**ГАПОУ «Казанский авиационно-технический колледж имени**

**П.В. Дементьева»**

|  |
| --- |
| **ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  **к курсовому проекту** |
| **КП 09.02.04.20И1.21.000** |
| (обозначение документа)  **МДК.02.04 Программные решения для бизнеса** |
| Тема: Разработка информационной системы на основе СУБД по теме:  «Конкурс творческих профессий» |

2023

Содержание

[Введение 4](#_Toc165286287)

[1 Анализ предметной области 6](#_Toc165286288)

[1.1 Описание области деятельности организации 6](#_Toc165286289)

[1.2 Анализ существующих аналогов ПО 6](#_Toc165286290)

[1.3 Определение требований к приложению. 8](#_Toc165286291)

[2 Проектирование приложения 10](#_Toc165286292)

[2.1. Определение сущностей и их характеристик 10](#_Toc165286293)

[2.2 Схема базы данных 11](#_Toc165286294)

[2.3 Словарь данных 11](#_Toc165286295)

[2.4 Сценарий работы приложения 12](#_Toc165286296)

[2.5 Дизайн интерфейса приложения 14](#_Toc165286297)

[2.6 Описание и формат входных данных. Надежность и достоверность данных 15](#_Toc165286298)

[2.7 Описание выходных данных. 16](#_Toc165286299)

[3. Разработка приложения 17](#_Toc165286300)

[3.1 Разработка БД 17](#_Toc165286301)

[3.2.1. Разработка стиля приложения 17](#_Toc165286302)

[3.2.2 Разработка подсистемы регистрации и авторизации 18](#_Toc165286303)

[3.2.3. Разработка подсистем приложения 19](#_Toc165286304)

[Тестирование 20](#_Toc165286305)

[4.1 Unit - тестирование 20](#_Toc165286306)

[4.2 Тестирование основных подсистем приложения. 20](#_Toc165286307)

[5. Инструкции по работе с приложением 23](#_Toc165286308)

[5.1 Руководство пользователя 23](#_Toc165286309)

[5.2. Руководство администратора БД 25](#_Toc165286310)

[Заключение 26](#_Toc165286311)

[Список литературы 27](#_Toc165286312)

[Приложения 28](#_Toc165286313)

[Приложение А 28](#_Toc165286314)

[Приложение Б 31](#_Toc165286315)

[Приложение В 34](#_Toc165286316)

[Приложение Г 37](#_Toc165286317)

[Приложение Д 41](#_Toc165286318)

[Приложение Ж 43](#_Toc165286319)

[Приложение З 46](#_Toc165286320)

[Приложение И 46](#_Toc165286321)

[Приложение К 48](#_Toc165286322)

# Введение

В эпоху информатизации, когда ремонт умных домофонов и выполнение гарантийных обязательств становятся критическими аспектами деятельности IT-компаний, задача эффективного управления и организации данных о сделках, клиентах и услугах приобретает особую значимость. Точная, структурированная и актуальная информация - залог гарантированного успеха и удовлетворения потребностей всех участников процесса ремонта умных домофонов.

Однако, IT-компании, специализирующиеся на производстве и ремонте умных домофонов, могут столкнуться с серьезными проблемами в области управления информацией, связанной с предоставлением этих услуг. Традиционные методы ведения данных и автоматизированные процессы могут привести к ошибкам, потере времени и, как результат, к недовольству клиентов.

Для решения указанных проблем и повышения эффективности работы отдела гарантии, целью курсовой работы является проектирование и разработка программного модуля «Отдел гарантий IT-компании» (АИС) для управления сделками по ремонту умного домофона. Данная система, является своего рода надстройкой над системой Битрикс 24 и станет мощным инструментом, способствующим улучшению процессов управления, минимизации ошибок и обеспечению высокого качества услуг.

Для достижения данной цели планируется выполнение следующих задач:

1. Анализ предметной области - детальное изучение и понимание специфики ремонта умных домофонов и особенностей гарантийного обслуживания.
2. Проектирование и разработка базы данных - создание структурированной и эффективной базы данных, являющейся основой для автоматизированной информационной системы.
3. Разработка настольного приложения - создание удобного и функционального приложения, которое будет интегрировано с базой данных и обеспечит удобный интерфейс для работы с информацией.
4. Тестирование
5. Разработка инструкций для работы с приложением

Важность данного проекта не может быть недооценена, учитывая растущую роль умных домофонов в повседневной жизни людей. Умные интеркомы

становятся неотъемлемой частью современных домов и офисов, обеспечивая безопасность и удобство для пользователей. Однако, как и любое сложное устройство, они подвержены поломкам и требуют профессионального обслуживания.

В свете этого, эффективное управление информацией о ремонте и гарантийном обслуживании умных домофонов становится ключевым фактором успеха для IT-компаний. Необходимость в автоматизированной информационной системе (АИС) для управления данными о сделках, клиентах и услугах становится очевидной.

АИС, разрабатываемая в рамках данного проекта, будет способствовать оптимизации рабочих процессов, уменьшению времени на обработку заявок и улучшению качества обслуживания клиентов. Она также поможет упростить процесс управления гарантийными обязательствами, уменьшив вероятность ошибок и улучшив управление статусами ремонта.

В дополнение к уже упомянутым задачам, в рамках проекта также планируется тестирование и внедрение системы. Это включает в себя проверку функциональности и производительности системы, а также ее адаптацию к конкретным условиям работы IT-компании.

В целом, данный проект имеет целью не только разработку АИС, но и внесение вклада в развитие отрасли ремонта умных домофонов, улучшение качества услуг и удовлетворенности клиентов.

1 Анализ предметной области

## Описание области деятельности организации

"ООО Спутник" является ведущим игроком на рынке информационных технологий, специализируясь на производстве, поддержке и восстановлении инновационных умных домофонов. Целью компании является интеграция и развитие передовых IT-решений, которые облегчают повседневную жизнь граждан, предоставляя им возможность осуществлять круглосуточный контроль и наблюдение за их жилищами.

"ООО Спутник" занимаемся производством умных домофонов различных классов, от доступных до премиум-сегмента, при этом накопив значительный опыт в работе с умными устройствами. Команда компании состоит из профессионалов в области IT, менеджмента, юридического сопровождения и технического обслуживания, каждый из которых вносит свой ценный вклад в создание готового уникального продукта.

Перед компанией стоят следующие задачи:

1. Улучшение качества наших продуктов: это включает в себя усовершенствование материалов корпуса, а также повышение качества комплектующих.
2. Развитие рекламной и маркетинговой стратегии: цель - привлечь к сотрудничеству крупные корпорации и индивидуальных потребителей.
3. Обеспечение качественного сервиса послепродажного обслуживания: компания предоставляет гарантийный ремонт, а также решает возникшие проблемы на расстоянии.

## Анализ существующих аналогов ПО

Среди существующих аналогов разрабатываемого приложения можно выделить такие платформы, как веб-сайт Битрикс-24, телеграмм-бот и мобильное приложение. Первые два варианта также позволяют изменять стадии сделки, однако это требует ввода номера панели вручную. В свою очередь, мобильная версия предлагает возможность сканирования NFC-метки панели с помощью смартфона, что значительно экономит время.

Для разработки приложения могут быть использованы различные платформы и технологии. В частности, можно использовать CRM-системы, которые предлагают широкий спектр инструментов для управления взаимоотношениями с клиентами. Также можно использовать технологии интеграции сервисов, такие как CSI (Crystal Service Integration).

Для создания коммерческого предложения могут быть выбраны различные ИТ-платформы и технологии. В частности, можно использовать платформы для создания диаграмм, такие как XMind, и технологии для моделирования процессов, такие как UML.

В процессе работы могут быть использованы различные технологии и методы. В частности, для управления проектами и задачами могут быть использованы методыуправления проектами, такие как Agile или Scrum.

В процессе моей работы я использую различные технологии и методы. Для управления проектами и задачами я могу использовать методы управления проектами, такие как Agile или Scrum. Также я могу использовать технологии для обработки данных, такие как Microsoft Office Excel. Кроме того, в моих средствах использования также присутствуют Visual Studio и Microsoft SQL Server для разработки и работы с базами данных.

Пример технических характеристик для приложения, разработанного для операционной системы Windows:

1. Операционная система: Приложение совместимо с Windows XP и более поздними версиями.
2. Процессор: процессор с частотой не менее 1 ГГц.
3. Оперативная память (RAM): Минимальное требование - 1 ГБ для 32-битной системы или 2 ГБ для 64-битной системы.
4. Жесткий диск: Минимальное свободное пространство на жестком диске - 4 ГБ
5. Видеокарта: DirectX 9 или более поздняя версия с драйвером WDDM 1.0.
6. Интернет-соединение: основные функции требуют подключение к интернету.

1.3 Определение требований к приложению.

Основная задача приложения - обеспечить возможность отслеживания и измени статуса ремонта домофонов. Пользователь вводит ID домофона в соответствующее поле и выбирает одну из четырех стадий: "Принята на склад", "Готово к отправке", "В ремонте", "В проверку". После этого отправляется WEB хук на удалённый сервер Битрикс-24 информация обновляется в системе, где заявка с указанным ID перемещается в соответствующую стадию или в случае когда заявка находится на стадии из которой нельзя перемещать возвращается сообщение с ошибкой. Все действия фиксируются в базе данных и логах.

