МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет *компьютерных наук*
Кафедра *технологий обработки и защиты информации*

Приложение «Мой ФКН»

Курсовой проект

по дисциплине

Технологии программирования

09.03.21 Информационные системы и технологии
Обработка информации и машинное обучение

6 семестр 2022/2023 учебного года

Зав. Кафедрой _____д. т. н., профессор А. А. Сирота
Обучающийся _____Н. В. Мерзляков, 3 курс, д/о
Обучающийся _____А. В. Сиваков, 3 курс, д/о
Обучающийся _____Д. И. Кулинченко, 3 курс, д/о
Обучающийся _____И.М. Кудинов, 3 курс, д/о
Руководитель _____ В.С. Тарасов, ст. преподаватель __.__.20__.

Воронеж 2023

Содержание

Введение

В наше время во время обучения в университете очень сложно обойтись без применения информационных технологий. ФКН стремится идти в ногу со временем и поэтому неудивительно, что всё новые и новые технологии находят свое применение в образовательном процессе. Тем не менее, за время обучения у авторов этой работы накопилось некоторое количество идей, реализация которых могла бы способствовать автоматизации и упрощению образовательного процесса как для студентов, так и для преподавателей. За основу была взята идея доработки уже имеющихся технологических решений. Основной идеей данной работы является упрощение использования уже имеющихся функций информационных систем, применяемых на факультете и добавление новых, которых не хватает.

Внедрение данных разработок поможет упростить ряд задач, неизбежно возникающих при реализации образовательных задач факультета, что в конечном счёте будет способствовать улучшению качества образовательного процесса.

Как уже отмечалось выше, за основу были взяты уже имеющиеся решения. В процессе разработки приложения одной из задач стало сохранение преимуществ этих решений и исправление недостатков. Был сделан упор на функциональность и простоту использования: приложение должно решать ряд задач, но при этом быть простым, интуитивно понятным для всех пользователей и не перегруженным лишними функциями.

Как показывают многочисленные исследования [1], одной из главных проблем российских ВУЗов является плохая коммуникация между студентами, преподавателями и деканатами. Именно поэтому факультету важно иметь мобильное приложение, чтобы решить проблему с устаревшими и неэффективными способами коммуникации.

1 Используемые определения

Таблица 1 - Используемые определения

Термин	**Определение термина**
Аватар	Изображение, используемое в учетной записи для персонализации пользователя.
Авторизация	Предоставление определённому лицу прав на выполнение определённых действий; а также процесс проверки (подтверждения) данных прав при попытке выполнения этих действий.
Авторизованный пользователь	Пользователь прошедший процесс авторизации
Агрегатор (приложение-агрегатор)	Приложение, объединяющее в себе услуги нескольких компаний, данные из нескольких источников и/или функции нескольких приложений или сайтов.
Администратор	Человек, имеющий доступ к расширенному функционалу веб-сервиса, имеющий знания о формате приема статей.
Боковое меню (сайд меню)	Меню, которое представляет собой панель, которая находится (или открывается, при помощи, каких-либо кнопок или жестов) снизу, слева или справа от области основного контента приложения, содержащая вертикальную, независимую от основного контента приложения прокрутку, и служит основным инструментом навигации в приложении.
Веб-сервис	Идентифицируемая уникальным веб-адресом (URL-адресом) программная система со стандартизированными интерфейсами, а также HTML-документ сайта, отображаемый браузером пользователя.
Виртуальный ассистент	Программный агент, который может выполнять задачи для пользователя на основе информации, введенной пользователем, данных о его местонахождении, а также информации, полученной из различных интернет-ресурсов.
Неавторизованный пользователь	Пользователь, не прошедший процесс аутентификации

Термин	Определение термина
Ошибка (Bug)	Общий термин, используемый для обозначения непредвиденной ошибки или дефекта в аппаратном или программном обеспечении, что приводит к его неисправности.
Пользователь	Лицо, которое использует действующую систему для выполнения конкретной функции.
Профиль (в веб-приложении)	Учетная запись пользователя в веб-приложении, вход в которую осуществляется с помощью логина / номера телефона / e-mail и пароля. В учетной записи содержится информация о пользователе.
СУБД	Система управления базами данных. Комплекс программ, позволяющих создать базу данных (БД) и манипулировать данными.
Фреймворк	Программные продукты, которые упрощают создание и поддержку технически сложных или нагруженных проектов.
Чат	Средство обмена сообщениями по компьютерной сети в режиме реального времени.
Android	Операционная система для мобильных устройств.
API	Программный интерфейс приложения. Описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой
CSS	Формальный язык, служащий для описания оформления внешнего вида документа, созданного с использованием языка разметки (HTML, XHTML, XML).
Django	Свободный фреймворк для веб-приложений на языке Python, использующий шаблон проектирования MVC.
Django ORM	Инструмент фреймворка Django, который позволяет взаимодействовать с базами данных, используя высокоуровневые методы Python, а не SQL-запросы.
Flutter	Комплект средств разработки и фреймворк с открытым исходным кодом для создания мобильных приложений под Android и iOS.

