

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ**
Федеральное государственное бюджетное образовательное учреждение высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОСКОВСКИЙ ГОСУДАРСТВЕННЫЙ
СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ»**

Институт Цифровых технологий моделирования в строительстве (ИЦТМС)
Кафедра Информационных систем, технологий и автоматизации в строительстве (ИСТАС)

КУРСОВАЯ РАБОТА

по дисциплине
«Архитектура прикладного программного обеспечения»

Тема:
«Разработка ИС «Аптечный склад»»

Выполнил обучающийся

ИЦТМС-3-8 Сафин Р. Р.

(институт, курс, группа, Ф.И.О.)

Руководитель курсовой работы

Ст. преподаватель Садовский Б. С.

(учёное звание, учёная степень, должность, Ф.И.О.)

К защите

(дата, подпись руководителя)

Курсовая работа защищена с оценкой

(оценка цифрой и прописью)

Руководитель курсовой работы

Доцент к.т.н., доц. Адамцевич Л. А.

(дата, подпись руководителя)

Председатель аттестационной
комиссии

(учёное звание, учёная степень, должность, Ф.И.О.)

Члены комиссии:

(дата, подпись члена комиссии)

Москва
2023

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ МОСКОВСКИЙ
ГОСУДАРСТВЕННЫЙ СТРОИТЕЛЬНЫЙ УНИВЕРСИТЕТ»

Институт Цифровых технологий моделирования в строительстве (ИЦТМС)

Кафедра Информационных систем, технологий и автоматизации в строительстве (ИСТАС)

Дисциплина Архитектура прикладного программного обеспечения

ЗАДАНИЕ НА ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

Студенту группы ИЦТМС-3-8 Сафину Ралифу Рафисовичу

1. Тема курсовой работы «Разработка ИС «Риэлторская контора»»
2. Исходные данные к курсовой работе:
 1. лекции, компьютерные практики и курсовая работа по курсу ООП;
 2. лекции и компьютерные практики по курсу «Архитектура прикладного ПО»;
 3. исходный код примеров решения отдельных деталей интерфейса Qt;
 4. документация по Qt;
 5. указания по оформлению отчёта;
 6. инструкция по использованию программы Doxygen.
3. Перечень подлежащих разработке вопросов:
 1. Программа должна быть написана на Си++ используя кросс-платформенный инструментарий разработки Qt или аналога.
 2. Программа должна хранить данные в СУБД PostgreSQL. То есть должна:
 - I. Подключаться к СУБД по сети.
 - II. Отображать данные из таблиц и/или представлений.
 - III. Добавлять данные в БД.
 - IV. Редактировать данные в БД.
 - V. Удалять данные из БД.
 3. ПО должно иметь сетевую клиент — серверную архитектуру (возможна установка сервера на виртуальной машине или на отдельном сервере).
 4. Работа с данными должна осуществляться с использованием языка SQL.
 5. В БД должно храниться не менее 1000-и записей в основной таблице. Каждая запись — это один пункт списка таблицы. По каждой записи должно быть не менее 6 свойств (характеристик). Все свойства определяются целесообразностью их упоминания.
 6. Исходный текст программы должен быть полностью задокументирован с помощью комментариев и комментариев программы Doxygen.

7. Программа должна состоять из главного окна и диалоговых окон (как минимум одного диалогового окна).
 8. В главном окне должно быть простое меню для реализации основных задач обработки данных.
 9. Объекты в окне не должны расползаться при изменении его размеров.
 10. В программе обязательно должно быть указано авторство (институт-курс-группа, фамилия, имя отчество полностью).
 11. Авторство указывается в диалоговом окне, вызываемым из меню главного окна. Оформить как пункт меню об авторе.
 12. Программа должна быть на русском языке, но при этом язык программы должен быть оформлен как перевод из файла **qm**. В файлах с исходным кодом (*.cpp, *.h, *.hpp) всё должно быть на английском через функцию tr(). Аналогично в файлах с пользовательским интерфейсом (*.ui) весь интерфейс должен быть на английском.
 13. Программа должна запоминать настройки интерфейса.
 14. Для хранения и отображения данных программа может использовать QTableWidget. Загрузка и работа с данными может быть реализована через QSqlQuery.
 15. Вместо использования QTableWidget, должен быть реализован архитектурный паттерн «Модель-представление-контроллер» (англ. Model-View-Controller, MVC).
 16. Для обработки корректности, вводимых пользователем данных, должны использоваться соответствующие объекты по написанным классам отображения элементов, а также рекомендуется применять различные валидаторы и маски ввода.
 17. Данные, с которыми происходит работа должны быть разделены на отдельные таблицы и нормализованы.
 18. У таблиц должны быть реализованы защита данных от аномалий обновления. Например, должна быть связь по ключам.
 19. Должно быть разделение обработки данных. На стороне сервера должна быть реализована обработка объёмных данных, а на стороне клиента оставить простые операции. Например, расчёт зарплат сотрудникам должен быть на стороне сервера, а отображение цен на товары по курсу валюты, на стороне клиента.
 20. Загрузка данных на клиент должна быть по частям.
 21. Должна быть реализована возможность поиска с фильтрацией и сортировки по отдельным столбцам таблицы выделенным пользователем.
 22. Должна быть реализована технология Drag-and-drop для перетаскивания записей из одного документа в другой, и во внешние программы.
 23. При изменении данных в таблицах в несколько запросов, необходимо использовать транзакции.
 24. Соединение с СУБД PostgreSQL надо организовать через шифрование SSL.
4. Предоставить отчёт о проделанной работе оформленные согласно нормам ГОСТ, UML, положению о курсовых проектах и курсовых работах обучающихся (Выпуск 3) и др. В отчёте обязательно должны быть разделы:
1. Введение.

2. Основная часть.
 - I. Входные данные.
 - II. Интерфейс пользователя.
 - III. Структура проекта.
 - IV. Описание классов и методов.
3. Заключение.
4. Библиографический список.
5. Приложение (опционально).
5. Основная часть отчёта формируется с использованием программы Doxygen на основе комментариев в исходном коде.