Описание основного бизнес-процесса приложения:

1. Пользователь вводит ID домофона вручную.
2. Пользователь нажимает на одну из четырех кнопок со следующими названиями: "Принята на склад", "Готово к отправке", "В ремонте", "В проверку».
3. В систему Битрикс-24 отправляется WEB-хук для получения сделки на номер этой панели.
4. После запроса приходят данные в json формате и от туда извлекается номер последней сделки.
5. Формируется новый json, который отправляется в систему Битрикс-24, где заявка с указанным ID перемещается в соответствующую стадию.
6. При успешном выполнении команды пользователю выводится соответствующее сообщение.
7. При ошибке выполнения команды пользователю выводится сообщение об ошибке.
8. Все действия фиксируются в базе данных и логах.

Дополнительные функции приложения:

1. Просмотр истории изменений стадий
2. Фильтрация по дате и времени изменения
3. Фильтрация по сотруднику
4. Фильтрация по стадии

В будущем в приложении может быть добавлена возможность работать с заявкой: указывать выявленные дефекты и проведенные работы.

Определение пользователей и их функционал:

1. Оператор: Вводит ID домофона, выбирает стадию, просматривает сообщения об успешном выполнении команды или об ошибке управление логами.

Защита от несанкционированного доступа:

Доступ к приложению должен быть защищен с помощью аутентификации и авторизации. Пользователи должны вводить токен для доступа к функционалу приложения.

Диаграмма прецедентов:

Диаграмма прецедентов будет включать в себя следующие действия: для мастера ввод ID домофона, смена стадии сделки, просмотр логов, для администратора просмотр логов и удаление записей.

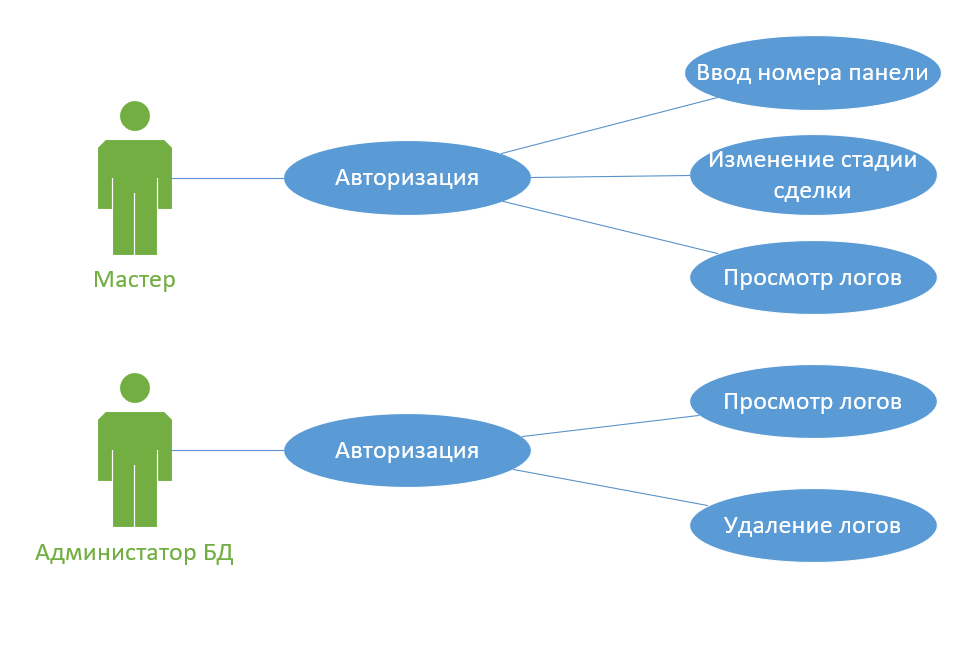


Рисунок 1 - Диаграмма прецедентов

2 Проектирование приложения

## 2.1. Определение сущностей и их характеристик

Были определены 2 сущности и также построена ER диаграмма

Сущности:

1. Сотрудник

Сотрудник будет иметь следующие поля:

* Идентификатор
* ФИО
* Уровень доступа
* Токен
* Должность

1. Логи

Таблица Логи будет содержать следующие поля:

- Идентификатор

- Номер панели

- ID сотрудника

- Начальную стадию

- Конечную стадию

- Дату создания

- Результат

- Описание

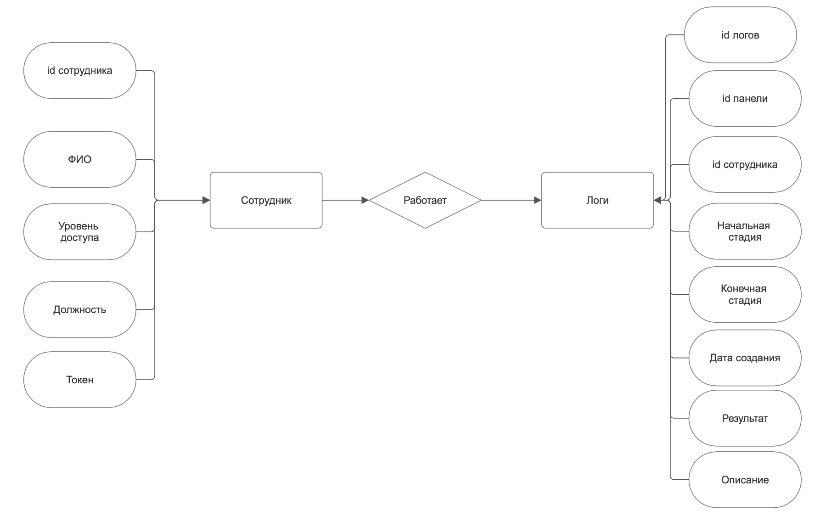


Рисунок 2 - ER модель

2.2 Схема базы данных

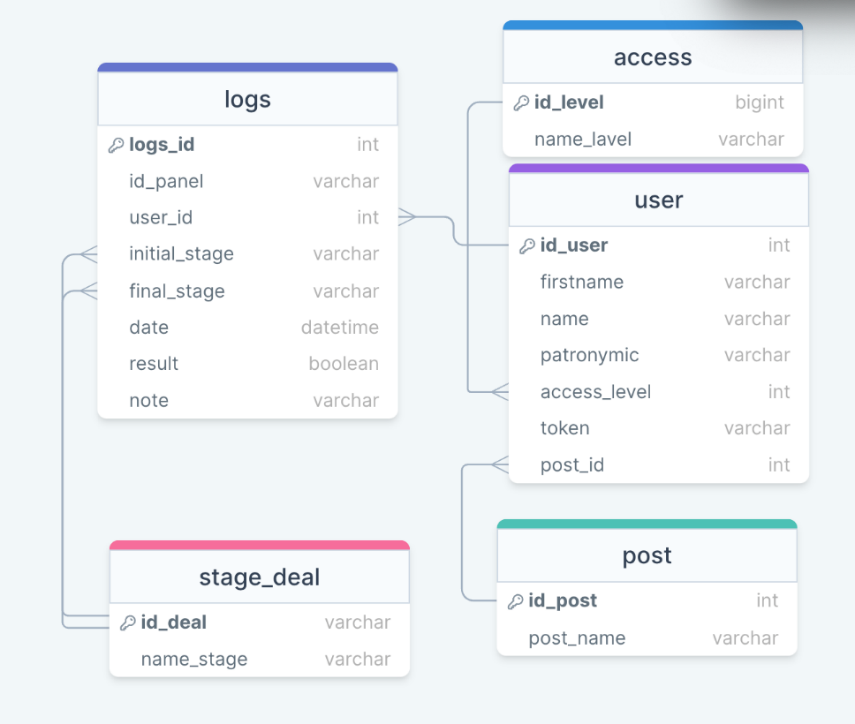


Рисунок 3 - Схема данных

## 2.3 Словарь данных

Таблица 1 - Словарь данных

|  |  |  |
| --- | --- | --- |
| logs | | |
| Поле | Тип данных | Примечание |
| logs\_id | int | PK |
| id\_panel | varchar(10) |  |
| user\_id | int | FK |
| initial\_stage | nvarchar(60) | FK |
| final\_stage | nvarchar(60) | FK |
| result | bit |  |
| note | nvarchar(MAX) |  |
| date | datetime |  |
| stage\_deal | | |
| Поле | Тип данных | Примечание |
| id\_deal | nvarchar(60) | FK |
| name\_stage | nvarchar(160) |  |
| access | | |
| Поле | Тип данных | Примечание |
| id\_level | int | PK |
| name\_lavel | nvarchar(60) |  |
| user | | |
| Поле | Тип данных | Примечание |
| id\_user | int | PK |
| firstname | nvarchar(100) |  |
| name | nvarchar(100) |  |
| patronymic | nvarchar(100) |  |
| access\_level | int | FK |
| token | nvarchar(200) |  |
| post\_id | int | FK |
| post | | |
| Поле | Тип данных | Примечание |
| id\_post | int | PK |
| post\_name | nvarchar(70) |  |

