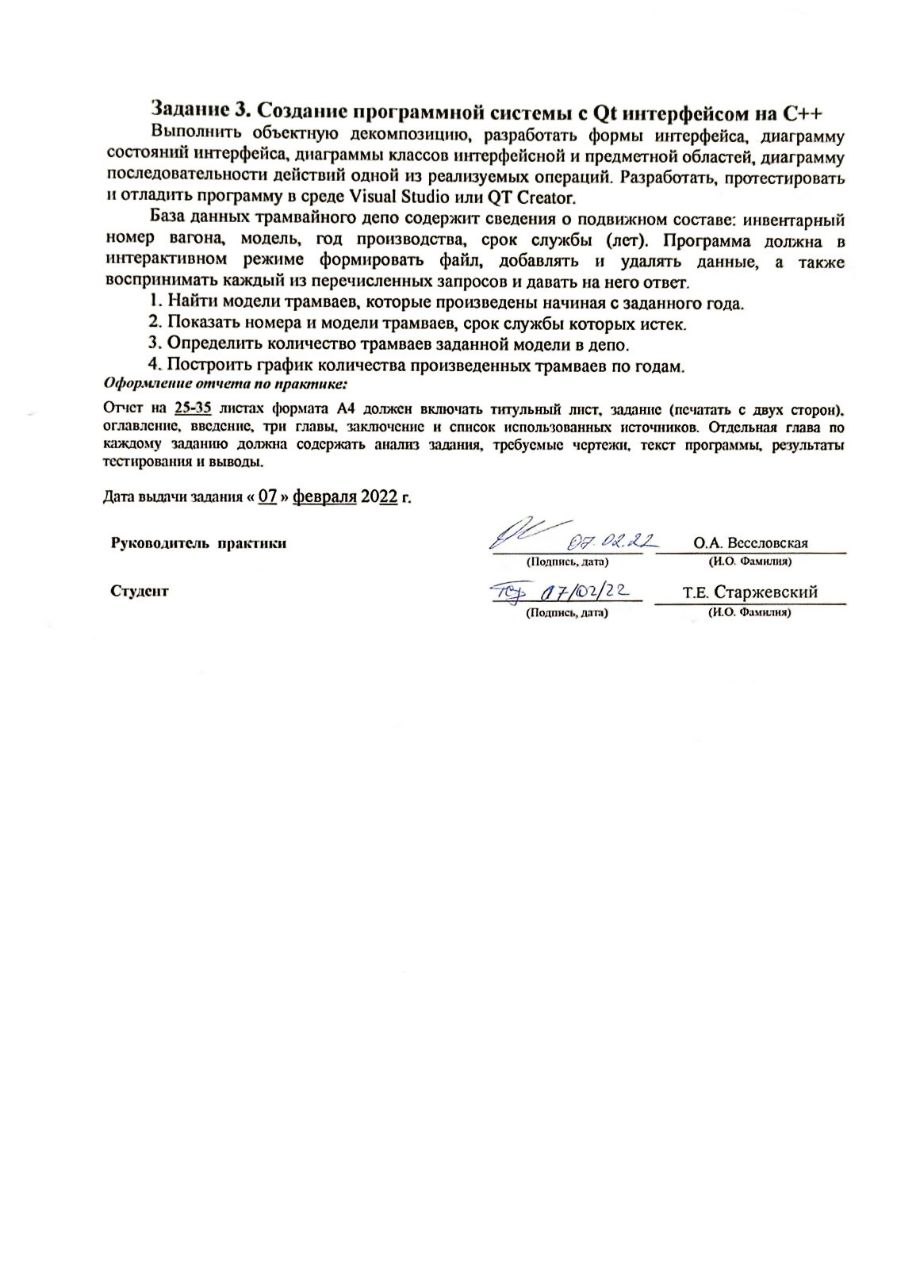
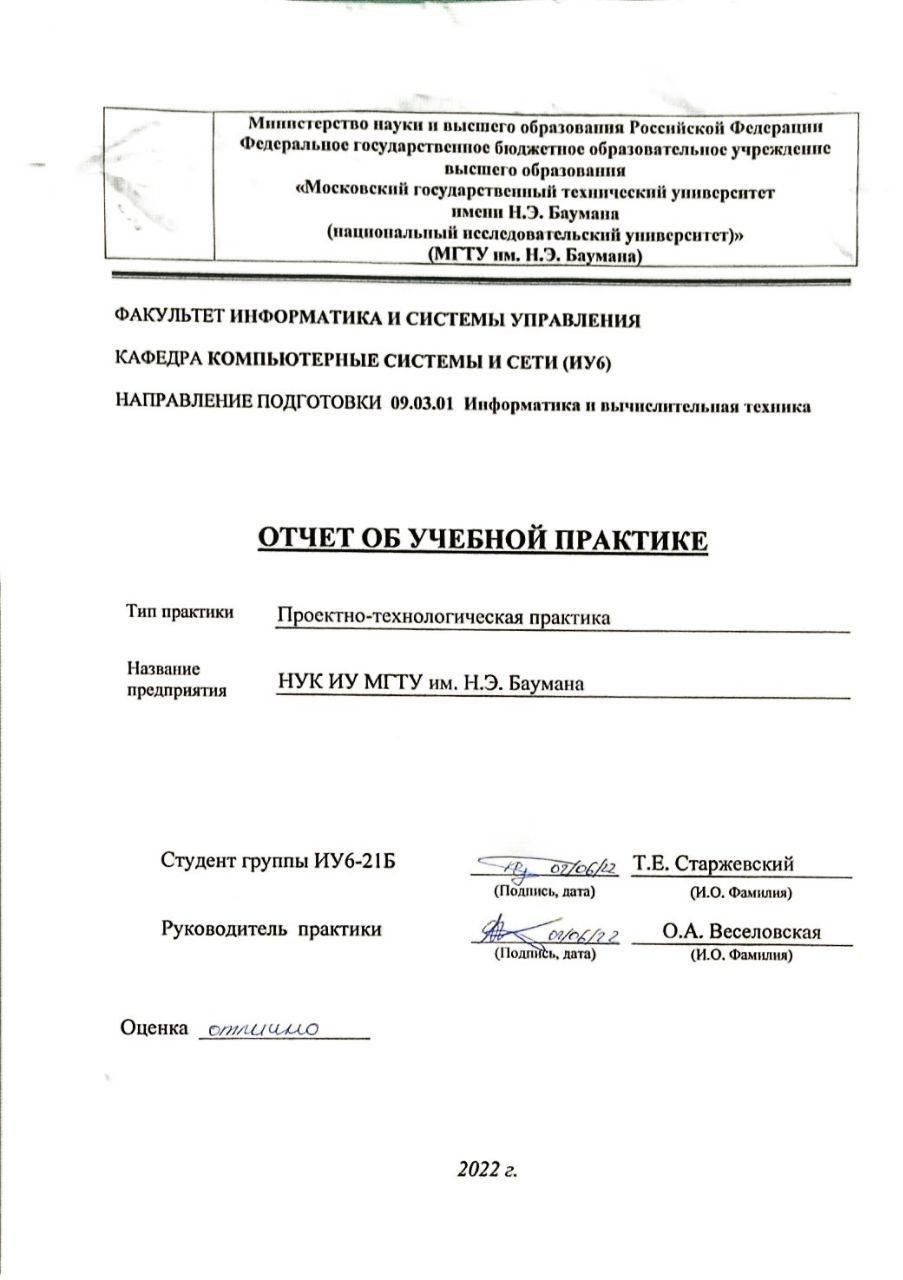
**

