

Министерство образования и науки Российской Федерации

федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа: Информационных технологий и робототехники

Направление подготовки: 09.04.01 Информатика и вычислительная техника

Отделение школы (НОЦ): Информационных технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Разработка мобильного приложения для абитуриентов ТУСУР

УДК: 004.451:378.662:621.37/.39-057.84

Студент

Группа	ФИО	Подпись	Дата
8ВМ6Г	Алмабек Галымжан Маратулы		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Лунёва Елена Евгеньевна	к.т.н		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Рыжакина Татьяна Гавриловна	к.э.н		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Авдеева Ирина Ивановна	—		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Кочегурова Елена Алексеевна	к.т.н		

Томск – 2018 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Код результатов	Результат обучения (выпускник должен быть готов)
	Общепрофессиональные компетенции
P1	Воспринимать и самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте.
P2	Владеть и применять методы и средства получения, хранения, переработки и трансляции информации посредством современных компьютерных технологий, в том числе в глобальных компьютерных сетях.
P3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений и высказываний, основанных на интерпретации данных, интегрированных из разных областей науки и техники, выносить суждения на основании неполных данных, анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями.
P4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью и готовностью к саморегулированию дальнейшего образования и профессиональной мобильности. Владеть, по крайней мере, одним из иностранных языков на уровне социального и профессионального общения, применять специальную лексику и профессиональную терминологию языка.
	Профессиональные компетенции
P5	Выполнять инновационные инженерные проекты по разработке аппаратных и программных средств автоматизированных систем различного назначения с использованием современных методов проектирования, систем автоматизированного проектирования, передового опыта разработки конкурентно способных изделий.
P6	Планировать и проводить теоретические и экспериментальные исследования в области проектирования аппаратных и программных средств автоматизированных систем с использованием новейших достижений науки и техники, передового отечественного и зарубежного опыта. Критически оценивать полученные данные и делать выводы.
P7	Осуществлять авторское сопровождение процессов проектирования, внедрения и эксплуатации аппаратных и программных средств автоматизированных систем различного назначения.
	Общекультурные компетенции
P8	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
P9	Осуществлять коммуникации в профессиональной среде и в обществе в целом, активно владеть иностранным языком, разрабатывать документацию,

	презентовать и защищать результаты инновационной инженерной деятельности, в том числе на иностранном языке.
P10	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
P11	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа: Информационных технологий и робототехники
Направление подготовки (специальность): 09.04.01 Информатика и вычислительная техника
Отделение школы (НОЦ): Информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП

(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

магистерской диссертации

Студенту:

Группа	ФИО
8ВМ6Г	Алмабек Галымжану Маратулы

Тема работы:

Разработка мобильного приложения для абитуриентов ТУСУР

Утверждена приказом директора (дата, номер)

Срок сдачи студентом выполненной работы:

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

<p>Исходные данные к работе</p> <p><i>(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.).</i></p>	<p>Объект исследования – программное приложение для мобильных устройств.</p> <p>Предмет исследования – мобильное приложение для абитуриентов ТУСУР.</p> <p>Цель исследования – разработать мобильное приложение.</p>
<p>Перечень подлежащих исследованию, проектированию и разработке вопросов</p> <p><i>(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</i></p>	<ol style="list-style-type: none">1. Проанализировать и определить требования к программному продукту.2. Разработать архитектуру программного обеспечения с учетом требований.3. Выбрать платформу для реализации разработки.

<i>работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).</i>	<p>4. Изучить возможные технологии выбранной платформы.</p> <p>5. Разработка мобильного гибридного приложения и графического интерфейса.</p>
Перечень графического материала	презентация в формате *.pptx
Консультанты по разделам выпускной квалификационной работы	
Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Рыжакина Татьяна Гавриловна
Социальная ответственность	Авдеева Ирина Ивановна
Названия разделов, которые должны быть написаны на русском и иностранном языках:	
<ul style="list-style-type: none"> • Введение • Краткий анализ литературы • Анализ предметной области • Designing of mobile application • Разработка программного продукта • Тестирование программного продукта • Финансовый менеджмент, ресурсоэффективность и ресурсосбережение • Социальная ответственность • Заключение 	

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

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Лунёва Елена Евгеньевна	К.Т.Н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ6Г	Алмабек Галымжан Маратулы		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Информационных технологий и робототехники
Направление подготовки 09.04.01 Информатика и вычислительная техника
Уровень образования магистратура
Отделение школы (НОЦ) Отделение информационных технологий
Период выполнения осенний/весенний семестр 2017/2018 учебного года

Форма представления работы:

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

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	
--	--

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
	Основная часть	75
	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
	Социальная ответственность	10

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Лунёва Елена Евгеньевна	К.Т.Н.		

СОГЛАСОВАНО:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Кочегурова Е.А.	К.Т.Н.		

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСООБЪЕКТИВНОСТЬ И
РЕСУРСОСБЕРЕЖЕНИЕ»**

Студенту:

Группа	ФИО
8ВМ6Г	Алмабек Галымжан Маратулы

Школа	ИШИТР	Отделение	Отделение информационных технологий
Уровень образования	Магистратура	Направление/специальность	09.04.01 Информатика и вычислительная техника

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:

1. <i>Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих</i>	Работа с информацией, представленной в российских и иностранных научных публикациях, аналитических материалах, статических бюллетенях и изданиях, нормативно-правовых документах; анкетирование; опрос.
2. <i>Нормы и нормативы расходования ресурсов</i>	
3. <i>Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования</i>	

Перечень вопросов, подлежащих исследованию, проектированию и разработке:

1. <i>Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения</i>	Проведение предпроектного анализа. Определение целевого рынка и проведение его сегментирования. Выполнение SWOT-анализа проекта
2. <i>Определение возможных альтернатив проведения научных исследований</i>	Определение целей и ожиданий, требований проекта. Определение заинтересованных сторон и их ожиданий.
3. <i>Планирование процесса управления НИ: структура и график проведения, бюджет, риски и организация</i>	Составление календарного плана проекта. Определение бюджета НИ
4. <i>Определение ресурсной, финансовой, экономической эффективности</i>	Проведение оценки экономической эффективности разработки мобильного приложения «Абитуриент ТУСУР», с помощью гибридных технологий.

Перечень графического материала (с точным указанием обязательных чертежей):

1. Оценка конкурентоспособности технических решений
2. Матрица SWOT
3. График проведения и бюджет НИ
4. Расчёт денежного потока
5. Оценка ресурсной, финансовой и экономической эффективности НИ

Дата выдачи задания для раздела по линейному графику

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Рыжакина Татьяна Гавриловна	К.Э.Н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ6Г	Алмабек Галымжан Маратулы		

**ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»**

Студенту:

Группа	ФИО
8ВМ6Г	Алмабек Галымжану Маратулы

Школа	ИШИТР	Отделение	Информационных технологий
Уровень образования	Магистратура	Направление/специальность	09.04.01 Информатика и вычислительная техника

Исходные данные к разделу «Социальная ответственность»:

1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения

Объект исследования является разрабатываемая гибридное приложение для абитуриентов, «Абитуриент ТУСУР». Приложение включает в себя различный функционал для удобства отслеживания абитуриентом своего места в рейтинги зачисления, новостей, ближайших мероприятий и оценки своего шанса на поступление. Использование мобильного приложения может происходить везде, где есть сеть, и не требует никаких дополнительных устройств, таким образом негативное влияние на здоровье и жизнедеятельность человека минимальная. Разработка системы происходит в помещениях и требует работы с компьютерами и другими электронными устройствами, которые являются источниками вредных излучений и могут оказывать негативное влияние на здоровье и жизнедеятельность человека.

Перечень вопросов, подлежащих исследованию, проектированию и разработке:

<p>1. Производственная безопасность</p>	<p>Возможные вредные факторы в офисном помещении:</p> <ul style="list-style-type: none"> • Недостаточная вентиляция; • Недостаточное или неправильное освещение; • Пониженная или повышенная температура воздуха; • шум • Повышенный уровень электромагнитного излучения; • Нервно-психические перегрузки. <p>Возможные опасные факторы в офисном помещении:</p> <ul style="list-style-type: none"> • Короткое замыкание; • Электрический ток. • Статическое эл-во
<p>2. Экологическая безопасность</p>	<p>В процессе разработки и эксплуатации системы идентификации возможно образование следующих видов отходов:</p> <ul style="list-style-type: none"> • Твердые отходы: бумага, канцелярские принадлежности, комплектующие; • Люминесцентные лампы.
<p>3. Безопасность в чрезвычайных ситуациях</p>	<p>Наиболее типичная чрезвычайная ситуация при работе в офисе – пожар. Превентивные меры включают инструктаж по пожарной безопасности, контроль состояния проводки и электрических приборов, своевременное профилактическое обслуживание.</p>
<p>4. Правовые и организационные вопросы обеспечения безопасности</p>	<p>Параметры рабочего места офисного работника регулируются ГОСТ 12.2.032–78 ССБТ, СанПиН 2.2.2/2.4.1340–03, «Трудовой кодекс Российской Федерации» от 30.12.2001 ФЗ-№197.</p>

Дата выдачи задания для раздела по линейному графику	14.03.2018
--	------------

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
ассистент	Авдеева Ирина Ивановна	–		14.03.2018

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ6Г	Алмабек Галымжан Маратулы		14.03.2018

Реферат

Выпускная квалификационная работа состоит из 144 с., 30 рис., 30 табл., 38 источников, 6 прил.

Ключевые слова: мобильное приложение, проектирование, гибридные приложения, кроссплатформенные приложения, REST API.

Объектом исследования является разрабатываемое программное приложение для мобильных устройств.

Цель работы – создание мобильного приложения для абитуриентов Томского государственного университета систем управления и радиоэлектроники (ТУСУР).

В процессе исследования проводились анализ требований к мобильному приложению, сравнительный анализ гибридной и нативной разработки, сравнительный анализ платформ для разработки кроссплатформенных приложений, анализ архитектур для реализации программного приложения.

В результате исследования было спроектировано и разработано мобильное кроссплатформенное приложение для абитуриентов ТУСУР.

Степень внедрения: в данный момент мобильное приложение находится на этапе внедрения в ТУСУР и используется в тестовом режиме. Область применения: взаимодействие и привлечение абитуриентов в университет.

Экономическая эффективность/значимость работы: обусловлена возможностью привлечения большего количества абитуриентов с помощью разрабатываемого приложения.

Функциональность приложения может быть расширена добавлением авторизации пользователей через социальные сети и возможностью делиться с друзьями своими достижениями через социальные сети. Также в дальнейшем планируется распространение данного программного продукта на другие вузы.

Определения, обозначения, сокращения, нормативные ссылки

ПП – программный продукт

REST – Representational state transfer

UML – Unified modeling language

JSON – JavaScript object notation

TCP – transmission control panel

Оглавление

Введение.....	15
1 Теоретический анализ предметной области и постановка задачи.....	18
1.1 Анализ литературы использованной в работе.....	18
1.2 Анализ в области использования мобильного трафика пользователями сети Интернет.....	19
1.3 Обзор существующих решений.....	23
1.4 Цели работы и задачи.....	25
2 Проектирование мобильного приложения.....	26
2.1 Концептуальная модель программного продукта.....	26
2.2 Компонентная архитектура.....	30
2.3 Определение общих требований к программному продукту.....	34
2.4 Анализ современных средств разработки гибридных мобильных приложений.....	35
2.4.1 Обзор методов разработки мобильных приложений.....	35
2.4.2 Анализ платформ для разработки гибридных мобильных приложений.....	37
2.4.3 Обзор принципа работы фреймворка Ionic 3.....	39
2.4.4 Инструменты для реализации приложения.....	41
2.5 Архитектура программного продукта.....	42
2.5.1 Интерфейс административной панели: SPA на AngularJS.....	46
2.5.2 Интерфейс мобильного приложения: Фреймворк Express.....	46
2.5.3 Гибридная архитектура программного продукта.....	48
2.5.4 Аутентификация пользователей.....	49
2.5.5 Защита передаваемых данных.....	50
3 Разработка программного продукта.....	52
3.1 Серверная сторона программного продукта.....	52
3.1.1 Создание необходимых коллекций в MongoDB.....	52
3.1.2 Создание HTTP-сервера с возможностью подключения к БД.....	53
3.1.4 Определение основных маршрутов REST API.....	55

3.1.5	Тестирование REST API с использованием инструмента Postman.....	56
3.2	Административная панель.....	57
3.2.1	Верстка пользовательского интерфейса в соответствии с макетом	59
3.2.2	Разработка компонентой структуры веб-приложения	61
3.3	Мобильное приложение.....	62
3.3.1	Верстка пользовательского интерфейса в соответствии с макетом	63
3.3.2	Разработка компонентой структуры мобильного приложения	64
3.3.3	Интеграция мобильного приложения с API	65
3.3.4	Сборка мобильного приложения под определенные платформы	66
4	Финансовый менеджмент, ресурсоэффективности и ресурсосбережение	69
4.1	Предпроектный анализ	69
4.1.1	Потенциальные потребители результатов проекта	69
4.1.2	Анализ конкурентных технических решений	70
4.1.3	SWOT – анализ	72
4.2	Инициация проекта	75
4.2.1	Цели и результаты проекта	75
4.2.2	Организационная структура проекта	76
4.2.3	Ограничения и допущения проекта.....	76
4.3	Планирование управления научно – техническим проектом	77
4.3.1	Иерархическая структура работ проекта	77
4.3.2	План проекта.....	78
4.4	Бюджет научного исследования	80
4.4.1	Расчет материальных затрат.....	80
4.4.2	Основная заработная плата	81
4.4.3	Дополнительная заработная плата научно-производственного персонала	83
4.4.4	Накладные расходы.....	84
4.4.5	Формирование бюджета затрат научно-исследовательского проекта...	85
4.5	Реестр рисков проекта.....	85
4.5.1	Чистая текущая стоимость (NPV).....	86

4.5.2	Дисконтированный срок окупаемости.....	88
4.5.3	Внутренняя ставка доходности.....	88
4.6	Оценка абсолютной эффективности.....	90
4.7	Оценка сравнительной эффективности исследования	91
5	Социальная ответственность	95
5.1	Анализ вредных и опасных факторов, которые могут возникнуть на рабочем месте при выполнении проекта	95
5.2	Производственная безопасность.....	96
5.2.1	Микроклимат	97
5.2.2	Производственный шум	99
5.2.3	Электромагнитные поля	99
5.2.4	Психофизиологические факторы.....	100
5.3	Экологическая безопасность	102
5.3.1	Безопасность в чрезвычайных случаях.....	103
5.3.2	Мероприятия по предотвращению ЧС.....	105
5.4	Правовые и организационные вопросы обеспечения безопасности.....	106
	Заключение	109
	Список использованной литературы.....	111
	Приложение А	115
	Приложение Б.....	116
	Приложение В.....	118
	Приложение Г	119
	Приложение Д.....	121
	Приложение Е.....	122

Введение

Смартфоны становятся незаменимыми гаджетами для каждого человека. Сейчас намного чаще встречаются люди без персонального компьютера, но с несколькими телефонами. По данным Gartner, в 2017 году по всему миру было продано почти 1,5 млрд смартфонов против 1,4 млрд штук годом ранее [1], в связи с этим и число мобильных приложений с каждым днем стремительно растет. По состоянию на третий квартал 2017 года, более чем один миллион приложений были разработаны для Android, с более чем 25 млрд загрузок приложений [2]. Анализ, который проводился на протяжении 2011 года, показал, что более 67% мобильных разработчиков использовали платформу Android для разработки и публикации приложений.

Для многих отраслей деятельности человека мобильные устройства стали критически важным и стратегическим каналом продаж, маркетинга и т.п. но, когда предприятие управляет своим брендом через веб-приложение, оно сталкивается с проблемой доставки разнообразной информации как с сервера на мобильные устройства, так и с мобильных устройств на сервер.

Мобильный доступ к данным ускоряет выполнение бизнес-процессов фирмы. Мобильное согласование записок или же согласование праздничных дней уже считается обыкновенной частью работы в корпорациях. Мобильные приборы есть у всякого работника, в следствие этого фирма имеет возможность применить эти приборы без инвестиций в них. Огромное количество разнообразных мобильных устройств нужно принимать во внимание при разработке корпоративного мобильного приложения: необходимо чтобы приложение работало одинаково хорошо на всех этих устройствах [3].

В данной ВКР рассматривается создание мобильного приложения с внедрением JavaScript фреймворка Angular.

Внедрение JavaScript фреймворков разрешает делать свежие мобильные приложения с имеющейся командой разработчиков. Для фирмы это значит, собственно, что одна и та же команда имеет возможность выполнять различные задачи – от разработки вебсайтов до разработки мобильных приложений.

Ноутбук, планшет, электронная книга, смартфон – всего лишь неполный список мобильных устройств, которые ежедневно используют люди для работы, учебы, общения и отдыха. Однако все эти устройства в основном работают под собственной операционной системой (ОС) (iOS, Android, Windows Phone, BlackBerry и т. д.). Это создает определенные неудобства для обмена информацией между ними через нативные приложения.

SDK – разработка программного обеспечения - специализированный пакет разработки, который учитывает особенности конкретной мобильной операционной системы. Например, Android разработчик должен знать язык программирования Java и пакет Android SDK. Это условие накладывает определенные ограничения на разработчиков.

Решением данной проблемы, с одной стороны, может считаться создание веб – приложений, данный вариант знаком веб – разработчикам, которые в своей работе используют HTML, JavaScript. Другими словами, данный вид приложения является веб – сервисом в Интернет. Чтобы использовать его, нужно открыть веб – браузер на своем мобильном устройстве, ввести адрес в адресной строке и начать использовать этот сервис. Вышеупомянутый алгоритм часто очень неудобен [4].

Эти две технологии имеют свои преимущества и недостатки. Нативные приложения, которые можно загрузить из хранилища приложений единойжды и в дальнейшем использовать его без доступа в Интернет, но с другой стороны процесс разработки нативных приложений сложный и трудозатратный. Веб-приложения являются кроссплатформенными и требуют только наличия браузера и подключения к Интернету. С другой стороны, им ограничен доступ к внутренним ресурсам устройства.

Своеобразным решением являются гибридные мобильные приложения. Для их создания необходимы минимальные знания встроенных мобильных приложений IOS и Android, а также знания веб-технологий. Гибридные приложения, как и веб-приложения, программируются с использованием веб-технологий, но пакетируются как встроенные приложения. Данные мобильные приложения распространяются через общедоступные хранилища приложений, также, как и встроенные. В отличие от веб-приложений они имеют прямой доступ к ресурсам мобильного устройства, что делает их более функциональными.

1 Теоретический анализ предметной области и постановка задачи

Перед тем как приступить к проектированию и разработке программного продукта, необходимо провести обзор существующих аналогов, обозначить их достоинства и недостатки. В данной главе будет проведён анализ существующих теоретических и практических работ по теме автоматизации процесса выбора направления абитуриентом направления подготовки в соответствующем вузе. Показана актуальность решаемой задачи, проведен анализ используемой литературы, также поставлена цель данной работы и определены основные задачи.

1.1 Анализ литературы использованной в работе

Литературу по основной части работы ВКР можно условно разделить на три группы.

В первую группу включены [1, 2, 3, 4, 5]. Источники из которых были взяты различная статистика, с помощью которых были проведен анализ.

Во вторую группу можно включить литературы, в которых описываются рекомендации и методологии решений [6, 7, 9, 14, 21]. Данная группа является наиболее важной для производственной практики и магистерской диссертации, так как на основе этих рекомендаций можно создать программный продукт, отвечающий всем требованиям разработки гибридных мобильных приложений. Статьи и публикации из этой группы были в основном взяты из известных и популярных книг и журналов таких как: «NgBook 5», «Разработка мобильных гибридных приложений». К третьей группе литератур, относятся статьи содержащие конкретные примеры разработки мобильных приложений на основе рассуждения авторов [условно 10, 11, 17, 18].

Проведённый анализ показал, что выделенные группы наиболее полно раскрывают предметную область и сущность проблемы. Первая группа литературы все специфические термины, характеризующие выбранные технологии. Вторая группа описывает общие понятия и стандарты разработки мобильных приложений под управлением административной панели. А третья дополнит теоретическую часть работы фактами, примерами и актуальными проблемами, связанными с разработкой мобильных приложений.

1.2 Анализ в области использования мобильного трафика пользователями сети Интернет

Анализ в области использования сети Интернет среди молодежи является довольно актуальной для академических исследований, в фокусе которых находятся тренды развития современного общества.

Данная актуальность обусловлена максимальной вовлеченностью именно молодежной аудитории в интернет среду, также факт того, что интернет достаточно широко проник в повседневную жизнь россиян. По числу активных пользователей сети Интернет Россия занимает одно из лидирующих мест по всему миру.

На данный момент люди во всем мире пользуются интернетом преимущественно со смартфонов [5]. Тем самым генерируя больше веб-трафика, чем все другие устройства суммарно. На 2018 год доля трафика, приходящаяся на мобильные телефоны, составляет 52% от общей суммы интернет трафика.

К тому же эти данные относятся только к веб-пользованию. Согласно исследованиям компании «App Annie», которая занимается исследованиями рынка мобильных приложений, на данный момент люди проводят в мобильных приложениях в 7 раз больше времени, чем в адаптированные под мобильные телефоны браузеры. Из чего следует, что доля смартфонов в интернете, вероятнее всего даже превосходит указанные цифры.

Последние сведения от Facebook только подтверждает данное предположение: всего лишь 5% мировой аудитории соцсети пользуются платформой исключительно с десктопа. К тому же пользователи тратят 89% мобильного времени на приложения, а остальные 11% на – веб-сайты [6].

Более того по данным «Retail & E-Commerce», потребители предпочитают приложения чем адаптивные сайты из-за пользовательского опыта, скорости, дополнительных функций и специальных предложений.

Один из российских филиалов исследовательского концерна GFK (Gesellschaft für Konsumforschung) Group, 16 января 2018 года опубликовал отчёт [6], «Проникновение Интернета в России: итоги 2017 года». По данным, которых аудитория пользователей сети интернет в возрасте от 16 лет и старше составила 87 миллиона человек, что на 3 миллиона больше прошлогодних результатов.



Рисунок 1.1 – Использование сети Интернет среди молодых россиян

Также по данным той же организации использование сети Интернет среди молодых россиян от 16 до 29 лет достигло предельных значений и составляет около 98%.

Согласно источнику [6] использование 30 наиболее посещаемых ресурсов сети Интернет производится через мобильные приложения, что означает, что мобильные приложения используются так же часто, как и веб-сайты. Это происходит потому, что большинство пользователей начинают использовать веб-сайт (или веб-приложение) перед загрузкой собственного мобильного приложения. После того, как определенный уровень взаимодействия был создан, пользователи переключаются на приложение для лучшего удобства и скорости работы. Другими словами, мобильные веб-сайты лучше подходят для начальной осведомленности и участия, но последние этапы привлечения лучше всего обслуживаются приложениями.

Блог «Swerve» недавно окрестил приложения «Новыми картами лояльности» из-за их сильного потенциала для улучшения лояльности клиентов [7]. Что они имели в виду, так это то, что приложения обладают несколькими ключевыми функциями и преимуществами, которых нет на веб-сайтах (включая мобильные оптимизированные). Например:

- Приложения могут отправлять push-сообщения, которые заставляют клиентов работать с вашим электронным магазином.
- Значки приложений видны, когда пользователь исследует свой телефон, сохраняя высокую узнаваемость бренда.
- Приложения могут сохранять пользовательские данные, позволяя проверять в один клик и минимизировать время загрузки.

По этим причинам 40,4% всех пользователей покупают больше товаров бренда после загрузки своего приложения для электронной коммерции. 45,9% также чаще посещают свой физический магазин. Эти номера показывают, что пользователи не только используют приложения, но и принимают меры после их

посещения. Сравните это с мобильными веб-сайтами, которые едва сохраняют 40% всех посетителей за первые несколько секунд, и понятно, почему приложения важны для лояльности к бренду.

Не мало важным фактором является, что более 16 миллионов россиян пользуются Интернетом только на мобильных устройствах.

Доля пользователей сети интернет только на мобильных устройствах составляет среди молодежи 16-29 лет – 18,4 % (Рисунок 1.2).

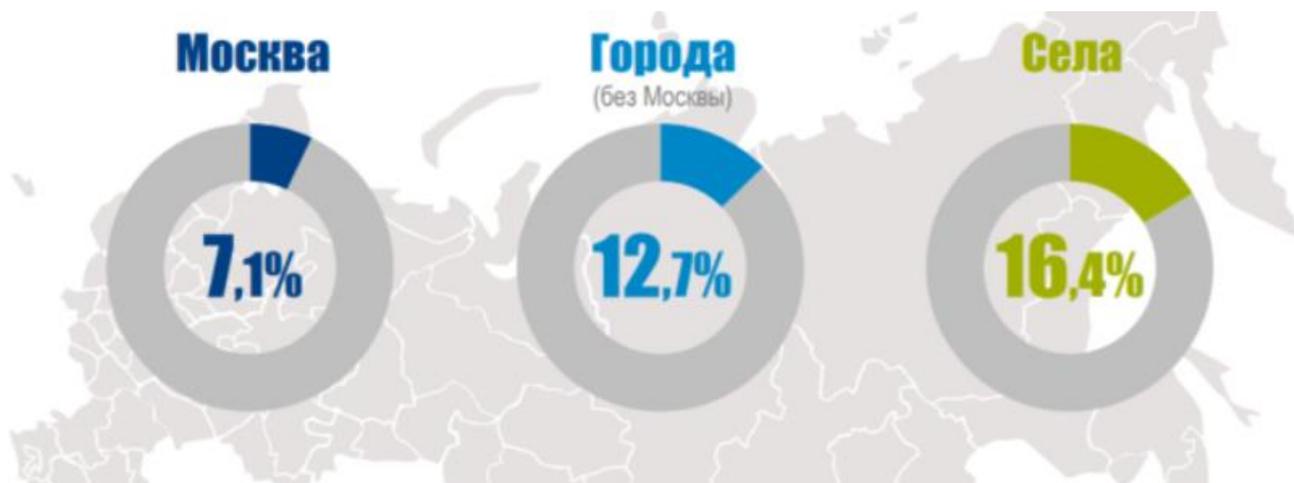


Рисунок 1.2 – Доля пользователей мобильного Интернет на российском рынке

При анализе интернет-практики молодых людей необходимо пересмотреть их с использованием новейших инструментов для преобразования реальных практик в виртуальную форму, то есть мобильных приложений. Актуальность этого исследования объясняется увеличением использования мобильного Интернета, а также установленным потребительским спросом для современных смартфонов, особенно среди молодежи. Исследование фокусируется на самых популярных приложениях для мобильных устройств Google Play. Предметом исследования является трансформация повседневных практик. Исследование приводит к тому, что в повседневной жизни существует общая тенденция к изменению.

1.3 Обзор существующих решений

Так как приложение разрабатывалось для вуза, расположенного в городе Томск, целесообразно рассматривать конкурентов именно в этом городе. На сегодняшний день есть 2 решения в вузах, также наиболее близкие по требуемому функционалу: «ТПУ Абитуриент» и «Хочу в ТГУ».

«ТПУ Абитуриент» мобильное приложение для абитуриентов, поступающих в Томский Политехнический Университет [8]. В приложении довольно приятный интерфейс, с возможностью войти в личный кабинет и отредактировать личные данные, так же в приложении реализована возможность отвечать на вопросы викторин тем самым зарабатывая баллы с возможностью выиграть один из главных призов.

«Хочу в ТГУ» мобильное приложения для абитуриентов Томского Государственного Университета [9]. В приложении много возможностей, но при клике на них половина перенаправляет на веб сайт, тем самым теряется автономность приложения. Так же реализован функционал «Калькулятор ЕГЭ», смысл, которого состоит в выборе предметов, сданных на Едином Государственном Экзамене, на основе которых составляется вероятность поступления абитуриента на определенные направления.

По характеристикам программной реализации все системы представляют собой законченные программные продукты и обеспечивают в той или иной мере наличие основных функций для комфортной работы абитуриентов. Для наглядного представления о достоинствах и недостатках систем составлена таблица 1.1.

Таблица 1.1 – Сравнение функционала систем

Параметры	Приложения		
	«Абитуриент ТУСУР»	«Абитуриент ТПУ»	«Хочу в ТГУ»
Возможность управления личными данными	✓	✓	–
Калькулятор ЕГЭ	✓	–	✓
Возможность просмотра рейтинга в списке	✓	✓	✓
Дополнительные приложения	✓	✓	–
Возможность просмотра новостей и мероприятий	✓	✓	✓
Дизайн	✓	–	–
Кроссплатформенность	✓	✓	✓
Автономность	✓	–	–
Фильтрация направлений по набранным баллам	✓	–	✓

Исходя из таблицы 1.1 можно заметить, что наиболее полно включает в себя характеристики эталонного приложения мобильное приложение для абитуриентов ТУСУР. Так же можно заметить, что наибольшим функционалом обладает именно данное приложение.