2.4 Сценарий работы приложения

1. Разработка алгоритма для изменения стадии сделки в системе Bitrix24 и ведения логов включает несколько важных шагов.
2. 1. Определение подсистем приложения: Приложение, которое разрабатывается для изменения стадии сделки и ведения логов, состоит из нескольких подсистем. Это подсистемы для отображения интерфейса пользователя, управления базой данных, управления WEB-хуками и ведения логов. Каждая из этих подсистем играет важную роль в работе приложения.
3. 2. Разработка алгоритма: Для успешной работы приложения необходимо разработать алгоритм, который будет определять последовательность действий для изменения стадии сделки и ведения логов. Этот алгоритм включает следующие шаги:
4. Получение последнего id сделки, которая заведена на введённый номер панели из системы Bitrix24.
5. Получение информации о текущей стадии сделки.
6. Изменение стадии сделки в соответствии с заданными правилами: переход на стадию «Принята на склад» если сделка находится в одном из следующих этапов: «Новая заявка» или «Проверка заявки» или «Заявка готова к отправке», в фазу «В ремонте» переходит если находится в стадии: «Принята на склад», «Отложено», «В ремонте», «Выходной контроль», «Готово к отправке», «Паркинг», в этап «Готов к отправке» можно перемещать если не находится в стадиях: «Заявка отменена», «Ошибка заведения», «Заявка завершена», в этап «В проверку» отправляется если на годится в фазе: «Принята на склад», «Отложено», «В ремонте», «Выходной контроль», «Готово к отправке», «Паркинг».
7. Запись действий в лог: Приложение ведёт логирование всех действий, выполняемых для изменения стадии сделки. Это помогает отслеживать историю изменений и обнаруживать возможные ошибки или проблемы.
8. Обновление информации о сделке в системе Bitrix24.

3. Разработка интерфейса: Важной частью разработки приложения является создание удобного и понятного интерфейса для пользователей. Интерфейс содержит такие элементы как: поле для ввода номера панели, выпадающие списки, всплывающий календарь, уточняющие пункты выбора для фильтрации истории взаимодействия.

2.5 Дизайн интерфейса приложения

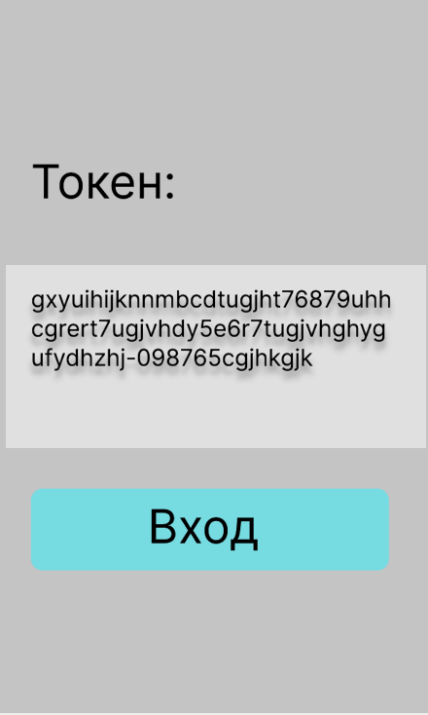


Рисунок 4 - Окно входа



Рисунок 5 - Главное окно

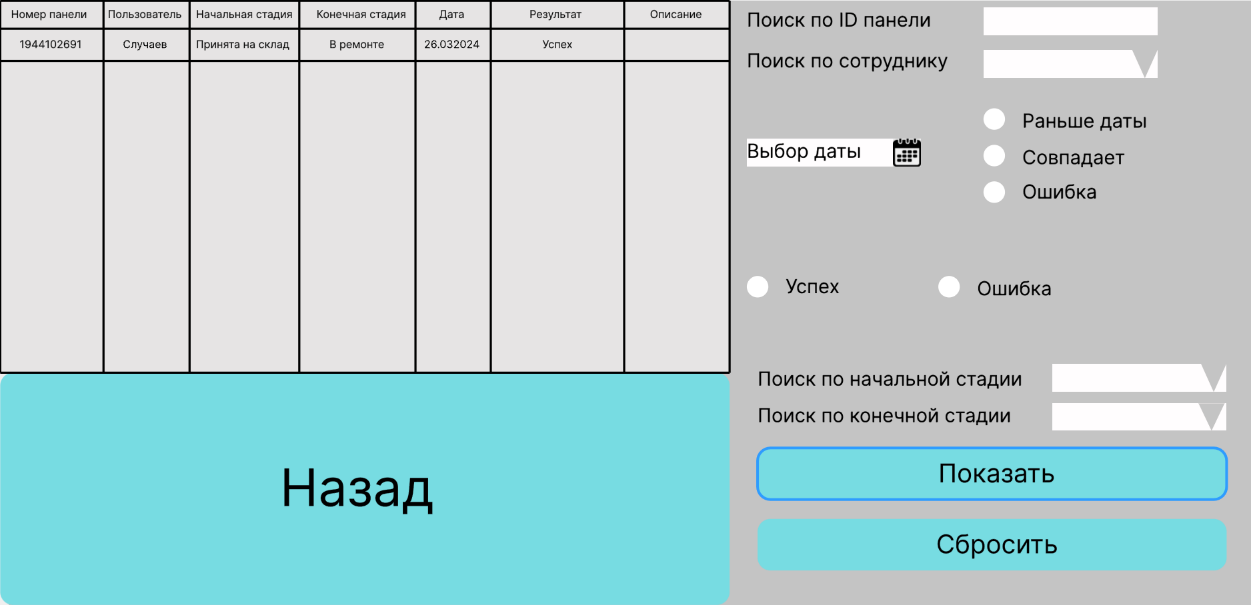


Рисунок 6 - Окно просмотра логов

## 2.6 Описание и формат входных данных. Надежность и достоверность данных

Для работы с моим приложением необходимо предоставить две составляющие входных данных.

Первая составляющая - это номер панели. Номер панели должен содержать 10 цифр без пробелов, букв и наличия специальных символов. Перед использованием номер панели проходит несколько шагов проверки. Сначала проверяется наличие символов в поле ввода, чтобы обеспечить гарантию того, что поле не осталось пустым. Затем проверяется длина номера панели, чтобы убедиться, что введено ровно 10 символов. После этого осуществляется проверка на наличие букв и пробелов в номере панели. Если в номере панели найдены буквы или пробелы, считается, что данные недостоверны и происходит вывод соответствующего сообщения.

Вторая составляющая - это файл в формате JSON, полученный от Битрикс-24 после отправки GET-запроса. Этот файл содержит дополнительные данные, необходимые приложению для выполнения задачи пользователя. Важно отметить, что достоверность и надежность данных, предоставленных пользователем после получения GET-запроса, уже были проверены и подтверждены системой Битрикс-24. Эта система считается автоматизированной и надежной, и поэтому шанс возникновения ошибки в предоставленных данных минимален.

## 2.7 Описание выходных данных.

Выходными данными работы приложения является json-файл, который отправляется в систему Битрикс-24. Этот файл содержит информацию, которую об изменяемом поле и новом значении, которое будет записано в соответствующее место.

# 3. Разработка приложения

## 3.1 Разработка БД

Открыл Microsoft SQL Server Management Studio (SSMS) и подключился к серверу баз данных, создал новую базу данных с именем courseproject. Затем, создал таблицу access с полями id\_level и name\_level, добавив столбцы id\_level (тип данных INT, PRIMARY KEY) и name\_level (тип данных NVARCHAR). После этого я создал таблицу logs с полями logs\_id, idpanel, userid, initialstage, finalstage, date, result, reason, добавив столбцы logs\_id (тип данных INT, PRIMARY KEY), id\_panel (тип данных INT, FOREIGN KEY связанный с id\_level в таблице access), user\_id (тип данных INT, FOREIGN KEY связанный с iduser в таблице user), initial\_stage (тип данных NVARCHAR), final\_stage (тип данных NVARCHAR), date (тип данных DATETIME), result (тип данных BIT), reason (тип данных NVARCHAR). Далее создал таблицу stage\_deal с полями id\_deal и name\_stage, добавив столбцы id\_deal (тип данных NVARCHAR, PRIMARY KEY) и name\_stage (тип данных NVARCHAR). После этого я создал таблицу user с полями id\_user, firstname, name, patronymic, access\_level, token, id\_post, добавив столбцы id\_user (тип данных INT, PRIMARY KEY), firstname (тип данных NVARCHAR), name (тип данных NVARCHAR), patronymic (тип данных NVARCHAR), access\_level (тип данных INT, FOREIGN KEY связанный с id\_level в таблице access), token (тип данных NVARCHAR), id\_post (тип данных INT, FOREIGN KEY). После этого я создал таблицу post с полями id\_post(тип данных INT, PRIMARY KEY),post\_name(тип данных NVARCHAR) Затем я связал таблицы между собой, установив соответствующие ключи FOREIGN KEY и связи, проверил работоспособность базы данных, выполнив несколько тестовых запросов и операций.

### 3.2.1. Разработка стиля приложения

Первое окно – это окно регистрации, состоит из текстового поля для ввода токена пользователя и кнопки для проверки соответствия введённых данных информации в БД.

Второе окно – это окно изменения стадии. Состоит из 6 кнопок и текстового поля ввода номера панели. Каждая кнопка имеет свою надпись, в которой заключён смысл её существования.

Третье окно – это окно просмотра логов, состоит из таблицы для отображения информации и различных элементов для поиска и фильтрации необходимых данных в БД.

В стиле приложения преобладают серые тона, на втором окне идёт секционная заливка цветом, уникальный цвет под каждый раздел. Во всех остальных случаях задний фон закрашен обычным серым, все кнопки имеют цвет морской волны, который гармонично вписывается в общий дизайн приложения.

