Министерство образования и науки РТ

ГАПОУ «Бугульминский машиностроительный техникум»

КУРСОВОЙ ПРОЕКТ

Автоматизированная информационная система предоставления услуг в ИТ отделе предприятия

КП.МДК.02.01.09.02.07.011.000ПЗ

|  |  |
| --- | --- |
| Выполнил | Малахов А.В. |
|  | Фамилия, И.О. |
| Руководитель проекта | Федотова Л.И. |
|  | Фамилия, И.О. |
| Оценка | / / |
| Подпись |  |
|  | подпись |

Бугульма, 2024г.

Содержание

[Введение 3](#_Toc163104588)

[1 Общая часть 6](#_Toc163104589)

[1.1 Постановка задачи 6](#_Toc163104590)

[1.2 Требования к программному средству 7](#_Toc163104591)

[1.3 Стадии и этапы разработки 8](#_Toc163104592)

[2 Технологическая часть 10](#_Toc163104593)

[2.1 Разработка алгоритма решения задачи 10](#_Toc163104594)

[2.2 Разработка пользовательского интерфейса 13](#_Toc163104595)

[2.3 Нормализация модели данных 14](#_Toc163104596)

[2.4 Физическая модель базы данных 20](#_Toc163104597)

[2.5 Описание структуры входной/выходной информации 23](#_Toc163104598)

[2.6 Руководство пользователя 26](#_Toc163104599)

[2.7 Руководство программиста 35](#_Toc163104600)

[Заключение 38](#_Toc163104601)

[Список используемых источников 39](#_Toc163104602)

[Список нормативных документов 40](#_Toc163104603)

# Введение

Темой данного курсового проекта является "Автоматизированная информационная система предоставления услуг в ИТ отделе предприятия". Данная система предназначена для предоставления различных услуг, таких как распечатка документов на разных форматах, использование 3D-принтера для печати деталей и другие подобные услуги. Важно отметить, что типы поддерживаемых услуг могут динамически изменяться.

Актуальность темы данного проекта заключается в том, что с развитием информационных технологий и повышением требований пользователей к сервисам, автоматизация процесса предоставления услуг становится необходимостью. Компании и организации нуждаются в эффективной системе, которая позволит им управлять и предоставлять различные ИТ-услуги своим клиентам.

Целью данного курсового проекта является анализ и разработка решения, связанного с созданием автоматизированной информационной системы для предоставления услуг в ИТ отделе предприятия. Это включает в себя исследование существующих проблем в процессе предоставления услуг, а также разработку и внедрение программного решения, которое обеспечит эффективное управление услугами и повысит удовлетворенность клиентов.

Для достижения поставленной цели необходимо выполнить следующие задачи:

* Описать назначение и область применения разрабатываемой информационной системы.
* Определить входную и выходную информацию, необходимую для функционирования системы.
* Разработать физическую модель работы системы, определив последовательность операций и взаимодействия с пользователями.
* Создать базу данных и провести нормализацию отношений для хранения информации о предоставляемых услугах, клиентах и других сущностях.
* Разработать внутренний и внешний интерфейсы системы, обеспечивающие удобство использования и эффективность работы.
* Провести тестирование разработанной системы для проверки ее функциональности и корректности работы.

Объектом изучения для курсового проекта является процесс предоставления услуг в ИТ отделе предприятия. Этот процесс включает в себя широкий спектр деятельности, связанный с обслуживанием клиентов и предоставлением различных ИТ-услуг. Автоматизация этого процесса позволит улучшить его эффективность и качество, а также повысить удовлетворенность клиентов.

Предметом изучения в данном курсовом проекте является разработка автоматизированной информационной системы предоставления услуг в ИТ отделе предприятия. Эта система будет предназначена для управления и контроля процесса предоставления услуг, включая учет и обработку заявок, планирование и распределение ресурсов, а также отчетность и аналитику.

Информационная база исследования включает в себя различные источники информации, такие как специализированные журналы и издательства, научные статьи, Интернет-ресурсы, а также опыт и знания экспертов в области информационных систем и предоставления услуг в ИТ отделе предприятия.

Методология, используемая в данном курсовом проекте, включает анализ требований и потребностей пользователей, проектирование системы, разработку программного решения, тестирование и внедрение. В процессе анализа требований будет проведено исследование существующих проблем и процессов в предоставлении услуг, а также определены функциональные и нефункциональные требования к системе. На основе этого анализа будет разработана концептуальная и физическая модель системы, определена структура базы данных и разработаны интерфейсы.

Для реализации программного решения могут использоваться различные технологии и инструменты, такие как язык программирования, базы данных, фреймворки и другие. Выбор конкретных технологий будет зависеть от требований проекта, возможностей предприятия и предпочтений разработчиков.

Структура проекта соответствует логике исследования и включает в себя введение, общую часть, постановку задачи, требования, стадии и этапы разработки, технологическую часть, разработку алгоритма решения задачи, разработку пользовательского интерфейса, нормализацию модели данных, физическую модель базы данных, описание структуры входной/выходной информации, руководство пользователя, руководство программиста, заключение, список использованных источников.

Успешная реализация данного курсового проекта позволит предприятию создать эффективную и удобную систему предоставления услуг в ИТ отделе. Это повысит производительность и качество работы, улучшит обслуживание клиентов и удовлетворенность пользователей.

# 1 Общая часть

## 1.1 Постановка задачи

Электронная система управления взаимоотношениями с клиентами «ServiTech» представляет собой веб-приложение, оказывающее услуги информационного или производственного характера, например: 3D печать детали, рефакторинг кода, печать документации на бумаге любого формата. В приложении должно быть 3 роли: администратор, пользователь и бухгалтер.

Не авторизованным пользователям доступ в систему запрещен. Необходимо это для сохранения конфиденциальности данных и предотвращение проблем с уязвимостью информационной системы.

Для отправки заявки, необходимо зарегестрироваться, либо войти в аккаунт. После чего заполнить личные данные и подтвердить аккаунт по почте. Данная функциональная возможность необходима для предотвращения создания множественных аккаунтов и последующего спама. После чего выбрать из списка поддерживаемых услуг нужную, ввести описание, прикрепить файл, отправить заявку и дождаться отклика от администратора.

Для обработки отправленных заявок, необходимо аналогично, иметь аккаунт с ролью администратора. В рабочей области выбрать конкретного пользователя, выбрать его заявку, скачать переданный им файл, установить соответствующий статус и цену. Администратор так же может изменять список поддерживаемых услуг. В соответствующем каталоге он может его модифицировать: добавлять, удалять или изменять конкретные элементы списка.

Для отчетности и хранения данных о выполненных или отклоненных заявок необходим учетная запись с ролью бухгалтера. Здесь можно просматривать и выводить на печать отчеты о выполненных за определенный день работах.

Приложение должно выполнять следующие задачи:

* просмотр заявок;
* получение сведений о поддерживаемых услугах;
* изменение статуса заявки;
* изменение цены услуги;
* размещение новых заявок;
* редактирование описания заявки;
* регистрация;
* авторизация;
* отправка отчетов в бухгалтерию;
* удаление выполненных или отклоненных заявок;
* вывод отчетов.

Для достижения поставленных задач необходимо реализовать:

* Создать обобщенную блок-схему.
* Реализовать базу данных.
* Реализовать программный код.
* Протестировать программу.
* Создать выводы на основе проведенных исследований.

## 1.2 Требования к программному средству

В состав системы предоставления услуг в ИТ отделе должны входить следующие возможности:

* предоставление информации о клиентах;
* предоставление информации о заявках;
* предоставление информации о поддерживаемых услугах;
* возможность выполнения добавления, удаления, поиска, обновления, редактирования данных о заявках, услугах, клиентах;
* выполнять авторизацию и регистрацию в приложение.

Техническое обеспечение системы должно максимально и наиболее эффективным образом использовать существующие технические средства.

Программное средство должно быть доступно для использования на компьютерах с операционными системами Windows, Linux, MacOS.

В состав комплекса входят следующие технические средства:

* Процессор: тактовая частота процессора не ниже 3 ГГц, минимальное количество ядер - 4 ядра;
* ОЗУ: объем оперативной памяти не ниже 8 ГБ;
* Свободного места на жёстком диске: не менее 1 Гб;
* Наличие периферийных устройств: монитор, клавиатура, компьютерная мышь.

Требования к информационной и программной совместимости:

Для того чтобы приложение работало, входят следующие программные совместимости:

* ЭВМ должна работать под управлением операционной системы не ниже, чем Windows 7;
* база данных работает под управлением СУБД PostgreSQL;
* администрирование базы данных осуществляется с помощью pgAdmin 5
* Браузер Chrome.

Требования к транспортированию и хранению не предъявляется.

## 1.3 Стадии и этапы разработки

Разработка продукта была поделена на 4 стадии: подготовка, проектирование, создание, документирование.

Таблица 1 – этапы разработки программного продукта

|  |  |  |
| --- | --- | --- |
| № | Постановка задачи ПП | Сроки выполнения |
| 1 | Подготовка | 23.02.24 – 27.02.24 |
| 2 | Проектирование | 28.02.24 – 01.03.24 |
| 3 | Создание | 04.03.24 – 06.03.24 |
| 4 | Документирование | 07.03.24 – 11.03.24 |

Все этапы разработки представлены на диаграмме Ганта (Рисунок 1)

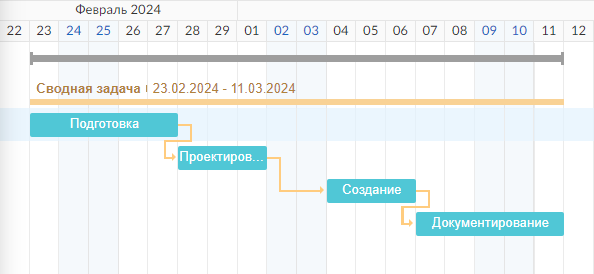


Рисунок 1 – Этапы разработки

# 2 Технологическая часть

## 2.1 Разработка алгоритма решения задачи

Для разработки алгоритма решения задачи, был выбран метод создания алгоритма логическим проектированием, который представлен на рисунке 2.

