

Федеральное государственное образовательное бюджетное  
учреждение высшего образования  
**«Финансовый университет  
при Правительстве Российской Федерации»  
(Финансовый университет)**

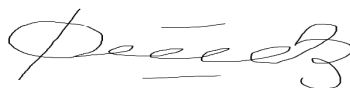
**Департамент анализа данных и машинного обучения**

**Пояснительная записка к курсовой работе по дисциплине  
«Современные технологии программирования»**

на тему:

**Информационно-справочная система «Ресторан»**

Выполнил:



Студент группы ПИ19-4  
Содиков Фарход Фирдавсович

Научный руководитель:

Доцент, к.т.н.



Дадян Эдуард Григорьевич

2021

## Содержание:

Введение .....	3
1. Постановка задачи .....	4
2. Описание предметной области .....	6
3. Актуальность автоматизации .....	7
4. Описание программы .....	8
4.1 Алгоритмические решения .....	8
4.2 Описание интерфейса программы .....	11
4.3 Состав приложения.....	24
5. Назначение и состав классов программы.....	27
6. Заключение .....	30
Список литературы: .....	31

## Введение

Основная идея курсового проекта состоит в создании готового приложения для пользователей с использованием языка программирования Java в среде разработки IntelliJ IDEA. В течение работы можно сильно улучшить навыки в области создания целостного пользовательского приложения, а точнее работу с языком программирования. Кроме Java можно окунуться в анализ предметной области и улучшить навыки визуального проектирования поставленной задачи. В работе также используется фреймворк Spring, который демонстрирует собой контейнер внедрения зависимостей, с несколькими слоями, что позволяет очень быстро и удобно делать Java-приложения.

В разработку входит создание *клиента*, который хранит и обрабатывает информацию и *сервера*, выполняющего учёт и отображение объектов предметной области, хранящихся в *базе данных* и также предоставляющего статистическую информацию.

В наше время цифровые технологии значительно развиваются и всё больше сфер нуждаются в поддержке и обработки данных. Чаще всего крупные и даже малые предприятия или компании нуждаются в автоматизированных информационных системах. Такие системы актуальны и используются не только в предпринимательской или IT деятельности, но и в других отраслях.

К окончанию работы должно быть готовое полноценное приложение для ресторанного бизнеса с использованием вышеперечисленных инструментов программирования. Чтобы достичь поставленной цели нужно выполнить ряд задач:

1. Постановка задачи
2. Анализ в области информационно-справочных систем ресторанного дела
3. Разработка база данных
4. Подбор нужных инструментов программирования (фреймворки, плагины, библиотеки и т. д.)
5. Реализация на программном уровне

## 1. Постановка задачи

В соответствии с выбранной темой курсовой работы необходимо разработать приложение в области ресторанного дела. Информационно-справочная система «Ресторан» предназначена для оптимизации работы в ресторанном бизнесе и взаимодействия с работниками.

Информационно-справочная система «Ресторан» обеспечивает хранение, сбор и обработку информации о заказе. Целями создания информационно-справочной системы «Ресторан» являются:

- Простота использования
- Хранение файлов в базе данных
- Ознакомление с современными технологиями, которые помогают в реальной жизни
- Хранение, сбор и обработка данных
- Поточковая передача данных

В созданном приложении реализована загрузка с помощью JSON-файла, в котором находится вся необходимая информация. Про JSON написано в следующих разделах записки.

Конкретизация требований:

1. В нашей работе должна присутствовать разработка информационной модели предметной области, предоставленная в формате пользовательских классов (двух или более). Должно быть также несколько форм пользовательского интерфейса.
2. Данная работа должна представлять собой web-приложение
3. Разработчик сам определяет интерфейс программы и её функционал, но приложение должно удовлетворять общим требованиям:
  - 3.1. Отображать в виде сеток данные предметной области.
  - 3.2. Сделать добавление в источник данных нового объекта, а также удаление объекта из источника данных и редактирование объекта источника данных.

- 3.3. Сделать фильтрацию записей источника данных, довольствующих введенному пользователем сложному параметру.
- 3.4. Создать сортировку записей источника данных, включая многоуровневую.
- 3.5. Сохранять источник данных для информационной модели, основанной на таблице БД, в базе данных.
- 3.6. При запуске программы загрузить сохранённые данные из нашей базы данных.
- 3.7. Используя панель инструментов или меню, вызвать приложение Блокнот для просмотра справки о программе: текстовый файл текущей папки программы.
- 3.8. Сделать в меню пункт «Об авторе» с выводом соответствующей информации.
- 3.9. Разработать некоторое количество полезных функций для отображения статистических данных, например, средних или минимальных или максимальных значений и т. д.
- 4. Программа не должна выключаться аварийно: сообщения о некорректном вводе данных, недопустимых или противоречивых значений данных, при отсутствии данных по функциональному запросу пользователя и других нештатных ситуациях отображать в окнах сообщений.
- 5. Программа также должна быть проста в прочтении и хранить полезные комментарии.
- 6. Интерфейс программы должен быть эргономичным и интуитивно понятным.

## **2. Описание предметной области**

В наше время очень много торговых центров, где находится большое количество мест питания: кафе и рестораны. В ТРЦ или в ТЦ зачастую приходят очень много людей за большими покупками или за развлечениями и конечно же они устают и идут в фуд корт. Поэтому в заведениях в большинстве случаев залы максимально заполняются. Работникам таких заведений сложно работать в таких условиях. Я часто видел у своих знакомых, которые занимаются ресторанным бизнесом проблемы связанные со случаями большой загруженности зала, официанты не могли запоминать большое количество информации в своей голове из-за этого физически не могли обслуживать вовремя все столики, а менеджеры получали за это жалобы в свою сторону и конечно же в сторону ресторана, которые портили репутацию заведения. Поэтому эта тема очень сильно заинтересовала меня.

Для решения этой проблемы я решил сесть и подумать насчет оптимизации и автоматизации данного дела. И конечно же в этом нам помогут современные технологии программирования. Подойдя с полной серьёзностью к этой проблеме, пришлось проанализировать несколько заведений.

### **3. Актуальность автоматизации**

Наверное, если вы были владельцем какого-нибудь ресторана, то хотели бы, чтобы в вашем заведении было всё удобно, автоматизировано и оптимизировано. Ведь традиционный метод работы в ресторанном деле очень рискованный, так как в наше время, время цифровизации, стоит использовать современные технологии. И зачастую работникам большого ресторана очень трудно хранить информацию (о столиках, о заказе) в голове и иногда молодые официанты и кассиры без опыта теряются и в итоге в ресторане творится хаос из-за чего менеджерам таких заведений достаётся по полной от своих начальников, боссов. Для избегания этой проблемы стоит разработать приложение для кассиров и их менеджеров. Это поможет менеджерам легко следить за официантами, а в свою очередь официантам и кассирам будет легко следить за заказами и столиками. К примеру, раньше им нужно было быстро записывать на листочек каждое блюдо и кол-во, а через приложение можно сразу увидеть все блюда и выбрать те блюда, которые тебе нужны, и обозначать количество каждого из блюд.

## 4. Описание программы

### 4.1 Алгоритмические решения

Данные о работниках (кассиры, повара, официанты и менеджеры и т.д.), о блюдах и о заказах хранятся в базе данных. В БД хранятся имена, фамилии, номера телефонов, заработная плата и должность работника. Framework Spring Boot используется для написания серверной части приложения, а интерфейс написан на JavaFX.

Во вкладке «Менеджерам» показываются объекты классов `employees` и `orders_history`, а во вкладке «Кассирам и поварам» отображаются объекты классов `orders`. Приложение сохраняет все изменения данных в базу данных. В большинстве классов использует специальный метод `TableView` для отображения списков (заказы, работники, блюда).