### 3.2.2 Разработка подсистемы регистрации и авторизации

Окно регистрации в системе (Window2), представленное на рисунке 4, разработано в соответствии со стилем приложения приложение В. Для создания окна, необходимо нажать ПКМ по проекту, выбрать "Добавить", "Окно" (см. рисунок 5). В всплывающем окне снизу указать название окна, нажать "Добавить" В файл Window2.xaml, расположенный в обозревателе решений, необходимо скопировать xaml код из приложения Б, а в Window2.xaml.cs C# код.

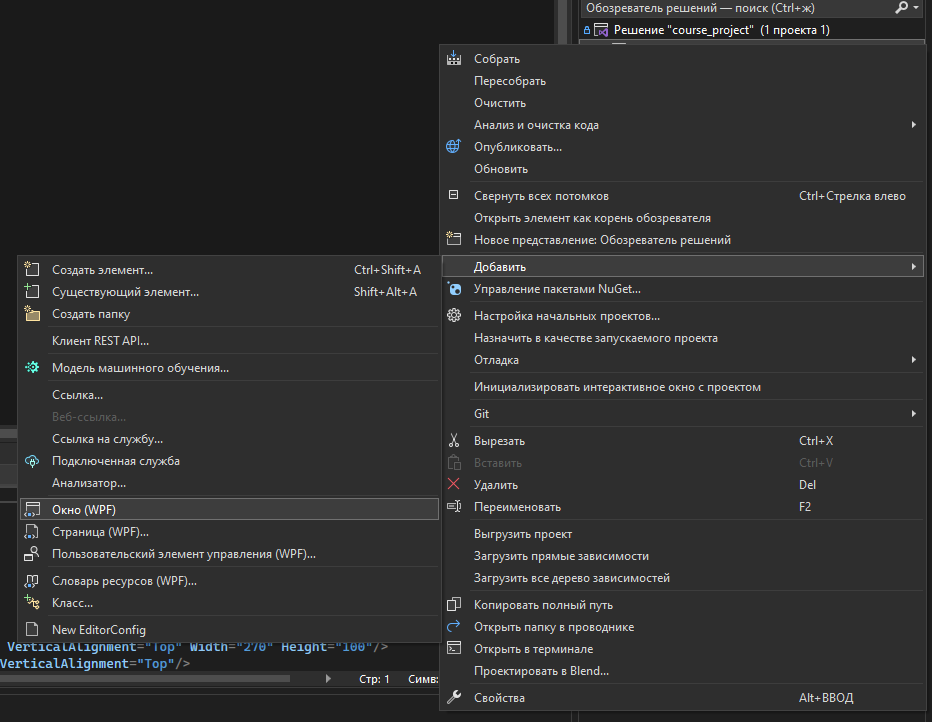


Рисунок 7 - Добавление окна регистрации

3.2.3. Разработка подсистем приложения

Пространство предназначенное для работы пользователя это главное окно и окно просмотра логов, были созданы согласно стилю приложения, содержат элементы, указанные в пункте 3.2.1. При разработке были добавлены таким же образом что и окно авторизации.

# Тестирование

## 4.1 Unit - тестирование

Для оценки корректности работы написанных методов были созданы следующие Unit – тесты для методов проверки длины введённого номера панели, наличия в строке недопустимых символов, правильно введённого номера панели, для метода поиска последней созданной сделки на номер домофона и 4 функций смены стадии сделки.

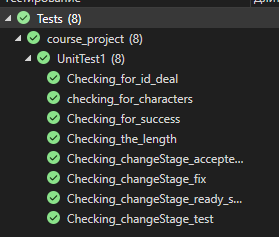


Рисунок 8 – Результат прохождения тестов

## 4.2 Тестирование основных подсистем приложения.

Система входа:

При вводе верного токена сотрудника и достаточного уровня доступа пользователя будет осуществляться переход на главною форму. Если сотрудник введёт не верный токен, то выйдет ошибка с сообщением, что токен является не верным(рисунок 9) .

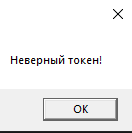


Рисунок 9 - Сообщение о неверном токене

При верном токене, не достаточном количестве прав у пользователя также будет ошибка, сообщение которой сообщает о недостаточном количестве прав(рисунок 10).

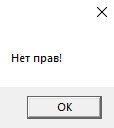


Рисунок 10 - Сообщение об отсутствии прав

Система изменения стадии сделки:

При попытке ввода некорректного номера панели выйдут сообщения с информацией, что не так с вводимой информацией(Рисунки 11, 12)

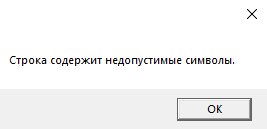


Рисунок 11 - Сообщение о недопустимых символах

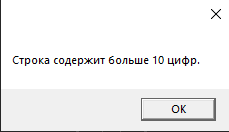


Рисунок 12 - Сообщение о несоответствии количества символов

При попытке изменить стадию сделки на недопустимую в данный момент также будет ошибка (Рисунок 13)

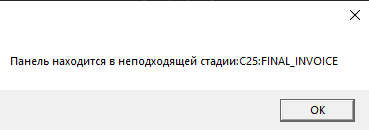


Рисунок 13 - Сообщение о неподходящей стадии

Система просмотров логов:

В данном окне возможно возникновение только одной ошибки с сообщением, что невозможно подключиться к БД(Рисунок 14).

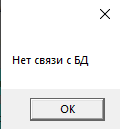


Рисунок 14 - Сообщение об отсутствии связи с БД

# 5. Инструкции по работе с приложением

## 5.1 Руководство пользователя

Данная инструкция предназначена для сотрудников компании Спутник, а именно участников отдела гарантии, к работе с созданной ИС допускаются люди, которые прошли инструктаж.

При запуске приложения пользователя встречает окно авторизации (рисунок 15), в соответствующее поле необходимо ввести свой уникальный токен сотрудника, при его корректности будет переход на следующее окно приложения, в противном случае выйдет сообщение с описание ошибки. После входа приложение запомнит пользователя и дальнейший запуск не потребует авторизации.

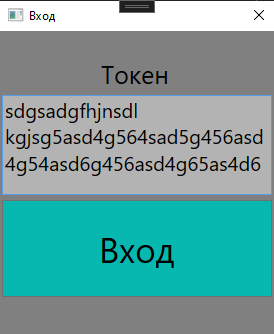


Рисунок 15 - Инструкция к окну входа

После авторизации пользователя встретит главное окно приложения (рисунок 16) в нём необходимо ввести ID панели, чей статус, в системе Битрикс-24, пользователь хочет изменить, каждая кнопка имеет своё название и перемещает сделку в соответствующую стадию, важно помнить порядок прохождения панелей этапов гарантийного обслуживания, при попытке переместить сделку в несоответствующую стадию, выйдет соответствующее, при прочих ошибках необходимо проверить наличие интернета и перезагрузить приложение. Кнопка «Log Out» предназначена для выхода из учётной записи.

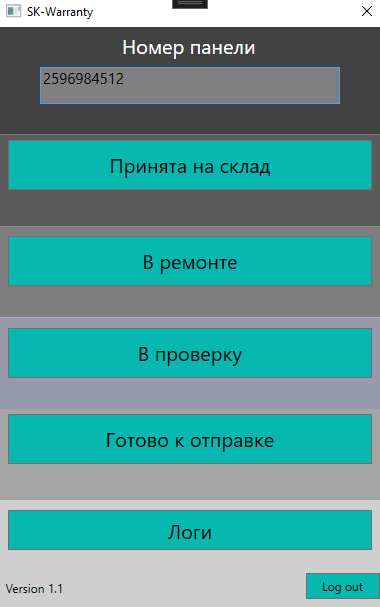


Рисунок 16 – Инструкция к главному окну

Окно просмотра логов содержит заметно больше элементов (рисунок 17), левый верхний угол занимает таблица и историей работы приложения, правее располагаются средства фильтрации данных, работа всех элементов интуитивно понятна, но фильтрация по дате может вызвать проблемы, сперва происходит выбор даты и дальше выбор идёт выбор трёх возможных вариантов: «Раньше даты», «Совпадает», «Позднее» выбор одного из 3 пунктов будет определять какие изменения будут отображены после применения фильтров.

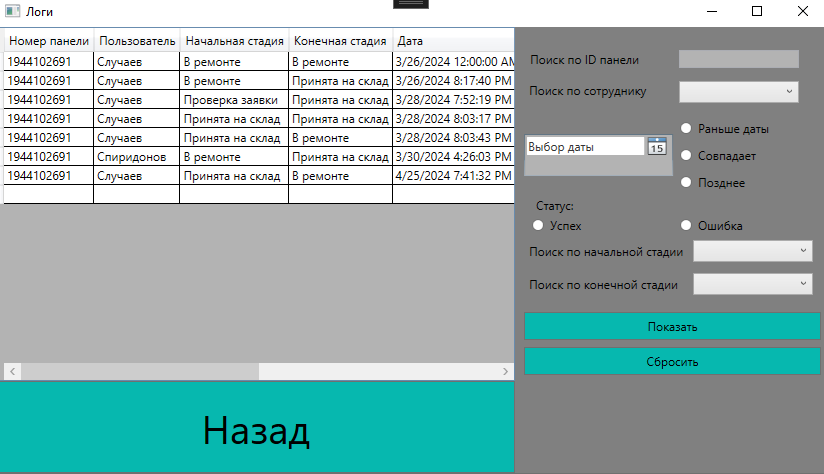


Рисунок 17 – Инструкция к окну логов

## 5.2. Руководство администратора БД

Данное руководство предназначено для пользователя с уровнем доступа администратор БД. При вводе токена, такого пользователя встретит окно с возможностью управления логами в БД включая возможность их удаления.

