

МИНОБРНАУКИ РОССИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет *компьютерных наук*

*Система управления рестораном*

*Курсовая работа по дисциплине «Технологии программирования»*

09.03.04 Программная инженерия

Обучающийся студент 3 курса

М. В. Злобин

Обучающийся студент 3 курса

А. Н. Пожидаев

Преподаватель

В.С. Тарасов

Воронеж 2020

## Содержание

Введение .....	4
Постановка задачи.....	5
Описание функционала системы.....	6
1. Анализ предметной области .....	7
1.1.Сравнение с аналогами .....	7
1.2 Анализ задач .....	9
1.3 Графическое описание работы системы .....	10
Диаграмма прецедентов.....	10
Диаграмма последовательностей .....	10
Диаграмма коммуникаций .....	14
Диаграмма состояний .....	16
Диаграмма активности.....	16
2. Реализация приложения.....	23
Анализ средств реализации .....	23
Архитектура системы .....	23
Диаграмма классов.....	24
Диаграмма объектов .....	25
Схема базы данных .....	26
Диаграмма развертывания .....	27
Серверная часть.....	29
Клиентская часть.....	31
Тестирование .....	40
Дымовое тестирование .....	40
Функциональное тестирование .....	41

Юзабилити тесты .....	43
Аналитика .....	46
Список использованных источников .....	49

## **Введение**

В современном мире тяжело представить свою жизнь без ресторанов. К сожалению, не все заведения работают корректно, и их владельцы не в состоянии исправить это самостоятельно, поэтому они вынуждены искать какую-либо систему для оптимизации. Однако, для самих ресторанов не существует системы, направленной на удобное распределение человеческих ресурсов, или они слишком дорогие.

Таким образом, целью нашей работы является создание такой системы, которая позволит управлять снабжением ресторана и оформлением заказов.

В свою очередь, работники ресторана смогут просматривать новые заказы и получать информацию о них.

## **Постановка задачи**

**Цель:** автоматизировать работу персонала внутри организации.

**Сфера применения:** ресторан.

**Требования к системе в целом:**

- Система должна быть реализована по клиент-серверной архитектуре на технологиях ASP NET API и Angular любых версий
- Система должна использовать MSSQL
- Система не должна позволять доступ неавторизованным пользователям доступ к данным системы
- Система должна предоставлять физический доступ к интерфейсу только той роли, под которой вошел в систему пользователь

**Задачи:**

1. Провести анализ рынка с целью выявления достоинств и недостатков схожих по функционалу систем
2. Спроектировать приложение с учетом информации, полученной ранее в ходе анализа
3. Реализовать прототип приложения, обладающий функционалом, описанным в требованиях
4. Описать процесс разработки и результат

## **Описание функционала системы**

Система управления рестораном представляет собой веб-приложение с ролевым доступом к компонентам. В системе должны быть предусмотрены следующие роли. В системе предусмотрены следующие роли:

- Администратор
- Повар
- Официант
- Провизор

Основные функциональные возможности Системы:

- Зарегистрировать сотрудника
- Настройка и просмотр меню
- Закупка ингредиентов и внесение их в систему
- Получение статистики о выручке, проданных блюдах
- Просмотр ингредиентов текущих и необходимых для закупки. Считать, что какой-либо продукт необходимо закупить если у оставшихся продуктов срок годности меньше недели или ингредиентов недостаточно для приготовления 10 порций блюда
- Составление заказа и пометка его готовым
- Оповещения о готовности заказов

## **1. Анализ предметной области**

В современном мире мы наблюдаем проникновение информационных технологий практически во все сферы человеческой деятельности. Сейчас система автоматизации работы ресторана, бара и кафе - это не модное новшество, а необходимое условие конкурентоспособности всего бизнеса. Без специальных программ не получится организовать контроль над работой персонала, продажи не поднимутся, а издержки будут постоянно расти. Все процессы, которые происходят в заведении, нужно отслеживать. Важно соединить в одном софте деятельность кухни и зала, а также учесть возможность просматривать статистику.

Установка ПО сделает проще жизнь работникам и руководству.

### **1.1.Сравнение с аналогами**

Среди аналогов, направленных на автоматизацию ресторанов/кафе/баров, можно выделить следующие приложения:

#### **1. Трактирь**

( <https://traktir.ru/production/> )

Программа, включает в себя целую линейку разнообразных продуктов. Автоматизирует процесс обслуживания, помогает управлять основной деятельностью, вести бухучет и руководить бизнесом.

Минусами этой программы можно считать:

- Сложный и непонятный интерфейс.
- Высокая стоимость.
- Отсутствие связи между поварами и официантами.

Плюсами этой программы можно считать:

- Отслеживает факты злоупотребления персоналом.
- Интегрируется с 1С.

## 2. Cafe Manager

( <https://cafe-manager.ru/> )

Еще один облачный сервис, помогающий контролировать работу персонала, управлять им. Доступна настройка под потребности конкретного кафе. Возможность оперативно добавлять товары в статистику, планировать дальнейшее развитие. Войти в систему можно с любого планшета, ПК или телефона с помощью логина и пароля. Подходит для любого заведения.

Минусами этой программы можно считать:

- В случае отсутствия интернета могут появиться сбои.
- Высокая стоимость
- Нет хранилища файлов.

Плюсами этой программы можно считать:

- Доступен в 99% случаев.
- Можно заказать разработку функционала под требования конкретного заведения.

Так же есть много других сервисов, которые представляют клиентам разнообразные системы под их заведения.:

## 1. R-Keeper

(<http://restaurants.rkeeper.ru/>)



2. Iiko

(<https://iiko.ru/>)

3. Tillypad

(<https://tillypad.ru/>)

4. Quick Resto

(<https://quickresto.ru/>)

5. Poster

(<https://joinposter.com/>)

6. Ivideon

(<https://ru.ivideon.com/>)

Перечисленные сервисы предусматривают работу системы, как для владельцев, так и для самих работников, но являются платными решениями и не всегда помогут удовлетворить владельцев нужным им функционалом.

## **1.2 Анализ задач**

Подробное описание всех прецедентов (пункт 4) и таблица ролевого доступа к компонентам (пункт 3.1) описаны в Настоящем Техническом Задании.

Вход в систему, то есть авторизация, будет осуществляться с помощью JWT токена (JSON Web Token), так как это самый популярный и эффективный способ авторизации для клиент-серверной архитектуры. Ее смысл состоит в том, чтобы на стороне сервера проверить введенный логин и пароль пользователя, далее сгенерировать уникальный токен доступа, подписанный секретным ключом, и передать его клиенту. Клиент же в свою очередь должен использовать возвращенный ему токен в заголовках в каждом своем

запросе на сервер, чтобы сервер мог идентифицировать текущего пользователя. Графически это изображено на диаграмме активности для прецедента «Вход в систему».

### 1.3 Графическое описание работы системы

#### Диаграмма прецедентов

Более подробно диаграмму можно рассмотреть на доске в Miro.

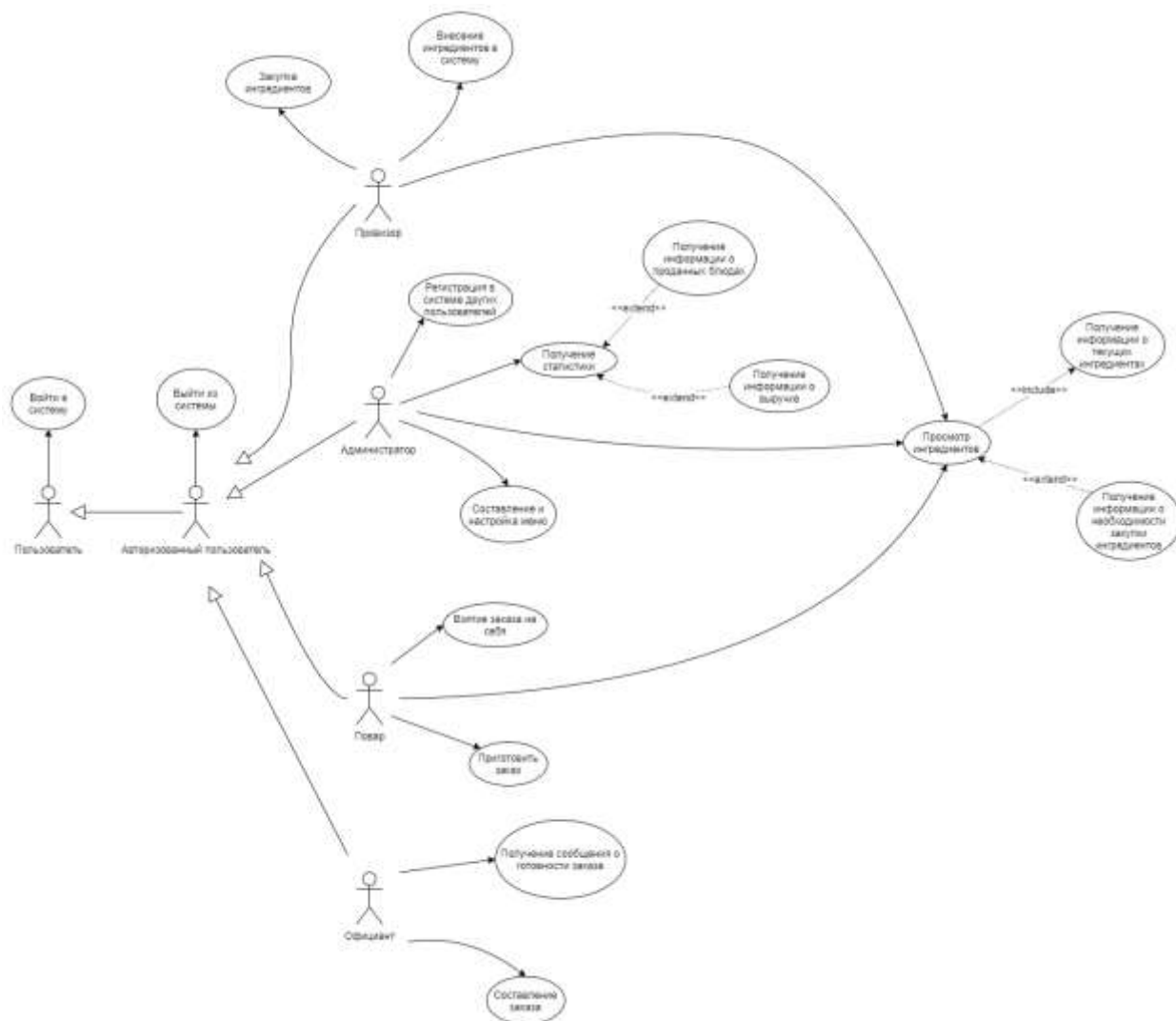


Рис. 1. Use-Case

#### Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Добавления ингредиентов». Подразумевается, что пользователь уже вошел в систему.

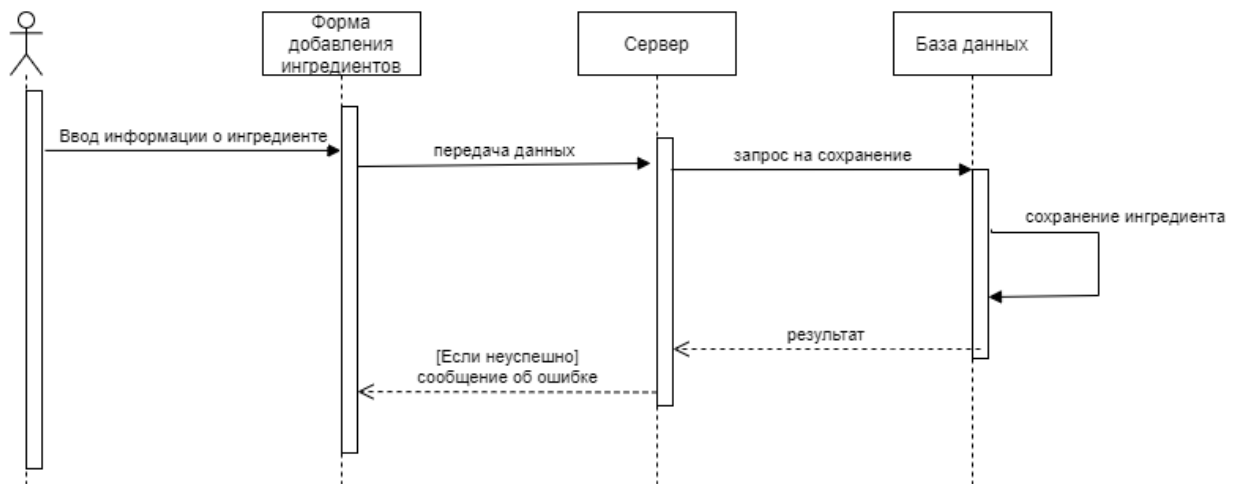


Рис. 2. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Закупки ингредиентов». Предполагается, что пользователь уже вошел в систему.