На основании этих факторов было принято решение начать разработку мобильного приложения «Абитуриент ТУСУР».

1.4 Цели работы и задачи

Проведенный анализ, посвященный определению подходящего программного продукта для упрощения процесса выбора абитуриентами направления подготовки, показал, что существующие программные продукты не в полной мере решают поставленную задачу.

Исходя из вышеуказанных результатов обзора, целью данной магистерской диссертации является разработка программного приложения «Абитуриент ТУСУР», позволяющий автоматизировать задачи, связанные с оповещением абитуриентов и взаимодействия с ними в рамках приёмной кампании.

Для достижения поставленной цели, требуется решить следующие задачи:

- Выявление требований, которыми должно обладать мобильное приложение;
- Выявление требований, которыми должно обладать программное приложение для управления контентом данного мобильного приложения;
- Спроектировать архитектуру программного продукта;
- Разработка удобного интерфейса для пользователей мобильного приложения;
- Внедрение разработанного программного продукта.

2 Проектирование мобильного приложения

После того как проведен анализ и обоснована необходимость выполнения данной работы, необходимо произвести проектирование реализуемого программного продукта.

2.1 Концептуальная модель программного продукта

Для описания работы системы необходимо построить концептуальную модель разрабатываемой системы. Такая модель должна быть адекватна для предметной области; следовательно, она должна содержать в себе знания всех участников процессов системы.

Для начала выделим актёров, которые будут взаимодействовать с программным продуктом:

- администратор программы – специалист, который будет управлять всем контентом в системе, и он же будет иметь полный доступ ко всем функциям;
- пользователь – в основном это абитуриенты, которые поступают в вуз, но в дальнейшем приложением может пользоваться и студент.

Так же выделим основные прецеденты:

Просмотр новостей и мероприятий

Исполнитель: пользователь. Производится в приложении.

Описание шагов:

1. авторизоваться в системе;
2. перейти к модулю «Новости» или «Мероприятия» с помощью навигационного меню;
3. выбрать интересную новость или мероприятие.

Участие в викторине

Исполнитель: пользователь. Производится в приложении.

Описание шагов:

1. авторизоваться в системе;
2. перейти к модулю «Викторина» с помощью навигационного меню;
3. выбрать не отвеченный вопрос;
4. ответить на вопрос, заработать очки.

Добавление новости или мероприятия

Исполнитель: администратор. Производится в административной панели.

Описание шагов:

1. авторизоваться в системе;
2. перейти к модулю «Новости» или «Мероприятия» с помощью навигационного меню;
3. перейти к добавлению новости или мероприятия;
4. оформить новость или мероприятие как нужно и добавить.

Взаимодействие актёров с программным продуктом и описание прецедентов графически представлена на рисунке 2.1 в виде диаграммы прецедентов.

Диаграмма прецедентов (диаграмма вариантов использования) в UML — диаграмма, отражающая взаимоотношения между актёрами и прецедентами и являющаяся составной частью модели прецедентов, позволяющей описать систему на концептуальном уровне [10].

Прецедент — вероятность имитируемой концепции (доля ее функциональности), вследствие которой юзер способен приобрести определенный, измеримый и необходимый ему итог. Факт отвечает единичному сервису концепции, устанавливает единственный с альтернатив ее применения и представляет классический метод взаимодействия юзера с системой. Варианты

использования как правило используются с целью спецификации наружных условий к концепции.

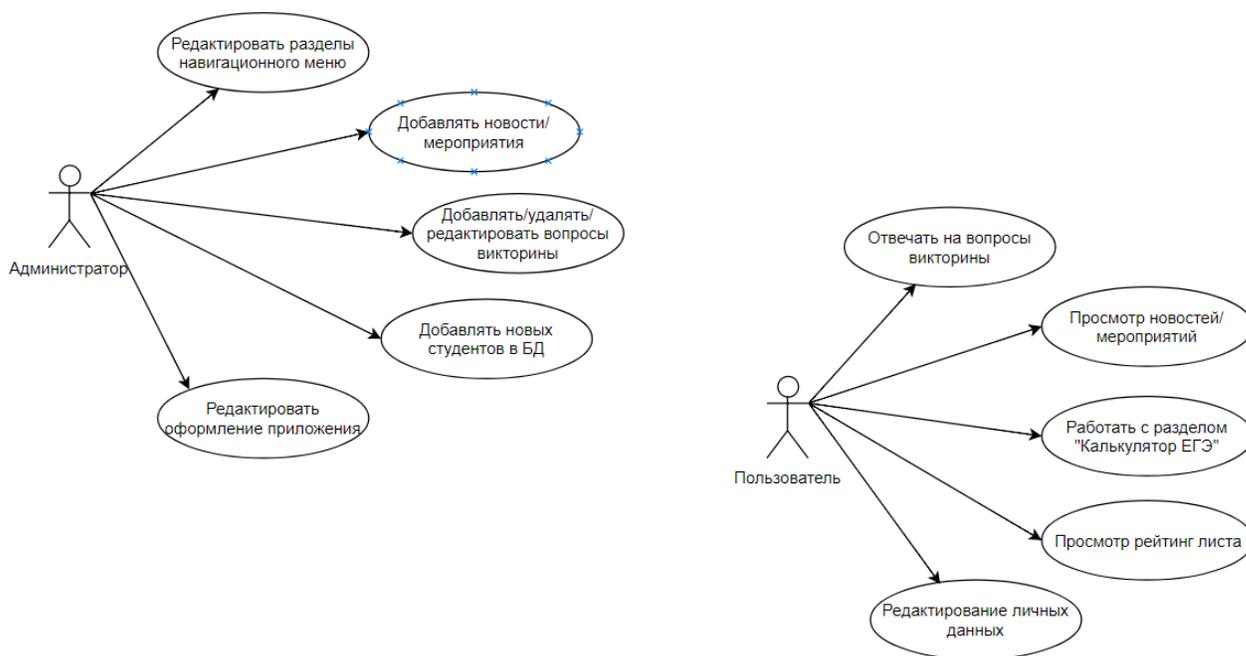


Рисунок 2.1 – Диаграмма прецедентов программного продукта

Процесс ответа пользователем на отправку запроса можно представить в виде диаграммы деятельности (рисунок 2.2).

Диаграмма активности – диаграмма UML, которая показывает действия в состояниях, описана на диаграмме состояний. Под деятельностью понимается спецификация исполняемого поведения в последовательной и параллельной координированной форме выполнения подчиненных элементов: вложенные действия и отдельные действия на английском языке. действия, связанные друг с другом потоками, которые идут от выходов одного узла к входам другого [10].

Диаграммы деятельности используются при моделировании бизнес-процессов, технологических процессов, параллельных и параллельных систем.

Диаграммы активности состоят из ограниченного числа фигур, соединенных стрелками. Основные фигуры:

1. Прямоугольники с искривлениями – действия
2. Ромбы – решения
3. Широкие полосы – начало (ветвь) и конец ветви действий
4. Черным кругом является начало процесса
5. Черным кругом с обводкой является концом процесса

Стрелки идут от начала до конца процесса и показывают управляющие потоки или потоки объектов (данных).

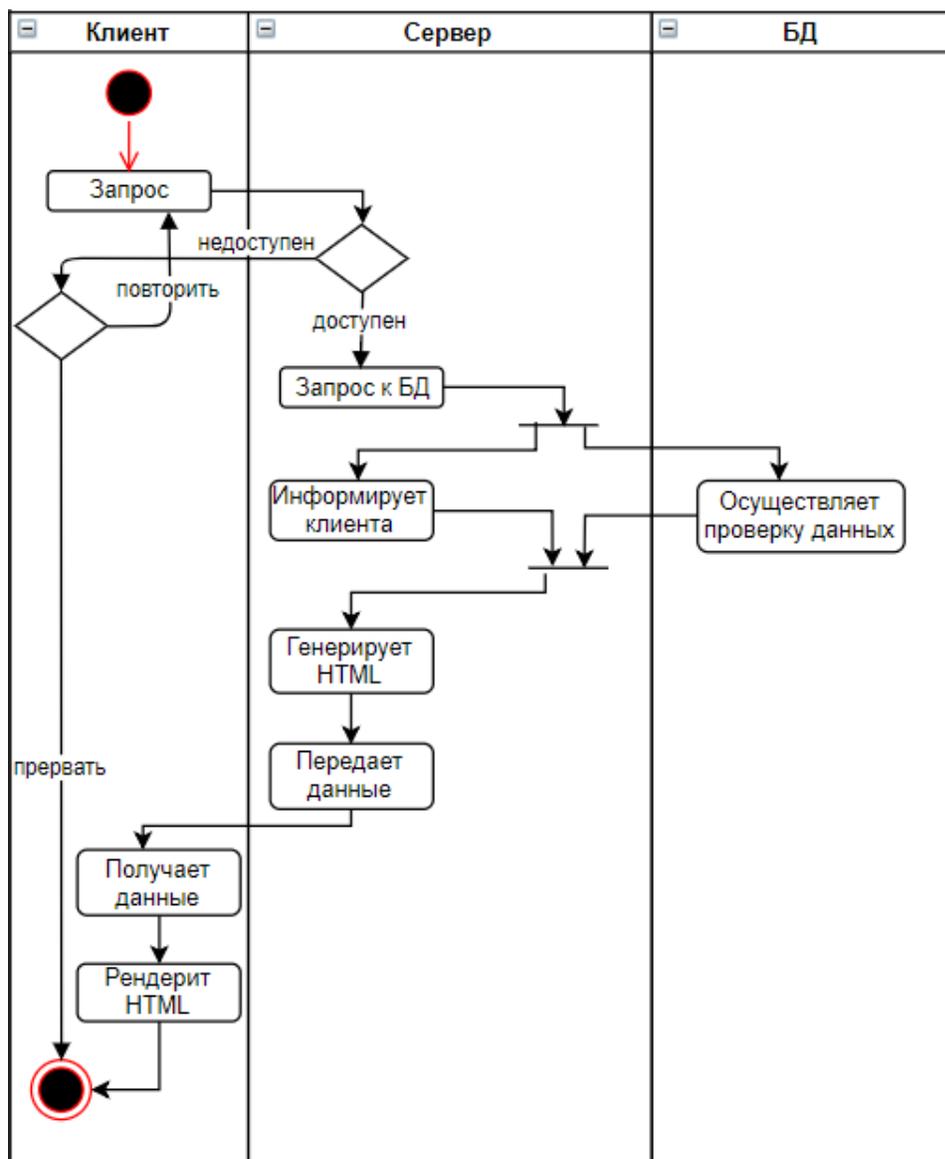


Рисунок 2.2 – Диаграмма деятельности процесса «Ответ на викторину»

2.2 Компонентная архитектура

Компонентная архитектура описывает подход к проектированию и разработке систем с использованием методов разработки программного обеспечения. Основное внимание в этом случае уделяется разложению дизайна на отдельные функциональные или логические компоненты, которые обеспечивают четко определенные интерфейсы, которые содержат методы, события и свойства. В этом случае обеспечивается более высокий уровень абстракции, чем при объектно-ориентированном развитии, и внимание не фокусируется на таких проблемах, как протоколы связи или общее состояние [11].

Существует два подхода к реализации приложения для абитуриентов ТУСУР:

- Однокомпонентная архитектура;
- Многокомпонентная архитектура.

Однокомпонентная архитектура:

В однокомпонентной архитектуре приложение устанавливается из магазина приложения с уже загруженным контентом. Вся логика изложена в одно приложение, индивидуально для каждой платформы.

Преимущества этой архитектуры:

- Меньше трудоемкости развития;
- Независимость подключения к Интернету;
- Безопасность передачи данных.

Недостатки этой архитектуры:

- Долгое время для обновлений контента;
- Не позволяет использовать социальное взаимодействие между пользователями;
- Трудности при переходе на новую платформу.

Многокомпонентная архитектура

Многокомпонентная архитектура или клиент-сервер – это приложение с графическим интерфейсом, который связывается с сервер базы данных, в котором хранится постоянно обновляемый контент и основная часть бизнес-логики. Как правило, этот стиль описывает отношения между клиентом и сервером (или серверами), где из клиент выполняет последовательность действий, запрос ожидает ответ - обработка при получении. Сервер, в свою очередь, разрешает пользователь - выполняет обработку, необходимую для получения результата – передает результат.

Основными преимуществами многокомпонентной архитектуры являются:

- Удобство поддержки. Уровни не зависят друг от друга, что позволяет выполнять обновления или изменения, не затрагивая приложение в целом.
- Масштабируемость. Уровни организованы на основе развертывания слоев, поэтому вы можете просто масштабировать приложение.
- Гибкость. Управление и масштабирование каждого уровня могут выполняться независимо, что обеспечивает повышенную гибкость.
- Доступность. Приложения могут использовать модульную архитектуру, которая позволяет системе использовать легко масштабируемые компоненты, что увеличивает доступность.

Сравнивая достоинства и недостатки рассмотренных архитектур, был сделан вывод о том, что наиболее подходящая архитектура для реализации руководство по применению – это многокомпонентная архитектура.

Как архитектурное решение используется один из видов многокомпонентной архитектуры – трехуровневая архитектура.

Данная архитектура позволяет разумно распределять модули обработки данных, которые передаются на один или несколько отдельных серверов. Одно из очевидных преимуществ этой архитектуры заключается в том, что сервера могут взаимодействовать друг с другом, это позволит разделить систему на более

подробные функциональный блоки с определенными ролями. Данная архитектура графически представлена на рисунке 2.3.

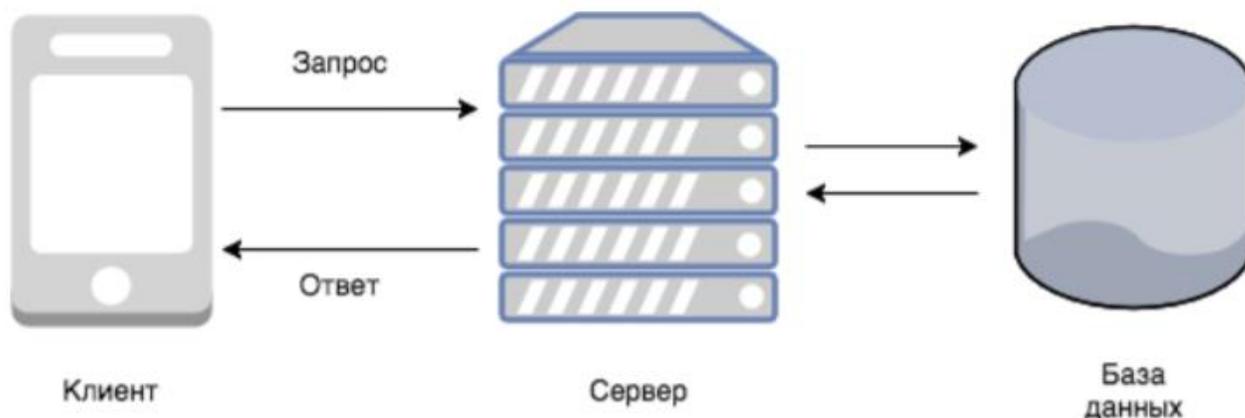


Рисунок 2.3 – Трехуровневая компонентная архитектура

Трехуровневая компонентная архитектура имеет три основных уровня:

Уровень клиента

Представляет собой графический интерфейс, который предоставляет функции ввода и отображения данных. В данном уровне хранится простая бизнес-логика и содержит состояние приложения. В разрабатываемой системе на данном уровне будет располагаться:

- Бизнес-логика для отправки запроса на сервер, получение и обработка полученных данных;
- Бизнес-логика взаимосвязи работающих компонентов через сервисы и провайдеры;
- Взаимодействие элементов графического интерфейса.

Уровень сервера приложений

Данный уровень является связующим звеном между уровнями клиента и базой данных, в котором хранятся большая часть бизнес-логики. На этом уровне в разрабатываемой системе будет содержаться:

- Бизнес-логика для приема и обработки запросов от клиента;
- Создание коллекций для базы данных;
- Бизнес-логика взаимодействия с базой данных.

Уровень базы данных

Данный уровень будет осуществлять хранение и обработку данных, будет взаимодействовать исключительно с уровнем сервера приложений.

Для представления взаимосвязи между логическими и физическими модулями, необходимо создать диаграмму компонентов. UML диаграмма компонентов графически показан на рисунке 2.4.

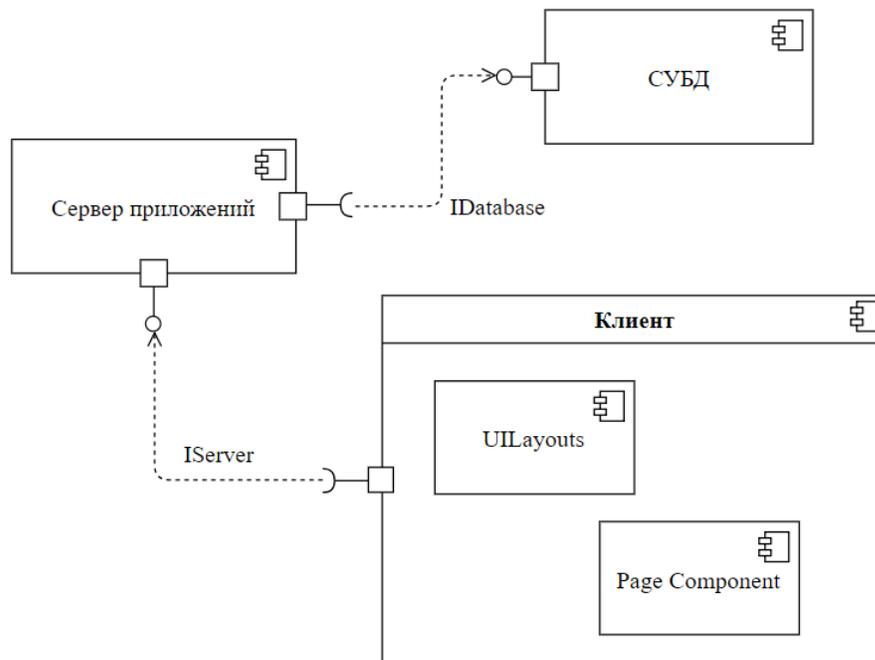


Рисунок 2.4 – UML диаграмма компонентов

Компонент «Сервер приложений» - реализует бизнес-логику для приема и обработки запросов от клиента, предоставляет интерфейс `IServer` и потребляет интерфейс `IDatabase`.

Компонент «СУБД» - реализует доступ к данным и предоставляет интерфейс `IDatabase`.

Компонент «Клиент» - потребляет интерфейс IServer, состоит из двух компонентных частей:

- UILayouts – компонентная часть, содержащая элементы пользовательского интерфейса;
- Page Component – компонентная часть, служит для вывода новых страниц и элементов меню.

2.3 Определение общих требований к программному продукту

В данном разделе будут определены требования к программному продукту. Требования со стороны администратора приложения:

- ПП должен позволять вносить новые разделы в навигационное меню;
- ПП должен позволять работать с различными данными;
- Все изменения должны вноситься через административную панель;
- ПП должен хранить все данные в базе данных;
- ПП должен быть кроссплатформенным;

Определим требования, предъявляемые со стороны пользователя:

- Все изменения должны быть автоматически доступны для приложения;
- ПП должен позволять просматривать рейтинг лист;
- ПП должен позволять просматривать новости и мероприятия;
- ПП должен позволять редактировать личные данные;
- У ПП должен быть понятные интерфейс;
- ПП должен позволять отвечать на вопросы викторины;

Функциональные требования к программному продукту

Исходя из вышеперечисленных требований, определим список функций разрабатываемой программы:

- Возможность создания новостей и мероприятий;
- Возможность создания викторины;
- Возможность создания новых страниц для приложения;
- Возможность создания новых разделов в приложении;
- Возможность редактирования пользователей;
- Возможность удаления пользователей;
- Проверка корректности введенной информации.

2.4 Анализ современных средств разработки гибридных мобильных приложений

2.4.1 Обзор методов разработки мобильных приложений

Вопрос, который часто встречается при разработке приложения, заключается в том, нужно ли создавать нативное мобильное приложение, гибридное приложение или сделать свой сайт адаптивным для мобильных устройств [12].

Нативные приложения предназначены для работы в конкретной мобильной среде. Они не будут работать на разных мобильных ОС. Например, если планируется создавать приложение для iOS, необходимо создать его в Swift или Objective-C.

Гибридные приложения – кроссплатформенные приложения, которые работают одинаково хорошо на разных платформах. Он разработан с использованием конкретного стандартного языка кодирования (например, JavaScript, CSS и HTML), а затем организован для работы на разных платформах.

Отдельные взаимодействия, связанные с различными устройствами, часто достигаются за счет применения плагинов для каждой операционной системы.

Важным преимуществом использования гибридного приложения является то, что он позволяет приложению запускаться на разных ОС без больших расходов. Если разработка приложения выполнена правильно, пользователи не смогут отличить, построено ли приложение с использованием гибридного или родного маршрута.

Далее проведем обзор этих двух подходов разработки (Таблица 2.1).

Таблица 2.1 – Обзор возможных решений разработки

	Нативный	Гибридный
Знание технологий	Objective-C, iOS, SDK, Java, Android SDK	HTML, CSS, JavaScript, фреймворк для мобильной разработки
Распространение	AppStore/PlayMarket	AppStore/PlayMarket
Скорость разработки	Медленная	Высокая
Стоимость разработки	Высокая	Низкая
Стоимость поддержки	Высокая	Низкая
Производительность	Высокая	Средняя
Доступ ко всему функционалу устройства	Да	Да
Лучше использовать	Во всех приложениях, где необходима производительность, график	В приложениях, которые не имеют высоких требований к производительности.

В ходе обзора возможных решений данной проблемы было решено, что лучшим решением будет разработка гибридного приложения, так как мы разрабатываем приложение, для абитуриентов которое не требует большой производительности и работы с нативным функционалом мобильного устройства.

После того как было решено разработать кроссплатформенное, гибридное приложение, необходимо выбрать какой из платформ для этих целей использовать.

2.4.2 Анализ платформ для разработки гибридных мобильных приложений

Кроссплатформенная разработка, позволяет разработчикам писать код один раз и применять некоторые его части на всех платформах. Обычно это происходит за счет производительности и поведения приложения.

Кроссплатформенные фреймворки Ionic 3 [13, 14], Xamarin [15], Unity, Qt и Appcelerator Titanium, Telerik Platform на сегодняшний день занимают 80% рынка кроссплатформенной разработки для мобильных устройств [16]. Будут учитываться следующие сравнительные характеристики:

- языковой стек;
- графический пользовательский интерфейс;
- сообщество;
- цена.

Ionic является open source проектом для разработки гибридных приложений, построенных на Angular JS и Apache Cordova. Инструменты и услуги, необходимые для разработки кроссплатформенных приложений, предоставляются с использованием известных веб-технологий, таких как CSS, HTML5 и SaaS.

Xamarin – фреймворк, используемый для разработки кроссплатформенных приложений. Он широко используется для связи с интерфейсом прикладных программ (API) нативных функций мобильных устройств, таких как контакты, камера и геолокация для операционных систем Android, iOS и Windows. Это позволяет разработчикам использовать почти 100% нативных библиотек Android и iOS, Xamarin лучше всего подходит для создания приложений с использованием

языка программирования C #, работающего на .NET Common Language Infrastructure (CLI).

Unity – мультиплатформенный инструмент для разработки 2D и 3D приложений и игр Unity, также один из лучших инструментов для демонстрации 3D контента. Созданные с помощью Unity приложения работают под операционными системами Windows, OS X, Linux, Android, Apple iOS, Windows Phone, BlackBerry, а также на игровых приставках Wii, PlayStation 3 и Xbox 360.

Titanium – это фреймворк для мобильных приложений с открытым исходным кодом, созданная компанией Appcelerator. Структура кодируется таким образом, что она предлагает интегрированную среду разработчикам для разработки широкого спектра мобильных приложений. На основе JavaScript он также влечет за собой HTML5, CSS3 и JQuery.

Telerik AppBuilder – одной из основных причин использовать AppBuilder является полноценная онлайн IDE. Она позволяет создавать, тестировать и даже публиковать гибридные приложения с любого компьютера или мобильного устройства, без необходимости в его загрузке.

Главные преимущества вышеперечисленных платформ рассмотрены в таблице 2.2.

Таблица 2.2 – Сравнительный анализ платформ

	Ionic 3	Xamarin	Unity	Appcelerator Titanium	Telerik Builder
Языки	JavaScript, HTML5, CSS3	C#, Xaml	C#, Unity Script, Boo	JavaScript, Python, Ruby, PHP	.NET, JavaScript, HTML5, Java, PHP
Платформы	iOS, Android, Windows	iOS, Android, Windows	Android, iOS, Windows Phone, Tizen, PS 4, Xbox One	iOS, Android, Windows, Tizen, Denso	iOS, Android, BlackBerry, Windows, Windows Phone

Продолжение таблицы 2.2

Цены	Платная версия: от 29\$ Бесплатная версия: доступна Adobe Creative Cloud Membership: доступно	Xamarin, Visual Studio Community: бесплатно Visual Studio Professional и Enterprise: доступно	Personal Edition: бесплатно Professional Edition: от 75 \$ в месяц	Есть бесплатный пробный период Indie: 39\$ в месяц Pro: \$99 в месяц	Есть бесплатный пробный период Цена от 39\$ в месяц
Межплатформенная переносимость	Свыше 98% исходного кода может использоваться повторно	Бизнес логика, доступ к данным, сетевая связь	Не полностью	Не полностью	Не полностью
Open source	+	–	–	+	–
UI	Web	Native	UI Canvas	Native	Web
Сообщество	Большое	Большое	Среднее	Маленькое	Маленькое

В ходе вышеуказанного сравнения было решено, что лучшим выбором для данной разработки будет Ionic 3

2.4.3 Обзор принципа работы фреймворка Ionic 3.

По своей сути, Ionic Framework — это дополнение над очень популярным фреймворком Apache Cordova, но со своим мощным CLI (Command Line Interface) и объемной документацией [17]. Следуя принципам Apache Cordova, приложения на Ionic Framework — это гибридные HTML приложения. Такие приложения на мобильном устройстве выполняются в специальной оболочке (UIWebView для iOS и WebView для Android), которая позволяет показывать HTML и выполнять JavaScript. На следующей диаграмме показан высокоуровневый вид архитектуры приложения Apache Cordova (Рисунок 2.3).

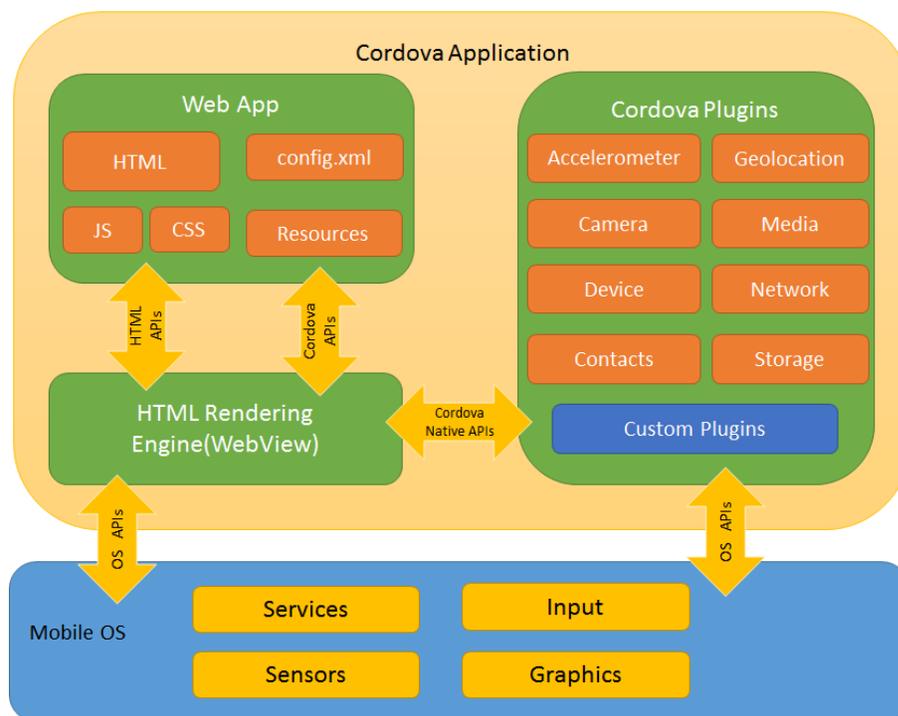


Рисунок 2.5 – Архитектура приложения Apache Cordova

Apache Cordova – своего рода прослойка между пользовательским интерфейсом и ресурсами устройства [18]. Любые виджеты пользовательского интерфейса или структуры MV* (Model – View) не включены в его структуру. Если необходимо использовать виджеты пользовательского интерфейса и / или рамки MV*, необходимо выбрать и включить их в приложение самостоятельно, в качестве сторонних ресурсов. Ionic 3 является одной из структур, которая предоставляет виджеты пользовательского интерфейса. Модель MVC для нее предоставляет Angular 2+.