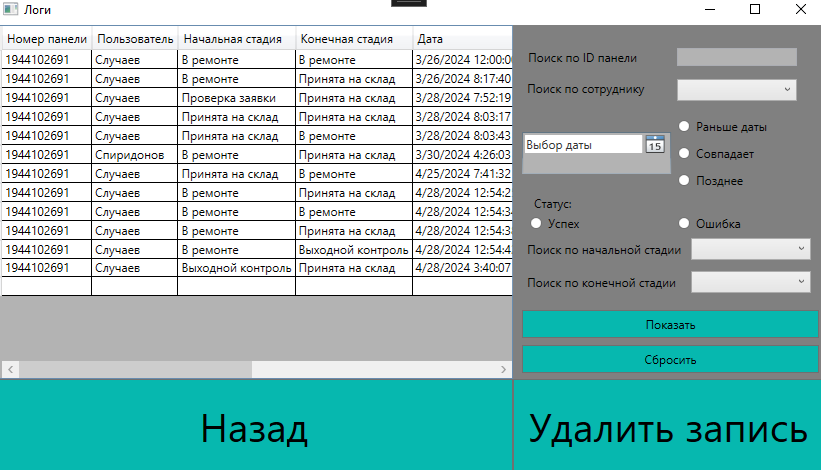


Рисунок 18 - Инструкция для администратора БД

Заключение

Результатом данного курсового проекта является АИС, предназначенная для управления отделом гарантии в IT компании. Во время работы был проведен анализ предметной области, проектирование и разработка базы данных, разработка приложения и руководства пользователя.

Разработанная система предоставляет удобное управление сделками на ремонт домофонов и просмотр истории их изменения, а гарантирует безопасность данных.

АИС имеет ряд преимуществ, среди которых:

* Простота добавления новых элементов
* Интуитивно понятный интерфейс
* Безопасность данных
* Возможность автоматизированной работы без связи с БД

В перспективе можно добавить возможность работать с заявкой: указывать выявленные дефекты и проведенные работы.

Следовательно, данная работа и в действительности обладает значительной ценностью, программный продукт может быть внедрен в деятельность организации и использоваться для решения вышеупомянутых задач.

# Список литературы

1. Билл Гейтс. Искусство программирования на C#. - Москва: Вильямс, 2021. - 800 с. (Дата обращения: 22.03.2024)
2. Эндрю Хант, Дэвид Томас. Разработка ПО с использованием WPF. - Санкт-Петербург: Питер, 2022. - 896 с. (Дата обращения: 21.04.2024)
3. Стив Макконнелл. Чистый код и эффективное программирование. - Москва: ДМК Пресс, 2020. - 650 с. (Дата обращения: 10.04.2024)
4. Максим Шаров. Паттерны проектирования в .NET. - Санкт-Петербург: Питер, 2022. - 320 с. (Дата обращения: 04.04.2024)
5. Coursera: [Электронный ресурс] Курс "Базы данных и SQL" на Coursera. URL: https://www.coursera.org/learn/sql-for-web-development (Дата обращения: 01.04.2024)
6. Основы работы с базами данных на платформе GeekBrains. - URL: https://geekbrains.ru/courses/68 (Дата обращения: 20.04.2024)
7. MozhnoEge: [Электронный ресурс] Информационные системы и базы данных. URL: https://mozhnoege.ru/informatika/informatsionnye-sistemy-i-bazy-dannyh (Дата обращения: 29.02.2024)
8. GeekBrains: [Электронный ресурс] Основы работы с базами данных. URL: https://geekbrains.ru/courses/68 (Дата обращения: 29.01.2024)
9. W3Schools: [Электронный ресурс] Руководство по SQL. URL: https://www.w3schools.com/sql/ (Дата обращения: 02.03.2024)
10. Docs.Microsoft: [Электронный ресурс] Документация по Windows Presentation Foundation (WPF). URL: https://docs.microsoft.com/en-us/dotnet/desktop/wpf/?view=netdesktop-7.0 (Дата обращения: 15.02.2024)

# Приложения

Приложение А

Скрипт создания БД

CREATE DATABASE [course\_project]

GO

USE [course\_project]

GO

/\*\*\*\*\*\* Object: Table [dbo].[access] Script Date: 16.04.2024 20:45:35 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[access](

[id\_level] [int] IDENTITY(1,1) NOT NULL,

[name\_level] [nvarchar](25) NOT NULL,

CONSTRAINT [PK\_access] PRIMARY KEY CLUSTERED

(

[id\_level] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[logs] Script Date: 16.04.2024 20:45:35 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[logs](

[logs\_id] [int] IDENTITY(1,1) NOT NULL,

[id\_panel] [nvarchar](10) NOT NULL,

[user\_id] [int] NOT NULL,

[initial\_stage] [nvarchar](50) NOT NULL,

[final\_stage] [nvarchar](50) NOT NULL,

[date] [datetime] NOT NULL,

[result] [bit] NOT NULL,

[reason] [nchar](100) NULL,

CONSTRAINT [PK\_logs] PRIMARY KEY CLUSTERED

(

[logs\_id] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[stage\_deal] Script Date: 16.04.2024 20:45:35 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[stage\_deal](

[id\_deal] [nvarchar](50) NOT NULL,

[name\_stage] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_stage\_deal] PRIMARY KEY CLUSTERED

(

[id\_deal] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

/\*\*\*\*\*\* Object: Table [dbo].[user] Script Date: 16.04.2024 20:45:35 \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[user](

[id\_user] [int] IDENTITY(1,1) NOT NULL,

[firstname] [nvarchar](100) NOT NULL,

[name] [nvarchar](75) NOT NULL,

[patronymic] [nvarchar](100) NOT NULL,

[access\_level] [int] NOT NULL,

[token] [nvarchar](100) NOT NULL,

[post] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_user] PRIMARY KEY CLUSTERED

(

[id\_user] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON) ON [PRIMARY]

) ON [PRIMARY]

GO

SET IDENTITY\_INSERT [dbo].[access] ON

INSERT [dbo].[access] ([id\_level], [name\_level]) VALUES (1, N'Низкий')

INSERT [dbo].[access] ([id\_level], [name\_level]) VALUES (2, N'Средний')

INSERT [dbo].[access] ([id\_level], [name\_level]) VALUES (3, N'')

SET IDENTITY\_INSERT [dbo].[access] OFF

GO

SET IDENTITY\_INSERT [dbo].[logs] ON

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (1, N'1944102691', 1, N'C25:2', N'C25:2', CAST(N'2024-03-26T00:00:00.000' AS DateTime), 1, NULL)

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (2, N'1944102691', 1, N'C25:2', N'C25:EXECUTING', CAST(N'2024-03-26T20:17:40.697' AS DateTime), 0, NULL)

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (3, N'1944102691', 1, N'C25:PREPARATION', N'C25:EXECUTING', CAST(N'2024-03-28T19:52:19.137' AS DateTime), 1, NULL)

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (5, N'1944102691', 1, N'C25:EXECUTING', N'C25:EXECUTING', CAST(N'2024-03-28T20:03:17.773' AS DateTime), 0, N'Панель находится в неподходящей стадии ')

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (6, N'1944102691', 1, N'C25:EXECUTING', N'C25:2', CAST(N'2024-03-28T20:03:43.683' AS DateTime), 1, N' ')

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (7, N'1111111111', 1, N'C25:APOLOGY', N'C25:EXECUTING', CAST(N'2024-03-28T20:05:03.180' AS DateTime), 0, N'Панель находится в неподходящей стадии ')

INSERT [dbo].[logs] ([logs\_id], [id\_panel], [user\_id], [initial\_stage], [final\_stage], [date], [result], [reason]) VALUES (8, N'1944102691', 2, N'C25:2', N'C25:EXECUTING', CAST(N'2024-03-30T16:26:03.980' AS DateTime), 0, N'Панель находится в неподходящей стадии ')

SET IDENTITY\_INSERT [dbo].[logs] OFF

GO

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:1', N'Отложено')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:2', N'В ремонте')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:7', N'Готово к отправке')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:APOLOGY', N'Ошибка заведения')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:EXECUTING', N'Принята на склад')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:FINAL\_INVOICE', N'Выходной контроль')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:LOSE', N'Заявка отменена')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:NEW', N'Новая заявка')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:PREPARATION', N'Проверка заявки')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:PREPAYMENT\_INVOIC', N'Заявка готова к отправке')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:UC\_DMZGI5', N'Паркинг')

INSERT [dbo].[stage\_deal] ([id\_deal], [name\_stage]) VALUES (N'C25:WON', N'Заявка завершена')

GO

SET IDENTITY\_INSERT [dbo].[user] ON

INSERT [dbo].[user] ([id\_user], [firstname], [name], [patronymic], [access\_level], [token], [post]) VALUES (1, N'Случаев', N'Максим', N'Константинович', 3, N'ылдфроапоырфвпароывраплрфыпвроаплрофыпвафыроваплрофыц', N'Мастер')