Термин	Определение термина
Front-end	Пользовательский интерфейс компьютера или любого устройства.
HTML	Стандартизированный язык разметки веб-страниц во Всемирной паутине.
iOS	Мобильная операционная система для смартфонов, электронных планшетов, носимых проигрывателей, разрабатываемая и выпускаемая американской компанией Apple
JavaScript	Мультипарадигменный язык программирования, используется как встраиваемый язык для программного доступа к объектам приложений.
MVC	Схема разделения данных приложения и управляющей логики на три отдельных компонента: модель, представление и контроллер.
PostgreSQL	Свободная объектно-реляционная система управления базами данных.
Python	Высокоуровневый язык программирования общего назначения.
REST	Архитектурный стиль взаимодействия компонентов распределенного приложения в сети.
WebView	Системный компонент, которое отвечает за открытие веб-страниц в рамках другого приложения.

2 Постановка задачи

Целью данного курсового проекта является разработка самостоятельного приложения для упрощения образовательного процесса на факультете ФКН. Приложение не должно полностью дублировать функции, уже реализованные при помощи других технических решений, в частности, платформы moodle, но должно содержать функции, которых еще нет, или те, которыми неудобно пользоваться в текущем виде.

2.1 Требования к функциональной части

Приложение должно отвечать следующим функциональным требованиям:

— Регистрация и авторизация пользователей;

— Реализация разделений ролей студент/преподаватель;

— Интеграция с moodle;

— Интеграция с BRS;

— Реализация карты факультета;

— Реализация чатов;

— Создание чат-бота;

— Реализация ленты событий.

2.2 Технические требования

Приложение должно удовлетворять следующим техническим требованиям:

— Возможность авторизации пользователей с помощью логина/пароля;

— Хранение данных для входа в сервисы moodle и BRS;

— Хранение необходимых данных в БД;

— Наличие панели администратора.

2.3 Требования к интерфейсу

— Интерфейс приложения должен удовлетворять следующим критериям:

— Логичность и интуитивная понятность для пользователя;

— Наличие единой цветовой гаммы и стиля исполнения;

— Читабельность текста;

— Отсутствие лишних деталей, отсутствие перегруженности;

— Оптимизация для разных экранов.

3 Анализ предметной области

3.1 Анализ существующих решений

Анализ существующих решений будем проводить на основе данных сервисов:

Таблица 2 - Примеры существующих решений

FlipTable	https://fliptable.ru/
Университет в кармане	http://moyuniver.ru/
Blackboard	https://www.blackboard.com/
SDU Informer	https://sdu2.software.informer.com/
Мобильное приложение «Цифровой университет МГЮА»	https://msal.ru/content/tsifrovoy-universitet/studentam/mobilnoe-prilozhenie/
SUAI Pocket: Расписание ГУАП	https://vk.com/suainav

3.1.1 FlipTable

Российская платформа, облегчающая создание и просмотр расписания. Может работать в браузере без установки каких-либо приложений. Обладает рядом полезных возможностей: экспорт расписания, многопользовательский доступ, импорт данных и т.д. Пользователи системы имеют возможность просматривать расписание в любом удобном для них формате: xls, *pdf, *iCal, в

онлайн-формате или мобильном приложении. По данным разработчиков, данная система успешно используется в 10 учебных заведения России [2].

Преимущества:

- Использует свободно распространяемое ПО;
- Поддерживает разнообразные форматы;
- Упрощает создание расписания: система автоматически обнаруживает «накладки»;
- Предоставляет инструменты для аналитики;
- Простая в использовании.

Недостатки:

- Платная: в бесплатной версии есть реклама и нет некоторых возможностей, например, экспорта в iCal и приложение «Расписание ВУЗов»;
- Практически невозможно найти отзывы на сторонних ресурсах. Их фактически нет после того, как приложение было удалено из Play Market. Разработчики также предоставляют только общую информацию о продукте. Следовательно, до начала использования невозможно оценить риск возможных проблем и наличие недостатков.

3.1.2 Университет в кармане.

Представляет из себя онлайн-платформу и экосистему мобильных приложений для решения любых учебных задач. Содержит свободно распространяемую информацию как для студентов, так и для преподавателей. Является не одним конкретным приложением, а целой экосистемой. Проект был спонсирован компанией Microsoft. На сайте можно задавать вопросы и искать учебные и методические материалы, что может помочь в образовательном

процессе. Сайт обладает простым и понятным интерфейсом. Однако можно отметить слишком минималистичный и устаревший дизайн.