2.4.4 Инструменты для реализации приложения

В ходе обсуждения с руководителем проекта было решено использовать стек MEAN. Одним из главных причин было, что в Ionic 3 используется Angular 2+, что ускорит скорость разработки. Далее более подробно рассмотрим стек MEAN.

Обычно при веб-разработке на всех уровнях стека используются разные языки программирования. База данных MongoDB, фреймворки Express и AngularJS и серверный язык Node.js вместе образуют стек MEAN — мощную платформу, на всех уровнях которой применяется всего один язык: JavaScript.

Переход от традиционной базы данных, такой как MySQL, к бессхемному, документно-ориентированному NoSQL-хранилищу, такому как MongoDB, представляет собой фундаментальный сдвиг в стратегии персистенции. Разработчик тратит меньше времени на написание операторов SQL и больше — на написание функций map на JavaScript. При этом исключаются огромные пласты логики преобразования, так как MongoDB изначально выдает формат JavaScript Object Notation (JSON). В результате предельно упрощается написание веб-сервисов REST.

Node.js – среда выполнения JavaScript на стороне сервера. Это платформа, встроенная в среду выполнения V8 JavaScript Google Chrome. Это помогает быстро создавать масштабируемые приложения [22].

Express.js – облегченный фреймворк, используемый для создания веб-приложений в Node. Он предоставляет ряд надежных функций для создания одно- и многостраничного веб-приложения. Структура Express заимствован с таких популярных языков как Ruby и Sinatra [23].

MongoDB – без схемная база данных NoSQL. MongoDB сохраняет данные в двоичном формате JSON, что упрощает передачу данных между клиентом и сервером [24]. Рисунок 2.5 представляет модель базы данных для данного приложения.

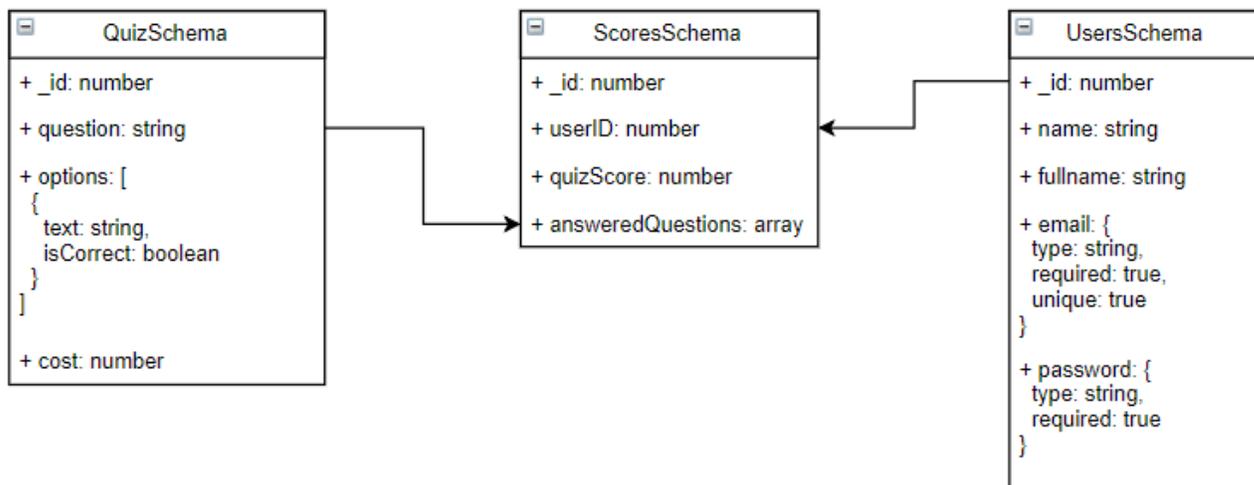


Рисунок 2.6 – Модель устройства базы данных в MongoDB

Angular – JavaScript фреймворк, разработанный Google. Он предоставляет некоторые полезные функции, такие как двусторонняя привязка данных. Это комплексное решение для быстрого развития интерфейса [25].

2.5 Архитектура программного продукта

Выбранная трехуровневая архитектура будет спроектирована при помощи сервис-ориентированного архитектурного стиля. Данная архитектура позволяет создавать распределенное программное обеспечение, состоящего из набора независимых сервисов.

Трехуровневая архитектура будет разработана с использованием сервис-ориентированный архитектурного стиля. Сервис-ориентированная архитектура позволяет создавать распределенное программное обеспечение, состоящий из набора независимых услуг [19, 20].

Эта архитектура имеет несколько преимуществ, и одна из них способность быстро адаптироваться к изменяющимся требованиям и задачам информационного мира.

В настоящий момент наиболее востребованы веб-сервисы SOAP и REST сервисы. Основные различия между подходами SOAP и REST [21]:

- Запросные и ответные пакеты в REST намного меньше размеры, чем соответствующие пакеты SOAP.
- SOAP рассматривает уровень передачи данных HTTP как пассивный наблюдатель и REST, напротив, как активный взаимодействие с использованием существующих методов HTTP, например GET, POST, PUT, DELETE.
- Модель SOAP поддерживает определенную степень интроспекции, позволяющая разработчикам услуг описать свой API в файле WSDL (Language Description Language), который является одной из самых простых задач. В свою очередь, модель REST избегает сложностей WSDL и использует более понятные интерфейс, основанный на стандартных методах HTTP.
- REST ограничивается операциями CRUD с помощью соответствующих методов HTTP. SOAP может содержать почти неограниченное количество методов.

При сравнении достоинств и недостатков двух подходов к внедрению веб-услуг, было показано, что веб-сервисы REST являются наиболее понятными и легко реализуемыми, имеют наименьшие размеры пакетов.

API RESTful также можно кэшировать. Это означает, что клиент имеет возможность хранить все ответы в кэше. В результате API имеет улучшенную производительность.

API RESTful обычно называют веб-службами RESTful, поскольку они реализуют принципы REST, а также протоколы HTTP.

В основном, он охватывает множество ресурсов с помощью HTTP-методов, а затем представляется в стандартном формате, который обычно является XML. Пока гипертекст является стандартным, он работает для допустимых типов интернет-медиа. Некоторые примеры представлены на рисунке 2.6.

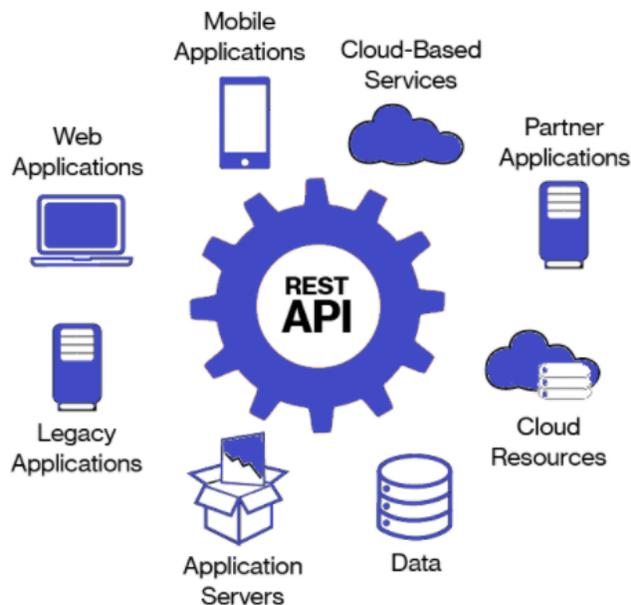


Рисунок 2.7 – Визуальное представление REST

REST использует 4 основных метода HTTP: GET, POST, PUT, DELETE. В большинстве случаев каждый из методов служит для выполнения определенного действия из CRUD (Create, Read, Update, Delete).

POST - создание, GET - чтение, PUT - обновление, DELETE - удаление. Пример реализации веб-сервиса REST графически показан на рисунке 2.7.

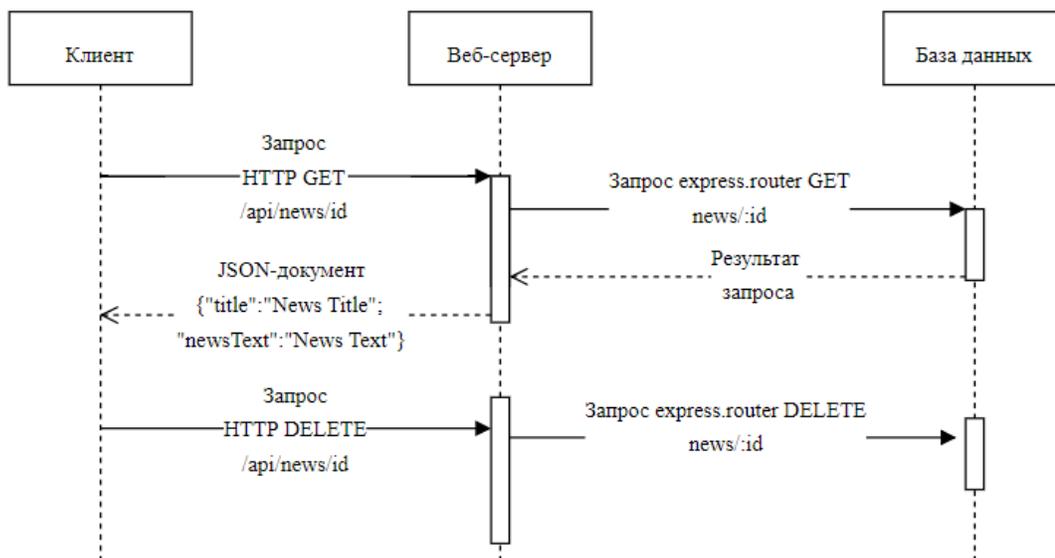


Рисунок 2.8 – REST веб-сервис

На рисунке 2.8 показана начальная форма, REST API, построенный с помощью Express и Node.js, чтобы обеспечить взаимодействие с базой данных MongoDB.

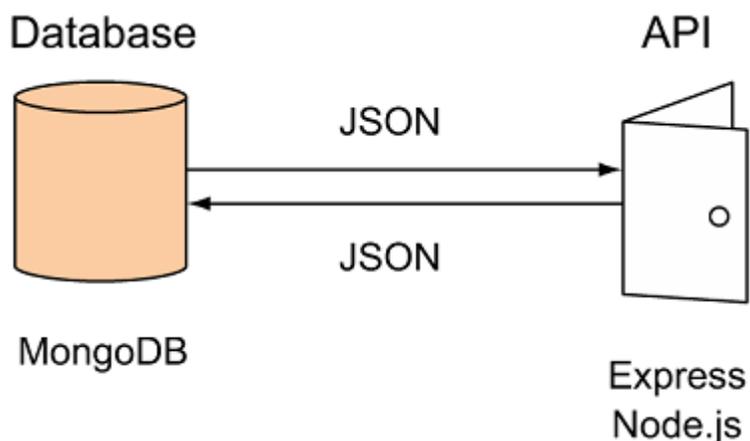


Рисунок 2.9 - Стандартный MEAN REST API, используя MongoDB, Express и Node.js

Данный программный продукт включает две части. Это ориентированная на пользователя интерфейс, выдающая пользователям новости, мероприятия и многое другое. У системы управления контентом мобильного приложения имеется также интерфейс администратора, в который администраторы входят для обновления содержимого и управления приложением.

В данном случае будет рассмотрена архитектура двух приложений по отдельности:

- Ориентированный на пользователя интерфейс,
- Интерактивный не общедоступный административный интерфейс.

2.5.1 Интерфейс административной панели: SPA на AngularJS

Для административного интерфейса идеально подходило бы SPA, построенное на основе AngularJS [26]. Архитектура этой части программного продукта будет выглядеть: API REST, построенный на MongoDB, Express и Node.js, с одностраничным приложением на AngularJS в интерфейсной части [27]. Рисунок 2.9 демонстрирует, как это будет выглядеть.

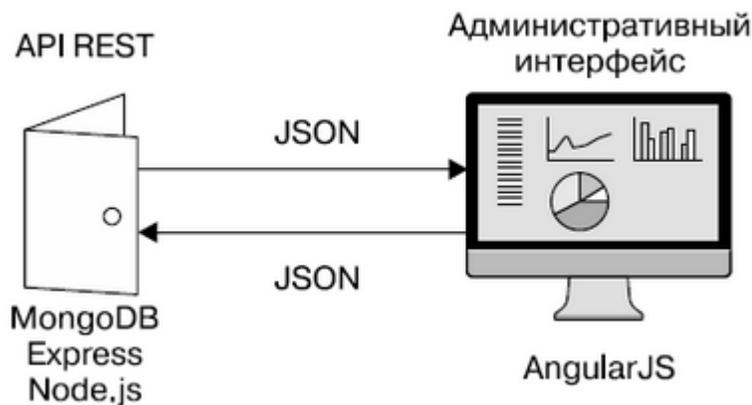


Рисунок 2.10 – Административный интерфейс представляет собой SPA на основе AngularJS, использующее REST API, построенное на MongoDB, Express и NodeJS

2.5.2 Интерфейс мобильного приложения: Фреймворк Express

В случае с мобильным приложением доставка HTML – кода и контента непосредственно с сервера – именно то, что будет необходимо. Express с самого начала обеспечивает выбор шаблонизаторов. HTML-содержимое потребует данные из БД, используем для этого API REST. Основу такой архитектуры демонстрирует рисунок 2.10.

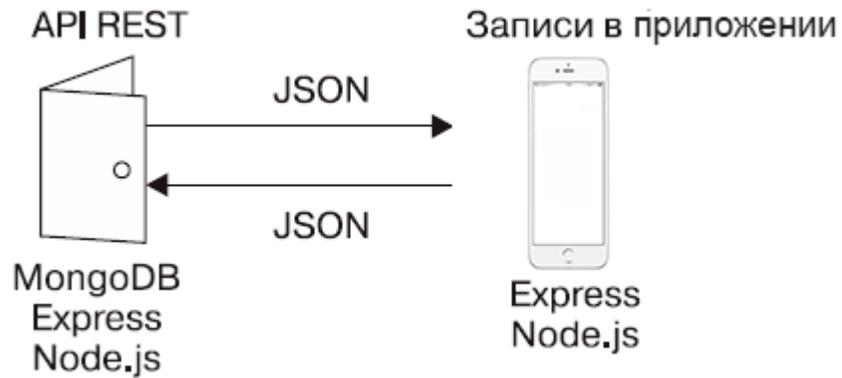


Рисунок 2.11 – Архитектура доставки HTML – кода непосредственно с сервера

Также к данной архитектуре можно добавить AngularJS, чтобы добавить в приложение динамики, к примеру чтобы отображать связанные сообщения или поисковое поле с авто дополнением.

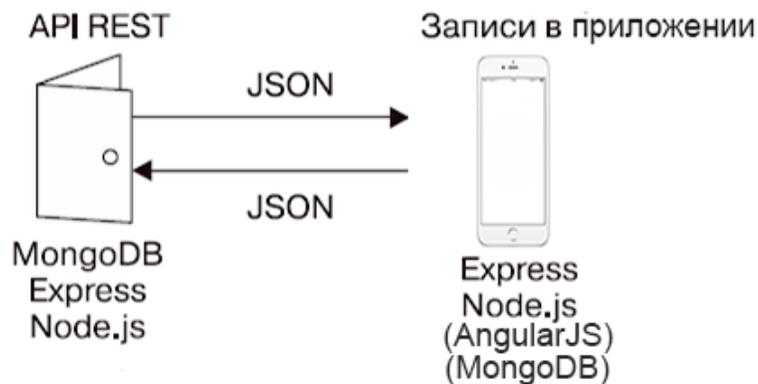


Рисунок 2.12 – Добавление параметров использования AngularJS и MongoDB как части интерфейсного аспекта

2.5.3 Гибридная архитектура программного продукта

В итоге было получено два отдельно работающих приложения, использующих REST API. Целостность архитектуры с одним API REST, взаимодействующим с двумя приложениями клиентской части, демонстрирует рисунок 2.12.

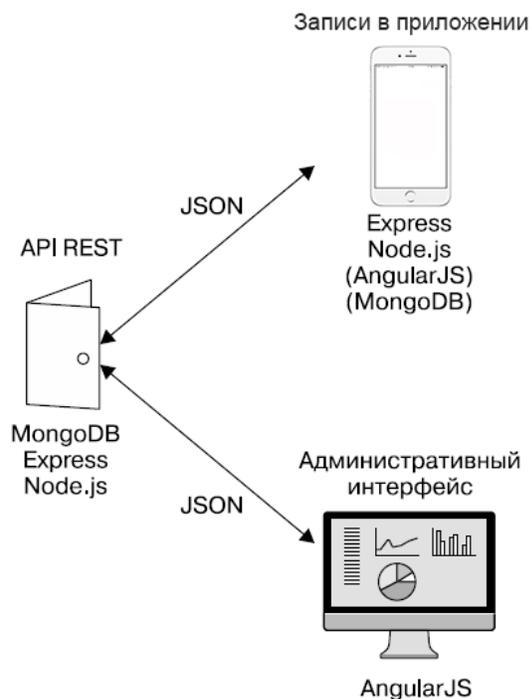


Рисунок 2.13 – Гибридная архитектура набора приложений MEAN

Как показано на рисунке 2.12 ради обеспечения наиболее подходящего решения один REST API поставляет данные двум ориентированным на взаимодействие с пользователями приложениям, основанным на различных частях набора технологий MEAN.

2.5.4 Аутентификация пользователей

Для авторизации, регистрации и защиты маршрутов была использована технология Json Web Token (JWT).

JSON Web Token (JWT) является открытым стандартом (RFC 7519), который определяет компактный и автономный способ для безопасной передачи информации между сторонами как JSON объект. Эта информация может быть проверена, поскольку она имеет цифровую подпись. JWT могут быть подписаны с использованием секретного (с алгоритмом HMAC) или пары открытого / закрытого ключей с использованием RSA.

Далее рассмотрим принцип действия JWT. Всякий раз, когда пользователь хочет получить доступ к защищенному маршруту или ресурсу, пользовательский агент должен отправить JWT, как правило, в заголовок авторизации, используя схему Bearer. Содержимое заголовка должно выглядеть следующим образом:

```
Authorization: bearer <token>
```

Это механизм аутентификации без учета состояния, поскольку пользовательское состояние никогда не сохраняется в памяти сервера. Защищенные маршруты сервера будут проверять правильность JWT в заголовке авторизации и, если он присутствует, пользователю будет разрешен доступ к защищенным ресурсам. Поскольку JWT являются автономными, существует вся необходимая информация, что уменьшает необходимость в запросе базы данных несколько раз.

Это позволяет полностью полагаться на API данных, которые не имеют апатрида и даже делают запросы к нисходящим службам.

Этот процесс графически показан на рисунке 2.13.

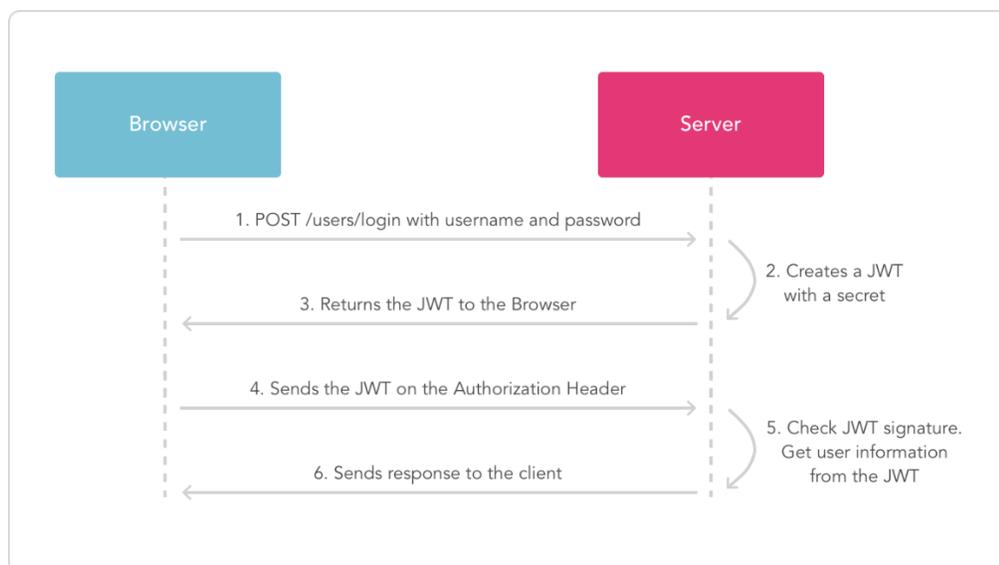


Рисунок 2.14 – Процесс работы JWT

2.5.5 Защита передаваемых данных

Для устранения возможности перехвата данных сторонними приложениями, при передаче данных от сервера клиенту по протоколам HTTP нужно обеспечить защиту передаваемых данных.

Для защиты передаваемых данных организуется HTTPS запросы, также идентификация и авторизация клиента на сервере. Для этого при выполнении запроса передаётся токен. JSON Web Token состоит из трех частей разделенных точками:

- HEADER
- PAYLOAD
- SIGNATURE

Вывод состоит из трех строк Base64-URL, разделенных точками, которые можно легко передать в среде HTML и HTTP, будучи более компактными по сравнению с стандартами на основе XML, такими как SAML.

На рисунке 2.15 с помощью диаграммы продемонстрирована идентификация клиент на сервере.

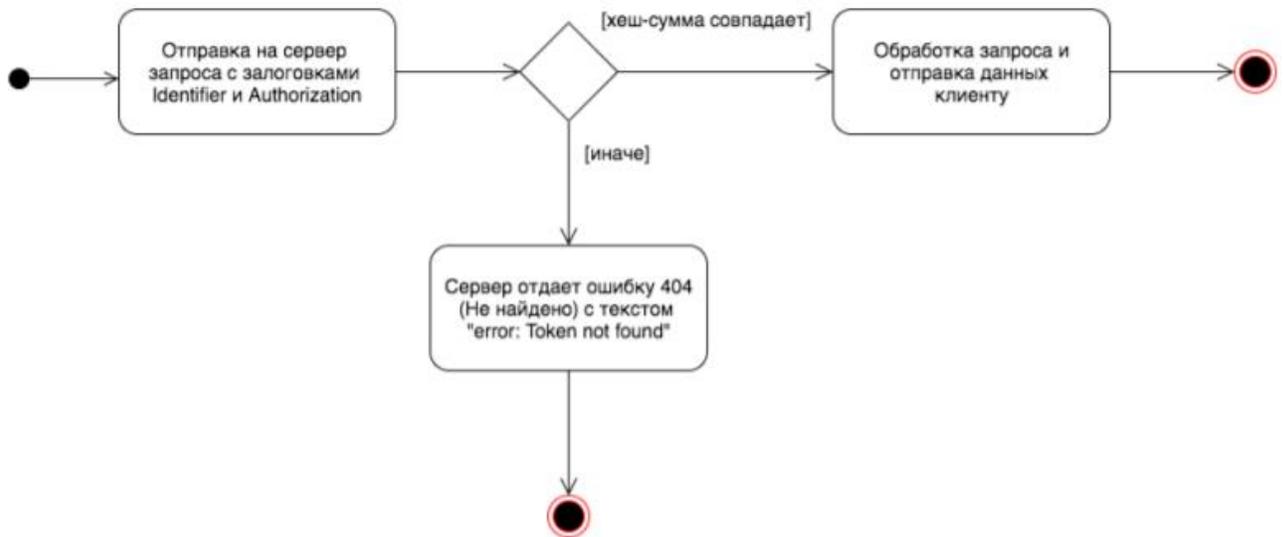


Рисунок 2.15 – UML диаграмма деятельности идентификации клиента на сервере

3 Разработка программного продукта

В данной главе будет рассмотрен весь процесс разработки программного продукта и выявлены возникшие проблемы и методы их решения.

3.1 Серверная сторона программного продукта

В процессе разработки серверной стороны программного продукта использовался язык JavaScript в среде NodeJS. В качестве платформы определяющую структуру программной системы использовался легковесный фреймворк Express. В качестве хранилища данных послужила документно-ориентированная база данных (БД) MongoDB.

3.1.1 Создание необходимых коллекций в MongoDB

Для создания коллекции необходимо установить MongoDB. Вся информация об установке MongoDB на Linux Elementary были взяты с официальной документации по установке.

После того как, MongoDB был установлен, можно приступить к созданию коллекций для дальнейшего хранения данных.

Листинг 3.1 – Модель базы данных news.js

```
const mongoose = require('mongoose');

const NewsSchema = new mongoose.Schema({
  id: String,
  title: String,
  newsPreview: String,
  newsText: String,
  publishedDate: { type: Date, default: Date.now }
});

module.exports = mongoose.model('News', NewsSchema);
```

На основе модели, показанного выше листинга можно создавать другие необходимые модели.

3.1.2 Создание HTTP-сервера с возможностью подключения к БД

Был создан простой MEAN проект и установлены нужные библиотеки. В результате получился каркас сервера. Настройка проекта происходит в файле `package.json`. После редактирования файла настроек проекта NodeJS нужно выполнить команду `npm install`. В данном случае файл настроек сервера выглядит как показано ниже:

```
{
  "name": "server",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "dependencies": {
    "bcrypt": "^1.0.3",
    "body-parser": "^1.18.2",
    "express": "^4.16.3",
    "jsonwebtoken": "^8.2.1",
    "mongodb": "^3.0.6",
    "mongoose": "^5.0.14",
    "morgan": "^1.9.0",
    "multer": "^1.3.0"
  },
  "devDependencies": {},
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "node server.js"
  },
  "author": "Galymzhan Almabek",
  "license": "ISC"
}
```

После того как все необходимые библиотеки были установлены и зарегистрированы в файле `package.json`, можно приступать к запуску сервера, листинг код запуска сервера указан ниже:

```

const port = process.env.PORT || '3000';
app.set('port', port);

const server = http.createServer(app);

server.listen(port, () => console.log(`Сервер запущен на localhost:${port}`));

```

В результате после запуска сервера командой «nodemon» стало возможным обратиться из браузера к HTTP-серверу, который прослушивает порт 3000.

3.1.3 Реализация проверки авторизации на сервере

Для доступа пользователей к своим аккаунтам нужно было разработать системы авторизации и аутентификации, для этого в NodeJS есть поддерживаемые сообществом разработчиков библиотека: jsonwebtoken. После подключения этой библиотеки в package.json в server.js была добавлена логика по работе с пользователями и описана схема документов в БД.

Листинг 3.2 – Логика для работы с пользователями

```

const jwt = require('jsonwebtoken');

module.exports = ( req, res, next ) => {
  try{
    const token = req.headers.authorization.split(" ")[1];
    const decoded = jwt.verify(token, "secret");
    req.userData = decoded;
    next();
  } catch (error) {
    return res.status(401).json({
      message: "Auth failed"
    });
  }
}

```

В данной реализации метод «verify» декодирует токен с полученный с запроса пользователя с помощью секретного ключа в данном случае это «secret». Далее сравнивает его с хранящимся в коллекции user токеном и отдает ответ 200 при успешном сравнении, в противном случае отдает ответ 401 не авторизован.

3.1.4 Определение основных маршрутов REST API

Для отправки ответов на запросы клиентов в NodeJS есть возможность описать маршрутизацию запросов при помощи определения соответствия пути, по которому клиент отправляет запрос и функции обработчика. Для этого в код главного модуля сервера было добавлено описание реализации маршрутов для запросов авторизации, веб-приложения и синхронизации мобильного приложения.

В папке routes хранятся файлы в которых реализованы маршруты ответа приложения на клиентские запросы к конкретному URL. Каждый маршрут может иметь одну или несколько функций обработки, которые выполняются при сопоставлении маршрута. Определение маршрута имеет следующую структуру:

```
app.METHOD(PATH, HANDLER)
```

где:

- app – это экземпляр express.
- METHOD – метод запроса HTTP.
- PATH – путь на сервере.
- HANDLER – функция, выполняемая при сопоставлении маршрута.

Ниже показан пример реализации маршрутов:

```
const Student = require("../models/students.js");
const News = require("../models/news.js");
const Events = require("../models/events.js");
const MenuItems = require("../models/menuItems.js");
const Pages = require("../models/pages.js");
const Users = require("../models/users.js");
const Quizes = require("../models/quizes.js");
const Scores = require('../models/scores');
router.post("/events", checkAuth, (req, res, next) => { });
router.post("/news", checkAuth, (req, res, next) => { });
router.post("/students", checkAuth, (req, res, next) => { });
router.post("/menus", checkAuth, (req, res, next) => { });
```

В соответствие с каждым маршрутом сопоставлена модель, в котором описана реализация. В вышеуказанном REST API показан не весь код в целях

безопасности, также в нем реализованы все операции CRUD (Create, Read, Update, Delete). Уникальностью данного подхода разработки является то, что описанный выше API будет использоваться дважды: административная панель и мобильное приложение.