INSERT [dbo].[user] ([id\_user], [firstname], [name], [patronymic], [access\_level], [token], [post]) VALUES (2, N'Спиридонов', N'Евгений', N'Андреевич', 3, N'выаплоыврапошлщрываолдпролд', N'Мастер')

SET IDENTITY\_INSERT [dbo].[user] OFF

GO

ALTER TABLE [dbo].[logs] WITH CHECK ADD CONSTRAINT [FK\_logs\_stage\_deal] FOREIGN KEY([initial\_stage])

REFERENCES [dbo].[stage\_deal] ([id\_deal])

GO

ALTER TABLE [dbo].[logs] CHECK CONSTRAINT [FK\_logs\_stage\_deal]

GO

ALTER TABLE [dbo].[logs] WITH CHECK ADD CONSTRAINT [FK\_logs\_stage\_deal1] FOREIGN KEY([final\_stage])

REFERENCES [dbo].[stage\_deal] ([id\_deal])

GO

ALTER TABLE [dbo].[logs] CHECK CONSTRAINT [FK\_logs\_stage\_deal1]

GO

ALTER TABLE [dbo].[logs] WITH CHECK ADD CONSTRAINT [FK\_logs\_user] FOREIGN KEY([user\_id])

REFERENCES [dbo].[user] ([id\_user])

GO

ALTER TABLE [dbo].[logs] CHECK CONSTRAINT [FK\_logs\_user]

GO

ALTER TABLE [dbo].[user] WITH CHECK ADD CONSTRAINT [FK\_user\_access] FOREIGN KEY([access\_level])

REFERENCES [dbo].[access] ([id\_level])

ON UPDATE CASCADE

GO

ALTER TABLE [dbo].[user] CHECK CONSTRAINT [FK\_user\_access]

GO

USE [master]

GO

ALTER DATABASE course\_project SET READ\_WRITE

GO

Приложение Б

Код главного окна

XAML:

<Window x:Class="course\_project.MainWindow"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:course\_project;assembly=course\_project"

mc:Ignorable="d"

Title="SK-Warranty" Height="620" Width="400" ResizeMode="NoResize"

Topmost="True">

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="100\*"/>

<RowDefinition Height="85\*"/>

<RowDefinition Height="85\*"/>

<RowDefinition Height="85\*"/>

<RowDefinition Height="85\*"/>

<RowDefinition Height="100\*"/>

</Grid.RowDefinitions>

<StackPanel Grid.Row="0" Background="#424242">

<Label Content="Номер панели" FontSize="20" Height="40" Width="146" Foreground="White" />

<TextBox x:Name="text\_panel" FontSize="15" Width="300" Height="37" Background="Gray"/>

</StackPanel>

<StackPanel Grid.Row="1" Background="#5B5B5B">

<Button x:Name="warehouse" Background="#06b8af" Content="Принята на склад" FontSize="20" Margin="10, 5, 10, 0" Height="50" Click="warehouse\_Click"/>

</StackPanel>

<StackPanel Grid.Row="2" Background="#7D7D7D">

<Button FontSize="20" x:Name="fix" Background="#06b8af" Content="В ремонте" Margin="10, 10, 10, 0" Height="50" Click="fix\_Click"/>

</StackPanel>

<StackPanel Grid.Row="3" Background="#979AAA">

<Button x:Name="check" FontSize="20" Background="#06b8af" Content="В проверку" Margin="10, 10, 10, 0" Height="50" Click="check\_Click"/>

</StackPanel>

<StackPanel Grid.Row="4" Background="#A5A5A5">

<Button x:Name="sending" FontSize="20" Background="#06b8af" Content="Готово к отправке" Margin="10, 5, 10, 0" Height="50" Click="sending\_Click"/>

</StackPanel>

<StackPanel Grid.Row="5" Background="#CFCFCF">

<Button x:Name="log" FontSize="20" Content="Логи" Background="#06b8af" Margin="10, 10, 10, 0" Height="40" Click="log\_Click"/>

<Label Content="Version 1.1" Margin="0, 25, 255, 0" Width="124"/>

<Button x:Name="out" Content="Log out" Margin="305, -30, 0, 0" Width="74" Height="26" Background="#06b8af" Click="out\_Click"/>

</StackPanel>

</Grid>

</Window>

C#:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Navigation;

using System.Windows.Shapes;

namespace course\_project

{

/// <summary>

/// Логика взаимодействия для MainWindow.xaml

/// </summary>

public partial class MainWindow : Window

{

public static string id\_panel;

public MainWindow()

{

InitializeComponent();

if (Window2.flag == true)

{

log.IsEnabled = false;

}

}

private void fix\_Click(object sender, RoutedEventArgs e)

{

try

{

string result1 = ID\_verification.CheckString(text\_panel.Text);

if(result1 != "Успех")

{

MessageBox.Show(result1);

}

else

{

id\_panel = text\_panel.Text;

string id\_deal = SearchDeal.SearchDealMethod(text\_panel.Text);

string result = DealManager.сhangeStage\_fix(id\_deal);

MessageBox.Show(result);

}

}

catch

{

MessageBox.Show("Произошла ошибка");

}

}

private void warehouse\_Click(object sender, RoutedEventArgs e)

{

try

{

string result1 = ID\_verification.CheckString(text\_panel.Text);

if (result1 != "Успех")

{

MessageBox.Show(result1);

}

else

{

id\_panel = text\_panel.Text;

string id\_deal = SearchDeal.SearchDealMethod(text\_panel.Text);

string result = DealManager.сhangeStage\_accepted\_warehouse(id\_deal);

MessageBox.Show(result);

}

}

catch

{

MessageBox.Show("Произошла ошибка");

}

}

private void check\_Click(object sender, RoutedEventArgs e)

{

try

{

string result1 = ID\_verification.CheckString(text\_panel.Text);

if (result1 != "Успех")

{

MessageBox.Show(result1);

}

else

{

id\_panel = text\_panel.Text;

string id\_deal = SearchDeal.SearchDealMethod(text\_panel.Text);

string result = DealManager.сhangeStage\_test(id\_deal);

MessageBox.Show(result);

}

}

catch

{

MessageBox.Show("Произошла ошибка");

}

}

private void sending\_Click(object sender, RoutedEventArgs e)

{

try

{

string result1 = ID\_verification.CheckString(text\_panel.Text);

if (result1 != "Успех")

{

MessageBox.Show(result1);

}

else

{

id\_panel = text\_panel.Text;

string id\_deal = SearchDeal.SearchDealMethod(text\_panel.Text);

string result = DealManager.сhangeStage\_ready\_ship(id\_deal);

MessageBox.Show(result);

}

}

catch

{

MessageBox.Show("Произошла ошибка");

}

}

private void log\_Click(object sender, RoutedEventArgs e)

{

Window1 window = new Window1();

window.Show();

this.Close();

}

private void out\_Click(object sender, RoutedEventArgs e)

{

file\_write\_and\_read.file\_write(" ");

Window2 window = new Window2();

window.Show();

this.Close();

}

}

}

Приложение В

Код окна входа

XAML:

<Window x:Class="course\_project.Window2"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:course\_project"

mc:Ignorable="d"

Title="Вход" Height="343" Width="290" ResizeMode="NoResize">

<Grid Background="Gray">

<TextBox x:Name="tokenT" HorizontalAlignment="Center" Background="#b3b3b3" FontSize="20" Margin="0,64,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="270" Height="100"/>

<TextBlock HorizontalAlignment="Left" Margin="101,26,0,0" TextWrapping="Wrap" Text="Токен" FontSize="25" VerticalAlignment="Top"/>

<Button Content="Вход" HorizontalAlignment="Center" Background="#06b8af" FontSize="35" VerticalAlignment="Top" Height="97" Width="270" Margin="0,169,0,0" Click="Button\_Click"/>

</Grid>

</Window>

C#:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace course\_project

{

/// <summary>

/// Логика взаимодействия для Window2.xaml

/// </summary>

public partial class Window2 : Window

{

public static int id\_user = 0;

public static int id\_level;

public static bool flag = false;

course\_projectEntities4 db = new course\_projectEntities4();

public Window2()

{

if (access\_verification() == true)

{

MainWindow window = new MainWindow();

window.Show();

this.Close();

}

InitializeComponent();

}

private void Button\_Click(object sender, RoutedEventArgs e)

{

if (tokenT.Text == "viva\_laswegas")

{

flag = true;

MessageBox.Show("Данные в БД не фиксируются");

MainWindow window = new MainWindow();

window.Show();

this.Close();

}

else

{

var list = db.user.Where(x => x.token == tokenT.Text);

int? level = 0;

if (list != null)

{

foreach (var item in list)

{

level = item.access\_level;

id\_user = item.id\_user;

id\_level = (int)level;

}

if (level == 4)

{

Window1 window = new Window1();

window.Show();

this.Close();

}

else

{

if (level > 1)

{

file\_write\_and\_read.file\_write(AESEncryption.Encrypt(tokenT.Text, "sAL4-AVlWoCGELM-"));

MainWindow window = new MainWindow();

window.Show();

this.Close();

}

else if (level == 1)

{

MessageBox.Show($"Нет прав!");

