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: Инженерная школа информационных технологий и робототехники (ИШИТР)
Направление подготовки: 09.03.04 «Программная инженерия»
ООП/ОПОП: Разработка программно-информационных систем
Отделение школы: Отделение информационных технологий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Тема работы
Разработка системы "HR-automation"

УДК: 004.4:005.95

Студенты

Группа	ФИО	Подпись	Дата
8К91	Байделюк Елизавета Андреевна		
8К91	Когут Арина Евгеньевна		
8К91	Сидоркин Александр Андреевич		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Чердынцев Евгений Сергеевич	к.т.н.		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Гасанов Магеррам Али оглы	д.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Мезенцева Ирина Леонидовна			

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Чердынцев Евгений Сергеевич	к.т.н.		

Томск – 2023 г.

Планируемые результаты освоения ООП по направлению 09.03.04 «Программная инженерия»

Код компетенции	Наименование компетенции
УК(У)-1	Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач
УК(У)-2	Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений
УК(У)-3	Способен осуществлять социальное взаимодействие и реализовывать свою роль в команде
УК(У)-4	Способен осуществлять деловую коммуникацию в устной и письменной формах на государственном языке Российской Федерации и иностранном(-ых) языке(-ах)
УК(У)-5	Способен воспринимать межкультурное разнообразие общества в социально-историческом, этическом и философском контекстах
УК(У)-6	Способен управлять своим временем, выстраивать и реализовывать траекторию саморазвития на основе принципов образования в течение всей жизни
УК(У)-7	Способен поддерживать должный уровень физической подготовленности для обеспечения полноценной социальной и профессиональной деятельности
УК(У)-8	Способен создавать и поддерживать безопасные условия жизнедеятельности, в том числе при возникновении чрезвычайных ситуаций
ОПК(У)-1	Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности
ОПК(У)-2	Способен использовать современные информационные технологии и программные средства, в том числе отечественного производства, при решении задач профессиональной деятельности
ОПК(У)-3	Способен решать стандартные задачи профессиональной деятельности на основе информационной и библиографической культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности
ОПК(У)-4	Способен участвовать в разработке стандартов, норм и правил, а также технической документации, связанной с профессиональной деятельностью

Код компетенции	Наименование компетенции
ОПК(У)-5	Способен инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем
ОПК(У)-6	Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов
ОПК(У)-7	Способен применять в практической деятельности основные концепции, принципы, теории и факты, связанные с информатикой
ОПК(У)-8	Способен осуществлять поиск, хранение, обработку и анализ информации из различных источников и баз данных, представлять ее в требуемом формате с использованием информационных, компьютерных и сетевых технологий.
ПК(У)-1	Владение навыками разработки требований и проектирования программного обеспечения
ПК(У)-2	Владение навыками разработки документов и стратегии тестирования программного обеспечения
ПК(У)-3	Владение навыками моделирования, анализа и использования формальных методов конструирования программного обеспечения
ПК(У)-4	Владение навыками использования операционных систем, сетевых технологий, средств разработки программного интерфейса, применения языков и методов формальных спецификаций, систем управления базами данных
ПК(У)-5	Владение концепциями и атрибутами качества программного обеспечения (надежности, безопасности, удобства использования), в том числе роли людей, процессов, методов, инструментов и технологий обеспечения качества

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: Инженерная школа информационных технологий и робототехники (ИШИТР)
Направление подготовки: 09.03.04 «Программная инженерия»
Отделение школы: Отделение информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП/ОПОП
_____ _____ Чердынцев Е.С.
(Подпись) (Дата) (ФИО)

ЗАДАНИЕ

на выполнение выпускной квалификационной работы

Обучающиеся:

Группа	ФИО
8К91	Байделюк Елизавета Андреевна
8К91	Когут Арина Евгеньевна
8К91	Сидоркин Александр Андреевич

Тема работы:

Разработка системы "HR-automation"	
Утверждена приказом директора (дата, номер)	№ 102-28_с от 12.04.2023

Срок сдачи студентом выполненной работы:	13.06.2023

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.).*	Работа направлена на разработку системы «HR-automation»

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 разделов, подлежащих разработке; заключение по работе).*	Изучение рынка приложений с нужным функционалом. Проектирование и реализация системы
Перечень графического материала *(с точным указанием обязательных чертежей)*	UML-диаграммы, описывающие проектируемую систему и ее функционал. Макеты экранов веб-приложения

Консультанты по разделам выпускной квалификационной работы
(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Гасанов Магеррам Али оглы
Социальная ответственность	Мезенцева Ирина Леонидовна

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	06.02.2023

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Чердынцев Евгений Сергеевич	к.т.н.		

Задание приняли к исполнению студенты:

Группа	ФИО	Подпись	Дата
8К91	Байделюк Елизавета Андреевна		
8К91	Когут Арина Евгеньевна		
8К91	Сидоркин Александр Андреевич		

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: Инженерная школа информационных технологий и робототехники (ИШИТР)
Направление подготовки: 09.03.04 «Программная инженерия»
Уровень образования: бакалавр
Отделение школы: Отделение информационных технологий
Период выполнения: весенний семестр 2022/2023 учебного года

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Обучающийся:

Группа	ФИО
8К91	Байделюк Елизавета Андреевна
8К91	Когут Арина Евгеньевна
8К91	Сидоркин Александр Андреевич

Тема работы:

Разработка системы "HR-automation»

Срок сдачи обучающимся выполненной работы:	08.06.2023

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
16.06.2023	Исследование предметной области	25
16.06.2023	Проектирование системы	25
16.06.2023	Реализация системы	20
16.06.2023	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
16.06.2023	Социальная ответственность	15

СОСТАВИЛ:
Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Чердынцев Евгений Сергеевич	к.т.н.		

Консультант (при наличии)

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Мезенцева Ирина Леонидовна			
Профессор ОСГН ШБИП	Гасанов Магеррам Али оглы	д.э.н.		

СОГЛАСОВАНО:
Руководитель ООП/ОПОП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ ИШИТР ТПУ	Чердынцев Евгений Сергеевич	к.т.н.		

Обучающийся

Группа	ФИО	Подпись	Дата
8К91	Байделюк Елизавета Андреевна		
8К91	Когут Арина Евгеньевна		
8К91	Сидоркин Александр Андреевич		

Реферат

Выпускная квалификационная работа содержит: 156 страниц, 56 рисунков, 25 таблиц, 17 источников, 9 приложений

Ключевые слова: информационная система, мобильное приложение, веб-приложение, сервер.

Объектом исследования является автоматизация процессов взаимодействия между сотрудниками офиса Тинькофф Центр Разработки (ТЦР).

Цель работы – разработка системы, позволяющей автоматизировать процессы взаимодействий между сотрудниками.

Область применения системы не ограничивается одним офисом, система внедряема и в другие среды, потребности которых совпадают с потребностями заказчика.

В ходе работы была спроектирована и реализована информационная система, состоящая из мобильного приложения, веб-приложения и сервера.

Степень внедрения: не внедрялось

Экономическая эффективность/значимость работы заключается в экономии времени сотрудников, затрачиваемого на взаимодействия друг с другом.

В будущем планируется завершение работ по реализации (тестирование, исправление ошибок).

Список терминов и сокращений

1. Android – это операционная система для смартфонов, планшетов, электронных книг, цифровых проигрывателей, наручных часов, и т.п.;
2. HR – специалист по работе с кадрами;
3. UML – язык графического описания для объектного моделирования в области разработки программного обеспечения, для моделирования бизнес-процессов, системного проектирования и отображения организационных структур;
4. CRUD – четыре базовые функции, используемые при работе с базами данных: создание (create), чтение (read), модификация (update), удаление (delete);
5. API – описание способов взаимодействия одной компьютерной программы с другими (Application Programming Interface);
6. ТЦР – Тинькофф центр разработки;
7. DevOps – специалист, отвечающий за синхронизацию, интегрирование, и автоматизацию процесса разработки программного обеспечения в течение всего жизненного цикла;
8. CRM-система – сервис для автоматизации бизнес-процессов и управления бизнесом;
9. Low-code – метод разработки с использованием графического интерфейса, но с написанием небольшого количества кода;
10. SDK – набор инструментов для разработки под определенные платформы;
11. Интерсептор – механизм, позволяющий встроить в заголовки каждого запроса необходимые данные;
12. Аутентификатор – интерфейс библиотеки okhttp3, реализация которого позволит обновлять ключ доступа.

Оглавление

ВВЕДЕНИЕ 14
ГЛАВА 1. Исследование предметной области 15
1.1 Описание предметной области 15
1.1.1 Общая информация 15
1.1.2 Аналогичные решения 24
1.2 Выбор и обоснование состава программного обеспечения 29
1.2.1 Веб-часть системы 29
1.2.2 Серверная часть системы 40
1.2.3 Мобильное приложение 44
ГЛАВА 2. Проектирование системы 49
2.1 Архитектура системы 49
2.1.1 Общая архитектура 49
2.1.2 DFD 50
2.1.3 EPC 50
2.1.4 Диаграмма вариантов использования 51
2.2 Проектирование административной панели 52
2.2.1 Введение 52
2.2.2 Архитектура 52
2.2.3 Пользовательский интерфейс 55
2.2.4 Функциональность 60
2.2.5 Безопасность и конфиденциальность 60
2.3 Проектирование серверной части 61
2.3.1 Диаграмма развертывания 61
2.3.2 Логическая модель базы данных 62
2.3.3 Проектирование системы аутентификации 64
2.4 Проектирование мобильного приложения 65
2.4.1 Введение 65
2.4.2 Архитектура 65
ГЛАВА 3. Реализация 72

3.1 Реализация административной панели 72
3.1.1 Раздел «Авторизация» 72
3.1.2 Раздел «Сотрудники» 73
3.1.3 Раздел «FAQ» 74
3.1.4 Раздел «Продукты» 75
3.1.5 Раздел «Рестораны» 77
3.1.6 Раздел «События» 78
3.1.7 Итоги 79
3.2 Реализация серверной части 79
3.2.1 Создание проекта 79
3.2.2 Аутентификация и авторизация пользователей 80
3.2.3 Раздел «Сотрудники» 80
3.2.4 Развертывание проекта и создание документации 81
3.2.5 Раздел «FAQ» 82
3.2.6 Раздел «Продукты» 82
3.2.7 Раздел «Рестораны» 83
3.2.8 Раздел «События» 84
3.3 Реализация мобильного приложения 85
3.3.1 Каркас приложения 85
3.3.2 Разработка представлений 86
3.4 Результаты разработки 96
ГЛАВА 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение 97
Задание для раздела 97
4.1.1 Краткое описание 99
4.2 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение 100
4.2.1 Предпроектный анализ 100
4.2.2 Определение возможных альтернатив проведения научного исследования 106

4.2.3 Планирование работ по научно-техническому исследованию 108
4.2.4 Бюджет научно-технического исследования (НТИ) 112
4.2.5 Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования 120
4.2.6 Вывод 122
ГЛАВА 5. Социальная ответственность 124
Задание для раздела 124
5.1 Социальная ответственность 128
5.1.1 Введение 128
5.1.2 Правовые и организационные вопросы обеспечения безопасности при разработке проектного решения 128
5.1.3 Производственная безопасность 130
5.1.4 Экологическая безопасность 134
5.1.5 Безопасность в чрезвычайных ситуациях 135
5.1.6 Вывод по разделу 137
ЗАКЛЮЧЕНИЕ 139
СПИСОК ЛИТЕРАТУРЫ 140
ПРИЛОЖЕНИЕ А (справочное) Функциональные требования для мобильного приложения 142
ПРИЛОЖЕНИЕ Б (справочное) Функциональные требования для веб-приложения 145
ПРИЛОЖЕНИЕ В (справочное) DFD диаграмма 149
ПРИЛОЖЕНИЕ Г (справочное) EPC диаграмма 150
ПРИЛОЖЕНИЕ Д (справочное) Диаграмма вариантов использования мобильного приложения 151
ПРИЛОЖЕНИЕ Е (справочное) Диаграмма вариантов использования административного приложения 152
ПРИЛОЖЕНИЕ Ж (справочное) Диаграмма компонентов мобильного приложения 153

ПРИЛОЖЕНИЕ И (обязательное) Описание работ, выполненных совместно всеми участниками групповой/комплексной ВКР 154
ПРИЛОЖЕНИЕ К (справочное) Результаты разработки 156

ВВЕДЕНИЕ

Успешные компании постоянно развиваются, чтобы увеличить свою прибыль. Этот процесс, как правило, сопровождается ростом численности сотрудников. Расширение штата влечет за собой появление новых правил и распорядков, развитие внутренней культуры в офисе компании. Повышение привлекательности и комфорта компании для сотрудников также играет немаловажную роль. Вышеперечисленное поднимает вопрос об эффективных способах адаптации сотрудников и организации взаимодействий между ними.

В большинстве организаций за поиск и адаптацию новых сотрудников отвечает HR-специалист. Чем больше компания и чем сложнее устройство внутренней среды офиса, тем больше труда необходимо для покрытия данной задачи. Проблему загрузки можно решить двумя способами: увеличивать число таких специалистов или автоматизировать выполнение часто повторяющихся задач и решения проблем, не требующих в таком случае вмешательства специалиста по работе с кадрами. Найм нового HR означает увеличение расходов, а также способствует возникновению проблем, связанных с распределением задач в рамках одного офиса, если количество сотрудников в нем растет и возникает необходимость в данном распределении.

Современные средства аппаратного и программного обеспечения предоставляют большое количество возможностей, в особенности, для автоматизации процессов, в которых ранее участвовал человеческий труд, который при выполнении простых повторяющихся задач ограничивает потенциал компании, а также подвержен человеческому фактору. Таким образом, некоторая информационная система может стать решением к уменьшению рутинной работы, выполняемой специалистом по работе с кадрами, и увеличить привлекательность компании, как работодателя, показывая сотрудникам, что эта система существует именно для них.

ГЛАВА 1. Исследование предметной области

1.1 Описание предметной области

1.1.1 Общая информация

1.1.1.1 Идея проекта

Тинькофф центр разработки (ТЦР) – это часть инфраструктуры Тинькофф, занимающаяся разработкой IT-проектов. Команда компании быстро растет, часто пополняется новыми сотрудниками, а значит возрастает нагрузка на HR-специалистов и менеджеров. Когда В ТЦР приходят новые сотрудники, они спрашивают однотипные вопросы у HR-а, менеджера или коллег, отвлекая их от другой работы. Такой подход неэффективен не только по затраченному времени, но и по достоверности – коллеги могут неправильно дать ответы новому работнику вследствие влияния человеческого фактора.

В коллективе часто возникают такие задачи, как оперативно узнать день рождения коллеги, контакты devOps-а или другие данные о сотруднике офиса. Для этого планируется раздел «Сотрудники» в автоматизированной системе с полезной информацией о каждом пользователе.

Также на данный момент в ТЦР реализован процесс заказа общей еды в офис. Сотрудники сообщают, какую продукцию нужно закупить, ответственное лицо собирает запросы и в конце недели заказывает еду. Коммуникации можно заменить автоматизированной системой для упрощения и структуризации заказа еды в офис.

В обеденный перерыв у сотрудников возникает потребность сходить поесть в близлежащее заведение. Коллеги тратят много времени на поиск подходящего места и обмен отзывами между друг другом.

Кроме того, существует потребность быстро и своевременно сообщать сотрудникам информацию о планирующихся внутренних мероприятиях,

включая дату и время проведения, а также предоставлять доступ к материалам (презентациям, полезным ссылкам).

Вышеперечисленные процессы составили запрос на разработку АС для ТЦР «HR-automation». Система нужна, в первую очередь, чтобы помочь HR-специалисту сократить время на работу с новыми сотрудниками и упорядочить текущие бытовые процессы в офисе.

1.1.1.2 Проблема

Краткая проблема: большие временные затраты на взаимодействие между сотрудниками ТЦР.

Проблема неавторизованных процессов в офисе воздействует, прежде всего, на компанию – HR-специалист тратит много времени на около-бытовые процессы, а значит меньше времени уделяет рабочей деятельности, от чего компания теряет в прибыли. Передавая информацию «из уст в уста», появляется риск недостоверности и неполноты информации.

На рисунке 1 представлена диаграмма Ishikawa, на которой визуально представлены причины – факторы, влияющие на проблему.

Рисунок 1 – диаграмма Ishikawa

1.1.1.3 IDEF0

IDEF0 – методология и графическая нотация, предназначенная для формализации и описания бизнес-процессов.

Рассмотрим описание бизнес-процесса заказа продуктов до и после внедрения системы и на этом примере рассмотрим, как изменится процесс для офис-менеджера.

Контекстная диаграмма IDEF0

Процесс: заказ продуктов в офис

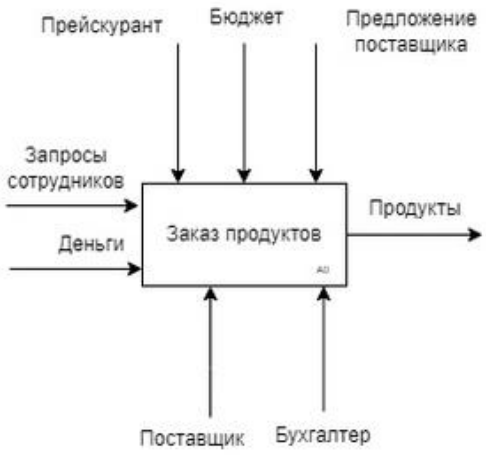

Рисунок 2 – Контекстная диаграмма IDEF0

Бизнес-процесс заказа продуктов в качестве входных данных принимает запросы сотрудников, содержащие информацию о том, какие продукты закончились, и денежные средства для закупки. Процедура выполняется двумя лицами – поставщиком и бухгалтером, контролируется прейскурантом – документом, устанавливающим стоимость продуктов, бюджетом – ограничивающим объем суммарных затрат, и предложение поставщика – определяющее наличие продуктов как таковых. Результат – заказанные продукты в офисе

Точка зрения: офис-менеджер

Цель: определить слабые стороны процесса заказа продуктов

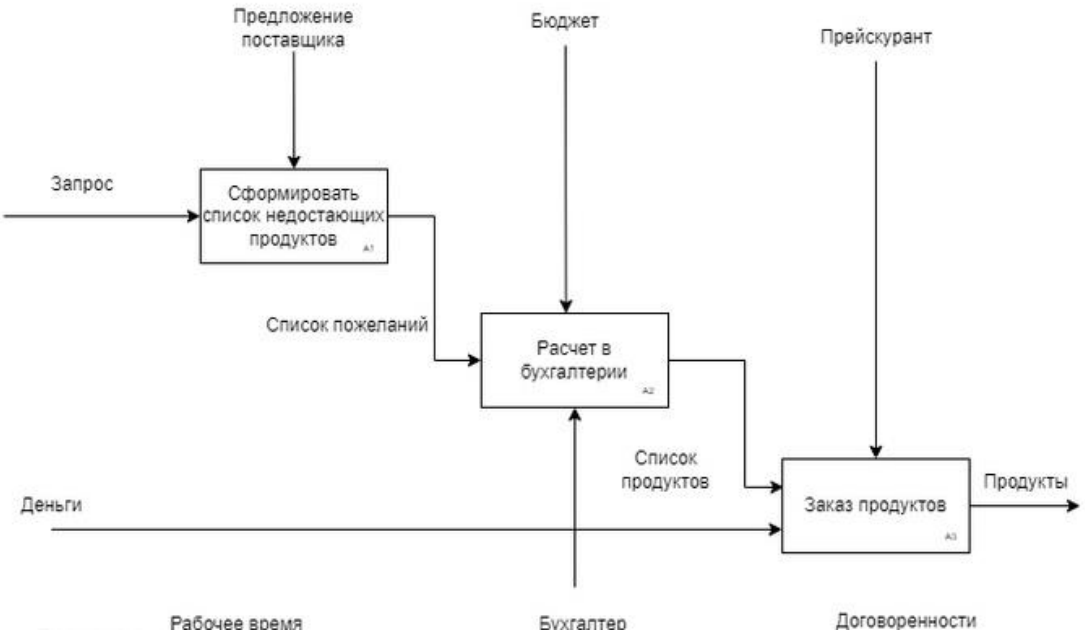

Рисунок 3 – Диаграмма IDEF0, уровень 1, до внедрения АС

Декомпозиция процесса позволяет визуально рассмотреть более маленькие процессы, их ресурсы и зависимости.

На диаграмме ниже декомпозируется процесс А1 «Сформировать список недостающих продуктов»

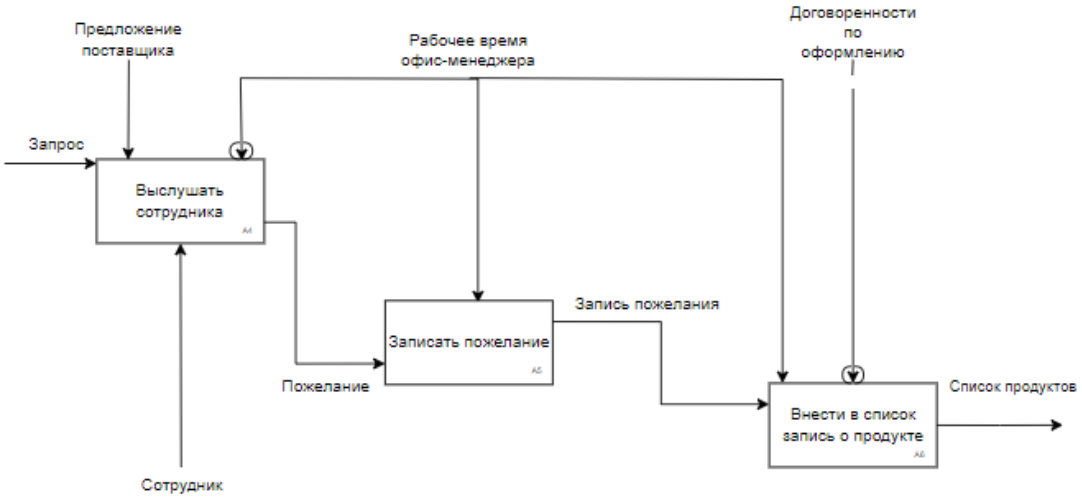

Рисунок 4 – Диаграмма IDEF0, уровень 2, до внедрения АС

Далее декомпозируем те же процессы с точки зрения офис-менеджера, но уже после внедрения системы.

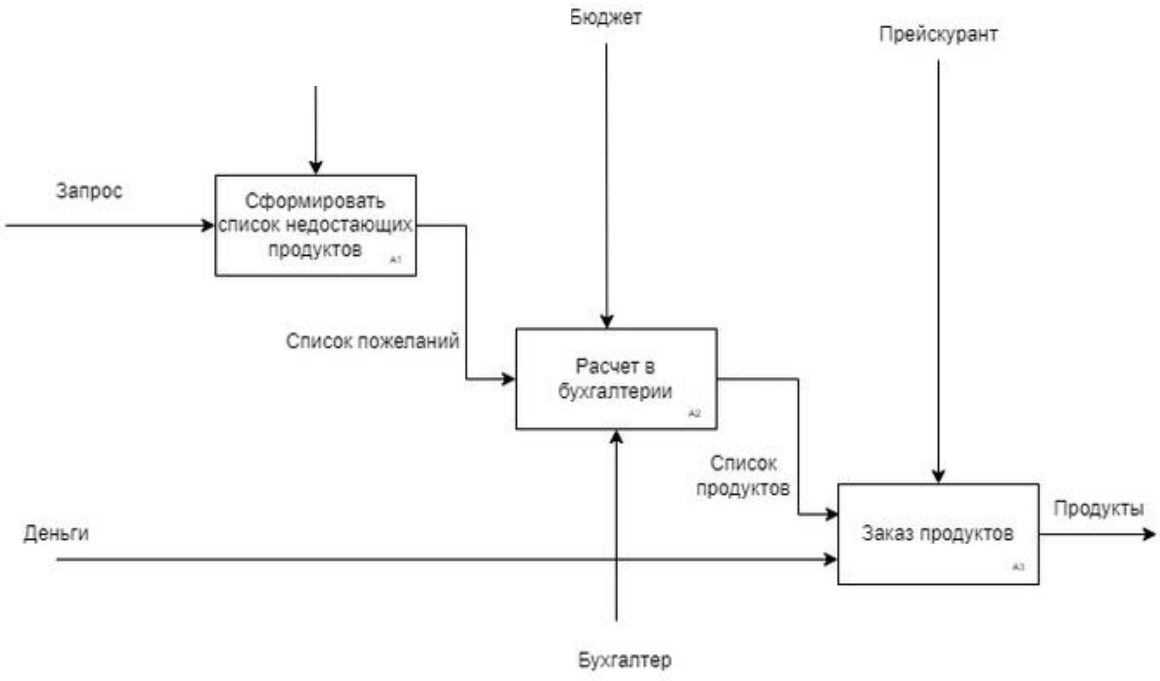

Рисунок 5 – Диаграмма IDEF0, уровень 1, после внедрения АС

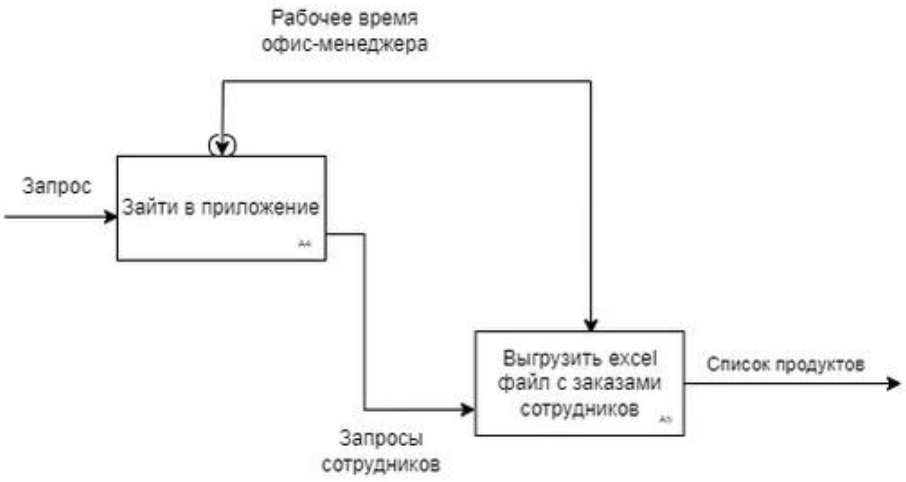

Рисунок 6 – Диаграмма IDEF0, уровень 2, после внедрения АС

Вывод: в результате функционального моделирования было определено, из каких частей состоит процесс «Сформировать список недостающих продуктов». После внедрения автоматизированной системы у покупателя процесс закупки становится менее ресурсозатратным, так как исчезают такие части, как «выслушать сотрудника» и «записать пожелание».

1.1.1.4 Цель и задачи

Цель разрабатываемой системы: уменьшение затрат времени на передачу информации между сотрудниками ТЦР.

Задачи разрабатываемой системы:

- создать реестр с информацией о сотрудниках офиса ТЦР;
- автоматизировать заказ товаров в офис;
- создать реестр с ответами на часто задаваемые вопросы;
- автоматизировать обмен информацией о заведениях рядом с офисом между сотрудниками;
- создать систему информирования сотрудников о внутренних событиях.

1.1.1.5 Контекст системы

Основные пользователи системы: сотрудники ТЦР. Доступ в административную панель получают HR-специалисты и лица, назначенные ответственными HR-специалистами.

За правильной работой системы следит главный программист. Офис-менеджер собирает отчет о заказанных продуктах.

Система взаимодействует с внешним сервисом онлайн-карт Google Maps, облачными сервисами Yandex Cloud и Firebase.

Объектами бизнеса являются сотрудники ТЦР. При регистрации сотрудника указывается следующая информацию о нем: имя, фамилия, отчество, дата рождения, должность, проект, на котором он работает. Границы для интерфейсов: система граничит с сервисом онлайн-карт. Размещение выполнения: веб-приложение для администраторов и сервер должны быть размещены на хостинге, мобильное приложение скачивается всеми сотрудниками ТЦР.

1.1.1.6 Основные функции системы

Как обязательные функции системы были выделены следующие:

для мобильного приложения:

- авторизация;
- личный кабинет;
- просмотр информации о коллегах;
- поиск коллег;
- просмотр ответов на часто задаваемые вопросы;
- заказ продуктов в офис;
- просмотр близлежащих к офису ресторанов;
- добавление отзывов к ресторанам;
- просмотр информации о мероприятиях.

Для административного приложения:

- редактирование/удаление/добавление/просмотр информации о сотрудниках;
- редактирование/удаление/добавление/просмотр ответов на часто задаваемые вопросы;
- редактирование/удаление/добавление/просмотр заказанных продуктов в офис;
- редактирование/удаление/добавление/просмотр близлежайщих ресторанов;

Желательные функции:

- фильтрация ресторанов;
- сортировка ресторанов.

Возможные функции:

для мобильного приложения:

- уведомление сотрудников о событиях;

- просмотр информации о событиях;
- фильтрация событий;

для административного приложения:

- редактирование/удаление/добавление/просмотр событий.

Отсутствующие функции:

- бронирование переговорных комнат;
- просмотр информации о переговорных комнатах.

Изначально планировался раздел «Переговорки», необходимый для бронирования комнат для проведения собраний и мероприятий, однако впоследствии от него было решено отказаться за ненадобностью.