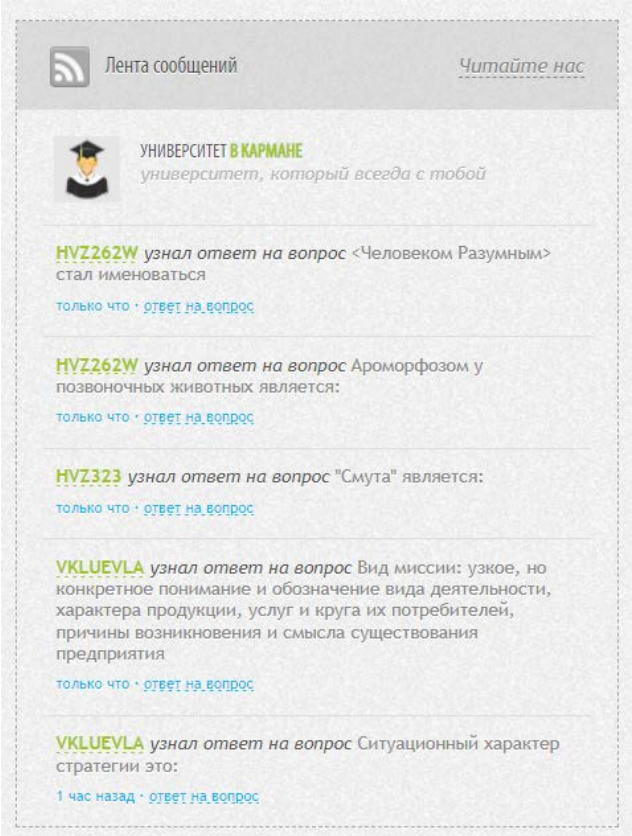

Рисунок 1 - Лента сообщений на сайте «Университет в кармане»

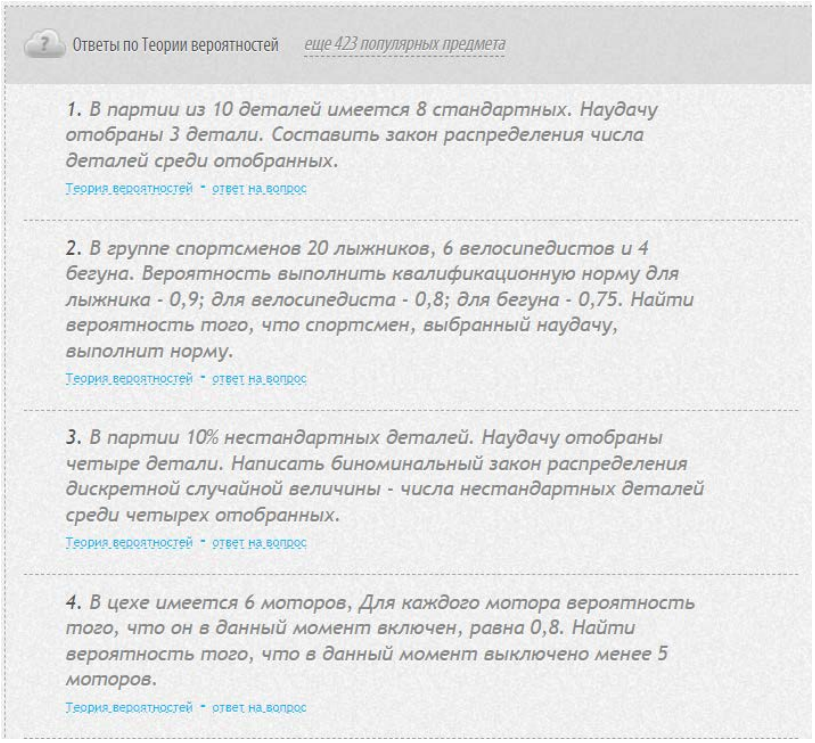

Рисунок 2 - Ответы на сайте «Университет в кармане»

Преимущества:

— Может применяться как студентами, так и преподавателями;

— Наличие экосистемы из приложений, которые могут удобно взаимодействовать друг с другом;

— Простой интерфейс и наличие поиска по сайту.

Недостатки:

— Ограниченность функций: сервис не имеет никаких функций, связанных, например, с расписанием. Фактически, его цель – лишь помощь в обмене данными и освоении учебного материала, а для преподавателей – помощь в распространении методических материалов;

— Сервис направлен скорее не на упрощение процесса образования, а на обман образовательной системы. Так, например, на нём можно найти генераторы текста для сочинений и предложения о выполнении учебных работ за деньги. Таким образом, внедрение подобной системы на факультете будет несколько неуместным и даже вредным действием;

— Устаревший дизайн;

— На момент написания данной работы могут возникать проблемы при попытке воспользоваться данным сервисом без VPN.

3.1.3 BlackBoard

Зарубежный коммерчески успешный продукт, используемый во многих учебных заведениях, например, США. Представляет из себя не полноценное приложение, а своеобразный конструктор, позволяющий каждому ВУЗу собрать своё приложение. BlackBoard предоставляет ряд возможностей: удобную

загрузку новостных лент, карты университетов, поиск контактов, возможность подключения DropBox, календарь событий и многое другое.

