TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа информационных технологий и робототехники
Направление подготовки 09.03.02 «Информационные системы м технологии»
Отделение школы (НОЦ) информационных технологий

БАКАЛАВРСКАЯ РАБОТА

Тема работы
ИНФОРМАЦИОННАЯ СИСТЕМА ДЛЯ РЕСТОРАНОВ БЫСТРОГО ПИТАНИЯ (СЕРВЕРНАЯ ЧАСТЬ)

УДК 004.455.2:640.433.045

Студент

Группа	ФИО	Подпись	Дата
8И5А	Миртов Сергей Павлович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Цапко И.В.	К.т.н., доцент		

КОНСУЛЬТАНТЫ:

По разделу «Концепция стартап-проекта»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Эксперт	Черний А. В.	—		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Немцова О. А			

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Цапко И.В.	к.т.н.		

Томск – 2019 г.

РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Код результатов	Результат обучения (выпускник должен быть готов)
Профессиональные и общепринятые компетенции	
Р1	Применять базовые и специальные естественнонаучные и математические знания для комплексной инженерной деятельности по созданию, внедрению и эксплуатации геоинформационных систем и технологий, а также информационных систем и технологий в бизнесе.
Р2	Применять базовые и специальные знания в области современных информационных технологий для решения инженерных задач.
Р3	Ставить и решать задачи комплексного анализа, связанные с созданием геоинформационных систем и технологий, информационных систем в бизнесе, с использованием базовых и специальных знаний, современных аналитических методов и моделей.
Р4	Выполнять комплексные инженерные проекты по созданию информационных систем и технологий, а также средств их реализации (информационных, методических, математических, алгоритмических, технических и программных).
Р5	Проводить теоретические и экспериментальные исследования, включающие поиск и изучение необходимой научно-технической информации, математическое моделирование, проведение эксперимента, анализ и интерпретация полученных данных, в области создания геоинформационных систем и технологий, а также информационных систем и технологий в бизнесе.
Р6	Внедрять, эксплуатировать и обслуживать современные геоинформационные системы и технологии, информационные системы и технологии в бизнесе, обеспечивать их высокую эффективность, соблюдать правила охраны здоровья, безопасность труда, выполнять требования по защите окружающей среды.
Универсальные (общекультурные) компетенции	
Р7	Использовать базовые и специальные знания в области проектного менеджмента для ведения комплексной инженерной деятельности.
Р8	Осуществлять коммуникации в профессиональной среде и в обществе в целом. Владеть иностранным языком (углублённый английский язык), позволяющем работать в иноязычной среде, разрабатывать документацию, презентовать и защищать результаты комплексной инженерной деятельности.
Р9	Эффективно работать индивидуально и в качестве члена команды, состоящей из специалистов различных направлений и квалификаций.
Р10	Демонстрировать личную ответственность за результаты работы и готовность следовать профессиональной этике и нормам ведения комплексной инженерной деятельности.
Р11	Демонстрировать знания правовых, социальных, экологических и культурных аспектов комплексной инженерной деятельности, а также готовность к достижению должного уровня физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности.

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа информационных технологий и робототехники
Направление подготовки 09.03.02 «Информационные системы м технологии»
Отделение школы (НОЦ) информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП
_____ _____ И.В. Цапко
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

бакалаврской работы

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	ФИО
8И5А	Миртову Сергею Павловичу

Тема работы:

Разработка системы прогнозирования параметров движения	
Утверждена приказом директора (дата, номер)	№3654/с от 13.05.2019 г.

Срок сдачи студентом выполненной работы:	4.06.2019

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Разработка серверной части информационной системы для ресторанов быстрого питания
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных*	– Анализ предметной области; – Проектирование системы; – Разработка системы; – Разработка программной документации.

разделов, подлежащих разработке; заключение по работе).	
Перечень графического материала *(с точным указанием обязательных чертежей)*	Презентация в формате *.ppt

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Концепция стартап-проекта	Черний А. В.
Социальная ответственность	Немцова О. А.

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Заключение

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	13.05.2019

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР	Цапко И. В.	К.т.н., доцент		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И5А	Миртов Сергей Павлович		

TOMSK POLYTECHNIC UNIVERSITY ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский Томский политехнический университет» (ТПУ)

Школа информационных технологий и робототехники
Направление подготовки 09.03.02 «Информационные системы м технологии»
Отделение школы (НОЦ) информационных технологий
Период выполнения – весенний семестр 2019 учебного года

Форма представления работы:

бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	04.06.2019

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
20.05.2019	Основная часть	75
21.05.2019	Концепция стартап-проекта	15
26.04.2019	Социальная ответственность	10

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Кочегурова Е. А.	к.т.н.		

СОГЛАСОВАНО:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Цапко И. В.	к.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«КОНЦЕПЦИЯ СТАРТАП-ПРОЕКТА»

Студенту:

Группа	ФИО
8И5А	**Миртову Сергею Павловичу**

Школа	ИШИТР	**Отделение (НОЦ)**	информационных технологий
Уровень образования	Бакалавриат	**Направление/специальность**	Информационные системы и технологии

Перечень вопросов, подлежащих разработке:	
Проблема конечного потребителя, которую решает продукт, который создается в результате выполнения НИОКР (функциональное назначение, основные потребительские качества)	Определены проблемы, которые должен решать разрабатываемый продукт. Выявлены уникальные торговые предложения.
Способы защиты интеллектуальной собственности	Определены способы защиты интеллектуальной собственности и методы распространения программного обеспечения.
Объем и емкость рынка	Рассчитан объем и емкость рынка, проведен анализ.
Современное состояние и перспективы отрасли, к которой принадлежит представленный в ВКР продукт	Выявлены приложения-конкуренты, рассмотрены текущие тренды области.
Себестоимость продукта	Рассчитана себестоимость продукта.
Конкурентные преимущества создаваемого продукта	Выявлены конкурентные преимущества продукта.
Сравнение технико-экономических характеристик продукта с отечественными и мировыми аналогами	Проведен сравнительный анализ конкурентов и разрабатываемого продукта.
Целевые сегменты потребителей создаваемого продукта	Определены целевые потребительские сегменты.
Бизнес-модель проекта	Разработана бизнес-модель по методологии Остервальдера.
Производственный план	Разработан производственный план.
План продаж	Выявлены способы продвижения продукта на рынок.
Перечень графического материала:	
При необходимости представить эскизные графические материалы (например, бизнес-модель)	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант по разделу «Концепция стартап-проекта» (со-руководитель ВКР:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Эксперт	**Черний А. В.**	**—**		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И5А	**Миртов С. П.**		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8И5А	Миртову Сергею Павловичу

Школа	ИШИТР	Отделение (НОЦ)	Информационных технологий
Уровень образования	Бакалавриат	Направление/специальность	Информационные системы и технологии

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	*Разработать серверное программное приложение, для взаимодействия с базой данных, клиентскими мобильными приложениями и веб приложением.*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	Требования к организации и оборудованию рабочих мест с ПЭВМ.
2. Производственная безопасность: 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	Вредные: 1. недостаточная освещенность рабочей зоны; 2. умственное перенапряжение 3. монотонный режим работы. 4. Отклонение показателей микроклимата в помещении Опасные: 1. опасность поражения электрическим током; 2. опасность возникновения пожара.
3. Экологическая безопасность:	Влияние объекта исследования на окружающую среду; мероприятия по защите окружающей среды.
4. Безопасность в чрезвычайных ситуациях:	Основные и типичные чрезвычайные ситуации в офисном помещении; установка общих правил поведения и рекомендаций во время ЧС.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Ассистент	Немцова О. А.			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8И5А	Миртов Сергей Павлович		

РЕФЕРАТ

Выпускная квалификационная работа содержит 76 страниц, 11 рисунков, 12 таблиц, 36 источников.

Ключевые слова: архитектура, сервер, разработка, клиент, приложение, architecture, server, development, client, application.

Объектом исследования является архитектура серверной части информационной системы для ресторанов быстрого питания.

Цель работы — разработка серверного компонента информационной системы.

В процессе исследования проводился анализ различных технологий, используемых при построении архитектура серверов.

В результате исследования была спроектирована и разработана архитектура серверного компонента информационной системы для ресторанов быстрого питания.

В будущем планируется внедрить и испытать систему в сети ресторанов быстрого питания.

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

API — Application Programming Interface

CSS — Cascading Style Sheets

ИС — Информационная Система

БП — Бизнес-процесс

POCO — Plain Old C# Object

DTO — Data Transfer Object

JSON — JavaScript Object Notation

JWT — Json Web Token

ORM — Object-Relational Mapping

REST — Representational State Transfer

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Во время обучения в Научно-Исследовательском Томском Политехническом Университете студенты проходят ежегодные медицинские осмотры. У большинства студентов состояние здоровья ухудшается от первого к четвертому курсу. Одной из основных причин ухудшения здоровья студентов являет нарушение режима питания.

Нарушение режима питания обуславливается многими факторами: отсутствие перерыва на принятие пищи, короткими перерывами между занятиями, необходимость перемещения между корпусами во время перерывов.

Рядом с корпусами ТПУ располагается большое количество заведений, основным видом деятельности которых является розничная продажа продуктов питания. К таким заведениям можно отнести сети ресторанов быстрого питания. Данные заведения позволяют студентам произвести прием пищи во время перерывов. Из-за длинных очередей многие студенты не успевают совершить заказ и получить его. В свою очередь рестораны быстрого питания вынуждены смириться с несбалансированной нагрузкой в течении дня, потерей прямых и потенциальных клиентов и потерей выручки.

Для уменьшения очередей возможно использование сервисов заказа еды онлайн, однако данные сервисы, в большинстве своем подразумевают доставку еды. Многие сервисы не производят доставку в здания корпусов университета.

Возможным решением вышеперечисленных проблем является разработка информационной системы, которая позволила бы выполнять предзаказ продуктов питания онлайн с последующим самовывозом. Такая система сбалансирует нагрузку на рестораны быстрого питания, уменьшит время на обслуживание клиента, снизит очередь, что в свою очередь позволит студентам, преподавателям и другим лицам выполнять приемы пищи регулярно.

1. АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

Целью данной исследовательской работы является разработка серверного компонента ИС с последующей реализацией. Перед началом разработки любой ИС необходимо выявить требования к ней.

После выявления требований необходимо спроектировать общую архитектуру информационной системы. Спроектировать начальную схему базы данных, определиться с протоколами взаимодействия между компонентами ИС.

1.1 Общие требования к информационной системе

В разрабатываемой информационной системе существуют три основных роли: администратор сети, работник сети, клиент. Ниже описаны основные функциональные требования к ИС, сгруппированные по ролям.

Администратор сети:

- Авторизоваться в системе.
- Зарегистрировать / удалить работников сети.
- Создать / изменить / удалить точки продаж.
- Создать / изменить / удалить категории товаров.
- Создать / изменить / удалить товары.
- Просмотреть информацию по текущим заказам.
- Изменить статус заказа.
- Просмотреть статистику по продажам для товара / категории товара / категории товара за выбранный промежуток времени.
- Произвести выход из системы.

Работник сети:

- Авторизоваться в системе.
- Просмотреть список заказов для торговой точки, в которой он находится.
- Перевести заказ в статус «в работе» / «готов» / «продан».

– Произвести выход из системы.

Клиент:

– Зарегистрироваться в системе.
– Авторизоваться в системе.
– Просмотреть список категорий товаров / товаров / торговых точек.
– Произвести заказ выбрав необходимые товары и точку продаж.
– Получить номер заказа.
– Просмотреть список своих заказов.
– Получить уведомление о смене статуса заказа.
– Произвести выход из системы.

Для отображения всех функциональных требований системы используется диаграмма вариантов использования на рисунке 1. Каждый вариант использования описывает с точки зрения действующего лица, группу действий в системе, которые приводят к конечному результату [1].

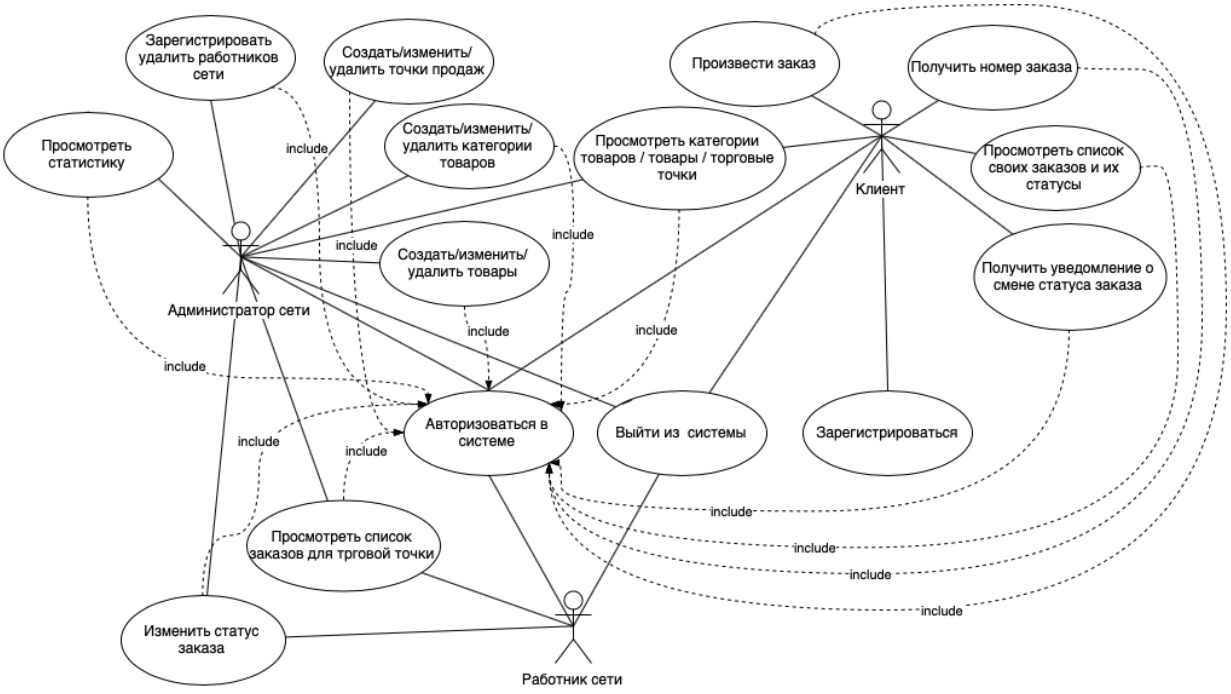

Рисунок 1 — Диаграмма вариантов использования информационной системы

1.2 Анализ конкурентных технических решений

В рамках анализа конкурентных технических решений необходимо определить ближайших конкурентов, подобрать факторы конкурентоспособности, определить оценочную шкалу факторов конкурентоспособности и их важности, составить таблицу «Оценочная карта для сравнения конкурентных технических решений (разработок)», произвести расчеты, построить многоугольник конкурентоспособности (с учетом важности факторов).

В качестве основных конкурентов выбраны: «eMenu», «CashPad», «iiko».