Логическое проектирование — это процесс создания логической модели данных, которая описывает структуру информации и отношения между различными элементами данных в системе. Логическое проектирование включает в себя разработку схемы базы данных на основе этой модели.

Блок-схема — распространённый тип схем, описывающих алгоритмы или процессы, в которых отдельные шаги изображаются в виде блоков различной формы, соединённых между собой линиями, указывающими направление последовательности.

Алгоритм сборки заявки:

* пользователь выбирает тип предоставляемой услуги из динамически подгружаемого списка;
* пользователь указывает описание к заявке;
* пользователь загружает файл соответствующий выбранному типу услуги;
* пользователь отправляет заявку на обработку.

Алгоритм обработки заявок:

* Администратор выбирает текущего пользователя
* Администратор выбирает заявку
* Администратор указывает статус и цену заявки
* Администратор отправляет обновленный объект заявки клиенту

Алгоритм отправки отчета о выполненной работе:

* Администратор может отправить отчет о выполненной работе, в случае, если имеется хотя бы одна выполненная или отклоненная заявка;
* Пользователь с ролью «бухгалтер» может просмотреть все отчеты по дате.
* Пользователь с ролью «бухгалтер» может вывести выбранный отчет на печать.

Программа «ServiTech» представляет собой автоматизированную информационную систему. Она предназначена для решения задач обработки данных.

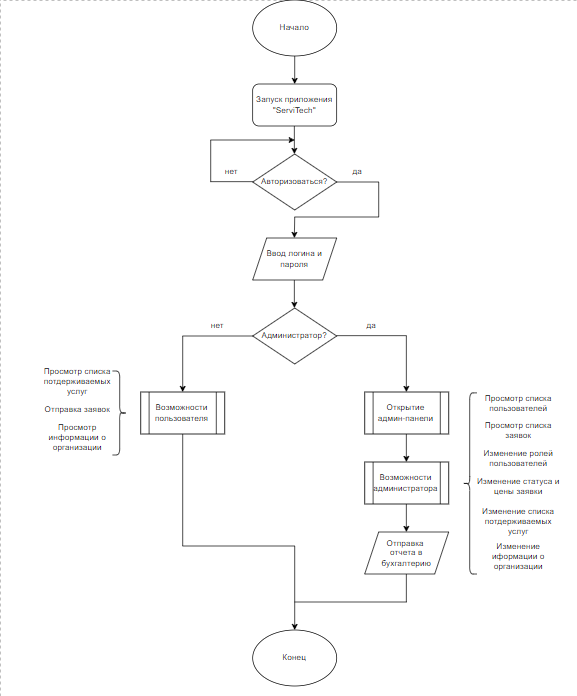


Рисунок 2 – Блок-схема программы

## 2.2 Разработка пользовательского интерфейса

Разработка пользовательского интерфейса программы является ключевым этапом, определяющим удобство взаимодействия пользователя с программным обеспечением. Интерфейс включает различные элементы, такие как кнопки, поля ввода, списки выбора и иконки, которые организованы для удобства в использовании.

Эргономичность интерфейса зависит от множества факторов, включая расположение элементов, контрастность, четкость и понятность подписей. Основные качества удобного интерфейса включают простоту использования, логичную структуру, информативность, приятную цветовую схему и шрифты, наличие обратной связи, отсутствие ошибок и багов, обучающие материалы и единообразие элементов.

Интерфейс разрабатываемого программного продукта ориентирован на максимальное удобство пользователя. Он прост и интуитивно понятен, обеспечивает быструю навигацию, содержит необходимую информацию без излишней детализации, обеспечивает хорошую обратную связь и лишен ошибок и багов.

Интерфейс понятен и удобен, все элементы находятся на своих традиционных местах. Для удобной навигации по приложению было разработано выдвижное меню – в свернутом виде оно не занимает много места, что позволяет эффективнее его использовать.

Эргономичность интерфейса:

* Простота использования: Интерфейс разработан с учетом принципов минимального числа кликов и легкости навигации, что делает его интуитивно понятным даже для новых пользователей;
* Логика и структура: Элементы интерфейса логически структурированы для удобства пользователя, облегчая навигацию и поиск нужных функций;
* Информативность: Интерфейс содержит достаточно информации для выполнения задач, но при этом избегает перегрузки лишними деталями, что упрощает восприятие информации пользователем;
* Цветовая схема и шрифты: Используется приятная для глаз цветовая гамма и четкие шрифты, что делает интерфейс приятным для длительного использования;
* Обратная связь: Пользователь получает надлежащую обратную связь от интерфейса о результатах своих действий, что помогает ему понять, что его запросы были успешно выполнены;
* Единообразие: Все элементы интерфейса оформлены единообразно, что облегчает ориентацию пользователя и повышает удобство использования.

Для удобства, простоты и эргономики, был использован стиль Material Design.

Страницы в приложении проектировались с учетом требований унификации:

* все страницы пользовательского интерфейса выполнены в едином графическом дизайне, с одинаковым расположением основных элементов управления и навигации;
* для обозначения похожих операций используются похожие по стилю элементы (кнопки, ползунки, текстовые поля).

Эргономичность интерфейса "ServiTech" обеспечивает удобство и эффективность работы пользователей с программой, что способствует повышению их удовлетворенности и продуктивности.

## 2.3 Нормализация модели данных

В современных условиях основным инструментом для разработки логической модели данных являются различные варианты ER-диаграмм (диаграммы сущность-связь).

ER-диаграмма представляет собой наглядное отображение взаимосвязей между объектами, что позволяет понять, как эти объекты взаимодействуют между собой и какие аспекты этого взаимодействия можно улучшить.

Эти диаграммы помогают разбивать сложные системы и базы данных на более простые элементы, что делает понимание функционирования системы более простым и наглядным. Они отображают, как соединяются и перекрываются различные объекты, что облегчает анализ и оптимизацию взаимосвязей в системе.

1. База данных включает в себя следующие таблицы:

* User (Пользователи) - таблица 1;
* Personal (Персональные данные) - таблица 2;
* Organization (Организация) - таблица 3;
* Order (Заявка) - таблица 4;
* Status (Статус) - таблица 5;
* File (Файл) - таблица 6;
* Type (Тип услуги) – Таблица 7;
* Token (Авторизационный токен) Таблица 8;
* Vapid (Ключи подписки) Таблица 9;
* Subscription (Объект подписки на push уведомления) Таблица 10;
* Keys (Ключи объекта подписки) Таблица 11.

1. Функциональные зависимости:

* От таблицы «User» исходит 1:М связь к таблице «Order», так как у одного пользователя может быть множество заявок;
* От таблицы «User» исходит 1:1 связь к таблице «Personal», так как у одного пользователя может быть только одна фамилия и другие персональные данные;
* От таблицы «User» исходит 1:1 связь к таблице «Vapid», так как у одного пользователя может быть только одна пара «Push» ключей.
* От таблицы «User» исходит связь 1:1 к таблице «Token», так как у одного пользователя может быть только одна пара авторизационных токенов;
* От таблицы «User» исходит связь 1:1 к таблице «Subscription», так как у одного пользователя может быть только один объект подписки;
* От таблицы «Subscription» исходит связь 1:1 к таблице «Keys», так как у одного объекта подписки может быть только одна пара ключей;
* От таблицы «Order» исходит связь 1:1 к таблице «Status», так как у одной заявки единовременно может быть только один статус;
* От таблицы «Order» исходит связь 1:1 к таблице «File», так как у одной заявки может быть единовременно только один файл;
* От таблицы «File» исходит связь 1:1 к таблице «Type», так как у файла может быть только один тип поддерживаемой услуги.

Процесс нормализации:

* Определение функциональных зависимостей между атрибутами таблиц;
* Проверка на соответствие первой нормальной форме (1НФ): каждый атрибут содержит только атомарные значения;
* Разделение таблиц на отдельные сущности для избегания избыточной информации и обеспечения целостности данных;
* Проверка на соответствие второй нормальной форме (2НФ): каждый не ключевой атрибут должен зависеть от ключа целиком, а не от его части.

1. При проведении нормализации были выполнены следующие изменения для нормализации:

* сохраняемые данные на пересечении строк и столбцов представляют скалярное значение, таблицы не содержат повторяющихся строк – эти изменения относятся к первой нормальной форме;
* каждый столбец, не являющийся ключом, зависит от первичного ключа – изменения для второй нормальной формы;
* каждый столбец, не являющийся ключом, зависит только от первичного ключа – изменения для третьей нормальной формы;

1. Для таблицы «User» (Пользователи) были включены следующие атрибуты:

* Id – первичный ключ;
* Email – электронная почта;
* passwordHash – захешированный пароль;
* isActivated – флаг, указывающий на тип аккаунта пользователя;
* activationLink – ссылка для активации аккаунта по почту;
* role – роль пользователя;
* personal – ссылка на модель «Personal» (Персональные данные);
* token – ссылка на модель «Token» (Авторизационный токен);
* order – ссылка на список моделей «Order» (Заявка);
* vapid – ссылка на модель «Vapid» (Ключи подписки);
* subscription – ссылка на модель «Subscription» (Объект подписки на push уведомления).

Для таблицы «Personal» (Персональные данные) были включены следующие атрибуты:

* name – имя пользователя;
* surname – фамилия пользователя;
* patronymic – отчество пользователя;
* phoneNumber – номер телефона пользователя;
* avatar – полное название изображения;
* userId – внешний ключ на таблицу «User» (Пользователи).

