МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет	*Кафедра*
вычислительной техники	МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

БАКАЛАВРСКАЯ РАБОТА

на тему

Мобильное приложение для молодежной общероссийской общественной организации «Российские студенческие отряды»

Студент _____ Колокольцев Кирилл Алексеевич
(подпись, дата) (ФИО полностью)

Руководитель _____ Прошкина Е.Н.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____№_____)

Заведующий кафедрой _____ Макарычев П.П.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____№____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2020

Федеральное государственное бюджетное образовательное учреждение высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет	*Кафедра*
Вычислительной техники	МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____П.П. Макарычев
«__»_____2020 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент_____Колокольцев Кирилл Алексеевич__гр.__16ВП1_факультета__ВТ__
<Фамилия, имя, отчество полностью>

2.Тема работы__Мобильное приложение для молодежной общероссийской общественной организации «Российские студенческие отряды»_____
Тема утверждена приказом ПГУ №_1488/о_от "_30_"___декабря____2019___

3. Руководитель работы____к.т.н., доцент кафедры «МОиПЭВМ» Прошкина Е.Н.____

4. Задание на работу (назначение разработки, исходные данные и т.п.)

1) *Разработать Мобильное приложение для молодежной общероссийской общественной организации «Российские студенческие отряды»*

2) Среда выполнения: Android 5.0 и выше

Языковые средства разработки: C#, UML

Инструментальные средства разработки: Visual Studio Xamarin

Технологии разработки: ООП

3) Основные функции приложения

а) обеспечить пользователю возможность обмениваться сообщениями с другими пользователями или группами пользователей

б) обеспечить возможность создания мероприятий, а также участвовать в них и в их организации

в) обеспечить возможность посещения информационной базы приложения, где можно ознакомиться с историей и законодательными документами

5. Перечень подлежащих разработке вопросов

1) Анализ предметной области и постановка задачи

2) Проектирование

3) Реализация

4) Контроль качества программного обеспечения

5) Оформление пояснительной записки

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	*Анализ предметной области и постановка задачи*	10%	15.03.2020–20.03.2020	
2	*Проектирование*	25%	20.03. 2020–30.03.2020	
3	*Реализация*	30%	30.03. 2020–30.04. 2020	
4	*Контроль качества программного обеспечения*	5%	30.04. 2020–10.05. 2020	
5	*Оформление пояснительной записки*	10%	10.05. 2020–20.05. 2020	

Дата выдачи задания "30" декабря 2019

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 104 листа, 48 рисунков, 16 использованных источников и 45 таблиц.

ООП, XAMARIN, ANDROID, UML, MVC, C#, MS SQL SERVER.

Объект разработки – программные средства поддержки корпоративного общения, а также организации мероприятий.

Целью выпускной квалификационной работы является разработка мобильного приложения на платформе Android, предоставляющего удобный и быстрый способ корпоративного общения и организации мероприятий.

Технология разработки – ООП, MVC.

В результате проведенной работы создано мобильное приложение, позволяющее пользователям обмениваться сообщениями, просматривать и организовывать мероприятия, а также предоставляющее базу знаний о движении «Российские студенческие отряды».

Разработка программного обеспечения проводилась на языке программирования C# в среде Visual Studio с дополнением Xamarin, в операционной системе Windows 10. В качестве хранилища данных был использован MS SQL Server.

Содержание

Введение 6
1 Постановка задачи и анализ предметной области 7
1.1 Анализ программного обеспечения аналогичного назначения 7
1.2 Постановка задачи 13
1.3 Анализ требований 14
1.3.1 Требования к интерфейсу пользователя. 14
1.3.2 Требования к программным средствам 15
2 Проектирование мобильного приложения 22
2.1 Проектирование структуры приложения 22
2.1.1 Проектирование классов представлений 22
2.1.2 Проектирование классов контроллеров 25
2.1.3 Проектирование структур данных 26
2.2 Проектирование интерфейса пользователя 33
2.3 Календарное планирование и оценка бюджета 38
3 Реализация мобильного приложения 48
3.1 Реализация интерфейса пользователя 48
3.2 Реализация классов представлений 52
3.3 Реализация классов контроллеров 56
3.4 Реализация структур данных 58
3.5 Развертывание приложения 62
4 Тестирование мобильного приложения 65
4.1 Функциональное тестирование 65
4.2 Структурное тестирование 66
4.3 UI-тестирование 70
Заключение 74
Список использованных источников 75
Приложение А. Исходный код приложения 77
Приложение Б. Результаты тестирования 97

Введение

В век стремительного развития IT-технологий очень сложно представить свою жизнь без какого-либо гаджета под рукой: компьютер, ноутбук, телефон, даже часы уже могут служить некой информационной системой. Доля общения в сети давно превзошла долю встреч вживую. Сейчас практически любую задачу можно решить благодаря интернету. Работа по сети стала уже не новой, и работодатели с удовольствием используют онлайн сервисы для связи с коллегами. В различных социальных сетях у многих компаний есть диалоги, в которых обсуждаются последние рабочие новости. Это довольно удобно, использовать такие системы не только для общения с друзьями, но и для решения каких-либо рабочих проблем. Такой подход не только повышает продуктивность, но и зачастую уменьшает затраты времени на выполнение определенных задач. Однако, все же есть и минусы использования социальных сетей в корпоративных целях. К примеру, в таких социальных сетях очень много избыточной информации, которая будет сбивать внимание на себя, отвлекать работника.

Организация «Российские студенческие отряды» не исключение. Главный ресурс данной организации – люди, а значит, ускорив процесс передачи информации между ними, будет в разы упрощено выполнение любой задачи. Этим и обуславливается актуальность создания тематического корпоративного мобильного приложения.

1 Постановка задачи и анализ предметной области

1.1 Анализ программного обеспечения аналогичного назначения

В 21 веке, веке информационных технологий каждый человек так или иначе взаимодействует с различными онлайн-сервисами, в том числе с социальными сетями. Социальные сети – это не только возможность личного общения с другими людьми, но и распространенный инструмент организации корпоративной работы. Различные беседы, в которых состоит большинство работников той или иной организации, поддерживают возможность передачи не только текстовых сообщений, но и передачи каких-либо документов, фотографий и видео. С течением времени большинство таких социальных сетей перешли к более новому и простому для использования виду – мобильному приложению.

В настоящее время настоящими лидерами в сфере социальных сетей являются: Вконтакте, Instagram.

Вконтакте – социальная сеть, созданная 10 октября 2006 года, стремительно набравшая пользователей и продолжающая распространяться и по сей день. Данная платформа предлагает своим пользователям множество возможностей, помимо простого общения – передача фото, видео, возможность совершения видео и аудио звонков. На данной платформе находится множество игр и музыки [1]. Но, несмотря на все это, можно выделить так же и минусы работы Вконтакте. К примеру, если использовать данную платформу для работы, то очень мало информации находится в сети об определенном человеке. Конечно, если человек захочет, он может расписать все свои увлечения и т.д., но, прочитав страницу пользователя, нельзя точно определить, где и в какой должности он работает и работает ли. Конечно, зачастую это никому и не нужно. Но, если мы рассматриваем Вконтакте как платформу для организации рабочей связи коллег, связи с начальниками, то, будь досье человека более информативное, работа организовывалась бы проще. К тому же данная система содержит крайне

мало информации о Российских студенческих отрядах, структуре, рассматриваемой в рамках данного проекта. К тому же платформа не рассматривает ведущей функцией планирования мероприятий, что для работы в системе РСО критично.

Instagram – это социальная сеть, она так же поддерживает передачу текстовых сообщений, но в основном направлена именно на распространение каких-либо фото и видео материалов. Для организации корпоративной работы практически не подходит из-за слабого ориентирования на обмен текстовой информации. К тому же в данной системе очень много отвлекающих факторов, таких, как постоянная и неконтролируемая реклама в новостной ленте. Из очень больших плюсов можно выделить удобство использования и просмотра фотографий человека.

Приложение ТрудКрут – вопросно-ответная система, посвященная определенному мероприятию, а именно 60-летию движения РСО. Из плюсов можно выбрать хорошую освещенность направления студенческих отрядов, однако для каких-то корпоративных манипулирований приложение не подходит. Это связано с тем, что диалоговая система практически не реализована – сообщение можно отправить только в общий чат, никаких личных диалогов либо бесед с несколькими участниками в данном приложении не предусмотрено.

Для сравнения платформ-аналогов с разрабатываемым в рамках выпускного проекта приложением «RSOnline» будет произведено сравнение вышеуказанных систем методом анализа иерархий [2].

Для проведения сравнения были выделены следующие критерии:

– функциональность чата;
– информационно-тематическая освещенность;
– организация мероприятий;
– UX/UI дизайн [3].

Матрицы попарных сравнений будут строиться на основе стандартного разделения баллов, а именно:

– равно, безразличие = 1;

– немного лучше (хуже) = 3 (1/3);

– лучше (хуже) = 5 (1/5);

– значительно лучше (хуже) = 7 (1/7);

– принципиально лучше (хуже) 9 (1/9).

При промежуточном мнении будут использоваться промежуточные баллы 2, 4, 6, 8.

Рассмотрим матрицу попарных сравнений критериев (таблица 1).

Таблица 1 – Матрица попарных сравнений критериев

	Функциональность чата	Информационно-тематическая освещенность	Организация мероприятий	UX/UI-дизайн
Функциональность чата	1	1/4	1/2	1/6
Информационно-тематическая освещенность	4	1	7	2
Организация мероприятий	2	1/7	1	1/5
UX/UI-дизайн	6	1/2	5	1

На таблице 2 представлена матрица попарных сравнений по критерию «Функциональность чата».

Таблица 2 – Матрица попарных сравнений по критерию «Функциональность чата»

Функциональность чата	«Вконтакте»	«Instagram»	«ТрудКрут»	«RSOnline»
«Вконтакте»	1	1/5	1/9	1/3
«Instagram»	5	1	1/6	4
«ТрудКрут»	9	6	1	8
«RSOnline»	3	1/4	1/8	1

Матрица попарных сравнений по критерию «Информационно-тематическая освещенность» показана ниже (таблица 3).

Таблица 3 – Матрица попарных сравнений по критерию «Информационно-тематическая освещенность»

Информационно-тематическая освещенность	«Вконтакте»	«Instagram»	«ТрудКрут»	«RSOnline»
«Вконтакте»	1	1/5	6	7
«Instagram»	5	1	7	9
«ТрудКрут»	1/6	1/7	1	5
«RSOnline»	1/7	1/9	1/5	1

Таблица 4 содержит результат попарных сравнений по критерию «Организация мероприятий».

Таблица 4 – Матрица попарных сравнений по критерию «Организация мероприятий»

Организация мероприятий	«Вконтакте»	«Instagram»	«ТрудКрут»	«RSOnline»
«Вконтакте»	1	1/5	1/2	2
«Instagram»	5	1	3	4
«ТрудКрут»	2	1/3	1	3
«RSOnline»	1/2	1/4	1/3	1

Матрица попарных сравнений по критерию «UX/UI-дизайн» представлена в таблице 5.

Таблица 5 – Матрица попарных сравнений по критерию «UX/UI-дизайн»

UX/UI-дизайн	«Вконтакте»	«Instagram»	«ТрудКрут»	«RSOnline»
«Вконтакте»	1	1/4	1/8	1/2
«Instagram»	4	1	1/6	1
«ТрудКрут»	8	6	1	7
«RSOnline»	2	1	1/7	1

После создания матриц попарных сравнений всех программ-аналогов по всем критериям была проведена нормировка матрицы попарных сравнений (таблица 6).

Таблица 6 – Весовые доли критериев

	Вес в долях	Вес в процентах
Функциональность чата	0,367	36,7%
Информационно-тематическая освещенность	0,149	14,9%
Организация мероприятий	0,381	38,1%
UX/UI-дизайн	0,103	10,3%

Из результата произведенных выше расчетов видно, что наибольший вес имеют критерии «Организация мероприятий» и «Функциональность чата» соответственно. Остальные 2 критерия не такие весомые – 14,9% критерий «Информационно-тематическая освещенность» и всего 10,3% UX/UI-дизайн.

Рассмотрим весовые доли в соответствии с критерием «Функциональность чата» (таблица 7).

Таблица 7 – Весовые доли рассматриваемых приложений в соответствии с критерием «Функциональность чата»

Функциональность чата	Вес в долях	Вес в процентах
«Вконтакте»	0,448	44,8%
«Instagram»	0,185	18,5%
«ТрудКрут»	0,035	3,5%
«RSOnline»	0,32	32%

Весовые доли аналогичных программ в соответствии с критерием «Информационно-тематическая освещенность» представлены в таблице 8.

Таблица 8 – Весовые доли аналогичных программ в соответствии с критерием «Информационно-тематическая освещенность»

Информационно-тематическая освещенность	Вес в долях	Вес в процентах
«Вконтакте»	0,142	14,2%
«Instagram»	0,032	3,2%
«ТрудКрут»	0,328	32,8%
«RSOnline»	0,498	49,8%

Ниже представлены весовые доли в соответствии с критерием «Организация мероприятий» (таблица 9).

Таблица 9 – Весовые доли аналогичных программ в соответствии с критерием «Организация мероприятий»

Организация мероприятий	Вес в долях	Вес в процентах
«Вконтакте»	0,338	33,8%
«Instagram»	0,072	7,2%
«ТрудКрут»	0,192	19,2%
«RSOnline»	0,398	39,8%

Весовые доли аналогичных программ в соответствии с критерием «UX/UI-дизайн» представлены в таблице 10.

Таблица 10 – Весовые доли аналогичных программ в соответствии с критерием «UX/UI-дизайн»

UX/UI-дизайн	Вес в долях	Вес в процентах
«Вконтакте»	0,44	44%
«Instagram»	0,24	24%
«ТрудКрут»	0,04	4%
«RSOnline»	0,27	28%

Данные всех приведенных выше таблиц были для удобства рассмотрения собраны в одну (таблица 11).

Таблица 11 – Матрица весов альтернативных платформ по всем критериям

	Функциональность чата	Информационно-тематическая освещенность	Организация мероприятий	UX/UI-дизайн
«Вконтакте»	0,448	0,142	0,338	0,44
«Instagram»	0,185	0,032	0,072	0,24
«ТрудКрут»	0,035	0,328	0,192	0,04
«RSOnline»	0,32	0,498	0,398	0,27

Для того чтобы найти лучшее по всем критериям приложение, умножим матрицу из таблицы «Матрица весов альтернативных платформ по всем критериям» на матрицу из таблицы «Весовые доли критериев». Результат приведен ниже в таблице 12.

Таблица 12 – Веса приложений-аналогов

	Вес в долях	Вес в процентах
«Вконтакте»	0,36	36%
«Instagram»	0,12	12%
«ТрудКрут»	0,14	14%
«RSOnline»	0,38	38%

Как видно из результата анализа методом иерархий – разрабатываемое в рамках выпускного проекта приложение «RSOnline» будет лучше по 2 выбранным критериям, один из которых является ведущим. Если рассматривать аналоги с учетом совокупности выбранных критериев, то «RSOnline» и «Вконтакте» имеют практически одинаковый вес, однако разрабатываемое приложение на 2% лучше.

1.2 Постановка задачи

В рамках дипломного проектирования должно быть разработано мобильное приложение молодежной общероссийской общественной организации «Российские студенческие отряды». Так как данное приложение будет работать как социальная сеть, оно должно иметь возможность:

– организации общения между двумя людьми;
– организации общения между группой людей (создание беседы);
– организации передачи различных документов, картинок, видео;
– разбиения пользователей на группы, в зависимости от принадлежности человека к отряду.

Данное приложение также может рассматриваться как некая информационная система, соответствующая данной тематике РСО. В ней должна быть возможность узнать что-то новое, поэтому в этой программе будет содержаться следующая информация:

– история РСО;
– законодательные документы, по которым существует система РСО;
– история отдельных отрядов;

– обучающая информация по работе в различных направления.

Так как каждому члену системы РСО при регистрации выдается определенный документ, то в приложении при создании аккаунта будет создаваться и некий «паспорт» пользователя, в него будут входить:

– ФИО;

– дата рождения;

– номер членского билета (документ, выданный при регистрации в системе РСО).

При завершении регистрации можно также добавить дополнительную информацию о себе:

– срок вступления в отряд;

– роли в отряде (если роль изменялась, то роли по годам);

– объекты, где работал пользователь;

– посещенные всероссийские и окружные мероприятия;

– список организованных мероприятий.

Так как помимо прочего участники РСО занимаются организацией различных мероприятий, то это будет так же отражаться в приложении. Будет добавлена возможность создания этого мероприятия в приложении, привлечение, как организаторов, так и посетителей.

Разработка данного приложения будет проводиться в среде программирования Visual Studio 2019 с дополнением Xamarin [4] на языке программирования C#.

1.3 Анализ требований

1.3.1 Требования к интерфейсу пользователя.

Интерфейс данного мобильного приложения должен иметь заметные и понятные элементы управления. Основные требования к интерфейсу:

– функциональность;

– понятность;

– логичность;

– адаптивность;

– обеспечение высокой скорости работы пользователя.

Так как основным выбранным прототипом является «Вконтакте», то и интерфейс пользователя будет во многом схож с вышеназванной платформой. Для удобства основная навигационная деятельность будет проводиться с помощью выдвигающегося бокового меню. Данное решение было взято в одной из версий интерфейса главного приложения-аналога, так как получило положительный отзыв и стало практически «привычным» для большинства пользователей мобильных приложений.

1.3.2 Требования к программным средствам

Постановка задачи для данного мобильного приложения позволяет выделить основные варианты использования (рисунок 1) [5].

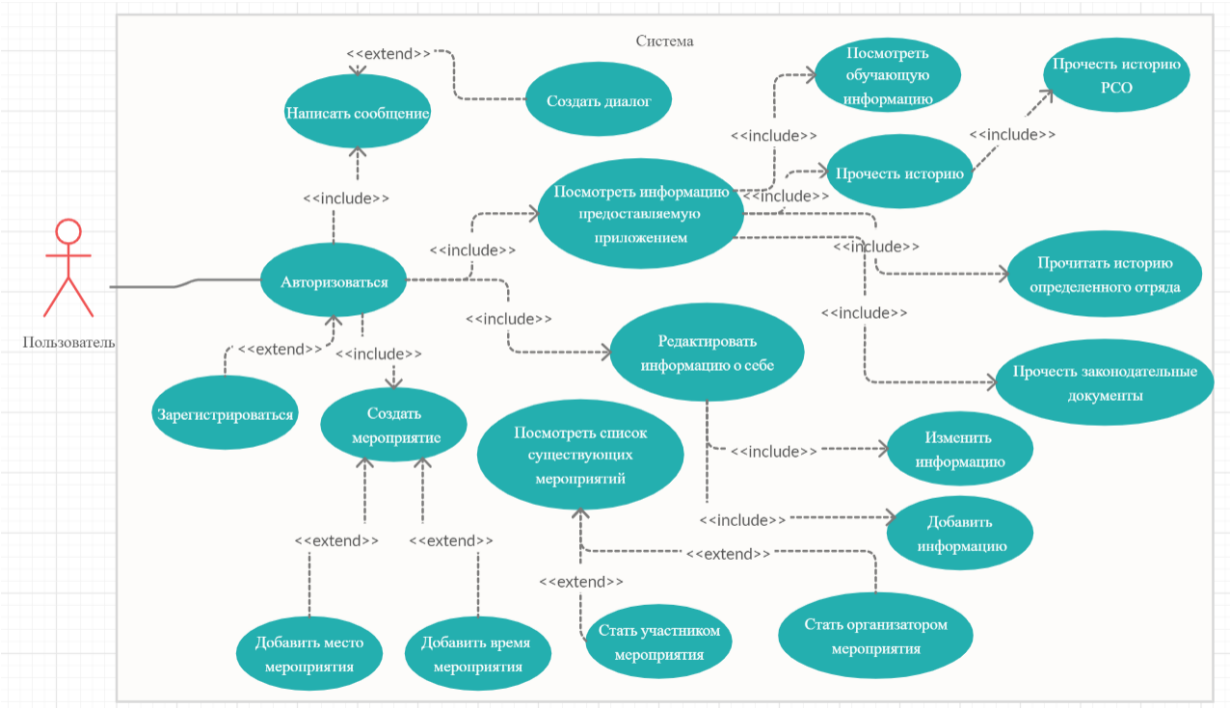

Рисунок 1 – Диаграмма вариантов использования

Опишем спецификацию прецедентов. Рассмотрим спецификацию прецедента «Зарегистрироваться» (рисунок 2).

Прецедент: Зарегистрироваться
ID: 1
Краткое описание: Пользователь регистрируется в системе
Действующие лица: Пользователь, система
Предусловия: Пользователь не зарегистрирован в системе
Основной поток: 1. Пользователь открывает приложение и нажимает кнопку регистрации 2. Пользователь вводит обязательные данные и нажимает кнопку «Регистрация»
Постусловия: -

Рисунок 2 – Спецификация прецедента «Зарегистрироваться»

Данная спецификация описана для варианта использования «Зарегистрироваться», который предназначен для таких случаев, когда пользователь впервые заходит в приложение и еще не имеет аккаунта в системе. Позволяет занести свои данные в систему, чтобы в дальнейшем сразу авторизоваться.

Рассмотрим спецификацию прецедента «Авторизоваться» (рисунок 3).

Прецедент: Авторизоваться
ID: 2
Краткое описание: Пользователь авторизуется в системе
Действующие лица: Пользователь, система
Предусловия: Пользователь зарегистрирован в системе
Основной поток: 1. Пользователь открывает приложение и вводит логин и пароль 2. Пользователь нажимает кнопку «Войти»
Постусловия: -

Рисунок 3 – Спецификация прецедента «Авторизоваться»

Данная спецификация описана для варианта использования «Авторизоваться», которая позволяет, если пользователь уже зарегистрирован, зайти в приложение.

Рассмотрим прецедент «Написать сообщение» (рисунок 4).

Прецедент: Написать сообщение
ID: 3
Краткое описание: Пользователь пишет сообщение кому-либо
Действующие лица: Пользователь, система, другой пользователь
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь выбирает контакт и открывает с ним диалог 2. Пользователь вводит сообщение и нажимает кнопку отправить
Постусловия: -

Рисунок 4 – Спецификация прецедента «Написать сообщение»

Данная спецификация описана для варианта использования «Написать сообщение», который позволяет вести диалог с другим пользователем, посредством передачи друг другу сообщений.

Рассмотрим спецификацию прецедента «Создать диалог» (рисунок 5).

Прецедент: Создать диалог
ID: 4
Краткое описание: Пользователь создает диалог на несколько пользователей(>2)
Действующие лица: Пользователь, система, другие пользователи
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь нажимает кнопку «Создать диалог» 2. Пользователь выбирает каких пользователей хочет добавить в диалог
Постусловия: -

Рисунок 5 – Спецификация прецедента «Создать диалог»

Данная спецификация описана для варианта использования «Создать диалог», который позволяет создать диалог из нескольких людей, дает возможность отправлять сообщение сразу группе пользователей.

На следующем рисунке будет рассмотрен вариант использования «Создать мероприятие» (рисунок 6).

Прецедент: Создать мероприятие
ID: 5
Краткое описание: Пользователь создает мероприятие
Действующие лица: Пользователь, система
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь нажимает кнопку «Добавить мероприятие» 2. Пользователь по желанию добавляет время мероприятия 3. Пользователь по желанию добавляет место мероприятия 4. Нажимает кнопку «Создать»
Постусловия: -

Рисунок 6 – Спецификация прецедента «Создать мероприятие»

Данная спецификация описана для варианта использования «Создать мероприятие», который позволяет создавать «Мероприятие» и тем самым искать организаторов и посетителей. Дает возможность доработать его с другими пользователями.