В качестве факторов конкурентоспособности определены следующие параметры:

- стоимость готового решения;
- гибкость (возможность адаптации под условия определенного предприятия);
- удобство пользовательского интерфейса;
- быстродействие системы;
- функциональные возможности.

Факторы были оценены по 10-балльной шкале, важность факторов по 5-балльной шкале.

Результаты проведенного анализа конкурентных технических решений приведены в таблице 1.

Таблица 1 – Оценочная карта для сравнения конкурентных технических решений (разработок)

№ п/п	Конкуренты	Факторы конкурентоспособности					Итоговая оценка
		Цена	Гибкость	Интерфейс	Быстродей-ствие	Возмож-ности	
1	eMenu	7/2,06	8/1,40	7/1,65	5/0,59	8/1,40	7,10
2	CashPad	6/1,76	6/1,06	2/0,47	4/0,47	9/1,58	5,34
3	iiko	3/0,88	9/1,58	9/2,12	10/1,18	10/1,76	7,52

4	Проект	10/2,94	9/1,58	10/2,35	8/0,94	8/1,40	9,21
	b_j	5	3	4	2	3	17
	w_j	0,294	0,176	0,235	0,118	0,176	-

Стоимость готового решения у разрабатываемой системы является самой низкой, поэтому по данному параметру получена максимальная оценка. Что касается гибкости системы, данный проект подразумевает легкую возможность адаптации под большинство имеющихся систем учета оборота в сетях быстрого питания, а решение «iiko» уже имеет интеграцию в наиболее распространенные системы. Тем не менее, под проприетарные решения необходима будет дополнительная доработка в любом из вариантов. Пользовательский интерфейс рассмотренных систем зачастую является устаревшим в связи с продолжительным сроком разработки, а также создан с расчетом на легкость его создания, а не на удобство пользователя, поэтому по данному параметру, например, «CashPad» получил самую низкую оценку. Быстродействие системы сильно различается у рассмотренных конкурентов, тем не менее, данный параметр является менее важным, в сравнении с остальными и имеет наименьший вес. Касательно имеющихся возможностей, решение «iiko» предоставляет самый широкий спектр функций, в связи, с чем и связана его крайне завышенная цена.

По сумме оценок с учётом их веса, наиболее предпочтительной является система, разрабатываемая в рамках данной работы.

1.3 Общая архитектура информационной системы

Исходя из выше сформулированных требований необходимо составить общую архитектуру системы.

Для взаимодействия клиентов с информационной системой нужен графический интерфейс. Графический интерфейс должен быть простым, приятным, удобным, покрывающим все функциональные требования. Одним из целевых сегментов клиентов системы являются студенты и молодые люди в возрасте от 16 до 40 лет. У большинства людей, составляющих данный сегмент имеется мобильный телефон, на одной из двух операционных системах: IOS, Android. Таким образом приложение, разработанные под данные платформы, является лучшим решением и способно реализовать необходимый графический интерфейс.

Помимо графических интерфейсов современные смартфоны предоставляют возможность взаимодействия с пользователем посредством голосового интерфейса. Одним из сервисов использующих голосовой интерфейс является сервис от компании Яндекс «Яндекс Алиса».[2]

Для использования данного сервиса внутри разрабатываемой ИС необходимо развернуть и настроить сервер, образ которого предоставляется компанией Яндекс.

Для взаимодействия работников торговой сети с информационной системой необходимо устройство с установленным программным модулем, которое будет отвечать следующим требованиям:

- Степень защиты не ниже IP65[3].
- Наличие креплений для возможности монтажа на кухне.
- Возможность питания от сети.
- Наличие автономного питания.
- Наличие доступа в интернет.
- Средство ввода и отображения информации.
- Приемлемая цена.
- Простота установки программного приложения.

Всем выше перечисленным требованиям отвечает такое решение как планшет на операционной системе Android. Примером такого устройства является планшет Lenovo Tab 3[4].

Варианты использования информационной системы со стороны роли администратора сети намного шире остальных ролей. Требования к интерфейсу взаимодействия администратора сети с информационной системной приведены ниже:

- Интуитивно понятный графический интерфейс.
- Возможность доступа к ИС из любой точки мира.
- Возможность доступа к ИС в любое время суток.
- Отсутствие привязки к определенному устройству.
- Отсутствие привязки к определенной операционной системе.

Таким образом, лучшим решением для взаимодействия администратора сети с информационной системой является веб-клиент. Веб клиент работает всех устройствах, на которых есть браузер[5], обеспечивает намного более гибкий интерфейс, не требует обновления на устройстве.

После определения способов взаимодействия всех участников информационной системы с самой системной необходимо спроектировать общую архитектуру информационной системы.

Веб-клиент, два мобильных клиента, сервер «Яндекс Алиса», клиент работника сети должны быть объединены в общую сеть, для взаимодействия с базой данных. Каждый из клиентов имеет возможность взаимодействовать с окружающим миром посредством HTTP протокола[6] и рядом других протоколов, подробнее об этом в разделе «1.3 Определение протоколов взаимодействия компонентов ИС».

Таким образом центральный сервер представляет собой веб-сервер. Общая архитектура информационной системы отображена на рисунке 2.

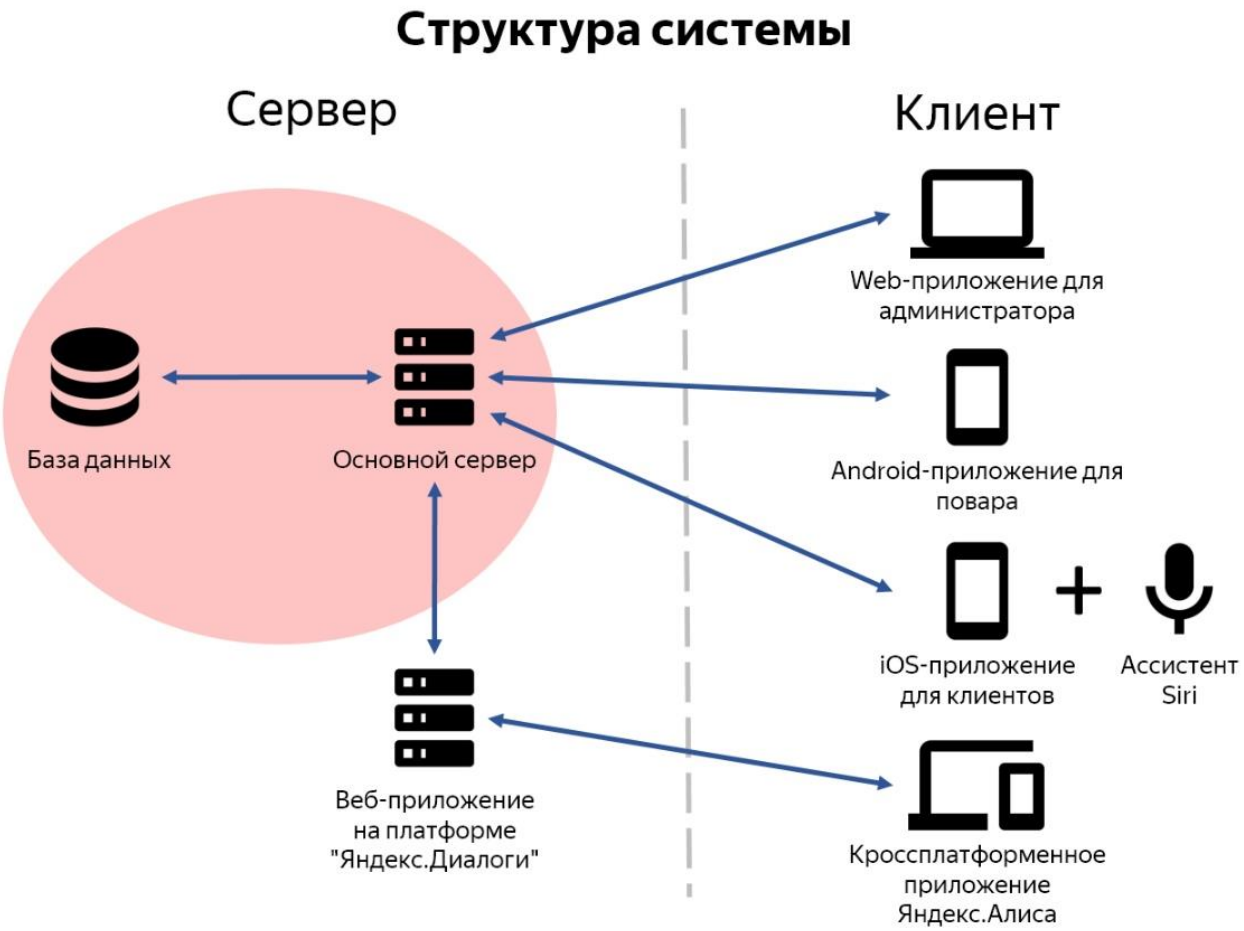

Рисунок 2 — Общая архитектура ИС.

Основной сервер, сервер базы данных, сервер «Яндекс Алиса», сервер веб-приложения находятся в Docker контейнерах[7], объединенных в общую виртуальную сеть, с возможностью последующей мультипликации и балансировки.

1.4 Определение протоколов взаимодействия компонентов ИС

Для того чтобы определиться с протоколами взаимодействия между компонентами ИС необходимо перечислить доступные протоколы. В таблице 2 представлены основные доступные протоколы для каждого модуля ИС.

Таблица 2 — Рассматриваемые протоколы модулей информационной системы:

Модуль ИС	Протоколы	Готовые библиотеки
Android Client	HTTP	OkHttp
	FTP	Apache common ftp
	TCP Socket	Andoid tcp client
	WEB Socket	Socket.IO
IOS Client	HTTP	IQHTTPService
	FTP	Не найден
	TCP Socket	CocoaAsyncSocket
	WEB Socket	Socket.io-client-swift
Web Client	HTTP	AXios.js
	FTP	AXios.js
	WEB Socket	Socket.io.js
Алиса Сервер	HTTP	AXios.js

Готовые библиотеки для различных модулей сильно удешевляют и ускоряют разработку. Клиент-серверная архитектура не является новинкой и при разработке такой архитектуры следует воспользоваться накопившемся опытом.

Таким образом мобильные приложения взаимодействуют с основным сервером посредством протокола HTTP, используя JSON API[8]. Использование JSON формата обусловлено легкой читаемостью, наличием большого количества готовых сериализаторов и десериализаторов, меньшим объемом памяти при том же объеме полезной нагрузки по сравнению с XML

форматом. Помимо этого, мобильные клиенты взаимодействуют с основным сервером посредством WEB Socket протокола, а именно Socket.io[9] надстройкой над данным протоколом. Использование сокетов позволяет установить соединение между клиентом и сервером. Инициатором создания сообщения в данном протоколе может выступать любая из сторон хоть сервер, хоть клиент. Все клиенты могут быть объединены в общие «комнаты». Данное решение позволяет отправлять информацию о смене статуса заказа с сервера на клиентские модули, без необходимости периодической отправки запросов со стороны клиента на сервер при использовании HTTP протокола.

Приложение повара использует те же протоколы по соображениям, описанным выше.

Web клиент использует классический HTTP протокол по понятным причинам. Данный клиент так же использует WEB Socket протокол с Socket.io надстройкой, так как часть функционала сервера реализована в виде микро сервисов, для каждого из которых нет необходимости поднимать веб-серверы, достаточно объединить их в сеть посредством Socket.io.

1.5 Начальная схема базы данных и выбор сервера

Определившись с требованиями к информационной системе в целом, примерно определив все компоненты и их реализацию необходимо сформировать требования к базе данных и разработать схему базы, для хранения необходимой информации.

К базе данных предъявляются следующие требования:

- Хранить данные о пользователях системах.
- Хранить данные о ролях пользователей.
- Хранить данные о торговых точках / об их рабочих часах.
- Хранить данные о привязках оборудования к торговым точкам.
- Хранить данные о категориях товаров / о товарах / о связке товара с категорией.
- Хранить данные о заказах / о товарах в заказе.
- СУБД должна поддерживать язык запросов.
- СУБД должна быть в списке готовых драйверов для выбранной ORM[10].
- Должен быть готовый Docker образ с поддержкой сервера базы данных из коробки.

Наиболее распространённые решения используют реляционные базы данных[11]. В последнее время набирают популярность не реляционные базы, такие как mongoDb[12], NoSQL и прочие. В данной работе они не рассматриваются, так как привычные реляционные базы данных покрывают все требования.

Исходя из требований были выбраны три основных сервера баз данных: PostgreSQL, MySQL, MS SQL. В таблице 3 представлены сравнения серверов по следующим параметрам:

- Поддержка ORM Entity Framework.
- Наличие готового Docker образа.
- Бесплатное использование.
- Наличие лицензии.

Таблица 3 — Сравнение рассматриваемых серверов баз данных.

	Поддержка ORM Entity Framework	Docker образ	Бесплатное использование	Наличие лицензии
PostgreSQL	Частично используя драйвер	Postgres-alpine-9.6.13	Да	Не требуется
MySQL	Частично используя драйвер	Mysql-8.0.16	Да	Не требуется
MS SQL	Из коробки	microsoft/mssql-server-linuX-2017-CU13	Нет	Учебная

Первые два участника имеют частичную поддержку Entity Framework. В драйверах для них отсутствует важная функция – поддержка миграций, что затрудняет разработку при использовании подхода Code First[13]

Таким образом на время разработки был выбран сервер MS SQL из-за удобства и наличия обучающей версии. При переходе на коммерческое использования готовая схема базы данных будет перенесена на сервер MySQL. Так же стоит отметить что по размерам и быстродействию Docker образов MySQL превосходит своих конкурентов.

Основываясь на требованиях с БД была разработана схема изображенная на рисунке 3.