График выполнения курсовой работы:

№	Наименование этапа выполнения курсовой работы	Срок выполнения	Процент выполнения курсовой работы
1	2	3	4
1	Получение задания	14.02.2023	5%
2	Создание БД, создание главного окна приложения и стандартные диалоговые окна. Меню программы	10.05.2023	20%
3	Организован интерфейс приложения с учётом изменения геометрии окна и политик. Сохранение настроек	10.05.2023	40%
4	Сделать подключение программы клиента к СУБД PostgreSQL	25.05.2023	60%
5	Организация работы с таблицами БД.	25.05.2023	75%
6	Поиск и сортировка. Шаблон проектирования MVC	25.05.2023	85%
7	Доработка и устранение ошибок в приложении	25.06.2023	90%
8	Написание отчёта	26.06.2023	100%

6. Дата выдачи задания 14.02.2023.

Задание получил студент
Сафин Р. Р.

(подпись)

Руководитель курсовой работы:
Ст. преподаватель Садовский Б. С.

(подпись)

Оглавление

ВВЕДЕНИЕ	6
1. Исходные данные	7
2. Технические требования	12
3. Функциональные возможности	13
4. Интерфейс программы.....	14
4.1. Главное окно.....	14
4.2. Работа с записями в документе.....	15
5. Описание исходного кода программы.....	17
5.1. Иерархия классов	17
5.2 Описание классов и их методов	17
5.2.1 Класс dialogAuthor	17
5.2.2 Класс dialogDelete	17
5.2.3 Класс dialogEdit	18
5.2.4 Класс dialogRedaction.....	19
5.2.5 Класс MainWindow.....	20
5.2.6 Класс Model.....	21
6. Список файлов с исходным кодом и вспомогательных	23
ЗАКЛЮЧЕНИЕ	24
БИБЛИОГРАФИЧЕСКИЙ СПИСОК.....	25

ВВЕДЕНИЕ

Целью курсовой работы является написание программы с применением механизмов ООП языка Си++/Qt, таких как инкапсуляция, полиморфизм и наследование. Программа должна вести учёт офисного оборудования строительной организации.

Программа должна подключаться к базе данных и взаимодействовать с ее таблицами.

Для достижения поставленной цели должен быть реализован ряд таких задач как:

Разработанное ПО должно отвечать следующим требованиям:

- выводить содержимое из таблиц БД на экран пользователя;
- добавлять, изменять или удалять данные из таблиц БД;
- иметь возможность сменить язык;
- иметь возможность сортировки данных;
- сохранять настройки интерфейса;
- иметь окно для добавления записей и т.д.

Курсовая работа нужна, чтобы понять, как использовать язык C++ для работы с базами данных. Это может включать в себя создание таблиц, добавление/удаление записей, выполнение запросов и т.д.

Также работа способствует подготовке к работе в индустрии, где работа с базами данных является одной из ключевых областей. Как правило, компании, которые разрабатывают программное обеспечение, всегда нуждаются в специалистах, которые знают, как работать с базами данных [1].

1. Исходные данные

Исходными данными для работы является база данных риэлторской конторы. Она состоит из 4 таблиц:

- Основная таблица «deals»;
- вспомогательная таблица «clients»;
- вспомогательная таблица «clientsrequests»;
- вспомогательная таблица «employees».
- вспомогательная таблица «immovables».
- вспомогательная таблица «immovablestypes».
- вспомогательная таблица «owners».
- вспомогательная таблица «ownersrequests».

Структура и часть данных основной таблицы приведены на рисунке 1.1.

	id [PK] integer	id_client integer	id_employee integer	id_immovable integer	trans_date date	trans_amount numeric	deal_desc text	id_owner integer
1	1	14306	38	4	2020-02-04	929.00	прошла успешно, обе стороны довольны	337
2	2	14051	13	70	2018-02-17	4184.00	прошла успешно, обе стороны довольны	111
3	3	13978	11	404	2015-06-10	4385.00	прошла успешно, обе стороны довольны	325
4	4	14256	23	479	2011-08-27	3793.00	прошла успешно, обе стороны довольны	188
5	5	14228	19	292	2017-10-12	2683.00	прошла успешно, обе стороны довольны	316
6	6	14171	45	290	2012-04-04	17.00	прошла гладко	435
7	7	14262	33	289	2008-08-21	544.00	прошла гладко	306
8	8	14087	40	376	2016-07-31	1069.00	прошла успешно, обе стороны довольны	420
9	9	14181	47	378	2019-05-22	1086.00	прошла успешно, обе стороны довольны	129
10	10	14085	49	249	2014-02-02	3346.00	прошла успешно, обе стороны довольны	453
11	11	14170	35	99	2017-05-24	3508.00	прошла успешно, обе стороны довольны	382
12	12	14216	22	187	2012-09-13	2310.00	прошла успешно, обе стороны довольны	131
13	13	13862	50	102	2017-08-17	2963.00	прошла гладко	479
14	14	14090	55	466	2015-09-10	2977.00	прошла успешно, обе стороны довольны	431
15	15	14119	8	279	2017-04-23	1719.00	прошла гладко	237
16	16	13997	28	124	2011-06-27	3943.00	прошла гладко	374
17	17	14306	9	7	2016-09-20	2106.00	прошла успешно, обе стороны довольны	443
18	18	14055	27	342	2022-12-25	3763.00	прошла гладко	63
19	19	13990	54	51	2014-11-23	179.00	прошла гладко	366
20	20	14255	48	321	2010-02-22	1163.00	прошла успешно, обе стороны довольны	36
21	21	13859	50	289	2012-01-13	4871.00	прошла успешно, обе стороны довольны	389
22	22	14296	51	123	2016-09-21	139.00	прошла успешно, обе стороны довольны	385

Рисунок 1.1 – Основная таблица

Структура и часть данных таблицы «clients» приведены на рисунке 1.2.