Рассмотрим спецификацию прецедента «Посмотреть список мероприятий» (рисунок 7).

Прецедент: Посмотреть список мероприятий
ID: 6
Краткое описание: Пользователь просматривает список мероприятий
Действующие лица: Пользователь, система
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь нажимает кнопку «Мероприятия» 2. Открывается окно со списком мероприятий
Постусловия: -

Рисунок 7 – Спецификация прецедента «Посмотреть список мероприятий»

Данная спецификация описана для варианта использования «Посмотреть список мероприятий».

Рассмотрим спецификацию прецедента «Стать организатором мероприятия» (рисунок 8).

Прецедент: Стать организатором мероприятия
ID: 6
Краткое описание: Пользователь подает заявку на участие в роли организатора
Действующие лица: Пользователь, система
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь нажимает кнопку «Мероприятия» 2. Выбирает из списка мероприятие 3. Переходит на страницу описания мероприятия 4. Нажимает кнопку «Стать организатором»
Постусловия: -

Рисунок 8 – Спецификация прецедента «Стать организатором мероприятия»

Данная спецификация описана для варианта использования «Стать организатором мероприятия». Позволяет пользователю стать организатором мероприятия.

Рассмотрим спецификацию прецедента «Редактировать информацию о себе» (рисунок 9).

Прецедент: Редактировать информацию о себе
ID: 6
Краткое описание: Пользователь редактирует свою информацию
Действующие лица: Пользователь, система
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь нажимает кнопку «Редактировать информацию» 2. Пользователь добавляет какую-либо информацию о себе 3. Пользователь удаляет какую-либо информацию о себе 4. Нажимает кнопку «Сохранить»
Постусловия: -

Рисунок 9 – Спецификация прецедента «Редактировать информацию о себе»

Данная спецификация описана для варианта использования «редактировать информацию о себе», позволяет более полно описать себя,

чтобы другие пользователи явно понимали, кто пользуется данным аккаунтом.

Рассмотрим спецификацию прецедента «Посмотреть информацию, предоставляемую приложением» (рисунок 10).

Прецедент: Посмотреть информацию, представляемую приложением
ID: 7
Краткое описание: Пользователь открывает информационную базу приложения
Действующие лица: Пользователь, система
Предусловия: Пользователь авторизован в системе
Основной поток: 1. Пользователь открывает информационную базу приложения 2. Пользователь выбирает, какая именно информация его интересует a. История i. История РСО ii. История определенного отряда b. Информация о рабочих объектах i. Информация о работе отрядов на определенном объекте c. Посмотреть обучающую информацию
Постусловия: -

Рисунок 10 – Спецификация прецедента «Посмотреть информацию, представляемую приложением»

Данная спецификация описана для варианта использования «Посмотреть информацию, представляемую приложением», позволяет познакомиться с историей отрядов и движения в целом, а также узнать законодательную часть системы «РСО».

Рассмотрим диаграмму последовательности [6] для варианта использования «Стать организатором мероприятия» (рисунок 11).

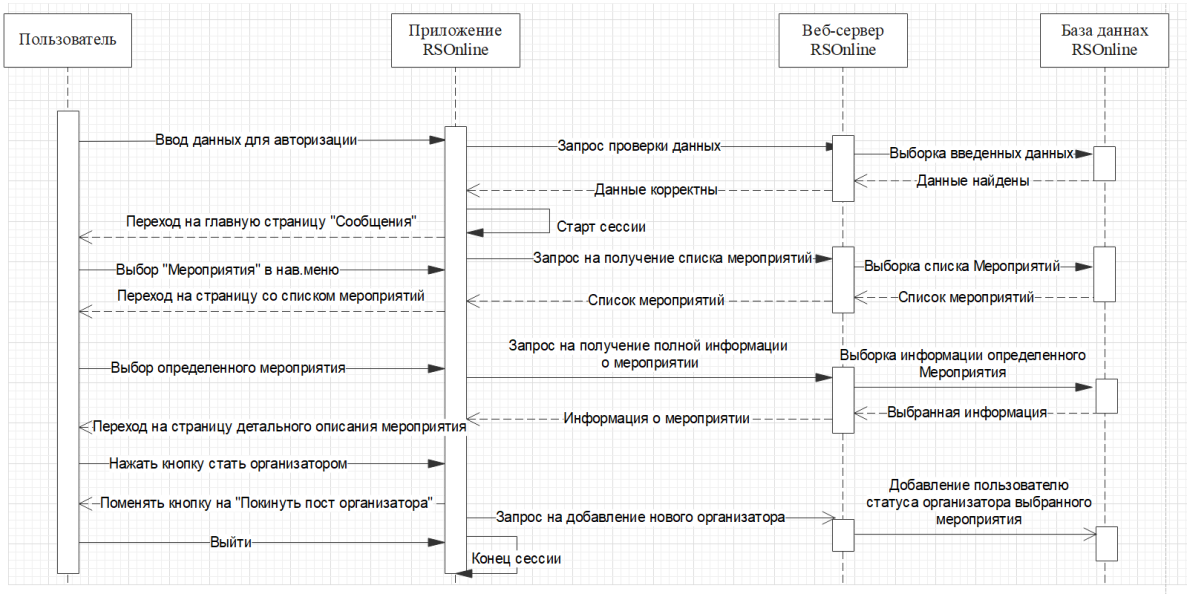

Рисунок 11 – Диаграмма последовательности для варианта использования «Стать организатором мероприятия»

На данной диаграмме видно весь спектр действий различных участников работы приложения RSOnline, а также видны все возможные связи элементов приложения. Из приведенных данных следует, что само приложение RSOnline никогда не будет иметь никаких отношений с базой данных, а эту роль выполняет сервер. При анализе данной информации начинает прослеживаться архитектура приложения, используемая при реализации проекта.

Таким образом, в данной главе был проведен анализ приложений-аналогов, был проанализирован пользовательский интерфейс, и выделены некоторые решения. Также была построена и проанализирована диаграмма вариантов использования.

2 Проектирование мобильного приложения

2.1 Проектирование структуры приложения

2.1.1 Проектирование классов представлений

Разрабатываемое приложение будет иметь архитектуру MVC(Model-View-Controller) [7,8].

Данная архитектура позволяет разделить данные приложения на 3 уровня:

– модель;
– контроллер;
– представление.

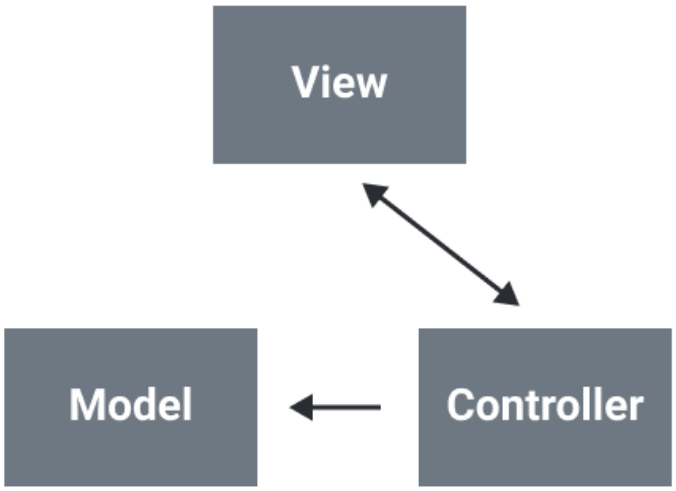

Рисунок 12 – Архитектура MVC

Рассмотрим на диаграмме деятельности вариант использования «Отправить сообщение», где явно прослеживается архитектура проектируемого приложения (рисунок 13).

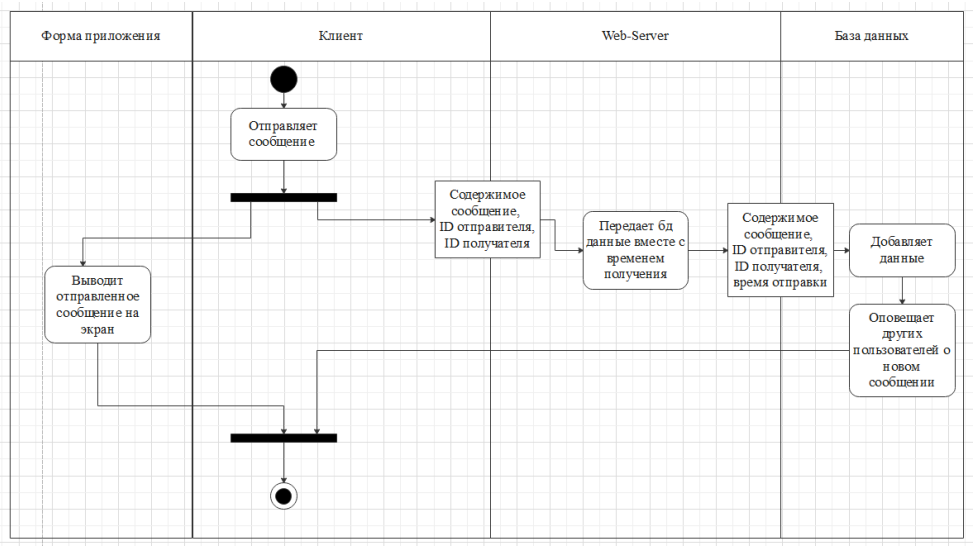

Рисунок 13 – Диаграмма деятельности варианта использования «отправить сообщение»

Роль представления здесь играет форма приложения, а именно окно Activity «Чат». Именно она генерирует все то, что видит пользователь.

Web-Server является контроллером, он обрабатывает все действия пользователя и отвечает на них, работает с базой данных.

Модель – это база данных, хранит в себе все данные приложения.

Рассмотрим диаграмму деятельности варианта использования «Посмотреть список мероприятий» (рисунок 14).

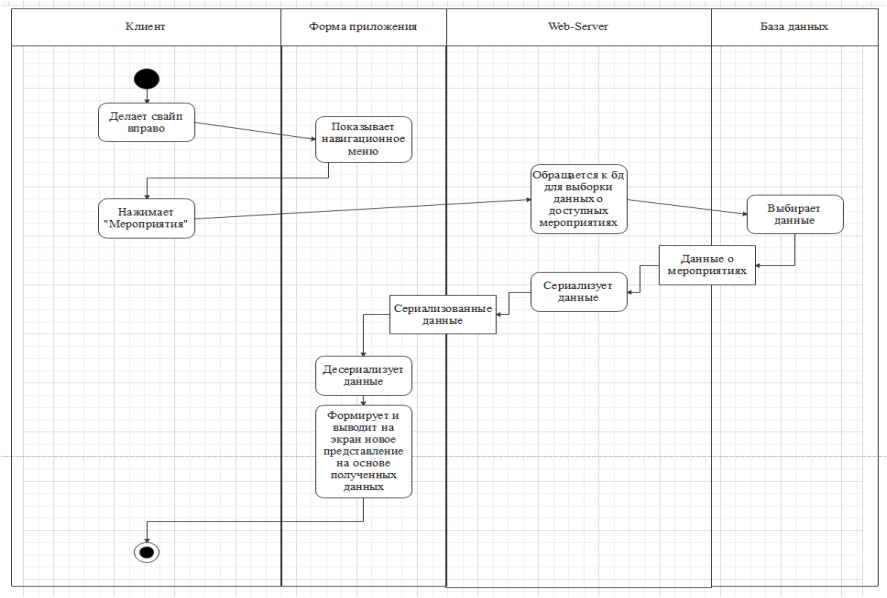

Рисунок 14 – Диаграмма деятельности варианта использования «Посмотреть список мероприятий»

Вышеприведенная диаграмма показывает, как и на основе каких данных создаются представления при переходе со страницы на страницу. Видно, что при передаче данных от контроллера к представлению они сериализуются. Это сделано для того, чтобы уменьшить вес ответов от сервера и увеличить скорость создания представлений соответственно, так как объемы передаваемых данных могут быть велики.

В данном подразделе будут спроектированы классы для работы представлений(View).

У дополнения Xamarin среды Visual Studio есть функция перетаскивания элементов представления непосредственно на экран, однако данный инструмент имеет недостатки. Поэтому для каждого представления был создан класс.

В ходе проектирования мобильного приложения, была построена диаграмма классов представлений (рисунок 15).

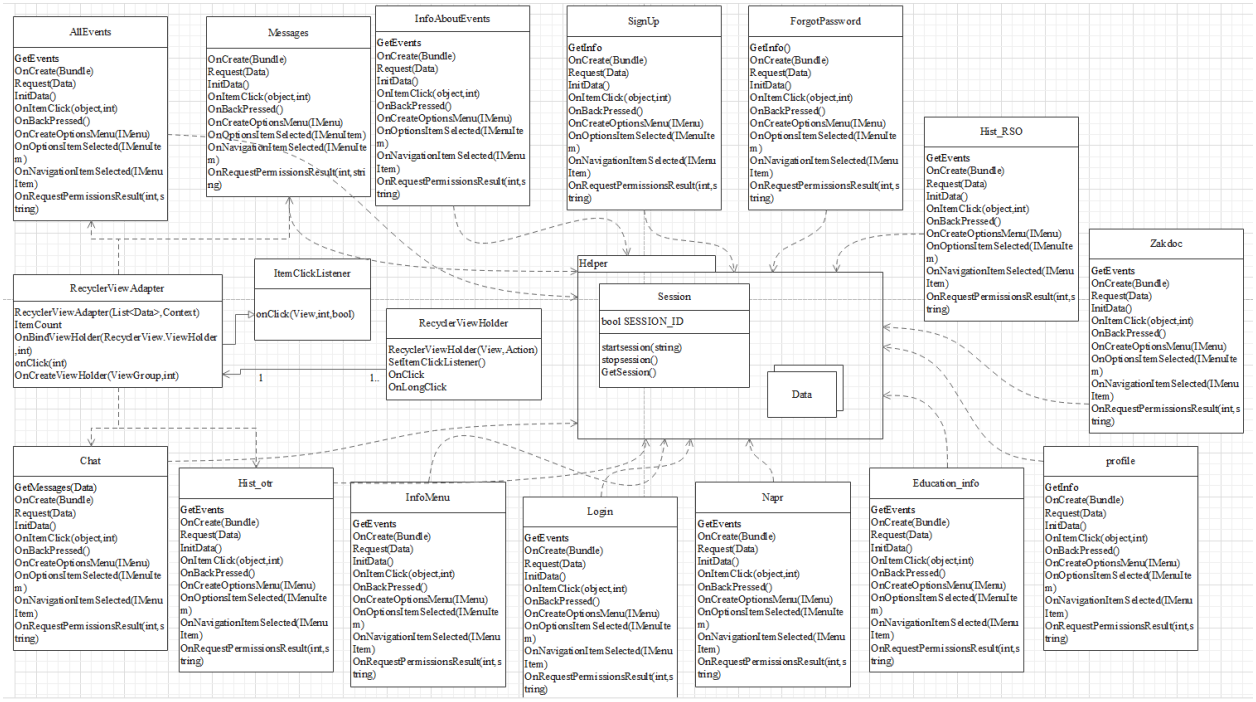

Рисунок 15 – Диаграмма классов представления

На получившейся диаграмме видно, какие классы были использованы для построения представления, а также какие методы они реализуют. Так, например, все классы используют каталог «Helper», где лежат

вспомогательные классы – Session и группа классов «Data», где находятся все структуры, используемые в работе приложения.

Классы – RecyclerViewAdater, RecyclerViewHolder отвечают за организацию цикличного вывода похожих записей, эта функция реализуется только в классах – Chat, Messages,Hist_otr, AllEvents. Эти циклично созданные записи будут в последующем заполняться данными из базы данных приложения. ItemClickListener отвечает за возможность нажатия на записи, позволяя переходить на страницу выбранной записи.

Помимо того, все классы, предназначенные для работы с окнами Activity, будут наследоваться от класса подключенной библиотеки Android.Support.V7.App. AppCompatActivity и реализовывать интерфейс NavigationView.IonNavigationItemSelectedListener. Благодаря подключенной библиотеке Android.Support.Design.Widget и реализации вышеназванного интерфейса классы реализуют левое выдвижное навигационное меню и верхнее вспомогательное меню.

2.1.2 Проектирование классов контроллеров

Так как проектируемое приложение будет иметь 4 функциональных раздела, то классы контроллеры так же поделены на 4 различных класса контроллеров. Рассмотрим диаграмму классов (рисунок 16).

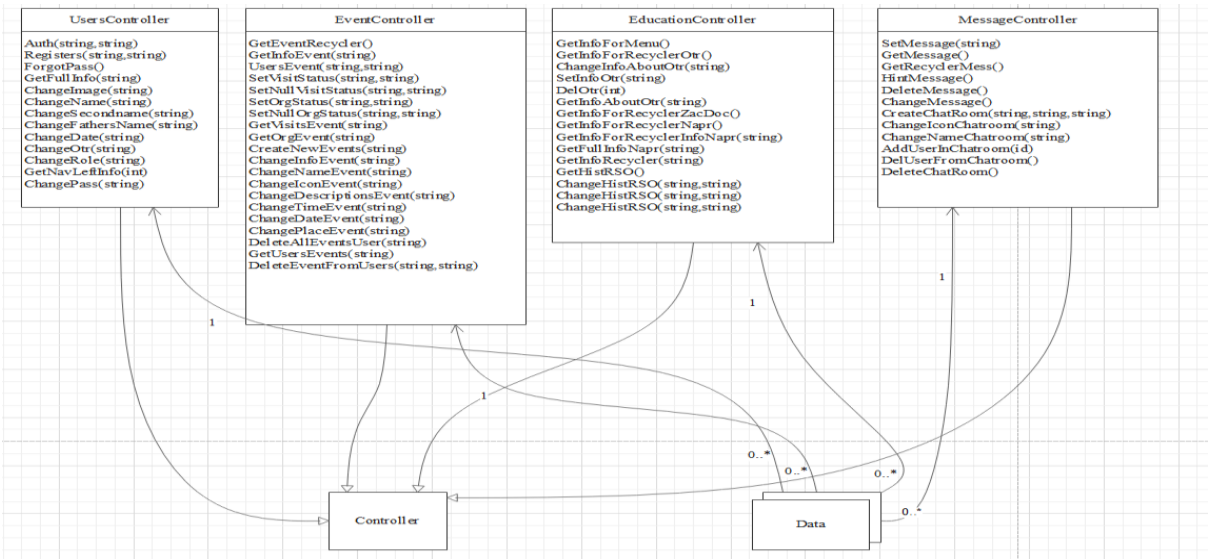

Рисунок 16 – Диаграмма классов контроллеров

Так как контроллеры хранятся в отдельном веб-приложении, то для работы с общими структурами данных требуется продублировать группу классов Data. На диаграмме представлены 4 класса-контроллера, рассмотрим каждый из них.

UsersController – данный контроллер обрабатывает все действия, которые могут быть совершены в отношении данных пользователя. Так, он проверяет введенные данные при регистрации или авторизации, меняет пароль. А также позволяет произвести выборку, изменение или удаление расширенных данных пользователя.

EventController – производит основные действия над «мероприятиями». Помимо работы с базой мероприятий, контроллер позволяет пользователю становиться участником или организатором, покидать определенное мероприятие, создавать новые, менять уже существующие, если имеется статус «организатор».

EducationController – благодаря этому контроллеру обрабатываются действия, совершенные в образовательном функциональном разделе. Позволяет добавлять новую, более актуальную информацию в базу знаний приложения, а также менять, удалять уже существующие записи.

MessageController – производит всю работу по отправке/получению сообщения, а также дает возможность пользователю изменить или удалить сообщение, скрыть диалог. Организует диалоговую систему, позволяя создавать беседы с несколькими участниками.

Как видно из Рисунка X все контроллеры наследуются от класса Controller. Именно он позволяет контроллерам принимать HTTP – запросы с последующей их обработкой и преобразованием в ответ.

2.1.3 Проектирование структур данных

Так как в проектируемом приложении роль модели играет база данных, то следующее рассмотрение структур данных будет являться проектированием модели [9].

В первую очередь, была разработана концептуальная модель базы данных (рисунок 17).

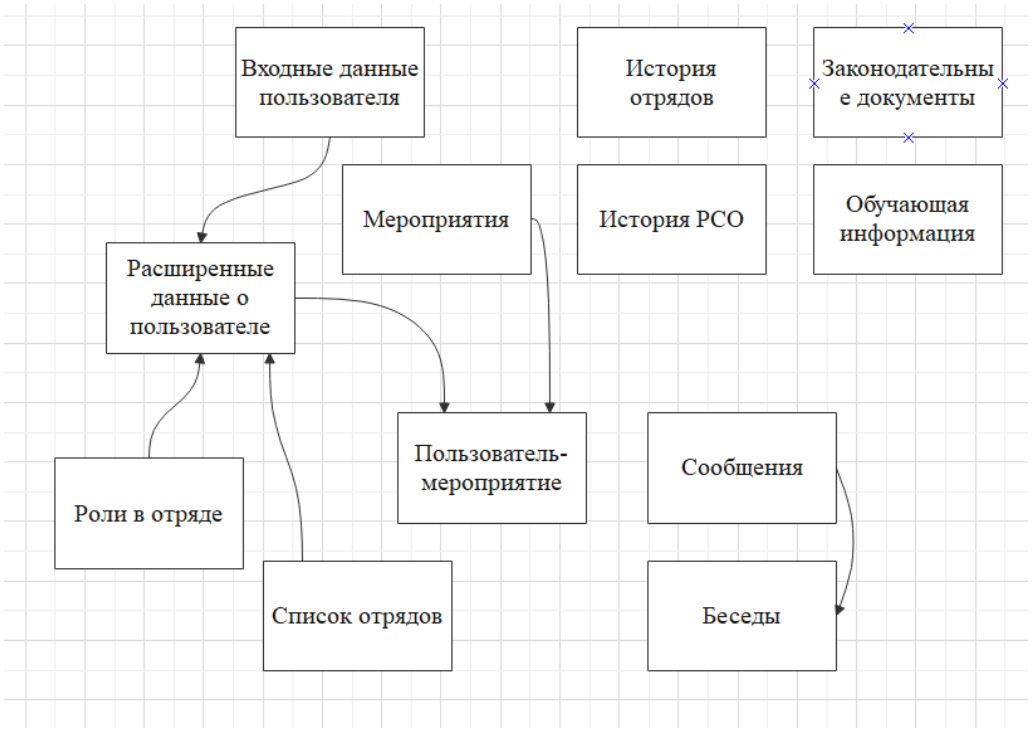

Рисунок 17 – Концептуальная модель базы данных

Для удобства база данных приложения будет состоять из нескольких групп таблиц, а именно:

– группа таблиц, содержащих информацию о пользователе;
– группа таблиц, формирующих базу знаний приложения;
– группа таблиц для работы с мероприятиями;
– группа таблиц для организации диалогов.

Описание сущностей, содержащих информацию о пользователе, представлено в таблице 13.

Таблица 13 – Описание сущностей

№	Наименование сущности	Описание
1	Расширенная информация о пользователе	Вся информация о пользователе
2	Входные данные пользователя	Электронная почта, пароль, номер членского билета пользователя

Продолжение таблицы 13

№	Наименование сущности	Описание
3	Список ролей в отряде	Список – какие роли есть в отряде
4	Список мероприятий	Список – какие мероприятия будут проходить или уже прошли

В таблице 14 приведенной далее описаны зависимости сущности.

Таблица 14 – Описание зависимостей сущностей

№	Тип сущности 1	Тип сущности 2	Описание	Мощность
1	Входные данные пользователя	Расширенная информация о пользователе	При входе в систему, будет получена расширенная информация пользователя	1 : 1
2	Расширенная информация о пользователе	Список ролей в отряде	Информация о пользователе содержит все роли пользователя в отряде	M : M
3	Расширенная информация о пользователе	Список мероприятий	Информация о пользователе содержит все роли данного пользователя в отряде	M : M

В таблице 15 приведено описание сущности «Входные данные».