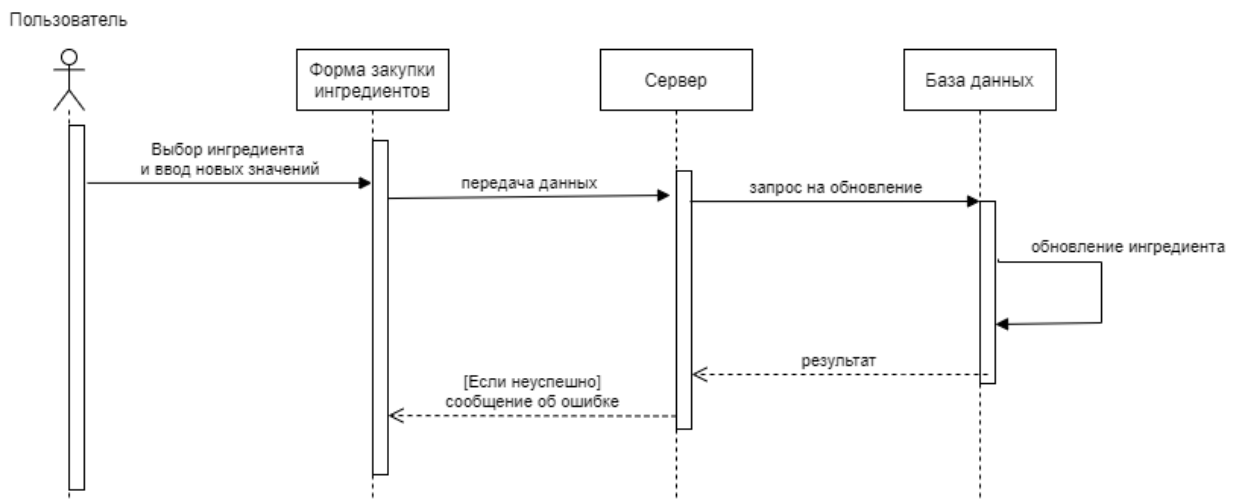


Рис. 3. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Регистрация сотрудника». Предполагается, что пользователь уже вошел в систему.

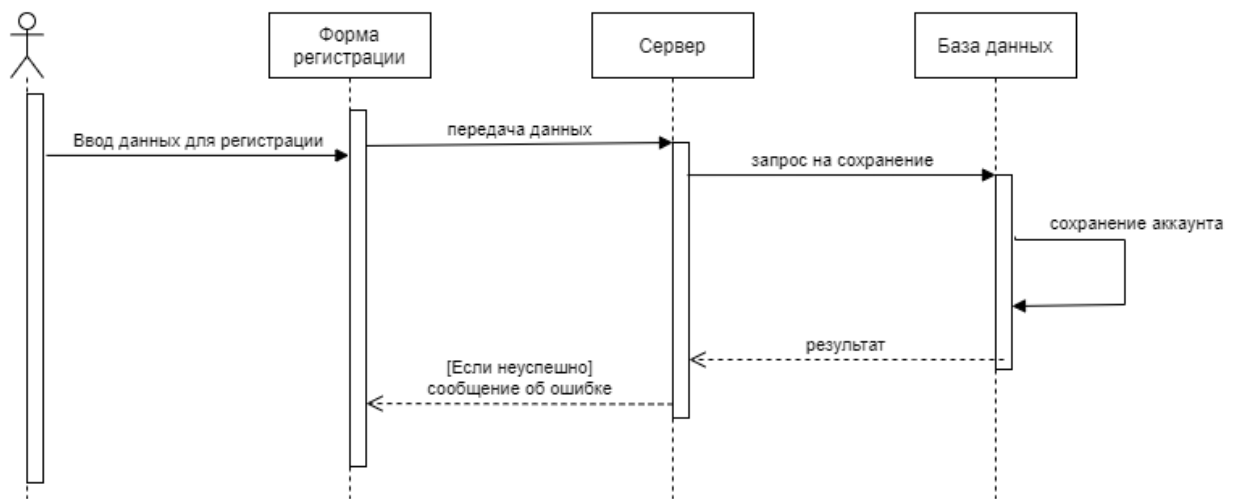


Рис. 4. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Создание заказа».  
 Подразумевается, что пользователь уже вошел в систему.

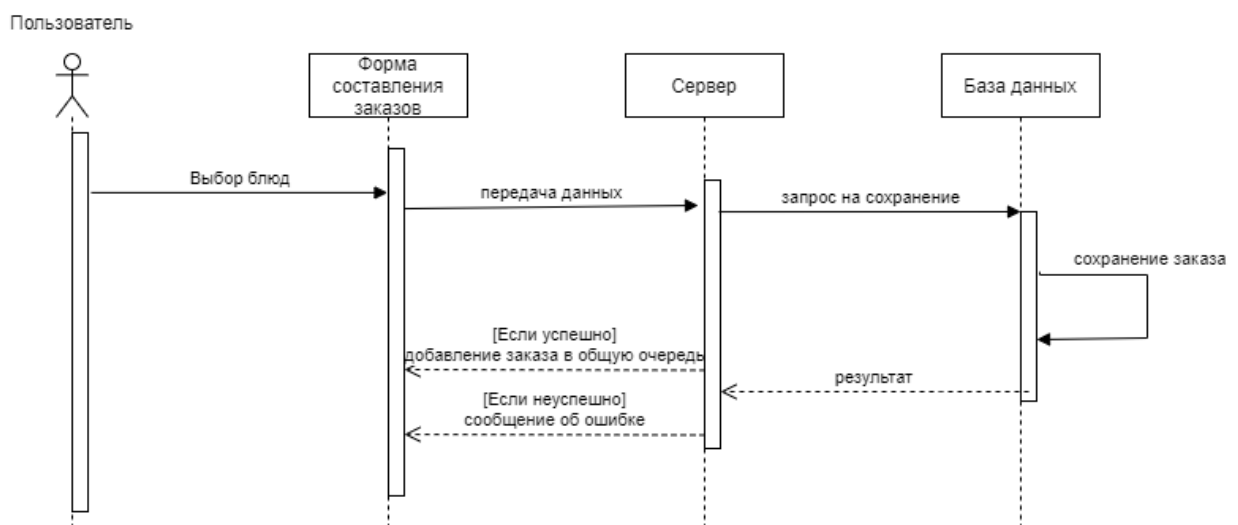


Рис. 5. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Настройка меню».  
 Подразумевается, что пользователь уже вошел в систему.

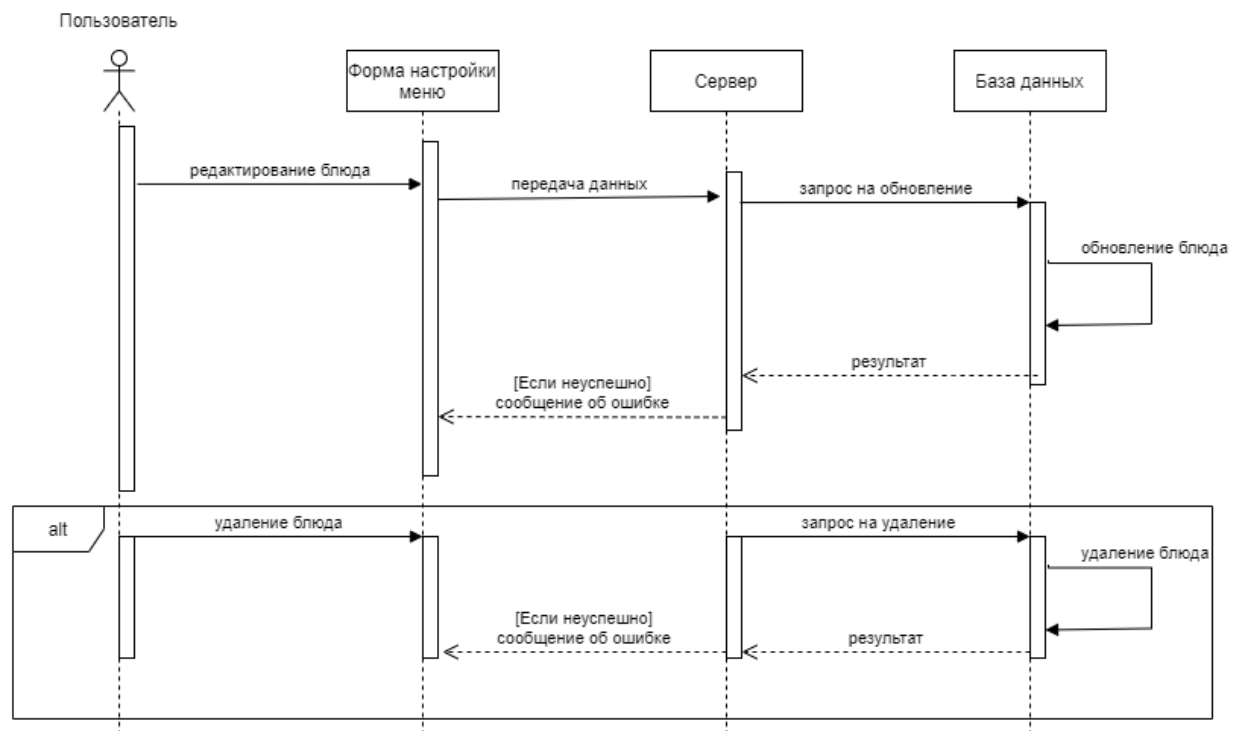


Рис. 6. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Взять заказ».

Подразумевается, что пользователь уже вошел в систему.

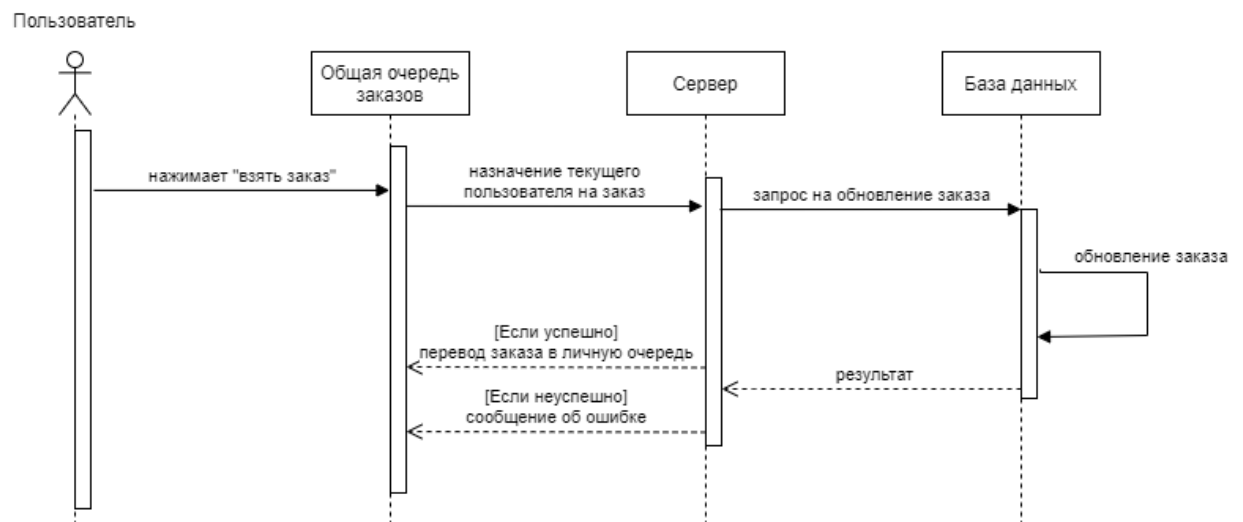


Рис. 7. Диаграмма последовательностей

Диаграмма последовательностей для прецедента «Приготовить заказ».

Подразумевается, что пользователь уже вошел в систему.

