

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение высшего образования
«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»

Кафедра компьютерных технологий и программной инженерии

ОТЧЁТ ПО ПРАКТИКЕ
ЗАЩИЩЁН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

Ст. преп.

М.Д. Поляк

должность, уч. степень, звание

подпись, дата

инициалы, фамилия

ОТЧЁТ ПО ПРАКТИКЕ

вид практики производственная

тип практики Научно-исследовательская работа

на тему индивидуального задания Разработка графического клиент-серверного
приложения на языке с# с использованием Entity Framework для ведения

картотеки десятичных номеров

выполнен Сидиропуло Хетагом Владимировичем

фамилия, имя, отчество обучающегося в творительном падеже

по направлению подготовки 09.03.04

код

Программная инженерия

наименование направления

направленности

наименование направления

02

код

Проектирование программных систем

наименование направленности

наименование направленности

Обучающийся группы № 4031

номер

подпись, дата

Х.В. Сидиропуло

инициалы, фамилия

Санкт–Петербург 2023

1. Введение

В ходе прохождения производственной практики было получено следующее задание:

Создание программы с графическим интерфейсом на языке C# взаимодействующей с базой данных PostgreSQL. В ходе практики было создано приложение для ведения картотеки десятичных номеров. Взаимодействие с базой выполнено через обращения к представлениям (create view) и функциям.

2. Входные данные

В качестве входных данных для данной задачи выступают данные, хранящиеся в базе.

3. Используемое программное обеспечение

Разработка приложения велась в среде MS Visual Studio 2022

Данные хранятся при помощи СУБД PostgreSQL.

В качестве менеджера СУБД использовался Dbeaver.

4. Ход работы

4.1 Добавление представлений и функций для более удобного взаимодействия с приложением

4.1.1 Code_2nums_wv – Представление для заполнения в интерфейсе корневого узла TreeView

```
CREATE OR REPLACE VIEW designation.code_2nums_wv
AS SELECT DISTINCT ON (("left"(designation.decimal_characteristics::text, 2)))
decimal_organization.code,
    "left"(designation.decimal_characteristics::text, 2) AS first_num
FROM designation.designation
    JOIN vpref.organization ON designation.organization_oid =
organization.iu_organization_code_oid
    JOIN vpref.decimal_organization ON decimal_organization.cid =
organization.decimal_oid
ORDER BY ("left"(designation.decimal_characteristics::text, 2));
```

4.1.2 character_device_wv – Представление для заполнения в интерфейсе узлов TreeView

```
CREATE OR REPLACE VIEW designation.character_device_wv
AS SELECT decimal_organization.code,
    designation.decimal_characteristics AS full_num,
    designation.decimal_name AS dec_name
FROM designation.designation
    JOIN vpref.organization ON designation.organization_oid =
organization.iu_organization_code_oid
    JOIN vpref.decimal_organization ON decimal_organization.cid =
organization.decimal_oid
ORDER BY designation.decimal_characteristics;
```

4.1.3 vpref_emp_vw – Представление для работы с данными о сотрудниках, различные данные используются в разных частях интерфейса.

```
CREATE OR REPLACE VIEW designation.vpref_emp_vw
AS SELECT employee.display_name,
    employee.sur_name,
    employee.initials,
    post_employee.name,
    employee.login,
    employee.cid AS dscid,
    group_employee.name AS department_name
FROM vpref.employee
    JOIN vpref.post_employee ON employee.position_oid = post_employee.cid
    JOIN vpref.group_employee ON employee.department_oid = group_employee.cid
ORDER BY employee.display_name;
```

4.1.4 decimal_autor_wv – Представление для отображения десятичных номеров и информации о сотрудниках

```
CREATE OR REPLACE VIEW designation.decimal_autor_wv
AS SELECT designation_number.number,
       designation_number.part_name,
       designation_number.occupied,
       designation_number.archived,
       designation_number.is_archived,
       designation.decimal_characteristics,
       designation_number.comment,
       ds_employee.employee_oid,
       employee.display_name,
       group_employee.name
FROM designation.designation_number
      JOIN designation.designation ON designation_number.designation_oid =
designation.cid
      JOIN designation.ds_employee ON designation_number.author_oid = ds_employee.cid
      JOIN vpref.employee ON ds_employee.employee_oid = employee.cid
```

4.1.5 add_new_number – функция для добавления нового десятичного номера. (если не существовал)

```
CREATE OR REPLACE FUNCTION designation.add_new_number(p_decimal_characteristic
character varying, p_decimal_name text, p_organization_oid bigint)
RETURNS integer
LANGUAGE plpgsql
AS $function$
DECLARE
    new_number int4;
BEGIN
    IF length(p_decimal_characteristic) < 6 THEN
        RAISE EXCEPTION 'p_decimal_characteristic должно содержать не менее 6
символов';
    END IF;
    INSERT INTO designation.designation (
        decimal_characteristics,
        decimal_name,
        organization_oid
    )
    VALUES (
        p_decimal_characteristic,
        p_decimal_name,
        p_organization_oid
    )
    RETURNING cid INTO new_number;
    RETURN new_number;
END;
$function$
;
```

4.1.6 archivation_number – функция для архивации десятичного номера

```
CREATE OR REPLACE FUNCTION designation.archivation_number(rownumber integer,
archiveddate timestamp with time zone, deccharacter character varying)
  RETURNS void
  LANGUAGE plpgsql
AS $function$
DECLARE
  designation_oid_check BIGINT;
BEGIN
  SELECT designation.cid INTO designation_oid_check
  FROM designation.designation
  WHERE decimal_characteristics = decCharacter;

  IF designation_oid_check IS NOT NULL THEN
    UPDATE designation.designation_number
    SET archived = ArchivedDate
    WHERE number = rowNumber AND designation_number.designation_oid =
designation_oid_check;
  END IF;
END;
$function$
;
```

4.1.7 insert_designation_number – функция для взятия десятичного номера

```
CREATE OR REPLACE FUNCTION designation.insert_designation_number(p_part_name text,
p_occupied timestamp with time zone, p_comment text, p_designation_oid bigint,
p_author_oid bigint)
  RETURNS text
  LANGUAGE plpgsql
AS $function$
DECLARE
  new_number int4;
  formatted_number text;
  new_cid int4;
BEGIN
  -- Получение нового значения cid
  --SELECT COALESCE(MAX(cid), 0) + 1 INTO new_cid FROM
designation.designation_number;
  -- Получение нового значения number
  SELECT COALESCE(MAX("number"), 0) + 1 INTO new_number FROM
designation.designation_number WHERE designation_oid = p_designation_oid;

  -- Проверка, если new_number больше 999, выбрасывается ошибка и функция
завершается
  IF new_number > 999 THEN
    RETURN 'over_number_error';
  END IF;

  -- Форматирование числа
  IF new_number < 100 THEN
    formatted_number := '.' || to_char(new_number, 'FM000');
  ELSE
    formatted_number := '.' || new_number::text;
  END IF;

  -- Вставка новой записи
  INSERT INTO designation.designation_number (
```

```

--      cid,
--      "number",
--      part_name,
--      occupied,
--      comment,
--      designation_oid,
--      author_oid
--    )
VALUES (
--      new_cid,
--      new_number,
--      p_part_name,
--      p_occupied,
--      p_comment,
--      p_designation_oid,
--      p_author_oid
);

-- Проверка, была ли выполнена вставка
IF FOUND THEN
-- Возврат отформатированного значения
RETURN formatted_number;
ELSE
-- Возвращение сообщения об ошибке
RETURN 'Ошибка вставки записи';
END IF;
END;
$function$
;

```


4.2 Описание программы

Программа используется для ведения картотеки десятичных номеров на предприятии

Децимальный номер состоит из:

- 1) Код организации
- 2) шестизначное целое число, обозначающее код изделия по ЕСКД
- 3) порядковый номер.