}

else

{

MessageBox.Show($"Неверный токен!");

}

}

}

}

}

public bool access\_verification()

{

try

{

string tokenAESE = file\_write\_and\_read.file\_read();

string token = AESEncryption.Decrypt(tokenAESE, "sAL4-AVlWoCGELM-");

var list = db.user.Where(x => x.token == token);

int? level = 1;

if (list != null)

{

foreach (var item in list)

{

level = item.access\_level;

id\_user = item.id\_user;

}

if (level > 1)

{

return true;

}

else

{

return false;

}

}

else

{

return false;

}

}

catch

{

return false;

}

}

}}

Приложение Г

Код окна логов

XAML:

<Window x:Class="course\_project.Window1"

xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"

xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"

xmlns:d="http://schemas.microsoft.com/expression/blend/2008"

xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"

xmlns:local="clr-namespace:course\_project"

mc:Ignorable="d"

Title="Логи" Height="485" Width="845" MinHeight="485" MinWidth="845"

Topmost="True">

<Grid Background="Gray">

<Grid.RowDefinitions>

<RowDefinition Height="385\*"/>

<RowDefinition Height="100\*"/>

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="250\*"/>

<ColumnDefinition Width="150\*"/>

</Grid.ColumnDefinitions>

<DataGrid Grid.Row="0" x:Name="logsG" AutoGenerateColumns="False" Background="#b3b3b3">

<DataGrid.Columns>

<DataGridTextColumn Header="Номер панели" Binding="{Binding id\_panel}"/>

<DataGridTextColumn Header="Пользователь" Binding="{Binding user.firstname}"/>

<DataGridTextColumn Header="Начальная стадия" Binding="{Binding stage\_deal.name\_stage}"/>

<DataGridTextColumn Header="Конечная стадия" Binding="{Binding stage\_deal1.name\_stage}"/>

<DataGridTextColumn Header="Дата" Binding="{Binding date}"/>

<DataGridTextColumn Header="Результат" Binding="{Binding result}"/>

<DataGridTextColumn Header="Причина" Binding="{Binding reason}"/>

</DataGrid.Columns>

</DataGrid>

<Button x:Name="comeback" FontSize="40" Content="Назад" Background="#06b8af" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" Grid.Row="1" Click="comeback\_Click" />

<Label Grid.Column="1" Content="Поиск по сотруднику" HorizontalAlignment="Left" Margin="9,50,0,0" VerticalAlignment="Top"/>

<ComboBox Grid.Column="1" Background="#b3b3b3" x:Name="find\_name" HorizontalAlignment="Left" Margin="164,54,0,0" VerticalAlignment="Top" Width="120" />

<RadioButton Grid.Column="1" x:Name="earlier" GroupName="first" Content="Раньше даты" HorizontalAlignment="Left" Margin="164,94,0,0" VerticalAlignment="Top" Height="20" Width="120"/>

<RadioButton Grid.Column="1" x:Name="matches" GroupName="first" Content="Совпадает" HorizontalAlignment="Left" Margin="164,121,0,0" VerticalAlignment="Top" Height="20" Width="120" />

<RadioButton Grid.Column="1" x:Name="later" GroupName="first" Content="Позднее" HorizontalAlignment="Left" Margin="164,148,0,0" VerticalAlignment="Top" Height="20" Width="120"/>

<DatePicker Background="#b3b3b3" Grid.Column="1" PreviewKeyDown="DatePicker\_PreviewKeyDown" x:Name="date" HorizontalAlignment="Left" Margin="9,107,0,0" VerticalAlignment="Top" Height="42" Width="149"/>

<Label Grid.Column="1" Content="Поиск по начальной стадии" HorizontalAlignment="Left" Margin="9,211,0,0" VerticalAlignment="Top"/>

<ComboBox Grid.Column="1" x:Name="stage" HorizontalAlignment="Left" Margin="178,213,0,0" VerticalAlignment="Top" Width="120" />

<Button Grid.Column="1" Background="#06b8af" x:Name="show" Content="Показать" HorizontalAlignment="Left" Margin="9,285,0,0" VerticalAlignment="Top" Width="297" Height="28" Click="show\_Click"/>

<Button Grid.Column="1" Background="#06b8af" x:Name="rest" Content="Сбросить" HorizontalAlignment="Left" Margin="9,320,0,0" VerticalAlignment="Top" Width="297" Height="28" Click="rest\_Click"/>

<Label Grid.Column="1" Content="Поиск по конечной стадии" HorizontalAlignment="Left" Margin="9,244,0,0" VerticalAlignment="Top"/>

<ComboBox Grid.Column="1" x:Name="stage\_fin" HorizontalAlignment="Left" Margin="178,246,0,0" VerticalAlignment="Top" Width="120" />

<RadioButton Grid.Column="1" x:Name="error" GroupName="second" Content="Ошибка" HorizontalAlignment="Left" Margin="164,191,0,0" VerticalAlignment="Top" Height="20" Width="120"/>

<RadioButton Grid.Column="1" x:Name="successful" GroupName="second" Content="Успех" HorizontalAlignment="Left" Margin="16,191,0,0" VerticalAlignment="Top" Height="20" Width="120"/>

<Label Grid.Column="1" Content="Поиск по ID панели" HorizontalAlignment="Left" Margin="10,19,0,0" VerticalAlignment="Top"/>

<TextBox Background="#b3b3b3" Grid.Column="1" x:Name="id" HorizontalAlignment="Left" Margin="164,23,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>

<Label Grid.Column="1" Content="Статус:" HorizontalAlignment="Left" Margin="16,165,0,0" VerticalAlignment="Top"/>

<Button Grid.Column="1" Content="Удалить запись" x:Name="del" FontSize="40" Visibility="Hidden" Background="#06b8af" Grid.Row="1" HorizontalAlignment="Stretch" VerticalAlignment="Stretch" Click="del\_Click"/>

</Grid>

</Window>

C#

using System;

using System.Collections.Generic;

using System.Linq;

using System.Reflection.Emit;

using System.Text;

using System.Threading.Tasks;

using System.Windows;

using System.Windows.Controls;

using System.Windows.Data;

using System.Windows.Documents;

using System.Windows.Input;

using System.Windows.Media;

using System.Windows.Media.Imaging;

using System.Windows.Shapes;

namespace course\_project

{

/// <summary>

/// Логика взаимодействия для Window1.xaml

/// </summary>

public partial class Window1 : Window

{

private course\_projectEntities4 db = new course\_projectEntities4();

public static List<logs> list;

public Window1()

{

InitializeComponent();

if (Window2.id\_level == 4)

{

del.Visibility = Visibility.Visible;

}

list = db.logs.ToList();

logsG.ItemsSource = db.logs.ToList();

find\_name.ItemsSource = db.user.Select(x => x.firstname).ToList();

stage.ItemsSource = db.stage\_deal.Select(x => x.name\_stage).ToList();

stage\_fin.ItemsSource = db.stage\_deal.Select(x => x.name\_stage).ToList();

}

private void comeback\_Click(object sender, RoutedEventArgs e)

{

if(Window2.id\_level == 4)

{

Window2 window = new Window2();

window.Show();

this.Close();

}

else

{

MainWindow window = new MainWindow();

window.Show();

this.Close();

}

}

private void show\_Click(object sender, RoutedEventArgs e)

{

try

{

var list2 = db.logs.ToList();

string value = stage.Text;

var id\_arr = db.stage\_deal.FirstOrDefault(x => x.name\_stage == value);

if (id.Text.Trim() != null)

{

var f = list2.Where(x => x.id\_panel.Contains(id.Text)).ToList();

list2 = f;

}

if (id\_arr != null)

{

var f = list2.Where(x => x.initial\_stage == id\_arr.id\_deal).ToList();

list2 = f;

}

string value1 = stage\_fin.Text;

var id\_arr1 = db.stage\_deal.FirstOrDefault(x => x.name\_stage == value1);

if (id\_arr1 != null)

{

var f = list2.Where(x => x.final\_stage == id\_arr1.id\_deal).ToList();

list2 = f;

}

string value2 = find\_name.Text;

var id\_arr2 = db.user.FirstOrDefault(x => x.firstname == value2);

if (id\_arr2 != null)

{

var f = list2.Where(x => x.user\_id == id\_arr2.id\_user).ToList();

list2 = f;

}

if (error.IsChecked == true)

{

var f = list2.Where(x => x.result == false).ToList();

list2 = f;

}

if (successful.IsChecked == true)

{

var f = list2.Where(x => x.result == true).ToList();

list2 = f;

}

if (earlier.IsChecked == true)

{

var f = list2.Where(x => x.date.Date > Convert.ToDateTime(date.Text).Date).ToList();

list2 = f;

}

if (matches.IsChecked == true)

{

var f = list2.Where(x => x.date.Date == Convert.ToDateTime(date.Text).Date).ToList();

list2 = f;

}

if (later.IsChecked == true)

{

var f = list2.Where(x => x.date.Date < Convert.ToDateTime(date.Text).Date).ToList();

list2 = f;

}

list = list2;

logsG.ItemsSource = list2;

}

catch

{

MessageBox.Show("Нет связи с БД");

}

}

private void rest\_Click(object sender, RoutedEventArgs e)