Рисунок 3 - Иллюстрация внешнего вида сайтов, созданных при помощи BlackBoard

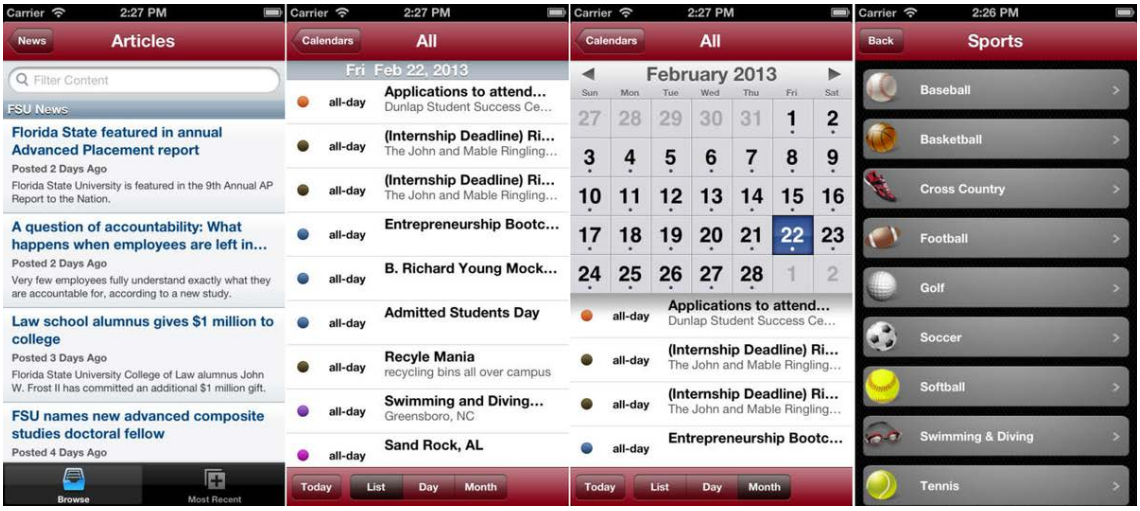

Рисунок 4 - Интерфейс приложения Florida State University, основанного на BlackBoard

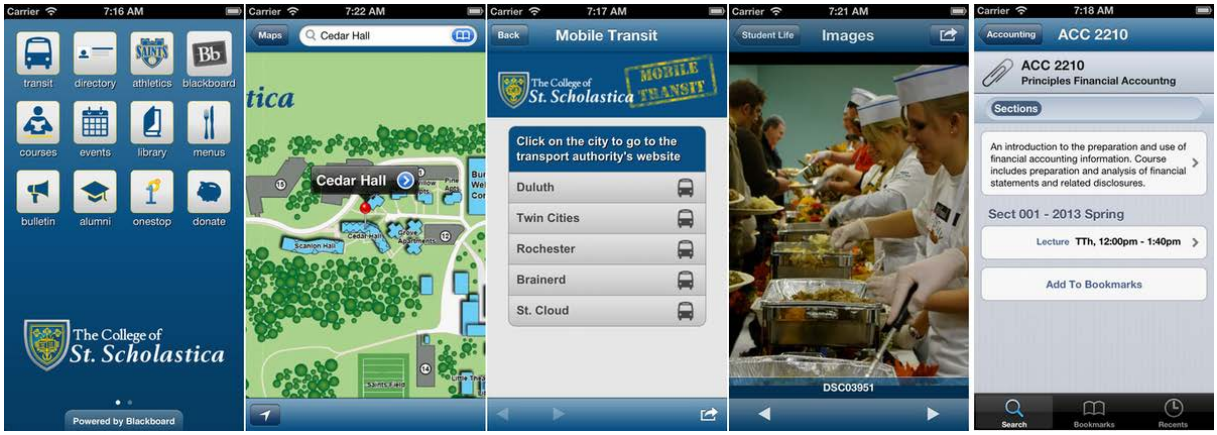

Рисунок 5 - Интерфейс приложения College of St. Scholastica, основанного на Blackboard.

Преимущества:

— Простота создания приложений;

— Большое количество разнообразный функций;

— Большинство необходимых функций уже реализовано – не надо ничего создавать самостоятельно.

Недостатки:

— Является конструктором, а не полноценным приложением. Следовательно, требуются дополнительные ресурсы на разработку;

— Является коммерческим и, следовательно, платным продуктом;

— Некоторые функции, например, фотогалерея, реализованы плохо;

— Не предусмотрен вывод практически никакой персонализированной информации для студентов.

3.1.4 SDU Informer.

Система, разработанная казахстанских университетом им. С. Демиреля. Обладает продвинутой лентой новостей, постоянно держит пользователя в курсе

событий. Также приложение содержит обновленную информацию об университете, факультетах, студенческих клубах, контактах, последних новостях и библиотеку с учебниками в формате e-pub, авторами которых являются преподаватели ВУЗа.