	id [PK] integer	full_name text	phone_number text	e_mail text	passport_data jsonb	address jsonb
1	13827	ААА ААБ	07070707070707070	aaaaa@a@a...	{ "num": "000000", "date": "08/03/2023", "series": "000..." }	{ "flat": 108, "home": 99, "street": "dddd" }
2	13828	ААВ ААГ	07070707070707071	aaaca@a@a...	{ "num": "000001", "date": "02/06/2020", "series": "000..." }	{ "flat": 73, "home": 116, "street": "1-й Квартал", "distr
3	13829	ААД ААЕ	07070707070707072	aaaea@a@a...	{ "num": "000002", "date": "03/19/2022", "series": "000..." }	{ "flat": 12, "home": 15, "street": "1-й Маевки Аллея",
4	13830	ААЕ ААЖ	07070707070707073	aaaga@a@a...	{ "num": "000003", "date": "04/22/2014", "series": "000..." }	{ "flat": 115, "home": 110, "street": "1-й Автозаводс
5	13831	ААЗ ААИ	07070707070707074	aaaila@a@a@j	{ "num": "000004", "date": "08/03/2008", "series": "000..." }	{ "flat": 49, "home": 16, "street": "1-й Амбулаторный
6	13832	ААЙ ААК	07070707070707075	aaaka@a@a@l	{ "num": "000005", "date": "09/17/2020", "series": "000..." }	{ "flat": 171, "home": 110, "street": "1-й Архивный Г
7	13833	ААЛ ААМ	07070707070707076	aaama@a@a...	{ "num": "000006", "date": "07/29/2023", "series": "000..." }	{ "flat": 113, "home": 106, "street": "1-й Бабыгород
8	13834	ААН ААО	07070707070707077	aaaoa@a@a...	{ "num": "000007", "date": "01/24/2012", "series": "000..." }	{ "flat": 123, "home": 31, "street": "1-й Бабыгородс
9	13835	ААП ААР	07070707070707078	aaaua@a@a...	{ "num": "000008", "date": "04/18/2022", "series": "000..." }	{ "flat": 97, "home": 62, "street": "1-й Балтийский П
10	13836	ААС ААТ	07070707070707079	aaawa@a@a...	{ "num": "000009", "date": "06/30/2010", "series": "000..." }	{ "flat": 74, "home": 60, "street": "1-й Балтийский п
11	13837	ААУ ААФ	07070707070707171	aaaya@a@a...	{ "num": "000011", "date": "01/25/2008", "series": "001..." }	{ "flat": 106, "home": 51, "street": "1-й Басманный Г
12	13838	ААХ ААЦ	07070707070707172	aabba@a@b...	{ "num": "000012", "date": "08/29/2021", "series": "001..." }	{ "flat": 144, "home": 19, "street": "1-й Белокаменн
13	13839	ААЧ ААШ	07070707070707173	aabda@a@b...	{ "num": "000013", "date": "08/06/2014", "series": "001..." }	{ "flat": 126, "home": 79, "street": "1-й Ботанически
14	13840	ААЩ ААЭ	07070707070707174	aabfa@a@b...	{ "num": "000014", "date": "04/22/2009", "series": "001..." }	{ "flat": 198, "home": 76, "street": "1-й Боткинский Л
15	13841	ААЮ ААЯ	07070707070707175	aabha@a@b@i	{ "num": "000015", "date": "03/27/2016", "series": "001..." }	{ "flat": 135, "home": 60, "street": "1-й Боткинский Л
16	13842	АББ АБВ	07070707070707176	aabja@a@b@k	{ "num": "000016", "date": "09/01/2017", "series": "001..." }	{ "flat": 16, "home": 69, "street": "1-й Варшавский Г
17	13843	АБГ АБД	07070707070707177	aabla@a@b...	{ "num": "000017", "date": "08/04/2021", "series": "001..." }	{ "flat": 28, "home": 24, "street": "1-й Верхний Миха
18	13844	АБЕ АБЕ	07070707070707178	aabna@a@b...	{ "num": "000018", "date": "04/10/2014", "series": "001..." }	{ "flat": 160, "home": 62, "street": "1-й Вешняковски
19	13845	АБЖ АБЗ	07070707070707179	aabpa@a@b...	{ "num": "000019", "date": "07/27/2020", "series": "001..." }	{ "flat": 66, "home": 38, "street": "1-й Войковский П
20	13846	АБИ АБЙ	07070707070707272	aabva@a@b...	{ "num": "000022", "date": "07/13/2013", "series": "002..." }	{ "flat": 108, "home": 109, "street": "1-й Волконский
21	13847	АБК АБЛ	07070707070707273	aabxa@a@b...	{ "num": "000023", "date": "04/10/2010", "series": "002..." }	{ "flat": 40, "home": 66, "street": "1-й Волоколамски
22	13848	АБМ АБН	07070707070707274	aabza@a@c...	{ "num": "000024", "date": "11/19/2012", "series": "002..." }	{ "flat": 23, "home": 113, "street": "1-й Вражский Пе

Рисунок 1.2 – таблицы «clients»

Структура и часть данных таблицы «clientsrequests» приведены на рисунке

1.3.

	id integer	id_client [PK] integer	type_immovable text	preferences jsonb
1	14	13827	2	{ "cost": 3091, "roof": 30, "other": "хочу подешевле", "street":
2	22	13828	2	{ "cost": 911, "roof": 11, "other": "хочу подешевле", "street": "
3	485	13829	1	{ "cost": 3851, "roof": 14, "other": "хочу подешевле", "street":
4	467	13830	4	{ "cost": 3573, "roof": 20, "other": "хочу подешевле", "street":
5	406	13831	4	{ "cost": 1379, "roof": 11, "other": "хочу подешевле", "street":
6	388	13832	1	{ "cost": 4438, "roof": 5, "other": "хочу подешевле", "street": "
7	271	13833	3	{ "cost": 4732, "roof": 9, "other": "хочу подешевле", "street": "
8	350	13834	2	{ "cost": 4776, "roof": 42, "other": "хочу подешевле", "street":
9	276	13835	4	{ "cost": 253, "roof": 2, "other": "хочу подешевле", "street": "4
10	395	13836	3	{ "cost": 1506, "roof": 25, "other": "хочу подешевле", "street":
11	407	13837	1	{ "cost": 2048, "roof": 15, "other": "хочу подешевле", "street":
12	357	13838	5	{ "cost": 2940, "roof": 5, "other": "хочу подешевле", "street": "
13	24	13839	5	{ "cost": 1634, "roof": 32, "other": "хочу подешевле", "street":
14	59	13840	3	{ "cost": 1742, "roof": 27, "other": "хочу подешевле", "street":
15	120	13841	6	{ "cost": 2682, "roof": 37, "other": "хочу подешевле", "street":
16	29	13842	3	{ "cost": 822, "roof": 8, "other": "хочу подешевле", "street": "1
17	369	13843	2	{ "cost": 1977, "roof": 3, "other": "хочу подешевле", "street": "