Таблица 15 – Сущность «Входные данные»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Номер членского билета	Целый	7	0000000 – 9999999	Нет	FK	1231231
Логин	Символьный	30	‘A-Z’,’a-z’,’@’,’.’,’_’	Нет		Kolokolcev2010@gmail.com
Пароль	Символьный	15		Нет		123123r

Сущность «Расширенная информация» описана в таблице 16.

Таблица 16 – Сущность «Расширенная информация»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Фотография	Символьный	20	‘А-я’	Нет		Колокольцев
Фамилия	Символьный	20	‘А-я’	Нет		Колокольцев
Имя	Символьный	20	‘А-я’	Нет		Кирилл
Отчество	Символьный	20	‘А-я’	Да		Алексеевич
Дата рождения	Дата	10	01.01.1950 – 31.12.2100	Нет		01.01.2003
Принадлежность к отряду	Символьный	20	‘А-я’	Нет		Спектр
Дата вступления в отряд	Дата	10	01.01.1950 – 31.12.2100	Нет		01.01.2003
Роль в отряде	Целый	2	01-99	Нет	FK	1
Номер членского билета	Целый	7	0000000 – 9999999	Нет	FK	1231231

Таблица 17 отображает сущность «Список ролей в отряде».

Таблица 17 – Сущность «Список ролей в отряде»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_роли	Целый	2	01-99	Нет	PK	1231231
Наименование роли	Символьный	20	‘А-я’	Нет		Комиссар

Сущность «Список мероприятий» описывается ниже (таблица 18).

Таблица 18 – Сущность «Список мероприятий»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_мероприятия	Целый	2	01-99	Нет	PK	1231231
Наименование мероприятия	Символьный	20	‘А-я’,’ ’	Нет		Областная школа комиссаров

Далее будет приведена концептуальная модель группы таблиц, формирующих информационную базу приложения. Рассмотрим описание сущностей (таблица 19).

Таблица 19 – Описание сущностей

№	Наименование сущности	Описание
1	История РСО	История развития РСО
2	Законодательные документы, по которым существует система РСО;	Список и описание законодательных документов
3	История отдельных отрядов;	История определенного отряда
4	Обучающая информация по работе в различных направлениях.	Обучающая информация по работе каждого отдельного направления

Ниже приведено описание сущности «История РСО» (таблица 20).

Таблица 20 – Сущность «История РСО»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
История РСО	Поле MEMO	64000	256-64000 символов	Нет		История РСО начинается …

Таблица 21 описывает сущность «Законодательные документы».

Таблица 21 – Сущность «Законодательные документы»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_документа	Целый	2	01-99	Нет	PK	1231231
Наименование документа	Символьный	20	‘А-я’,’ ’	Нет		Мирный атом
Содержимое документа	MEMO	64000	256-64000 символов	Нет		История РСО начинается…

Сущность «История отрядов» описывается в таблице 22.

Таблица 22 – Сущность «История отрядов»

Наименование	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_Отряда	Целый	2	01-99	Нет	PK	1231231
Наименование отряда	Символьный	20	‘А-я’,’ ’	Нет		Мирный атом
История отряда	MEMO	64000	256-64000 символов	Нет		История РСО начинается…

Таблица 23 описывает сущность «Обучающая информация».

Таблица 23 – Сущность «Обучающая информация»

Наименование	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_направления	Целый	2	01-99	Нет	PK	1231231
Наименование направления	Символьный	20	‘А-я’,’ ’	Нет		Мирный атом
Информация по направлениям	MEMO	64000	256-64000	Нет		История РСО

Далее будет спроектирована группа таблиц «Мероприятия». В данной группе будет задействовано всего 2 таблицы: пользователи и мероприятия. Рассммотрим описание сущностей (таблица 24).

Таблица 24 – Описание сущностей

№	Наименование сущности	Описание
1	Мероприятия	Вся информация о мероприятии
2	Расширенная информация о пользователе	Информация о пользователе

Данные таблицы будут относиться друг к другу правилом M:M. Рассмотрим подробнее сущность «Мероприятия» (таблица 25).

Таблица 25 – Сущность «Мероприятия»

Наименование	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_Мероприятия	Целый	2	01-99	Нет	PK	12
Изображение мероприятия	Символьный	20	‘А-я’,’ ’	Нет		Oshk.jpg
Название мероприятия	Символьный	20	‘А-я’,’ ’	Нет		Областная
Краткое описание	Символьный	100	‘А-я’,’ ’	Нет		Мирный атом
Количество участников	Целый	3	001-999	Нет		123
Полное описание	MEMO	64000	256-64000 символов	Нет		История РСО начинается…
Дата проведения	Дата	10	01.01.1950 – 31.12.2100	Нет		01.01.2003
Время проведения	Время	8	00:00:00	Нет		23:59:59
Место проведения	Символьный	100	‘А-я’,’ ’	Нет		Мирный атом

Далее будет спроектирована группа таблиц для организации диалогов. Рассмотрим сущность «Сообщения» (таблица 26).

Таблица 26 – Сущность «Сообщение»

Наименование	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Время отправки	Время	8	00:00:00	Нет		23:59:59
ID_отправителя	Целый	7	0000001-00000099	Нет	PK	11
ID_получателя	Целый	7	0000001-00000099	Нет	PK	123123
ID_беседы	Целый	7	0000001-00000099	Нет	PK	123123
Сообщение	Полное описание	MEMO	64000	256-64000 символов	Нет	
Видимость	Целочисленное	1	0,1	Нет		1
Прочитано	Целочисленное	1	0,1	Нет		1

Сущность, позволяющая создавать диалоги с двумя или более пользователями, описана в таблице 27.

Таблица 27 – Сущность «Беседа»

Наименование	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_беседы	Целый	2	01-99	Нет	PK	12
Название беседы	Символьный	100	‘А-я’,’ ’	Нет		Мирный атом
Пользователи	Бинарный	100	‘A-z,’ ’ ,’=’ ,’{ ’ ,’ ’	Нет		{ID = 123,ID = 123}
Картинка беседы	Символьный	100	‘А-я’,’ ’	Нет		1.jpg
Создатель	Целый	7	0000001 - 00000099	Нет	PK	123123

Таким образом, была спроектирована логическая модель базы данных разрабатываемого приложения (рисунок 18).

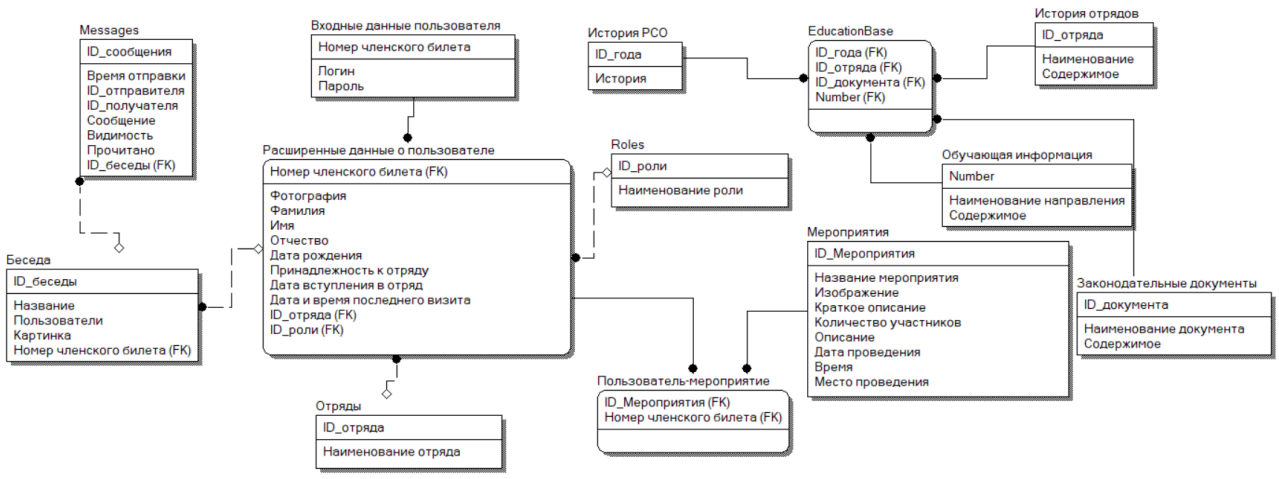

Рисунок 18 – Логическая модель разрабатываемой базы данных

Из приведенной выше диаграммы видно, как будет выглядеть разработанная структура данных. Видны связи, и как передаются вторичные ключи (FK).

2.2 Проектирование интерфейса пользователя

На основе выявленных функциональных требований был спроектирован пользовательский интерфейс мобильного приложения.

Рассмотрим схему сущностей представлений, а так же переходы по страницам (рисунок 19) [10].

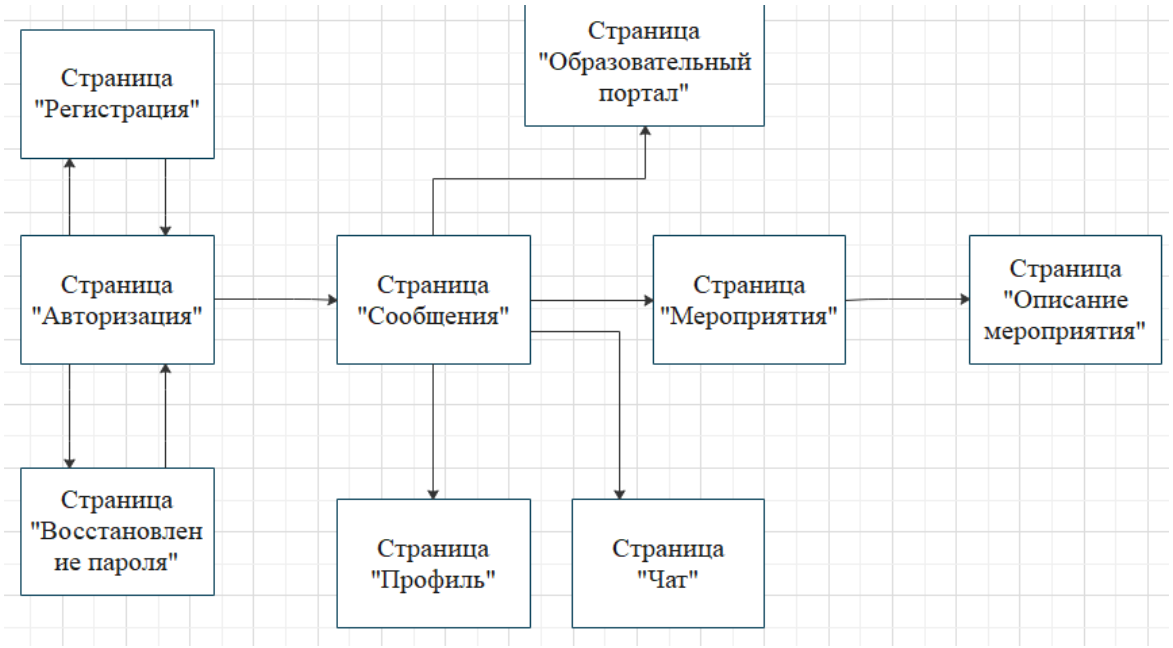

Рисунок 19 – Схема переходов по страницам приложения

К основным страницам приложения относятся следующие:

– страница «Авторизация»;
– страница «Регистрация»;
– страница «Восстановление пароля»;
– страница «Сообщения»;
– страница «Чат»;
– страница «Профиль»;
– страница «Образовательный портал»;
– страница «Мероприятия»;
– страница «Описание мероприятия».

Так как интерфейс некоторых страниц очень схож, то будут рассмотрены только 5 прототипов страниц, а именно:

– страница «Авторизация»;
– страница «Сообщения»;
– страница «Чат»;
– страница «Мероприятия»;
– страница «Профиль».

Рассмотрение вышеприведенных страниц начнется с прототипа «Авторизация» (рисунок 20).

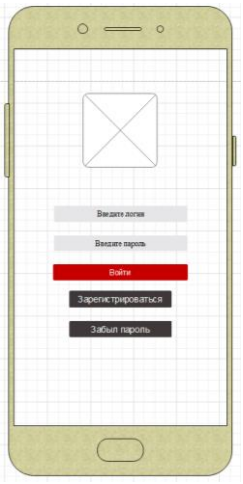

Рисунок 20 – Прототип страницы «Авторизация»

Как видно из прототипа, окно будет иметь несколько элементов:

– место под логотип приложения;

– две строки для ввода данных;

– три кнопки – «Войти», «Зарегистрироваться», «Забыл пароль»;

При вводе корректных пользователем данных будет осуществлен переход на страницу «Сообщения» (рисунок 21).

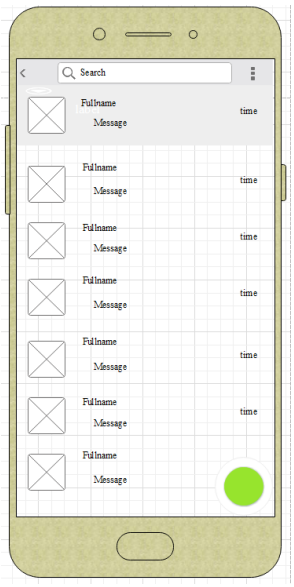

Рисунок 21 – Прототип страницы «Сообщения»

На данной странице будут отображаться все диалоги, в которых участвует пользователь. На данной странице, нажав на «плавающую кнопку» с иконкой сообщения (На прототипе отображена зеленым кругом), пользователь может написать сообщение другому пользователю, диалога с которым пока что нет. При нажатии на уже существующий диалог, будет осуществлен переход к чату с человеком либо к беседе с несколькими людьми. Данный чат будет приведен на следующем рисунке (рисунок 22).

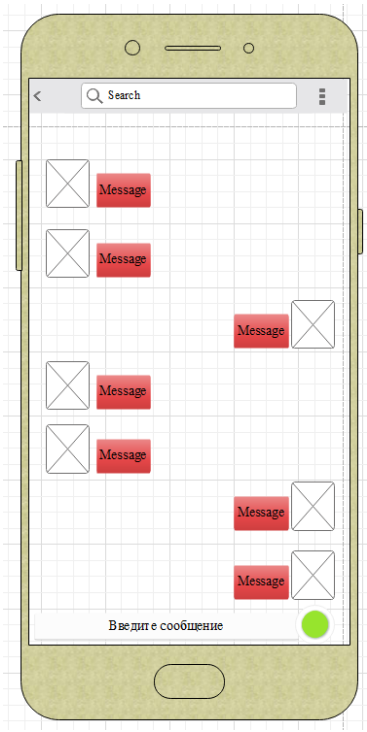

Рисунок 22 – Прототип страницы «Чат»

Страница чата предназначена для обмена сообщениями с другим пользователем.

Далее будет рассмотрена страница со списком мероприятий. Чтобы перейти к ней, нужно нажать в навигационном меню кнопку «Мероприятия». Данная страница будет содержать список предстоящих мероприятий с их кратким описанием, а также датой и временем проведения (рисунок 23).

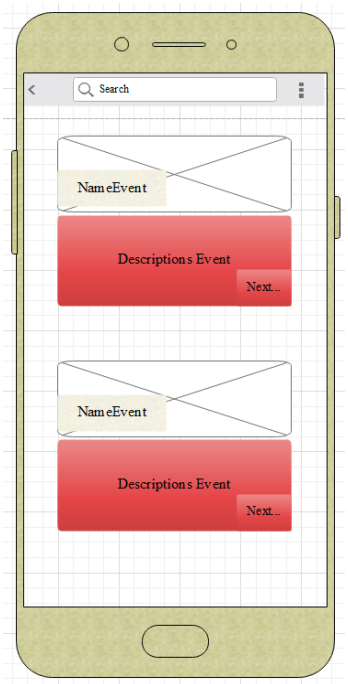

Рисунок 23 – Страница со списком мероприятий

Данная страница нацелена на ознакомление пользователя со списком предстоящих мероприятий.

Все страницы после прохождения авторизации будут иметь боковое навигационное меню, позволяющее перемещаться по приложению. Данное меню будет вызываться свайпом вправо. Рассмотрим его (рисунок 24).

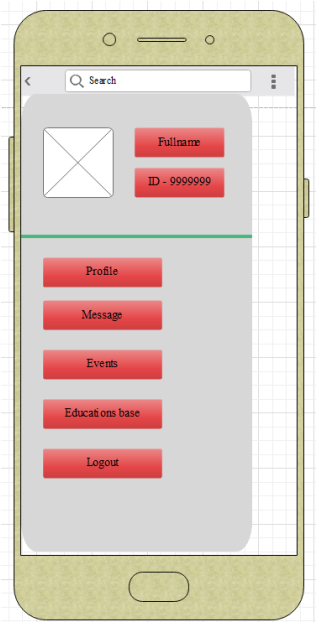

Рисунок 24 – Прототип «навигационного меню»

Данное меню имеет разделено на 2 области – головная область меню и область, содержащая ссылки для навигации по приложению.

Головная область содержит в себе информацию об авторизованном пользователе, а именно:

– фотография пользователя;

– его фамилия и имя;

– номер членского билета.

Навигационная область содержит в себе ссылки:

– «Профиль» – при нажатии осуществляется переход на страницу с информацией пользователя;

– «Сообщения» – осуществляет переход к списку входящих сообщений;

– «Мероприятия» – осуществляет переход к списку доступных мероприятий;

– «Информационная база» – переход к информационному порталу приложения;

– «Выйти» – выход из аккаунта.

Таким образом, были разработаны макеты основных страниц для приложения «RSOnline».

2.3 Календарное планирование и оценка бюджета

Для успешного управления любым проектом просто необходимо произвести качественное планирование. Под планированием понимается оптимальное распределение ресурсов для достижения поставленных целей. Для планирования был составлен перечень задач, отображающих процесс создания мобильного приложения для Пензенского регионального отделения молодежной общероссийской общественной организации «Российские студенческие отряды» (таблица 28). За начало проекта была принята дата – 23 марта [11].

Таблица 28 – Перечень задач проекта

№	Название задачи	Длительность	Предшественники
1	Начало проекта	-	-
2	Анализ требований проекта разработки мобильного приложения для Пензенского регионального отделения молодежной общероссийской общественной организации «Российские студенческие отряды»	-	-
3	Обзор приложений-аналогов	2 дня	1
4	Составление списка функциональных возможностей приложения	4 дня	1
5	Выбор программного средства для реализации приложения	3 дня	3;4
6	Анализ требований завершен	-	5
7	Проектирование для проекта разработки мобильного приложения для Пензенского регионального отделения молодежной общероссийской общественной организации «Российские студенческие отряды»	-	-
8	Проектирование интерфейса приложения	5 дней	6
9	Проектирование структур данных	7 дней	6
10	Выбор архитектуры программного обеспечения	3 дня	9
11	Проектирование завершено	-	8;9;10
12	Реализация	-	-
13	Программирование классов приложения	8 дней	11
14	Создание базы данных	3 дня	11
15	Интеграция клиентской и серверной частей приложения	5 дней	13;14
16	Реализация завершена	-	15
17	Отладка	-	-
18	Тестирование работы приложения	4 дня	16
19	Исправление ошибок	7 дней	18
20	Отладка завершена	-	19
21	Конец проекта	-	20

Из таблицы 28 можно увидеть, что необходимо выполнить некое количество задач, а именно 21. В таблице 29 отображен список ресурсов, используемых во время реализации проекта.

Таблица 29 – Список ресурсов проекта

Название ресурса	Тип	Стандартная ставка	Сверхурочная ставка
Программист 1	Трудовой	90 Р/ч	150 Р/ч
Программист 2	Трудовой	90 Р/ч	150 Р/ч
Программист 3	Трудовой	90 Р/ч	150 Р/ч
Интернет	Затраты	90 Р/ч	150 Р/ч
Компьютер	Материальный	1500 Р	
Смартфон на платформе Android	Материальный	800 Р	

После определения списка ресурсов, требуемых для реализации проекта, необходимо отобразить список этих ресурсов в задачах (таблица 30).

Таблица 30 – Назначения ресурсов

№	Название задачи	Ресурс	Единицы
1	Начало проекта	-	-
2	Анализ требований проекта разработки мобильного приложения для Пензенского регионального отделения молодежной общероссийской общественной организации «Российские студенческие отряды»	-	-
3	Обзор приложений-аналогов	Программист1 Интернет Компьютер	100% 1 шт.
4	Составление списка функциональных возможностей приложения	Программист1 Интернет Компьютер	100% 1 шт.
5	Выбор программного средства для реализации приложения	Программист1 Интернет Компьютер	100% 1 шт.
6	Анализ требований завершен		
7	Проектирование для проекта разработки мобильного приложения для Пензенского регионального отделения молодежной общероссийской общественной организации «Российские студенческие отряды»		

Продолжение таблицы 30

8	Проектирование интерфейса приложения	Программист1 Программист2 Интернет Компьютер	100% 50% 2 шт.
9	Проектирование структур данных	Программист2 Программист3 Интернет Компьютер	50% 100% 2 шт.
10	Выбор архитектуры программного обеспечения	Программист1 Программист2 Программист3 Интернет Компьютер	100% 50% 50% 3 шт.
11	Проектирование завершено		
12	Реализация		
13	Программирование классов приложения	Программист1 Программист2 Интернет Компьютер	100% 50% 2 шт.
14	Создание базы данных	Программист2 Программист3 Интернет Компьютер	50% 100% 2 шт.
15	Интеграция клиентской и серверной частей приложения	Программист1 Программист2 Программист3 Интернет Компьютер	100% 50% 50% 3 шт.
16	Реализация завершена		
17	Отладка		
18	Тестирование работы приложения	Программист2 Программист3 Интернет Компьютер Телефон на платформе Android	100% 100% 2 шт. 1 шт.
19	Исправление ошибок	Программист1 Программист2 Программист3 Интернет Компьютер Телефон на платформе Android	100% 100% 100% 3 шт. 1 шт.
20	Отладка завершена		
21	Конец проекта		

Из представленных таблиц видно, что решением задач будут заниматься 3 программиста, которым для этого необходимы компьютеры, доступ к Интернету, а также смартфон на платформе Android для тестирования приложения.

При помощи системы управления проектами Microsoft Project выполним планирование задач проекта (рисунок 25).

Название задачи	Длительность	Начало	Окончание	Предшественники	Названия ресурсов
Начало проекта	0 дней	Пн 23.03.20	Пн 23.03.20		
Анализ требований проекта	**7 дней**	**Пн 23.03.20**	**Вт 31.03.20**		
Обзор приложений-аналогов	2 дней	Пн 23.03.20	Вт 24.03.20	1	Программист1;Интернет;Компьютер[1]
Составление списка функциональных возможностей приложения	4 дней	Пн 23.03.20	Чт 26.03.20	1	Программист1;Интернет;Компьютер[1]
Выбор программного средства для реализации приложения	3 дней	Пт 27.03.20	Вт 31.03.20	3;4	Программист1;Интернет;Компьютер[1]
Анализ требований завершен	0 дней	Вт 31.03.20	Вт 31.03.20	5	
Проектирование для проекта	**10 дней**	**Ср 01.04.20**	**Вт 14.04.20**		
Проектирование интерфейса приложения	5 дней	Ср 01.04.20	Вт 07.04.20	6	Программист1;Программист2[50%];Интернет;Компьютер[2]
Проектирование структур данных	7 дней	Ср 01.04.20	Чт 09.04.20	6	Программист2[50%];Программист3;Интернет;Компьютер[2]
Выбор архитектуры программного обеспечения	3 дней	Пт 10.04.20	Вт 14.04.20	9	Программист1;Программист2[50%];Программист3[50%];Интернет;Компьютер[3]
Проектирование завершено	0 дней	Вт 14.04.20	Вт 14.04.20	8;9;10	
Реализация	**13 дней**	**Ср 15.04.20**	**Пт 01.05.20**		
Программирование классов приложения	8 дней	Ср 15.04.20	Пт 24.04.20	11	Программист1;Программист2[50%];Интернет;Компьютер[2]
Создание базы данных	3 дней	Ср 15.04.20	Пт 17.04.20	11	Программист2[50%];Программист3;Интернет;Компьютер[2]
Интеграция клиентской и серверной частей приложения	5 дней	Пн 27.04.20	Пт 01.05.20	13;14	Программист1;Программист2[50%];Программист3[50%];Интернет;Компьютер[3]
Реализация завершена	0 дней	Пт 01.05.20	Пт 01.05.20	15	
Отладка	**11 дней**	**Пн 04.05.20**	**Пн 18.05.20**		
Тестирование работы приложения	4 дней	Пн 04.05.20	Чт 07.05.20	16	Программист2;Программист3;Интернет;Телефон на платформе Android[1];Компьютер[3]
Исправление ошибок	7 дней	Пт 08.05.20	Пн 18.05.20	18	Программист1;Программист2;Программист3;Интернет;Компьютер[3];Телефон на платформе Android[1]
Отладка завершена	0 дней	Пн 18.05.20	Пн 18.05.20	19	
Конец проекта	0 дней	Пн 18.05.20	Пн 18.05.20	20	

Рисунок 25 – Планирование задач в Microsoft Project

Microsoft Project по задачам, введенным в системе, автоматически выстроил диаграмму Ганта (рисунок 26).