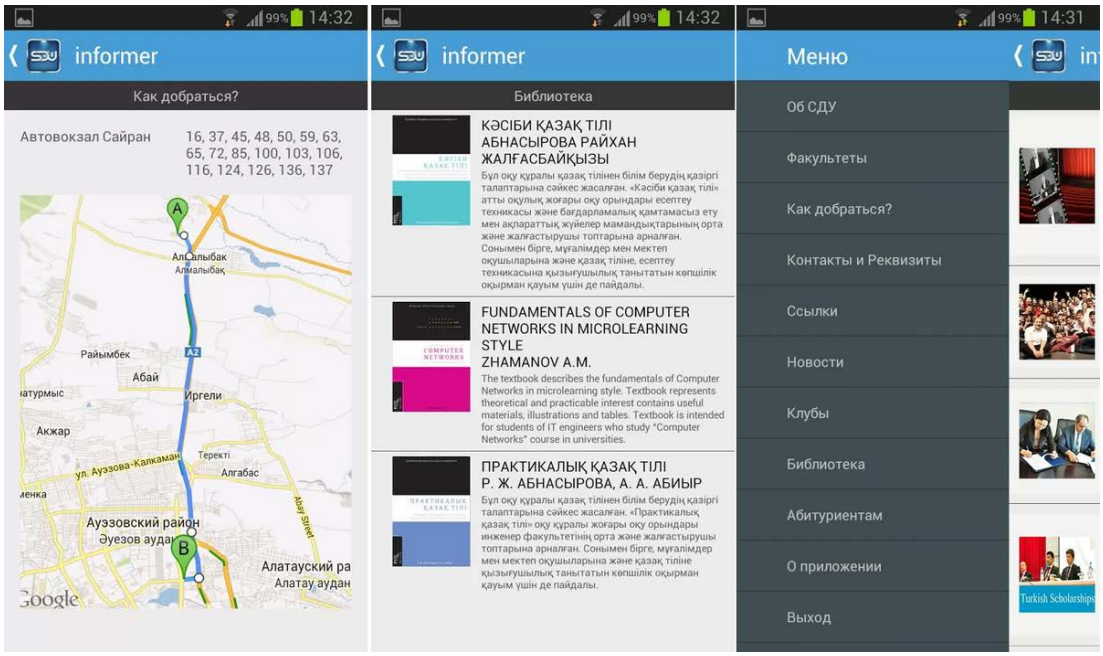

Рисунок 6 - Интерфейс SDU Informer.

Преимущества:

— Удобный интерфейс;

— Современный дизайн;

— Функциональность.

Недостатки:

— Пользователи отмечают нестабильную работу.

3.1.5 Мобильное приложение «Цифровой университет МГЮА»

Приложение, используемое Московским Государственным Юридическим Университетом. В приложении публикуется личная информация по расписанию занятий, новостям и мероприятиям. Имеет простой интерфейс. Личные кабинеты

студентов и преподавателей отличаются и в зависимости от роли приложение предлагает разные функции.

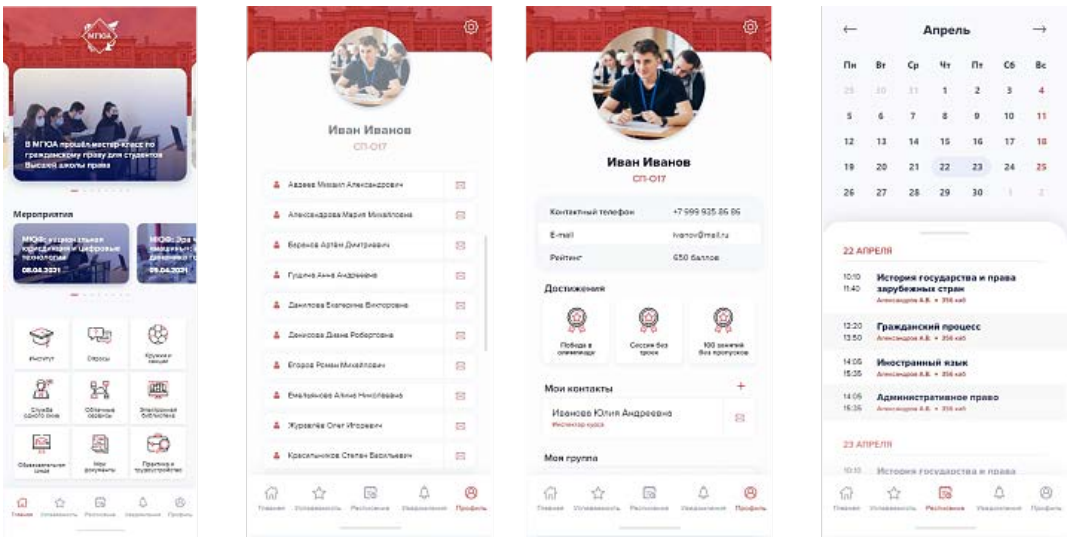

Рисунок 7 - Интерфейс приложения «Цифровой университет МГЮА»

Преимущества:

— Простой, понятный, не перегруженный интерфейс;

Недостатки:

— Низкая функциональность;

— Проблемы со входом в учётную запись и нестабильная работа.