Регистратор Децимальных номеров АО "Вибро-прибор". Версия 5.0.1



Взять децимальный номер

Поиск

Параметр поиска:

Номер

Найти

№	ФИО	Подразделение	Наименование д
001	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция по входному кон
002	Селякова Ирина	Конструкторский сектор	Планка торцевая передняя
003	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Панель
004	Руденко Валерий	Направление II	Крышка
005	Демиденко Анастасия Владимировна	Конструкторский сектор	Датчик токовых ревой AR2100

Обновить

Выход

Взятие нового номера:

Регистратор Децимальных номеров АО "Виб

Взять децимальный н

ЖЯИУ. 11

ЖЯИУ. 111111 - 111

ЖЯИУ. 111112 - 112

ЖЯИУ. 30

ЖЯИУ. 32

ЖЯИУ. 36

ЖЯИУ. 40

ЖЯИУ. 41

ЖЯИУ. 42

ЖЯИУ. 43

ЖЯИУ. 44

ЖЯИУ. 46

ЖЯИУ. 56

ЖЯИУ. 68

ЖЯИУ. 71

ЖЯИУ. 72

ЖЯИУ. 73

ЖЯИУ. 74

ЖЯИУ. 75

Обновить

osoft.Extensions.Caching.Abstraction... 14.07.2023

osoft.Extensions.Caching.Memory.dll 14.07.2023

Добавление/Взятие характеристики

Характеристика ЖЯИУ.

Характеристика (6 цифр) 666666

ЕСКД

Описание характеристики

Ввод описания характеристики

Описание детали/сборки

ФИО Брюзгин Антон Евгеньевич

Другой пользователь

Подразделение Отдел разработки первичных преобразов

Название детали/сборки Название

Ввод коммента

Комментарий

Дата взятия 21 июля 2023 г.

Взять

Наименование детали/сбор

23

23

23

Выход

После взятия:

ЖЯИУ. 11	№	ФИО	Подразделение	Наименование детали/сборки
ЖЯИУ. 30	001	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Название
ЖЯИУ. 32				
ЖЯИУ. 36				
ЖЯИУ. 40				
ЖЯИУ. 41				
ЖЯИУ. 42				
ЖЯИУ. 43				
ЖЯИУ. 44				
ЖЯИУ. 46				
ЖЯИУ. 56				
ЖЯИУ. 66				
ЖЯИУ. 666666 - Ввод описания характ				
ЖЯИУ. 68				
ЖЯИУ. 71				
ЖЯИУ. 72				
ЖЯИУ. 73				
ЖЯИУ. 74				
ЖЯИУ. 75				

До архивации:

ЖЯИУ. 30	ЖЯИУ. 300004 - Сборочные единицы об	№	ФИО	Подразделение	Наименование детали/сборки
	ЖЯИУ. 300536 - Сборочные единицы об	019	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Основание МВ-44М
	ЖЯИУ. 300543 - Сборочные единицы об	020	Селякова Ирина	Конструкторский сектор	Консоль
	ЖЯИУ. 301111 - Сборочные единицы об	021	Черников Вячеслав	Конструкторский сектор	Основание
	ЖЯИУ. 301122 - Сборочные единицы об	022	Черников Вячеслав	Конструкторский сектор	Основание
	ЖЯИУ. 301191 - Сборочные единицы об	023	Селякова Ирина	Конструкторский сектор	Основание
	ЖЯИУ. 301261 - Сборочные единицы об				
	ЖЯИУ. 301262 - Сборочные единицы об				
	ЖЯИУ. 301264 - Сборочные единицы об				
	ЖЯИУ. 301314 - Сборочные единицы об				

Регистратор Децимальных номеров АО "Вибро-прибор". Версия 5.0.1

Взять децимальный номер

ЖЯИУ. 30

ЖЯИУ. 300004 - Сборочные единицы об

ЖЯИУ. 300536 - Сборочные единицы об

ЖЯИУ. 300543 - Сборочные единицы об

ЖЯИУ. 301111 - Сборочные единицы об

ЖЯИУ. 301122 - Сборочные единицы об

ЖЯИУ. 301191 - Сборочные единицы об

ЖЯИУ. 301261 - Сборочные единицы об

ЖЯИУ. 301262 - Сборочные единицы об

ЖЯИУ. 301264 - Сборочные единицы об

ЖЯИУ. 301314 - Сборочные единицы об

ЖЯИУ. 301411 - Сборочные единицы об

ЖЯИУ. 301413 - Сборочные единицы об

ЖЯИУ. 301511 - Сборочные единицы об

ЖЯИУ. 301561 - Сборочные единицы об

ЖЯИУ. 301716 - Сборочные единицы об

ЖЯИУ. 301732 - Сборочные единицы об

ЖЯИУ. 302634 - Сборочные единицы об

ЖЯИУ. 302643 - Сборочные единицы об

ЖЯИУ. 303659 - Сборочные единицы об

ЖЯИУ. 303664 - Сборочные единицы об

ЖЯИУ. 303717 - Сборочные единицы об

Обновить

Поиск

Параметр поиска

№

ФИО

019 Брюзгин Антон Евгеньевич

020 Селякова Ирина

021 Черников Вячеслав

022 Черников Вячеслав

023 Селякова Ирина

Отдел разработки первичных преобразователей

Конструкторский сектор

Конструкторский сектор

Конструкторский сектор

Конструкторский сектор

Сдача в архив

Характеристика ЖЯИУ.

Характеристика (6 цифр)

301314

ЕСКД

Описание характеристики

Сборочные единицы общемашиностроительные. Устройства корпусные, опорные, несущие, крепления. Устройства опорные. Элементы опорные. Основания, башмаки.

Описание детали/сборки

ФИО

Черников Вячеслав

Название детали/сборки

Основание МВ-44М

Дата сдачи

21 июля 2023 г.

Сдать

После:

	№	ФИО	Подразделение	Наименование детали/сборки
▶	019	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Основание МВ-44М
	020	Селякова Ирина	Конструкторский сектор	Консоль
	021	Черников Вячеслав	Конструкторский сектор	Основание
	022	Черников Вячеслав	Конструкторский сектор	Основание
	023	Селякова Ирина	Конструкторский сектор	Основание

Функции поиска:

Поиск

Параметр поискаНомер

30

Найти

	№	Номер	ФИО	Подразделение	
▶	001	304134	Виноградова Анна Михайловна	Конструкторский сектор	Направляю
	001	300004	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция
	001	302634	Демиденко Анастасия Владимировна	Конструкторский сектор	Штуцер (МД
	001	304246	Селякова Ирина	Конструкторский сектор	Демпфер
	001	304515	Селякова Ирина	Конструкторский сектор	Эксцентрик
	001	305651	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Вибропрео
	001	303664	Селякова Ирина	Конструкторский сектор	Шкив
	001	303712	Селякова Ирина	Конструкторский сектор	Вал
	001	304599	Демиденко Анастасия Владимировна	Конструкторский сектор	Кольцо Вал
	001	301511	Селякова Ирина	Конструкторский сектор	Фланец
	001	305653	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект за
	001	300543	Селякова Ирина	Конструкторский сектор	Ярлык
	001	305632	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект уг
	001	305312	Черников Вячеслав	Конструкторский сектор	Втулка
	001	304132	Виноградова Анна Михайловна	Конструкторский сектор	Призма (УП
	001	300536	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	М4 для ВС-
	001	301122	Герасимов Сергей Александрович	Конструкторский сектор	Тройник каб
	002	305653	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект за
	002	305651	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Вибропрео
	002	305312	Черников Вячеслав	Конструкторский сектор	Втулка уплс
	002	305137	Герасимов Сергей Александрович	Конструкторский сектор	Рычаг МР-5

Поиск

Параметр поискаФИО

Брюз

Найти

	№	Номер	ФИО	Подразделение	
▶	001	467851	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	БС-19-34 ППП
	001	468157	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Блок АСП
	001	468241	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	БС-19-34 АКБ
	001	300004	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция по вк
	001	320113	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция по уп
	001	715451	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Втулка переход. Д
	001	468344	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Коммутатор сигнал
	001	305632	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект упаковки
	001	430309	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция по вы
	001	300536	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	М4 для ВС-14
	001	714567	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Втулка
	001	406234	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Датчик переменн
	001	305653	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект запасны
	001	666666	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Название
	001	111112	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	123
	001	305651	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Вибропреобразов
	001	758449	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Гайка Адаптер АМС
	001	430329	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	ПМ-АО для ТМАМ
	001	758786	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Проводник (МВ-52
	001	680208	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Технология произ
	001	680204	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция по уп