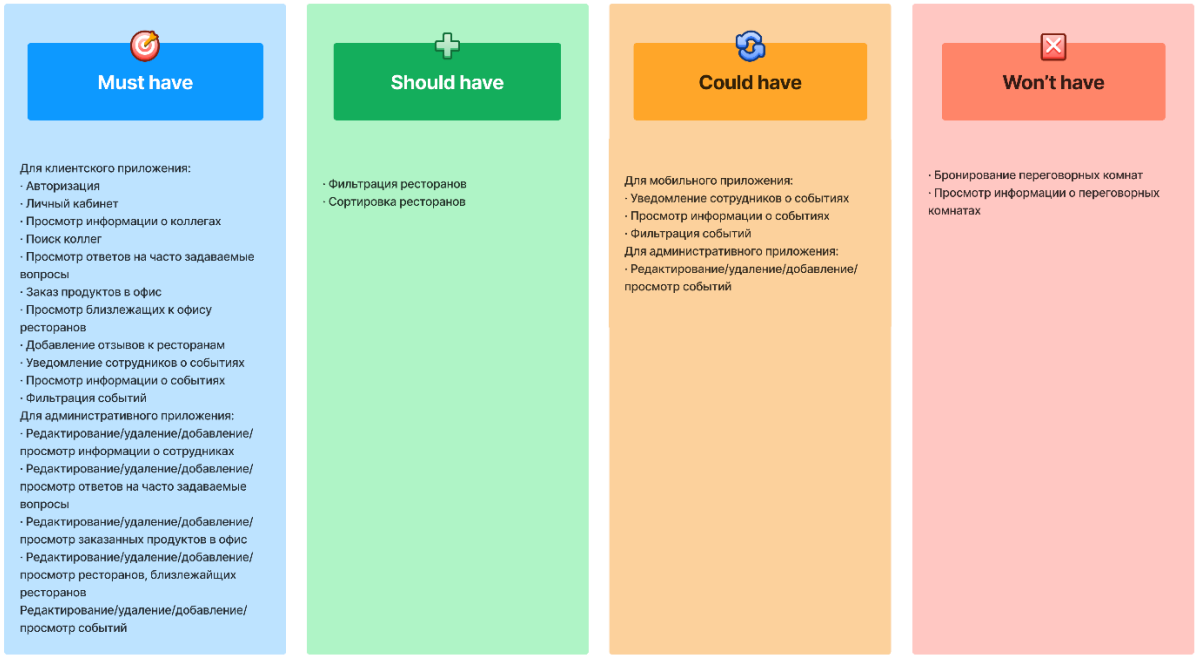

Рисунок 7 – Классификация функций методом MosCow

1.1.1.7 Ограничения

Таблица 1 – Ограничения

Источник	Ограничение	Комментарий

Эксплуатационный	Система должна быть интуитивно понятна без предварительного обучения пользования ей	Отсутствие ресурсов на обучение сотрудников пользованием системой
Технологические требования	Должны использоваться актуальные технологии для разработки	Не исключено добавление новых функций в систему позже
Эксплуатационный	Данные о клиентах должны храниться в зашифрованном виде	Для информационной безопасности
Эксплуатационный	Сотрудники без выданных прав администратора не могут получить доступ в административную панель	Для сотрудников предназначено мобильное приложение с меньшим количеством прав

1.1.1.8 Функциональные требования

Для мобильного приложения (см. приложение А)

Для веб-приложения (см. приложение Б)

1.1.2 Аналогичные решения

1.1.2.1 Типовые проблемы новых сотрудников

Согласно пользовательской истории заказчика, наиболее часто HR-специалисты отвечают на организационные вопросы, например, порядок поездки в командировку или на конференцию, скидки партнеров и бонусы на продукты компании, клиники по ДМС и даже на вопросы о том, как пользо-

ваться кофемашиной. Ответ на такие вопросы может быть сформирован и получен устно, от HR-специалиста или менеджера, что, как уже было выяснено во введении к данной работе, не эффективно в связи с частотой возникновения данных вопросов, поэтому такого способа нужно избежать. Далее следовало бы подумать, можно ли решить такую проблему составлением текстового документа с заранее заготовленными ответами, тогда было бы возможным свести человеческое вмешательство к нулю. Тем не менее в условиях постоянно растущей и изменяющейся среды офиса, количество таких вопросов может изменяться, могут добавиться новые, а ответы на старые изменяться. В таком случае необходимо не только изменить документ, но и предоставить к нему общий доступ, например, в рабочем пространстве. Уверенно можно сказать, что новоприбывший сотрудник с трудом разбирается, где и что искать в рабочем пространстве, даже если речь идет о небольшой компании штатом в 100 человек, в таком случае вопрос вновь будет адресован HR-специалисту. Значит ответы на вопросы должны быть отделены от рабочей информации и распространены локально для одного офиса, при этом список вопросов и ответов на них может редактироваться и пополняться.

Вторым по популярности вопросом от сотрудников является вопрос о том, где можно поесть. Проблема заключается не только в том, что сотруднику хочется узнать заведения по близости, но и оценить, в какой лучше пойти по мнению коллег, а также быстро сориентироваться для определения местоположения ресторана. Таким образом, требуется решение, которое предоставит информацию о ресторанах, в которые ходят сотрудники офиса, при этом необходимость беспокоить коллег или менеджеров должна отсутствовать.

Третьим вопросом обозначим информацию о сотрудниках офиса. Ответ может включать в себя, например, почту или проект интересующего сотрудника, а также дополнительную информацию.

Заказчик также выдвинул требования:

- наличие мобильного клиента на операционной системе android;
- наличие отдельной административной панели;
- возможность помечать закончившиеся на кухне продукты и выгружать список заказанного из административной панели в виде таблицы;
- возможность уведомлять сотрудников о новых событиях внутри компании.

Таким образом, необходима информационная система, которая для конечного пользователя позволяет отображать и редактировать следующую информацию:

1. Часто задаваемые вопросы и ответы;
2. Сотрудники и информация о них;
3. Карта с заведениями, которые посещают сотрудники и их отзывы об этих заведениях;
4. Продукты, доступные для заказа на кухню офиса;
5. События, информация о них и уведомления о новых событиях.

1.1.2.2 Имеющиеся решения

Прямых конкурентов, напрямую удовлетворяющих запросам, нет, но необходимый функционал рассредоточен в нескольких уже существующих продуктах. Определим критерии сравнения:

Таблица 2 - Критерии сравнения

Приватность	Имидж	Расширяемость	Соответствие требованиям

Приватность определяет, могут ли данные компании быть доступны третьим лицам, несет ли поставщик решения ответственность за утечки данных.

Имидж показывает удобство и соответствие статусу компании и ее бренда. Расширяемость – возможность легко добавить новый функционал. Соответствие требованиям проверяет удовлетворение конкретно запроса заказчика.

1.1.2.3 Сотрудники, часто задаваемые вопросы и события

Для покрытия потребностей данных разделов можно использовать канал в мессенджере и чат-бота, группу в социальных сетях или CRM-систему.

Таблица 3 – Сравнение аналогов по разделам «сотрудники» и «часто задаваемые вопросы»

Аналог	Приватность	Имидж	Расширяемость	Соответствие требованиям
Мессенджер	Нет	Нет	Нет	Да
Социальная сеть	Нет	Нет	Нет	Да
CRM-система	Да	Да	Да	Нет

Мессенджер и социальная сеть не подходят в качестве удовлетворительного решения по ряду причин: в случае утечки данных соответствующие лица не понесут ответственность за нанесенный ущерб, не соответствует статусу компании заказчика, канал в мессенджере не предоставляет удобств для просмотра ответов на вопросы, не могут быть дополнены функционалом, который может в будущем понадобиться заказчику.

CRM-система гарантирует защиту данных и возмещение ущерба, удовлетворяет элегантному статусу компании. Существуют современные решения, поддерживающие модульное подключение функций или придерживается принципов low-code и конструктор. К серьезным недостаткам отнесем необходимость оплаты тарифного плана и излишний функционал.

1.1.2.4 Закупка продуктов

Каждый сотрудник может отмечать продукт, который закончился, в электронной таблице или канале мессенджера. Такой способ имеет недостатки связанные с отсутствием наглядности информации и удобством использования – нужно придумать такую структуру документа, в которой можно использовать картинки и вести учет запросов от каждого сотрудника.

Таблица 4 – Сравнение по функционалу закупки продуктов

Аналог	Приватность	Имидж	Расширяемость	Соответствие требованиям
Электронная таблица	Да	Нет	Нет	Да
Мессенджер	Нет	Нет	Нет	Да

1.1.2.5 Рестораны

Имеющиеся аналоги, наиболее похожие по функционалу – 2GIS, Google maps и Яндекс карты. Они позволяют наглядно увидеть заведения по близости, однако носят публичный характер и потому отзывы, оставленные на этих сервисах, не гарантируют соответствие действительности в силу существования накруток и прямых конкурентов.

Таблица 5 – Сравнение по функционалу обзора ресторанов

Аналог	Приватность	Имидж	Расширяемость	Соответствие требованиям
2GIS	Нет	Нет	Нет	Нет
Google maps	Нет	Нет	Нет	Нет
Яндекс карты	Нет	Нет	Нет	Нет

1.1.2.6 Итог

Итого имеем, что закрытие потребностей с помощью нескольких различных продуктов не целесообразно. Проект «HR-automation» содержит все необходимые функции, а также легко интегрируется с корпоративной системой заказчика, соответствует требованиям заказчика к дизайну и безопасности.

1.2 Выбор и обоснование состава программного обеспечения

1.2.1 Веб-часть системы

«Административная панель» представляет собой клиентское веб-приложение. При выборе инструментов разработки были использованы следующие критерии:

1. Масштабируемость

Следует учитывать масштабируемость инструментов для обеспечения будущего роста информационной системы. Так как информационная система разрабатывается в первую очередь под потребности людей – сотрудников ТЦР, нельзя исключать расширение системы для нового функционала.

2. Пользовательский опыт

Перед тем как выбрать инструмент, следует оценить опыт программистов, а также простоту использования программного обеспечения и время, затрачиваемое на обучение. Так как на разработку ИС было выделено 6 месяцев, нужно было использовать средства, базовый функционал которых можно быстро изучить, а более сложные функции продолжить изучать в процессе разработки веб-приложения.

3. Стоимость

Бюджета на разработку информационной системы не выделялось (проект является учебным), поэтому важно было выбрать бесплатные инструменты разработки.

4. Надежность

Нужно учитывать надежность программного обеспечения, включая доступность и стабильность инструментов, а также поддержку, предоставляемую поставщиком. Нельзя было допустить использование средств, которые перестанут поддерживаться через какое-то время, чтобы избежать переписывание кода.

5. Сообщество

Важно оценить, насколько активно сообщество поддерживает выбранные инструменты, включая доступность документации, примеров кода и форумов для поддержки. К веб-разработчику был приставлен куратор из «проектного практикума», который помогал с разработкой, давал советы и проводил code review (процесс проверки и анализа кода). Но при этом веб-разработчик обращался к куратору только по вопросам, которые не смог решить самостоятельно, а именно найти ответы на появляющиеся вопросы на тематических интернет-форумах.

Таким образом, по выдвинутым параметрам для разработки выбирались следующие инструменты:

- масштабируемые;
- простые в использовании;
- бесплатные;
- стабильные;
- с доступной документацией;
- с активным сообществом.

1.2.1.1 Выбор языка программирования

Перед разработкой клиентской части «Административной панели» необходимо было выбрать язык программирования. Мы не стали рассматривать

для разработки zero-code или low-code технологии, так как веб-приложение предполагалось слишком сложным и большим для таких технологий.

JavaScript — это язык программирования, который широко используется для веб-разработки. В основном он используется для написания сценариев на стороне клиента. Это означает, что он выполняется в веб-браузере пользователя, в отличие от серверных сценариев, которые выполняются на сервере. JavaScript предоставляет широкий спектр функций, которые делают его мощным инструментом для создания пользовательских интерфейсов, обработки взаимодействий с пользователем и выполнения динамической обработки на стороне клиента. JavaScript используется в качестве клиентского языка программирования на 98% всех веб-сайтов и хорошо знаком веб-разработчику, а другие решения, такие как AdobeFlash или SilverLight, не имеют широкой поддержки и устарели. Поэтому было решено использовать для разработки JavaScript или языки программирования, транслируемые в JavaScript.

Далее нужно было выбрать, будет ли использоваться для разработки фреймворк, язык программирования, расширяющий возможности JavaScript или чистый JavaScript.

В разработке веб-приложений фреймворки ускоряют написания кода и позволяют сосредоточиться на логике приложения, а не на написании основных функций. За счёт того, что фреймворк переиспользует готовые модули, уменьшается время релиза новой функции. Если использовать готовый шаблон, только дополняя некоторые функциональные блоки, и наполнять его контентом, то шансы ошибиться уменьшаются.

Также в пользу фреймворков можно сказать, что основная часть из них защищена и протестирована. Таким образом, можно не бояться взломов и

XSS-атак (тип атаки на веб-системы, заключающийся во внедрении в выдаваемую веб-системой страницу вредоносного кода и взаимодействии этого кода с веб-сервером злоумышленника).

Чистый же JavaScript используется, когда

- Веб-приложение небольшое и простое;
- Веб-приложение имеет высокие требования к производительности.

Разрабатываемое клиентское веб-приложение содержит в себе 5 блоков разной сложности и авторизацию, не требует быстро обрабатывать большие данные. Также приложение требуется написать быстро (6 месяцев) при небольшом опыте веб-разработки у программиста. Из этого следует выбор в пользу фреймворков.

1.2.1.2 Выбор фреймворка

Самыми популярными фреймворками и библиотеками, основанными на языке программирования JavaScript, на которых пишутся клиентские веб-приложения, являются:

- React;
- Angular;
- Vue.js;
- Svelte.

Менее популярные не рассматривались, так как не подходили по важным критериям «активное сообщество» или «простота в использовании». Разработка с помощью таких фреймворков не уложилась бы в поставленные сроки,

так как из-за недостатка образовательных ресурсов, гайдов и обсуждений вопросов по разработке на данном фреймворке обучение заняло бы значительно больше времени по сравнению с более популярными аналогами.

С помощью ресурсов сети Интернет (тематических форумов и статей) мы сравнили фреймворки по пятибалльной шкале (1 – меньшая оценка по критерию, 5 – наибольшая оценка по критерию) по следующим критериям:

- активное сообщество;
- простота в использовании;
- быстрота обучения;
- уровень знаний по фреймворку у фронтенд-разработчика;
- уровень знаний по фреймворку у куратора.

Результаты представлены в таблице 6:

Таблица 6 – Сравнение фреймворков веб разработки

Название фреймворка/библиотеки	Активное сообщество	Простота в использовании	Быстрота обучения	Уровень знаний по фреймворку у фронтенд-разработчика	Уровень знаний по фреймворку у куратора
React	5	4	4	2	3
Angular	5	3	4	1	5
Vue.js	3	5	5	1	1
Svelte	3	5	5	1	1

По полученным результатам была построена гистограмма, представленная ниже.

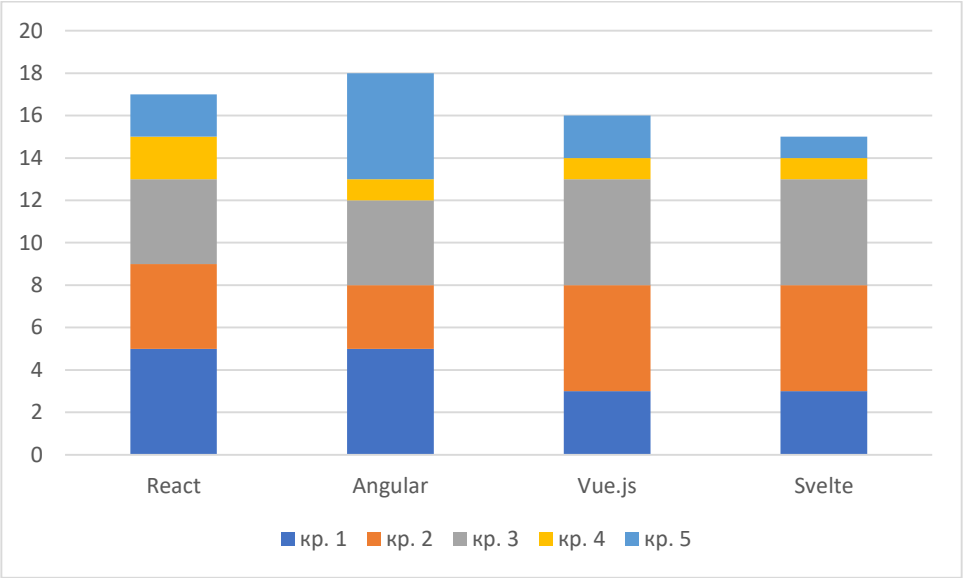

Рисунок 8 – Гистограмма фреймворков

Таким образом, было принято решение использовать фреймворк Angular последней версии (14) в связке с языком программирования TypeScript.

1.2.1.3 Angular

Angular — это фреймворк от компании Google для создания веб-приложений — SPA (Single Page Applications) — на языке программирования TypeScript. Angular является не самым простым в освоении фреймворком, так как имеет нестандартную структуру и поддерживает начало работы «из коробки». Разработанные на этом фреймворке проекты состоят из независимых элементов – компонентов. Они выполняют определенный набор задач и функций, при этом отражают визуальный элемент (разметку) или группу элементов. Основное отличие Angular от других фреймворков в том, что отображение имеет возможность менять модель, а модель — имеет возможность менять отображение.

В связке с Angular использовалась группа библиотек NgRx. NgRx Store это инструмент, позволяющий обеспечивать реактивное управление состоянием в приложениях Angular.

1.2.1.4 TypeScript

Язык программирования TypeScript — это расширенная версия языка JavaScript, созданная изначально в Microsoft для разработки крупных веб-приложений. Данный язык программирования решает такие типичные проблемы JavaScript, как

- ошибки типов в среде выполнения;
- неконтролируемо «разрастающийся» код.

Обучение TypeScript не предполагало больших временных затрат, так как фронтенд-разработчик имел базовые знания JavaScript, а так как TypeScript является расширением JavaScript, их синтаксисы очень похожи.

1.2.1.5 HTML и CSS

В разработке проекта безусловно были задействованы HTML (язык гипертекстовой разметки) версии 5 и CSS (язык таблиц стилей) версии 3 для создания визуальных интерфейсов. Обоснование однозначного выбора этих инструментов заключалось в отсутствии аналогов, сопоставимых с выбранным фреймворком.

Препроцессоры (программы, позволяющие генерировать код CSS, используя свой уникальный синтаксис) не использовались в разработке веб-приложения, так как проект был направлен больше на изучение и практику языка программирования и фреймворка, чем на написание разметки и таблицы стилей. Оптимизация написания CSS-кода посчиталась не актуальна, так как данного кода планировалось немного.

1.2.1.6 Taiga UI

В качестве библиотеки визуальных компонентов использовалась Taiga UI. Taiga UI – это набор компонентов с открытым исходным кодом, разработанная в Tinkoff. Данное решение было советом куратора, так как идеально подходит под данный проект (проста в использовании, поддержание внутреннего стиля компании) и хорошо знакома куратору, а значит, он сможет быстро подсказывать при возникших проблемах.

Библиотека имеет очень понятную документацию с удобным интерфейсом на официальном сайте (пример на рисунке ниже)

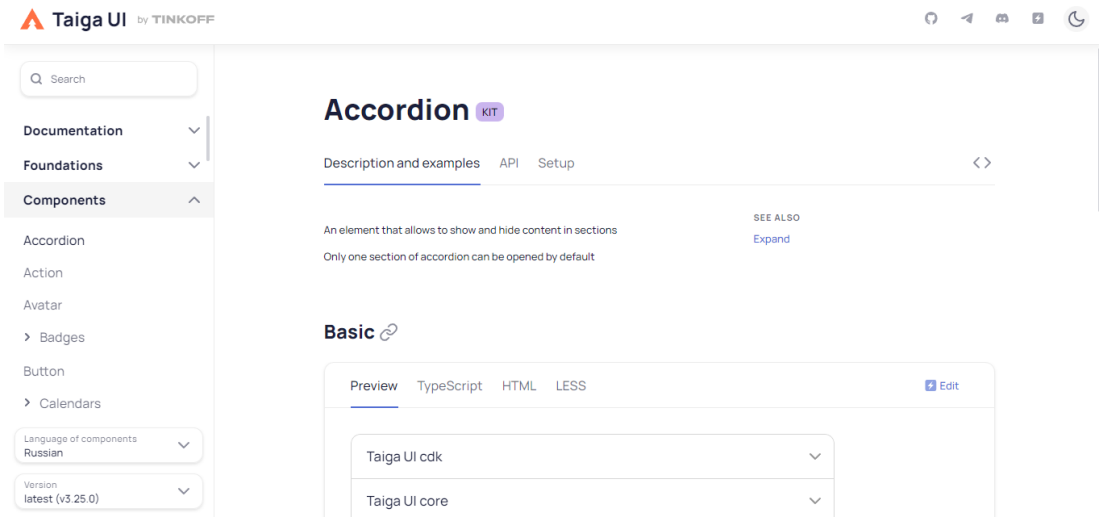

Рисунок 9 – Taiga UI

1.2.1.7 Google Maps

В проекте планировалось использовать компоненты карт для отображения ресторанов и мест проведения событий на карте. Среди возможных инструментов, которые охватывают российские города, были выделены API (программный интерфейс приложения) следующие:

- Яндекс.Карты;
- Google Maps;

- 2ГИС.

Сервис Google Maps выделялся своей простотой в использовании и многочисленными гайдами в сети Интернет, поэтому было принято решение использовать для разработки его. Данный набор интерфейсов хорош тем, что подходит, как и для веб-сайтов, так и для Android, а значит в проекте будет меньше несовместимых инструментов.

1.2.1.8 Git

В проекте была необходима система контроля версий, чтобы отслеживать и вести историю изменения файлов. Git является самой популярной распределённой системой управления версиями и хорошо знакома разработчикам. Аналоги системы git не имеют такой большой поддержки и большого активного сообщества, а значит, использование данных средств будет затруднительнее. Таким образом, было решено использовать во всем проекте git.

1.2.1.9 IDE

Для быстрой и удобной разработки необходимо использовать редактор кода. Данная программа позволит подсвечивать синтаксис – это улучшает читабельность кода, и автоматически дополняет код – данная функция ускоряет написание кода.

В качестве IDE (интегрированная среда разработки) или редактора кода рассматривались следующие, бесплатные и популярные у веб-разработчиков, варианты:

- WebStorm;
- Visual Studio Code;
- Atom;
- Notepad++;
- Vim.

С помощью ресурсов сети Интернет (тематических форумов и статей) для оптимального выбора были сравнены редакторы кода по пятибалльной шкале (1 – меньшая оценка по критерию, 5 –наибольшая оценка по критерию) по следующим критериям:

- Легкость настройки;
- Наличие встроенных команд Git;
- Быстрота отладки;
- Наличие проверки кода в режиме реального времени;
- Легкость нагрузки на ОС.

Таблица 7 – Сравнение сред разработки

Название редактора кода	Легкость настройки	Наличие встроенных команд Git	Быстрота отладки	Наличие проверки кода в режиме реального времени	Легкость нагрузки на ОС
WebStorm	4	5	4	5	2
Visual Studio Code	5	5	4	5	3
Atom	2	1	3	5	4
Notepad++	5	2	5	2	5
Vim	1	1	5	5	5

По данным результатам была построена гистограмма, представленная ниже.

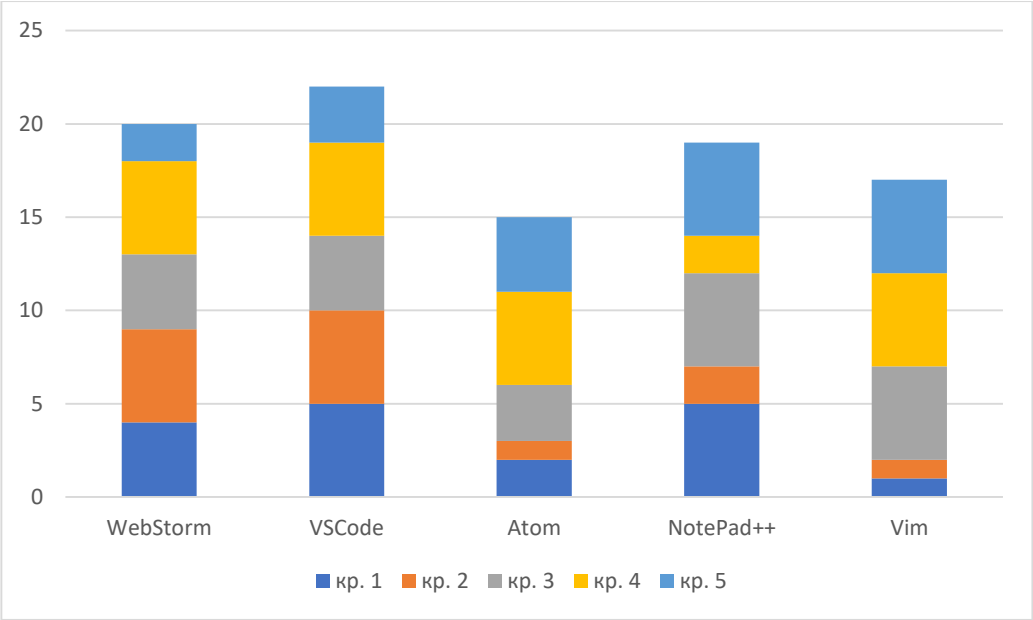

Рисунок 10 – Гистограмма сравнения сред разработки

Таким образом, было принято решение использовать редактор кода Visual Studio Code для разработки веб-части информационной системы.

1.2.1.10 Итог

Таким образом для разработки веб-части системы был выбран следующий стек технологий:

1. Type-Script;
2. Angular;
3. NgRx;
4. HTML;
5. CSS;
6. Taiga UI;
7. Google Maps;
8. Git;
9. Visual Studio Code.

1.2.2 Серверная часть системы

1.2.2.1 Выбор языка программирования и фреймворков

Для разработки серверной части использовался язык Java. Этот язык отлично подходит для написания больших веб-приложений, так как он объектно-ориентирован и строго типизирован. На его основе уже существует множество библиотек и фреймворков, один из них – Spring. Spring позволяет структурно разделять части приложения и использовать всевозможные паттерны, что делает код более читаемым и «правильным». Большое количество различного встроенного функционала позволяет гибко и точно настраивать веб-приложение для нужных целей, не прописывая вручную большие повторяющиеся куски кода. Также для настройки проекта использовался Spring Boot. Этот фреймворк тоже помогает сократить количество ручного труда, но уже в области конфигурации приложения. Так как разрабатываемый проект не требует какой-то специфической настройки, то Spring Boot идеально подходит для автоматического координирования частей приложения между собой. Важным критерием при выборе стека технологий является надежность. Java и Spring в хорошем смысле старые технологии, а значит вероятность столкнуться с багом на стороне их разработчиков значительно меньше, чем у более новых языков и фреймворков. Также существует большое количество разных библиотек, расширяющих Java, поэтому довольно легко подобрать что-то для себя, не беспокоясь при этом о совместимости разных средств. Кроме того, уже написано огромное множество документаций и справочной информации по этим технологиям.

1.2.2.2 Выбор сборщика кода

Ручная сборка приложения занимает много времени и может вызывать определенные трудности. В данном проекте в этом нет необходимости, поэтому была выбрана система автоматической сборки Gradle. Ближайший его

аналог – Maven – обладает одним значительным недостатком. Он использует язык xml, что в больших проектах приводит к неоправданному нагромождению кода и делает его трудночитаемым. Код же на Gradle выглядит намного лаконичнее и понятнее.

1.2.2.3 Выбор системы управления базами данных и сопутствующих фреймворков

Сложно представить веб-приложение, в котором бы не понадобилось использовать базу данных, поэтому необходимо также подобрать средства для работы с ней. В качестве системы управления базами данных была выбрана PostgreSQL. Данная СУБД является бесплатной, подходит для любых операционных систем. Доступность системы также дополняется ее относительной простотой. Многие SQL-запросы в PostgreSQL имеют свои упрощенные аналоги, что сокращает время на их написание. В системе PostgreSQL представлены разные форматы и типы данных, в том числе пользовательские. Это помогает сделать базу данных гибкой, позволяет хранить информацию в наиболее удобном виде. Еще одним важным требованием, предъявляемым к системе управления базами данных, является надежность. PostgreSQL соответствует четырем основным требованиям ACID, а именно является атомарной, согласованной, изолированной и устойчивой, а значит не придется переживать о том, что данные каким-то образом повредятся. Также стоит отметить популярность этой системы, она широко используется многими разработчиками и большинство сторонних сервисов и инструментов поддерживают PostgreSQL.

Для связи ООП-сущностей непосредственно с реляционными базами данных использовался Hibernate – еще один популярный фреймворк, написанный для Java. Hibernate реализует спецификацию JPA (Java Persistence API) для удобного хранения Java-объектов в базе данных. Главное преимущество Hibernate это упрощение работы с БД. Нет необходимости вручную прописывать

SQL-запросы, достаточно описать сущность в виде обычного Java-объекта, с которым впоследствии удобно работать. Это ускоряет процесс написания кода, делает его более читаемым. Популярность данного фреймворка дает все те же преимущества в виде большого количества доступной информации и обучающих материалов. Кроме того, Hibernate не привязывается к конкретной СУБД и поддерживается многими технологиями на Java.

1.2.2.4 Выбор средств для развертывания сервера

Для оперативной командной работы необходимо было разместить сервер в открытый доступ. Для этого был выбран облачный сервис Render. Важнейшими преимуществами относительно других аналогов являются доступность и простота использования. Он предоставляет бесплатный пробный период и доступен из России. Также не менее важно то, что Render бесплатно позволяет создать базу данных с помощью PostgreSQL. Расположение приложения и базы данных в одном месте крайне удобно, так как их легко связать средствами используемого сервиса. Кроме того, Render можно подключить к GitHub и тогда сервис автоматически находит конфигурационный файл и разворачивает приложение. Это сокращает время на настройку проекта и значительно упрощает работу. Все изменения в репозитории на GitHub сразу же подгружаются в Render и сервер перезапускается. Такая возможность позволяет оперативно загружать все изменения и не отвлекаться на развертывание приложения.

Также для упрощения развертывания приложения использовался Docker. Docker использует технологию контейнеризации, которая приходит на замену виртуальным машинам. С помощью этого инструмента написанное приложение можно одинаково успешно запускать на различных платформах и эффективно перемещать между ними. Благодаря контейнеризации в разы упрощается процесс развертывания приложения и обнаружения возможных

проблем. Кроме того, Docker легковесен относительно виртуальных машин и позволяет экономить аппаратные ресурсы и оптимизировать их использование. Docker является одной из самых популярных и эффективных технологий в данной сфере.

1.2.2.5 Выбор облачного сервиса, для хранения изображений

В разрабатываемом приложении предусмотрена работа с изображениями, которые необходимо где-то хранить. Для этого был выбран облачный сервис Yandex Cloud. Хранение изображений в облаке позволяет значительно высвободить и оптимизировать ресурсы сервера. В базе данных же хранятся только ссылки на файлы. Yandex Cloud реализует технологию S3 (Simple Storage Service), которая идеально подходит для приложений, использующих протокол HTTP. В объектном хранилище каждый файл представлен собственным идентификатором и может содержать различные метаданные, что позволяет хранить дополнительную информацию об объектах и обеспечивает быстрый поиск. Yandex Cloud это отечественная разработка, поэтому большое количество обучающих материалов (в том числе и документация) представлены на русском языке. Также Yandex Cloud предоставляет бесплатный пробный период, что прекрасно подходит для разработки в учебных целях.

1.2.2.6 Итог

Таким образом для бэкенд-разработки был выбран следующий стек технологий:

1. Java;
2. Spring Framework;
3. Spring Boot;
4. Gradle;
5. PostgreSQL;

6. Hibernate;
7. Render;
8. Docker;
9. Yandex Cloud.

1.2.3 Мобильное приложение

1.2.3.1 Выбор языка и основного инструмента разработки

Приложения под операционные системы Android могут быть написаны на следующих языках и с использованием соответствующих им инструментов разработки:

- Java – язык использовался для разработки в качестве основного до 2019 года, когда компания Google объявила, что Kotlin теперь является приоритетным в разработке под Android. Использует стандартный Android SDK;
- Kotlin – был включен в официальный инструмент разработки в 2017 году. Простота, понятность и краткость языка обеспечивают удобство при написании и чтении кода, а полная совместимость с Java позволяет большей части имеющейся кодовой базы легко адаптироваться. Использует Android SDK;
- C\C++ - Google рекомендует исключительно для разработки приложений, требовательных к ресурсам устройства. Малое количество статей и документации для его инструментов разработки;
- JavaScript с использованием фреймворка React Native. Позволяет разработку приложений под множество платформ, в том числе Android и IOS. Не соответствует стеку разработки заказчика, что означает отсутствие расширяемости. Расширение до уровня крупных нагруженных проектов в целом невозможно в связи со следующими недостатками:
  - Низкая производительность;
  - Низкий потенциал при разработке сложного интерфейса;

- Отсутствие модулей со специфичными компонентами;
- Чрезвычайная сложность обновления версий React Native.

- Dart с фреймворком Flutter. Аналогично React Native обеспечивает разработку под несколько платформ – быстрее и дешевле для бизнеса. Flutter это относительно свежая технология, это означает, что его сообщество и кодовая база довольно малы, многие библиотеки еще на стадии разработки. Dart и Flutter неразделимы, в первую очередь отсюда следует, что недостатки Dart'а так же унаследованы: язык все еще развивается, отсутствует стабильность и уверенность в возможности поддержки в долгосрочной перспективе.