Рисунок 1.3 – Таблица «clientsrequests»

Структура и часть данных таблицы «employees» приведены на рисунке 1.4.

	id [PK] integer	full_name text	job text	phone_number text	e_mail text	passport_data jsonb
1	1	ААА ААБ	courier	07070707070707070	aaaaa@a@a@b	{"num": "000000", "date": "05/23/2020", "series": "000..."}
2	2	ААВ ААГ	courier	07070707070707071	aaaca@a@a@d	{"num": "000001", "date": "09/20/2015", "series": "000..."}
3	3	ААД ААЕ	courier	07070707070707072	aaaea@a@a@f	{"num": "000002", "date": "03/01/2012", "series": "000..."}
4	4	ААЕ ААЖ	manager	07070707070707073	aaaga@a@a@h	{"num": "000003", "date": "03/12/2018", "series": "000..."}
5	5	ААЗ ААИ	manager	07070707070707074	aaiaa@a@a@j	{"num": "000004", "date": "01/22/2022", "series": "000..."}
6	6	ААЙ ААК	manager	07070707070707075	aaaka@a@a@l	{"num": "000005", "date": "03/27/2010", "series": "000..."}
7	7	ААЛ ААМ	courier	07070707070707076	aaama@a@a@...	{"num": "000006", "date": "06/26/2023", "series": "000..."}
8	8	ААН ААО	realtor	07070707070707077	aaaoa@a@a@p	{"num": "000007", "date": "11/28/2010", "series": "000..."}
9	9	ААП ААР	manager	07070707070707078	aaaua@a@a@v	{"num": "000008", "date": "08/02/2011", "series": "000..."}
10	10	ААС ААТ	manager	07070707070707079	aaawa@a@a@x	{"num": "000009", "date": "08/14/2019", "series": "000..."}
11	11	ААУ ААФ	realtor	07070707070707171	aaaya@a@a@z	{"num": "000011", "date": "12/24/2017", "series": "001..."}
12	12	ААХ ААЦ	realtor	07070707070707172	aabba@a@b@c	{"num": "000012", "date": "10/26/2020", "series": "001..."}
13	13	ААЧ ААШ	operator	07070707070707173	aabda@a@b@e	{"num": "000013", "date": "06/24/2009", "series": "001..."}
14	14	ААЩ ААЭ	trainee	07070707070707174	aabfa@a@b@g	{"num": "000014", "date": "11/07/2018", "series": "001..."}
15	15	ААЮ ААЯ	accountant	07070707070707175	aabha@a@b@i	{"num": "000015", "date": "08/01/2021", "series": "001..."}
16	16	АББ АБВ	operator	07070707070707176	aabja@a@b@k	{"num": "000016", "date": "05/10/2020", "series": "001..."}
17	17	АБГ АБД	manager	07070707070707177	aabla@a@b@m	{"num": "000017", "date": "04/06/2018", "series": "001..."}
18	18	АБЕ АБЁ	courier	07070707070707178	aabna@a@b@o	{"num": "000018", "date": "09/10/2017", "series": "001..."}

Рисунок 1.4 – Таблица «employees»

Структура и часть данных таблицы «immovables» приведены на рисунке 1.5.

	id [PK] integer	id_type smallint	address jsonb	parameters jsonb
1	1	2	{ "flat": 18, "home": 104, "street": "4-й Котельнический Переулок", "district": "Отрадное" }	{ "cost": 4217, "other": "в хорошем сос..." }
2	2	2	{ "flat": 56, "home": 64, "street": "Большая Грузинская Улица", "district": "Хорошёвский" }	{ "cost": 2559, "other": "в хорошем сос..." }
3	3	4	{ "flat": 189, "home": 92, "street": "6-я Кожуховская Улица", "district": "Дегунино Западное" }	{ "cost": 2656, "other": "в хорошем сос..." }
4	4	4	{ "flat": 79, "home": 99, "street": "Валовая Улица", "district": "Фили-Давыдково" }	{ "cost": 4244, "other": "в хорошем сос..." }
5	5	2	{ "flat": 190, "home": 86, "street": "1-й Рогозининский Переулок", "district": "Метрогородок" }	{ "cost": 2284, "other": "в хорошем сос..." }
6	6	2	{ "flat": 54, "home": 88, "street": "8-я Текстильщиков Улица", "district": "Строгино" }	{ "cost": 4653, "other": "в хорошем сос..." }
7	7	5	{ "flat": 196, "home": 25, "street": "7-я Текстильщиков Улица", "district": "Таганский" }	{ "cost": 1210, "other": "в хорошем сос..." }
8	8	5	{ "flat": 186, "home": 88, "street": "3-й Дунайский Переулок", "district": "Орехово-Борисово Северное" }	{ "cost": 767, "other": "в хорошем сос..." }
9	9	1	{ "flat": 32, "home": 89, "street": "Венецианова Улица", "district": "не указано" }	{ "cost": 797, "other": "в хорошем сос..." }
10	10	2	{ "flat": 139, "home": 48, "street": "5 Квартал", "district": "Митино" }	{ "cost": 444, "other": "в хорошем сос..." }
11	11	6	{ "flat": 164, "home": 70, "street": "Газгольдерная Улица", "district": "Черёмушки" }	{ "cost": 3060, "other": "в хорошем сос..." }
12	12	1	{ "flat": 13, "home": 109, "street": "Введенского Улица", "district": "Молжаниновский" }	{ "cost": 1301, "other": "в хорошем сос..." }
13	13	5	{ "flat": 135, "home": 92, "street": "4-й Донской Проезд", "district": "Бибирево" }	{ "cost": 3908, "other": "в хорошем сос..." }
14	14	3	{ "flat": 73, "home": 17, "street": "1-й Спасоаликовский переулок Переулок", "district": "Черёмушки" }	{ "cost": 344, "other": "в хорошем сос..." }
15	15	5	{ "flat": 62, "home": 22, "street": "Академика Самарского Улица", "district": "Ломоносовский" }	{ "cost": 183, "other": "в хорошем сос..." }
16	16	5	{ "flat": 60, "home": 12, "street": "Врачебный Проезд", "district": "Тушино Южное" }	{ "cost": 1714, "other": "в хорошем сос..." }
17	17	3	{ "flat": 65, "home": 97, "street": "2-я Белогорская Улица", "district": "Фили-Давыдково" }	{ "cost": 820, "other": "в хорошем сос..." }
18	18	5	{ "flat": 129, "home": 92, "street": "16 Квартал", "district": "Крюково" }	{ "cost": 143, "other": "в хорошем сос..." }
19	19	3	{ "flat": 189, "home": 17, "street": "Бескудниковский проезд Проезд", "district": "Гагаринский" }	{ "cost": 4300, "other": "в хорошем сос..." }
20	20	4	{ "flat": 118, "home": 28, "street": "Васильевский Спуск Площади", "district": "Старое Крюково" }	{ "cost": 60, "other": "в хорошем сос..." }
21	21	1	{ "flat": 140, "home": 59, "street": "9-й Марьиной Роши Проезд", "district": "Аэропорт" }	{ "cost": 3135, "other": "в хорошем сос..." }

