпись (ЭЦП)* – идентификатор документа, полученный в результате криптографического преобразования информации с использованием закрытого ключа подписи и позволяющий проверить отсутствие искажения информации в электронном документе, а также принадлежность подписи владельцу сертификата ключа подписи [3].

*Сертификат ключа подписи* – документ, содержащий открытый ключ, информацию о владельце ключа, подтверждающий принадлежность открытого ключа владельцу [4].

### Процесс отправки сообщения

Отправка сообщения происходит асинхронно, т.е. отправитель вовлечен в процесс только до момента подтверждения отправки.

На первом шаге отправитель составляет сообщение, может прикрепить к нему какие-либо файлы, выбирает получателя или список получателей, при желании может выбрать предпочтительный сервис отправки или разные сервисы отправки для разных получателей, может отложить отправку на запланированное время, либо указать предпочтительные диапазоны времени отправки. По окончании конфигурирования сообщения пользователь нажимает кнопку отправить.

Далее сервис пытается отправить сообщение получателям посредством выбранных отправителем сервисов или сервисами, выбранными по умолчанию для адресов выбранных получателей. Сервер отдает всю информацию о доставке сервисам доставки и ожидает от них ответ. Если сообщение не доставлено, сообщение передается следующему в очереди сервису доставки, и от него ожидается ответ. Данный цикл продолжается, пока сервер не получит ответ с успешным статусом доставки или пока не перепробует все доступные сервисы доставки.

На заключительном этапе формируется подробная информация о доставке. Эта информация заносится в журнал отправителя.

Процесс создания и отправки сообщения схематично изображен на рис. 1.1.



Рис. 1.1. Процесс отправки сообщения

Как уже отмечалось выше, сервисы для отправки сообщений предоставляют свой функционал бесплатно до достижения определенного объема трафика (например, определенное количество отправленных сообщений в месяц). Таким образом, для использования некоторых сервисов придется приобретать один из платных тарифов.

## Обзор программ-аналогов

В ходе обзора аналогов учитывались следующие функции:

1. отправка сообщения;
2. возможность конфигурирования способа доставки;
3. наличие повторной отправки недоставленного сообщения;
4. наличие и объем информации о доставке.

Полных программ-аналогов найдено не было. Поэтому были рассмотрены сервисы, частично реализующие рассмотренные функции:

1. Amazon Simple Email Service [5];
2. Sendpulse [27];
3. Tin-cat [7].

### Amazon SES

Amazon Simple Email Service [5] – это экономичный, гибкий и масштабируемый сервис электронной почты, с помощью которого разработчики могут отправлять электронные письма из любого приложения. Amazon SES предоставляет гибкую конфигурацию, в том числе возможность выбрать несколько вариантов использования электронной почты, включая отправку транзакций, маркетинговых писем или выполнение массовой рассылки.

Amazon SES обладает целым рядом сильных сторон:

1. быстрая интеграция;
2. эффективная отправка сообщений;
3. оптимизация доставки;
4. безопасное масштабирование.

Данный сервис широко применяется для отправки мгновенных сообщений в ответ на действия пользователя, например, для подтверждения регистрации или восстановления пароля, а также хорошо подходит для массовых рассылок.

При этом Amazon SES не лишен недостатков:

1. скорость отправки сообщений снижена из-за сложного алгоритма определения пути доставки;
2. относительная ненадежность (в случае неудачной доставки сервис лишь попытается поменять маршрут доставки).

### Sendpulse

Sendpulse [27] – сервис, с помощью которого можно создавать и распространять рассылки E-mail и SMS-сообщений.

Сервис предлагает следующие возможности:

* возможность рассылать различные типы сообщений: по электронной почте, через мессенджеры, в виде смс или Push-сообщений;
* интегрирование работы с мессенджерами Facebook и Viber;
* удобный конструктор, который сочетает богатые возможности с простотой использования;
* 130 шаблонов для рассылаемых писем;
* возможность автоматизации сложных процедур;
* средства для проведения сегментации и персонализации;
* возможность работы с 40 сервисами;
* продуманная и удобная админ-панель;
* бесплатный тариф, который подойдет тем, кто делает первые шаги.

Достоинства сервиса:

* возможность автоматизации сложных процессов работы;
* удобный встроенный блочный конструктор;
* много встроенных возможностей;
* работа с различными типами рассылок;
* гибкие и недорогие тарифы, возможность бесплатной работы.

Недостатки SendPulse:

* мало подписчиков на бесплатном тарифе;
* медленная работа техподдержки;
* часто изменяющееся API, из-за которого может страдать обратная совместимость;
* мало возможностей для создания писем-триггеров;
* если получатель пометил сообщение как «Спам», то сервис может блокировать аккаунт отправителя и приостанавливать дальнейшую рассылку.

### Tin-cat email queue

Tin-cat email queue [7] – система очереди для отправки сообщений. При попытке отправить сообщение, это сообщение отправляется в очередь. При этом каждую минуту система проверяет наличие сообщений в очереди и отправляет их.

Tin-cat позволяет регулировать частоту проверки очереди, а также количество сообщений, отправляемых за один период (за одну проверку).

Данная система является примером асинхронной отправки сообщений.

Недостатком системы является ее надежность. Если сообщение не будет доставлено, будет осуществлена попытка отправить его тем же способом, что не является эффективным способом. Также стоит отметить недостаточно гибкую конфигурацию данного сервиса (например, невозможность, регулировать интервал отправки сообщений).

### Сравнение

При анализе программ-аналогов были выделены ключевые недостатки рассматриваемого ПО, которые необходимо избежать при проектировании разрабатываемого программного комплекса.

Итоговое сравнение программ-аналогов представлено в таблице 1.1.

Знаком «+» обозначены преимущества соответствующих программ-аналогов, а знаком «-» недостатки (в рамках темы дипломной работы).

**Таблица 1.1.**

Сравнение программ-аналогов

|  |  |  |  |
| --- | --- | --- | --- |
| Критерий | Amazon SES | Sendpulse | Tin-cat email queue |
| Цена | + | + | + |
| Асинхронная отправка | + | + | + |
| Гибкость | + | + | - |
| Надежность | - | - | - |

Рассмотренные программы-аналоги, разработанные сторонними компаниями, не реализуют все указанные функции или реализуют их не в полной мере. Данные ПО позволяют лишь частично контролировать отправку сообщений. Также важным моментом является ненадежность API (Sendpulse). Важно: если сообщение не удается отправить, данные сервисы не предпринимают попыток отправить его другим способом, что отрицательно сказывается на надежности рассылки.

## Диаграмма вариантов использования

Так как в программном комплексе существует разделение пользователей по ролям (оператор и администратор), рассмотрим доступный функционал для каждой из ролей.

### Оператор

Оператору доступны отправка сообщения и просмотр истории сообщений.

При отправке сообщений оператор должен заполнить поле «Кому». Дополнительноон может добавить получателей, выбрать сервис доставки, время доставки или период, во время которого необходимо доставить сообщение. Также он может прикрепить к сообщению файлы. Из диаграммы видно, что ввод темы сообщения, а также ввод текста опциональны, что сделано для упрощения отправки сообщений и повышения гибкости разрабатываемого программного комплекса.

Оператор может сохранять контакты. Контакт – запись, представляющая собой пару из имени получателя и его почтового адреса. И затем в дальнейшем выбирать получателей из существующих аккаунтов (в форме создания сообщения при вводе получателя появляются контекстные подсказки из списка контактов).

Оператору доступен просмотр сообщений в своём журнале сообщений. Здесь оператор может проверить статус доставки, чтобы убедиться, что сообщение доставлено, и просмотреть время доставки.

Также оператор может просмотреть список доступных на данный момент сервисов доставки и особенности выбранного сервиса доставки.

### Администратор

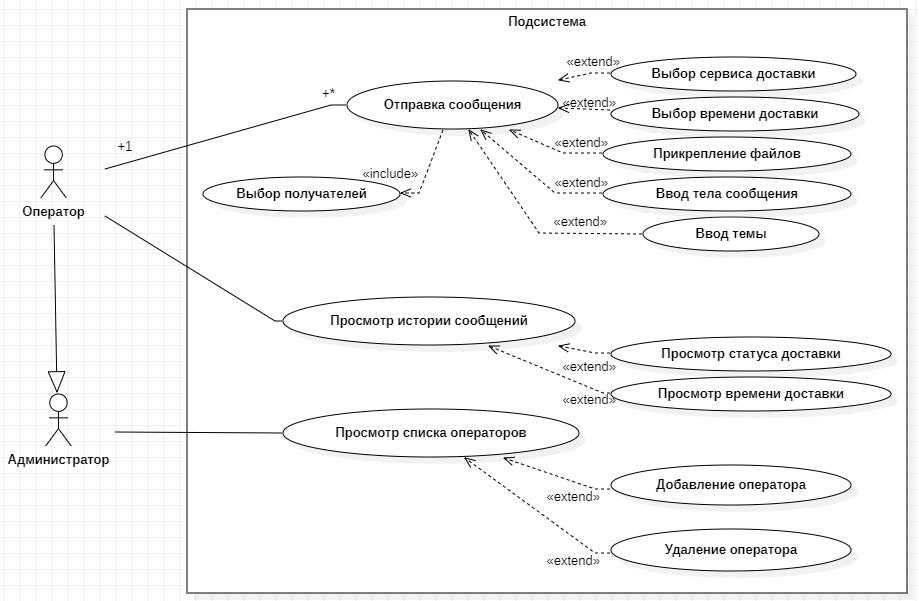
Администратор наследуется от оператора, т.е. ему доступен весь доступный оператору функционал.