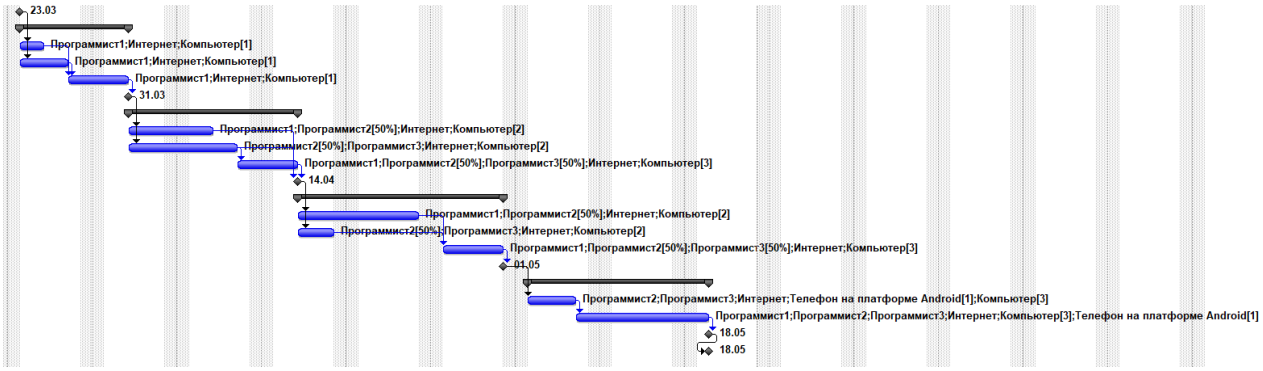

Рисунок 26 – Диаграмма Ганта

Сетевой график представлен на рисунках 27 - 29.

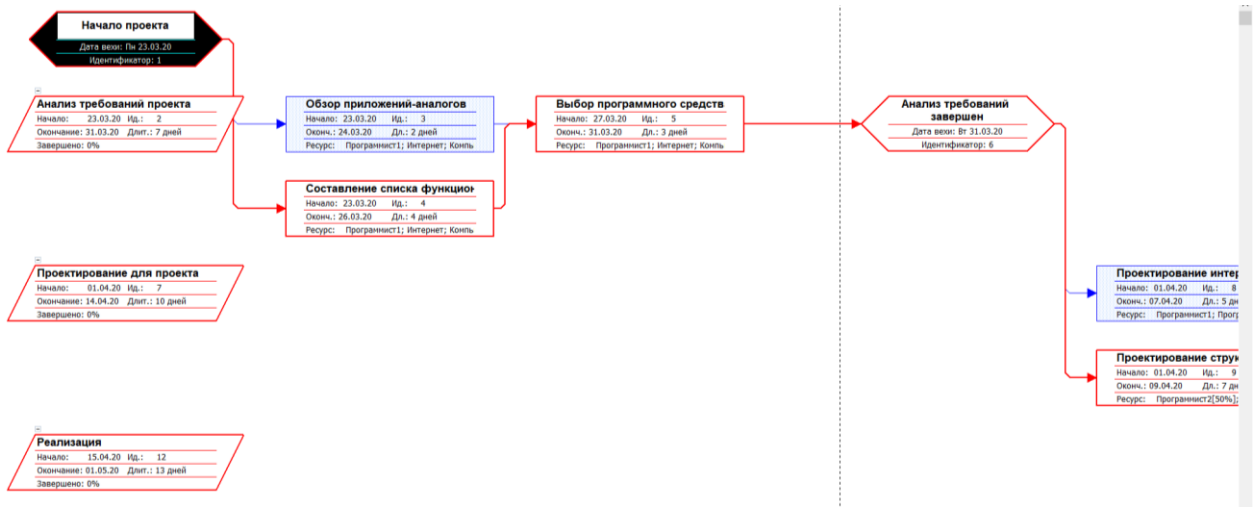

Рисунок 27 – Начало сетевого графика

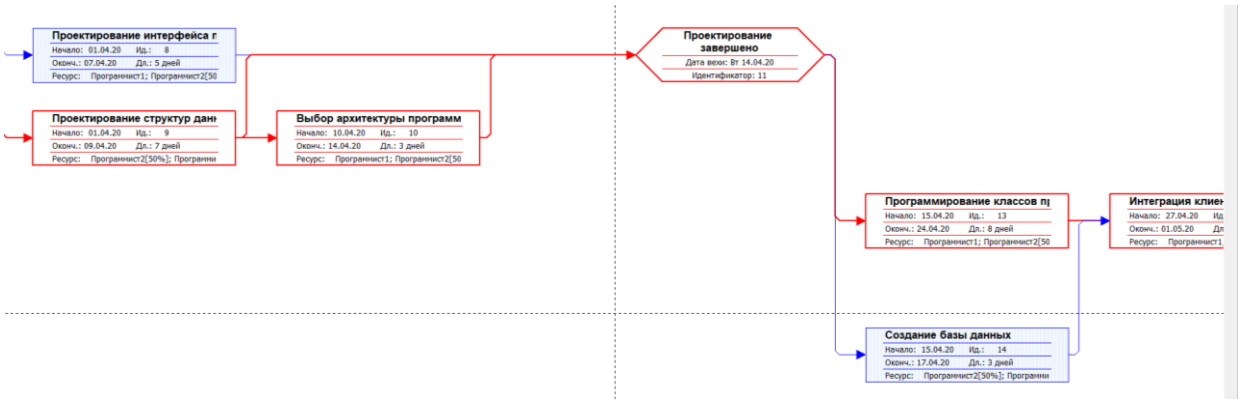

Рисунок 28 – Продолжение сетевого графика

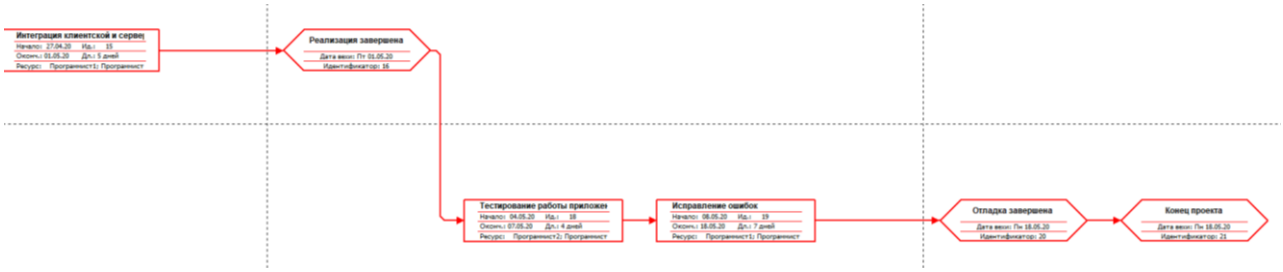

Рисунок 29 – Окончание сетевого графика

Сетевой график показывает, что критический путь проходит через задачи с номерами 1, 4, 5, 6, 9, 10, 11, 13, 15, 16, 18, 19, 20, 21.

На рисунке 30 изображен обзор проекта, на котором можно увидеть подсчитанную Microsoft Project длительность реализации проекта – 41 день и предполагаемые затраты – 89740 руб.

Даты			
Начало:	Пн 23.03.20	Окончание:	Пн 18.05.20
Баз. начало:	НД	Баз. окончание:	НД
Факт. начало:	НД	Факт. окончание:	НД
Отклон. начала:	0 дней	Отклон. окончания:	0 дней

Длительность			
Текущий план:	41 дней	Оставшаяся:	41 дней
Базовый план:	0 дней?	Фактическая:	0 дней
Отклонение:	41 дней	Процент завершени	0%

Трудозатраты			
Текущий план:	708 ч	Оставшиеся:	708 ч
Базовый план:	0 ч	Фактические:	0 ч
Отклонение:	708 ч	Процент завершени	0%

Затраты			
Текущий план:	р.89 740,00	Оставшиеся:	р.89 740,00
Базовый план:	р.0,00	Фактические:	р.0,00
Отклонение:	р.89 740,00		

Состояние задачи	
Еще не начатые задачи:	21
Выполняющиеся задачи	0
Завершенные задачи:	0
Итого задач:	21

Состояние ресурса	
Трудовые ресурсы:	3
Трудовые ресурсы с прев. дост	1
Материальные ресурсы:	2
Итого ресурсов:	6

Рисунок 30 – Обзор проекта

На рисунках 31 - 35 представлены графики загруженности ресурсов.

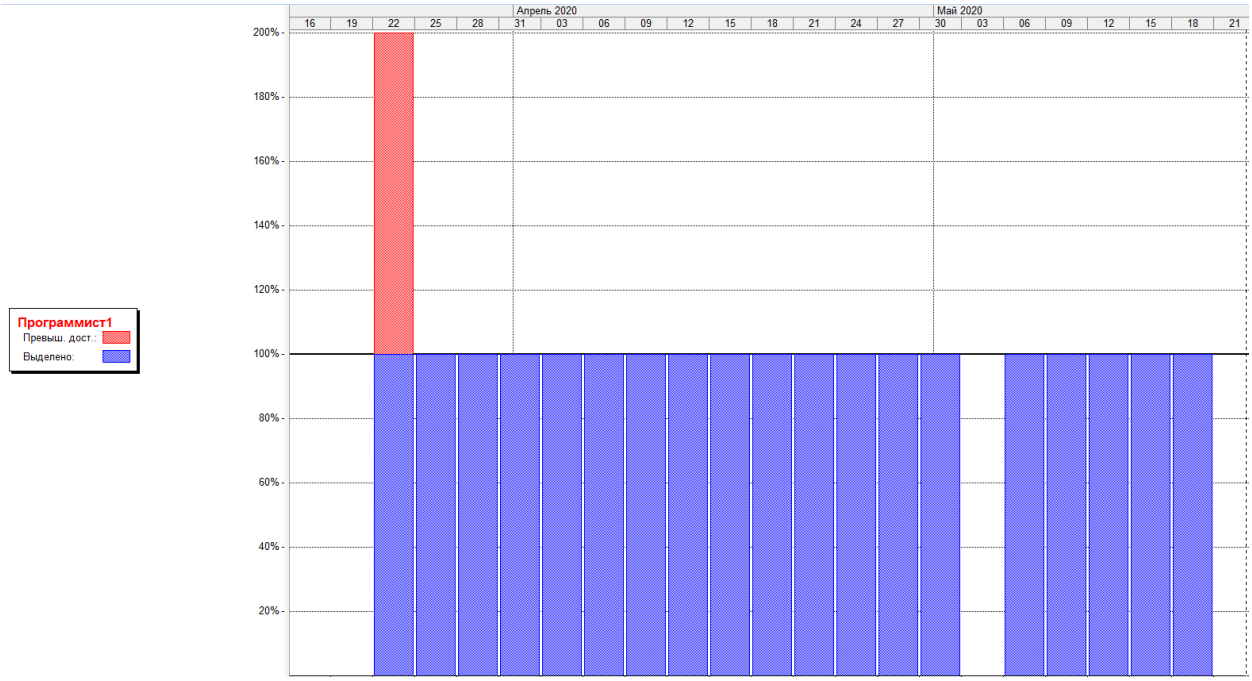

Рисунок 31 – График загруженности ресурса «Программист1»

Из приведенного выше графика видно, что «Программист1» перегружен только 22 апреля 2020 года.

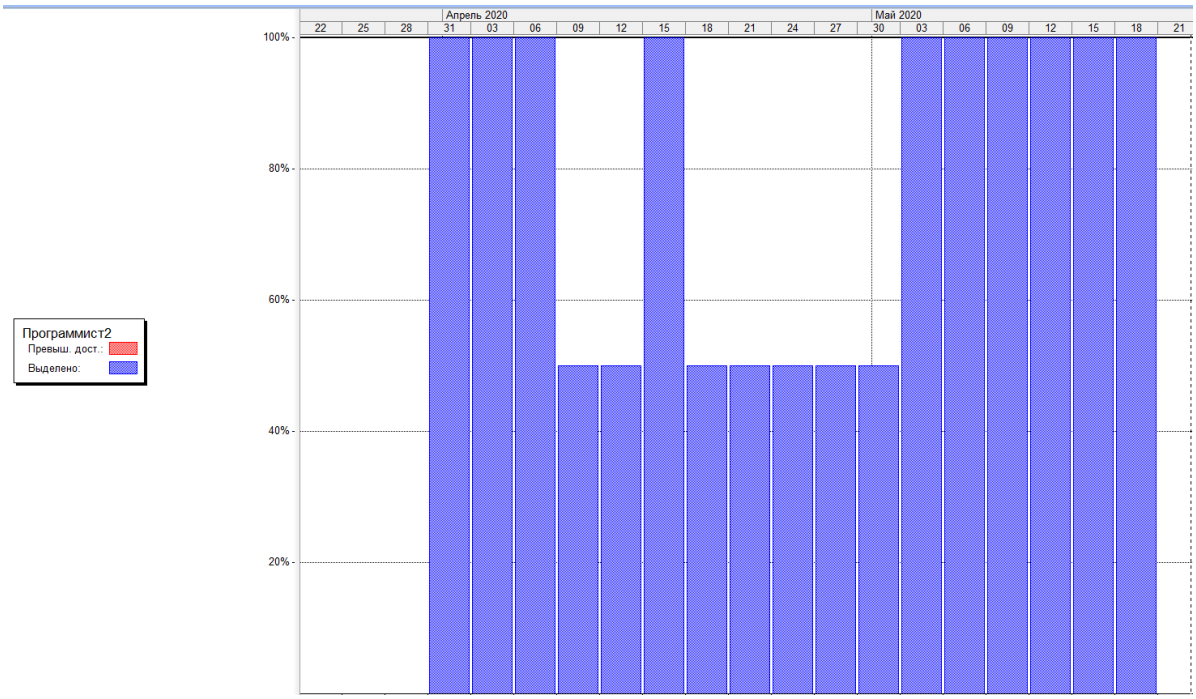

Рисунок 32 – График загруженности ресурса «Программист2»

График показывает, что «Программист2» не перегружается на протяжении всего проекта, а с 9 по 12 и с 18 по 30 апреля будет загружен только на 50%.

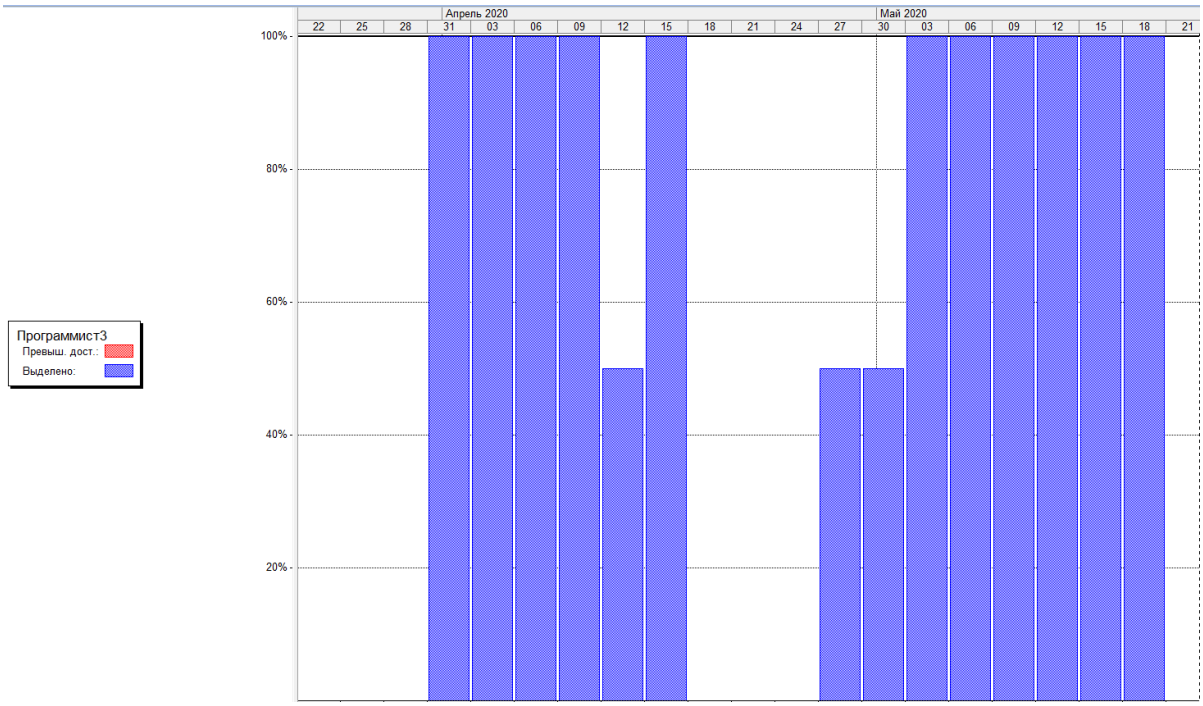

Рисунок 33 – График загруженности ресурса «Программист3»

Из графика видно, что «Программист3» не будет работать с 18 по 30 апреля. В рабочие дни он не перегружается.

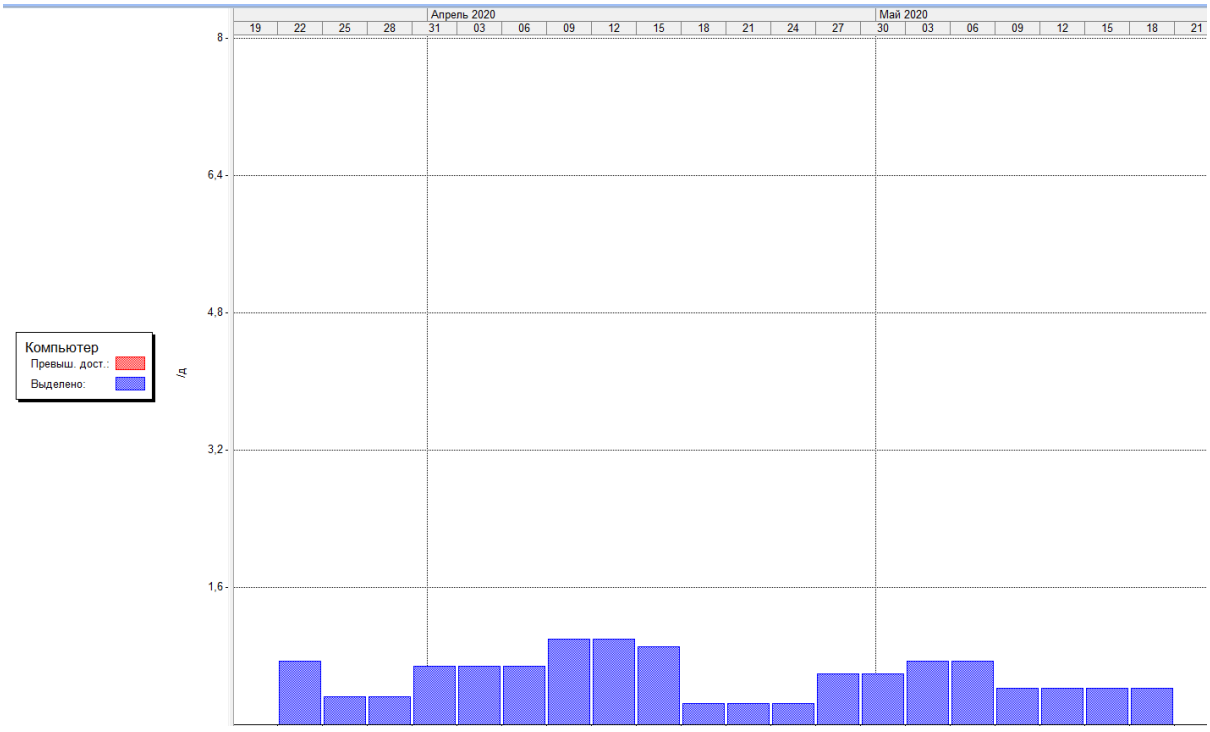

Рисунок 34 – График загруженности ресурса «Компьютер»

Как видно из приведенной выше диаграммы, ресурс «Компьютер» будет использоваться каждый день на протяжении всего проекта, но иметь разную загруженность.

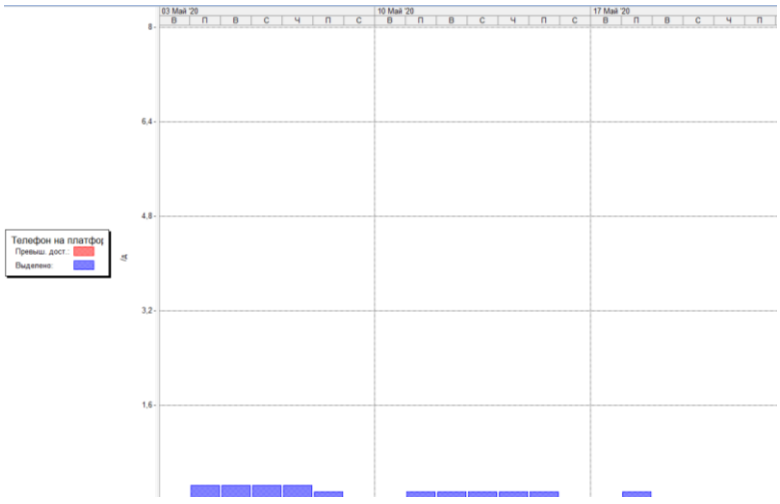

Рисунок 35 – График загруженности ресурса «Телефон на платформе Android»

Так как ресурс «Интернет» имеет тип – затраты, программа MS Project не может предоставить график использования данного ресурса.

В данной главе были спроектированы классы для всех 3х элементов архитектуры MVC. Были построены и рассмотрены диаграммы классов для представлений и контроллеров. Была выбрана и спроектирована структура данных в виде логической модели базы данных. Также был подобран подходящий интерфейс, отвечающий требованиям.

3 Реализация мобильного приложения

3.1 Реализация интерфейса пользователя

Дополнение Xamarin в среде разработке Visual Studio имеет несколько вариаций реализации пользовательского интерфейса. Среди них есть инструмент, который позволяет переносить объекты на форму – экран телефона. Помимо этого способа имеется второй вариант – программирование XML-файлов, добавляя объекты исключительно программно.

Пользовательский интерфейс разрабатываемого приложения был реализован преимущественно программно. В проектировании были выделены несколько основных страниц, которые имеют схожий интерфейс с другими:

– страница «Авторизация»;
– страница «Сообщения»;
– страница «Чат»;
– страница «Мероприятия»;
– страница «Профиль».

