

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Южно-Уральский государственный университет
(национальный исследовательский университет)»

Высшая школа электроники и компьютерных наук
Кафедра «Электронные вычислительные машины»

РАБОТА ПРОВЕРЕНА
Рецензент

«__»_____ 2019 г.

ДОПУСТИТЬ К ЗАЩИТЕ
Заведующий кафедрой ЭВМ
Г.И. Радченко

«__»_____ 2019 г.

Разработка веб-приложения для организации научной конференции

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

Руководитель работы,
к.т.н., доцент каф. ЭВМ
Ю.Г. Плаксина

«__»_____ 2019 г.

Автор работы,
студент группы КЭ-452
А.А. Шакиров

«__»_____ 2019 г.

Нормоконтролёр,
ст. преп. каф. ЭВМ
С.В. Сяськов

«__»_____ 2019 г.

Челябинск-2019

АННОТАЦИЯ

A. A. Шакиров. Разработка веб-приложения для проведения научных конференций. – Челябинск: ФГАОУ ВО «ЮУрГУ (НИУ)», ВШЭКН; 2019, 64 с., 26 ил., библиогр. список – 16 наим., 0 прил.

В рамках выпускной квалификационной работы разработано веб-приложение для организации научных конференций. В данной работе был проведен анализ рынка аналогичных веб-приложений. Рассмотрены и изучены различные аналоги разрабатываемого приложения. В рассмотренных аналогах отсутствует возможность сопровождения участников конференции на момент ее проведения. На основе проведенного анализа были детализированы функциональные требования к программному продукту.

В процессе выполнения выпускной квалификационной работы были спроектированы архитектура приложения и минимальный пользовательский интерфейс.

Разработанное веб-приложение позволит упростить взаимодействие участников и организаторов конференции, оперативно предоставлять актуальную информацию пользователям и организовать обратную связь от участников к организаторам.

ОГЛАВЛЕНИЕ

АННОТАЦИЯ.....	5
ОГЛАВЛЕНИЕ	6
ВВЕДЕНИЕ.....	7
1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ	9
1.1 ОБЗОР АНАЛОГОВ	9
1.2. АНАЛИЗ ОСНОВНЫХ ТЕХНОЛОГИЧЕСКИХ РЕШЕНИЙ.....	11
1.2.1 Backend	11
1.2.2 Выбор системы управления базой данных.....	15
1.2.3 Frontend	17
1.2.4 Карты	17
1.3 ВЫВОД.....	20
2 ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ.....	21
2.1 ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ.....	21
2.2 НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ.....	22
3 ПРОЕКТИРОВАНИЕ.....	23
3.1 АРХИТЕКТУРА ПРОГРАММЫ	23
3.2 АРХИТЕКТУРА ПРЕДЛАГАЕМОГО РЕШЕНИЯ	25
4 РЕАЛИЗАЦИЯ.....	42
4.1 РЕАЛИЗАЦИЯ ИНТЕРФЕЙСОВ	42
5 ТЕСТИРОВАНИЕ.....	56
5.1 МЕТОДОЛОГИЯ ТЕСТИРОВАНИЯ	56
5.2 ПРОВЕДЕНИЕ ПРОЦЕДУРЫ ТЕСТИРОВАНИЯ	58
ЗАКЛЮЧЕНИЕ.....	61
БИБЛИОГРАФИЧЕСКИЙ СПИСОК.....	62

ВВЕДЕНИЕ

Проведение научных конференций является неотъемлемой частью научной деятельности университетов. Конференция - это одновременно семинары, тренинги, выставки, десятки симпозиумов и круглых столов. Она позволяет собрать большое количество ученых, занимающихся исследованиями в областях науки и техники. Направлена на представление результатов новейших научных достижений и исследований посредством докладов и организации конструктивных дискуссий, обмен знаниями и передовым опытом в продвижении научных разработок, развитие контактов специалистов на стыке наук, развитие междисциплинарного, межрегионального и международного сотрудничества. Содействие подготовке научно-педагогических кадров высшей квалификации и повышению научной квалификации профессорско-преподавательских кадров.

Количество участников конференции может достигать тысячи человек. В настоящее время при организации практически любой серьезной конференции регистрация заявок для участия происходит через сайт конференции.

На сайте, как минимум, размещается самая необходимая информация:

- краткое сообщение о целях и теме конференции, об актуальности и необходимости проведения мероприятия;
- контакты организаторов конференции, тематические направления, название секций;
- информация о том, как зарегистрироваться для участия, что и в какие сроки необходимо предоставить;
- язык конференции, а также информация о том, где будут опубликованы материалы конференции.

Указывают организационный комитет, известных ученых, давших согласие на участие в конференции, сумму взноса, счет, по которому его можно оплатить, возможность поддержки молодых ученых в качестве спонсора, мероприятия в рамках конференции. На сайте обычно доступна информация о возможных вариантах расселения участников конференции и визовой поддержке, если это необходимо; приведены схемы проезда с вокзалов, аэропортов, план части города.

При проведении конференции организаторы сталкиваются с рядом проблем:

- сложная схема места проведения конференции;
- оперативное уведомление участников об изменениях в расписании и каких-либо новостях.

Разработке веб-приложения при помощи которого, возможно, решение указанных проблем, посвящена данная работа. Разрабатываемое веб-приложение содержит всю необходимую информацию: расписание и место расположения секций, схема проезда к месту конференции, схема помещений конференций. Спроектирована система коммуникации между участниками и организаторами.

1 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

1.1 ОБЗОР АНАЛОГОВ

Для создания информативного, удобного для пользователя и конкурентоспособного приложения, необходимо рассмотреть решения, имеющиеся на рынке, проанализировать их достоинства и недостатки.

Timepad

Позволяет создавать лэндинг мероприятия с формой регистрации и приема платежей. Есть возможность создать схему зала, собирать базу участников и платно размещать таргетированную рекламу о мероприятии.

Event.4Science

Позволяет создавать лэндинг мероприятия. Есть поддержка локализации для английского языка. Возможно сделать отдельные анкеты для разных категорий участников. Заявки возможно принимать, отклонять и отправлять их на доработку.

Lomonosov-msu

Позволяет создавать лэндинг мероприятия. Есть поддержка массовых рассылок, переписок с участниками и поддержка расписания секций.

В таблице 1.1 представлена сравнительная характеристика аналогичных продуктов на рынке.

Таблица 1.1 – характеристика рассмотренных аналогов

Веб-приложение	Разработчик	Преимущества	Недостатки
Timepad [1]	ООО«ТаймПэдЛтд».	Возможность создать страницу события	Часть функционала платная
		Возможность провести регистрацию участников и сбор контактов	Отсутствуют уведомления об изменениях в расписаниях секций
		Возможность проведения почтовых рассылок	Отсутствуют личные сообщения
		Возможность рассылать новость об анонсе	Отсутствуют схемы проезда
Event.4science [2]	компания NAUMEN	Возможность обрабатывать заявки	Часть функционала платная
			Отсутствуют уведомления об изменениях в расписаниях секций
			Отсутствуют личные сообщения
			Отсутствует схема проезда
			Отсутствует возможность составлять расписание
Lomonosov-msu [3]	Научная сеть «Ломоносов»	Можно создать страницу мероприятия	Отсутствуют уведомления об изменениях в расписаниях секций
		Можно приложить необходимые файлы	Отсутствуют личные сообщения
		Есть возможность массовой рассылки	Отсутствует схема проезда Отсутствует возможность составлять расписание

1.2. АНАЛИЗ ОСНОВНЫХ ТЕХНОЛОГИЧЕСКИХ РЕШЕНИЙ

1.2.1 Backend

Выбор языка программирования

Для разработки веб-приложения можно использовать следующие языки: Python, Java, Ruby, C, C++, PHP. Чтобы отсечь нецелесообразные варианты, здесь и далее прежде всего были проанализированы недостатки перечисленных решений (см. таблицу 1.2).

Таблица 1.2 – недостатки языков программирования для веб-приложений

Язык программирования	Недостатки
C++ и C	Высокий порог входления
	Низкий уровень абстракции
	Медленная скорость разработки веб-приложений
Java	Высокий порог входления
	Медленная скорость разработки
	Низкий уровень абстракции.
Ruby	Высокий порог входления;
	Недостаток информационных ресурсов
	Медленно развивается
PHP	Отсутствие многопоточности
	Безопасность
	Противоречия в коде

Для создания веб-приложения выбран язык программирования Python [5], так как данный интерпретируемый язык программирования высокого уровня пригоден для решения задач веб-разработки. Главным преимуществом языка Python является практичность. Он предоставляет программисту инструменты для быстрого и эффективного решения поставленных задач. Практический характер Python обусловлен четырьмя важными характеристиками:

1. Читабельностью;

2. Простотой;
3. Эффективностью;
4. Безопасностью.

Так же Python имеет ряд следующих преимуществ:

1. Является свободным программным обеспечением, распространяемым под особой лицензией;
2. Несложен в освоении;
3. Имеет большое сообщество пользователей;
4. Имеет развитую поддержку БД;
5. Есть возможность использования в изолированной среде;
6. Возможно развертывание на любом веб-сервере;
7. Python является кроссплатформенным и работает почти на всех известных платформах. Существуют порты под Windows, все варианты UNIX и Mac OS.

Для упрощения разработки веб-приложений можно использовать CMS либо фреймворк.

CMS (Content Management System) [6] – программное обеспечение, предназначенное для создания, организации структуры, редактирования веб-сайта и управления им.

CMS-движки дают возможность добавлять и редактировать сайт, не изменяя внутренний механизм организации и вывода страниц. Условно CMS разделена на два хранилища информации: для баз данных с контентом страниц и для элементов визуализации, позволяющих показывать содержимое сайта посетителям (графические элементы, шаблоны и т.д.)

Недостатки использования CMS:

1. Низкая производительность. У всех CMS производительность ниже, чем у аналогичных решений, разработанных на базе фреймворков, из-за высокого уровня абстрактности;

2. Избыточность функциональности отдельных модулей. Обычно взгляды разработчиков и пользователей на необходимую функциональность модуля кардинально различаются. Поэтому в большинстве случаев используется лишь незначительная часть возможностей;

3. Необходимость изучать сложное устройство системы шаблонов. Дизайнеру придется разобраться с правилами создания шаблонов и их возможностями.

Фреймворк [7] - это своего рода надстройка над языком, набор библиотек, которые хорошо отлажены и прекрасно взаимодействующих между собой.

Преимущества использования фреймворков:

1. Высокая производительность кода. По скорости работы могут уступать только веб-приложениям, полностью написанным на Python в следствие различий в уровне абстракций;

2. Безопасность. Фреймворки пишутся одними программистами для других программистов и тщательно тестируются всем сообществом. Это позволяет вовремя найти недочеты кода и устраниить ошибки;

3. Гибкость. Фреймворки позволяют решать широкий круг задач. Имеется возможность использования готовых классов и библиотек, разработанных другими программистами.

Фреймворки являются отличным выбором для программистов, желающих разрабатывать сложные уникальные проекты с упором на быстродействие. CMS подойдут в случае разработки веб-приложения при необходимости быстрого создания стандартного веб-приложения.

Таким образом, в данной работе будет использоваться фреймворк.

Выбор фреймворка

Существует множество фреймворков для создания веб-приложений. Наиболее популярные фреймворки для разработки на Python: Django, Pyramid, Flask.

Недостатки фреймворка Pyramid:

1. Малое количество документации;
2. Слабая система кэширование;
3. Отсутствие генератора кода.

Недостатки фреймворка Flask:

1. Малое количество готовых модулей;
2. Низкий уровень абстракции готовых компонентов;
3. Малое количество документации.

Для разработки проекта выбран Python-фреймворк Django. Django – универсальный фреймворк, который может быть задействован для всех типов веб-приложений. Благодаря его структуре, набору компонент и поддержке кэширования, фреймворк подходит для разработки таких типов проектов, как порталы, форумы, интернет-магазины или RESTful-приложения, так же на нем написан Instagram.

Особенности фреймворка Django:

1. Для организации кода Django использует интуитивную-понятную MVC (Model-View-Controller) архитектуру;
2. Django - full-stack фреймворк и включающий в себя проверенные, а так же хорошо зарекомендованные возможности, такие как ActiveRecord для баз данных, поддержку REST API, систему многоуровневого кэширования и другие;
3. Django хорошо поддаются расширению. Можно настраивать или заменять практически любую часть основного кода;
4. Одно из главных преимуществ Django – хорошая производительность;
5. Наличие простого и понятного генератора кода.

1.2.2 Выбор системы управления базой данных

Данные можно хранить в файлах или базах данных (БД). Особенности организации данных в БД по сравнению с файловыми системами:

1. Базы данных позволяют использовать одни и те же данные в различных приложениях;
2. БД сводят к минимуму дублирование данных, прибегая к нему только ради ускорения доступа к данным или для восстановления БД при утрате целостности данных;
3. Одна из важных черт БД – независимость данных от устройства прикладных программ, использующих их, а также она дает возможность создавать программы в таком виде, что изменение особенностей хранения, логической структуры или значений данных не потребует изменения программ их обработки;
4. Возможность изменять физические особенности хранения данных без изменения их логической структуры.