3.1.5 Тестирование REST API с использованием инструмента Postman

Все описанные операции CRUD были изначально протестированы в инструменте для отладки HTTP запросов Postman (Рисунок 3.1).

Postman – удобный HTTP-клиент для тестирования веб-сайтов, входит в число лучших расширений каталога Chrome Web Store в категории «инструменты для работы» (productivity tools). С помощью расширения можно составлять и редактировать простые или сложные HTTP-запросы

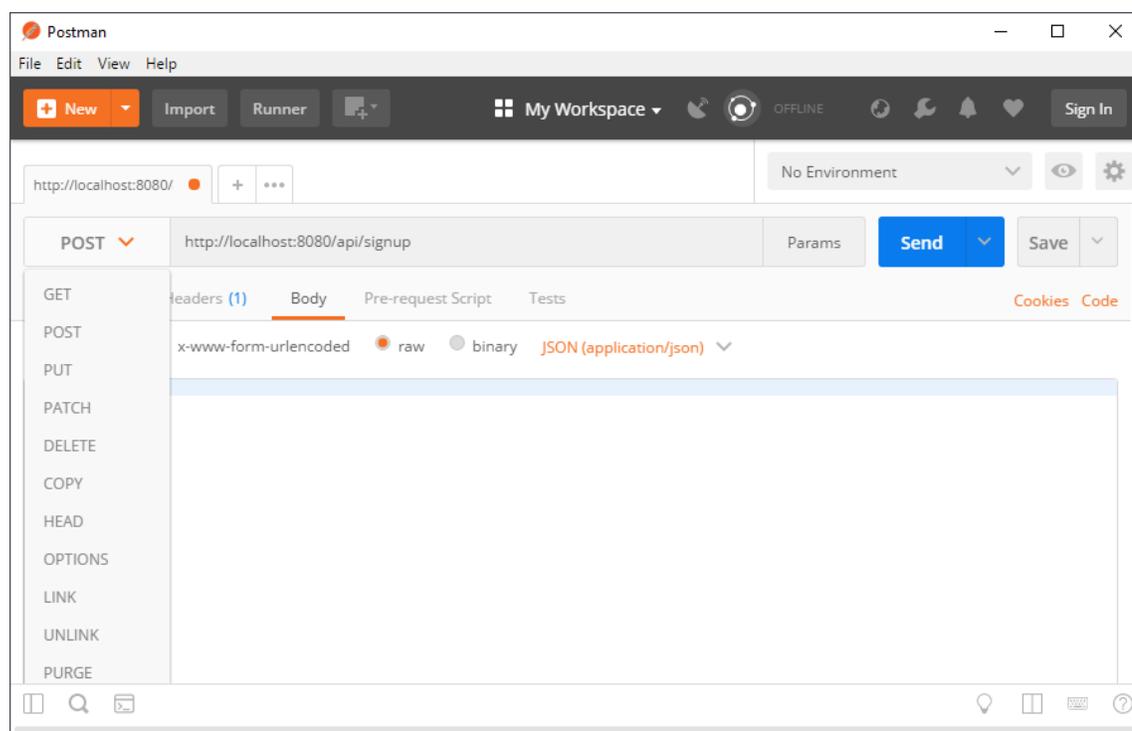


Рисунок 3.1 – Интерфейс инструмента отладки Postman

3.2 Административная панель

Данный модуль проектируется и реализуется по шаблону MVVM (Model – View – ViewModel). С помощью шаблона MVVM можно разделить модель приложения и ее представление для изменения их отдельно друг от друга. С использованием данного шаблона элементы представления связываются со свойствами и событиями представления – модели. В итоге каждый слой MVVM не знает о существовании другого.

Для реализации данного приложения был выбран подход SPA(Single Page Application). В традиционных веб-приложениях клиент (браузер) инициирует связь с сервером, запрашивая страницу. Затем сервер обрабатывает запрос и отправляет HTML-код клиенту. В последующих взаимодействиях со страницей, например, пользователь переходит к ссылке или отправляет форму с данными, на сервер отправляется новый запрос, и поток начинается снова. Сервер обрабатывает запрос и отправляет новую страницу в браузер в ответ на новое действие, запрошенное клиентом. [29]

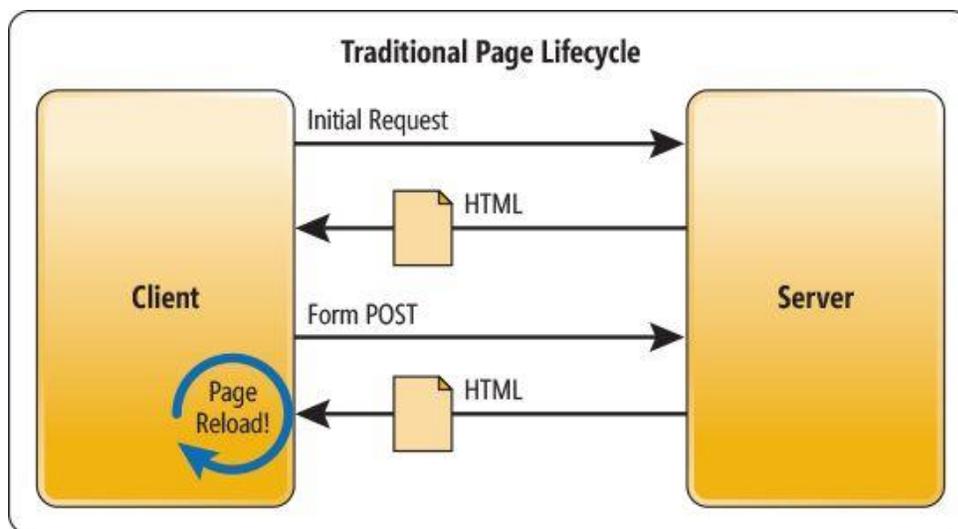


Рисунок 3.2 – Жизненный цикл обычных страниц

В одностраничных приложениях (SPA) вся страница загружается в браузер после первоначального запроса, но последующие взаимодействия происходят с использованием запросов Ajax. Это означает, что браузер должен обновлять только часть измененной страницы; нет необходимости перезагружать всю страницу. SPA-подход сокращает время, затрачиваемое приложением на реагирование на действия пользователя, что приводит к более интенсивному воздействию с приложением

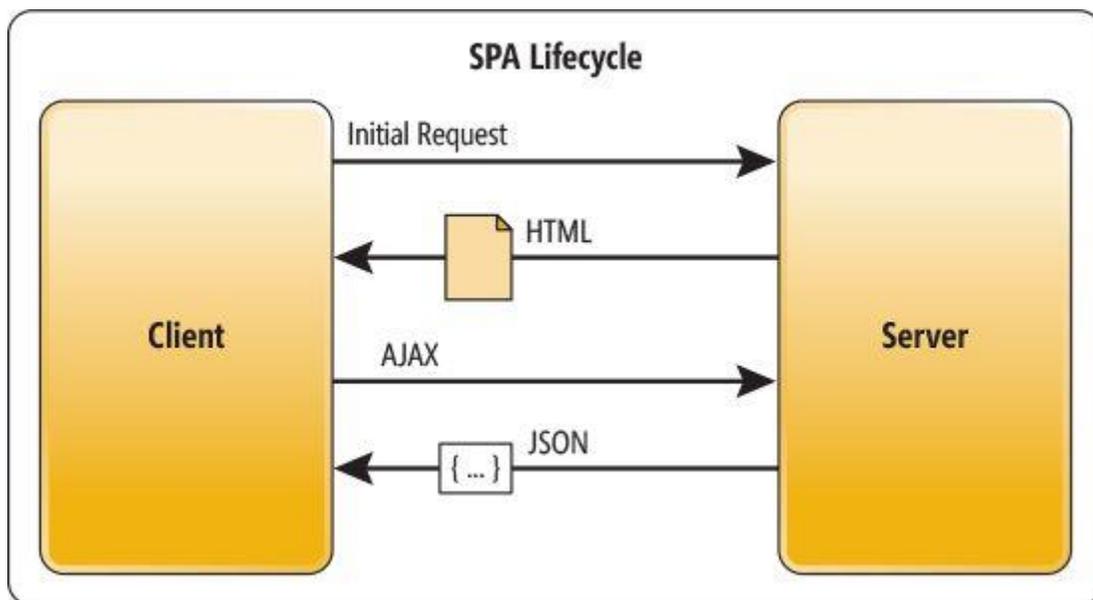


Рисунок 3.3 – Жизненный цикл SPA

Визуальное представление зависимостей, которые использует административная панель представлено на UML диаграмме пакетов, рисунок 3.4.

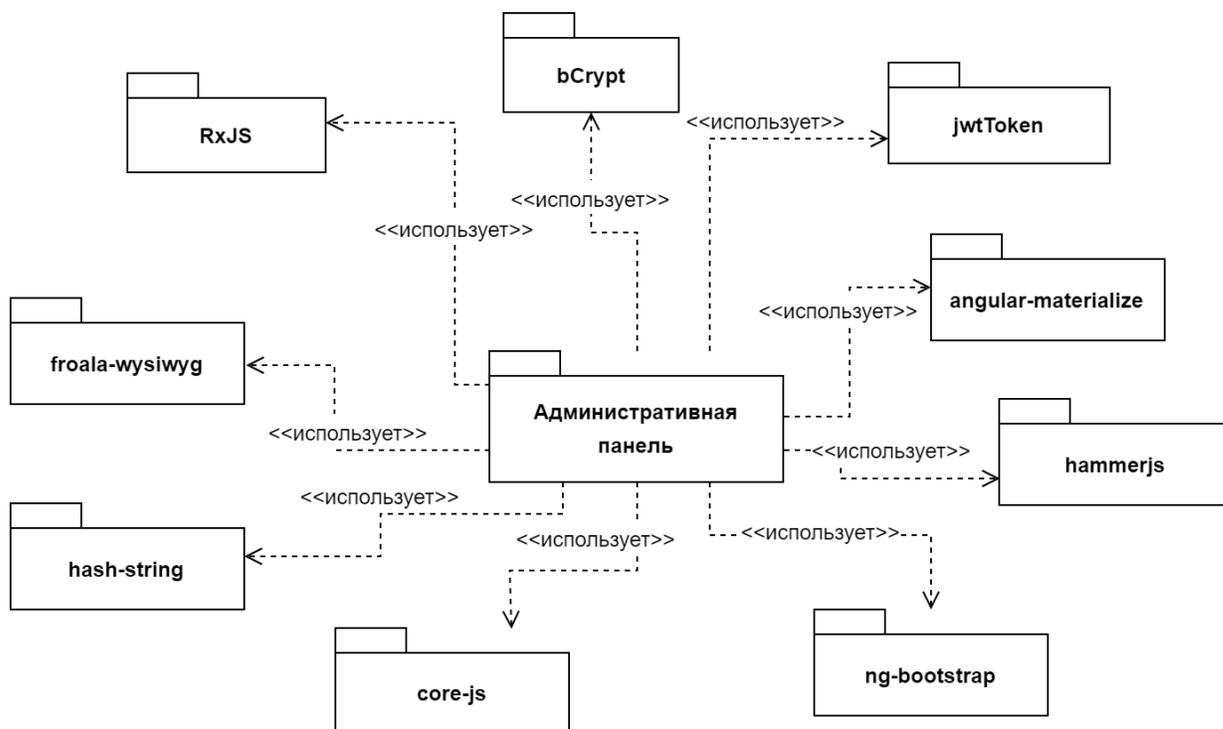


Рисунок 3.4 – UML детализированная диаграмма пакетов

3.2.1 Верстка пользовательского интерфейса в соответствии с макетом

Для создания шаблона административной панели было решено использовать библиотеку Angular Material. Вся верстка административной панели соответствует предоставленному макету.

Первое что увидит при входе в административную панель администратор приложения будет страницы авторизации. Страница авторизации графически показан на рисунке 3.4.

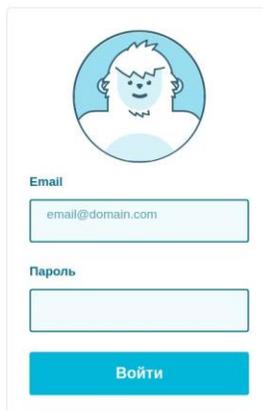
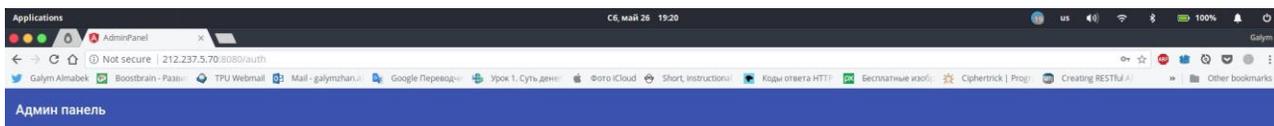


Рисунок 3.5 – Окно авторизации административной панели

У администратора приложения будут неограниченные права на создание, чтение, редактирование и удаление данных.

Так как Angular Material является готовыми компонентами, для их использования необходимо для начала импортировать необходимые компоненты в файл `app.module`. Импорт для `material` кнопок, форм, карт и иконок показан ниже:

```
import {MatButtonModule,  
        MatFormFieldModule,  
        MatCardModule,  
        MatIconModule} from 'angular/material';
```

После успешной авторизации администратору будет открыт доступ к главному окну приложения. Где он может просматривать, редактировать и удалять данные. Основные окна приложения показаны в приложении А.

3.2.2 Разработка компонентой структуры веб-приложения

Для создания компонентов использовался инструмент Angular 5 – CLI (Command Line Interface). Пример использования Angular CLI:

```
ng generate component "name of module" "component name"
```

где,

module – название модуля к которому принадлежит компонент

После создания необходимых компонентов, необходимо реализовать запрос к API сервера. Придерживаясь компонентного подхода, создаем запрос в сервисе, а в компоненте обращаемся к тому же сервису.

Реализация функционала «добавления новостей» с указанными выше шагами:

файл «add-news.component.ts»:

```
saveNews(title, newsPreview) {  
    this.dataservice.saveNews(hash(title), title, this.previewImage, newsPreview,  
this.froalaContent);  
    this.http.get("/api/news").subscribe(data => {  
        this.news = data;  
        setTimeout(() => {  
            this.router.navigate(['/news']);  
        }, 3000);  
    });  
}
```

файл «data-service.ts»

```
saveNews(id, title, imagePreview, newsPreview, newsText) {  
    const obj = {  
        id: id,  
        title: title,  
        imagePreview: imagePreview,  
        newsPreview : newsPreview,  
        newsText: newsText  
    };  
    this.http.post("/api/news", obj)
```

```

        .subscribe(res => console.log("done"));
    }

```

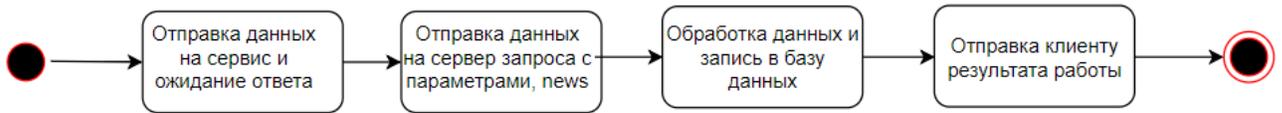


Рисунок 3.6 – Выполнение запроса на запись новости

Файл «add-news.component.ts» представлен в приложении Б.

3.3 Мобильное приложение

Структура приложения на Ionic 3 с поддержкой Angular 2+ и Typescript, графически представлен на рисунке 3.7.

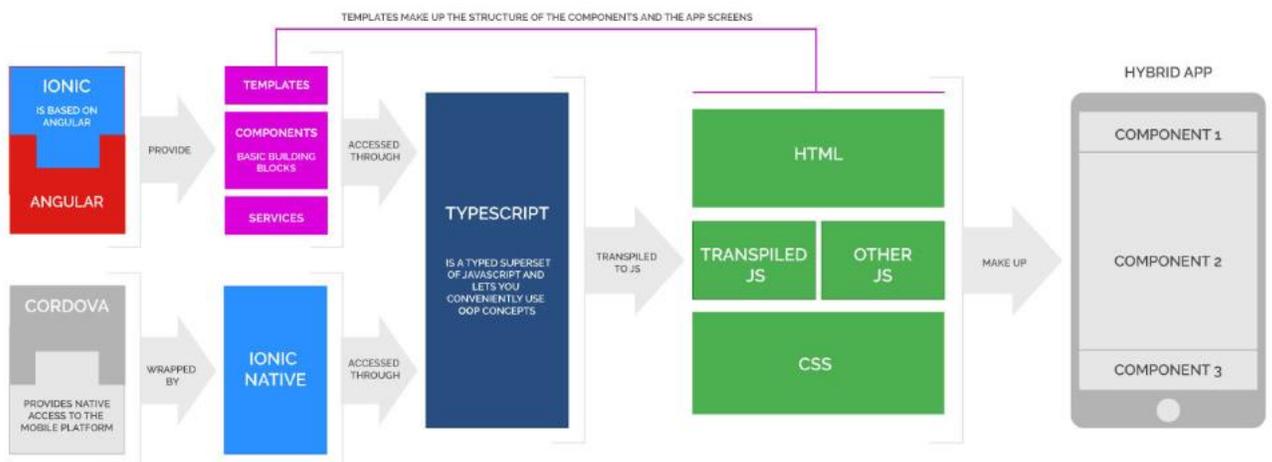


Рисунок 3.7 – Структура гибридного Ionic приложения

Визуальное представление зависимостей, которые использует мобильное приложение представлено на UML диаграмме пакетов, рисунок 3.8.

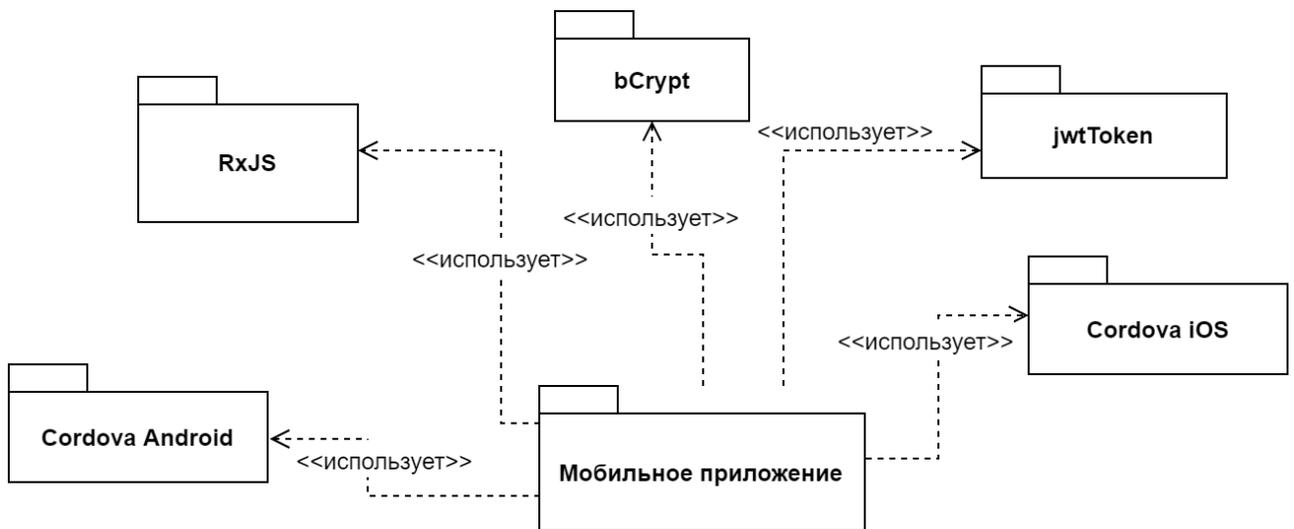


Рисунок 3.8 – UML детализированная диаграмма пакетов

3.3.1 Верстка пользовательского интерфейса в соответствии с макетом

Интерфейс мобильного приложения реализовывался при помощи готовых компонентов, которые предоставляются документацией Ionic 3 [30]. Пример использования готового компонента списка показан ниже:

```

<form (ngSubmit)="login()" #registerForm="ngForm">
  <ion-list>
    <ion-item>
      <ion-label floating>Email</ion-label>
      <ion-input type="text" name="email" [(ngModel)]="registerCredentials.email" required></ion-input>
    </ion-item>
    <ion-item>
      <ion-label floating>Пароль</ion-label>
      <ion-input type="password" name="password" [(ngModel)]="registerCredentials.password" required></ion-input>
    </ion-item>
  </ion-list>
  <button ion-button class="primary" block type="submit" [disabled]="!registerForm.form.valid">Логин</button>
  <button ion-button block clear (click)="createAccount()">Создать аккаунт</button>
</form>
  
```

В Ionic 3 при разработке у программиста есть возможность выбрать препроцессор CSS или использовать его без препроцессоров.

3.3.2 Разработка компонентой структуры мобильного приложения

Так же, как и в Angular 5, у Ionic 3 есть мощный CLI (Command Line Interface) с помощью которого были созданы страницы, провайдеры и необходимые модули. Пример использования CLI показан ниже:

```
$ ionic generate [<type>] [<name>]  
ionic generate page Login
```

Каждое Angular – приложение имеет как минимум один компонент, корневой компонент, который соединяет иерархию компонентов со страницей DOM. Каждый компонент определяет класс, который содержит данные приложения и логику, и связан с HTML-шаблоном, который определяет представление, отображаемое в целевой среде.

Ниже представлена иерархия взаимодействия компонентов (Рисунок 3.9):

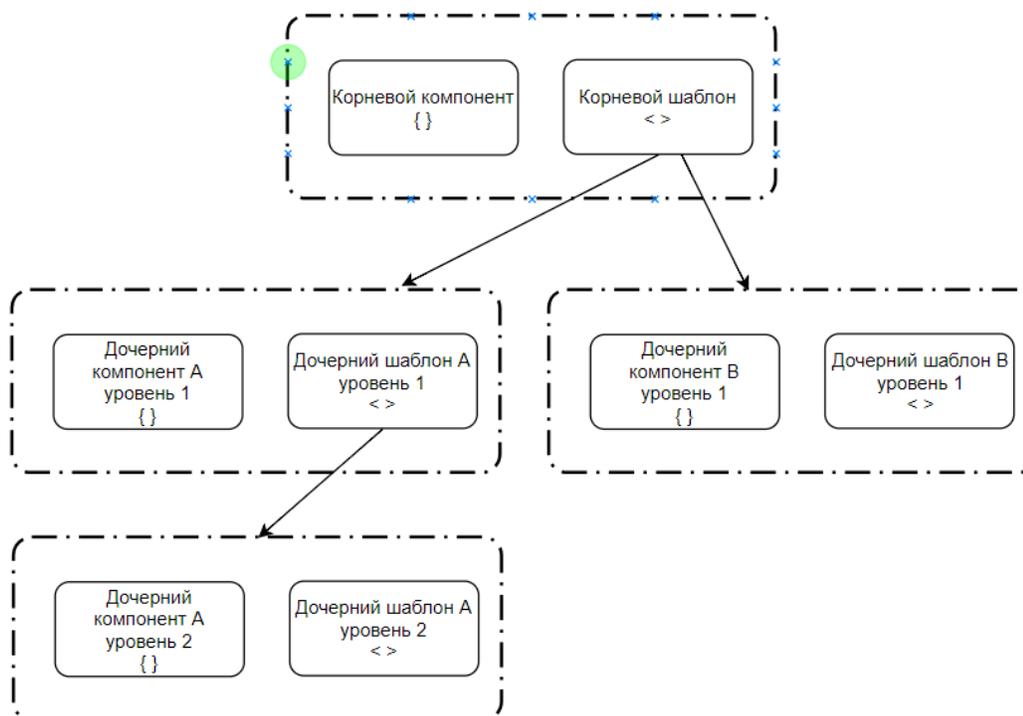


Рисунок 3.9 – Иерархия взаимодействия представлений

Структура каждого компонента состоит из импорта необходимых модулей, объявлений декораторов, интерфейсов и класса, в котором хранятся вся логика компонента. Пример реализации компонента показан в Листинге В.

3.3.3 Интеграция мобильного приложения с API

Так же, как и в приложении для административной панели обращение к серверу идет через провайдеры. Чтобы определить класс как провайдер в Angular, необходимо использовать декоратор `@Injectable`, чтобы предоставить метаданные, которые позволяют Angular внедрить его в компонент как зависимость.

Процесс введения Auth-Service в компонент графически показан на рисунке 3.10.

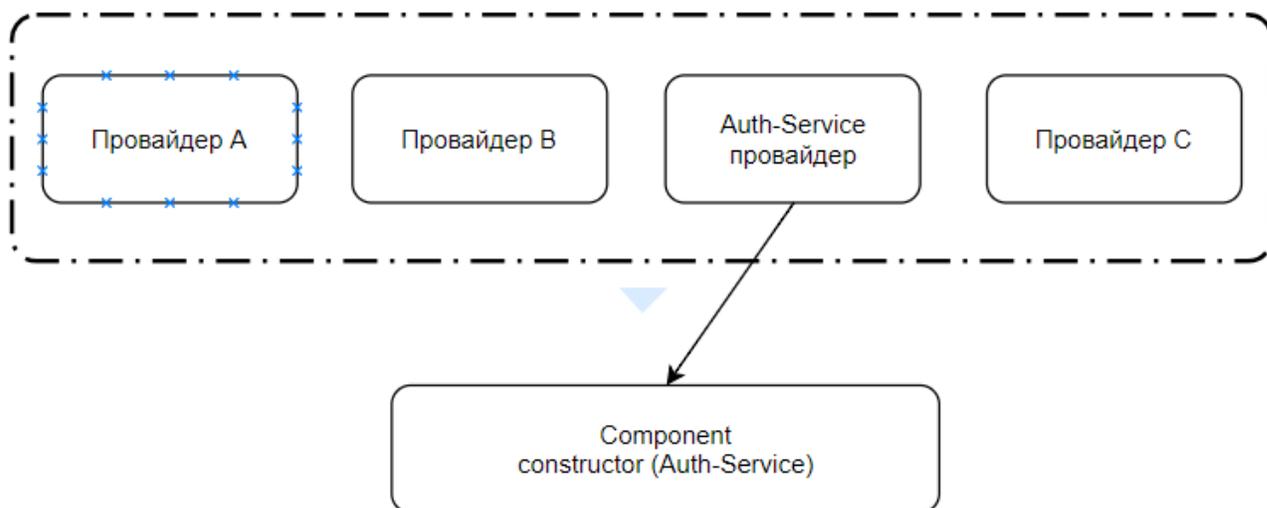


Рисунок 3.10 – Внедрение Auth-Service провайдера в компонент

Запрос на сервер с приложения по паттерну MVVM отправляется через провайдеры, пример отправки запроса на регистрацию нового пользователя показан ниже, полное содержимое файла показано в листинге Г.

```
public register(credentials) {
```

```

    if (credentials.email.toLowerCase().trim() === null ||
credentials.password.trim() === null) {
    return Observable.throw("Пожалуйста введите данные");
    } else {
    return Observable.create(observer => {
    this.http.post(`${this.rootUrl}/signup`, {
    name: credentials.name.toLowerCase().trim(), fullname:
credentials.fullname.toLowerCase().trim(),
    email: credentials.email.toLowerCase().trim(), password:
credentials.password.trim(),
    })
    .map(res => res.json())
    .subscribe( data => {
    console.log(data);
    });
    observer.next(true);
    observer.complete();
    });
    }
}

```

3.3.4 Сборка мобильного приложения под определенные платформы

В Ionic 3 существует четыре способа тестирования приложения при разработке:

1. браузер WebKit;

```
$ ionic serve
$ ionic --lab
```
2. симулятор iOS или Android;

```
$ ionic cordova build ios
$ ionic cordova emulate ios
```
3. мобильный браузер;

```
$ ionic cordova prepare ios
```
4. приложение в телефоне.

```
$ ionic cordova run android
```

В данном случае использовался первый подход весь процесс разработки на Ionic 3 происходил в браузере с помощью ionic CLI (Command Line Interface), запускалась команда:

```
$ ionic -lab
```

Результат выполнения данной команды графически показан на рисунке 3.11.