- C# с фреймворком Xamarin. Заметно более надежное кроссплатформенное решение, но не без недостатков:
  - Для промышленной разработки Xamarin платный;
  - В качестве IDE для разработки с использованием продуктов Microsoft лучше всего подойдет Visual Studio, которая тоже будет платной для комфортной промышленной разработки;
  - После выпуска новых возможностей платформ, их использование на Xamarin потребует времени, в отличие от нативных решений;
  - Производительность все еще будет заметно отставать от нативных решений для проектов с высокими требованиями к ней;
  - Большой размер приложения.

Таким образом, для данного проекта был выбран Kotlin в качестве языка программирования. Это позволит использовать наиболее популярные инструменты и подходы, рекомендуемые при разработке приложений под Android, и поэтому эти технологии будет проще изучить, обеспечит возможность масштабируемости и поддержки проекту, что критически важно для нашего заказчика.

1.2.3.2 Выбор библиотек

Выбор библиотеки для многопоточности сводится к выбору между RxJava и Coroutines. RxJava рекомендует себя как зрелый, проверенный фреймворк. Coroutines это относительно новый инструмент, который набирает популярность, проекты и разработчики переходят именно на него. Основные преимущества Coroutines над RxJava это простой API, структурированность многопоточности, простота операторов, более высокая производительность и покрытие некоторых недостатков, например, с переизбытком входящих событий.

Для упрощения взаимодействия с сервером через HTTP запросы необходимо подобрать соответствующую библиотеку. Наиболее часто для этих целей выбирают Retrofit или Volley. Volley имеет некоторые преимущества, такие как поддержка загрузки картинок, приоритизация, отмена и повтор запроса из коробки с минимумом усилий, кэширование. К существенным недостаткам отнесем отсутствие наглядности и понимания работы написанного кода, не способен приводить ответ к нужному виду автоматически. Код написанный с Retrofit понятен и удобен при тестировании, не выглядит как черный ящик на первых этапах использования. Другие недостатки давно можно решить небольшим количеством кода, который не трудно найти и изменять под нужды. Именно поэтому большая часть разработчиков предпочитают Retrofit.

Для упрощения тестирования, переиспользования и переосмысления кода необходим механизм внедрения зависимостей. Существуют следующие общеизвестные фреймворки, использующиеся с выбранными технологиями: Dagger, Hilt, Koin, Codeine. Dagger предоставляет полный набор необходимых инструментов для разработки даже крупных приложений и повсеместно используется заказчиком при разработке собственных продуктов, что означает возможность последующего сопровождения и разработки. Этот инструмент

часто используется, примеры и документация доступны. Вышеперечисленного достаточно для того, чтобы выбрать этот фреймворк для нашего проекта, однако стоит упомянуть еще один, тесно связанный с Dagger'ом. Hilt – это надстройка над Dagger, значительно упрощающая и ускоряющая разработку и понимание основ внедрения зависимостей, уменьшает количество необходимого к написанию кода для получения необходимых результатов. Однако простота данного фреймворка и является его изъяном для крупных проектов и поэтому не используется заказчиком. Отдадим предпочтение фреймворку Dagger в связи с большими перспективами для развития разработчиков и потенциального развития проекта.

Иногда необходимо логировать информацию в консоль чтобы проверить ее достоверность или протестировать некоторый функционал, не пользуясь встроенным дебаггером среды разработки. Для логирования наиболее известное решение – Timber, позволяющий логирование исключительно при отработке приложения в состоянии разработки, а также предоставляющий некоторые правила для линтера, контролирующего качество кода.

Для загрузки изображений с Yandex Cloud используем фреймворк Glide. Этот инструмент способен быстро скачать изображения, в том числе и для множества позиций в списке, подогнать по размеру и вставить их в элемент ImageView системы Android. Согласно заявлениям разработчиков фреймворк подходит под любой сетевой стек, может загружать не только картинки, но и короткие видео

Практически ни в одном Android проекте не обходится без библиотеки Adapter Delegates, которая позволяет удобно создать несколько различных представлений для отображения в списке и подавать их адаптеру списка как родственные представления. Стоит отметить, что при использовании данной

библиотеки скорее всего придется разработать собственный инструмент для пагинации списков, так как имеющееся решение Google – Paging 3 можно использовать совместно лишь с большим трудом, а уже разработанных решений пагинации именно для Adapter Delegates найти не удалось.

Для работы с картами наиболее приемлемым вариантом будет взять уже готовую библиотеку. Наиболее известные варианты — это Яндекс Карты и Google Maps. Яндекс Карты имеет чрезвычайно неудобную документацию с недостатком реальных практических примеров, в том числе и в интернете в целом. Google Maps – популярный инструмент, имеющий множество гайдов и примеров по всему интернету, в особенности при использовании с такими технологиями Google как Android и Angular.

ГЛАВА 2. Проектирование системы

2.1 Архитектура системы

2.1.1 Общая архитектура

На рисунке 11 схематично представлена архитектура компонентов приложения

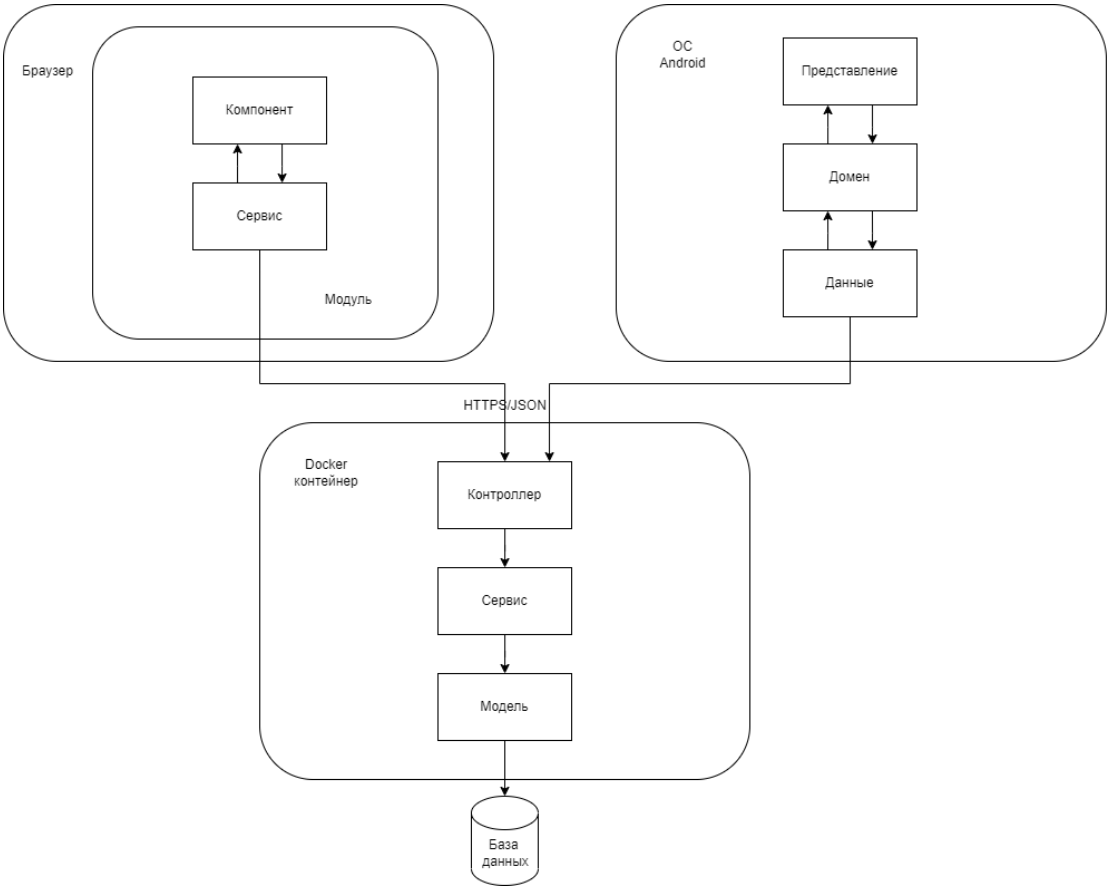

Рисунок 11 – Общая архитектура

Архитектура мобильного приложения состоит из трех слоев:

1. слой данных - слой, содержащий api взаимодействия с источником данных;
2. доменный слой - содержит логику бизнеса, методы и сущности для работы и перехода между слоями;
3. презентационный слой - содержит логику отображения.

Архитектура сервера состоит из трех основных компонентов:

1. контроллер – класс, обрабатывающий запросы от клиента;
2. сервис – класс, который содержит всю основную логику приложения;
3. модель – представление сущностей базы данных в виде java объектов, необходимое для простого оперирования данными.

Angular-приложение состоит из модулей:

1. модуль группирует компоненты и сервисы, связанные друг с другом;
2. компонент отвечает за представление информации из вне, которая извлекается сервисом;
3. сервис – это класс, который выступает в качестве поставщика данных.

Мобильное и веб-приложения связываются с сервером по протоколу HTTPS с помощью JSON файлов

2.1.2 DFD

Диаграммы потоков данных (DFD) используются для описания документооборота в компьютерных системах и в организации блок-схемы процедур. На диаграмме показан процесс потока данных для системы HR-automation. На уровне 1 рассматриваются процессы «Просмотр сотрудника» и «Добавление сотрудника». В контекстной диаграмме выделены следующие внешние сущности: Сотрудник, HR-специалист (см. приложение В)

2.1.3 EPC

Событийная цепочка процессов используется для описания процессов нижнего уровня. Диаграмма процесса в нотации EPC представляет собой упорядоченную комбинацию событий и функций.

Процесс заказа продуктов в офис для администратора начинается с события возникновения потребности заказа, такая потребность появляется каждые две недели. Ответственный за заказ продуктов сотрудник открывает веб-приложение административной панели. Сначала нужно пройти авторизацию – пользователь вводит адрес электронной почты, на которую был зарегистрирован, а после вводит одноразовый код, который ему прислала система на почту. Если авторизация прошла успешно, пользователь переходит во вкладку «Продукты». Там он нажимает кнопку «Выгрузить список продуктов», на его локальную машину скачивается файл с расширением xlsx, который содержит названия, артикулы и нужное количество продуктов, которые сотрудники выбрали для заказа. Процесс заканчивается, администратор может заказывать продукты по списку в сторонних системах (см. приложение Г)

2.1.4 Диаграмма вариантов использования

Диаграмма вариантов использования– диаграмма, описывающая, какой функционал разрабатываемой информационной системы доступен каждой группе пользователей. На диаграмме (см. приложение Д) рассмотрены варианты использования для мобильного приложения системы для авторизованного пользователя и неавторизованного пользователя. Так как разрабатываемая система является корпоративной, ее функции закрыты для сторонних пользователей (не сотрудников). Таким образом неавторизованному пользователю доступна только функция авторизации.

Авторизованный пользователь может использовать следующие разделы системы «Сотрудники», «Часто задаваемые вопросы», «Продукты», «Рестораны» и «События».

2.2 Проектирование административной панели

2.2.1 Введение

Административная панель является частью разрабатываемой автоматизированной информационной системы и предназначается для управления и администрирования предоставляемых функций. Ключевыми функциями административной панели являются:

- Управление страницами зарегистрированных сотрудников:

Редактирование, добавление и удаление аккаунтов сотрудников;

- Управление часто задаваемыми вопросами (FAQ):

Добавление и удаление категорий вопросов. Редактирование, удаление и добавление отчетов на часто задаваемые вопросы;

- Управление продуктами:

Добавление, редактирование и удаление возможных к заказу продуктов. Редактирование списка заказанных продуктов. Выгрузка списка заказанных продуктов в файл;

- Управление близлежащими к офису заведениями:

Добавление, редактирование и удаление ресторанов. Удаление отзывов к ресторанам;

- Управление событиями:

Добавление, редактирование и удаление проходящих в Тинькофф внутренних мероприятий;

2.2.2 Архитектура

2.2.2.1 Пользователи и роли пользователей

Административная панель предназначена для пользования определенной группе сотрудников. Это HR-специалисты, менеджеры офиса и прочие со-

трудники, наделенные администраторскими правами. Сотрудники, не наделенные правами администратора, не имеют доступ к административной панели.

2.2.2.2 Системные требования

Веб-приложение полностью должно быть написано на фреймворке Angular, язык программирования Typescript. Для управление состоянием должна использоваться группа библиотек NgRx.

Разметка должна быть написана с помощью HTML, а для написания таблицы стилей используется CSS.

Для управления состоянием приложения используется группа библиотек NgRx.

Встроенные карты должны базироваться на сервисе Google Maps API.

Для проектирования интерфейсов и дизайна использовалась Figma.

2.2.2.3 Архитектура системы

В приложении используется шаблон хранилища (Store) и вся архитектура обмена данными строится на нем. Хранилище — это набор Actions, Selectors, Reducers и Effects. Store является единственным источником достоверности для состояния приложения и инкапсулирует его с помощью глобального объекта.

Основные компоненты NGRX:

- Reducer — это функция, которая принимает текущее состояние и action и возвращает новое состояние. Reducer обрабатывает ту часть состояния, которую охватывает action. В разрабатываемом приложении данные функции формируют дерево редукторов;

- Actions представляют изменения состояния в приложении. Action исполняет задачу идентификатора, который сопоставляется с редуктором для обновления состояния. В разрабатываемом приложении actions также формируют дерево;
- Effects отвечают за обновление состояния на основе действия. Они используются для выполнения асинхронных действий (например, запросы на сервер). Также представлены в виде дерева эффектов;
- Selectors – функции, которые сопоставляют состояние с определенным значением. Селекторы объединены с другими селекторами в один, с помощью функции createSelector().

Сценарий обмена данными начинается с представления компонентов. Некоторые взаимодействия, осуществляемые пользователем, могут привести к тому, что компонент отправит action. Если этот action не вызывает effect, reducer анализирует этот action и возвращает новое состояние, которое будет результатом слияния старого состояния со значением, измененным вызовом этого действия. Если effect запускается в результате отправки action, то перед вызовом reducer произойдут некоторые побочные эффекты. После того, как effect завершился, из него выстреливает новый action. Теперь в Store появилось новое состояние. С помощью селекторов настраивается подписка на изменения состояния с использованием Observable, поэтому компоненты сразу же «узнают» об изменениях в хранилище и перерисовывают интерфейс, если требуется. На схеме ниже представлен круговорот данных по шаблону Store.

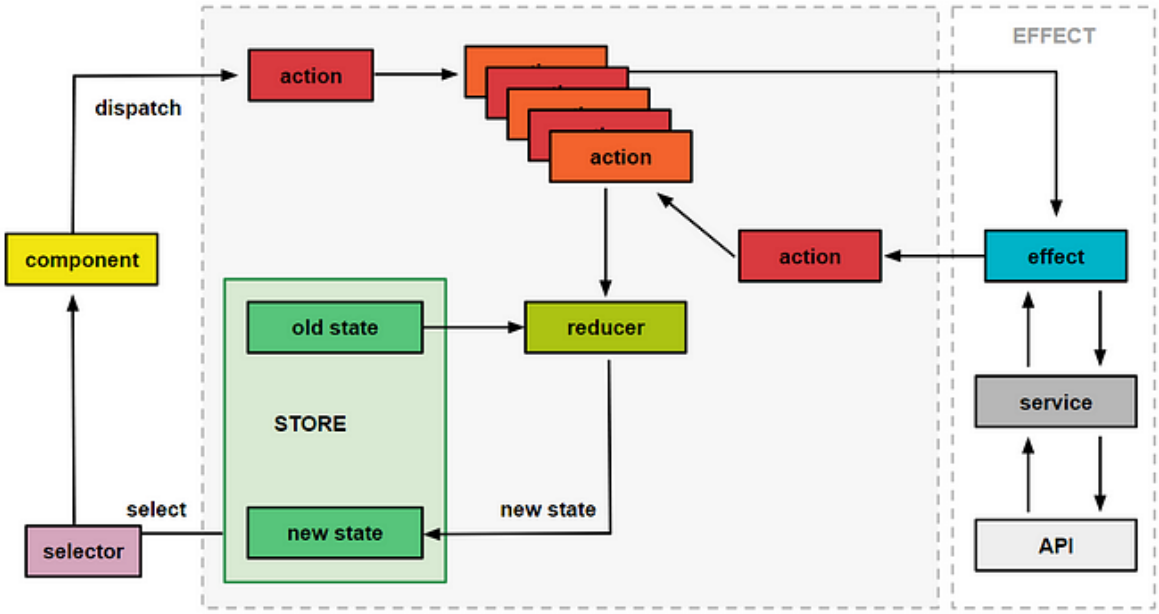

Рисунок 12 – Схема обмена данными в NgRx

2.2.3 Пользовательский интерфейс

В веб-приложении запланировано 6 разделов – «Сотрудники», «Рестораны», «FAQ», «Продукты», «События» и раздел авторизации. Всего в административной панели было создано 22 экрана. Для всех экранов приложения

были спроектированы макеты, ниже представлены макеты для главных экранов.

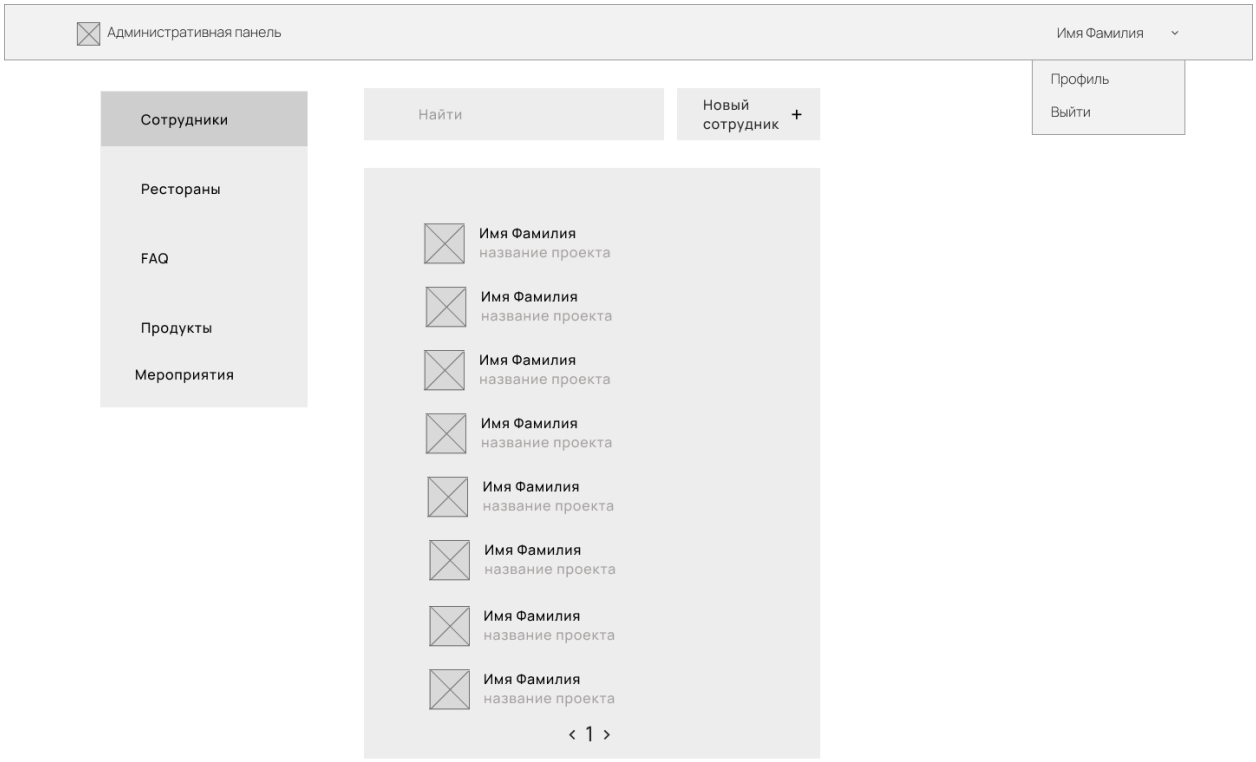

Рисунок 13 – Макет экрана "Сотрудники"

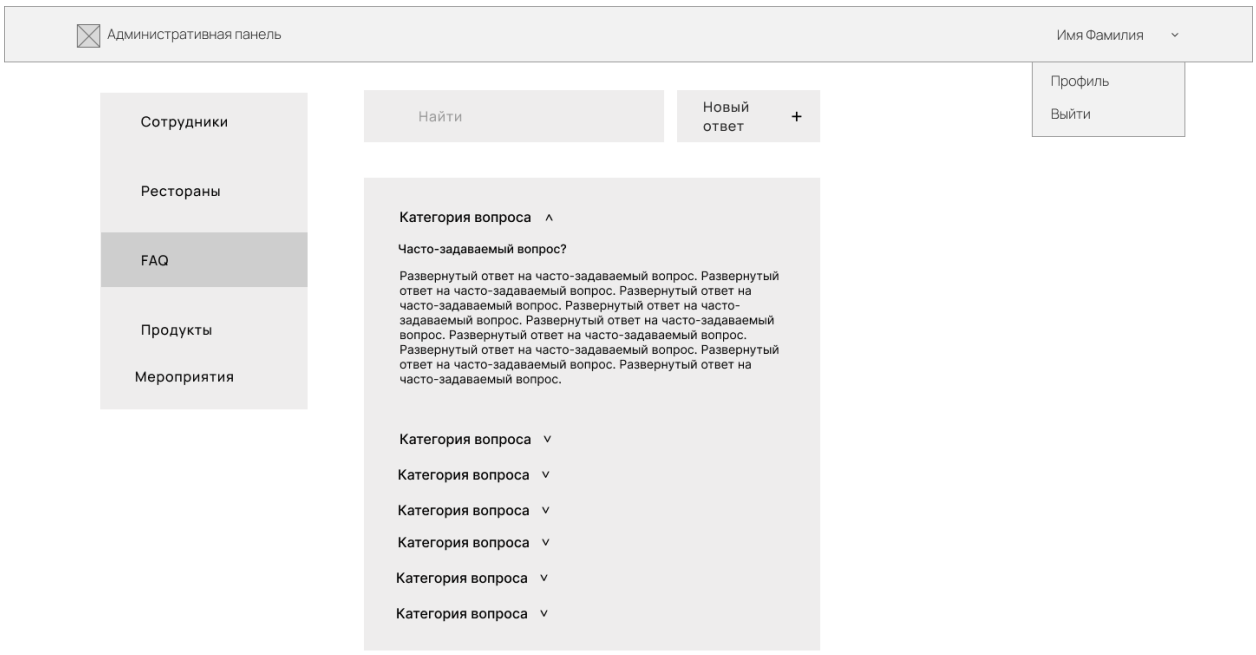

Рисунок 14 – Макет экрана "FAQ"

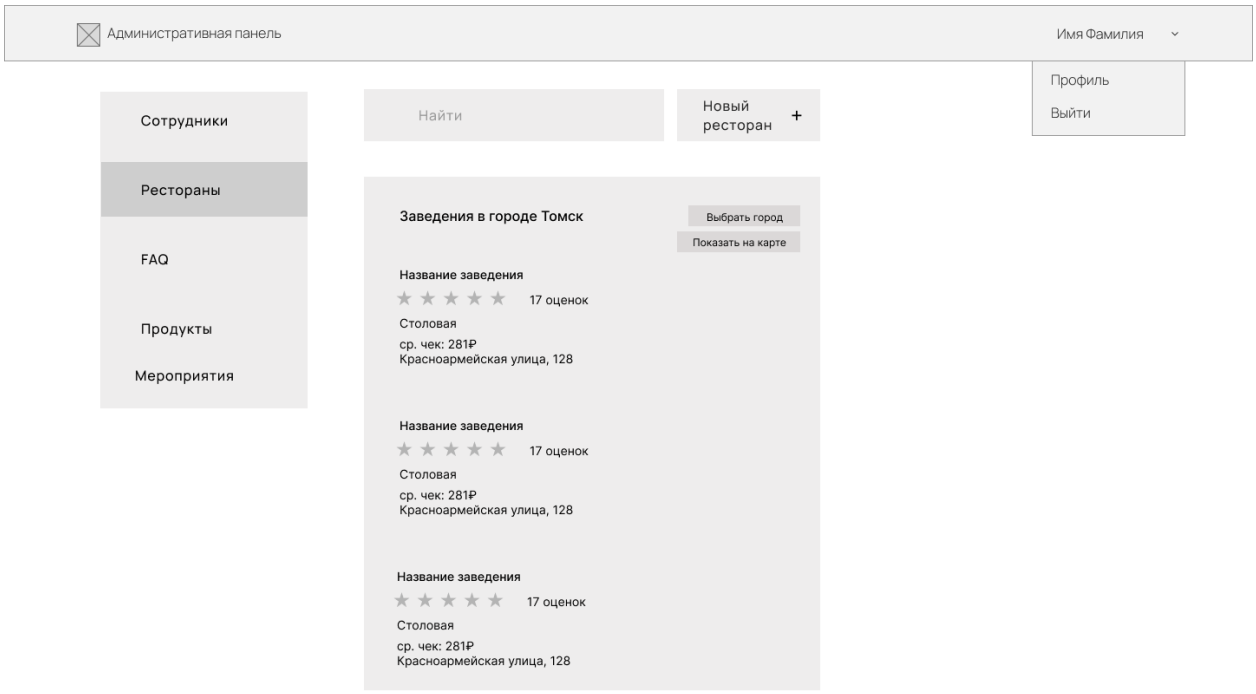

Рисунок 15 – Макет экрана "Рестораны"

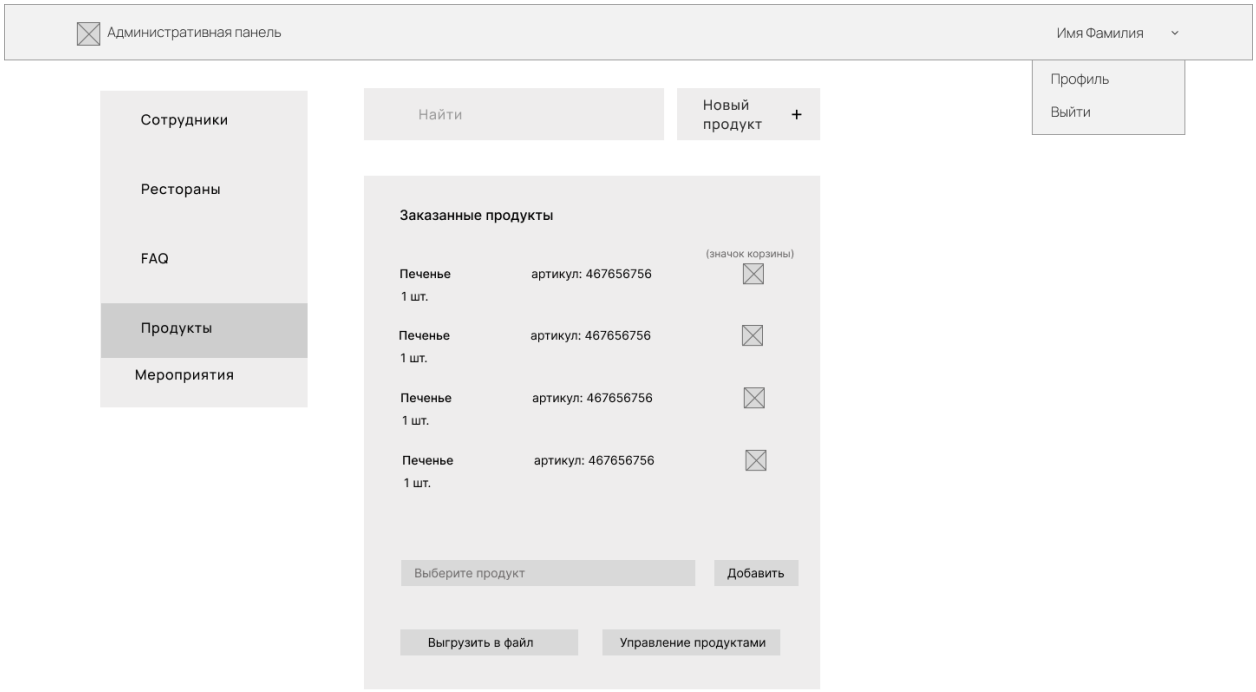

Рисунок 16 – Макет экрана "Заказанные продукты"

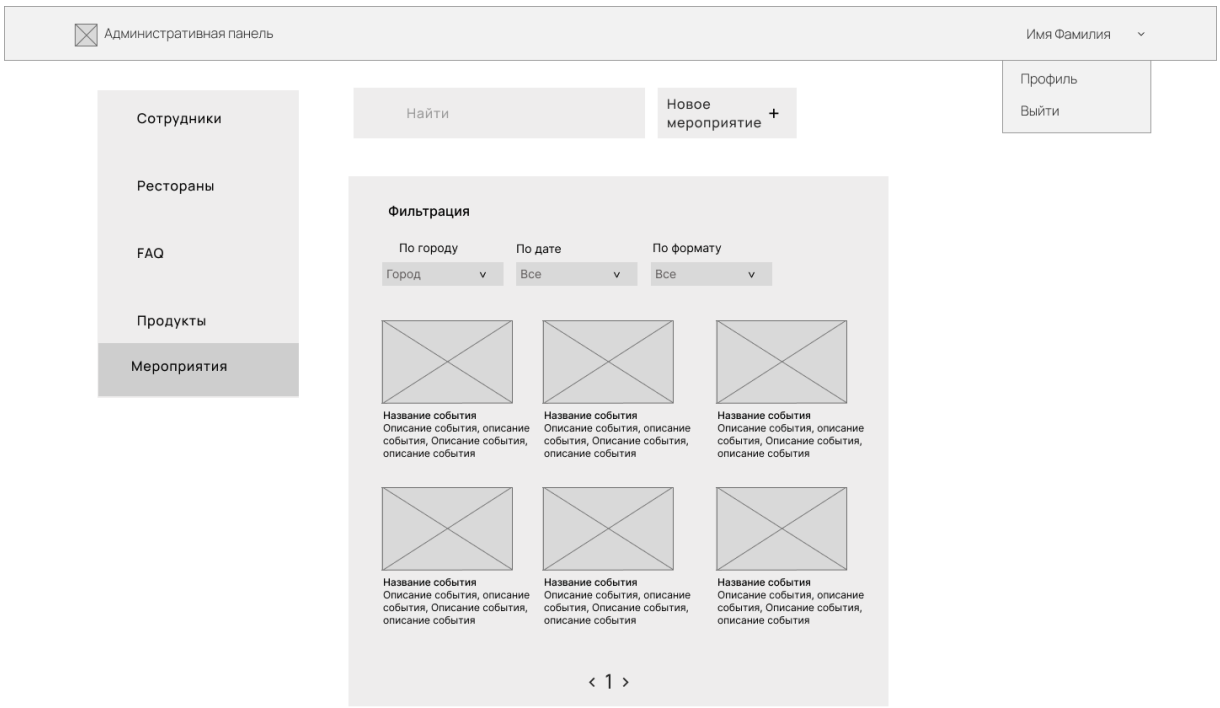

Рисунок 17 – Макет экрана "События"

Как видно из макетов, рабочая область веб-приложения состоит из трех компонентов:

- меню;
- шапка;
- контент.

В меню должны отображаться названия разделов, должна быть функция перехода на раздел по нажатию. Также при переходе в раздел в меню должна становится активной соответствующая разделу кнопка.

В шапке сайта размещены два блока – это название приложения (административная панель) и имя текущего пользователя (если пользователь уже прошел аутентификацию). Кликнув по имени пользователя, должно открываться меню с двумя кнопками: «Выйти» и «Профиль».

Компонент с контентом представляет из себя блок справа от меню, занимающий большее количество пространства сайта. Соответственно этот блок должен содержать основные интерфейсы и информацию приложения.