В приложении также присутствует кнопка удаления объекта из списка. К примеру, убираем работника из списка работников. В списке показывается имя, фамилия и должность человека. При нажатии любого объекта из списка, в правой части экрана появляется вся информация о работнике, которую мы упоминали чуть выше. Также в этом разделе, помимо информации и кнопки, присутствует кнопка «Редактировать», которая позволяет изменить информацию о служащем.

Конечно же, имеется кнопка удаления, которая при наведении на объект списка и нажатии кнопки «Удалить» удаляет работника из списка и, соответственно, из базы данных.

В нашей структуре базы данных, включая `Employee`, имеется 6 сущностей:

- Dish (Блюдо)
- Employee (Работник)
- Ingredient (Ингредиент)
- Order (Заказ)
- Position (Должность)
- Unit (Единицы измерения ингредиентов)



Сущность «Dish» имеет две связи ManyToMany с «Order» и с «Ingredient», так как, соответственно, в первом случае нам нужно указать какие блюда входят в заказ, а во втором случае для описания блюда - то из чего состоит. В «Dish» несколько атрибутов:

- ID (Идентификатор)
- Name (Название блюда)
- Price (Стоимость)
- Time (Время готовки)

Сущность «Unit» нужен для определения единицы измерения ингредиента, например 500 граммов картошки или 200 миллилитров воды, поэтому он имеет связь с «Ingredient» ManyToOne. Её атрибуты:

- ID (Идентификатор)
- Unit (Название единицы измерения)

Сущность «Ingredient» описывает ингредиенты блюд, поэтому она имеет связь с «Dish» через новую сущность «Dish\_ingredients», так как связь между «Ingredients» и «Dish» ManyToMany. У ингредиентов следующие атрибуты:

- ID (Идентификатор)
- Name (Название ингредиента)
- Unit\_price (Стоимость одной единицы ингредиента)
- Unit\_ID (Внешний ключ, идентификатор единицы измерения)

Сущность «Order» имеет две связи, как мы говорили ранее, связь ManyToMany с «Dish» через «Dish\_order» и вторую связь с «Employee» для связи с работниками, к которым чуть позже перейдём. Заказ имеет не так много атрибутов:

- ID (Идентификатор)
- Locator (Локатор, номер локатора)
- Employee\_ID (Внешний ключ, идентификатор работника)

Сущность «Employee» описывает работников и имеет две связи, одна из которых упомянута чуть выше, и вторая связь с сущностью «Positions».

Атрибуты «Employee»:

- ID (Идентификатор)
- First\_name (Имя)
- Last\_name (Фамилия)
- Phone\_number (Телефонный номер)
- Positions\_ID (Внешний ключ, идентификатор должности)

Сущность «Positions» нужна для описания должности работника и их заработной платы, поэтому имеет связь с «Employee». И атрибуты должности:

- ID (Идентификатор)
- Position (Название должности)
- Salary (Заработная плата)

У всех идентификаторов тип данных – целое число, у названий или имён тип данных – строка, у даты или время тип данных – дата и время (datetime).

## 4.2 Описание интерфейса программы

При входе в приложение, на экране высвечивается окно «О программе» (см. рис №1), в которой говорится всё о программе. Окно, открывающееся при открытии, то есть, главную страницу можно изменить.

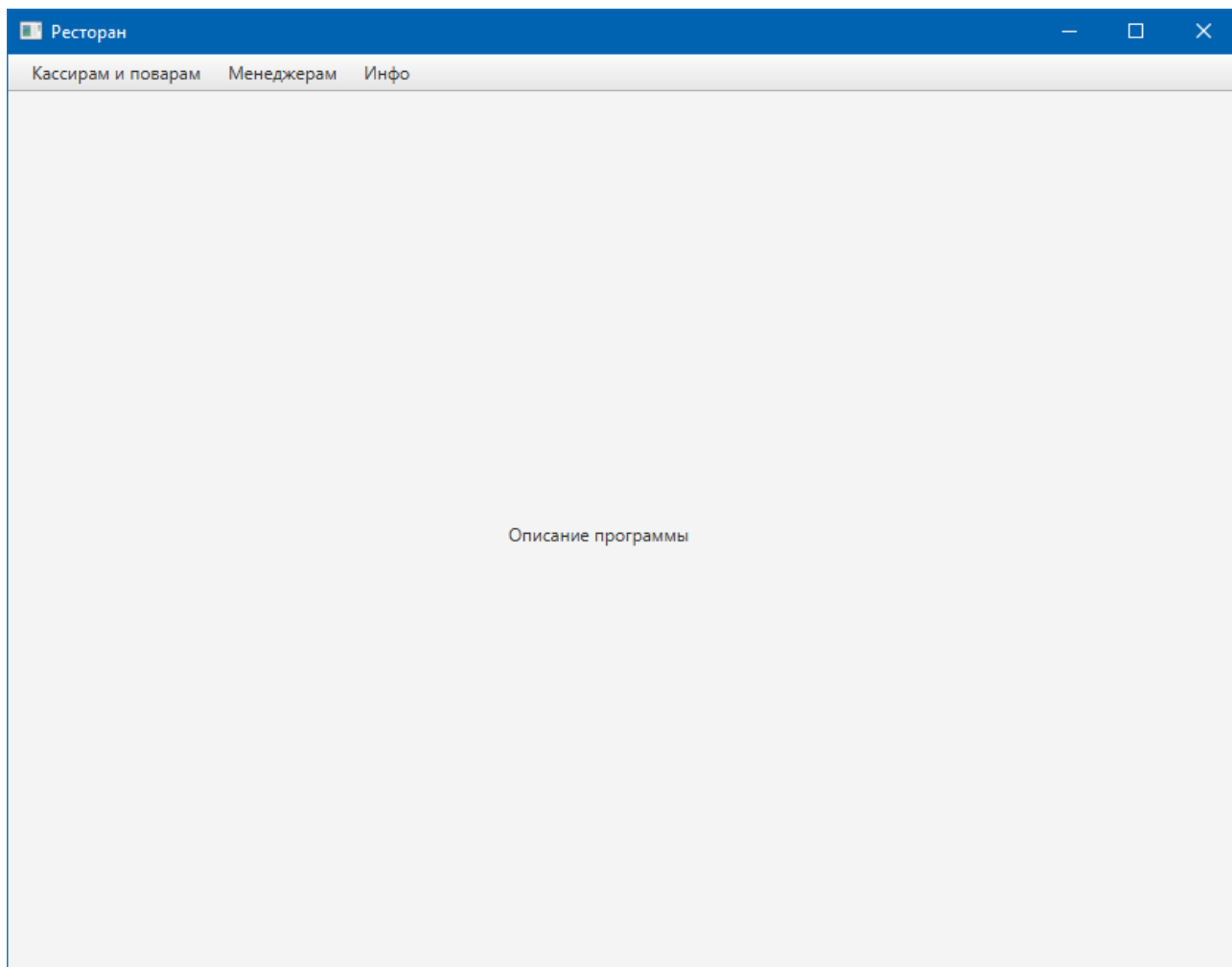


Рис. 1

Это окно пока что пустое, но его легко можно заполнить, добавив Label в Scene Builder. На самом деле, там уже есть Label, в котором написано: «Описание программы».

Также во вкладке «Инфо» есть раздел «Об авторе» (см. рис №2), в котором будет записана вся информация о создателе приложения в Label, который стоит по центру.