Существует множество различных СУБД, которые можно использовать для веб-приложений, например MySQL, Microsoft SQL Server, PostgreSQL, MongoDB. Чтобы отсечь нецелесообразные варианты, прежде всего были проанализированы недостатки перечисленных решений (см. таблицу 1.3).

Таблица 1.3 – обзор СУБД

СУБД	Недостатки
Microsoft SQL Server	Является тяжеловесной
	Может занять все доступные ресурсы
	Есть проблемы с использованием службы интеграции для импорта файлов
MySQL	Неразборчивая документация
	Слабая надежность
	Скорость работы может падать во время проведения пакетных операций или выполнения запросов чтения

Продолжение таблицы 1.3

СУБД	Недостатки
MongoDB	<p>SQL не используется в качестве языка запросов</p> <p>программа установки может занять много времени</p> <p>MongoDB подходит для проектов с разнородными данными, которые тяжело поддаются классификации.</p> <p>Для внедрения потребуются высококлассные специалисты</p>

PostgreSQL [8] является самым профессиональным из выше рассмотренных СУБД. Она свободно распространяется и соответствует стандартам SQL. В PostgreSQL стараются полностью применять стандарты ANSI/ISO SQL своевременно с выходом новых версий.

От других СУБД PostgreSQL отличается поддержкой объектно-ориентированного и/или реляционного подхода к базам данных. Например, полная поддержка надежных транзакций. Благодаря использованию мощных технологических решений PostgreSQL имеет высокий уровень производительности. Параллельность достигнута не за счет блокировки операций чтения, а версионностью, что также обеспечивает соответствие ACID. PostgreSQL можно расширять своими хранимыми процедурами, Эти функции упрощают использование постоянно повторяемых операций.

Достоинства PostgreSQL:

1. Открытое ПО, соответствующее стандарту SQL. PostgreSQL - бесплатное ПО с открытым исходным кодом.
2. Большое сообщество – существует довольно большое сообщество, в котором можно найти ответы на интересующие вопросы.
3. Большое количество дополнений – несмотря на огромное количество встроенных функций, существует очень много дополнений, позволяющих разрабатывать данные для этой СУБД и управлять ими.
4. Расширения – существует возможность расширения функционала за счет использования своих процедур.

5. Объектность – PostgreSQL это не только реляционная СУБД, но также и объектно-ориентированная с поддержкой наследования.

PostgreSQL идеально подходит для проектов с ограниченным бюджетом, но требует привлечения квалифицированных специалистов.

Таким образом, при разработке веб-приложения будет использоваться СУБД PostgreSQL.

1.2.3 Frontend

Для разработки клиентской части используются следующие средства:

1. Язык разметки HTML5;
2. Таблицы каскадных стилей CSS3;
3. Язык программирования JavaScript с библиотекой jQuery 3.3.1;
4. Фреймворк Bootstrap 4.3.1;

jQuery [4] – это инструмент, упрощающий написание общих задач JavaScript. Кроме того, jQuery обладает кросбраузерной совместимостью, а это значит, что можно быть уверенным в том, что любой современный браузер корректно отобразит документ, при использовании данной библиотеки.

Главным достоинством Bootstrap является адаптивная сетка, благодаря которой можно создавать адаптивные шаблоны с предсказуемым поведением, на устройствах с различной разрешающей способностью экранов.

1.2.4 Карты

Одним из требований к разрабатываемому приложению является наличие карты, на которой можно посмотреть месторасположение конференции и схему проезда до него.

На рынке поставщиков картографических и справочных сервисов и услуг можно выделить трех основных: Яндекс.Карты; 2ГИС; Google Maps. Сравнение карт по конкурирующим критериям приведено в таблице 1.7.

Таблица 1.7 – Сравнительная характеристика карт

Критерий	Яндекс.Карты	2ГИС	Google Maps
Покрытие	Лучшее покрытие России, уступает Google в покрытии мира	Уступает конкурентам в покрытии как в России, так и в других странах	Лучшее покрытие всего мира
Детализация	Хорошая детализация России, достаточная в мире	Одна из лучших детализаций в городах присутствия	Хорошая детализация по всему миру. На карте России могут отсутствовать крупные города. В плане отображения невнятная детализация. Объекты хорошо видны только при достаточно сильном приближении.
Детализация на уровне здания	Нет	Небольшие склады, кафедры университетов	Крупные торговые центры
Вариант выбора отображения ландшафта	Карта, спутник, народная карта	Карта	Карта, спутник, велокарта, общественный транспорт
Отображение пробок в крупных городах	Да. Отображение доп. информации о дорожной обстановке	Не все города	Не все города. Интеграция с сервисом Waze

Продолжение таблицы 1.7

Критерий	Яндекс.Карты	2ГИС	Google Maps
Актуализация справочной информации	Нет информации	Обновления каждый месяц	Нет информации
Стоимость	Бесплатная, если количество запросов в сутки не превышает 25 тысяч.	Бесплатная при использовании не в коммерческих целях	Бесплатная, если количество запросов в месяц не превышает 28 тысяч.

Также из достоинств 2ГИС карт стоит отметить то, что у них хорошее покрытие в Челябинске.

Таким образом, в веб-приложении будут использоваться 2ГИС карты.

1.3 ВЫВОД

Исследование рынка приложений для организации научных конференций, показало, что ни одно из них не является полнофункциональным и удовлетворяющих требованию заказчика. Следовательно, создание приложения, свободного от выявленных недостатков, является актуальной задачей.

Для реализации веб-приложения выбраны следующие технологические решения:

1. backend: язык программирования Python 2.7, фреймворк Django 1.11;
2. СУБД: PostgreSQL 11.2;
3. frontend: язык разметки HTML5, таблицы каскадных стилей CSS3, язык программирование JavaScript с библиотекой jQuery 3.3.1, фреймворк Bootstrap4.3.1.

2 ОПРЕДЕЛЕНИЕ ТРЕБОВАНИЙ

2.1 ФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

Основные требования к функционалу системы.

Организатор должен иметь следующие возможности:

1. Создание конференции;
2. Ручная регистрация участников;
3. Ручная регистрация членов оргкомитета, модераторов;
4. Составление/изменение расписания секций;
5. Публикация новостей (с возможностью уточнения тегами секций конференции);
6. Возможность вести диалог с участником конференции.

Модератор должен иметь следующие возможности:

1. Просмотр программы секции, формируемой с учетом зарегистрированных участников;
2. Возможность изменять очередность докладов.

Участник должен иметь следующие возможности: регистрация по QR коду с указанием ID номера доклада, Фамилии, Имени и Отчества в отдельных полях.

Состав меню:

1. Программа конференции (формируемая предварительно, без учета зарегистрированных участников);
2. Лента новостей;
3. Карта проезда (с использованием 2ГИС-сервиса);
4. Схема помещений (опционально по необходимости для конкретной конференции);
5. Добавление дополнительных докладов по ID номеру;
6. Получать push-уведомления о новостях, а также о том, что участник докладывает следующим;

7. Возможность задать вопрос организаторам.

2.2 НЕФУНКЦИОНАЛЬНЫЕ ТРЕБОВАНИЯ

Требования к пользователям

Каждый пользователь, работающий в системе, обязан зарегистрироваться в системе и заполнить следующие обязательные поля в регистрационной форме:

1. Имя пользователя;
2. Номер доклада.

В системе должны быть выделены следующие типы пользователей:

1. Организатор;
2. Модератор;
3. Участник.

Требования к системе уведомлений

Приложение должно обладать следующим минимальным набором уведомлений:

1. Уведомление о входящем сообщении;
2. Уведомление об изменении расписания секции.

3 ПРОЕКТИРОВАНИЕ

3.1 АРХИТЕКТУРА ПРОГРАММЫ

Как правило машины и программы, входящие в информационную систему, не являются равноправными. Часть из них используют ресурсы, другие имеют возможность обращаться к этим ресурсам. Компьютер управляющий ресурсом, называют сервером (файловый сервер, сервер базы данных, вычислительный сервер). Клиент и сервер какого-либо ресурса могут находиться как в рамках одной вычислительной машины, так и на различных компьютерах, связанных сетью.

Главный принцип архитектуры "клиент-сервер" заключается в разделении функций приложения на три группы:

1. Ввод и отображение данных (взаимодействие с пользователем);
2. Прикладные функции, характерные для данной предметной области;
3. Функции использования ресурсов;

Исходя из этого в любом приложении выделяются следующие компоненты:

1. Компонент, отвечающий за пользовательский интерфейс;
2. Прикладной компонент, реализующий алгоритм решения конкретной задачи;
3. Компонент управляющий ресурсом;

Клиент-серверная[9] архитектура определяет общие правила организации взаимодействия в сети, где имеются серверы, узлы-поставщики некоторых специфичных функций (сервисов) и клиенты, использующие эти функции.

Для реализации данного веб-приложения будет удобно использовать трехзвенную клиент-серверную архитектуру, так как в проектируемой системе присутствует сервер БД.

Данная архитектура реализуется на основе модели сервера приложений, где сетевое приложение разделено на два и более компонента, каждый из

которых может находиться на отдельном компьютере. Выделенные части приложения взаимодействуют друг с другом, обмениваясь сообщениями в заранее согласованном формате (см. рисунок 3.1). Компоненты в модели располагаются следующим образом:

1. Представление данных – на стороне клиента;
2. Контроллер – на выделенном сервере приложений;
3. Управление ресурсами – на сервере БД, который и предоставляет запрашиваемые данные;

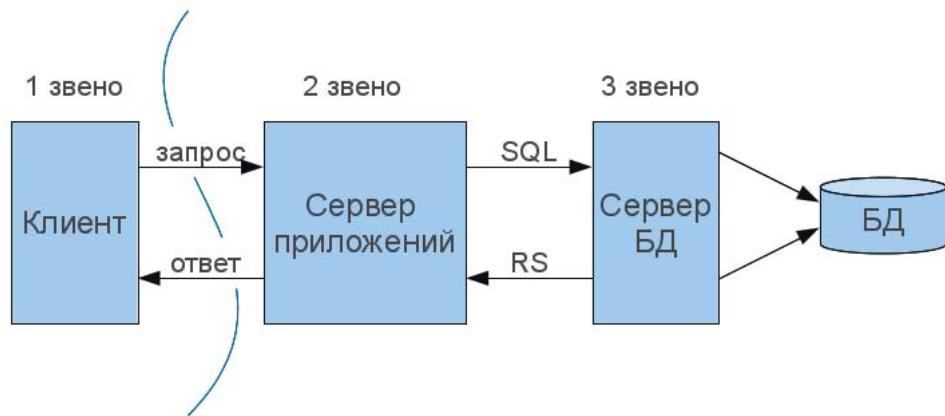


Рисунок 3.1 – Трехзвенная архитектура «клиент-сервер»

3.2 АРХИТЕКТУРА ПРЕДЛАГАЕМОГО РЕШЕНИЯ

Паттерн проектирования – архитектурное решение, предназначенное для проектирования некоторых типовых контекстов.

Подходящими шаблонами для веб-приложений являются MVC. В контексте разработки веб-приложения допустимо использовать классификацию шаблонов проектирования Мартина Фаулера [10]. Согласно ей паттерны можно распределить по категориям:

1. Базовые шаблоны;
2. Шаблоны веб-представления;
3. Шаблоны архитектурных источников данных;
4. Шаблон объектно-реляционной логики;
5. Шаблоны объектно-реляционного структурирования;
6. Шаблоны логики сущности;
7. Шаблоны распределения данных;
8. Шаблоны локальной конкуренции.

Любая из этих категорий включает в себя некоторый набор паттернов, одним из которых является паттерн MVC (Model – View – Control) и производные:

1. MVP (Model – View – Presenter);
2. MVVM (Model – View – View – Model);
3. HMVC (Hierarchical MVC);
4. PAC (Presentation – Abstraction – Control).

MVC [11] (см. рисунок 3.2) позволяет реализовать бизнес-логику приложения без необходимости затрачивать время на программирование. Он делит работу веб-приложения на три отдельные функциональные части: модель данных (model), представление (view) и управляющую логику

(controller). Таким образом, изменения, вносимые в один из компонентов, оказывают минимальное влияние на другие компоненты. В этом паттерне компоненты не зависят друг от друга, что позволяет проектировать модель как независимый компонент и, например, создавать несколько представлений для одной модели.