Отличие администратора от оператора заключается в возможности просмотра списка операторов, где администратор может добавить или удалить оператора.

На данный момент разработан модуль авторизации, но все зарегистрировавшиеся и авторизованные пользователи обладают правами оператора.

В целях визуализации описанного выше функционала составлена диаграмма вариантов использования. Она представлена на рис. 1.2.

|  |
| --- |
| Рис. 1.2. Диаграмма вариантов использования |



## IDEF0-Диаграмма

Разработана IDEF0-диаграмма для процесса отправки сообщения.

На контекстной диаграмме (рис. 1.3) определен функциональный блок «Отправить сообщение через систему», механизм, предписание, входные и выходные данные.

На диаграмме первого уровня (рис. 1.4) выполнена декомпозиция базового блока «Отправить сообщение через систему». Этот блок содержит четыре подфункции: «Сформировать письмо», «Инициировать отправку сообщения», «Занести информацию о доставке в журнал». Проведена детализация функций «Сформировать письмо» (рис. 1.5) и «Инициировать отправку сообщения» (рис. 1.6)



Рис. 1.3. Контекстная диаграмма модели «Поиск объектов»



Рис. 1.4. Диаграмма первого уровня функциональной модели



Рис. 1.5. Процесс «Сформировать письмо»



Рис. 1.6. Процесс «Инициировать отправку сообщения»

## Модель бизнес-процесса отправки сообщения

В качестве бизнес-процесса был рассмотрен процесс отправки сообщения. В данном процессе участвуют четыре исполнителя:

1. пользователь как инициатор процесса;
2. клиентская часть, обрабатывающая введенные пользователем данные и отправляющая запрос на сервер;
3. серверная часть принимает запрос от клиентской части, формирует письмо, заносит его в базу данных и отправляет письмо в очередь;
4. сторонние сервисы доставки.

Клиентская часть обрабатывает введенные пользователем данные и отправляет запрос на сервер. Также клиентская часть получает ответ от сервера и представляет пользователю обратную связь в удобном виде.

Серверная часть принимает запрос от клиентской части, формирует письмо, заносит его в базу данных и отправляет письмо в очередь. Когда очередь доходит до данного письма, происходит выбор сервера доставки и передача письма выбранному сервису.

При получении неудовлетворительного ответа предпринимается попытка заново отправить письмо, но посредством других доступных сервисов доставки.

Фрагменты модели бизнес-процесса представлены на рис. 1.7-1.8.

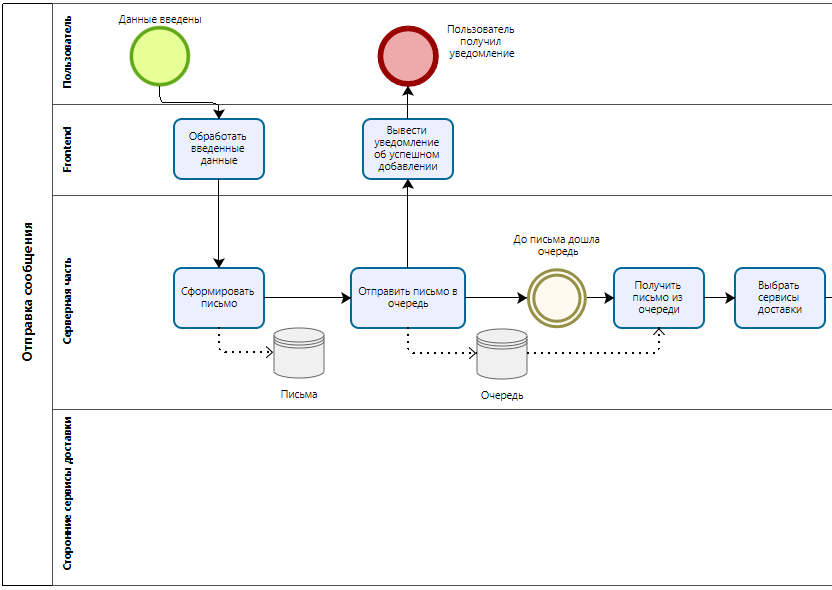


Рис. 1.7. Фрагмент диаграммы модели бизнес-процесса

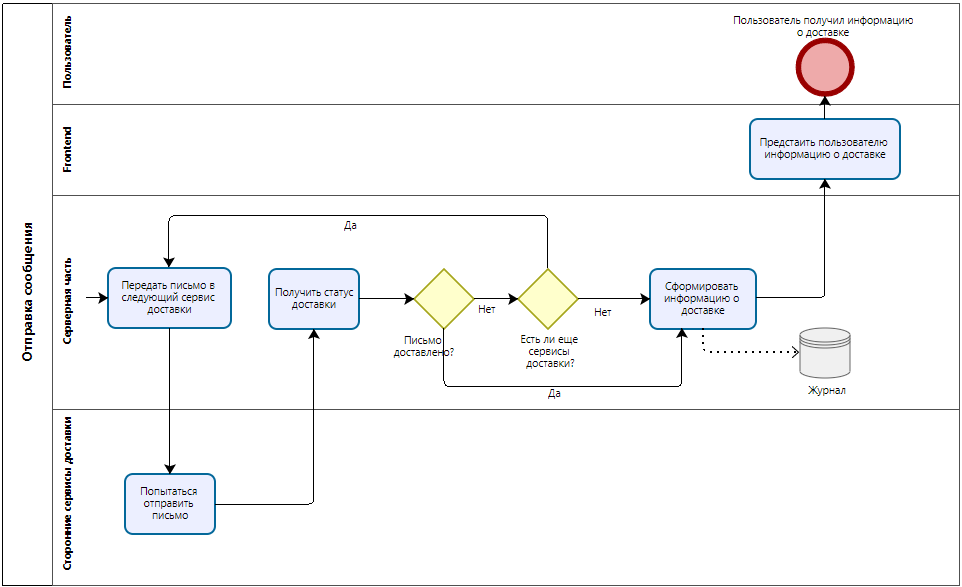


Рис. 1.8. Фрагмент диаграммы модели бизнес-процесса

## Выводы

В результате проведенного анализа была изучена предметная область (процесс отправки электронных сообщений), проведено сравнение и выявлены ключевые недостатки существующих программ-аналогов, описаны варианты использования программного комплекса, разработаны IDEF0-диаграмма и BPMN.

**Тема работы**: «Проектирование программного комплекса управления надежной отправкой электронных писем с реализацией основного механизма рассылки и пользовательского интерфейса».

Так как все рассмотренные программы-аналоги не предоставляют требуемый уровень надежности доставки сообщений, можно сделать вывод о необходимости разработки программного продукта, который должен решать все поставленные задачи.

**Целью дипломной работы** является проектирование программного комплекса управления рассылкой электронных сообщений с возможностью последующего внедрения в системы, требующие повышенной надежности отправки.

Поставленная цель достигается путем решения следующих основных **задач**:

1. анализ процесса отправки электронного письма;
2. сравнительный анализ уже имеющихся систем и платформ для обучения;
3. разработка и анализ требований;
4. проектирование программного комплекса;
5. программная реализация базы данных, API, серверной части, WEB-интерфейса;
6. тестирование разработанного программного комплекса;
7. внедрение в существующую информационную систему.

**Объектом исследования** является процесс отправки сообщений.

**Предметом исследования** в дипломной работе являются методы и средства улучшения надежности и контроля рассылки электронных сообщений.

# ТЕХНИЧЕСКОЕ ЗАДАНИЕ

Наименование программы: «Программный комплекс управления надежной отправкой электронных писем».

## Основание для разработки

Основанием для разработки подсистемы управления надежностью отправки электронных писем является задание на дипломную работу, выданное доцентом Трубаковым А.О, а также заявка на разработку от компании ООО «Максимум Веб», представленная в прилож. Г на рис. Г.1.

## Назначение и область применения

Разрабатываемый программный комплекс должен выполнять следующие задачи:

* асинхронная рассылка сообщений выбранным адресам с помощью выбранных сервисов доставки;
* обеспечение повышенной надежности доставки;
* возможность получения и просмотра подробной информации о статусе доставки сообщения.

## Требования к программному комплексу

### 2.3.1. Требования к функциональным характеристикам

Реализуемый программный комплекс должен включать в себя нижеупомянутые функциональные особенности:

1. Авторизация пользователей в web-интерфейсе для администрирования:

Роли:

оператор – может управлять рассылкой сообщений, просматривать соответствующую историю сообщений;

администратор – имеет все возможности оператора, но также может просматривать список операторов, создавать новых операторов и удалять существующих.

1. После осуществления входа в программный комплекс пользователь может:

отправлять сообщения на существующие электронные почтовые адреса;

выбирать сервисы доставки из списка доступных на данный момент;

прикреплять файлы к сообщению;

просматривать список доставленных сообщений;

просматривать информацию о доставке сообщений.

1. Разрабатываемый комплекс должен обладать следующими функциями:

по умолчанию в первую очередь осуществлять попытку отправлять сообщение с помощью SMTP-сервера;

возможность повторной отправки недоставленного сообщения, но уже посредством другого доступного сервиса;

возможность отправлять сообщения посредством SMTP-сервера и минимум **трех** различных сервисов для отправки сообщений;

в случае неудачной попытки отправить сообщение посредством каждого доступного сервера сообщение помещается в конец очереди;

наличие API, с помощью которого разрабатываемый программный комплекс можно будет встраивать в различные сервисы, использующие рассылку сообщений;

независимость от пользовательского интерфейса – возможность выполнять все функции с помощью методов Web API;

возможность прикреплять файлы к сообщению (до 5 файлов в сообщении общим объемом до 10 Мб). Доступные расширения: .doc, .docx, .xlsx, .csv, .png, .svg, .rar;

возможность хранить историю сообщений, т.е. хранить ранее отправленные сообщения и давать доступ к просмотру информации об их доставке;

возможность скачать ранее прикрепленные файлы.