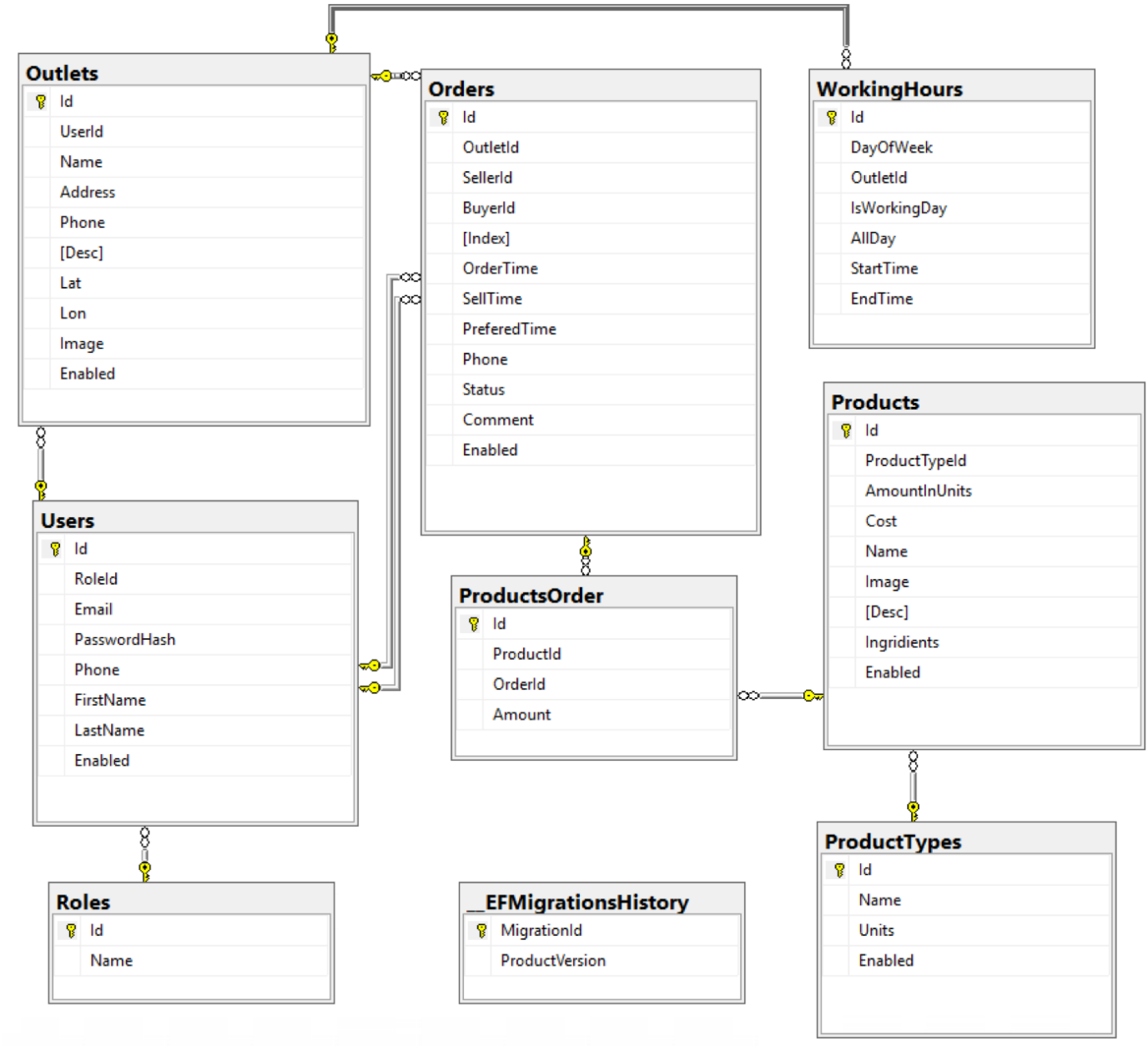

Рисунок 3 — Схема базы данных.

Таблица Users и таблица Roles описывают пользователей и их роли. Один пользователь может иметь одну роль, так что связь один к одному. Таблица Users не хранит в себе информацию о токене пользователя, так как используются JWT токены подробнее в разделе «3 Программная реализация».

Таблицы Outlets и WorkingHours описывают торговые точки и время их работы. На каждую запись в таблице Outlets хранится семь записей в таблице WorkingHours. Поддержка целостности данных в этом случае обеспечивает основной сервер.

Таблицы Products и ProductType описывают продукты и категории, в которых они находятся.

Таблица Orders описывает заказы. Данная сущность имеет связь многие ко многим по отношению к сущности Product. Решением данной связи является классическая промежуточная таблица ProductsOrder.

2. ПРОЕКТИРОВАНИЕ СЕРВЕРНОГО КОМПОНЕНТА ИС

Основной целью данной исследовательской работы является разработка серверного компонента информационной системы. Прежде чем переходить к выбору технологий и разработки архитектуры необходимо определить функциональные требования к компоненту. Стоит отметить, что в данной работе под серверным компонентом подразумевается совокупность нескольких программных приложений, которые взаимодействуют между собой, взаимодействуют с клиентскими компонентами. Некоторые из программных приложений являются самостоятельными серверами, но не являются серверным компонентом ИС.

2.1 Определение функциональных требований к серверному компоненту

Исходя из общей архитектуры ИС, протоколов, используемых для взаимодействия между компонентами и требований к ИС были сформулированы требования к серверному компоненту описанные в таблице 4.

Таблица 4 — Функциональные требования к серверному компоненту

Требование	Объяснение необходимости
Поддержка протоколов HTTP, Web-socket и надстройки Socket.io.	Раннее были определены основные протоколы взаимодействия между компонентами ИС.
Поддержка JSON REST API.	Обусловлено удобством использования на клиентских компонентах, наиболее распространённое решение, наличие готовых библиотек для взаимодействия с JSON объектами.
Поддержка multipart/form запросов.	Необходимо загружать различные файлы в том числе картинки,

	использование FTP протокола усложняет взаимодействие между компонентами, а перевод файлов в base64 кодировку излишне нагружает кодировку.
Поддержка авторизации по средствам использования токенов.	В информационной системе существуют несколько ролей. Каждая из ролей наделена своими привилегиями. Для разграничения привилегий между пользователями необходима авторизация.
Наличие документации по основным методам API и описание DTO[14] объектов.	Так как разработка ИС ведется несколькими разработчиками одновременно, необходима документация, для возможности реализации компонентов во время отсутствия полной реализации серверного компонента.
Общее изолированное пространство для выполнения методов API.	Для отладки и понимания работы методов API необходимо общее пространство с синхронизацией между разработчиками.

2.2 Выбор стека технологий для реализации сервера

После определения основных функциональных требований можно перейти к выбору технологий, используемых для реализации серверного компонента. При выборе технологий очень важно использовать опыт применения технологий в Enterprise решениях, а так же наличием собственного опыта, так как сроки и ресурсы выполнения разработки ограниченны.

В качестве сервера базы данных на время разработки ИС был выбран MS SQL Server подробное обоснование в пункте «1. 4 Начальная схема базы данных и выбор сервера».

Исходя из опыта существующих Enterprise решений[15], на роль основного веб сервера предоставляющего REST API, выделены три основных претендента: PHP с использованием Symphony Framework, J2EE с использованием Hypermedia Controls, ASP .Net или ASP .Net Core. Сравнение решений представлено в таблице 5.

Таблица 5 — сравнение решений в качестве основного веб-сервера.

	PHP	J2EE	ASP .Net
Встроенный веб сервер	нет	Apache TomCat	Kestrel
ORM	Doctrine	Hibernate	Entity Framework
Библиотека для Socket.Io	socket.Io-client	нет	dotnet-socket.io-client dotnet-cosket.io-server
Личный опыт разработки	2 месяца	нет	3 года

Ключевым фактором при выборе технологии для основного веб-сервера стало наличие личного опыта применения, таким образом основным сервером является ASP .Net Core 2.2.

ORM для взаимодействия с базой данных является Entity Framework core, выбор был сделан исходя из наличия опыта, частому применению в Enterprise решениях.

В качестве Web-socket сервера выбран сервер с надстройкой Socket.io. Данный сервер основан на протоколе web-socket, что позволяет взаимодействовать с клиентами на уровне абстракции данного протокола, однако имеет надстройку, несколько увеличивающую уровень абстракции, что позволяет упростить процесс настройки сервера. Так же данная надстройка упрощает взаимодействие с клиентскими приложениями. Примером таких упрощений являются автоматические мутации для связки Vue.js + Vuex[16].

Помимо основного веб сервера, необходим веб-сервер, который отвечает за доставку файлов для веб-клиента. Простым и наиболее распространенным решением является веб-сервер Nginx. Данный сервер может быть настроен следующим образом: запросы на сервер веб-клиента обрабатывает непосредственно Nginx, запросы на статические файлы обрабатывает Nginx, запросы к API Nginx проксирует к основному серверу на Asp .Net.

Одним из требований к информационной системе является просмотр статистики, как правило подсчет статистики требует большого количества операций чтения из базы, из-за высокой абстракции Entity Framework данные операции выполняются дольше в отличии от выполнения оптимизированных SQL запросов. Так же операции по расчету статистики могут излишне нагружать основной сервер, что снизит быстродействие всей системы в целом.

Было принято решение вынести операции по расчету статистики в отдельный микро сервис, написанный на языке с уровнем абстракции ниже чем уровень абстракции в ASP .Net. Данное решение позволяет снизить нагрузку на основной сервер, уменьшить время на расчет статистики.

Микро сервис по расчету разработан на языке от компании Google — GO[17]. Данный микро сервис должен взаимодействовать либо напрямую с клиентскими приложениями, с использование Nginx в качестве прокси, либо с основным сервером. Вариант прямого взаимодействия с основным сервером имеет серьезный недостаток, он повлечет за собой изменения логики в клиентских приложениях, так как потребует дополнительных удаленных вызовов процедур. Вариант взаимодействия с основным сервером — наиболее предпочтителен.

В реальных условиях возможны ситуации, в которых несколько компонентов внутри серверного компонента будут недоступны по каким-либо причинам. В то же время изменение сущностей, влияющих на статистику будут происходить регулярно. Исходя из этого появляется необходимость в некотором буфере или очереди сообщений, который будет обработан после восстановления функционирования сервиса по расчету или любого другого сервиса.

Для данной проблемы уже существуют готовые решения. Одним из таких решений является программный брокер сообщений, работающий поверх протокола AMQP[18] под названием RabbitMQ. Основной сервер отправляет сообщение в очередь RabbitMQ об изменении сущностей, RabbitMQ помещает сообщения в очередь и хранит до тех пор, пока микро сервис не прочитает сообщения. Микро сервис восстанавливает свою работу, читает сообщения из очереди и производит перерасчет статистики. После перерасчета отправляет данные в socket.io сервер.

Помимо этого для «боевой» развертки, упрощения CI/CD[19] процессов, возможности мультиплицирования и балансирования используются Docker контейнеры. Все контейнеры объединены в общую виртуальную сеть посредством инструмента Docker-compose[20].

Благодаря использованию Docker контейнеров все используемые модули серверного компонента можно развернуть на большинстве существующих платформ, в том числе ARM платформы, например Raspberry

Pi[21]. Эта возможность позволяет развернуть сервер локально на недорогом оборудовании, либо на облачных сервисах, например Яндекс облако[22].

Для документации методов API и синхронизации песочницы выполнения используется сервис Postman[23]

В результате анализа готовых решений и лучших практик применения различных технологий в Enterprise решений был определен стек технологий изображенный на рисунке 4.

Рисунок 4 — Стек используемых технологий

Выбранный стек технологий позволяет легко масштабировать серверный компонент, подключать различные микро сервисы, опционально настраивать различные модули. Так же такое решение позволяет развернуть все необходимое окружение для разработки на рабочей станции в несколько команд.

2.3 Разработка архитектуры сервера

Заключительным этапом проектирования серверного компонента информационно системы является разработка архитектуры. Этот этап очень важен, так как именно архитектура определяет насколько легко вносить изменения и масштабировать систему. При использовании методологии гибкой разработки[24] архитектура приложения должна позволять вносить изменения с минимальными затратами. Именно гибкость и модульность разработанной ИС является одним из конкурентных преимуществ.

Исходя из требований к серверу, из стека технологий была разработана архитектура, представленная на рисунке 5.

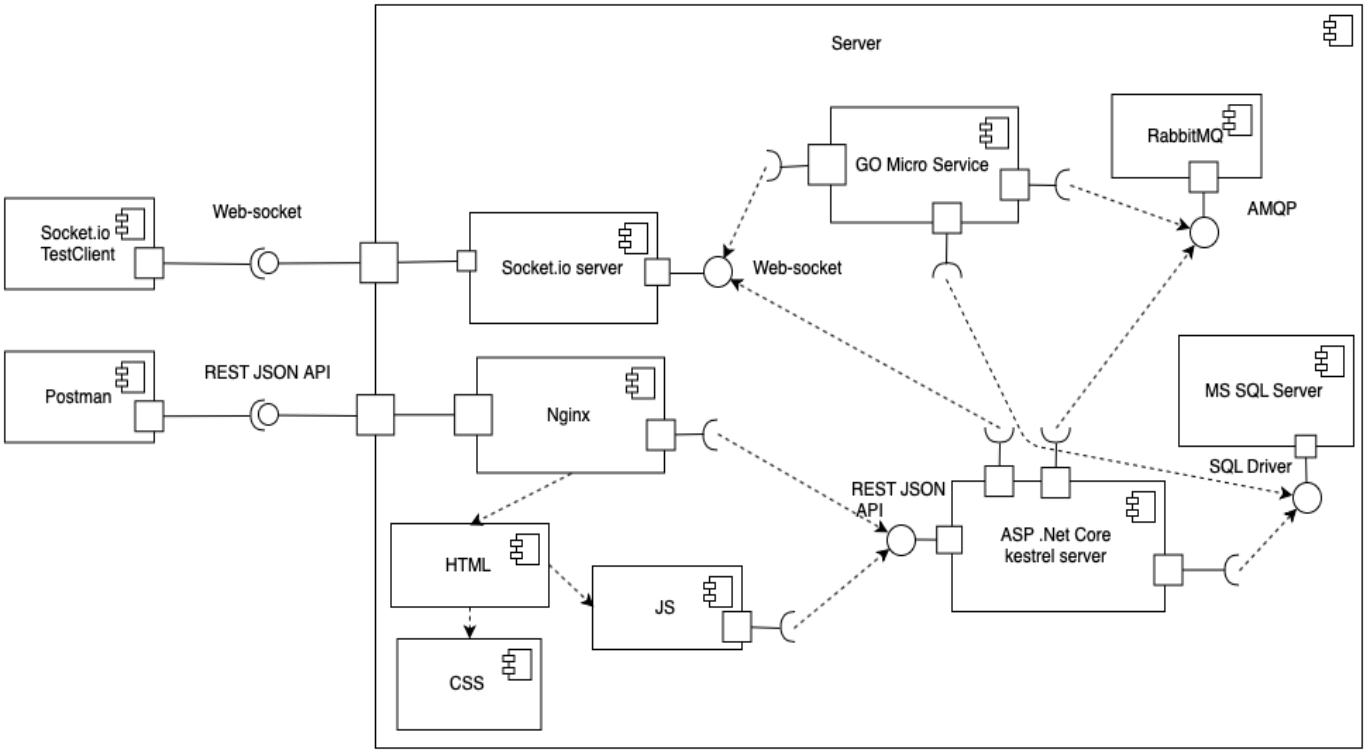

Рисунок 5 — Диаграмма компонентов серверной составляющей ИС.

Серверный компонент имеет два порта. Первый порт пробрасывает Web-socket протокол до Socket.io сервера. Второй порт пробрасывает HTTP протокол до Nginx сервера. В зависимости от URL в заголовке HTTP запроса Nginx либо отдает HTML страницу для Веб-клиента, либо если URL

удовлетворяет маске «*/api/*», проксирует запрос на ASP .Net Core kestrel сервер.

Asp .Net Core сервер взаимодействует с MS SQL сервером, а так же с RabbitMQ по протоколу AMQP и Socket.io по протоколу Web-socket.