Рисунок 1.5 – Таблица «immovables»

Структура и часть данных таблицы «immovalestypes» приведены на рисунке 1.6.

	id [PK] smallint	name text
1	1	частный дом
2	2	квартира
3	3	комната
4	4	промышленное помещение
5	5	офис
6	6	торговое помещение

Рисунок 1.6 – Таблица «immovalestypes»

Структура и часть данных таблицы «owners» приведены на рисунке 1.7.

	id [PK] integer	full_name text	address jsonb	phone_number text	e_mail text	passport_data jsonb
1	1	ААА ААБ	{"flat": 151, "home": 92, "st...	07070707070707070	aaaaa@a@a...	{"num": "000000", "date": ...
2	2	ААВ ААГ	{"flat": 165, "home": 77, "st...	07070707070707071	aaaca@a@a...	{"num": "000001", "date": ...
3	3	ААД ААЕ	{"flat": 34, "home": 97, "str...	07070707070707072	aaaea@a@a...	{"num": "000002", "date": ...
4	4	ААЁ ААЖ	{"flat": 93, "home": 54, "str...	07070707070707073	aaaga@a@a...	{"num": "000003", "date": ...
5	5	ААЗ ААИ	{"flat": 71, "home": 82, "str...	07070707070707074	aaail@a@a@j	{"num": "000004", "date": ...
6	6	ААЙ ААК	{"flat": 170, "home": 43, "st...	07070707070707075	aaaka@a@a@l	{"num": "000005", "date": ...
7	7	ААЛ ААМ	{"flat": 93, "home": 13, "str...	07070707070707076	aaama@a@a...	{"num": "000006", "date": ...
8	8	ААН ААО	{"flat": 109, "home": 61, "st...	07070707070707077	aaaoa@a@a...	{"num": "000007", "date": ...
9	9	ААП ААР	{"flat": 147, "home": 24, "st...	07070707070707078	aaaua@a@a...	{"num": "000008", "date": ...
10	10	ААС ААТ	{"flat": 178, "home": 32, "st...	07070707070707079	aaawa@a@a...	{"num": "000009", "date": ...

Рисунок 1.7 – Таблица «owners»

Структура и часть данных таблицы «ownersrequests» приведены на рисунке 1.7.

id integer	id_owner [PK] integer	id_immovable integer	preferences text
583	1	144	в хорошем состоянии, быстро, возможен торг
880	2	353	в хорошем состоянии, быстро, возможен торг
907	3	17	в хорошем состоянии, быстро, возможен торг
976	4	433	в хорошем состоянии, быстро, возможен торг
690	5	386	в хорошем состоянии, быстро, возможен торг
992	6	279	в хорошем состоянии, быстро, возможен торг
958	7	334	в хорошем состоянии, быстро, возможен торг
912	8	307	в хорошем состоянии, быстро, возможен торг
721	9	359	в хорошем состоянии, быстро, возможен торг
776	10	458	в хорошем состоянии, быстро, возможен торг
724	11	108	в хорошем состоянии, быстро, возможен торг
883	12	498	в хорошем состоянии, быстро, возможен торг
933	13	340	в хорошем состоянии, быстро, возможен торг
916	14	243	в хорошем состоянии, быстро, возможен торг
645	15	220	в хорошем состоянии, быстро, возможен торг
977	16	97	в хорошем состоянии, быстро, возможен торг
630	17	179	в хорошем состоянии, быстро, возможен торг

Рисунок 1.7 – Таблица «ownersrequests»

2. Технические требования

Основные технические требования:

- Программа должна быть написана на Си++ используя кросс-платформенный инструментальный разработки Qt или аналога;
- программа должна хранить данные в БД;
- программа должна уметь работать по сети с СУБД PostgreSQL;
- программа должна подключаться к СУБД;
- программа должна отображать данные из таблиц и/или представлений;
- программа должна добавлять данные в БД;
- программа должна редактировать данные в БД;
- программа должна удалять данные из БД [2].

3. **Функциональные возможности**

Программа написана на Си++ с использованием кроссплатформенного инструментария разработки Qt. Программа хранит данные в БД и умеет работать по сети с СУБД PostgreSQL. То есть она:

- подключается к СУБД;
- отображает данные из таблиц;
- добавляет данные в БД;
- редактирует данные в БД;
- удаляет данные из БД.

Программа состоит из главного окна и диалоговых окон. В главном окне простое меню для реализации основных задач обработки данных.

Весь интерфейс программы на английском языке, но есть возможность переключиться на русский и немецкий. Программа запоминает настройки интерфейса [3].

Реализована возможность поиска и сортировки по отдельным данным.

В БД реализована схема из нескольких таблиц, связанных через внешние ключи.

4. Интерфейс программы

4.1. Главное окно

На рисунке 4.1 представлено главное окно программы.