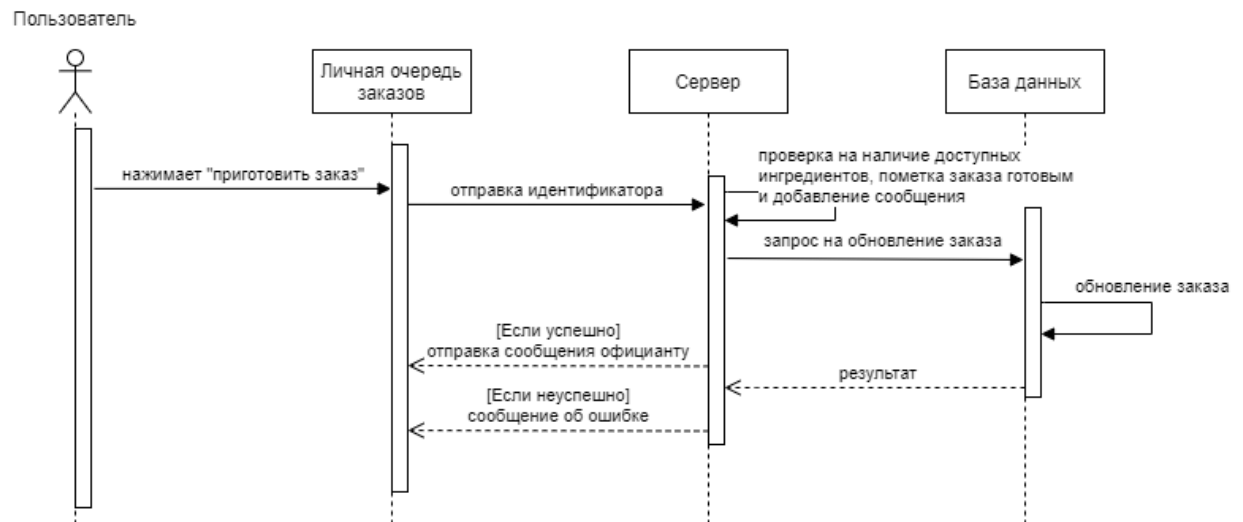


Рис. 8. Диаграмма последовательностей

Диаграмма последовательностей для прецедентов «Получение статистики» и «Просмотр ингредиентов». Подразумевается, что пользователь уже вошел в систему.

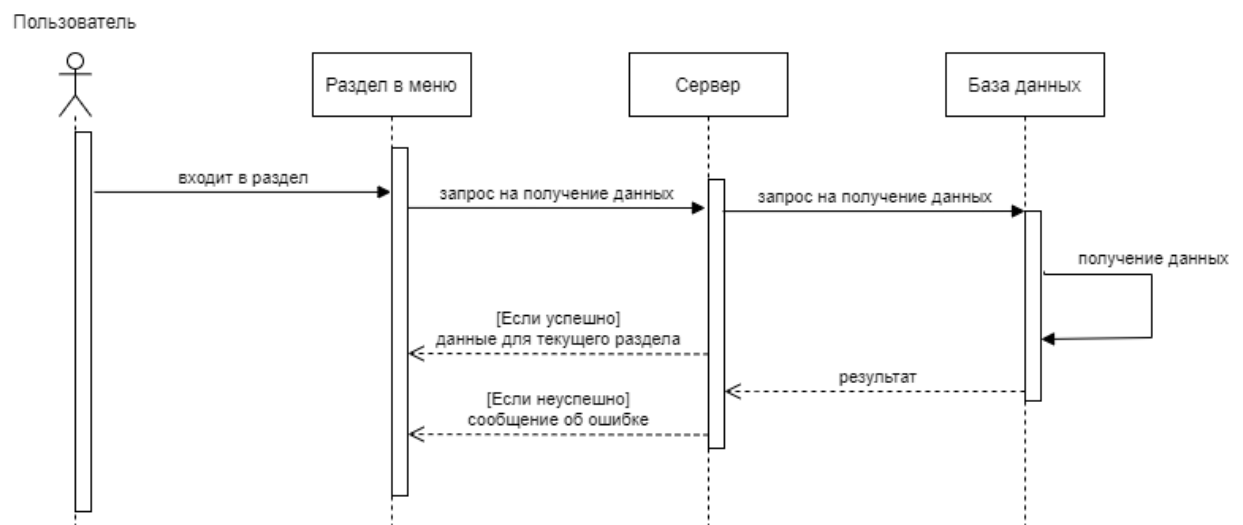


Рис. 9. Диаграмма последовательностей

## Диаграмма коммуникаций

Диаграмма коммуникаций для прецедента «Взять заказ». Подразумевается, что пользователь уже вошел в систему.

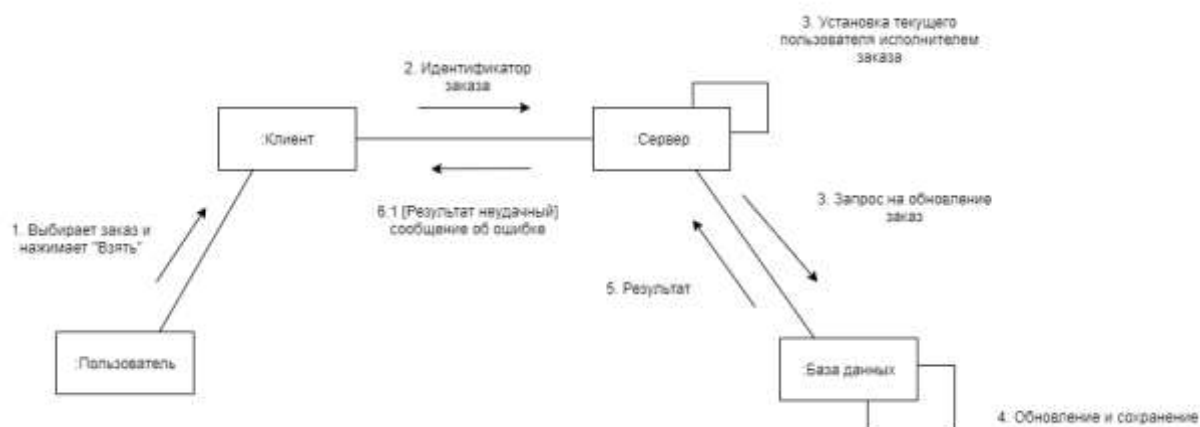


Рис. 10. Диаграмма коммуникаций

Диаграмма коммуникаций для прецедента «Приготовить заказ». Подразумевается, что пользователь уже вошел в систему.

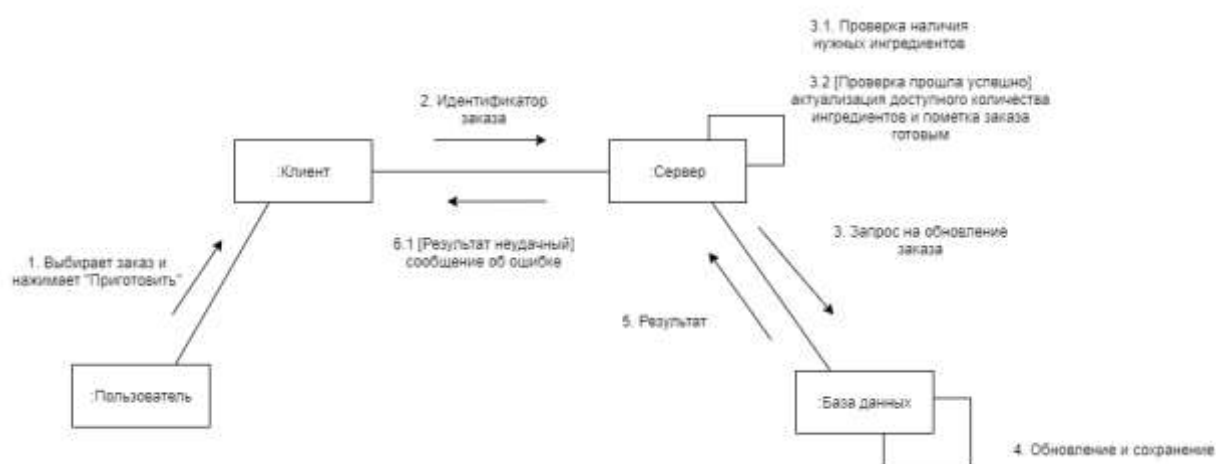


Рис. 11. Диаграмма коммуникаций

Диаграмма коммуникаций для прецедентов «Добавить ингредиент», «Создать заказ», «Регистрация сотрудника». Подразумевается, что пользователь уже вошел в систему.

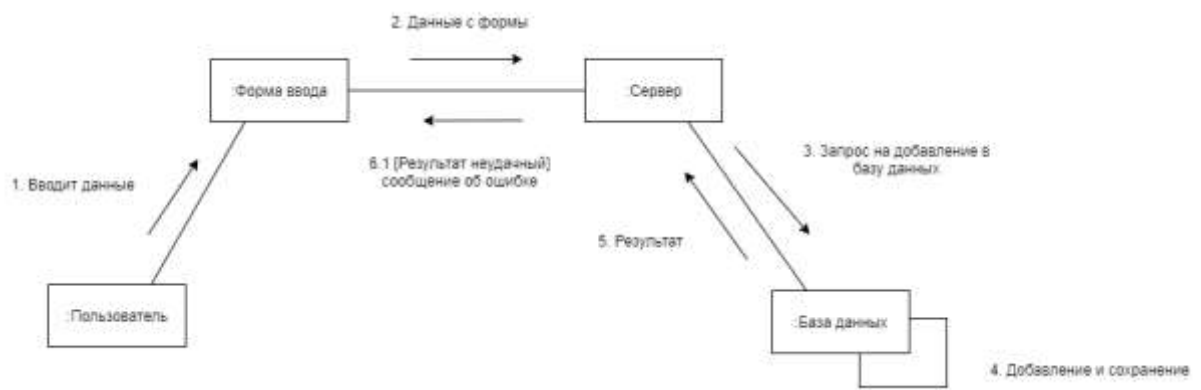


Рис. 12. Диаграмма коммуникаций

Диаграмма коммуникаций для прецедентов «Настройка меню», «Закупка ингредиентов». Подразумевается, что пользователь уже вошел в систему.

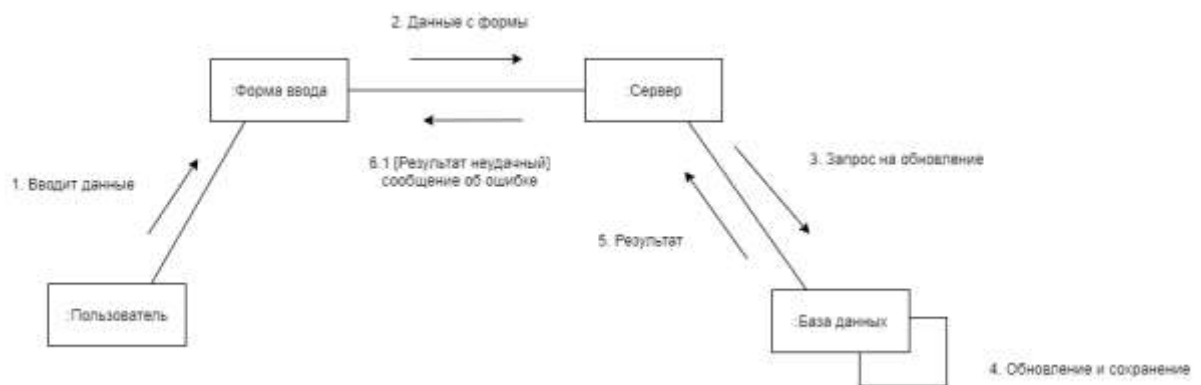


Рис. 13. Диаграмма коммуникаций

## Диаграмма состояний

Более подробно диаграмму можно рассмотреть на доске в Miro.

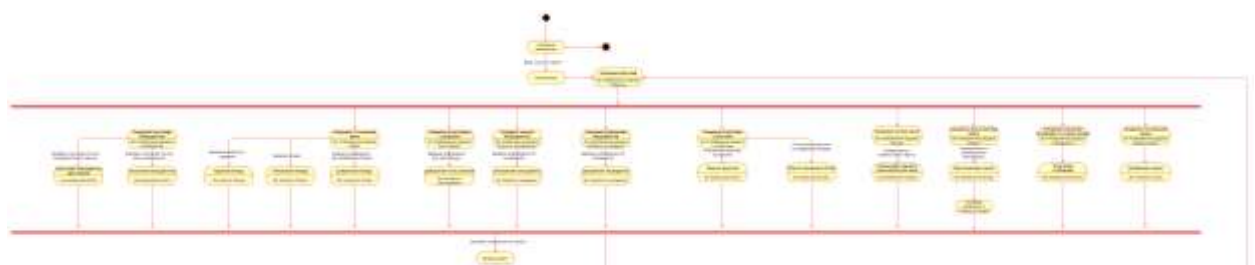


Рис. 14. Диаграмма состояний

## Диаграмма активности



Диаграмма активности для прецедента «Вход в систему» (Авторизация).