Поиск


Параметр поискаПодразделение

Отдел

Найти

	№	Номер	ФИО	Подразделение	
▶	001	758449	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Гайка Адапте
	001	430329	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	ПМ-АО для Т
	001	468344	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Коммутатор с
	001	666666	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Название
	001	305651	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Вибропреобр
	001	467851	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	БС-19-34 ПП
	001	468157	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Блок АЦП
	001	468241	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	БС-19-34 АК
	001	563351	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Блок АКБ
	001	680204	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция г
	001	680208	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Технология п
	001	758786	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Проводник (Р
	001	411521	Селихов Александр Михайлович	Отдел специальной электроники	БС-9234.01
	001	300004	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция г
	001	715451	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Втулка перех
	001	305632	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Комплект упа
	001	320113	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция г
	001	300536	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	М4 для ВС-14
	001	430309	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Инструкция г
	001	714567	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Втулка
	001	406234	Брюзгин Антон Евгеньевич	Отдел разработки первичных преобразователей	Патрик пера

Регистратор Децимальных номеров АО "Вибро-прибор". Версия 5.0.1

 Взять децимальный номер

Поиск

Параметр поискаСборка/деталь

Жгут

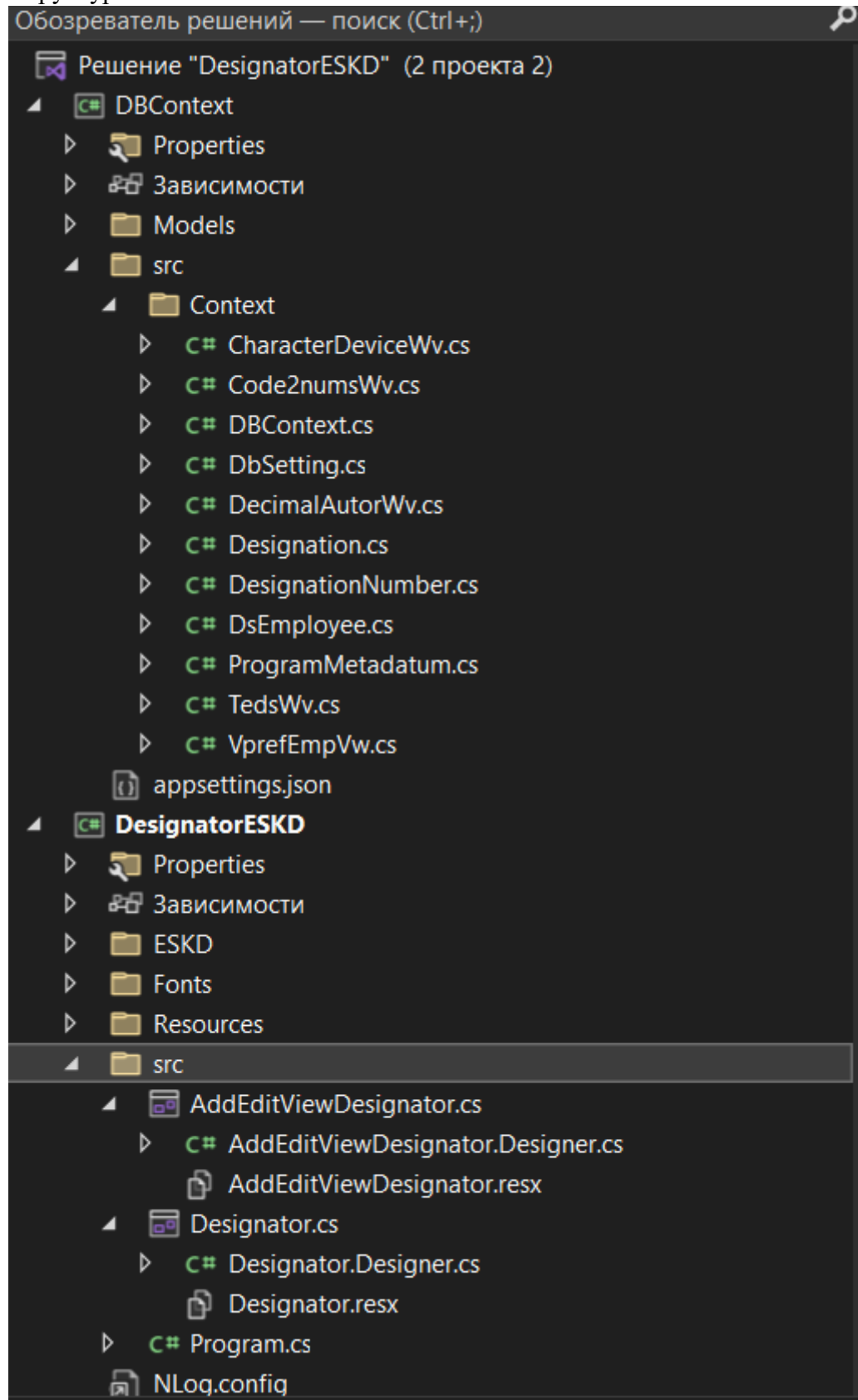
Найти

- ЖЯИУ. 30
- ЖЯИУ. 300004 - Сборочные единицы об
 - ЖЯИУ. 300536 - Сборочные единицы об
 - ЖЯИУ. 300543 - Сборочные единицы об
 - ЖЯИУ. 301111 - Сборочные единицы об
 - ЖЯИУ. 301122 - Сборочные единицы об
 - ЖЯИУ. 301191 - Сборочные единицы об
 - ЖЯИУ. 301261 - Сборочные единицы об
 - ЖЯИУ. 301262 - Сборочные единицы об
 - ЖЯИУ. 301264 - Сборочные единицы об
 - ЖЯИУ. 301314 - Сборочные единицы об
 - ЖЯИУ. 301411 - Сборочные единицы об
 - ЖЯИУ. 301413 - Сборочные единицы об
 - ЖЯИУ. 301511 - Сборочные единицы об
 - ЖЯИУ. 301561 - Сборочные единицы об
 - ЖЯИУ. 301716 - Сборочные единицы об
 - ЖЯИУ. 301732 - Сборочные единицы об
 - ЖЯИУ. 302634 - Сборочные единицы об
 - ЖЯИУ. 302643 - Сборочные единицы об
 - ЖЯИУ. 303659 - Сборочные единицы об
 - ЖЯИУ. 303664 - Сборочные единицы об
 - ЖЯИУ. 303712 - Сборочные единицы об

Подразделение	Наименование детали/сборки	Занято	Сдано в ар
Жгут	Жгут	25.08.2015	
Отдел разработки первичных преобразователей	Жгут Р7-5,0Р-Н-000. Комплект монтажных частей	18.11.2022	
Отдел разработки первичных преобразователей	Жгут 082-5,0Р-Н-000. Комплект монтажных частей	18.11.2022	
Жгут	Жгут	07.07.2021	
Отдел разработки первичных преобразователей	Жгут 642	12.09.2019	
Жгут	Жгут	22.07.2021	
Отдел разработки первичных преобразователей	Жгут 022-1,0Р-П-000	03.09.2020	
Отдел разработки первичных преобразователей	Жгут	29.06.2021	
Жгут	Жгут Ж-Ц-В7-О-xxx-244	30.06.2021	
Жгут	Жгут Ж-Ц-Р7-В7-xxx-245	30.06.2021	
Отдел разработки первичных преобразователей	Жгут Ж-Ц-Р8-В8-xxx-247	30.06.2021	
Жгут	Жгут ЖС-ПД14-248	16.08.2021	
Жгут	Жгут ЖС-ПД14-249	16.08.2021	
Отдел разработки первичных преобразователей	Жгут СИЭЛ-166Д-10Д	30.08.2021	
Жгут	Жгут 642-0,3Р/1,0Р-Н-631	05.10.2021	
Жгут	Жгут ЖЭ-К1-РН6-Р4А-О-С1-xxx-252	05.10.2021	
Жгут	Жгут Ж-П-Р2-В2-XXX-253	14.10.2021	
Отдел разработки первичных преобразователей	Жгут Ж-П-В2-О-xxx-254	25.02.2022	
Жгут	Жгут	21.03.2022	
Жгут	Жгут	25.03.2022	
Жгут	Жгут	29.03.2022	