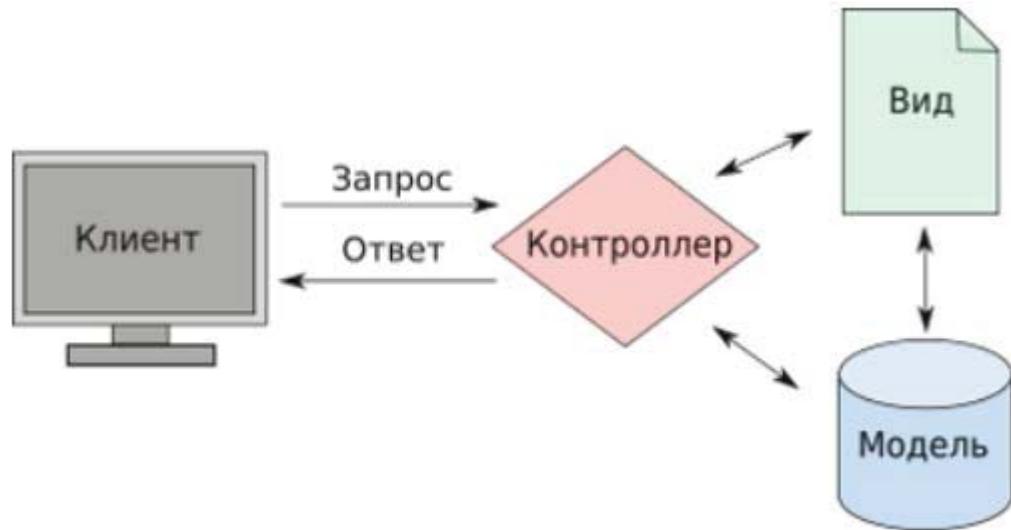


Рисунок 3.2 – Концепция паттерна MVC

Паттерн РАС [12] (Presentation–Abstraction–Control) – программный архитектурный паттерн. Это ориентированный на взаимодействие шаблон, который разделяет интерактивную систему на три типа компонентов, отвечающих за конкретные аспекты функциональности приложения. Компонент абстракции извлекает и обрабатывает данные, компонент представления форматирует визуальное и звуковое представление данных, а контроллер обрабатывает такие вещи, как поток управления и связь между двумя другими компонентами. В отличие от MVC, РАС используется в качестве иерархической структуры агентов, каждый из которых состоит из триады частей представления, абстракции и контроля (концепция паттерна РАС см. на рисунке 3.3). Агенты (или триады) связаны друг с другом только через контрольную часть каждой триады. Он также отличается от MVC тем, что в каждой триаде он полностью изолирует представление (представление в MVC) и абстракцию (модель в MVC). Это обеспечивает возможность

раздельной многопоточности модели и представления, что может дать пользователю возможность очень короткого времени запуска программы, поскольку пользовательский интерфейс (представление) может быть показан до полной инициализации абстракции.

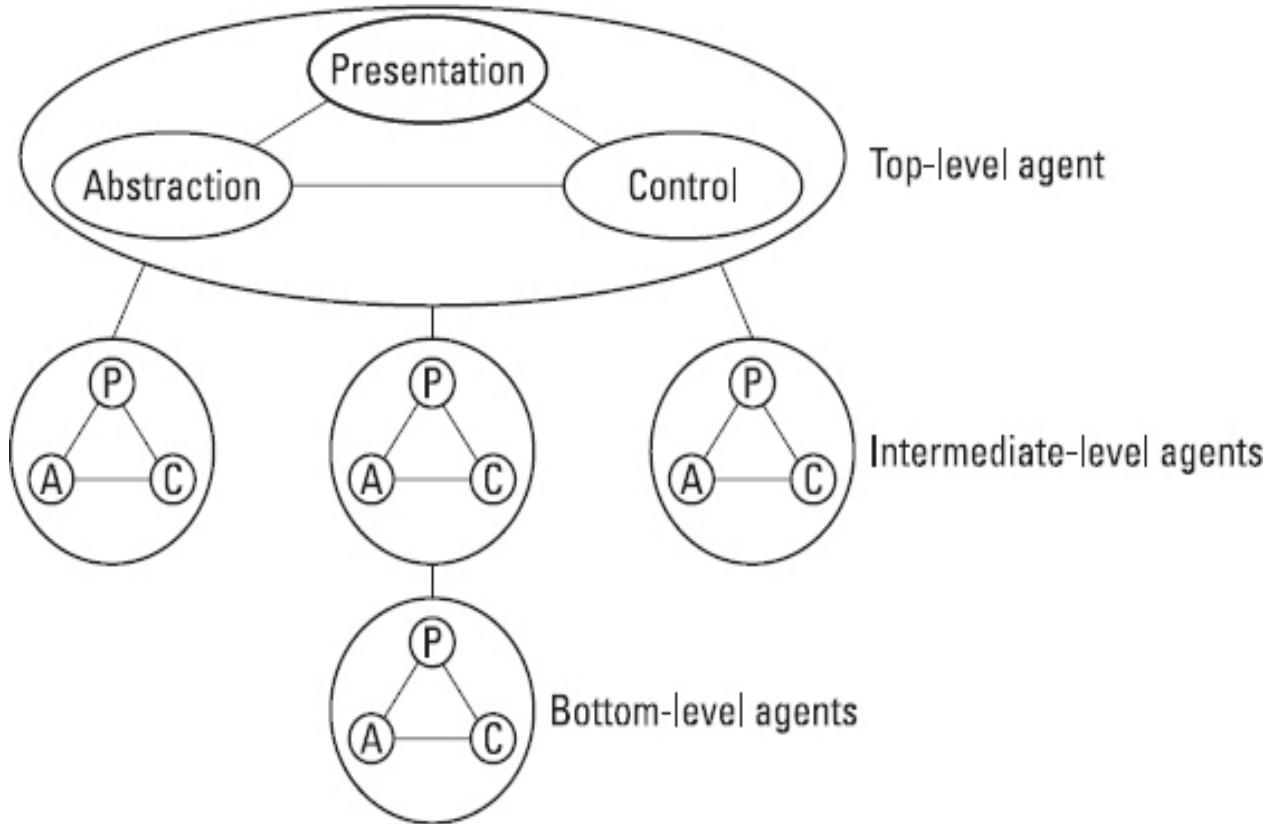


Рисунок 3.3 – Концепция паттерна РАС

Паттерн HMVC [13] (Hierarchical Model–View–Controller) – каждый отдельный набор MVC компонентов используется в качестве слоя в иерархической структуре. При этом каждый набор в этой иерархии независим от других, и может обратиться к контроллеру другого набора. Данный подход существенно упрощает и ускоряет разработку сложных приложений, упрощает их дальнейшую поддержку и масштабирование, способствует повторному использованию кода. Некоторые разработчики отмечают, что HMVC, по сути, является переосмыслением более строгого паттерна РАС. Концепция HMVC представлена на рисунке 3.4.

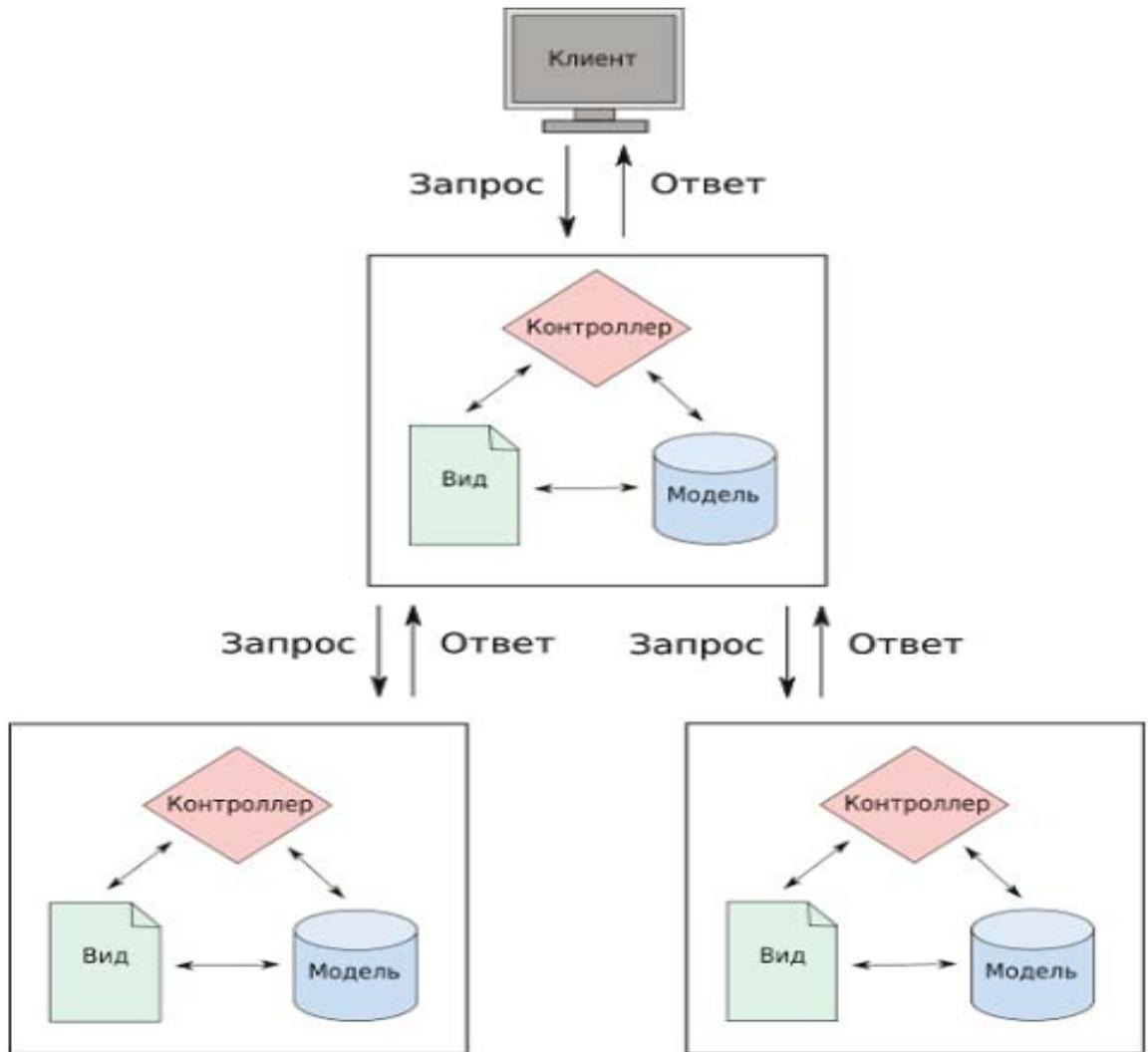


Рисунок 3.4– Концепция HMVC

Паттерн MVP [14] (Model-View-Presenter) – шаблон проектирования пользовательского интерфейса, разработанный для легкого автоматического тестирования и разделения ответственности в логике представления путем разделения логики и отображения. В MVP Presenter выполняет роль посредника, такую же, как контроллер в MVC. Также Presenter отвечает за управление событиями интерфейса пользователя, обработка которых в MVC отводится представлениям (View). Концепция данного паттерна представлена на рисунке 3.5.

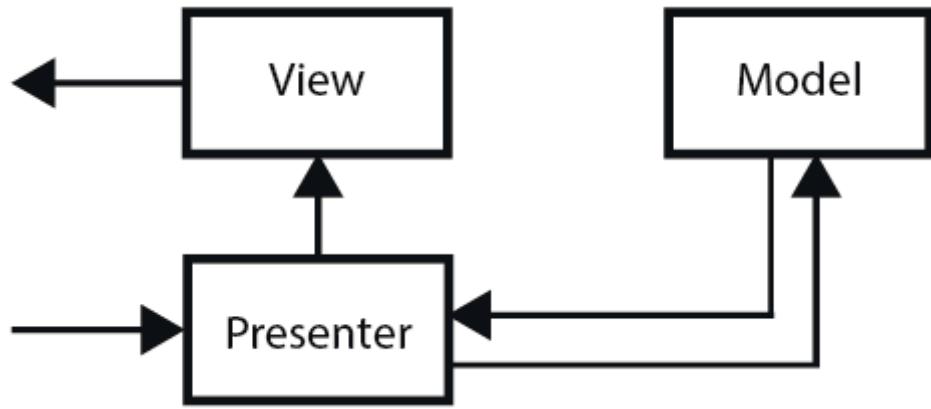


Рисунок 3.5 – концепция MVP

Паттерн MVVM [15] (Model-View-View-Model) – допускает отделение логики приложения от представления. Данный паттерн задает общую архитектуру приложения и имеет более тесную связь между моделью и представлением. Данная концепция реализована в WPF и Silverlight. На рисунке 3.6 приведена концепция MVVM.