Страница «Авторизация» представлена на рисунке 36.

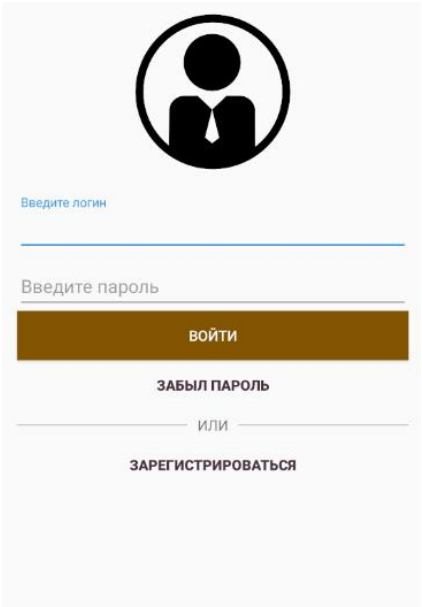

Рисунок 36 – Страница авторизации

Страница авторизация содержит 2 текстовых поля:

– первое поле для ввода логина;

– второе поле для ввода пароля.

Так же данная страница содержит 3 кнопки:

– «Войти» – при нажатии происходит вход в систему под введенными данными;

– «Забыл пароль» – после нажатия переходит к странице восстановления пароля;

– «Зарегистрироваться» – после нажатия переходит на страницу регистрации.

Все последующие страницы будут иметь боковое навигационное меню, позволяющее перемещаться по приложению. Данное меню будет вызываться свайпом вправо. Рассмотрим его (рисунок 37).

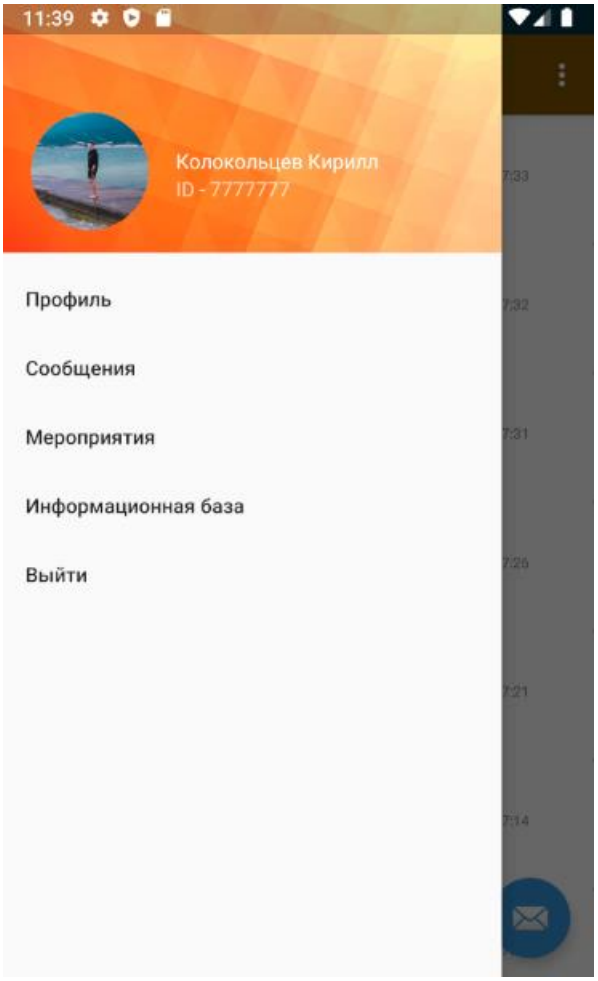

Рисунок 37 – Навигационное меню

Данное меню имеет разделено на 2 области - головная область меню и область, содержащая ссылки для навигации по приложению.

Головная область содержит в себе информацию об авторизовавшемся пользователе, а именно:

– фотография пользователя;

– его фамилия и имя;

– номер членского билета.

Навигационная область содержит в себе ссылки:

– «Профиль» – переход на страницу с информацией пользователя;

– «Сообщения» – осуществляет переход к списку входящих сообщений;

– «Мероприятия» – переход к списку доступных мероприятий;

– «Информационная база» – переход к информационному порталу;

– «Выйти» – выход из аккаунта.

После авторизации пользователь попадает на страницу диалогов. Рассмотрим рисунок 38.

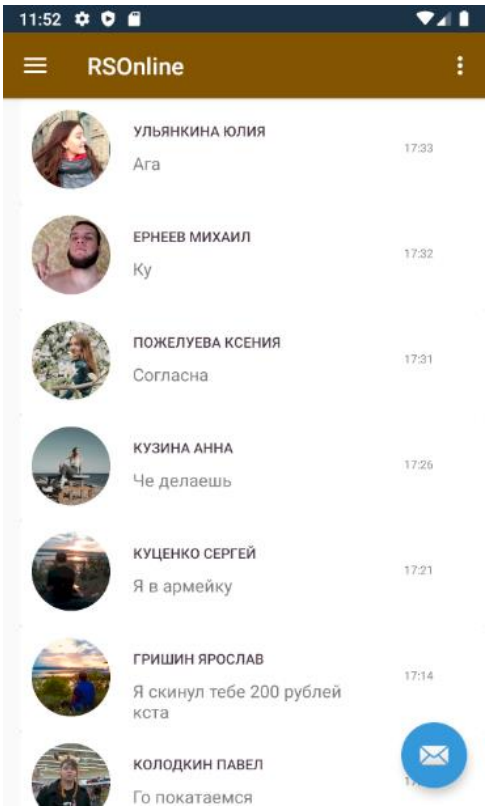

Рисунок 38 – Страница диалогов

На данной странице будут отображаться все диалоги, в которых участвует пользователь. На данной странице нажав на «плавающую кнопку» с иконкой сообщения, пользователь может написать сообщение другому пользователю, диалога с которым пока что нет. Так же, нажав на уже существующий диалог, будет осуществлен переход к чату с человеком, на чьё сообщение было совершено нажатие, либо к «беседе» с несколькими людьми. Данный чат будет приведен на следующем рисунке (рисунок 39).

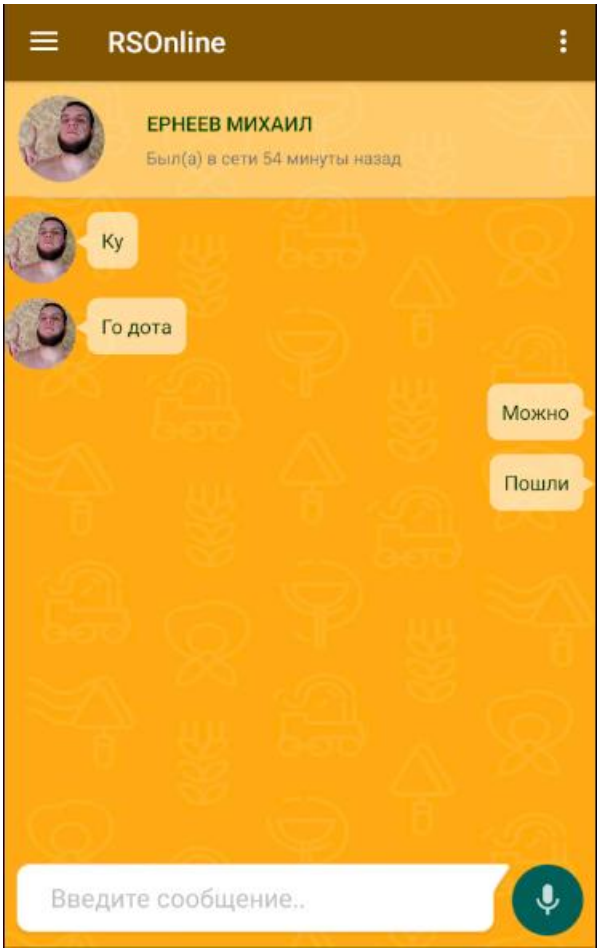

Рисунок 39 – Страница чата

Страница чата предназначена для обмена сообщениями с другим пользователем. При вводе текста значок микрофона заменяется значком треугольника – «Отправить сообщение».

Далее будет рассмотрена страница со списком мероприятий (рисунок 40). Чтобы перейти к ней, нужно нажать в навигационном меню кнопку «Мероприятия».

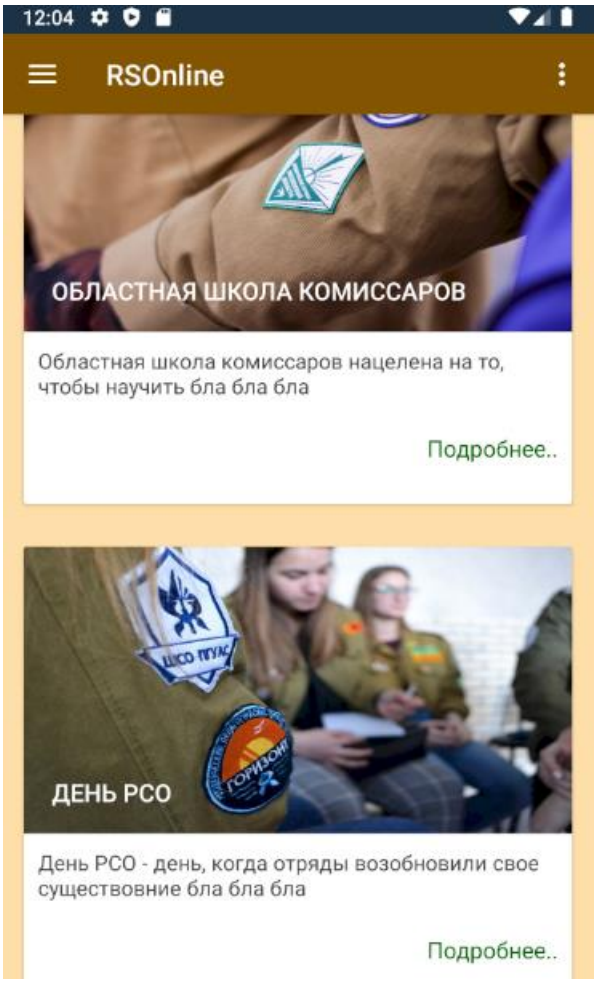

Рисунок 40 – Страница со списком мероприятий.

Данная страница нацелена на ознакомление пользователя со списком предстоящих мероприятий. Чтобы узнать более подробную информацию или записаться в участники или организаторы, нужно нажать кнопку «Подробнее…».

Таким образом, был реализован пользовательский интерфейс согласно поставленной задаче, а также требованиям к интерфейсу пользователя.

3.2 Реализация классов представлений

Рассмотрим реализацию классов представлений на примере левого навигационного меню.

Данное меню будет состоять из нескольких отдельных фрагментов. Подключенные xml-файлы, определяющие внешний вид данного класса, будут видны на рисунке 41.

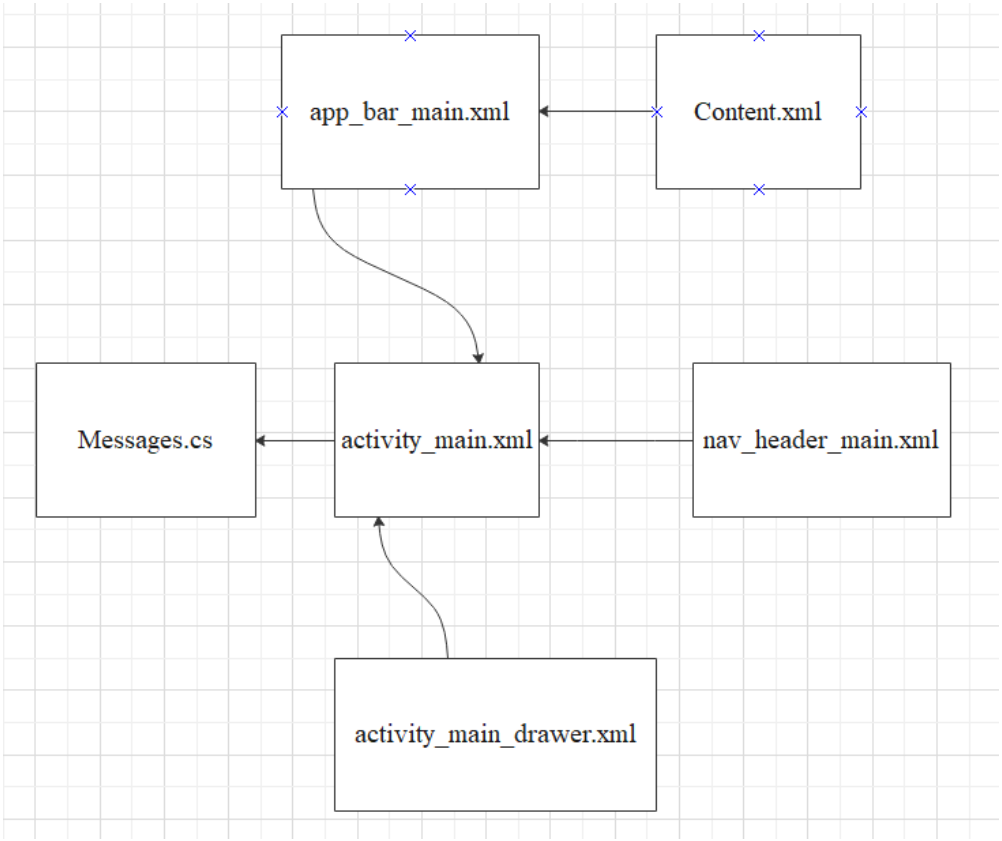

Рисунок 41 – Структура навигационного меню.

Рассмотрим подробнее подключенные к классу файлы.

Content.xml – отвечает за расположение элементов на основной контент-странице, в данном контексте рассмотрения ценности не несет.

Nav_header_main.xml – отвечает за расположение элементов в «Шапке» (верхняя часть) навигационного меню.

App_bar_name.xml разделяет основное контентное окно и верхнее вспомогательное меню.

Activity_main_drawer.xml содержит в себе список элементов меню. Именно он отвечает за видимость ссылок на другие страницы приложения.

Activity_main.xml – собирает воедино все элементы получившейся страницы, содержащей в себе все выше оговоренные элементы.

Класс Message.cs работает напрямую с activity_main.xml, имея в тоже время доступ ко всем элементам других подключенных ранее xml-файлов.

Рассмотрим заполнение «Шапки» навигационного меню. Класс обращается к статическому классу Sessions, получая ID авторизованного пользователя (листинг 1).

Листинг 1 – Передача запроса контроллеру

```
public DataForLeftMenu request(DataForLeftMenu Dsql1)
        {
            WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/UsersCont
roller/GetInfoUser");
            request1.Method = "POST";
            string idS = Sessions.GetSession();
            string data1 = "id=" + idS;
            byte[] byteArray1 =
System.Text.Encoding.UTF8.GetBytes(data1);
            request1.ContentType = "application/x-www-form-
urlencoded";
            request1.ContentLength = byteArray1.Length;
            using (System.IO.Stream dataStream =
request1.GetRequestStream())
            {
                dataStream.Write(byteArray1, 0,
byteArray1.Length);
            }
            WebResponse response1 = request1.GetResponse();
            string json_string1 = "";
            using (System.IO.Stream stream =
response1.GetResponseStream())
            {
                using (StreamReader reader = new
StreamReader(stream))
                {
                    json_string1 = reader.ReadToEnd();
                }
            }
            Dsql1 =
JsonConvert.DeserializeObject<DataForLeftMenu>(json_string1);
            response1.Close();
            return Dsql1;
        }
```

В приведенном выше листинге видно, что класс содержит метод request. Данный метод передает запрос, содержащий ID пользователя, контроллеру UsersController. Далее он считывает ответ от сервера и возвращает заполненную структуру данных. После вызова метода, уже имею

заполненную структуру данных, класс соединяется с элементами представления и меняет их значения.

Нижняя, основная часть навигационного меню содержит ссылки на другие страницы, структуру переходов между страницами можно увидеть в проектировании интерфейса пользователя (раздел 2.2).

Далее будет рассмотрена реализация функционала RecyclerView на примере страницы с входящими сообщениями. RecyclerView – инструмент для вывода однотипных элементов с возможностью прокрутки основной страницы до конца входных данных.

На диаграмме классов представлений в проектировании классов представлений (раздел 2.1.1) видно, что в этом процессе задействованы классы RecyclerViewAdapter,RecyclerViewHolder и интерфейс ItemClickListener. Данные классы представлены на рисунке 42.

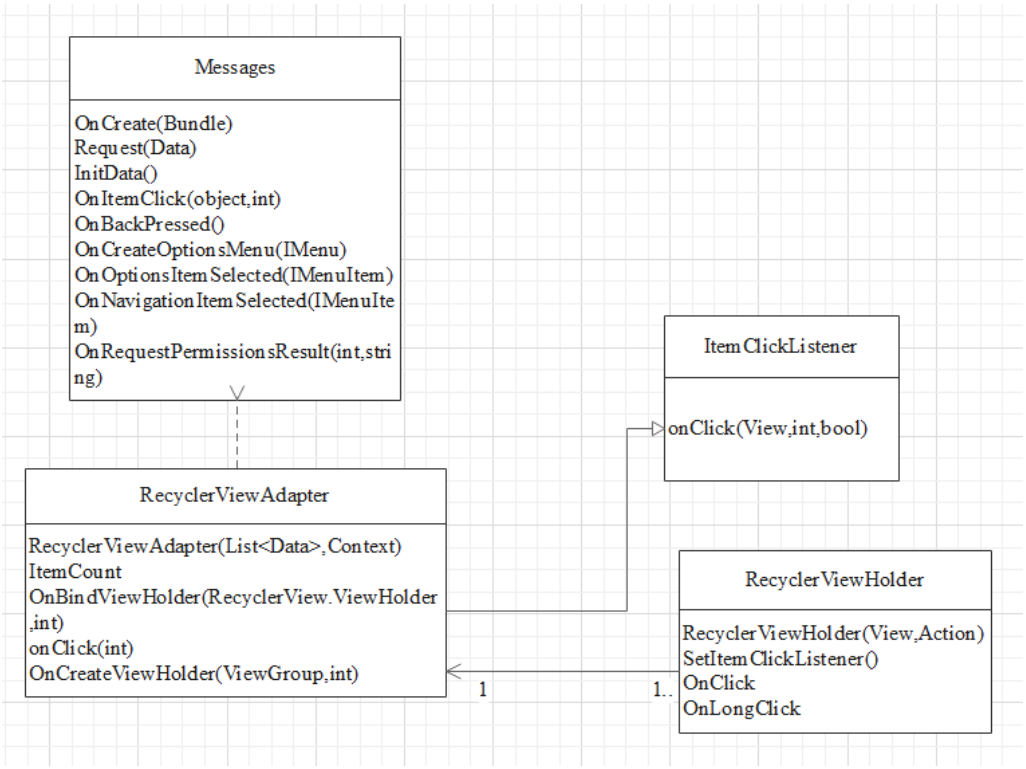

Рисунок 42 – Классы реализующие работу RecyclerView

Класс Message создает объект RecyclerView и передает ему данные, полученные путем отправки запроса соответствующему контроллеру. Аналогичный такому запросу листинг можно увидеть выше (листинг 1).

RecyclerViewAdapter создает новое представление используя xml-файл ItemMessage и заполняет его данными, полученными от класса Message. Данное действие будет производиться, пока входная структура не окажется пустой.

Далее новосозданное представление отправляется в RecyclerViewHolder, где оно будет добавлено в уже существующий контекст.

3.3 Реализация классов контроллеров

При проектировании классов контроллеров было принято решение разделить на 4 функциональных раздела:

– usersController;
– eventController;
– educationController;
– messageController.

Контроллеры получают HTTP- запросы в зависимости от действий пользователя в представлении. Возвращаемые структуры данных преимущественно сериализуются, потому что сервер передает различные структуры данных. Сериализация нужна, чтобы представление смогло получить не просто одно значение, а целый объект, сразу при получении преобразовав его в структуру.

Для примера реализации будет рассмотрен контроллер UsersController. Данный контроллер обрабатывает запросы, непосредственно связанные с пользователем. Например, заполнение личной информации, регистрация, изменение данных, добавление фотографии и тд.

Метод GetFullInfo класса UsersController представлен на листинге 2.

Листинг 2 – Метод выборки данных о пользователе

```
public string GetFullInfo(int id)
        {
            string json = "0";
            DataForFullInfo data = new DataForFullInfo();
            var conString =
ConfigurationManager.ConnectionStrings["LocalSqlServer"];
            string strConnString = conString.ConnectionString;
```

```
            using (SqlConnection con = new SqlConnection(strConnString))
            {
                string sql = "SELECT * FROM [UsersInfo] WHERE id = '" + id +
"'";
                string otr="0", role ="0";
                con.Open();
                SqlCommand command = new SqlCommand(sql, con);
                using (SqlDataReader oReader = command.ExecuteReader())
                {
                    if (oReader.Read())
                    {
                        data.image = oReader["image"].ToString();
                        data.Login = oReader["Login"].ToString();
                        data.Secondname = oReader["Secondname"].ToString();
                        data.Name = oReader["Name"].ToString();
                        data.Fathersname = oReader["Fathersname"].ToString();
                        data.date = oReader["date"].ToString();
                      data.dateOtr = oReader["dateOtr"].ToString();
                        otr = oReader["otr"].ToString();
                        role = oReader["role"].ToString();
                    }
                }
                string sql1 = "SELECT * FROM [Otr] WHERE id = '" + otr + "'";
                string sql2 = "SELECT * FROM [Roles] WHERE id = '" + role +
"'";
                SqlCommand command1 = new SqlCommand(sql1, con);
                using (SqlDataReader oReader1 = command1.ExecuteReader())
                {
                    if (oReader1.Read())
                    {
                        data.otr = oReader1["otr"].ToString();
                    }
                }
                SqlCommand command2 = new SqlCommand(sql2, con);
                using (SqlDataReader oReader2 = command2.ExecuteReader())
                {
                    if (oReader2.Read()) { data.role =
oReader2["role"].ToString(); }
                }
                con.Close();
            }
            json = JsonConvert.SerializeObject(data);
            return json;
        }
```

Данный метод получает на вход ID авторизованного пользователя и проводит выборку в базе данных всей информации, где ID равен полученному от представления. Затем он ищет отряд, в котором состоит пользователь, и его роль. Всю выбранную информацию метод помещает в заранее созданную структуру данных и, с помощью функции JsonConvert.SerializeObject() сериализует их.

3.4 Реализация структур данных

Для реализации модели разрабатываемого приложения «RSOnline» в качестве инструмента был выбран MS SQL server. MS SQL Server – программный продукт, нацеленный на хранение базы данных, позволяющий обращаться к данным из других программ. SQL server поддерживает язык XML и поддерживает протокол HTTP, что позволяет работать бесперебойно, а так же повысить скорость и доступность базы данных. Так же SQL сервер поддерживает возможность резервного копирования в любой момент, независимо от подключенных пользователей. Проанализировав вышеупомянутую информацию, был сделан вывод – данная СУБД оптимально подходит для реализации выпускного проекта.

Опираясь на проектирование структур данных, была разработана физическая модель данных. В таблице 31 приведена сущность «UsersAcc».

Таблица 31 – Сущность «UsersAcc»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID	integer	7	0000000 – 9999999	Нет	FK	1231231
Login	varchar	20	‘A-Z’,’a-z’,’@’,’.’,’ ’,’_’	Нет		Kolokolcev2010 @gmail.com
Password	varchar	20		Нет		123123r

Сущность «UsersInf» описана в таблице 32.