4.3 Структура проекта и исходный код

Структура:



Файл appsettings.json отвечает за строку подключение к базе данных:

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Host=localhost; Port=5432;Database=db_vp;Username=postgres;Password=1234"
  }
}
```

Файл Nlog.config отвечает за правила логирования в приложении:

```
<?xml version="1.0" encoding="utf-8" ?>
<nlog xmlns="http://www.nlog-project.org/schemas/NLog.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.nlog-project.org/schemas/NLog.xsd NLog.xsd"
  autoReload="true"
  throwExceptions="false"
  internalLogLevel="Off" internalLogFile="c:\temp\nlog-internal.log">

  <!-- optional, add some variables
  https://github.com/nlog/NLog/wiki/Configuration-file#variables
  -->
  <variable name="myvar" value="myvalue"/>

  <!--
  See https://github.com/nlog/nlog/wiki/Configuration-file
  for information on customizing logging rules and outputs.
  -->
  <targets>

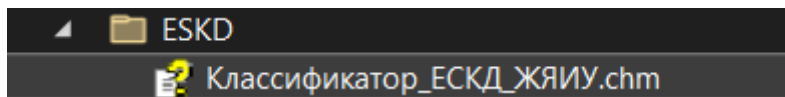
    <!--
    add your targets here
    See https://github.com/nlog/NLog/wiki/Targets for possible targets.
    See https://github.com/nlog/NLog/wiki/Layout-Renderers for the possible layout renderers.
    -->

    <!--
    Write events to a file with the date in the filename.
    <target xsi:type="File" name="f" fileName="${basedir}/logs/${shortdate}.log"
      layout="${longdate} ${uppercase:${level}} ${message}" />
    -->
  </targets>

  <rules>
    <!-- add your logging rules here -->

    <!--
    Write all events with minimal level of Debug (So Debug, Info, Warn, Error and Fatal, but not Trace) to "f"
    <logger name="*" minlevel="Debug" writeTo="f" />
    -->
  </rules>
</nlog>
```

Файл Классификатор ЕСКД является справочником для составления десятичного номера



Классификатор ЕСКД v18.07

Введение

Классы Классификатора ЕСКД

04	Оборудование для обработки резанием, прессовое, литейное и сварочное механическое
06	Оборудование гидромеханических, тепловых, массообменных процессов
10	Оборудование упаковочное и продовольственное
16	Оборудование полиграфическое. Средства оргтехники. Оборудование учебное и технические средства обучения
20	Средства оптико-механические, оптико-электронного наблюдения управления движением. Средства фотометрические, голографические, спектральные, микрофильмирования, фотокиноаппаратура
27	Оборудование сельско- и лесохозяйственное, рыбоводства и водного промысла
28	Оснастка технологическая. Инструмент режущий
29	Оснастка технологическая, кроме инструмента режущего
30	Сборочные единицы общемашиностроительные
31	Подшипники качения
32	Тара. Мебель
33	Изделия культурно-бытового назначения и хозяйственного обихода
36	Суда, судовое оборудование
38	Двигатели (кроме электрических)
40	Средства измерений линейных и угловых размеров, параметров, движения, времени, силы, массы, температуры, давления, расхода, количества и уровня
41	Средства измерений электрических и магнитных величин, ионизирующих излучений, средства интроскопии, определения состава и физико-химических свойств веществ
42	Устройства и системы контроля и регулирования параметров технологических процессов, средства телемеханики, охранной и пожарной сигнализации
43	Микросхемы. Приборы полупроводниковые, электровакуумные пьезоэлектрические квантовой электроники. Резисторы. Соединители. Преобразователи электроэнергии. Средства вторичного электропитания. Модули СВЧ
44	Оборудование технологическое специфическое
45	Средства безрельсового транспорта
46	Средства радиоэлектронные управления, связи, навигации и вычислительной техники
47	Комплексы, агрегаты, машины и аппараты металлургические
48	Оборудование подъемно-транспортное и погрузочно-разгрузочное
49	Арматура трубопроводная
52	Машины электрические вращающиеся
56	Источники электрической энергии, системы электроснабжения. Комплекты электрооборудования
61	Оборудование буровое, горно-шахтное, нефтепромысловое, коксовое. Оборудование для дробления, разделения, окускования и перемешивания

Исходный код основной формы Designator.cs:

```
using DesignatorESKD.DatabaseContext;
using System.Data;
using System.Data.Entity.Core.Common.CommandTrees.ExpressionBuilder;
using NLog;
using System.Windows.Forms;

namespace EF_form
{
    //EF_form - ver.2
    public partial class Designator : Form
    {
        private static readonly Logger logger =
            LogManager.GetCurrentClassLogger();

        public Designator()
        {
            InitializeComponent();
            button1.Click += button1_Click;
            button1.Text = "Выход";
            buttonSearch.Click += buttonSearch_Click;
            AutoSize = false;
            dgvViewDesignation.SelectionMode =
                DataGridViewSelectionMode.FullRowSelect;
        }
    }
}
```

```

        InsertComboBoxSearchParametr();
        TreeView_S.BeforeExpand += TreeView_S_BeforeExpand;
        Load += Designator_Load;

    }

    private void Designator_Load(object sender, EventArgs e)
    {
        // инициализация дерева
        InitializeTreeView();
        // Выполняем установку фокуса после задержки с помощью
метода BeginInvoke

        // выбор стандартного элемента
        comboBox1.SelectedIndex = 0;
    }
    // для параметра поиска
    public void InsertComboBoxSearchParametr()
    {
        try
        {
            comboBox1.Items.Insert(0, "Номер");
            comboBox1.Items.Insert(1, "ФИО");
            comboBox1.Items.Insert(2, "Подразделение");
            comboBox1.Items.Insert(3, "Сборка/деталь");
        }
        catch (Exception ex)
        {
            logger.Error(ex, "Ошибка при вставке в
combobox. Исключение: {exceptionMessage}", ex.Message);
            CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка загрузки приложения");
            customMessageBox.ShowDialog();
            this.Close();
        }
    }
    private void button1_Click(object? sender, EventArgs e)
    {
        Application.Exit();
        logger.Info("Пользователь вышел из приложения\n");
    }
    private void comboBox1_SelectedIndexChanged(object sender,
EventArgs e)
    {
        textBoxNumber.Text = string.Empty;
    }
    // кнопка "Поиск"
    private void buttonSearch_Click(object sender, EventArgs e)
    {
        string searchParameter = textBoxNumber.Text;
        string selectedComboBoxItem =
comboBox1.SelectedItem.ToString();
        using (var dbContext = new DBcontext())
        {
            try
            {
                var query =
dbContext.DecimalAutorWvs.AsQueryable();

                switch (selectedComboBoxItem)
                {
                    case "Номер":

```



```

        if
        (searchParameter.Length == 6)
        {
            query =
            query.Where(ddnu => ddnv.DecimalCharacteristics == searchParameter);
        }
        else if
        (searchParameter.Length < 6 && searchParameter.Length >= 2)
        {
            query =
            query.Where(ddnu => ddnv.DecimalCharacteristics.StartsWith(searchParameter));
        }
        else
        {
            MessageBox.Show("Номер состоит из 6 цифр \n Вы можете ввести от 2 до 6
            для поиска");

            dgvViewDesignation.DataSource = null;

            return;
        }
        break;
    case "ФИО":
        query = query.Where(ddnu
        => ddnv.DisplayName.StartsWith(searchParameter));
        break;
    case "Сборка/деталь":
        query = query.Where(ddnu
        => ddnv.PartName.StartsWith(searchParameter));
        break;
    case "Подразделение":
        query = query.Where(ddnu
        => ddnv.Name.StartsWith(searchParameter)).OrderBy(ddnu =>
        ddnv.DecimalCharacteristics);
        break;
    default:
        MessageBox.Show("Недопустимый выбор в ComboBox.");
        return;
    }
    var designation = query.Select(dawv =>
new
    {
        dawv.Number,
        dawv.DecimalCharacteristics,
        dawv.DisplayName,
        dawv.Name,
        dawv.PartName,
        dawv.Occupied,
        dawv.Archived,
        dawv.IsArchived,
        dawv.Comment,
    })
    .OrderBy(ddnum => ddnum.Number)
    .ToList();

    if (designation.Count > 0)
    {
        dgvViewDesignation.DataSource =
        designation;
    }
}

```