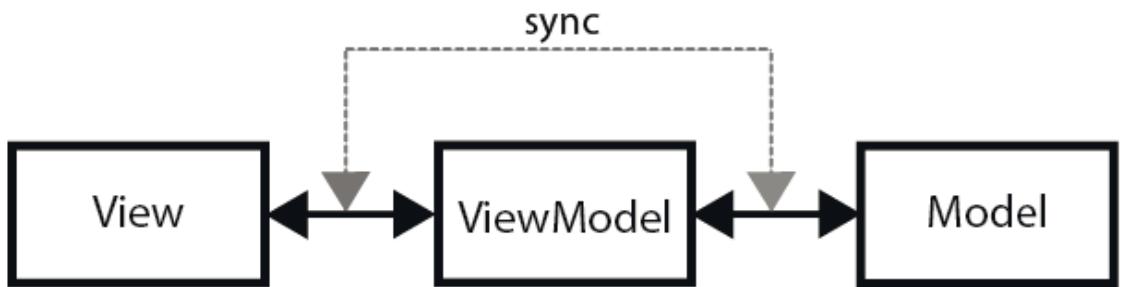
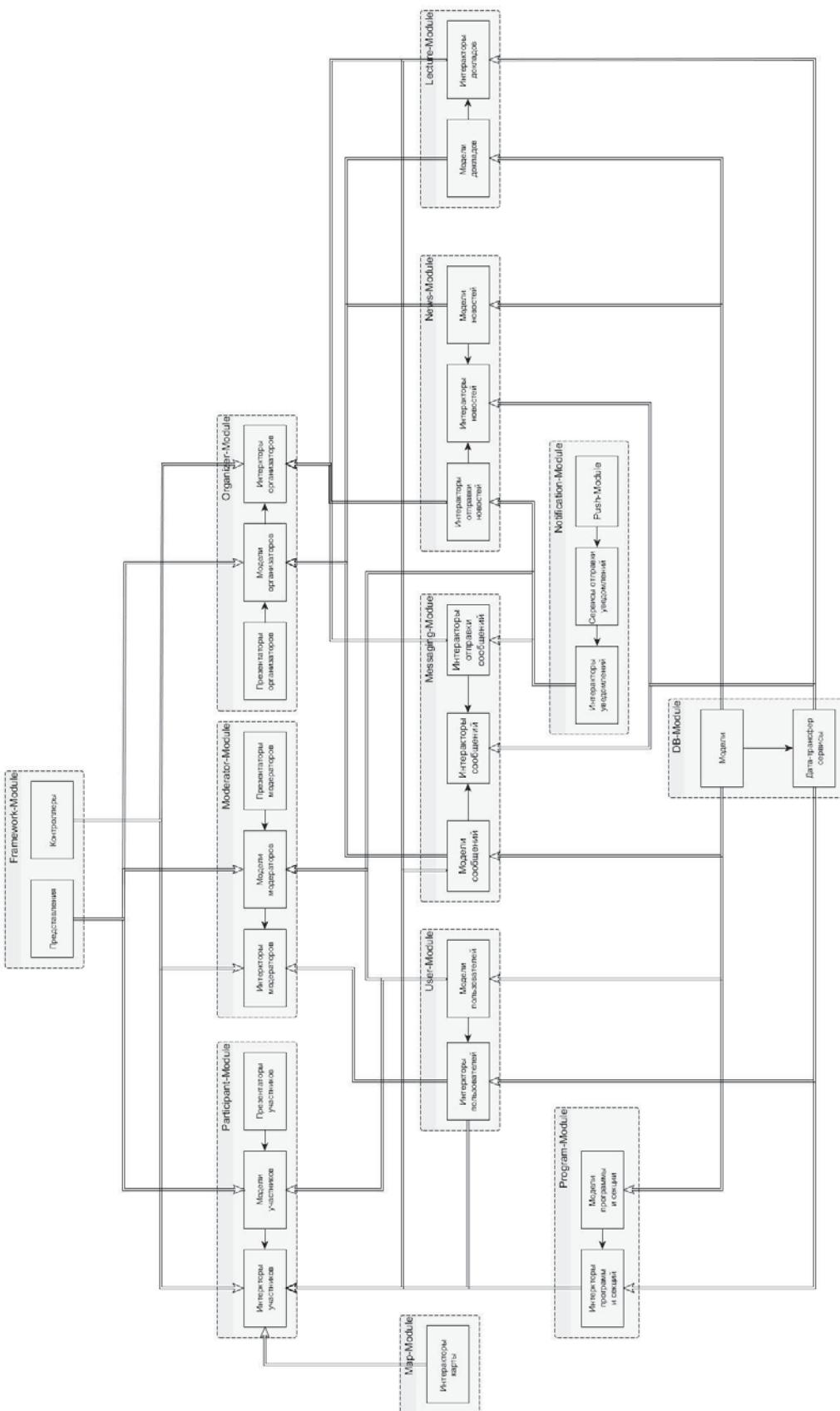


Рисунок 3.6 – Концепция MVVM

Рисунок 3.7 – Предварительная архитектура приложения



Для реализации приложения был выбран шаблон MVC, поскольку он обеспечивает простую реализацию бизнес-логики и является типичным

решением при разработке веб-приложений. Поэтому рассмотрим подробно компоненты MVC. В свою очередь фреймворк Django организации кода может использовать архитектуру паттерна MVC (Model-View-Controller). Идея, которая лежит в основе шаблона MVC, следующая: нужно чётко разделять ответственность за различное функционирование в приложении. Приложение разделяется на три основных компонента, каждый из которых отвечает за различные задачи. Рассмотрим этим компоненты подробнее:

1. Контроллер – управляет запросами пользователя (получаемыми в виде запросов HTTP GET или POST, когда пользователь использует элементы интерфейса для выполнения различных задач). Его основное предназначение – вызывать и координировать действие необходимых ресурсов и объектов, нужных для выполнения действий, вызванных пользователем. Обычно контроллер вызывает необходимую модель для задачи и выбирает подходящий вид.

2. Модель – это данные и правила, используемые для работы с данными, которые представляют систему управления приложением. В каждом приложении вся система моделируется как данные, которые обрабатываются определённым образом. Модель даёт контроллеру представление данных, которые запросил пользователь. Модель данных будет одинаковой вне зависимости от того, как мы хотим представлять их пользователю. Поэтому выбирается любой доступный вид для отображения данных.

Модель содержит наиболее важную часть логики приложения, которая лежит в основе решения задачи. Контроллер содержит, в основном, организационную логику для самого приложения.

3. Вид – обеспечивает разные средства для представления данных, которые получены из модели. Он может быть шаблоном, который заполняется данными. Может быть несколько разных видов, и контроллер выбирает, какой подходит самым лучшим образом для текущего случая.

Веб приложение обычно состоит из набора контроллеров, моделей и представлений. Контроллер может быть устроен как ведущий, который получает все запросы и вызывает другие контроллеры для реализации задач в зависимости от ситуации. Таблица с контроллерами веб-приложения приведена в таблице 3.1.

Таблица 3.1 – Контроллеры веб-приложения

Наименование	Назначение	Количество action, шт.	Взаимодействие с другими компонентами	
			View	Model
Conference	Работа с конференциями	1	• create;	• Conference
Event	Работа с событиями.	1	• create;	• Event
Lecture	Работа с докладами	1	• create;	• Lecture
Messaging	Работа с сообщениями	6	• send • load/old • load/new • load • read • available-users	• Message
Moderator	Работа с модераторами	1	• index	• User • Section
Organizer	Работа с организаторами	3	• index • create • user-registration	• User • Section • Post • Message • WebpushSettings

Продолжение таблицы 3.1

Наименование	Назначение	Количество action, шт.	Взаимодействие с другими компонентами	
			View	Model
Participant	Работа с участниками	1	<ul style="list-style-type: none"> index 	<ul style="list-style-type: none"> Section User Message WebpushSettings
Post	Работа с новостями	2	<ul style="list-style-type: none"> create tape 	<ul style="list-style-type: none"> Post WebpushSettings User Tag
Push Settings	Работа с настройками уведомлениями	1	<ul style="list-style-type: none"> update 	<ul style="list-style-type: none"> WebpushSettings
Section	Работа с секциями	2	<ul style="list-style-type: none"> create update 	<ul style="list-style-type: none"> Section Event Moderator
Tags	Работа с тегами новостей	2	<ul style="list-style-type: none"> most- popular find-tags 	<ul style="list-style-type: none"> Tag
User	Работа с пользователями	2	<ul style="list-style-type: none"> registration login 	<ul style="list-style-type: none"> User

Таблица 3.2 – Модели веб-приложения

Наименование	Назначение
Tag	Теги новостей
WebpushSettings	Настройки уведомлений
User	Пользователь
Post	Новость
Lecture	Доклад
Conference	Конференция
Section	Секция
Event	Логическая единица секции
Message	Сообщение

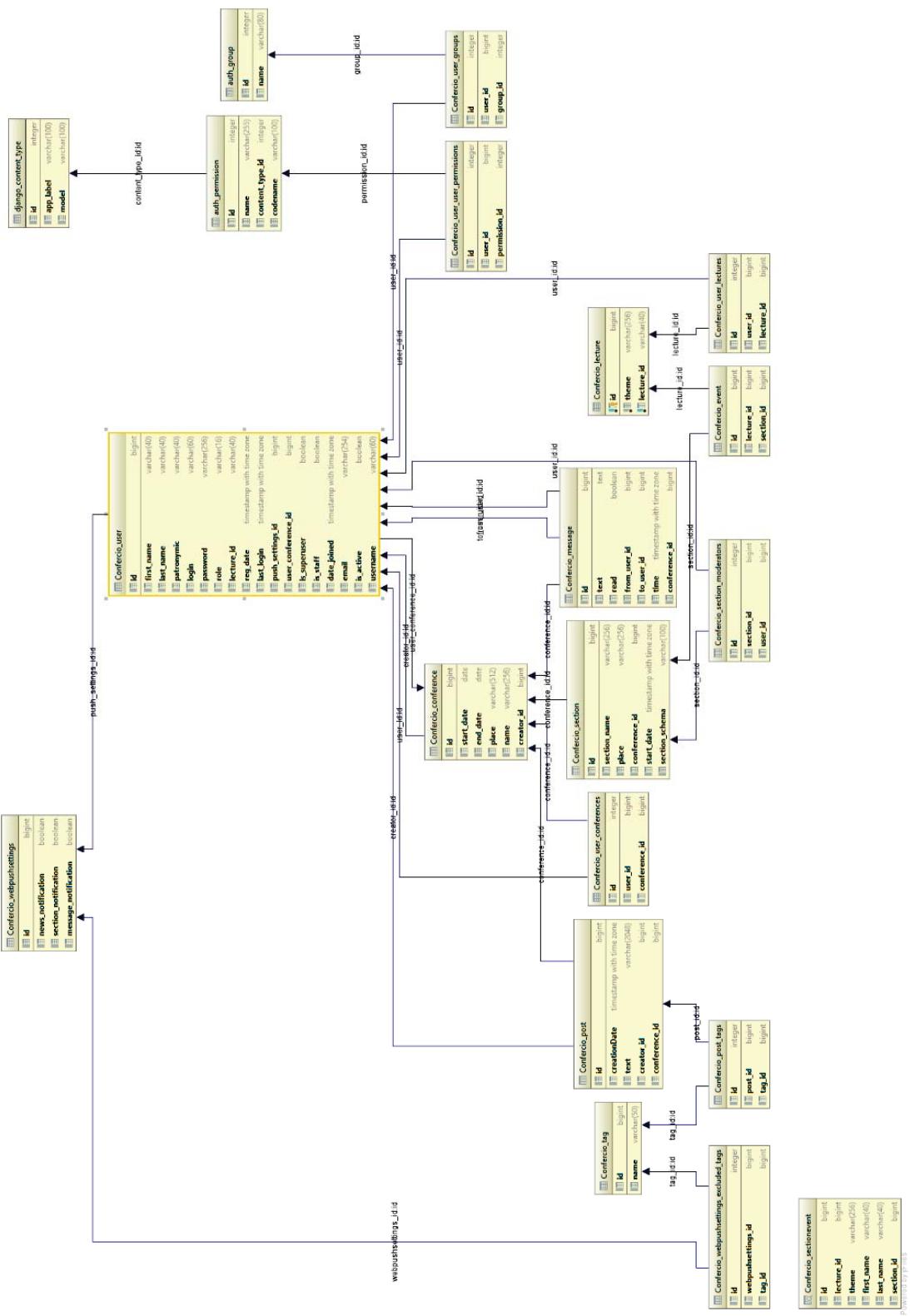
Таблица 3.3 – Представления веб-приложения:

Привязка к контроллеру	Наименование	Назначение
Conference	conference_create	Отображение формы создания конференции.
Event	event_create	Отображение формы создания события
Lecture	lecture_create	Отображение формы создания доклада

Таблица 3.3 – Представления веб-приложения:

Привязка к контроллеру	Наименование	Назначение
Message	choose-conversationalist	Модальное окно выбора собеседника
	dialog_container	Контейнер диалога
	dialog_line	Шаблон диалога из списка
	dialog_list	Список диалогов
	dialog_view	Виджет диалогов
Moderator	main	Страница модератора
	moderator_create	Форма создания модератора
	tableRowBreakTemplate	Шаблон строки перерыва
	tableRowEventTemplate	Шаблон строки события
Organizer	main	Страница организатора
	organizer_create	Форма создания организатора
Participant	main	Страница пользователя
Post	news_tape	Новостная лента
Section	section_create	Форма создания секции

Рисунок 3.8 – Схема базы данных с использованием UML-диаграмм



Ниже представлены таблицы БД. Таблица со списком таблиц БД и их назначением приведена в таблице 3.3.