1. Разрабатываемый комплекс предоставляет асинхронную отправку сообщений: когда оператор или администратор отправляет сообщение, сообщение отправляется в очередь на удаленном сервере. Система с настраиваемой периодичностью проверяет эту очередь и передает сообщения соответствующим сервисам доставки.
2. В целях улучшения надежности разрабатываемый комплекс должен включать в себя кластерную систему серверов.
3. Требования к интерфейсу

Реализуемый программный комплекс должен включать в себя нижеупомянутые функциональные особенности:

интерфейс для работы с подсистемой (интерфейс администратора) – web-страница, включает в себя список со всей информацией об отправленных сообщениях, предоставляет возможность вручную отправить сообщение через выбранный сервис, настраивать почтовые шлюзы;

необходимо обеспечить совместимость с основными браузерами (последние версии Chrome, Firefox, Safari, Opera; IE начиная с 10 версии). Верстка должна быть адаптивной и рассчитана на минимальное разрешение экрана 1280×720;

цветовая гамма приложения не должна быть излишне яркой, но сочетания цветов должны быть контрастными. Дизайн должен быть простым и понятным, нужно избегать непонятных иконок. Списки, с помощью которых также может быть реализовано меню, не должны превышать 5-9 элементов;

если время загрузки страницы составляет более 5с., необходимо обеспечить пользователю уверенность в том, что процесс действительно происходит при помощи индикатора выполнения процесса;

на длинных страницах необходимо применять ссылки, возвращающие пользователя в верхнюю часть страницы.

**Получение данных.** Система получает на вход:

1. тему;
2. тело сообщения;
3. список прикрепленных файлов;
4. список адресов, на которые нужно отправить сообщения;
5. выбранный сервис доставки, посредством которого необходимо доставить сообщение;

6) запланированное время доставки или диапазоны времени, в пределах которых сообщение должно быть доставлено.

**Выходные данные.** Система предоставляет пользователю историю сообщений с информацией о доставке, список контактов, список доступных сервисов доставки.

**Отправка данных.** Система обрабатывает полученные от пользователя данные через Web API [2] и передает их указанным сервисам доставки.

### 2.3.2. Требования к надежности

* + - 1. Требования к обеспечению надежного функционирования программы

Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением Заказчиком совокупности организационно-технических мероприятий, перечень которых приведен ниже:

1. организацией бесперебойного питания технических средств;
2. использованием лицензионного программного обеспечения;
3. регулярным выполнением рекомендаций Министерства труда и социального развития РФ, изложенных в Постановлении от 23 июля 1998 г. «Об утверждении межотраслевых типовых норм времени на работы по сервисному обслуживанию ПЭВМ и оргтехники и сопровождению программных средств»;
4. регулярным выполнением требований ГОСТ 51188-98. Защита информации. Испытания программных средств на наличие компьютерных вирусов.
   * + 1. Время восстановления после отказа.

Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать 30-ти минут при условии соблюдения условий эксплуатации технических и программных средств.

Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.

* + - 1. Отказы из-за некорректных действий оператора

Отказы программы возможны вследствие некорректных действий пользователя при взаимодействии с системой.

Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу оператора без предоставления ему административных привилегий, что реализовано посредством разделения ролей.

## Условия эксплуатации

### 2.4.1. Климатические условия эксплуатации

Климатические условия эксплуатации, при которых должны обеспечиваться заданные характеристики, должны удовлетворять требованиям, предъявляемым к техническим средствам в части условий их эксплуатации.

### 2.4.2. Требования к квалификации и численности персонала

Минимальное количество персонала, требуемого для работы программного комплекса, должно составлять не менее 1 человека: администратор, управляющий настройкой подсистемы.

### 2.4.3. Требования к составу и параметрам технических средств

Программный комплекс функционирует в составе уже имеющегося сервиса и не предъявляет дополнительных требований к составу и параметру технических средств. Подсистема должна работать на тех же самых устройствах, что и вся система учета.

Сервер приложения:

* наличие веб-сервера IIS.
* процессор: 3,1 ГГц и выше;
* ОЗУ: 16Гб;
* SSD диск.

Сервер баз данных:

* процессор: 3,1 ГГц и выше;
* ОЗУ: 16Гб и выше;
* SSD диск.

### 2.4.4. Требования к программной совместимости

В перечень программного обеспечения, работающего на стороне сервера входит следующее ПО:

* лицензионная версия операционной системы «Microsoft Windows Server 2019»;
* базы данных «Microsoft SQL Server 2019»;
* СУБД «SQL Server Management Studio»;
* web-браузер Google Chrome 89.0 или Safari 14, или Mozilla Firefox 91.6, или Opera 12;
* антивирусное программное обеспечение.

Дополнительные требования к защите программного обеспечения и информации не предъявляются.

## Программная документация

### 2.5.1. Предварительный состав программной документации

Состав программной документации:

* техническое задание;
* программа и методики испытаний;
* руководство оператора.

## Стадии и этапы разработки

### 2.6.1. Стадии разработки

Разработка подсистемы состоит из следующих этапов:

* разработка технического задания;
* проектирование программного комплекса;
* разработка программного комплекса;
* тестирование разработанного программного комплекса;
* написание программной документации;
* внедрение разработанного программного комплекса.

### 2.6.2. Содержание работ по этапам

На этапе разработки технического задания должны быть выполнены перечисленные ниже работы:

* постановка задачи;
* определение и уточнение требований к техническим средствам;
* определение требований к программе;
* определение стадий, этапов и сроков разработки программы;
* согласование и утверждение технического задания.

На этапе разработки программы должна быть выполнена работа по программированию (кодированию) и отладке программы.

Этап тестирования включает в себя выполнение тестов по указанным заранее шагам и сверку полученных результатов с указанными заранее ожидаемыми результатами.

Этап внедрения подразумевает развертывание разработанного программного комплекса на выделенном сервере с дальнейшим приведением подсистемы в стабильно работающее состояние. Результатом данного заключительного этапа является подписанный Заказчиком акт о внедрении.

## Порядок контроля и приемки

### 2.7.1. Виды испытаний

Для проверки корректности работы разработанного программного комплекса должны быть проведены следующие виды испытаний:

1. тестирование Web API методом черного ящика;
2. модульное тестирование;
3. кроссбраузерное тестирование.

В результате проведенного тестирования методом черного ящика и модульного тестирования все установленные заранее тесты должны быть пройдены успешно.

Кроссбраузерное тестирование должно подтвердить идентичное поведение разработанного программного комплекса в указанных в разделе «Требования к информационной и программной совместимости» браузерах.

Приемо-сдаточные испытания должны проводиться на объекте Заказчика в оговоренные сроки.

Приемо-сдаточные испытания программы должны проводиться согласно разработанной Исполнителем и согласованной Заказчиком Программы и методик испытаний.

Ход проведения приемо-сдаточных испытаний Заказчик и Исполнитель документируют в Протоколе проведения испытаний

### 2.7.2. Общие требования к приемке работы

На основании Протокола проведения испытаний Исполнитель совместно с Заказчиком подписывает акт о внедрении.

Программа должна обеспечивать возможность выполнения перечисленных ниже функций:

1. При входе в веб-сервис под учетной записью пользователь должен иметь возможность:
   * просматривать список сообщений;
   * создавать новые сообщения;
   * выбирать несколько получателей;
   * прикреплять до 5 файлов общим объемом до 10 МБ с расширением: .doc, .docx, .xlsx, .csv, .rar, .pdf, .txt.
2. Программный комплекс должен хранить сведения о:
   * сообщениях: дата доставки, тема сообщения, текст сообщения, размер, запланировано ли сообщение, запланированная дата доставки, отправлено ли сообщение, идентификатор очереди, в которой находится сообщение, выбранный пользователем сервис доставки, сервис доставки, посредством которого сообщение доставлено, идентификатор отправителя, статус доставки, дата создания сообщения, является ли текст сообщения html-страницей;
   * адресах получателей: электронный адрес получателя, сообщение, отправляемое на этот адрес;
   * очередях доставки: интервал доставки, дата последней отправки;
   * прикрепленных файлах: название файла, содержание файла, тип файла, сообщение, к которому файл прикреплен;
   * пользователях: имя и фамилия пользователя, электронный адрес почты пользователя, хэш пароля, статус активности пользователя, роль;
   * контактах: электронный адрес почты контакта, имя контакта, идентификатор пользователя, которому принадлежит данный контакт;
   * доступах: дата начала действия доступа, дата окончания действия доступа, идентификатор действующего тарифа, идентификатор пользователя, которому принадлежит данный доступ;
   * тарифах: название тарифа, цена;
   * скидках: наименование скидки, значение.

# ЭКОНОМИЧЕСКИЙ АНАЛИЗ

Перед началом разработки любой программной системы необходимо выяснить, насколько целесообразна ее разработка, как с точки зрения полезности, так и с точки зрения экономической эффективности. В технико-экономической части работы рассматриваются вопросы организации работ по созданию и внедрению программной системы, а также приводится расчёт ее себестоимости.

Структура себестоимости программного продукта отражена в табл. 3.1 и представлена на рис. 3.1. Расчеты себестоимости приведены в прилож. А.