```

// заполнение DGV поисковыми
данными
GenerateDGVCColumns(true);
}
else
{
    MessageBox.Show("Результаты не
найденны.", "Поиск", MessageBoxButtons.OK, MessageBoxIcon.Information);
    dgvViewDesignation.DataSource =
null;
}
}
#pragma warning disable CS0168 // Переменная объявлена, но не используется
catch (Exception ex)
{
    logger.Fatal(ex, "Ошибка при поиске в
базе. Исключение: {exceptionMessage}", ex.Message);
    CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка загрузки базы данных");
    customMessageBox.ShowDialog();
    this.Close();
}
#pragma warning restore CS0168 // Переменная объявлена, но не используется
}
private void TreeView_S_AfterSelect(object sender,
TreeViewEventArgs e)
{
    TreeNode selectedNode = e.Node;

    TreeView_S.SelectedNode = selectedNode;
    TreeView_S.Focus();

    // Если выбран корневой узел
    if (selectedNode.Parent == null)
    {
        // Проверяем наличие дочерних узлов
        if (selectedNode.Nodes.Count > 0)
        {
            TreeNode firstChildNode =
selectedNode.Nodes[0];
            if (firstChildNode.Tag != null &&
firstChildNode.Tag is string tagValue)
            {
                ViewDesignatorNumber(tagValue);
            }
        }
        // Если выбран дочерний узел
        else
        {
            if (selectedNode.Tag != null &&
selectedNode.Tag is string tagValue)
            {
                ViewDesignatorNumber(tagValue);
            }
        }
    }
    private void TreeView_S_BeforeExpand(object sender,
TreeViewCancelEventArgs e)
    {
        TreeNode selectedNode = e.Node;

```

```

        // Если выбран корневой узел
        if (selectedNode.Parent == null)
        {
            // Проверяем наличие дочерних узлов
            if (selectedNode.Nodes.Count > 0)
            {
                TreeNode firstChildNode =
selectedNode.Nodes[0];
                if (firstChildNode.Tag != null &&
firstChildNode.Tag is string tagValue)
                {
                    ViewDesignatorNumber(tagValue);
                    // Установка фокуса на первый
дочерний узел
                    TreeView_S.SelectedNode =
firstChildNode;
                }
            }
        }

        // TreeView для раскрытия
        private void ViewDesignatorNumber(string search_parameter)
        {
            using (var dbContext = new DBcontext())
            {
                try
                {
                    var designation =
dbContext.DecimalAutorWvs.Where(dawv => dawv.DecimalCharacteristics ==
search_parameter)

                    .Select(dawv => new
{
                        dawv.Number,
                        dawv.DecimalCharacteristics,
                        dawv.DisplayName,
                        dawv.Name,
                        dawv.PartName,
                        dawv.Occupied,
                        dawv.Archived,
                        dawv.IsArchived,
                        dawv.Comment,
                    })

                    .OrderBy(dawv => dawv.Number)

                    .ToList();

                    if (designation.Count > 0)
                    {
                        dgvViewDesignation.DataSource =
designation;
                    }
                }
            }
        }
    }
}

```

```

GenerateDGVCOLUMNS(false);

foreach (DataGridViewRow row in
dgvViewDesignation.Rows)
{
    row.Cells["Number"].Tag
= row.Cells["Number"].Value;
}
foreach (DataGridViewRow rowA in
dgvViewDesignation.Rows)
{
    rowA.Cells["Archived"].Tag = rowA.Cells["Archived"].Value;
}
foreach (DataGridViewRow rowB in
dgvViewDesignation.Rows)
{
    rowB.Cells["DisplayName"].Tag = rowB.Cells["DisplayName"].Value;
}
else
{
    dgvViewDesignation.DataSource =
null;
}
}

#pragma warning disable CS0168 // Переменная объявлена, но не используется
catch (Exception ex)
{
    logger.Error(ex, "Ошибка при загрузке
данных в viewDesNum_DVG. Исключение: {exceptionMessage}", ex.Message);
    CustomMessageBox customMessageBox = new
CustomMessageBox("Возникла ошибка при подключении к базе данных десятичных
номеров. \nОбратитесь к администратору.");
    customMessageBox.ShowDialog();

    this.Close();
}
#pragma warning restore CS0168 // Переменная объявлена, но не используется
}

private void DgvViewDesignation_CellDoubleClick(object sender,
DataGridViewCellEventArgs e)
{
    if ((int)e.RowIndex < 0) return;

    // Если номер уже сдан в архив
    if
(dgvViewDesignation.Rows[e.RowIndex].Cells["Archived"].Tag != null)
{
        MessageBox.Show("Нельзя редактировать запись,
которая внесена в архив.\nЕсли Вам необходимо отредактировать данную запись
обратитесь к разработчику программы.", "Информация", MessageBoxButtons.OK,
MessageBoxIcon.Information);
        return;
    }
    int decNum =
Convert.ToInt32(TreeView_S.SelectedNode.Tag);
    // передача значения номера взятия десятилки
    ЖЯИУ.[1].666666

```

```

        if
(dgvViewDesignation.Rows[e.RowIndex].Cells["Number"].Tag is int number_id)
        {
            if
(dgvViewDesignation.Rows[e.RowIndex].Cells["DisplayName"].Tag is string
employeeFio)
            {
                AddEditViewDesignator p_AddEditfromDgv
= new AddEditViewDesignator(decNum, true, number_id, employeeFio);
                p_AddEditfromDgv.ShowDialog(this);

                ViewDesignatorNumber(decNum.ToString());
                TreeView_S.Focus();
            }
        }
    }
    // Отображение dgv
    public void GenerateDGVColumns(bool print_dec_char)
    {
        try
        {
            dgvViewDesignation.ColumnHeadersDefaultCellStyle = new
DataGridViewCellStyle()
            {
                BackColor = Color.Azure,
                Alignment =
DataGridViewContentAlignment.BottomCenter
            };
            dgvViewDesignation.AutoSizeColumnsMode =
DataGridViewAutoSizeColumnsMode.AllCellsExceptHeader;
            dgvViewDesignation.DefaultCellStyle.WrapMode =
DataGridViewTriState.True;
            dgvViewDesignation.AutoSizeColumnsMode();
            // Аннулирование номера

            dgvViewDesignation.Columns["IsArchived"].Visible = false;
            dgvViewDesignation.Columns["Number"].HeaderText
= "№ п/п";

            dgvViewDesignation.Columns["Number"].DefaultCellStyle = new
DataGridViewCellStyle() { Format = "000", Font = new Font("PT Sans", 10F,
System.Drawing.FontStyle.Bold) };

            dgvViewDesignation.Columns["DecimalCharacteristics"].Visible =
print_dec_char;

            dgvViewDesignation.Columns["DecimalCharacteristics"].HeaderText =
"Номер";

            dgvViewDesignation.Columns["Comment"].HeaderText = "Комментарий";

            dgvViewDesignation.Columns["PartName"].HeaderText = "Наименование
детали/сборки";

            dgvViewDesignation.Columns["Occupied"].HeaderText = "Занято";

            dgvViewDesignation.Columns["Archived"].HeaderText = "Сдано в архив";

            dgvViewDesignation.Columns["DisplayName"].HeaderText = "ФИО";
            dgvViewDesignation.Columns["Name"].HeaderText =
"Подразделение";
        }
    }

```