3.1.6 SUAI Pocket: Расписание ГУАП

Приложение, используемое Санкт-Петербургским государственным университетом аэрокосмического приборостроения. Имеет ряд полезных функций для студентов. Отдельно стоит отметить удобный просмотр расписания. Также есть полезная функция просмотра заданий с приближающимся сроком сдачи. Также есть возможность просмотра полезной информации об университете и преподавателей. В отличие от большинства аналогов, имеет функцию загрузки отчётов по заданиям. При этом преподаватели имеют возможность прямо в приложении дать отзыв на задание.

Преимущества:

— Современный дизайн;

— Удобный интерфейс;

— Множество полезных функций.

Недостатки:

— Перегруженность функциями, необходимость которых сомнительна.

3.2 Итог анализа

В процессе анализа предметной области было установлено, что при разработке приложения следует придерживаться следующих аспектов:

— Чат для удобной коммуникации;

— Простой и понятный интерфейс;

— Наличие хорошей ленты новостей и календаря событий;

— Стабильность работы;

— Возможность использования как студентами, так и преподавателями;

— Удобный просмотр расписания;

— Современный дизайн;

— Самостоятельность приложения;

— Наличие карты факультета;

— Своевременное обновление информации и информирование о происходящих событиях.

Также было установлено, что необходимо избегать таких ошибок, как:

— Нестабильная работа;

— Устаревший дизайн и неудобный интерфейс;

— Низкая, ограниченная функциональность;

— Недостаток возможностей для коммуникации;

— Создание «конструктора» вместо полноценного приложения.

3.3 Анализ потребности

Можно заключить, что факультет нуждается в простом, но при этом функциональном и стабильном мобильном приложении. Имеющиеся на рынке решения либо недоступны для приобретения факультетом, либо обладают определёнными недостатками. Существуют альтернативы разрабатываемого приложения, например, такие как уже используемая платформа moodle. Однако, их недостатки негативно сказываются на удобстве образовательного процесса. Поэтому приложение, разрабатываемое в ходе данного проекта, является лучшим решением, чем имеющиеся на данный момент сервисы.