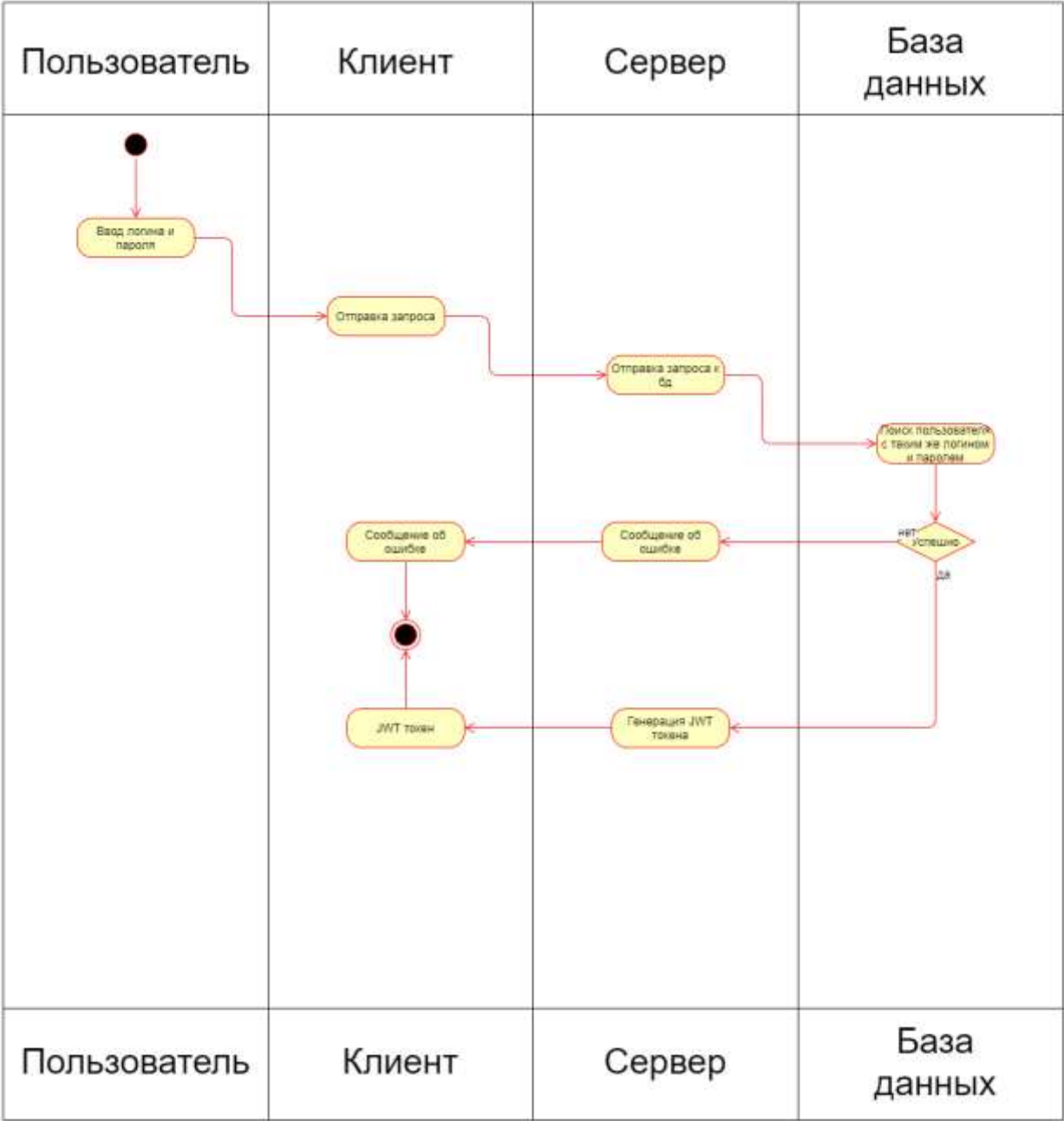


Рис. 15. Диаграмма активности

Диаграмма активности для прецедента «Взять заказ».

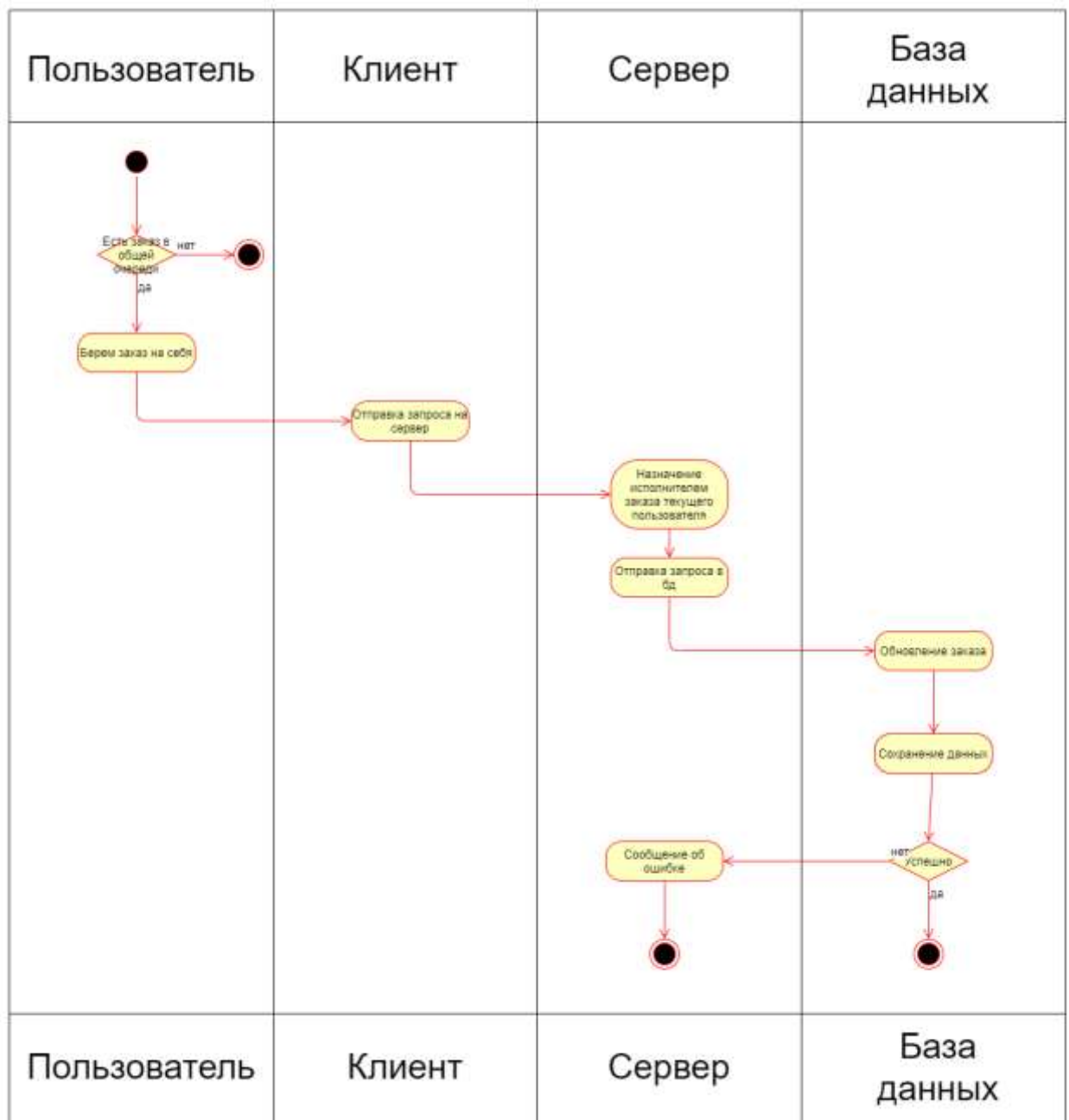


Рис. 16. Диаграмма активности

Диаграмма активности для прецедента «Приготовить заказ».

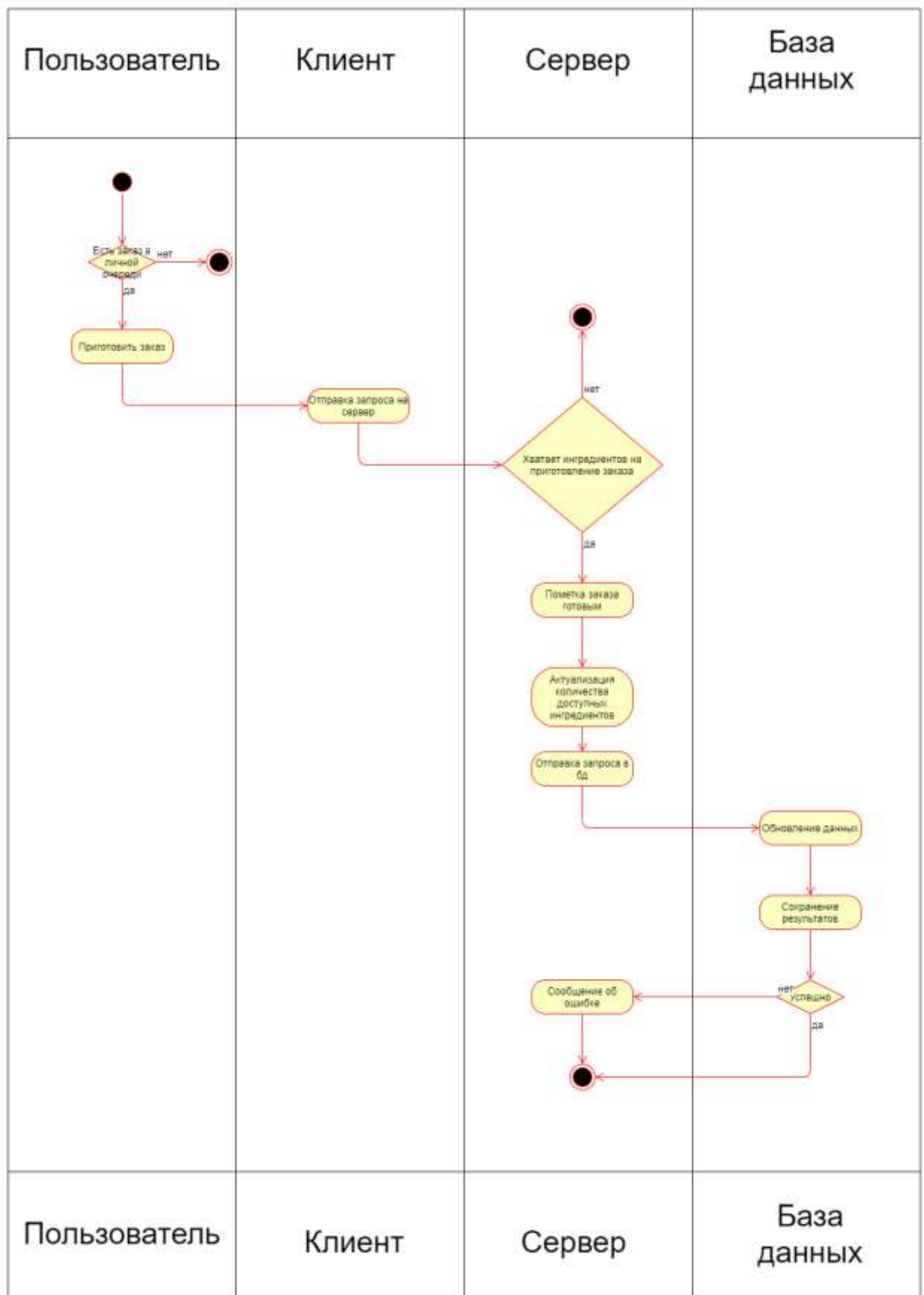


Рис. 17. Диаграмма активности

Диаграмма активности для прецедентов «Добавление ингредиентов», «Создание заказа» и «Регистрация сотрудника».