Оглавление

[**Задание 1. Создание программной системы на Object Pascal.** 5](#_Toc103452662)

[**Объектная декомпозиция программы:** 5](#_Toc103452663)

[**Диаграмма пользовательского интерфейса:** 6](#_Toc103452664)

[**Диаграмма классов интерфейсной и предметной области программы:** 7](#_Toc103452665)

[**Полученные формы интерфейса:** 8](#_Toc103452666)

[**Диаграмма последовательности действий процедуры** *TMain.Button1Click:* 9](#_Toc103452667)

[**Задание 2. Создание программной системы с элементарным интерфейсом консольного режима на С++** 10](#_Toc103452668)

[**Структурная декомпозиция:** 10](#_Toc103452669)

[**Структурная схема:** 11](#_Toc103452670)

[**Код программы:** 14](#_Toc103452671)

[**Задание 3. Создание программной системы c Qt интерфейсом на С++** 17](#_Toc103452672)

[**Объектная декомпозиция программы:** 17](#_Toc103452673)

[**Диаграмма пользовательского интерфейса:** 18](#_Toc103452674)

[**Диаграмма классов интерфейсной и предметной области программы:** 19](#_Toc103452675)

[**Полученные формы интерфейса:** 20](#_Toc103452676)

[**Диаграмма последовательности действий процедуры** *Admin::****on\_pushButton\_BACK\_clicked****()***:** 21](#_Toc103452677)

[**Заключение:** 22](#_Toc103452678)

***Вариант 16.***

# **Задание 1. Создание программной системы на Object Pascal.**

Выполнить объектную декомпозицию, разработать формы интерфейса, диаграмму состояний интерфейса, диаграммы классов интерфейсной и предметной областей, диаграмму последовательности действий одной из реализуемых операций. Разработать, протестировать и отладить программу.

База данных (файл) трамвайного депо содержит сведения о подвижном составе: инвентарный номер вагона, модель, год производства, срок службы (лет). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

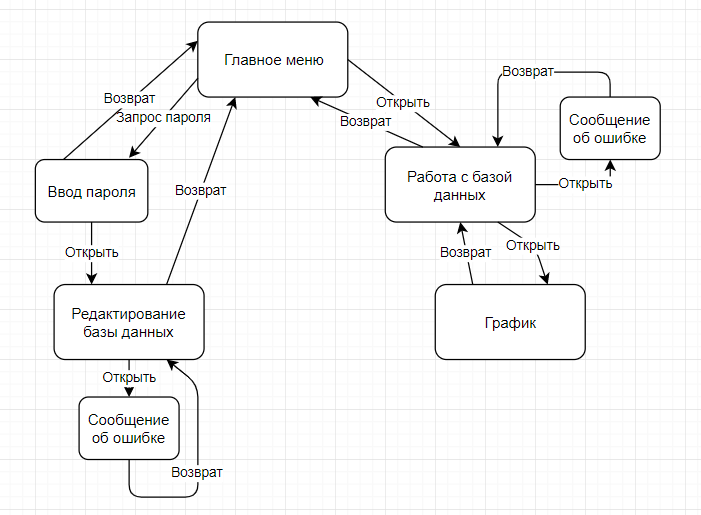
1. Найти модели трамваев, которые произведены начиная с заданного года.

2. Показать номера и модели трамваев, срок службы которых истек.

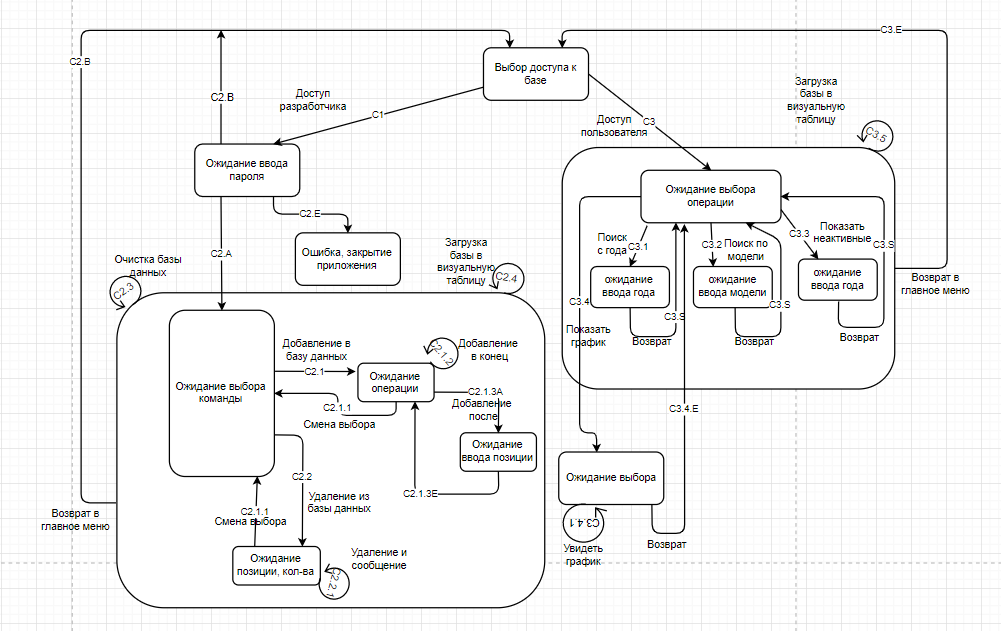
3. Определить количество трамваев заданной модели в депо.