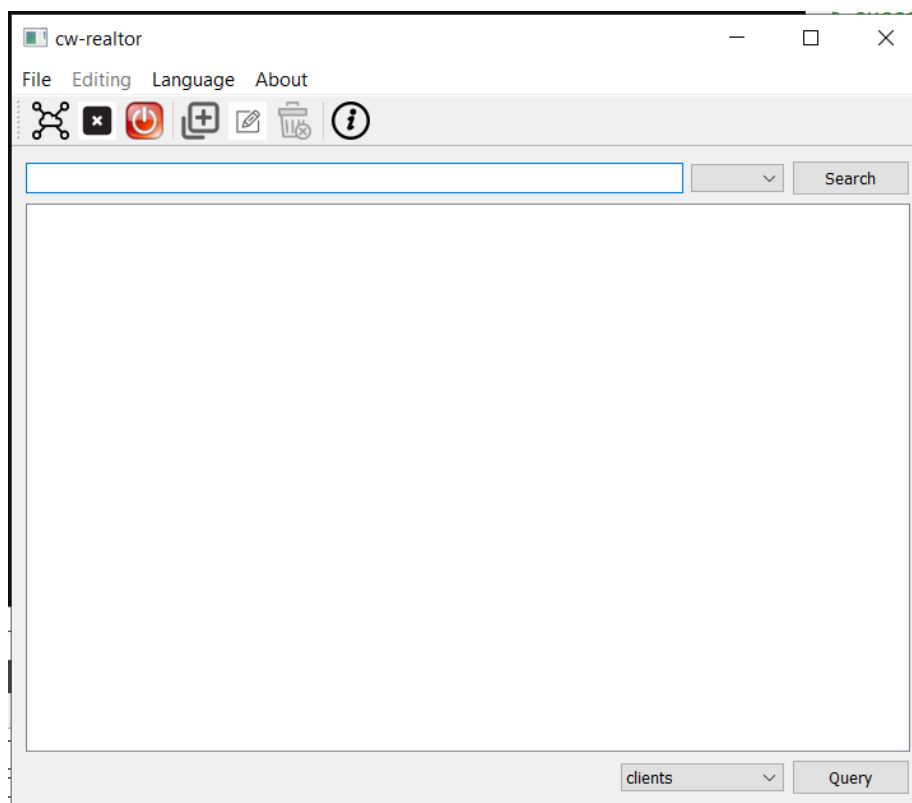


Рисунок 4.1 – Главное окно программы

На рисунке 4.2 представлен пример работы одно из пунктов меню. Кнопка «Connect» не активна, так как мы уже подключились к базе данных [4].

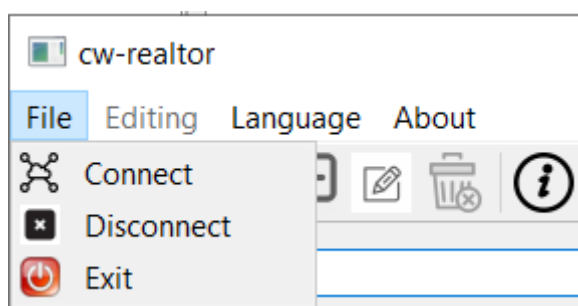


Рисунок 4.2 – Пункт меню

На рисунке 4.3 представлен вывод информации об авторе.

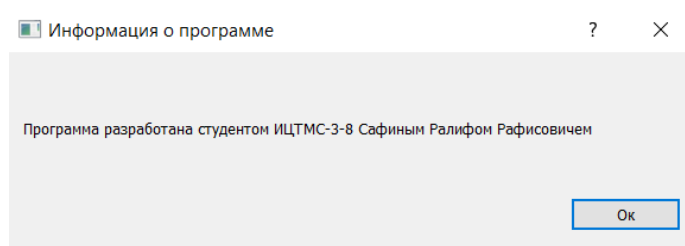


Рисунок 4.3 – Информация об авторе

Программа позволяет выбрать один из трех языков: русский, английский и немецкий. Пример интерфейса программы на английском языке представлен на рисунке 4.4.

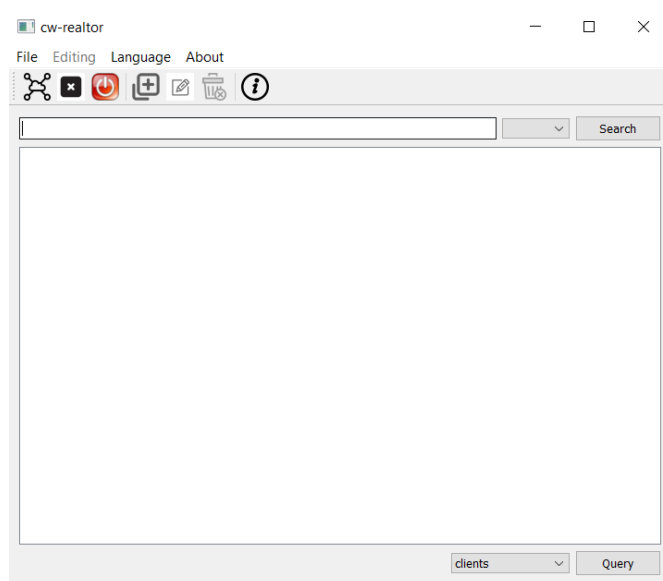


Рисунок 4.4 – Английский интерфейс

4.2. Работа с записями в документе

Для получения возможности работать с базой данных, к ней изначально нужно подключиться. Это делается нажатием кнопки «Подключение» (см. рис. 4.5) [5].

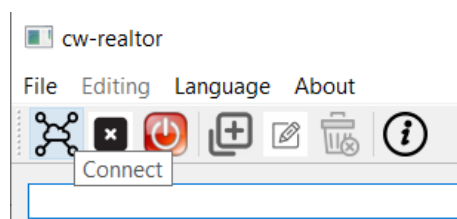


Рисунок 4.5 – Диалоговое окно «Подключение»

Добавление строки в базу данных или ее редактирование происходит в диалоговом окне «Редактирование»

Удаление строки из базы данных происходит путем выбора конкретной строки и нажатия кнопки «Удалить».

В программе также реализован поиск по названию каждым столбцам таблиц. Также реализована сортировка по столбцам.

5. Описание исходного кода программы

5.1. Иерархия классов

На рисунке 5.1 представлена иерархия классов, созданных в проекте.