**Таблица 3.1**

Структура себестоимости программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Элементы себестоимости** | **Сумма (руб.)** | **% в общ. сумме себестоимости** |
| 1 | Общая заработная плата исполнителя | 786 125 | 56,20 |
| 3 | Отчисления на социальные нужды (страховые взносы) | 95 702 | 6,84 |
| 4 | Арендные платежи за производственные (офисные) помещения | 134 572 | 9,62 |
| 5 | Амортизация используемых основных средств и нематериальных активов | 78 986 | 5,65 |
| 6 | Расходы на модернизацию и приобретение основных средств | - | - |
| 7 | Расходы на приобретение необходимого ПО | - | - |
| 8 | Расходы на интернет, связь | - | - |
| 9 | Расходы на канцелярские товары и расходные материалы | 4 200 | 0,30 |
| 10 | Прочие расходы | 299 114 | 21,39 |
| **Итого:** | | **1 398 699** | **100** |

|  |
| --- |
| Рис. 3.1. Структура себестоимости программного продукта |



# ПРОЕКТИРОВАНИЕ ПРОГРАММНОГО КОМПЛЕКСА

В процессе проектирования было принято следующее: система должна быть централизованной, т.е. все данные должны располагаться в центральном хранилище. Система должна иметь трехуровневую архитектуру, состоящую из следующих уровней: первый - клиент, второй - сервер, третий – хранилище в связи с ее масштабируемостью, гибкой настройкой, высокой безопасностью и надежностью, а также наличием большого объема документации.

Схема архитектуры ПС представлена на рис. 4.1.

## Клиентская часть

В процессе анализа было рассмотрено несколько альтернативных инструментов разработки web-интерфейса:

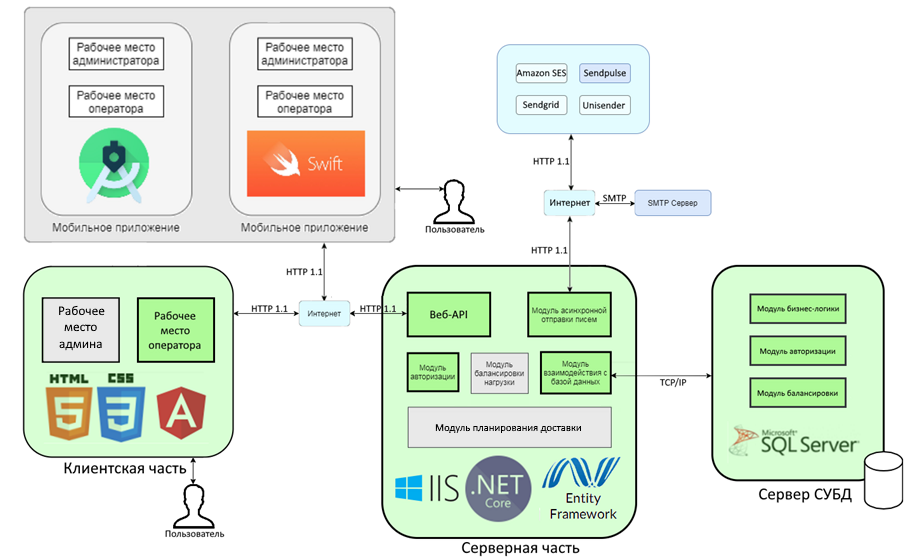
* + 1. React [18];
    2. Angular [17];
    3. Vue [19].

Среди выбранных фреймворков предпочтение было отдано Angular благодаря его декларативности, наличию развитого сообщества, модульности и MVC из коробки [17]. В качестве скриптового языка из-за ряда преимуществ над JavaScript был выбран TypeScript [21]. Для вёрстки будет применяться следующий набор инструментов: HTML + CSS [20], а также адаптивные компоненты Material и некоторые стили Bootstrap. Bootstrap и Material уже входят в Angular, поэтому отдельно подключать их не требуется.

## Серверная часть

Серверная часть включает в себя API, используемый для обработки запросов от клиентской части, набор функций для взаимодействия с хранилищем данных, модуль асинхронной отправки писем.

|  |
| --- |
| ***Рис. 4.1. Схема архитектуры ПС*** |



Серверная часть условно поделена на ряд модулей:

1. Web API;
2. модуль авторизации;
3. модуль балансировки нагрузки;
4. модуль асинхронной отправки писем;
5. модуль взаимодействия с базой данных.

Web API служит для взаимодействия пользовательской части и серверной части. Он включает в себя методы получения, добавления, изменения и удаления пользователей, сообщений и других сущностей [8].

Модуль авторизации отвечает за процедуру подтверждения подлинности конкретного человека, блокировку пользователей, а также за управление ролями зарегистрированных в системе пользователей.

Модуль балансировки нагрузки обеспечивает корректную работу кластерной системы серверов: координирует работу нескольких серверов, позволяет проверять их работоспособность и эффективность их работы, собирает статистику запросов, задавать подходящий метод распределения нагрузки. Подробная

Модуль асинхронной отправки писем является фоновой службой и служит для взаимодействия с доступными сервисами доставки и организации отправки писем, а модуль планирования доставки позволяет откладывать отправку сообщения на указанную дату и время.

Модуль взаимодействия с базой данных предоставляет интерфейсы для взаимодействия с таблицами сущностей. Для разработки данного модуля был выбран Entity Framework Core.

В качестве инструмента разработки серверной части была выбрана технология .NET Core, благодаря ее кроссплатформенности и удобству языка C# [22], а также благодаря детальной технической документации от официального разработчика [9].

## Модель данных

В ходе выполнения работы был проанализирован ряд различных СУБД, а именно:

1. MySQL [25];
2. MS SQL [24];
3. PostgreSQL [26].

Для разработки подошла бы каждая СУБД из указанных выше, но предпочтение было отдано MS SQL из-за большего опыта работы с ней.

База данных условно разделена на ряд модулей:

* + - 1. модуль бизнес-логики;
      2. модуль авторизации;
      3. модуль балансировки.

В базе данных хранятся все основные сущности системы:

1. сообщения;
2. пользователи;
3. ссылки на прикрепленные файлы;
4. контакты пользователей;
5. очереди доставки;
6. доступы пользователей;
7. тарифы;
8. скидки.

Также для осуществления отношения «многие ко многим» необходимы промежуточные таблицы для взаимодействия следующих таблиц:

* Пользователи – Доступы;
* Тарифы – Скидки.

Модуль бизнес-логики включает в себя таблицы основных бизнес-сущностей (таких как сообщения, файлы и другие) и отношения между ними.

Модуль авторизации включает в себя таблицы, в которых хранится информация о зарегистрированных пользователях, их ролях и доступах.

Модуль балансировки состоит из таблиц доступных серверов и их приоритетов.

## Низкоуровневое проектирование

### 4.4.1. ER-диаграмма

Разработана модель данных для программного комплекса управления рассылкой электронных сообщений.

Оператор передает информацию о сообщении в пользовательской части и при помощи API передает его на сервер, где информация о сообщении формализуется, заносится в базу данных и передается сервисам доставки.

В результате анализа процесс обучения можно обозначить несколько сущностей. Вот описание некоторых основных сущностей:

* + - 1. Messages – таблица для хранения сообщений:

messageId (long) – суррогатный первичный ключ;

userId (long) – id отправителя;

destinationDate (dateTime) – дата и время успешной доставки;

theme (nvarchar(max)) – тема сообщения;

body (text) – текст сообщения;

size (int) – размер сообщения в единицах (байт);

isScheduled (int) – логическое поле – запланирована ли отправка сообщения;

scheduleDate (dateTime) – запланированная дата и время, когда необходимо отправить сообщение;

isSent (int) – логическое поле – успешно ли доставлено сообщение;

deliveryQueueId (int) – id очереди, в которой содержится данное сообщение;

chosendDeliveryService (int) – id выбранного пользователем сервиса доставки;

usedDeliveryService (int) – id сервиса доставки, использованного для отправки;

deliveryStatus (int) – id статуса доставки;

creationDate (dateTime) – дата создания сообщения;

isHtml (bit) – логическое поле – является ли текст html-страницей

* + - 1. AttachedFiles – таблица для хранения ссылок на прикрепленные файлы:

id (long) – суррогатный первичный ключ;

title (int) – название файла;

fileType – тип файла;

messageId – сообщение, к которому файл прикреплен.

* + - 1. Users – таблица для хранения информации о пользователях:

userId (long) – суррогатный первичный ключ;

userName (nvarchar(max)) – имя пользователя;

userSecondName (nvarchar(max)) – фамилия пользователя;

email (nvarchar(max)) – почтовый адрес пользователя;

isActive (bit) – логическое поле – является ли пользователь активным

passwordHash (nvarchar(max)) – хэш, полученный из пароля пользователя;

role (int) – текущая роль пользователя.

* + - 1. UsersAccesses – таблица для связи таблиц пользователей и доступов

userId (long) – суррогатный первичный ключ пользователя;

accessId (long) – суррогатный первичный ключ доступа

* + - 1. Accesses – таблица для хранения информации о доступах

accessId (long) – суррогатный первичный ключ;

userId (long) – id пользователя, которому принадлежит данный доступ

startDate (dateTime) – дата и время начала подписки пользователя;

endDate (dateTime) – дата и время конца подписки пользователя;

tariffId (int) – id тарифа пользователя.

* + - 1. Contacts – таблица для хранения контактов пользователей

id (long) – суррогатный первичный ключ;

contactName (nvarchar(MAX)) – имя контакта;

contactEmail (ncarchar(MAX)) – электронный адрес контакта;

userId (long) – id пользователя, которому принадлежит эта запись.

* + - 1. DeliveryQueues – таблица для хранения информации об очередях

id (int) – суррогатный первичный ключ;

sendingIntervalSec (int) – интервал отправки сообщений.