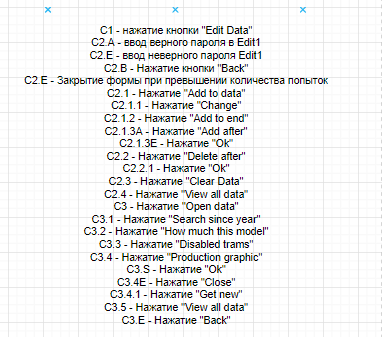
4. Построить график количества произведенных трамваев по годам.

## **Объектная декомпозиция программы:**

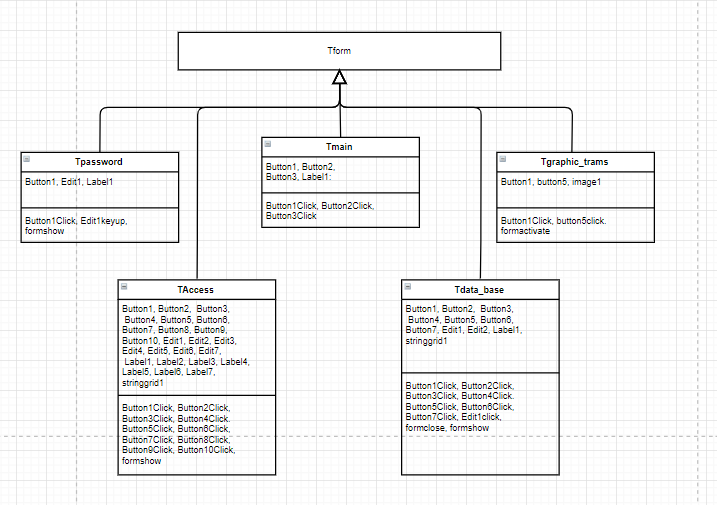
****

## **Диаграмма пользовательского интерфейса:**

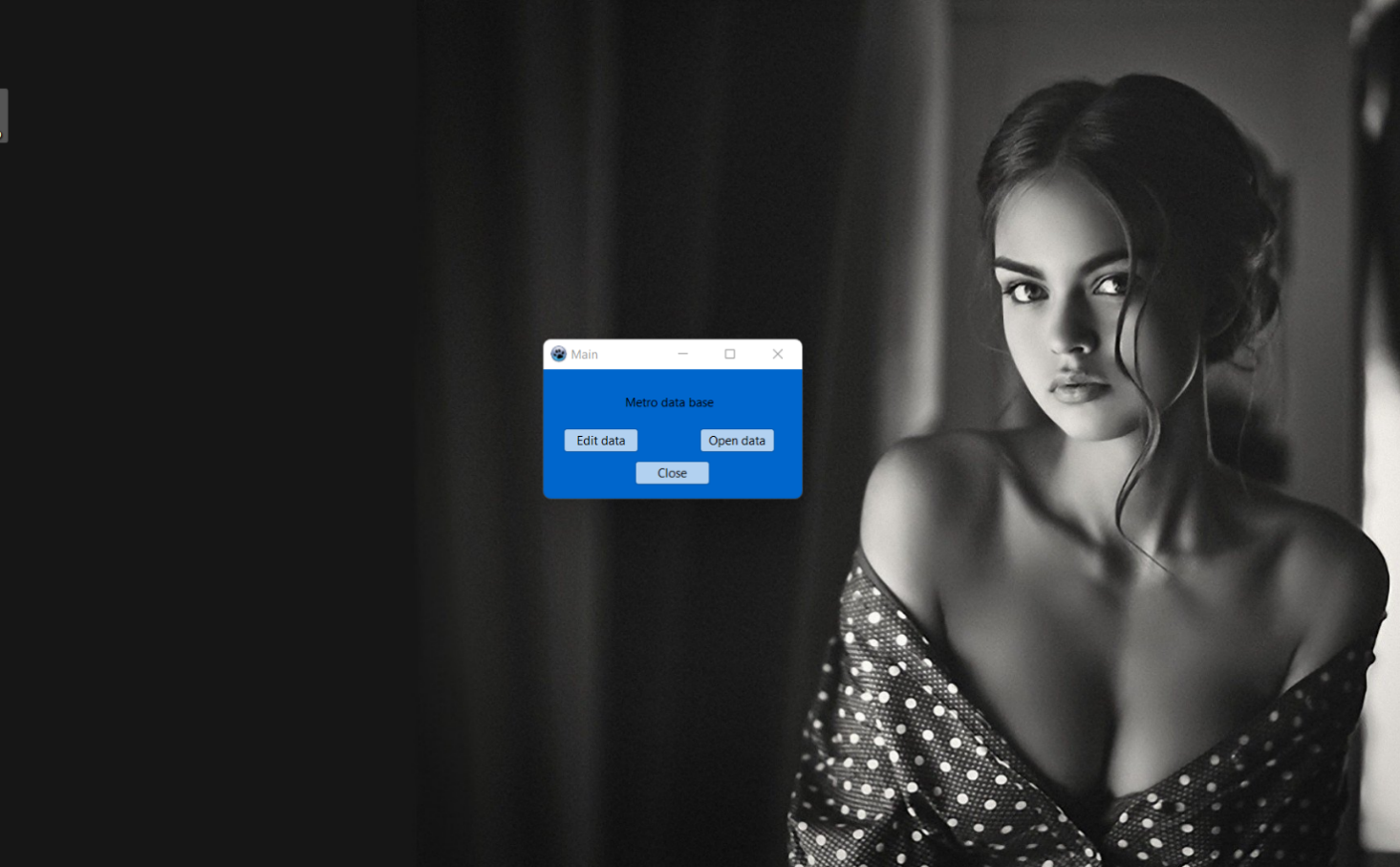
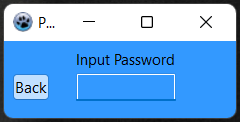
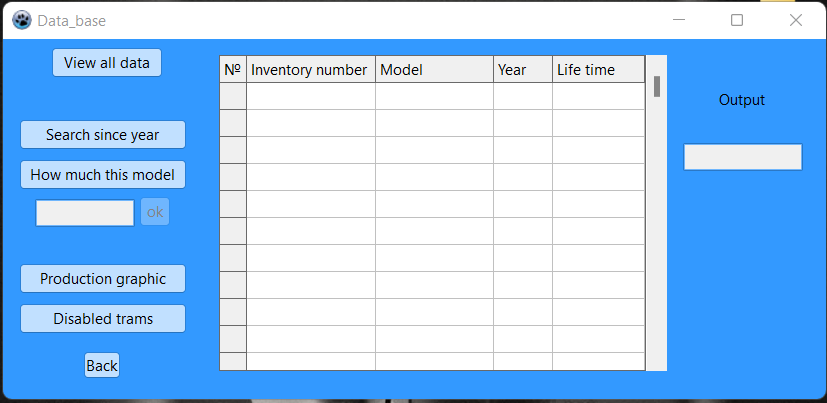
****