Главной страницей в веб-приложении является раздел «Сотрудники». Именно на него направляется пользователь после аутентификации.

2.2.4 Функциональность

Все функции административной панели представлены на диаграмме вариантов использования (см. приложение Е). Веб-приложением могут воспользоваться только сотрудники ТЦР, наделенные правами администратора. Поэтому для неавторизованных пользователей вариант использования один – авторизация.

Для авторизованного пользователя существует пять разделов и выход из аккаунтов. Разделы: «Сотрудники», «FAQ», «Рестораны», «Продукты», «События». Все разделы имеют базовый набор манипуляций с элементами раздела – CRUD (создать, посмотреть, редактировать, удалить).

2.2.5 Безопасность и конфиденциальность

Аутентификация в административной панели представлена посредством JWT (JSON Web Tokens). Данная технология подробно описана в разделе проектирования серверной части.

На фронтенд-части JWT-токен клиент получает после прохождения входа в систему. Выданный токен подставляется в заголовки запросов на сервер с помощью Angular HTTP Interceptor.

Через 5 минут токен становится неактуальным («протухает») и с первым отправляемым запросом уходит запрос на обновление запроса.

2.3 Проектирование серверной части

2.3.1 Диаграмма развертывания

На рисунке 18 представлен фрагмент диаграммы развертывания, подробно демонстрирующий серверную составляющую приложения.

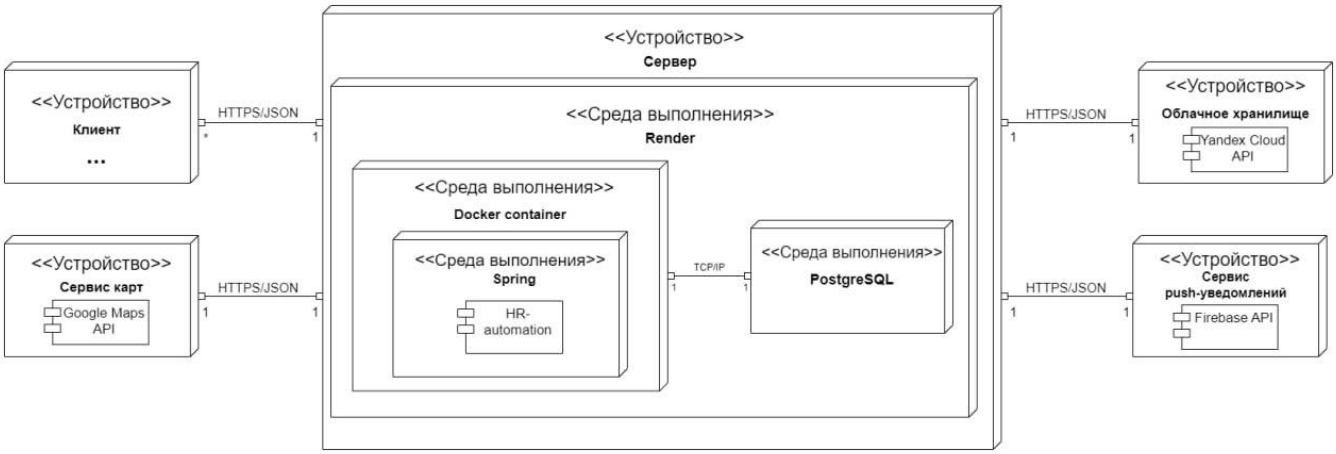

Рисунок 18 – Диаграмма развертывания

Данная диаграмма отражает, что проект, написанный на Spring, находится внутри Docker контейнера, который в свою очередь развернут на облачном сервисе Render. На этом же сервисе размещена база данных, которая общается с приложением по стеку протоколов TCP/IP. Также диаграмма демонстрирует интеграцию сервера с тремя сторонними сервисами: сервис карт Google Maps (для работы с картами в разделах «Рестораны» и «События»), облачное хранилище данных Yandex Cloud (для хранения изображений в разделах «Сотрудники», «Продукты», «События») и сервис отправки push-уведомлений Firebase (для информирования пользователей о появлении новых событий в разделе «События»). Со всеми этим сервисам, так же, как и с клиентами, связь происходит по протоколу HTTPS посредством JSON файлов.

2.3.2 Логическая модель базы данных

На рисунке 19 представлена логическая модель базы данных

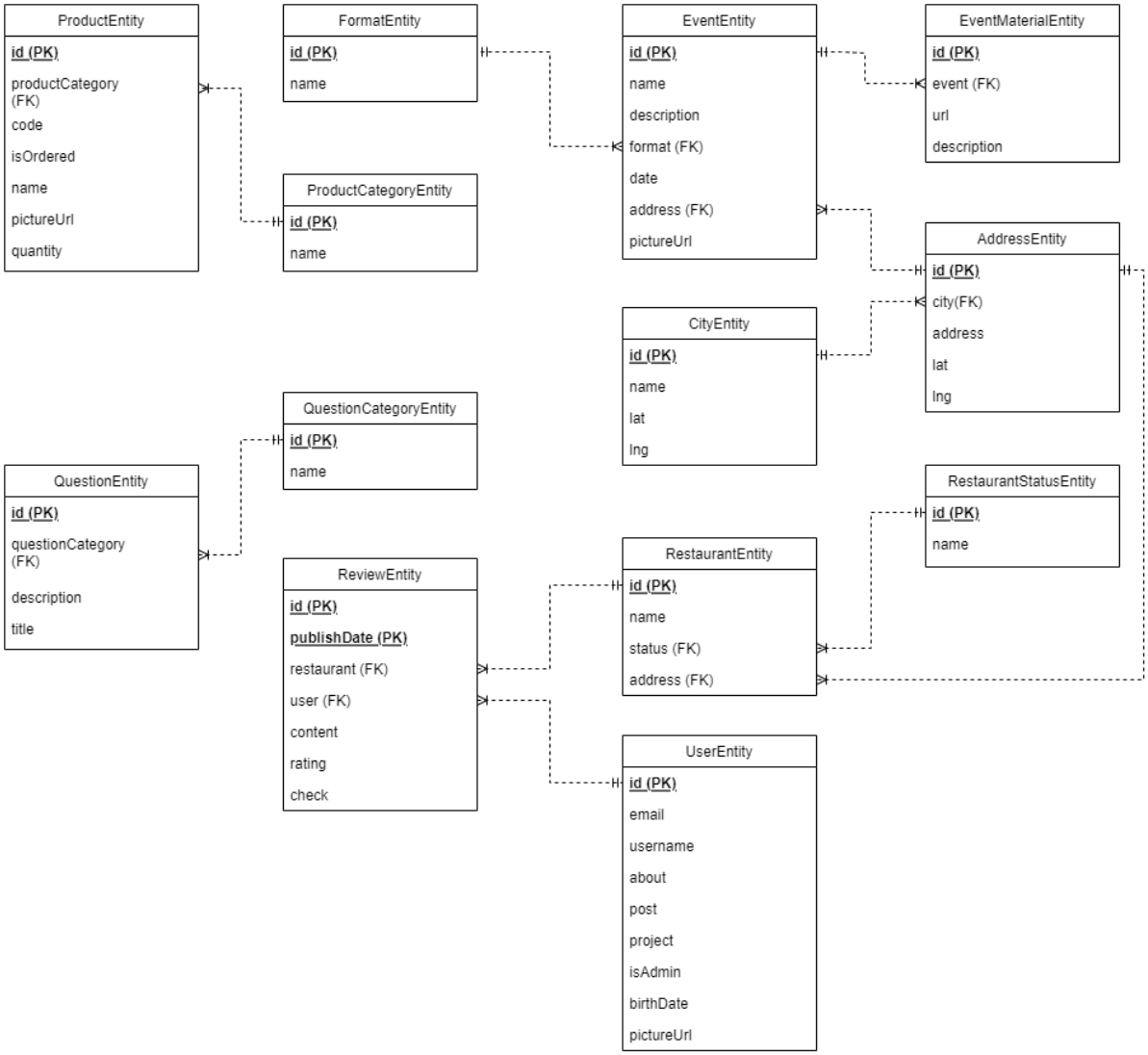

Рисунок 19 – Логическая модель базы данных

При проектировании базы данных было составлено 13 сущностей:

1. Сущность UserEntity содержит информацию о пользователях системы (адрес электронной почты, имя, информация «о себе», должность, проект, роль (администратор/не администратор), дату рождения, ссылка на фотографию профиля);

2. Сущность QuestionEntity содержит информацию о часто задаваемых вопросах (заголовок вопроса, ответ на вопрос, категория вопроса);

3. Сущность QuestionCategoryEntity представляет собой перечень категорий вопросов;

4. Сущность ProductEntity содержит информацию о продуктах, доступных для заказа (категория продукта, артикул, название, статус (заказан/не заказан), количество, ссылка на фотографию продукта);

5. Сущность ProductCategoryEntity представляет собой перечень категорий продукта;

6. Сущность CityEntity содержит информацию о городах, в которых расположены ТЦР (название, широта, долгота);

7. Сущность EventEntity содержит информацию о мероприятиях, проводимых для сотрудников (название, описание, формат проведения, дата проведения, адрес проведения, ссылка на фотографию);

8. Сущность FormatEntity представляет собой перечень форматов проведения мероприятий;

9. Сущность EventMaterialEntity содержит информацию о материалах к мероприятию (мероприятие, ссылка на материал, описание);

10. Сущность AddressEntity содержит информацию об адресах (город, адрес, широта, долгота);

11. Сущность RestaurantEntity содержит информацию о ресторанах (название, статус, адрес);

12. Сущность RestaurantStatusEntity представляет собой перечень статусов заведений;

13. Сущность ReviewEntity содержит информацию об отзывах на рестораны (дата публикации, ресторан, пользователь, содержание, рейтинг, чек).

2.3.3 Проектирование системы аутентификации

Проект разрабатывается для пользования сотрудниками ТЦР, поэтому в нем отсутствует возможность прямой регистрации. Всех пользователей регистрирует администратор, используя корпоративную почту сотрудника. Также невозможно самостоятельно удалить свой профиль – удаление профилей осуществляет администратор (например, при увольнении сотрудника). Вход в приложение осуществляется с помощью одноразового четырехзначного кода, который приходит сотруднику на корпоративную почту. Это делает систему более безопасной и надежной, так как не требует постоянного пароля (который можно забыть или передать третьим лицам). Во время использования мобильного и веб-приложения для аутентификации пользователя используется технология JSON Web Token (JWT). С помощью нее в заголовке каждого запроса передается уникальный для каждого пользователя токен, позволяющий серверу идентифицировать того, кто отправляет ему запрос. Токен может содержать любые необходимые данные о пользователе (например, его id, имя, электронную почту), что позволяет сократить количество обращений к базе данных и увеличить производительность сервера. Для того, чтобы защитить токены от перехвата злоумышленниками, используется токен обновления. При таком подходе основной токен доступа имеет очень короткий срок жизни (обычно несколько минут) и по его истечении он обновляется с помощью дополнительного токена, время жизни которого достигает нескольких дней, однако он одноразовый. Таким образом при истечении срока жизни токена доступа клиент запрашивает его обновление, присылая токен обновления, после чего сервер отдает ему новую пару из токена доступа и токена обновления.

2.4 Проектирование мобильного приложения

2.4.1 Введение

Согласно имеющимся требованиям, мобильное приложение предназначено для клиентской части пользователей – сотрудников ТЦР и должно содержать следующие функциональные модули:

- «Сотрудники» - просмотр и поиск сотрудников, просмотр детальной информации о сотруднике;
- «Продукты» - просмотр и заказ продуктов с возможностью фильтрации по категориям;
- «FAQ» - просмотр категорий часто задаваемых вопросов, просмотр ответов на вопрос;
- «События» - просмотр временной линии событий, прошедших и предстоящих, фильтрация событий по параметрам, просмотр деталей события;
- «Рестораны» - просмотр карты с заведениями города, в котором есть ТЦР. Просмотр списка заведений города. Просмотр деталей заведения. Просмотр и создание отзывов о заведении;
- «Профиль» - экран для просмотра и редактирования личной информации сотрудника;
- «Авторизация» - комплексный модуль, содержащий экраны ввода почты и кода авторизации, механизмы по предоставлению, хранению и обновлению токена авторизации и идентификатора пользователя.

2.4.2 Архитектура

2.4.2.1 Общая архитектура системы

Рекомендуемая и наиболее часто используемая архитектура для Android приложений – это так называемая «Чистая Архитектура», призванная решить сразу несколько проблем при разработке:

- тестируемость;
- независимость от пользовательского интерфейса;
- независимость от баз данных;
- независимость от внешних фреймворков и библиотек.

Для достижения поставленной задачи необходимо разделить систему на некоторые слои, при этом изменения в одном слое не затронут изменения в другом. В нашем случае, при разработке Android приложения необходимо разделить систему на слои:

- Презентационный – зависимый от фреймворка. технологий и платформы слой. Содержит исключительно логику отображения уже поступивших данных, обработанных бизнес-правилами;
- Доменный – абстрактный слой, определяющий сущности системы в общем виде и содержащий бизнес-правила;
- Data-слой – содержит источники данных и модели, принятые для взаимодействия с этими источниками.

Описанная модель изображена на рисунке 20

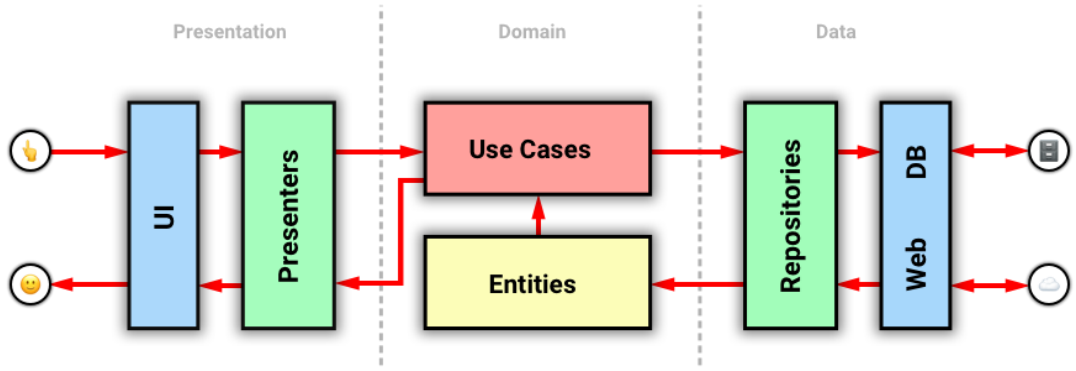

Рисунок 20 – Чистая архитектура

Таким образом, поток данных можно описать следующим образом: пользователь взаимодействует с графическим интерфейсом, UI логика обрабатывает его взаимодействие и передает данные в Use Case – вариант использования, который определен бизнес-логикой приложения, далее Use Case запрашивает у репозитория получение или передачу данных с некоторого источника, которым может являться как локальная база данных, так и удаленный сервер, к которому необходимо совершить запрос. Таким образом репозиторий ответственен за исходные данные. В каждом слое могут использоваться свои сущности, для перехода этих сущностей между слоями используются маппперы, получающие на вход модель одного типа и возвращающие другой.

2.4.2.2 Архитектура презентационного слоя

Для каждого слоя может использоваться своя внутренняя архитектура. Например, при разработке приложений для операционной системы Android, презентационный слой описывается одним из нескольких паттернов:

MVVM – Model-View-View Model, где Model – сущности с которыми происходит взаимодействие, View – представление сущностей, View Model – логика представления сущностей. Важно отметить, что в данном подходе View Model не знает о том, какая View за ним закреплена, но View знает о своей View Model. Визуальное представление паттерна изображено на рисунке 21

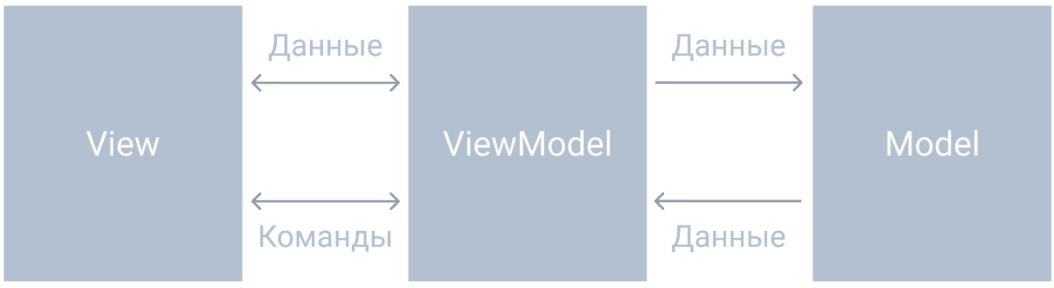

Рисунок 21 – Паттерн MVVM

MVI – Model-View-Intent. Паттерн особенно полезный для разработки сложных экранов, на которых состояние элементов зависит от состояний других. Использует понятие «Состояние экрана», для хранения состояния всех элементов. Отметим большие недостатки паттерна – уязвимость к гонке состояний, необходимость отслеживания разрушения процесса, изменение состояния вызывает рендеринг всех элементов. Визуально представление паттерна изображено на рисунке 22

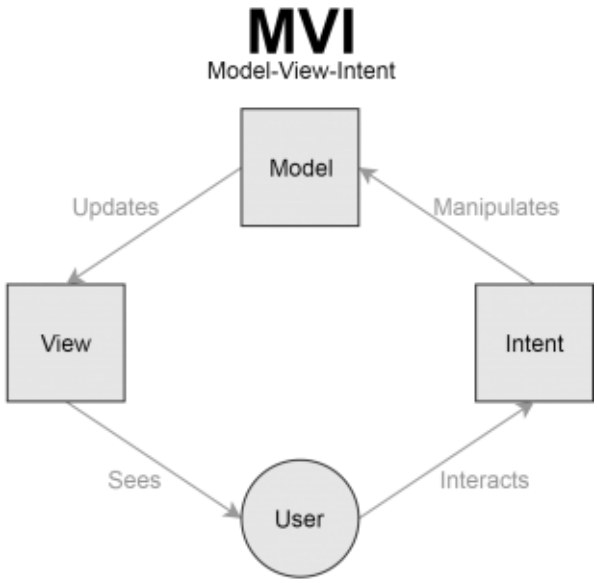

Рисунок 22 – Паттерн MVI

MVP – Model-View-Presenter. Ключевое отличие от MVVM – Presenter знает о том, какое представление View с ним взаимодействует. Отличия паттерна можно увидеть на рисунке 23

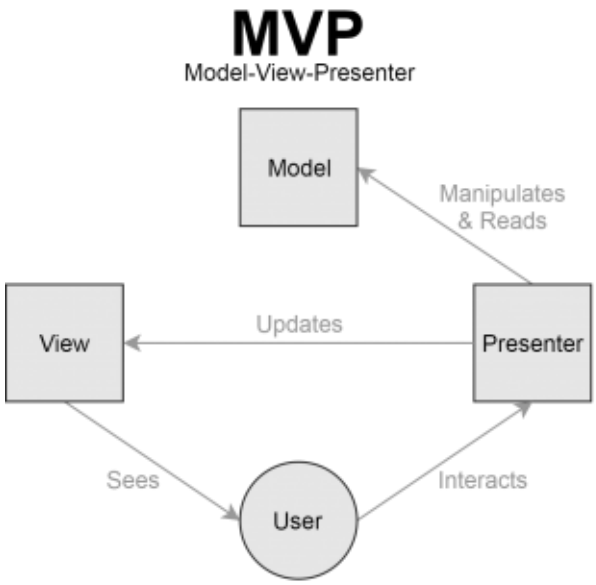

Рисунок 23 – Паттерн MVP

MVC – Model-View-Controller. Отличается от MVP тем, что Controller, в отличие от Presenter’а, не обновляет View самостоятельно, View отслеживает изменения полей Controller’а. Визуализация паттерна отображена на рисунке 24.

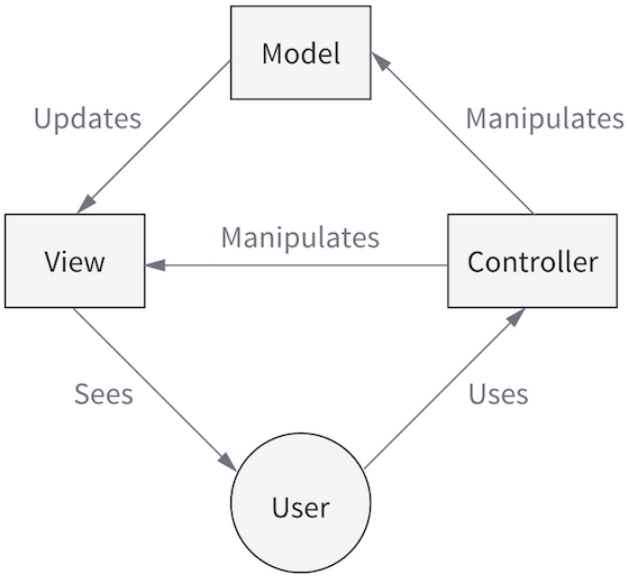

Рисунок 24 – Паттерн MVC

Рассматривая перечисленные паттерны проектирования, можно заметить, что наименьшим количеством зависимостей и наибольшей простотой обладает MVVM, так как его модель отображения – View Model, не знает о своей View и реализуема в Android фреймворке готовым набором инструментов, в связи с этим, возьмем его как основной для разработки.

Заметим, что проект по большей части не имеет сложной бизнес-логики, в связи с чем Use Case в доменном слое будет излишним, поэтому опустим данный элемент при проектировании.

2.4.2.3 Компоненты

Обладая вышеупомянутой информацией по архитектуре, можно спроектировать диаграмму компонентов разрабатываемого приложения для более общего понимания предстоящих задач. Рассмотрим часть слоя презентации на рисунке ниже

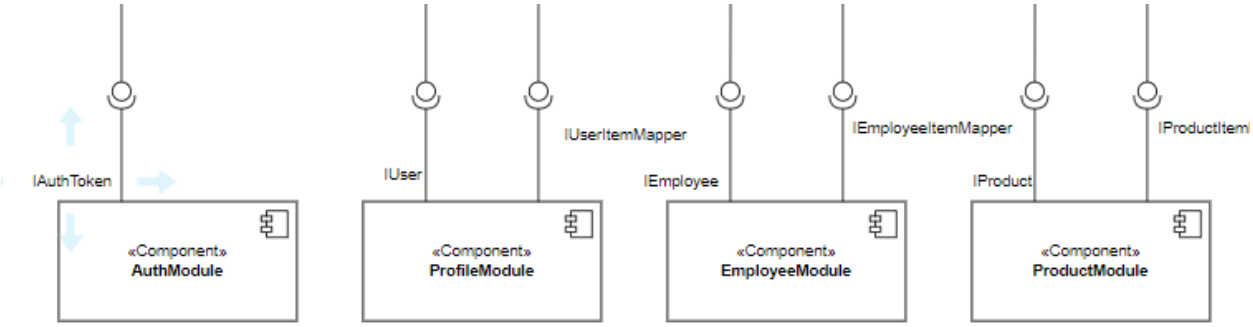

Рисунок 25 – Компоненты слоя презентации

На данной диаграмме модули — это совокупность классов, ответственных за соответствующий функциональный модуль приложения. Компоненты принимают на вход объект реализующий определенный интерфейс. Все принимаемые объекты используются соответствующей реализацией маппера, который также предоставляется модулю, для преобразования из доменной модели в презентационный. Фрагмент слоя данных изображен на рисунке 26.

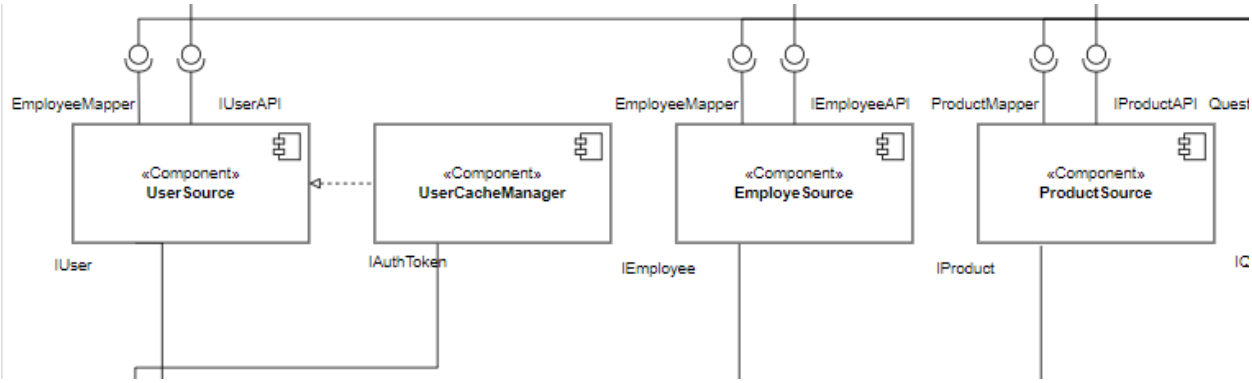

Рисунок 26 – Компоненты слоя данных

Компоненты с постфиксом «Source» представляют собой репозиторий для управления объектами слоя данных, и, соответственно, сами классы этих объектов. Здесь также необходимы соответствующие мапперы, но уже для преобразования из моделей данных в модели домена. Репозиторий получает данные с помощью API, реализующего соответствующий репозиторию интерфейс. API предоставляет методы для запроса данных с сервера.

Рисунком ниже рассмотрим компоненты, предоставляющие API и выполняющие запросы к серверу, так как некоторые из них, используют различные реализации клиента, в зависимости от того, необходима ли для его работы авторизация.

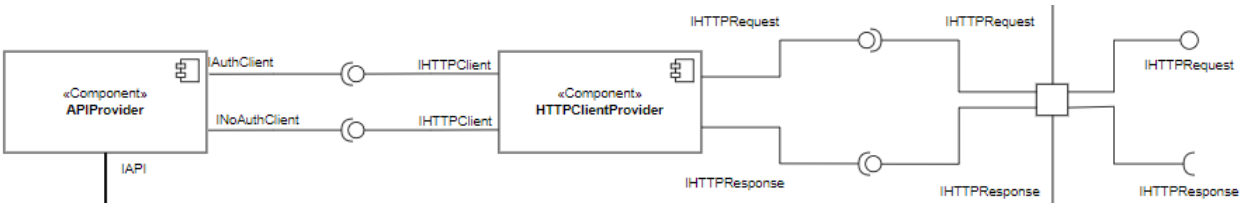

Рисунок 27 – Компоненты API

Полная диаграмма компонентов (см. приложение Ж).

ГЛАВА 3. Реализация

3.1 Реализация административной панели

3.1.1 Раздел «Авторизация»

Разработка административной панели началась с раздела аутентификации пользователя (в коде соответствующий компонент называется «auth»). В процессе работы предстояло создать две страницы: с полем для ввода электронной почты и с полем для ввода ключа, который приходит на почту сотрудника после отправки запроса на аутентификацию.

Но основная сложность была не в создании разметки, а в написании логики. Задачи были следующие: отправлять два запроса на сервер (первый с введенной электронной почтой, второй с введённым ключом) и сохранять в системе токен, полученный с сервера после успешного прохождения аутентификации.

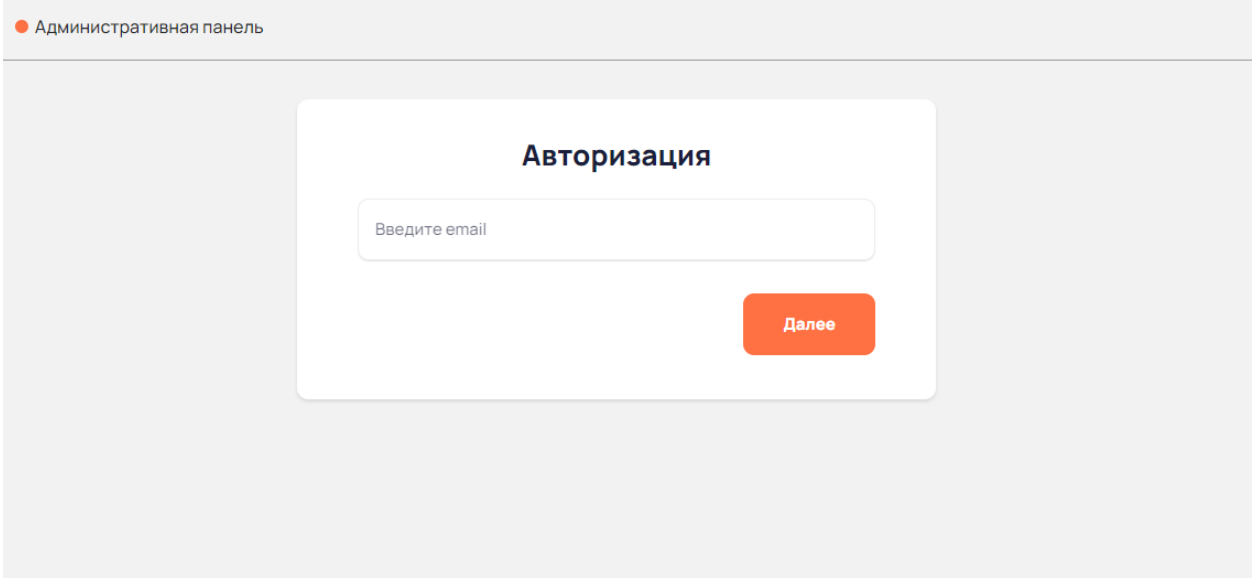

Рисунок 28 – Экран с авторизацией

Далее был разработан раздел «Сотрудники». Так как это уже часть приложения, а приложение, согласно требованиям, должно быть доступно только

авторизованным пользователям, этот и последующие разрабатываемые разделы были защищены от неавторизованных посетителей с помощью Route Guards.

Следующая часть кода описывает класс AuthGuard, который отвечает за перенаправление на страницу с авторизацией, если в системе не сохранен токен.

```
export class AuthGuard implements CanActivate {
  constructor(private tokenService: TokenService, private router: Router)
{}
  canActivate(
    route: ActivatedRouteSnapshot,
    state: RouterStateSnapshot
  ):
    | Observable<boolean | UrlTree>
    | Promise<boolean | UrlTree>
    | boolean
    | UrlTree {
    if (!this.tokenService.isLoggedIn()) {
      this.router.navigate(['login']);
      return false;
    } else {
      return true;
    }
  }
}
```

3.1.2 Раздел «Сотрудники»

В разделе «Сотрудники» было создано 4 экрана: страница со списком всех сотрудников и поиском по имени, страница с подробной информацией и фото выбранного сотрудника, страница с редактированием информации выбранного сотрудника и страница с созданием нового сотрудника. На страницу создания можно перейти, кликнув по кнопке «Добавить» в верхней части блока с контентом. На страницу сотрудника, кликнул на фото или имя любого сотрудника из списка. Переход на страницу с редактированием информации

осуществляется после нажатия кнопки «Редактировать» на странице пользователя. На рисунке 29 представлен экран со списком сотрудников, отсортированных по id.