Таблица 3.3 – Таблицы базы данных

Наименование	Назначение
Confercio_conference	Конференции
Confercio_event	Информация о позиции доклада в секции
Confercio_lecture	Доклады
Confercio_message	Сообщения
Confercio_post	Новости
Confercio_post_tags	Теги новостей
Confercio_section	Секции
Confercio_section_moderators	Модераторы секций
Confercio_tag	Теги
Confercio_user	Пользователи
Confercio_user_lectures	Связь между участниками и их докладами
Confercio_webpushsettings	Настройки уведомлений
Confercio_webpushsettings_excluded_tags	Исключенные теги

Таблица с описанием полей таблицы «Confercio_conference» приведена в таблице 3.4.

Таблица 3.4 – Описание полей таблицы «Confercio_conference»

Наименование	Тип данных	Назначение
id	bigint	Идентификатор конференции
start_date	date	Дата начала
end_date	date	Дата окончания
place	varchar(512)	Место проведения
name	varchar(256)	Наименование конференции
creator_id	bigint	Ссылка на создателя

Таблица с описанием полей таблицы «Conference_event» приведена в таблице 3.5.

Таблица 3.5 – Описание полей таблицы «Conference_event»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор события
lecture_id	bigint	Ссылка на доклад
section_id	bigint	Ссылка на секцию

Таблица с описанием полей таблицы «Confercio_lecture» приведена в таблице 3.6.

Таблица 3.6 – Описание полей таблицы «Confercio_lecture»

Наименование	Тип данных	Назначение
id	bigint	Идентификатор доклада
theme	varchar(256)	Тема доклада
lecture_id	varchar(40)	Номер доклада

Таблица с описанием полей таблицы «Confercio_message» приведена в таблице 3.7.

Таблица 3.7 – Описание полей таблицы «Confercio_message»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор сообщения
text	text	Текст
read	boolean	Признак того что сообщение прочитано
from_user_id	bigint	Ссылка на отправителя
to_user_id	bigint	Ссылка на получателя
time	timestamp	Время отправления

Таблица с описанием полей таблицы «Confercio_post» приведена в таблице 3.8.

Таблица 3.8 – Описание полей таблицы «Confercio_post»

Наименование	Тип данных	Назначение
id	bigint	Идентификатор новости
creationDate	timestamp	Время создания
text	varchar(2048)	Текст
creator_id	bigint	Ссылка на создателя

Таблица с описанием полей таблицы «Confercio_post_tags» приведена в таблице 3.9.

Таблица 3.9 – Описание полей таблицы «Confercio_post_tags»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор связи
post_id	bigint	Ссылка на новость
tag_id	bigint	Ссылка на тег

Таблица с описанием полей таблицы «Conference_section» приведена в таблице 3.10.

Таблица 3.10 – Описание полей таблицы «Conference_section»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор секции
section_name	varchar(256)	Название секции
place	varchar(256)	Место секции
conference_id	bigint	Ссылка на конференцию
start_date	timestamp with time zone	Время начала секции
section_schema	varchar(100)	Путь к файлу со схемой месторасположения секции

Таблица с описанием полей таблицы «Conference_section_moderators» приведена в таблице 3.11.

Таблица 3.11 – Описание полей таблицы «Conference_section_moderators»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор связи
section_id	bigint	Ссылка на секцию
moderator_id	bigint	Ссылка на модератора.

Таблица с описанием полей таблицы «Conference_tag» приведена в таблице 3.12.

Таблица 3.12 – Описание полей таблицы «Conference_tag»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор тега
name	varchar(50)	Название тега

Таблица с описанием полей таблицы «Conference_user» приведена в таблице 3.13.

Таблица 3.13 – Описание полей таблицы «Conference_user»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор пользователя
first_name	varchar(40)	Имя
last_name	varchar(40)	Фамилия
patronymic	varchar(40)	Отчество
login	varchar(60)	Логин
password	varchar(60)	Пароль
role	varchar(16)	Роль
lecture_id	varchar(40)	Номер доклада, по которому зарегистрировался участник
reg_date	timestamp	Дата регистрации
push_settings_id	bigint	Ссылка на настройки уведомлений
user_conference_id	bigint	Ссылка на конференцию на которую зарегистрировался участник
is_superuser	boolean	Флаг админа

Таблица с описанием полей таблицы «Confercio_webpushsettings» приведена в таблице 3.14.

Таблица 3.14 – Описание полей таблицы «Confercio_webpushsettings»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор настроек уведомлений
messages_notifications	boolean	Уведомления сообщений
news_notifications	boolean	Уведомления новостей
section_notification	boolean	Уведомления секций

Таблица с описанием полей таблицы «Confercio_webpushsettings_excluded_tags» приведена в таблице 3.15.

Таблица 3.15 – Описание полей таблицы
 «Confercio_webpushsettings_excluded_tags»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор связи.
webpushsettings_id	bigint	Ссылка на настройки
tag_id	bigint	Ссылка на исключенный тег

Таблица с описанием полей представления «Confercio_sectionevent» приведена в таблице 3.16.

Таблица 3.16 – Описание полей представления
 «Confercio_sectionevent»:

Наименование	Тип данных	Назначение
id	bigint	Идентификатор события.
lecture_id	bigint	Ссылка на лекцию
theme	varchar(256)	Тема лекции
first_name	varchar(40)	Имя участника
last_name	varchar(40)	Фамилия участника
section_id	bigint	Ссылка на секцию

4 РЕАЛИЗАЦИЯ

4.1 РЕАЛИЗАЦИЯ ИНТЕРФЕЙСОВ

Был разработан пользовательский интерфейс, который удовлетворяет минимальным требованиям:

- соответствие задачам пользователя;
- соответствие технологии;
- понятность и логичность;
- обеспечение высокой скорости работы пользователя;
- обеспечение защиты от человеческих ошибок;
- быстрое обучение пользователя.