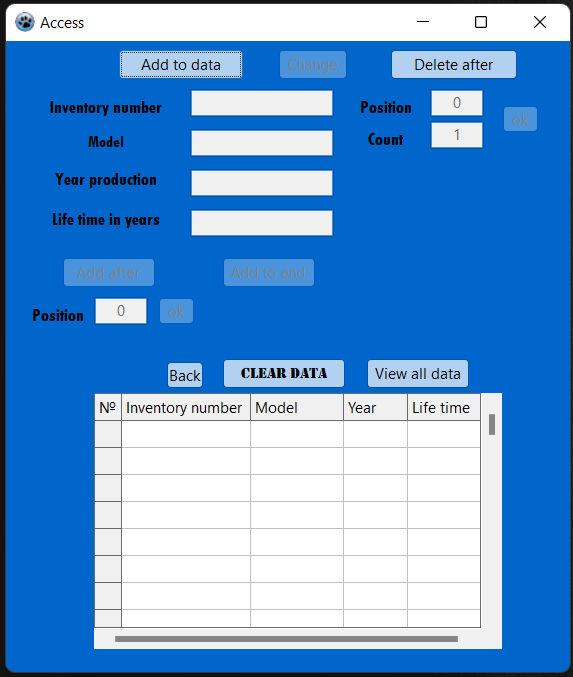
****

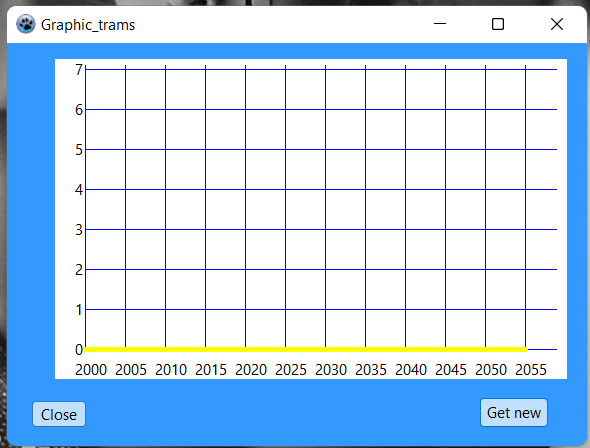
## **Диаграмма классов интерфейсной и предметной области программы:**

