

Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и кибербезопасности
Высшая школа программной инженерии

Курсовой проект
по дисциплине
«Базы данных»
«Автоматизация автосервиса»

Выполнил
Студент гр. 5130202/10201

 **Хакимуллина А.М.**

Руководитель

Юркин В.А.

Санкт-Петербург
2023 г.

Содержание

Постановка задачи	3
Анализ предметной области	4
Хеширование паролей	12
Реализация	15
Запросы	16
Клиентская часть	21
Регистрация/Авторизация	21
Вход в качестве мастера	24
Вход в качестве оператора	27
Вход в качестве бухгалтера	42
Вывод	45
Список использованной литературы	46

Постановка задачи

Целью данной работы является разработка GUI для базы данных, представляющую собой работу автосервиса.

БД представляет собой вид представленный на рисунке 1.

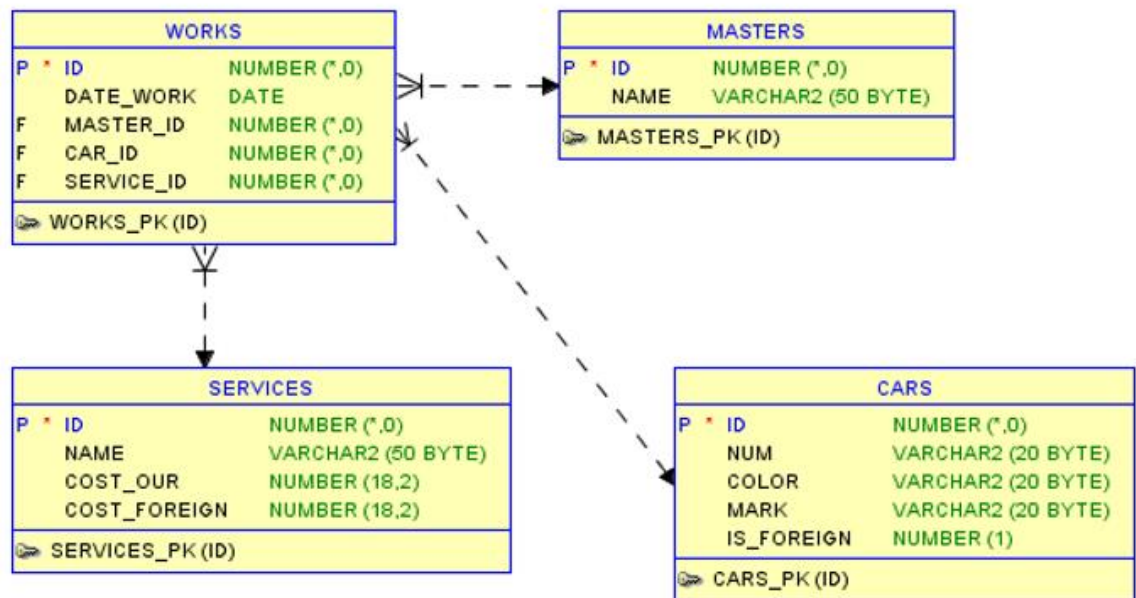


Рисунок 1 База данных автосервиса

Разработка осуществлялась с помощью языка программирования Java, для создания графического интерфейса использовался javaFX.

Разрабатываемый сервис предоставляет следующие функции:

- Регистрация и авторизация с хешированием паролей в базе данных;
- Возможность выбора типа пользователя при регистрации;
- Добавление, редактирование, удаление записей из таблицы мастеров;
- Добавление, редактирование, удаление записей из таблицы автомобилей;
- Добавление, редактирование, удаления записей из таблицы услуг;
- Добавление, редактирование, удаление записей проведенных работ;
- Реализован интерфейс, позволяющий оператору назначать работы из перечня услуг мастерам автосервиса для обслуживания имеющейся базы клиентов с возможностью указания даты проведения работы.
- Возможность просмотра оператором следующих показателей:
 - Общая стоимость обслуживания отечественных и импортных автомобилей (с возможностью фильтрации по датам оказания услуги).
 - Пять мастеров, которые в заданном месяце выполнили наибольшее число работ для разных автомобилей.

Анализ предметной области

Группы пользователей разрабатываемой информационной системы:

№ пп	Наименование пользователя
1	Оператор
2	Мастер
3	Бухгалтер

Функции оператора:

№ пп	Выполняемая функция	Входные данные	Выходные данные	Функции, которые должны быть реализованы в ИС
1	Вход	Логин, пароль	Доступ к базе данных в качестве оператора	Вход в систему в качестве оператора
2	Просмотр всех мастеров		Таблица masters	Вывод всех мастеров
3	Добавление мастера	Id, имя мастера	Таблица masters	Добавление нового мастера
4	Удаление мастера	Id мастера	Таблица masters	Удаление данного мастера
5	Изменение мастера	Id мастера, имя	Таблица masters	Изменение имени мастера
6	Просмотр		Таблица cars	Вывод всех

	р всех автомоб илей			существующих автомобилей
7	Добавление автомобиля	Id, номер автомобиля, цвет, марка, иностраннй или нет	Таблица cars	Добавление нового автомобиля
8	Удаление автомобиля	Id автомобиля	Таблица cars	Удаление заданного автомобиля
9	Просмотр всех услуг		Таблица services	Вывод всех услуг
10	Изменение стоимости для отечествен ного автомобиля	Id услуги, стоимость	Таблица services	Изменение стоимости услуги для отечественного автомобиля
11	Изменение стоимости для зарубежног о автомобиля	Id услуги, стоимость	Таблица services	Изменение стоимости услуги для зарубежного автомобиля
12	Добавление услуги	Id, наименование услуги,	Таблица services	Добавление новой услуги

		стоимость для отечественного автомобиля , стоимость для зарубежного автомобиля		
13	Просмотр всех работ		Таблица works	Вывод всех существующих работ
14	Удаление работы	Id работы	Таблица works	Удаление заданный работы
15	Назначение работы	Id работы, дата, id мастера, id автомобиля, id услуги	Таблица works	Назначение новой работы
16	Изменение работы	Id работы, дата	Таблица works	Изменение даты работы
17	Просмотр общей стоимости работ в определенн ый период	Дата начала, дата концы	Общая стоимость работ в заданный период	Вывод общей стоимости работ в заданный период
18	Просмотр 5 мастеров, которые в	Номер месяца, год	Список 5 мастеров	Вывод списка и мастеров, которые в заданном месяце

	заданном месяце выполнили наибольшее количество работ для разных автомобилей			выполнили наибольшее количество работ для разных автомобилей
19	Вывод отчета в формате txt	Имя файла	Отчет в формате txt	Вывод отчета в формате txt
20	Вывод отчета в формате xls	Имя файла	Отчет в формате xls	Вывод отчета в формате xls

Функции мастера:

№ пп	Выполняемая функция	Входные данные	Выходные данные	Функции, которые должны быть реализованы в ИС
1	Вход	Логин, пароль	Доступ к базе данных в качестве мастера	Вход в систему в качестве мастера
2	Просмотр всех мастеров		Таблица masters	Вывод всех мастеров

3	Просмотр всех автомоби лей		Таблица cars	Вывод всех существующих автомобилей
4	Просмотр всех услуг		Таблица services	Вывод всех услуг
5	Просмотр всех работ		Таблица works	Вывод всех существующих работ

Функции бухгалтера:

№ пп	Выполняемая функция	Входные данные	Выходные данные	Функции, которые должны быть реализованы в ИС
1	Вход	Логин, пароль	Доступ к базе данных в качестве бухгалтера	Вход в систему в качестве бухгалтера
2	Просмотр всех мастеров		Таблица masters	Вывод всех мастеров
3	Просмотр всех автомоби лей		Таблица cars	Вывод всех существующих автомобилей
4	Просмотр		Таблица	Вывод всех

	всех услуг		services	услуг
5	Просмотр всех работ		Таблица works	Вывод всех существующих работ
6	Подсчет зарплаты с премией	Id мастера, зарплата	Зарплата с премией	Вывод зарплаты с премией

Подключение к базе данных

Чтобы прочитать параметры подключения к базе данных используется файл конфигурации config.xml, который включает элементы для каждого параметра подключения:

```
<database>
  <parameter
name="url">jdbc:postgresql://localhost:5432/car_service</parameter>
  <parameter name="username">postgres</parameter>
  <parameter name="password">1234</parameter>
</database>
```

В этом примере мы создаем корневой элемент <database>, который содержит три элемента <parameter>, представляющих параметры подключения к базе данных. Каждый элемент <parameter> имеет атрибут name, который указывает имя параметра, и текстовое содержимое, которое указывает значение параметра.

Чтобы прочитать параметры подключения из файла конфигурации в Java, мы используем классы javax.xml.parsers.DocumentBuilder и javax.xml.parsers.DocumentBuilderFactory для создания документа XML и извлечения параметров из него. Я создала класс Config, который содержит метод initializeConfig, чтобы прочитать параметры подключения из файла конфигурации:

```
package project;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;
import java.io.File;

public class Config {
    public static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";
    public static String DB_URL;
    public static String USER;
    public static String PASS;
    public static void initializeConfig() {
        try {
            // Создать новый документ XML
            DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
            DocumentBuilder builder = factory.newDocumentBuilder();
            Document document;
```

```

// Прочитать XML-файл и создать документ XML
File inputFile = new File("config.xml");
document = builder.parse(inputFile);
document.getDocumentElement().normalize();

// Извлечь параметры из документа XML
NodeList nodeList = document.getElementsByTagName("parameter");
for (int i = 0; i < nodeList.getLength(); i++) {
    Node node = nodeList.item(i);
    if (node.getNodeType() == Node.ELEMENT_NODE) {
        Element element = (Element) node;
        String name = element.getAttribute("name");
        String value = element.getTextContent();

        if (name.equals("url")) {
            DB_URL = value;
        } else if (name.equals("username")) {
            USER = value;
        } else if (name.equals("password")) {
            PASS = value;
        }
    }
}

System.out.println("Параметры успешно извлечены из файла
конфигурации XML.");
} catch (Exception e) {
    e.printStackTrace();
}
}
}

```

Эта функция используется в классе HelloApplication:

```

public class HelloApplication extends Application {
    @Override
    public void start(Stage stage) throws Exception {
        initializeConfig();
        ...
    }
    ...
}

```

Хеширование паролей

Для хеширования паролей в PostgreSQL был использован алгоритм SHA-256.

SHA-256 - это криптографический алгоритм хеширования, который принимает на вход некоторую строку данных и генерирует из нее хеш-значение фиксированной длины (256 бит). Хеш-значение представляет собой уникальную последовательность символов, которая является цифровой подписью входных данных.

SHA-256 основан на алгоритме SHA-1, но использует более длинный блок данных (256 бит) и более сложную функцию хеширования. Это делает его более устойчивым к атакам подбора паролей и другим формам криптографического взлома.

Алгоритм SHA-256 работает следующим образом:

1. Получение входных данных: алгоритм принимает на вход некоторую строку данных, которую необходимо хешировать.
2. Создание экземпляра SHA-256: для выполнения хеширования мы должны создать экземпляр алгоритма SHA-256. В Java мы можем использовать класс `MessageDigest` для этого.
3. Инициализация экземпляра SHA-256: после создания экземпляра SHA-256 мы должны инициализировать его с использованием алгоритма SHA-256.
4. Поэтапное хеширование: алгоритм SHA-256 выполняет несколько раундов хеширования, каждый из которых включает в себя следующие шаги:
 - 4.1 Расширение входных данных: входные данные расширяются путем добавления нулевых символов до достижения длины, кратной 512 бит.

4.2 Загрузка блока данных: блок данных, состоящий из 128 бит, загружается из входных данных.

4.3 Циклический сдвиг: каждый символ входных данных циклически сдвигается влево на определенное количество позиций.

4.4 Хеширование с использованием функции хеширования: блок данных хешируется с использованием функции хеширования SHA-256.