* + - 1. Tariffs – таблица для хранения информации о тарифах

id (int) – суррогатный первичный ключ;

tariffName (nvarchar(255)) – название тарифа;

cost (int) – стоимость тарифа.

* + - 1. TariffsSales – таблица для связи таблиц тарифов и скидок

tariffId (int) – суррогатный первичный ключ тарифа;

saleId (int) – суррогатный первичный /ключ скидки.

* + - 1. Sales – таблица для хранения информации о скидках

id (int) – суррогатный первичный ключ;

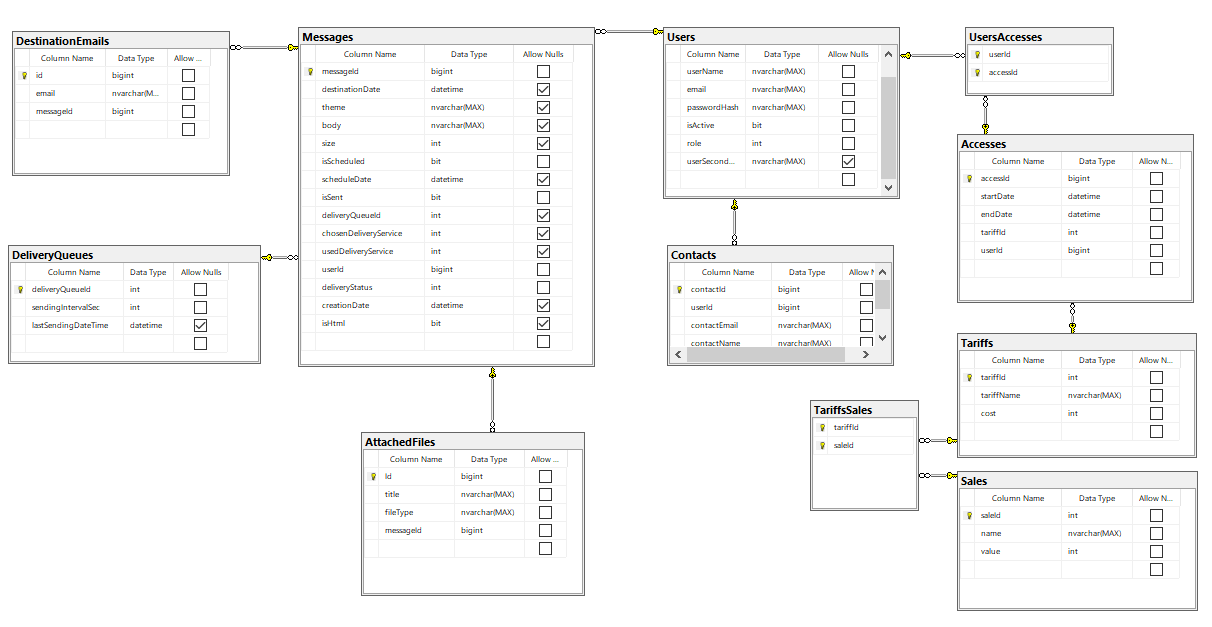
name (nvarchar(255)) – название скидки на тариф;

value (int) – размер скидки на тариф.

ER диаграммы применяются для моделирования и проектирования реляционных баз данных, причем как в плане логических и бизнес-правил (логические модели данных), так и в плане внедрения конкретных технологий (физические модели данных). В сфере разработки программного обеспечения ER-диаграмма, как правило, служит первым шагом в определении требований проекта по созданию информационных систем.

Спроектированная ER диаграмма находится в НФБК, так как каждая нетривиальная и неприводимая слева функциональная зависимость обладает потенциальным ключом в качестве детерминанта [10]. ER диаграмма представлена на рис. 4.2.

|  |
| --- |
| ***Рис. 4.2. ER диаграмма*** |



### 4.4.2. Диаграмма потоков данных

Разработана модель потоков данных для процесса создания сообщения и отправки сообщения.

На контекстной диаграмме (рис. 4.3) определены базовый блок «Система рассылки и учета электронных сообщений» и внешняя сущность «Пользователь», хранилища данных «Журнал», «Очередь», «Письма», а также информационные потоки между ними.

На диаграмме первого уровня (рис. 4.4) выполнена декомпозиция базового процесса на три подпроцесса. На диаграмме второго уровня проведена детализация процесса «Инициировать отправку письма» (рис. 4.5). На диаграммах третьего уровня проведена детализация процесса «Отправить письмо» (рис. 4.6).



Рис. 4.3. Контекстная диаграмма



Рис. 4.4. Диаграмма первого уровня модели потоков данных



Рис. 4.5. Процесс «Инициировать отправку письма»



Рис. 4.6. Процесс «Отправить письмо»

### 4.4.3. Диаграмма классов для службы отправки сообщений

Служба отправки сообщений разрабатываемого программного комплекса представляет собой фоновый процесс, запущенный на сервере.

Этот процесс циклически получает из базы данных сообщение из очереди и пытается отправить его с помощью интерфейса ISenderService. Класс SenderService реализует указанный интерфейс и последовательно передает сообщение классам через интерфейс ISender.

Классы, реализующие ISender:

1. SmtpSender;
2. SendpulseSender;
3. SendGrideSender.

В зависимости от результата SenderService формирует и возвращает соответствующий объект класса SendingResult.

Если результат положительный, то WorkerMain обновляет письмо: удаляет его из очереди и меняет его статус.

Диаграмма классов представлена на рис. 4.7.

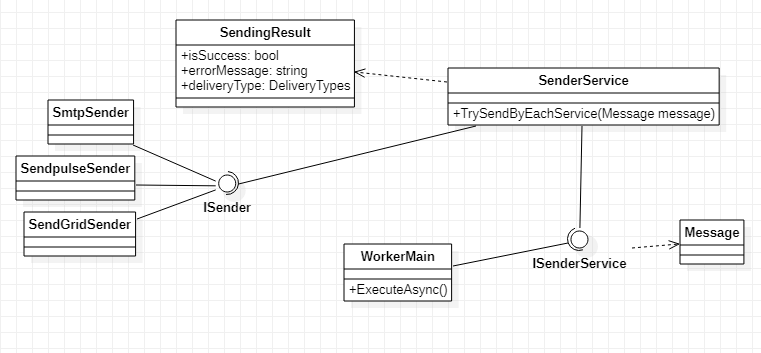


Рис. 4.7. Диаграмма классов

## Проектирование интерфейса

При проектировании интерфейса должны быть учтены следующие критерии эргономики:

1. скорость работы пользователя.
2. количество человеческих ошибок.
3. скорость обучения;
4. субъективная удовлетворенность пользователя.

Макеты интерфейсов создания сообщения, списка сообщений и списка доступных сервисов доставки представлены на рис. 4.8-4.11.