4 Графическое описание работы системы

4.1 Диаграмма IDEF0

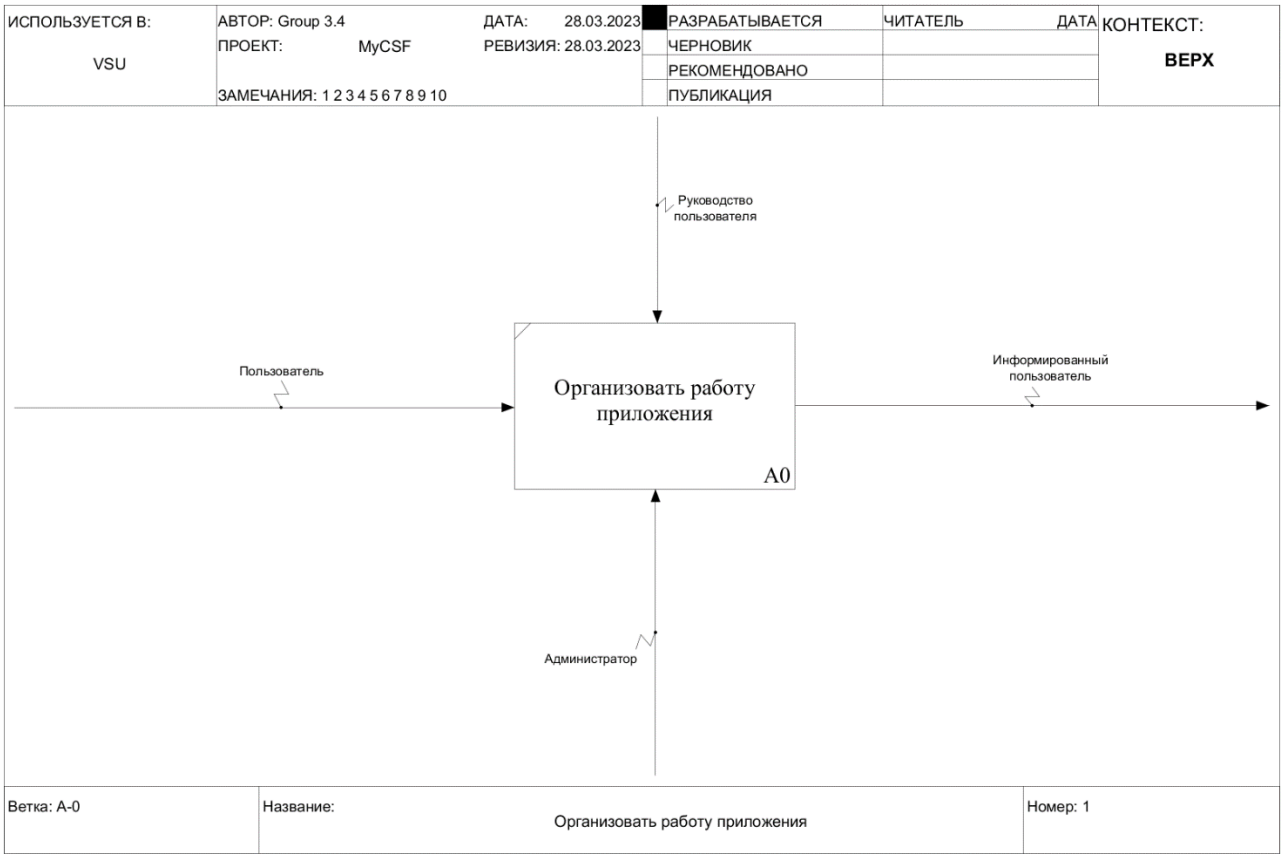

Рисунок 8 - диаграмма IDEF0

На рисунке 8 представлена диаграмма IDEF0. Данная диаграмма используется для создания функциональной модели,
отображающей структуру и функции системы, а также потоки информации и материальных объектов, связывающих эти функции. Как видно, на вход блоку «Организовать работу приложения» поступают пользователь, администратор и руководство пользователя, а на выходе получается информированный пользователь.

4.2 Диаграмма активности

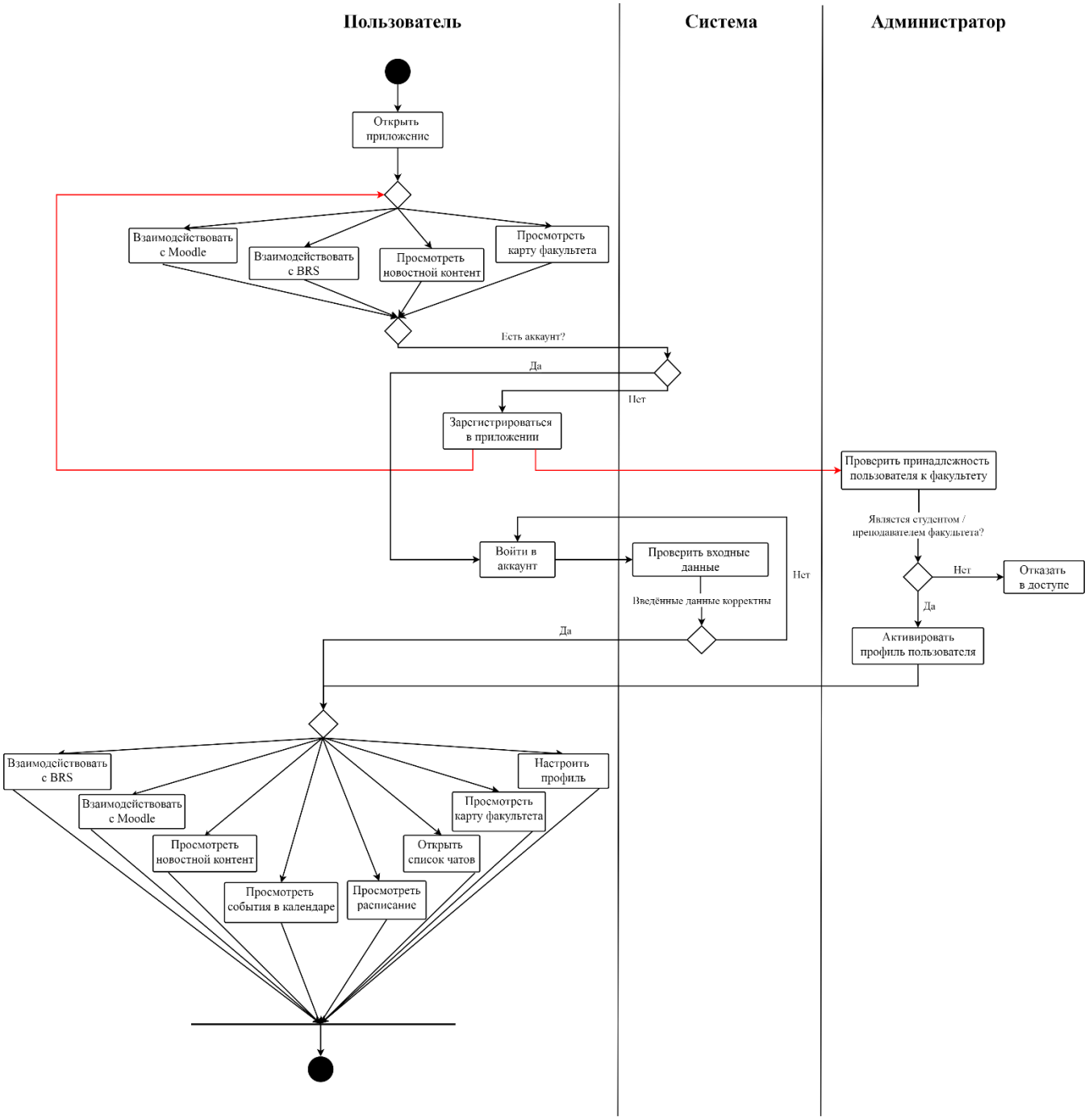

Рисунок 9 - Диаграмма активности

На рисунке 9 приведена диаграмма активностей, используемая для моделирования происходящих процессов и выполняемых последовательностей действий. Как видим, пользователь может открыть приложение, но если у него нет аккаунта, то для выполнения некоторых действий ему придётся зарегистрироваться. В противном же случае для выполнения этих же действий

ему будет необходимо авторизоваться. На диаграмме можно видеть последовательность процедур, которые будут при этом происходить.