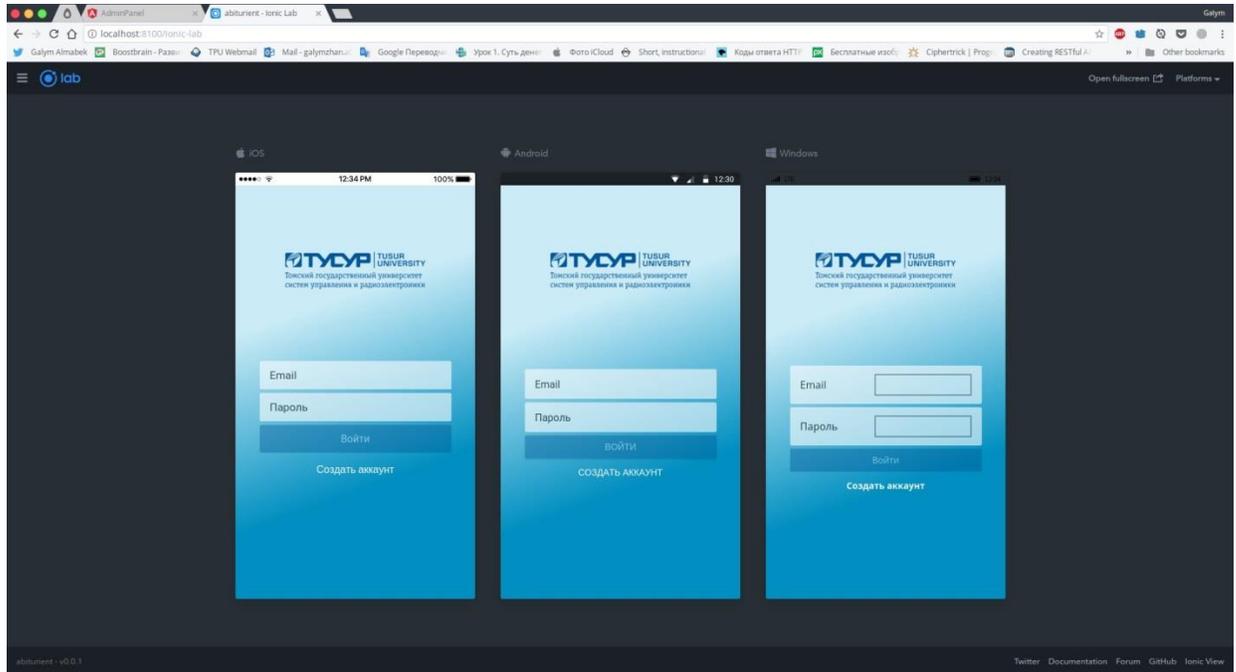


Рисунок 3.11 – Окно авторизации приложения в браузере на трех платформах

Для того чтобы, протестировать приложение на мобильном устройстве, необходимо собрать арк файл. Для этого с помощью CLI запускаем команду

```
$ ionic cordova build android
```

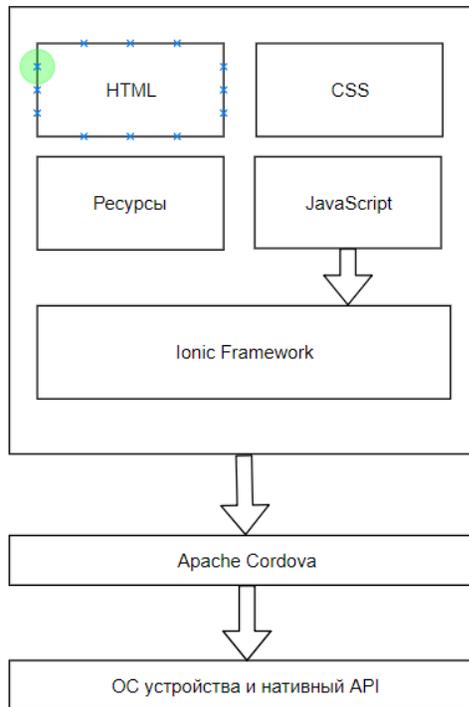


Рисунок 3.12 – Процесс сборки приложения на Ionic 3

4 Финансовый менеджмент, ресурсоэффективности и ресурсосбережение

В настоящее время для университетов важно сохранения и увеличение количества ежегодно поступающих абитуриентов. В свою очередь для абитуриента в процессе выбора университета важно иметь полное представление о процессе подачи документов, списке направлений и шансах на поступление. Данная ВКР решает все эти проблемы и делает процесс поступления удобным, путем разработки мобильного приложения «Абитуриент ТУСУР».

В данном разделе осуществляется оценка коммерческого потенциала, перспективности и альтернатив проведения разработки с позиции ресурсоэффективности и ресурсосбережения, а также планирование и формирование бюджета.

Работа заключалась в автоматизации процесса поступления абитуриента, а именно в создании мобильного приложения для мониторинга общего списка поступающих по определенному направлению, разработка велась с использованием гибридных технологий, также в процессе разработана административная панель.

4.1 Предпроектный анализ

4.1.1 Потенциальные потребители результатов проекта

Целевым рынком данной разработки являются абитуриенты, которые находятся в режиме ожидания после сдачи экзаменов на поступление, приложение вовремя уведомит об изменениях в рейтинг листе, также приложение будет полезно для тех, кто хочет оценить свои шансы на поступление с помощью функции «Калькулятор».

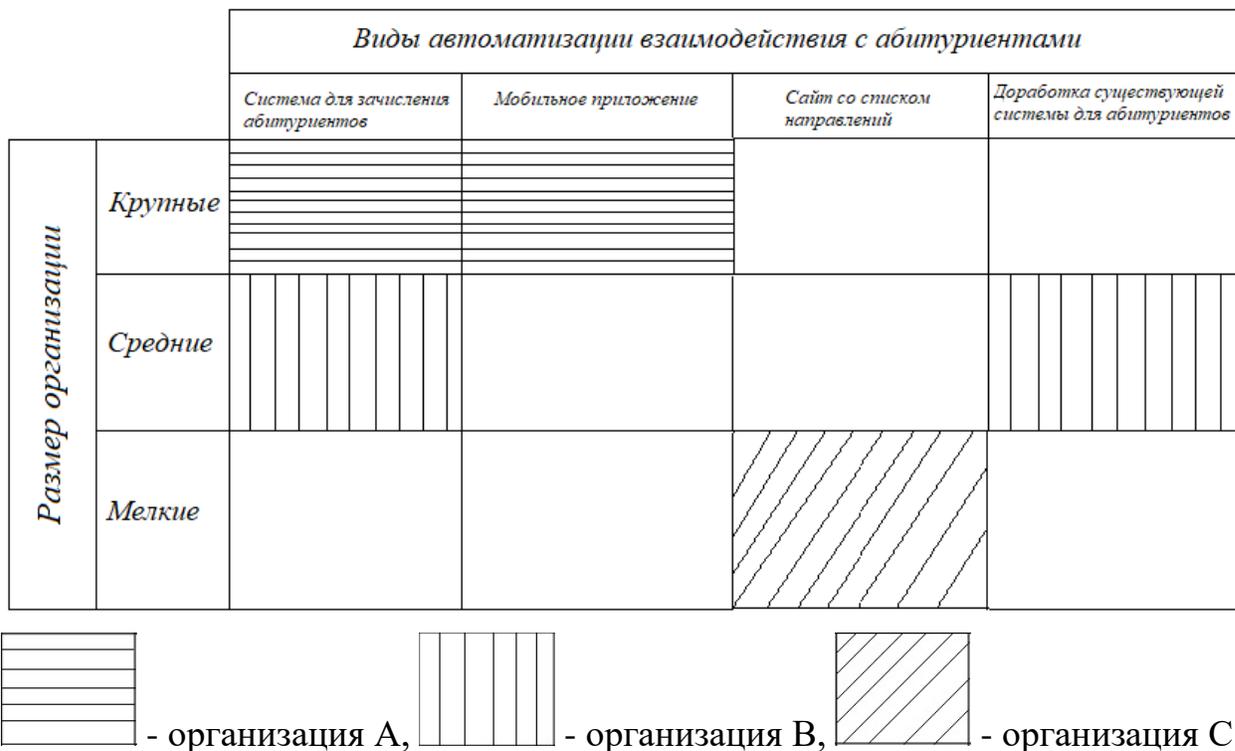


Рисунок 4.1 – Карта сегментирования рынка разработки

4.1.2 Анализ конкурентных технических решений

Разработанное решение является уникальным в своем роде, так как является гибридным, в следствии чего можно мобильное приложения безболезненно перевести в веб приложение, что является выгодным по всем, трём параметрам: цена, время и качество. Поэтому в качестве конкурентов целесообразно рассматривать именно гибридные мобильные приложения.

Таблица 4.1– оценочная карта сравнения конкурентных решений

№	Критерии оценки	Вес критерия	Баллы			Конкурентоспособность		
			Бф	Бк1	Бк2	Кф	К1	К2
	1	2	3	4	5	7	8	9
Технические критерии оценки ресурсоэффективности								
1	Повышение производительности	0,1	9	7	9	0,9	0,7	0,9

	ти труда пользователя							
--	--------------------------	--	--	--	--	--	--	--

Продолжение таблицы 4.1

2	Удобство в эксплуатации (соответствует требованиям потребителей)	0,15	10	10	8	1,5	1,5	1,2
3	Функциональная мощность	0,13	7	8	9	0,91	1,04	1,17
4	Простота эксплуатации	0,09	10	7	5	0,9	0,63	0,45
5	Качество интеллектуального интерфейса	0,15	10	10	7	1,5	1,5	1,05
Экономические критерии оценки эффективности								
1	Конкурентоспособность продукта	0,07	7	7	8	0,49	0,49	0,56
2	Уровень проникновения на рынок	0,05	5	10	10	0,25	0,5	0,5
3	Цена	0,1	10	6	6	1	0,6	0,6
4	Послепродажное обслуживание	0,03	10	5	5	0,3	0,15	0,15
5	Финансирование научной разработки	0,04	9	8	8	0,36	0,32	0,32
6	Срок выхода на рынок	0,09	10	5	5	0,9	0,45	0,45
	Итого	1				9,01	7,88	7,35

Исходя из проведенного анализа можно заключить, что уязвимость конкурентных технологических решений связана, прежде всего с высокими затратами на разработку решения, то есть при разработке приложения расходовалась большая сумма. Реализации двух конкурентов очень схожи, «ТПУ Абитуриент» и «Хочу в ТГУ», поэтому коэффициенты их конкурентоспособности едва различимы. Данные приложений обладают практически единственным более значительным достоинством – это уровень их проникновения на рынок, за счёт раннего выхода на рынок они располагают широкой популярностью среди студентов из разных регионов страны.

Наиболее сильным конкурентом можно считать стандартный функционал, приложения «Хочу в ТГУ». Его основным достоинством является количество

предоставляемой информации в виде разделов, однако, как и «ТПУ Абитуриент», данное приложение не удовлетворяет требованиям автономности приложения.

Преимуществом собственной разработки помимо того, что, она в несколько раз сокращает время выполнения процесса, можно считать то, что данный продукт на рынке является уникальным. Также сильной стороной является то, что данная система проста в поддержке и доработке дополнительных модулей.

4.1.3 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 4.2.

Таблица 4.2– SWOT-анализ

	Сильные стороны	Слабые стороны
	<p>С1 Грамотно спроектированная архитектура модуля, web-приложения и их интеграция</p> <p>С2 Широкий спектр дополнительного функционала, облегчающий работу сотрудникам</p>	<p>СЛ1 Неудобство использования IP-адреса, при обращении к web-приложению</p> <p>СЛ2 Меньшая производительность по сравнению с нативными приложениями</p>
	<p>С3 Дружественный интерфейс и понятная документация</p> <p>С4 Постоянная поддержка разработчика</p> <p>С5 Минимальные денежные затраты на разработку</p>	<p>СЛ3 Медленная работа на старых устройствах.</p> <p>СЛ4 Привязка некоторого функционала к браузеру</p> <p>СЛ5 Сложности в работе с компонентами устройства</p>

Продолжение таблицы 4.2

<p>Возможности: В1 Покупка доменного имени В2 Интеграция API личного кабинета приложения и Web</p>	<p>В1С5 Финансирование разработки позволит приобрести доменное имя для удобства использования. В2С1С4 Уже реализованная интеграция, а так же поддержка разработчика позволит настроить взаимодействие программ.</p>	<p>В1СЛ2 Возможная необходимость разработать нативное приложение В4СЛ1СЛ3СЛ4 Выходу системы на рынок может воспрепятствовать использование IP-адреса для обращения к web-приложению, а также отсутствие специального дизайна для старых устройств.</p>
<p>В3 Прикрепление видео ответов на вопросы викторины В4 Выход системы на рынок В5 Доработка в связи с пожеланиями</p>	<p>В2В5С4С5 Грамотно спроектированная архитектура модуля, поддержка разработчика и финансирование предприятием позволит доработать систему. В4С2С3С4 Широкий спектр функционала, дружелюбный интерфейс, документация, а также поддержка разработчика способствуют распространению системы на рынок.</p>	<p>В3В2В5СЛ5 Сложности в работе с компонентами устройств может быть помехой в расширении функционала.</p>
<p>Угрозы У1 Неприятие автоматизации пользователями У2 Неверное выполнение инструкций пользователем У3 Медленная работа системы</p>	<p>У1У2С2С3С4 Дружелюбный интерфейс, понятная документация позволят избежать неправильного выполнения инструкций, а также неприятия автоматизации. Так же избежанию неприятия способствует автоформирование отчетности и другие возможности системы. У3С4 Грамотная поддержка разработчика снизит вероятность медленной работы системы</p>	<p>У4СЛ1 Из-за несвоевременного финансирования не возможно купить доменное имя. У1У2У3СЛ1СЛ3СЛ4 Неудобная работа на старых устройствах, медленная работа системы, несвоевременная поддержка системных администраторов, медленная работа системы.</p>

Второй этап состоит в выявлении соответствия сильных и слабых сторон научно-исследовательского проекта внешним условиям окружающей среды. Соотношения параметров представлены в таблицах 4.3-4.6.

Таблица 4.3 – Матрица для сильных сторон и возможностей

Сильные стороны						
Возможности		C1	C2	C3	C4	C5
	V1	+	0	-	-	+
	V2	+	-	-	+	+
	V3	+	-	-	+	+
	V4	-	+	+	-	-
	V5	+	-	-	+	+

Таблица 4.4 – Интерактивная матрица для слабых сторон и возможностей

Слабые стороны						
Возможности		СЛ1	СЛ2	СЛ3	СЛ4	СЛ5
	V1	+	+	-	-	0
	V2	-	0	-	-	+
	V3	-	-	0	-	+
	V4	+	0	+	+	-
	V5	-	-	0	-	+

Таблица 4.5 – Интерактивная матрица для сильных сторон и угроз

Сильные стороны						
Угрозы		C1	C2	C3	C4	C5
	V1	-	+	+	-	-
	V2	-	0	+	-	-
	V3	+	+	0	-	-
	V4	-	-	-	+	+
	V5	+	+	+	-	-

Таблица 4.6 – Интерактивная матрица для слабых сторон и угроз

Слабые стороны						
Угрозы		C1	C2	C3	C4	C5
	V1	+	-	+	+	-
	V2	-	-	+	+	-
	V3	-	+	+	-	-
	V4	+	+	-	-	0
	V5	+	-	+	+	+

4.2 Инициация проекта

В рамках процессов инициации определяются изначальные цели и содержание и фиксируются изначальные финансовые ресурсы. Определяются внутренние и внешние заинтересованные стороны проекта, которые будут взаимодействовать и влиять на общий результат научного проекта.

Таблица 4.7 – Заинтересованные стороны проекта

Заинтересованные стороны проекта	Ожидания заинтересованных сторон
Организация – заказчик	Разработанный функционал приложения по низкой цене
Пользователь	Удобное взаимодействие с университетом
Разработчик	Прибыль
Научный руководитель, студент	Готовая магистерская диссертация

4.2.1 Цели и результаты проекта

Цели и результаты проекта представлены в таблице 4.8:

Таблица 4.8 – Цели и результат проекта

Цели проекта:	<ul style="list-style-type: none">• Спроектировать функционал в соответствии с требованиями заказчика• Создать техническое задание и проектные решения• Разработать спроектированный функционал• Произвести тестирование• Внедрить разработку
----------------------	---

Ожидаемые результаты проекта:	Успешно внедренная разработка у организации-заказчика
Критерии приемки результата проекта:	Успешное тестирование функционала в соответствии с техническим заданием

Продолжение таблицы 4.8

Требования к результату проекта:	Требование:
	Выполненные все пункты технического задания
	Разработанный функционал полностью соответствует проектным решениям

4.2.2 Организационная структура проекта

Таблица 4.9 – Рабочая группа процесса

№ п/п	ФИО, основное место работы, должность	Роль в проекте	Функции	Трудозатраты, час.
1	Алмабек Галымжан Маратулы, ТОО «Паравеб», практикант	Исполнитель	<ul style="list-style-type: none"> • Проектирование функционала • Разработка функционала • Тестирование • Внедрение разработки 	6
2	Ахтямов Эмиль Камильевич, ТОО «Паравеб», директор компании	Руководитель проекта от предприятия, заказчик	<ul style="list-style-type: none"> • Проверка разработки • Помощь во внедрении 	5
ИТОГО:				19

4.2.3 Ограничения и допущения проекта

Ограничения проекта представлены в таблице 5.8:

Таблица 4.10 – Ограничения проекта

Фактор	Ограничения/ допущения
--------	------------------------

Бюджет проекта	34,5 тыс рублей
Источник финансирования	ТОО «Паравеб»
Сроки проекта:	03.02.2018-31.05.2018

Продолжение таблицы 4.10

Дата утверждения плана управления проектом	03.02.2018
Дата завершения проекта	31.05.2018

4.3 Планирование управления научно – техническим проектом

4.3.1 Иерархическая структура работ проекта

Группа процессов планирования состоит из процессов, осуществляемых для определения общего содержания работ, уточнения целей и разработки последовательности действий, требуемых для достижения данных целей. На рисунке 4.2 представлен шаблон иерархической структуры.

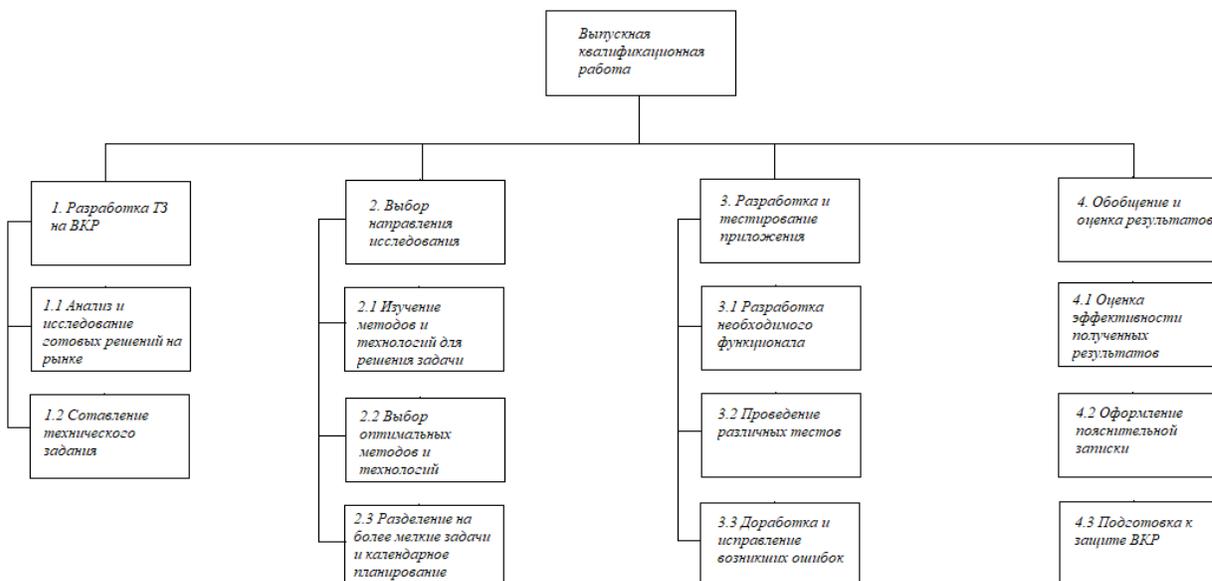


Рисунок 4.2 – Иерархическая структура по ВКР

4.3.2 План проекта

Таблица 4.11 – Календарный план проекта

Код работ-ты	Название	Длительность, дни	Дата начала работ	Дата окончания работ	Состав участников
1	Разработка ТЗ на ВКР	7	11.02.2018	18.02.2018	Ахтямов Э.К.
1.1	Анализ и исследование готовых решений на рынке	1	18.02.2018	19.02.2018	Ахтямов Э.К., Алмабек Г.М.
1.2	Составление и утверждение технического задания	5	20.02.2018	25.02.2018	Ахтямов Э.К.
2	Выбор направления исследования	5	26.02.2018	03.03.2018	Ахтямов Э.К., Алмабек Г.М.
2.1	Изучение методов и технологий для решения задач	15	04.03.2018	19.03.2018	Алмабек Г.М.
2.2	Выбор оптимальных методов решений	2	22.03.2018	24.03.2018	Алмабек Г.М.
3	Разработка и тестирование приложения	30	24.03.2018	21.04.2018	Алмабек Г.М.
3.1	Доработка и исправление возникших ошибок	15	21.04.2018	06.05.2018	Алмабек Г.М.
4	Обобщение и оценка результатов	3	07.05.2018	10.05.2018	Ахтямов Э.К., Алмабек Г.М.
4.2	Оформление пояснительной записки	6	11.05.2018	17.05.2018	Алмабек Г.М.
4.3	Подготовка к защите ВКР	18	20.05.2018	08.06.2018	Алмабек Г.М.
Итого:		92			

Для удобства построения графика, длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни.

Коэффициент календарности определяется по следующей формуле:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - T_{\text{вых}} - T_{\text{пр}}} = \frac{365}{365 - 104 - 14} = 1,48 \quad (4.1)$$

$$T_{ki} = T_{pi} \cdot k_{\text{кал}} \quad (4.2)$$

Таблица 4.12 – Календарный план-график проведения НИОКР

№ работ	Вид работ	Исполн и.	Т _{ки} , кал. дни	Продолжительность выполнения работ												
				февр.		март			апрель			май			июнь	
				2	3	1	2	3	1	2	3	1	2	3	1	
1	Разработка ТЗ на ВКР	Руковод .	6	█												
2	Анализ и исследование готовых решений на рынке	Руковод ., студент	1	█												
3	Составление и утверждение технического задания	Руковод .	4		█											
4	Выбор направления исследования	Руковод ., студент	3		█											
5	Изучение методов и технологий для решения задач	студент	18			█	█	█								
6	Выбор оптимальных методов решений	студент	2					█								
7	Разработка и тестирование приложения	студент	31						█	█	█					
8	Доработка и исправление возникших ошибок	студент	15									█	█			
9	Обобщение и оценка результатов	Руково д., студент	1										█			
10	Оформление пояснительной записки	студент	5											█		
11	Подготовка к защите ВКР	студент	17												█	█



– руководитель,



– студент

4.4 Бюджет научного исследования

4.4.1 Расчет материальных затрат

Этот пункт включает в себя стоимость всех материалов, необходимых для выполнения НИР. К категории материалов относят:

- 1) Хостинг, доменное имя;
- 2) Электроэнергия.

Разработка проводилась в течении 4 месяцев (в среднем 20 дней в месяц) по 6 часов (480 часов), официально заявленная мощность оборудования 0,06 кВт/час.

Затраты на электроэнергию рассчитываются по формуле:

$$C_{эл} = C_{эл} \times P \times F_{об} \quad (4.3)$$

где $C_{эл}$ – тариф на электроэнергию (3,5руб за 1 кВт-ч);

P – мощность оборудования, кВт;

$F_{об}$ – время использования оборудования, ч.

$$C_{эл} = 3,5 \times 0,06 \times 480 = 100,8 \text{ руб.}$$

Так же в статью материальных расходов можно занести покупку хостинга и доменного имени:

$$C_{м} = C_{эл} + C_{х} + C_{ди} \quad (4.4)$$

Необходимо рассчитать затраты на хостинг за 12 месяцев пользования:

$$C_{х} = 12 \times 619,11 = 7429,32 \text{ руб.}$$

$$C_{м} = 100,8 + 7429,32 + 595 = 8125,12 \text{ руб.}$$

Аккаунт разработчика на платформах Android и iOS стоит в сумме за 12 месяцев 7768 рублей.

Таблица 4.13 – Материальные расходы

Материальные затраты	Сумма, руб
Затраты на электроэнергию	100,8
Хостинг	7429,32
Доменное имя	595
Аккаунт разработчика	7768
Итого	15893,12

Таблица 4.13 – Расчет затрат по статье «Спецоборудование»

№ п/п	Наименование оборудования	Кол-во единиц оборудования	Цена единицы оборудования, тыс.руб.	Общая стоимость оборудования, тыс.руб.
1.	Персональный компьютеры	1	-	-
2.	Электроэнергия	-	-	100,8
3.	Среда разработки Visual Studio Code	1	-	-
	Итого		-	-

4.4.2 Основная заработная плата

В данную статью включается основная заработная плата руководителя от предприятия и студента, также премия, выплачиваемая ежемесячно из фонда заработной платы в размере 20 –30 % от тарифа или оклада. Расчет выполняется на основе трудоемкости выполнения каждого этапа и величины месячного оклада исполнителя.

Расчет основной заработной платы приведен в таблице 4.14.

Таблица 4.14 – Расчет основной заработной платы

Исполнители	Исполнители по категориям	Трудо-емкость, чел.-дн., Т _р	Заработная плата, приходящаяся на один чел.-дн., тыс. руб.з/дн	Всего заработная плата по тарифу (окладам), руб.З _{осн}
Студент		62,9		2410
Руководитель	старший преподаватель	9,9	934,7	9253,53
Итого				11663,53

$$Z_{\text{зп}} = Z_{\text{осн}} + Z_{\text{доп}} \quad (4.5)$$

где $Z_{\text{осн}}$ – основная заработная плата;

$Z_{\text{доп}}$ – дополнительная заработная плата (12% от $Z_{\text{осн}}$).

Основная заработная плата ($Z_{\text{осн}}$) руководителя от ТПУ (при наличии руководителя от предприятия) рассчитывается по следующей формуле:

$$Z_{\text{осн}} = Z_{\text{дн}} \cdot T_p \quad (4.6)$$

где $Z_{\text{осн}}$ – основная заработная плата одного работника;

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

Значит, для руководителя:

$$Z_{\text{осн}} = 934,7 \times 9,9 = 9253,53$$

$Z_{\text{дн}}$ – среднедневная заработная плата работника, руб.

Среднедневная заработная плата рассчитывается по формуле:

$$Z_{\text{дн}} = \frac{Z_m \cdot M}{F_d} \quad (4.7)$$

где Z_m – месячный должностной оклад работника, руб.;

M – количество месяцев работы без отпуска в течение года:

при отпуске в 48 раб.дней $M=10,4$ месяца, 6-дневная неделя;

F_d – действительный годовой фонд рабочего времени научно-технического персонала, раб.дн.(таблица 4.15).

Для руководителя:

$$Z_{\text{дн}} = \frac{23100 \times 10,4}{257} = 934,7$$

Баланс рабочего времени представлен в таблице 4.15.

Таблица 4.15 – Баланс рабочего времени

Показатели рабочего времени	Руководитель	Студент
Календарное число дней	365	365
Количество нерабочих дней		
-выходные дни	46	16
-праздничные дни	14	14
Потери рабочего времени		
-отпуск	48	48
-невыходы по болезни	–	–
Действительный годовой фонд рабочего времени	257	257

Месячный должностной оклад работника рассчитывается по формуле:

$$Z_M = Z_6 * K_p \quad (4.8)$$

где Z_6 – базовый оклад, руб;

K_p – районный коэффициент, равный 1,3.

Заработная плата старшего преподавателя составляет 23100 руб., согласно «Положению об оплате труда» ТПУ.

Для руководителя: $Z_M = 23100 * 1,3 = 30030$ руб.

Результаты расчета основной заработной платы представлены в таблице 4.16.

Таблица 4.16 - Результаты расчета основной заработной платы

Исполнители	Z_6 ,руб.	K_p	Z_M ,руб	$Z_{дн}$,руб.	$T_{раб.раб. дн.}$	$Z_{осн}$,руб.
Руководитель	23100	1,3	30030	1201,2	48	9253,53
Итого $Z_{осн}$						9253,53

4.4.3 Дополнительная заработная плата научно-производственного персонала

Затраты по дополнительной заработной плате исполнителей темы учитывают величину предусмотренных Трудовым кодексом РФ доплат за отклонение от нормальных условий труда, а также выплат, связанных с обеспечением гарантий и компенсаций.