Для таблицы «Organization» (Организация) были включены следующие атрибуты:

* phoneNumber – номер телефона организации;
* email – электронная почта организации;
* address – адрес организации;
* avatar – полное название изображения;
* accNumber – номер банковского счета;
* description – описание организации.

Для таблицы «Token» (авторизационный токен) были включены следующие атрибуты:

* refreshToken – токен обновления;
* userId – внешний ключ на таблицу «User» (Пользователи).

Для таблицы «Order» (Заявка) были включены следующие атрибуты:

* id – первичный ключ;
* description – описание заявки;
* price – цена предоставляемой услуги;
* status – ссылка на модель «Status» (Статус);
* userId – внешний ключ на таблицу «User» (Пользователи);
* file – ссылка на модель «File» (Файл).

Для таблицы «Status» (Статус) были включены следующие атрибуты:

* status – уникальное значение статуса;
* message – сообщение для пользователя;
* orderId – внешний ключ на таблицу «Order» (Заявка).

Для таблицы «File» (Файл) были включены следующие атрибуты:

* path – полный путь к файлу на сервере;
* type – ссылка на модель «Type» (Тип услуги);
* orderId – внешний ключ на таблицу «Order» (Заявка).

Для таблицы «Type» (Тип услуги) были включены следующие атрибуты:

* fileId – внешний ключ на таблицу «File» (Файл);
* id – первичный ключ;
* name – имя типа;
* type – уникальное значение типа.

Для таблицы «Subscription» (Объект подписки на push уведомления) были включены следующие атрибуты:

* endpoint – метаданные о клиентской части приложения;
* expirationTime – время жизни подписки;
* userId – внешний ключ на таблицу «User» (Пользователи);
* keys – ссылка на модель «Keys» (Ключи объекта подписки).

Для таблицы «Keys» (Ключи объекта подписки) были включены следующие атрибуты:

* subscriptionId – внешний ключ на таблицу «Subscription» (Объект подписки на push уведомления);
* p256dh – открытый ключ;
* auth – секретный ключ аутентификации.

На рисунке 3 представлена нормализация проекта базы данных.

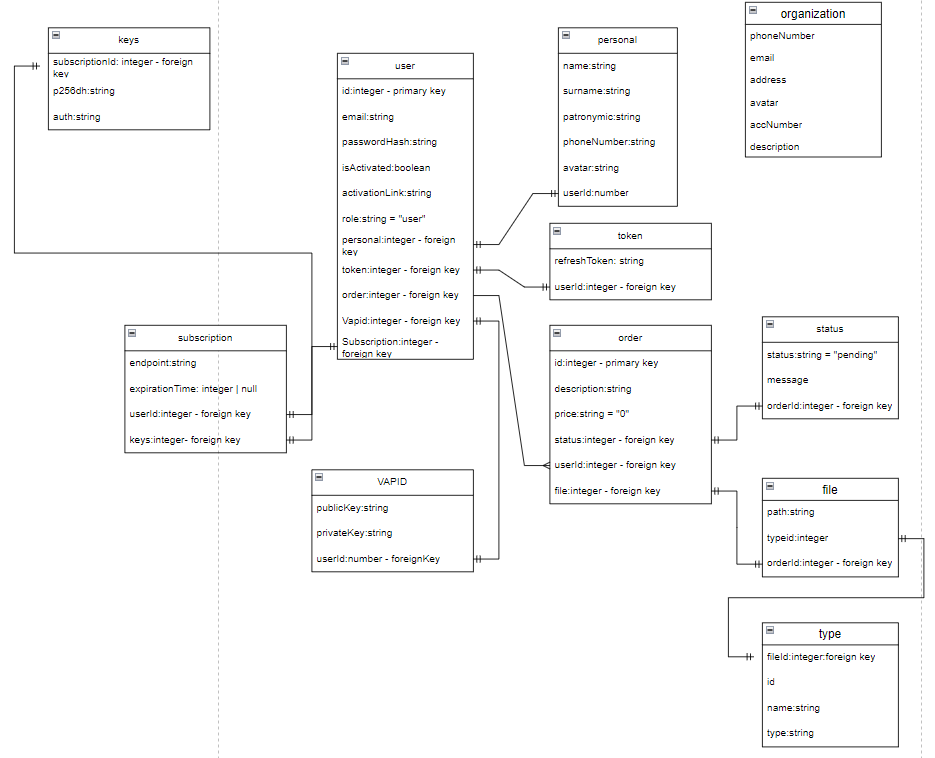


Рисунок 3 – Нормализация проекта базы данных

## 2.4 Физическая модель базы данных

Физическая модель данных описывает способы хранения информации в конкретной системе управления базами данных. На этапе разработки логической модели отношения между данными преобразуются в таблицы, где атрибуты становятся столбцами, а для ключевых атрибутов создаются уникальные индексы. Также осуществляется преобразование доменов в соответствующие типы данных, используемые в выбранной СУБД.

Физическая модель базы данных "ServiTech" спроектирована таким образом, чтобы обеспечить эффективное хранение и доступ к данным, а также масштабируемость и надежность работы программы.

В таблице 2 содержится информация о пользователях.

Таблица 2 – User (Пользователи)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| id | integer | - | + | Код |
| email | string | 40 | - | Электронная почта |
| passwordHash | string | - | - | Захешированный пароль |
| isActivated | boolean | - | - | Флаг на активацию аккаунта |
| activationLink | string | 35 | - | Ссылка на активацию |
| role | string | - | - | Роль |
| personal | integer | - | - | Персональные данные |
| token | integer | - | - | Токен |
| order | integer | - | - | Заявка |
| vapid | integer | - | - | Пара ключей |
| subscription | integer | - | - | Обьект подписки |

В таблице 3 содержится информация о персональных данных пользователей.

Таблица 3 – Personal (Персональные данные)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| name | String | 30 | - | Имя |
| surname | String | 30 | - | Фамилия |
| patronymic | String | 30 | - | Отчество |
| phoneNumber | String | 16 | - | Номер телефона |
| avatar | String | 27 | - | Фото профиля |
| userId | Integer | - | - | Код пользователя |

В таблице 4 содержится информация о оперативной памяти.

Таблица 4 – Token (Авторизационный токен)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| refreshToken | Счётчик | - | - | Токен обновления |
| userId | Числовой | - | - | Код пользователя |

В таблице 5 содержится информация о заявках.

Таблица 5 – Order (Заявка)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| id | Integer | - | + | Код |
| description | string | 100 | - | Описание |
| price | integer | - | - | Цена |
| status | string | 40 | - | Статус |
| userId | integer | - | - | Идентификатор пользователя |
| file | string | - | - | файл |

В таблице 6 содержится информация о статусе заявок.

Таблица 6 – Status (Статус)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| status | string | - | + | Статус |
| message | string | 52 | - | Сообщение |
| orderId | integer | - | - | Идентефикатор заявки |

В таблице 6 содержится информация о прикрепленных файлах.

Таблица 6 – File (Файл)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| path | string | - | + | Путь |
| typeId | integer | 52 | - | Код типа |
| orderId | integer | - | - | Код заявки |

В таблице 6 содержится информация о типах заявок.

Таблица 6 – Type (Тип заявки)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| fileId | integer | - | + | Код файла |
| name | string | 52 | - | Имя |
| type | string | - | - | тип |

В таблице 6 содержится информация о ключах «push» подписок.

Таблица 6 – Vapid (Ключи подписки)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| publicKey | text | - | + | Публичный  ключ |
| privateKey | text | 52 | - | Приватный  ключ |
| userId | integer | - | - | Код  пользователя |

В таблице 6 содержится информация о организации.

Таблица 6 – Organization (Организация)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| phoneNumber | string | - | + | Номер  Телефона |
| email | string | 52 | - | Электронная  Почта |
| address | string | - | - | Адрес |
| avatar | string | - | - | Изображение |
| accNumber | string | - | - | Номер  Счета |
| description | string | - | - | описание |

В таблице 6 содержится информация о статусе заявок.

Таблица 6 – Subscription (Подписка)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| endpoint | text | - | + | Контрольная  Точка |
| expirationTime | Text | null | 52 | - | Время жизни |
| userId | integer | - | - | Код  Пользователя |
| keys | text | - | - | ключи |

В таблице 6 содержится информация о статусе заявок.

Таблица 6 – Keys (Ключи подписки)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поя | Тип данных | Ограничения | Первичный ключ | Расшифровка |
| status | string | - | - | Статус |
| message | string | 52 | - | Сообщение |
| orderId | integer | - | - | Код  заявки |

## 2.5 Описание структуры входной/выходной информации

Для данного приложения, входной информацией является данные, предоставляемые пользователем для заполнения соответствующих полей в базе данных.

Для разрабатываемого программного продукта входной информация будет служить:

* Тип услуги;
* Описание заявки;
* Файл;
* Статус заявки;
* Цена услуги;
* Статус услуги.

Требования и спецификации:

* Функциональные требования, определяющие необходимый набор функций и возможностей программы;
* Технические требования, указывающие на используемые технологии и платформы.

Графические материалы:

* Логотипы и другие графические элементы, используемые для создания пользовательского интерфейса программы.

Технические данные:

* Метаданные о пользователях, версиях браузеров, авторизационных токенов и объектах подписок;

Входными данными также являются добавление, изменение и удаление данных из таблиц и карт, представленных в программе.

Так же были выполнены следующие требования:

* Дизайн был выполнен в спокойных и приятных для глаз тонах;
* Просмотр и функциональные возможности над таблицами базы данных.