▼ C QDialog	
C dialogAuthor	Класс dialogAuthor - наследует QDialog и предоставляет функциональность диалогового окна автора
C dialogDelete	Класс dialogDelete - наследует QDialog и предоставляет функциональность диалогового окна удаления
C dialogEdit	Класс dialogEdit - наследует QDialog и предоставляет функциональность диалогового окна редактирования
C dialogRedaction	Класс dialogRedaction - наследует QDialog и предоставляет функциональность диалогового окна редакции
▼ C QMainWindow	
C MainWindow	
▼ C QSqlQueryModel	
C modell	

Рисунок 5.1 – Иерархия классов

5.2 Описание классов и их методов

5.2.1 Класс dialogAuthor


На рисунке 5.2 представлено полное описание класса database из Doxygen.

Класс dialogAuthorОткрытые члены | Полный список членов класса

Класс **dialogAuthor** - наследует QDialog и предоставляет функциональность диалогового окна автора [Подробнее...](#)

```
#include <dialogauthor.h>
```

Граф наследования:dialogAuthor:



```
graph BT
    dialogAuthor --> QDialog
```

Открытые члены

dialogAuthor (QWidget *parent=nullptr)
Конструктор класса Подробнее...
~dialogAuthor ()
Деструктор класса

[Подробнее описание](#)

Класс **dialogAuthor** - наследует QDialog и предоставляет функциональность диалогового окна автора

Рисунок 5.2 – Класс dialogAuthor

5.2.2 Класс dialogDelete

На рисунке 5.3 представлено полное описание класса dialogDelete.

Класс dialogDelete

Класс `dialogDelete` - наследует `QDialog` и предоставляет функциональность диалогового окна удаления Подробнее...

#include <dialogDelete.h>

Граф наследования:dialogDelete:

QDialog

dialogDelete

Открытые члены

dialogDelete (QWidget *parent=nullptr)

Конструктор класса Подробнее...

~dialogDelete ()

Деструктор класса

int num ()

Получение числового значения, введенного пользователем Подробнее...

Подробное описание

Класс `dialogDelete` - наследует `QDialog` и предоставляет функциональность диалогового окна удаления

Конструктор(ы)

◆ dialogDelete()

dialogDelete::dialogDelete (QWidget * parent = nullptr)

Конструктор класса

Аргументы

parent Родительский виджет диалогового окна (необязательный)

Методы

◆ num()

int dialogDelete::num ()

Получение числового значения, введенного пользователем

Возвращает

Введенное число

Рисунок 5.3 – Класс dialogDelete.

5.2.3 Класс dialogEdit

На рисунке 5.4 представлено полное описание класса dialogEdit.

Класс dialogEdit

Класс `dialogEdit` - наследует `QDialog` и предоставляет функциональность диалогового окна редактирования Подробнее...

#include <dialogedit.h>

Граф наследования:dialogEdit:

QDialog

dialogEdit

Открытые члены

dialogEdit (QWidget *parent=nullptr)

Конструктор класса Подробнее...

~dialogEdit ()

Деструктор класса

int num ()

Получение числового значения, введенного пользователем Подробнее...

Подробное описание

Класс `dialogEdit` - наследует `QDialog` и предоставляет функциональность диалогового окна редактирования

Конструктор(ы)

◆ dialogEdit()

dialogEdit::dialogEdit (QWidget * parent = nullptr)

Конструктор класса

Аргументы

parent Родительский виджет диалогового окна (необязательный)

Методы

◆ num()

int dialogEdit::num ()

Получение числового значения, введенного пользователем

Возвращает

Введенное число

Рисунок 5.4 – Класс dialogEdit

5.2.4 Класс dialogRedaction

На рисунке 5.5 представлено полное описание класса dialogRedaction.

Класс dialogRedaction

Класс `dialogRedaction` - наследует `QDialog` и предоставляет функциональность диалогового окна редакции Подробнее...

#include <dialogredaction.h>

Граф наследования: dialogRedaction:

QDialog

dialogRedaction

Открытые члены

dialogRedaction (QWidget *parent=nullptr)

Конструктор класса Подробнее...

~dialogRedaction ()

Деструктор класса

void getData (QString ¶m1, QString ¶m2, QString ¶m3, QString ¶m4, QString ¶m5, QString ¶m6, QString ¶m7)

Получение значений полей из диалогового окна Подробнее...

void editLabel (QString text)

Устанавливает текст на метке (label) диалогового окна Подробнее...

Подробнее описание

Класс `dialogRedaction` - наследует `QDialog` и предоставляет функциональность диалогового окна редакции

Конструктор(ы)

◆ dialogRedaction()

dialogRedaction: dialogRedaction (QWidget * parent = nullptr)

Конструктор класса

Аргументы

parent Родительский виджет диалогового окна (необязательный)

Методы

◆ editLabel()

void dialogRedaction::editLabel (QString text)

Устанавливает текст на метке (label) диалогового окна

Аргументы

text Текст, который будет установлен на метке

◆ getData()

void dialogRedaction::getData (QString & param1, QString & param2, QString & param3, QString & param4, QString & param5, QString & param6, QString & param7)

Получение значений полей из диалогового окна

Аргументы

[in,out] param1 Значение первого поля

[in,out] param2 Значение второго поля

[in,out] param3 Значение третьего поля

[in,out] param4 Значение четвертого поля

[in,out] param5 Значение пятого поля

[in,out] param6 Значение шестого поля

[in,out] param7 Значение шестого поля

Рисунок 5.5 – Класс dialogRedaction.

5.2.5 Класс MainWindow

20

На рисунке 5.6 представлено полное описание класса MainWindow.



Рисунок 5.6 – Класс MainWindow

5.2.6 Класс Model

На рисунке 5.7 представлено полное описание класса Model.