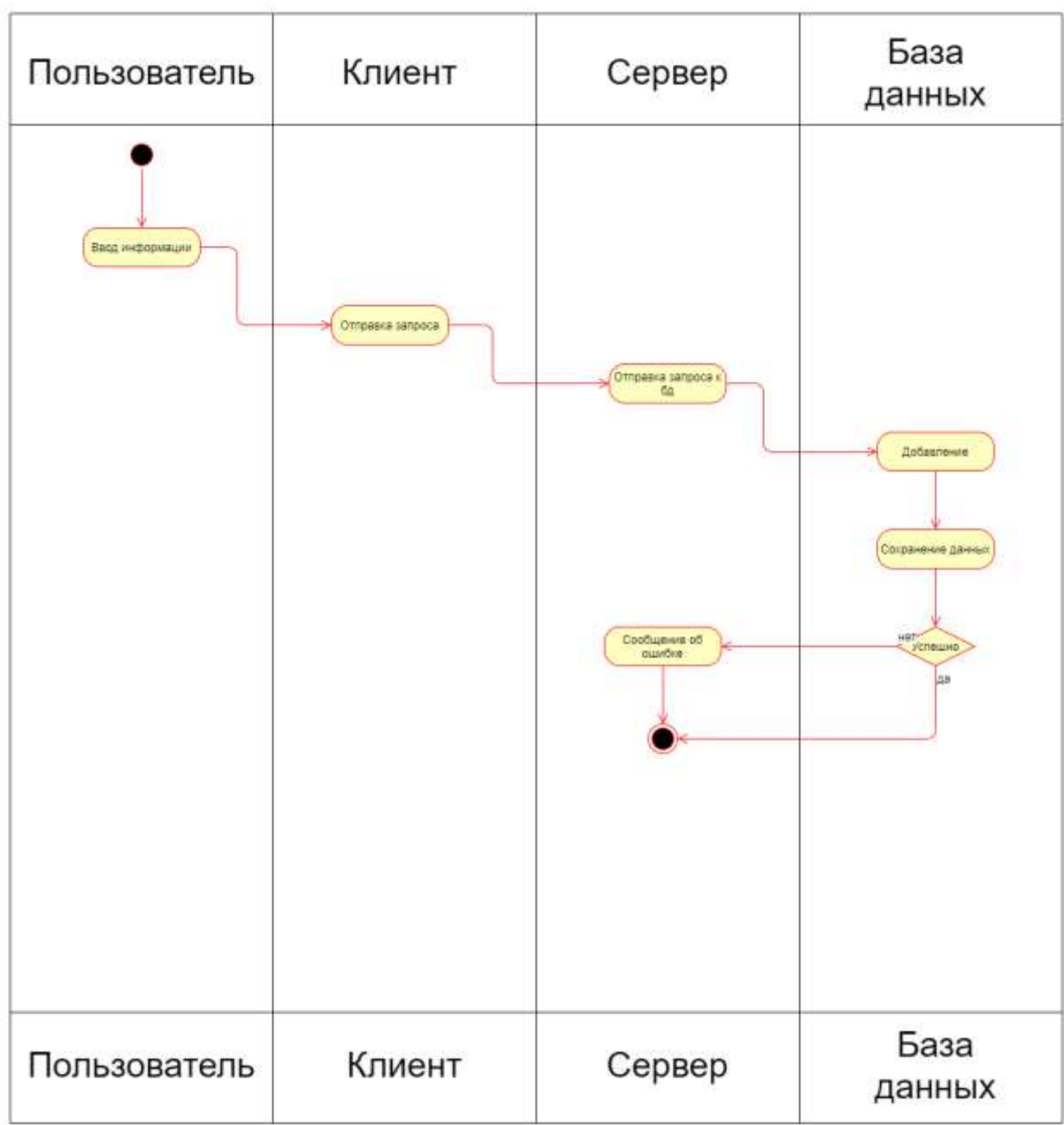


Рис. 18. Диаграмма активности

Диаграмма активности для прецедентов «Закупка ингредиентов», «Настройка меню».

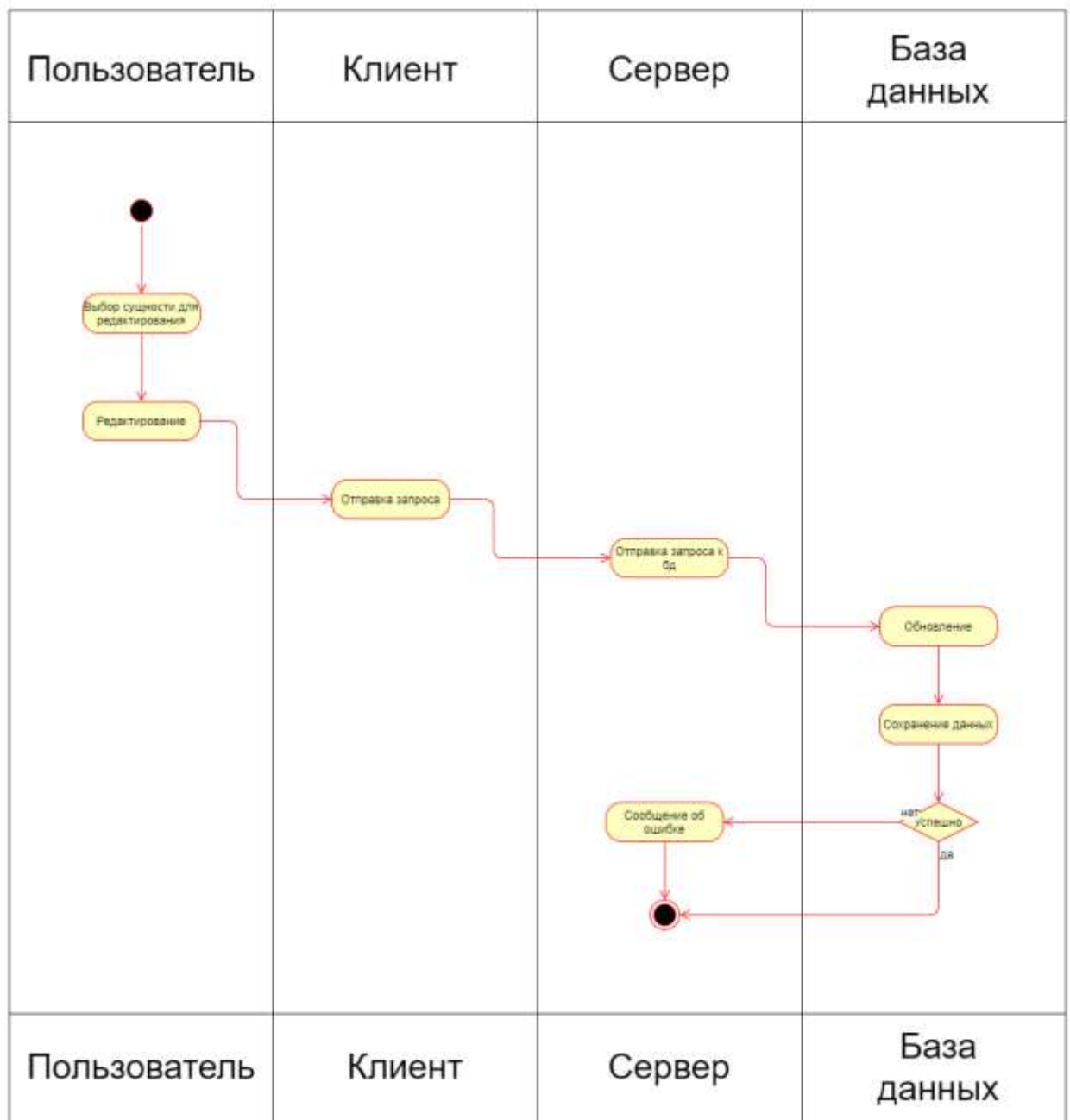


Рис. 19. Диаграмма активности

Диаграмма активности для прецедентов «Просмотр статистики», «Просмотр ингредиентов» и «Просмотр сообщений о готовых заказах».

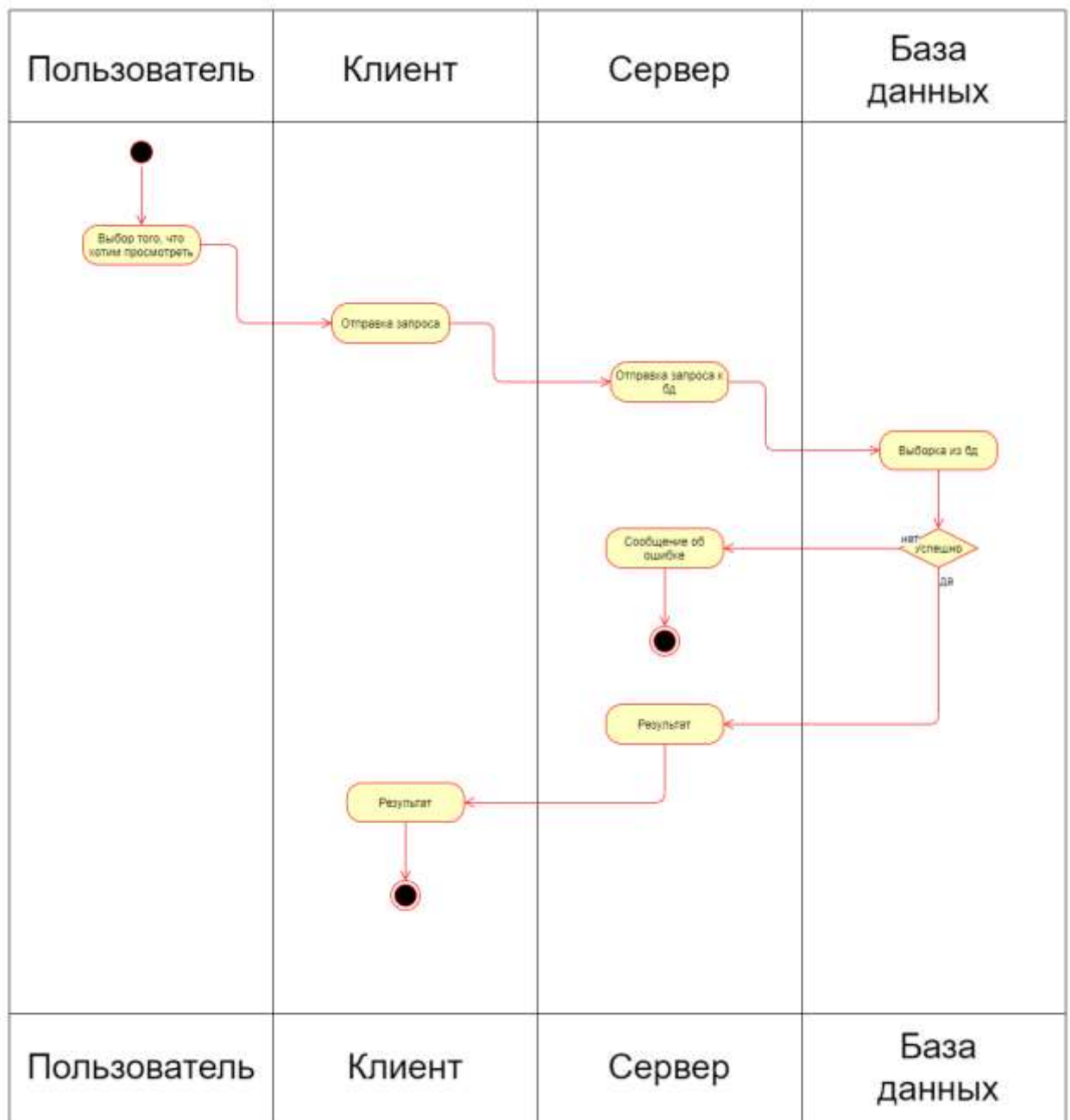


Рис. 20. Диаграмма активности

## **2. Реализация приложения**

### **Анализ средств реализации**

Учитывая требования, которые указаны в Настоящем Техническом Задании, система будет построена на следующих технологиях:

- В качестве API сервера был выбран .NET Core [1] – популярная и прогрессирующая платформа для подобного рода проектов, основанная на базе языка C#.
- В качестве СУБД – MSSQL [3], так как это классическая связка .NET + MSSQL.
- Entity Framework [2] – самый популярный фреймворк для работы с базой данных в .NET.
- В качестве клиента будет выступать SPA (Single Page Application) приложение на Angular [4], так как это очень удобный фреймворк на базе языка TypeScript.

### **Архитектура системы**

Приложение будет построено по клиент-серверной архитектуре, с использованием реляционной базы данных.

В качестве клиента используется SPA приложение на Angular, так как генерация одной страницы по тем JSON-данным, которые пришли от сервера, гораздо эффективнее того, если бы сервер возвращал отдельные страницы. Таким образом, мы снимаем нагрузку с сервера, а также это позволяет при необходимости подключать разные API сервисы, причем написаны они могут быть на разных технологиях. К тому же, само слово Angular уже означает больше, чем просто фреймворк. Angular – это front-end фреймворк со своей экосистемой и своими паттернами. Самое популярное – это дробление всего на мелкие компоненты, которые

взаимодействуют друг с другом, а также подход реактивного программирования.

В качестве API сервера будет выступать ASP.NET Core, построенный по REST-архитектуре, для взаимодействия с клиентом.

Также в системе используется внедрение зависимостей, то есть Dependency Injection (DI).

В качестве реляционной базы данных выступает Microsoft SQL (MSSQL).