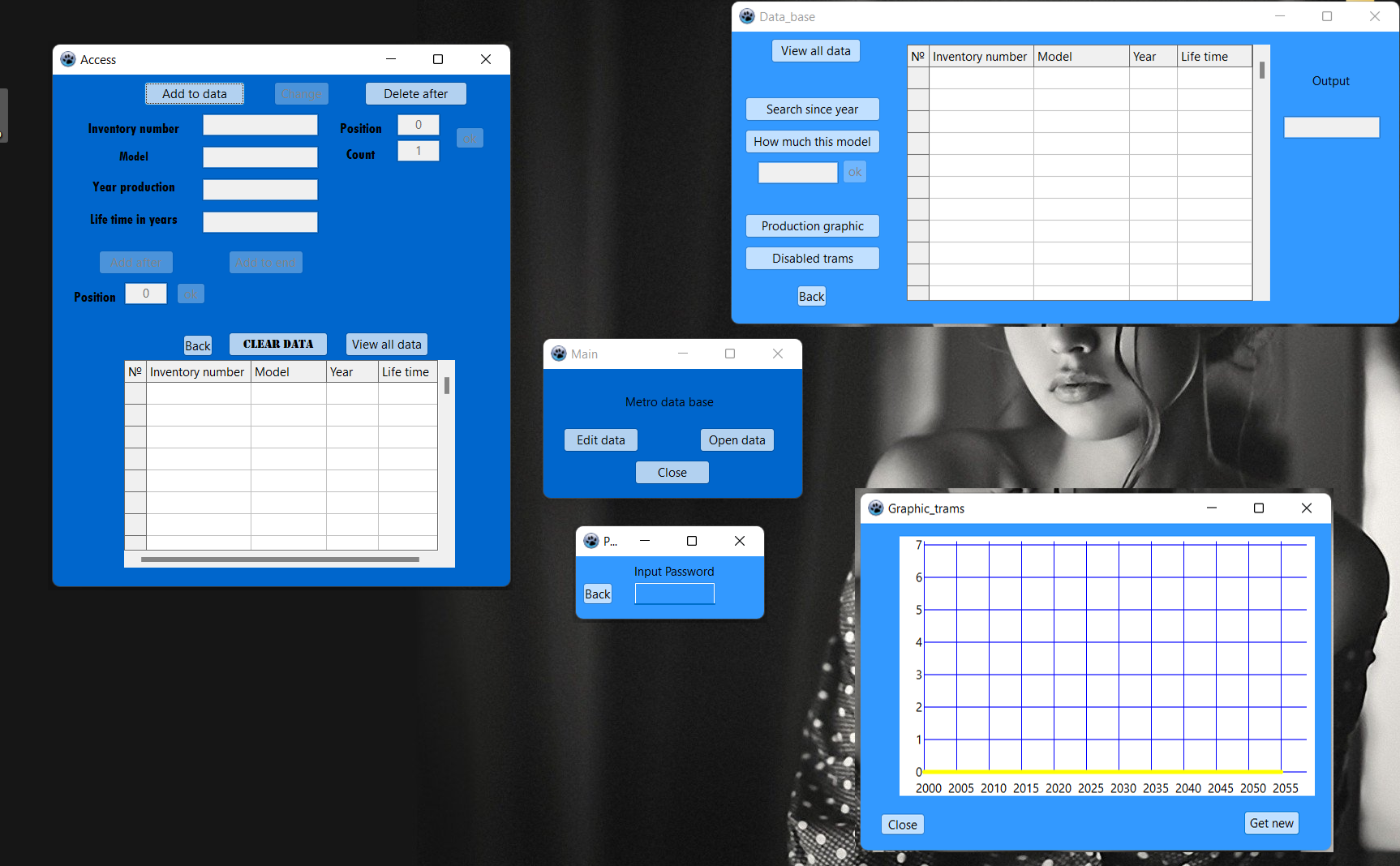
****

## **Полученные формы интерфейса:**

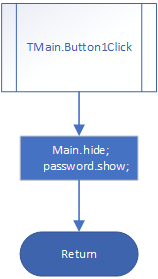
****

****

****



## **Диаграмма последовательности действий процедуры** *TMain.Button1Click:*

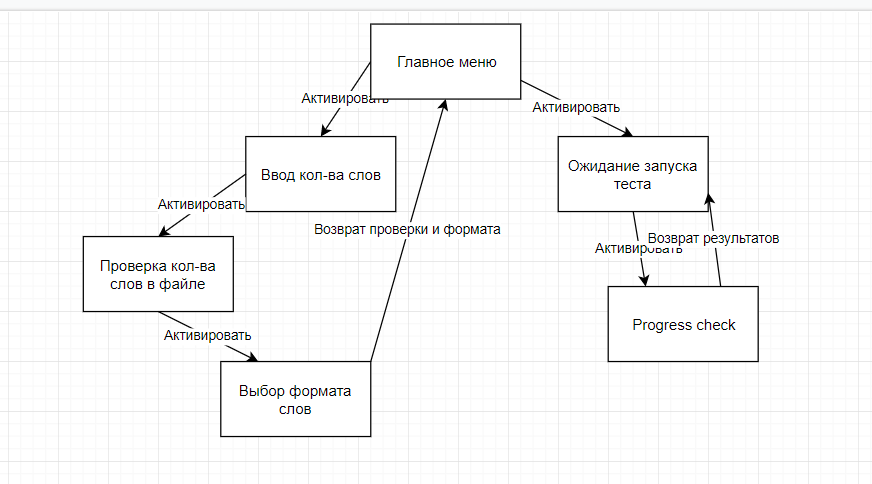


# **Задание 2. Создание программной системы с элементарным интерфейсом консольного режима на С++**

Выполнить структурную декомпозицию, разработать структурную схему, содержащую не менее 3 подпрограмм, и алгоритмы этих подпрограмм. Реализовать на С++ в консольном режиме. Предусмотреть примитивный интерфейс типа меню, позволяющий выбирать нужную подпрограмму.

Написать программу, помогающую запоминать слова иностранного языка. Программа предлагает слова, выбранные из некоторого множества случайным образом, а обучающийся должен ввести его эквивалент. По желанию пользователя предусмотреть смену языка. Придумать методику оценивания ответов и формирования итогового балла.

## **Структурная декомпозиция:**



## **Структурная схема:**







## **Код программы:**

#include <iostream>

#include <fstream>

#include <algorithm>

#include <time.h>

#include <set>

#include <conio.h>

#include <string>

//#define Teacher

//#define DEBUG

#define DEBUG\_AFTER

#define DEBUG\_CHEAT

//File format: "<English word>'\n'<Rus word>"

using namespace std;

struct slovo

{

string str1, str2;

};

int main()

{

string file\_name = "Wordlist\_5\_1251.txt";

system("chcp 1251 > nul");

//system("chcp 1251");

#ifdef Teacher

cout << "Input file name, standard it: " << file\_name << "\nChange it? [y/n]\t";

char ch = \_getch();

if ( ch == 'y' || ch == 'Y' || ch == 'í')

{

cout << "Input new file name: "; cin >> file\_name;

}

else cout << "\n Using standard file name (pls don`t input 'Y' instead of 'y') \n";

#endif

#ifdef DEBUG

cout << "\nÒåñòèðóþ ðóññêèé ÿçûê\n";

#endif

fstream file;

file.open(file\_name, fstream::in);

if (!file.is\_open())

{

cout << "File wasn`t open! Please make sure the name is: " << file\_name << endl;

system("pause");

return -1;

}

else

{

int count\_words = 0, i = 0,main\_language = 2;

slovo mas[255]; string input;

cout << "How much words u want to test (<256): ";

cin >> count\_words;

cout << "Choose format:\n(1)Eng - rus || (2)Rus - eng \tMy choose: ";

cin >> main\_language;

if (count\_words > 255)

count\_words = 255;

if (main\_language == 1)

while (!file.eof())

{

getline(file, mas[i].str1);

getline (file,mas[i].str2); //str1 - eng str2 - rus

if (mas[i].str1 == "") i--;

i++;

}

else while (!file.eof())

{

getline(file, mas[i].str2);

getline (file, mas[i].str1); //str1 - rus str2 - eng

if (mas[i].str1 == "") i--;

i++;

}

file.close();

if (i < count\_words)

{

count\_words = i;

cout << "File consist only " << count\_words << " words, so let`s start with it" << endl;

}

//Start counting

cout << "Press something to start or '-' for end...\n";

cin.ignore();

int for\_rand = i, tryes = 0;

ofstream Results;

Results.open("Results.txt");

while (\_getch() != '-')

{

cout << "============= Progress Check ============\n";

srand(time(NULL)); set<int> check; int total = 0, correct = 0;

i = rand() % count\_words;

while (total < count\_words)

{

while (check.count(i))

i = rand() % for\_rand;

check.insert(i);

cout << "[" << total+1 << "/" << count\_words << "] " << mas[i].str1;

#ifdef DEBUG\_CHEAT

cout << "(" << mas[i].str2 << ")";