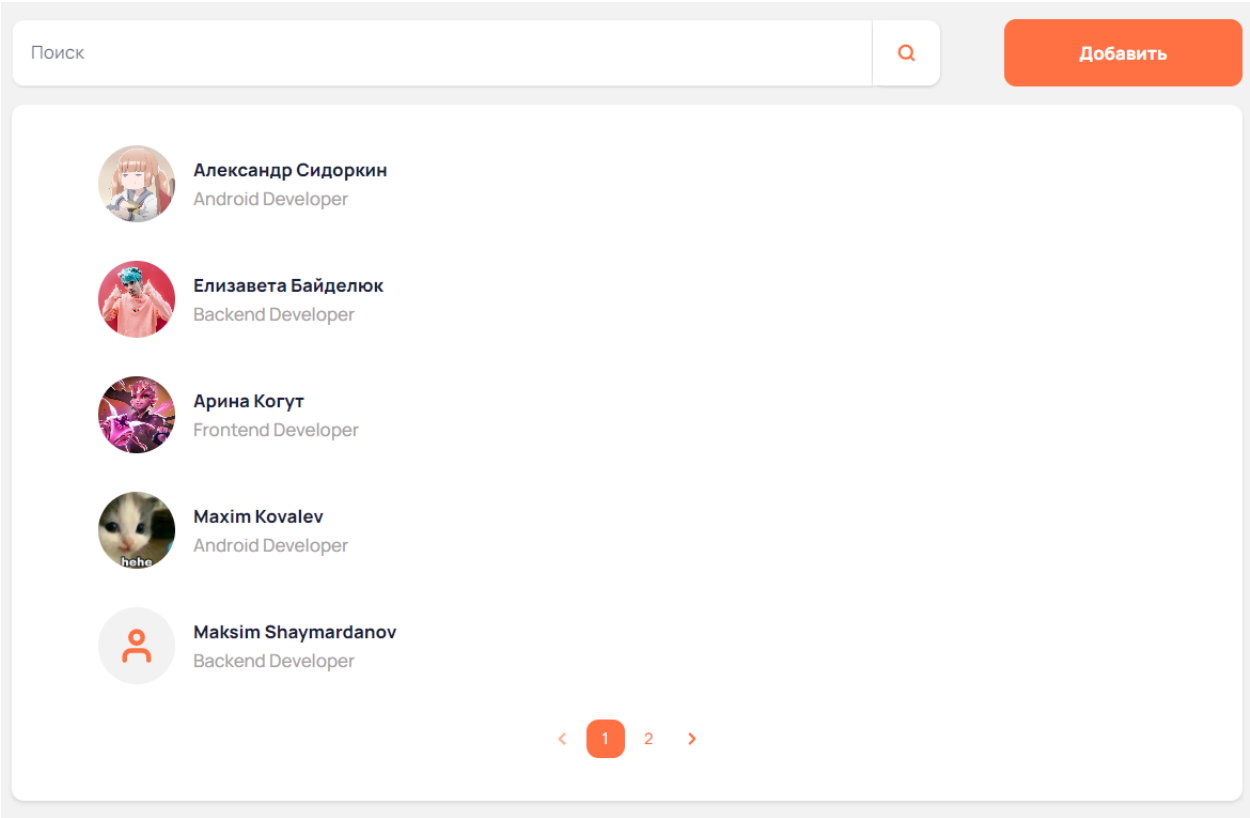

Рисунок 29 – Экран со списком сотрудников

3.1.3 Раздел «FAQ»

Следующим разделом в разработке стал раздел «FAQ». Это раздел с ответами на часто-задаваемые вопросы. Для пользователей доступны следующие страницы: список вопросов с ответами, разделенными по категориям, добавление нового ответа на часто-задаваемый вопрос, редактирование ответа и создание новой категории вопросов. На рисунке 30 представлен экран со списком часто-задаваемых вопросов.

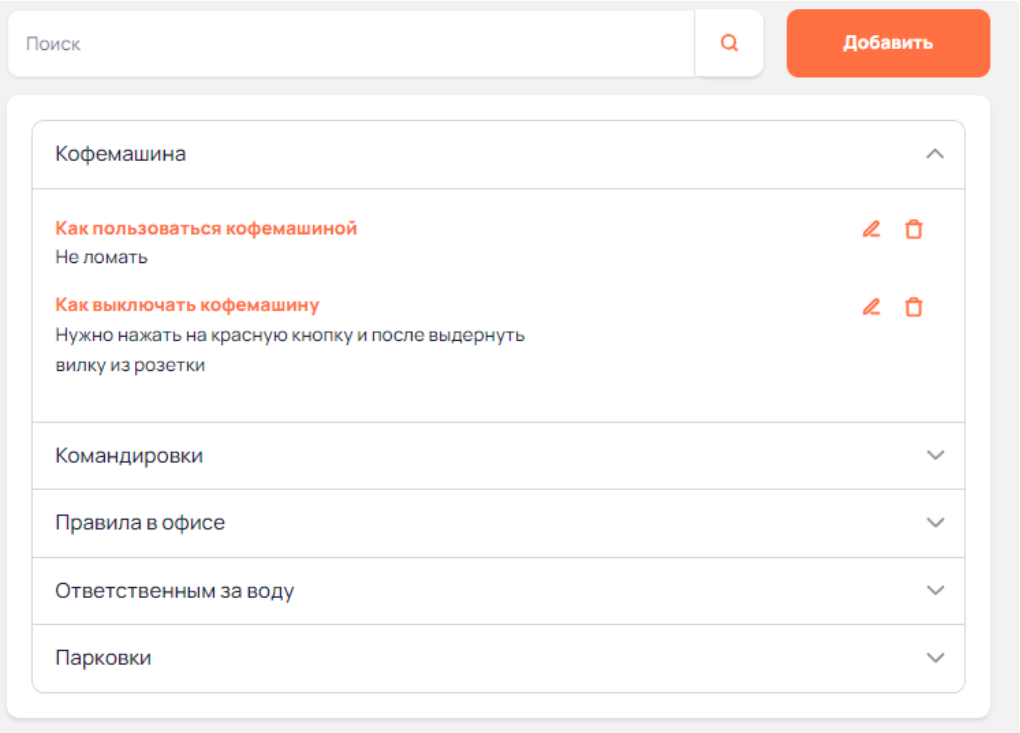

Рисунок 30 – Экран со списком часто-задаваемых вопросов

Представленная страница является главной в разделе, то есть появляется при переходе на раздел. На другие страницы раздела «FAQ» пользователь попадает именно через нее.

3.1.4 Раздел «Продукты»

Следующим разрабатывался раздел «Продукты». Отличительной чертой этого раздела является дополнительный экран со списком заказанных продуктов, полем для добавления продукта в заказанные и кнопкой для выгрузки файла со списком (рисунок 31). Остальные экраны (список всех продуктов, редактирование и создание) были сделаны по аналогии с предыдущими разделами.

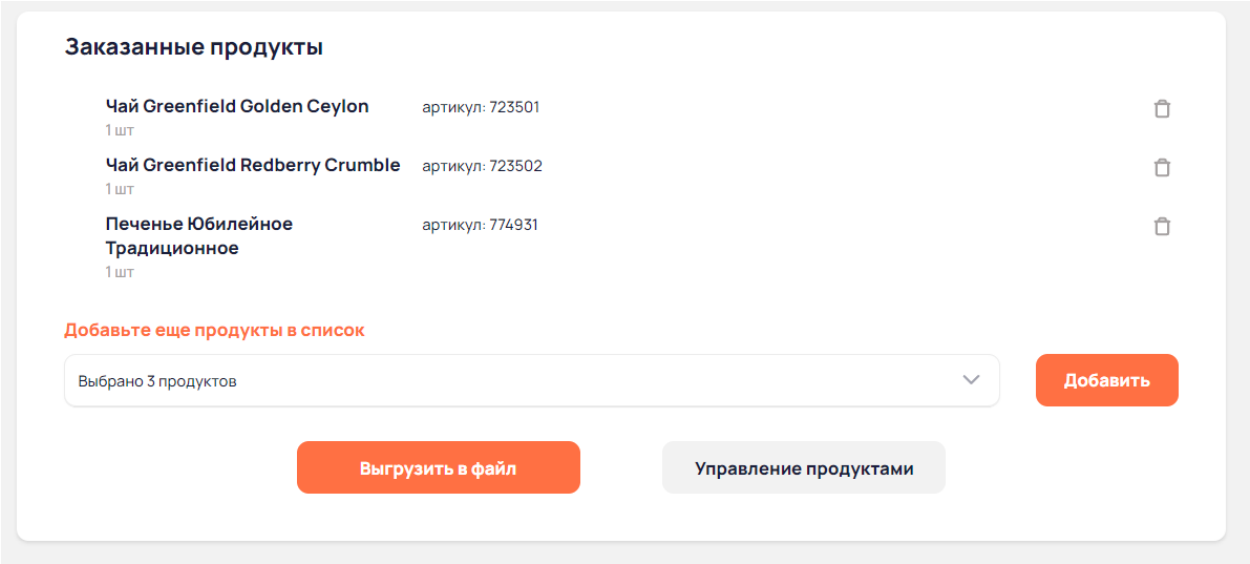

Рисунок 31 – Экран с заказанными продуктами

Таким образом, сложностью реализации раздела «Продукты» являлась логика создания выгрузки файла на локальную машину. Ниже представлена часть кода с функцией, закрепленной за кнопкой «Выгрузить в файл».

```
submitProducts() {
    // формирование имени файла с текущей датой
    this.today =
      `Заказанные продукты на ` +
      new Date().getDate() +
      '.' +
      (new Date().getMonth() + 1) +
      '.xls';
      // кнопка переходит в состояние загрузки
    this.download = true;
    // находим элемент с id "link"
    const element = document.getElementById('link');
    // получаем файл, помещаем в store
    this.store$.dispatch(getFile());
    // "забираем файл" из store
    const file = this.store$.select(selectFile);
    // подписываемся на поток данных "file"
    file
      .pipe(
        tap((file) => {
          if (file != null) {
            // создаем object URL для файла
```

```
            const url = URL.createObjectURL(file);
            // динамически добавляем атрибут к DOM-элементу
            element?.setAttribute('href', url);
            // создаем имитацию клика по ссылке
            if (element != null) element.click();
            URL.revokeObjectURL(url);
            // кнопка переходит в обыычное состояние
            this.download = false;
          }
        }),
        takeUntil(this.destroy$)
      )
      .subscribe();
  }
```

Основная логика заключается в том, что создается невидимый элемент (ссылка с атрибутом «display: none») на странице. Далее формируется url на файл и прикрепляется атрибутом к этой ссылке. Когда url прикреплен, имитируется нажатие на ссылку и файл скачивается.

3.1.5 Раздел «Рестораны»

Далее разрабатывался раздел «Рестораны». Он, так же, как и остальные, имел четыре экрана (список, создание, ресторан, редактирование) и дополнительно два модальных окна: с выбором города и просмотром ресторанов на карте.

Много времени ушло на внедрение google-maps-api в веб-приложение. Нужно было сначала ознакомиться с документацией, а после написать код для карты с маркерами ресторанов и карты для страницы отдельного ресторана (с маркером, но без модального окна при нажатии на него).

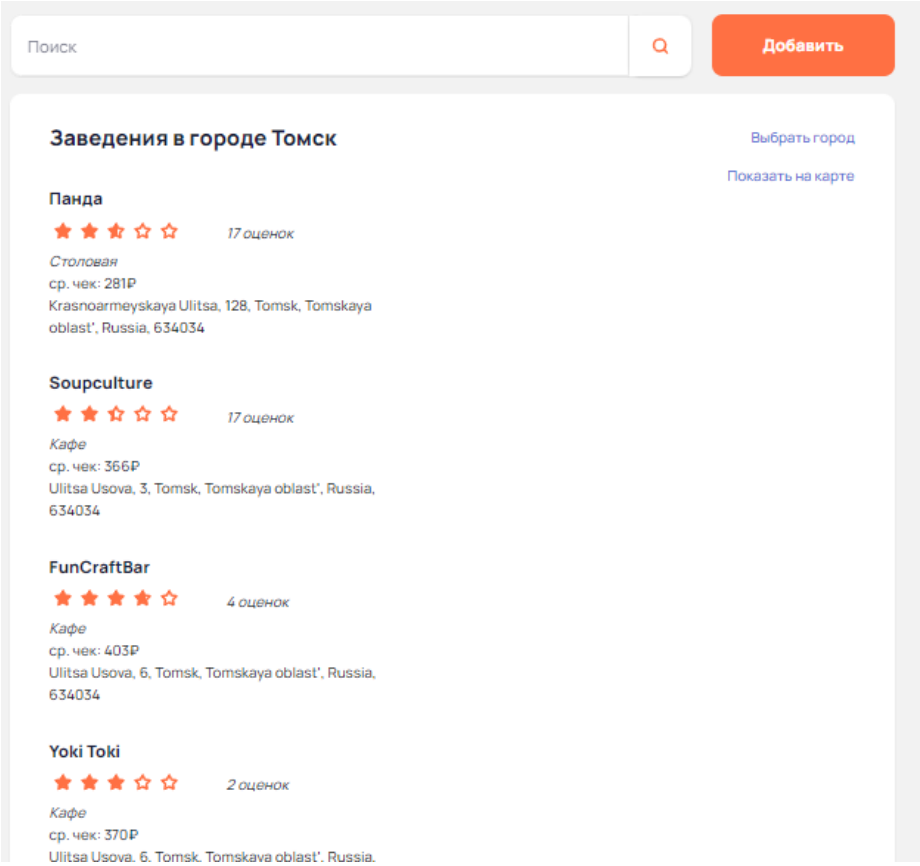

Рисунок 32 – Экран со списком ресторанов

3.1.6 Раздел «События»

Последним разрабатываемым разделом стали «События». Раздел посвящён внутренним мероприятиям и должен предоставлять интерфейсы для создания мероприятия, редактирования мероприятия, просмотра списка доступных мероприятий и просмотра отдельного события с подробной информацией о нем.

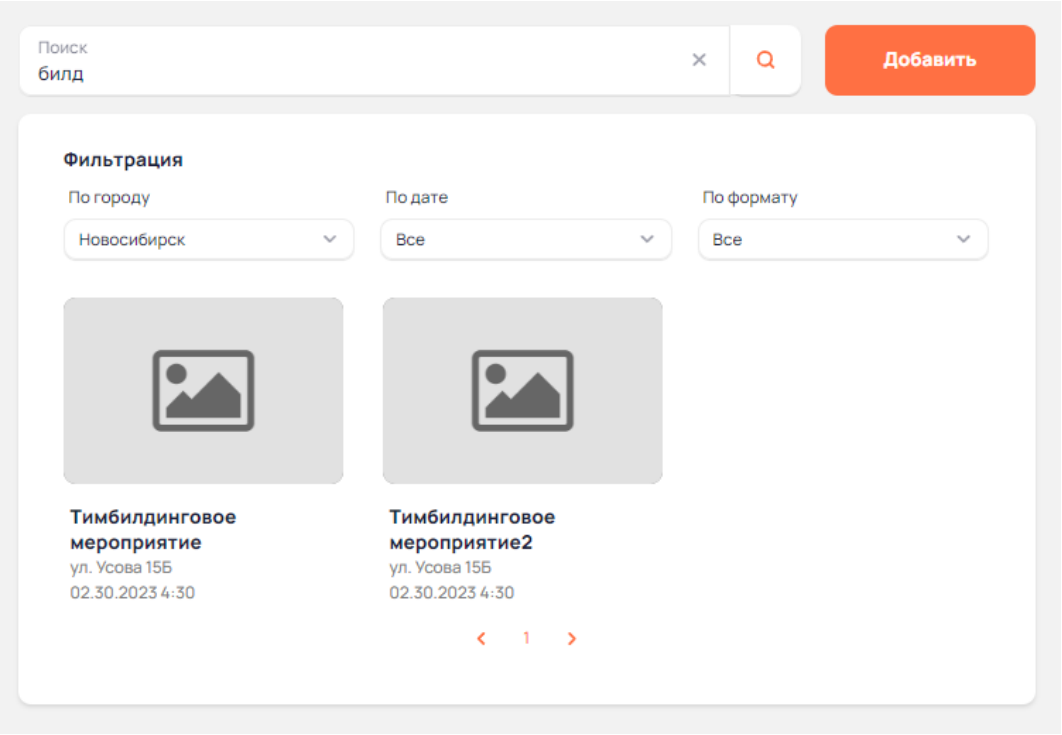

Рисунок 33 – Экран со списком событий

3.1.7 Итоги

Почти все задачи фронтенд-разработчиком были выполнены самостоятельно без помощи куратора, такой подход был полезен для глубокого погружения в язык программирования и фреймворк. Однако куратор проекта проводил code review, указывая на ошибки в коде. Все замечания были исправлялись, и с каждым code review выявленных проблем становилось меньше.

3.2 Реализация серверной части

3.2.1 Создание проекта

Для быстрого создания проекта на Spring был использован ресурс start.spring.io. Он позволяет выбрать язык и его версию, сборщик и сразу добавить некоторые необходимые зависимости. Проект был создан для Java 17, Gradle – Groovy, Spring Boot 2.7.4. Также первоначально были добавлены следующие зависимости:

1. Spring Web – стартер для создания веб-приложений, в том числе RESTful, с использованием Spring MVC;
2. Spring Data JPA – стартер для использования Spring Data JPA с Hibernate;
3. Spring Boot DevTools – инструмент, обеспечивающий быстрый перезапуск приложения при обнаружении изменений, LiveReload и расширяет возможности разработки с помощью конфигураций.

3.2.2 Аутентификация и авторизация пользователей

Первым этапом работы стала разработка системы регистрации и аутентификации пользователей. Важно отметить, что для Spring уже существует фреймворк Spring Security, реализующий функции, связанные с регистрацией и аутентификацией. Однако было принято решение не использовать данный фреймворк для того, чтобы лучше понять механизмы работы данных процессов. Для аутентификации было разработано два эндпоинта – для обычных пользователей и для администраторов. Сервер получает от клиента электронную почту и проверяет существует ли пользователь с такой электронной почтой в базе данных. Во втором случае дополнительно проверяется есть ли у сотрудника с этой почтой права доступа к административной панели. Если все проверки прошли успешно, то генерируется одноразовый код и высылается на указанную почту. Для реализации этой части процесса аутентификации используется интерфейс JavaMailSender, который входит в Spring Framework. Если пользователь вводит верный код, то сервер отдает пару из двух токенов для дальнейшей авторизации в процессе использования приложения.

3.2.3 Раздел «Сотрудники»

Так как все вышеописанные процессы уже подразумевают работу с пользователями, далее было решено разрабатывать раздел «Сотрудники». Для сущности сотрудника были реализованы стандартные CRUD-операции, а

также поиск по имени. Также на данном этапе понадобилось подключить S3 хранилище, для чего был использован Yandex Cloud. В сущности каждого пользователя хранится ссылка на фотографию профиля, которая хранится в бакете на Yandex Cloud.

3.2.4 Развертывание проекта и создание документации

Параллельно с разработкой также шел процесс развертывания сервера на Render. Изначально для развертывания использовался OpenJDK, однако на этапе добавления в проект S3 хранилища выяснилось, что в используемой версии jdk есть баг при подключении метрик AWS S3. Поэтому в качестве jdk был выбран Eclipse Temurin.

После того как сервер стал легкодоступен другим участникам разработки, возникла необходимость создания документации для быстрого координирования работы внутри команды. Для автоматического создания документации использовался инструмент apiDoc. Он обладает очень простым синтаксисом и позволяет с помощью конфигурационного файла и тегов в контроллерах создавать удобную документацию в формате веб-страницы. Данная страница была размещена на облачном хостинге GitHub Pages. На рисунке 34 представлен пример сгенерированной документации.

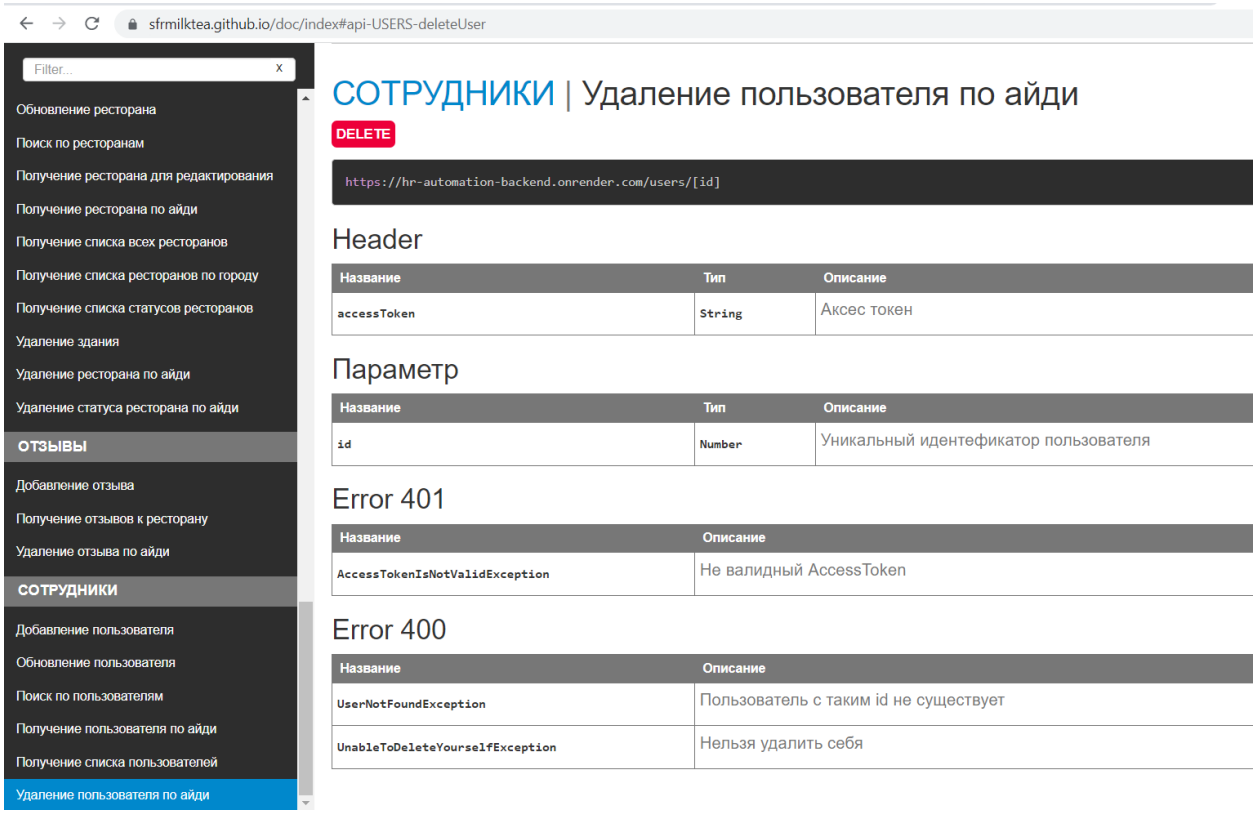

Рисунок 34 – Пример документации

3.2.5 Раздел «FAQ»

Следующим разрабатываемым разделом был выбран «FAQ». Часто задаваемые вопросы разделены на категории, поэтому помимо CRUD-операций для самих вопросов также были написаны CRUD-операции для категорий вопросов. Кроме того, реализован GET запрос для получения вопросов, принадлежащих определенной категории и поиск по ключевым словам.

3.2.6 Раздел «Продукты»

Одной из основных функций приложения является автоматизация процесса сбора информации о необходимости заказать продукты в офис, поэтому следующим разделом был разработан раздел «Продукты». Аналогично разделу «FAQ» продукты разделяются на категории, поэтому как для продуктов, так и для категорий были реализованы CRUD-операции, получение продуктов по категории и поиск. Также были написаны методы для изменения статуса

продукта (заказан/не заказан). Еще одной из функций приложения является возможность для администратора выгрузить Excel файл с продуктами, которые необходимо заказать. Для работы с Excel документами был интегрирован API Apache POI. С помощью него формируется документ, содержащий названия продуктов, их артикулы и количество необходимое к заказу. Пример формируемого файла представлен на рисунке 35.

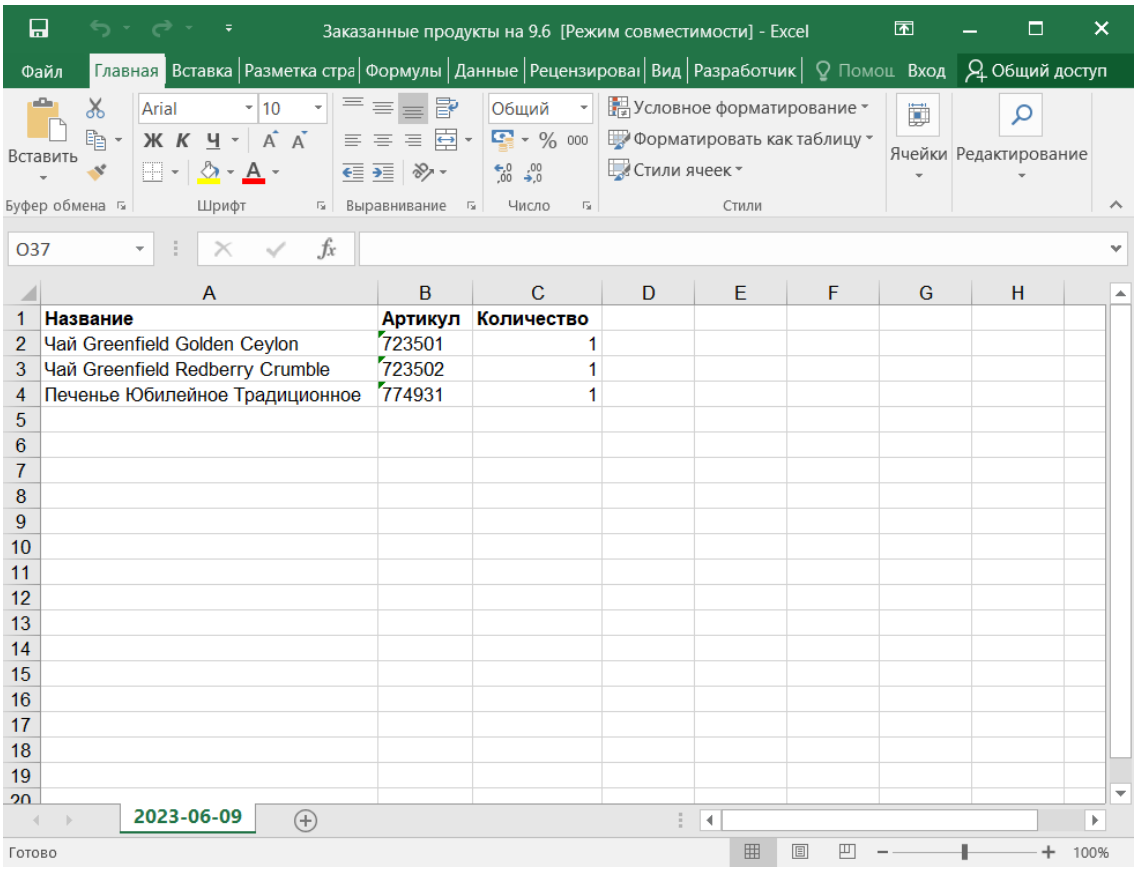

Рисунок 35 – Пример Excel файла с продуктами для заказа

3.2.7 Раздел «Рестораны»

Вслед за разделом «Продукты» был разработан раздел «Рестораны», позволяющий пользователям просматривать информацию о ближайших заведениях и делиться отзывами на них. Для ресторанов и отзывов были написаны все общие для всех разделов методы. Для наглядного просмотра заведений

было принято решение использовать Google Maps API. Для отображения ресторанов на карте при добавлении или обновлении каждого объекта введенный пользователем адрес преобразуется в координаты с помощью геокодера. В базе данных хранятся широта и долгота ресторана, которые используются клиентом для размещения маркеров на карте. Также возможно и обратное – пользователь размещает маркер на карте, клиент передает координаты маркера, а сервер делает запрос на преобразует координаты в адрес. В процессе работы возникла проблема с тем, что иногда в одном здании могут находиться несколько ресторанов. В таком случае маркеры накладываются друг на друга и пользоваться картой в таком случае становится неудобно. Было принято решение использовать кластеризацию ресторанов и добавить новую сущность – Здание. Таким образом координаты сохраняются для зданий, а рестораны содержат лишь идентификатор здания, в котором находятся. Для ресторанов также были написаны алгоритмы расчета среднего рейтинга и среднего чека на основе отзывов пользователей.

3.2.8 Раздел «События»

Последним из разрабатываемых разделов стали «События». Для событий были написаны стандартные CRUD-операции. Также было принято решение предоставить пользователю возможность фильтровать события по городу, формату, дате и названию. Для такой единовременной фильтрации был реализован отдельный интерфейс, использующий Criteria API. Данный инструмент позволяет имитировать SQL-запросы для получения из базы данных объектов, соответствующих необходимым фильтрам. Одной из отличительных задач в разделе «События» была реализация push-уведомлений о появлении новых событий в приложении. Для этого в проект была интегрирована система Firebase. При установке мобильного приложения сервер получает токен устройства, который впоследствии используется для отправки уведомлений.

3.3 Реализация мобильного приложения

3.3.1 Каркас приложения

Разметим пакеты приложения соответственно намеченной архитектуре. В слое данных будем разрабатывать API с помощью Retrofit, интерсептор аутентификации, аутентификатор токенов, модели слоя по каждому разделу, репозитории для каждого раздела и медиаконтента устройства, а также кэш-менеджеры и репозиторий токенов.

Для пакета, предназначенного для внедрения зависимостей с помощью Dagger, также определим разработку модуля, предоставляющего API соответствующий интерфейсу, требуемому в репозитории, а в каждый из них клиент, требующий авторизации или нет. Второй класс-провайдер условно можно назвать TokenApiProvider, так как необходим только для запросов по проверке почты, получению и обновлению токена. Там же будем разрабатывать классы предоставляющие утилиты по типу мапперов и ресурс-провайдеров, фабрику новых View Model, предоставление контекста, а также репозиториев.

В доменном слое разместим лишь доменные модели и интерфейсы репозиториев, именно их будут требовать View Model при создании, а Dagger будет создавать их, предоставляя необходимые зависимости.

В презентационном слое разместим модели слоя, сервис для уведомлений, имплементацию провайдера строковых ресурсов, представления, являющиеся важнейшей частью нашего приложения, определяющие логику представлений и работу с репозиториями (подробнее о ключевых моментах и особенностях при работе с представлениями будет рассказано позже), а также базовые классы, позволяющие вынести общий и часто повторяющийся код:

- базовые классы для активностей и фрагментов системы android;

- базовые делегаты для удаления повторяющегося кода при использовании библиотеки adapter delegates;
- базовые view model.

Наконец, отдельно создадим пакет для утилит, в который поместим классы для работы с графическими ресурсами, датами, публикаторы программных событий, утилиты, необходимые для работы конкретных модулей. В этом пакете будем разрабатывать интерфейсы провайдера строковых ресурсов, маппера и других вспомогательных классов, фабрику View Model, утилиты для адаптера списков, расширения для многопоточности. Там же разместим очень важную утилиту для работы со сменой изображений загрузочных представлений.

3.3.2 Разработка представлений

3.3.2.1 Представления авторизации

Начнем разработку с создания загрузочной активности, на которой будет отображен так называемый Splash-screen, обычно показывающий логотип компании, а в нашем случае, отображать иконку приложения во весь экран.

Рисунок 36 – Ресурс для отображения на загрузочном экране

Загрузочная активность будет перенаправлять пользователя на активность, содержащую два фрагмента: фрагмент для ввода кода и фрагмент для ввода почты. После ввода почты, View Model фрагмента отправит ее для проверки на сервер. В случае положительного ответа почта считается валидной, и пользователь сможет ввести код с почты, который так же будет проверен сервером. В случае отправки верного кода сервер вернет токен авторизации и идентификатор пользователя, который только что авторизовался, мы запишем их в соответствующих репозиториях, которые для запоминания таких параметров могут использовать Shared Preferences – небольшое приватное пространство для настроек приложения.