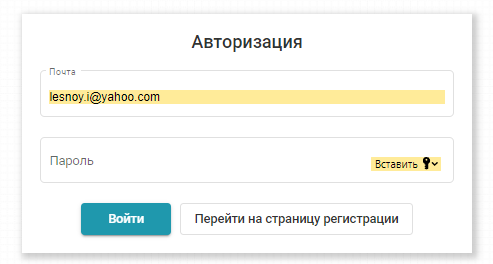
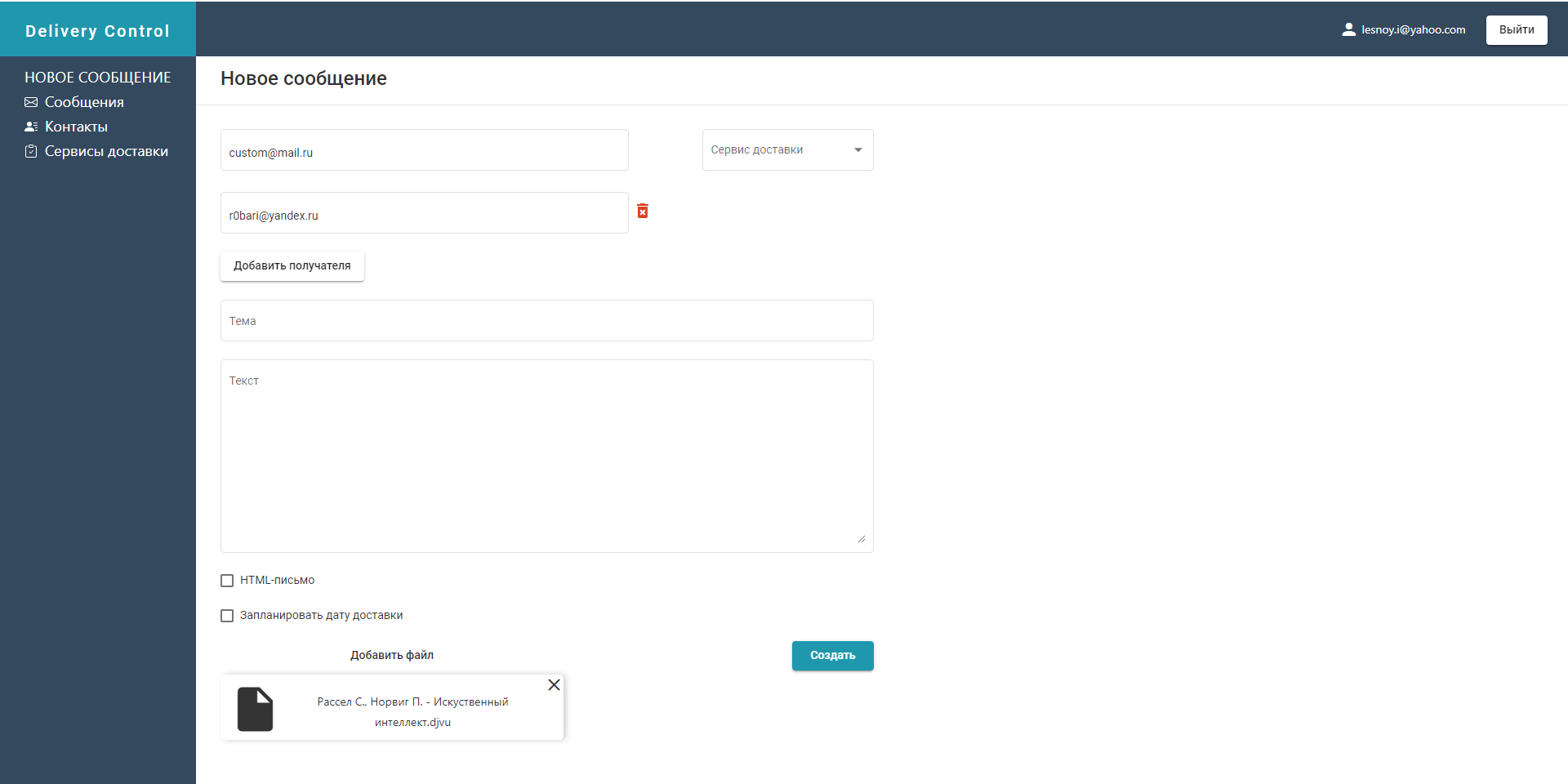
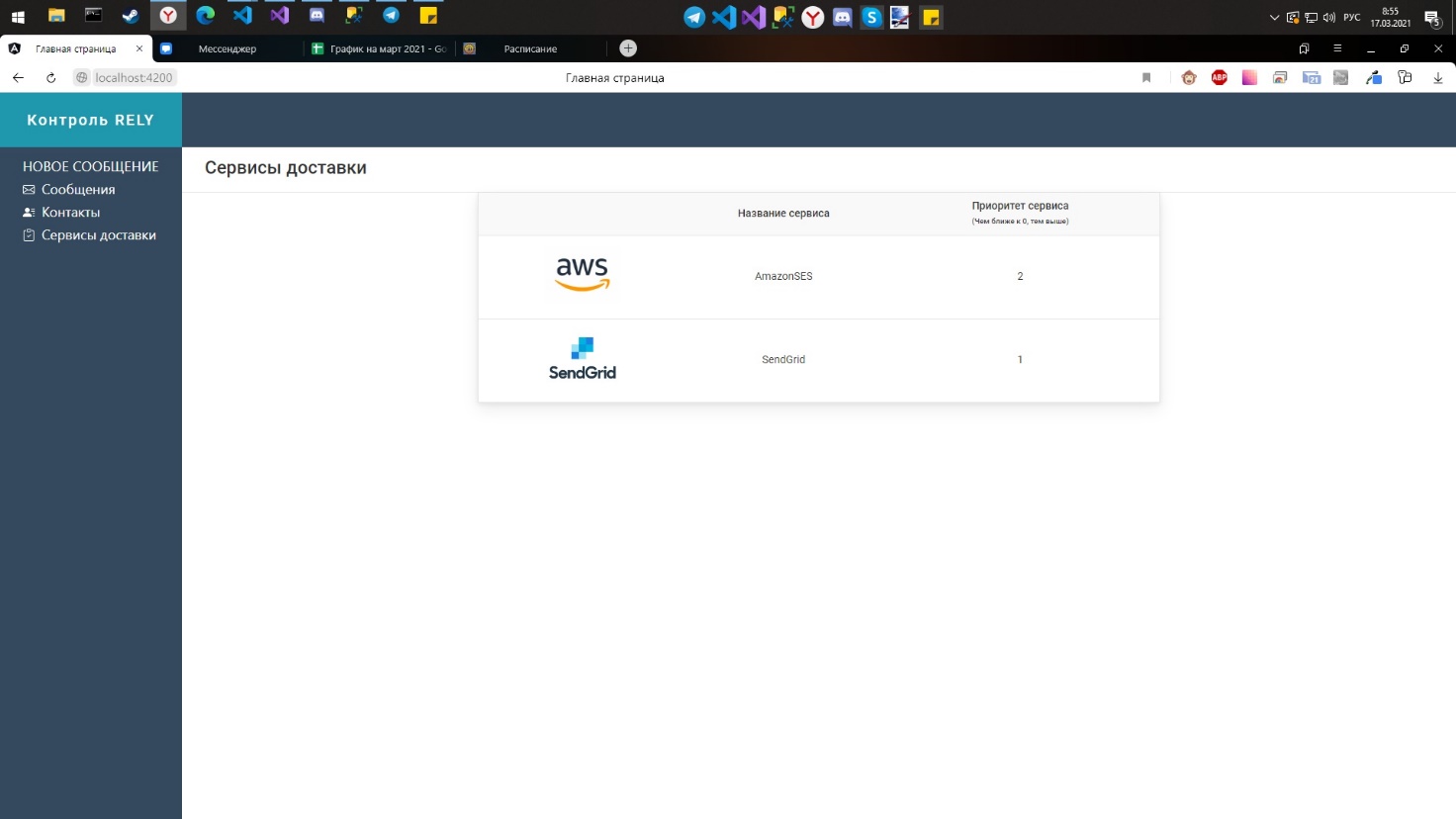


Рис. 4.8. Форма авторизации

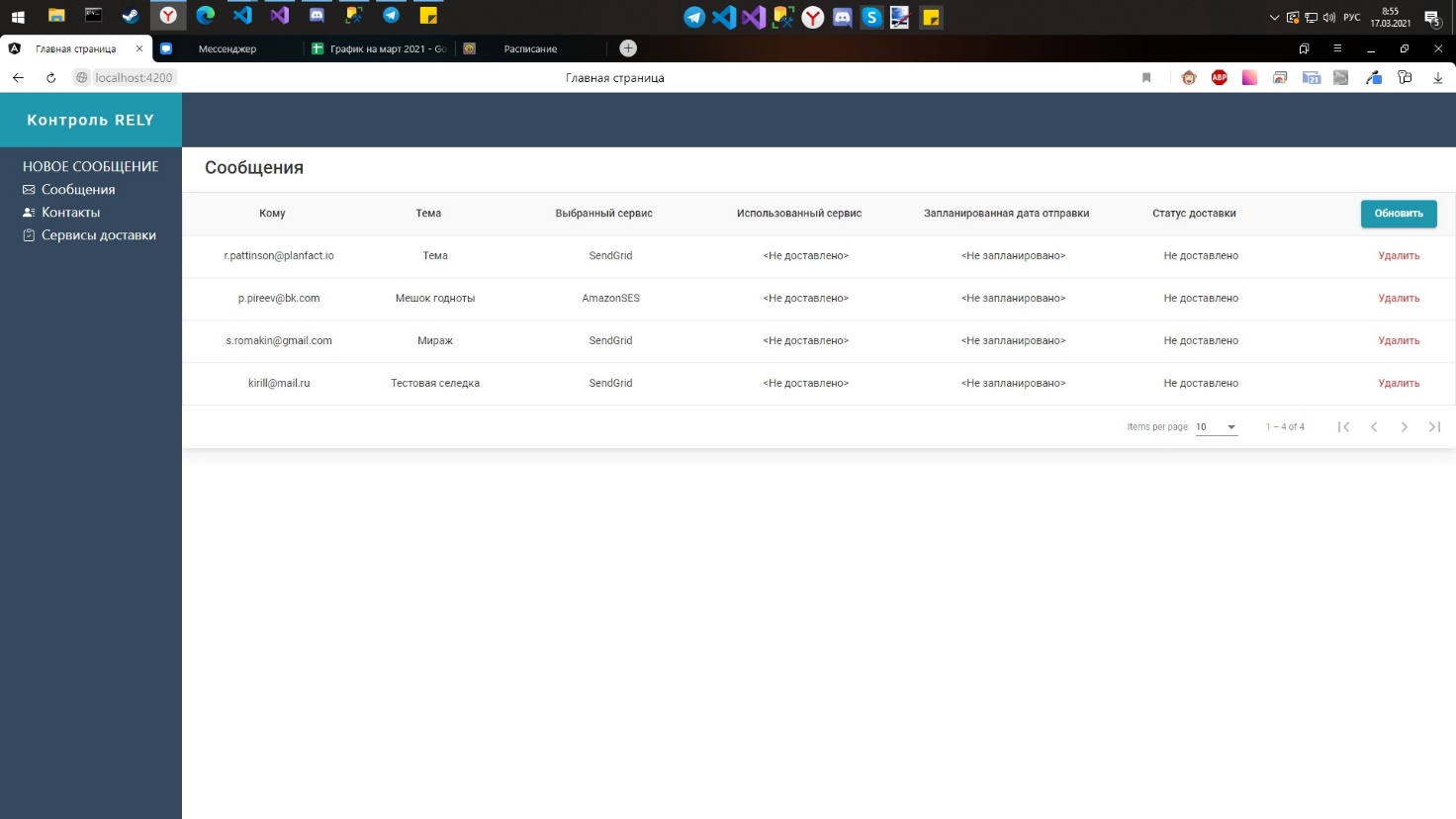
|  |
| --- |
| Рис. 4.9. Страница с доступными сервисами доставки |



|  |
| --- |
| Рис. 4.10. Страница с доступными сервисами доставки |



|  |
| --- |
| Рис. 4.11. Журнал сообщений |



## Проектирование системы балансировки нагрузки

В случае, когда на сервер одновременно поступает большое количество запросов, время обработки запросов закономерно вырастает, что делает использование программного комплекса менее удобным. Чтобы избежать этой проблемы, можно использовать несколько идентичных серверов и распределять по ним поступающие запросы по определенному правилу.