Таблица 32 – Сущность «UsersInf»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
image	image	20	‘А-я’	Нет		Колокольцев
Secondname	Nvarchar	50	‘А-я’	Нет		Колокольцев
Name	Nvarchar	50	‘А-я’	Нет		Кирилл
Fathersname	Nvarchar	50	‘А-я’	Да		Алексеевич

Продолжение таблицы 32

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Date	Datetime	10	01.01.1950 – 31.12.2100	Нет		01.01.2003
Otr	Nvarchar	50	‘А-я’	Нет		Спектр
Date_otr	Datetime	10	01.01.1950 – 31.12.2100	Нет		01.01.2003
Role	integer	2	01-99	Нет	FK	1
ID	Integer	7	0000000 – 9999999	Нет	FK	1231231
Last_Online	Datetime	10	01.01.1950 – 31.12.2100	Нет		01.01.2003 00:00:00

Таблица 33 содержит информацию о сущности «Roles».

Таблица 33 – Сущность «Roles»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_role	integer	2	01-99	Нет	PK	1231231
Name	Nvarchar	20	‘А-я’	Нет		Комиссар

Информацию о сущности «Hist_RSO» можно увидеть на таблице 34.

Таблица 34 – Сущность «Hist_RSO»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_years	integer	2	01-99	Нет	PK	1231231
Hist_rso	Text	64000	256-64000 символов	Нет		История РСО начинается…

Таблица 35 содержит информацию о сущности «Zakdoc».

Таблица 35 – Сущность «Zakdoc»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_doc	integer	2	01-99	Нет	PK	1231231
Name	Nvarchar	50	‘А-я’,’ ’	Нет		Мирный атом
Descriptions	Text	64000	256-64000 символов	Да		История РСО начинается…

Информацию о сущности «Hist_otr» можно увидеть в таблице 36.

Таблица 36 – Сущность «Hist_otr»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_otr	integer	2	01-99	Нет	PK	1231231
Name	Nvarchar	50	‘А-я’,’ ’	Нет		Мирный атом
Hist	Text	64000	256-64000	Да		История РСО

Таблица 37 содержит информацию о сущности «EducationsInfo».

Таблица 37 – Сущность «EducationsInfo»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_napr	integer	2	01-99	Нет	PK	1231231
Name	Nvarchar	50	‘А-я’,’ ’	Нет		Мирный атом
Descriptions	Text	64000	256-64000 символов	Да		История РСО начинается…

Далее будет рассмотрена сущность «Events»(таблица 38).

Таблица 38 – Сущность «Events»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
ID_event	integer	2	01-99	Нет	PK	12
image	image	-	‘А-я’,’ ’	Нет		Oshk.jpg
Name	Nvarchar	50	‘А-я’,’ ’	Нет		Областная
Quick_descriptions	Nvarchar	50	‘А-я’,’ ’	Нет		Мирный атом

Продолжение таблицы 38

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Count	integer	3	001-999	Да		123
Descriptions	Nvarchar	MAX	256-64000 символов	Да		История РСО начинается…
Date	Datetime	10	01.01.1950 – 31.12.2100	Да		01.01.2003
Time	Datetime	8	00:00:00	Да		23:59:59
Place	Nvarchar	50	‘А-я’,’ ’	Да		Мирный атом

Далее была реализована группа таблиц для организации диалогов. Рассмотрим сущность «Message» (таблица 39).

Таблица 39 – Сущность «Message»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Time	Datetime	8	00:00:00	Нет		23:59:59
ID_sender	integer	7	0000001-00000099	Нет	PK	11
ID_receive	integer	7	0000001-00000099	Да	PK	123123
ID_chatroom	integer	7	0000001-00000099	Да	PK	123123
Message	varchar	Max	‘А-я’,’ ’,”.”	Нет	Нет	
Visible	integer	1	0,1	Нет		1
Reader	integer	1	0,1	Нет		1

Сущность «Chatroom» описана в таблице 40.

Таблица 40 – Сущность «Chatroom»

Название	Тип данных	Размерность	Область ДЗ	Возможность Null	Роль	Пример
Chatroom	Nvarchar	50	‘А-я’,’ ’	Нет		Мирный атом
Users	Nvarchar	50	‘A-z,’ ’ ,’=’	Нет		{ID = 1}
image	image	-	‘А-я’,’ ’	Да		1.jpg

Таким образом, была спроектирована физическая модель базы данных разрабатываемого приложения (рисунок 43).

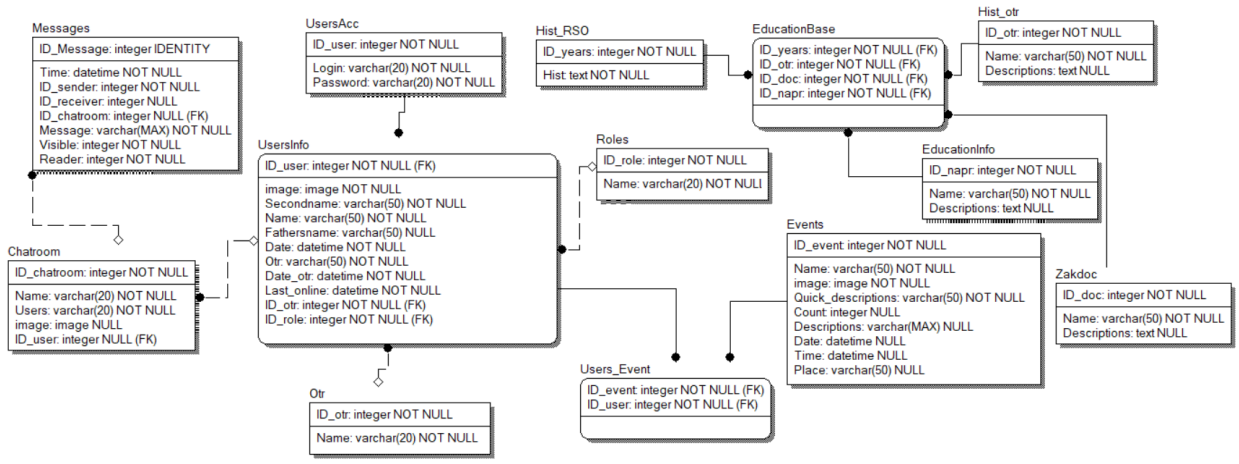

Рисунок 43 – Физическая модель разрабатываемой базы данных

Данная диаграмма показывает структуру используемой базы данных, а также, какие сущности связаны между собой и позволяет проследить перемещение вторичных ключей, для оптимизации выборки контроллерами.

3.5 Развертывание приложения

Разработка мобильного приложения осуществлялась на языке программирования C# в среде Visual Studio с дополнением Xamarin. Листинг классов приведен в приложении А. На рисунке 44 приведена диаграмма компонентов, показывающая структуру приложения, включая исходные коды, файлы интерфейса и библиотеки.

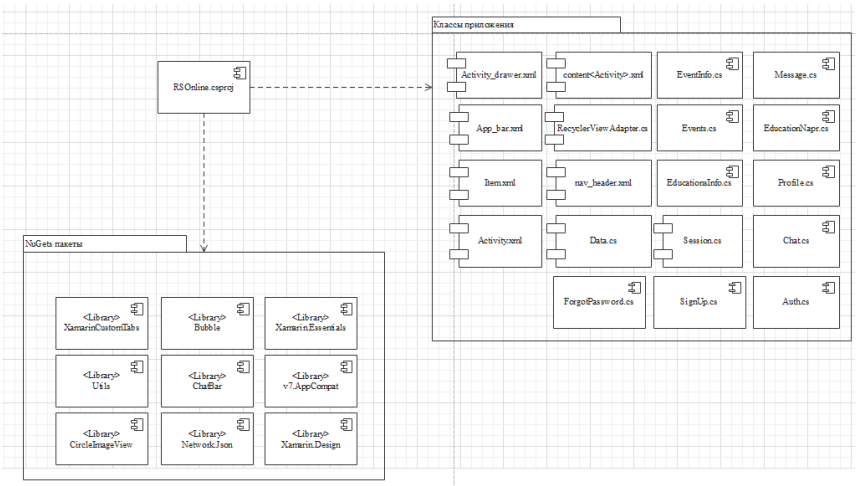

Рисунок 44 – Диаграмма компонентов

Главный файл проекта зависит от библиотек, устанавливаемых с помощью NuGet (система управления пакетами), и от разработанных классов приложения, включающих в себя классы представлений. Исходные коды всех контроллеров приведены в приложении А. В таблице 41 описано назначение использованных библиотек.

Таблица 41 – Назначение использованных библиотек

Название пакета	Назначение
Bubble	Библиотека для создания привычного окна чата, а именно всплывающих «облаков», содержащих текст сообщения.
Название пакета	Назначение
AppCompat	Библиотека-помощник при создании пользовательского интерфейса
ChatBar	Для создания чатбара (Необычное поле ввода + кнопки ввода, голосового ввода)
Xamarin.CustomTabs	Библиотека-помощник при создании таблиц
CircleimageView	Для обрезания фотографий к привычному круглому виду
Network.Json	Библиотека для сериализации и десериализации данных. Используется при передаче/получении запросов к серверу
Xamarin.Design	Библиотека-помощник при создании пользовательского интерфейса
Xamarin.Essentials	Библиотека для использования языка Xamarin в Visual Studio.
Xamarin.Utils	Библиотека для использования языка Xamarin в Visual Studio.

На рисунке 45 показана диаграмма развертывания приложения.

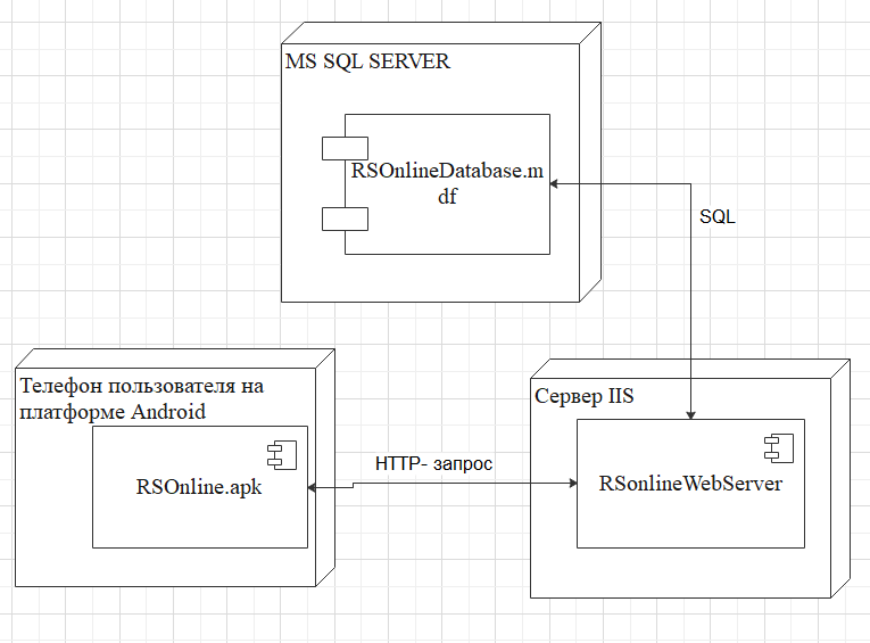

Рисунок 45 – Диаграмма развертывания

На вышеприведенной диаграмме показано, что клиентская часть приложения работает на мобильных устройствах с платформами Android. Приложение связывается с локальным сервером IIS через HTTP – запросы и подключается к контроллерам, обрабатывающим их. В свою очередь сервер непосредственно работает с базой данных приложения через SQL - запросы, а именно с MS SQL Server.

Таким образом, приложение «RSOnline» было реализовано в соответствии с поставленным заданием и на основе анализа требований данного проекта.

4 Тестирование мобильного приложения

4.1 Функциональное тестирование

На основе выявленных при анализе требований вариантов использования, был построен план функционального тестирования (таблица 42) [12,13].

Таблица 42 – План функционального тестирования

Вариант использования	Тест	Результаты
Зарегистрироваться в приложении	Зарегистрироваться в приложении с логином «kolokolcev», паролем «123» и номером членского билета «1234567»	Тест пройден. Новый аккаунт успешно создан. Авторизация с новыми данными прошла успешно (рисунок Б.1 и Б.2).
Авторизоваться в приложении	Вход в приложение с логином «kolokolcev» и паролем «123»	Тест пройден. Проверка данных прошла успешно и пользователь был переведен на страницу с входящими сообщениями (рисунок Б.2 и Б.3).
Написать сообщение	Написать любому контакту «Привет от функционального тестирования»	Тест пройден. Сообщение успешно было отправлено в чат (рисунок Б.4 и Б.5).
Создать диалог	Создать диалог «Областная школа»	Тест пройден. Диалог был создан (рисунок Б.6 и Б.7).
Создать мероприятие	Создать мероприятие и добавить информацию о нем	Тест пройден. Мероприятие было успешно создано и отображено в доступных у всех пользователей (рисунок Б.8 и Б.9, Б.10).
Посмотреть информацию предоставляемую приложением	Зайти в раздел «информационная база»	Тест пройден. Переход на страницу информационного портала был успешно осуществлен (рисунок Б.11).
Посмотреть информацию об определенном отряде	Зайти в раздел «информационная база» и выбрать подраздел	Тест пройден. Выбор отряда и прочтение подробной информации о нем было произведено успешно (рисунок Б.11).

Продолжение таблицы 42

Вариант использования	Тест	Результаты
Редактировать информацию о себе	Изменить фамилию	Тест пройден. Фамилия была успешно изменена как в представлении, так и в базе данных (рисунок Б.12).
Посмотреть список существующих мероприятий	Просмотреть список доступных мероприятий в разделе «Мероприятия»	Тест пройден. Переход на страницу со списком мероприятий был произведен успешно (рисунок Б.13).
Стать участником мероприятия	Зайти в подробное описание мероприятия и стать участником	Тест пройден. Просмотр и нажатие на кнопку «Стать участником» было произведено успешно. База данных была обновлена (рисунок Б.12).

Вышеперечисленные тесты показывают полную работоспособность разработанного приложения, а так же соответствие функциональным требованиям. Результат тестирования был приведен на рисунках в приложении Б.

4.2 Структурное тестирование

Далее будет рассмотрено тестирование метода считывания общей информации о пользователе контроллера UserController (листинг 3). Тестирование будет произведено методом белого ящика, для учета всех возможных результатов метода [14,15].

Листинг 3 – Метод GetFullInfo

```
public string GetFullInfo(int id)
        {
            string json = "0";
           if (id != 0){
            DataForFullInfo data = new DataForFullInfo();
            Try{var conString =
ConfigurationManager.ConnectionStrings["LocalSqlServer"];
            string strConnString = conString.ConnectionString;
            using (SqlConnection con = new SqlConnection(strConnString))
            {
                string sql = "SELECT * FROM [UsersInfo] WHERE id = '" + id +
"'";
                string otr="0", role ="0";
```

```
            con.Open();
            SqlCommand command = new SqlCommand(sql, con);
            using (SqlDataReader oReader = command.ExecuteReader())
            {
           Try{
                if (oReader.Read())
                {
                    data.image = oReader["image"].ToString();
                    data.Login = oReader["Login"].ToString();
                    data.Secondname = oReader["Secondname"].ToString();
                    data.Name = oReader["Name"].ToString();
                    data.Fathersname = oReader["Fathersname"].ToString();
                    data.date = oReader["date"].ToString();
                  data.dateOtr = oReader["dateOtr"].ToString();
                    otr = oReader["otr"].ToString();
                    role = oReader["role"].ToString();
                }}catch{ json = "Ошибка считывания данных";}
            }
            con.Close();
        }
        json = JsonConvert.SerializeObject(data);
        return json;
    }}catch{

            json = "Ошибка соединения с Sql сервером"}

        }

}else{ json = "Переданные данные с запросом некорректны"}
```

Блок-схема на основе листинга 3 приведена на рисунке 46.

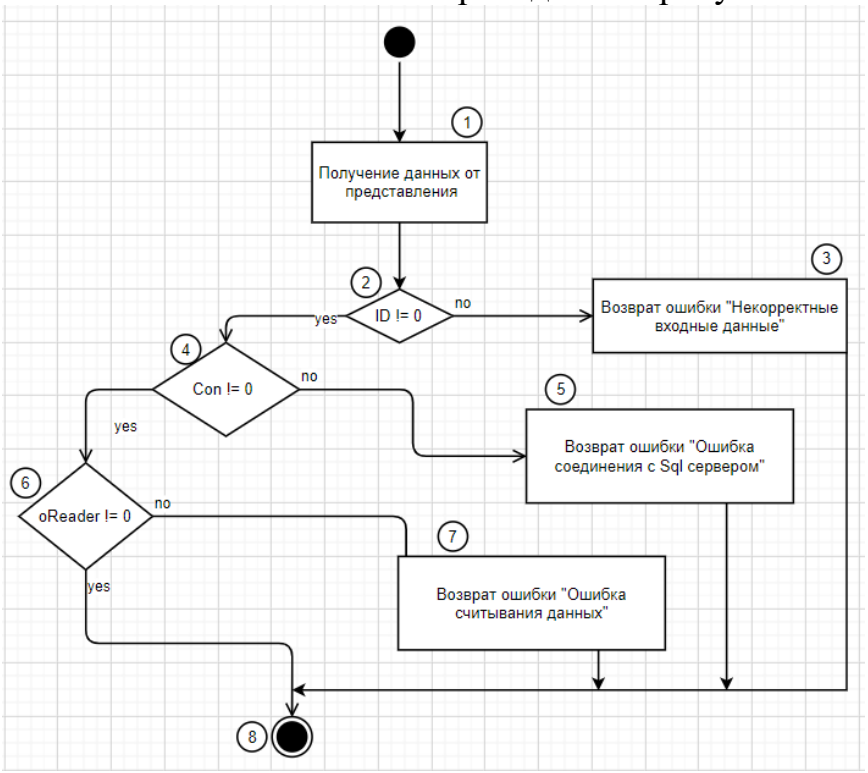

Рисунок 46 – Блок-схема для листинга 3

Создадим управляющий граф на основе построенной блок-схемы (рисунок 47).

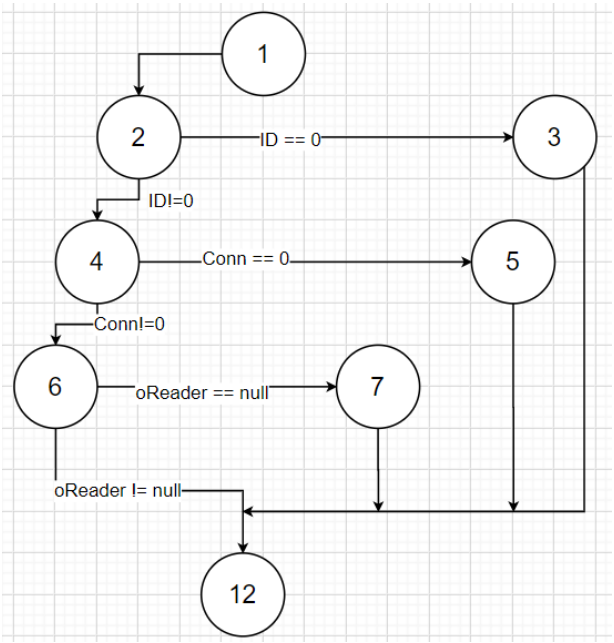

Рисунок 47 – Управляющий граф

Построим таблицы с тестами проверки покрытия операторов, решений и маршрутов, согласно вышеприведенным диаграммам.

Пусть ID, Conn, oReader, oReader1, oReader2 – это A, B, C, D, E соответственно. Результаты тестов представлены в таблицах ниже. Рассмотрим покрытие операторов (таблица 43).

Таблица 43 – Покрытие операторов

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
ID == 0; Conn != 0; oReader != null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке	Успешно
ID != 0; Conn != 0; oReader != null;	Данные успешно выбраны	Данные успешно выбраны	Успешно

Рассмотрим покрытие решений на таблице 44.

Таблица 44 – Покрытие решений

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
ID == 0; Conn != 0; oReader != null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке «Переданные данные с запросом некорректны»	Успешно
ID != 0; Conn == 0; oReader != null;	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Успешно
ID != 0; Conn != 0; oReader == null;	Сообщение об ошибке «Ошибка считывания данных»	Сообщение об ошибке «Ошибка считывания данных»	Успешно
ID != 0; Conn != 0; oReader != null;	Данные успешно считаны из базы данных	Данные успешно считаны из базы данных	Успешно

Покрытие условий описано в таблице 45.

Таблица 45 – Покрытие условий

Тест	Ожидаемый результат	Фактический результат	Результат тестирования
ID != 0; Conn != 0; oReader != null;	Данные успешно считаны из базы данных	Данные успешно считаны из базы данных	Успешно
ID != 0; Conn != 0; oReader == null;	Сообщение об ошибке	Сообщение об ошибке	Успешно
ID != 0; Conn == 0; oReader != null;	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Успешно
ID != 0; Conn == 0; oReader == null;	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Сообщение об ошибке «Ошибка соединения с Sql сервером»	Успешно

Продолжение таблицы 45

ID == 0; Conn != 0; oReader != null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке «Переданные данные с запросом некорректны»	Успешно
ID == 0; Conn != 0; oReader == null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке «Переданные данные с запросом некорректны»	Успешно
ID == 0; Conn == 0; oReader != null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке «Переданные данные с запросом некорректны»	Успешно
ID == 0; Conn == 0; oReader == null;	Сообщение об ошибке «Переданные данные с запросом некорректны»	Сообщение об ошибке «Переданные данные с запросом некорректны»	Успешно

По рассмотренным выше таблицам видно, что для покрытия всех операторов необходимо провести всего 2 теста, для покрытия всех решений 4 теста, а для покрытия всех маршрутов 8 тестов. Используя эти данные, можно эффективнее создавать автоматические тесты.

4.3 UI-тестирование

Инструментал Visual Studio позволяет автоматизированно проверить работоспособность представлений в зависимости от пользовательских действий.

Тестирование интерфейса было задано программно, с помощью подключенных библиотек NUnit3TestAdapter. Рассмотрим код, написанный для работы тестов (листинг 4) [16].

Листинг 4 – Класс выполняющий тестирование интерфейса

```
 [TestFixture(Platform.Android)]
    [TestFixture(Platform.iOS)]
    public class Tests
    {
        IApp app;
        Platform platform;
        public Tests(Platform platform)
        {
            this.platform = platform;
        }
        [SetUp]
        public void BeforeEachTest()
        {
            app = ConfigureApp
             .Android

.ApkFile("C:/Users/kolok/source/repos/App5/bin/Debug/com.companyname.app5.apk
")
            .StartApp();
        }

        [Test]

        public void Login()
        {
            app.EnterText(c => c.Id("login_email"), "kolokolcev");
            app.EnterText(c => c.Id("login_password"), "123");
            app.Tap(c => c.Marked("login_btn_login"));
        }
        [Test]
        public void LetsChat()
        {
            app.perform(RecyclerViewActions.actionOnItemAtPosition(o,
click()));
        }
        [Test]
        public void SendMessage()
        {
            app.EnterText(c => c.Id("chatbar"), "Hello");
            app.Tap(c => c.Marked("chatbar"));
        }
        [Test]
        public void GetLeftMenu()
        {
            app.SwipeLeftToRight();
        }
        [Test]
        public void GetListEvent()
        {
            app.SwipeLeftToRight();
            app.Tap(c => c.Marked("nav_event"));
        }
        [Test]
        public void GetDescriptionsEvent()
        {
            app.Tap(RecyclerViewActions.actionOnItemAtPosition(o, click()));
        }
        [Test]
        public void GetInformbase()
        {
            app.SwipeLeftToRight();
```

```
            app.Tap(c => c.Marked("infbase"));
        }
    }
```

Функция «Login» автоматически выполняет проверки ввода данных в текстовые поля логин и пароль, а так же тестирует реакцию интерфейса на нажатие кнопки «Войти».