Микро сервис, написанный на языке GO, использует интерфейс, предоставляемый MS SQL, так же использует интерфейсы от RabbitMQ и Socket.io.

Общий поток данных выглядит следующим образом: клиентские приложения отправляют запрос на api, ASP .Net Core обрабатывает запрос, подтягивает данные из базы, генерирует ответ, отправляет ответ на запрос. Затем ASP .Net Core отправляет сообщение в Socket.io об изменениях каких-либо сущностей, данное сообщение приходит всем клиентским приложениям, которые установили соединение с Socket.io. Помимо этого ASP .Net Core помещает сообщение в очередь RabbitMQ, о необходимости вызова процедуры у любого из микро сервисов. Микро сервис считывает сообщение из очереди, выполняет какие либо действия и отправляет сообщение в Socket.io сервер.

Для тестирования и отладки работоспособности серверного компонента используются два основных инструмента: Postman и Socket.io TestClient.

Postman позволяет валидировать JSON данные, генерировать HTTP запросы, отправлять их, отображать ответы на запросы и метаданные, такие как заголовки, время выполнения и тому подобное. Помимо прочего, данный инструмент позволяет вести коллекции запросов, генерировать документацию по запросам в автоматическом режиме, синхронизировать запросы между рабочими пространствами разроботчиков.

Socket.io Client — плагин для браузера Google Chrome, позволяющий подключаться к серверу, просматривать и отправлять сообщения.

Архитектура развертывания серверного компонента ИС представлена на рисунке 6.

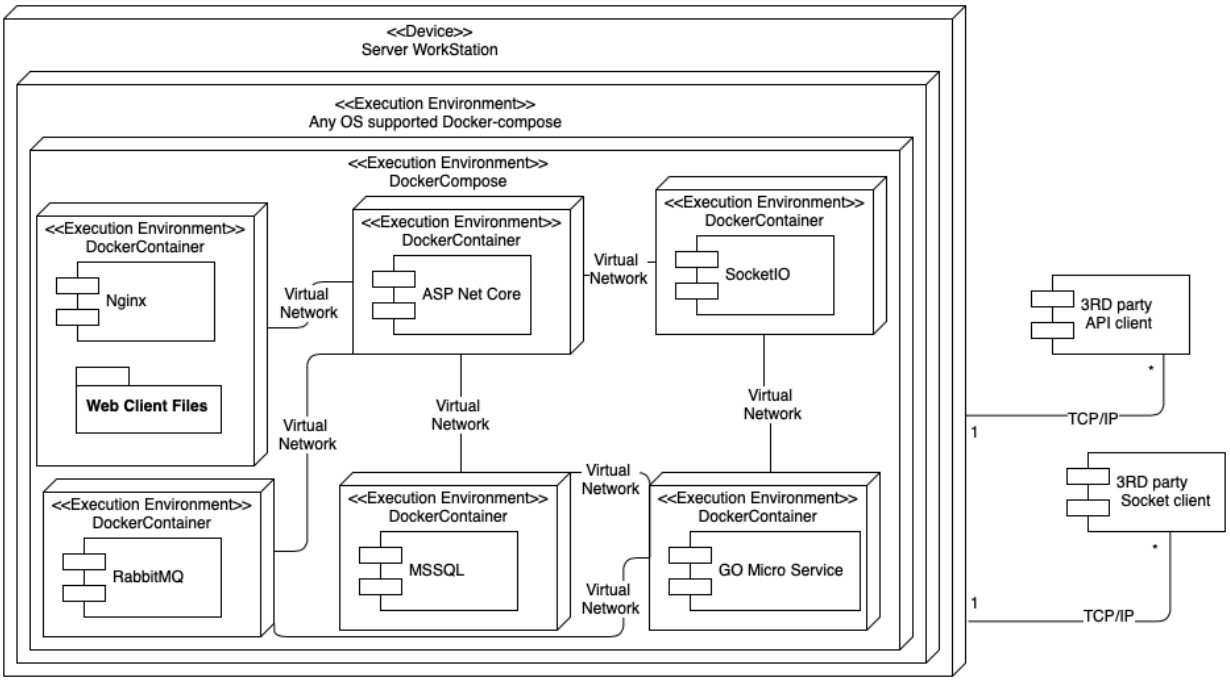

Рисунок 6 — Упрощенная архитектура развертывания серверного компонента ИС.

Как видно из диаграммы, устройством, на котором будет развернут серверный компонент может быть любым устройством с поддержкой ОС, которая в свою очередь поддерживает Docker-compose. Каждый программный компонент сервера упакован в свой Docker контейнер. Все Docker контейнеры объединены между собой виртуальной сетью посредством Docker-compose.

Развертывание и обновление компонентов происходит путем доставки на сервер «docker-compose.yml» файла, и выполнение команды «docker-compose run docker-compose.yml -d». Инструмент Docker-compose самостоятельно подтягивает все необходимые зависимости и docker образы, запускает контейнеры и поддерживает их работоспособность. В случае завершения любого из контейнеров в аварийном режиме docker-compose уведомит разработчиков и будет пробовать перезапускать контейнер.

3. ПРОГРАММНАЯ РЕАЛИЗАЦИЯ СЕРВЕРА

После определения требований, выбора стека технологий и разработки архитектуры наступает черед реализации компонентов.

Компонент Nginx и веб-клиент уже реализованы и требуют только конфигурации. Компоненты RabbitMQ, Socket.io так же реализованы и требуют только настройки. Так как используется подход CodeFirst компонент MSSQL так же не требует реализации.

Настройка компонентов подразумевает редактирование файлов конфигурации и пересборку контейнеров. Данный этап не рассматривается в исследовательской работе.

Необходимо разработать веб приложение на платформе ASP .Net Core 2.2, микросервис на языке GO.

3.1 Разработка Веб-приложения на платформе ASP .Net Core

Выбор версии Core вместо классического .Net выбран из-за следующих преимуществ:

- Меньший объем потребляемых ресурсов.
- Частичная кросс-платформенность.
- Упрощенная настройка редактированием одного .json файла.
- Наличие встроенного сервера kestrel.

Первым делом были созданы POCO классы, описывающие сущности бизнес процессов. Затем установлен Entity Framework Core посредством менеджера пакетов NuGet, сгенерирован контекст и начальная миграция. После чего миграция была применена. Следующим шагом был написан помощник инициализации базы и контроллер для вызова данного помощника.

После этого были разработаны DTO модели на основе документации, сервисы, взаимодействующие с базой данных, контроллеры взаимодействующие с сервисами.

Упрощенно архитектура приложения изображена на рисунке 7.

Рисунок 7 — Упрощенная архитектура веб приложения.

Отношение компонентов к слоям отображено в таблице 6.

Таблица 6 — Отношение компонентов приложения к слоям.

Слой доступа к данным	POCO классы, контекст ORM, драйвера доступа к базе данных
Слой бизнес логики	Сервисы, реализующие логику обработки данных, статические классы упрощающие реализацию сервисов, инструменты ограничения прав доступа.
Слой представления	Контроллеры, сериализаторы, механизмы авторизации, DTO объекты, мапперы, механизмы внедрения зависимостей.

Условное разделение компонентов приложения позволяет переиспользовать часть компонентов в других приложениях и сервисах, однако в современных решениях данной возможностью пользуются редко.

Так как разработка веб приложения является одной из главных задач данной исследовательской работы ниже представлены подробности реализации некоторых компонентов приложения.

3.2 Внедрение зависимостей или Dependency Injection

ASP .Net Core имеет встроенный механизм внедрения зависимостей, основанный на интерфейсе IServiceProvider. Обычно механизм внедрения зависимостей используется для создания связей между абстракциями, например интерфейсами, и конечными реализациями. В данной исследовательской работе не используются интерфейсы, в связи с ограниченным временем разработки. В таком случае механизм внедрения зависимостей служит для того, чтобы однажды настроив создание какой-либо конечной реализации того или иного компонента, можно была использовать его внутри других компонентов приложения.

Встроенный механизм внедрения зависимостей имеет несколько различных вариантов передачи зависимостей, в данном приложении, в основном, используется передачи зависимостей через конструктор класса.

Для того, чтобы не усложнять читаемость кода Startup класса, настройка некоторых встроенных сервисов была вынесена в отдельные методы расширений. Методы расширений позволяют вынести часть функционала в отдельные файлы и методы, используя их потом как методы класса. Однако с методами расширения нужно быть осторожными, так как это статические методы и при использовании unit тестов невозможно создать mock объект, реализующий функционал статического метода или класса.

3.3 Аутентификация и авторизация

В разработанной ИС существует несколько ролей, на основе ролей происходит разграничение прав доступа. Для понимания какой роли принадлежит тот или иной пользователь необходима аутентификация с последующей авторизацией.

Простейший вариант аутентификации является передача логина и пароля в метаданных HTTP запросов. Данный вариант не является безопасным, так как данные о логине и пароле можно перехватить и использовать в корыстных целях.

Современным решением является использование токенов. Токен представляет собой строку зашифрованный каким либо алгоритмом. Пользовательские приложение однажды передают логин и пароль, сервер проверяет подлинность данных и генерирует токен, который передается клиенту в ответе. При каждом последующем запросе клиентское приложение передает в заголовке токен, таким образом сервер может произвести аутентификацию.

Классический вариант использования токенов авторизации имеет ряд недостатков:

- Информацию о токенах и о связи пользователь-токен необходимо хранить.
- Для того чтобы произвести процесс аутентификации необходимо произвести поиск по базе, что занимает время и с ростом количества токенов в базе увеличивается время поиска.
- Информацию об устаревших токенах нужно удалять, нужно понимать какие токены устарели.

JSON Web Token (JWT)[25] лишен данных недостатков. JWT состоит из трех частей: header, payload, crypt-data. В заголовке передаются мета-данные об алгоритме шифрования и типе токена, в полезной нагрузке могут находиться любые полезные данные, будь то роль, имя или id пользователя. В третьей части токена передается подпись и время жизни токена.

JWT генерируется на сервере и ключ подписи храниться только на сервере. Таким образом, когда поступает запрос с таким токеном на сервер, сервер валидирует подпись и время жизни токена, если токен валиден данные о пользователе извлекаются из строки. Отпадает необходимость хранить токены, очищать устаревшие данные и производить поиск по базе.

В разработанном приложении применяется собственный сервис по генерации токена и ПО промежуточного слоя, которая подмешивает информацию о пользователе и его роли в контекст запроса.

На рисунке 8 представлена диаграмма последовательности вызовов при получении токена.

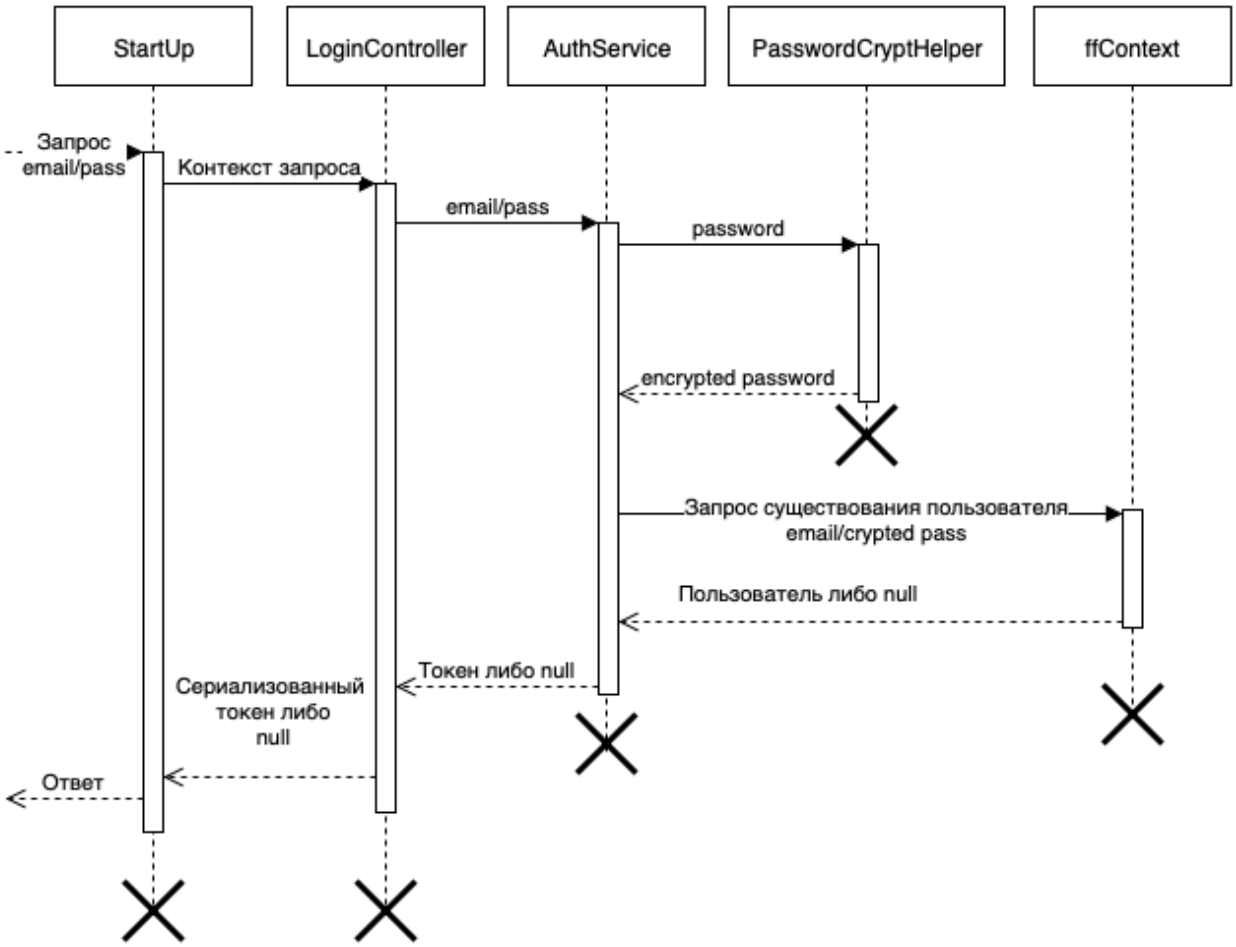

Рисунок 8 — Последовательность вызовов при получении токена.

Пользователь передает запрос с информацией о email и пароле. Считаем, что все компоненты уже созданы механизмом обратного контроля. Далее передается контекст запроса в контроллер, контроллер валидирует наличие

почты и пароля. Контроллер вызывает метод сервиса, в который передает информацию о почте и пароле. Далее сервис хеширует пароль с помощью помощника и проверяет при помощи контекста существование пользователя. Если пользователь существует генерирует токен, иначе возвращает пустое значение. В конце по цепочке пользователю возвращается сериализованый токен либо пустое значение.

На рисунке 9 изображен процесс обработки запроса с токеном.