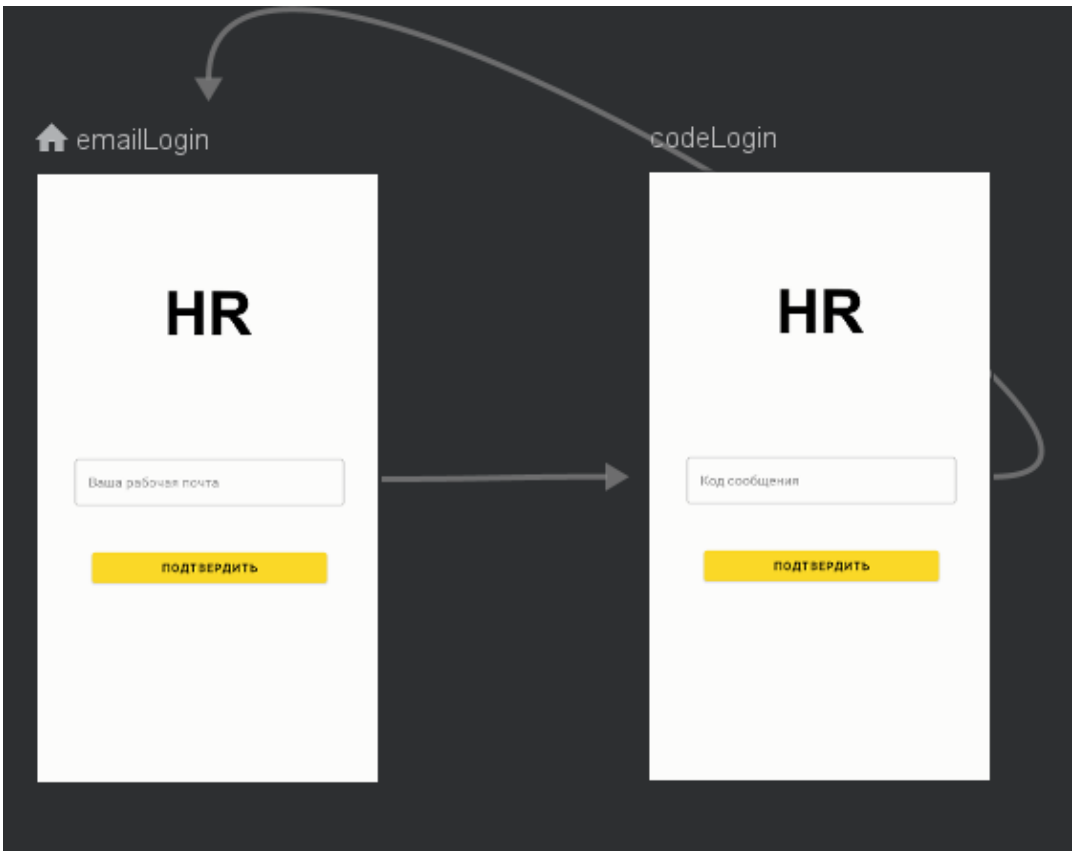

Рисунок 37 – Навигационный граф активности авторизации

Обладание токеном теперь позволяет создать авторизованный клиент и использовать в нем наш токен авторизации, который будет подставляться в каждый запрос при помощи интерсептора, и обновляться, если будет просрочен, при помощи аутентификатора.

3.3.2.2 Главная активность

После авторизации пользователю будет показан главный экран, который реализован в виде активности, содержащей пять фрагментов, организованных с помощью нижней навигационной панели и графа навигации. фрагмента раздела «Сотрудники» и нижнюю навигационную панель можно рассмотреть на рисунке 38.

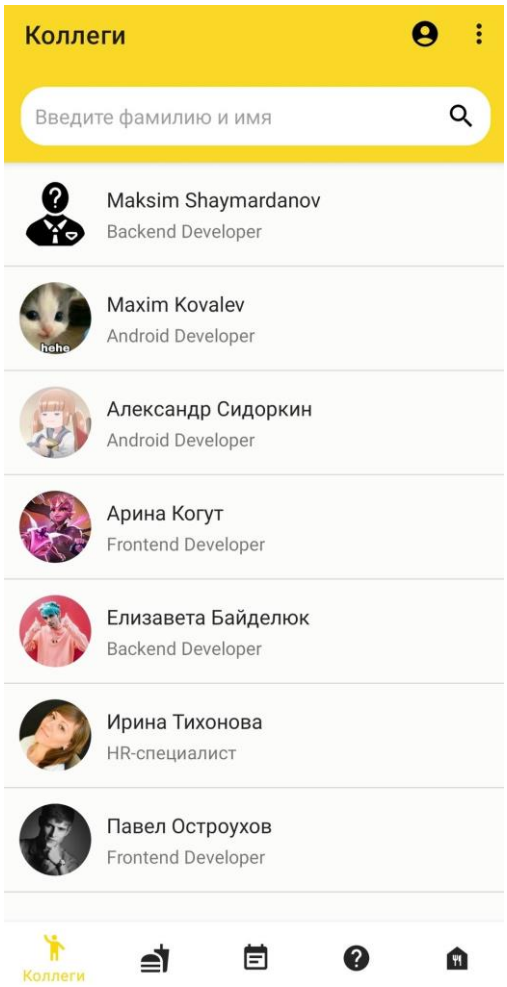

Рисунок 38 – Экран раздела «Сотрудники»

Отметим, что главная активность отслеживает навигацию пользователя по приложению и самостоятельно управляет панелью инструментов, которая также видна на каждом фрагменте, как и навигация. Панель инструментов всегда содержит кнопку перехода в личный кабинет и опции с кнопкой выхода из аккаунта. Главная активность также будет отвечать за выход из приложения – для этого достаточно дважды нажать на системную кнопку «назад».

3.3.2.3 Фрагмент «Сотрудники»

Общий план работы для создания функционала по показу информации и, в частности, в списках следующий:

1. View Model должна запросить у репозитория данные, которые в свою очередь поступают с сервера и проходят стадии маппинга до презентационного слоя. Пока данные не пришли, утилита переключения между состояниями экрана «загрузка» и «отображение» показывает соответствующий значок загрузки;

2. Реализация паттерна «Наблюдатель» позволяет View Model отправить новые данные, а View – получить эти данные. Если при получении данных в какой-то момент произошла ошибка – View Model поймает ошибку с помощью упомянутой утилиты для корутин и отправит ошибку отдельному наблюдателю ошибок во View, который с использованием утилиты отображения заменит экран на состояние ошибки, который позволит перезагрузить экран заново;

3. View отображает данные в списке при помощи элемента Recycler View и соответствующего адаптера.

Функционал фильтрации по введенным символам в поисковой строке осуществляет поиск соответствий строки имени сотрудника с введенной пользователем строкой. При нажатии на сотрудника открывается активность деталей сотрудника. Просмотр деталей объекта происходит по общей схеме:

1. Идентификатор объекта передается в новую активность;

2. Активность через свою View Model отправляет запрос в репозиторий на получение детальной информации по идентификатору;

3. Данные отслеживаются представлением и распределяются по полям.

3.3.2.4 Фрагмент «Продукты»

Отличительной особенностью фрагмента «Продукты» от сотрудников является дополнительное диалоговое окно для заказа продукта, а также использование специального элемента Chips из библиотеки Material Design для

фильтрации продуктов по категориям. Таким образом, сперва необходимо определить категорию продуктов, которые нужно запросить с бэкенда, по умолчанию запросятся все продукты. После выбора категории список соответствующих продуктов будет предоставлен сервером. Кнопки выбора категорий будут помещены в панель инструментов. Выделенные особенности отображены на рисунках 39-40.

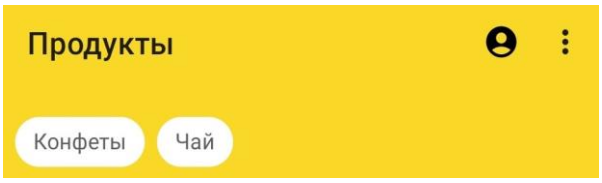

Рисунок 39 – Панель инструментов с категориями продуктов

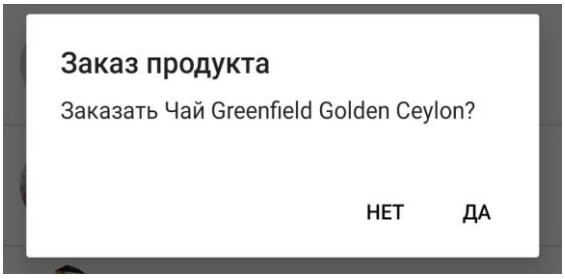

Рисунок 40 – Диалоговое окно заказа продуктов

3.3.2.5 Фрагмент «FAQ»

Фрагмент не обладает отличительными особенностями, которые бы мы не обсуждали: необходимо получить данные списка – категории вопросов. Затем на активности выбранной категории вопроса показать вопросы и ответы на них.

3.3.2.6 Активность «Профиль»

Особенностью активности является необходимость отправить на сервер данные, обновленные пользователем. Для этого достаточно сообщить об этом репозиторию. В случае с обновлением картинки профиля все не так просто. Для отправления таких файлов необходимо отправлять их в теле запроса как

файл или, что более желательно, файл из множества частей, что в нашем случае означает формирование Multipart Body, очень удобно предоставляемым библиотекой Retrofit.

Чтобы получить фотографию с устройства нужно выполнить следующие действия:

1. Получить разрешения от пользователя на доступ к требуемым данным;
2. Использовать контракты активностей результатов с указанием типа контракта для получения контента и указания типа контента как изображения;
3. Далее могут возникнуть трудности, ведь вместо самой картинки мы получим ссылку на нее в директории устройства в виде URI. По данной ссылке получим представление изображения – Bitmap и получим из него массив байт, чтобы отправить в репозиторий для составления Multipart Body.

После оповещения об успешной смене картинки необходимо обновить список сотрудников на совершенно другом фрагменте «Сотрудники». Для этого используем утилиту «Публикатор» и оповестим фрагмент о необходимости перезагрузить список.

3.3.2.7 Фрагмент «Рестораны»

Крупный фрагмент с множеством новых особенностей:

1. Использование вкладок в панели для разделения карты с заведениями от списка заведений города;
2. Нижняя выдвигающаяся панель для выбора города;
3. Нижняя выдвигающаяся панель для обзора заведений в одном здании;
4. Выдвигающаяся карточка для просмотра краткой информации о заведении;

5. Карта с маркерами заведений.

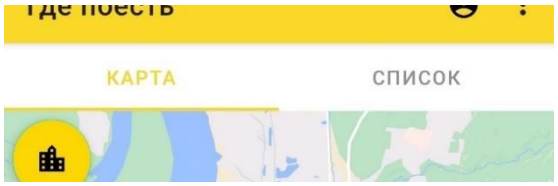

Рисунок 41 – Вкладка карты

Рисунок 42 – Вкладка списка

Нижняя выдвигающаяся панель – это компонент Android, который можно вызвать как диалоговое окно, но он при этом может быть выполнен в виде фрагмента, показывающегося поверх другого. Реализацию компонента в данном приложении можно увидеть на рисунках 43-45.

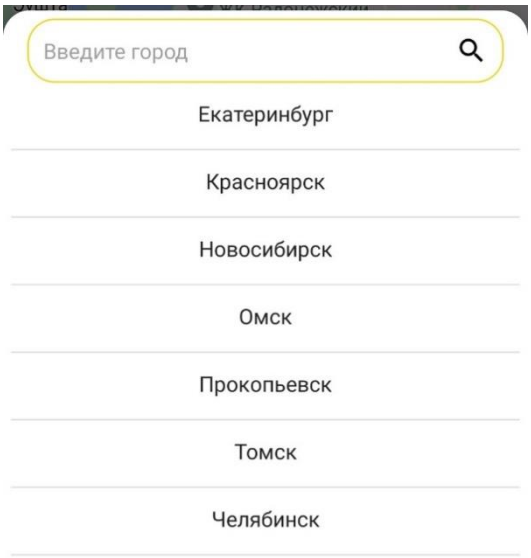

Рисунок 43 – Фрагмент с городами

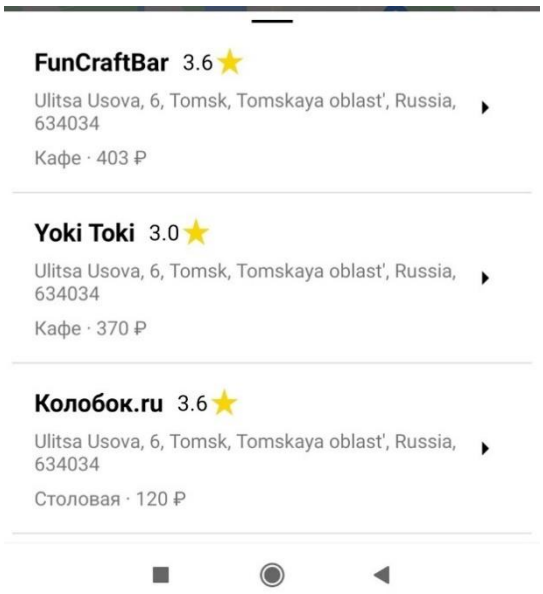

Рисунок 44 – Фрагмент с заведениями

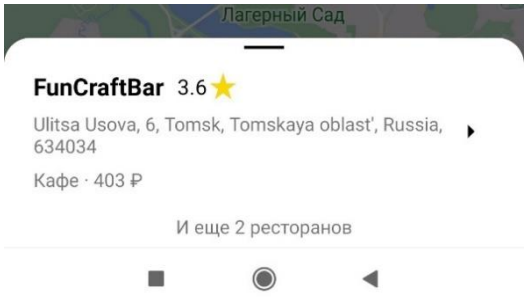

Рисунок 45 – Фрагмент с заведениями - скрытый

Выдвигающаяся карточка – компонент, который требуется создать самостоятельно. В данном случае ее можно получить совмещением других компонентов в одном представлении, описав предоставив дополнительно методы и анимацию для появления и скрытия карточки, смены данных.

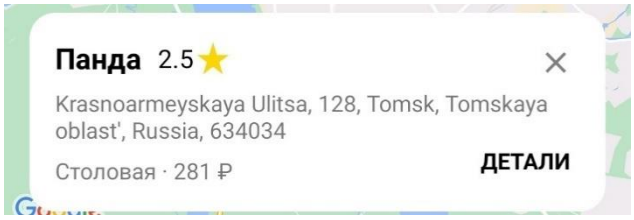

Рисунок 46 – Выдвигающаяся карточка

Для более удобной работы с картой и вынесения этой логики в отдельный класс требовалось написать адаптер, который в итоге будет осуществлять выбор маркера, постановку маркеров на карту, устанавливать слушатель нажатия на маркер, перемещать карту по городам. Так как наша карта отображает здания с одним рестораном внутри или несколькими, то необходимо разработать различные маркеры для обеспечения некоторой кластеризации. О том, какой маркер необходимо применить, мы будем судить о количестве ресторанов внутри. Таким образом, к адаптеру карты разработан класс Marker Delegate, предоставляющий к созданию своих наследников – маркер с одним рестораном и маркер с числом, показывающим количество ресторанов. Для маркера с числом разработаем и используем утилиту-провайдер, которая будет рисовать на исходном маркере, представленном классом Drawable, число ресторанов с использованием инструмента Canvas. На рисунке ниже можно увидеть маркеры на карте.

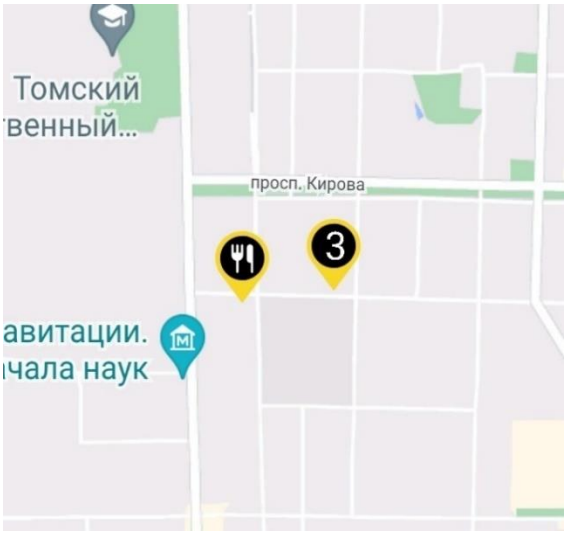

Рисунок 47 – Маркеры на карте

В разделе «Рестораны» можно посмотреть детали ресторана и отзывы сотрудников, написать собственный отзыв.

3.3.2.8 Фрагмент «События»

Отличием раздела является наличие временной линии, по которой эти события разделяются на предстоящие и прошедшие. Определение происходит по сравнению дат. События можно отфильтровать по нескольким параметрам: название, дата, город проведения, формат. Параметры фильтрации отправляются на сервер, откуда ответом приходит список искомых событий. По нажатию на событие открывается активность с полной информацией по событию и небольшой картой, по нажатию на которую можно посмотреть детальнее место проведения.

3.4 Результаты разработки

Результатами разработки являются: приобретенные навыки разработки клиент-серверных приложений, командной работы (вклад каждого участника см. в приложении И), собственно система «HR-automation» (см. приложение К).

ГЛАВА 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Задание для раздела

Студентам:

Группа	ФИО
8К91	Байделюк Елизавета Андреевна
8К91	Когут Арина Евгеньевна
8К91	Сидоркин Александр Андреевич

Школа	ИШИТР	Отделение (НОЦ)	ОИТ
Уровень образования	Бакалавриат	**Направление/ специальность**	09.03.04 Программная инженерия

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	*Стоимость материальных ресурсов и специального оборудования определены в соответствии с рыночными ценами г. Томска* *Тарифные ставки исполнителей определены штатным расписанием НИ ТПУ*
2. *Нормы и нормативы расходования ресурсов*	*Норма амортизационных отчислений на специальное оборудование*
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	*Отчисления во внебюджетные фонды 30 %*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Анализ конкурентных технических решений (НИ)*	*Расчет конкурентоспособности SWOT-анализ*
2. *Формирование плана и графика разработки и внедрения (НИ)*	*Структура работ. Определение трудоемкости. Разработка графика проведения исследования*

3. *Составление бюджета инженерного проекта (НИ)*	*Расчет бюджетной стоимости НИ*
4. *Оценка ресурсной, финансовой, бюджетной эффективности (НИ)*	*Интегральный финансовый показатель.* *Интегральный показатель ресурсоэффективности.* *Интегральный показатель эффективности.*
Перечень графического материала	
1. Оценка конкурентоспособности ИР 2. Матрица SWOT 3. Диаграмма Ганта 4. Бюджет НИ 5. Основные показатели эффективности НИ	

Дата выдачи задания для раздела по линейному графику	20.05.23

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Профессор	Гасанов Магеррам Али оглы	д.э.н.		20.05.23

Задание приняли к исполнению студенты:

Группа	ФИО	Подпись	Дата
8К91	Байделюк Елизавета Андреевна		20.05.23
8К91	Когут Арина Евгеньевна		20.05.23
8К91	Сидоркин Александр Андреевич		20.05.23

4.1.1 Краткое описание

Проектная группа по реализации продукта состоит из трех лиц: научный руководитель и 3 разработчика (бэкенд-разработчик, фронтенд-разработчик, Android-разработчик).

Описываемая выпускная квалификационная работа заключается в проектировании и разработке системы «HR-automation», которая помогает HR-специалистам в решении повторяющихся задач. Состоит из мобильного приложения, веб-приложения и сервера. Целью раздела «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение» является выбор наиболее конкурентоспособных методологий разработки, оценка эффективности, определение рисков и стратегий их устранения, формирование состава работ и бюджета проекта.

Для достижения поставленной цели были сформулированы следующие задачи:

1. Сформировать альтернативные варианты реализации проекта;
2. Оценить коммерческий потенциал и перспективность разработки проекта;
3. Произвести оценку научно-технического уровня исследования и оценку рисков;
4. Составить план работ по реализации проекта;
5. Рассчитать бюджет проекта.

4.2 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

4.2.1 Предпроектный анализ

4.2.1.1 Потенциальные потребители результатов исследования

Тинькофф центр разработки (ТЦР) – это часть инфраструктуры Тинькофф, занимающаяся разработкой IT-проектов. Система разрабатывается для сотрудников офисов ТЦР, они и являются главными и единственными потребителями результатов разработки.

4.2.1.2 Анализ конкурентных решений

Анализ конкурентных технических решений важен в выборе наиболее эффективного подхода к реализации проекта. Необходимо учитывать все сильные и слабые стороны конкурентов, чтобы иметь возможность развивать конкурентные преимущества собственного продукта.

Прямых конкурентов, напрямую удовлетворяющих запросам, нет, но необходимый функционал рассредоточен в нескольких уже существующих продуктах.

Для проведения анализа составлена оценочная карта. Выбрано 8 критериев и выбран их удельный вес, чтобы сформировать интегральный показатель конкурентоспособности продукта. Для сравнения используются три наиболее сильных конкурентных разработок. Оценочная карта для сравнения конкурентных разработок представлена в таблице 8.

Для покрытия потребностей разделов системы можно использовать канал в мессенджере (или бота), группу в социальных сетях или электронную таблицу в облачном хранилище.

Таблица 8 – Оценочная карта сравнений конкурентных и аналоговых технических решений

№	Критерии оценки	Вес критерия	Баллы			Конкурентоспособность		
			Бк1	Бк2	Бк3	К1	К2	К3
	1	2	3	4	5	6	7	8
Технические критерии оценки ресурсоэффективности								
1	Отказоустойчивость	0,15	3	4	4	0,45	0,6	0,6
2	Скорость	0,2	3	5	4	0,6	1	0,8
3	Простота реализации	0,1	4	4	4	0,4	0,4	0,4
4	Потребность в вычислительных ресурсах	0,1	5	4	4	0,5	0,4	0,4
5	Простота поддержки	0,1	4	4	5	0,4	0,4	0,5
6	Масштабируемость	0,15	3	5	5	0,45	0,75	0,75
Экономические критерии оценки эффективности								
7	Цена	0,15	5	4	4	0,75	0,6	0,6
8	Стоимость обслуживания	0,05	4	4	4	0,2	0,2	0,2
	Итого	**1**	**29**	**34**	**34**	**3,75**	**4,35**	**4,25**

Анализ конкурентных технических решений определяется по формуле:

$К = \sum В! \times Б!$ (1)

где К – конкурентоспособность вида;

В! – вес критерия (в долях единицы);

Б! – балл i-го показателя.

По данным оценочной карты получается, что наиболее конкурентоспособным решением является *мессенджер*. Однако данное решение имеет слишком низкую расширяемость, а значит, что сотрудники будут тратить слишком много времени на добавление информации в общую базу.

4.2.1.3 Технология QuaD

Технология QuaD (QUality ADvisor) представляет собой гибкий инструмент измерения характеристик, описывающих качество новой разработки и ее перспективность на рынке и позволяющие принимать решение целесообразности вложения денежных средств в научно-исследовательский проект.

В данном разделе необходимо провести анализ по технологии QuaD, чтобы дать предварительную оценку эффективности проекта и целесообразности его разработки.

Первым шагом необходимо определить показатели оценки коммерческого потенциала и качества разработки:

1. Показатели оценки коммерческого потенциала разработки:

– конкурентоспособность продукта;
– перспективность рынка;
– стоимость разработки;
– послепродажное обслуживание;
– финансовая эффективность;
– срок выхода на рынок.

2. Показатели оценки качества разработки:

– время отклика системы;

– отказоустойчивость;
– безопасность данных;
– потребность в вычислительных ресурсах;
– потребность в ресурсах памяти;
– качество пользовательского интерфейса;
– масштабируемость;
– простота реализации;
– качество технической документации.

Следующий шаг QuaD заключается в проведении расчётов показателей. Оценочная карта проекта по технологии QuaD приведена в таблице 9.

Таблица 9 – Оценочная карта проекта по технологии QuaD

Критерии оценки	**Вес критерия**	**Баллы**	**Максимальный балл**	**Средневзвешенное значение**
Показатели оценки коммерческого потенциала разработки				
1. Конкурентоспособность продукта	0,1	90	100	9
2. Перспективность рынка	0,05	20	100	1
3. Стоимость разработки	0,1	90	100	9
4. Послепродажное обслуживание	0,1	80	100	8
5. Финансовая эффективность	0,05	70	100	3,5
6. Срок выхода на рынок	0,05	85	100	4,25

Показатели оценки качества разработки				
7. Время отклика системы	0,1	95	100	9,5
8. Отказоустойчивость	0,1	95	100	9,5
9. Безопасность данных	0,1	90	100	9
10. Потребность в вычислительных ресурсах	0,025	85	100	2,125
11. Потребность в ресурсах памяти	0,025	80	100	2
12. Качество пользовательского интерфейса	0,05	95	100	4,75
13. Масштабируемость	0,1	100	100	10
14. Простота реализации	0,025	90	100	2,25
15. Качество технической документации	0,025	100	100	2,5
Итого	**1**			**86,375**

Оценка качества и перспективности по технологии QuaD определяется по формуле:

$$П_{ср} = \sum П_i \ = \ \sum В_i \times Б_i, \qquad (2)$$

где $П_{ср}$ – средневзвешенное значение показателя качества и перспективности научной разработки;

$В_i$ – вес показателя (в долях единицы);

$Б_i$ – баллы i-го показателя.

$$П_{ср} = 86,375$$

По результатам оценки делается вывод, что разработка является перспективной, так как средневзвешенное значение показателя качества попадает в диапазон 100-80.

4.2.1.4 SWOT-анализ

SWOT-анализ применяют для исследования внешней и внутренней среды проекта. Матрица составляется на основе анализа рынка и конкурентных технических решений, и показывает сильные и слабые стороны проекта, возможности и угрозы для разработки.

Первый этап заключается в описании сильных и слабых сторон проекта, в выявлении возможностей и угроз для реализации проекта, которые проявились или могут появиться в его внешней среде. Матрица SWOT представлена в таблице 10.

Таблица 10 – SWOT-анализ

Сильные стороны	**Слабые стороны**
С1. Гибкая и масштабируемая архитектура системы;	СЛ1. Отсутствие опыта разработки
С2. Стабильность и скорость работы системы;	СЛ2. Отсутствие макетов дизайна
С3. Многофункциональность системы	СЛ3. Нечеткие выдвинутые требования
Возможности	**Угрозы внешней среды**
В1. Спрос на новый функционал систмы	У1. Небезопасность системы
В2. Использование актуальных и популярных инструментов разработки	У2. Нехватка специалистов для дальнейшей поддержки проекта

	У3. Нехватка финансирования

Второй этап состоит в выявлении соответствия сильных и слабых сторон научно-исследовательского проекта внешним условиям окружающей среды. Это соответствие или несоответствие должны помочь выявить степень необходимости проведения стратегических изменений.

Соотношения параметров представлены в таблице 11.

Таблица 11 – Интерактивная матрица проекта

Сильные стороны проекта					Слабые стороны проекта		
Возможности проекта		С1	С2	С3	Сл1	Сл2	Сл3
	В1	+	+	+	-	+	+
	В2	+	-	+	+	-	-
Угрозы проекта	У1	-	-	-	-	+	+
Угрозы проекта	У2	+	-	+	+	+	+
Угрозы проекта	У3	+	+	+	-	-	-

4.2.2 Определение возможных альтернатив проведения научного исследования

В данном разделе необходимо предложить альтернативные варианты разработки продукта. В рамках проекта будет сформулирован основной вариант и два альтернативных, которые будут использоваться в дальнейших расчётах в виде вариантов исполнения.

Для определения альтернатив используется морфологически подход.

Первый шаг состоит в формулировке проблемы. Проблема состоит в больших временных затратах на взаимодействие между сотрудниками ТЦР.

На втором шаге необходимо выделить все важные морфологические характеристики объекта исследования:

– сервис карт;
– база данных;
– инструмент разработки сервера;
– инструмент разработки веб-приложения;
– инструмент разработки мобильного приложения.

На третьем шаге расписываются различные варианты характеристик. Морфологическая матрица проекта представлена в таблице 12.

Таблица 12 – Морфологическая матрица проекта

	1	2	3	4
А. Сервис карт	Google Maps API	Яндекс карты	2GIS	
Б. Базы данных	MySQL	PostgreSQL	SQLite	MongoDB
В. Инструмент разработки сервера	Spring	Django	Node.js	Golang
Г. Инструмент разработки веб-приложения	React	Angular	Vue.js	Svelte
Д. инструмент разработки мобильного приложения	Java	Kotlin	C\C++	JavaScript

Далее выделяются различные комбинации, которые составляют новое техническое решение. Необходимо привести один основной и один альтернативный вариант исполнения:

1. А1Б2В1Г2Д2;
2. А2Б1В2Г2Д1;

4.2.3 Планирование работ по научно-техническому исследованию

4.2.3.1 Структура работ в рамках научного исследования

Планирование комплекса предполагаемых работ осуществляется в следующем порядке:

- определение структуры работ в рамках научного исследования;
- определение участников каждой работы;
- установление продолжительности работ;
- построение графика проведения научных исследований.

Для выполнения научных исследований формируется рабочая группа, в состав которой могут входить научные сотрудники и преподаватели, разработчики, инженеры, техники и лаборанты. По каждому виду запланированных работ устанавливается соответствующая должность исполнителей.

Принятая рабочая группа: научный руководитель, Android-разработчик, backend-разработчик, frontend-разработчик.

Перечень этапов и работ, распределение исполнителей по данным видам работ представлен в таблице 13.

Таблица 13 – Перечень этапов, работ и распределение исполнителей

Основные этапы	№ раб	Содержание работ	Должность исполнителя
Разработка технического задания	1	Составление и утверждение технического задания	Руководитель

Выбор направления исследований	2	Выбор направления исследований	Разработчики
	3	Подбор и изучение материалов по теме	Разработчики
	4	Календарное планирование работ	Руководитель Разработчики
Теоретические и экспериментальные исследования	5	Исследование методологии построения архитектуры системы	Руководитель Разработчики
	6	Концептуализация системы	Разработчики
Обобщение и оценка результатов	7	Оценка эффективности полученных результатов	Разработчики
	8	Определение целесообразности проведения ОКР	Разработчики
Проведение ОКР			
Разработка технической документации и проектирование	9	Проектирование системы и составление диаграмм	Разработчики
	10	Оценка эффективности разработки и применения системы	Разработчики
Изготовление и испытание макета (опытного образца)	11	Разработка системы	Разработчики
Оформление отчета по НИР	12	Составление пояснительной записки	Разработчики

4.2.3.2 Определение трудоемкости выполнения работ

Трудовые затраты в большинстве случаях образуют основную часть стоимости разработки, поэтому важным моментом является определение трудоемкости работ каждого из участников научного исследования.

Трудоемкость выполнения научного исследования оценивается экспертным путем в человеко-днях и носит вероятностный характер, который зависит от множества трудно учитываемых факторов. Для определения ожидаемого (среднего) значения трудоемкости $t_{ожi}$ используется следующая формула:

$$t_{ожi} = \frac{3t_{\min i} + 2t_{\max i}}{5}, \quad (3)$$

где $t_{ожi}$ – ожидаемая трудоемкость выполнения i-ой работы чел.-дн.;

$t_{\min i}$– минимально возможная трудоемкость выполнения заданной i-ой работы, чел.-дн.;

$t_{\max i}$– максимально возможная трудоемкость выполнения заданной i-ой работы, чел.-дн.;

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях T_p, учитывающая параллельность выполнения работ по нескольким исполнителями.

$$T_{pi} = \frac{t_{ожi}}{Ч_i}, \quad (4)$$

где T_{pi} – продолжительность одной работы, раб.дн.;

$t_{ожi}$ – ожидаемая трудоемкость выполнения одной работы, чел.-дн.;

$Ч_i$ – численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

4.2.3.3 Разработка графика проведения научного исследования

Наиболее удобным и наглядным представлением проведения научных работ является построение ленточного графика в форме диаграммы Ганта.