В таком случае один из серверов отвечает за прием запросов от пользователей и перенаправление запросов подходящим серверам для обработки (распределительный сервер).

Сервера, обрабатывающие запросы, принимают запрос от распределительного сервера, обрабатывают полученный запрос и возвращают ответ распределительному серверу. А тот в свою очередь возвращает полученный ответ пользователю.

Распределительный сервер выбирает, какому из серверов передать запрос на обработку по определенному алгоритму. Рассмотрим несколько доступных алгоритмов балансировки нагрузки.

Round Robin. Самый простой алгоритм, при котором поступающие запросы распределяются по серверам поочередно и циклически. Таким образом, каждый сервер будет обрабатывать равное количество запросов. Возможна ситуация, когда один сервер получит ряд очень простых запросов, и его ресурсы будут простаивать, а другой сервер получит такое же количество очень длительных запросов, из-за чего будет перегружен.

Weighted Round Robin. Модификация предыдущего алгоритма, в которой каждому обрабатывающему запросы серверу присваивается какой-то удельный вес. И распределительный сервер передает запросы серверам пропорционально их весам.

Least Connections. Алгоритм балансировки, при котором распределительный сервер передает очередной запрос тому серверу, на который в данный момент времени назначено наименьшее число запросов.

Заданная последовательность. Разновидность алгоритма Round Robin, при котором запросы распределяются по серверам в заданном, конфугирируемом порядке.

В целях улучшения гибкости в разрабатываемом программном комплексе возможно переключение между тремя описанными выше алгоритмами.

Схема взаимодействия серверов представлена на рис. 4.12.

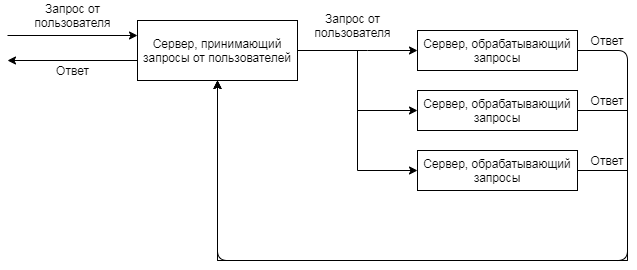


Рис. 4.12. Схема взаимодействия серверов

# ТЕСТИРОВАНИЕ ПРОГРАММНОГО КОМПЛЕКСА

Для проверки корректности разработанного программного комплекса планируется проведение следующих видов тестирования:

1. тестирование корректности работы Web API методом черного ящика;
2. модульное тестирование;
3. кроссбраузерное тестирование интерфейсов пользовательской части.

## Тестирование Web API методом черного ящика

Тестирование методом черного ящика – это функциональное и нефункциональное тестирование без доступа к внутренней структуре компонентов системы [12]. Метод тестирования «черного ящика» – процедура получения и выбора тестовых случаев на основе анализа спецификации (функциональной или нефункциональной), компонентов или системы без ссылки на их внутреннее устройство.

В рамках данного тестирования будут проверены методы GET, POST, PUT и DELETE следующих контроллеров Web API:

1. MessagesController;
2. UsersController;
3. ContactsController;
4. DeliveryQueuesController;
5. AttachedFilesController;
6. AuthController.

Пример тестов представлены в таблице 5.1.

**Таблица 5.1**

Тесты для тестирования методом черного ящика

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Входные данные** | **Ожидаемый результат** | **Соответствие полученного результата ожидаемому** |
| 1 | Запрос: GET https://localhost:44306/api/messages/list HTTP/1.1 | {  "attachedFiles": null,  "deliveryQueue": {  "deliveryQueueId": 2,  "sendingIntervalSec": 40  },  "chosenDeliveryService": {  "deliveryServiceId": 3,  "deliveryServiceName": "SendGrid",  "standartPriority": 1,  "connectionString": "'sendGrid road'"  },  "usedDeliveryService": null,  "user": {  "userId": 4,  "userName": "Evgeny Sukharev",  "email": "r0bari@yandex.ru",  "isActive": true,  "role": 2  },  "messageId": 7,  "theme": "Тема",  "body": "Обычный текст",  "destinationDate": "2021-01-16T21:05:00",  "destinationEmail": "r.pattinson@planfact.io",  "size": 50,  "isScheduled": false,  "scheduleDate": null,  "isSent": false,  "deliveryQueueId": 2,  "chosenDeliveryServiceId": 3,  "usedDeliveryServiceId": null,  "deliveryStatus": 1,  "userId": 4  } | Да |

**Таблица 5.1. Продолжение**

Тесты для тестирования методом черного ящика

|  |  |  |  |
| --- | --- | --- | --- |
| **№ теста** | **Входные данные** | **Ожидаемый результат** | **Соответствие полученного результата ожидаемому** |
| 2 | Запрос: POST <https://localhost:44306/api/auth/sign-in>  Body:  {  "email": "r0bari@yandex.ru",  "password": "Eviguf@77"  } | {  "data": {  "userId": 4,  "userName": "Evgeny Sukharev",  "email": "r0bari@yandex.ru",  "isActive": true,  "role": 2  },  "isSuccess": true,  "errorMessage": null  } | Да |
| 3 | Запрос: GET <https://localhost:44306/api/contacts/3> | {  "data": {  "user": {  "userId": 4,  "userName": "Evgeny Sukharev",  "email": "r0bari@yandex.ru",  "isActive": true,  "role": 2  },  "contactId": 3,  "userId": 4,  "contactEmail": "maxonfjvipon@yandex.ru",  "contactName": "Максим Трунников"  },  "isSuccess": true,  "errorMessage": null  } | Да |

## Модульное тестирование

Модульное тестирование, или юнит-тестирование (англ. unit testing) – процесс в программировании, позволяющий проверить на корректность отдельные модули исходного кода программы [13].

Идея состоит в том, чтобы писать тесты для каждой нетривиальной функции или метода. Это позволяет достаточно быстро проверить, не привело ли очередное изменение кода к регрессии, то есть к появлению ошибок в уже оттестированных местах программы, а также облегчает обнаружение и устранение таких ошибок.

В рамках модульного тестирования будут созданы новые проекты DeliveryRely.ServiceLayer.Tests и DeliveryRely.Domain.Tests.

В проекте Delivery.Domain.ServiceLayer.Tests содержатся тесты для методов класса Hash и других вспомогательных классов, таких как AutoMapper. Пример модульного теста из данного проекта представлен в прилож. В в листинге В.6.

В проекте DeliveryRely.Domain.Tests содержатся тесты для методов получения (методы Get…), добавления (методы Insert…), изменения (методы Update…) и удаления (методы Delete…) класса EFRepository. Пример теста представлен в прилож. В в листинге В.7.

## Кроссбраузерное тестирование интерфейсов

Тестирование кроссбраузерности — вид тестирования, направленный на поддержку и правильное полное отображение программного продукта в разных браузерах, мобильных устройствах, планшетах, экранах различного размера [14].

В рамках данного тестирования в самых распространенных браузерах были проверены основные интерфейсы клиентской части.

Список проверяемых браузеров:

1. Google Chrome 89.0;
2. Safari 14;
3. Mozilla Firefox 91.6 ESR;
4. Opera 12.

Список сравниваемых интерфейсов:

1. страница авторизации;
2. главная страница;
3. список сервисов доставки;
4. список сообщений;
5. список контактов;
6. страница создания сообщения;
7. страница информации о сообщении.

В качестве примера приводятся скриншоты интерфейсов страницы сообщений в браузерах Google Chrome (рис. 5.1) и Opera (рис. 5.2).

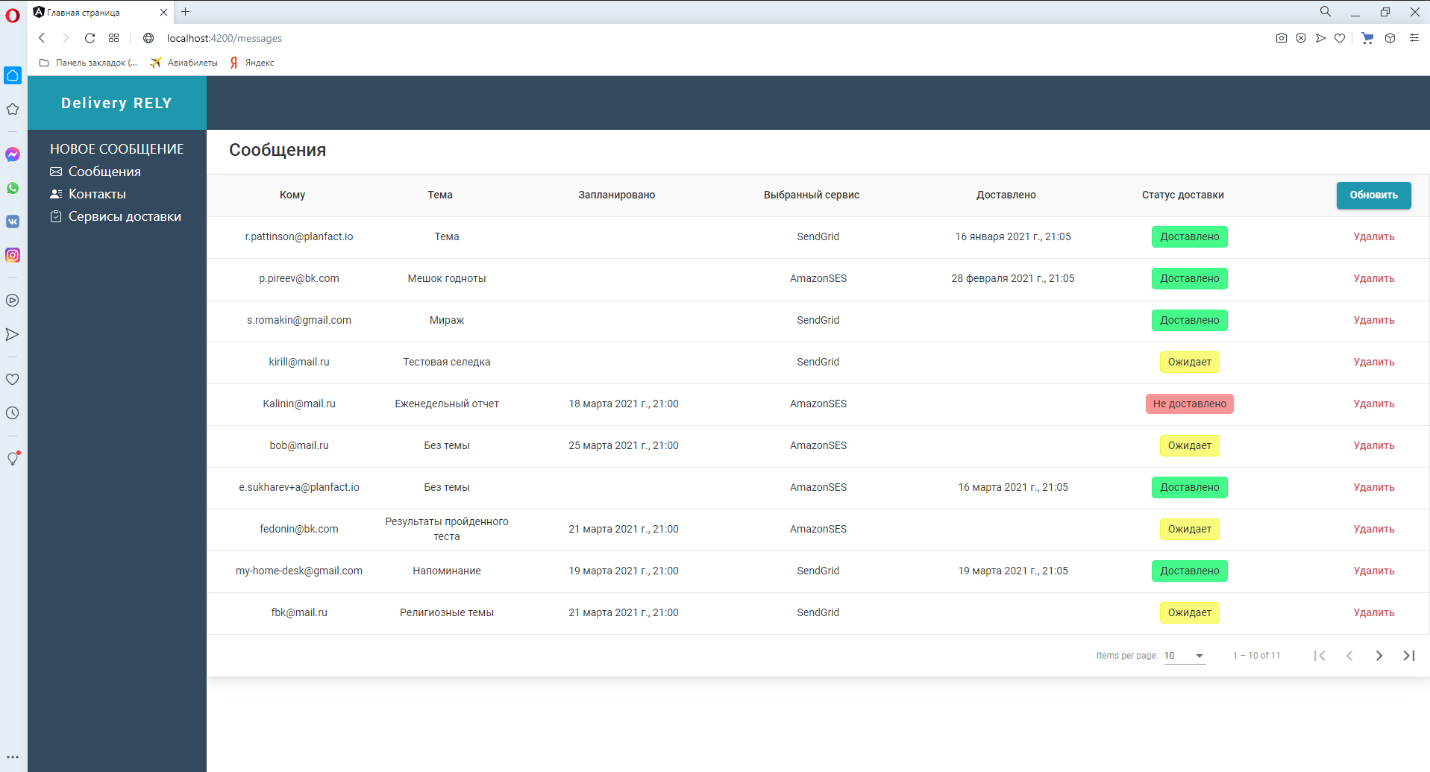
По итогам данного тестирования были выявлены определенные особенности отображения интерфейсов в некоторых браузерах. Например, последний столбец в компоненте таблицы из библиотеки Material отображался некорректно в браузере Mozilla Firefox, из-за чего таблица становилась шире, в браузере появлялась горизонтальная полоса прокрутки, и не было видно столбца с доступными действиями.