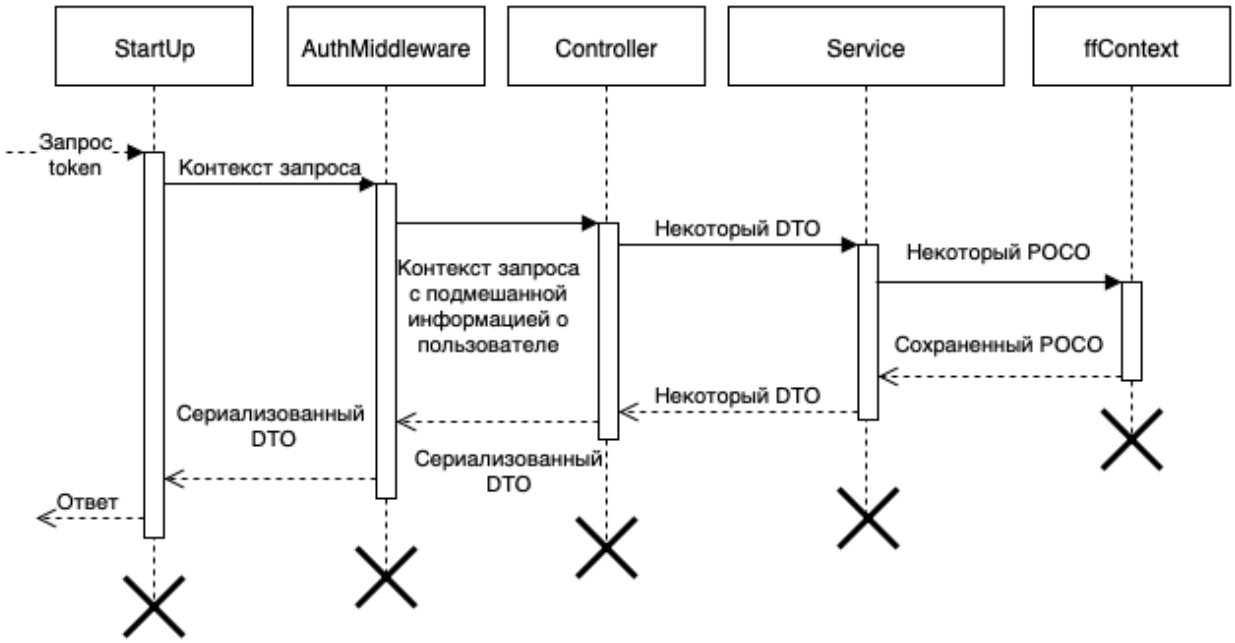

Рисунок 9 — Обработка запроса с токеном.

Как и в предыдущем запросе все компоненты созданы. Запрос проходит через начальный класс, далее роутер перенаправляет его в ПО промежуточного слоя. ПО промежуточного слоя расшифровывает и валидирует токен из заголовка запроса, подмешивает информацию о пользователе в контекст запроса. Далее контроллер создает и валидирует DTO из тела запроса и передает его в некоторый сервис. Сервис преобразовывает DTO в POCO и передает в контекст. Контекст базы данных возвращает сохраненный POCO, сервис преобразовывает его в DTO, контроллер сериализует его и возвращает по цепочке пользователю.

3.4 Преобразование DTO объектов или AutoMapper

Одним из классических подходов при разработке приложений на языках со строгой типизацией является использование объектов DTO. Объекты передачи данных имеют структуру, которая при сериализации будет отправлена в другие программные модули. Так же используя объекты DTO в приложениях ASP .Net Core можно настроить валидацию входных данных.

Объекты DTO относятся к уровню представления и мало пригодны для переиспользования в других приложениях без изменения.

Для того чтобы преобразовать POCO объект в DTO объект и наоборот существует два основных способа: ручное преобразование, автоматическое преобразование. При использовании ручного преобразования разработчик сам описывает логику преобразования. Обычно, вся логика преобразования сводится к переприсвоению одноименных свойств объекта.

Для автоматического преобразовании объектов существуют готовые библиотеки, однако автоматическим подобные преобразования назвать сложно. Прежде чем выполнить преобразование, необходимо создать объект преобразователя, потом сконфигурировать его и только после этого выполнить преобразование. Для упрощения данного процесса была написана обертка в виде статического класса, который позволяет выполнить преобразования двух объектов или коллекции объектов.

Пример вызова преобразования:

«MapHelper.Map<Product, ProductOut>(product)»

3.5 Загрузка файлов и multipart запросы

В современных решениях все чаще в HTTP запросах в качестве полезной нагрузки выступают данные в формате JSON. Возможным решением является отправка запросов с полезными данными в теле типа form-data. form-data это формат ключ-значение. В качестве значения могут выступать строки. Третий вариант — multipart/form-data, в таком случае в качестве значения могут быть как строки так и файлы.

В разработанной информационной системе, при создании торговой точки необходимо загрузить фотографию, информацию о точке и информацию рабочих часах торговой точки. Информацию о часах торговой точки невозможно описать используя формат ключ – значение.

В данном случае можно разделить загрузку информацию и загрузку фотографии на два запроса, однако данный вариант имеет ряд недостатков.

Вторым возможным решением является загрузка фотографии как одно поле multipart/form-data, а загрузка информации в формате json как второе поле. Данный запрос так же имеет недостатки:

- DTO объект в контроллере веб приложения должен иметь только два поля.
- Валидация данных JSON не произойдет в автоматическом режиме.

Для решения данных проблем было принято решение описать логику связывания свойств объекта DTO с полями запроса, настроив валидацию таким образом.

В ASP .Net Core за связывание данных запроса со свойствами объектов отвечает интерфейс IModelBinder. Разработав класс, который реализует данный интерфейс можно описать собственную логику связывания данных.

Таким образом был разработан класс «JsonWithFilesFormDataModelBinder.cs». Для применение собственного объекта связывания данных необходимо воспользоваться механизмом аннотации данных и указать название объекта перед объявлением класса DTO.

3.6 Отказ от DTO объектов в пользу JSON объектов

Раннее в работе был рассмотрен способ сериализации данных с использованием объектов DTO. Данный способ широко распространен, однако требует дополнительных затрат со стороны разработчика, а так же при изменении основных классов, зачастую требуются изменения в DTO классах.

Вторым возможным способом создания желаемого JSON объекта на выходе, является создание и сериализация JObject объекта.

JObject объекты — это объекты классов библиотеки NewtonSoft JSON[26]. Данная библиотека позволяет создавать объекты с динамическими свойствами, которые можно изменять во время выполнения программы. Так же данная библиотека имеет встроенный сериализатор и десериализатор, что избавляет от необходимости использования различных мапперов. Однако несмотря на привлекательность использования таких объектов делает невозможным использование встроенных механизмов валидации данных.

Использование JObject актуально для данных, у которых могут динамически изменяться свойства, или требования ко входным и выходным данным могут меняться во время развития проекта.

4. ПРИМЕР ИСПОЛЬЗОВАНИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ

В данном разделе будут рассмотрены примеры использования информационной системы с точки зрения реализации сервера.

4.1 Пример создания и обработки заказа

На рисунке 10 изображена последовательность вызовов и действий при создании и обработке заказа. В рассматриваемом примере все компоненты сервера запущены и функционируют, все компоненты ASP .Net Core приложения созданы механизмом обратного контроля. Клиентские приложения уже запущены и установлены соединения по веб-сокетам

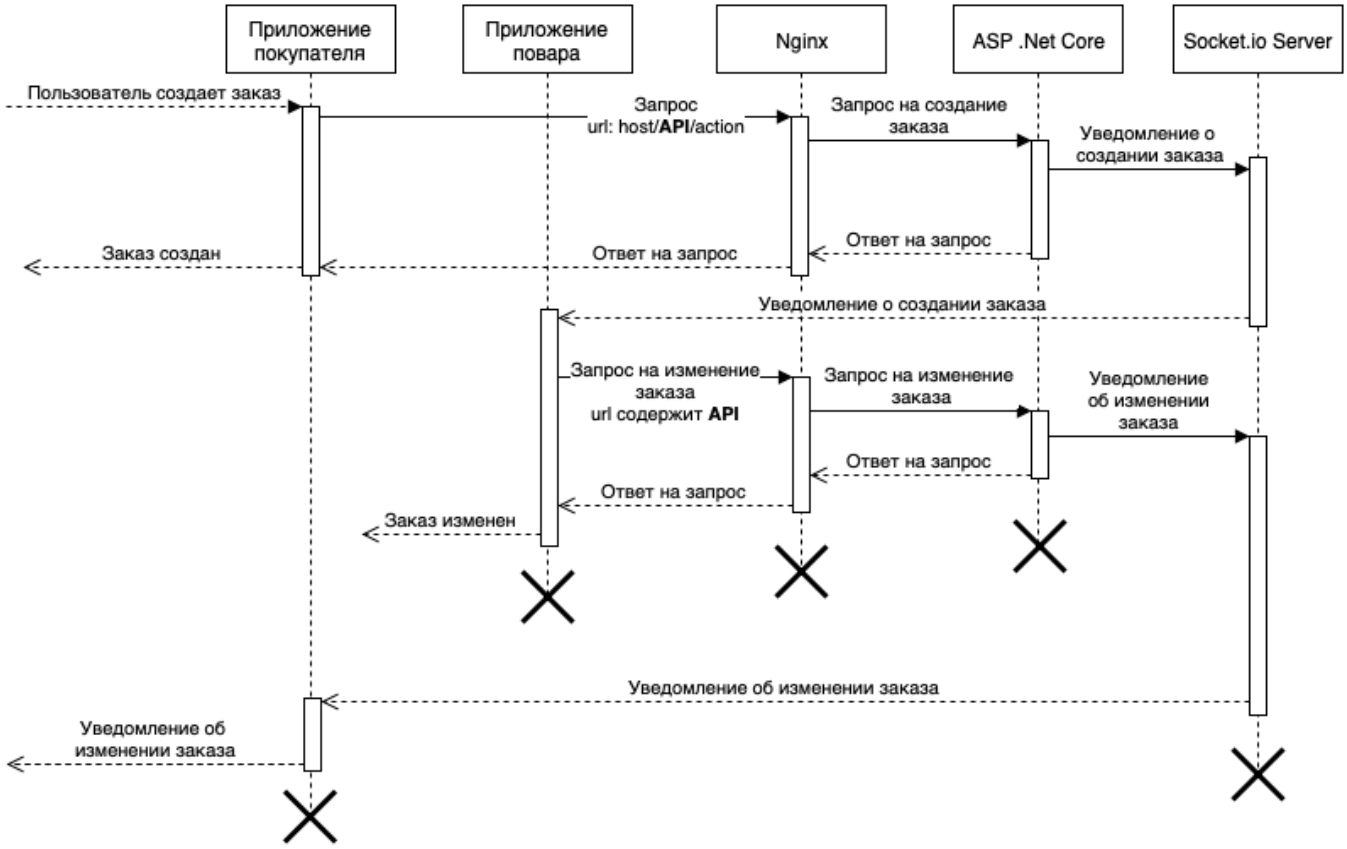

Рисунок 10 — последовательность вызовов при создании и обработке заказа.

Пользователь, воспользовавшись приложением создает заказ. Информация о товарах уже загружена приложением раннее. Приложение передает запрос о создании заказа на Nginx сервер. Так как URL в запросе содержит слово API, Nginx проксирует его в kestrel сервер ASP .Net Core приложения. ASP .Net Core обрабатывает запрос, подробнее в пункте 3.3,

возвращает ответ серверу Nginx и отправляет сообщение в Socket.io сервер о создании заказа. Далее ответ об успешном создании заказа по цепочке возвращается в пользовательское приложение. Socket.io сервер уведомляет приложение повара, по заранее установленному сокет соединению. Повар обрабатывает заказ и изменяет его статус. Приложение повара отправляет запрос на Nginx сервер. Как и предыдущий запрос, текущий доходит до ASP .Net приложения. Приложение обрабатывает его, возвращает ответ об успешном изменении статуса и отправляет сообщение в Socket.io сервер. Socket сервер уведомляет клиентское приложение об изменении статуса заказа.

4.2 Использование ИС администратором сети, расчет статистики микросервисом.

На рисунке 11 изображен процесс обработки запроса от администратора на расчет статистики по продажам. Как и в прошлый раз, сокет соединения установлены, все компоненты уже созданы и функционируют, все компоненты ASP .Net Core приложения созданы механизмом обратного контроля.

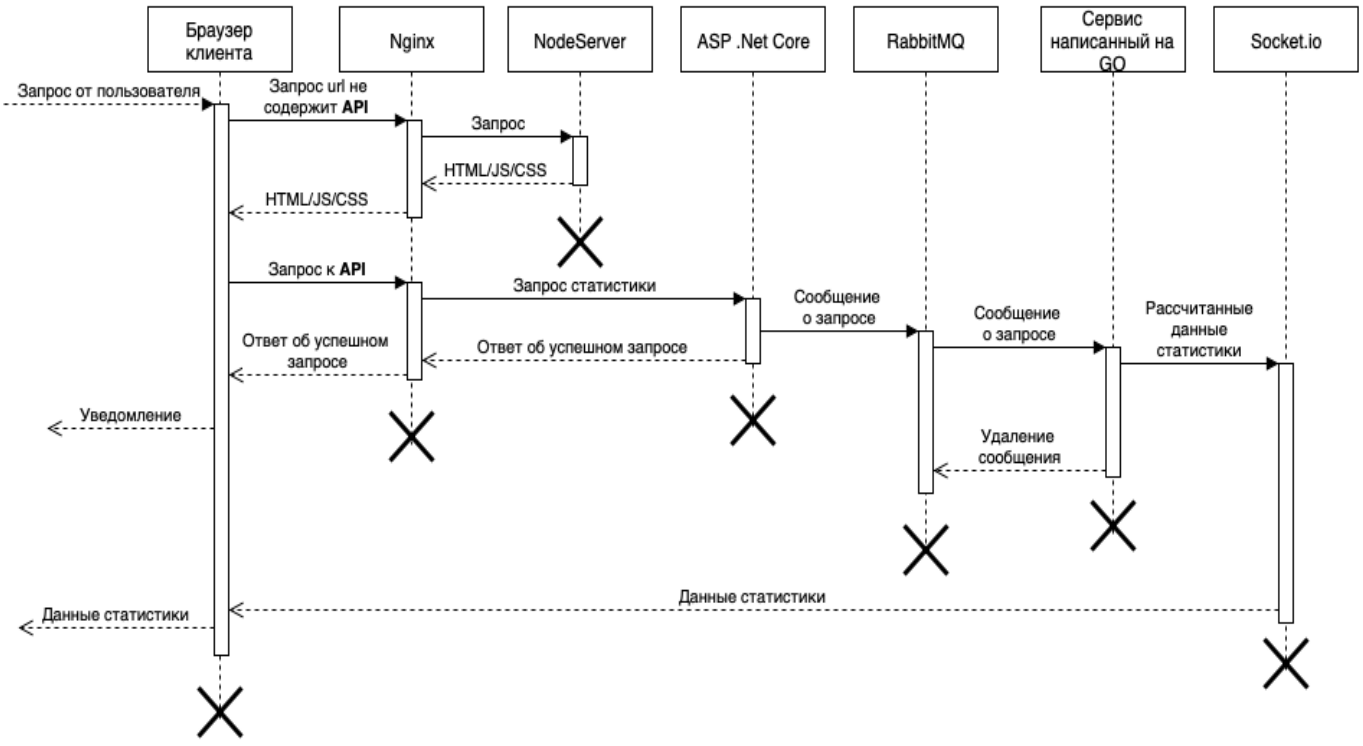

Рисунок 11 — Обработка запроса от администратора сети

на получение статистики.