Выходной информацией для программы является вывод отчета о проделанной работе для отправки в бухгалтерию, а так же для сбора статистики. Отчет в приложении представлен на рисунке 4, отчет, выведенный на печать, представлен на рисунке 5

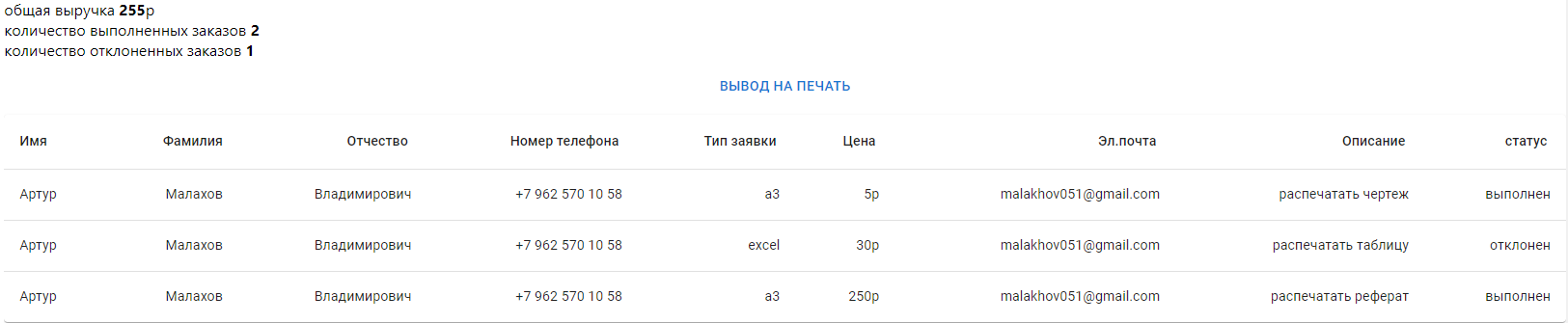


Рисунок 4 – Отчет о выполненной работе в приложении

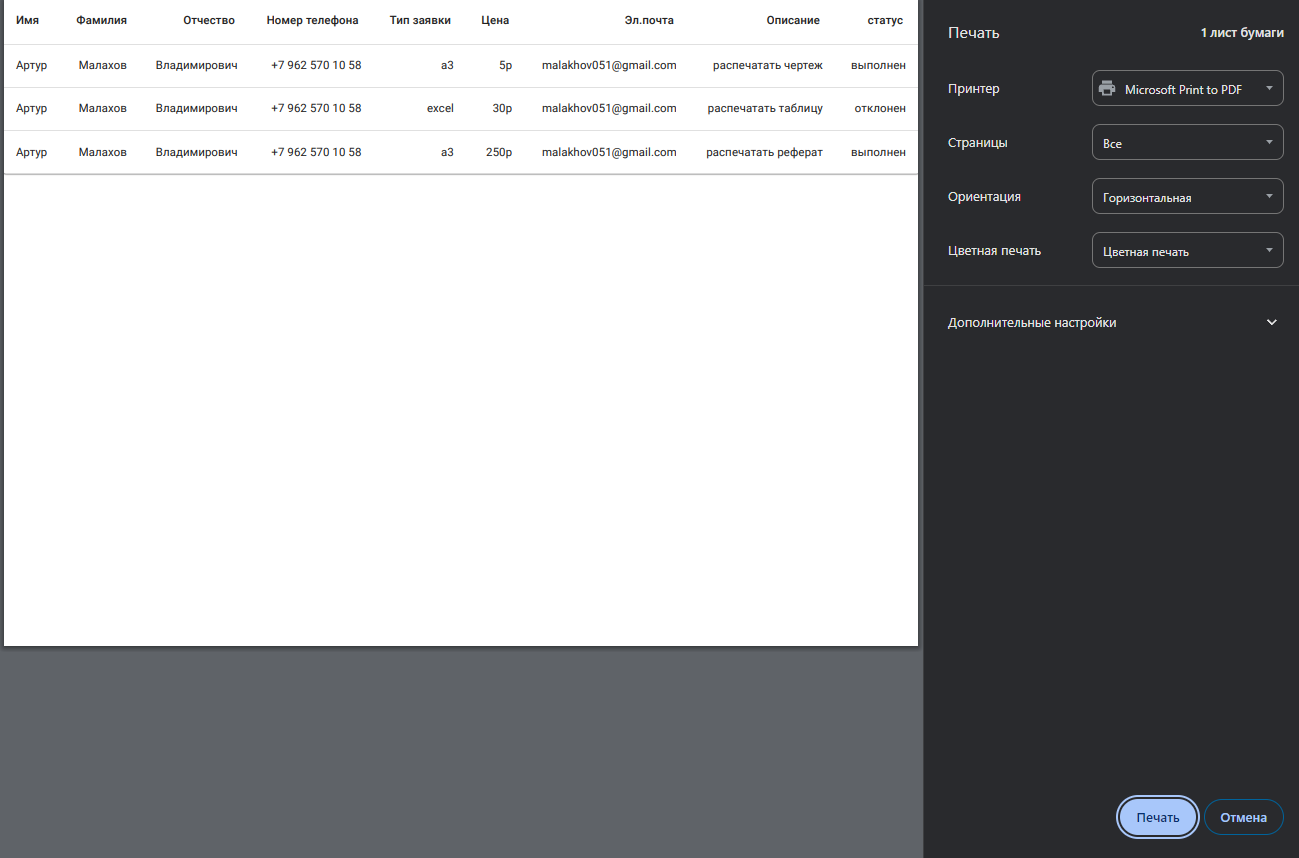


Рисунок 5 – Отчет о выполненной работе выведенный на печать

Также приложение выводит необходимые диалоговые окна:

* При успешном создании и редактировании объявления, программа выводит сообщения о том, что записи добавлены/изменены успешно.
* Если при добавлении и редактировании объявления пропущено поле, то программа выводит сообщение, о том, что не все поля заполнены.

## 2.6 Руководство пользователя

Система предоставления услуг в ИТ отделе предназначена для удаленного взаимодействия с клиентами, автоматизации обработки заявок.

Система реализует следующие основные группы функций:

* Предоставление информации о типах предоставляемых услуг;
* Предоставление информации о заявках;
* Предоставление информации о организации, занимающейся обработкой заявок;
* Предоставление информации о статусе заявки;
* Предоставление информации о цене заявки;
* Возможность добавления, изменения и удаления заявок;
* Возможность добавления, изменения и удаления типов услуг;
* Возможность вывода отчета по проделанной работе;

Пользователь программы должен обладать простыми практическими навыками работы с графическим пользовательским интерфейсом.

Для запуска приложения необходимо перейти по адресу: «http://servitech.ru». После чего пользователь попадает на страницу авторизации. Важно отметить, что для неавторизованных пользователей доступ запрещен по всем маршрутам, так что перед началом работы придется авторизоваться или зарегестрироваться (рисунок 6).

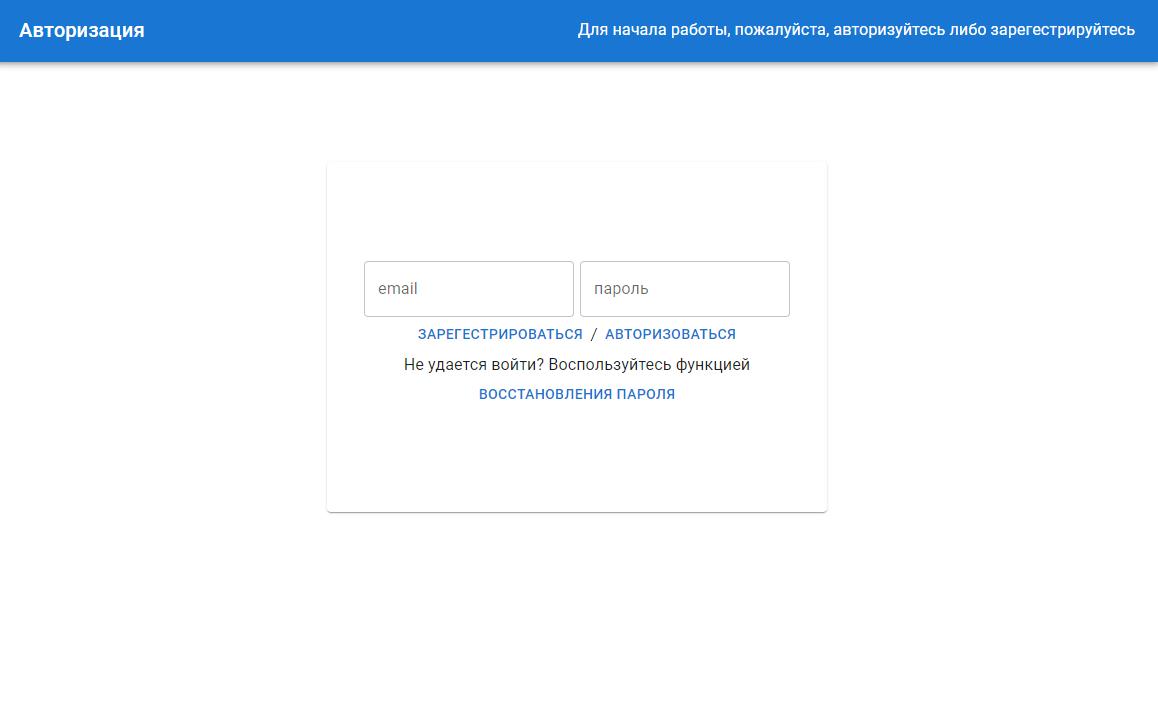


Рисунок 6 – Форма авторизации

После авторизации пользователя перенаправляет в его личный кабинет, где необходимо заполнить личные данные такие как фамилия и имя (рисунок 7).