Функция «LetsChat» имитирует нажатие на первый в списке входящих сообщений чат.

Функция «SendMessage» вводит в текстовое поле сообщение и затем отправляет сообщение, проверяет, добавляется ли новая запись в списке сообщений с пользователем.

Функция «GetLeftMenu» производит свайп вправо и ждет появления выдвижного навигационного меню.

Функция «GetListEvent» имитирует переход на страницу «Мероприятия», нажав на ссылку в навигационном меню.

Функция «GetDescriptionsEvent» нажимает на первый элемент RecyclerView списка, ожидая перехода на страницу с подробным описанием мероприятия.

Функция «Getinformbase» тестирует, совершается ли переход на страницу информационного портала при нажатии на одноименную ссылку в навигационном меню.

Результаты произведенных тестов отображены на рисунке 48.

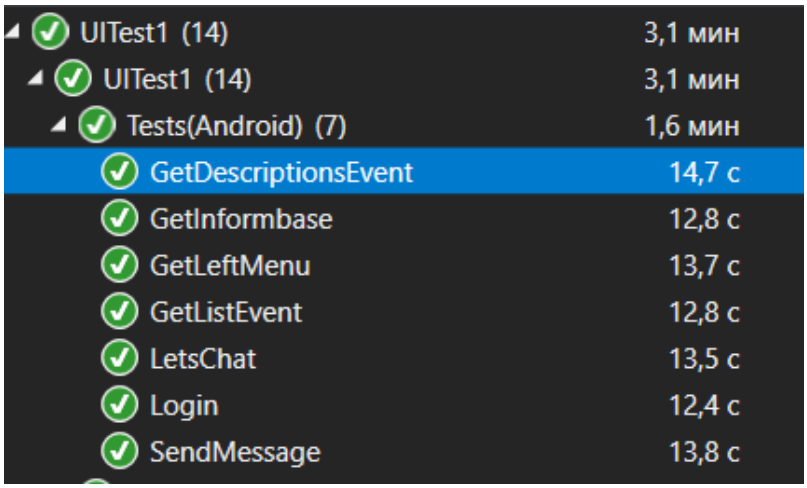

Рисунок 48 – Результаты тестирования

Как видно из результатов, все тесты успешно были выполнены, а значит, пользовательский интерфейс работает без ошибок.

Заключение

При выполнении выпускной квалификационной работы было разработано мобильное приложение «RSOnline» для облегчения корпоративных отношений, а так же планирования и проведения мероприятий в системе молодежной общероссийской общественной организации Российские студенческие отряды. Приложение было сделано для мобильных устройств на платформе Android 5 и выше.

В ходе выполнения был проведен анализ предметной области и постановка задачи. Был произведен анализ аналогичных продуктов на Российском рынке и сравнение их с разрабатываемым приложением методом иерархий.

Была выбрана архитектура реализации приложения, а именно MVC. Так же были спроектированы классы и пользовательский интерфейс приложения. Проведение планирования разработки и оценка бюджета показали, что оптимальная длительность проекта составляет 41 день и предполагаемые затраты 89740 рублей.

При реализации приложения «RSOnline» была использована среда разработки Visual Studio с дополнением Xamarin на языке программирования C#.

При тестировании были использованы методы ручного функционального тестирования, структурное тестирование методом «белого ящика», а так же тестирование пользовательского интерфейса с помощью вложенных методов Visual Studio – UI-tests. Результаты тестирования показали корректную работу приложение и соответствие функциональным и нефункциональным требованиям.

Результатом выполнения выпускной квалификационной работы является приложение «RSOnline», используемое для упрощения корпоративного общения, организации мероприятий в системе Российские студенческие отряды.

Список использованных источников

1. Статья на Wikipedia о ВКонтакте // Электронная энциклопедия [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/ВКонтакте (дата обращения: 13.06.2020).

2. Статья о методе анализа иерархий // Сайт по теме теории и практики оценки экономических величин [Электронный ресурс]. URL: http://vamocenka.ru/metod-analiza-ierarxij-procedura-primeneniya/ (дата обращения: 15.06.2020).

3. Статья «UX/UI-дизайн: что это такое?» // Онлайн-университет Skillbox [Электронный ресурс]. URL: https://skillbox.ru/media/design/ux_ui_dizayn_chto_eto_takoe/ (дата обращения: 20.05.2020).

4. Статья «52 Weeks of Xamarin: Week 11 – Customized Controls for Xamarin.Forms // Информационный ресурс [Электронный ресурс]. URL: http://jesseliberty.com/2015/10/20/52-weeks-of-xamarin-week-11-customized-controls-for-xamarin-forms/ (дата обращения: 20.05.2020).

5. Статья «UML — диаграмма вариантов использования (use case diagram)» // Информационный ресурс [Электронный ресурс]. URL: https://habr.com/ru/post/47940/ (дата обращения: 14.05.2020).

6. Статья «Введение в UML» // Национальный открытый институт [Электронный ресурс]. URL: https://www.intuit.ru/studies/courses/1007/229/lecture/5954 (дата обращения: 02.06.2020).

7. Статья про MVC на Wikipedia // Информационный ресурс [Электронный ресурс]. URL: https://ru.wikipedia.org/wiki/Model-View-Controller (дата обращения: 29.05.2020).

8. Руководство по ASP.Net MVC 5 // Сайт о программировании [Электронный ресурс]. URL: https://metanit.com/sharp/mvc5/ (дата обращения: 29.05.2020).

9. Статья «Создание логической модели хранилища данных» // Информационный ресурс [Электронный ресурс]. URL: https://studwood.ru/1600339/informatika/sozdanie_logicheskoy_modeli_bazy_dannyh (дата обращения: 02.06.2020).

10. Статья «Основы разработки интерфейсов мобильных приложений» // Национальный открытый институт [Электронный ресурс]. URL: https://www.intuit.ru/studies/courses/12643/1191/lecture/21986 (дата обращения: 07.06.2020).

11. Статья о планировании стоимости в MS Project // Информационный ресурс [Электронный ресурс]. URL: http://www.taurion.ru/project/13/2 (дата обращения: 29.05.2020).

12.Функциональное тестирование ПО // Информационный ресурс [Электронный ресурс]. URL: https://www.softwaretesting.ru/library/around-testing/processes/2286-pair-testing (дата обращения: 04.06.2020).

13. Статья о технике анализа классов эквивалентности // Сайт с записями о тестировании программных продуктов и методах тестирования [Электронный ресурс]. URL: https://qaevolution.ru/testovayadokumentaciya/test-dizajn/texnika-analiza-klassov-ekvivalentnosti/ (дата обращения: 21.06.2020).

14.Статья о методе функциональных диаграмм // Сайт с общедоступной информацией для студентов разных предметных областей [Электронный ресурс]. URL: https://studopedia.ru/8_201409_metodfunktsionalnih-diagramm.html (дата обращения: 09.06.2020).

15. Статья о тестирование методом белого ящика // Центр подготовки IT-специалистов QALight [Электронный ресурс]. URL: https://qalight.com.ua/baza-znaniy/white-black-grey-box-testirovanie/ (дата обращения: 18.06.2020).

16. Статья «Тестирование UI (пользовательского интерфейса)» // Сайт компании WOXAPP, занимающейся разработкой мобильных приложений [Электронный ресурс]. URL: https://woxapp.com/ru/our-blog/testing-the-ui-userinterface/ (дата обращения: 06.06.2020).

Приложение А
ИСХОДНЫЙ КОД ПРИЛОЖЕНИЯ

(рекомендуемое)

Класс AllEvents.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using Android;
using Android.App;
using Android.Content;
using Android.Media;
using Android.OS;
using Android.Runtime;
using Android.Support.Design.Widget;
using Android.Support.V4.View;
using Android.Support.V4.Widget;
using Android.Support.V7.App;
using Android.Support.V7.Widget;
using Android.Views;
using Android.Widget;
using Auth.Helper;
using Newtonsoft.Json;

namespace App5
{

   [Activity(Label = "@string/app_name", Theme = "@style/AppTheme.NoActionBar")]
   public class AllEvents : AppCompatActivity, NavigationView.IOnNavigationItemSelectedListener
   {
      private RecyclerView recycler;
      private RecyclerViewAdapterEvents adapter;
      private RecyclerView.LayoutManager layoutManager;
      private List<DataEvents> lstData = new List<DataEvents>();
      List<DataEvents> Devents = new List<DataEvents>();
      private DataForLeftMenu Dsql1 = new DataForLeftMenu();
      private ImageView imageSrc;
      private TextView Fullname;
      private TextView id;

      protected override void OnCreate(Bundle savedInstanceState)
      {
         base.OnCreate(savedInstanceState);
         Xamarin.Essentials.Platform.Init(this, savedInstanceState);
         SetContentView(Resource.Layout.activity_mainAllEvents);
         Android.Support.V7.Widget.Toolbar toolbar =
FindViewById<Android.Support.V7.Widget.Toolbar>(Resource.Id.toolbar);
         SetSupportActionBar(toolbar);

         DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
         ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar,
Resource.String.navigation_drawer_open, Resource.String.navigation_drawer_close);
         drawer.AddDrawerListener(toggle);
         toggle.SyncState();

         NavigationView navigationView = FindViewById<NavigationView>(Resource.Id.nav_view);
         View hv = navigationView.GetHeaderView(0);
         Fullname = hv.FindViewById<TextView>(Resource.Id.Fullname);
         id = hv.FindViewById<TextView>(Resource.Id.ID);
         imageSrc = hv.FindViewById<ImageView>(Resource.Id.imageView);
         navigationView.SetNavigationItemSelectedListener(this);

         Dsql1 = req(Dsql1);
```

```
            string Full = Dsql1.Secondname + " " + Dsql1.Name;
            Fullname.Text = Full;
            id.Text = "ID - " + Sessions.GetSession();
            int resourceId = (int)typeof(Resource.Drawable).GetField(Dsql1.image).GetValue(null);
            imageSrc.SetImageResource(resourceId);

            Devents = GetEvents();
            recycler = FindViewById<RecyclerView>(Resource.Id.recycler);
            recycler.HasFixedSize = true;
            layoutManager = new LinearLayoutManager(this);
            recycler.SetLayoutManager(layoutManager);
            adapter = new RecyclerViewAdapterEvents(Devents, this);
            adapter.ItemClick += OnItemClick;
            recycler.SetAdapter(adapter);


        }

        private void OnItemClick(object sender, int position)
        {
            Intent intent = new Intent(this, typeof(InfoEvents));
            intent.PutExtra("image", Devents[position].image);
            StartActivity(intent);
            Finish();
        }

        public DataForLeftMenu req(DataForLeftMenu Dsql1)
        {
            WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/GetImageFullname");
            request1.Method = "POST";
            string idS = Sessions.GetSession();
            string data1 = "id=" + idS;
            byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
            request1.ContentType = "application/x-www-form-urlencoded";
            request1.ContentLength = byteArray1.Length;
            using (System.IO.Stream dataStream = request1.GetRequestStream())
            {
                dataStream.Write(byteArray1, 0, byteArray1.Length);
            }
            WebResponse response1 = request1.GetResponse();
            string json_string1 = "";
            using (System.IO.Stream stream = response1.GetResponseStream())
            {
                using (StreamReader reader = new StreamReader(stream))
                {
                    json_string1 = reader.ReadToEnd();
                }
            }
            Dsql1 = JsonConvert.DeserializeObject<DataForLeftMenu>(json_string1);
            response1.Close();
            return Dsql1;
        }

        public List<DataEvents> GetEvents()
        {
            WebRequest request =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/GetEventsRecycler");
            request.Method = "POST";
            string data = "";
            byte[] byteArray = System.Text.Encoding.UTF8.GetBytes(data);
            request.ContentType = "application/x-www-form-urlencoded";
```

```
    request.ContentLength = byteArray.Length;
    using (System.IO.Stream dataStream = request.GetRequestStream())
    {
        dataStream.Write(byteArray, 0, byteArray.Length);
    }
    WebResponse response = request.GetResponse();
    string json_string = "";
    using (System.IO.Stream stream = response.GetResponseStream())
    {
        using (StreamReader reader = new StreamReader(stream))
        {
            json_string = reader.ReadToEnd();
        }
    }
    var dsql = JsonConvert.DeserializeObject<List<DataEvents>>(json_string);
    response.Close();
    return dsql;
}
public override void OnBackPressed()
{
    DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
    if(drawer.IsDrawerOpen(GravityCompat.Start))
    {
        drawer.CloseDrawer(GravityCompat.Start);
    }
    else
    {
        base.OnBackPressed();
    }
}

public override bool OnCreateOptionsMenu(IMenu menu)
{
    MenuInflater.Inflate(Resource.Menu.menu_main, menu);
    return true;
}

public override bool OnOptionsItemSelected(IMenuItem item)
{
    int id = item.ItemId;
    if (id == Resource.Id.action_settings)
    {
        return true;
    }

    return base.OnOptionsItemSelected(item);
}


public bool OnNavigationItemSelected(IMenuItem item)
{
    int id = item.ItemId;

    if (id == Resource.Id.nav_profile)
    {
        StartActivity(new Android.Content.Intent(this, typeof(Profile)));
        Finish();
    }
    else if (id == Resource.Id.nav_message)
    {
        StartActivity(new Android.Content.Intent(this, typeof(MainActivity)));
        Finish();
```

```
            }
            else if (id == Resource.Id.nav_event)
            {
            }
            else if (id == Resource.Id.nav_infbase)
            {
               StartActivity(new Android.Content.Intent(this, typeof(Infomenu)));
               Finish();
            }
            else if (id == Resource.Id.nav_Logout)
            {
               Sessions.stopsession();
               StartActivity(new Android.Content.Intent(this, typeof(Login)));
               Finish();
            }

            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            drawer.CloseDrawer(GravityCompat.Start);
            return true;
         }
         public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
Android.Content.PM.Permission[] grantResults)
         {
            Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

            base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
         }
      }
}
```

Класс Message.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Net;
using System.Text;

using Android.App;
using Android.Content;
using Android.OS;
using Android.Runtime;
using Android.Support.V7.Widget;
using Android.Views;
using Android.Widget;
using App5;
using Auth.Helper;
using Newtonsoft.Json;

namespace Auth.Activities
{
   [Activity(Label = "Messages")]
   public class Messages : Activity
   {
      private RecyclerView recycler;
      private RecViewAdapterMessage adapter;
      private RecyclerView.LayoutManager layoutManager;
      private List<DataForMessage> lstData = new List<DataForMessage>();
      private List<DataForMessageFromSql> Dsql = new List<DataForMessageFromSql>();
      protected override void OnCreate(Bundle savedInstanceState)
      {
         base.OnCreate(savedInstanceState);
```

```
    SetContentView(Resource.Layout.Messages);
    InitData();

    recycler = FindViewById<RecyclerView>(Resource.Id.recycler);
    recycler.HasFixedSize = true;
    layoutManager = new LinearLayoutManager(this);
    recycler.SetLayoutManager(layoutManager);
    adapter = new RecViewAdapterMessage(lstData, this);
    adapter.ItemClick += OnItemClick;
    recycler.SetAdapter(adapter);

    // Create your application here
}

private void InitData()
{
    WebRequest request = WebRequest.Create("http://192.168.56.1/WebApplication1/Home/Message");
    request.Method = "POST";
    string idSS = Sessions.GetSession();
    string login = "kolokolcev";
    string data = "id="+"1234567";
    byte[] byteArray = System.Text.Encoding.UTF8.GetBytes(data);
    request.ContentType = "application/x-www-form-urlencoded";
    request.ContentLength = byteArray.Length;
    using (System.IO.Stream dataStream = request.GetRequestStream())
    {
        dataStream.Write(byteArray, 0, byteArray.Length);
    }
    WebResponse response = request.GetResponse();
    string json_string = "";
    using (System.IO.Stream stream = response.GetResponseStream())
    {
        using (StreamReader reader = new StreamReader(stream))
        {
            json_string = reader.ReadToEnd();
        }
    }

    Dsql = JsonConvert.DeserializeObject<List<DataForMessageFromSql>>(json_string);
    response.Close();
    int i = 0;

    while (i < Dsql.Count)
    {
        lstData.Add(new DataForMessage()
        {
            image = Dsql[i].image,
            Fullname = Dsql[i].Secondname + " " + Dsql[i].name,
            time = Dsql[i].time,
            message = Dsql[i].message
        });
    }



}
void OnItemClick(object sender, int position)
{
    //Intent intent = new Intent(this, typeof(AboutOtr));
   // intent.PutExtra("id", position);
   // StartActivity(intent);
```

```
            //Finish();
        }
    }
}
```

Класс Chat.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Net;
using Android;
using Android.App;
using Android.Media;
using Android.OS;
using Android.Runtime;
using Android.Support.Design.Widget;
using Android.Support.V4.View;
using Android.Support.V4.Widget;
using Android.Support.V7.App;
using Android.Support.V7.Widget;
using Android.Views;
using Android.Widget;
using App5.Helper;
using Auth.Helper;
using Cenkgun.ChatBar;
using Newtonsoft.Json;

namespace App5
{
    [Activity(Label = "@string/app_name", Theme = "@style/AppTheme.NoActionBar")]
    public class Chat : AppCompatActivity, NavigationView.IOnNavigationItemSelectedListener
    {

        private RecyclerView recycler;
        private RecyclerViewAdapterChat adapter;
        private RecyclerView.LayoutManager layoutManager;

        private DataForLeftMenu Dsql1 = new DataForLeftMenu();
        private List<DataForChat> Sqlchat = new List<DataForChat>();
        private ImageView imageSrc;
        private TextView Fullname;
        private TextView id;
        string image, Name, Secondname;
        protected override void OnCreate(Bundle savedInstanceState)
        {
            image = Intent.GetStringExtra("image");
            Name = Intent.GetStringExtra("Name");
            Secondname = Intent.GetStringExtra("Secondname");
            base.OnCreate(savedInstanceState);
            Xamarin.Essentials.Platform.Init(this, savedInstanceState);
            SetContentView(Resource.Layout.activity_mainChat);
            Android.Support.V7.Widget.Toolbar toolbar =
FindViewById<Android.Support.V7.Widget.Toolbar>(Resource.Id.toolbar);
            SetSupportActionBar(toolbar);

            //FloatingActionButton fab = FindViewById<FloatingActionButton>(Resource.Id.fab);
            //fab.Click += FabOnClick;

            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar,
Resource.String.navigation_drawer_open, Resource.String.navigation_drawer_close);
            drawer.AddDrawerListener(toggle);
```

```
            toggle.SyncState();

            NavigationView navigationView = FindViewById<NavigationView>(Resource.Id.nav_view);
            View hv = navigationView.GetHeaderView(0);
            Fullname = hv.FindViewById<TextView>(Resource.Id.Fullname);
            id = hv.FindViewById<TextView>(Resource.Id.ID);
            imageSrc = hv.FindViewById<ImageView>(Resource.Id.imageView);
            navigationView.SetNavigationItemSelectedListener(this);

            Dsql1 = req(Dsql1);
            ChatBarView chb = FindViewById<ChatBarView>(Resource.Id.chatbar);
            chb.SetMessageBoxHint("Введите сообщение..");
            string Full = Dsql1.Secondname + " " + Dsql1.Name;
            Fullname.Text = Full;
            id.Text = "ID - " + Sessions.GetSession();
            int resourceId = (int)typeof(Resource.Drawable).GetField(Dsql1.image).GetValue(null);
            imageSrc.SetImageResource(resourceId);
            ImageView img_top = (ImageView)FindViewById(Resource.Id.image_chat);
            TextView txt = (TextView)FindViewById(Resource.Id.Fname);
            int resourceId1 = (int)typeof(Resource.Drawable).GetField(image).GetValue(null);
            img_top.SetImageResource(resourceId1);
            txt.Text = Secondname + " " + Name;



            Sqlchat = GetMessages(Sqlchat);

            recycler = FindViewById<RecyclerView>(Resource.Id.recycler);
            recycler.HasFixedSize = true;
            layoutManager = new LinearLayoutManager(this);
            recycler.SetLayoutManager(layoutManager);
            adapter = new RecyclerViewAdapterChat(Sqlchat, this);
            recycler.SetAdapter(adapter);
        }
        public DataForLeftMenu req(DataForLeftMenu Dsql1)
        {
            WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/GetImageFullname");
            request1.Method = "POST";
            string idS = Sessions.GetSession();
            string data1 = "id=" + idS;
            byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
            request1.ContentType = "application/x-www-form-urlencoded";
            request1.ContentLength = byteArray1.Length;
            using (System.IO.Stream dataStream = request1.GetRequestStream())
            {
                dataStream.Write(byteArray1, 0, byteArray1.Length);
            }
            WebResponse response1 = request1.GetResponse();
            string json_string1 = "";
            using (System.IO.Stream stream = response1.GetResponseStream())
            {
                using (StreamReader reader = new StreamReader(stream))
                {
                    json_string1 = reader.ReadToEnd();
                }
            }
            Dsql1 = JsonConvert.DeserializeObject<DataForLeftMenu>(json_string1);
            response1.Close();
            return Dsql1;
        }
        public List<DataForChat> GetMessages(List<DataForChat> Dsql1)
```

```
{
    WebRequest request1 = WebRequest.Create("http://192.168.56.1/WebApplication1/Home/Chat");
    request1.Method = "POST";
    string idS = Sessions.GetSession();
    string data1 = "id=" + idS+"&image="+image;
    byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
    request1.ContentType = "application/x-www-form-urlencoded";
    request1.ContentLength = byteArray1.Length;
    using (System.IO.Stream dataStream = request1.GetRequestStream())
    {
        dataStream.Write(byteArray1, 0, byteArray1.Length);
    }
    WebResponse response1 = request1.GetResponse();
    string json_string1 = "";
    using (System.IO.Stream stream = response1.GetResponseStream())
    {
        using (StreamReader reader = new StreamReader(stream))
        {
            json_string1 = reader.ReadToEnd();
        }
    }
    Dsql1 = JsonConvert.DeserializeObject<List<DataForChat>>(json_string1);
    response1.Close();
    return Dsql1;
}
public override void OnBackPressed()
{
    DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
    if(drawer.IsDrawerOpen(GravityCompat.Start))
    {
        drawer.CloseDrawer(GravityCompat.Start);
    }
    else
    {
        base.OnBackPressed();
    }
}

public override bool OnCreateOptionsMenu(IMenu menu)
{
    MenuInflater.Inflate(Resource.Menu.menu_main, menu);
    return true;
}

public override bool OnOptionsItemSelected(IMenuItem item)
{
    int id = item.ItemId;
    if (id == Resource.Id.action_settings)
    {
        return true;
    }

    return base.OnOptionsItemSelected(item);
}

private void FabOnClick(object sender, EventArgs eventArgs)
{
    View view = (View) sender;
    Snackbar.Make(view, "Replace with your own action", Snackbar.LengthLong)
        .SetAction("Action", (Android.Views.View.IOnClickListener)null).Show();
}
```

```
        public bool OnNavigationItemSelected(IMenuItem item)
        {
            int id = item.ItemId;

            if (id == Resource.Id.nav_profile)
            {
                StartActivity(new Android.Content.Intent(this, typeof(Profile)));
                Finish();
            }
            else if (id == Resource.Id.nav_message)
            {
                StartActivity(new Android.Content.Intent(this, typeof(MainActivity)));
                Finish();
            }
            else if (id == Resource.Id.nav_event)
            {

            }
            else if (id == Resource.Id.nav_infbase)
            {
                StartActivity(new Android.Content.Intent(this, typeof(Infomenu)));
                Finish();
            }

            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            drawer.CloseDrawer(GravityCompat.Start);
            return true;
        }
        public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
Android.Content.PM.Permission[] grantResults)
        {
            Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

            base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
        }
    }
}
```

Класс InfoEvents.cs