В первую очередь, администратор, используя браузер, отправляет запрос на сервер. Так как URL запроса не содержит слово API, Nginx проксирует его на сервер Node, а возвращает. Node сервер возвращает по цепочке файлы HTML, CSS, JS. На следующем этапе браузер, исполняя JS код выполняет запрос на сервер о расчете статистики. В данном случае URL содержит слово API и Nginx сервер проксирует его в ASP .Net Core приложение. Веб приложение отправляет по цепочке ответ об успешном запросе и одновременно отправляет в очередь RabbitMQ сообщение о необходимости произвести расчет статистики.

Если сервис расчета недоступен, сообщение в очереди будет находится до тех пор, пока оно не будет прочитано потребителем сообщений. Docker-compose, в случае если контейнер с сервисом расчета недоступен, будет предпринимать попытки переподнять его и уведомит администратора ИС.

Сервис расчета статистики, написанный на языке GO, считывает сообщение из очереди, тем самым удаляя его. Затем сервис обращается к базе данных и производит расчет статистики. После расчета статистики, сервис отправляет данные в Socket.io сервер, который, в свою очередь по предустановленному соединению отправляет его на веб-клиент администратора.

5. КОНЦЕПЦИЯ СТАРТАП-ПРОЕКТА

5.1 Описание продукта как результата НИР

Разрабатываемая ИС предназначена для автоматизации предварительного заказа еды в сети кафе быстрого питания. Потребителей системы можно разделить на две категории: бизнес (сеть кафе) как потребитель и конечные пользователи, осуществляющие заказы еды.

Основным сегментом потребителей, среди конечных пользователей, являются люди, которые имеют ограниченное время на прием пищи, что не позволяет стоять в очередях и тратить время. Примером такого сегмента являются студенты. Как правило, у студентов мало времени на прием пищи, перерывы между парами наступают одновременно и в кафе быстрого питания скапливаются очереди что приводит к нехватке времени. Разрабатываемая ИС позволит сократить время на обслуживание заказа, что приведет к сокращению времени у пользователя на получение заказа.

Второй группой потребителей ИС рассматривается бизнес. Зачастую сети быстрого питания децентрализованы, это пагубно сказывается на возможности присоединения к существующим решениям онлайн заказа еды. Помимо этого, существующие решения в большинстве случаев являются агрегаторами, которые не следят за качеством исполнения заказов. Для бизнеса разрабатываемая ИС решает следующие проблемы: соединяет децентрализованные кафе в единую сеть, собирает статистику и отчетность по каждому кафе и позволяет проследить динамику изменения продаж, предоставляет конечному пользователю информацию обо всех точках продаж, позволяет интегрировать ИС с используемыми программными решениями (мультикасса ЭВАТОР, табло статусов заказов и т.п.) для упрощения бизнес-процессов.

Основным предметом НИР в разработке данной ИС является использование технологий, позволяющие придать системе максимальную гибкость и модульность. Необходимость в гибкости обусловлена не

стандартизированными БП при продаже еды. Внутри одной сети могут находиться как торговые точки, которые продают еду только на вынос, так и кафе с посадочными местами. Таким образом каждая конкретная торговая точка, кафе, ресторан должны иметь свой уникальный набор опций таких как экран статусов заказов, печать чеков с QR кодом, лазер, позволяющий нанести символику на продукцию и т.д.

Для реализации вышеперечисленных требований были использованы следующие технологии: ASP.NET Core — основной сервер, взаимодействующий с базой данных, MS SQL — СУБД, RabbitMQ — сервис доставки сообщений, Socket.io — сокет-сервер, Docker-composer — менеджер контейнеров, микросервисы, разработанные на языке GOLang. Данные технологии позволяют упаковать каждый компонент в свой контейнер, объединить их в единую сеть и использовать опционально.

5.2 Целевые сегменты потребителей создаваемого продукта

Целевой сегмент разрабатываемого продукта — бизнесы, работающие в сфере быстрого питания. Это рестораны с блинами, шаурмой, шашлычные, кофейни, то есть все торговые точки, которые работают с едой на вынос.

У ресторанов быстрого питания, особенно расположенных в студенческом центре с большим трафиком, основная аудитория — студенты. Можно заметить, что во время перерывов в вузе в таких торговых точках скапливается большая очередь; продавцы не успевают обслужить такой поток посетителей, и многие студенты, боясь опоздать на следующую пару, отказываются от покупки. Со временем студенты начинают выбирать торговые точки, в которых очередей либо почти нет, либо где обслуживание посетителей происходит очень быстро.

Разрабатываемый продукт поможет ресторанам быстрого питания сохранить текущих покупателей, а также привлечь тех студентов, которые не успевают перекусить во время перерыва. Это достигается с помощью

мобильного приложения, которое брендируется и дорабатывается под конкретную торговую точку или сеть. Владельцы торговых точек смогут привлекать новых посетителей с помощью своего приложения, а возможность сделать заказ онлайн позволит конечным пользователям (студентам) заранее заказывать еду, а затем просто забирать ее на перерыве.

5.3 Объем и емкость рынка

Больше половины россиян – 58% – заказывают готовую еду онлайн (NPD, апрель 2017 – март 2018). При этом доля интернет-заказов очень быстро растет: в первом квартале 2018-го года по сравнению с тем же периодом 2017-го в среднем по России рост составил 7,5%.

73% миллениалов делают покупки через свои смартфоны. Миллениалы — люди, рожденные в период с 1980 по 2000 год. Эта аудитория намного превосходит в тратах предыдущие поколения, поэтому является «лакомым кусочком» для компаний в разных сферах. Большинство людей, которые пользуются доставкой — молодежь до 35 лет (Исследовательский холдинг «Ромир»).

Мировой рынок доставки продуктов питания стремительно растет и составляет, по данным McKinsey, около $96,2 млрд. Это примерно 1% от всего рынка продуктов питания и около 4% от рынка продуктов питания, продаваемых через рестораны и сети быстрого питания. Заказать еду в пару кликов для американца или европейца давно стало обыденностью.

В России общий объем рынка доставки продуктов питания в 2017 году составил $3,5 млрд, подсчитали в Target Global. Рынок доставки готовой еды достиг $1,6 млрд. Несмотря на столь впечатляющие цифры, доля онлайн-сервисов по-прежнему очень мала — менее 1%. В целом рынок доставки повторяет американские и европейские тенденции, но сильное влияние на развитие оказывает менталитет и экономическое состояние. В России слабый средний класс, и средние расходы на питание вне дома в 15 раз ниже, чем в США. К тому же, наши соотечественники достаточно консервативны: на

формирование привычки заказывать еду потребуется время. Сегодня это скорее способ отдохнуть и уйти от приготовления пищи дома в конкретный момент времени, чем ежедневное автоматическое действие.

5.4 Анализ современного состояния и перспектив развития отрасли

По данным компании The NPD Group в период с апреля 2017 года по март 2018 года рынок доставки в России вырос на 19%. В первом квартале 2018 количество онлайн заказов выросло на 7,3% по сравнению с аналогичным периодом прошлого года. Уже 14% россиян пробовали покупать еду в интернете. (PriceWaterhouseCoopers). Если говорить о жителях России в целом, то регулярно пользуются доставкой готовой еды на дом 36% населения. Впереди всех — Москва, а крупные провинциальные города отстают на один–два года. Это можно связать с внедрением новых технологий в обычную жизнь людей. Тяжело встретить того, кто ни разу в жизни не заказывал ту же самую пиццу.

Самые быстрорастущие игроки на рынке доставки — рестораны фастфуд и кофейни. С апреля 2017 года по март 2018 года доставка в этих сегментах выросла на 27% и 48% соответственно. При этом в фастфуде лидируют бургеры и курица — их рост составил 77%.

5.5 Планируемая стоимость продукта

Весь процесс разработки продукта можно разделить на две части: единоразовая разработка white label-основы и процесс брендирования под конкретного заказчика.

Вместо платного Github используется бесплатный GitLab. В качестве хостинга для сервера используется Яндекс.Облако.

Основная статья расходов – приобретение лицензий на программное обеспечение, для разработки и проектирование пользовательского интерфейса. Структура расходов представлена в таблице 7 и 8.

Таблица 7 – Расходы на единоразовую разработку

Что	**Количество**	**Цена за единицу, ₽**	**Сумма, ₽**
Лицензия Sketch	1	6500	6500
Яндекс.Облако	12	1400	16800
Человекодни	90	2400	216000
Лицензия IntelliJ IDEA (Приложение под Android)	1	32500	32500
Лицензия IntelliJ IDEA Rider (Для C#)	1	9100	9100
Лицензия для App Store	1	6500	6500
Лицензия для Google Play	1	1600	1600
Итого			289000

В расходах на брендирование под конкретного заказчика играют роль только человекодни, так как лицензии уже куплены на этапе разработки white label:

Таблица 8 – Расходы на брендирование

Что	**Количество**	**Цена за единицу, ₽**	**Сумма, ₽**
Человекодни	20	2400	48000
Итого			48000

5.6 Конкурентные преимущества создаваемого продукта, сравнение технико-экономических характеристик с отечественными и мировыми аналогами

Создаваемая система поддерживает русский язык, имеет более низкую стоимость по сравнению с аналогами и ее функциональные возможности могут быть расширены под конкретного заказчика.

Идея онлайн заказов еды не нова, на рынке существуют готовые решения. Готовые решения можно разделить на три основные группы: агрегаторы продаж еды, white label решения, разработанные для конкретной сети, универсальные системы, адаптируемые для каждой сети.

Агрегаторы продаж — сервисы, позволяющие подключить бизнес к общей сети. Примером таких сервисов являются: Delivery club, Яндекс еда, Zaka Zaka. Данную группу существующих решений нельзя рассматривать как прямых конкурентов, так как они отличаются идеологией и спецификой продаж. Основным отличием является специализированность на доставке продуктов питания, для сетей, у которых отсутствует данный функционал.

White label решения — сервисы, разработанные для конкретной франшизы. Примерами таких решений являются сервисы по продаже еды для сетей KFC, Burger King, Mcdonalds и т.п. Такие решения так же нельзя рассматривать как прямых конкурентов, так как их нельзя использовать для продажи еды сети, не являющейся частью франшизы.

Универсальные решения адаптируемые для каждой сети — основные конкуренты. Примерами таких решений являются R Keeper System, DriveThru, RK Order и прочие. Основным преимуществом подобных решений перед нашей системой является низкая стоимость. R Keeper System базовый пакет — стоимость 1590 рублей в месяц, DriveThru аренда ИС для одной торговой точки — 790 рублей в месяц, RK Order пакет услуг “новичок” — 990 рублей в месяц.

В зависимости от выбранной модели монетизации (ежемесячная плата или процент от заказа) стоимость разрабатываемого продукта будет

варьироваться от 2 до 4 тысяч рублей в месяц для каждой торговой точки. Данная стоимость рассчитывается из стоимости брендирования системы, аренды серверов и срока окупаемости 6 месяцев.

Конкурентные решения навязывают свои БП, решения однотипные и не учитывают специфики деятельности бизнеса, имеют малую гибкость и интегрированность с используемыми программными решениями.

Основными преимуществами перед решениями конкурентов являются гибкость и модульность системы как описано в пункте «Описание продукта как результата НИР».

Система разрабатывается таким образом, что она может взаимодействовать с используемыми технологиями, мультикассами, pos терминалами и т.п. Разрабатываемая система может быть интегрирована с 1С Предприятием и 1С бухгалтерией. Помимо вышеперечисленного наше решение подразумевает разработку персональных мобильных приложений и публикацию в маркетах приложений Google Play и App Store. Также возможна разработка индивидуальных функций по требованию бизнеса. Например, ведение упрощенного бухгалтерского учета, акции и специальные предложения и тому подобное.

5.7 Интеллектуальная собственность

Информационная система и все ее компании является узкоспециализированными, что снимает необходимость регистрации свидетельства ЭВМ, так как копирование ИС становится нецелесообразным.

Архитектура клиент-сервер и идея заказа еды онлайн не являются новыми, поэтому защитить их как ноу-хау не предоставляется возможным.

Также система будет развертываться на собственных серверах, а клиентам и заказчикам будет предоставлен доступ только через интерфейс пользователя.

5.8 Бизнес-модели проекта. Производственный план и план продаж

План проекта: маркетинговое исследование, анализ требований, проектирование интерфейса, разработка архитектуры и модулей, тестирование и отладка системы, запуск ИС, маркетинговая кампания.

Опишем бизнес-модель проекта по модели Остервальда.

Ключевые партнеры. Партнеров нет. Компания действует самостоятельно и независимо.

Ключевые виды деятельности. Основной вид деятельности — разработка программного обеспечения.

Ключевые ресурсы. Материальные ресурсы: 3 рабочих станции с необходимым ПО. Интеллектуальные ресурсы: разработанное ПО, презентации для предпродаж. Человеческие ресурсы: разработчик .NET, мобильный разработчик, веб-разработчик и дизайнер интерфейсов.

Ценностные предложения. Уникальное торговое предложение продукта: ИС позволяет бизнесу сохранить конечных потребителей, которые отказываются от покупки в виду длинной очереди и большого времени приготовления заказа, а также привлечь новый поток студентов.

Взаимоотношения с клиентами. Клиенты продукта — владельцы сетей и конкретных торговых точек. Взаимодействует с ними менеджер по продажам, который будет заниматься предпродажами с помощью пресейл-презентации продукта, заключать договора, а также вести процесс брендирования продукта под клиента.

Если после запуска возникнет необходимость в доработках, то работа будет оцениваться в человекочасах и отдельно согласовываться с клиентом.

Потребительские сегменты. Потребительские сегменты описаны в разделе «Целевые сегменты потребителей создаваемого продукта». Основной потребительский сегмент — рестораны быстрого питания, чьей аудиторией являются студенты, которые хотят поесть в перерыве между парами в вузе.

Структура издержек. Подробная структура издержек описана в разделе «Планируемая стоимость продукта».

Потоки поступления доходов. В зависимости от выбранной модели монетизации (ежемесячная плата или процент от заказа) стоимость разрабатываемого продукта будет варьироваться от 2 до 4 тысяч рублей в месяц для каждой торговой точки. Данная стоимость рассчитывается из стоимости брендирования системы, аренды серверов и срока окупаемости 6 месяцев.

5.9 Стратегия продвижения продукта на рынок

Так как клиенты не массовые — владельцы ресторанов быстрого питания, продвижение будет осуществляться через менеджера по продажам. Необходимо разработать презентацию для продажи с описанием возможностей и стоимости системы, четко описать «боли», которая система закрывает, договориться о встречах с ЛПР на стороне бизнеса, презентовать им решение и заключить договор на брендирование.