```

        catch (Exception ex)
        {
            logger.Error(ex, "Ошибка при загрузке данных в
genDVG. Исключение: {exceptionMessage}", ex.Message);
        }
    }
    // Инициализация дерева
    private void InitializeTreeView()
    {
        try
        {
            TreeView_S.Nodes.Clear();
            TreeView_S.Dock = DockStyle.None;
            Controls.Add(TreeView_S);
            TreeView_S.Scrollable = true;
            using (var dbContext = new DBcontext())
            {
                var result =
dbContext.Code2numsWvs.ToList();

                foreach (var item in result)
                {

                    var items =
dbContext.CharacterDeviceWvs
                        .Where(c =>
c.FullNum.StartsWith(item.FirstNum))
                        .ToList();
                    // Заполнение корня:
ЖЯИУ          30
                    TreeNode parentNode =
TreeView_S.Nodes.Add($"{item.Code} {item.FirstNum}");
                    Font boldFont = new
Font(TreeView_S.Font.FontFamily, 12, FontStyle.Bold);
                    parentNode.NodeFont = boldFont;

                    foreach (var childItem in items)
                    {
                        // ограничение для
вывода названия в листьях дерева
                        string decName =
childItem.DecName.Length > 20 ? childItem.DecName.Substring(0, 20) :
childItem.DecName;
                        // заполнение листьев
ЖИЯУ          300000          ИМЯ

                        TreeNode childNode =
parentNode.Nodes.Add($"{childItem.Code} {childItem.FullNum} - {decName}");
                        childNode.Tag =
childItem.FullNum; // Установка тега для использования в событии AfterSelect
                    }
                }
            }
        }
        catch (Exception ex)
        {
            logger.Error(ex, "Произошла ошибка при
инициализации дерева: {ErrorMessage}", ex.Message);
            CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка загрузки приложения");
            this.Close();
        }
    }
}

```

```

e)
        private void BtnAddDesignation_Click(object sender, EventArgs
        {
            try
            {
                // добавление номера с дерева
                int listik;
                if (TreeView_S.SelectedNode != null &&
TreeView_S.SelectedNode.Tag != null)
                {
                    listik =
Convert.ToInt32(TreeView_S.SelectedNode.Tag);
                }
                else
                {
                    // обработка события если на форме
ничего не выделено
                    listik = 0;
                }
                // отключение фокусировки на кнопке
                ActiveControl = null;
                AddEditViewDesignator p_AddEditDesignator = new
AddEditViewDesignator(listik, false);
                p_AddEditDesignator.ShowDialog(this);

                TreeView_S.Focus();
                ViewDesignatorNumber(listik.ToString());
            }
            catch (Exception ex)
            {
                logger.Error(ex, "Произошла ошибка взятия
номера: {ErrorMessage}", ex.Message);
                CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка загрузки приложения\nОбратитесь к системному
администратору");
            }
        }

        // Прибавление 3 часов для времени взятия
        // Отображение в DGV лагает на -3 часа (?)
        private void dgvViewDesignation_CellFormatting(object sender,
DataGridViewCellFormattingEventArgs e)
        {
            if (dgvViewDesignation.Columns[e.ColumnIndex].Name ==
"Occupied" || dgvViewDesignation.Columns[e.ColumnIndex].Name == "Archived" &&
e.Value != null)
            {
                if (e.Value is DateTime occupied)
                {
                    // Добавляем 3 часа к значению поля
Occupied
                    DateTime adjustedOccupied =
occupied.AddHours(3);
                    e.Value = adjustedOccupied;
                }
            }
        }

        private void dgvViewDesignation_RowPrePaint(object sender,
DataGridViewRowPrePaintEventArgs e)
        {
            DataGridViewRow row =
dgvViewDesignation.Rows[e.RowIndex];

```

```

        if
(!string.IsNullOrEmpty(row.Cells["Archived"].Value?.ToString()))
        {
            row.DefaultCellStyle.BackColor =
Color.LightGray;
        }
    }
    private void button2_Click(object sender, EventArgs e)
    {
        InitializeTreeView();
    }
    public class CustomMessageBox : Form
    {
        public CustomMessageBox(string message)
        {
            AddControls(message);
            ControlBox = false;
        }

        private void AddControls(string message)
        {
            // Создание и настройка контролов формы
            Label label = new Label()
            {
                Text = message,
                Location = new Point(15, 50),
                AutoSize = true,
                TextAlign =
ContentAlignment.MiddleCenter,
                Font = new Font(Font.FontFamily, 12,
FontStyle.Bold)
            };

            Button closeButton = new Button()
            {
                Text = "Закреть",
                Size = new Size(80, 30),
                Font = new Font(Font.FontFamily, 12,
FontStyle.Regular)
            };
            Size = new Size(415, 200);
            closeButton.Location = new
Point(ClientSize.Width - closeButton.Width, ClientSize.Height -
closeButton.Height + 30);
            closeButton.Click += (sender, e) =>
            {
                Close(); // Закреть форму
            }
        }

        // Добавление контролов на форму
        Controls.Add(label);
        Controls.Add(closeButton);
        // Настройка размеров и отображения формы
        FormBorderStyle = FormBorderStyle.FixedDialog;
        StartPosition = FormStartPosition.CenterScreen;
    }

    private void BtnAddDesignation_Click_1(object sender, EventArgs e)
    {
        try
        {
            // добавление номера с дерева

```



```

        int listik;
        if (TreeView_S.SelectedNode != null &&
TreeView_S.SelectedNode.Tag != null)
        {
            listik = Convert.ToInt32(TreeView_S.SelectedNode.Tag);
        }
        else
        {
            // обработка события если на форме ничего не выделено
            listik = 0;
        }
        // отключение фокусировки на кнопке
        ActiveControl = null;
        AddEditViewDesignator p_AddEditDesignator = new
AddEditViewDesignator(listik, false);
        p_AddEditDesignator.ShowDialog(this);

        TreeView_S.Focus();
        ViewDesignatorNumber(listik.ToString());
    }
    catch (Exception ex)
    {
        logger.Error(ex, "Произошла ошибка взятии номера:
{ErrorMessage}", ex.Message);
        CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка загрузки приложения\nОбратитесь к системному
администратору");
    }
}
}
}

```

Исходный код формы AddEditViewDesignator используемой для взятия и архивации десятичных номеров:

```

using DesignatorESKD.DatabaseContext;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.Logging;
using Npgsql;
using NpgsqlTypes;
using System.Data;
using Microsoft.EntityFrameworkCore.Diagnostics;
#pragma warning disable CS8629 // Тип значения, допускающего NULL, может быть
NULL.
#pragma warning disable CS8600 // Преобразование литерала, допускающего
значение NULL или возможного значения NULL в тип, не допускающий значение
NULL.

namespace EF_form
{
    public partial class AddEditViewDesignator : Form

```

```

{
    // константа для организации // должна быть перемещена в
    conf file
    const int organization_id = 1;

    // Создайте экземпляр логгера для вашего класса или контекста

    // для хранения десятичного номера
    private int m_DsDesignationId = 0;

    // для хранения порядкового номера десятичной
    int number_id = 0;
    string employeeDisplay = "";
    // для скрывания кнопок при архивации
    bool m_Flag;

    // логин пользователя компьютера
    static string localEmployee = Environment.UserName;
    long authorOidParam = 0;

    // имя aebruzgin взято для тестов
    //string userName = "aebruzgin";
    string userName = localEmployee;
    bool checkAuthorOid;
    public AddEditViewDesignator(int decimal_number, bool p_Flag)
    {
        // false - treeView
        InitializeComponent();
        m_DsDesignationId = decimal_number;
        m_Flag = p_Flag;
        try
        {
            Init(m_Flag);
        }
        catch (Exception ex)
        {
            logger.Error(ex, "Ошибка Init в AddEdit.
Исключение: {exceptionMessage}", ex.Message);
            CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка инициализации приложения");
            customMessageBox.ShowDialog();
            this.Close();
        }
    }
    // Перегрузка для считывания порядкового номера десятичной

    public AddEditViewDesignator(int decimal_number, bool p_Flag,
int number, string employeeFio)
    {
        // true - dgv
        InitializeComponent();
        m_DsDesignationId = decimal_number;
        // для отображения ФИО при взятии
        employeeDisplay = employeeFio;

        m_Flag = p_Flag;
        // номер взятия 001
        number_id = number;

        button1.Text = "Сдать";
        Text = "Сдача в архив";
        dtpArchived.Focus();
    }
}

```