Данная ошибка была исправлена изменением стилей таблицы, после чего поведение таблицы во всех браузерах стало идентичным и корректным.

|  |
| --- |
| Рис. 5.1. Список сообщений в Google Chrome |



|  |
| --- |
| Рис.5.2. Страница сообщений в Opera |

******

# ОРГАНИЗАЦИОННАЯ ЧАСТЬ

Тема дипломной работы связана с разработкой программного комплекса управления надежной отправки электронных писем, предусматривающей эксплуатацию вычислительной техники и решение задач в помещениях, оборудованных ПЭВМ.

Все требования к помещению и используемому компьютерному оборудованию отражены в Санитарных правилах и нормах СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания» [28].

При проектировании рабочего места должна быть решена проблема освещения как искусственного, так и естественного.

Заданные параметры:

1. тип лампы – люминесцентная;
2. тип светильника – ЛД;
3. освещенность – *E* = 300 лк;
4. высота помещения – *H* = 3,3 м*;*
5. длина помещения – *a* = 9 м;
6. ширина помещения – *b* = 4 м;
7. высота подвеса светильника – *hp* = 2,5 м.

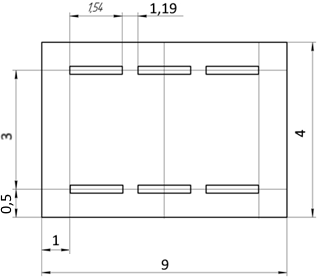
Результаты расчетов освещения обобщены в таблице 6.1, схема расположения светильников представлена на рис. 6.1, а сами расчеты представлены в прилож. Б.

**Таблица 6.1**

Результаты расчётов параметров осветительной установки

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| №  п/п | Тип  лампы | Световой  поток лампы Ф, лм | Количество  светильников | | Отклонение  nпр от nрасч, % | Мощность  лампы, Вт | Полная мощность N, Вт |
| расчётное  nрасч | принятое  nпр |
| 1 | ЛД-65 | 3390 | 6,37 | 6 | -5,80 | 65 | 780 |
| 2 | ЛД-80 | 3865 | 5,58 | 6 | 7,53 | 80 | 960 |

Вывод: оптимальным вариантом осветительной установки является ЛД-65, так как значение её полной потребляемой мощности является меньше, у осветительной установки ЛД-80.



***Рис. 6.1. Схема расположения светильников***

# ЗАКЛЮЧЕНИЕ

В ходе выполнения дипломной работы был проведён анализ процесса отправки и доставки электронных сообщений и анализ аналогов, разработана архитектура программного комплекса и структура базы данных, выполнена программная реализация рабочего места оператора, спроектирован основной механизм отправки посредством SMTP-сервера, сервисов доставки Sendpulse, Unisender, Amazon SES. В рамках дипломной работы была реализована отправка сообщений через SMTP-сервер и сервис Sendpulse, осуществляемая фоновой службой, запускаемой на серверы. Спроектированы основные интерфейсы клиентской части, разработаны сервисы для взаимодействия с Web-API. Добавлены тесты для всего разработанного функционала.

Спроектированный программный комплекс позволяет контролировать отправку сообщений, увеличивать надежность доставки. Комплекс ориентирован на компании, для которых крайне важно, чтобы письмо дошло до получателя.

Проведена интеграция программного комплекса с веб-приложением ПланФакт. Акт внедрения представлен в прилож. Г на рис. Г.2. и в данный момент находится на этапе тестирования в компании ООО «Максимум Веб».

Спроектированный программный комплекс может быть изменен для дальнейшего увеличения надежности доставки путем реализации доставки сообщений большим числом сервисов доставки. Таким образом, доступно горизонтальное масштабирование механизма доставки.

Перспективой развития является программная реализация спроектированного рабочего места администратора, спроектированной доставки по расписанию, а также доставки с помощью сервисов Unisender и Amazon SES.

# СПИСОК ЛИТЕРАТУРЫ

1. Что такое web-интерфейс: [Электронный ресурс]. URL: <https://semantica.in/blog/veb-interfejs.html/>.
2. Проектирование веб-API / Перевод с английского Д.А. Беликова: [Текст] / Арно Лоре – Оформление, издание, ДМК Пресс, 2020.
3. Виды электронных подписей: [Электронный ресурс]. URL:  <http://www.consultant.ru/document/cons_doc_LAW_112701/d9cd621c949a3c9efef51c2884c247e18ab9908b/>.
4. Сертификат ключа подписи: [Электронный ресурс]. URL:  https://uc.iitrust.ru/articles/chto-takoe-sertifikat-klyucha-proverki-elektronnoy-podpisi/.
5. Официальный сайт Amazon SES: [Электронный ресурс]. URL:  https://aws.amazon.com/ru/ses/.
6. Официальный сайт Sendgrid: [Электронный ресурс]. URL:  https://sendgrid.com.
7. Проект Email queue пользователя tin-cat на Github: [Электронный ресурс]. URL:  <https://github.com/tin-cat/emailqueue>.
8. Введение в Web API: [Электронный ресурс]. URL: <https://metanit.com/sharp/mvc/12.1.php>.
9. Документация по ASP.NET: [Электронный ресурс]. URL: <https://docs.microsoft.com/ru-ru/aspnet/core/?view=aspnetcore-3.1>.
10. Нормализация сущностей: [Электронный ресурс]. URL: <https://habr.com/ru/post/254773>.
11. UML основы. Краткое руководство по стандартному языку объектного проектирования: [Текст] / Мартин Фаулер – Санкт-Петербург, 2005.
12. Тестирование методом черного ящика: [Электронный ресурс]. URL:  <https://habr.com/ru/post/462837/>.
13. Модульное тестирование или юнит-тестирование: [Электронный ресурс]. URL:  <https://habr.com/ru/post/169381/>.
14. Кроссбраузерное тестирование: [Электронный ресурс]. URL:  <https://imajor.ru/razrabotka/verstka/krossbrauzernost-sayta>.
15. Язык UML. Руководство пользователя: [Текст] / Буч Г., Рамбо Д., Якобсон И. – 2-е издание: Пер. с англ Мухин Н., М.: ДМК Пресс, 2012.
16. Документация по C#: [Электронный ресурс]. URL: <https://docs.microsoft.com/ru-ru/dotnet/csharp/>
17. Официальная страница Angular: [Электронный ресурс]. URL: <https://angular.io/?developerstash>.
18. React в действии: [Текст] / Марк Тиленс Томас – СПб.: Питер, 2019.
19. Vue.js в действии [Текст] / Эрик Хэнчетт, Бенджамин Листоун – СПб: Питер, 2019.
20. Большая книга CSS: [Текст] / Дэвид Макфарланд – СПб.: Питер, 2016.
21. Современный учебник JavaScript: [Электронный ресурс]. URL: <https://learn.javascript.ru/>
22. Язык программирования C# 7 и платформы .NET и .NET Core [Текст] / Троелсен, Джепикс – 2018.
23. Software Engineering for Internet Applications by Eve Andersson [Текст] / Philip Greenspun, Andrew Grumet – 2006.
24. Microsoft SQL Server: [Текст] / Душан Петкович – Санкт-Петербург «БХВ-Петербург» – 2013.
25. My SQL 5.0. Библиотека программиста: [Текст] / Виктор Гольцман – Питер, Санкт-Петербург, 2010.
26. Техническая документация по PostgreSQL: [Электронный ресурс]. URL: <https://postgrespro.ru/docs/postgresql>
27. Официальный сайт Senpulse: [Электронный ресурс]. URL: <https://sendpulse.com/ru>
28. СанПиН 1.2.3685-21 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы. Введ. 30.06.03. – М.: Федеральный центр госсанэпиднадзора Минздрава России, 2003. – 54с.