### **Проектирование системы**

Проектирование системы включает в себя:

- Проектирование диаграммы классов
- Проектирование диаграммы объектов
- Проектирование схемы базы данных
- Проектирование диаграммы развертывания

### **Диаграмма классов**



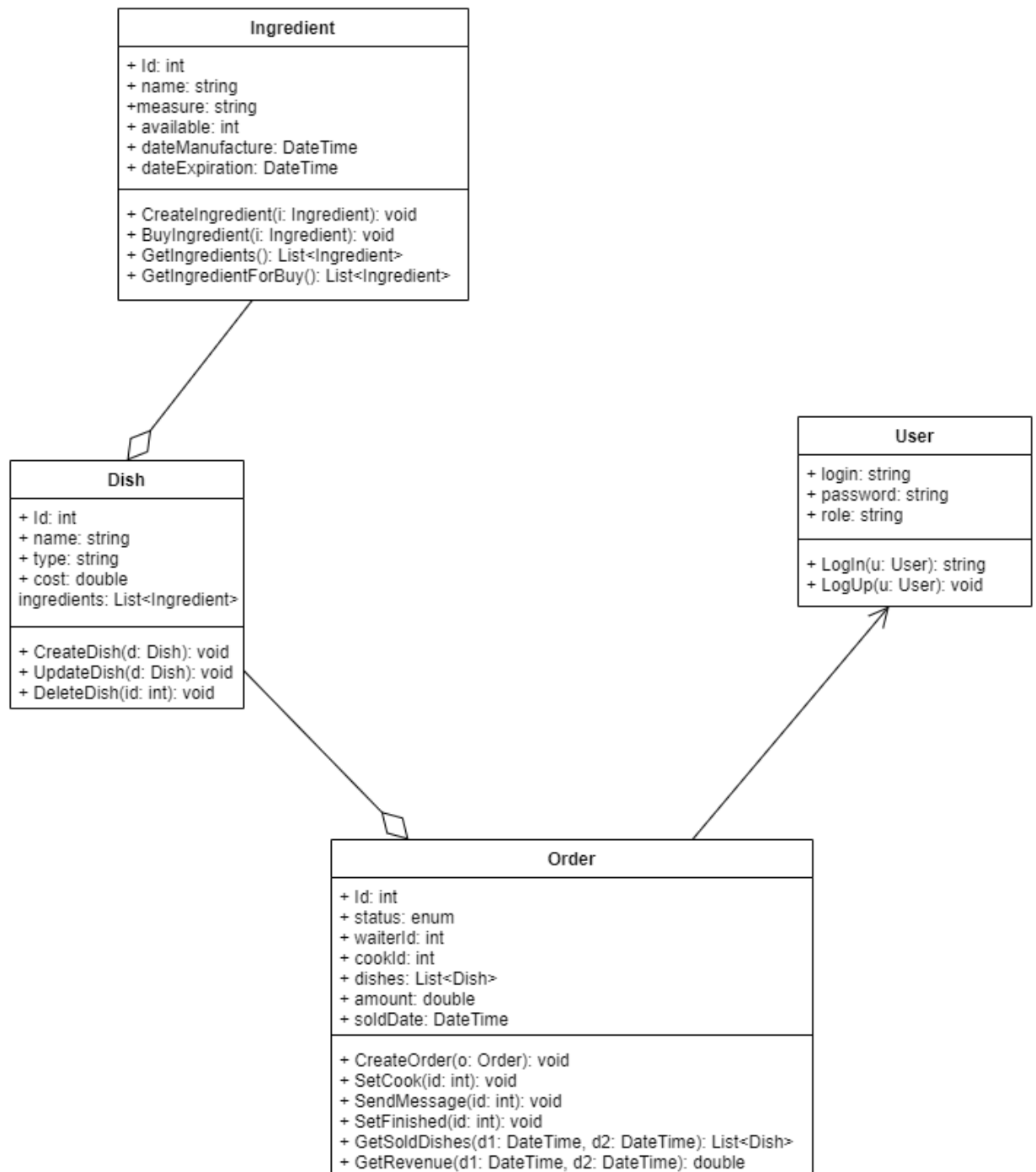


Рис. 21. Диаграмма классов

## Диаграмма объектов

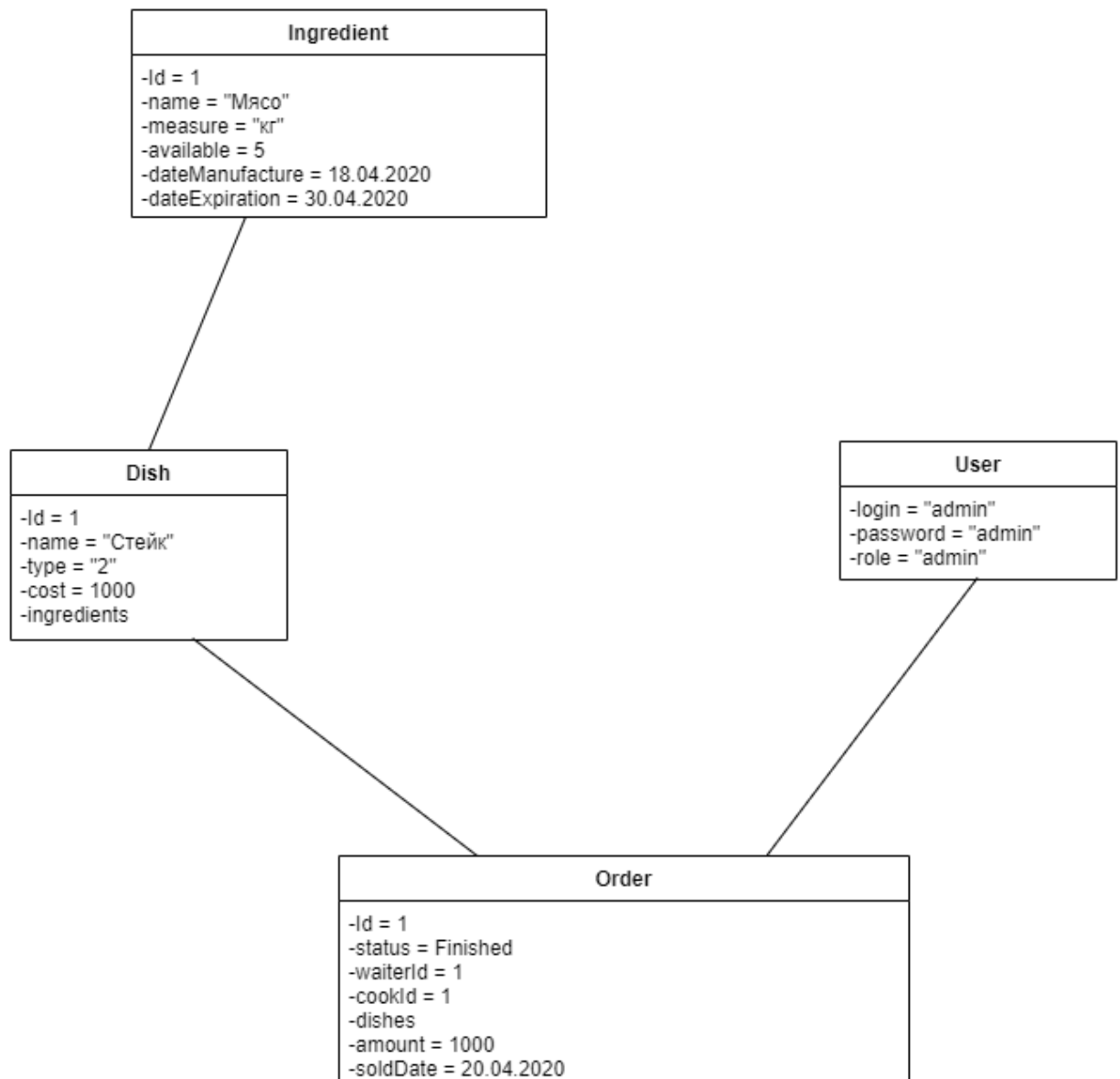


Рис. 22. Диаграмма объектов

## Схема базы данных

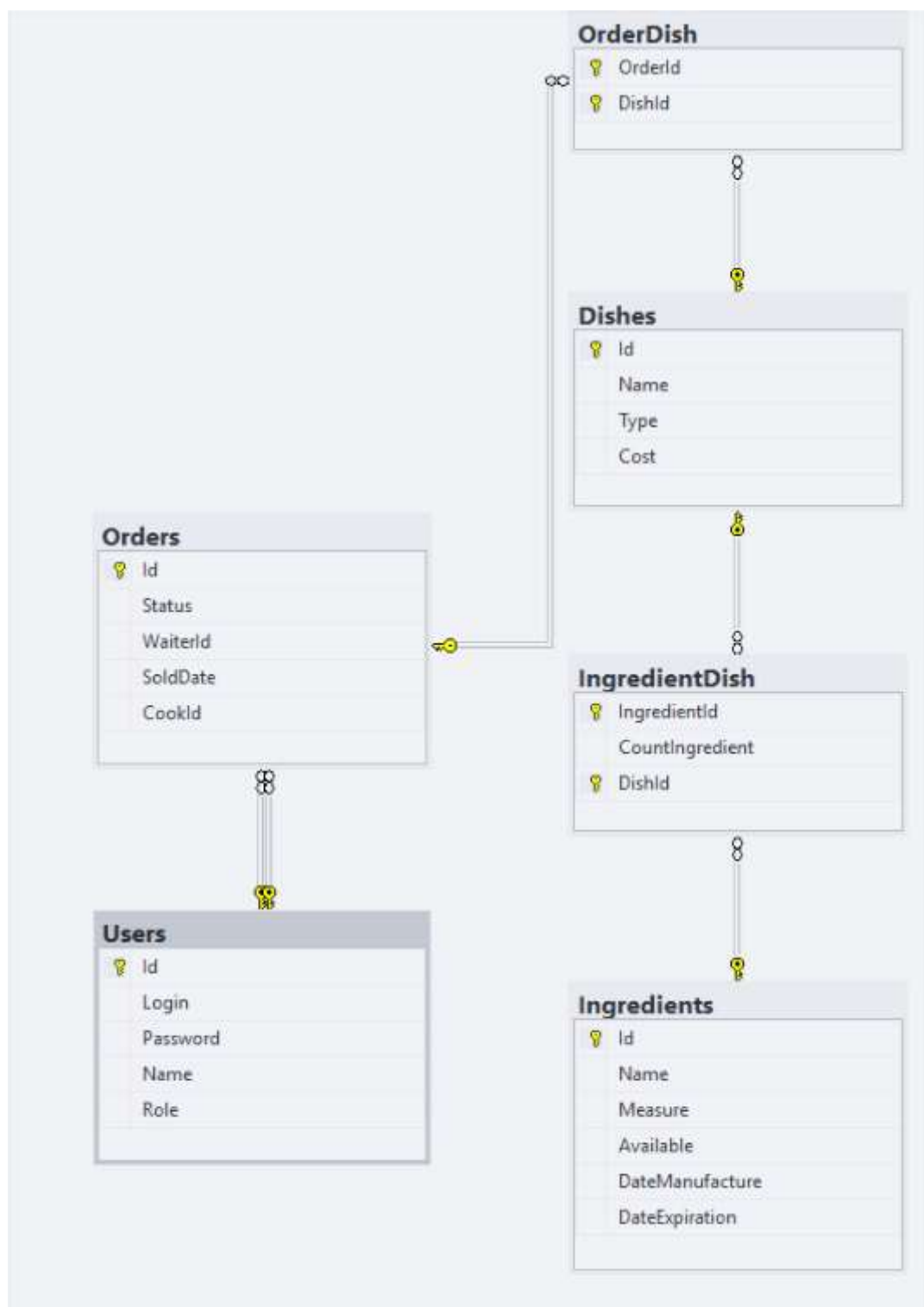


Рис. 23. Диаграмма базы данных

## Диаграмма развертывания

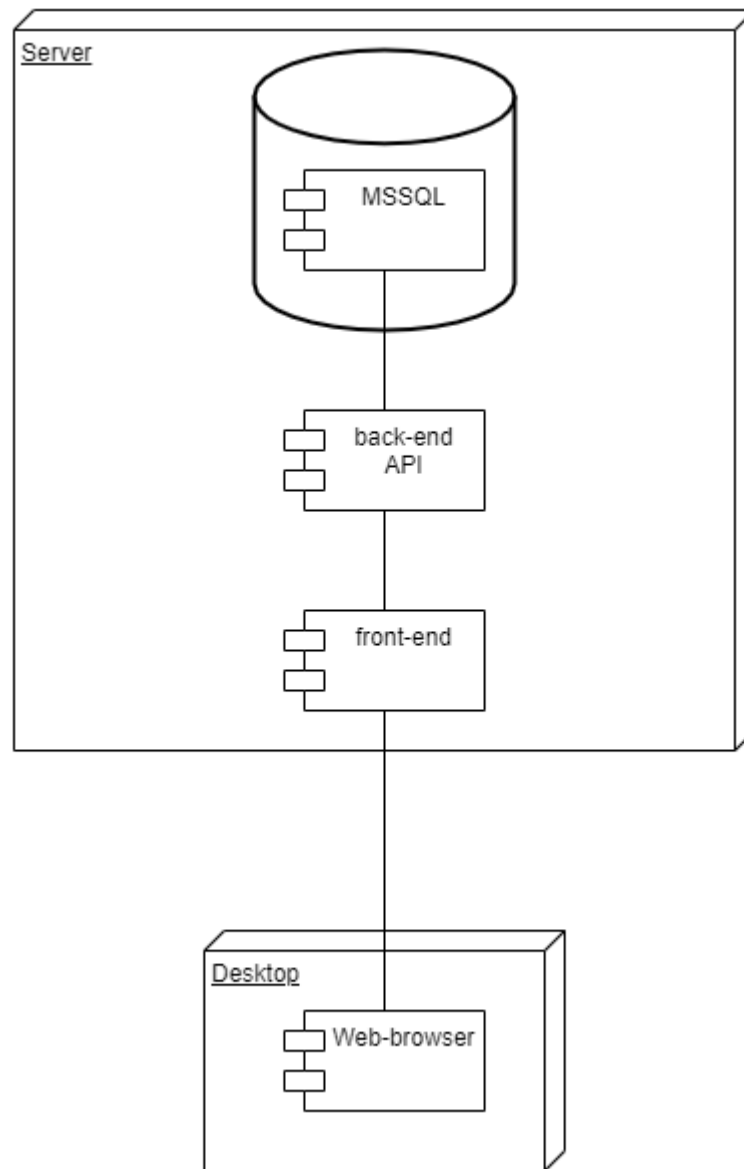


Рис. 24. Диаграмма развертывания

## Серверная часть

Основные контроллеры серверной части:

- AccountController – контроллер отвечает за авторизацию и регистрацию пользователей.
- DishesController – контроллер отвечает за взаимодействие с блюдами и меню.
- IngredientsController – контроллер отвечает за взаимодействие с ингредиентами.
- OrdersController – контроллер отвечает за взаимодействие с заказами.

Авторизация происходит JWT (JSON Web Token) методом. Пользователь обращается на сервер, сервер генерирует уникальный токен доступа и возвращает его пользователю, а затем клиент отправляет запросы на сервер с этим токеном, поэтому он является авторизованным.

Скриншоты API из Swagger отображены ниже.

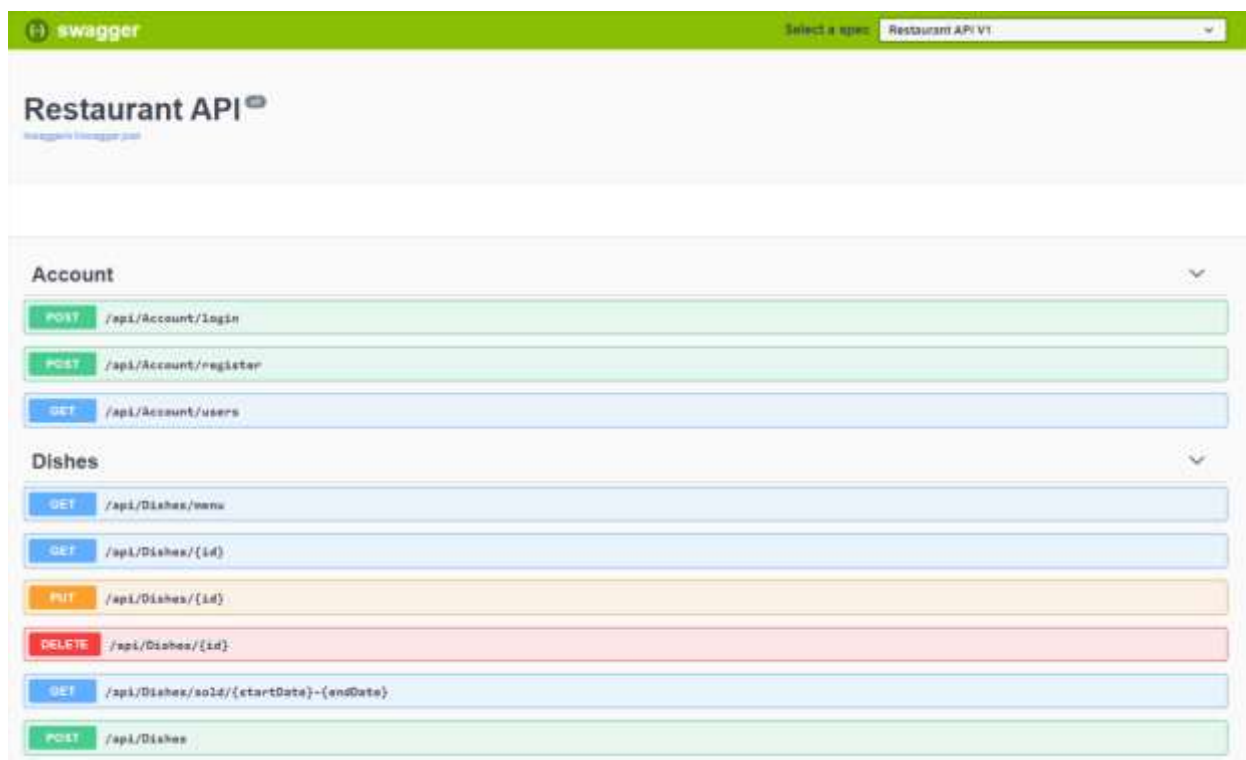


Рис. 25. Swagger

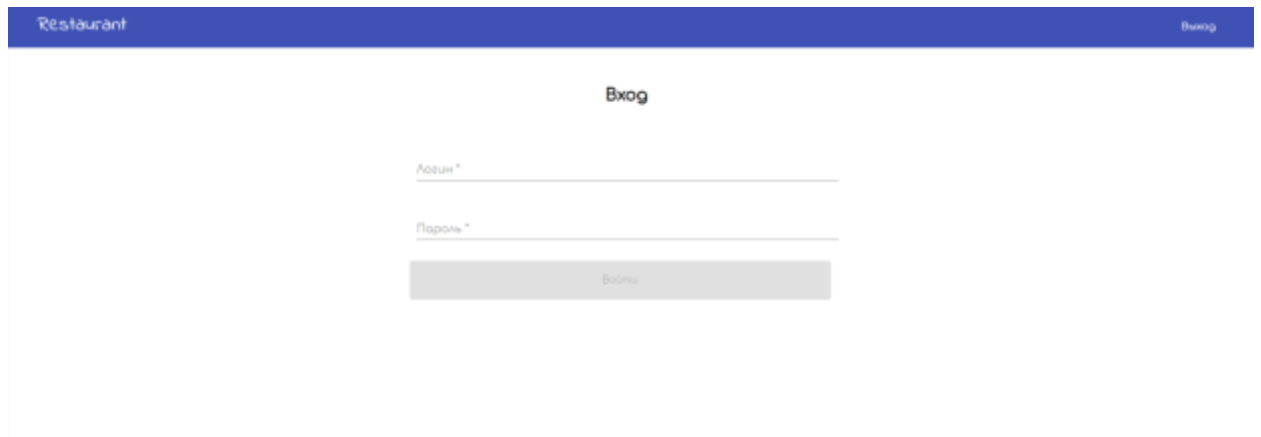
Ingredients		▼
GET	/api/Ingredients	
POST	/api/Ingredients	
GET	/api/Ingredients/low	
GET	/api/Ingredients/used/{startDate}-{endDate}	
GET	/api/Ingredients/{id}	
PUT	/api/Ingredients/{id}	
DELETE	/api/Ingredients/{id}	
Orders		▼
GET	/api/Orders/free	
GET	/api/Orders/in-process	
GET	/api/Orders/{id}	
PUT	/api/Orders/{id}	
DELETE	/api/Orders/{id}	
PUT	/api/Orders/{id}/accept	
PUT	/api/Orders/{id}/ready	
POST	/api/Orders	
GET	/api/Orders/revenue/{startDate}-{endDate}	
GET	/api/Orders/message	

Рис. 26. Swagger

## Клиентская часть

Клиентская часть представляет собой SPA (Single Page Application) на Angular. Для стилизации приложения использовалась библиотека Angular Material [5].

Страница входа пользователя выглядит следующим образом.



The screenshot shows a web application interface for a restaurant. At the top, there is a dark blue header bar with the word "Restaurant" on the left and a "Вход" (Login) button on the right. The main content area is white and contains a centered login form. The form has a title "Вход" (Login) at the top. Below the title, there are two input fields: "Логин \*" (Login) and "Пароль \*" (Password). Below these fields is a "Войти" (Login) button. The form is enclosed in a light gray border.

Рис. 27. Вход

Главная страница администратора выглядит следующим образом.