#endif

cout << " - ";

getline (cin,input);

transform(input.begin(), input.end(), input.begin(), ::tolower);

transform(mas[i].str2.begin(), mas[i].str2.end(), mas[i].str2.begin(), ::tolower);

#ifdef DEBUG\_AFTER

cout << "Input: " << input << "\tCorrect: " << mas[i].str2 << "\n";

#endif

if (input == mas[i].str2)

correct++;

else

cout << "Mistake! Try harder\n";

total++;

}

int var = correct \* 100 / total;

cout << "===> Results <====\nTotal words: " << total << "\tCorrect translated: " << correct << "\tProcent: " << var << "\n--> Your mark: ";

Results << ++tryes << ") Total words : " << total << "\tCorrect translated : " << correct << "\tProcent : " << var << "\n-- > Your mark : ";

if (var > 84)

{

cout << "5\n";

Results << "5\n";

}

else if (var > 64)

{

cout << "4\n";

Results << "4\n";

}

else if (var > 49)

{

cout << "3\n";

Results << "3\n";

}

else {

cout << "2\n";

Results << "2\n";

}

cout << "\nEnter '-' to end test or press enter to start new Test...\n";

}

Results.close();

return 0;

}

}

# **Задание 3. Создание программной системы c Qt интерфейсом на С++**

Выполнить объектную декомпозицию, разработать формы интерфейса, диаграмму состояний интерфейса, диаграммы классов интерфейсной и предметной областей, диаграмму последовательности действий одной из реализуемых операций. Разработать, протестировать и отладить программу в среде Visual Studio или QT Creator.

База данных трамвайного депо содержит сведения о подвижном составе: инвентарный номер вагона, модель, год производства, срок службы (лет). Программа должна в интерактивном режиме формировать файл, добавлять и удалять данные, а также воспринимать каждый из перечисленных запросов и давать на него ответ.

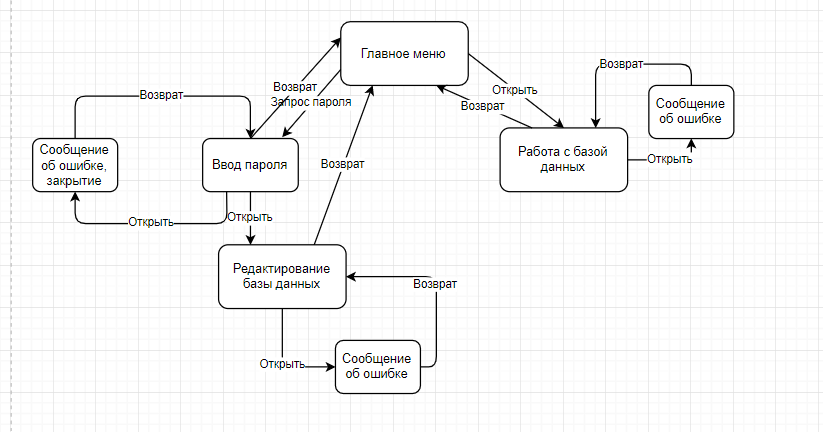
1. Найти модели трамваев, которые произведены начиная с заданного года.

2. Показать номера и модели трамваев, срок службы которых истек.

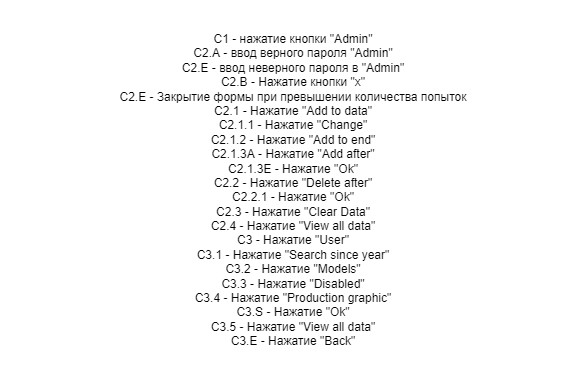
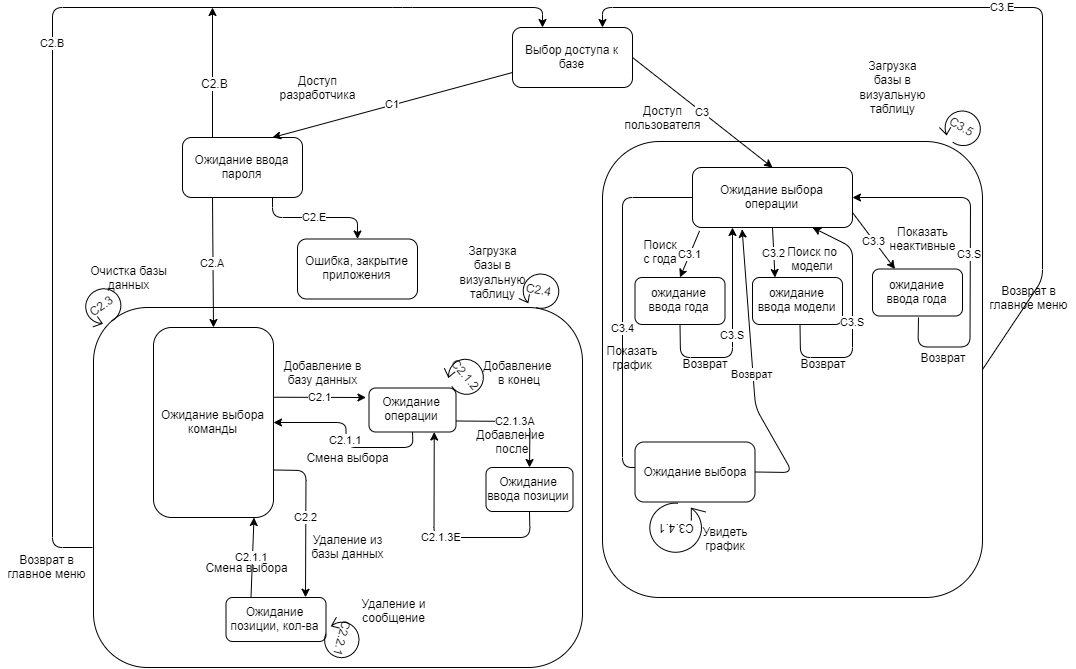
3. Определить количество трамваев заданной модели в депо.