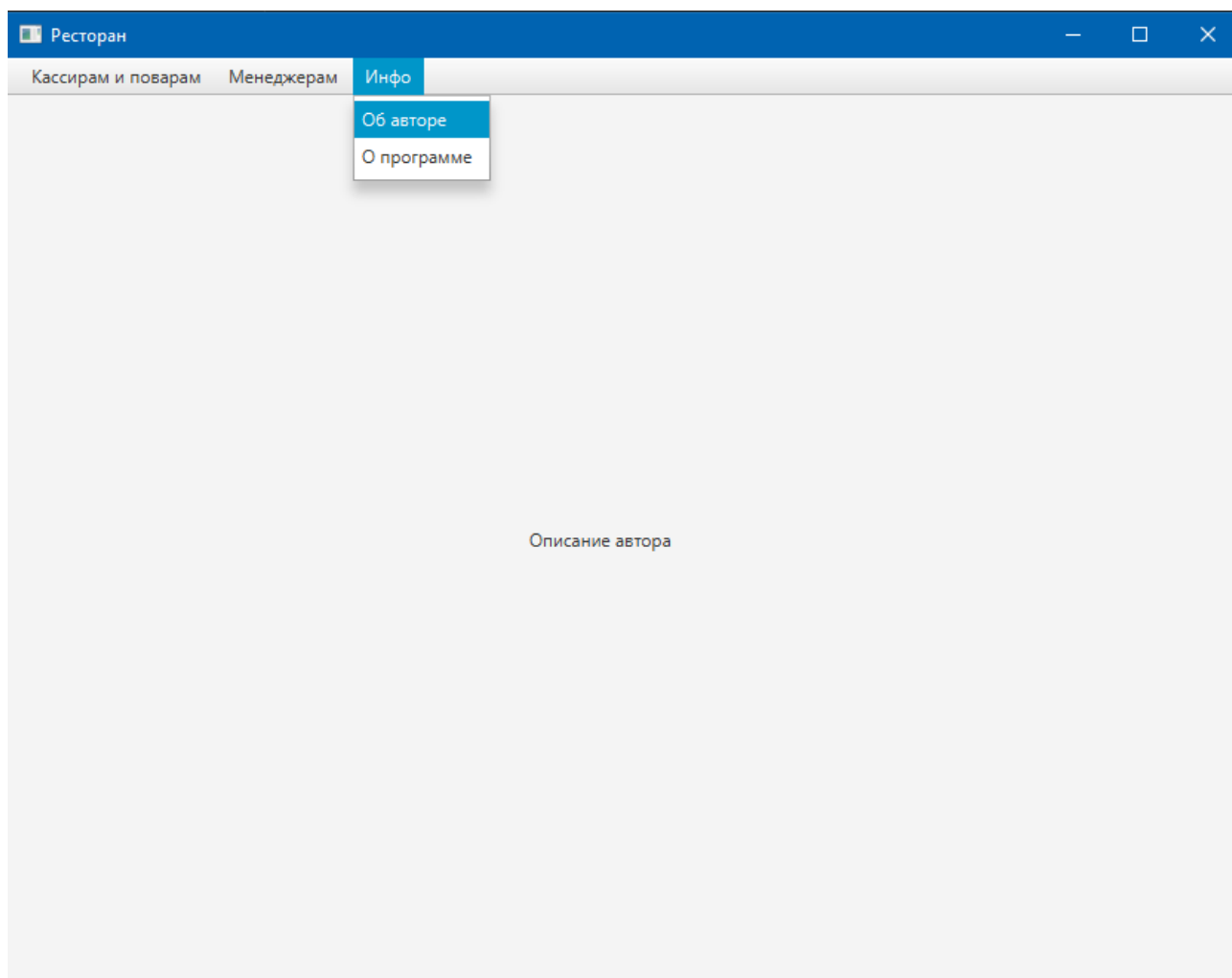


Рис. 2

Также при входе можно заметить в верхнем левом углу название и логотип ресторана. Это можно сделать, перейдя в класс «MainApp» в методе «start».

Но это всего лишь окна с текстом, поэтому далее будет весь функционал нашего приложения, разбор классов и различных объектов Scene Builder'а. Для удобного отображения всех блюд мы будем использовать таблицу с ценами и названиями. Также таблицу будем использовать для отображения всех сотрудников, которая будет удобна для менеджеров, которым будет легче следить за посещаемостью работников. Ко всему вышеперечисленному, будут различные кнопки, о которых мы говорили ранее.

Во вкладке «Менеджерам» (см. рис. 3) есть переход к окнам «Список сотрудников» и «История заказов».

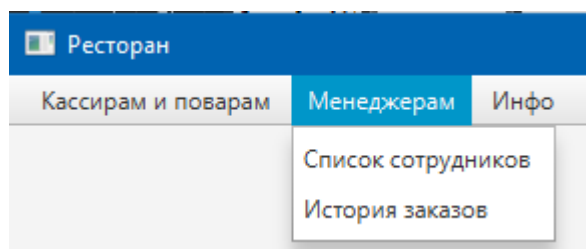


Рис. 3

В списке сотрудников отображаются наши работники (кассиры, официанты, повара и менеджеры).

Для создания этого окна в Scene Builder ставим пустой AnchorPane, который помогает позиционировать элементы вдоль одной из сторон контейнера. На AnchorPane накладываем SplitPane для разделения экрана на два вертикальных окна.