Диаграмма Ганта – горизонтальный ленточный график, на котором работы по теме представляются протяженными во времени отрезками, характеризующимися датами начала и окончания выполнения данных работ.

Для удобства построение графика, длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$\mathrm{T_{ki}} = \mathrm{T_{pi}} \cdot \mathrm{k_{кал}}\,, \qquad (5)$$

где T_{ki} – продолжительность выполнения i-й работы в календарных днях;

T_{pi} – продолжительность выполнения i-й работы в рабочих днях;

$k_{\text{кал}}$– коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - (T_{\text{вых}} + T_{\text{пр}})}\,, \qquad (6)$$

где $T_{\text{кал}}$ – количество календарных дней в году;

$T_{\text{вых}}$ – количество выходных дней в году;

$T_{\text{пр}}$ – количество праздничных дней в году.

Расчет коэффициента календарности:

$$\mathrm{k_{кал}} = \frac{\mathrm{T_{кал}}}{\mathrm{T_{кал}} - (\mathrm{T_{вых}} + \mathrm{T_{пр}})} = \frac{365}{365 - 118} = 1{,}48$$

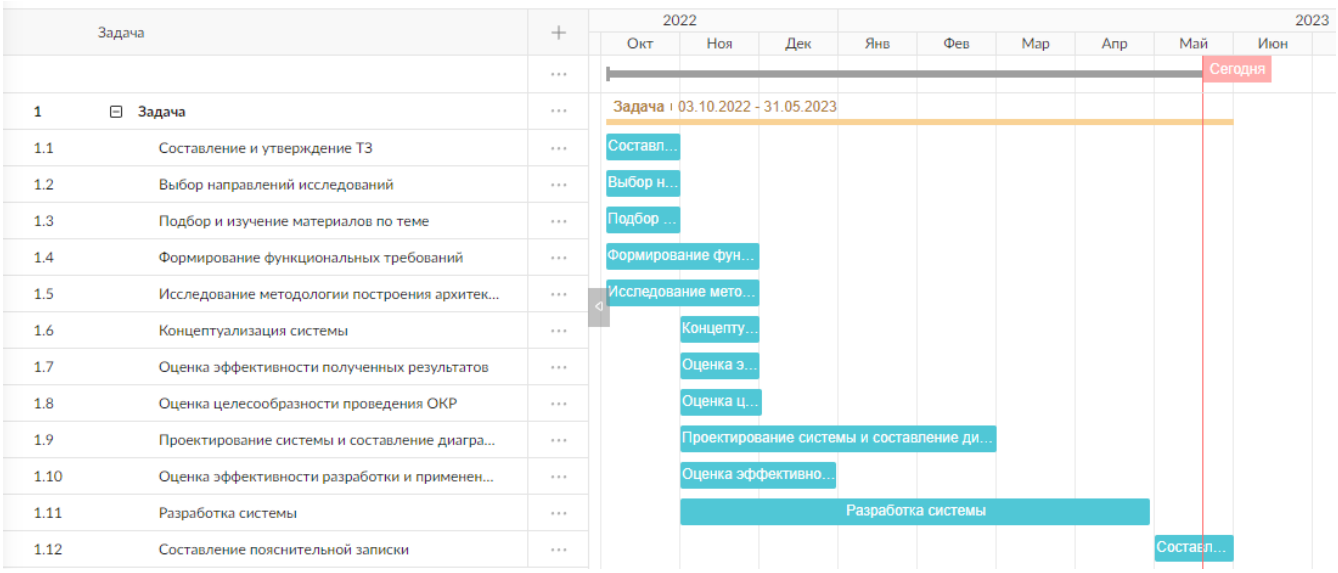

Рисунок 48 – Диаграмма Ганта

4.2.4 Бюджет научно-технического исследования (НТИ)

1. Материальные затраты;
2. Основная и дополнительная ЗП;
3. Социальные отчисления;
4. Прямые затраты;
5. Накладные расходы.

4.2.4.1 Расчет материальных затрат

В виду того, что все прочие принадлежности, необходимые для работы, имелись у исполнителей, при расчете материальных затрат была учтена только электроэнергия, необходимая для работы оборудования.

Расчет материальных затрат осуществляется по формуле:

$$З_М = (1 + k_T) \cdot \sum_{i=1}^{m} Ц_i \cdot N_{расхi}\,, \qquad (7)$$

где m – количество видов материальных ресурсов, потребляемых при выполнении научного исследования;

$N_{расхi}$ – количество материальных ресурсов i-го вида, планируемых к использованию при выполнении научного исследования (шт., кг, м, м2 и т.д.);

$Ц_i$ – цена приобретения единицы i-го вида потребляемых материальных ресурсов (руб./шт., руб./кг, руб./м, руб./м2 и т.д.);

$k_Т$ – коэффициент, учитывающий транспортно-заготовительные расходы.

Таблица 14 – Материальные затраты

Наименование	Единица измерения	Количество		Цена за ед., руб.	Затраты на материалы, ($З_М$), руб.	
		Исп.1	Исп. 2		Исп.1	Исп.2
Электроэнергия	кВт*ч	1050	1200	4,39	4609,5	5268
Итого, руб.					4609,5	5268

Общие материальные затраты составили **9877,5 руб**.

4.2.4.2 Основная заработная плата исполнителя темы

В настоящую статью включается основная заработная плата научных и инженерно-технических работников, рабочих макетных мастерских и опытных производств, непосредственно участвующих в выполнении работ по данной теме. Величина расходов по заработной плате определяется исходя из трудоемкости выполняемых работ и действующей системы окладов и тарифных ставок.

В состав основной заработной платы включается премия, выплачиваемая ежемесячно из фонда заработной платы в размере 20-30 % от тарифа или оклада. Расчет основной заработной платы приводится в таблице 15.

Таблица 15 – Расчет основной заработной платы

№ п/п	Наимен ование этапов	Испол нители по катег	Трудоемкость, чел.-дн.	Заработная плата, приходящаяся на	Всего заработная плата по тарифу

		ория м			один чел-дн.		(окладам), тыс. руб.	
			Исп.1	Исп.2	Исп.1	Исп.2	Исп.1	Исп.2
1.	Составление и утверждение ТЗ	НР	3	3	5		15	15
2.	Выбор направлений исследований	НР, Р	1	1	2		2	2
3.	Подбор и изучение материалов по теме	Р	2	2	4		8	8
4.	Календарное планирование работ	НР, Р	1	1	6		6	6
5.	Исследование методологии построения архитектуры системы	Р	5	3	4		20	12
6.	Концептуализация системы	Р	3	3	3		9	9
7.	Оценка эффективности полученных результатов	Р	1	1	3		3	3
8.	Оценка целесообразности проведения ОКР	Р	1	1	2		2	2
9.	Проектирование системы и составление диаграмм	Р	10	13	2		20	26
10.	Оценка эффективности разработки и применения системы	Р	1	1	2		2	2
11	Разработка системы	Р	38	40	5		190	200

12	Составление пояснительной записки	Р	15	15	3	45	45
Итого						322	330

Статья включает основную заработную плату работников, непосредственно занятых выполнением проекта, (включая премии, доплаты) и дополнительную заработную плату и рассчитывается по формуле:

$$З_{зп} = З_{осн} + З_{доп} \quad (8)$$

где $З_{осн}$ – основная заработная плата;

$З_{доп}$ – дополнительная заработная плата (12–20 % от $З_{осн}$).

Основная заработная плата руководителя рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot Т_р \quad (9)$$

где $З_{осн}$– основная заработная плата одного работника;

$Т_р$ – продолжительность работ, выполняемых научно-техническим работником, раб. дн.;

$З_{дн}$ – среднедневная заработная плата работника, руб.

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot М}{F_д} \quad (10)$$

где $З_м$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года:

при отпуске в 24 раб. дня М =11,2 месяца, 5–дневная неделя;

при отпуске в 48 раб. дней М=10,4 месяца, 6–дневная неделя;

$F_д$ – действительный годовой фонд рабочего времени научно–технического персонала, раб. дн.

Таблица 16 – Баланс рабочего времени

Показатели рабочего времени	Научный руководитель	Разработчик
Календарное число дней	365	365
Количество нерабочих дней - выходные дни - праздничные дни	118	118
Потери рабочего времени - отпуск - невыходы по болезни	480	720
Действительный годовой фонд рабочего времени	199	175

Месячный должностной оклад работника (руководителя):

$$З_м = З_{тс} \cdot (1 + k_{пр} + k_д) \cdot k_р \qquad (11)$$

где $З_{тс}$ – заработная плата по тарифной ставке, руб.;

$k_{пр}$ – премиальный коэффициент, равный 0,3;

$k_д$ – коэффициент доплат и надбавок составляет примерно 0,2 – 0,5;

$k_р$ – районный коэффициент, равный 1,3 (для Томска).

Для предприятий, не относящихся к бюджетной сфере, тарифная заработная плата (оклад) рассчитывается по тарифной сетке, принятой на данном предприятии.

Расчет основной заработной платы представлен в таблице 17.

Таблица 17 – Расчет основной заработной платы

Исполнители	Разряд	$З_{тс}$, руб.	$k_{пр}$	$k_д$	$k_р$	$З_м$, руб.	$З_{дн}$, руб.	$Т_р$, раб. дн.	$З_{осн}$, руб.

Научный руководитель	Старший преподаватель	30000	0,3	0,4	1,3	66300	3731,46	5	18657,3
Android-Разработчик	Разработчик	15000	0,3	0,2	1,3	39000	1872,00	78	146016
Backend-Разработчик	Разработчик	15000	0,3	0,2	1,3	39000	1872,00	78	146016
Frontend-Разработчик 2	Разработчик	15000	0,3	0,2	1,3	39000	1872,00	78	146016
Итого									**456705,3**

4.2.4.3 Расчет дополнительной заработной платы

Дополнительная заработная плата учитывает величину предусмотренных Трудовым кодексом РФ доплат за отклонение от нормальных условий труда, а также выплат, связанных с обеспечением гарантий и компенсаций (при исполнении государственных и общественных обязанностей, при совмещении работы с обучением, при предоставлении ежегодного оплачиваемого отпуска и т.д.).

Расчет дополнительной заработной платы рассчитывается по формуле:

$$З_{доп} = k_{доп} \cdot З_{осн}\,, \qquad (12)$$

где $k_{доп}$ – коэффициент дополнительной заработной платы, принятый на стадии проектирования за 0,15.

4.2.4.4 Отчисления во внебюджетные фонды

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников.

Расчет произведен в соответствии с Федеральным законом от 24.07.2009 №212-ФЗ.

Так как предстоящий проект является частью сферы информационных технологий, проводим дальнейший расчет с учетом письма ФНС России от 01.03.2022 N БС-4-11/2441:

– 6% на обязательное пенсионное страхование;
– 1,5% на обязательное социальное страхование;
– 0,1% на обязательное медицинское страхование.

Таким образом общий тариф составляет 7,6%.

Отчисления во внебюджетные фонды представлены в таблице 18.

Таблица 18 – Отчисления во внебюджетные фонды

Исполнитель	Основная заработная плата, руб.		Дополнительная заработная плата, руб.	
	Исп.1	Исп.2	Исп.1	Исп.2
Руководитель проекта	18657,3	18657,3	2798,6	2798,6
Разработчик 1	146016	151632	21902,4	22744,8
Разработчик 2	146016	151632	21902,4	22744,8
Разработчик 3	146016	151632	21902,4	22744,8
Итого				
Исполнение 1	39916,04			
Исполнение 2	41388,56			

4.2.4.5 Накладные расходы

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов. Их величина определяется по формуле:

$$З_{накл} = (\sum статей) \cdot k_{нр} \quad (13)$$

где $k_{нр}$ – коэффициент, учитывающий накладные расходы.

Величину коэффициента накладных расходов можно взять в размере 15%.

Накладные расходы для исполнения 1 составили:

$$З_{накл} = (4609{,}5 + 18657{,}3 + 146016 * 3 + 2798{,}6 + 21902{,}4 * 3 + 39916{,}04) \cdot 0{,}15 = 85460{,}5 \text{ руб.}$$

Накладные расходы для исполнения 2 составили:

$$З_{накл} = (5268 + 11194.4 + 151632 * 2 + 1679.16 + 22744{,}8 * 2 + 41388{,}56) \cdot 0{,}15 = 179270{,}76 \text{ руб.}$$

4.2.4.6 Формирование бюджета затрат научно-исследовательского проекта

Рассчитанная величина затрат научно–исследовательской работы является основой для формирования бюджета затрат проекта. Определение бюджета затрат на научно–исследовательский проект приведено в таблице 19.

Таблица 19 – Расчет бюджета затрат НТИ

Наименование статьи	Сумма, руб.		Примечание
	Исп.1	Исп.2	
1. Материальные затраты	4609,5	5268	Пункт 1.4.1
2. Затраты на специальное оборудование для научных (экспериментальных) работ	-	-	Отсутствуют
3. Затраты по основной заработной плате исполнителей темы	456705,3	473553,3	Пункт 1.4.2

4. Затраты по дополнительной заработной плате исполнителей темы	68505,8	71033	Пункт 1.4.3
5. Отчисления во внебюджетные фонды	39916,04	41388,56	Пункт 1.4.4
6. Затраты на научные и производственные командировки	-	-	Отсутствуют
7. Контрагентские расходы	-	-	Отсутствуют
8. Накладные расходы	85460,5	179270,76	Пункт 4.5.6
9. Бюджет затрат НТИ	655197,14	770513,62	

4.2.5 Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности происходит на основе расчета интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

Интегральный показатель финансовой эффективности научного исследования определяется как:

$$I_{\text{фин.р}}^{\text{исп.}i} = \frac{\Phi_{pi}}{\Phi_{max}} \quad (14)$$

где $I_{\text{фин.р}}^{\text{исп.}i}$ – интегральный финансовый показатель разработки;

$\Phi_{\text{р}i}$ – стоимость i-го варианта исполнения;

$Ф_{max}$ – максимальная стоимость исполнения научно-исследовательского проекта.

$$I_{\text{фин.р}}^{\text{исп1}} = \frac{655197,14}{770513,62} = 0,85;$$

$$I_{\text{фин.р}}^{\text{исп2}} = \frac{770513,62}{770513,62} = 1;$$

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить следующим образом:

$$I_{pi} = \sum_{i=1}^{n} a_i \times b_i \qquad (15)$$

где $I_{\text{р}i}$ – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки;

a_i – весовой коэффициент i-го варианта исполнения разработки;

b_i^a, b_i^p – бальная оценка i-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания;

n – число параметров сравнения.

Таблица 20 – Сравнительная оценка характеристик вариантов исполнения

Объект исследования / Критерии	Весовой коэффициент параметра	Исп. 1	Исп. 2
1. Удобство конечным пользователям	0,3	5	4
2. Удобство администраторам	0,3	4	4
3. Масштабируемость	0,1	5	5
4. Гибкость	0,1	5	4
5. Отказоустойчивость	0,2	4	4
Итого	1	4,5	4,1

$$I_{\text{р}-\text{исп1}} = 0,3 \cdot 5 + 0,3 \cdot 4 + 0,1 \cdot 5 + 0,1 \cdot 5 + 0,2 \cdot 4 = 4,5;$$

$$I_{\text{р}-\text{исп2}} = 0,3 \cdot 4 + 0,3 \cdot 4 + 0,1 \cdot 5 + 0,1 \cdot 4 + 0,2 \cdot 4 = 4,1;$$

Интегральный показатель эффективности вариантов исполнения разработки ($I_{\text{испi}}$) определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формуле:

$$I_{\text{исп1}} = \frac{I_{\text{р-исп1}}}{I_{\text{фин.р}}^{\text{исп1}}} = \frac{4{,}5}{0{,}85} = 5{,}3;$$

$$I_{\text{исп2}} = \frac{I_{\text{р-исп2}}}{I_{\text{фин.р}}^{\text{исп2}}} = \frac{4{,}1}{1} = 4{,}1;$$

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволит определить сравнительную эффективность проекта и выбрать наиболее целесообразный вариант из предложенных.

Сравнительная эффективность проекта ($Э_{\text{ср}}$):

$$Э_{\text{ср}} = \frac{I_{\text{исп2}}}{I_{\text{исп1}}} \quad (16)$$

Таблица 21 – Сравнительная эффективность разработки

№	Показатели	Исп.1	Исп.2
1	Интегральный финансовый показатель разработки	0,85	1
2	Интегральный показатель ресурсоэффективности разработки	4,5	4,1
3	Интегральный показатель эффективности	5,3	4,1
4	Сравнительная эффективность вариантов исполнения	1	0,77

Сравнив значения интегральных показателей эффективности, можно сделать вывод, что реализация технологии в **первом** исполнении является более эффективным вариантом решения задачи, поставленной в данной работе с позиции финансовой и ресурсной эффективности.

4.2.6 Вывод

В ходе выполнения раздела финансового менеджмента проведен анализ финансово-экономических аспектов разработки информационной системы.

Составлен перечень проводимых работ, назначены исполнители для них и продолжительность выполнения этапов работ.

Основываясь на результатах проведенного анализа, разработка системы HR-automation является конкурентоспособной и перспективной в финансовом плане. Длительность разработки составила 240 календарных дней, а рассчитанная стоимость – около 650 тыс. рублей.

ГЛАВА 5. Социальная ответственность

Задание для раздела

Студентам:

Группа	ФИО
8К91	Байделюк Елизавете Андреевне
8К91	Когут Арине Евгеньевне
8К91	Сидоркину Александру Андреевичу

Школа	**ИШИТР**	**Отделение (НОЦ)**	**ОИТ**
Уровень образования	Бакалавриат	**Направление/ специальность**	09.03.04 Программная инженерия

Тема ВКР:

Разработка системы «HR-automation»	
Исходные данные к разделу «Социальная ответственность»:	
Введение – Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика) и области его применения. – Описание рабочей зоны (рабочего места) при разработке проектного решения/при эксплуатации	*Объект исследования:* приложение, направленное на уменьшение затрат времени на передачу информации между сотрудниками ТЦР *Область применения:* информационные технологии *Рабочая зона:* офис *Размеры помещения:* 35 м². *Количество и наименование оборудования рабочей зоны:* компьютерный стол – 3 шт., ноутбук – 3 шт., настольная лампа – 3 шт., компьютерная мышь – 3 шт.

	Рабочие процессы, связанные с объектом исследования, осуществляющиеся в рабочей зоне: проектирование, разработка и тестирование веб-приложения.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности при разработке проектного решения: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ. ГОСТ 12.2.032-78. «ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования»
2. Производственная безопасность при разработке проектного решения: – Анализ выявленных вредных и опасных производственных факторов	**Вредные факторы:** 1. Отсутствие или недостаток необходимого искусственного освещения; 2. Повышенный уровень шума; 3. Нагрузка на зрительный аппарат; **Опасные факторы:** Факторы, связанные с электрическим током **Требуемые средства коллективной защиты от выявленных факторов:** системы естественного

	освещения, приборы искусственного освещения, изоляционные средства, предохранительные устройства.
3. Экологическая безопасность при разработке проектного решения	Воздействие на селитебную зону не выявлено Воздействие на атмосферу не выявлено Воздействие на гидросферу из-за неверного способа утилизации рабочей техники Воздействие на литосферу из-за неверного способа утилизации рабочей техники
4. Безопасность в чрезвычайных ситуациях при разработке проектного решения	**Возможные ЧС:** − Природные (землетрясения, наводнения) − Техногенные (пожары) − Биолого-социального (пандемия) **Наиболее типичная ЧС:** − Пожар
Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Старший преподаватель	Мезенцева Ирина Леонидовна			17.05.2023

Задание приняли к исполнению студенты:

Группа	ФИО	Подпись	Дата
8К91	Байделюк Елизавета Андреевна		17.05.2023

8К91	Когут Арина Евгеньевна		17.05.2023
8К91	Сидоркин Александр Андреевич		17.05.2023

5.1 Социальная ответственность

5.1.1 Введение

В рамках выпускной квалификационной работы была разработана информационная система «HR-automation», направленная на сокращение затраченного HR-специалистом времени на работу с новыми сотрудниками и на выполнение повторяющихся задач. Пользователями системы являются сотрудники ТЦР. Система состоит из трех частей – сервер, веб-приложение и мобильное приложение для ОС Android. Приложение разделено на 5 функциональных разделов: «Сотрудники» – содержит информацию о сотрудниках ТЦР, «FAQ» – содержит ответы на часто задаваемые вопросы, «Продукты» – содержит интерфейс для заказов продуктов в офис, «Рестораны» – содержит перечень близлежащих заведений с отзывами на них, «События» – содержит информацию о проводящихся в ТЦР событиях.

Разработка велась в помещении (35 м^2), оборудованном рабочими местами для комфортной работы за компьютером: компьютерными столами (3 шт.), ноутбуками (3 шт.), настольными лампами (3 шт.), компьютерными мышками (3 шт.). В помещении присутствуют окна, предоставляющие естественное освещение и вентиляцию. Также присутствует искусственное освещение. Целью данной главы является определение и анализ вредных и опасных факторов труда, анализ параметров рабочей среды, а также исследование социальной ответственности при работе в офисе и описание правомерных способов предупреждения неблагоприятных факторов работы и уменьшения их последствий.

5.1.2 Правовые и организационные вопросы обеспечения безопасности при разработке проектного решения

5.1.2.1 Правовые нормы трудового законодательства

Законодательный акт «Трудовой кодекс Российской Федерации» от

30.12.2001 N 197-ФЗ (ред. от 19.12.2022) служит основным инструментом регулирования трудовых отношений между работником и работодателем. Ниже представлены некоторые важные положения из ТК РФ:

1. Работодатель обязан обеспечивать безопасность и условия труда, соответствующие государственным нормативным требованиям охраны труда;

2. Список обязанностей, режим работы и размер заработной платы должны быть должны быть зафиксированы в трудовом договоре;

3. Нормальная продолжительность рабочего времени не может превышать 40 часов в неделю;

4. В течение рабочего дня работнику должен быть предоставлен перерыв для отдыха и питания продолжительностью не более двух часов и не менее 30 минут;

5. Всем работникам предоставляются выходные дни (еженедельный непрерывный отдых).

5.1.2.2 Эргономические требования к правильному расположению и компоновке рабочей зоны

Рабочее место должно быть организовано с учетом требований ГОСТ 12.2.032-78 «ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования». Согласно ГОСТ 12.2.032-78, взаимное расположение элементов рабочего места должно обеспечивать возможность осуществления всех необходимых движений для эксплуатации и технического обслуживания оборудования.

Если работник постоянно загружен работой с ПЭВМ, приемлемой является поза сидя. В положении сидя основная нагрузка падает на мышцы, поддерживающие позвоночный столб и голову. В связи с этим при длительном сидении время от времени необходимо менять фиксированные рабочие позы. При организации работы с ЭВМ, согласно указанным выше требованиям,

должны быть соблюдены следующие условия:

1. Рабочие места с ПЭВМ должны располагаться на расстоянии не менее 1,5 м от стены с оконными проемами, от других стен на расстоянии 1 м, между собой на расстоянии не менее 1,5 м;

2. Конструкция рабочей мебели должна обеспечивать возможность индивидуальной регулировки соответственно росту пользователя и создавать удобную позу для работы;

3. При размещении рабочих мест необходимо исключить возможность прямой засветки экрана источником естественного освещения;

4. Окна в помещениях с ПК должны быть оборудованы регулирующими устройствами – жалюзи, занавески, внешние козырьки;

5. При размещении ЭВМ на рабочем месте должно обеспечиваться пространство для пользователя величиной не менее 850 мм;

6. Высота рабочего стола с клавиатурой должна составлять 680 - 800 мм над уровнем стола.

При выполнении выпускной квалификационной работы правовых и организационных нарушений по указанным требованиям не было выявлено, рабочее место было оборудовано согласно всем нормам и правилам.

5.1.3 Производственная безопасность

Согласно ГОСТ 12.0.003-2015 «ССБТ. Опасные и вредные производственные факторы. Классификация» на оператора ПЭВМ в течение рабочего дня воздействует множество различныхпроизводственных факторов, каждый из которых влияет на производительность, работоспособность и физическое состояние.

Все выявленные факторы приведены в таблице 22.

Таблица 22 – Возможные опасные и вредные производственные факторы на рабочем месте инженера-программиста

Факторы (ГОСТ 12.0.003-2015)	Нормативные документы
Отсутствие или недостаток необходимого искусственного освещения	СП 52.13330.2016 «Естественное и искусственное освещение». Актуализированная редакция СНиП 23-05-95
Повышенный уровень шума	СП 51.13330.2011 Защита от шума. Актуализированная редакция СНиП 23-03-2003
Нагрузка на зрительный аппарат	СП 52.13330.2016 «Естественное и искусственное освещение». Актуализированная редакция СНиП 23-05-95
Факторы, связанные с электрическим током	ГОСТ 12.1.019-2017 «Электробезопасность. Общие требования и номенклатура видов защиты»

5.1.3.1 Отсутствие или недостаток необходимого искусственного освещения

Недостаток освещения или его отсутствие влияют на работоспособность зрения, а также оказывают воздействие на психику и эмоциональное состояние работника, вызывая усталость нервной системы. Для освещения помещений рекомендуется использовать систему общего равномерного освещения. В случае, когда работа в основном выполняется на персональном компьютере, рекомендуется применять системы комбинированного освещения.

Согласно СП 52.13330.2016 зрительную работу разработчика программного обеспечения можно характеризовать как работу разряда Б – высокой точности (наименьший эквивалентный размер объекта различения составляет 0,3-0,5 мм), подразряда 1 (относительная продолжительность зрительнойработы при направлении зрения на рабочую поверхность не менее 70%).

В таблице 23 представлены требования к освещению рабочего помещения для указанного разряда.

Таблица 23 – Требования к освещению рабочего помещения для разряда Б1

Искусственное освещение	
Освещенность на рабочей поверхности от системы общего освещения, лк	300
Цилиндрическая освещенность, лк	100
Объединенный показатель дискомфорта, не более	21
Коэффициент пульсации освещенности, Кп, %, не более	15

Для уменьшения влияния недостаточного освещения на рабочем месте рекомендуется размещать осветительные приборы таким образом, чтобы свет падал непосредственно на рабочую поверхность, минимизируя появление теней. Необходимо избегать сильного света в периферийной зоне зрения, так как это может привести к напряжению глаз и быстрой утомляемости. Чтобы решить проблему недостаточного освещения помещения, можно рассмотреть возможность расширения оконных проемов или установку качественных источников искусственного освещения.

5.1.3.2 Повышенный уровень шума

Источником повышенного уровня шума в офисе может быть различная техника, издающая монотонный гул, открытые окна, разговоры других разработчиков. Шум на рабочем месте может негативно влиять на концентрацию и уровень усталости работника, что может отрицательно сказаться на результативности работы.

Уровень шума на рабочих местах разработчика-программиста не должен превышать значений, которые указаны в СП 51.13330.2011. Согласно СП 51.13330.2011 (пункт 6.3), уровень шума в офисе не должен превышать значение в 65 дБА.

Чтобы снизить уровень шума на рабочем месте, можно установить материалы для звукоизоляции на стены и потолок помещения, чтобы уменьшить проникновение шума извне.

5.1.3.3 Нагрузка на зрительный аппарат

Работа на ПК сопровождается постоянным и значительным напряжением функций зрительного анализатора.

Спектр излучения компьютера включает в себя рентгеновскую и ультрафиолетовую области спектра, а также широкий диапазон электромагнитных волн других частот. Опасность рентгеновских лучей считается сейчас специалистами пренебрежимо малой, поскольку этот вид лучей поглощается веществом экрана.

Допустимые уровни ультрафиолетового излучения для мониторов регулируются в соответствии с СанПиН 1.2.3685-21 «Гигиенические нормативы и требования к обеспечению безопасности и (или) безвредности для человека факторов среды обитания» и указаны в таблице 24.

Таблица 24 – Допустимые уровни ультрафиолетового излучения

Вид изделий	**Спектральный диапазон длин волн, нм**	**Допустимая интенсивность облучения, Вт/м2**
Экраны телевизоров, видеомониторов, осциллографов измерительных и других приборов, средств отображения информации с визуальным контролем	Свыше 315 до 400	Не более 0,1
	Свыше 280 до 315	Не более 0,0001
	От 200 до 280	Не допускается

Чтобы снизить зрительное напряжение, необходимо выбрать такой монитор, параметры которого удовлетворяли бы параметрам таблицы 3. Соблюдение этих норм позволит снизить зрительное напряжение.

5.1.3.4 Факторы, связанные с электрическим током

В связи с близостью программиста к электрическим сетям и устройствам, необходимо соблюдать дополнительные меры безопасности при работе с электропроводкой и компьютером, осознавая риск поражения электрическим током. Поражение электрическим током может быть опасно термическим повреждением кожи и тканей, воздействием на сердце и нервную систему.

Основными причинами поражения электрическим током являются:

- прикосновение к токовыводящим частям под напряжением;
- неисправность изоляции и защитных устройств.

Мероприятия, направленные на предотвращение возможности поражения электрическим током, включают в себя следующее:

- при выполнении монтажных работ необходимо использовать только исправно работающий инструмент, аттестованный службой кипиаф;
- запрет на выполнение работ на задней панели при включенном сетевом напряжении;
- постоянное наблюдение за исправностью электропроводки, при обнаружении неисправности незамедлительное ее устранение;
- выполнение работ по устранению неисправности проводится только компетентными в данной области людьми.

Работа программистов требует 1 группы по электробезопасности, так как является работой неэлектротехнического персонала.

5.1.4 Экологическая безопасность

Загрязнение почвы веществами от компьютерной техники, батареек и других элементов питания, может серьезно повлиять на литосферу. Токсичные вещества, такие как свинец, кадмий, ртуть и другие, могут проникать в почву и загрязнять грунтовые воды. Это может иметь серьезные последствия для

окружающей среды и здоровья людей, включая отравления, рак и другие заболевания. Загрязнение почвы также может снизить плодородие почвы и ухудшить качество растительности.

Неправильная утилизация рабочей техники может привести к серьезному загрязнению гидросферы. Если техника утилизируется неправильно, отходы могут попадать в реки, озера и другие водные источники, загрязняя их. Токсичные вещества, включая ртуть, свинец и другие тяжелые металлы, могут попадать в воду, отравляя ее и нанося вред живым организмам, включая рыб и других морских животных. Это может привести к уменьшению популяций рыб и других водных организмов, а также снижению качества воды для человеческого использования. Неправильная утилизация техники также может привести к образованию свалок на берегах рек и озер, что способствует загрязнению водных ресурсов. Поэтому важно правильно утилизировать рабочую технику с использованием безопасных методов, соответствующих законодательству и нормативам, а также соблюдать меры предосторожности при работе с техникой. Это включает переработку и утилизацию техники с использованием безопасных методов. Необходимо обратить внимание, что описанные влияния не оказываются на атмосферу и селитебную зону.

5.1.5 Безопасность в чрезвычайных ситуациях

Разработка информационной системы проходила в офисе. Ниже перечислены возможные ЧС:

1. Техногенные (взрывы, пожары, обрушение помещений);
2. Природные (наводнения, ураганы, бури, природные пожары);
3. Биологические (эпидемии, пандемии).