Расчет дополнительной заработной платы ведется по следующей формуле:

$$Z_{\text{доп}} = k_{\text{доп}} \cdot Z_{\text{осн}} \quad (4.9)$$

Для руководителя:

$$Z_{\text{доп}} = 9253,53 \times 0,12 = 1110,4 \text{ рублей}$$

В таблице 4.17 приведен расчёт основной и дополнительной заработной платы.

Таблица 4.17 – Заработная плата исполнителей ВКР

Заработная плата	Руководитель	Студент
Основная зарплата	9253,53	2410
Дополнительная зарплата	1110,4	–
Зарплата исполнителя	10363,9	2410
Итого по статье С _{зп}		12773,9

4.4.4 Накладные расходы

Расчет накладных расходов ведется по следующей формуле:

$$C_{\text{накл}} = k_{\text{накл}} * (Z_{\text{осн}} + Z_{\text{доп}}) \quad (4.10)$$

где $k_{\text{накл}}$ – коэффициент накладных расходов.

$$C_{\text{накл}} = 0,8 \times (9253,53 + 1110,4) = 8291,14 \text{ руб.}$$

4.4.5 Формирование бюджета затрат научно-исследовательского проекта

Таблица 4.18 – Группировка затрат по статьям

Вид работ	Статьи						
	Материальные расходы	Специальное оборудование для научных (экспериментальных) работ	Основная заработная плата	Дополнительная заработная плата	Отчисления на социальные нужды	Накладные расходы	Итого плановая себестоимость
1.	15893,12	–	9253,53	1110,4	3109,17	8291,14	34548,19
2.	21000	45000	14000	3000	5100	13500	96500

В результате было получено, что бюджет затрат НИИ составит 34548,19 руб. При этом затраты у конкурента составляют 96500 рублей, из чего можно сделать вывод что полученный продукт будет экономичней, чем у конкурентов.

4.5 Реестр рисков проекта

Идентифицированные риски проекта включают в себя возможные неопределенные события, которые могут возникнуть в проекте и вызвать последствия, которые повлекут за собой нежелательные эффекты. Информацию по данному разделу сведены в таблицу 4.19.

Таблица 4.19 – Реестр рисков

№	Риск	Потенциальное воздействие	Вероятность наступления (1-5)	Влияние риска (1-5)	Уровень риска*	Способы смягчения риска	Условия наступления
---	------	---------------------------	-------------------------------	---------------------	----------------	-------------------------	---------------------

Продолжение таблицы 4.19

1	Изменение требований к продукту	Задержка срока выхода продукта на рынок, увеличение себестоимости	4	4	4	Разделить требования на "абсолютно необходимые" и "хорошо бы было иметь", до запуска системы выполнять только абсолютно необходимые требования	Смена ответственного о лица со стороны заказчика, переориентация конечного продукта
2	Затруднения с финансированием	Отказ разработчика поддерживать продукт	2	5	5	Поднять вопрос на уровень директора компании	Нехватка бюджета заказчика

4.5.1 Чистая текущая стоимость (NPV)

Данный метод основан на сопоставлении дисконтированных чистых денежных поступлений от операционной и инвестиционной деятельности.

Если инвестиции носят разовый характер, то NPV определяется по формуле:

$$NPV = \sum_{t=1}^n \frac{ЧДП_{опt}}{(1+i)^t} - I_0 \quad (4.11)$$

где $ЧДП_{опt}$ – чистые денежные поступления от операционной деятельности;

I_0 – разовые инвестиции, осуществляемые в нулевом году;

t – номер шага расчета ($t=0, 1, 2 \dots n$);

n – горизонт расчета;

i – ставка дисконтирования (желаемый уровень доходности инвестируемых средств).

Чем больше NPV, тем больше влияние инвестиционного проекта на экономический потенциал предприятия, реализующего данный проект, и на экономическую ценность этого предприятия.

Таким образом, инвестиционный проект считается выгодным, если **NPV** является положительной.

Таблица 4.20 – Расчет чистой текущей стоимости по проекту в целом

№	Наименование показателей	Шаг расчета				
		0	1	2	3	4
1.	Выручка от реализации, тыс.руб	0	51,822	51,822	51,822	51,822
2.	Итого приток, тыс.руб	0	51,822	51,822	51,822	51,822
3.	Инвестиционные издержки, тыс.руб.	-34,548	0	0	0	0
4.	Операционные затраты, тыс. руб С+Ам+ФОТ	0	16,324	16,324	16,324	16,324
5.	Налогооблагаемая прибыль		35,498	35,498	35,498	35,498
6.	Налоги, тыс. руб Выр-опер=донал.приб*20%		7,100	7,100	7,100	7,100
7.	Итого отток, тыс.руб. Опер.затр+налоги	-34,548	23,424	23,424	23,424	23,424
8.	Чистый денежный поток, тыс.руб. ЧДП=Пчист+Ам Пчист=Пдонал.-налог	-34,548	28,398	28,398	28,398	28,398
9.	Коэффициент дисконтирования (приведения при $i = 20\%$)	1,0	0,833	0,694	0,578	0,482
10.	Дисконтированный чистый денежный поток, тыс.руб. (с7*с8)	-34,548	23,655	19,708	16,414	13,688
11.	То же нарастающим итогом, тыс.руб. (NPV =38,917тыс.руб.)	-34,548	-10,893	8,815	25,229	38,917

4.5.2 Дисконтированный срок окупаемости

Рассчитывается данный показатель примерно по той же методике, что и простой срок окупаемости, с той лишь разницей, что последний не учитывает фактор времени.

Наиболее приемлемым методом установления дисконтированного срока окупаемости является расчет кумулятивного (нарастающим итогом) денежного потока (таблица 4.20).

Таблица 4.21– Дисконтированный срок окупаемости.

№	Наименование показателя	Шаг расчета				
		0	1	2	3	4
1.	Дисконтированный чистый денежный поток ($i=0,20$)	-34,548	23,655	19,708	16,414	13,688
2.	То же нарастающим итогом	-34,548	-10,893	8,815	25,229	38,917
3.	Дисконтированный срок окупаемости	$PP_{диск} = 1 + 10,893/19,708 = 1,5$ года				

4.5.3 Внутренняя ставка доходности

Для установления показателя чистой текущей стоимости (NPV) необходимо располагать информацией о ставке дисконтирования, определение которой является проблемой, поскольку зависит от оценки экспертов. Поэтому, чтобы уменьшить субъективизм в оценке эффективности инвестиций на практике широкое распространение получил метод, основанный на расчете внутренней ставки доходности (IRR).

Между чистой текущей стоимостью (NPV) и ставкой дисконтирования (i) существует обратная зависимость. Эта зависимость следует из таблицы 4.22 и графика 5.1.

Таблица 4.22– Зависимость NPV от ставки дисконтирования.

№	Наименование показателя	0	1	2	3	4	NPV
1	Чистые денежные потоки	-34,548	28,398	28,398	28,398	28,398	
2	коэффициент дисконтирования						
	$i=0,1$	1	0,909	0,826	0,751	0,683	
	$i=0,2$	1	0,833	0,694	0,578	0,482	
	$i=0,3$	1	0,769	0,592	0,455	0,350	
	$i=0,4$	1	0,714	0,51	0,364	0,26	
	$i=0,5$	1	0,667	0,444	0,295	0,198	
	$i=0,6$	1	0,625	0,390	0,244	0,095	
	$i=0,7$	1	0,588	0,335	0,203	0,070	
	$i=0,8$	1	0,556	0,309	0,171	0,095	
	$i=0,9$	1	0,526	0,277	0,146	0,077	
3	Дисконтированный денежный поток, тыс. руб						
	$i=0,1$	-34,548	25,814	23,457	21,327	19,396	55,446
	$i=0,2$	-34,548	23,656	19,708	16,414	13,688	38,918
	$i=0,3$	-34,548	21,838	16,812	12,921	9,939	26,962
	$i=0,4$	-34,548	20,276	14,483	10,337	7,383	17,931
	$i=0,5$	-34,548	18,941	12,609	8,377	5,623	11,002
	$i=0,6$	-34,548	17,749	11,075	6,929	2,698	3,903
	$i=0,7$	-34,548	16,698	9,513	5,765	1,988	-0,584
	$i=0,8$	-34,548	15,789	8,775	4,856	2,698	-2,43
	$i=0,9$	-34,548	14,937	7,866	4,146	2,187	-5,412

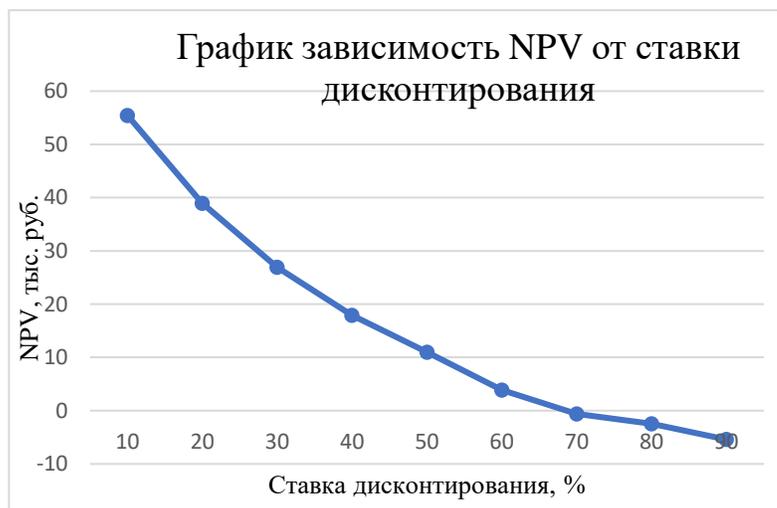


График 4.1 – Зависимость NPV от ставки дисконтирования

Из таблицы и графика следует, что по мере роста ставки дисконтирования чистая текущая стоимость уменьшается, становясь отрицательной. Значение ставки, при которой NPV обращается в нуль, носит название «внутренней ставки доходности» или «внутренней нормы прибыли». Из таблицы и графика следует, что по мере роста ставки дисконтирования чистая текущая стоимость уменьшается, становясь отрицательной. Значение ставки, при которой обращается в нуль, носит название «внутренней ставки доходности» или «внутренней нормы прибыли». Из графика получаем, что IRR составляет 0,69.

4.5.4 Индекс доходности (рентабельности) инвестиций (PI)

Индекс доходности показывает, сколько приходится дисконтированных денежных поступлений на рубль инвестиций.

Расчет этого показателя осуществляется по формуле

$$PI = \sum_{t=1}^n \frac{ЧПД_t}{(1+i)^t} / I_0, \quad (4.12)$$

где I_0 – первоначальные инвестиции.

$$PI = \frac{23,655 + 19,708 + 16,414 + 13,688}{34,548} = 2,126$$

$PI = 2,126 > 1$, следовательно, проект эффективен при $i = 0,2$; $NPV = 38,917$ тыс. руб.

4.6 Оценка абсолютной эффективности

Для оценки социальной эффективности научного проекта магистранту необходимо выявить критерии социальной эффективности, на которые влияет реализация научного проекта и оценить степень их влияния.

Таблица 4.22 – Критерии социальной эффективности

ДО	ПОСЛЕ
Затрудненность просмотра списка направлений ВУЗа, через портативные устройства	Вся важная информация о поступлении доступна в одном приложении, без доступа в Интернет
Необходимость постоянного доступа в Интернет	Приложение работает в автономном режиме без ссылок на портал университета (данные хранятся в кэш)
Поиск на различных форумах и сайтах о шансах на поступление по определенному направлению	В приложение реализован функционал «Калькулятор», с помощью которого возможно оценить свои шансы на поступление

4.7 Оценка сравнительной эффективности исследования

Интегральный показатель финансовой эффективности научного исследования получают в ходе оценки бюджета затрат трех (или более) вариантов исполнения научного исследования. Для этого наибольший интегральный показатель реализации технической задачи принимается за базу расчета (как знаменатель), с которым соотносятся финансовые значения по всем вариантам исполнения.

Интегральный финансовый показатель разработки определяется:

$$I_{финр}^{исп.i} = \frac{\Phi_{pi}}{\Phi_{max}}, \quad (4.13)$$

где $I_{финр}^{исп.i}$ – интегральный финансовый показатель разработки;

Φ_{pi} – стоимость i -го варианта исполнения;

Φ_{max} – максимальная стоимость исполнения научно-исследовательского проекта (в т.ч. аналоги).

Полученная величина интегрального финансового показателя разработки отражает соответствующее численное увеличение бюджета затрат разработки в размах (значение больше единицы), либо соответствующее численное удешевление стоимости разработки в размах (значение меньше единицы, но больше нуля).

Так как разработка имеет одно исполнение, то

$$I_{\text{финр}}^p = \frac{\Phi_p}{\Phi_{\text{max}}} = \frac{34548,19}{34548,19} = 1$$

Для аналогов (с использованием дополнительного оборудования, которое стоит 4000 руб и 7000 руб) соответственно:

$$I_{\text{фин1}}^{a1} = \frac{\Phi_{a1}}{\Phi_{\text{max}}} \quad (4.14)$$

$$I_{\text{фин a1}}^{a1} = \frac{\Phi_{a1}}{\Phi_{\text{max}}} = \frac{38548,19}{34548,19} = 1,11;$$

$$I_{\text{фин a2}}^{a2} = \frac{\Phi_{a2}}{\Phi_{\text{max}}} = \frac{41548,19}{34548,19} = 1,2;$$

Расчёт интегрального показателя ресурсоэффективности представлен в таблице 4.23.

Таблица 4.23 – Сравнительная оценка вариантов исполнения проекта

Критерии \ Объект исследования	Весовой коэффициент параметра	Текущий проект	Аналог 1	Аналог 2
1. Способствует росту производительности труда пользователя	0,4	4	3	5
2. Удобство в эксплуатации (соответствует требованиям потребителей)	0,2	5	5	3
3. Помехоустойчивость	0,1	5	4	4
4. Надёжность	0,25	4	3	5
5. Материалоёмкость	0,15	5	3	3
ИТОГО	1	4,6	3	3,3

$$I_{\text{тп}} = 4 \cdot 0,4 + 5 \cdot 0,2 + 5 \cdot 0,1 + 4 \cdot 0,25 + 5 \cdot 0,15 = 4,85;$$

$$\text{Аналог 1} = 3 \cdot 0,4 + 5 \cdot 0,2 + 4 \cdot 0,1 + 3 \cdot 0,25 + 3 \cdot 0,15 = 3,8;$$

$$\text{Аналог 2} = 5 \cdot 0,4 + 3 \cdot 0,2 + 4 \cdot 0,1 + 5 \cdot 0,25 + 3 \cdot 0,15 = 4,7.$$

В результате:

$$I_{\text{финр}}^p = \frac{I_m^p}{I_{\text{финр}}^p} = \frac{4,85}{1} = 4,85$$

$$I_{\text{финр}}^p = \frac{I_m^p}{I_{\text{финр}}^p} = \frac{3,8}{1,11} = 3,42$$

$$I_{\text{финр}}^p = \frac{I_m^p}{I_{\text{финр}}^p} = \frac{4,7}{1,2} = 3,91$$

Сравнение интегрального показателя эффективности текущего проекта и аналогов позволит определить сравнительную эффективность проекта.

Сравнительная эффективность проекта:

$$I_{\text{финр}}^p = \frac{I_{\text{финр}}^p}{I_{\text{финаi}}^{\text{ai}}}$$

Результат вычисления сравнительной эффективности проекта и сравнительная эффективность анализа представлены в таблице 4.24.

Таблица 4.24– Сравнительная эффективность разработки

№	Показатели	Аналог 1	Аналог 2	Разработка
1	Интегральный финансовый показатель разработки	1,11	1,2	1
2	Интегральный показатель ресурсоэффективности разработки	3,8	4,7	4,85
3	Интегральный показатель эффективности	3,42	3,91	4,85
4	Сравнительная эффективность вариантов исполнения	3,08	3,25	4,85

Сравнение значений интегральных показателей эффективности позволяет судить о приемлемости существующего варианта решения поставленной в

магистерской диссертации технической задачи с позиции финансовой и ресурсной эффективности.

В ходе проведения анализа показателей эффективности инвестиций была получена чистая текущая стоимость (NPV) – 38,917тыс. руб. Таким образом, данный инвестиционный проект можно считать выгодным, NPV является положительной величиной. Дисконтированный срок окупаемости проекта ($PP_{\text{диск}}$) составляет 1,5 года. Внутренняя ставка доходности (IRR) – 0,69, что позволяет признать инвестиционный проект экономически оправданным, так как выполняется условие неравенства $IRR > i$. Индекс доходности (PI) – 2,126, и, основываясь на том, что данная величина превышает единицу, можно утверждать, что данная инвестиция приемлема.

5 Социальная ответственность

Для рассмотрения производственной безопасности проекта необходимо выявить вредные и опасные факторы, которые могут возникнуть на рабочем месте, и описать мероприятия по защите исследователя и пользователей конечных продуктов от действия опасных и вредных факторов.

5.1 Анализ вредных и опасных факторов, которые могут возникнуть на рабочем месте при выполнении проекта

На данном этапе выполнения работы необходимо выявить источники опасности, то есть части производственных систем, производственного оборудования и элементы среды, формирующие эти опасности.

Производственные условия на рабочем месте характеризуются наличием некоторых опасных и вредных факторов, которые в соответствии с ГОСТ 12.0.003-74 опасные и вредные производственные факторы подразделяются по природе действия на следующие группы: физические, химические, биологические и психофизиологические [31].

В общих случаях к определенным признакам опасных и вредных факторов относятся: затруднение осуществления физиологических функций дыхания, возможность непосредственного воздействия на организм, кровообращения, работы центральной нервной системы, органов пищеварения, выделения.

В соответствии с [32] приведены основные опасные и вредные производственные факторы, воздействующие на персонал, пользующийся компьютером при выполнении данной разработки:

Вредные:

а) микроклимат (ГОСТ 12.1.005-88 с изм. №1 от 2000 г., СанПиН 2.2.4.548-96);

- б) шум (ГОСТ 12.1.003-83) [28];
- в) электромагнитное поле [33].
- г) психофизиологические факторы.

Опасные:

- а) статическое электричество;
- б) электрический ток;
- в) короткое замыкание.

5.2 Производственная безопасность

Производственное помещение – это пространство, производственная среда, где осуществляется трудовая деятельность человека. В производственных помещениях должны быть обеспечены и соблюдены нормативные санитарно-технические условия.

При планировании рабочего помещения необходимо соблюдать нормы полезной площади и объема помещения.

Размеры рабочего кабинета представлены в таблице 5.1.

Таблица 5.1 – Размеры рабочего кабинета

длина помещения	7 м
ширина	6 м
высота	5 м

Рабочее помещение представляет собой комнату площадью 42 м² и объемом 210 м³. Одновременно в рабочем помещении находится 6 человек, следовательно на одного человека приходится около 35 м³ объема помещения и 7 м² площади, что удовлетворяет требованиям санитарных норм [34], согласно которым для одного работника должны быть предусмотрены площадь величиной не менее 6 м² и объем

не менее 20 м³ с учетом максимального числа одновременно работающих в смену [34].

Выбор типа производственного помещения определяется производственным процессом, и при анализе опасных и вредных факторов необходимо ориентироваться на конкретное рабочее место и конкретные условия труда.

5.2.1 Микроклимат

Микроклимат помещения – это комплекс метеорологических условий в данном помещении. Оптимальные и допустимые значения характеристик микроклимата устанавливаются в соответствии с СанПиН 2.2.4.548-96 и СанПиН 2.2.4.1294-03.

Требования к микроклимату определяются исходя из категории тяжести работ. Работа разработчика-программиста относится к первой категории тяжести в соответствии с [35].

Далее приводится анализ микроклимата в помещении, где находится рабочее место.

В таблице 5.2 приведены параметры микроклимата.

Таблица 5.2 - Оптимальные и допустимые значения микроклимата

Период года	Температура, °С	Относительная влажность, %	Скорость движения воздуха, м/с
Холодный	23-25	40-60	0,1
Тёплый	20-22	40	0,1

Для анализа микроклимата в помещении, где находится рабочее место необходимо определить следующие параметры: температура, относительная

влажность, скорость движения воздуха. Эти параметры по отдельности и в комплексе влияют на организм человека, определяя его самочувствие.

По степени физической тяжести работа программиста относится к категории лёгких работ. Параметры микроклимата в помещении, где находится рабочее место, регулируются системой центрального отопления и приточно-вытяжной вентиляцией, и имеют следующие значения: влажность 45%, скорость движения воздуха 0,15 м/с, температура летом (теплый период года) 23..25°C, зимой (холодный период года) 20..25°C.

К мероприятиям по оздоровлению воздушной среды в производственном помещении относятся: правильная организация вентиляции и кондиционирования воздуха, отопление помещений. Вентиляция может осуществляться естественным и механическим путём. В помещении должны подаваться следующие объёмы наружного воздуха: при объёме помещения до 20 м³ на человека – не менее 30 м³ в час на человека; при объёме помещения более 40 м³ на человека и отсутствии выделения вредных веществ допускается естественная вентиляция.

В рабочем помещении осуществляется естественная вентиляция. Воздух поступает и удаляется через щели, окна, двери. Основной недостаток такой вентиляции в том, что приточный воздух поступает в помещение без предварительной очистки и нагревания. Также подобная вентиляция создает сквозняки, что негативно сказывается на работниках.

В зимнее время в помещении необходимо предусмотреть систему отопления. Она должна обеспечивать достаточное, постоянное и равномерное нагревание воздуха. В помещениях с повышенными требованиями к чистоте воздуха должно использоваться водяное отопление. В рассматриваемом офисе используется водяное отопление со встроенными нагревательными элементами и стояками.

5.2.2 Производственный шум

Одним из важнейших параметров, наносящим большой ущерб для здоровья и резко снижающим производительность труда, является шум. Действие шума различно: затрудняет разборчивость речи, вызывает снижение работоспособности, повышает утомляемость, вызывает необратимые изменения в органах слуха человека. Шум воздействует не только на органы слуха, но и на весь организм человека через центральную нервную систему. Ослабляется внимание, ухудшается память, снижается реакция, увеличивается число ошибок при работе [36].

Производственные помещения, в которых для работы используются ПЭВМ, не должны граничить с помещениями, в которых уровень шума и вибрации превышают нормируемые значения. При выполнении основной работы на ПЭВМ уровень шума на рабочем месте не должен превышать 50 дБ. Допустимые уровни звукового давления в помещениях для персонала, осуществляющего эксплуатацию ЭВМ при разных значениях частот [37].

По субъективным ощущениям шумовая обстановка на рабочем месте соответствует норме.

5.2.3 Электромагнитные поля

Как любые электрические приборы, видеотерминалы (ВДТ) и системные блоки производят электромагнитное излучение. Большая часть его происходит не от экрана монитора (ВДТ), а от видеокабеля и системного блока. В портативных компьютерах практически все электромагнитное излучение идет от системного блока, располагающегося под клавиатурой. Современные машины выпускаются заводом-изготовителем со специальной металлической защитой внутри системного блока для уменьшения фона электромагнитного излучения.

При воздействии полей, имеющих напряженность выше предельно допустимого уровня, развиваются нарушения со стороны нервной, сердечно-сосудистой систем, органов пищеварения и некоторых биологических показателей крови.

Согласно СанПиН 2.2.2/2.4.1340-03 напряженность электромагнитного поля на расстоянии 50 см вокруг ВДТ по электрической составляющей должна быть не более:

В диапазоне частот 2 кГц ÷ 400кГц – 2,5 В/м.

В диапазоне частот 5 Гц ÷ 2 кГц – 25 В/м.

Плотность магнитного потока должна быть не более:

В диапазоне частот 2 кГц ÷ 400кГц – 25 нТл.

В диапазоне частот 5 Гц ÷ 2 кГц – 250 нТл.

Возможные способы защиты от ЭМП:

- Основной способ – увеличение расстояния от источника, во избежание последствий экран видеомонитора должен находиться на расстоянии не менее 50 см от пользователя.
- Применение экранных фильтров, специальных экранов и других средств индивидуальной защиты, прошедших испытание в аккредитованных лабораториях и имеющих соответствующий гигиенический сертификат.

5.2.4 Психофизиологические факторы

Наиболее эффективные средства предупреждения утомления при работе на ПК – это средства, нормализующие активную трудовую деятельность человека. Уменьшение плотности рабочего времени, наличие простоев на протяжении рабочего дня не только не отдалают наступление и развитие утомления, но могут ускорить и углубить его.

Перерывы различаются по своему значению и продолжительности. В середине рабочего дня назначается обеденный перерыв, продолжительность которого должна составлять 1 час. Расположение дополнительных перерывов на протяжении рабочего дня их количество и продолжительность определяется на основании физиологического и психологического изучения динамики работоспособности.

В соответствии с утвержденной инструкцией по охране труда рекомендуется 30 минутный перерыв после каждых двух часов работы или 15 минутный перерыв на каждый час. Доказано что частые паузы до развития утомления намного ценнее длительных, но менее частых перерывов, начинающихся уже после снижения уровня работоспособности.

Основными элементами рабочего места, оснащенного дисплеем, является рабочее кресло, рабочая поверхность, экран дисплея и клавиатура.

Рабочее кресло обеспечивает и поддержание рабочей позы, в положении сидя, и чем длиннее это положение в течении рабочего дня тем настоятельнее должны быть требования к созданию удобных и правильных рабочих сидений.

Сидение должно быть удобным, иметь закругленные края, наклоняться по отношению к горизонтали вперед на 2 градуса и назад на 14 градусов. Его размеры не должны превышать 40*40см. Сиденье должно быть покрыто латексом толщиной 1 см, сверху которого накладывается влагопроницаемый материал.

Высота спинки кресла рекомендуется 48-50 см от поверхности сидения и с регулировкой в переднезаднем направлении. На высоте 10-20 см от поверхности сидения ее следует оборудовать поясничным опорным валиком.

Рабочий стол должен иметь стабильную конструкцию. Высоту стола необходимо регулировать в диапазоне 65-85 см. При этом высота от горизонтальной линии зрения до рабочей поверхности стола при выпрямленной рабочей позе должна быть 45-50 см.

Так же для предотвращения психофизиологических факторов рабочее место должно удовлетворять следующим нормативам:

- Освещение [32];
- Уровень шума [36]
- Температура и влажность воздуха в помещении [37];
- Приём пищи на рабочем месте [38].

Главным направлением работы по коренному улучшению условий труда должны быть профилактические меры технического характера, поскольку они направлены на устранение причин, порождающих неблагоприятные условия труда. Поэтому, прежде всего предусматривается проведение мероприятий по внедрению новейших технологий, технологий, отвечающих эргономическим (гигиеническим, психофизиологическим, эстетическим и др.) Требованиям, по реконструкции зданий и сооружений, модернизации действующего оборудования, совершенствованию организации труда и производства. Эффективность и эффективность благоприятных условий труда, способствующих увеличению производственного травматизма, производственно обусловленной заболеваемости и текучести кадров.

5.3 Экологическая безопасность

Экологическая безопасность и охрана окружающей среды являются одними из важнейших факторов при выполнении работ любого характера. При работе в офисном помещении за персональным ПК отсутствуют выбросы в окружающую среду и нет влияния на жилищную зону.

Поскольку при разработке данной магистерской диссертации использовался компьютер, необходимо помнить о правильной утилизации компьютерного лома после выхода из строя данного ПК. В соответствии с постановлением

правительства №340 [34] юридическим лицам запрещено самостоятельно утилизировать компьютерную технику. Необходимо найти организацию, которая занимается утилизацией в частном порядке. Это относится к следующим видам отходов:

– образование твердых отходов, относящихся к IV классу опасности (системный блок компьютера, принтеры, сканеры, клавиатура, манипулятор "мышь") и жидких отходов, образование твердых отходов, относящихся к IV классу опасности (системный блок компьютера, принтеры, сканеры, клавиатура, манипулятор "мышь") и жидких отходов;

– Жидкие отходы: сточные воды;

– Люминесцентные лампы.

5.3.1 Безопасность в чрезвычайных случаях

Персональный компьютер – электроприбор. Основным отличием персонального компьютера от других электроприборов является длительное время эксплуатации без отключения от электрической сети. В связи с этим следует особое внимание уделить качеству организации электропитания. Во время работы за компьютером запрещается:

- прикасаться к задней панели системного блока и переключать разъемы кабелей периферийных устройств при включенном питании;

- загромождать верхние панели устройств бумагами и посторонними предметами;

- производить отключение питания во время выполнения активных задач;

- включать охлажденное оборудование;

- работать на офисной технике при снятых кожухах, при отсутствии или неисправности предусмотренных конструкцией оборудования предохранительных приспособлений, блокировок;
- отключать оборудование от электросети и выдергивать электрическую вилку, держась за шнур;
- самостоятельно производить вскрытие, обслуживание и ремонт офисной техники.