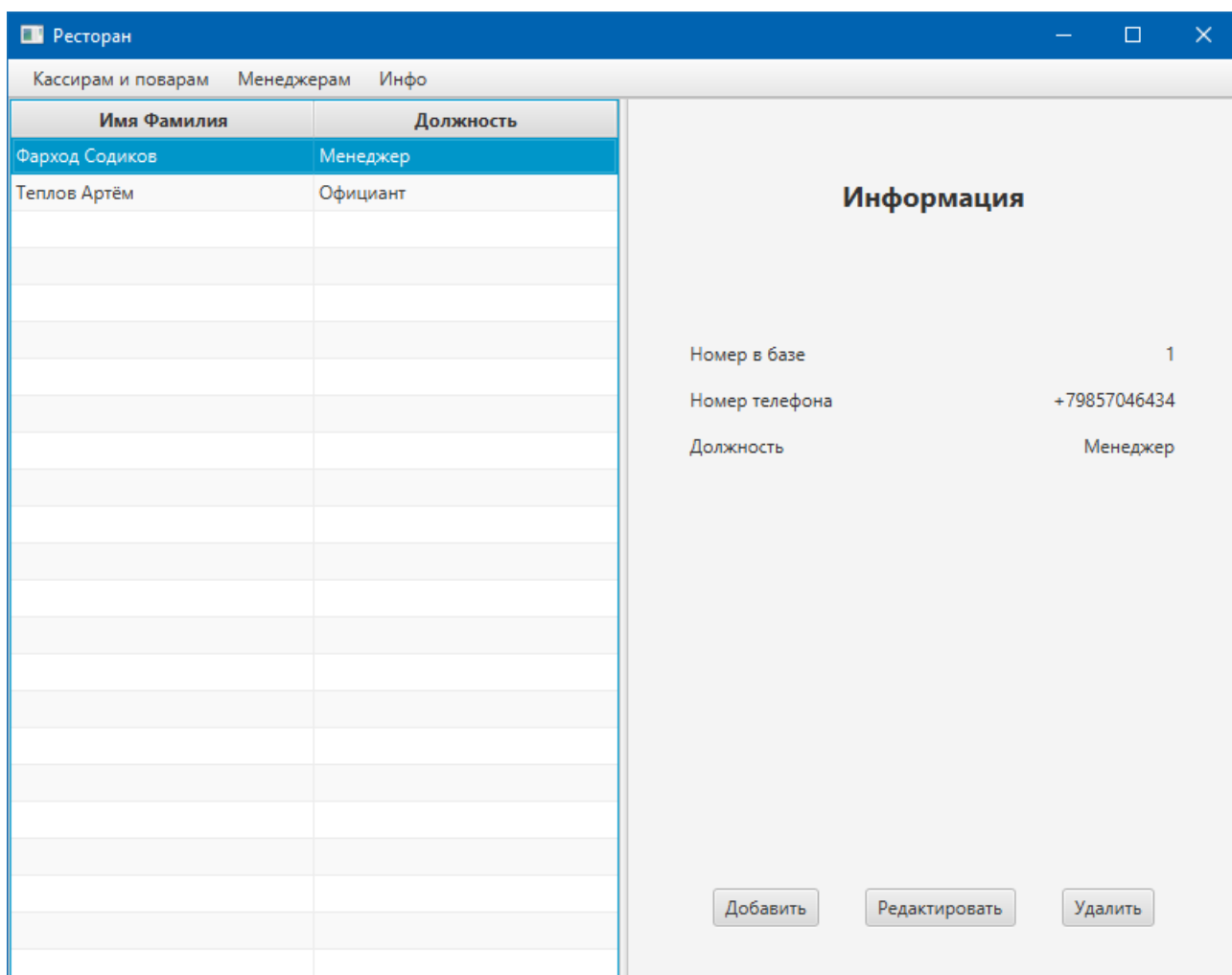


Рис. 4

В левой части создан `AnchorPane`, на который наложен `TableView` для отображения всех сотрудников ресторана. В правой части также присутствует `AnchorPane`, в которой наложен `GridPane`, который содержит информацию о работнике при нажатии на какого-либо сотрудника (см. рис. 4).

Как можно заметить, в окне в рисунке №4 можно заметить 3 кнопки («Добавить», «Редактировать», «Удалить»), о которых я говорил ранее. Первая из них помогает нам добавить нового сотрудника.

Рис. 5

При нажатии появляется новое окно для добавления нового сотрудника (см. рис. 5).

И наш новый официант оказывается в списке своих коллег в приложении (см. рис. 6).

Ресторан

— □ ×

Кассирам и поварамМенеджерамИнфо

Имя Фамилия	Должность
Фарход Содиков	Менеджер
Теплов Артём	Официант
Примеров Пример	Официант

### Информация

Номер в базе

4

Номер телефона

+992928883280

Должность

Официант

Добавить

Редактировать

Удалить

Рис. 6

Следующая кнопка помогает нам изменить информацию о работнике. При нажатии на кнопку нас переносит в окно, которое мы попадали ранее (см. рис. 6).

Последняя кнопка позволяет пользователю удалить выбранного сотрудника из списка. У нас было 3 работника. Попробуем удалить Теплова Артёма из нашего списка (см. рис. 7).

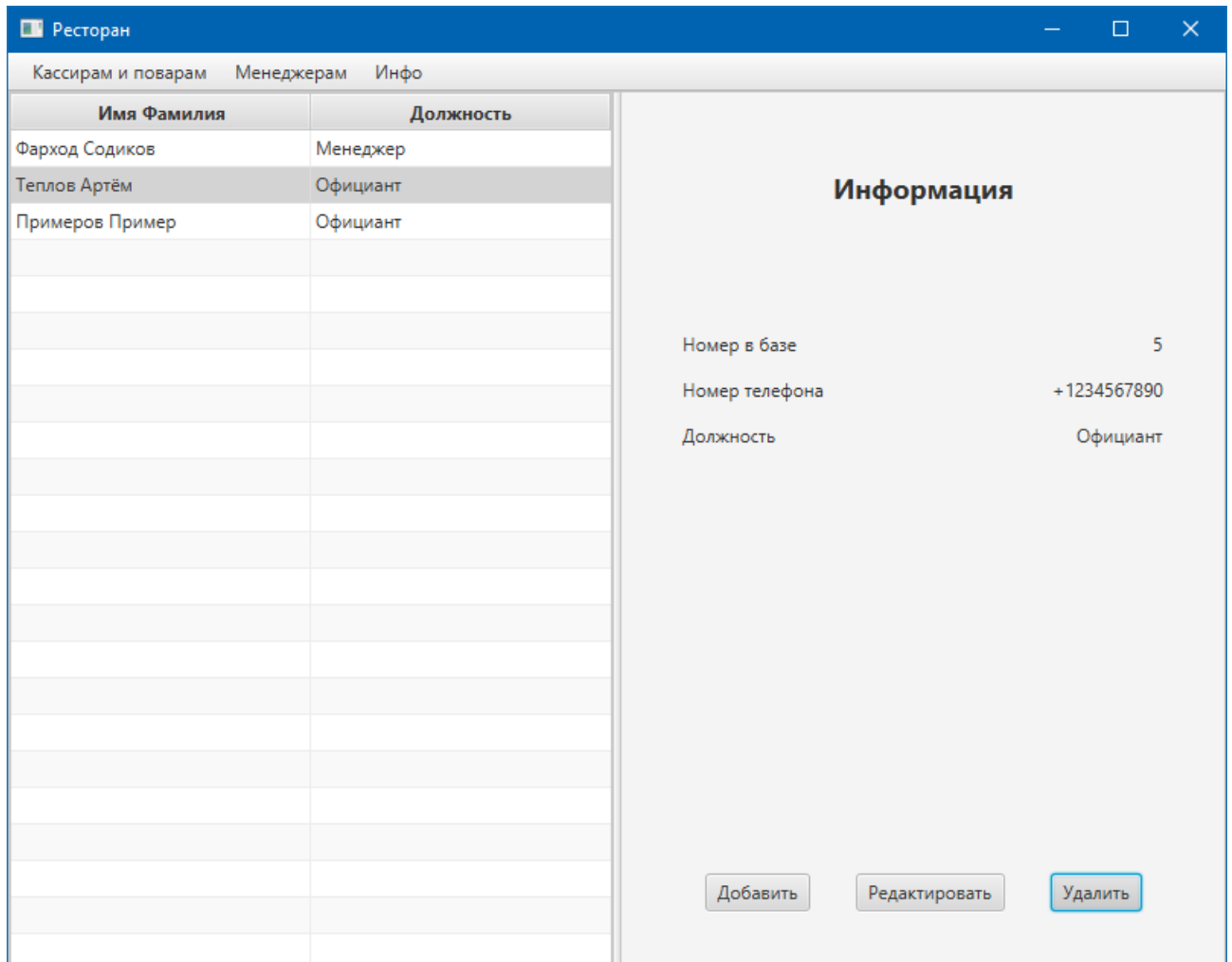


Рис. 7

Как можно видеть (см. рис 8), в списке его больше нет.

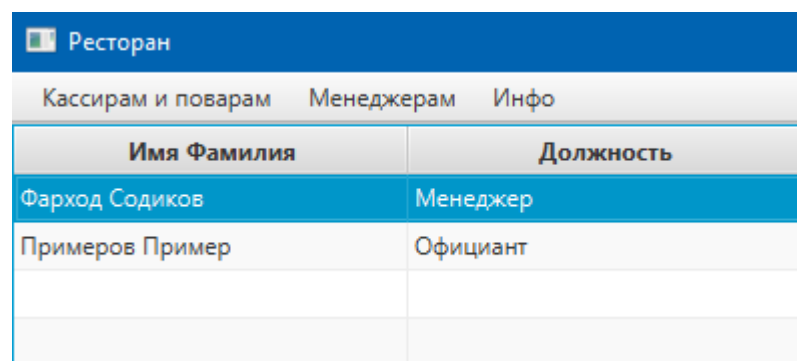


Рис. 8

Во вкладке «Менеджерам» есть ещё один пункт «История заказов», к ней мы вернёмся позже.