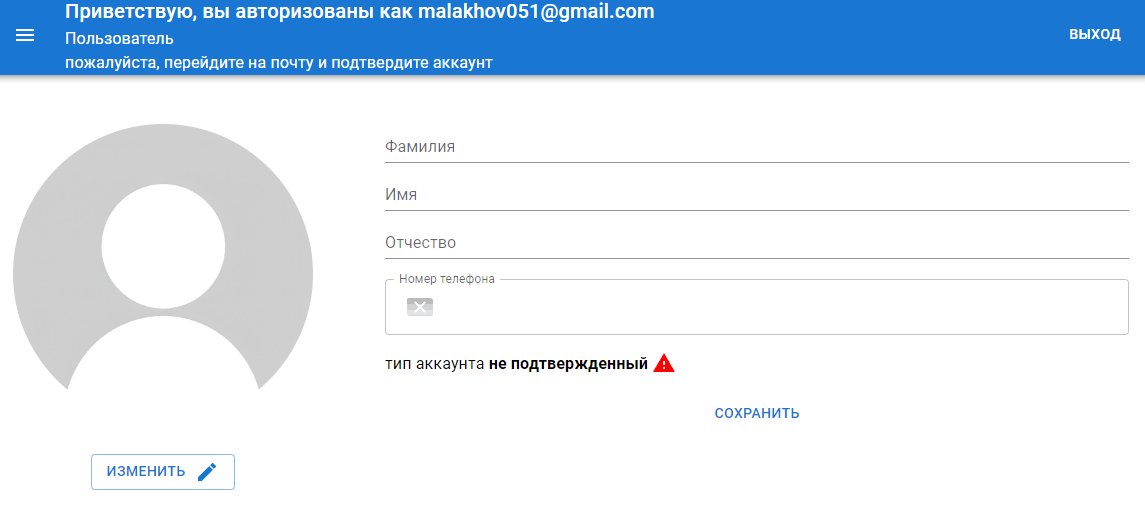


Рисунок 7 – Форма авторизации

Пользователи, не заполнившие личные данные и не подтвердившие учетную запись по почте, могут лишь просматривать информацию о организации и поддерживаемых услугах. Следовательно, помимо заполнения личных данных, необходимо перейти на указанную почту и подтвердить аккаунт (рисунок 8).

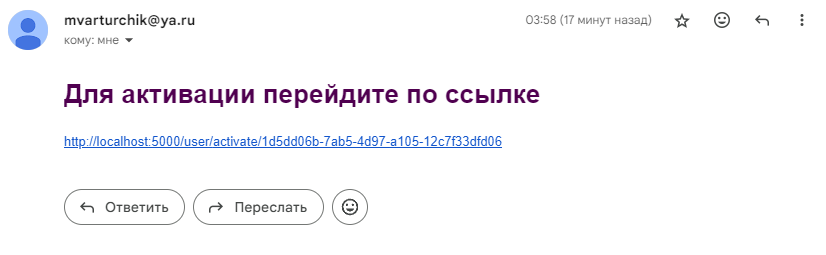


Рисунок 8 – Пример письма для активации аккаунта

После перехода по ссылке, пользователя перенаправляет обратно в приложение и тип аккаунта становиться подтвержденной, следовательно, теперь пользователю доступен весь функционал приложения (рисунок 9).

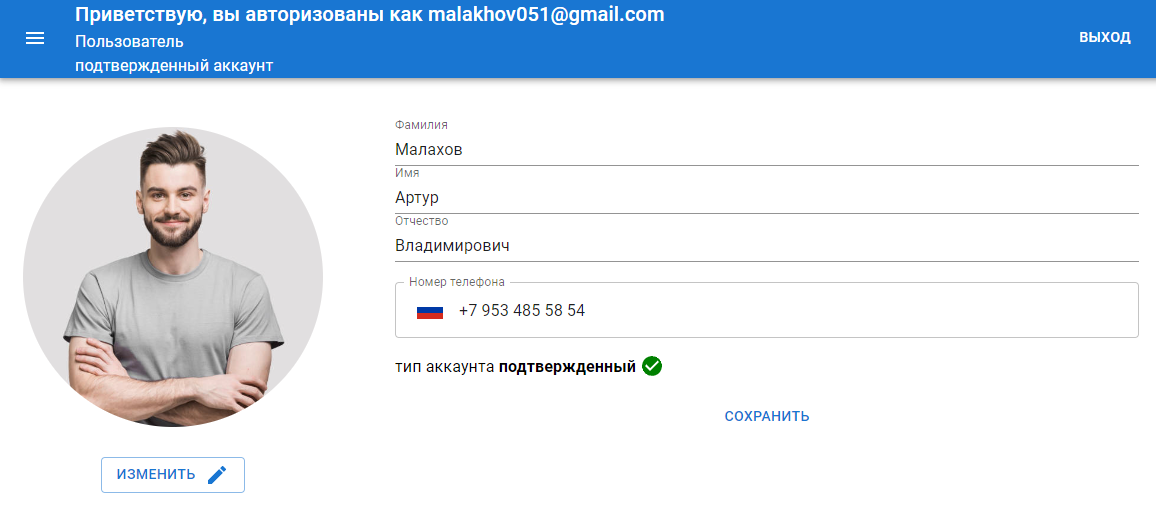


Рисунок 9 – Подтвержденный аккаунт

Следом пользователь может отправить заявку на обработку. Для этого необходимо перейти в раздел заявки (рисунок 10).

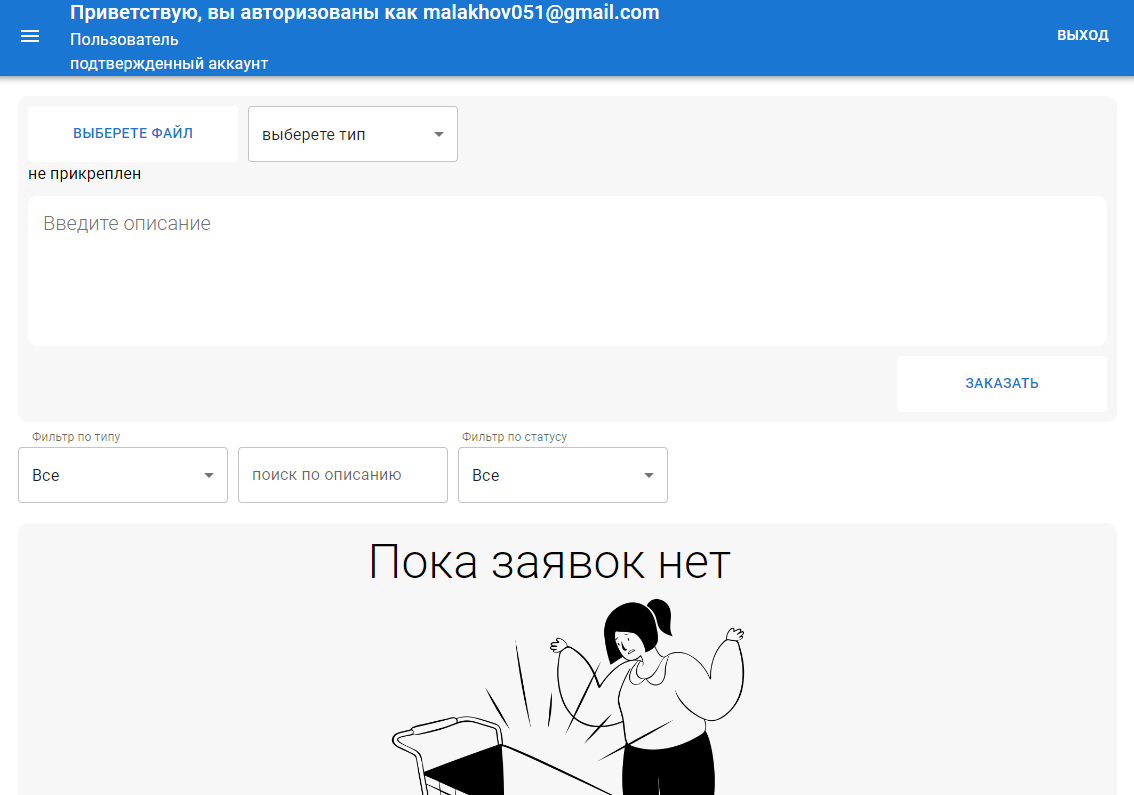


Рисунок 10 – Страница заявок.

После заполнения формы, новая заявка появится в списке ниже (рисунок 11), а также отправится в личный кабинет администратора.

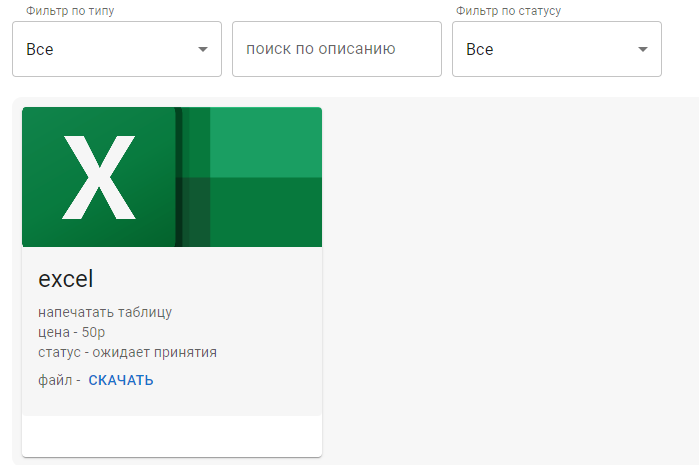


Рисунок 11 – Заявка со стороны клиента

После того, как заявка была отправлена на обработку, в рабочей области администратора для конкретного пользователя она отобразится (рисунок 12).