Во избежание воздействия возможных опасных факторов необходимо:

- занулять электрооборудование;
- следить за своевременным техническим обслуживанием оборудования; корпус системного блока необходимо очищать от пыли не реже 1 раза в шесть месяцев, во избежание замыкания электрических контактов;
- использовать только исправное электрооборудование; исключить резкие перегибы электрических проводов.

Также не следует работать с персональными электронно-вычислительными машинами (ПЭВМ) в условиях повышенной влажности (относительная влажность воздуха длительно превышает 75%), высокой температуры (более 35°C), наличии токопроводящей пыли, токопроводящих полов и возможности одновременного соприкосновения к имеющим соединение с землёй металлическим элементам и металлическим корпусом электрооборудования. Правилами запрещено работать с ПЭВМ в таких условиях. Таким образом, работа с ПЭВМ может проводиться только в помещениях без повышенной опасности, и возможность поражения током может быть только при прикосновении непосредственно с элементами ПЭВМ.

Помещение, в котором проводились работы, относится к помещению класса 1 - без повышенной опасности поражения электрическим током, то есть отсутствуют условия, создающие повышенную опасность согласно ПУЭ 1.1.13.

5.3.2 Мероприятия по предотвращению ЧС

Для предупреждения пожаров от коротких замыканий и перегрузок необходимы правильный выбор, монтаж и соблюдение установленного режима эксплуатации электрических сетей, дисплеев и других электрических средств автоматизации.

Следовательно, необходимо предусмотреть ряд профилактических мероприятий технического, эксплуатационного, организационного плана.

На каждом этаже здания, на видном месте вывешен план эвакуации с этажа (здания). На плане эвакуации кроме путей выхода (стрелками), указываются места размещения средств пожаротушения, телефонов.

Для предупреждения возникновения пожара необходимо соблюдать следующие правила пожарной безопасности:

- а) исключение образования горючей среды (герметизация оборудования, контроль воздушной среды, рабочая и аварийная вентиляция);
- б) применение при строительстве и отделке зданий негорюемых или трудно сгораемых материалов.

Необходимо в офисе проводить следующие пожарно-профилактические мероприятия:

- 1) Организационные мероприятия:
 - а. противопожарный инструктаж обслуживающего персонала;
 - б. обучение персонала правилам техники безопасности;
 - с. издание инструкций, плакатов, планов эвакуации.
- 2) Эксплуатационные мероприятия:
 - а. соблюдение эксплуатационных норм оборудования;

- b. обеспечение свободного подхода к оборудованию.
- c. содержание в исправности изоляции токоведущих проводников.

3) Технические мероприятия:

a. соблюдение противопожарных мероприятий при устройстве электропроводок, оборудования, систем отопления, вентиляции и освещения. В помещении аудитории имеется порошковый огнетушитель типа ОУ-5, установлен рубильник, обесточивающий все помещение, на двери аудитории приведен план эвакуации в случае пожара, и на досягаемом расстоянии находится пожарный щит. Если возгорание произошло в электроустановке, для его устранения должны использоваться углекислотные огнетушители или порошковые;

- b. профилактический осмотр, ремонт и испытание оборудования.

Кроме устранения самого очага пожара, нужно своевременно организовать эвакуацию людей.

5.4 Правовые и организационные вопросы обеспечения безопасности

В соответствии с ГОСТ 12.2.032-78 ССБТ «Рабочее место при выполнении работ сидя. Общие эргономические требования» к рабочему месту предъявляются следующие основные требования:

- Конструкцией рабочего места должно быть обеспечено выполнение трудовых операций в пределах зоны досягаемости моторного поля;
- При организации рабочего места следует учитывать антропометрические показатели женщин (если работают только женщины) и мужчин (если работают только мужчины); если работают и женщины, и мужчины – общие средние показатели женщин и мужчин;

– Конструкцией рабочего места должно быть обеспечено оптимальное положение работающего, которое достигается регулированием высоты рабочей поверхности, сиденья и пространства для ног [35].

В условиях выполнения данной работы шум и вибрация находятся в пределах норм. Доза облучения при работе с компьютером составляет 100мкР/час, с уменьшением расстояния до экрана доза облучения увеличивается. В целом, дозы облучения, создаваемые монитором, невелики по сравнению с естественным фоном ионизирующего облучения, что достигается применением средств коллективной защиты.

Способом защиты от излучения является использование защитных экранов, улавливающих электромагнитные поля. Также рекомендуется в помещении больше устанавливать растений, поглощающих вредное излучение.

Деятельность человека причиняет ущерб окружающей среде, а потому перед обществом стоит задача сделать это воздействие наименее пагубным. В процессе трудовой деятельности в вычислительных центрах, также, как и обычной жизнедеятельности, человек является источником твердых бытовых отходов. Эти отходы, как пищевые, так и промышленные, сильно загрязняют окружающую среду. Как правило, в качестве промышленных отходов выступают бумага, диски, строительные отходы, коробки и т.п. Этот мусор с другими отходами вывозится на территории, выделенные под складирование бытовых отходов. Сжигание этих отходов уменьшает их объем на 90%, но в результате сжигания происходит выделение вредных газов и дымов, загрязняющих атмосферу.

Компьютерная техника постоянно устаревает и выходит из строя, что ведет к ее обновлению и появлению большого количества отходов электроники. Утилизация электроники является обязательным и производится специализированными отделами.

Заключение

Результатом работы является мобильное приложение для абитуриентов Томского государственного университета систем управления и радиотехники.

В процессе выполнения данной выпускной квалификационной работы, был проведен анализ и выявлены необходимые требования, которыми должен обладать мобильное приложение для абитуриентов. На основе данных требований было создано мобильное приложение «Абитуриент ТУСУР», позволяющее иметь под рукой информацию о списке рейтинга, новостях и мероприятиях, также в приложения абитуриент может оценить свои возможности на поступление по конкретному направлению.

Данное приложение отличается от конкурирующих решений способом разработки, также обновление приложения происходит без установки обновлений.

Мобильное приложение позволяет при помощи участия в викторине зарабатывать баллы и выиграть различные призы. В личном кабинете пользователь может увидеть на каком месте он стоит в рейтинг листе, также приложение позволяет пользователю добавлять, изменять и удалять персональные данные в том числе и личную фотографию. В целом в приложении приятный интерфейс и достаточно большое количество функционала. Используя данное приложение, абитуриент будет всегда в курсе всех происходящих событий.

Функциональность приложения может быть расширена добавлением авторизации пользователей через социальные сети и возможностью делиться с друзьями своими достижениями через социальные сети. Также в дальнейшем планируется распространение данного программного продукта на другие вузы.

Conclusion

The result of the work is a mobile application for the applicants of Tomsk State University of Control Systems and Radio Engineering.

In the process of performing this final qualifying work, an analysis was carried out and the necessary requirements that a mobile application for applicants should possess are identified. On the basis of these requirements, a mobile application "TUSUR Applicant" was created, which allows to have information on the list of ratings, news and events at hand, and also to applicants can assess their opportunities for admission to a specific direction.

This application differs from competing solutions in the way of development, also the application update occurs without installing updates.

The mobile application allows you to earn points and win various prizes with the help of participation in the quiz. In the personal account, the user can see where he is in the rating sheet, and the application allows the user to add, change and delete personal data including personal photos. In general, the application has a nice interface and a lot of functionality. Using this application, the entrant will always be aware of all current events.

The functionality of the application can be extended by adding users' authorization through social networks and the ability to share their achievements through social networks with friends. Also, in the future it is planned to distribute this software product to other universities.

Список использованной литературы

1. Gartner Inc. // [Электронный ресурс]. URL: <http://www.gartner.com/newsroom/id/3215217/> (дата обращения: 16.03.18);
2. Mashable. // [Электронный ресурс]. URL: <http://mashable.com/2013/07/24/google-play-1-million/> (дата обращения: 16.03.18).
3. Cross platform App Frameworks // [Электронный ресурс]: Stats and Forecast [Infographic]. URL: <https://www.rishabhsoft.com/blog/cross-platform-app-tools-infographic> (дата обращения: 15.04.2018)
4. Гибридные, нативные и мобильные веб приложения // [Электронный ресурс]: URL: <https://magora-systems.ru/hybrid-vs-mobile-web-vs-native-applications/> (дата обращения: 16.03.18).
5. Интернет 2017-2018 в мире и в России: Статистика и тренды // [Электронный ресурс]: URL: <https://www.web-canape.ru/business/internet-2017-2018-v-mire-i-v-rossii-statistika-i-trendy/> (дата обращения: 16.03.18)
6. Статистика мобильного интернета 2018 // [Электронный ресурс]: URL: <https://sdvv.ru/articles/elektronnaya-kommertsiya/statistika-interneta-2018-sayty-blogi-domeny-elektronnaya-kommertsiya-interesnye-tsifry-i-fakty-so-v/> (дата обращения: 16.03.18)
7. Mobile app versus Mobile Website statistics // [Электронный ресурс]: URL: <https://jmango360.com/wiki/mobile-app-vs-mobile-website-statistics/> (дата обращения: 16.03.18)
8. Официальная страница мобильного приложения «Абитуриент ТПУ» // [Электронный ресурс]. URL: <https://www.mobile.tpu.ru/> (дата обращения: 15.03.18)

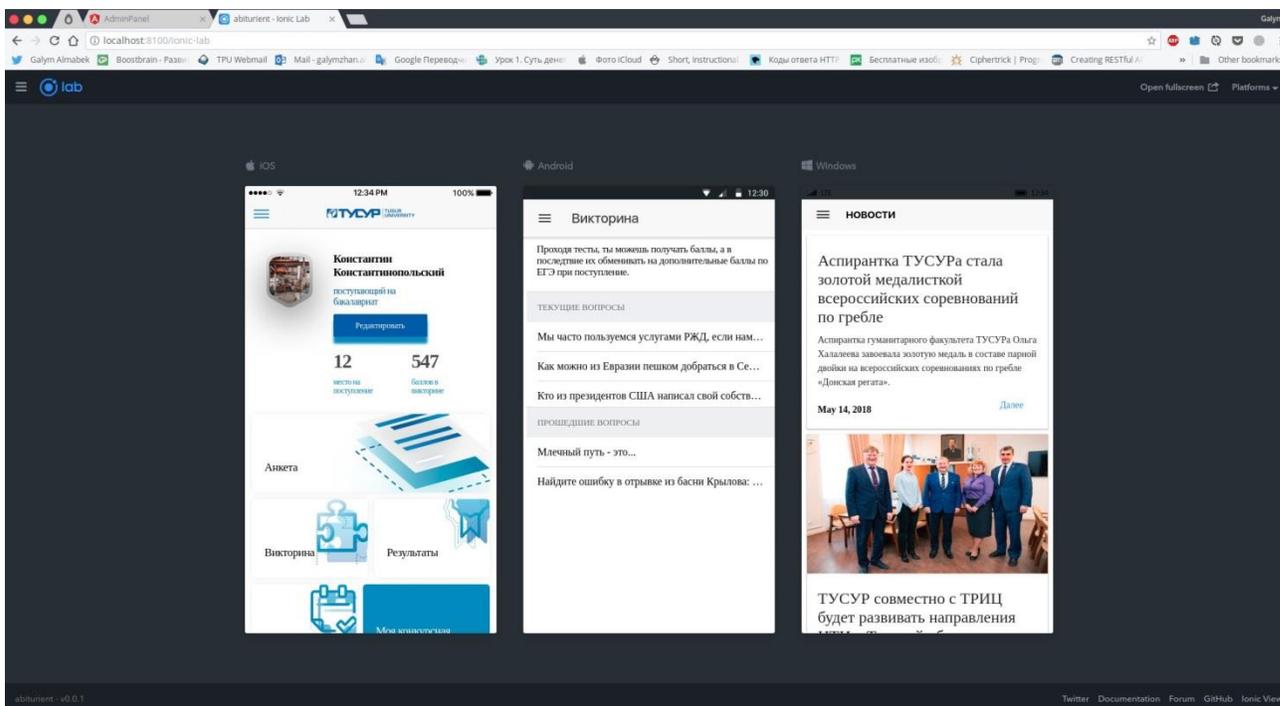
9. Новость о запуске приложения мобильного приложения «Хочу в ТГУ» // [Электронный ресурс]. URL: <http://www.tsu.ru/news/tgu-zapustil-mobilnoe-prilozhenie-dlya-abituriento/> (дата обращения: 15.03.18)
10. Моделирование на UML. Общие диаграммы // [Электронный ресурс]: Моделирование на UML. URL: http://book.uml3.ru/sec_1_5#p7 (дата обращения: 09.05.2018).
11. Использование компонентной архитектуры в веб приложениях // [Электронный ресурс]: URL: <https://fwdays.com/en/event/js-frameworks-day-2015/review/komponentnaia-arkhitektura-v-web-prilozheniia> (дата обращения: 29.03.18).
12. Мобильные приложения: нативные, веб и гибридные // [Электронный ресурс]: URL: <https://agilie.com/ru/blog/mobilnyie-prilozheniia-nativnyie-veb-i-ghibridnyie> (дата обращения: 5.04.18).
13. Официальный сайт, фреймворка Ionic 3 // [Электронный ресурс]: URL: <https://ionicframework.com/> (дата обращения: 20.04.18)
14. Инфраструктура разработки приложения на PhoneGap для iOS и Android // [Электронный ресурс]: Habr. URL: <https://habr.com/company/arcadia/blog/257749/> (дата обращения: 20.04.18)
15. Xamarin: что это такое, кому стоит использовать и ответы на другие важные вопросы о фреймворке // [Электронный ресурс]: Tproger. URL: <https://tproger.ru/articles/xamarin-answers/> (дата обращения: 20.04.18)
16. Native Vs Hybrid Apps // [Электронный ресурс]: URL: <http://www.ijcter.com/papers/volume-2/issue-6/native-apps-vs-hybrid-appsionic-titanium.pdf> (дата обращения: 20.04.18)
17. Ionic framework. Обзор экосистемы // [Электронный ресурс]: Habr. URL: <https://habr.com/company/simpleweek/blog/254681/> (дата обращения: 23.04.18).
18. Нововведения в Cordova 5.0 // [Электронный ресурс]: Habr. URL: <https://habr.com/post/258915/> (дата обращения: 22.04.18)

19. OASIS SOA Reference Model TC | OASIS // [Электронный ресурс]: Advancing open standards for the information society. URL: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm (дата обращения: 09.05.2018).
20. Банокин П.И. Методы и средства проектирования информационных систем и технологий: учебное пособие / П.И. Банокин; Томский политехнический университет. – Томск: Изд-во Томского политехнического университета, 2015. – 92 с.
21. SOAP vs. REST: The Differences and Benefits Between the Two Widely-Used Web Service Communication Protocols // [Электронный ресурс]: URL: <https://stackify.com/soap-vs-rest/> (дата обращения: 28.04.18)
22. NodeJS official documentation. About Node.js // [Электронный ресурс]: URL: <https://nodejs.org/en/about/> (дата обращения: 30.04.18)
23. Express.js official documentation. Basic routing // [Электронный ресурс]: URL: <http://expressjs.com/en/starter/basic-routing.html> (дата обращения: 30.04.18)
24. MongoDB official documentation. Getting Started // [Электронный ресурс]: URL: <https://docs.mongodb.com/> (дата обращения: 30.04.18)
25. Angular. One framework. Mobile & desktop. // [Электронный ресурс]: URL: <https://angular.io/guide/quickstart> (дата обращения: 30.04.18)
26. Manning. Getting MEAN with Mongo, Express, Angular, and Node / Simon Holmes // Adding a dynamic frontend with Angular. – 2016 – part 3.
27. Learn to Build Modern Web Apps with MEAN // [Электронный ресурс]: URL: <https://thinkster.io/tutorials/mean-stack> (дата обращения: 23.04.18)
28. Web API Design: Crafting Interfaces that Developers Love / Brian Mulloy // [Электронный ресурс]: Advancing open standards for the information society. URL: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm (дата обращения: 09.05.2018).

29. Single Page Application (SPA) Using Angular JS, Web API and MVC // [Электронный ресурс]: URL: https://www.c-sharpcorner.com/uploadfile/rahul4_saxena/single-page-application-spa-using-angularjs-web-api-and-m/
30. Ionic in Action: Hybrid Mobile Apps with Ionic and AngularJS (англ) / Jeremy Wilken // Ionic in Action. – Нью-Йорк: 2016. — Vol. 1. — P. 9-36.
31. ГОСТ 12.0.003-74 ССБТ. Опасные и вредные производственные факторы. Классификация.
32. СНиП 23-05-95 Естественное и искусственное освещение.
33. СанПиН 2.2.2/2.4.1340-03 Гигиенические требования к персональным электронно-вычислительным машинам и организации работы.
34. СанПиН 2.2.2. 542-96 «Гигиенические требования к видео дисплейным терминалам, персональным электронно-вычислительным машинам и организации работы».
35. СанПиН 2.2.4.548-96 «Гигиенические требования к микроклимату производственных помещений».
36. СНиП 2.2.4/2.1.8.562-96 «Шум на рабочих местах, в помещениях жилых, общественных зданий и на территории жилой застройки».
37. СанПин 2.2.4.3359-16 «Санитарно–эпидемиологические требования к физическим факторам на рабочих местах».
38. Статья 108 ТК РФ

Приложение А (Обязательное)

Экранные снимки основных окон приложения



Приложение Б (Обязательное)

Исходный код реализации компонента добавления новостей

```
export class AddNewsComponent implements OnInit {
  public alerts: Array<IAlert> = [];
  public showAlert: Boolean = false;
  news: any;
  public froalaContent: string = '';
  previewImage: any;
  labelText: string = "Загрузить изображение";

  edtContent($event : string){
    this.froalaContent = $event;
  }

  constructor(
    private http: HttpClient,
    private dataservice: DataService,
    private router: Router,
  ) {

    this.alerts.push({
      id: 1,
      type: "success",
      message: "Новость добавлена! Вы будете перенаправлены на страницу
      новостей..."
    });

  }

  previewFile() {
    let file = (document.querySelector('.upload_file') as
    HTMLInputElement).files[0];
    const fileReader: FileReader = new FileReader();
    this.labelText = file.name;

    fileReader.onloadend = (event : Event) => {
      let previewImage = fileReader.result;
      this.previewImage = previewImage;
    }
    fileReader.readAsDataURL(file);
  }

  saveNews(title, newsPreview) {
    this.dataservice.saveNews(hash(title), title, this.previewImage,
    newsPreview, this.froalaContent);
    this.http.get("/api/news").subscribe(data => {
      this.news = data;
      setTimeout(() => {
        this.router.navigate(['/news']);
      }, 3000);
    });
  }

  public closeAlert(alert: IAlert) {
    const index: number = this.alerts.indexOf(alert);
    this.alerts.splice(index, 1);
  }

  onClickMe(){
    this.showAlert = true;
    setTimeout(
```

```
function() {  
  this.showAlert = false;  
}.bind(this),  
3000);  
}
```

Приложение В (Обязательное)

Исходный код реализации страницы авторизации

```
export class LoginPage {
  data: any;
  loading: Loading;
  registerCredentials = { email: '', password: '' };

  constructor(
    public navCtrl: NavController,
    public navParams: NavParams,
    public auth: AuthServiceProvider,
    private alertCtrl: AlertController,
    private loadingCtrl: LoadingController
  ) {
  }

  public createAccount() {
    this.navCtrl.push( RegisterPage );
  }

  public login() {
    this.showLoading()
    console.log(this.registerCredentials);
    this.auth.login(this.registerCredentials).subscribe(allowed => {
      if (allowed) {
        this.navCtrl.setRoot( ProfilePage );
      } else {
        this.showError("данные не правильны!");
        console.log(allowed);
      }
    },
    error => {
      this.showError(error);
    });
  }

  showLoading() {
    this.loading = this.loadingCtrl.create({
      content: 'Подождите...',
      dismissOnPageChange: true
    });
    this.loading.present();
  }

  showError(text) {
    this.loading.dismiss();

    let alert = this.alertCtrl.create({
      title: 'Ошибка',
      subTitle: text,
      buttons: ['OK']
    });
    alert.present(alert);
  }
}
```

Приложение Г (Обязательное)

Исходный код реализации авторизации и регистрации пользователя

```
export class AuthServiceProvider {

  data: any;
  message: "";
  currentUser: User;
  rootUrl: string = 'http://localhost:8080/api';

  constructor(
    public jwtHelper: JwtHelperService,
    public _http: HttpClient,
    public http: Http
  ) { }

  public isAuthenticated(): boolean {
    const token = localStorage.getItem('jwtToken');
    return !this.jwtHelper.isTokenExpired(token);
  }

  public login(credentials) {
    if (credentials.email === null || credentials.password === null) {
      return Observable.throw("Пожалуйста введите данные");
    } else {
      return Observable.create(observer => {
        this._http.post(`${this.rootUrl}/signin`, {
          email: credentials.email, password:
credentials.password
        }).subscribe(res => {
          this.data = res;
          localStorage.setItem('jwtToken',
this.data.token);
          observer.next(true);
          observer.complete();
        }, err => {
          this.message = err;
        });
      });
    }
  }
}
```

```

        observer.next(false);
        observer.complete();
    });
});
}
}

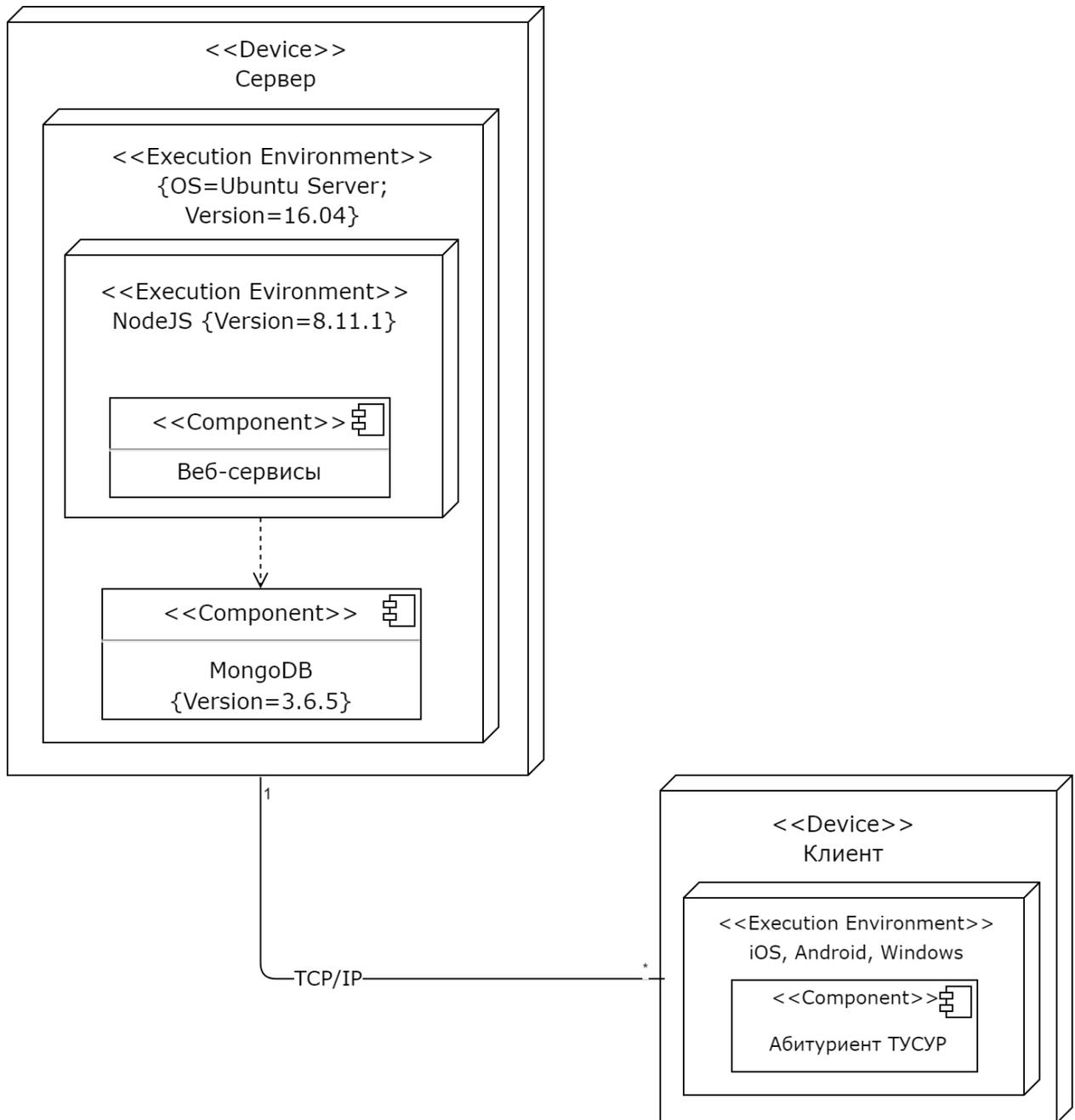
public register(credentials) {
    if (credentials.email.toLowerCase().trim() === null ||
        credentials.password.trim() === null) {
        return Observable.throw("Пожалуйста введите данные");
    } else {
        return Observable.create(observer => {
            this.http.post(`${this.rootUrl}/signup`, {
                name: credentials.name.toLowerCase().trim(),
                fullname: credentials.fullname.toLowerCase().trim(),
                email:
                credentials.email.toLowerCase().trim(), password: credentials.password.trim(),
            })
                .map(res => res.json())
                .subscribe( data => {
                    console.log(data);
                });
            observer.next(true);
            observer.complete();
        });
    }
}

public getUserInfo() : User {
    return this.currentUser;
}
}
}

```

Приложение Д (Обязательное)

UML диаграмма развертывания программного продукта



**Приложение Е
(Обязательное)**

Designing Mobile Applications

Студент:

Группа	ФИО	Подпись	Дата
8ВМ6Г	Алмабек Галымжан Маратулы		

Консультант ИШИТР отделения информационных технологий

Должность	ФИО	Ученая степень, звание	Подпись	Дата
доцент	Лунёва Елена Евгеньевна	К.Т.Н		

Консультант – лингвист отделения иностранных языков ШБИП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
	Кудряшова Александра Владимировна	Старший преподаватель		

1.1 Conceptual model of mobile application

To describe system's operation, it is necessary to build a conceptual model of developing system. Such a model should be adequate for the subject matter; therefore, it must contain the knowledge of all system processes' participants.

First, we will select the actors who will interact with the software:

- Program administrator is an expert who will manage all the content in the system;
- User – basically they are applicants who going to enter the university, but in the future students can also use the application.

Let's also highlight the main precedents:

View news and events

Perpetrator: user. Take place in the application.

Description of steps:

1. Authorize in the system,
2. Go to the news or events module using navigation menu,
3. Choose interesting news or event.

Participation in the quiz

Perpetrator: user. Take place in the application.

Description of steps:

1. Authorize in the system,
2. Go to quiz module using the navigation menu,
3. Choose an unanswered question,
4. Answer the question, earn points.

Add news or events

Perpetrator: administrator. Take place in the administrative panel.

Description of steps:

1. Authorize in the system,
2. Go to the events or news module using navigation menu,
3. Switch to adding news or events,
4. Arrange the news or event as needed and add it.

The interaction of actors with the software and the description of use cases is graphically presented in Figure 1.1 in the form of a use case diagram.

Use case diagram – is a diagram that reflects the relationship between actors and use cases and is an integral part of the use case model that allows you to describe the system at the conceptual level.

The precedent is the probability of a simulated concept (the proportion of its functionality), due to which the user is able to acquire a certain, measurable and necessary result. The fact corresponds to a single service concept, establishes the only one with alternatives to its application and represents the classic method of interaction between the user and the system. Usage options are usually used to specify the external conditions to the concept.