Наиболее типичной ЧС для помещения, в котором проводилась работа, является пожар. Он может возникнуть вследствие причин электрического и неэлектрического характеров. К причинам электрического характера можно

отнести короткое замыкание, искрение, статическое электричество. К причинам неэлектрического характера относится неосторожное обращение с огнём, курение, оставление без присмотра нагревательных приборов.

Наиболее частыми причинами возникновения пожара можно назвать короткое замыкание, перегрузку сетей, с последующим нагревом токоведущих частей и неисправность оборудования.

Рабочее помещение, где работает программист, относится к категории В согласно стандарту пожарной безопасности. Это связано с наличием твёрдых горючих и трудногорючих материалов. По классификации пожаров, установленной в Федеральном законе от 22 июля 2008 года № 123-ФЗ, возможный пожар в данном помещении относится к классу А в связи с наличием горючих твердых веществ, а также к классу Е из-за присутствия электроустановок под напряжением.

Для тушения пожара такого класса, причиной которого стало возгорание ПЭВМ, достаточно воспользоваться переносным и передвижным огнетушителем, находящимся в здании офиса.

Для предотвращения возникновения пожара необходимо:

- регулярно проводить инструктажи сотрудников предприятия по пожарной безопасности;
- разместить в помещении план эвакуации и плакаты с краткой информацией с действиями при возникновении пожара;
- соблюдать правила и нормы при монтаже электронных приборов и проведении электрической проводки;
- оборудовать помещение пожарной сигнализацией и красными кнопками, а также средствами тушения пожара.

если все же не удалось предотвратить пожар, то каждый сотрудник должен:

- незамедлительно сообщить об этом в пожарную охрану;
- принять меры по эвакуации людей, каких-либо материальных ценностей согласно плану эвакуации;
- отключить электроэнергию, приступить к тушению пожара первичными средствами пожаротушения.

5.1.6 Вывод по разделу

В ходе проведенной работы были исследованы правовые и организационные аспекты обеспечения безопасности в рабочем окружении при разработке информационной системы. В данной главе были определены потенциально опасные и вредные факторы, связанные с работой программиста, которые соответствуют установленным нормам.

Рабочее место тщательно оборудовано с учетом требований по освещению, обеспечивая оптимальные условия для работы с минимальной нагрузкой на зрительный аппарат. Светильники правильно расположены, обеспечивая равномерное освещение без засветок и теней. Мониторы имеют регулируемую яркость и контрастность, а также адаптивное освещение, что помогает снизить усталость глаз и предотвратить негативные последствия для зрения. Также были предусмотрены периодические перерывы и упражнения для глаз, чтобы снизить напряжение и поддерживать здоровье глазных мышц.

Согласно правилам устройства электроустановок, рабочие места программистов относятся к зонам без повышенной опасности от поражения электрическим током. Рабочее помещение оборудовано с учетом требований по электро- и пожарной безопасности. Работа программистов относится к категории тяжести труда Iа и требует 1 группы по электробезопасности.

В отношении пожарной безопасности, рабочее помещение программиста классифицируется как пожароопасное категории В, а возможный пожар может быть классифицирован как класс А или Е.

В отношении негативного воздействия на окружающую среду, рабочее помещение инженера-программиста классифицируется как объект IV категории.

Таким образом, при выполнении проекта не было обнаружено нарушений в организации рабочего процесса, и все необходимые требования и нормы безопасности были соблюдены.

ЗАКЛЮЧЕНИЕ

В ходе выполнения работы был осуществлен ряд мероприятий по разработке решения проблемы заказчика о сокращении трудозатрат на второстепенные задачи менеджеров и HR-специалистов (вклад каждого участника см. в приложении И). Была получена проблема заказчика и собраны требования для поиска аналогов и составления функциональных требований. Анализ аналогичных решений показал, что лишь проект «HR-automation» предоставляет необходимый функционал и удовлетворяет требованиям заказчика, поэтому его разработку можно считать оправданной.

Проектирование системы включало в себя задачи по определению общей архитектуры, на ее основании был составлен стек технологий для каждой платформы. На основании стека технологий были спроектированы части системы по отдельности. Решены вопросы по взаимодействию между подсистемами при реализации конкретного функционала. Установлен формат взаимодействия между членами команды и подход к разработке и взаимодействию с заказчиком.

В ходе разработки системы решались проблемы по реализации внутренних работ подсистем. Были приняты решения по внутренней организации кода, структур и классов. Получены знания о подходах в коммерческой разработке программного обеспечения. Результатом разработки является готовая система, удовлетворяющая требованиям заказчика и выложенная на хостинг-сервисах для демонстрации, (см. приложение К).

СПИСОК ЛИТЕРАТУРЫ

1. Документация Angular [Электронный ресурс] URL: https://angdev.ru/ngrx/about/ (дата обращения: 05.10.2022)
2. Расширяемые приложения с NgRx [Электронный ресурс] URL: https://medium.com/trade-me (дата обращения: 20.10.2022)
3. Введение в NgRx [Электронный ресурс] URL: https://habr.com/ru/articles/489674/ (дата обращения: 20.10.2022)
4. Проектирование ИС [Электронный ресурс] URL: https://habr.com/ru/articles/416525/ (дата обращения: 05.10.2022)
5. Пояснение о чистой архитектуре [Электронный ресурс] URL: https://habr.com/ru (дата обращения: 05.10.2022)
6. Документация Android [Электронный ресурс] URL: https://developer.android.com/docs (дата обращения: 06.10.2022)
7. Презентационные паттерны [Электронный ресурс] URL: https://habr.com/ru (дата обращения: 07.10.2022)
8. Документация Dagger и примеры [Электронный ресурс] URL: https://dagger.dev/ (дата обращения: 23.10.2022)
9. Документация Retrofit [Электронный ресурс] URL: https://square.github.io/retrofit/ (дата обращения: 22.10.2022)
10. Сравнение Volley и Retrofit [Электронный ресурс] URL: https://www.javainhand.com (дата обращения: 05.10.2022)
11. Документация Google Maps [Электронный ресурс] URL: https://developers.google.com (дата обращения: 10.02.2023)
12. Документация Spring [Электронный ресурс] URL: https://docs.spring.io (дата обращения: 05.10.2022)
13. Документация Java [Электронный ресурс] URL: https://docs.oracle.com/javase/8 (дата обращения: 05.10.2022)

14. Краткое обучение Spring [Электронный ресурс] URL: https://www.baeldung.com/spring-tutorial (дата обращения: 05.10.2022)

15. Документация Docker [Электронный ресурс] URL: https://hub.docker.com/ (дата обращения: 14.11.2022)

16. Документация Liquibase [Электронный ресурс] URL: https://docs.liquibase.com/home.html (дата обращения: 14.11.2022)

17. Создание документации Apidoc [Электронный ресурс] URL: https://apidocjs.com/ (дата обращения: 05.10.2022)

ПРИЛОЖЕНИЕ А

(справочное)

Функциональные требования для мобильного приложения

Личный кабинет:

ФК1. система должна предоставлять возможность авторизации пользователя;

ФК2. система должна предоставлять возможность выхода из аккаунта авторизованного пользователя;

ФК3. система должна предоставлять возможность просмотра информации о своем профиле;

ФК4. система должна предоставлять возможность редактирования полей «проект» и «о себе»;

ФК5. система должна предоставлять возможность загрузки нового аватара профиля.

Сотрудники:

ФК6. система должна предоставлять возможность просмотра информации о профилях коллег, зарегистрированных в этой системе;

Часто-задаваемые вопросы:

ФК7. Система должна предоставлять возможность просмотра категорий ответов на часто задаваемые вопросы;

ФК8. Система должна предоставлять возможность просмотра ответов на часто задаваемые вопросы категории.

Продукты:

ФК9. система должна предоставлять возможность просмотра доступных для заказа в офис товаров;

ФК10. система должна предоставлять возможность фильтрации товаров по категориям;

ФК11. Система должна предоставлять возможность отправлять запрос на заказ недостающих товаров.

Рестораны:

ФК12. система должна предоставлять возможность просмотра заведений города по местоположению тцр в виде списка;

ФК13. система должна предоставлять возможность просмотра заведений города по местоположению тцр заведений на карте;

ФК14. система должна предоставлять возможность просмотра отзывов на заведения;

ФК15. система должна предоставлять возможность оставлять отзыв на заведения.

События:

ФК16. система должна предоставлять возможность просмотра мероприятий, организуемых тцр, в виде списка;

ФК17. система должна предоставлять возможность просмотра подробной информации о мероприятии;

ФК18. система должна предоставлять возможность просмотра места проведения мероприятия на карте;

ФК19. система должна предоставлять возможность фильтрации мероприятий по дате проведения;

ФК20. система должна предоставлять возможность фильтрации мероприятий по месту проведения;

ФК21. система должна предоставлять возможность фильтрации мероприятий по названию;

ФК22. система должна предоставлять возможность фильтрации мероприятий по формату проведения;

ФК23. система должна уведомлять пользователя о добавлении нового мероприятия посредством отправки push-уведомления.

ПРИЛОЖЕНИЕ Б

(справочное)

Функциональные требования для веб-приложения

Личный кабинет:

ФК1. Система должна предоставлять возможность авторизации для сотрудников-администраторов

ФК2. Система должна предоставлять возможность выхода из аккаунта

ФК3. Система должна предоставлять возможность просмотра информации о своем профиле

ФК4. Система должна предоставлять возможность редактирования информации своего профиля

Сотрудники:

ФК5. Система должна предоставлять возможность добавлять новый профиль сотрудника

ФК6. Система должна предоставлять возможность просмотра информации о профилях зарегистрированных сотрудников

ФК7. Система должна предоставлять возможность редактирования информации профилей зарегистрированных сотрудников

ФК8. Система должна предоставлять возможность удаления профилей зарегистрированных сотрудников

ФК9. Система должна предоставлять возможность фильтрации сотрудников по имени

Часто-задаваемые вопросы:

ФК10. Система должна предоставлять возможность добавлять ответ на часто-задаваемый вопрос

ФК11. Система должна предоставлять возможность добавлять новую категорию часто-задаваемых вопросов

ФК12. Система должна предоставлять возможность удаления категорий часто-задаваемых вопросов

ФК13. Система должна предоставлять возможность просмотра ответов на часто-задаваемые вопросы

ФК14. Система должна предоставлять возможность редактирования ответов на часто-задаваемые вопросы

ФК15. Система должна предоставлять возможность удаления ответов на часто-задаваемые вопросы

ФК16. Система должна предоставлять возможность фильтрации часто-задаваемых вопросов по категории и названию

Продукты:

ФК17. Система должна предоставлять возможность добавлять новый товар в список доступных для офиса товаров

ФК18. Система должна предоставлять возможность просмотра информации о доступных для офиса товарах

ФК19. Система должна предоставлять возможность редактирования информации доступных для офиса товарах

ФК20. Система должна предоставлять возможность удаления доступных для офиса товаров

ФК21. Система должна предоставлять возможность просмотра информации о заказанных сотрудниками товарах

ФК22. Система должна предоставлять возможность добавлять товары в список заказанных

ФК23. Система должна предоставлять возможность выгрузить в файл .xlsx данные о заказанных сотрудниками продуктах

Рестораны:

ФК24. Система должна предоставлять возможность выбора города, в котором находится офис ТЦР

ФК25. Система должна предоставлять возможность добавления города, в котором находится офис ТЦР

ФК26. Система должна предоставлять возможность удаления города, в котором находится офис ТЦР

ФК27. Система должна предоставлять возможность добавления нового заведения в список близлежащих ресторанов

ФК28. Система должна предоставлять возможность просмотра информации о близлежащих заведениях в виде списка

ФК29. Система должна предоставлять возможность просмотра информации о близлежащих заведениях на карте

ФК30. Система должна предоставлять возможность редактирования информации о близлежащих заведениях

ФК31. Система должна предоставлять возможность удаления близлежащих заведений

ФК32. Система должна предоставлять возможность просмотра отзывов к заведению

ФК33. Система должна предоставлять возможность удаления отзывов к заведению

События:

ФК34. Система должна предоставлять возможность просмотра информации о мероприятиях в виде списка

ФК35. Система должна предоставлять возможность добавления нового мероприятия в список мероприятий

ФК36. Система должна предоставлять возможность просмотра места проведения мероприятия на карте

ФК37. Система должна предоставлять возможность редактирования информации о мероприятиях

ФК38. Система должна предоставлять возможность удаления мероприятий

ФК39. Система должна предоставлять возможность фильтрации мероприятий по дате проведения

ФК40. Система должна предоставлять возможность фильтрации мероприятий по месту проведения

ФК41. Система должна предоставлять возможность фильтрации мероприятий по названию

ФК42. Система должна предоставлять возможность фильтрации мероприятий по формату проведения

ПРИЛОЖЕНИЕ В

(справочное)

DFD диаграмма

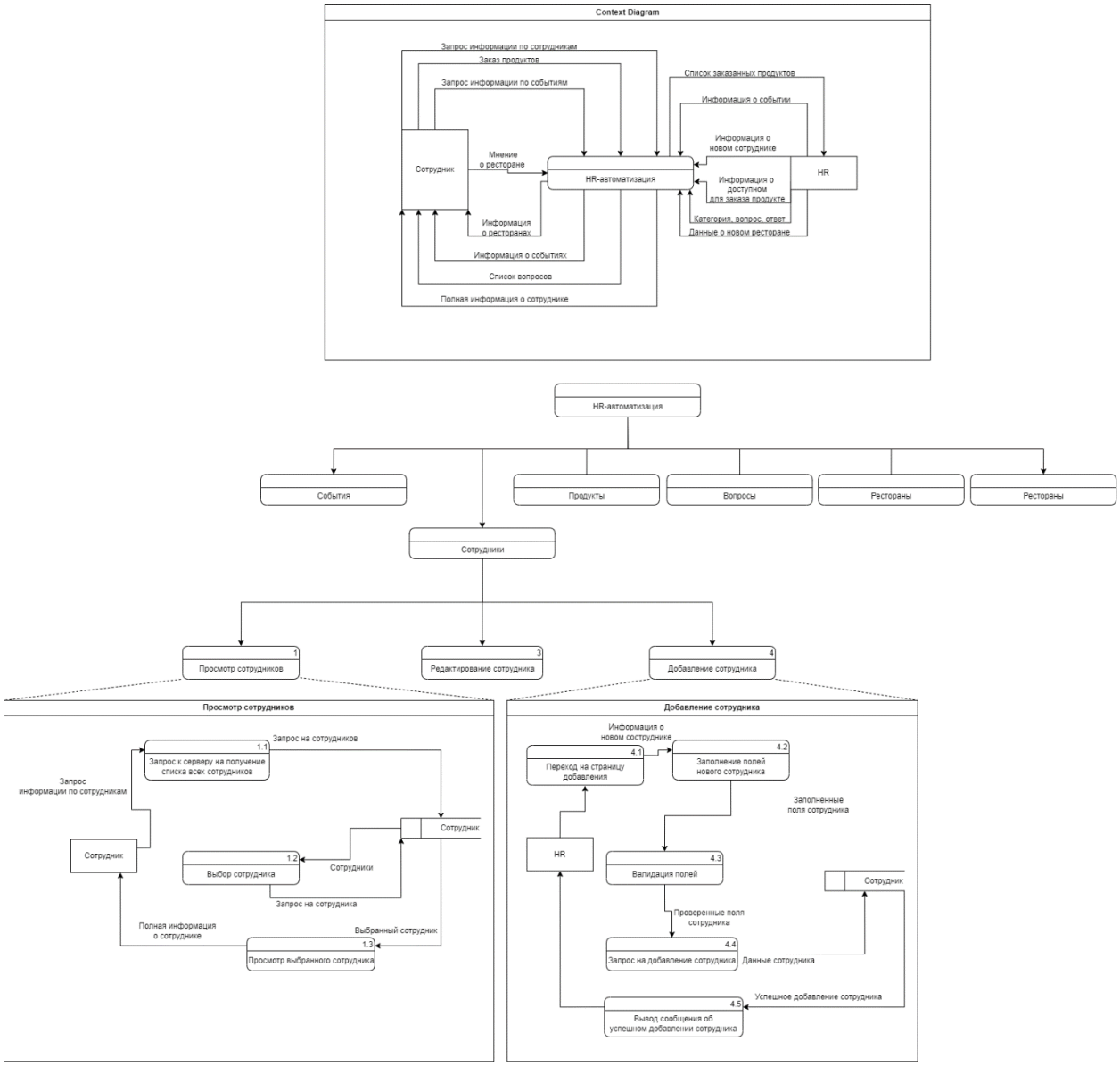

Рисунок В.1 - DFD диаграмма

ПРИЛОЖЕНИЕ Г

(справочное)

EPC диаграмма

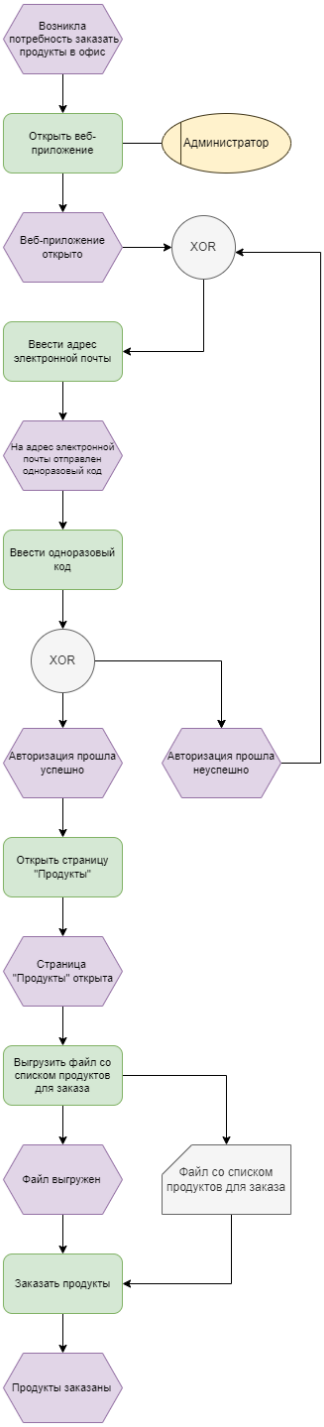

Рисунок Г.1 - EPC диаграмма

ПРИЛОЖЕНИЕ Д

(справочное)

Диаграмма вариантов использования мобильного приложения

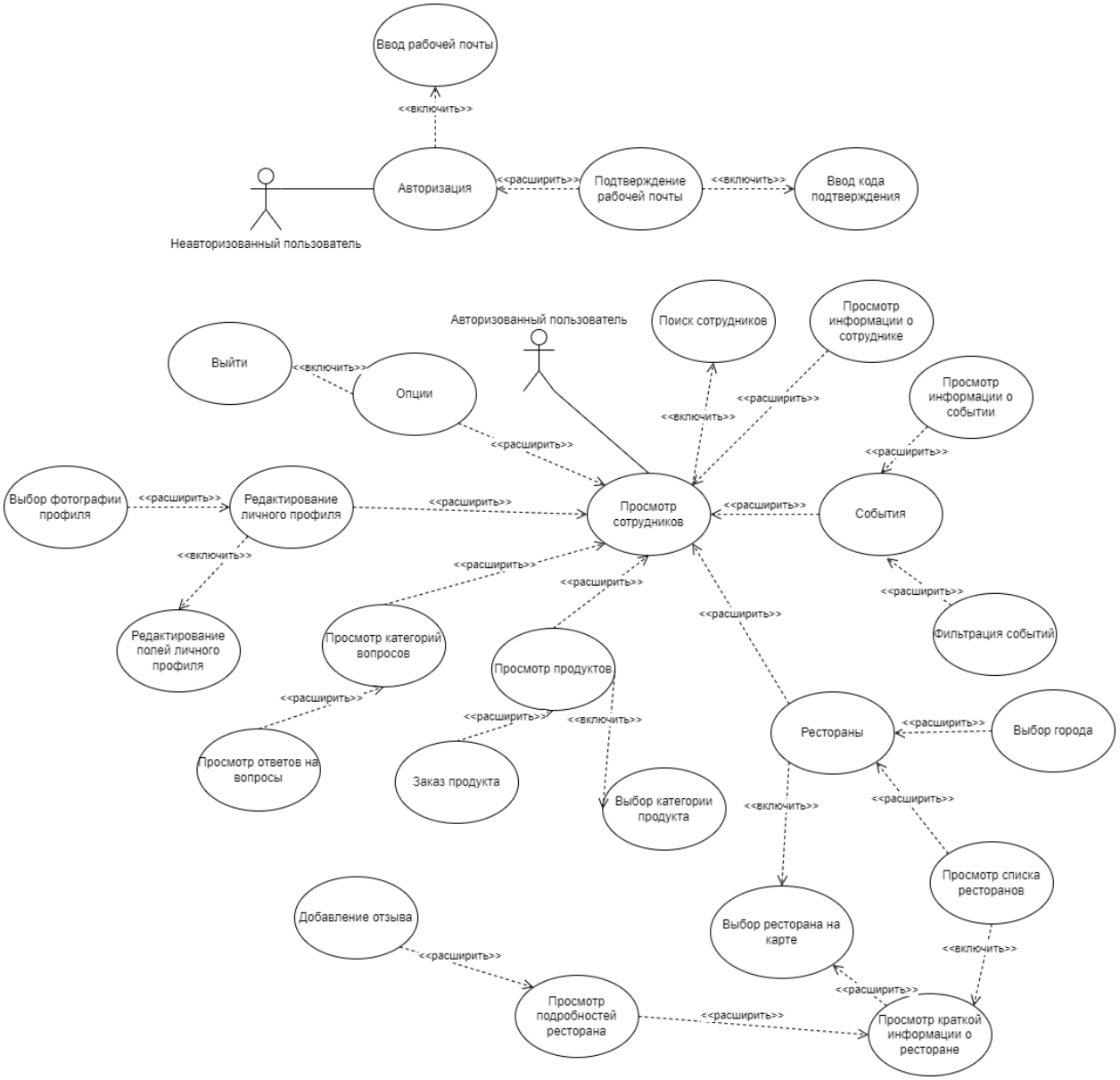

Рисунок Д.1 - Диаграмма вариантов использования мобильного приложения

ПРИЛОЖЕНИЕ Е

(справочное)

Диаграмма вариантов использования административного приложения

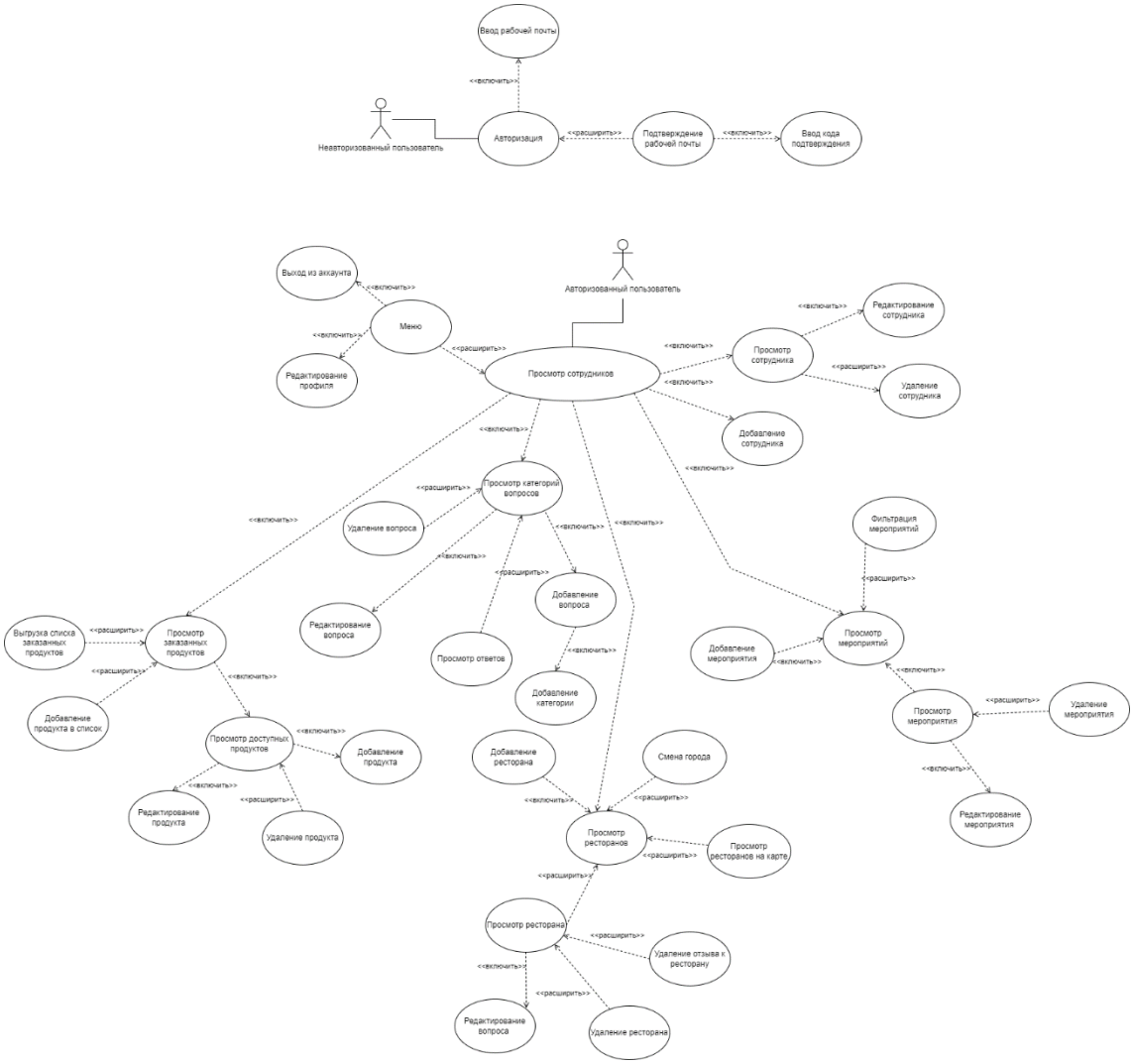

Рисунок Е.1 - Диаграмма вариантов использования административного приложения

ПРИЛОЖЕНИЕ Ж

(справочное)

Диаграмма компонентов мобильного приложения

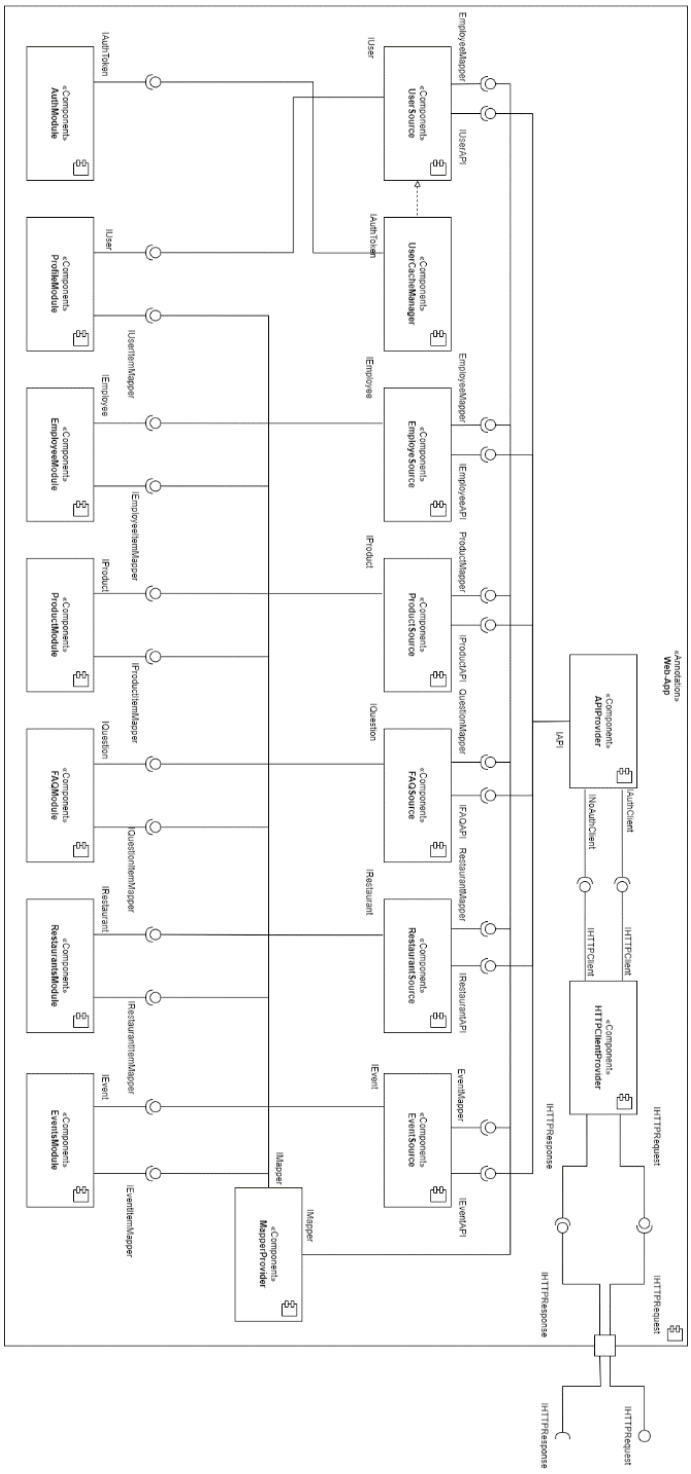

Рисунок Ж.1 - Диаграмма компонентов мобильного приложения

ПРИЛОЖЕНИЕ И

(обязательное)

Описание работ, выполненных совместно всеми участниками групповой/комплексной ВКР

Таблица И.1 - Описание работ, выполненных совместно всеми участниками групповой/комплексной ВКР

Общая тема работы	ФИО обучающегося	Индивидуальная тема работы
Разработка системы "HR-automation"	Байделюк Елизавета Андреевна	Разработка серверной части системы "HR-automation"
	Когут Арина Евгеньевна	Разработка веб-части системы "HR-automation"
	Сидоркин Александр Андреевич	Разработка мобильного приложения системы "HR-automation"

Вклад в групповую ВКР	Байделюк Елизавета Андреевна
В проекте Елизавета выступала в качестве backend-разработчика. Она разработала API для взаимодействия клиента с сервером, а также описала его в документации. Помимо этого, Елизавета участвовала в проектировании системы (Классификация функций методом MosCow, построение диаграммы	

вариантов использования, диаграмма общей архитектуры системы) и написании пояснительной записки для ВКР (Глава 5 «социальная ответственность», серверная часть в каждой основной главе).

Вклад в групповую ВКР	Когут Арина Евгеньевна
Арина занимала роль frontend-разработчика. Она разработала веб-часть системы, а именно административную панель. Была проделана работа по созданию макетов интерфейсов, реализации приложения и выгрузки на хостинг. Помимо этого Арина участвовала в проектировании системы (построение диаграммы idef0, описание контекста системы, написание функциональных требований) и написании пояснительной записки для ВКР (Глава 4. Финансовый менеджмент, ресурсоэффективность и ресурсосбережение», веб-часть в каждой основной главе).	

Вклад в групповую ВКР	Сидоркин Александр Андреевич
Александр занимал роль Android-разработчика в проекте. Он разработал мобильное приложение для платформы Android. Помимо этого, Александр участвовал в проектировании системы (построение диаграмм DFD, EPC и развертывания, поиск аналогичных решений) и написании пояснительной записки для ВКР (мобильная часть в каждой основной главе, общие части в основных главах).	

ПРИЛОЖЕНИЕ К

(справочное)

Результаты разработки

Рисунок К.1 – qr-код на мобильное приложение

Рисунок К.2 – qr-код на веб-приложение

Рисунок К.2 – qr-код на документацию к серверу