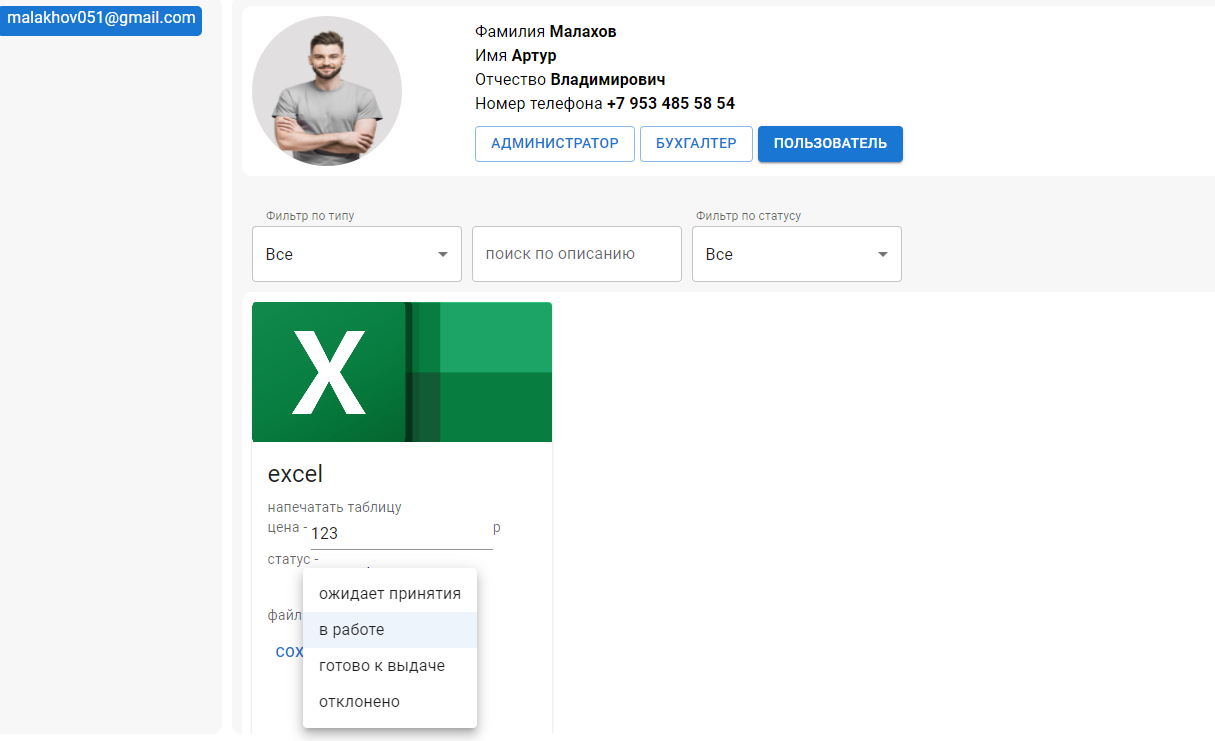


Рисунок 12 – Заявка со стороны администратора

Администратор может указать статус выполнения заявки, а также цену. После того, как цена и статус были указаны, администратору следует сохранить изменения, после этого со стороны клиента эти изменения отобразятся. Администратор так же может раздавать роли пользователям, для этого, в рабочей области под личными данными необходимо нажать на одну из кнопок, соответствующей роли (рисунок 13)

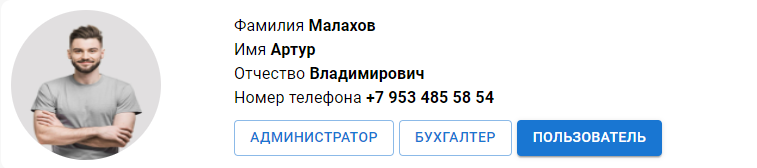


Рисунок 13 – Выдача ролей

Каждое действие с заявкой пользователя, будь то изменение статуса или изменение цены, сопровождается отправкой push уведомления клиенту о актуализации каких – либо данных (рисунок 14).

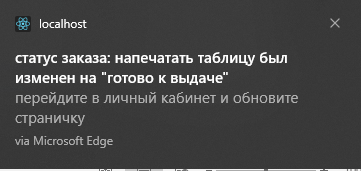


Рисунок 14 – Пример push уведомления

После изменения администратором статуса заявки на «в работе», пользователю следует перейти в раздел, о компании и перевести средства на указанный банковский счет (рисунок 15).

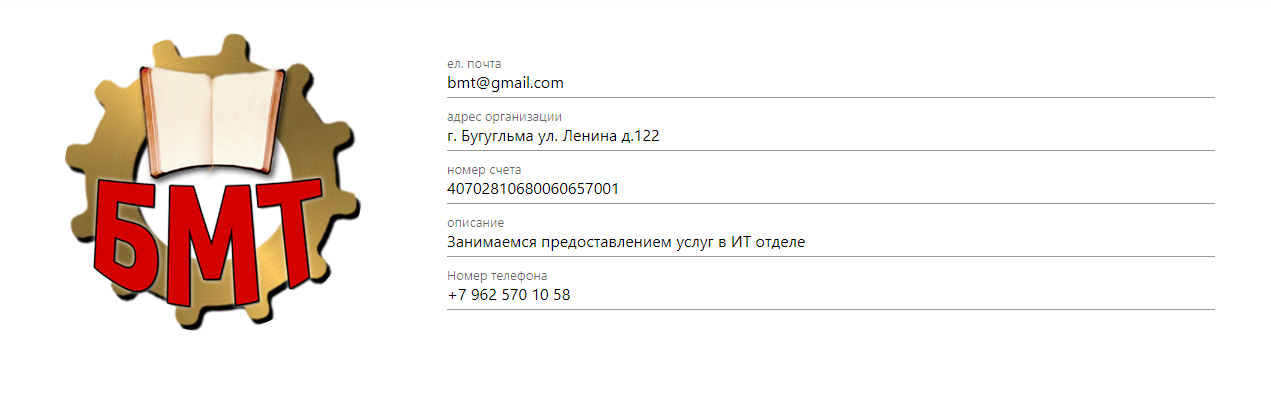


Рисунок 15 – Раздел «о компании»

После того, как статус заявки был изменен на «готово к выдаче», пользователь может забрать заказ по указанному в разделе «о компании» адресу.

По завершению работы, администратор может отправить отчет в бухгалтерию, для этого, в рабочей области следует нажать на кнопку «отправить отчет», после чего откроется модальное окно с просьбой подтвердить действие (рисунок 16).

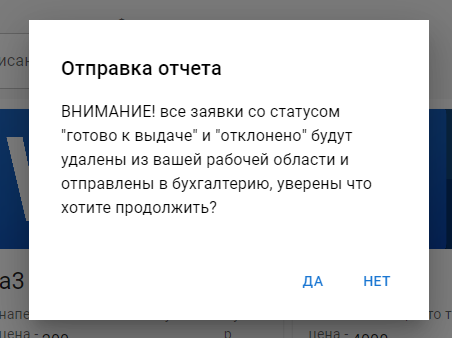


Рисунок 16 – Подтверждение операции

По нажатию на кнопку «да» все заявки с статусом «готово к выдаче» и «отклонено» будут отправлены в аккаунт с ролью «бухгалтерия».

Бухгалтер может изменять данные о организации, просматривать отчеты и выводить их на печать а так же просматривать некоторую статистику о количестве выполненных и отклоненных заказов и о общей выручке за день. Чтобы вывести отчет на печать, нужно выбрать соответствующий и нажать на кнопку «вывод на печать» (рисунок 17).



Рисунок 17 – Вывод отчетов

После чего его можно будет сохранить в PDF (рисунок 18) или распечатать на принтере (рисунок 19).