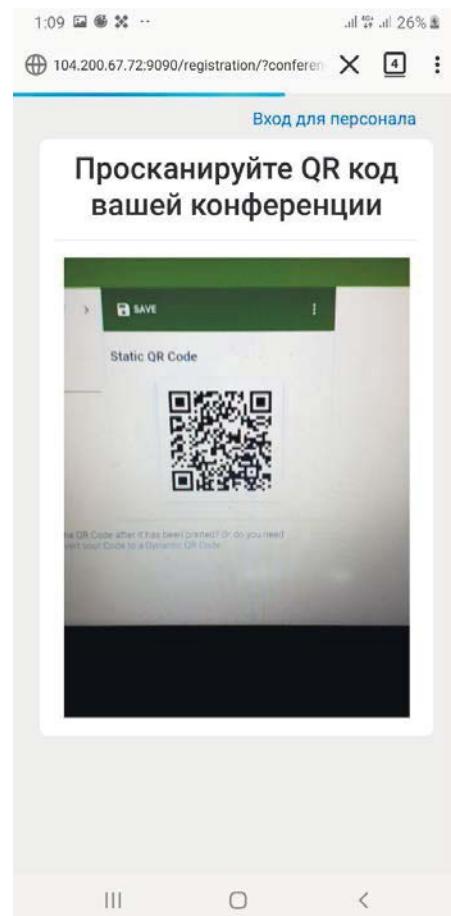


Рисунок 3.9 - Главная страница приложения для пользователя

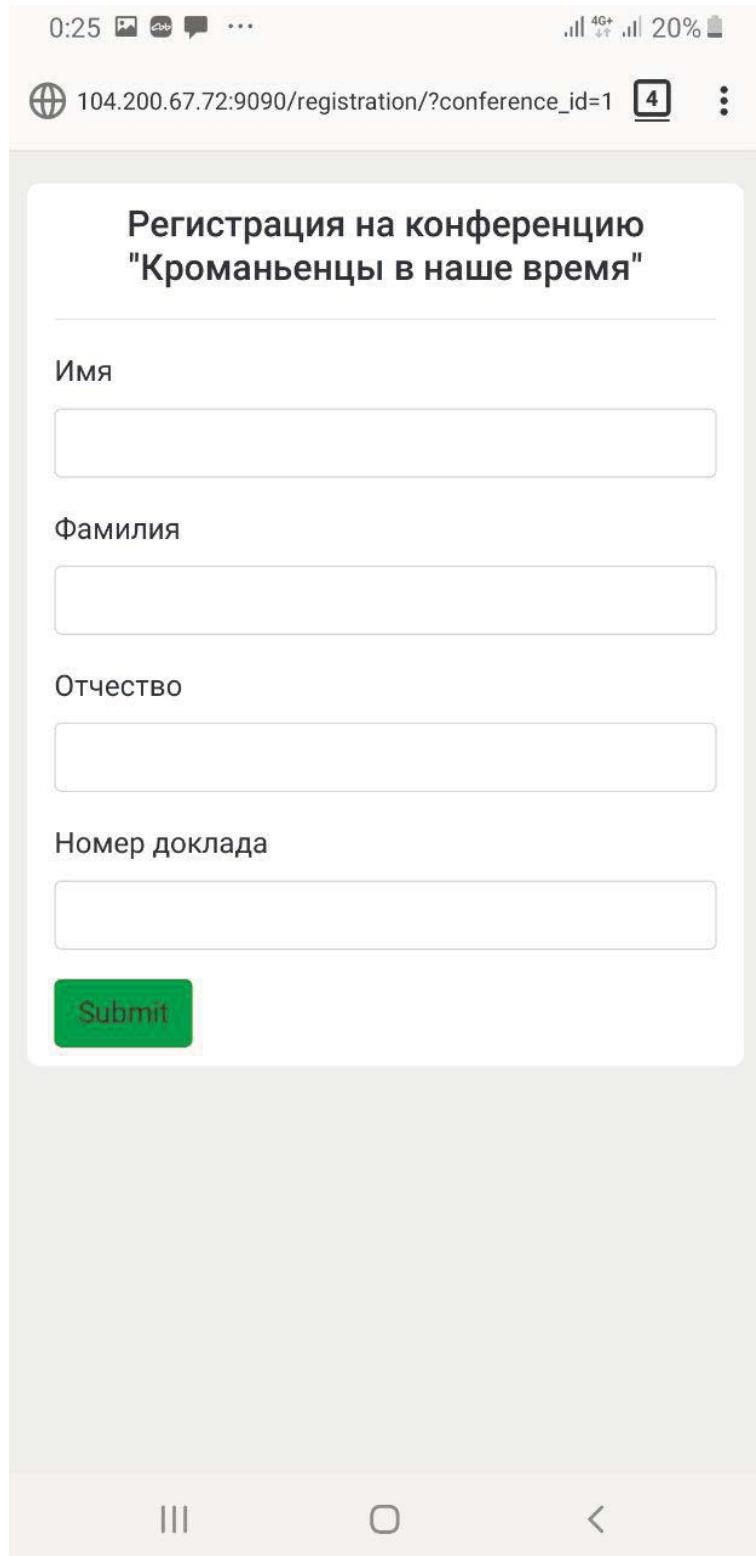


Рисунок 3.10 - Форма регистрации участника

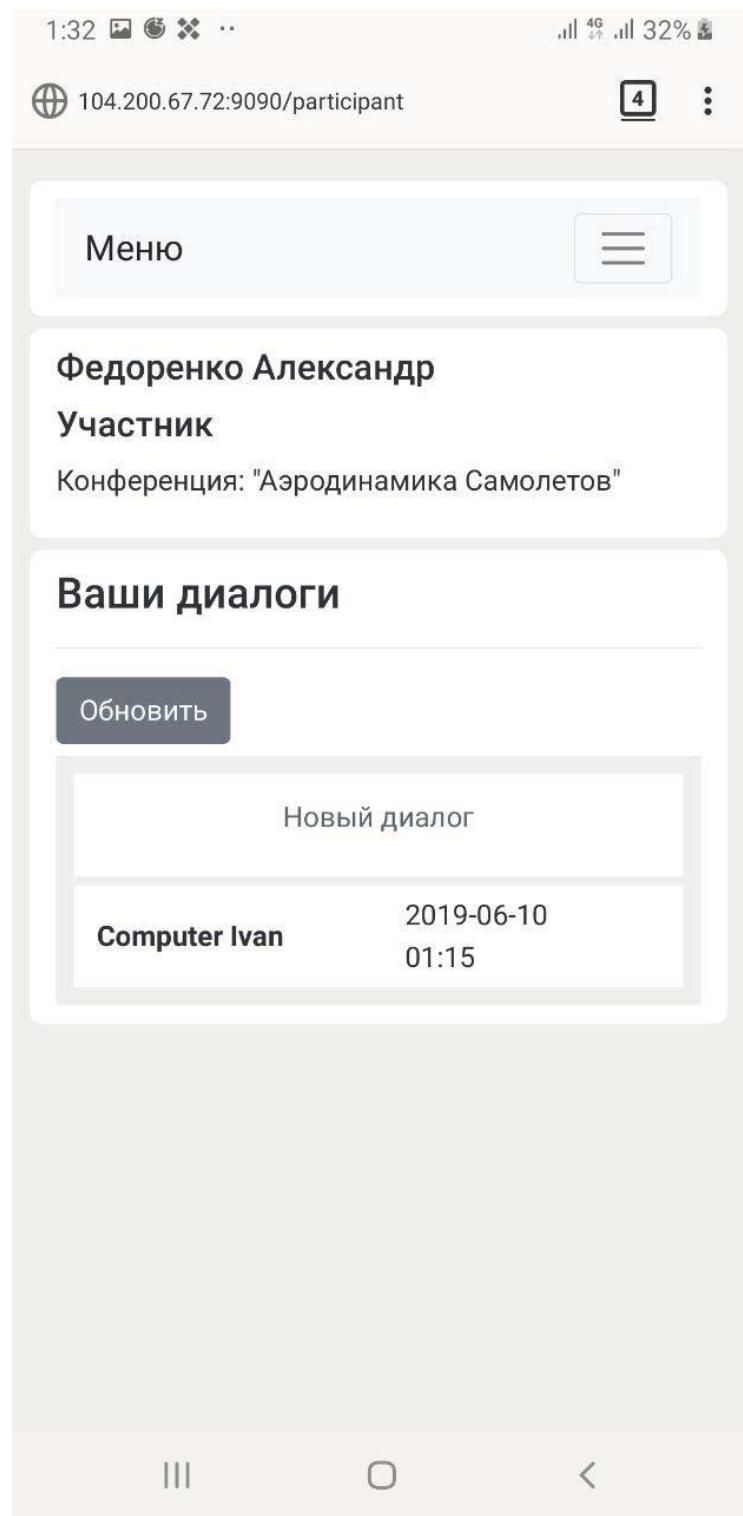


Рисунок 3.11 - Сообщения участника

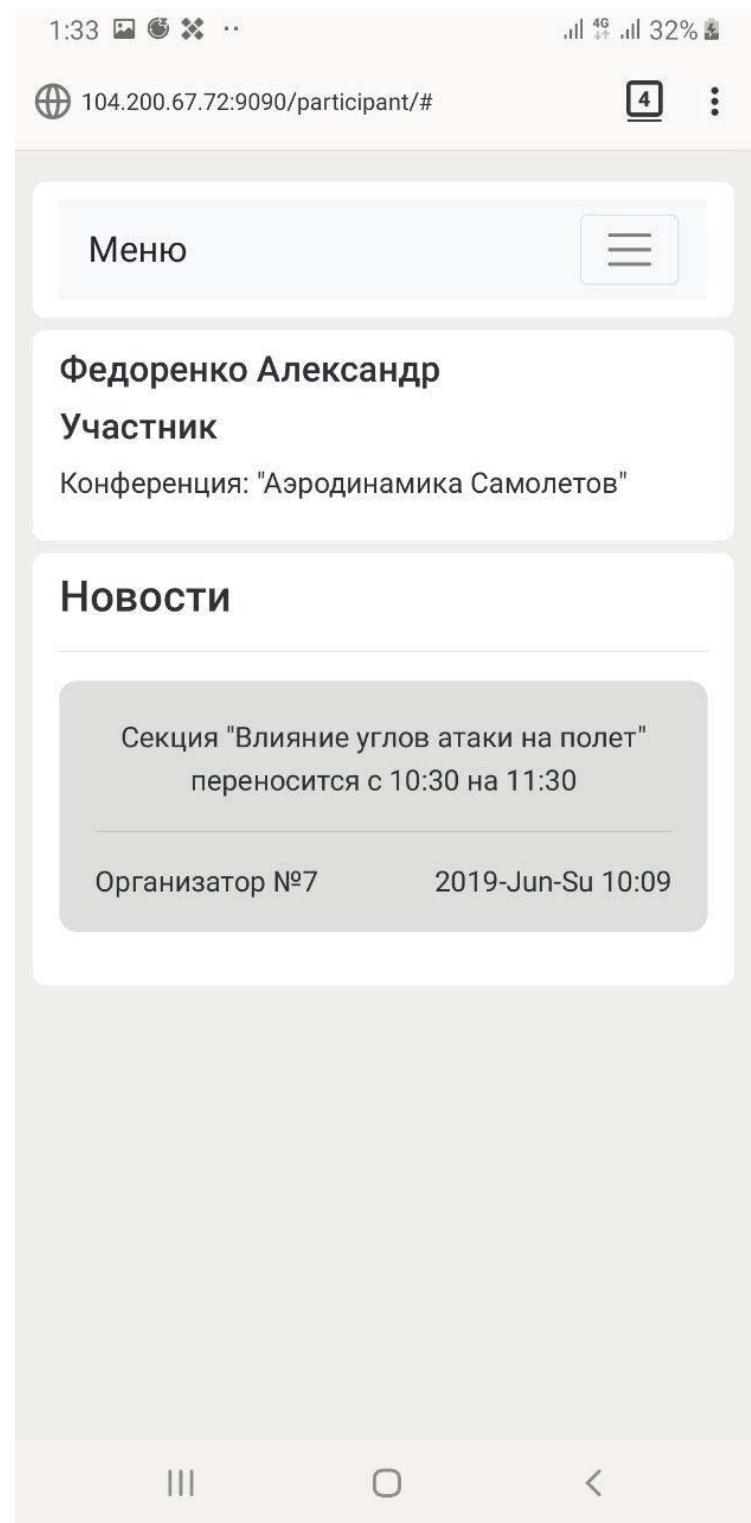


Рисунок 3.12 - Новости участника

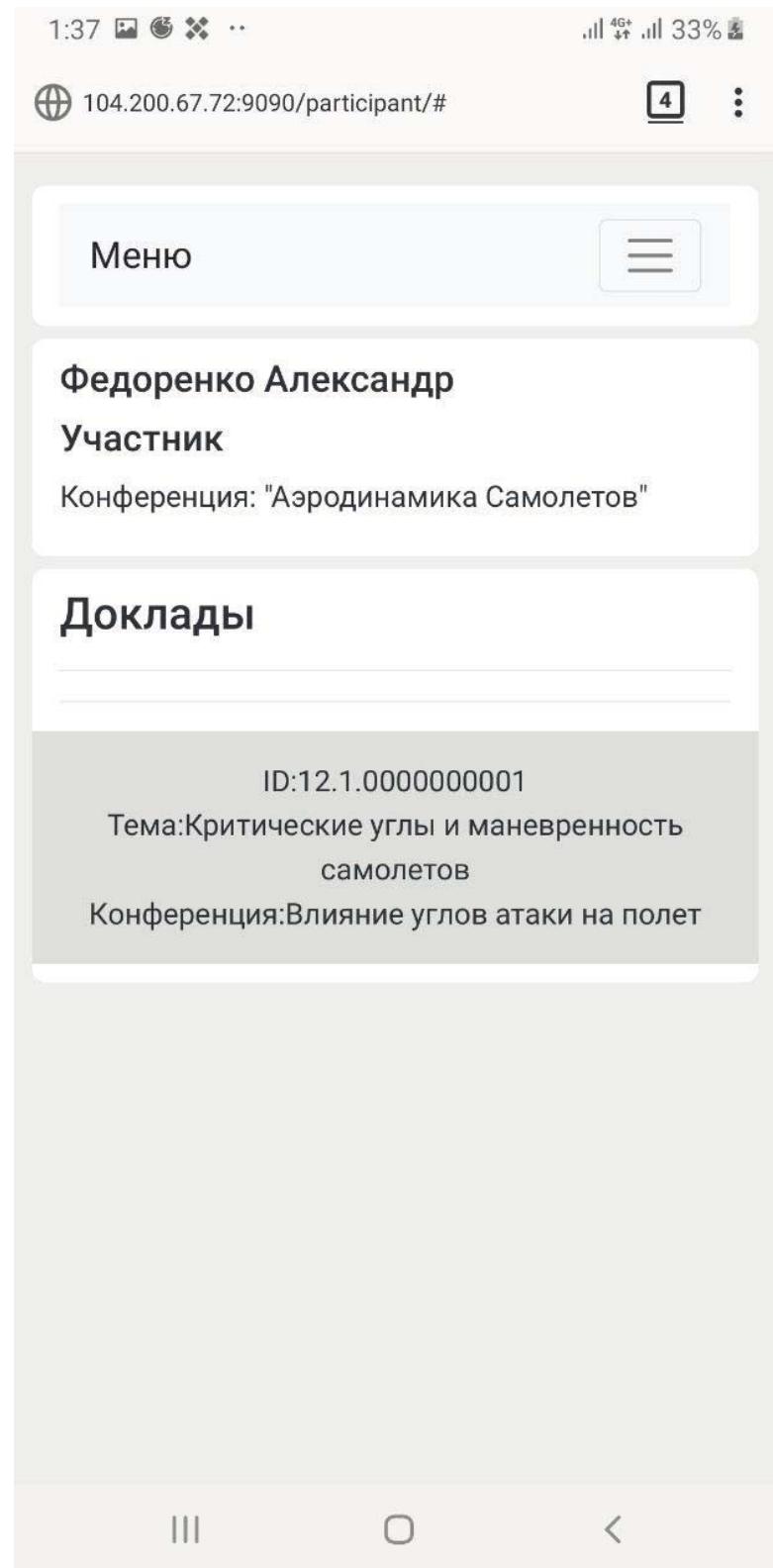


Рисунок 3.13 - Доклады участника

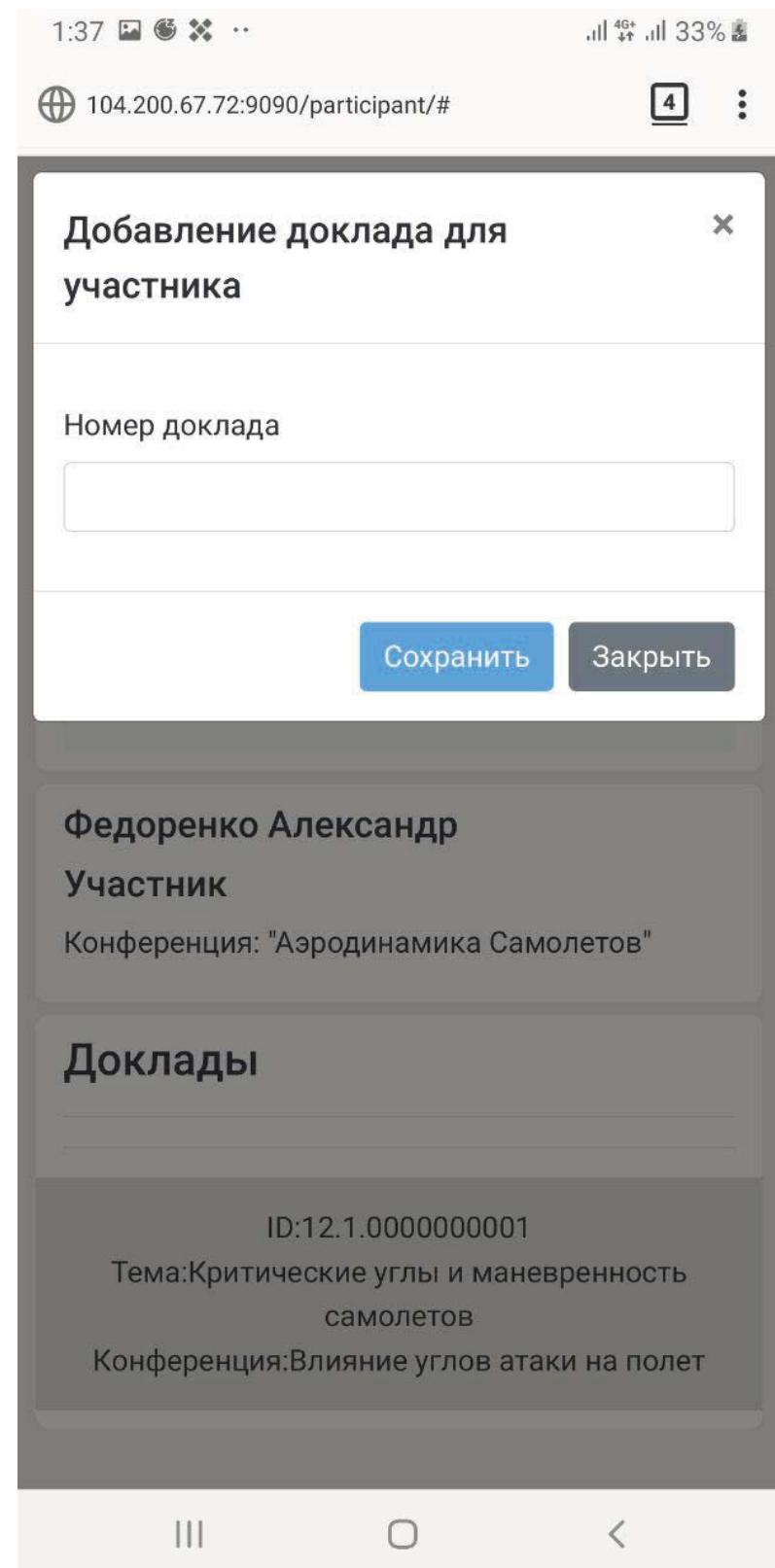


Рисунок 3.14 - Добавление доклада участника

14:07 80%

⊕ 104.200.67.72:9090/participant/#

4 :

Уведомления ▾

**Федоренко Александр
Участник**
Конференция: "Аэродинамика Самолетов"

Расписание секций

Влияние углов атаки на полет Закрыть

2019-06-09 11:30

Критические углы и маневренность самолетов	Федоренко Александр
Штопор как способ передвежения	Райт Уилбур
Целесообразно ли сбрасывать скорость шатлов закритическими углами	Маск Илон

||| ○ <

The screenshot shows a mobile application interface for a conference. At the top, there's a header with the time (14:07), signal strength, battery level (80%), and a URL (104.200.67.72:9090/participant/#). Below the header is a notification bar with a '4' icon and a three-dot menu. The main content area has a light gray background. It starts with a participant profile for 'Федоренко Александр' (Participant). Below that is a section titled 'Расписание секций' (Schedule). A specific session is highlighted with a blue rounded rectangle: 'Влияние углов атаки на полет' (Effect of attack angles on flight) scheduled for '2019-06-09 11:30'. This session is attributed to 'Федоренко Александр' (Fedorenko Alexander) and has a 'Закрыть' (Close) button. Below this are two other sessions listed in a table: 'Штопор как способ передвежения' (Stall as a way of forward movement) by 'Райт Уилбур' (Wright Wilbur) and 'Целесообразно ли сбрасывать скорость шатлов закритическими углами' (Is it reasonable to release shuttle speed at critical angles) by 'Маск Илон' (Elon Musk). At the bottom of the screen are three navigation icons: vertical lines, a circle, and a left arrow.