После запуска приложения можно договариваться о дополнительных продажах функций клиенту. Например, разработать необходимую функцию или расширить возможности сервиса.

6. СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Целью выпускной квалификационной работы является разработка программного серверного приложения, взаимодействующего с базой данных а так же с тремя клиентскими мобильными приложениями и одним веб приложением. Разрабатываемое серверное приложение обеспечивает взаимодействие между клиентскими приложениями, объединяя их в единую информационную систему. Информационная система позволяет клиентам совершать заказы в ресторанах быстрого питания удаленно, а владельцам сети питания настраивать точки продажи, категории товаров, продукты питания. Так же владельцам сети предоставляется возможность просматривать статистику и отчетность по продажам, анализировать и выявлять наиболее продаваемые товары, категории товаров, торговые точки, приносящие наибольшую прибыль.

Областью применения разрабатываемой информационной системы является область общественного питания и розничных продаж продуктов питания.

Особенность применения большинства информационных заключается в необходимости использовать средства вычислительной техники. Использование средств вычислительной техники, накладывает целый ряд вредных факторов на человека, что впоследствии снижает производительность его труда и может привести к существенным проблемам со здоровьем сотрудника.

В данном разделе рассмотрены вредные и опасные факторы, которые могут возникнуть при использовании средств вычислительной техники. Также раздел включает в себя выявление возможных вредных воздействий на окружающую среду, программ по их снижению и экономии невосполнимых ресурсов и способах защиты в чрезвычайных ситуациях, которые могут возникнуть на рабочем месте.

6.1 Правовые и организационные вопросы обеспечения безопасности

6.1.1 Специальные правовые нормы трудового законодательства

Режим труда и отдыха предусматривает соблюдение определенной длительности непрерывной работы на персональном компьютере (ПК) и перерывов, регламентированных с учетом продолжительности рабочей смены, видов и категории трудовой деятельности.

Вид трудовой деятельности на персональном компьютере в рамках данной работы соответствует группе В – творческая работа в режиме диалога с ПК, категория трудовой деятельности – III (до 6 часов непосредственной работы на ПК).

При 8-часовой рабочей смене и работе на ПК, соответствующей описанным выше критериям необходимо через 1,5- 2,0 часа от начала рабочей смены и через 1,5-2,0 часа после обеденного перерыва устраивать регламентированные перерывы продолжительностью 20 минут каждый или продолжительностью 15 минут через каждый час работы.

Продолжительность непрерывной работы на ПК без регламентированного перерыва не должна превышать 2 часа.

Эффективными являются нерегламентированные перерывы (микропаузы) длительностью 1-3 минуты.

Регламентированные перерывы и микропаузы целесообразно использовать для выполнения комплекса упражнений и гимнастики для глаз, пальцев рук, а также массажа. Комплексы упражнений целесообразно менять через 2-3 недели.

Продолжительность рабочего дня не должна быть меньше указанного времени в договоре, но не больше 40 часов в неделю. Для работников до 16 лет – не более 24 часов в неделю, от 16 до 18 лет и инвалидов I и II группы – не более 35 часов.

При работе в ночное время продолжительность рабочей смены сокращается на один час. К работе в ночную смену не допускаются беременные женщины; работники, не достигшие возраста 18 лет; женщины, имеющие детей в возрасте до трех лет, инвалиды, работники, имеющие детей-инвалидов, а также работники, осуществляющие уход за больными членами их семей в соответствии с медицинским заключением, матери и отцы-одиночки детей до пяти лет.

Организация обязана предоставлять ежегодный отпуск продолжительностью 28 календарных дней. Дополнительные отпуска предоставляются работникам, занятым на работах с вредными или опасными условиями труда, работникам имеющими особый характер работы, работникам 64 с ненормированным рабочим днем и работающим в условиях Крайнего Севера и приравненных к нему местностях.

6.1.2 Организационные мероприятия при компоновке рабочей зоны

Рабочее место должно быть организовано в соответствии с требованиями стандартов, технических условий и (или) методических указаний по безопасности труда. Оно должно удовлетворять следующим требованиям:

- обеспечивать возможность удобного выполнения работ;
- учитывать физическую тяжесть работ;
- учитывать размеры рабочей зоны и необходимость передвижения в ней работающего;
- учитывать технологические особенности процесса выполнения работ.

Невыполнение требований к расположению и компоновке рабочего места может привести к получению работником производственной травмы или развития у него профессионального заболевания. Рабочее место программиста должно соответствовать требованиям СанПин 2.2.2/2.4.1340-03.

Конструкция оборудования и рабочего места при выполнении работ в положении сидя должна обеспечивать оптимальное положение работающего, которое достигается регулированием высоты рабочей поверхности, высоты сидения, оборудованием пространства для размещения ног и высотой подставки для ног. Схемы размещения рабочих мест с персональными компьютерами должны учитывать расстояния между рабочими столами с мониторами: расстояние между боковыми поверхностями мониторов не менее 1,2 м, а расстояние между экраном монитора и тыльной частью другого монитора не менее 2,0 м. Клавиатура должна располагаться на поверхности стола на расстоянии 100-300 мм от края, обращенного к пользователю. Быстрое и точное считывание информации обеспечивается при расположении плоскости экрана 65 ниже уровня глаз пользователя, предпочтительно перпендикулярно к нормальной линии взгляда (нормальная линия взгляда 15 градусов вниз от горизонтали). Рабочие места с компьютерами при выполнении творческой работы, требующей значительного умственного напряжения или высокой концентрации внимания, рекомендуется изолировать друг от друга перегородками высотой 1,5 - 2,0 м.

6.2 Производственная безопасность

Для обеспечения производственной безопасности необходимо проанализировать воздействия на человека вредных и опасных производственных факторов, которые могут возникать при разработке или эксплуатации проекта.

Производственные условия на рабочем месте характеризуются наличием различных опасных и вредных производственных факторов, оказывающих негативное влияние на работников.

Производственный фактор считается вредным, если воздействие этого фактора на работника может привести к его заболеванию. Производственный

фактор считается опасным, если его воздействие на работника может привести к его травме.

Вредные факторы характеризуются потенциальной опасностью для здоровья, в частности способствуют развитию каких-либо заболеваний, приводят к повышенной утомляемости и снижению работоспособности. При этом, вредные факторы проявляются при определенных условиях таких как интенсивность и длительность воздействия. Опасные производственные факторы способны моментально оказать влияние на здоровье работника: привести к травмам, ожогам или к резкому ухудшению здоровья работников в результате отравления или облучения.

В таблице 9 представлены возможные вредные и опасные факторы, возникающие при работе за ПЭВМ.

Таблица 9 – Вредные и опасные факторы, возникающие при работе за ПЭВМ

Наименование видов работ	**Факторы по ГОСТ 12.0.003-2015**	**Нормативные документы**
Вредные факторы		
Работа за персональным компьютером (ПК)	Отклонение показателей микроклимата (температуры и влажности воздуха)	СанПиН 2.2.2/2.4.1340-03 [27] СанПиН 2.2.4.548-96 [28]
	Недостаточная освещенность рабочей зоны	СанПиН 2.2.2/2.4.1340-03
Опасные факторы		
Работа за персональным компьютером (ПК)	Опасность поражения электрическим током	ГОСТ 12.1.038–82 [29]
	Пожаровзрывоопасность	ФЗ от 22.07.2008 №123-ФЗ [30]

6.2.1 Анализ вредных и опасных факторов

6.2.1.1 Отклонение показателей микроклимата в помещении

Одним из необходимых благоприятных условий труда является обеспечение в помещениях нормальных условий микроклимата, оказывающих существенное влияние на тепловое самочувствие человека. Микроклимат в производственных помещениях, зависит от особенностей технологического

процесса, а также внешних условий (категории работ, периода года, условий вентиляции и отопления).

К параметрам, характеризующим микроклимат в производственных помещениях, относятся:

- температура воздуха (t, °С);
- температура поверхностей (t, °С);
- относительная влажность воздуха (φ, %);
- скорость движения воздуха (υ, м/с);
- интенсивность теплового облучения (I, Вт/м^2).

В производственных помещениях для работы с ПЭВМ происходит постоянное выделение тепла самой вычислительной техникой, вспомогательными приборами и средствами освещения. Поскольку оператор расположен в непосредственной близости с источниками выделения тепла, то данный фактор является одним из важнейших вредных факторов производственной среды оператора ПЭВМ, а высокая температура воздуха способствует быстрому перегреву организма и быстрой утомляемости [31].

Влажность оказывает большое влияние на терморегуляцию организма. Так, например, высокие показатели относительной влажности (более 85 %) затрудняют терморегуляцию снижая возможность испарения пота, низкие показатели влажности (менее 20 %) вызывают пересыхание слизистых оболочек человека [32].

Работа программиста относиться к категории Iа, которые производятся сидя и сопровождаются незначительными физическим напряжением. Интенсивность энерготрат организма для данной категории работ составляет до 120 ккал/ч (до 139 Вт).

Оптимальные значения показателей микроклимата на рабочих местах производственных помещений согласно СанПиН 2.2.4.548-96 для категории работ Iа представлены в таблице 9.

Таблица 9 – Оптимальные величины показателей микроклимата на рабочих местах производственных помещений

Период года	**Категория работ**	**Температура воздуха, °С**	**Относительная влажность воздуха, %**	**Скорость движения воздуха, м/с**
Холодный	Ia	22 – 24	60 – 40	0,1
Теплый	Ia	23 – 25	60 – 40	0,1

Допустимые микроклиматические условия не вызывают повреждений или нарушений состояния здоровья, но могут приводить к возникновению общих и локальных ощущений теплового дискомфорта, напряжению механизмов терморегуляции, ухудшению самочувствия и понижению работоспособности.

В таблице 11 приведены допустимые величины показателей микроклимата на рабочих местах производственных помещений согласно СанПиН 2.2.4.548-96 для категории работ Ia. [33]

Таблица 11 – Допустимые величины показателей микроклимата на рабочих местах производственных помещений

Период года	**Категория работ**	**Температура воздуха, °С**		**Относительная влажность воздуха, %**	**Скорость движения воздуха, м/с**	
		Диапазон ниже оптимальных величин	Диапазон выше оптимальных величин		Для диапазона температур воздуха ниже оптимальных величин, не более	Для диапазона температур воздуха выше оптимальных величин, не более
Холодный	Ia	20,0 – 21,9	24,1-25,0	15 – 75	0,1	0,1
Теплый	Ia	21,0-22,9	25,1-28,0	15 – 75	0,1	0,2

Согласно требованиям СанПиН 2.2.2/2.4.1340–03, в кабинете поддерживается температура равная 19–20 С°, при относительной влажности в 55–58%. [34] Чтобы добиться этого, необходимо проводить в помещении ежедневную влажную уборку и систематическое проветривание.

6.2.1.2 Недостаточная освещенность рабочей зоны

Недостаточная освещенность рабочей зоны – вредный производственный фактор, который регламентируется СП 52.13330.2011.

Недостаточный уровень освещенности в помещении приводит к снижению остроты зрения, головным болям, снижению концентрации внимания и, как следствие, к ухудшению производительности труда. Причиной недостаточной освещенности являются недостаточность естественного освещения, недостаточность искусственного освещения, пониженная контрастность.

В рабочем помещении должны присутствовать естественное и искусственное освещение. Коэффициент естественного освещения должен быть не менее 1,2%. Согласно СанПиН 2.2.2/2.4.1340-03 освещенность на поверхности рабочего стола в зоне размещения документа должна быть 300 – 500 лк, что может достигаться установкой местного освещения, не создающего бликов на поверхности экрана. Освещенность поверхности экрана не должна превышать 300 лк. Яркость светящихся поверхностей (окон, светильников), находящихся в поле зрения должна быть не более 200 кд/м^2. Для источников искусственного освещения следует применять люминесцентные лампы типа ЛБ и компактные люминесцентные лампы (КЛЛ). Коэффициент пульсации при работе с компьютером не должен превышать 5%.

Помимо обеспечения достаточного уровня освещения, для минимизации данного вредного фактора следует ограничить отраженную блескость на рабочих поверхностях (экран, стол, клавиатура) за счет правильного выбора и расположения светильников, яркость бликов на экране не должна превышать 40 кд/м^2. Светильники местного освещения должны иметь непросвечивающий отражатель.

6.2.2 Опасные производственные факторы

6.2.2.1 Опасность поражения электрическим током

Поражение электрическим током является опасным производственным фактором и, поскольку оператор ПЭВМ имеет дело с электрооборудованием, то вопросам электробезопасности на его рабочем месте должно уделяться много внимания.

Опасность поражения человека электрическим током оценивается величиной тока I (А), проходящего через его тело, или напряжением прикосновения U (В). Степень опасного воздействия на человека электрического тока зависит от рода и величины напряжения тока, частоты электрического тока, пути тока через тело человека, продолжительности его воздействия на организм человека, а также условий внешней среды.

Электрический ток, протекая через тело человека, производит термическое, механическое и световое воздействие – электролитическое разложение жидкости (в том числе и крови), судорожное сокращение мышц, разрыв тканей и поражение глаз.

Работа с ПЭВМ является опасной с точки зрения поражения током, так как практически во всех частях компьютера течет электрический ток. Поражение электрическим током при работе в ПЭВМ возможно при наличии оголенных участков на кабеле, нарушении изоляции распределительных устройств и от токоведущих частей компьютера в случае их пробоя и нарушении изоляции, при работе с ПЭВМ во влажной одежде и влажными руками.

Помещение, где расположено рабочее место оператора ПЭВМ, относится к помещениям без повышенной опасности ввиду отсутствия следующих факторов: сырость, токопроводящая пыль, токопроводящие полы, высокая температура, возможность одновременного прикосновения человека к имеющим соединение с землей металлоконструкциям зданий,

технологическим аппаратам, механизмам и металлическим корпусам электрооборудования.

К мероприятиям по предотвращению возможности поражения электрическим током относятся:

- При производстве монтажных работ необходимо использовать только исправный инструмент, аттестованный службой КИПиА;
- С целью защиты от поражения электрическим током, возникающим между корпусом приборов и инструментом при пробое сетевого напряжения на корпус, корпуса приборов и инструментов должны быть заземлены;
- При включенном сетевом напряжении работы на задней панели должны быть запрещены;
- Все работы по устранению неисправностей должен производить квалифицированный персонал;
- Необходимо постоянно следить за исправностью электропроводки [35].

Согласно ГОСТ 12.1.038-82 на рабочем месте программиста допускаются уровни напряжений прикосновения и токов, представленные в таблице 12. [36]

Таблица 12 – Предельно допустимые напряжения прикосновения и токи

Род тока	**Напряжение прикосновения, В**	**Ток, м/А**
Переменный, 50 Гц	Не более 2,0	Не более 0,3
Постоянный	Не более 8,0	Не более 1,0

6.2.2.2 Пожаровзрывобезопасность

Пожарная безопасность представляет собой единый комплекс организационных, технических, режимных и эксплуатационных мероприятий по предупреждению пожаров и взрывов.