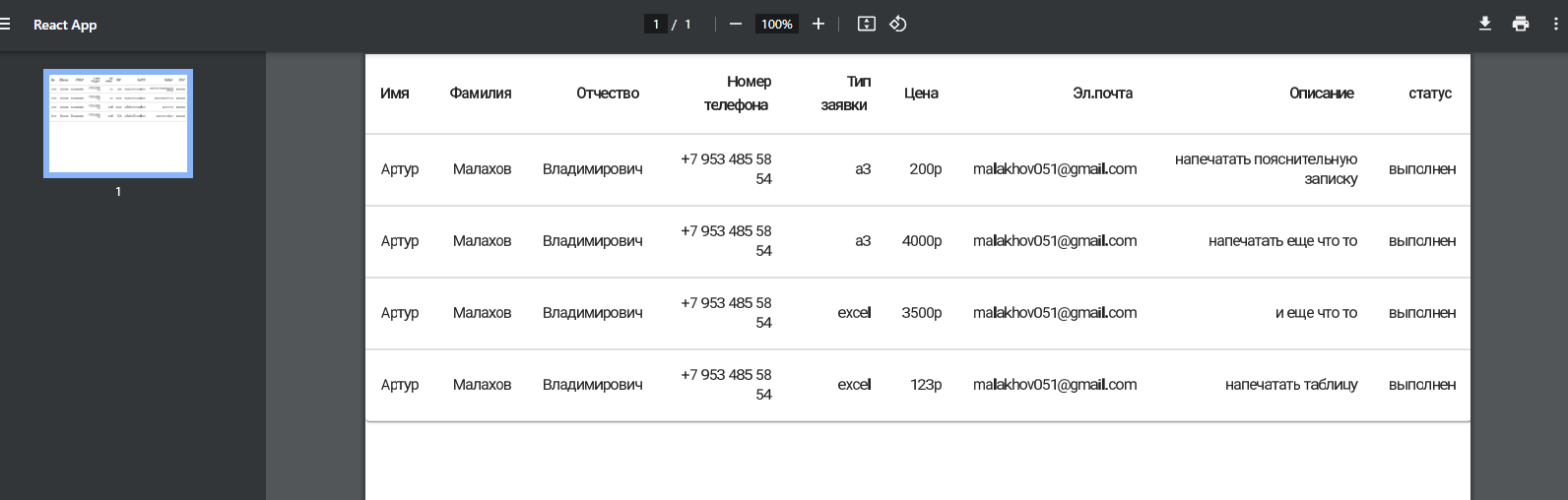


Рисунок 18 – Пример отчета в PDF формате

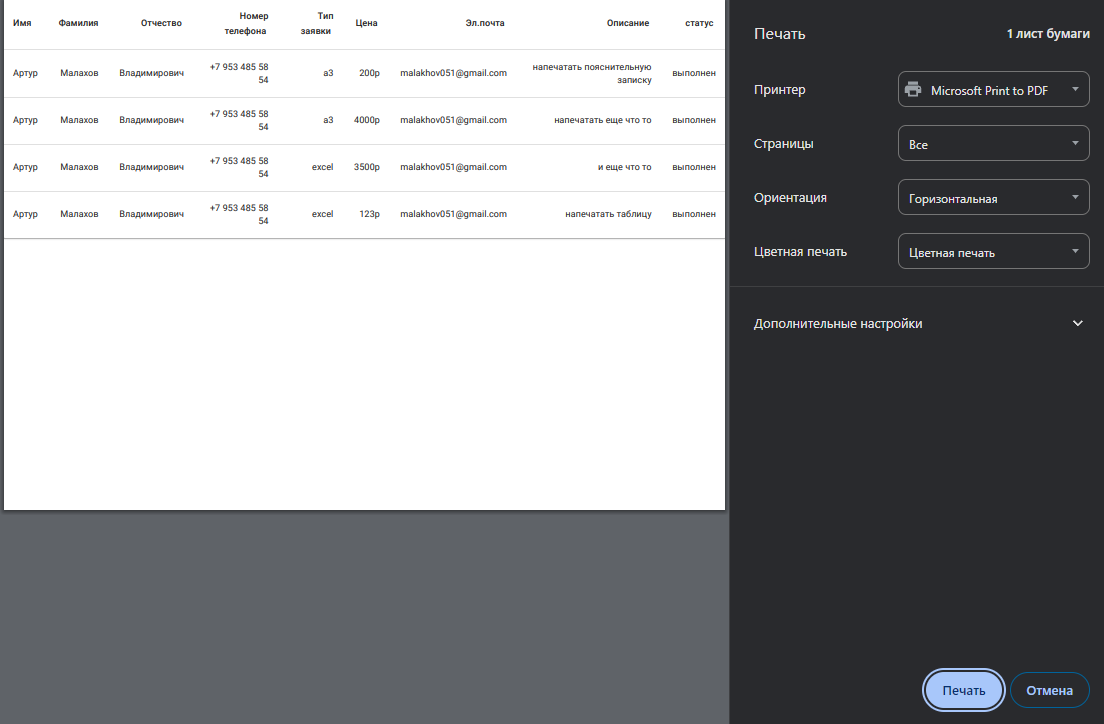


Рисунок 19 – Пример вывода отчета на печать

В случае некорректного ввода email адреса при регистрации выводится сообщение (Рисунок )



Рисунок – некорректный ввод email адреса

В случае некорректного ввода пароля при регистрации выводится сообщение (Рисунок )

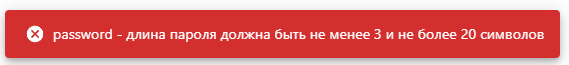


Рисунок – некорректный ввод пароля

В случае некорректного ввода авторизационных данных выводится сообщение (Рисунок )



Рисунок – некорректный ввод авторизационных данных

В случае некорректного имени выводится сообщение (Рисунок )

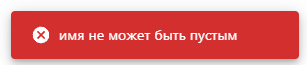


Рисунок – некорректный ввод персональных данных

В случае некорректного фамилии выводится сообщение (Рисунок )



Рисунок – некорректный ввод персональных данных

В случае успешной отправки отчета выводится следующее сообщение (Рисунок )

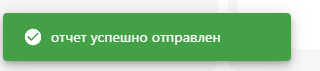


Рисунок – положительный результат выполнения запроса на отправку отчета

В случае, если аккаунт пользователя не подтвержден по почте, выводится следующее сообщение (Рисунок )



Рисунок – нет доступа

В случае, если при отправке заявки файл был не прикреплен, выводится следующее сообщение (Рисунок )

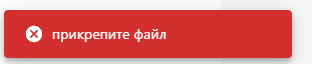


Рисунок – не корректный ввод данных

В случае некорректного ввода описания выводится следующее сообщение (Рисунок )



Рисунок – не корректный ввод данных при заполнении заявки

В случае успешной отправки заявки выводится следующее сообщение (Рисунок )



Рисунок – положительный результат выполнения запроса на отправку заявки

В случае корректного изменения данных выводится следующее сообщение (Рисунок )

## 2.7 Руководство программиста

В данном разделе содержатся общие сведения о программе, структуре, настройке, процесс установки программы.

Данным программным продуктом могут воспользоваться все пользователи, однако все функции станут доступными после авторизации.

Для работы приложения перед запуском необходимо установить следующие дополнительные компоненты:

* Браузер Chrome или любой другой на движке Chromium;
* Доступ в сеть интернет;
* Adobe PDF Reader.

Для внесения изменений в программный код и для настройки «.env» конфигурации необходимо установить:

* Node.js LTS version;
* Yarn package manager;
* postgeSQL v16;
* Visual Studio Code.

Перед началом установки node.js:

1. Проверить [требования к системе](https://learn.microsoft.com/ru-ru/visualstudio/releases/2022/system-requirements). Может ли компьютер поддерживать node.js.
2. Обновить Windows до последней версии. Эти обновления гарантируют, что на компьютере установлены последние обновления для системы безопасности и необходимые системные компоненты для node.js.
3. Перезагрузить компьютер. Перезагрузка гарантирует, что ожидающие установки или обновления компоненты не будут препятствовать установке node.js.

Скачать node.js можно перейдя на официальный сайт Node https://nodejs.org/en (рисунок 20).



Рисунок 20 – Сайт загрузки Node.js

Скачать postgres можно на официальном сайте EDB по адресу: <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads> (рисунок 21).

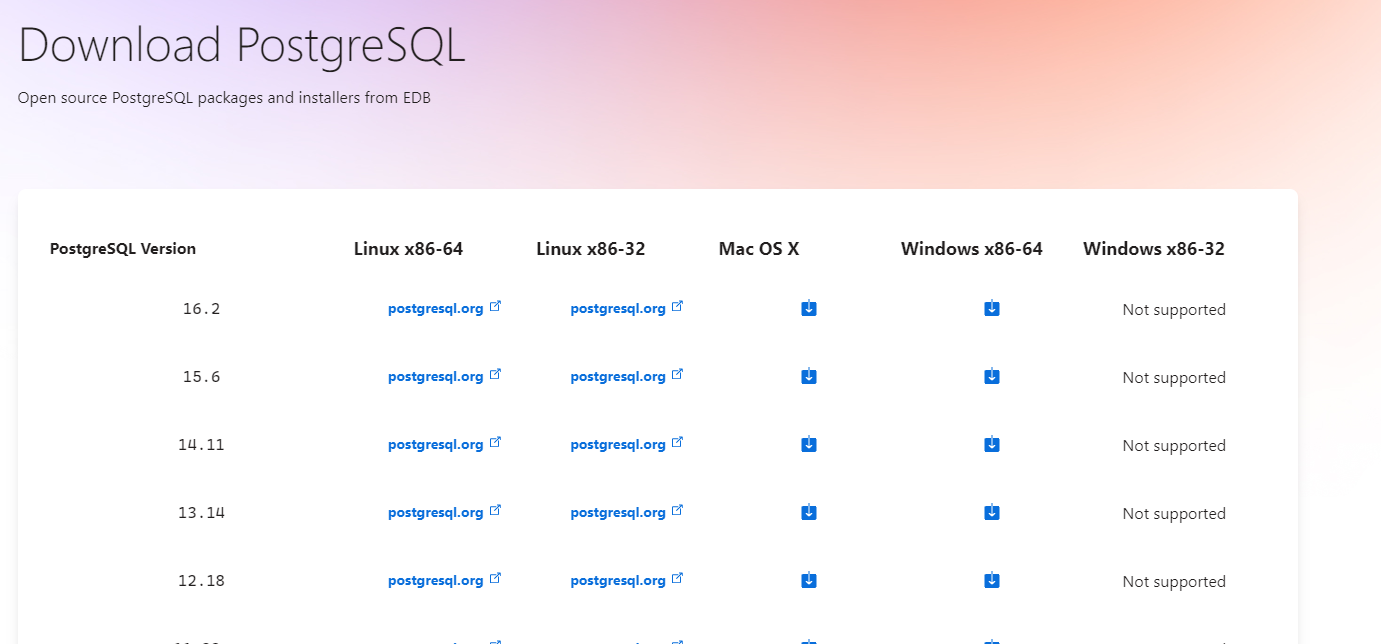


Рисунок 21 – Cайт загрузки postgres

Скачать Visual Studio Code можно на официальном сайте Microsoft по адресу: <https://code.visualstudio.com/> (рисунок 22).