Теперь перейдём во вкладку «Кассирам и поварам», а точнее в пункт «Добавить заказ» (смю рис. 9). Как можно уже догадаться это окно нужно для записи нового заказа официантами.

Блюдо	Цена
Пицца	125.0 руб.
Паста	152.0 руб.

Блюдо	Цена	Кол-во
No content in table		

Сумма заказа: 0.0 руб.

Очистить -

Локатор

Создать

Рис. 9

Это окно также как и окно списка разделён на две вертикальные части с помощью SplitPane. В левой верхней части можно увидеть «Поиск» для легкого поиска блюда. Ниже можно увидеть наложенный TableView для создания списка блюд с ценами и названием. Правее от поиска можно увидеть обычный Label с надписью «В заказе». Чуть ниже можно увидеть ещё один TableView. Он показывает все наши блюда, добавленные в заказ, а именно название блюда, цену и количество. Ниже происходит подсчет общей суммы заказа. «Локатор» — это одна из хороших вещей в приложении и в целом в ресторане. Он помогает понять к какому столику относиться определенный заказ. Такие приборы используются в популярных ресторанах и даже кафе России.

Чтоб наконец-таки создать заказ для начала нам нужно из списка блюд выбрать те, которые заказал клиент. Например, закажем три пиццы и одну пиццу. Для этого наводясь на пиццу нажимаем на неё и кликаем по кнопке «Добавить в заказ» три раза и также делаем с пастой, только нажимаем один раз кнопку. Если список блюд большой, то воспользуемся поисковиком (см. рис. 10):

- Наводимся на поисковик и нажимаем
- Пишем название блюда
- Нажимаем на кнопку «ОК»

Ресторан

Кассирам и поварамМенеджерамИнфо

Поиск

Блюдо	Цена
Пицца	125.0 руб.

Рис. 10

После этого пишем номер локатора (1). Вот что в итоге вышло (см. рис. 11).

[illegible]

Рис. 11

Также есть кнопки «Очистить» и «-». Соответственно первая очищает весь заказа со всеми блюдами. И у нас опять как на рисунке №9 пустой заказ (см. рис. 9), а в свою очередь, кнопочка «-» удаляет из списка выбранных блюд одну её единицу. К примеру, удалим одну пиццу из заказа (см. рис. 12).

The screenshot shows a window titled "Ресторан" (Restaurant) with three tabs: "Кассирам и поварам" (For cashiers and cooks), "Менеджерам" (For managers), and "Инфо" (Info). The "Кассирам и поварам" tab is active.

On the left, there is a search bar labeled "Поиск" with an "OK" button. Below it is a table with two columns: "Блюдо" (Dish) and "Цена" (Price).

Блюдо	Цена
Пицца	125.0 руб.
Паста	152.0 руб.

At the bottom of this section is a button labeled "Добавить в заказ" (Add to order).

On the right, there is a section titled "В заказе" (In order). It contains a table with three columns: "Блюдо" (Dish), "Цена" (Price), and "Кол-во" (Quantity).

Блюдо	Цена	Кол-во
Пицца	125.0 руб.	2
Паста	152.0 руб.	1

Below the table, it shows "Сумма заказа: 402.0 руб." (Order total: 402.0 rub.). There are two buttons: "Очистить" (Clear) and a blue button with a minus sign "-". Below these is a label "Локатор" (Locator) and a text input field containing the number "1". At the bottom right is a button labeled "Создать" (Create).

Рис. 12

И наконец-таки, выбрав все блюда, что нам нужны, мы можем создать заказ, нажав на кнопку «Создать» в правом нижнем углу под локатором.

После нажатия высветиться окно об окончательном оформлении заказа (см. рис. 13)

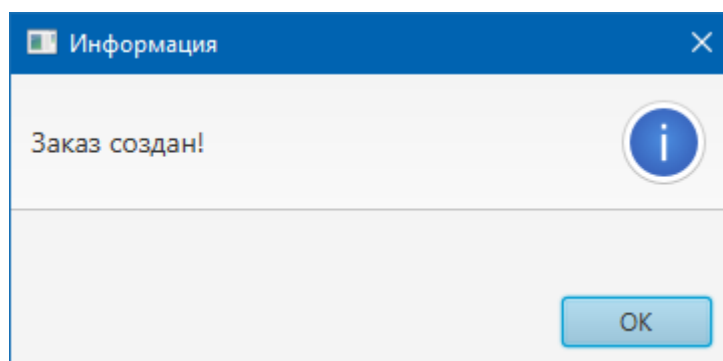


Рис. 13

И сразу же после нажатия на кнопку «ОК» наше окно исчезнет и нас перенесёт во подвкладку «Заказы в обработке» (см. рис. 14).

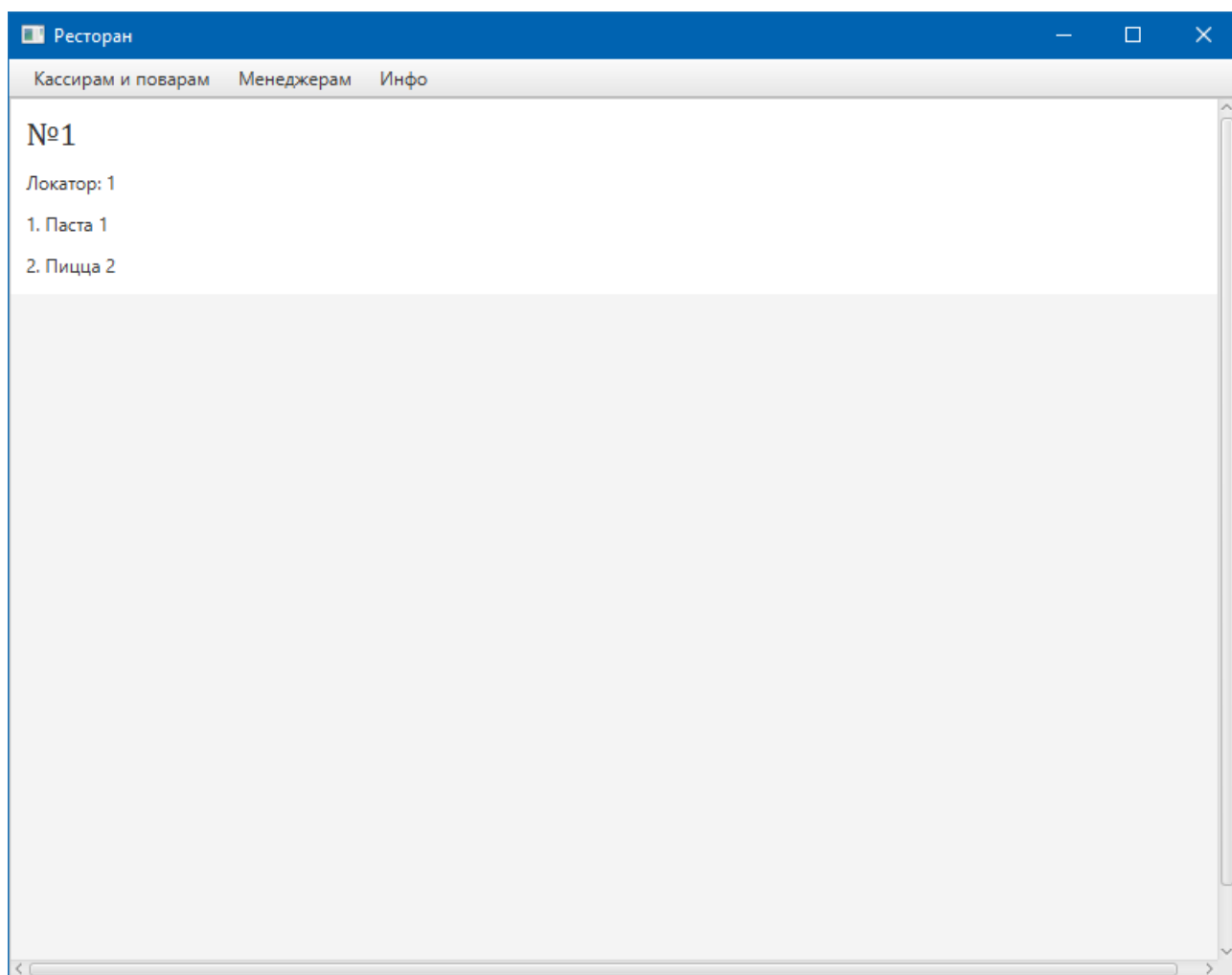


Рис. 14

В это окне можно увидеть список всех заказов, на которых указан номер заказа, номер локатора и список блюд. При нажатии один раз на заказ он загорится в жёлтый цвет, который в свою очередь означает, что заказ готовится (см. рис. 15).