Рисунок 3.15 - Расписание секций

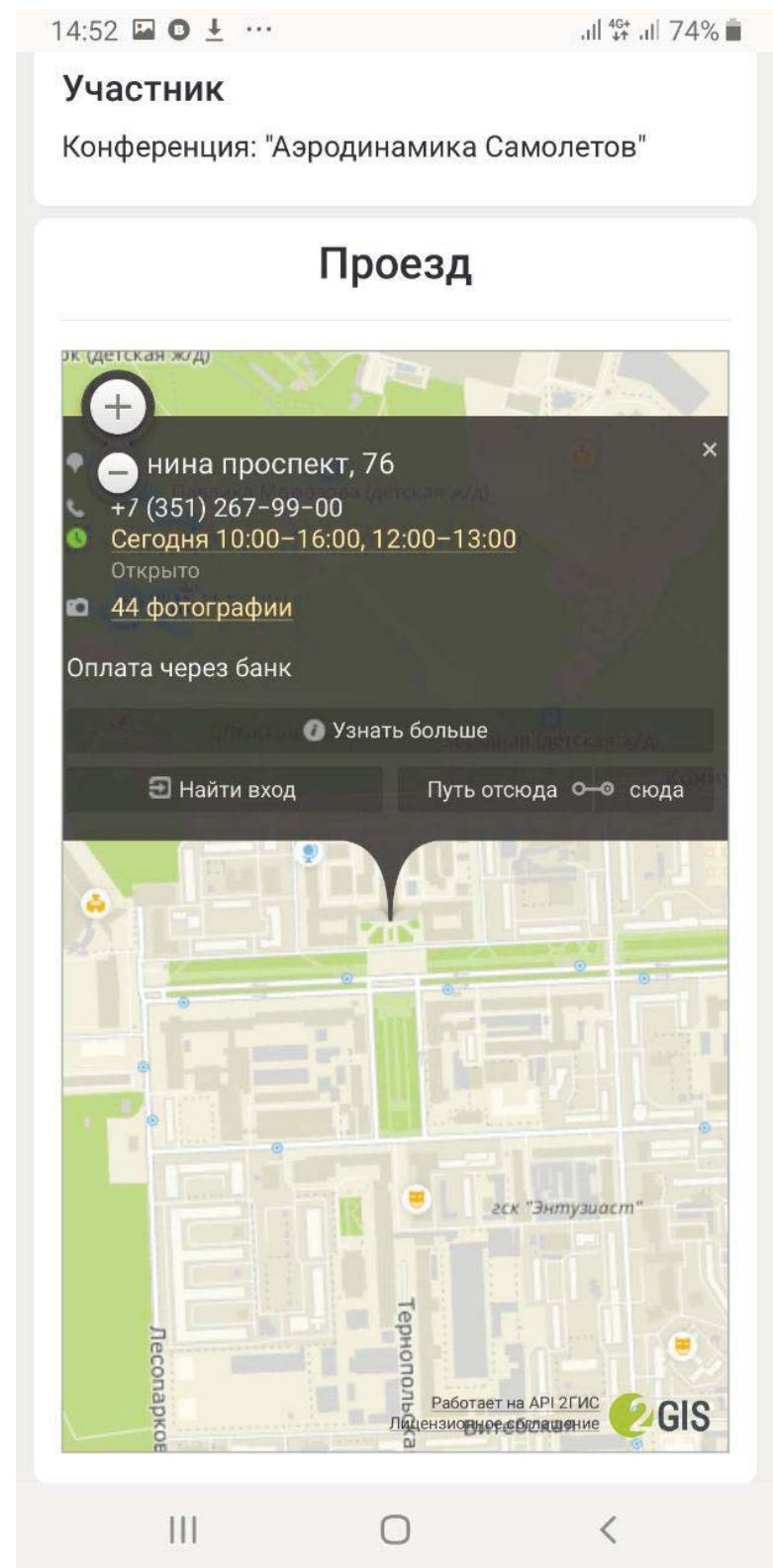


Рисунок 3.16 – Карта проезда

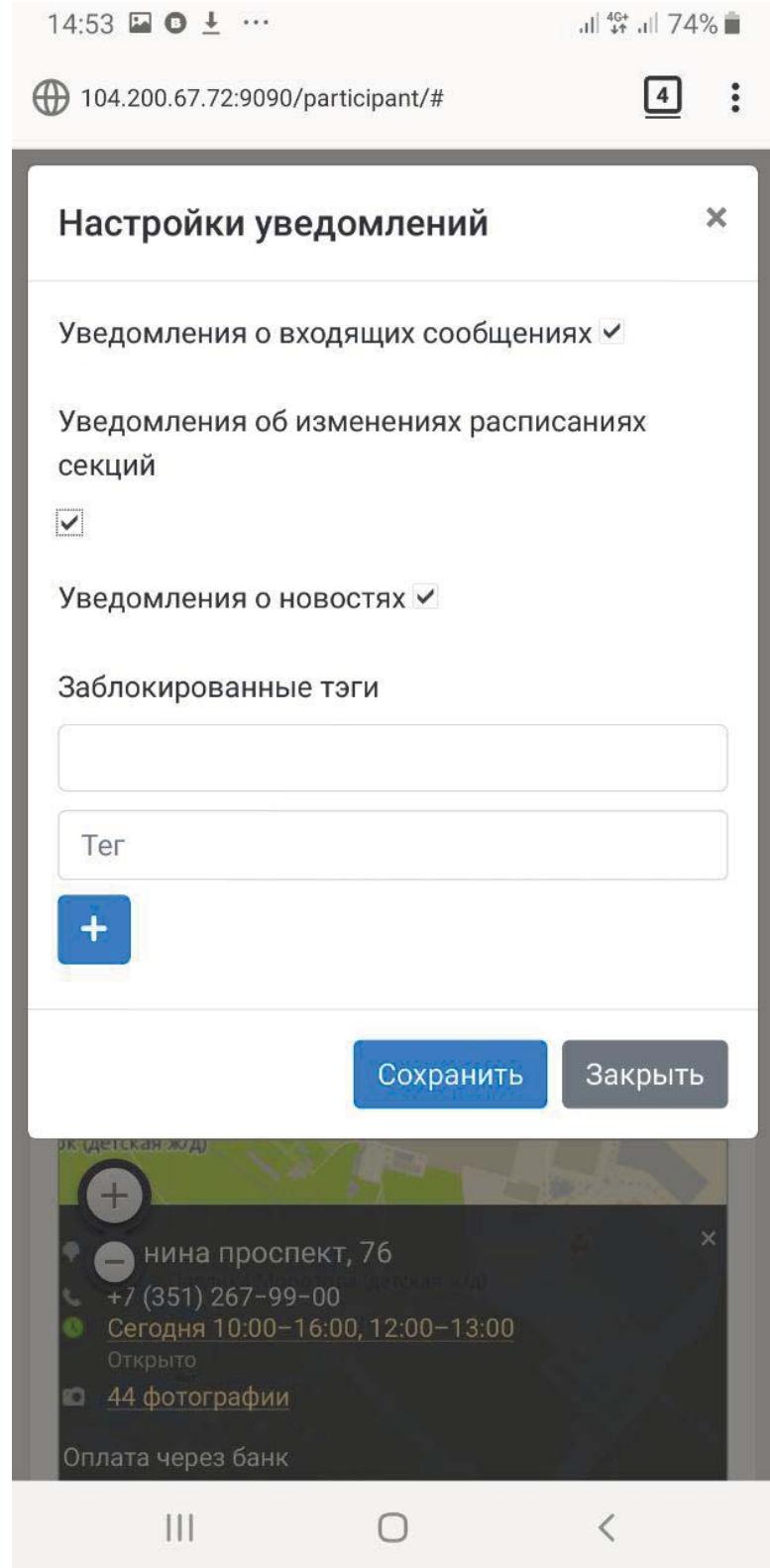


Рисунок 3.17 - Настройки уведомлений

Создание конференции ×

Название конференции
Аэродинамика самолетов

Место проведения
ЮУрГУ

Начало проведения
10.06.2019 ×

Конец проведения
14.06.2019 ×

Сохранить Закрыть

Рисунок 3.18 - Создание конференции для организатора

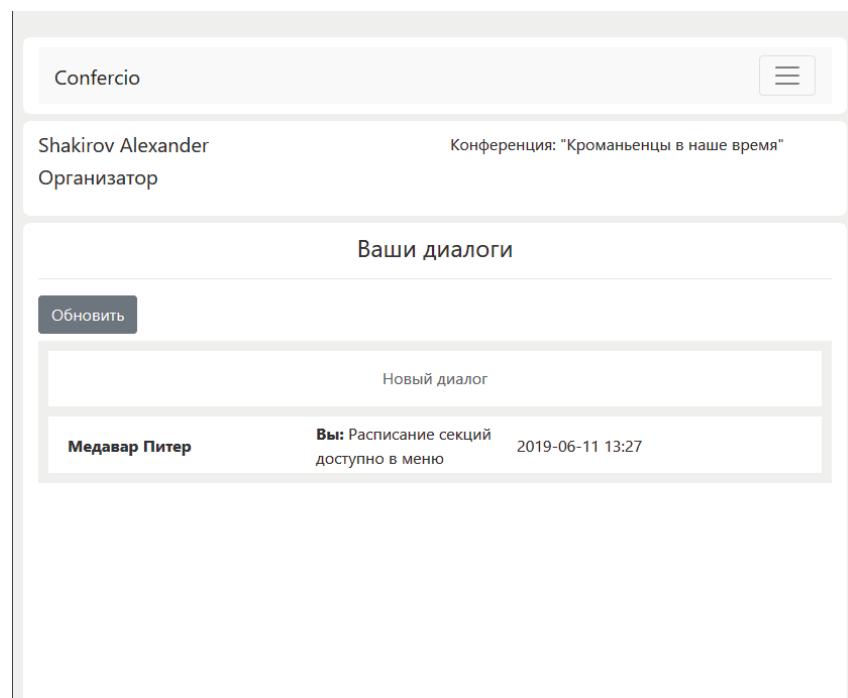


Рисунок 3.19 - Сообщения организатора

Создание секции ×

Название секции

Место

Дата начала

Модераторы

Dementev Andrey
Lisiy Arkady

Схема секции Файл не выбран.