4. Построить график количества произведенных трамваев по годам.

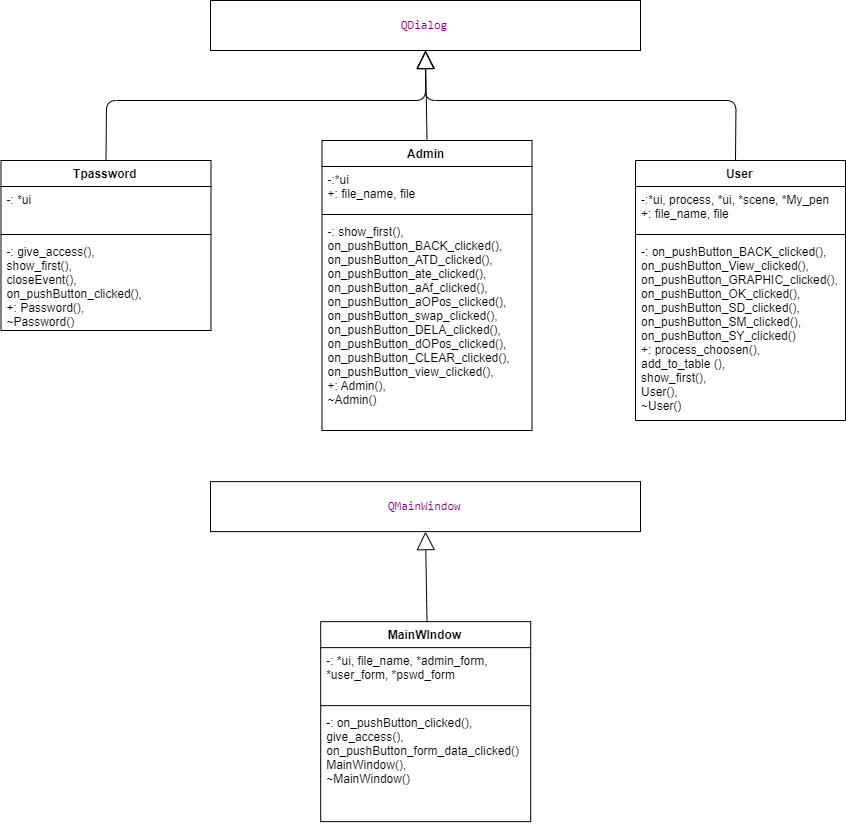
**Объектная декомпозиция программы:**

****

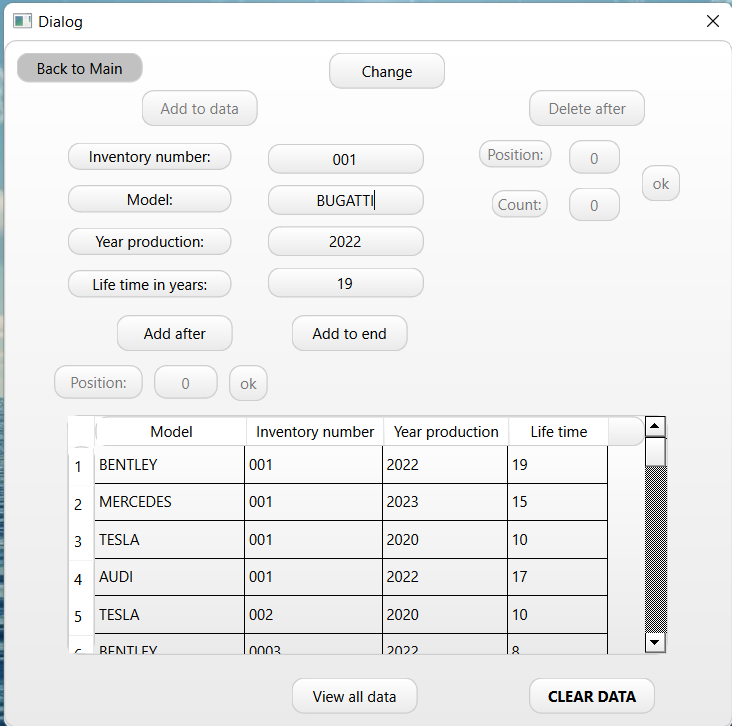
## **Диаграмма пользовательского интерфейса:**

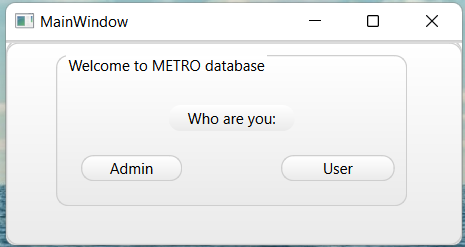
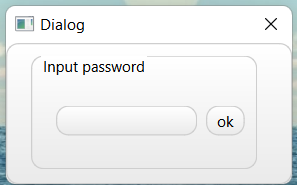
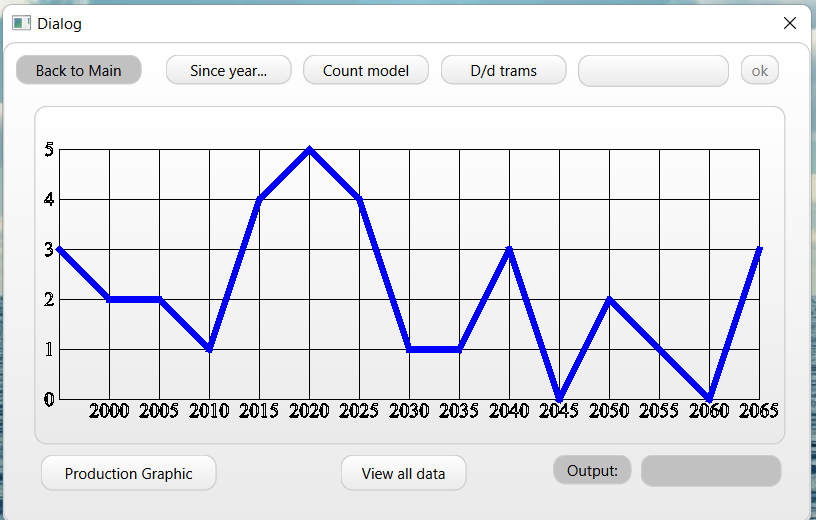
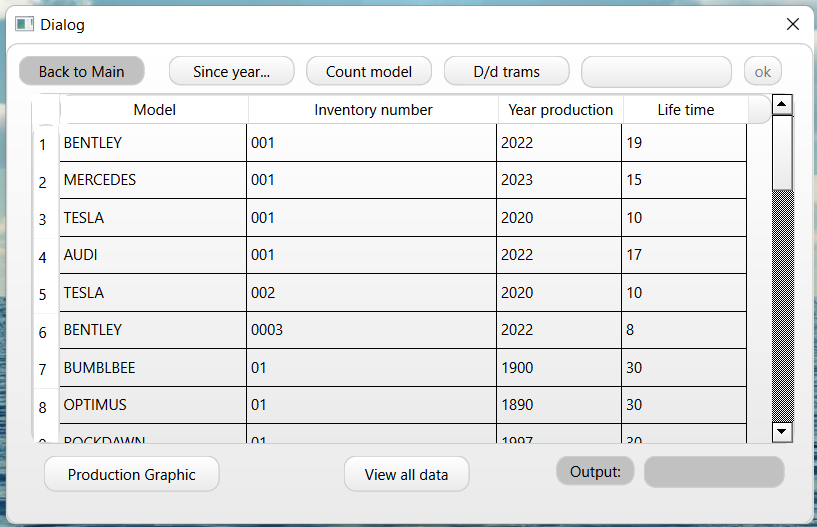
****