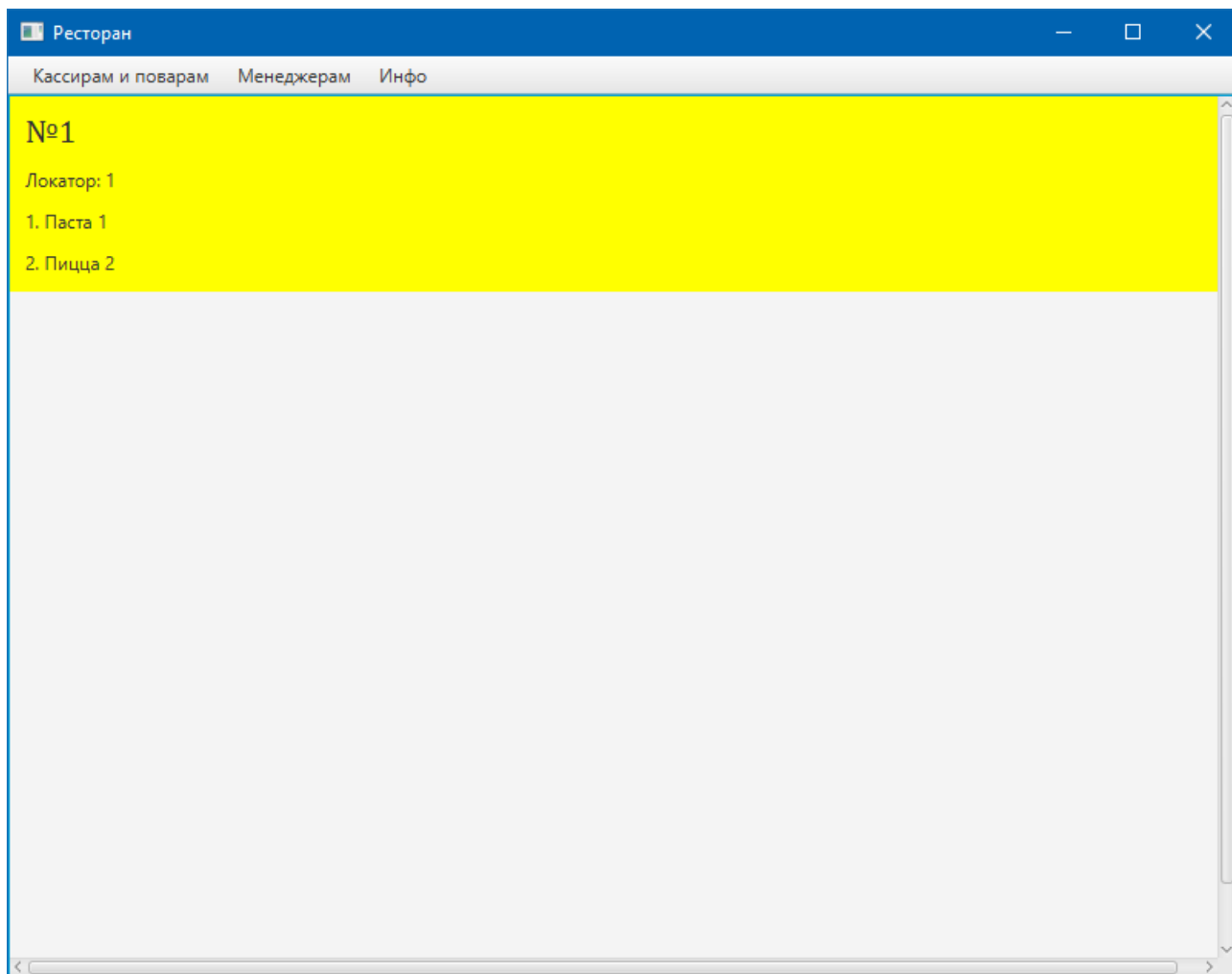


Рис. 15

Если нажать второй раз, то он поменяет цвет в зелёный означающий, что заказ готов к выдаче (см. рис. 16).



Рис. 16

Последнее нажатие удаляет наш заказ из реестра, что означает одобрение заказа клиентом заведения (см. рис. 17).

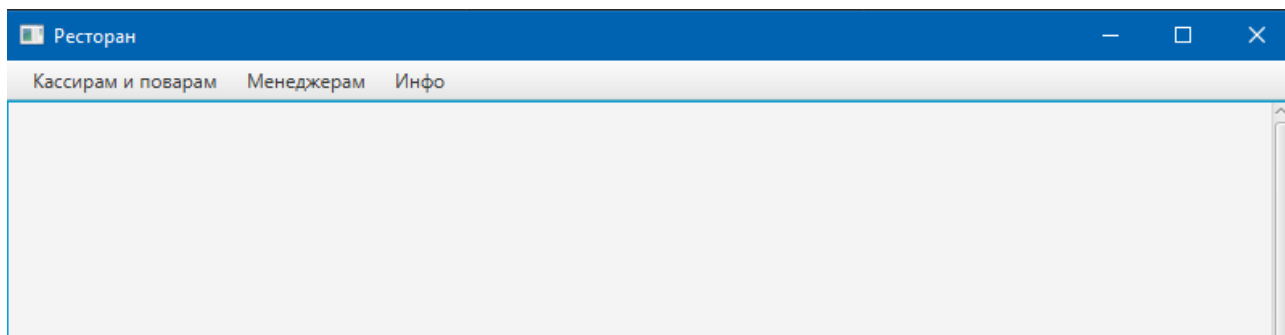


Рис. 17

На самом деле наш заказа удалился всего лишь из окна «Заказы в обработке». Уже одобренные заказы идут в окошко «История заказов», где хранится вся база принятых заказов ресторана (см. рис. 18).