The screenshot shows the administrator's main page. It features a dark blue header bar with "Restaurant" on the left and "Админ" (Admin) and "Вход" (Login) buttons on the right. The main content area is white and contains a sidebar on the left with a list of menu items: "Регистрация", "Показать все заведения", "Заведения для заказа", "Меню", "Поданные блюда", and "Выход". The sidebar is enclosed in a light gray border.

Рис. 28. Главная страница админа

Главная страница официанта выглядит следующим образом.



The screenshot shows the waiter's main page. It features a dark blue header bar with "Restaurant" on the left and "Официант" (Waiter) and "Вход" (Login) buttons on the right. The main content area is white and contains a sidebar on the left with a list of menu items: "Новый заказ" and "Сообщения". The sidebar is enclosed in a light gray border.

Рис. 29. Главная страница официанта

Главная страница провизора выглядит следующим образом.



Рис. 30. Главная страница провизора

Главная страница повара выглядит следующим образом.



Рис. 31. Главная страница повара

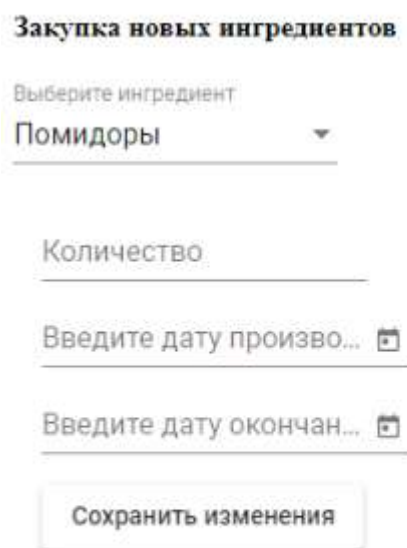
Компонент добавления ингредиентов выглядит следующим образом.

Название ингредиента
Единица измерения и...
Количество
Введите дату произво... 📅
Введите дату окончан... 📅
Добавить

Рис. 32. Добавление ингредиентов



Компонент закупки ингредиентов выглядит следующим образом.



The screenshot shows a web form titled "Закупка новых ингредиентов" (Purchase new ingredients). It contains the following elements:

- A dropdown menu labeled "Выберите ингредиент" (Select ingredient) with "Помидоры" (Tomatoes) selected.
- A text input field labeled "Количество" (Quantity).
- A date input field labeled "Введите дату произво..." (Enter production date) with a calendar icon.
- A date input field labeled "Введите дату окончан..." (Enter completion date) with a calendar icon.
- A button labeled "Сохранить изменения" (Save changes).

Рис. 33. Закупка ингредиентов

Компонент регистрации пользователей выглядит следующим образом.

**Регистрация**

Введите имя \*

Введите логин \*

Введите пароль \*

Выберите роль \*

Зарегистрировать

Рис. 34. Регистрация

Компонент просмотра ингредиентов выглядит следующим образом.

Таблица ингредиентов					
Id	Имя	Ед. измерения	Кол-во единиц	Дата производства	Конец срока годности
1	Мука	кг	5	6/8/20	9/10/20
2	Помидоры	кг	10	6/8/20	10/11/20
3	Яйца	шт	50	6/8/20	8/7/20
4	Лук	кг	5	6/8/20	9/10/20
5	Свинина	кг	64	6/8/20	9/10/20
6	Куриное филе	кг	42	6/8/20	6/1/20
7	Картошка	кг	25	6/8/20	1/1/20
8	Пиво	л	30	6/8/20	1/1/20
9	Рыба	кг	35	6/8/20	8/25/20
10	Рис	кг	12	6/8/20	11/3/20

Рис. 35. Ингредиенты

Компонент просмотра ингредиентов для закупки выглядит следующим образом.