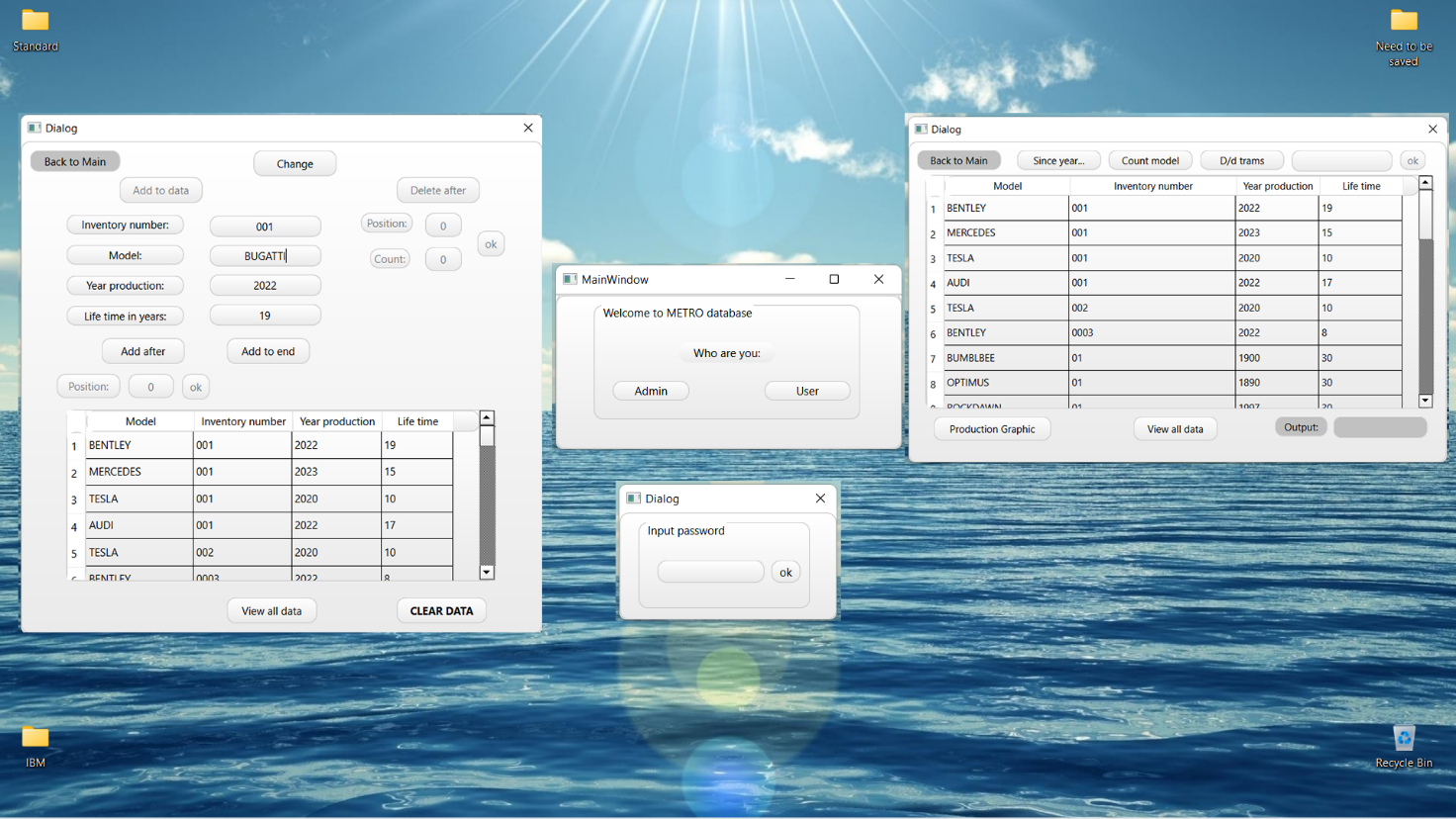
## **Диаграмма классов интерфейсной и предметной области программы:**

****

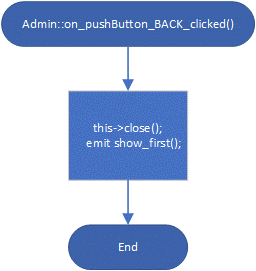
## **Полученные формы интерфейса:**

****

****

****

## **Диаграмма последовательности действий процедуры** *Admin::****on\_pushButton\_BACK\_clicked****()***:**



# **Заключение:**

По заданным техническим заданиям было создано многооконное приложение для работы с базой данных в среде “Lazarus” и “QT” на разных языках программирования: Pascal и C++ соответственно. База данных хранится в файле, доступ к которому возможен с точки зрения администратора и пользователя. Работа с файлом протестирована и утечки данных внутри приложения не возникали. На языке C++ по второму техническому заданию была написана и протестирована программа для изучения русских/английских слов. При заданных условиях все работало корректно.