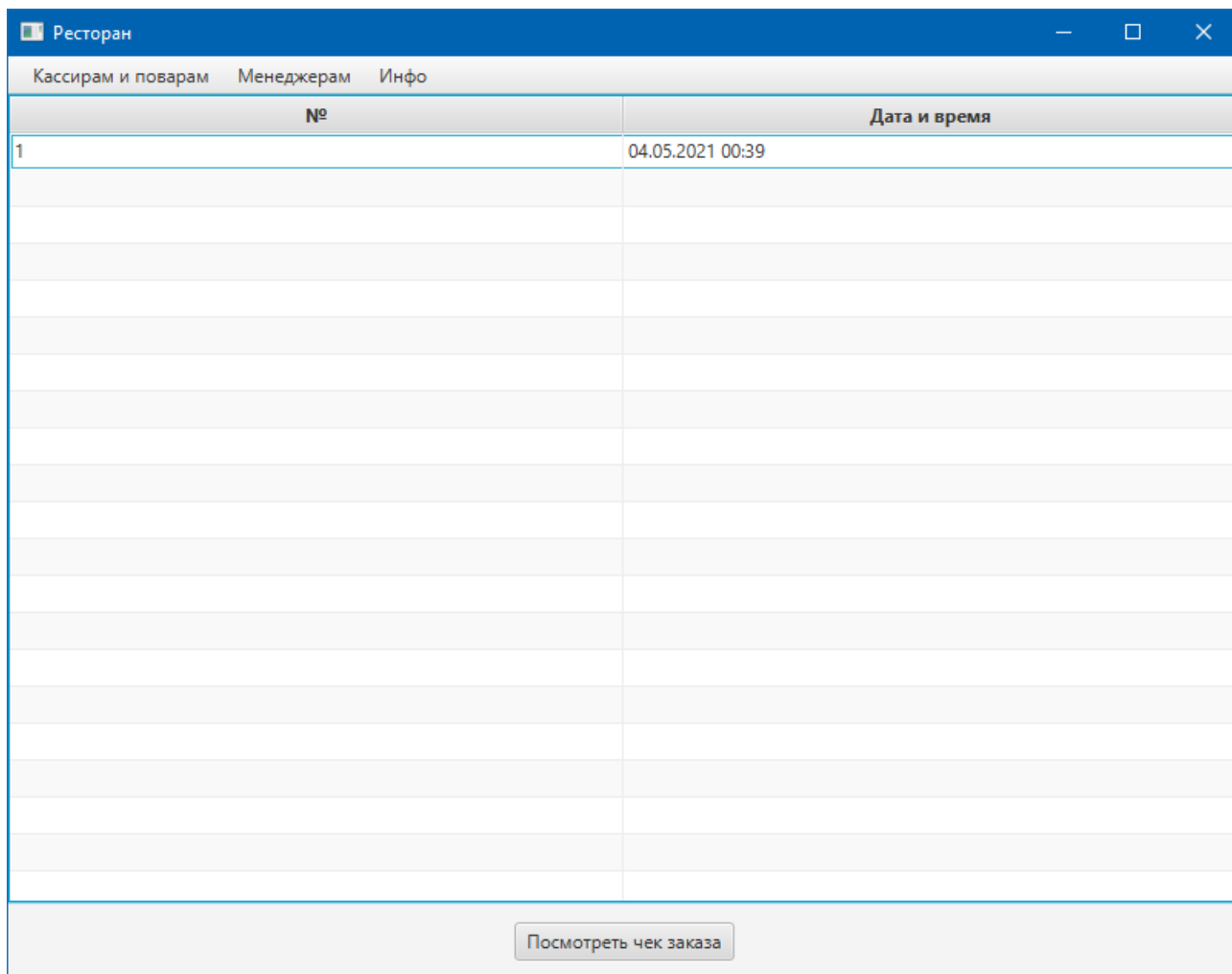


Рис. 18

Это обычное окно с TableView, в котором хранятся наши заказы, именно номер заказа и его дата и время.

Чуть ниже расположена кнопка «Посмотреть чек заказа». Она нужна для того, чтоб посмотреть менеджерам, что заказали клиенты. При нажатии на эту кнопку нас переносит в отдельное окошечко с чеком (см. рис. 19). В верхней части обычный Label с надписью «Общая сумма: <общая сумма> руб.». В центральной части находится TableView заполненная списком блюд, а точнее их названиями, стоимостью и количеством. И чуть ниже можно видеть обычную кнопку для закрытия окна чека.

Чек №1 от 04.05.2021 00:39

**Общая сумма: 402.0 руб.**

Блюдо	Стоимость	Кол-во
Паста	152.0 руб.	1
Пицца	125.0 руб.	2

Закрыть

Рис. 19

### 4.3 Состав приложения

Информационно-справочная система состоит из 3 составляющих:

1. Сервер
2. Клиент
3. База данных

Для реализации **серверной** части использовались ряд зависимостей:

1. Lombok – это специальный плагин, который помогает превращать аннотации в Java-код.
2. Spring Web – это веб-фреймворк, который позволяет легко создавать веб-приложения, с использованием системы MVC (Model-View-Controller).  
Для запуска Spring Web MVC требуется специальный контейнер Servlet.
3. Spring Data JPA – это механизм для легкого и удобного взаимодействия с сущностями БД, извлечения данных и т.п.
4. Spring Boot DevTools – это один из инструментов Spring Boot, который даёт дополнительный функционал времени разработки.
5. Apache Maven – это фреймворк, который автоматизирует складку, склейку проекта.
6. Spring-boot-maven-plugin – это плагин, который нужен для обеспечения Spring Boot в Maven.

Для реализации **клиентской** части использовались следующие инструменты:

1. JavaFX – инструмент Java для отображения графического интерфейса приложения. Включает в себя следующие компоненты:
  - a) JavaFX-controls – модуль для создания элементов управления, скинов и диаграмм.
  - b) JavaFX-fxml – нужен для работы с языком разметки FXML, который в свою очередь основан на XML.
  - c) JavaFX-graphics – это модуль позволяющий рисовать, анимировать использовать CSS. Нужен для работы функционала окон.



2. GSON – это разработка программистов Google. Эта библиотека нужна для серкализации и десериализацией объектов Java в JSON. Сereaлизация – это процесс преобразования структур данных в формат, который может быть сохранён или передан или реконструирован позже.
3. OkHttp – это HTTP-клиент, который снижает задержку запросов и помогает избегать повторных запросов в сети. Также OkHttp работает, когда сеть вызывает проблемы тем, что он незаметно восстановится после распространённых проблем с подключением.

В качестве СУБД использовалась популярная система H2. Она нам подходит тем, что её можно интегрировать в Java и использовать в клиент-серверных приложениях.