4.5 Комбинирование и повторение: полученное значение хеширования комбинируется с предыдущим значением хеширования и повторяется до тех пор, пока все входные данные не будут хешированы.

5 Получение хеш-значения: после завершения всех раундов хеширования мы получаем окончательное хеш-значение, которое является цифровой подписью входных данных.

Пример применения данного алгоритма:

```
private static final String ALGORITHM = "SHA-256";

...

private void register() {
    ...
    String hashedPassword = hashPassword(password);
    User user = new User(login, hashedPassword, role);
    ...
}

public static String hashPassword(String password) {
    try {
        MessageDigest messageDigest = MessageDigest.getInstance(ALGORITHM);
        byte[] passwordBytes = password.getBytes();
        byte[] hashedPassword = messageDigest.digest(passwordBytes);
        return bytesToHex(hashedPassword);
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e);
    }
}

private static String bytesToHex(byte[] bytes) {
    StringBuilder hex = new StringBuilder();
    for (byte b : bytes) {
```

```

        hex.append(String.format("%02x", b));
    }
    return hex.toString();
}

```

Как выглядят данные о пользователях в базе данных:

Query		Query History	
1		select * from users	
Data Output		Notifications	
Messages			
	login character varying (100)	password character varying (100)	role character varying (50)
1	oper12	cbfad02f9ed2a8d1e08d8f74f5303e9eb93637d47f82ab6f1c15871cf8dd04...	operator
2	master34	5f395d07369071a505ef926527de2ac53e8c29e103dc63398315bc276224...	master
3	acc09	9552e277ebcc7fa191292c6e900d94dfe6e837d8abf76a2da927d4530d2c...	accountant
4	master678	d2dc85fed2104598a780a20f44dff7573ee5a67c3bb1d38fd1ffa2c8b66f4a...	master

Рисунок 2 Таблица с данными о пользователях

SHA-256 широко используется в криптографии, безопасности и других областях, где требуется генерация уникальных хеш-значений. Он обеспечивает высокую степень безопасности и устойчивости к атакам, что делает его идеальным для защиты конфиденциальности и целостности данных в различных приложениях.

Реализация

SQL-скрипт, создающий необходимые для обеспечения функциональности и работоспособности системы таблицы:

Query	Query History
1	CREATE TABLE cars
2	(
3	car_id serial PRIMARY KEY ,
4	num int UNIQUE NOT NULL ,
5	color varchar(30) NOT NULL ,
6	mark varchar(50) NOT NULL ,
7	is_foreign bool NOT NULL
8);
9	
10	CREATE TABLE masters
11	(
12	master_id serial PRIMARY KEY ,
13	name varchar(50) NOT NULL
14);
15	
16	
17	CREATE TABLE services
18	(
19	service_id serial PRIMARY KEY ,
20	name varchar(50) NOT NULL ,
21	cost_our int NOT NULL ,
22	cost_foreign int NOT NULL
23);
24	
25	CREATE TABLE works
26	(
27	work_id serial PRIMARY KEY ,
28	date_work date NOT NULL ,
29	fk_master_id int REFERENCES masters(master_id) NOT NULL ,
30	fk_car_id int REFERENCES cars(car_id) NOT NULL ,
31	fk_service_id int REFERENCES services(service_id) NOT NULL
32);

Таблица с данными о пользователях:

```
Query  Query History
1  CREATE TABLE users(
2      id serial PRIMARY KEY,
3      login varchar(50) UNIQUE NOT NULL,
4      password varchar(50) NOT NULL,
5      role varchar(50) NOT NULL
6  );
```

Таблица users:

Query	Query History
1	<pre>select * from users</pre>

Data Output	Notifications	Messages
<div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div><div></div></div>		
login	password	role
character varying (100)	character varying (100)	character varying (50)
1 oper12	cbfad02f9ed2a8d1e08d8f74f5303e9eb93637d47f82ab6f1c15871cf8dd04...	operator
2 master34	5f395d07369071a505ef926527de2ac53e8c29e103dc63398315bc276224...	master
3 acc09	9552e277ebcc7fa191292c6e900d94dfe6e837d8abf76a2da927d4530d2c...	accountant
4 master678	d2dc85fed2104598a780a20f44dff7573ee5a67c3bb1d38fd1ffa2c8b66f4a...	master

Была создана дополнительная таблица для пользователей users, со столбцами login, password, role. Пользователи хранятся в базе, и их пароли хранятся там же в захешированном виде. При входе сравниваются переданный захешированный пароль с паролем в таблице. При регистрации записывается в базу логин, хешированный пароль и роль пользователя.

Запросы

Просмотр всех мастеров:

```
SELECT * FROM masters ORDER BY master_id
```

Изменение имени мастера:

```
update masters set name= 'Анна крылова' where master_id = 4
```

Добавить мастера:

```
INSERT INTO masters (master_id, name) VALUES (7, 'Михаил Карпов')
```

Удалить мастера:

DELETE FROM masters WHERE master_id = 2

Просмотр всех автомобилей:

SELECT * FROM cars ORDER BY car_id

Добавить автомобиль:

INSERT INTO cars VALUES (11, 'л486пи122', 'white', 'Lada', false)

Триггер, не позволяющий добавить автомобиль с уже существующим номером:

```
CREATE OR REPLACE FUNCTION warn_duplicate_car_num()
RETURNS TRIGGER
AS $$
BEGIN
IF EXISTS
(SELECT 1 FROM cars WHERE num = NEW.num) THEN
RAISE EXCEPTION 'Автомобиль с таким номером уже существует';
END IF;
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
CREATE TRIGGER check_unique_car_num
BEFORE INSERT OR UPDATE ON cars
FOR EACH ROW
EXECUTE FUNCTION warn_duplicate_car_num();
```

Удалить автомобиль:

DELETE FROM cars WHERE car_id =5

Просмотр всех услуг:

SELECT * FROM services ORDER BY service_id

Изменение стоимости услуги для отечественных автомобилей:

update services set cost_our = 5500 where service_id =8

Изменение стоимости услуги для импортных автомобилей:

update services set cost_foreign= 8000 where service_id =3

Добавить услугу:

INSERT INTO services VALUES (13, 'Замена колес', 5000, 6000)

Удалить услугу:

DELETE FROM services WHERE service_id = 6

Просмотр всех работ:

```
SELECT * FROM works ORDER BY work_id
```

Назначение работы:

```
INSERT INTO works VALUES (14, 2023-12-04, 7, 3, 9)
```

Триггер на назначение работы, если мастер уже выполнил в этот день больше одной работы:

```
CREATE OR REPLACE FUNCTION warn_master_work_count()
RETURNS TRIGGER
AS $$
BEGIN
IF ((SELECT COUNT(+) FROM works WHERE fk_master_id = NEW.master_id AND
date_work = NEW.date_work) > 1)
THEN
RAISE EXCEPTION 'Мастер уже выполнил более одной работы в этот день.';
END IF;
RETURN NEW;
END;
$$ LANGUAGE plpgsql;
```

```
CREATE TRIGGER check_master_work_count BEFORE INSERT OR UPDATE ON works
FOR EACH ROW
EXECUTE FUNCTION warn_master_work_count()
```

Изменить дату работы:

```
update works set date_work = '2023-10-15' where work_id = 7
```

Триггер, не позволяющий изменить дату работы более чем на один день:

```
CREATE OR REPLACE FUNCTION warn_invalid_date()
RETURNS TRIGGER
AS $$
BEGIN
IF EXISTS
(SELECT 1 FROM works WHERE ((NEW.date_work > date.work + interval '1' day) OR
(NEW.date_work < date.work - interval '1' day)) AND NEW.work.id=work_id) THEN
RAISE EXCEPTION 'Нельзя изменить дату работы более чем на один день!'
END IF;
RETURN NEW;
END;
$$ LANGUAGE plpgsql
```

```
CREATE TRIGGER check_date_work
BEFORE UPDATE ON works
FOR EACH ROW
EXECUTE FUNCTION warn_invalid_date()
```

Удалить работу:

```
DELETE FROM works WHERE work_id = 9
```

Подсчет общей стоимости обслуживания отечественных и импортных автомобилей в определенный период:

```
SELECT SUM(CASE WHEN cars.is_foreign = false THEN services.cost_our ELSE
services.cost_foreign END) AS total_cost
FROM works
JOIN cars ON works.fk_car_id = cars.car_id
JOIN services ON works.fk_service_id = services.service_id
WHERE works.date_work >= '2023-04-15' AND works.date_work <= '2023-05-15'
```

Просмотр 5 мастеров, которые в заданном месяце выполнили наибольшее число работ для разных автомобилей:

```
SELECT masters.master_id, masters.name, COUNT(DISTINCT works.fk_car_id) AS
num_cars_serviced
FROM works +
JOIN masters ON works.fk_master_id = masters.master_id
WHERE works.date_work >= '2023-07-01' AND works.date_work < '2023-08-01'
GROUP BY masters.master_id
ORDER BY num_cars_serviced DESC
LIMIT 5
```

Подсчет зарплаты с учетом премии:

```
CREATE OR REPLACE PROCEDURE get_salary_with_bonus(t_master_id INT, t_salary
NUMERIC, OUT salary_with_bonus NUMERIC)
LANGUAGE plpgsql
AS $$
DECLARE
bonus NUMERIC = 0;
count_of_works INT = 0;
salary_cursor CURSOR FOR
SELECT COUNT (work_id)
FROM works
WHERE fk_master_id = t_master_id AND date_work > date_trunc('month', CURRENT_DATE)
GROUP BY fk_car_id
ORDER BY fk_car_id;
BEGIN
salary_with_bonus:=0;
open salary_cursor;
loop
fetch salary_cursor into count_of_works;
exit when not found;
IF count_of_works < 3 THEN
```

```
bonus := bonus + t_salary*0.05 * count_of_works;  
ELSE  
bonus := bonus + t_salary *0.07 * count_of_works;  
END IF;  
end loop;  
close salary_cursor;  
salary_with_bonus := t_salary + bonus;  
END $$
```

Клиентская часть

Регистрация/Авторизация

Запускаем приложение, открывается окно авторизации:

Авторизация

Логин:

Пароль:

Рисунок 3 Авторизация

При нажатии на кнопку “Нет аккаунта” открывается окно регистрации.