Таблица ингредиентов			
Id	Имя	Ед. измерения	Кол-во единиц
1	Мука	кг	5
2	Помидоры	кг	10
4	Лук	кг	5
6	Куриное филе	кг	42
7	Картошка	кг	25
8	Пиво	л	30

Рис. 36. Ингредиенты для закупки

Компонент создания заказа выглядит следующим образом.

**Составление заказа**

Выберите блюдо

Стейк

Стейк

Оформить заказ

Рис. 37. Составление заказа

Компонент просмотра сообщений готовых заказов выглядит следующим образом.

### Сообщения для официанта

Заказ №2 готов.

Рис. 38. Сообщения о готовности заказов

Компонент заказов для поваров выглядит следующим образом.

The interface is divided into two main sections: "Свободные заказы" (Free orders) and "Текущие заказы" (Current orders). Under "Свободные заказы", there is a card for "Заказ №1" with the text "Борщ, Стейк" and a "Взять" (Take) button. Under "Текущие заказы", there is a card for "Заказ №2" with the text "Борщ, Стейк" and a "Приготовить заказ" (Prepare order) button.

Рис. 39. Страница заказов

Компоненты статистики выглядят следующим образом.

The background interface shows date input fields: "Введите дату начала" with "6/1/2020" and "Введите дату конца" with "6/12/2020", along with a "Узнать" (Know) button. A white notification overlay is present in the center, titled "Уведомление от сайта localhost" (Notification from the localhost site), displaying "Выручка = 2060" (Revenue = 2060) and a "Закрыть" (Close) button.

Рис. 40. Получение выручки

Введите дату начала  
6/1/2020

Введите дату конца  
6/12/2020

Выбрать проданные блюда

Борщ : 2

Стейк : 2

Рис. 41. Статистика проданных блюд

Компонент просмотра меню выглядит следующим образом.

#	Название	Цена	Количество заказов		
1	Борщ	230	0	Редактировать	Удалить
2	Стейк	800	63	Редактировать	Удалить
3	Жареное филе	380	14	Редактировать	Удалить
4	Шаурма	130	4	Редактировать	Удалить
5	Пиво	97	30	Редактировать	Удалить
6	Бургер	260	8	Редактировать	Удалить
Добавить					

Рис. 42. Меню

Компонент редактирования блюда в меню выглядит так.

Название

Стейк

Тип

2

Цена

800

ID	Название	Количество ингредиентов	Единица	
5	Свинина	1	кг	удалить ингредиент из блюда

Сохранить

Рис. 43. Редактирование блюда

Компонент добавления нового блюда в меню выглядит следующим образом.

Добавить

Название

Тип

Цена

ID	Название	Количество ингредиентов	Единица			
ID	Название	Количество на складе	Единица	Дата изготовления	Дата истечения срока годности	
1	Мука	1	кг	6/8/20	9/10/20	Добавить ингредиент
2	Помидоры	8	кг	6/8/20	10/11/20	Добавить ингредиент
3	Яйца	50	шт	6/8/20	8/7/20	Добавить ингредиент
4	Лук	5	кг	6/8/20	9/10/20	Добавить ингредиент
5	Свинина	60	кг	6/8/20	9/10/20	Добавить ингредиент
6	Куриное филе	42	кг	6/8/20	6/1/20	Добавить ингредиент
7	Картошка	25	кг	6/8/20	1/1/20	Добавить ингредиент
8	Лаваш	30	л	6/8/20	1/1/20	Добавить ингредиент
9	Рыба	35	кг	6/8/20	8/25/20	Добавить ингредиент
10	Рис	12	кг	6/8/20	11/3/20	Добавить ингредиент

Рис. 44. Добавление блюда

# Тестирование

## Дымовое тестирование

Для данного типа тестирования необходимо было проверить работоспособность приложения на следующих основных сценариях:

- войти в систему;
- выйти из системы;
- закупка ингредиентов;
- внесение ингредиентов в систему;
- получение сообщения о готовности заказа;
- составление заказа;
- взятие заказа на себя;
- приготовить заказ;
- регистрация в системе других пользователей;
- составление и настройка меню;
- получение информации о проданных блюдах;
- получение информации о выручке;
- получение информации о текущих ингредиентах;
- получение информации о необходимости закупки ингредиентов.

Дымовое тестирование проводилось ручным способом, на заранее развернутом сайте.

Результаты, полученные в ходе тестирования, представлены в Таблице 1.

Таб. 1. Результаты дымового тестирования

Сценарий	Результат
Войти в систему	Пройден
Выйти из системы	Пройден
Закупка ингредиентов	Пройден
Внесение ингредиентов в	Пройден



систему	
Получение сообщения о готовности заказа	Пройден
Составление заказа	Пройден
Взятие заказа на себя	Пройден
Приготовить заказ	Пройден
Регистрация в системе других пользователей	Пройден
Составление и настройка меню	Пройден
Получение информации о проданных блюдах	Пройден
Получение информации о выручке	Пройден
Получение информации о текущих ингредиентах	Пройден
Получение информации о необходимости закупки ингредиентов	Пройден

### **Функциональное тестирование**

Для данного типа тестирования необходимо было проверить работоспособность приложения на следующих основных сценариях:

- войти в систему;
- выйти из системы;
- закупка ингредиентов;
- внесение ингредиентов в систему;
- получение сообщения о готовности заказа;

- составление заказа;
- взятие заказа на себя;
- приготовить заказ;
- регистрация в системе других пользователей;
- составление и настройка меню;
- получение информации о проданных блюдах;
- получение информации о выручке;
- получение информации о текущих ингредиентах;
- получение информации о необходимости закупки ингредиентов.

Результаты, полученные в ходе тестирования представлены в Таблице 2.

Таб. 2. Результаты функционального тестирования

Сценарий	Результат
Войти в систему	Пройден
Выйти из системы	Пройден
Закупка ингредиентов	Пройден
Внесение ингредиентов в систему	Пройден
Получение сообщения о готовности заказа	Пройден
Составление заказа	Пройден
Взятие заказа на себя	Пройден
Приготовить заказ	Пройден
Регистрация в системе других пользователей	Пройден
Составление и настройка меню	Пройден
Получение информации о проданных блюдах	Пройден

Получение информации о выручке	Пройден
Получение информации о текущих ингредиентах	Пройден
Получение информации о необходимости закупки ингредиентов	Пройден

### Юзабилити тесты

Для проведения юзабилити тестирования было случайно отобрано 3 человека, не пользовавшиеся заранее приложением. Для данного тестирования необходимо проверить следующие основные сценарии взаимодействия пользователя с приложением:

- войти в систему;
- выйти из системы;
- закупка ингредиентов;
- внесение ингредиентов в систему;
- получение сообщения о готовности заказа;
- составление заказа;
- взятие заказа на себя;
- приготовить заказ;
- регистрация в системе других пользователей;
- составление и настройка меню;
- получение информации о проданных блюдах;
- получение информации о выручке;
- получение информации о текущих ингредиентах;
- получение информации о необходимости закупки ингредиентов.

Таб. 3. Результаты

Сценарий	Пользователь 1	Пользователь 2	Пользователь 3
Войти в систему	Пройден	Пройден	Пройден
Выйти из системы	Пройден	Пройден	Пройден
Закупка ингредиентов	Пройден	Пройден	Пройден
Внесение ингредиентов в систему	Пройден	Пройден	Пройден
Получение сообщения о готовности заказа	Пройден	Пройден	Пройден
Составление заказа	Пройден	Пройден	Пройден
Взятие заказа на себя	Пройден	Пройден	Пройден
Приготовить заказ	Пройден	Пройден	Пройден
Регистрация в системе других пользователей	Пройден	Пройден	Пройден
Составление и настройка меню	Пройден	Пройден	Пройден
Получение информации о проданных блюдах	Пройден	Пройден	Пройден

Получение информации о выручке	Пройден	Пройден	Пройден
Получение информации о текущих ингредиентах	Пройден	Пройден	Пройден
Получение информации о необходимости закупки ингредиентов	Пройден	Пройден	Пройден

## Аналитика

Так как система рассчитана на бизнес и автоматизацию внутренних процессов, то анализировать действия сотрудников ресторана нет смысла, а получение статистики для администратора ресторана в системе имеется, то был создан маркетинговый сайт для продвижения данного ПО.

Для маркетингового сайта была создана основная воронка заинтересованности данной системы.



Рис. 45. Воронка

Скриншот данной воронки в Яндекс Метрике.

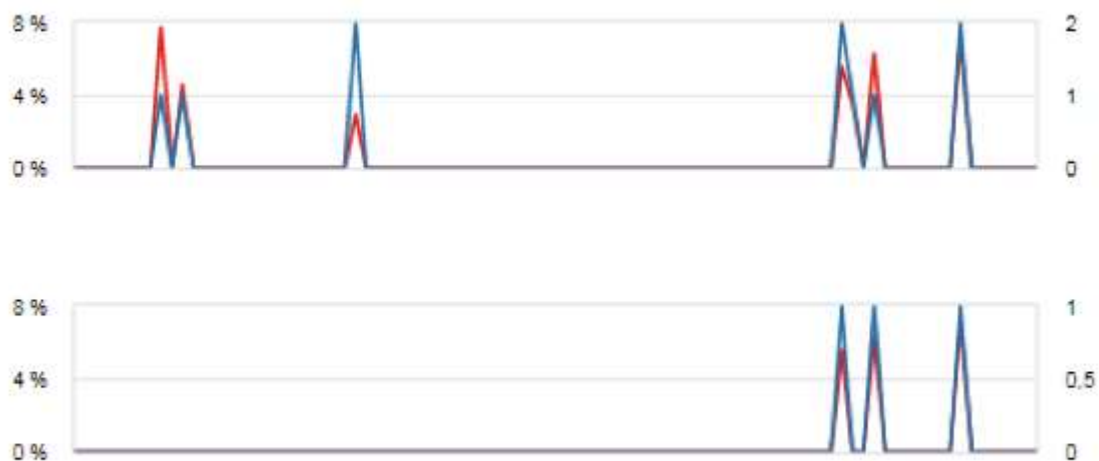


Рис. 46. Скриншот из Яндекс.Метрики

Вывод по данной воронке.

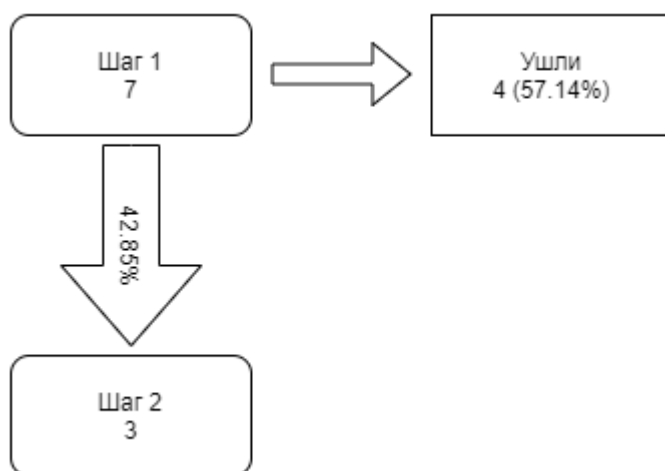


Рис. 47. Вывод

## **Заключение**

В результате работы было реализована система, которая автоматизирует работу внутри ресторана между его сотрудниками. Были выполнены следующие задачи:

1. Была разработана база данных в MSSQL.
2. Был реализован сервер на .NET.
3. Была реализована клиентская часть на Angular.



## Список использованных источников

1. Создание веб-API с помощью ASP.NET Core | Microsoft Docs – 2020. [Электронный ресурс]. Режим доступа: <https://docs.microsoft.com/ru-ru/aspnet/core/web-api/?view=aspnetcore-2.2> (Дата обращения: 13.03.2020)
2. Документация по Entity Framework | Microsoft Docs – 2009 - 2020. [Электронный ресурс]. Режим доступа: <https://docs.microsoft.com/ru-ru/ef/> (Дата обращения: 15.03.2020)
3. SQL Server 2019 | Microsoft – 2009 - 2020. [Электронный ресурс]. Режим доступа: <https://www.microsoft.com/ru-ru/sql-server/sql-server-2019> (Дата обращения: 13.03.2020)
4. Angular – 2020. [Электронный ресурс]. Режим доступа: <https://angular.io/> (Дата обращения: 03.04.2020)
5. Angular Material UI component library – 2009 - 2020. [Электронный ресурс]. Режим доступа: <https://material.angular.io/> (Дата обращения: 03.04.2020)