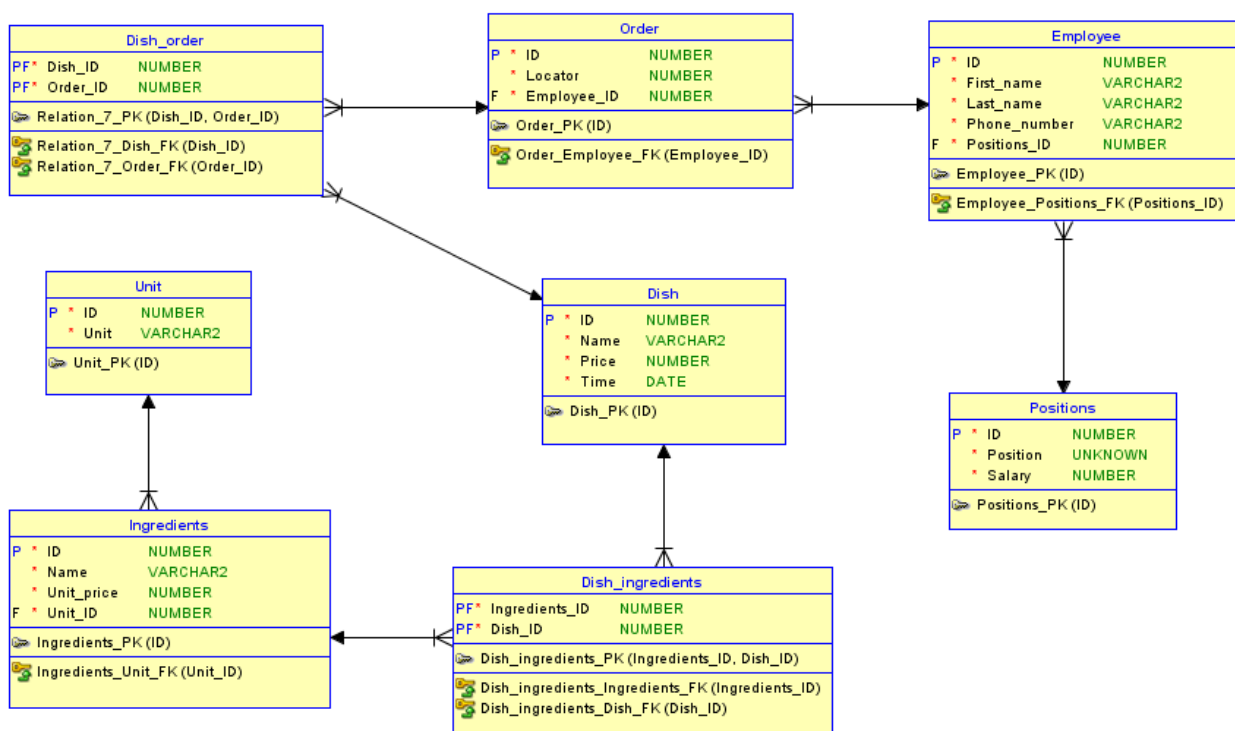


Рис. 20

На рисунке №20 показана наша система базы данных. Она состоит из сущностей, которые содержат поля, и связей. В каждой сущности есть первичные ключи (Primary Key – PK) и в некоторых внешние ключи (Foreign Key – FK). Количество полей у нас равно восьми:

- Unit (Поля: ID, Unit)
- Ingredients (Поля: ID (PK), Name, Unit\_price, Unit\_ID(FK))

- Dish\_ingredients (Поля: Ingredients\_ID (FK), Dish\_ID (FK))
- Dish (Поля: ID (PK), Name, Price, Time)
- Dish\_order (Поля: Dish\_ID (FK), Order\_ID (FK))
- Order (Поля: ID (PK), Locator, Employee\_ID (FK))
- Employee (Поля: ID (PK), First\_name, Last\_name, Phone\_number, Positions\_ID (FK))
- Positions (Поля: ID (PK), Position, Salary)

Звездочка, стоящая слева от названия поля, означает, что поле обязательно.

Буква P (Primary) означает, что поле первичное.

Буква F (Foreign) означает, что поле внешнее и взято из другой сущности.

Сочетание букв PF означает, что мы используем несколько полей из разных сущностей для связи. В рисунке №20 можно увидеть два примера таких связей:

- Сущности Order и Dish имеют связь вида ManyToMany и связаны они через сущность Dish\_order.
- Сущности Ingredients и Dish также имеют связь вида ManyToMany и связаны они через сущность Dish\_ingredients.

При создании связи «Многие ко многим» в логической модели, программа автоматически создаёт новую сущность в реляционной с атрибутами FP.

## 5. Назначение и состав классов программы

### Сервер

Сервер имеет 24 класса, которые делятся на 4 группы, в каждом из которых имеется по 6 классов:

- Controller (Контроллер) – классы, которые обрабатывают входящие запросы от клиентов.
- Entity (Сущность) – данные, которые передаются сервером к клиенту.
- Repository (Репозиторий) – это наборы стандартных методов JPA для работы с базами данных.
- Service (Сервис) – классы-посредники между контроллерами и репозиториями. Нужны они для обработки данных, полученных на контроллере.

Контроллеры:

- DishController – это класс для HTTP-запросов, который связан с данными о блюдах.
- EmployeeController – это класс для HTTP-запросов который связан с данными о сотрудниках.
- IngredientController – это класс для HTTP-запросов, который связан с данными об ингредиентах.
- OrderController – это класс для HTTP-запросов, который связан с данными о заказах.
- PositionController – это класс для HTTP-запросов, который связан с данными о должностях.
- UnitController – это класс для HTTP-запросов, который связан с данными об единицах измерения.

#### Сущности:

- Dish – это сущность для описания блюд в СУБД.
- Employee – это сущность для описания сотрудников в СУБД.
- Ingredient – это сущность для описания ингредиентов в СУБД.
- Order – это сущность для описания заказов в СУБД.
- Position – это сущность для описания должностей в СУБД.
- Unit – это сущность для описания единиц измерения в СУБД.

#### Репозитории:

- DishRepository – это JpaRepository интерфейс для работы с массивом данных о блюдах.
- EmployeeRepository – это JpaRepository интерфейс для работы с массивом данных о сотрудниках.
- IngredientRepository – это JpaRepository интерфейс для работы с массивом данных об ингредиентах.
- OrderRepository – это JpaRepository интерфейс для работы с массивом данных о заказах.
- PositionRepository – это JpaRepository интерфейс для работы с массивом данных о должностях.
- UnitRepository – это JpaRepository интерфейс для работы с массивом данных об единицах измерения.

#### Сервисы:

- DishService – это сервис для работы с блюдами.
- EmployeeService – это сервис для работы с сотрудниками.
- IngredientService – это сервис для работы с ингредиентами.
- OrderService – это сервис для работы с заказами.
- PositionService – это сервис для работы с должностями.
- UnitService – это сервис для работы с единицами измерения.

Клиент имеет 7 классов и несколько FXML-файлов:

- EmployeeEditDialog – это класс окна настройки сотрудника.
- EmployeeLayout – это класс окна информации сотрудника.
- OrdersEditDialog – это класс окна настройки заказа.
- OrdersLayout – это класс окна информации о заказе.
- CheckDialog – это класс окна заказов в обработке.
- OrdersHistoryLayout – это класс окна прошлых заказов.
- RootController – это контроллер-обработчик вкладок.

СУБД имеет 6 основных моделей сущности и 2 вспомогательных:

- Unit – это модель сущности единицы измерения.
- Ingredients – это модель сущности ингредиентов.
- Dish\_ingredients – это вспомогательная модель для связи ManyToMany двух сущностей Dish и Ingredients.
- Dish – это модель сущности блюда.
- Dish\_order – это вспомогательная модель для связи ManyToMany двух сущностей Dish и Order.
- Order – это модель сущности заказа.
- Employee – это модель сущности сотрудника.
- Positions – это модель сущности должностей.

## 6. Заключение

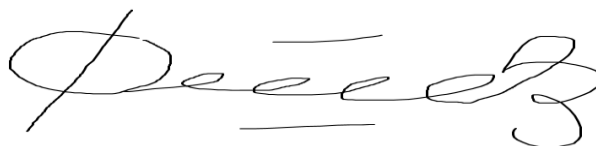
В данной курсовой работе все поставленные задачи были выполнены успешно: программа работает стабильно, без аварийного выключения. Была разработана информационно-справочная система «Ресторан» с использованием языка программирования – Java и фреймворка Spring. Программа проста в использовании для любых пользователей и поэтому к приложению не нужны никакие инструкции. Разработка имеет возможность дальнейшего улучшения в функционале и интерфейсе.

В результате разработки данного приложения был получен большой опыт работы с одним из сложных языков программирования с использованием дополнительных фреймворков. Также не малый опыт с работами структур баз данных. Большой объём программного кода даёт уверенность при работе с крупными проектами.

Выполнил:

Студент группы ПИ19-4

Содиков Фарход Фирдавсович



---

(Подпись)

### **Список литературы:**

- Мартин Роберт К. Чистый код. Создание анализ и рефакторинг. - СПб: Питер, 2019. - 464 с.
- Козмина Ю., Харроп Р. Spring 5 для профессионалов. - Киев: Диалектика-Вильямс, 2019. - 1120 с.
- Коузен К. Современный Java. Рецепты программирования. - М.: ДМК Пресс, 2018. - 274 с.
- Шилдт Г. Java. Полное руководство.-Киев: Диалектика, 2018.-1488 с.
- Прохоренок Н.А. JavaFX. - СПб: БХВ-Петербург, 2020. - 768 с.