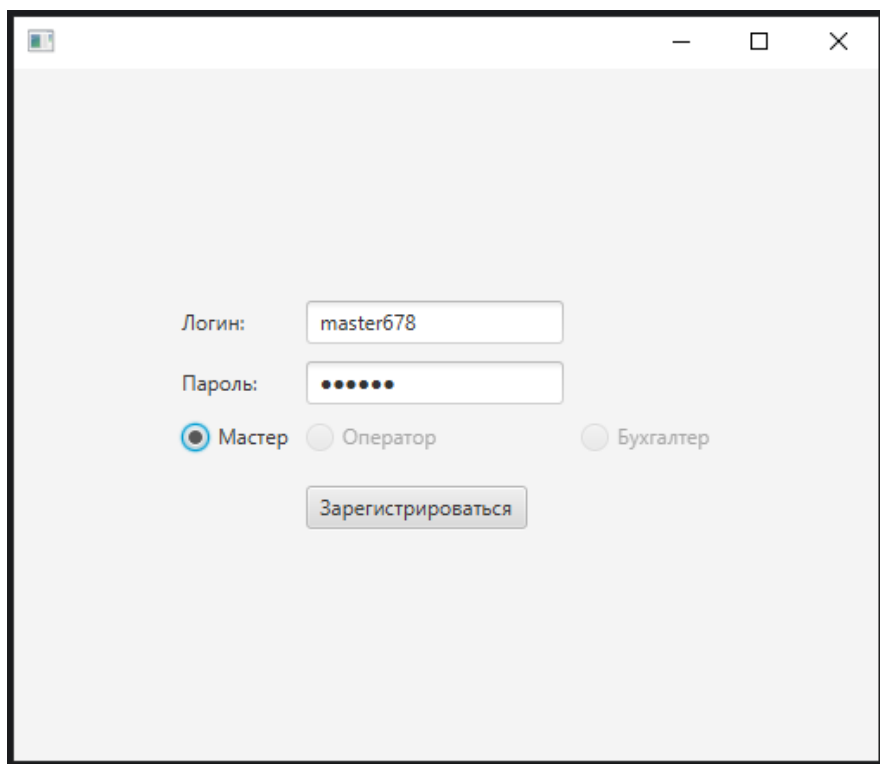
Логин:

Пароль:

☐ Мастер ☐ Оператор ☐ Бухгалтер

Рисунок 4. Регистрация

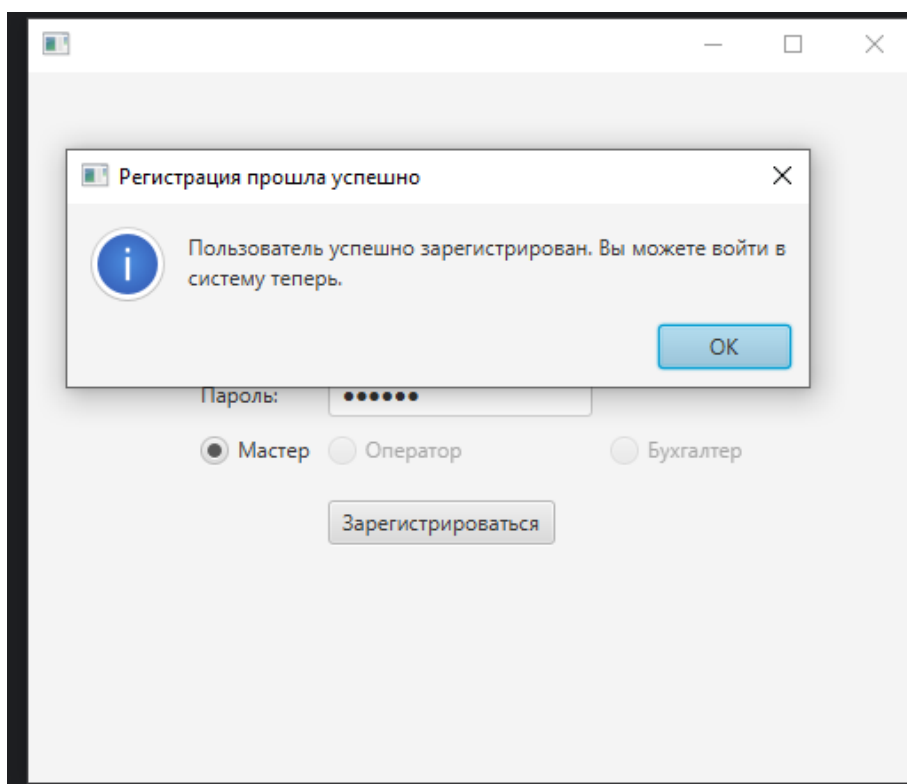
Попробуем зарегистрировать нового мастера:



The screenshot shows a registration window with a light gray background. At the top, there are standard window control buttons (minimize, maximize, close). Below them, the form contains the following elements:

- A label "Логин:" followed by a text input field containing the text "master678".
- A label "Пароль:" followed by a password input field with seven dots.
- Three radio buttons for user roles: "Мастер" (selected with a blue dot), "Оператор", and "Бухгалтер".
- A button labeled "Зарегистрироваться" (Register) located below the radio buttons.

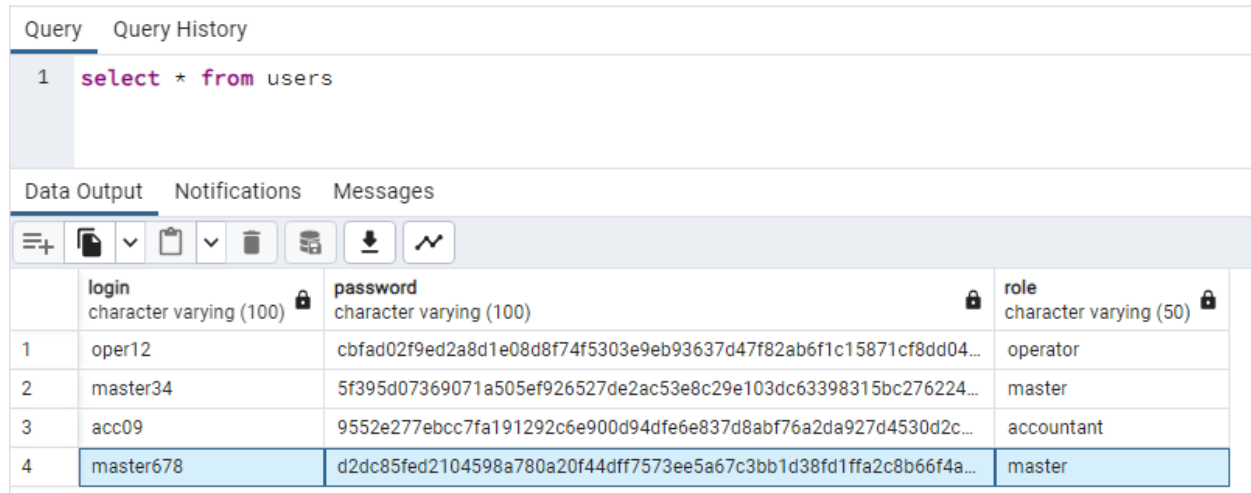
Рисунок 5. Регистрация мастера



This screenshot shows the same registration form as in Figure 4, but with a modal dialog box overlaid on top. The dialog box has a title bar that says "Регистрация прошла успешно" (Registration successful) and a close button (X). Inside the dialog, there is an information icon (i in a blue circle) and the text: "Пользователь успешно зарегистрирован. Вы можете войти в систему теперь." (User successfully registered. You can now log in to the system). At the bottom right of the dialog is an "OK" button. The registration form is visible in the background, showing the password field, the "Мастер" radio button selected, and the "Зарегистрироваться" button.

Рисунок 6. Успешная регистрация мастера

Данные о пользователях сохраняются в таблицу users базы данных, пароли хранятся в хешированном виде:



Query		Query History	
1		select * from users	
Data Output		Notifications	
		Messages	
	login character varying (100)	password character varying (100)	role character varying (50)
1	oper12	cbfad02f9ed2a8d1e08d8f74f5303e9eb93637d47f82ab6f1c15871cf8dd04...	operator
2	master34	5f395d07369071a505ef926527de2ac53e8c29e103dc63398315bc276224...	master
3	acc09	9552e277ebcc7fa191292c6e900d94dfe6e837d8abf76a2da927d4530d2c...	accountant
4	master678	d2dc85fed2104598a780a20f44dff7573ee5a67c3bb1d38fd1ffa2c8b66f4a...	master

Рисунок 7. Таблица users в базе данных

Можем заметить, что новый мастер появился в таблице. Нажимаем на кнопку “Ок”, открывается окно авторизации.

При вводе неправильного логина или пароля выходит сообщение об ошибке:

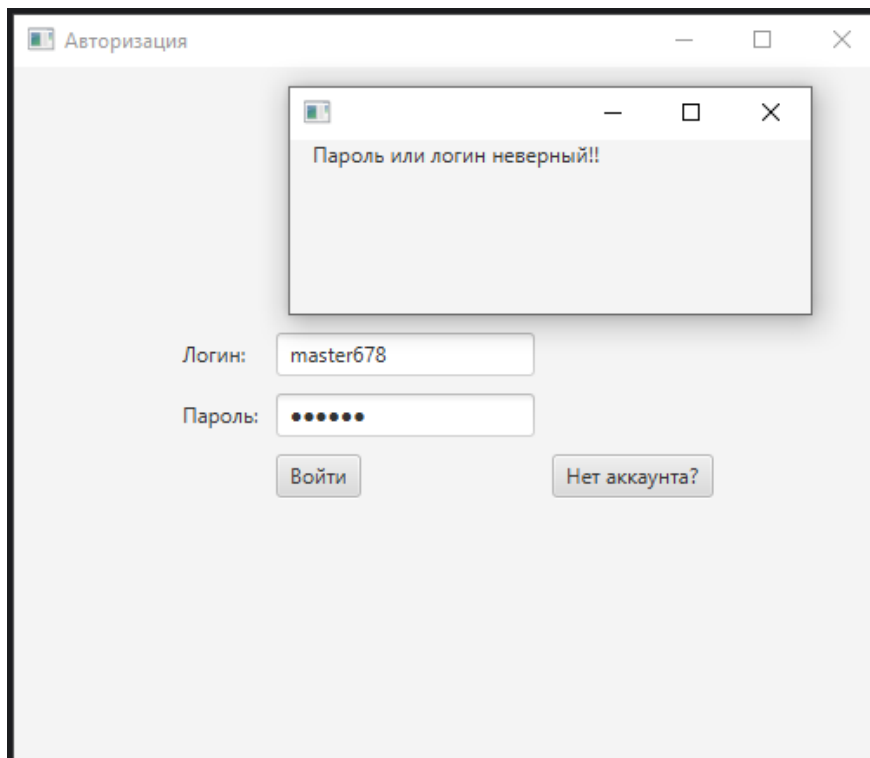


Рисунок 8. Ввод неправильного пароля

Вход в качестве мастера

Войдем в систему как мастер:

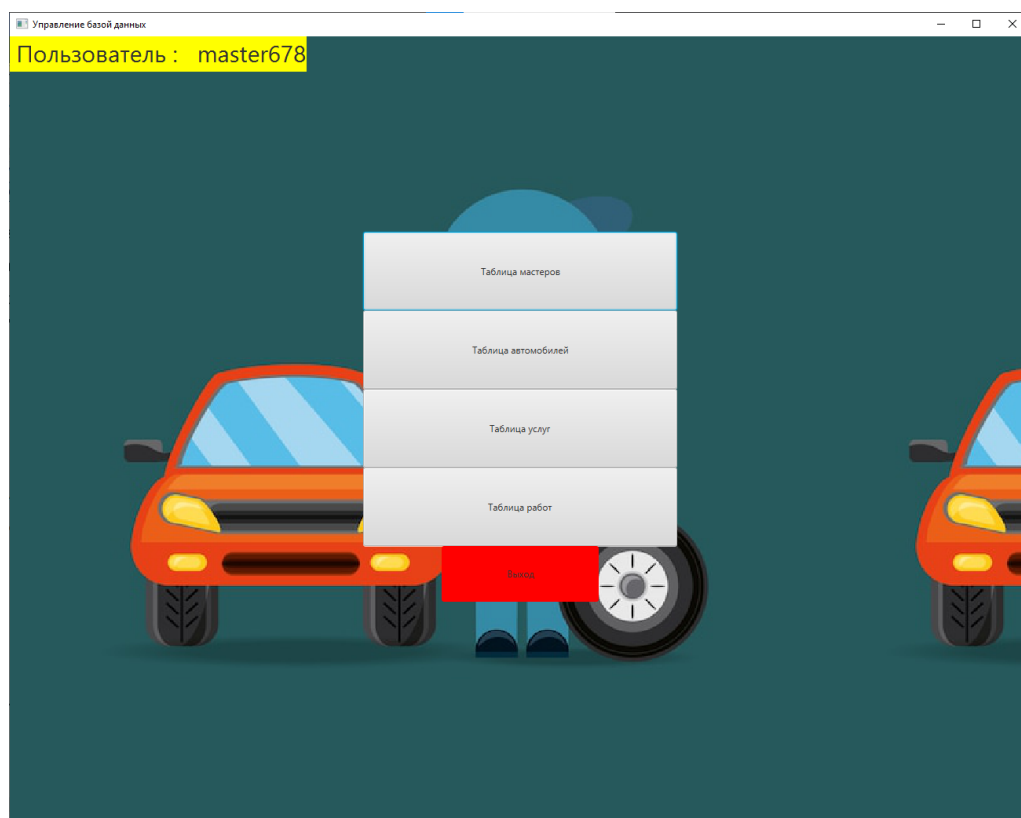


Рисунок 9. Вход в качестве мастера

Мастер может посмотреть данные разных таблиц. Откроем, к примеру, таблицу автомобилей.