{

find\_name.Text = "";

stage\_fin.Text = "";

stage.Text = "";

logsG.ItemsSource = list;

error.IsChecked = false;

successful.IsChecked = false;

later.IsChecked = false;

matches.IsChecked = false;

earlier.IsChecked = false;

date.Text = "";

id.Text = "";

}

private void DatePicker\_PreviewKeyDown(object sender, KeyEventArgs e)

{

e.Handled = true;

}

private void del\_Click(object sender, RoutedEventArgs e)

{

try

{

var del = logsG.SelectedItem as logs;

if (del != null)

{

db.logs.Remove(del);

MessageBox.Show("Успех");

logsG.ItemsSource = list;

}

else

{

MessageBox.Show("Выберите строку в таблице");

}

}

catch

{

MessageBox.Show("Нет связи с БД");

}

}

}

}

Приложение Д

Класс для поиска сделки

using System;

using System.IO;

using System.Net;

using System.Xml.Linq;

using Newtonsoft.Json.Linq;

public class SearchDeal

{

public static string SearchDealMethod(string idPanel)

{

string idDeal = "";

try

{

// URL for getting JSON

string urlString =

// Creating URL object

Uri url = new Uri(urlString);

// Creating HttpWebRequest object

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(url);

request.Method = "GET";

// Getting response from server

HttpWebResponse response = (HttpWebResponse)request.GetResponse();

if (response.StatusCode == HttpStatusCode.OK)

{

// Reading data from input stream

StreamReader reader = new StreamReader(response.GetResponseStream());

string responseContent = reader.ReadToEnd();

// Processing received JSON

JObject jsonObject = JObject.Parse(responseContent);

JArray dealsArray = (JArray)jsonObject["result"];

JObject lastDeal = (JObject)dealsArray[dealsArray.Count - 1];

idDeal = lastDeal["ID"].ToString();

}

else

{

idDeal = "Error executing request. Response code: " + response.StatusCode;

}

}

catch

{

idDeal = "Error executing";

}

return idDeal;

}

}

Приложение Е

Класс для проверки стадии

using System.Net;

using Newtonsoft.Json.Linq;

using System.Text;

using System.IO;

using System;

public class CheckStage

{

public static string Check\_Stage(string id\_deal)

{

string status = "";

try

{

// URL for getting JSON

string urlString =

// Create a HttpWebRequest object

HttpWebRequest request = (HttpWebRequest)WebRequest.Create(urlString);

request.Method = "GET";

// Get the response from the server

HttpWebResponse response = (HttpWebResponse)request.GetResponse();

if (response.StatusCode == HttpStatusCode.OK)

{

// Read data from the input stream

using (StreamReader reader = new StreamReader(response.GetResponseStream(), Encoding.UTF8))

{

string inputLine;

StringBuilder jsonResponse = new StringBuilder();

while ((inputLine = reader.ReadLine()) != null)

{

jsonResponse.Append(inputLine);

}

reader.Close();

// Processing the received JSON

JObject jsonObject = JObject.Parse(jsonResponse.ToString());

JObject dealsArray = (JObject)jsonObject["result"];

status = (string)dealsArray["STAGE\_ID"];

}

}

else

{

status = "Error executing request. Response code: " + response.StatusCode;

}

response.Close();

}

catch (Exception e)

{

status = "Error executing: " + e.Message;

}

return status;

}

}

Приложение Ж

Код для изменения стадии сделки

using course\_project;

using System;

using System.Net.Http;

using System.Xml.XPath;

public static class DealManager

{

private static string ChangeStage\_universal(string dealId, string stageId, string stage)

{

string result = "";

HttpClient client = new HttpClient();

try

{

// URL и JSON-запрос

// Создание запроса POST

StringContent content = new StringContent(json);

content.Headers.ContentType = new System.Net.Http.Headers.MediaTypeHeaderValue("application/json");

HttpResponseMessage response = client.PostAsync(url, content).Result;

if (response.IsSuccessStatusCode)

{

// Преобразование ответа в строку

string responseBody = response.Content.ReadAsStringAsync().Result;

result = "Успех";

if(Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, stageId, true, "");

}

}

else

{

result = "Запрос отправлен, но сообщеня об успехе нет";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, stageId, false, result);

}

}

}

catch

{

result = "Ошибка запроса";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, stageId, false, result);

}

}

return result;

}

public static string сhangeStage\_accepted\_warehouse(string dealId)

{

string result = "";

string stage = CheckStage.Check\_Stage(dealId);

if (stage.Length <= 30)

{

if (stage == "C25:NEW" || stage == "C25:PREPARATION" || stage == "C25:PREPAYMENT\_INVOIC")

{

result = ChangeStage\_universal(dealId, "C25:EXECUTING", stage);

}

else

{

result = $"Панель находится в неподходящей стадии:{stage}";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, "C25:EXECUTING", false, "Панель находится в неподходящей стадии");

}

}

}

else

{

result = stage;

}

return result;

}

public static string сhangeStage\_fix(string dealId)

{

string result = "";

string stage = CheckStage.Check\_Stage(dealId);

if (stage.Length <= 30)

{

if (stage == "C25:1" || stage == "C25:2" || stage == "C25:PREPAYMENT\_INVOIC" || stage == "C25:7" || stage == "C25:UC\_DMZGI5" || stage == "C25:EXECUTING")

{

result = ChangeStage\_universal(dealId, "C25:2", stage);

}

else

{

result = $"Панель находится в неподходящей стадии:{stage}";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, "C25:2", false, "Панель находится в неподходящей стадии");

}

}

}

else

{

result = stage;

}

return result;

}

public static string сhangeStage\_ready\_ship(string dealId)

{

string result = "";

string stage = CheckStage.Check\_Stage(dealId);

if (stage.Length <= 30)

{

if (stage != "C25:LOSE" && stage != "C25:APOLOGY" && stage != "C25:WON")

{

result = ChangeStage\_universal(dealId, "C25:7", stage);

}

else

{

result = $"Панель находится в неподходящей стадии:{stage}";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, "C25:7", false, "Панель находится в неподходящей стадии");

}

}

}

else

{

result = stage;

}

return result;

}

public static string сhangeStage\_test(string dealId)

{

string result = "";

string stage = CheckStage.Check\_Stage(dealId);

if (stage.Length <= 30)

{

result = ChangeStage\_universal(dealId, "C25:FINAL\_INVOICE", stage);

}

else

{

result = $"Панель находится в неподходящей стадии:{stage}";

if (Window2.flag == false)

{

logs\_save.save(Window2.id\_user, MainWindow.id\_panel, stage, "C25:FINAL\_INVOICE", false, "Панель находится в неподходящей стадии");

}

}

return result;

}

}

Приложение З

Код для сохранения логов

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace course\_project

{

public class logs\_save

{

public static void save(int user\_id, string id\_panel, string first\_stage, string final\_stage, bool result, string reason)

{

course\_projectEntities4 db = new course\_projectEntities4();

logs logs = new logs();

logs.user\_id = user\_id;

logs.id\_panel = id\_panel;

logs.result = result;

logs.initial\_stage = first\_stage;

logs.final\_stage = final\_stage;

logs.date = DateTime.Now;

logs.reason = reason;

db.logs.Add(logs);

db.SaveChanges();

}

}

}

## Приложение И

Код для шифрования токена

using System;

using System.IO;

using System.Security.Cryptography;

using System.Text;

namespace course\_project

{

public class AESEncryption

{

public static string Encrypt(string plainText, string key)

{

using (Aes aesAlg = Aes.Create())

{

aesAlg.Key = Encoding.UTF8.GetBytes(key);

aesAlg.IV = new byte[16]; // Используем IV длиной 16 байт для AES

ICryptoTransform encryptor = aesAlg.CreateEncryptor(aesAlg.Key, aesAlg.IV);

using (MemoryStream msEncrypt = new MemoryStream())

{

using (CryptoStream csEncrypt = new CryptoStream(msEncrypt, encryptor, CryptoStreamMode.Write))

{

using (StreamWriter swEncrypt = new StreamWriter(csEncrypt))

{

swEncrypt.Write(plainText);

}

}

return Convert.ToBase64String(msEncrypt.ToArray());

}

}

}

public static string Decrypt(string cipherText, string key)

{

using (Aes aesAlg = Aes.Create())

{

aesAlg.Key = Encoding.UTF8.GetBytes(key);

aesAlg.IV = new byte[16]; // Используем IV длиной 16 байт для AES

ICryptoTransform decryptor = aesAlg.CreateDecryptor(aesAlg.Key, aesAlg.IV);

using (MemoryStream msDecrypt = new MemoryStream(Convert.FromBase64String(cipherText)))

{

using (CryptoStream csDecrypt = new CryptoStream(msDecrypt, decryptor, CryptoStreamMode.Read))

{

using (StreamReader srDecrypt = new StreamReader(csDecrypt))

{

return srDecrypt.ReadToEnd();

}

}

}

}

}

}

}

Приложение К

Код для записи зашифрованного токена в TXT файл

using System;

using System.IO;

namespace course\_project

{

public class file\_write\_and\_read

{

public static void file\_write(string text)

{

try

{

string filePath = "user.txt";

File.WriteAllText(filePath, text);

}

catch (Exception e)

{

}

}

public static string file\_read()

{

try

{

string filePath = "user.txt";

string data = File.ReadAllText(filePath);

return data;

}

catch (Exception e)

{

return "Ошибка при чтении данных из файла: " + e.Message;

}

}

}

}