```
using System;
using System.IO;
using System.Net;
using Android;
using Android.App;
using Android.Media;
using Android.OS;
using Android.Runtime;
using Android.Support.Design.Widget;
using Android.Support.V4.View;
using Android.Support.V4.Widget;
using Android.Support.V7.App;
using Android.Views;
using Android.Widget;
using Auth.Helper;
```

```
using Newtonsoft.Json;
using static Android.Views.View;

namespace App5
{
    [Activity(Label = "@string/app_name", Theme = "@style/AppTheme.NoActionBar")]
    public class InfoEvents : AppCompatActivity, NavigationView.IOnNavigationItemSelectedListener,
IOnClickListener
    {
        string image;
        private DataForLeftMenu Dsql1 = new DataForLeftMenu();
        private DataEventsUsers DsqlEv = new DataEventsUsers();
        private ImageView imageSrc;
        private TextView Fullname;
        private TextView id;
        LinearLayout LLBack;
        TextView Visit,org;
        TextView Name, Descriptions, Date, Time, Place, Count;
        protected override void OnCreate(Bundle savedInstanceState)
        {
            image = Intent.GetStringExtra("image");
            base.OnCreate(savedInstanceState);
            Xamarin.Essentials.Platform.Init(this, savedInstanceState);
            SetContentView(Resource.Layout.activity_mainInfoEvents);
            Android.Support.V7.Widget.Toolbar toolbar =
FindViewById<Android.Support.V7.Widget.Toolbar>(Resource.Id.toolbar);
            SetSupportActionBar(toolbar);

           // FloatingActionButton fab = FindViewById<FloatingActionButton>(Resource.Id.fab);
           // fab.Click += FabOnClick;

            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar,
Resource.String.navigation_drawer_open, Resource.String.navigation_drawer_close);
            drawer.AddDrawerListener(toggle);
            toggle.SyncState();

            NavigationView navigationView = FindViewById<NavigationView>(Resource.Id.nav_view);
            View hv = navigationView.GetHeaderView(0);
            Fullname = hv.FindViewById<TextView>(Resource.Id.Fullname);
            id = hv.FindViewById<TextView>(Resource.Id.ID);
            imageSrc = hv.FindViewById<ImageView>(Resource.Id.imageView);
```

```
navigationView.SetNavigationItemSelectedListener(this);

Dsql1 = req(Dsql1);

string Full = Dsql1.Secondname + " " + Dsql1.Name;
Fullname.Text = Full;
id.Text = "ID - " + Sessions.GetSession();
int resourceId = (int)typeof(Resource.Drawable).GetField(Dsql1.image).GetValue(null);
imageSrc.SetImageResource(resourceId);
DsqlEv = GetEventsInfo(DsqlEv,image);

LLBack = (LinearLayout)FindViewById(Resource.Id.LLBackInfoEvent);
Name = (TextView)FindViewById(Resource.Id.NameEventsInfo);
Descriptions = (TextView)FindViewById(Resource.Id.DescEventsInfo);
Date = (TextView)FindViewById(Resource.Id.DataEvent);
Time = (TextView)FindViewById(Resource.Id.Timestart);
Place = (TextView)FindViewById(Resource.Id.PlaceEvent);
//Count = (TextView)FindViewById(Resource.Id.CountEvent);

int resourceId1 = (int)typeof(Resource.Drawable).GetField(image).GetValue(null);
LLBack.SetBackgroundResource(resourceId1);
Name.Text = DsqlEv.Name;
Descriptions.Text = DsqlEv.Descriptions;
Date.Text = DsqlEv.Date;
Time.Text = DsqlEv.Time;
Place.Text = DsqlEv.Place;
// Count.Text = DsqlEv.Count;

Visit = (TextView)FindViewById(Resource.Id.Visits);
org = (TextView)FindViewById(Resource.Id.org);
if (DsqlEv.Status == "0")
{
    Visit.Text = "Стать участником";
    org.Text = "Стать организатором";
}
else if (DsqlEv.Status == "1")
{
    Visit.Text = "Покинуть мероприятие";
    org.Text = "Стать организатором";
}
else if (DsqlEv.Status == "2")
{
```

```
            Visit.Text = "Стать участником";
            org.Text = "Покинуть статус организатора";
        }
        Visit.SetOnClickListener(this);

    }
    public DataForLeftMenu req(DataForLeftMenu Dsql1)
    {
        WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/GetImageFullname");
        request1.Method = "POST";
        string idS = Sessions.GetSession();
        string data1 = "id=" + idS;
        byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
        request1.ContentType = "application/x-www-form-urlencoded";
        request1.ContentLength = byteArray1.Length;
        using (System.IO.Stream dataStream = request1.GetRequestStream())
        {
            dataStream.Write(byteArray1, 0, byteArray1.Length);
        }
        WebResponse response1 = request1.GetResponse();
        string json_string1 = "";
        using (System.IO.Stream stream = response1.GetResponseStream())
        {
            using (StreamReader reader = new StreamReader(stream))
            {
                json_string1 = reader.ReadToEnd();
            }
        }
        Dsql1 = JsonConvert.DeserializeObject<DataForLeftMenu>(json_string1);
        response1.Close();
        return Dsql1;
    }
    public DataEventsUsers GetEventsInfo(DataEventsUsers Dsql1,string img)
    {
        WebRequest request1 = WebRequest.Create("http://192.168.56.1/WebApplication1/Home/UsersEvents");
        request1.Method = "POST";
        string idS = Sessions.GetSession();
        string data1 = "image=" + img + "&users_id="+idS;
        byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
        request1.ContentType = "application/x-www-form-urlencoded";
        request1.ContentLength = byteArray1.Length;
```

```
    using (System.IO.Stream dataStream = request1.GetRequestStream())
    {
        dataStream.Write(byteArray1, 0, byteArray1.Length);
    }
    WebResponse response1 = request1.GetResponse();
    string json_string1 = "";
    using (System.IO.Stream stream = response1.GetResponseStream())
    {
        using (StreamReader reader = new StreamReader(stream))
        {
            json_string1 = reader.ReadToEnd();
        }
    }
    Dsql1 = JsonConvert.DeserializeObject<DataEventsUsers>(json_string1);
    response1.Close();
    return Dsql1;
}

public override void OnBackPressed()
{
    DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
    if(drawer.IsDrawerOpen(GravityCompat.Start))
    {
        drawer.CloseDrawer(GravityCompat.Start);
    }
    else
    {
        base.OnBackPressed();
    }
}

public override bool OnCreateOptionsMenu(IMenu menu)
{
    MenuInflater.Inflate(Resource.Menu.menu_main, menu);
    return true;
}

public override bool OnOptionsItemSelected(IMenuItem item)
{
    int id = item.ItemId;
    if (id == Resource.Id.action_settings)
    {
```

```
        return true;
    }

    return base.OnOptionsItemSelected(item);
}

private void FabOnClick(object sender, EventArgs eventArgs)
{
    View view = (View) sender;
    Snackbar.Make(view, "Replace with your own action", Snackbar.LengthLong)
        .SetAction("Action", (Android.Views.View.IOnClickListener)null).Show();
}

public bool OnNavigationItemSelected(IMenuItem item)
{
    int id = item.ItemId;

    if (id == Resource.Id.nav_profile)
    {
        StartActivity(new Android.Content.Intent(this, typeof(Profile)));
        Finish();
    }
    else if (id == Resource.Id.nav_message)
    {
        StartActivity(new Android.Content.Intent(this, typeof(MainActivity)));
        Finish();
    }
    else if (id == Resource.Id.nav_event)
    {

    }
    else if (id == Resource.Id.nav_infbase)
    {
        StartActivity(new Android.Content.Intent(this, typeof(Infomenu)));
        Finish();
    }

    DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
    drawer.CloseDrawer(GravityCompat.Start);
    return true;
}
```

```
    public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
Android.Content.PM.Permission[] grantResults)
    {
        Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

        base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
    }

    public void OnClick(View v)
    {//Кликает на кнопку только один раз, дальше не слушает

        if (v.Id == Resource.Id.Visits)
        {
            if (Visit.Text == "Стать участником")
            {
                WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/SetVisitStatus");
                request1.Method = "POST";
                string idS = Sessions.GetSession();
                string data1 = "image=" + image + "&users_id=" + idS;
                byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
                request1.ContentType = "application/x-www-form-urlencoded";
                request1.ContentLength = byteArray1.Length;
                using (System.IO.Stream dataStream = request1.GetRequestStream())
                {
                    dataStream.Write(byteArray1, 0, byteArray1.Length);
                }

                Visit = (TextView)FindViewById(Resource.Id.Visits);
                Visit.Text = "Покинуть мероприятие";
                Count.Text = Convert.ToString(Convert.ToInt32(Count.Text) + 1);
            }
            if (Visit.Text == "Покинуть мероприятие")
            {



                Visit = (TextView)FindViewById(Resource.Id.Visits);
                Visit.Text = "Стать участником";
                Count.Text = Convert.ToString(Convert.ToInt32(Count.Text) - 1);
            }
        }
```

```
        }
    }
}
```

Класс InfoMenu.cs

```
using System;
using System.IO;
using System.Net;
using Android;
using Android.App;
using Android.Media;
using Android.OS;
using Android.Runtime;
using Android.Support.Design.Widget;
using Android.Support.V4.View;
using Android.Support.V4.Widget;
using Android.Support.V7.App;
using Android.Views;
using Android.Widget;
using Auth.Helper;
using Newtonsoft.Json;
using static Android.Views.View;

namespace App5
{
    [Activity(Label = "@string/app_name", Theme = "@style/AppTheme.NoActionBar")]
    public class Infomenu : AppCompatActivity, NavigationView.IOnNavigationItemSelectedListener,
IOnClickListener
    {

        private DataForLeftMenu Dsql1 = new DataForLeftMenu();
        private ImageView imageSrc;
        private TextView Fullname;
        private TextView id;

        TextView Zakdoc, HistRSO, otrhist, obrp;
        protected override void OnCreate(Bundle savedInstanceState)
        {
            base.OnCreate(savedInstanceState);
            Xamarin.Essentials.Platform.Init(this, savedInstanceState);
            SetContentView(Resource.Layout.activity_mainInfo);
            Android.Support.V7.Widget.Toolbar toolbar =
FindViewById<Android.Support.V7.Widget.Toolbar>(Resource.Id.toolbar);
            SetSupportActionBar(toolbar);


            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            ActionBarDrawerToggle toggle = new ActionBarDrawerToggle(this, drawer, toolbar,
Resource.String.navigation_drawer_open, Resource.String.navigation_drawer_close);
            drawer.AddDrawerListener(toggle);
            toggle.SyncState();

            NavigationView navigationView = FindViewById<NavigationView>(Resource.Id.nav_view);
            View hv = navigationView.GetHeaderView(0);
            Fullname = hv.FindViewById<TextView>(Resource.Id.Fullname);
            id = hv.FindViewById<TextView>(Resource.Id.ID);
            imageSrc = hv.FindViewById<ImageView>(Resource.Id.imageView);
            navigationView.SetNavigationItemSelectedListener(this);

            Zakdoc = (TextView)FindViewById(Resource.Id.Zakdoc);
            HistRSO = (TextView)FindViewById(Resource.Id.histRSO);
```

```
            otrhist = (TextView)FindViewById(Resource.Id.Histotr);
            obrp = (TextView)FindViewById(Resource.Id.obrp);
            Zakdoc.SetOnClickListener(this);
           HistRSO.SetOnClickListener(this);
           otrhist.SetOnClickListener(this);
           obrp.SetOnClickListener(this);



            Dsql1 = req(Dsql1);

            string Full = Dsql1.Secondname + " " + Dsql1.Name;
            Fullname.Text = Full;
            string idS = Sessions.GetSession();
            id.Text = "ID - " + idS;
            int resourceId = (int)typeof(Resource.Drawable).GetField(Dsql1.image).GetValue(null);
            imageSrc.SetImageResource(resourceId);
        }
        public DataForLeftMenu req(DataForLeftMenu Dsql1)
        {
            WebRequest request1 =
WebRequest.Create("http://192.168.56.1/WebApplication1/Home/GetImageFullname");
            request1.Method = "POST";
            string idS = Sessions.GetSession();
            string data1 = "id=" + idS;
            byte[] byteArray1 = System.Text.Encoding.UTF8.GetBytes(data1);
            request1.ContentType = "application/x-www-form-urlencoded";
            request1.ContentLength = byteArray1.Length;
            using (System.IO.Stream dataStream = request1.GetRequestStream())
            {
                dataStream.Write(byteArray1, 0, byteArray1.Length);
            }
            WebResponse response1 = request1.GetResponse();
            string json_string1 = "";
            using (System.IO.Stream stream = response1.GetResponseStream())
            {
                using (StreamReader reader = new StreamReader(stream))
                {
                    json_string1 = reader.ReadToEnd();
                }
            }
            Dsql1 = JsonConvert.DeserializeObject<DataForLeftMenu>(json_string1);
            response1.Close();
            return Dsql1;
        }

        public override void OnBackPressed()
        {
            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            if (drawer.IsDrawerOpen(GravityCompat.Start))
            {
                drawer.CloseDrawer(GravityCompat.Start);
            }
            else
            {
                base.OnBackPressed();
            }
        }

        public override bool OnCreateOptionsMenu(IMenu menu)
        {
            MenuInflater.Inflate(Resource.Menu.menu_main, menu);
```

```
            return true;
        }

        public override bool OnOptionsItemSelected(IMenuItem item)
        {
            int id = item.ItemId;
            if (id == Resource.Id.action_settings)
            {
                return true;
            }

            return base.OnOptionsItemSelected(item);
        }

        private void FabOnClick(object sender, EventArgs eventArgs)
        {
            View view = (View)sender;
            Snackbar.Make(view, "Replace with your own action", Snackbar.LengthLong)
                .SetAction("Action", (Android.Views.View.IOnClickListener)null).Show();
        }

        public bool OnNavigationItemSelected(IMenuItem item)
        {
            int id = item.ItemId;

            if (id == Resource.Id.nav_profile)
            {
                StartActivity(new Android.Content.Intent(this, typeof(Profile)));
                Finish();
            }
            else if (id == Resource.Id.nav_message)
            {
                StartActivity(new Android.Content.Intent(this, typeof(MainActivity)));
                Finish();
            }
            else if (id == Resource.Id.nav_event)
            {
                StartActivity(new Android.Content.Intent(this, typeof(AllEvents)));
                Finish();
            }
            else if (id == Resource.Id.nav_infbase)
            {
                StartActivity(new Android.Content.Intent(this, typeof(Infomenu)));
                Finish();
            }
            else if (id == Resource.Id.nav_Logout)
            {
                Sessions.stopsession();
                StartActivity(new Android.Content.Intent(this, typeof(Login)));
                Finish();
            }

            DrawerLayout drawer = FindViewById<DrawerLayout>(Resource.Id.drawer_layout);
            drawer.CloseDrawer(GravityCompat.Start);
            return true;
        }
        public override void OnRequestPermissionsResult(int requestCode, string[] permissions, [GeneratedEnum]
Android.Content.PM.Permission[] grantResults)
        {
            Xamarin.Essentials.Platform.OnRequestPermissionsResult(requestCode, permissions, grantResults);

            base.OnRequestPermissionsResult(requestCode, permissions, grantResults);
```

```
        }

        public void OnClick(View v)
        {
            if (v.Id == Resource.Id.Zakdoc)
            {
                //StartActivity(new Android.Content.Intent(this, typeof(ForgotPass)));
                //Finish();
            }
            else if (v.Id == Resource.Id.histRSO)
            {

                //StartActivity(new Android.Content.Intent(this, typeof(SignUp)));
               // Finish();
            }
            else if (v.Id == Resource.Id.Histotr)
            {
                StartActivity(new Android.Content.Intent(this, typeof(histotr)));
                Finish();

            }
            else if (v.Id == Resource.Id.obrp)
            {


            }
        }
    }
}
```

Приложение Б

РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

(рекомендуемое)

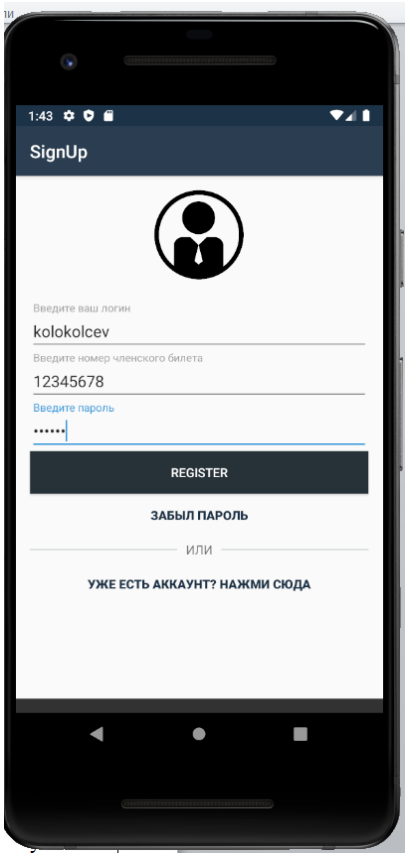

Рисунок Б.1 – Регистрация

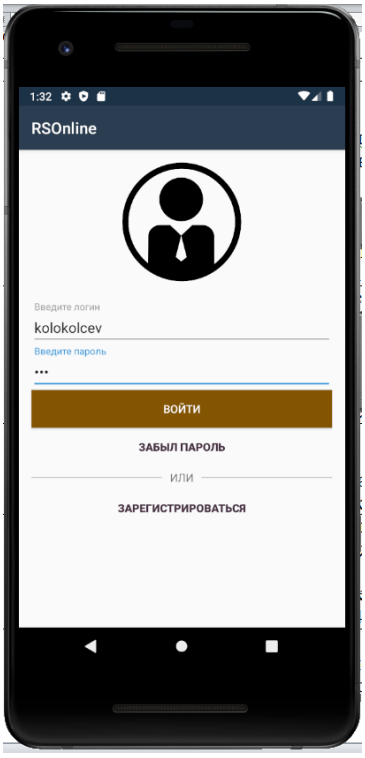

Рисунок Б.2 – Ввод данных для авторизации

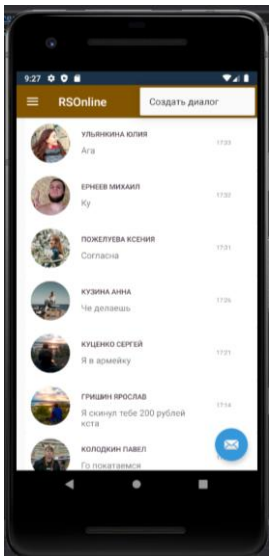

Рисунок Б.3 – Переход на страницу с сообщениями

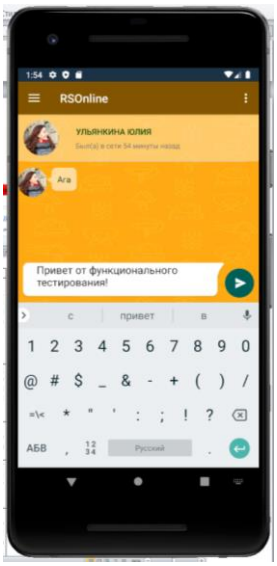

Рисунок Б.4 – Отправка сообщения

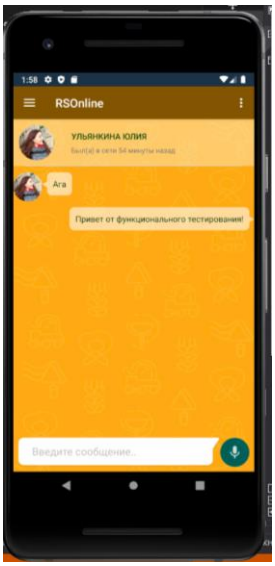

Рисунок Б.5 – Отображение сообщения в чате

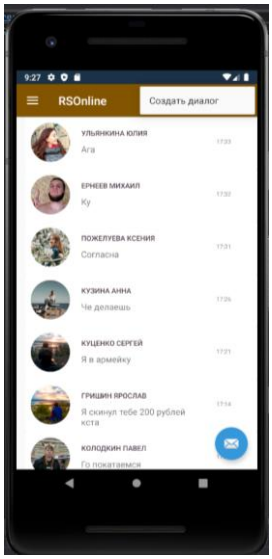

Рисунок Б.6 – Создание диалога

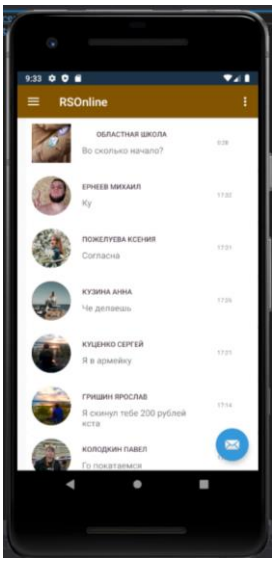

Рисунок Б.7 – Отображение нового диалога

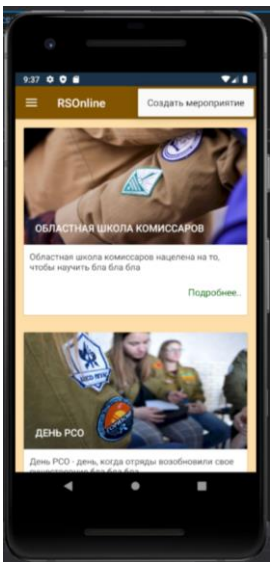

Рисунок Б.8 – Создание мероприятия

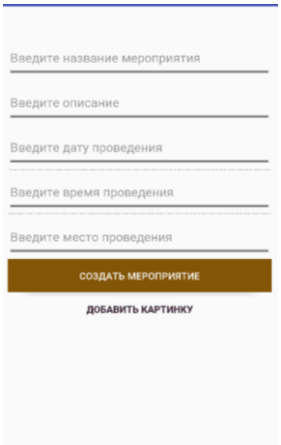

Рисунок Б.9 – Форма создания мероприятия

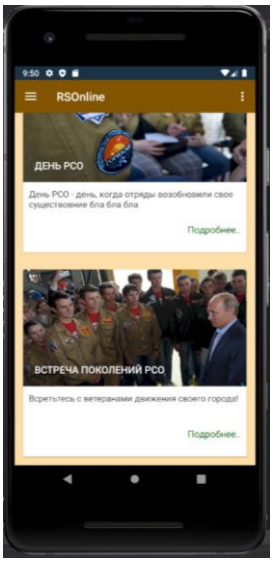

Рисунок Б.10 – Отображение нового мероприятия

Рисунок Б.11 – Заход в информационный раздел и просмотр страницы определенного отряда

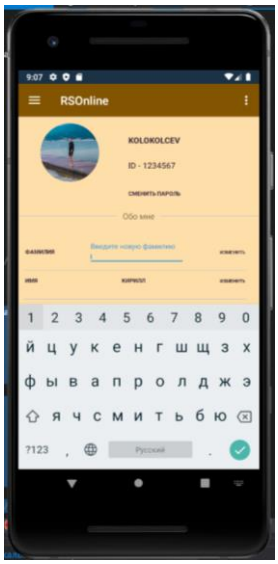

Рисунок Б.12 – Изменение фамилии

Рисунок Б.13 – Просмотр списка мероприятий

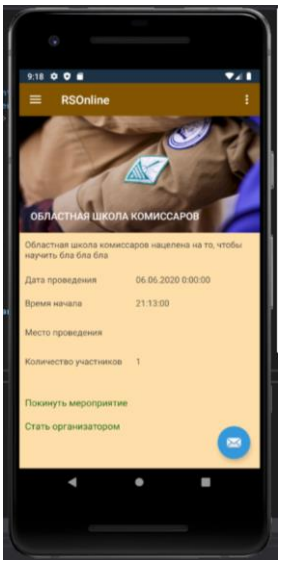

Рисунок Б.14 – Становление участником мероприятия