Car Table

car_id	num	color	mark	is_foreign
1	к120то78	blue	Lada Vesta	false
2	и523ро23	red	Sollers Atlant	false
3	вп476дз119	pink	Datsun mi-do	true
4	ф174щц68	black	UAZ Patriot	false
5	у140го78	red	Lexus LC	true
6	ы912шс78	blue	Mazda 6	true
7	х879мк116	white	Sollers Argo	false
8	ц891па92	black	Toyota Camry	true
10	ф812зн116	black	Mazda CX-5	true
11	а411ау85	pink	Audi A6	true
23	к120то178	red	lada	false

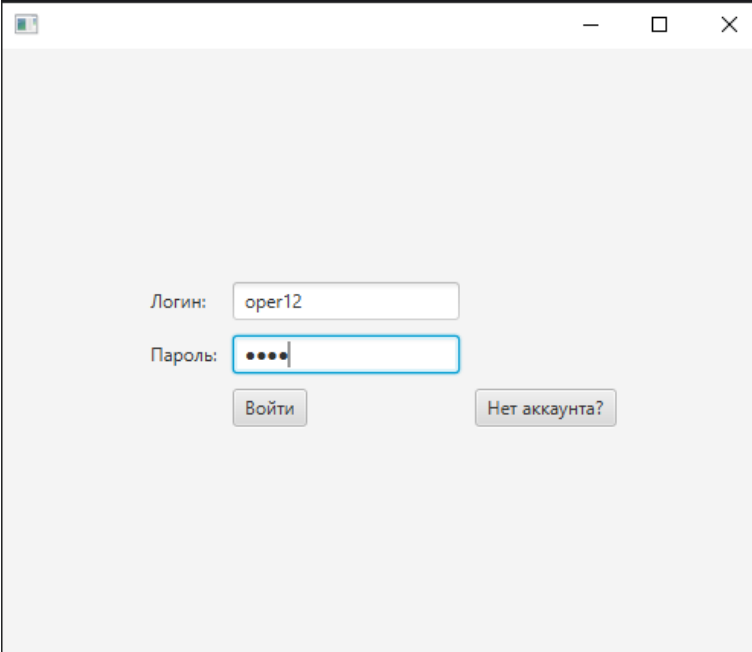
OK

Рисунок 10. Таблица автомобилей

Других возможностей, кроме просмотра данных, у мастера нет. При выходе из аккаунта, снова открывается окно авторизации.

Вход в качестве оператора

Войдем в систему как оператор.



The image shows a standard Windows-style window with a title bar containing a minimize button, a maximize button, and a close button. The window's content area has a light gray background. In the center, there is a login form. It consists of two labels: 'Логин:' and 'Пароль:'. The 'Логин:' label is followed by a text input field containing the text 'oper12'. The 'Пароль:' label is followed by a password input field containing four black dots. Below the password field, there is a blue rectangular border. At the bottom of the form, there are two buttons: 'Войти' (Login) on the left and 'Нет аккаунта?' (No account?) on the right.

Рисунок 11. Вход в качестве оператора

После успешной авторизации открывается следующее окно:

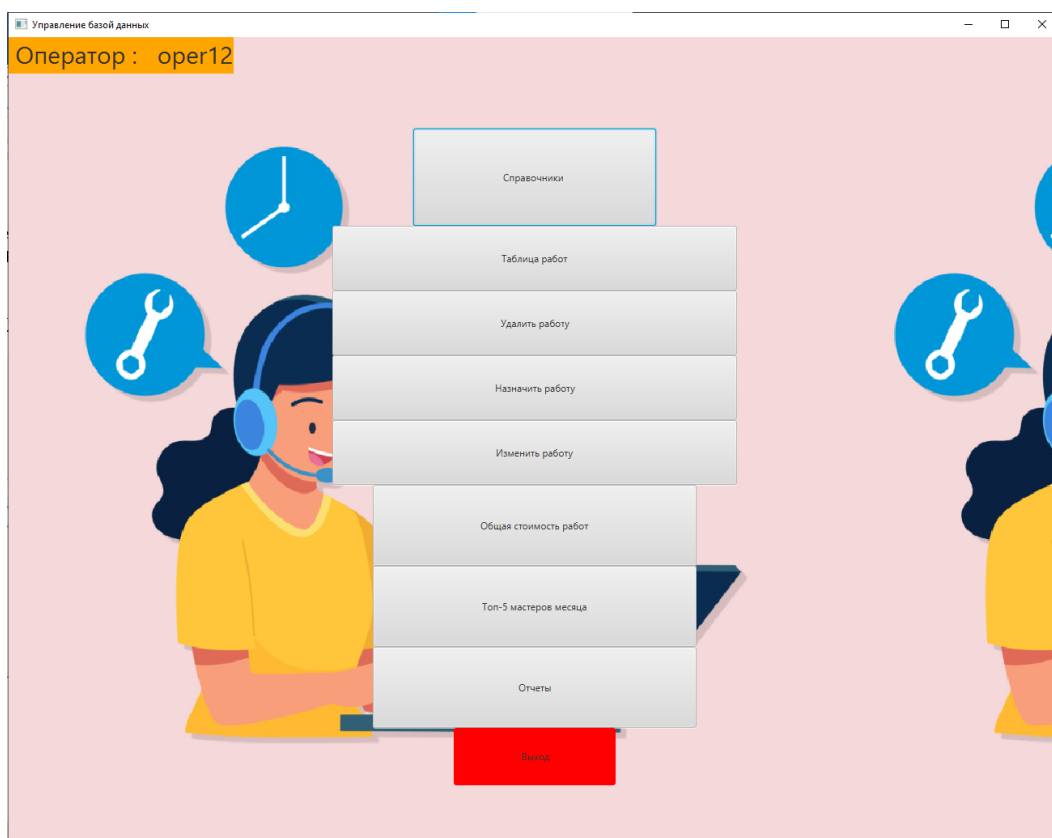


Рисунок 12. Окно оператора

Возможности оператора:

В разделе справочники оператор может работать с таблицами мастеров, услуг и автомобилей

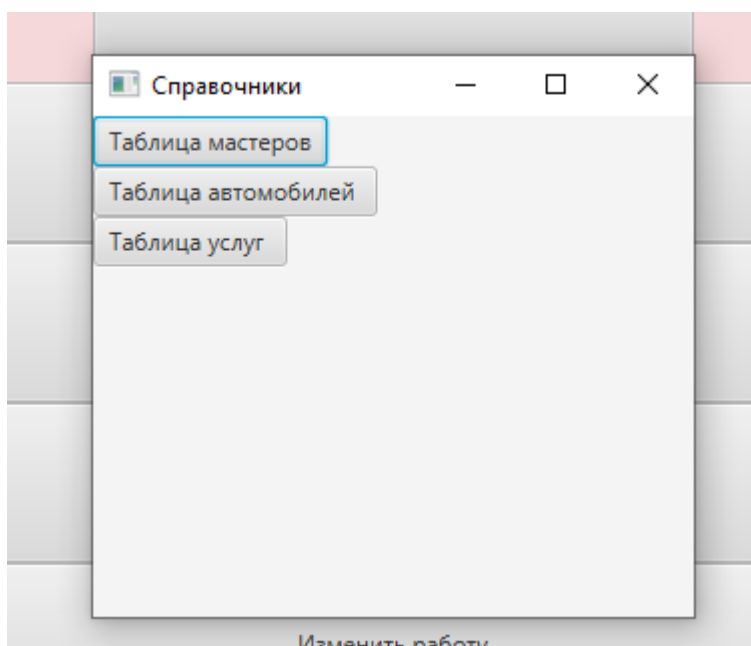


Рисунок 13. Справочники

Он может добавлять новые данные в таблицу, удалять их и модифицировать.

Рассмотрим подробнее работу с таблицей мастеров:

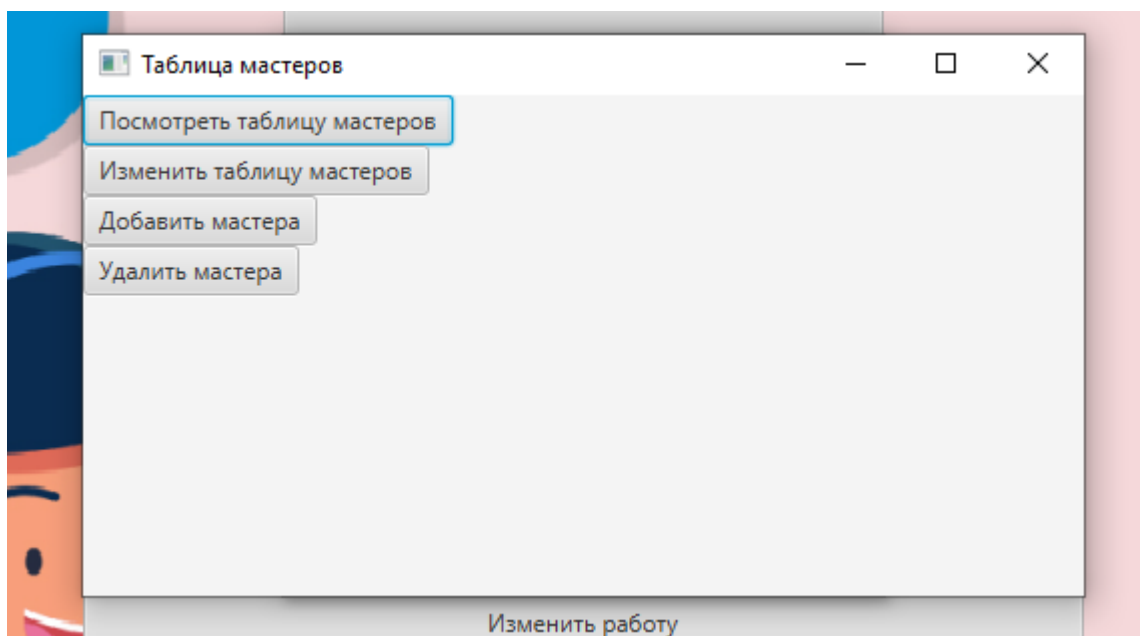


Рисунок 14. Работа с таблицей мастеров

Для начала посмотрим какие мастера есть в таблице:

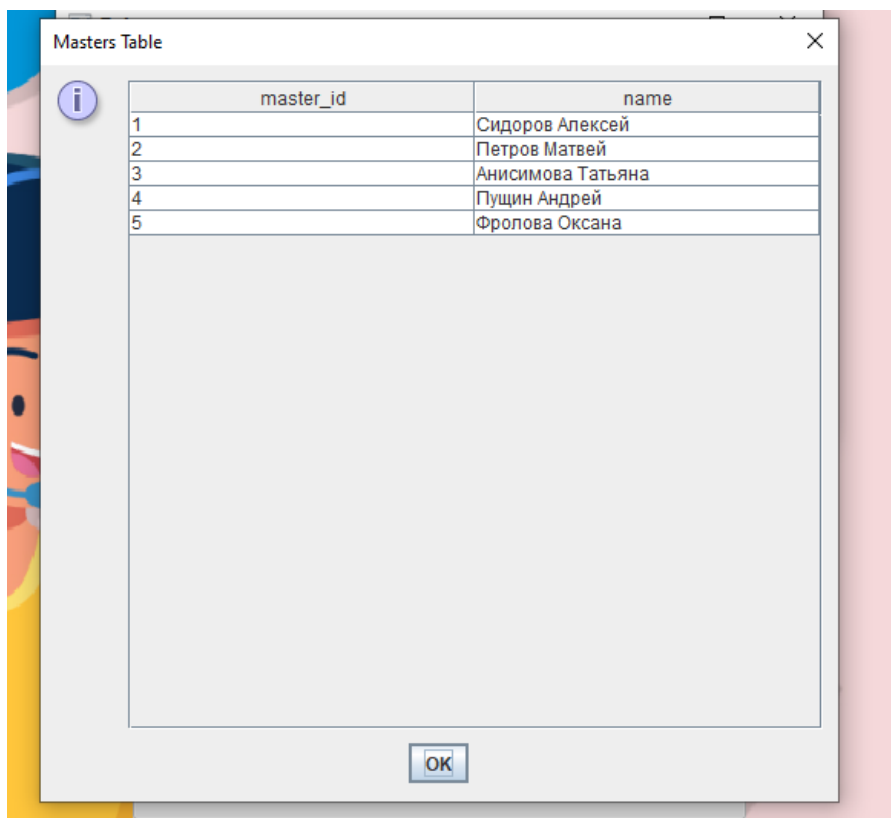


Рисунок 15. Таблица мастеров

Добавим нового мастера в таблицу:

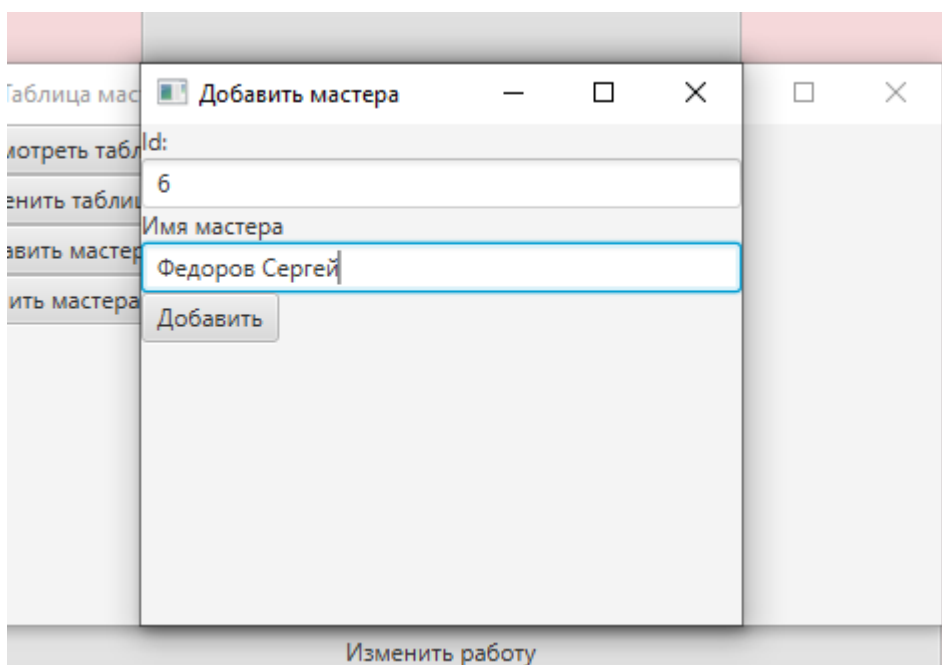
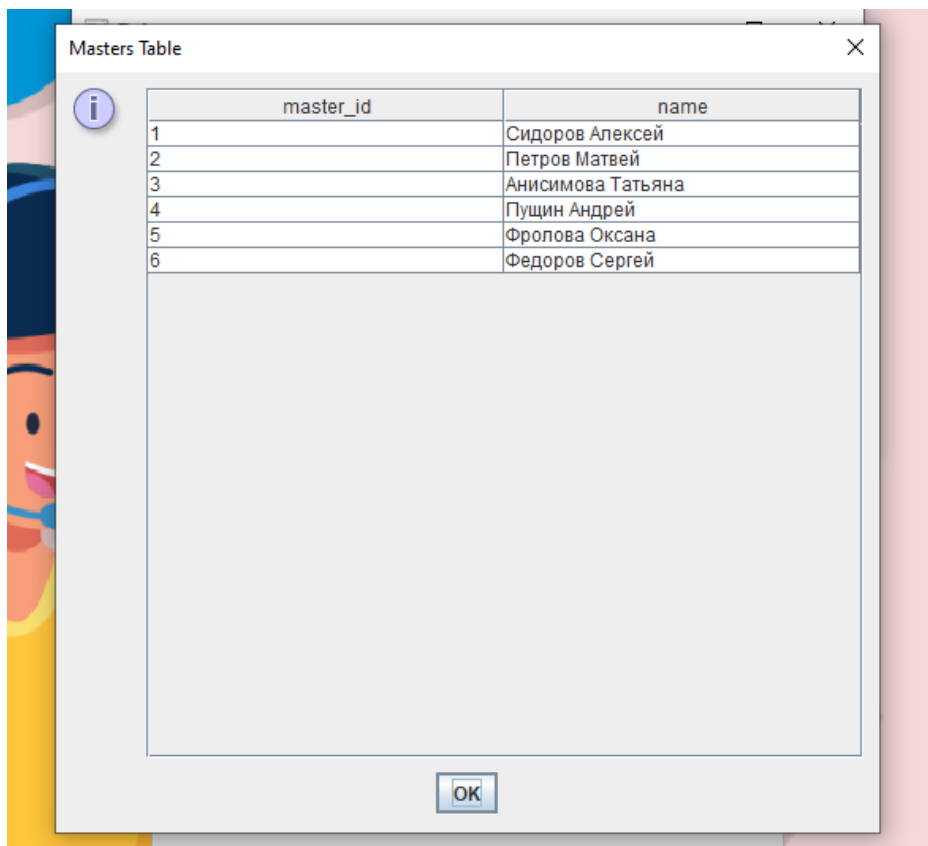


Рисунок 16. Добавление нового мастера в таблицу

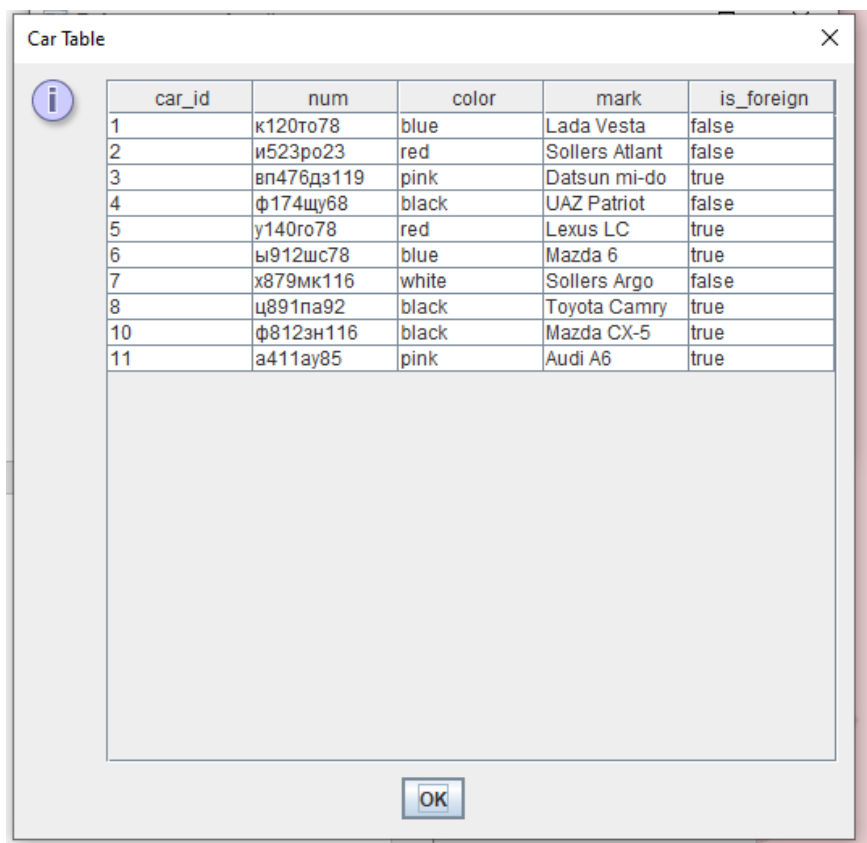
Рассмотрим снова таблицу мастеров, можем заметить что мастер был успешно добавлен:



master_id	name
1	Сидоров Алексей
2	Петров Матвей
3	Анисимова Татьяна
4	Пушин Андрей
5	Фролова Оксана
6	Федоров Сергей

Рисунок 17. Обновленная таблица мастеров

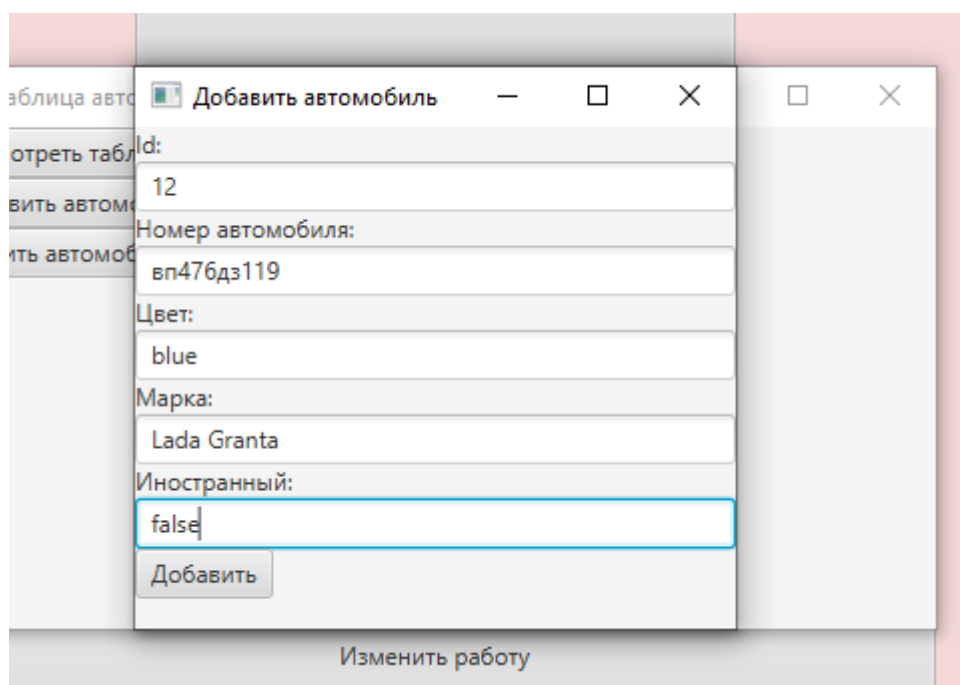
Посмотрим таблицу автомобилей:



car_id	num	color	mark	is_foreign
1	к120то78	blue	Lada Vesta	false
2	и523ро23	red	Sollers Atlant	false
3	вп476дз119	pink	Datsun mi-do	true
4	ф174щу68	black	UAZ Patriot	false
5	у140ро78	red	Lexus LC	true
6	ы912шс78	blue	Mazda 6	true
7	х879мк116	white	Sollers Argo	false
8	ц891па92	black	Toyota Camry	true
10	ф812зн116	black	Mazda CX-5	true
11	а411ау85	pink	Audi A6	true

Рисунок 18. Таблица автомобилей

Попробуем добавить автомобиль с таким же номером, как у третьего автомобиля:



Добавить автомобиль

Id: 12

Номер автомобиля: вп476дз119

Цвет: blue

Марка: Lada Granta

Иностраный: false

Добавить

Изменить работу

Рисунок 19. Добавление автомобиля с уже существующим номером

Как и предполагалось, возникает ошибка:

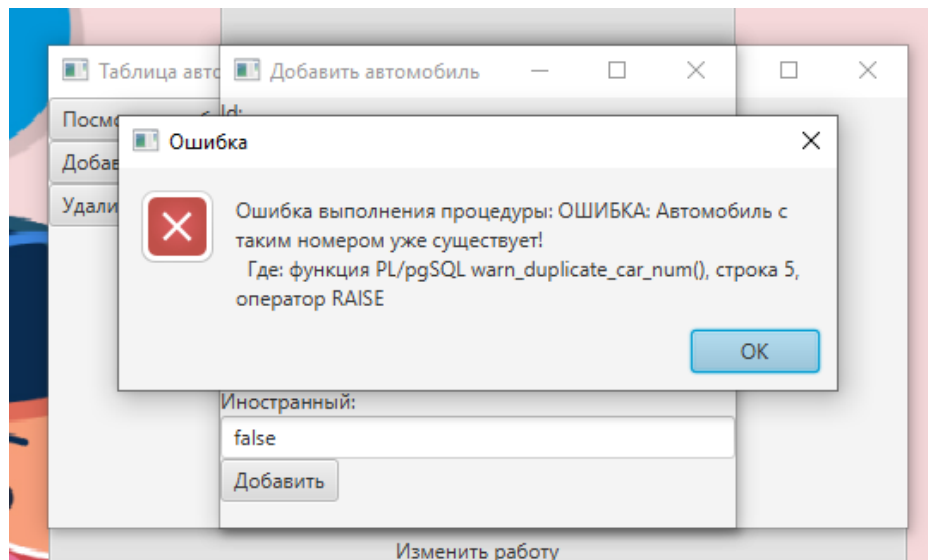


Рисунок 20. Вывод ошибки

Попробуем добавить автомобиль с уже существующим id:

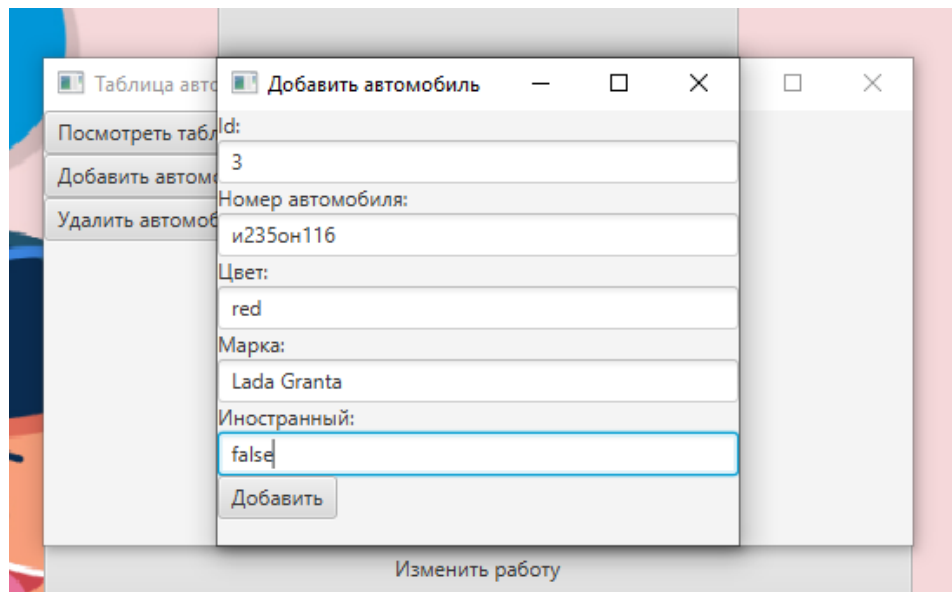


Рисунок 21. Добавление автомобиля с уже существующим id

Также возникает ошибка:

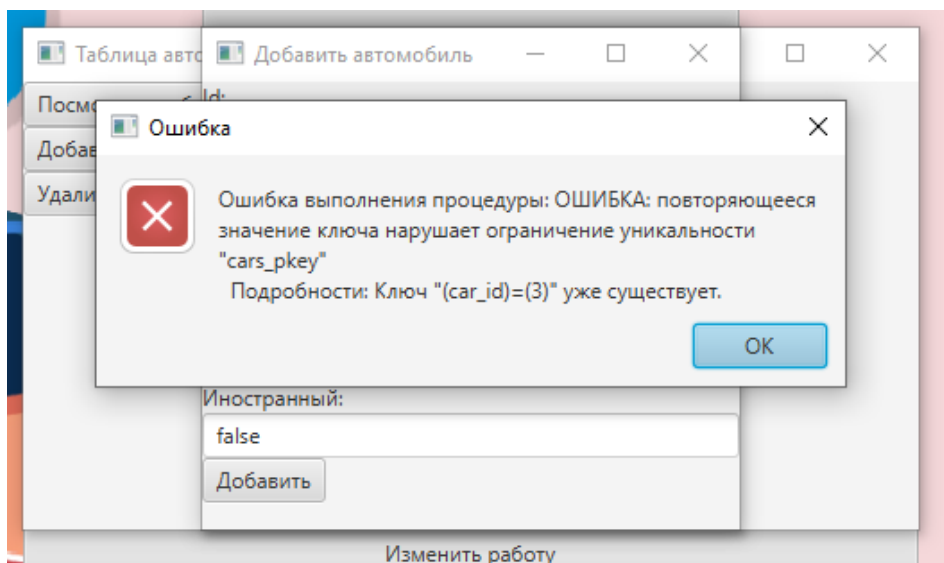


Рисунок 22. Вывод ошибки

Во вкладке услуг, кроме добавления и удаления записей, можно изменить стоимости услуг:

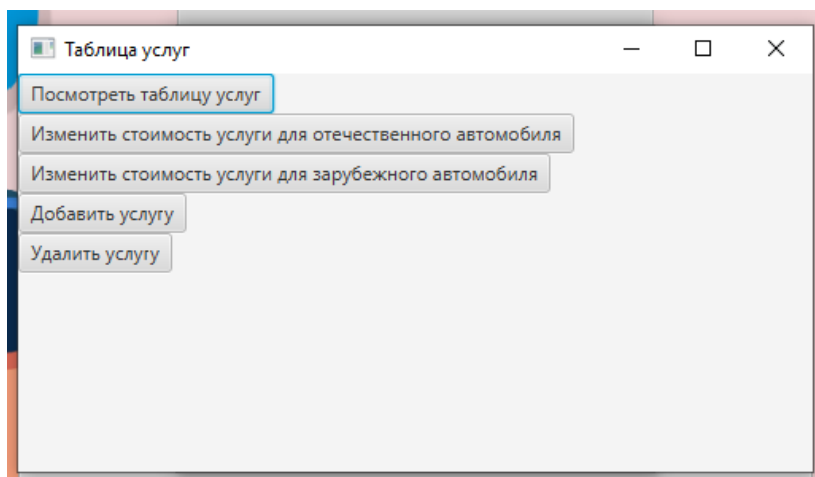


Рисунок 23. Работа с таблицей услуг

Оператор может работать с таблицей работ, просмотреть ее, удалить работу, назначить работу, или изменить запись в ней.

Рассмотрим подробнее функцию назначения работы. Открывается окошко, в котором оператор может переключаться между различными таблицами, вписывая в поле для ввода необходимые ему данные:

master_id	name
1	Сидоров Алексей
2	Петров Матвей
3	Анисимова Татьяна
4	Пушин Андрей
5	Фролова Оксана
6	Федоров Сергей

Id: Дата: Id мастера: Id автомобиля: Id услуги:

Рисунок 24. Окно добавления новой работы

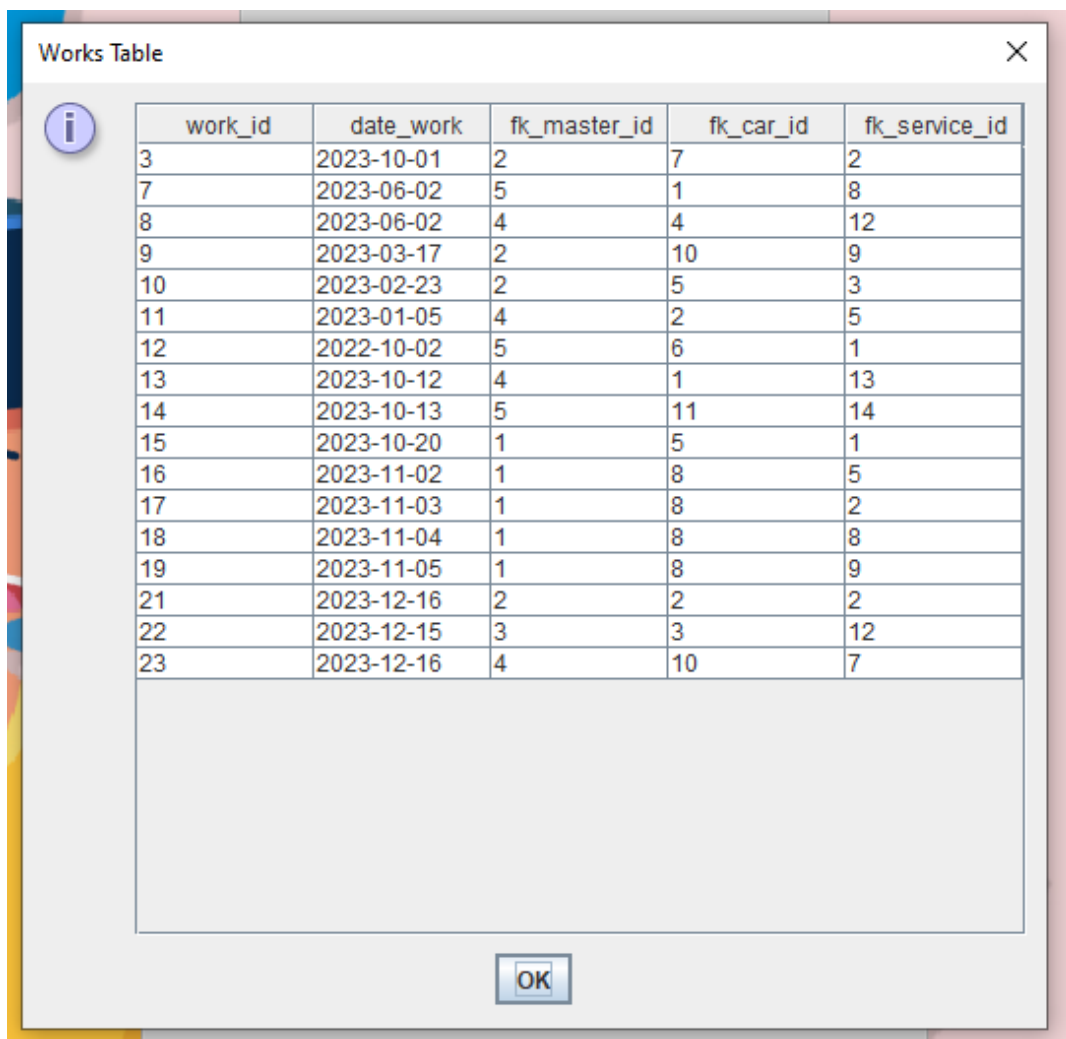
Назначим новую работу:

work_id	date_work	fk_master_id	fk_car_id	fk_service_id
3	2023-10-01	2	7	2
7	2023-06-02	5	1	8
8	2023-06-02	4	4	12
9	2023-03-17	2	10	9
10	2023-02-23	2	5	3
11	2023-01-05	4	2	5
12	2022-10-02	5	6	1
13	2023-10-12	4	1	13
14	2023-10-13	5	11	14
15	2023-10-20	1	5	1
16	2023-11-02	1	8	5
17	2023-11-03	1	8	2
18	2023-11-04	1	8	8
19	2023-11-05	1	8	9
21	2023-12-16	2	2	2
22	2023-12-15	3	3	12

Id: 23 Дата: 2023-12-16 Id мастера: 4 Id автомобиля: 10 Id услуги: 7

Рисунок 25. Назначение новой работы

В таблице работ можем убедиться, что работа была добавлена:



Works Table

work_id	date_work	fk_master_id	fk_car_id	fk_service_id
3	2023-10-01	2	7	2
7	2023-06-02	5	1	8
8	2023-06-02	4	4	12
9	2023-03-17	2	10	9
10	2023-02-23	2	5	3
11	2023-01-05	4	2	5
12	2022-10-02	5	6	1
13	2023-10-12	4	1	13
14	2023-10-13	5	11	14
15	2023-10-20	1	5	1
16	2023-11-02	1	8	5
17	2023-11-03	1	8	2
18	2023-11-04	1	8	8
19	2023-11-05	1	8	9
21	2023-12-16	2	2	2
22	2023-12-15	3	3	12
23	2023-12-16	4	10	7

OK

Рисунок 26. Таблица работ

Оператор может вывести общую стоимость обслуживания отечественных и импортных автомобилей за какой-то период:

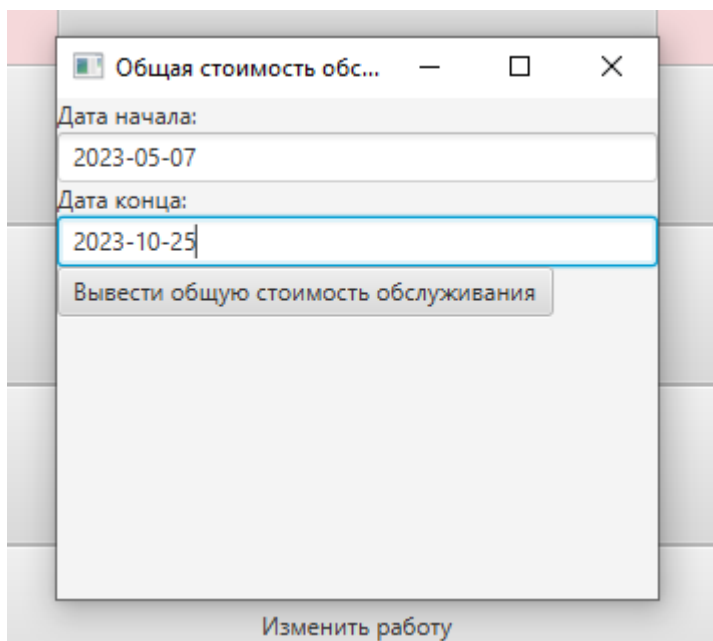


Рисунок 27. Подсчет стоимости обслуживания за определенный период

Получаем следующий результат:

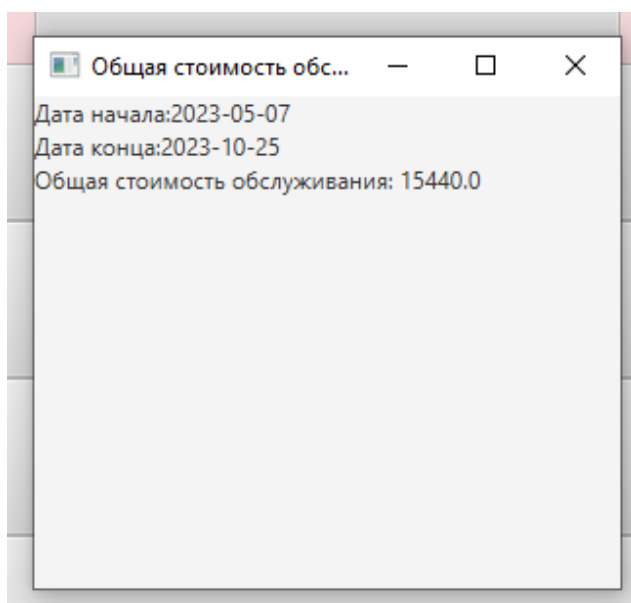


Рисунок 28. Результат подсчета стоимости обслуживания

Оператор может вывести пять мастеров, которые в заданном месяце выполнили наибольшее число работ для разных автомобилей.

Вводим номер месяца и год :

Рисунок 29. Получение списка мастеров, которые в октябре выполнили наибольшее число работ для разных автомобилей

Получаем список мастеров:

master_id	name	num_cars_served
1	Сидоров Алексей	1
2	Петров Матвей	1
4	Пушин Андрей	1
5	Фролова Оксана	1

Рисунок 30. Список мастеров

Оператор может вывести в отдельный файл отчеты.

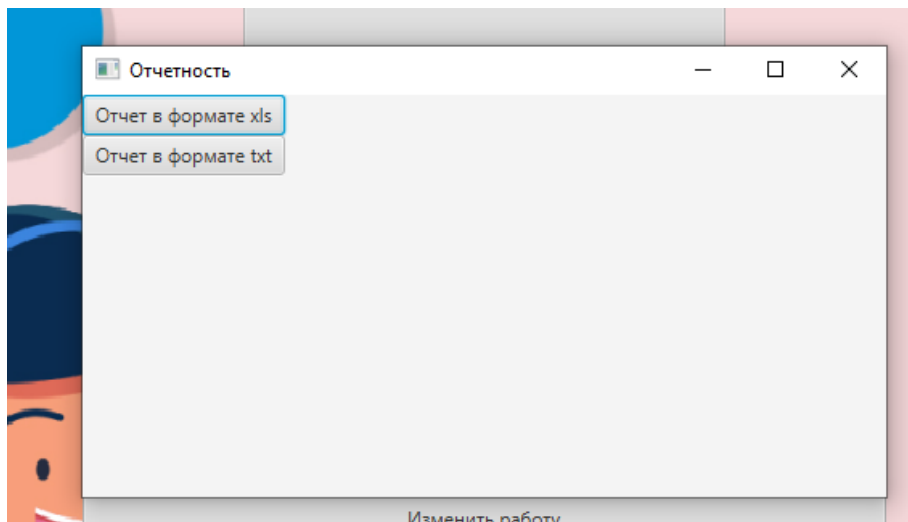


Рисунок 31. Выбор формата отчета

Выведем данные в txt файл:

Отчет – Блокнот

Файл Правка Формат Вид Справка

Таблица masters:

1	Сидоров Алексей			
2	Петров Матвей			
4	Пущин Андрей			
3	Анисимова Татьяна			
5	Фролова Оксана			
6	Федоров Сергей			

Таблица cars:

1	к120то78	blue	Lada Vesta	f
2	и523ро23	red	Sollers Atlant	f
3	вп476дз119	pink	Datsun mi-do	t
4	ф174щу68	black	UAZ Patriot	f
5	у140го78	red	Lexus LC	t
6	ь912шс78	blue	Mazda 6 t	
7	х879мк116	white	Sollers Argo	f
8	ц891па92	black	Toyota Camry	t
10	ф812эн116	black	Mazda CX-5	t
11	а411ау85	pink	Audi A6 t	

Таблица services:

14	замена рулевой трапеции	2885	3230	
3	капитальный ремонт	70000	104000	
12	покраска заднего бампера		6000	6325
1	замена тормозных колодок		1150	1380
2	замена тормозных дисков	2875	3335	
4	замена масла	805	1035	
5	диагностика ДВС	1725		
6	контрольный осмотр	575	690	
7	замена сцепления	5175	6325	
8	замена рулевой тяги	1495	1725	
9	регулировка фар	575		
10	диагностика электронных блоков	575	575	
13	замена картера с/у	460	460	

Таблица works:

3	2023-10-01	2	7	2
8	2023-06-02	4	4	12
9	2023-03-17	2	10	9
10	2023-02-23	2	5	3
11	2023-01-05	4	2	5
12	2022-10-02	5	6	1
13	2023-10-12	4	1	13
14	2023-10-13	5	11	14
7	2023-06-02	5	1	8
15	2023-10-20	1	5	1
16	2023-11-02	1	8	5
17	2023-11-03	1	8	2
18	2023-11-04	1	8	8
19	2023-11-05	1	8	9
21	2023-12-16	2	2	2
22	2023-12-15	3	3	12
23	2023-12-16	4	10	7

Рисунок 32. Отчет в txt-формате

Попробуем вывести данные в файл формата xlsh:

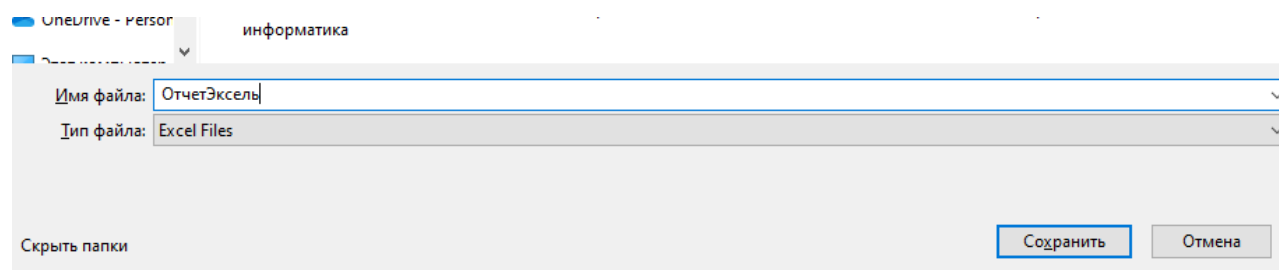


Рисунок 33. Сохранение отчета в формате xlsh

Откроем файл:

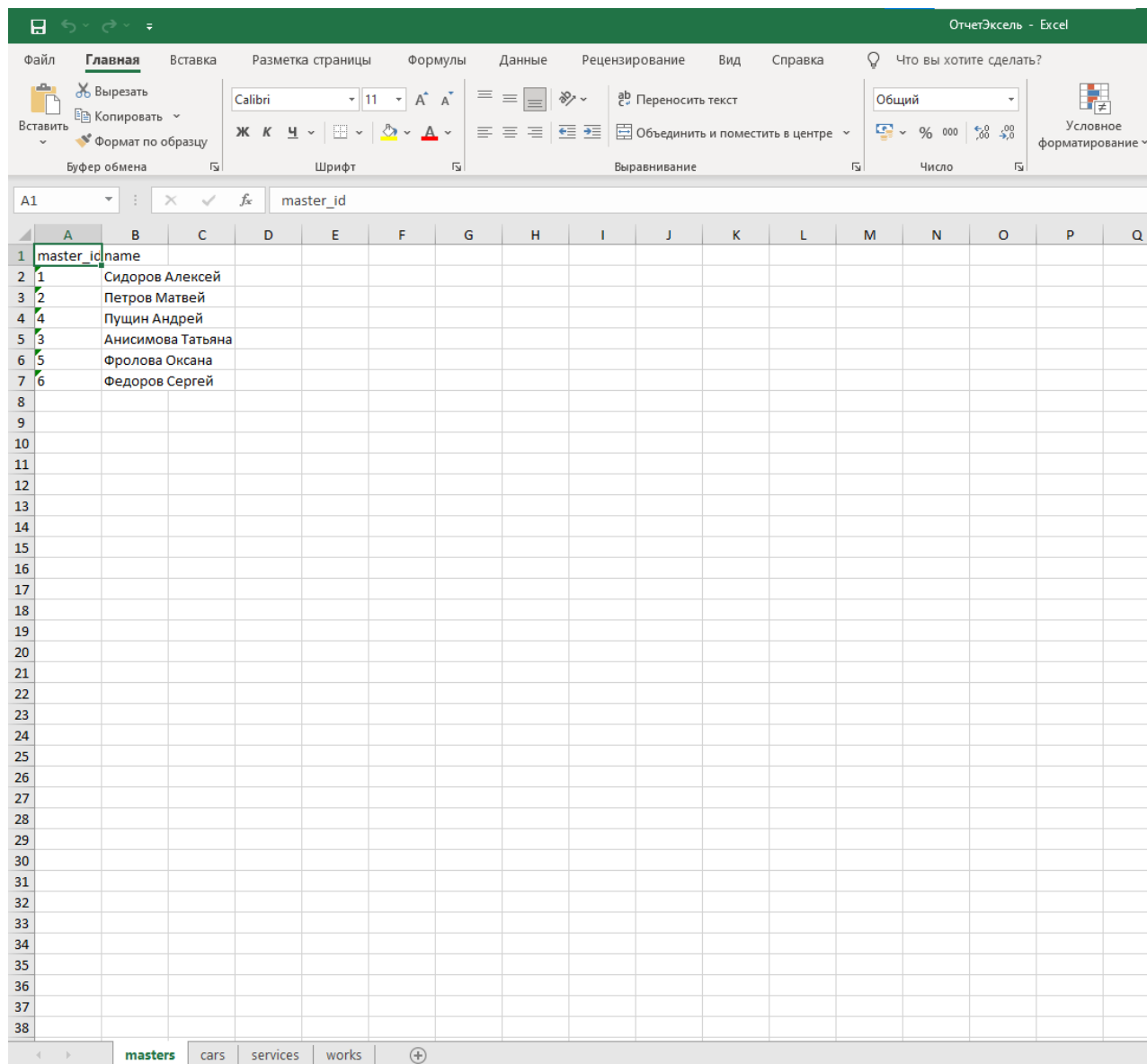


Рисунок 34. Отчет в формате *xlsx*

Вход в качестве бухгалтера

При входе как бухгалтер открывается следующее окно:

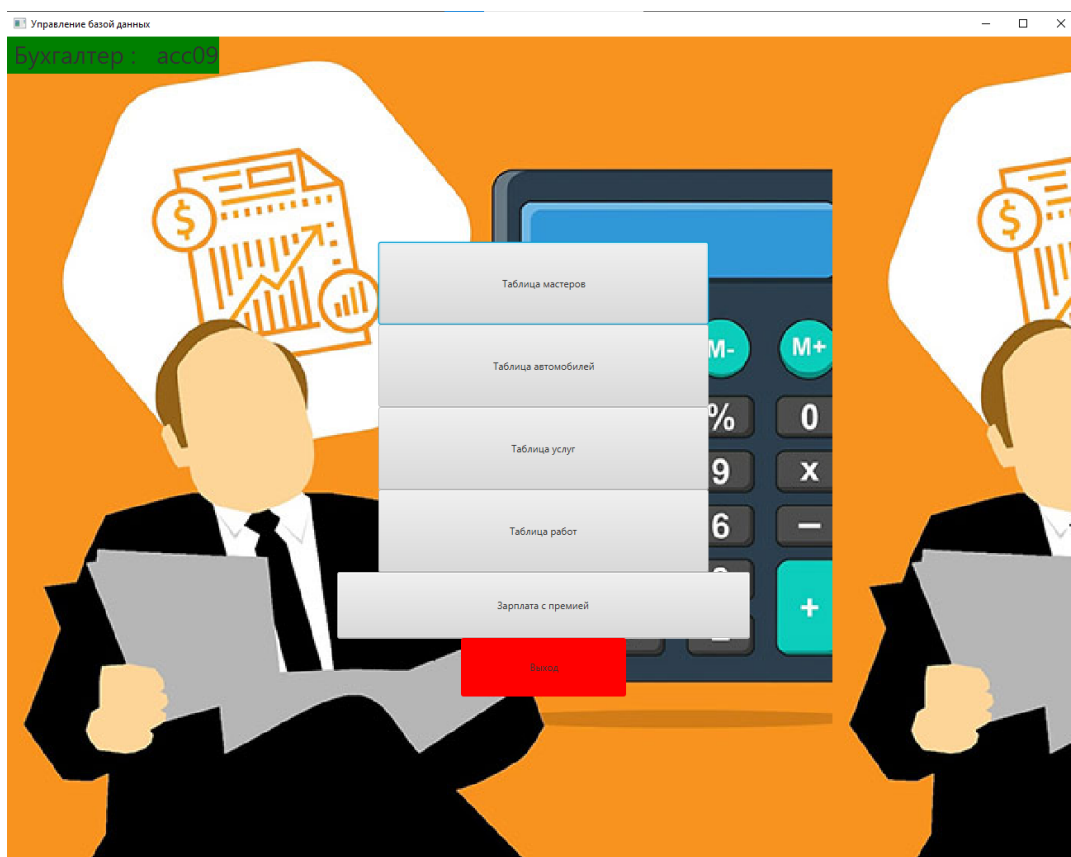
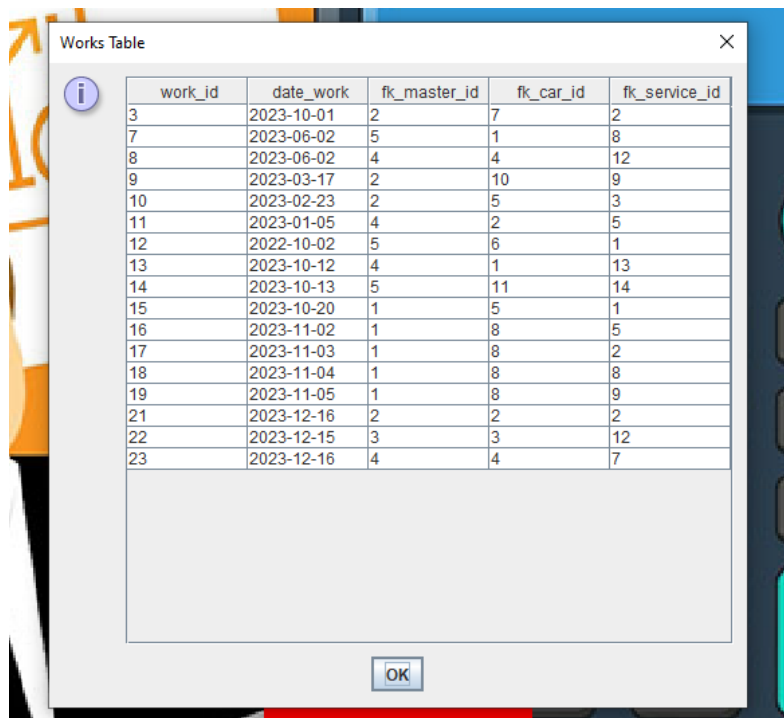


Рисунок 35. Вход в систему в качестве бухгалтера

Он так же, как и мастер может просмотреть данные из различных таблиц. Кроме того, у него есть возможность рассчитать зарплату с учетом премии в текущем месяце.

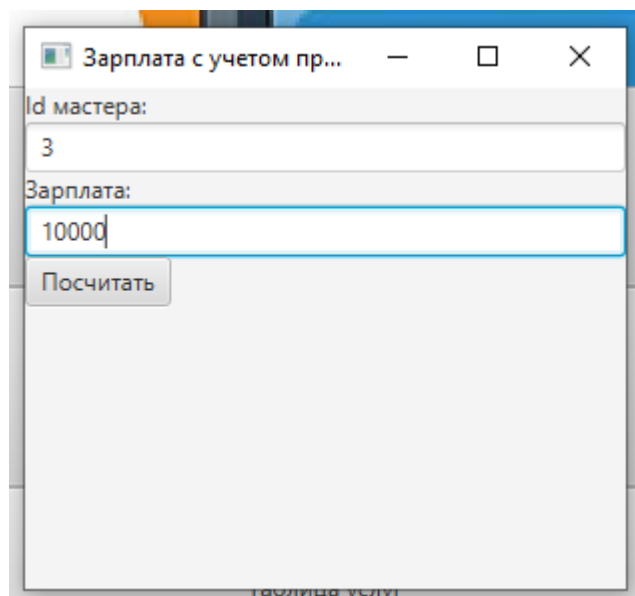
Рассмотрим сначала таблицу работ:



work_id	date_work	fk_master_id	fk_car_id	fk_service_id
3	2023-10-01	2	7	2
7	2023-06-02	5	1	8
8	2023-06-02	4	4	12
9	2023-03-17	2	10	9
10	2023-02-23	2	5	3
11	2023-01-05	4	2	5
12	2022-10-02	5	6	1
13	2023-10-12	4	1	13
14	2023-10-13	5	11	14
15	2023-10-20	1	5	1
16	2023-11-02	1	8	5
17	2023-11-03	1	8	2
18	2023-11-04	1	8	8
19	2023-11-05	1	8	9
21	2023-12-16	2	2	2
22	2023-12-15	3	3	12
23	2023-12-16	4	4	7

Рисунок 36. Таблица работ

Рассчитаем зарплату с учетом премии мастера с id=3 и с зарплатой 10000 р.:



Зарплата с учетом пр...

Id мастера:
3

Зарплата:
10000

Посчитать

Рисунок 37. Вычисление зарплаты с учетом премии

Так как по разным машинам в текущем месяце он выполнил лишь одну работу, его премия составляет 5% от зарплаты:

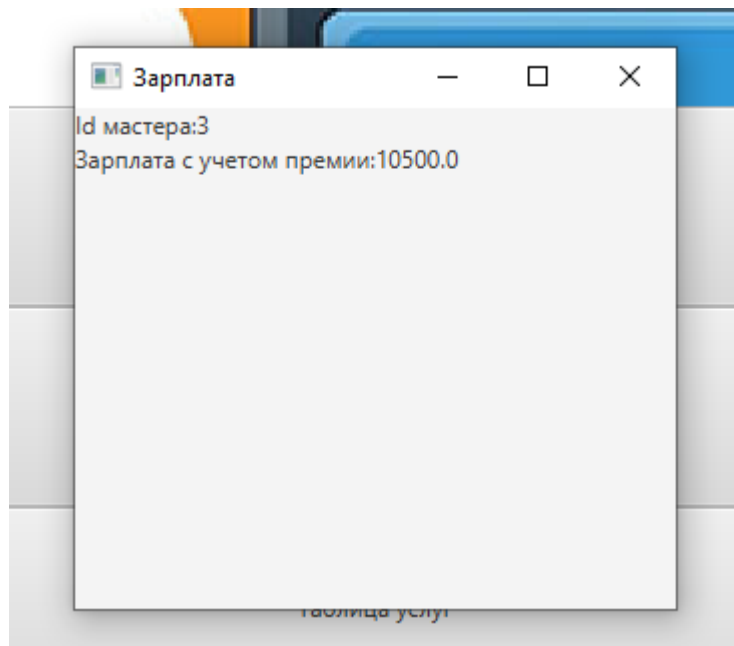


Рисунок 38. Вывод зарплаты с учетом премии

Вывод

В результате выполнения работы была создана функциональная система, позволяющая пользователям взаимодействовать с базой данных автосервиса через удобный графический интерфейс. Приложение обеспечивает возможность просмотра информации о клиентах, автомобилях, услугах и других сущностях, связанных с работой автосервиса. Были изучены современные инструменты для работы с БД и реализации графического интерфейса. Благодаря использованию современных инструментов разработки, таких как JavaFX для создания графического интерфейса и JDBC для работы с базой данных PostgreSQL, приложение обладает высокой производительностью и надежностью. Прделанная работа является для меня полезным опытом, так как позволила применить на практике навыки в области программирования и баз данных.

Список использованной литературы

1. Шилд, Герберт. "Java 8. Полное руководство". 9-е изд. Перевод с английского. М.: ООО "И.Д. Вильямс", 2015. 1376 с.
2. Прохоренок, Н. А. "П84 JavaFX". СПб.: БХВ-Петербург, 2020. 768 с.
3. Молинаро, Энтони. "SQL. Сборник рецептов". 2-е изд. Перевод с английского. М.: Издательство "ДМК Пресс", 2018. 560 с.
4. Оливер, Эд. "JavaFX: разработка RIA-приложений". 2-е изд. Перевод с английского. М.: Издательство "ДМК Пресс", 2017. 800 с.
5. Бек, Адам. "SQL для начинающих: практический подход". 2-е изд. Перевод с английского. М.: Издательство "ДМК Пресс", 2019. 528 с.