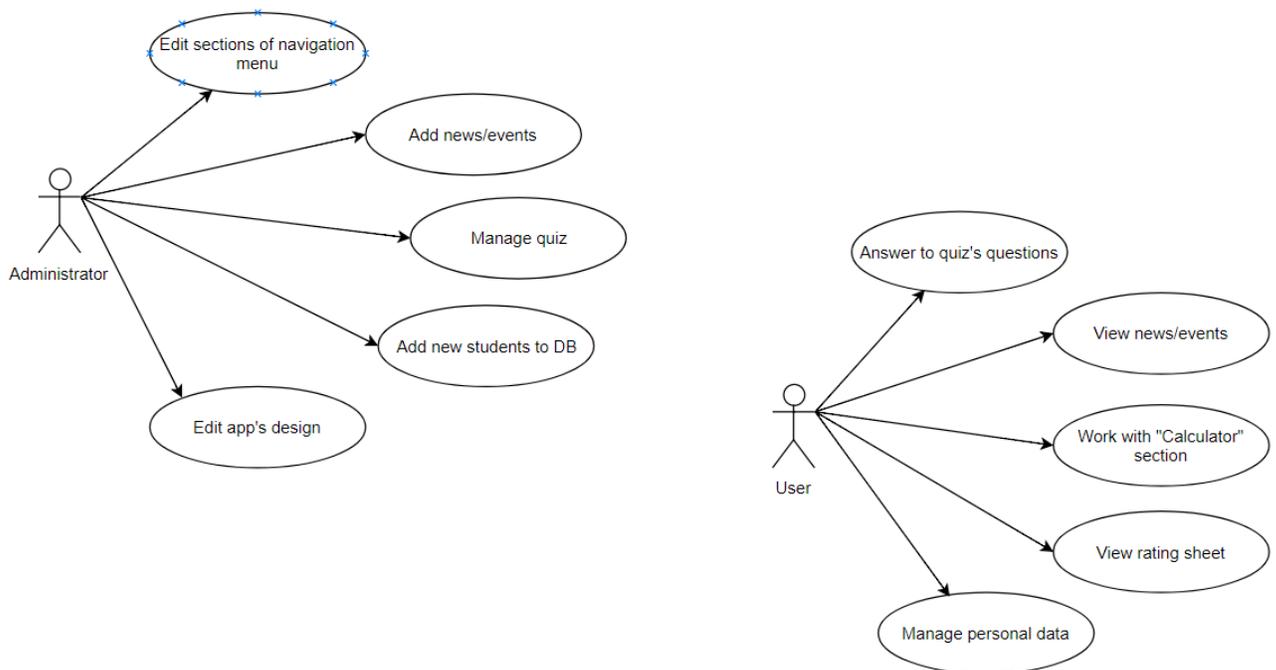


Figure 1.1 – Use case diagram of a software

The user's response to quiz questions can be presented in the form of an activity diagram (Figure 1.2).

Activity Diagram - The UML diagram, which shows the actions in states, is described in the state diagram. Activity is defined as the specification of the executable behavior in a coherent and parallel coordinated form for the performance of subordinate elements: nested actions and individual actions. Actions connected with each other by flows that go from the outputs of one node to the inputs of the other.

Activity diagrams are used in modeling business processes, technological processes, parallel and parallel systems.

Activity diagrams consist of a limited number of figures connected by arrows. The main figures:

- Rectangles with curvatures – actions,
- Rhombuses – solutions,
- Wide bands – the beginning (branch) and the end (convergence) of the branch actions,
- The black circle is the beginning of the process,
- The black circle with stroke is the end of process.

Arrows go from the beginning to the end of the process and show control flows or streams of objects (data).

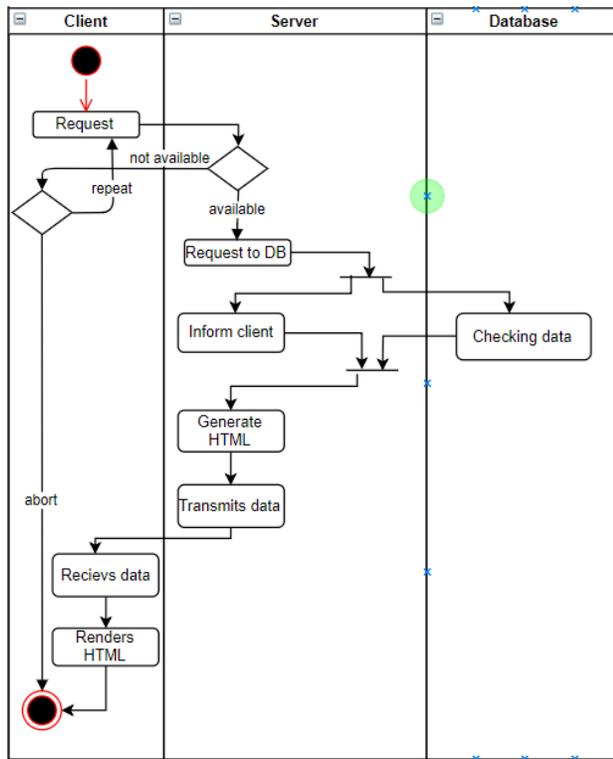


Figure 1.2 – Activity diagram of the process “Answer to the quiz”

1.2 Definition of general requirements for the software

In this section, the requirements for the software will be defined. Requirements from the application administrator:

- The software must allow new sections to be entered in the navigation menu;
- The software must be able to work with different data;
- All changes must be made through the administration panel;
- The software must store all data in the database;
- The software must be cross-platform;

Define the requirements for the user:

- All changes must be automatically available for the application;
- The software should allow viewing the rating sheet;

- The software should allow viewing news and events;
- The software must allow the editing of personal data;
- The UI must have a clear interface;
- The software should allow answers to quiz questions;

Functional requirements for the software

Based on the above requirements, we define the list of functions of the program being developed:

- Ability to create news and events;
- Ability to create a quiz;
- Ability to create new pages for the application;
- Ability to create new sections in the application;
- Ability to edit users;
- Ability to delete users;
- Verify the correctness of the information entered.

1.3 Analysis of modern means of developing hybrid mobile applications

1.3.1 Overview of mobile application development methods

The question that often occurs when developing an application is whether to create a native mobile application, a hybrid application, or make your site adaptive for mobile devices.

Native applications are designed to work in a specific mobile environment. They will not work on different mobile OS. For example, if you plan to create your own application for iOS, you must create it in Swift.

Hybrid applications are cross-platform applications that work equally well on different platforms. It is designed using a specific standard encoding language (for example, JavaScript, CSS and HTML), and then organized to work on different platforms.

Individual interactions associated with various devices are often achieved through the use of plug-ins for each operating system.

An important advantage of using a hybrid application is that it allows the application to run on different OSes without large expenses. If the development of the application is done correctly, users will not be able to tell if the application is built using a hybrid or native route.

Next, review these two development approaches (Table 1.1).

	Native	Hybrid
Technology	Objective-C, iOS, SDK, Java, Android SDK	HTML, CSS, JavaScript, фреймворк для мобильной разработки
Spread	AppStore/PlayMarket	AppStore/PlayMarket
Development speed	Slow	High
Development cost	High	Low
Cost of support	High	Low
Performance	High	Average
Access to native functions	Yes	Yes
Better to use	In all applications where performance is needed	In applications that do not have high performance requirements.

During the review of possible solutions to this problem, it was decided that the best solution would be to develop a hybrid application, since we are developing an application for applicants that does not require great performance and work with the native functionality of the mobile device.

Several other factors were:

- rapid development;
- less development costs;

- less support costs.

After it was decided to develop a cross-platform, hybrid application, it is necessary to choose which platform to use for these purposes.

1.3.2 Platform analysis for developing hybrid mobile application

In 2018, the market of cross-platform applications reached \$ 7.5 billion, and the volume of tools for developing mobile applications for cross-platform applications is growing. So, what are the best platforms, resources and tools for encoding for iOS, Android, Windows?

Native applications have the advantage, but a well-made cross-platform mobile application will make the differences insignificant and will have the advantage that users on multiple platforms will have access to your product or service.

Cross-platform mobile development has long been a viable alternative to a fully native application. Following the classical native approach, you can create two different applications: one will be written in Java for Android, and the other on Swift or Objective-C for iOS. This path is quite laborious and expensive, because the same code should be written twice. On the other hand, it ensures maximum code consistency with the original appearance of the application.

Cross-platform development, on the contrary, allows developers to write code once and apply some of its parts on all platforms. This is usually due to the performance and behavior of the application.

Next, consider the use of cross-platform platforms Ionic, PhoneGap, Xamarin and Titanium to create mobile applications. The following comparative characteristics will be taken into account:

- language stack;
- graphical user interface;
- community;

– price.

Let's compare 6 solutions for cross-platform development.

Cross-platform frameworks Ionic 3, Xamarin, Unity, Qt and Appcelerator Titanium, Telerik Platform currently occupy 80% of the cross-platform development market for mobile devices.

Ionic is an open source project for developing hybrid applications built on Angular JS and Apache Cordova. The tools and services needed to develop cross-platform applications are provided using well-known web technologies such as CSS, HTML5 and SaaS.

Xamarin is a framework used to develop cross-platform applications. It is widely used to communicate with the application programming interface (API) of native functions of mobile devices, such as contacts, camera and geolocation for Android, iOS and Windows operating systems. This allows developers to use almost 100% native Android and iOS libraries, Xamarin is best suited for building applications using the .NET Common Language Infrastructure(CLI) programming language C #.

Unity is a multi-platform tool for developing 2D and 3D applications and Unity games and is also one of the best tools for demonstrating 3D content. Unity-based applications run under Windows, OS X, Linux, Android, Apple iOS, Windows Phone, BlackBerry, and Wii, PlayStation 3 and Xbox 360 game consoles.

Titanium is the framework for mobile applications with open source, created by Appcelerator. The structure is coded in such a way that it offers an integrated environment for developers to develop a wide range of mobile applications. Based on JavaScript, it also entails HTML5, CSS3, and JQuery.

Telerik AppBuilder - one of the main reasons to use AppBuilder is a full-fledged online IDE. It allows you to create, test and even publish hybrid applications from any computer or mobile device, without the need to download it.

Consider the main advantages of the above platforms in Table 1.2.

Table 1.2 – Comparative analysis of platforms

	Ionic 3	Xamarin	Unity	Appcelerator Titanium	Telerik Builder
Technology	JavaScript, HTML5, CSS3	C#, Xaml	C#, Unity Script, Boo	JavaScript, Python, Ruby, PHP	.NET, JavaScript, HTML5, Java, PHP
Platforms	iOS, Android, Windows	iOS, Android, Windows	Android, iOS, Windows Phone, Tizen, PS 4, Xbox One	iOS, Android, Windows, Tizen, Denso	iOS, Android, BlackBerry, Windows, Windows Phone
Cost	Price paid: from 29\$ Free version: available Adobe Creative Cloud Membership: available	Xamarin, Visual Studio Community: free Visual Studio Professional and Enterprise: available	Personal Edition: free Professional Edition: or 75 \$ per month	Free period Indie: 39\$ per month Pro: \$99 per month	Free period From 39\$ per month
Cross platform portability	Over 98% of the source code can be reused	Business logic, data access, network connection	Not completely	Not completely	Not completely
Open source	+	-	-	+	-
UI	Web	Native	UI Canvas	Native	Web
Community	Huge	Huge	Average	Small	Small

With the help of Table 2.2 it was determined that the best choice for our development would be Ionic 3. But at the time of development, such technology as React Native was gaining popularity. It was decided to compare the Ionic 3 platform with this technology.

- Ionic uses web technologies to improve portability, whereas React Native uses a functional interface;
- The language used by Ionic is TypeScript, and React Native uses JSX;
- Ionic uses a simple method of learning such as pre-created components; on the other hand, React Native can be studied once and written on any platform;

- Ionic provides code reuse for Android, iOS, web applications, desktop applications, while React Native provides reuse of code for Android, iOS applications;
- Ionic code performance can be seen in Web View, and React Native does not have such a web view;
- In Ionic testing can be performed in any browser, and React Native uses emulators and real mobile devices;
- In Ionic, Apache Cordova is used to access the hardware of the phone, and React Native has the ability to access the phone's hardware on its own;
- Ionic and React Native have a huge community;
- Documentation in Ionic is very simple, transparent and correct, whereas React Native does not have very complete documentation.

In the above comparison, it was decided to leave since the strategically important aspects of the tasks did not improve in any way, in some cases Ionic even surpassed React Native.

Further, we will take a deeper look at the Ionic ecosystem.

1.3.3 Overview of the Ionic 3 framework

At its core, the Ionic Framework is an addition to the very popular Apache Cordova framework, but with its powerful CLI (Command Line Interface) and extensive documentation. Following the principles of Apache Cordova, applications on the Ionic Framework are hybrid HTML applications. Such applications on the phone run in a special shell (UIWebView for iOS and WebView for Android), which allows you to display HTML and perform JavaScript. Accordingly, when working in the application, the user works as if in a web browser. There are several components for the Cordova application. The following diagram shows a high-level view of the Cordova application architecture (Figure 1.3).

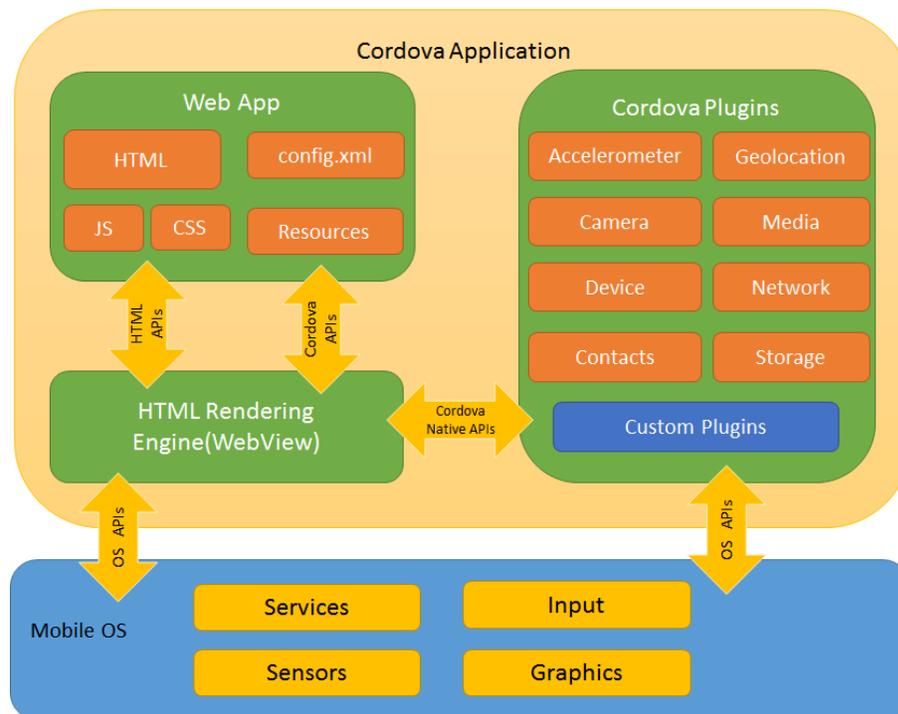


Figure 1.3 – Apache Cordova application architecture

Apache Cordova - a kind of layer between the user interface and device resources. Any widgets of the user interface or the MV* (Model - View) structure are not included in its structure. If you want to use the widgets of the user interface and/or the MV* framework, you must select and include them in the application yourself, as third-party resources. Ionic 3 is one of the structures that provides widgets for the user interface. The MVC model for it is provided by Angular 2+.

1.4 Software architecture

To start development, it was necessary to determine the stack of technologies with the help of which the interface and server part will be developed.

During the discussion with the project manager, it was decided to use the MEAN stack. One of the main reasons was that the above Ionic technology uses Angular 2+,

which would speed up the development speed. Next, we'll take a closer look at the MEAN stack.

Usually, in web development, different programming languages are used at all levels of the stack. The MongoDB database, the Express and AngularJS frameworks, and the server language Node.js together form the MEAN stack - a powerful platform, at all levels of which only one language is used: JavaScript. The MEAN stack is attractive for developers and business because of its simplicity and economy, and end users love MEAN applications for their speed and responsiveness. Next, we will consider in more detail why each technology is used from the above stack.

The transition from a traditional database, such as MySQL, to an incomprehensible, document-oriented NoSQL repository, such as MongoDB, represents a fundamental shift in persistence strategy. The programmer spends less time writing SQL statements and more - writing the map / JavaScript functions. This eliminates the huge layers of transformation logic, since MongoDB initially produces the JavaScript Object Notation (JSON) format. As a result, writing of REST web services is extremely easy.

Node.js is the server-side JavaScript runtime environment. This is the platform built into the V8 JavaScript runtime of Google Chrome. This helps quickly create scalable and parallel applications.

Express is a lightweight framework used to create Web applications in the Node. It provides a number of reliable functions for creating single- and multi-page web applications. Structure Express is borrowed from such popular languages as Ruby and Sinatra.

MongoDB - without schema NoSQL database. MongoDB stores the data in a binary JSON format, which simplifies the transfer of data between the client and the server.

Angular is a JavaScript framework developed by Google. It provides some useful functions, such as two-way data binding. This is a complete solution for the rapid development of the interface.

Next, consider the method of communicating applications with the server. Most of the applications that currently exist must be connected to the Internet for proper operation.

API means "application program interface". More specifically, it allows developers to access the platform or application of the other party.

When it comes to the architecture of the client server, the client and the server are completely independent of each other. The frontend component is the client, and the backend service is the server.

The RESTful API can also be cached. This means that the client has the ability to store all the responses in the cache. As a result, the API has improved performance.

When processing requests is processed, the data is not stored on the server. Instead, the data is stateless, which means that the session is stored on the client side.

RESTful APIs are commonly called RESTful web services, because they implement the principles of REST, as well as HTTP protocols.

Basically, it covers a lot of resources using HTTP methods, and then is presented in a standard format, which is usually XML. While hypertext is standard, it works for valid types of Internet media. Some examples are shown in Figure 1.4.

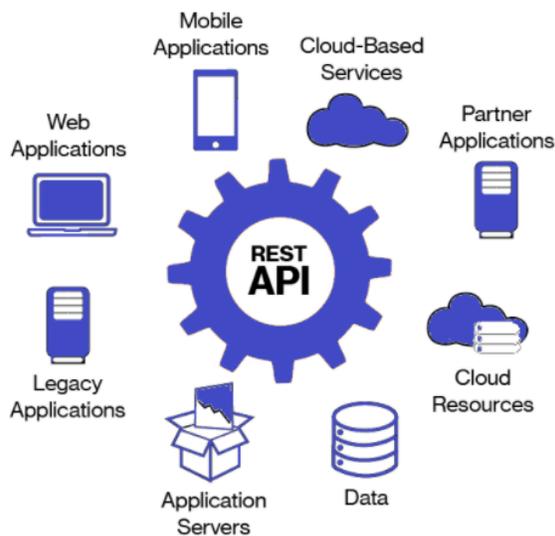


Figure 1.4 – Visual representation of REST

Next, it was necessary to define an HTTP method that will use our REST-service, because even with the same name, but with different methods, REST-services perform completely different actions.

REST uses 4 basic HTTP methods: GET, POST, PUT, DELETE. In most cases, each of the methods is used to perform a predefined action from the CRUD (creating, reading, updating, deleting - "creating, reading, updating, deleting").

POST - creation, GET - reading, PUT - update, DELETE - deletion.

Since the application will use the database and transfer data, we will start building the architecture using the part that we definitely need. Figure 1.5 shows the initial form, the REST API, built using Express and Node.js, to provide interaction with the MongoDB database.

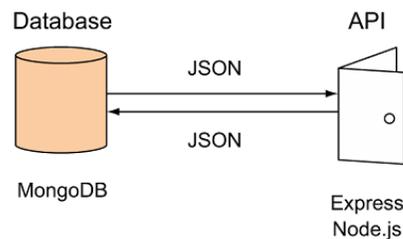


Figure 1.5 – Standard MEAN architecture of the REST API

This software product includes two parts. This is a user-oriented interface that gives readers articles, news, and more. The content management system of the mobile application also has an administrator interface, in which administrators enter to write new articles and manage the application. Some of the main characteristics of these two parts are shown in Figure 1.6.

Mobile application



Characteristics:

- Content Saturation
- Weak interactivity
- Quick Startup
- Short user work time

Administrative interface



Characteristics

- Saturation with features
- High interactivity
- Fast response to actions
- Long user work time
- Lack of general accessibility

Figure 1.6 – Conflicting characteristics of two parts of the content management system of the application

As shown in Figure 1.6, there is a high level of conflict between the characteristics of these two parts. On the one hand, you need a content-rich and weakly interactive environment for news, events, and so on, and a rich, highly-interactive environment for the administrative interface.

In this case, the architecture of the two applications will be considered separately:

- User-friendly interface,
- An interactive non-public administrative interface.

1.4.1 Interface of administrative panel: SPA and AngularJS

For the administrative interface, the SPA, built on the basis of AngularJS [4], would be ideally suited. So, the architecture of this part of the software product will look very familiar: the REST API built on MongoDB, Express and Node.js, with a one-page application on AngularJS in the interface part. Figure 2.7 shows how this will look.

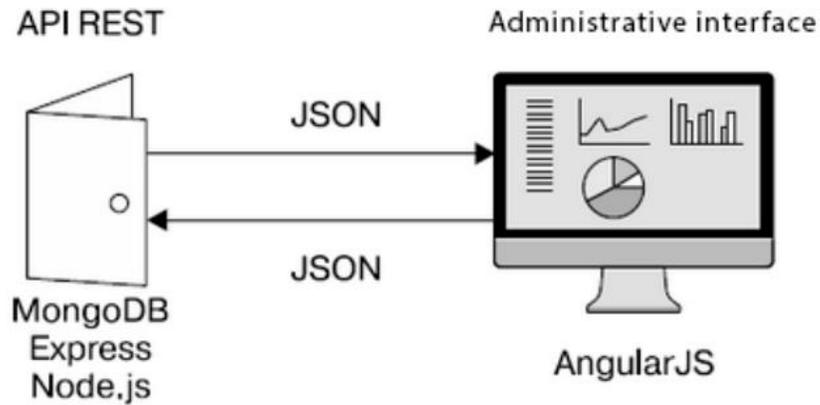


Figure 2.7 – The administrative interface is a SPA based on AngularJS, using the REST API built on MongoDB, Express and NodeJS

1.4.2 Interface of mobile application: Express framework

In the case of a mobile application, delivering HTML code and content directly from the server is exactly what will be needed. Express from the outset provides a choice of templating. HTML content will require data from the database, using the REST API for this. The basis of this architecture is shown in Figure 1.8.

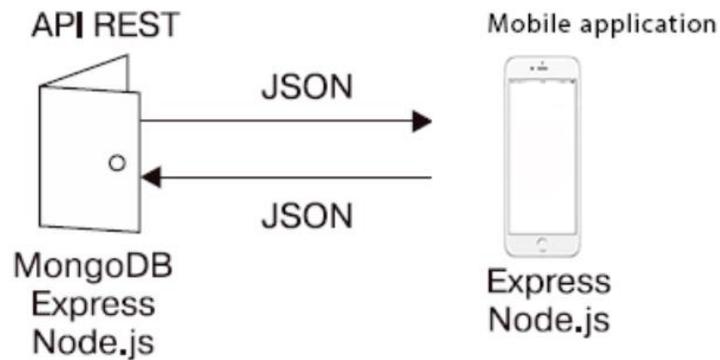


Figure 1.8 – Architecture of delivering HTML tags from server

As shown in Figure 1.8 on the front of the application, there are Express and NodeJS interacting with the REST API built on MongoDB, Express and NodeJS.

Also, to this architecture, you can add an AngularJS to add dynamics to the application, for example, to display related messages or a search field with auto-completion.

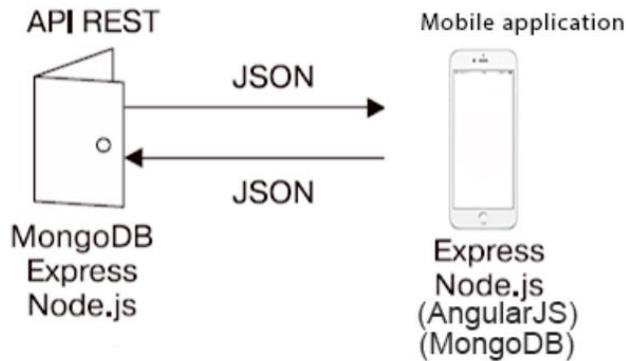


Figure 1.9 – Adding AngularJS and MongoDB usage parameters as part of the interface aspect

1.4.3 Hybrid software architecture

As a result, we get two separately running applications that use the REST API. If you plan a little, it can be a common REST API, applied by both parts of the application. The integrity of the architecture with one REST API, interacting with two client-side applications, demonstrates Figure 1.10.

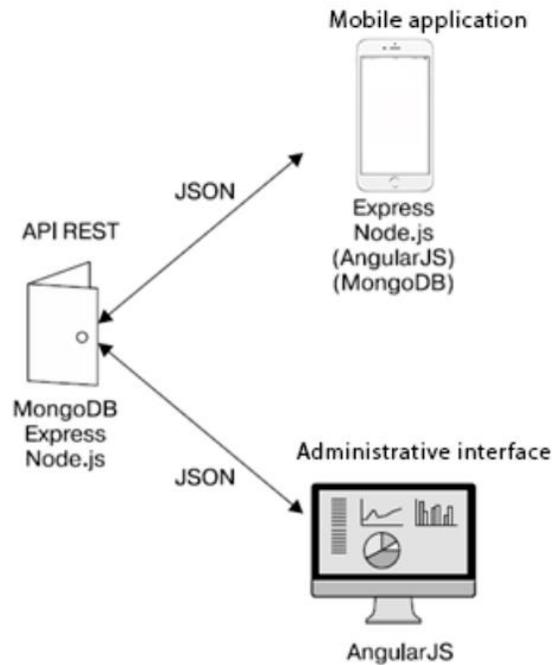


Figure 1.10 – Hybrid architecture of MEAN application

As shown in Figure 1.10, for the sake of providing the most appropriate solution, one REST API delivers data to two user-oriented applications based on various parts of the MEAN technology suite.

1.4.4 User Authentication

Json Web Token (JWT) technology was used to authorize, register and protect routs.

JSON Web Token (JWT) is an open standard (RFC 7519), which defines a compact and standalone method for securely transferring information between parties as a JSON object. This information can be verified because it has a digital signature. JWT can be signed using a secret (with the HMAC algorithm) or an open / private key pair using RSA.

Next, consider the principle of the JWT. Whenever a user wants to access a protected route or resource, the user agent must send the JWT, usually in the authorization header, using the Bearer schema. The contents of the header should look like this:

```
Authorization: Bearer <token>
```

This is a non-state authentication mechanism, because the user state is never stored in the server's memory. Secure server routes will verify the validity of JWT in the authorization header and, if present, the user will be allowed access to protected resources. Because JWTs are autonomous, all the necessary information exists, which reduces the need for the database query several times.

This allows you to completely rely on API data that does not have a stateless and even make requests to the top-down services. No matter which domains are served by your APIs, Cross-Origin Resource Sharing (CORS) will not be a problem, because it does not use cookies.

The following Figure 1.11 shows this process.

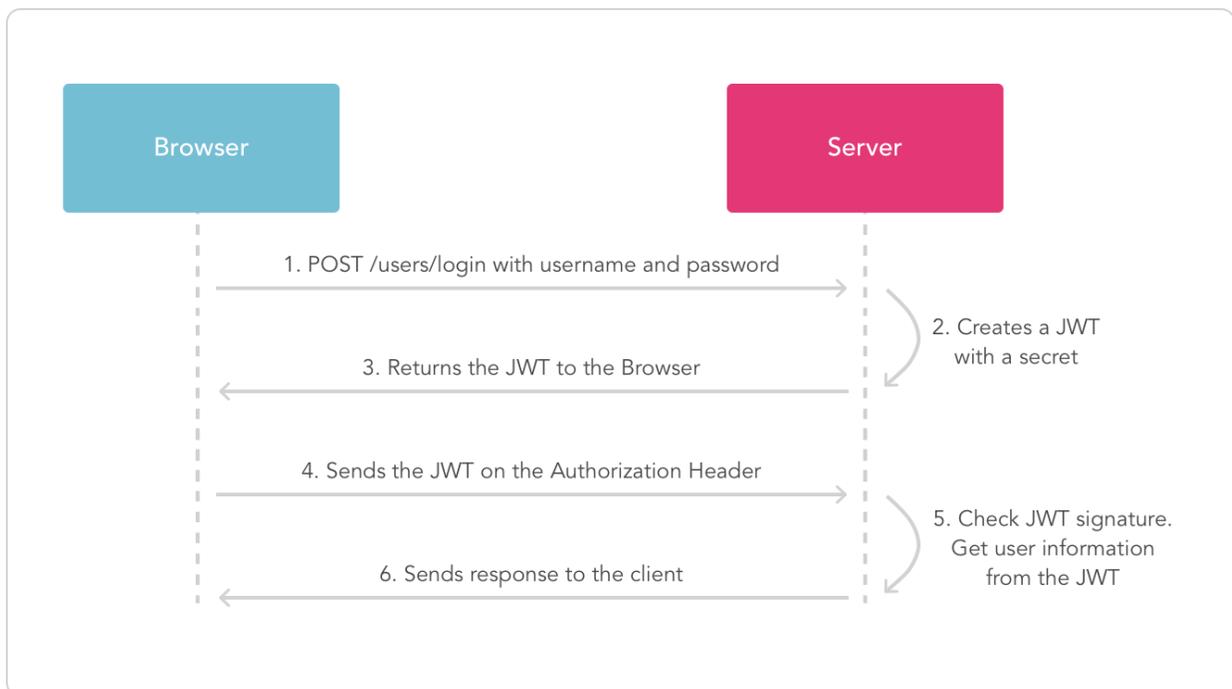


Figure 1.11 - JWT workflow