```

        try
        {
            Init(m_Flag);
        }
        catch (Exception ex)
        {
            logger.Error(ex, "Ошибка Init в AddEdit.
Исключение: {exceptionMessage}", ex.Message);
            CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка инициализации приложения ы");
            customMessageBox.ShowDialog();
            this.Close();
        }
    }

    private DBcontext CreateDbContext()
    {
        // Здесь создается и возвращается новый экземпляр
контекста базы данных
        return new DBcontext();
    }

    private void Init(bool m_Flag)
    {
        // Если данные взяты с DGV => их нельзя изменять,
только сдавать в архив
        ///|                                     |\\
        // дата сдачи видна только с DGV == true
treeView==false

        dtpArchived.Visible = m_Flag;
        // блокировка вводимых компонентов
        rtbDecimalName.ReadOnly = m_Flag; tbDecemal.ReadOnly =
m_Flag; tbComment.ReadOnly = m_Flag; label8.Visible = m_Flag;

        cdPartName.Enabled = !m_Flag;
        // скрыть кнопку "другой пользователь" при архивации
        otherUser.Visible = !m_Flag;
        cdSurname.Enabled = false;
        dtpOccupied.Enabled = !m_Flag;
        dtpOccupied.Visible = !m_Flag;
        label7.Visible = !m_Flag;
        tbDepartment.Visible = !m_Flag;
        label4.Visible = !m_Flag;
        label6.Visible = !m_Flag;
        tbComment.Visible = !m_Flag;
        // идет из автора//
        tbDepartment.ReadOnly = true;
        //
        if (m_DsDesignationId == 0)
        {
            tbDecemal.Text = string.Empty;
        }
        else
        {
            tbDecemal.Text =
Convert.ToString(m_DsDesignationId);
        }
        using (var dbContext = CreateDbContext())
        {
            var decimalAuthors =
dbContext.DecimalAutorWvs.ToList();
            var vpdecimalAuthors =
dbContext.VprefEmpVws.ToList();

```

```

var vpDisplayNames =
dbContext.VprefEmpVws.Select(x => x.DisplayName).ToList();
var displayNames = decimalAuthors.Select(x =>
x.DisplayName).ToList();
// оставить список для кнопки "Другой
сотрудник"
if (m_Flag == true)
{
    cdSurname.Enabled = false;
    cdSurname.Text = employeeDisplay;
}
else
{
    cdSurname.DataSource =
vpDisplayNames.ToList();

var t =
dbContext.VprefEmpVws.FirstOrDefault(x => x.Login == userName)?.DisplayName;

int defaultIndex =
vpDisplayNames.FindIndex(x => x == t);

cdSurname.SelectedIndex = defaultIndex;
}
UpdateTbDepartament(vpdecimalAuthors);
// Обработчик события изменения выбранного
значения в комбобоксе
cdSurname.SelectedIndexChanged += (sender, e)
=>
{
    UpdateTbDepartament(vpdecimalAuthors);
};
// ----- //
// Заполнение cdPartName значениями из свойства
PartName
string designationId =
m_DsDesignationId.ToString();
var partNames =
GetFilteredPartNames(decimalAuthors, designationId);
cdPartName.DataSource = partNames;

// достаем описание
var rtb = dbContext.CharacterDeviceWws
.Where(x => x.FullNum ==
m_DsDesignationId.ToString())
.Select(x => x.DecName)
.FirstOrDefault();
rtbDecimalName.Text = rtb ?? string.Empty;
// Обработчик события изменения текста в
tbDecemal
tbDecemal.TextChanged += (sender, e) =>
{
    if (tbDecemal.Text.Length == 6)
    {
        using (var dbContext =
CreateDbContext())
        {
            designationId =
tbDecemal.Text;
            partNames =
GetFilteredPartNames(decimalAuthors, designationId);
            cdPartName.DataSource = partNames;

```

```

rtb =
dbContext.CharacterDeviceWvs
rtb =
.Where(x
=> x.FullNum == tbDecemal.Text)
rtb =
.Select(x
=> x.DecName)

.FirstOrDefault();
}
rtbDecimalName.Text = rtb ??

string.Empty;
}
else
{
// деталь сборки
cdPartName.DataSource = null;
// описание характеристики
rtbDecimalName.Text =

string.Empty;
}
};
}
}

private List<string> GetFilteredPartNames(List<DecimalAutorWv>
decimalAuthors, string designationId)
{
#pragma warning disable CS8619 // Допустимость значения NULL для ссылочных
типов в значении не соответствует целевому типу.
return decimalAuthors.Where(x =>
x.DecimalCharacteristics == designationId).Select(x => x.PartName).ToList();
#pragma warning restore CS8619 // Допустимость значения NULL для ссылочных
типов в значении не соответствует целевому типу.
}

private void UpdateTbDepartment(List<VprefEmpVw>
vpDisplayNames)
{
if (cdSurname.SelectedIndex != -1)
{
string selectedDisplayName =
cdSurname.SelectedItem.ToString();
var selectedAuthor =
vpDisplayNames.FirstOrDefault(x => x.DisplayName == selectedDisplayName);
tbDepartment.Text =
selectedAuthor?.DepartmentName;
}
}

private bool ChechRecord()
{
if ((tbDecemal.Text == "") || (tbDepartment.Text ==
"") || (cdPartName.Text == ""))
{
MessageBox.Show("Для взятия кода необходимо
заполнить все строковые поля,\n а так-же выбрать дату, если она отличается от
текущей", "Ошибка", MessageBoxButtons.OK, MessageBoxIcon.Error);
return false;
}
else return true;
}
}

```

```

private bool CheckNum()
{
    string input = tbDecemal.Text;
    int designator_check;

    if (input.Length == 6 && int.TryParse(input,
out designator_check))
    {
        return true;
    }
    else
    {
        string message = "Децимальный номер
введен неверно";
        CustomMessageBox customMessageBox = new
CustomMessageBox(message);
        customMessageBox.ShowDialog();
        return false;
    }
}

private void button1_Click(object sender, EventArgs e)
{
    TimeZoneInfo systemTimeZone = TimeZoneInfo.Local;
    string currentTimeZone = systemTimeZone.DisplayName;

    // MessageBox.Show("Текущий часовой пояс: " +
currentTimeZone, "Информация о часовом поясе");
    // взятие с дерева
    if (m_Flag == false)
    {
        // проверка на пустые поля
        try
        {
            AddRecord();
        }
        catch (Exception ex)
        {
            logger.Error(ex, "Ошибка
ChechRecord/CheckNum в AddEdit. Исключение: {exceptionMessage}", ex.Message);
            CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка проверки данных в приложении");
            customMessageBox.ShowDialog();
            this.Close();
        }
    }
    // архивация с dgv
    else
    {
        ArchiveRecord();
    }
    this.Close();
}

private void ArchiveRecord()
{
    // -- Архивирует запись с помощью номера взятия и
децимального номера --
    // 6 цифр
    string tbDecemalValue = tbDecemal.Text;
    // дата сдачи с формы

```

```

        DateTime archivedParam = dtpArchived.Value;
        using (var dbContext = new DBcontext())
        {
            string sql = $"SELECT
designation.archivation_number({number_id}, '{archivedParam}',
'{tbDecemalValue}')";

            try
            {
                dbContext.Database.ExecuteSqlRaw(sql);
                logger.Info($"| Архивация: ( Децимальный
номер: {tbDecemalValue}, {number_id}, '{archivedParam}')");

                var name_org =
dbContext.CharacterDeviceWvs.Where(d => d.FullNum == tbDecemal.Text).Select(d
=> d.Code).FirstOrDefault();

                string formattedNumberId =
number_id.ToString().PadLeft(3, '0'); // Добавление ведущих нулей до ширины 3

                string message = $"Децимальный номер:
{name_org}{tbDecemal.Text}.{formattedNumberId}\nСдан в архив";
#pragma warning restore CS8602 // Разыменование вероятной пустой ссылки.
                CustomMessageBox customMessageBox = new
CustomMessageBox(message);

                customMessageBox.ShowDialog();
            }
            catch (Exception ex)
            {
                logger.Error(ex,
$"Архивация: ({tbDecemalValue}, number: {number_id}, '{archivedParam}')\"";
                ex.Message);