В помещениях с компьютерами повышен риск возникновения пожара из-за присутствия множества факторов: наличие большого количества

электронных схем, устройств электропитания, устройств кондиционирования воздуха; возможные неисправности электрооборудования, освещения, или неправильная их эксплуатация может послужить причиной пожара.

Для устранения возможных причин возникновения пожаров необходимо проводить следующие мероприятия:

- организационные мероприятия:
  - противопожарный инструктаж обслуживающего персонала;
  - обучение персонала техники безопасности;
  - разработка инструкций, плакатов, планов эвакуации;
- эксплуатационные мероприятия:
  - соблюдение эксплуатационных норм оборудования;
  - выбор и использование современных автоматических средств тушения пожаров;
- технические мероприятия:
  - профилактический осмотр и ремонт оборудования;
  - соблюдения противопожарных мероприятий при устройстве электропроводок, оборудования, систем отопления, вентиляции и освещения

6.3 Экологическая безопасность

6.3.1 Анализ воздействия продукта на окружающую среду

Вследствие развития научно-технического прогресса постоянно увеличивается возможность воздействия на окружающую среду, создаются предпосылки для возникновения экологических кризисов.

Увеличение количества компьютерных систем, внедряемых в производственную сферу, приводит к увеличению объема потребляемой электроэнергии, что влечет за собой увеличение мощностей электростанций и их количества. И то, и другое содействует нарушению экологической обстановки и, выбросы со станций оказывают существенное влияние на атмосферу.

Основным фактором, оказывающим негативное влияние на гидросферу и литосферу, является образование отходов. В помещении образуются следующие виды отходов: бумага (макулатура), отходы от продуктов питания и личной гигиены (упаковка, органические отходы), отходы от канцелярских принадлежностей, отходы от офисной техники (использованные картриджи, упаковка, неисправные компоненты), лампы.

6.3.2 Решения по обеспечению экологической безопасности

Наиболее активной формой защиты окружающей среды от вредного воздействия выбросов промышленных предприятий является полный переход к безотходным и малоотходным технологиям и производствам. Это потребует решения целого комплекса сложных технологических, конструкторских и организационных задач, основанных на использовании новейших научно-технических достижений.

Необходимо стремиться к снижению энергопотребления, то есть разрабатывать и внедрять системы с малым энергопотреблением. Следует использовать современные ЭВМ с режимом пониженного потребления электроэнергии при длительном простое.

Для каждого вида отходов должны применяться свои методы переработки и утилизации, недопустимо организовывать свалки мусора, проводить самостоятельно утилизацию. Все отходы следует собирать, сортировать и направлять на переработку в соответствующие организации.

В настоящее время существует ряд способов хранения и переработки твердых бытовых отходов, а именно: предварительная сортировка, санитарная земляная засыпка, сжигание, биотермическое компостирование, низкотемпературный пиролиз, высокотемпературный пиролиз.

6.4 Безопасность в чрезвычайных ситуациях

6.4.1 Перечень возможных ЧС при разработке и эксплуатации научно-исследовательского проекта

Чрезвычайные ситуации, которые могут возникнуть при разработке и эксплуатации проектируемого решения:

- техногенные (взрывы, пожары, обрушение помещений, аварии на системах жизнеобеспечения);
- природные (наводнения, ураганы, бури, природные пожары);
- биологические (эпидемии, пандемии);
- антропогенные (война, терроризм).

Общие правила поведения при чрезвычайных ситуациях:

1) Не паниковать и не поддаваться панике. Призывать окружающих к спокойствию.

2) По возможности немедленно позвонить по телефону «01», сообщить что случилось, указать точный адрес места происшествия, назватьсвою фамилию и номер своего телефона.

3) Включить устройства передачи звука (радио, телевизор), а также прослушать информацию, передаваемую через уличные громкоговорители и громкоговорящие устройства. В речевом сообщении будут озвучены основные рекомендации и правила поведения.

4) Выполнять рекомендации специалистов (сотрудников полиции, медицинских работников, пожарных, спасателей).

5) Не создавать условия, которые препятствуют и затрудняют действия сотрудников полиции, медицинских работников, спасателей, пожарных.

Наиболее характерной для объекта, где размещаются рабочие помещения, оборудованные электронно-вычислительными машинами, чрезвычайной ситуацией является пожар.

Причинами возникновения данного вида ЧС могут являться:

- возникновением короткого замыкания в электропроводке;

- возгоранием устройств вычислительной техники из-за неисправности аппаратуры;
- возгоранием устройств искусственного освещения;
- возгоранием мебели по причине нарушения правил пожарной безопасности, а также неправильного использования дополнительных бытовых электроприборов и электроустановок.

Помещение для работы операторов ПЭВМ по системе классификации категорий помещений по взрывопожарной и пожарной опасности относится к категории Д (из 5-ти категорий А, Б, В1-В4, Г, Д), т.к. относится к помещениям с негорючими веществами и материалами в холодном состоянии.

6.4.2 Разработка действий в результате возникшей ЧС и меры по ликвидации ее последствий

Пожарная безопасность подразумевает надлежащее состояние объекта с исключением возможности возникновения очага возгорания (пожара) и его распространения в пространстве. Обеспечение пожарной безопасности — приоритетная задача для любого предприятия. Создание системы защиты регламентировано законом и нормативными документами различных ведомств.

Каждый сотрудник организации должен быть ознакомлен с инструкцией по пожарной безопасности, пройти инструктаж по технике безопасности и строго соблюдать его.

Запрещается использовать электроприборы в условиях, не соответствующих требованиям инструкций изготовителей, или имеющие неисправности, которые в соответствии с инструкцией по эксплуатации могут привести к пожару, а также эксплуатировать электропровода и кабели с поврежденной или потерявшей защитные свойства изоляцией. Электроустановки и бытовые электроприборы в помещениях по окончании рабочего времени должны быть обесточены (вилки должны быть вынуты из розеток). Под напряжением должны оставаться дежурное освещение и

пожарная сигнализация. Недопустимо хранение легковоспламеняющихся, горючих и взрывчатых веществ, использование открытого огня в помещениях офиса.

Перед уходом из служебного помещения работник обязан провести его осмотр, закрыть окна, и убедиться в том, что в помещении отсутствуют источники возможного возгорания, все электроприборы отключены и выключено освещение. С периодичностью не реже одного раза в три года необходимо проводить замеры сопротивления изоляции токоведущих частей силового и осветительного оборудования.

Работник при обнаружении пожара или признаков горения (задымление, запах гари, повышение температуры и т.п.) должен:

- немедленно прекратить работу и вызвать пожарную охрану по телефону «01», сообщив при этом адрес, место возникновения пожара и свою фамилию;
- принять по возможности меры по эвакуации людей и материальных ценностей;
- отключить от сети закрепленное за ним электрооборудование; – Приступить к тушению пожара имеющимися средствами пожаротушения;
- сообщить непосредственному или вышестоящему начальнику и оповестить окружающих сотрудников;
- при общем сигнале опасности покинуть здание согласно «Плану эвакуации людей при пожаре и других ЧС».

Для тушения пожара необходимо применять ручные углекислотные огнетушители (типа ОУ-2, ОУ-5), находящиеся в помещениях офиса, и пожарный кран внутреннего противопожарного водопровода. Они предназначены для тушения начальных возгораний различных веществ и материалов, за исключением веществ, горение которых происходит без доступа воздуха.

ЗАКЛЮЧЕНИЕ

В результате выполнения выпускной квалификационной работы была достигнута ее основная цель — спроектирован и разработан серверный компонент информационной системы для сетей ресторанов быстрого питания.

Архитектура была разработана и реализована с использованием современных технологий, используемых в enterprise решениях. Так же использование docker-compose позволило упростить процессы развертывания окружения на персональных рабочих местах до трех команд. Так же были настроены процессы CI/CD, так как параллельно с разработкой серверного компонента велись разработки клиентских компонентов.

Главный серверный модуль был реализован с использованием платформы ASP .Net Core 2.2. При проектировании структуры данных использовался подход code first. При разработке приложения были рассмотрены различные подходы сериализации данных. Созданы собственные реализации объектов связывания данных. Разработан собственный механизм создания JWT токенов и подмешивания данных в контекст запроса с использованием ПО промежуточного слоя. Архитектура приложения была выстроена основываясь на существующие решения, используемые в бизнесе. Были задействованы механизмы обратного контроля и внедрения зависимостей.

Финансовая часть ВКР была защищена как стартап проект. После защиты поступило предложение о финансирование проекта со стороны организации «Опора России».

В дальнейшем планируется отладить взаимодействие с клиентскими приложениями, наполнить базу демо данными, провести несколько презентаций для представителей сетей быстрого питания.

CONCLUSION

As a result of the final qualifying work, its main goal was achieved. The server component of the information system for fast-food restaurants — was designed and developed.

The architecture was developed and implemented using modern technologies used in enterprise solutions. Also, using docker-compose technology allowed us to simplify the process of deploying the environment on personal workplaces of up to three commands. The CI / CD processes were also set up, since in parallel with the development of the server component, the development of client components was carried out.

The main server module was implemented using the ASP .Net Core 2.2 platform. During designing the data structure, the code first approach was used. During developing the application, various approaches to data serialization were considered. Created own implementations of data binding objects. A custom mechanism has been developed for creating JWT tokens and mixing data into the request context using middleware. The application architecture was built based on existing solutions used in business. Were involved in the mechanisms of reverse control and dependency injection.

The financial part of the final qualifying work was protected as a startup project. After the defense, a proposal was received to finance the project from the organization «Opora Rossii».

In the future, it is planned to debug interaction with client applications, fill the database with demo data, and hold several presentations for representatives of fast-food restaurants.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Use Cases // Школа аналитиков. URL: https://systems.education/use-case/, свободный (дата обращения 23.02.2019)

2. Алиса — голосовой помощник от компании Яндекс // Алиса. URL: https://alice.yandex.ru/, свободный (дата обращения 25.02.2019)

3. Что такое IP65, степени защиты IP, пылевлагозащищенность и ГОСТ 14254-96 // Аэротема. URL: http://aerotema.ru/articles/stepeni-ip-gost-14254-96/, свободный (дата обращения 28.05.2019)

4. Планшет Lenovo Tab 3 // Леново. URL: https://www.lenovo.com/ru/ru/tablets/android-tablets/tab3-series/Tab-3-7-/p/ZZITZTATB3F, свободный (дата обращения 18.02.2019)

5. Что такое браузер - Компьютер для новичков // Компьютеры для начинающих. URL: https://beginpc.ru/internet/chto-takoe-browser, свободный (дата обращения 17.04.2019)

6. Простым языком об HTTP / Хабр // Хабр. URL: https://habr.com/ru/post/215117/, свободный (дата обращения 09.05.2019)

7. Enterprise Container Platform | Docker // Докер. URL: https://www.docker.com/, свободный (дата обращения 11.02.2019)

8. JSON API – работаем по спецификации / Блог компании Конференции Олега Бунина (Онтико) / Хабр // Хабр. URL: https://habr.com/ru/company/oleg-bunin/blog/433322/, свободный (дата обращения 26.05.2019)

9. Socket.IO // socket.io. URL: https://socket.io/, свободный (дата обращения 30.04.2019)

10. ORM или как забыть о проектировании БД / Хабр // Хабр. URL: https://habr.com/ru/post/237889/, свободный (дата обращения 28.03.2019)

11. Что такое реляционная база данных? – Amazon Web Services;(AWS) // Амазон. URL: https://aws.amazon.com/ru/relational-database/, свободный (дата обращения 08.03.2019)

12. MySQL и MongoDB — когда и что лучше использовать / Хабр // Хабр. URL: https://habr.com/ru/post/322532/, свободный (дата обращения 30.04.2019)

13. Разработка баз данных с Code First / Хабр // Хабр. URL: https://habr.com/ru/post/234827/, свободный (дата обращения 30.04.2019)

14. DTO vs POCO vs Value Object / Хабр // Хабр. URL: https://habr.com/ru/post/268371/, свободный (дата обращения 05.05.2019)

15. About Enterprise Programming | The Ilardi Family // Иларди. URL: https://blog.ilardi.com/about-enterprise-programming, свободный (дата обращения 30.03.2019)

16. Введение | Vuex // Vuex. URL: https://vuex.vuejs.org/ru/guide/, свободный (дата обращения 30.03.2019)

17. The Go Programming Language // GoLang. URL: https://golang.org/, свободный (дата обращения 30.04.2019)

18. AMQP по-русски / Хабр // Хабр. URL: https://habr.com/ru/post/62502/, свободный (дата обращения 05.02.2019)

19. CI/CD: принципы, внедрение, инструменты – Southbridge – Medium // Medium. URL: https://medium.com/southbridge/ci-cd, свободный (дата обращения 19.02.2019)

20. Docker Compose | Docker Documentation // Docker. URL: https://docs.docker.com/compose/, свободный (дата обращения 30.04.2019)

21. Teach, Learn, and Make with Raspberry Pi – Raspberry Pi // Raspberry. URL: https://www.raspberrypi.org/, свободный (дата обращения 18.02.2019)

22. Яндекс.Облако // Yandex Cloud. URL: https://cloud.yandex.ru/, свободный (дата обращения 05.05.2019)

23. Postman | API Development Environment // Postman. URL: https://www.getpostman.com/, свободный (дата обращения 20.02.2019)

24. Как объяснить бабушке, что такое Agile за 15 минут с картинками / Блог компании Edison / Хабр // Habr. URL:

https://habr.com/ru/company/edison/blog/313410/, свободный (дата обращения 02.04.2019)

25. Пять простых шагов для понимания JSON Web Tokens (JWT) / Хабр // Habr. URL: https://habr.com/ru/post/340146/, свободный (дата обращения 02.05.2019)

26. Json.NET - Newtonsoft // Newton Soft. URL: https://www.newtonsoft.com/json, свободный (дата обращения 25.04.2019)

27. Слизистые оболочки. Строение тела человека, органы, системы тела человека, слизистые оболочки, клетки и хромосомы, обмен веществ, гомеостаз, анатомия человека. // Анатомический атлас. URL: http://ru5ru.ru/atlas.php?id_article=38, свободный (дата обращения 25.05.2019)

28. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 27.12.2018)

29. ГОСТ 12.2.032-78 ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования.

30. ГОСТ 12.0.003-2015 ССБТ. Опасные и вредные производственные факторы. Классификация.

31. ГОСТ 12.1.028-82 ССБТ. Электробезопасность. Предельно допустимые значения напряжений прикосновения и токов

32. Правила устройства электроустановок ПУЭ.

33. СанПиН 2.2.4.548-96. Гигиенические требования к микроклимату производственных помещений.

34. СанПиН 2.2.1/2.1.1.1278-03. Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий.

35. СН 2.2.4/2.1.8.562-96. Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории застройки.

36. СанПиН 2.2.4.3359-16. Санитарно-эпидемиологические требования к физическим факторам на рабочих местах.