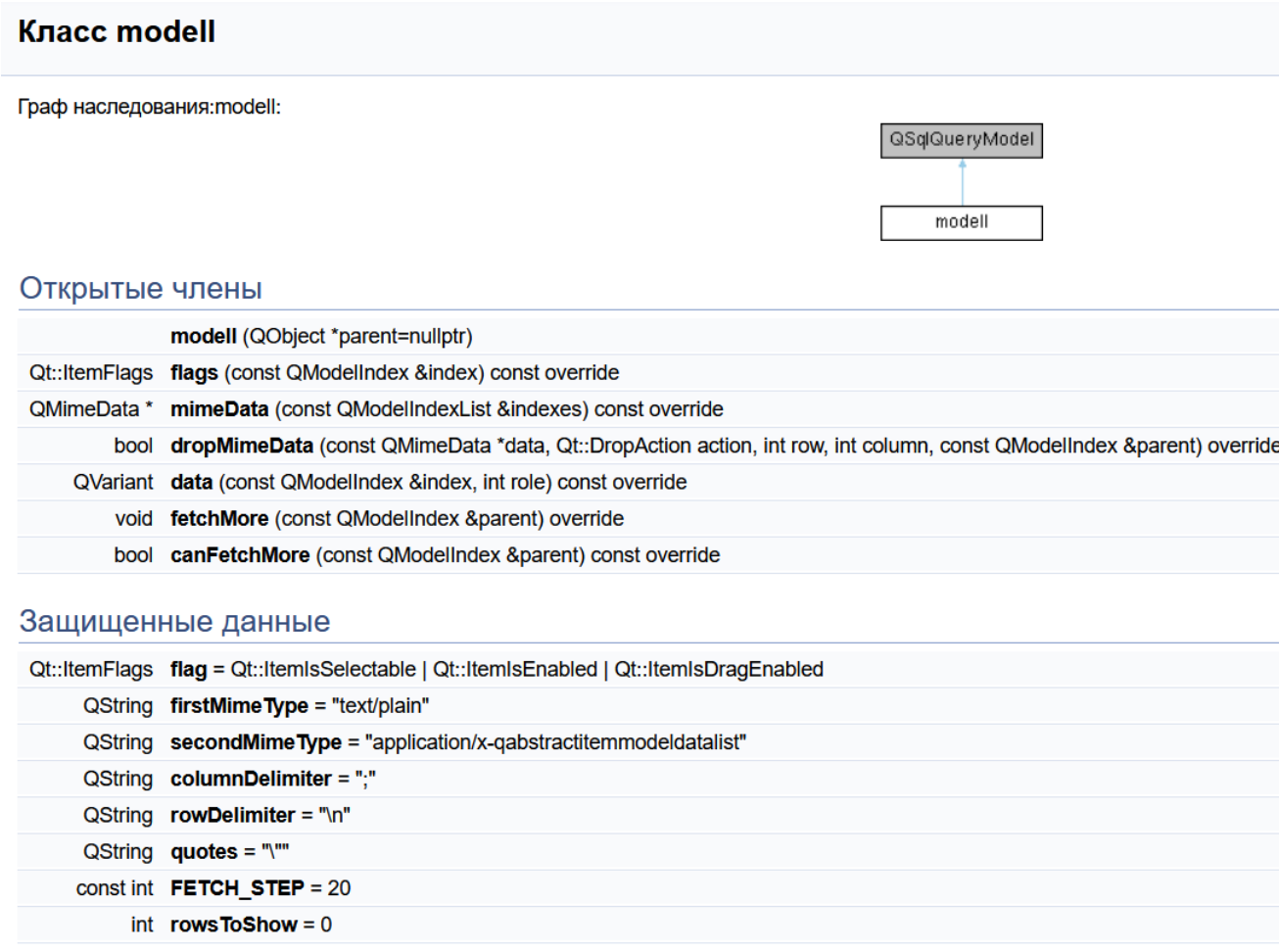


Рисунок 5.7 – Класс Model

6. Список файлов с исходным кодом и вспомогательных

На рисунке 6.1 изображена вся файловая структура программы, включая вспомогательные файлы.

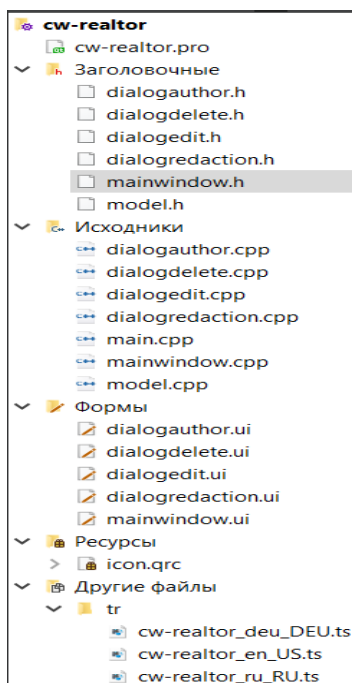


Рисунок 6.1 – Список файлов проекта

ЗАКЛЮЧЕНИЕ

В данной программе с помощью графических инструментов Qt Creator, используя основные механизмы ООП, такие как: наследование, полиморфизм и инкапсуляция, было создано ПО для работы с базой данных риэлторской конторы.

Данная программа позволяет:

- хранить данные в БД;
- подключается к СУБД;
- отображать данные из таблиц и/или представлений;
- добавлять данные в БД;
- редактировать данные в БД;
- удалять данные из БД.

Выполнив эту работу, улучшил навыки по работе с пользовательским интерфейсом, классами и объектами Си++, а также научился работать с изменением языка в Qt с помощью программы Linguist. Получил навыки использования PostgreSQL [7]. Получил навыки установки и настройки ОС Linux.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Садовский Б. С. Конспект лекций по курсу “Объектно-ориентированное программирование” – Изд. № Б/Н – Москва: МИСИ-МГСУ, 2019.
2. Основы программирования на языках Си и С++ для начинающих [Электронный ресурс] URL: <http://cppstudio.com> (дата обращения 10.05.2023)
3. Д., Раскин. Интерфейс: новые направления в проектировании компьютерных систем / Раскин Д. — 3-е изд. — СПб.: Символ-плюс, 2007. — 272 с.
4. Керниган, Брайан У. Язык программирования С / Брайан У. Керниган, Деннис М. Ритчи. — 2-е изд. — М.: ООО “И. Д. Вильямс”, 2008. — 304 с.
5. Форум программистов, системных администраторов, администраторов баз данных, компьютерный форум, форум по электронике и бытовой технике, обсуждение софта. [Электронный ресурс] URL: <https://www.cyberforum.ru> (дата обращения 20.05.2023)
6. Форум программистов и системных администраторов [Электронный ресурс] URL: <https://stackoverflow.com> (дата обращения 20.06.2023)
7. Хабибуллин, И. Программирование на языке высокого уровня. С/С++ / И. Хабибуллин. - М.: БХВ-Петербург, 2006. — 512 с.