                CustomMessageBox customMessageBox = new
CustomMessageBox("Ошибка архивации");

                customMessageBox.ShowDialog();
            }
        }
    }

    private void AddRecord()
    {
        // -- Взятие децимального номера -- //
        string cdSurnameValue =
cdSurname.SelectedItem?.ToString();
        string rtbDecimalNameValue = rtbDecimalName.Text;
        string tbDecemalValue = tbDecemal.Text;
        string partNameParam = cdPartName.Text;
        DateTime occupiedParam = dtpOccupied.Value;
        string commentParam = tbComment.Text;

        using (var dbContext = new DBcontext())
        {
            // вставка во взятие происходит с помощью
            // внешнего ключа на cid децимального номера
            // Значение id от децимального номера
            long designationOidParam =

dbContext.Designations
                .Where(d => d.DecimalCharacteristics ==
tbDecemalValue)
                .Select(d => d.Cid)
                .FirstOrDefault();

            // значение cid от автора
            localEmployee

            if (checkAuthorOid == false) {

```

```

        authorOidParam =
(long)dbContext.VprefEmpVws.Where(d => d.Login == userName).Select(d =>
d.Dscid).FirstOrDefault();
    }
    if (checkAuthorOid == true)
    {
        // Логин запросом, в случае если
человек выбирает окошко "Другой пользователь"

        authorOidParam =
(long)dbContext.VprefEmpVws.Where(d => d.DisplayName ==
cdSurnameValue).Select(d => d.Dscid).FirstOrDefault();
    }
    if (authorOidParam == 0)
    {
        MessageBox.Show("Сотрудник не найден,
обратитесь к системного администратору", "authorOidParamError");
    }
    // если введен новый десятичный номер
var newDesignatorCid = new
NpgsqlParameter("@result", NpgsqlDbType.Text)
{
    Direction = ParameterDirection.Output
};

    if (designationOidParam == 0)
    {
        try
        {

            dbContext.Database.ExecuteSqlRaw("SELECT designation.add_new_number
(@decimalCharacteristics, @decimalName, @organizationOid)",
                new
NpgsqlParameter("@decimalCharacteristics", tbDecemalValue),
                new
NpgsqlParameter("@decimalName", rtbDecimalNameValue),
                new
NpgsqlParameter("@organizationOid", organization_id), newDesignatorCid);

            logger.Info($"Добавление: (|
Децимальный номер: {tbDecemalValue}, 'Имя: '{rtbDecimalNameValue}',
Организация: '{organization_id}'| )");

#pragma warning disable CS8602 // Разыменованная вероятная пустая ссылка.
                string getNewCid =
newDesignatorCid.Value.ToString();
#pragma warning restore CS8602 // Разыменованная вероятная пустая ссылка.
                designationOidParam =
Convert.ToInt64(getNewCid);
            }
            catch (Exception ex)
            {
                logger.Error(ex, $"Добавление:
(| Децимальный номер:{tbDecemalValue}, 'Имя: '{rtbDecimalNameValue}',
Организация: '{organization_id}'|)", ex.Message);
                CustomMessageBox
customMessageBoxAdd = new CustomMessageBox("Ошибка добавления");

                customMessageBoxAdd.ShowDialog();
            }
        }

        // взятие самого десятичного номера

```



```

        var resultParam = new
NpgsqlParameter("@result", NpgsqlDbType.Text)
        {
            Direction = ParameterDirection.Output
        };
        try
        {
            dbContext.Database.ExecuteSqlRaw("SELECT
designation.insert_designation_number(@partName, @occupied, @comment,
@designationOid, @authorOid)",
            new NpgsqlParameter("@partName",
partNameParam),
            new NpgsqlParameter("@occupied",
occupiedParam),
            new NpgsqlParameter("@comment",
commentParam),
            new
NpgsqlParameter("@designationOid", designationOidParam),
            new
NpgsqlParameter("@authorOid", authorOidParam),
            resultParam);
            logger.Info($"Взятие номера: (|
Дecimalный номер: {tbDecemalValue}, part_name: {partNameParam},
occupied:'{occupiedParam}', comment: '{commentParam}',
des_oid:'{designationOidParam}, aut_oid: '{authorOidParam}, номер:
{resultParam.Value.ToString()} '");
        }
        catch (Exception ex)
        {
            logger.Error(ex, $"Взятие номера: (|
Дecimalный номер: {tbDecemalValue}, part_name: {partNameParam},
occupied:'{occupiedParam}', comment: '{commentParam}',
des_oid:'{designationOidParam}, aut_oid: '{authorOidParam}, номер:
{resultParam.Value.ToString()} ')", ex.Message);
            CustomMessageBox customMessageBoxTake =
new CustomMessageBox("Ошибка взятия номера");
            customMessageBoxTake.ShowDialog();
        }
#pragma warning disable CS8602 // Разыменованная вероятная пустая ссылка.
        authorOidParam =
(long)dbContext.VprefEmpVws.Where(d => d.Login == userName).Select(d =>
d.Dscid).FirstOrDefault();
        string message;
        var name_org =
dbContext.CharacterDeviceWvs.Where(d => d.FullNum == tbDecemal.Text).Select(d
=> d.Code).FirstOrDefault();
        string result = resultParam.Value.ToString();
        try
        {
            if (result == "insert_error")
            {
                message = "Ошибка вставки.
\nОбратитесь к системному администратору.";
            }
            else if (result == "over_number_error")
            {
                message = "Количество номеров
превысило лимит. \nОбратитесь к системному администратору.";
            }
            else
            {
                message = $"Дecimalный номер:
{name_org}{tbDecemal.Text}{result}";
            }
        }

```

```

        }
    }
    catch (Exception)
    {
        message = "Неизвестная ошибка.
\nОбратитесь к системному администратору.";
    }

#pragma warning restore CS8602 // Разыменование вероятной пустой ссылки.

CustomMessageBox customMessageBox = new
CustomMessageBox(message);
customMessageBox.ShowDialog();
    }

    // открытие файла ЕСКД
    // нужно поменять при выпуске exe файла
    private void btnOpenClassificier_Click(object sender,
EventArgs e)
    {
        var path = Path.Combine(Application.StartupPath,
"ESKD\\Классификатор_ЕСКД_ЖЯИУ.chm");

        if (File.Exists(path) )
        {
            Help.ShowHelp(ParentForm, path,
HelpNavigator.TableOfContents);
        }
        else
        {
            CustomMessageBox customMessageBox = new
CustomMessageBox("файл ЕСКД не найден");
            customMessageBox.ShowDialog();
        }
    }
    private void otherUser_CheckedChanged(object sender, EventArgs
e)
    {
        CheckBox checkBox = (CheckBox)sender; // приводим
отправителя к элементу типа CheckBox

        cdSurname.Enabled = checkBox.Checked;
        checkAuthorOid = checkBox.Checked;
    }
}

public class CustomMessageBox : Form
{
    public CustomMessageBox(string message)
    {
        AddControls(message);
        ControlBox = false;
    }
    private void AddControls(string message)
    {
        // Создание и настройка контролов формы
        Label label = new Label()
        {
            Text = message,
            Location = new Point(15, 50),
            AutoSize = true,
            TextAlign = ContentAlignment.MiddleCenter,

```

```

FontStyle.Bold)
Font = new Font(Font.FontFamily, 13,
);

Button closeButton = new Button()
{
    Text = "Заккрыть",
    Size = new Size(80, 30),
    Font = new Font(Font.FontFamily, 12,
FontStyle.Regular)
};
Size = new Size(415, 200);
closeButton.Location = new Point(ClientSize.Width -
closeButton.Width , ClientSize.Height - closeButton.Height + 30);
closeButton.Click += (sender, e) => Close();
// Добавление контролов на форму
Controls.Add(label);
Controls.Add(closeButton);
// Настройка размеров и отображения формы
FormBorderStyle = FormBorderStyle.FixedDialog;
StartPosition = FormStartPosition.CenterScreen;
}
}
}

```