Сохранить Закрыть

Рисунок 3.20 - Форма создания секции организатора

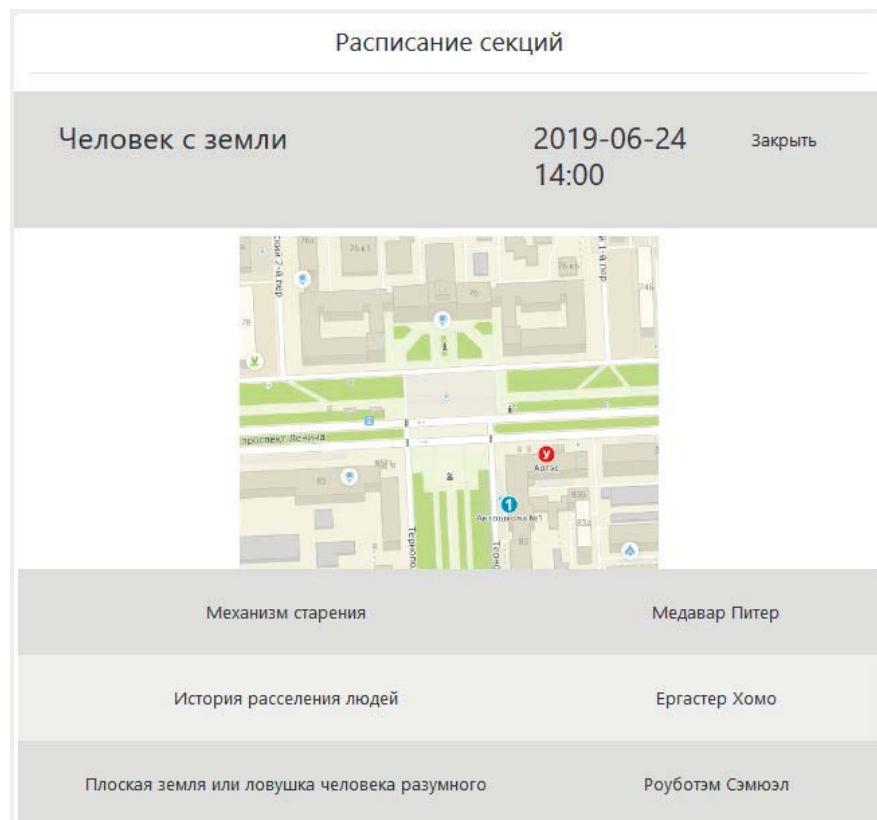


Рисунок 3.21 - Расписание секций для организатора

The screenshot shows a registration form titled 'Регистрация' with an 'X' button in the top right corner. It contains five input fields with placeholder text and two buttons at the bottom: 'Сохранить' (Save) and 'Закрыть' (Close).

Имя	Александр
Фамилия	Шакиров
Отчество	Анатольевич
Номер доклада	3.12.000000012

Сохранить Закрыть

Рисунок 3.22 - Регистрация участника для организатора

Добавление доклада для участника

Фильтр по ФИО

Участник

Номер доклада

Сохранить **Закрыть**

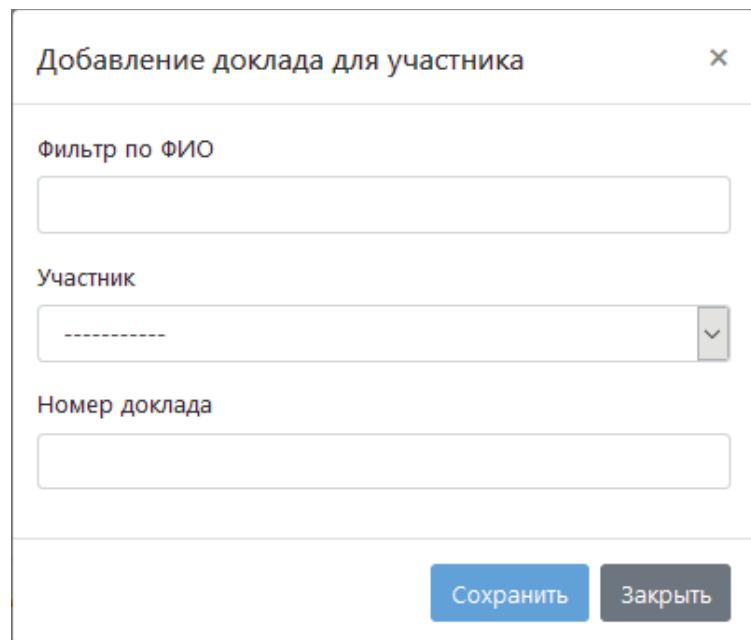


Рисунок 3.23 - Добавить доклад

Создание новости

Текст

Выбранные теги

Тег **+**

Сохранить **Закрыть**

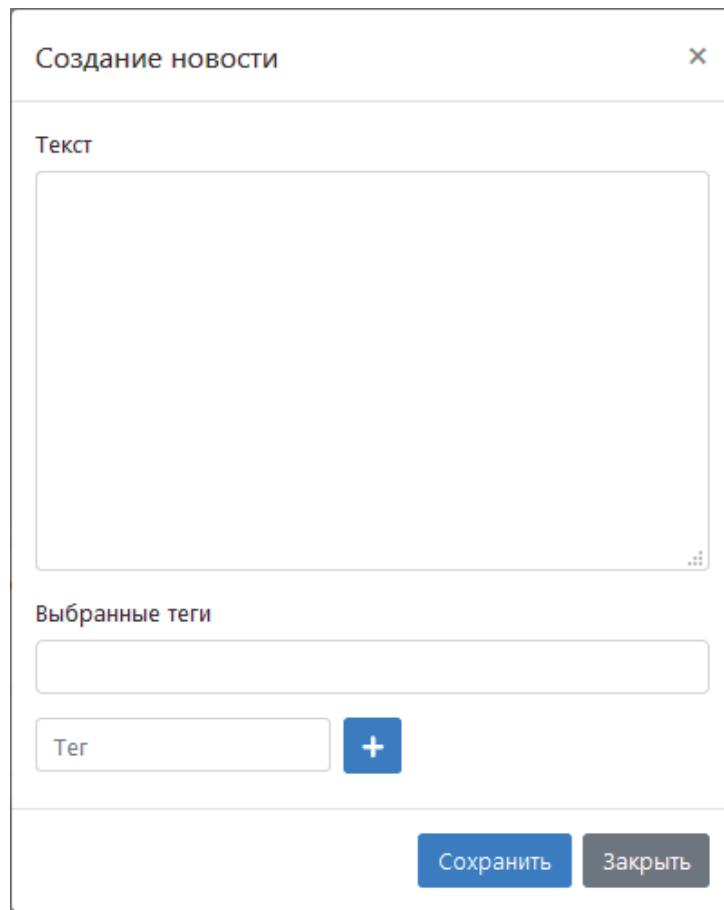


Рисунок 3.24 - Добавить новость для организатора

Shakirov Alexander
Организатор

Конференция: "Аэродинамика Самолетов"

Новости

Секция "Влияние углов атаки на полет" переносится с 10:30 на 11:30

Организатор №7

2019-Jun-Su 10:09

Рисунок 3.25 - Новости для организатора

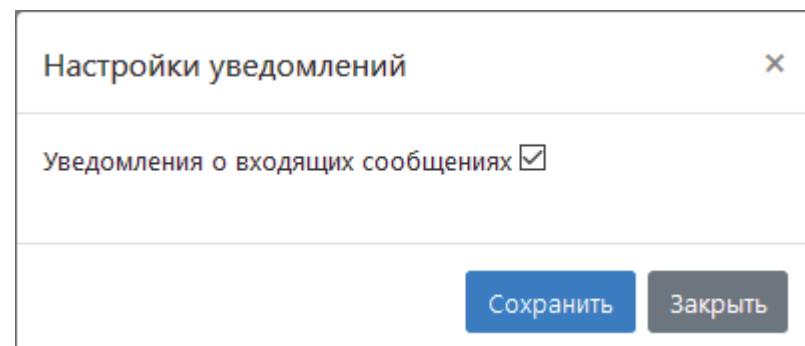


Рисунок 3.26 - Настройки уведомления для организатора

5 ТЕСТИРОВАНИЕ

5.1 МЕТОДОЛОГИЯ ТЕСТИРОВАНИЯ

В данной работе проверка качества реализации веб-приложения была реализована путем ручного функционального тестирования.

Функциональное тестирование является одним из главных видов тестирования, предназначение которого – установить соответствие реализованного программного обеспечения заданным функциональным требованиям заказчика. Другими словами, проведение функционального тестирования позволяет проверить информационную систему на способность решать задачи, необходимые пользователям.

В зависимости от уровня доступа к коду системы можно выделить два типа функциональных проверок:

- тестирование black box (черный ящик) – проведение функционального тестирования без доступа к программному коду системы,
- тестирование white box (белый ящик) – функциональное тестирование с доступом к программному коду системы.

Тестирование black box проводится без доступа к внутренним механизмам работы системы и полагается на внешние проявления ее работы. При этом тестировании проверяется поведение ПО при разных входных данных и внутреннем состоянии системы. В случае тестирования white box создаются тест-кейсы, основанные в основном на исходном коде. Также существует расширенный тип проведения black-box тестов, допускающих изучение кода, – так называемые grey box тесты (серый ящик).

Ключевые преимущества:

- Функциональное тестирование ПО полностью повторяет фактическое использование системы.
- Позволяет вовремя выявить системные ошибки ПО и, тем самым, избежать множества проблем при работе с ним в дальнейшем.
- Экономия за счет исправления ошибок на более раннем этапе разработки ПО.

5.2 ПРОВЕДЕНИЕ ПРОЦЕДУРЫ ТЕСТИРОВАНИЯ

Таблица 5.1 – Описание и результаты тестов

Тест	Последовательность действий	Результат	Пройден
Регистрация участника	<ul style="list-style-type: none"> Перейти на главную страницу участника Отсканировать QR код с ссылкой на регистрацию Заполнить форму 	<ul style="list-style-type: none"> Получить перенаправление на страницу участника 	Да
Отправка сообщения	<ul style="list-style-type: none"> Нажать кнопку новый диалог Выбрать получателя Отправить сообщение Войти под получателем 	<ul style="list-style-type: none"> Отправленное сообщение должно быть в диалогах 	Да
Добавить доклад	<ul style="list-style-type: none"> Ввести номер доклада в форму 	<ul style="list-style-type: none"> В расписании секций появился доклад Во вкладке «Ваши доклады» появился доклад 	Да
Добавить конференцию	<ul style="list-style-type: none"> Открыть форму создания конференции Заполнить форму 	На странице «Все конференции» появилась соответствующая конференция	Да

Продолжение таблицы 5.1

Добавить секцию	<ul style="list-style-type: none"> Открыть форму создания секции <p>Заполнить форму</p>	На странице расписание секций появилась соответствующая секция	Да
Добавить доклад организатором	<ul style="list-style-type: none"> Открыть форму добавления доклада <p>Заполнить форму</p>	На странице расписание секций должен появиться доклад	Да
Добавить участника	<ul style="list-style-type: none"> Открыть форму добавления участника <p>Заполнить форму</p>	В базе данных должен появиться соответствующий участник	Да
Добавить новость	<ul style="list-style-type: none"> Открыть форму добавления новости <p>Заполнить форму</p>	На странице «Новости» должна появиться соответствующая новость	Да
Изменение настроек уведомлений	<ul style="list-style-type: none"> Открыть форму изменения настроек Изменить <p>Перезагрузить страницу</p>	Настройки сохранены	Да

Продолжение таблицы 5.1

Изменение расписания секции	<ul style="list-style-type: none">• Изменить расписаниеСохранить	После перезагрузки страницы порядок докладов должен сохраниться	Да
-----------------------------	---	---	----

ЗАКЛЮЧЕНИЕ

В разрабатываемом веб-приложении сосредоточена вся необходимая информация: расписание и место расположения секций, схема проезда к месту конференции, схема помещений конференций. Налажена система коммуникации между участниками и организаторами.

В ходе дипломного проектирования выполнено следующее:

1. Проведен анализ существующих решений и предметной области;
2. Выбраны средства реализации и среда разработки, разработаны основные архитектурные решения;
3. Разработана архитектура база данных;
4. Реализованы пользовательская и серверная части приложения;
5. Протестировано разработанное веб-приложение.

В веб-приложении были реализованы следующие функции:

1. Регистрация пользователей;
2. Создание конференций;
3. Создание секций;
4. Рассылка уведомлений;
5. Закрепление модераторов на секциях;
6. Автоматическое составление расписаний;
7. Регистрация с выбором конференции по QR коду;
8. Личные сообщения;
9. Адаптивный дизайн.

В настоящее время веб-приложения готово к эксплуатации, но его необходимо доработать, в частности: доработать ролевую модель и дизайн.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Timepad. – <https://timepad.ru/about/contacts/>
2. Event.4science. – <https://event.4science.ru/>
3. Lomonosov-msu. – <https://lomonosov-msu.ru/>
4. jQuery. – <https://www.8host.com/blog/osnovy-jquery/>
5. Python. – https://studbooks.net /obzor_yazykov_programmirovaniya
6. CMS. – <https://wiki.rookee.ru/cms/>
7. Фреймворки и CMS. – <https://vc.ru/flood/33985-cms-ili-freymvork-chto-vybrat>
8. Postgresql. – <https://devacademy.ru/posts/sqlite-vs-mysql-vs-postgresql/>
9. Архитектура клиент-сервер. –
<http://www.4stud.info/networking/lecture5.html>
10. Классификацию шаблонов проектирования Мартина Фаулера. –
<http://design-pattern.ru/patterns/>
11. Патерн проектирования MVC. – <http://design-pattern.ru/patterns/mvc.html>
12. Патерн проектирования PAC. – <http://ru.knowledgr.com>
13. Патерн проектирования HMVC. – <https://ru.wikipedia.org/wiki/HMVC>
14. Патерн проектирования MVP. – <https://ru.wikipedia.org/wiki/Model-View-Presenter>
15. Патерн проектирования MVVM. – <https://metanit.com/sharp/wpf/22.1.php>
16. Функциональное тестирование. –
<http://aplana.ru/services/testing/functionalnoe-testirovanie>