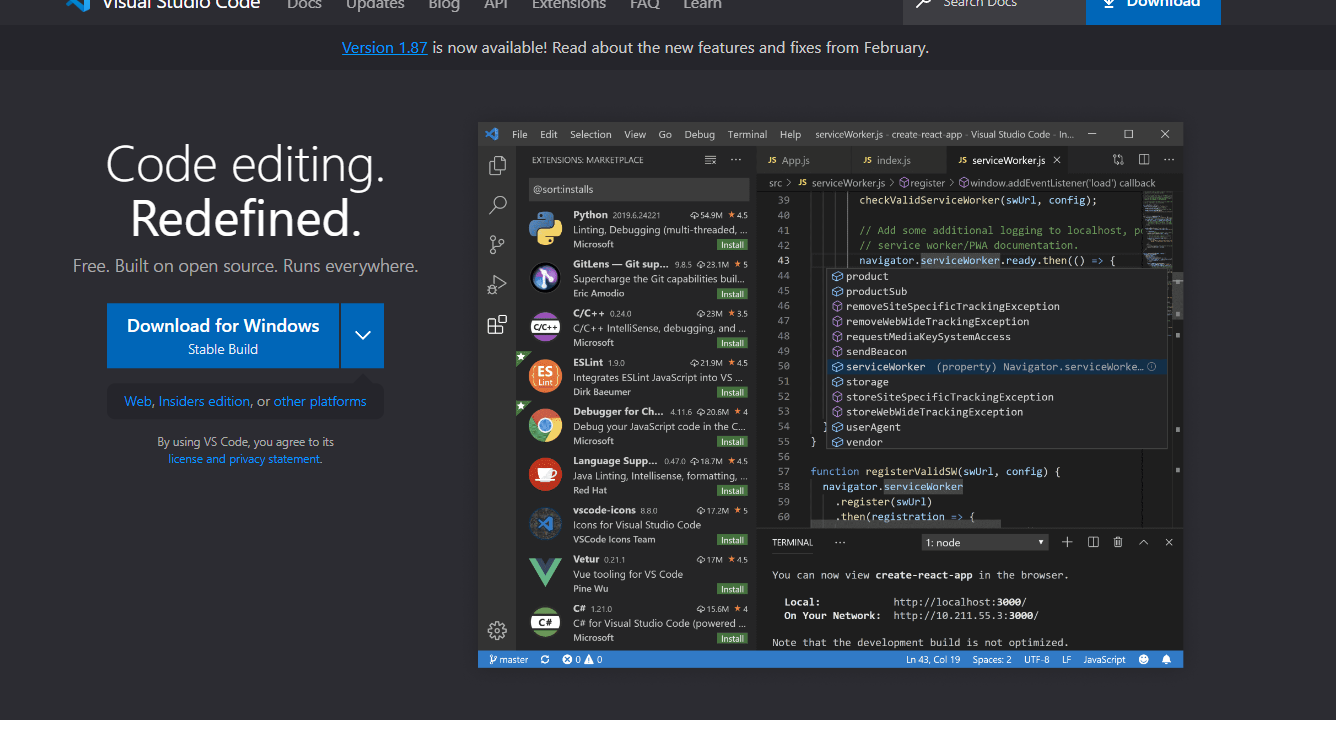


Рисунок 22 – Cайт загрузки Visual Studio Code

После установки всего необходимого программного обеспечения, запустить проект следует установить все зависимости командой «yarn». Далее можно запустить проект командой yarn start:dev (рисунок 23).

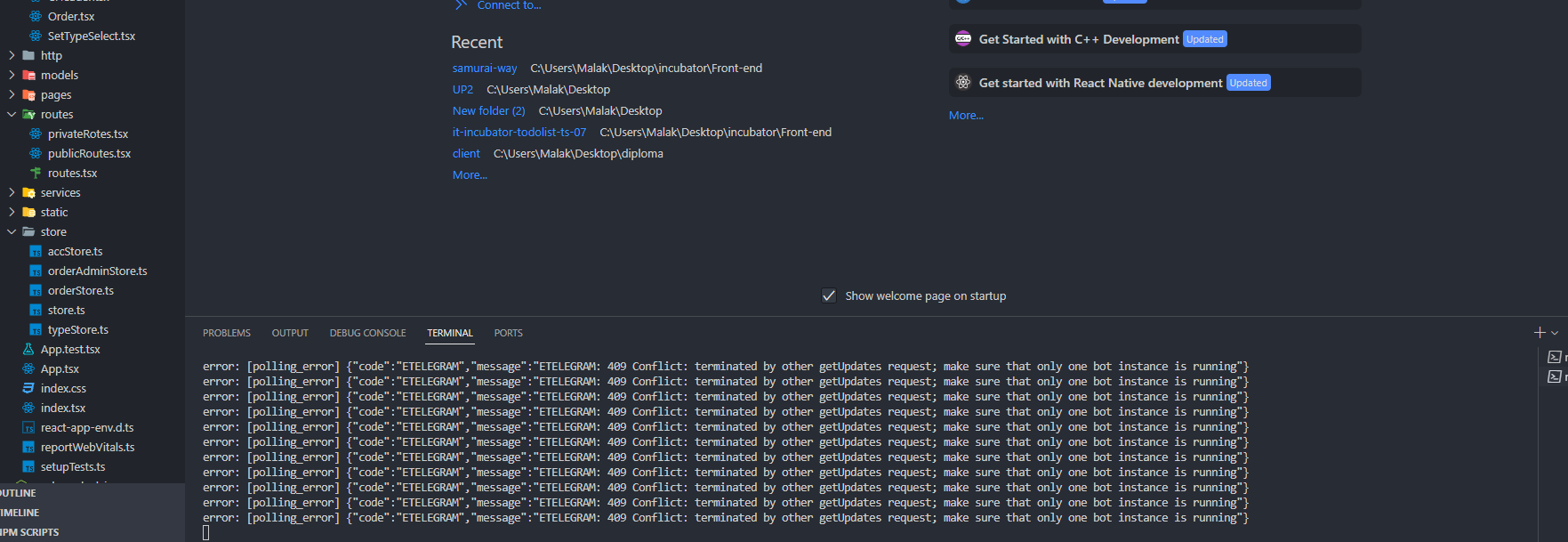


Рисунок 23 – Проект в Visual Studio Code

При размещении проекта на удаленном сервере необходимо изменить соответствующие поля в файле «.env» и «http.ts»

# Заключение

Система предоставления услуг в IT отделе "ServiTech" является мощным инструментом для пользователей, позволяющим легко и удобно удаленно заказывать выполнение определенных услуг информационного и производственного характера. В ходе разработки данного программного средства были выполнены следующие этапы:

* Были определены функциональные характеристики, включая возможности создания и редактирования заявок, а также управления их состоянием.
* Разработаны требования к техническим средствам, включая необходимость работы на различных операционных системах и совместимость с различными типами оборудования.
* Произведена нормализация модели данных и физическая модель базы данных, обеспечивая эффективное хранение и доступ к информации.
* Структура входной и выходной информации была определена, обеспечивая понимание пользовательских потребностей и функциональных возможностей программы.
* Проведено логическое и даталогическое проектирование, гарантируя оптимальную работу и удобство использования интерфейса программы.
* Разработано руководство пользователя и руководство программиста, обеспечивая удобство в установке, настройке и использовании программы, а также быстрое решение возможных проблем при разработке и эксплуатации.

# Список используемых источников

1. Албахари Д., Албахари Б. C# 6.0. Справочник. Полное описание языка [Текст]: учебное пособие / Д. Албахари, Б. Албахари – Вильямс – М., 2020 – 623 с.
2. Джепикс Ф., Троелсен Э. Язык программирования C# 7 и платформы .NET и .NET Core [Текст]: учебное пособие / Ф. Джепикс, Э. Троелсен – М.: И.Д. Вильямс, 2019 г. – 1328 с.
3. Дэвис А. Асинхронное программирование в C# 5.0 [Текст]: учебное пособие/ А. Дэвис – М.: ДМК Пресс, 2019. – 120 c.
4. Культин Н. MicrosoftVisual C# в задачах и примерах [Текст]: учебное пособие / Н. Культин – БХВ-Петербург - М., 2019. – 320 c.
5. РихтерД. CLRviaC#. Программирование на платформе Microsoft.NET Framework 4.5 на языке C# [Текст]: учебное пособие / Д. Рихтер – СПб.: Питер, 2020. – 896 c.
6. Скит Д. C# для профессионалов. Тонкости программирования [Текст]: учебное пособие / Д. Скит – М.: И.Д. Вильямс,2019г., – 608 с.
7. Стиллмен Э., Грин Дж. Изучаем C#. 3-е издание [Текст]: учебное пособие / Э. Стиллмен Дж. Грин – СПб.: Питер,2020г. – 816 с.
8. Тепляков С. Паттерны проектирования на платформе .NET[Текст]: учебное пособие / С. Тепляков – СПб.: Питер,2019г. – 320с.
9. Фримен А. ASP.NET Core MVC 2 с примерами на C# для профессионалов [Текст]: учебное пособие / А. Фримен – М.: Диалектика, 2019г. – 1008 с.
10. Шилдт Г. C# 4.0. Полное руководство [Текст]: учебное пособие / Г. Шилдт – М.: И.Д. Вильямс.,2019. – 291 c.

# Список нормативных документов

1. ГОСТ 19.101-77. Виды программ и программных документов. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 4 с.
2. ГОСТ 19.104-77. Основные надписи. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 7 с.
3. ГОСТ 19.105-77. Общие требования к программным документам. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 2 с.
4. ГОСТ 19.106-77. Требования к программным документам, выполненным печатным способом. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 3с.
5. ГОСТ 19.201-78. Техническое задание. Требования к содержанию и оформлению. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 3 с.
6. ГОСТ 19.202-78. Спецификация – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 3 с.
7. ГОСТ 19.301-79. Программа и методика испытаний. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 2 с.
8. ГОСТ 19.401-78. Текст программы. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 1 с.
9. ГОСТ 19.402-78. Описание программы. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 2 с.
10. ГОСТ 19.404-79. Пояснительная записка. Требования к содержанию и оформлению. – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 2 с.
11. ГОСТ 19.502-78. Описание применения. Требования к содержанию и оформлению – Введ. 01.01.80. – М.: Изд-во стандартов, 1987 . – 2 с.