4.3 Диаграммы прецедентов

Далее будут приведены диаграммы Use-Case (диаграммы прецедентов) для разных пользователей. Эти диаграммы демонстрируют различные сценарии, возникающие при использовании приложения.

4.3.1 Диаграмма прецедентов (admin)

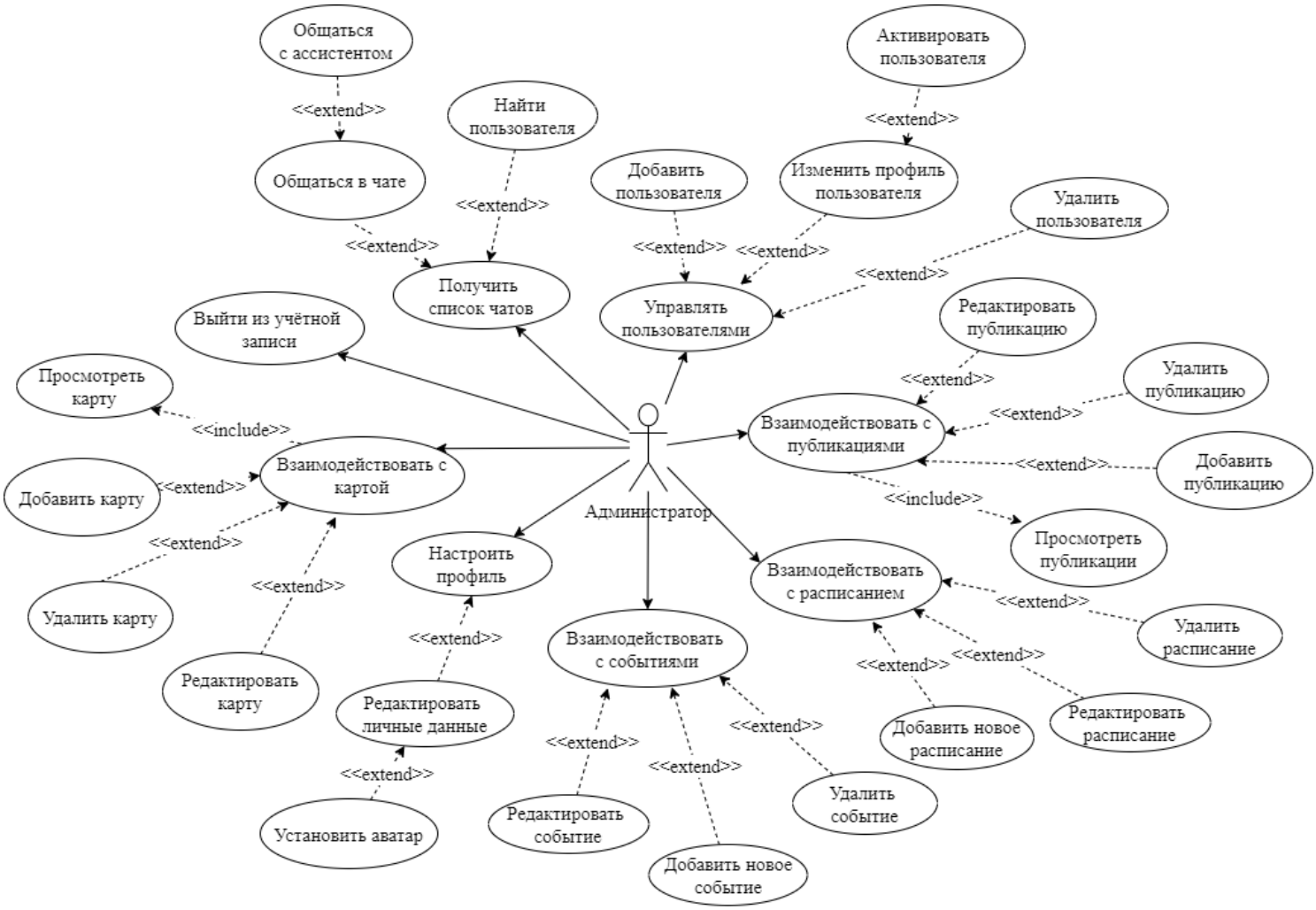

Рисунок 10 - Диаграмма прецедентов (admin)

На данной диаграмме видны все действия администратора а также их возможные расширения и включения:

— Получение списка чатов;

— Управление пользователями;

— Взаимодействие с публикациями, расписанием, событиями;

— Настройка профиля;

— Взаимодействие с картой;

— Выход из учётной записи.

4.3.2 Диаграмма прецедентов (student)

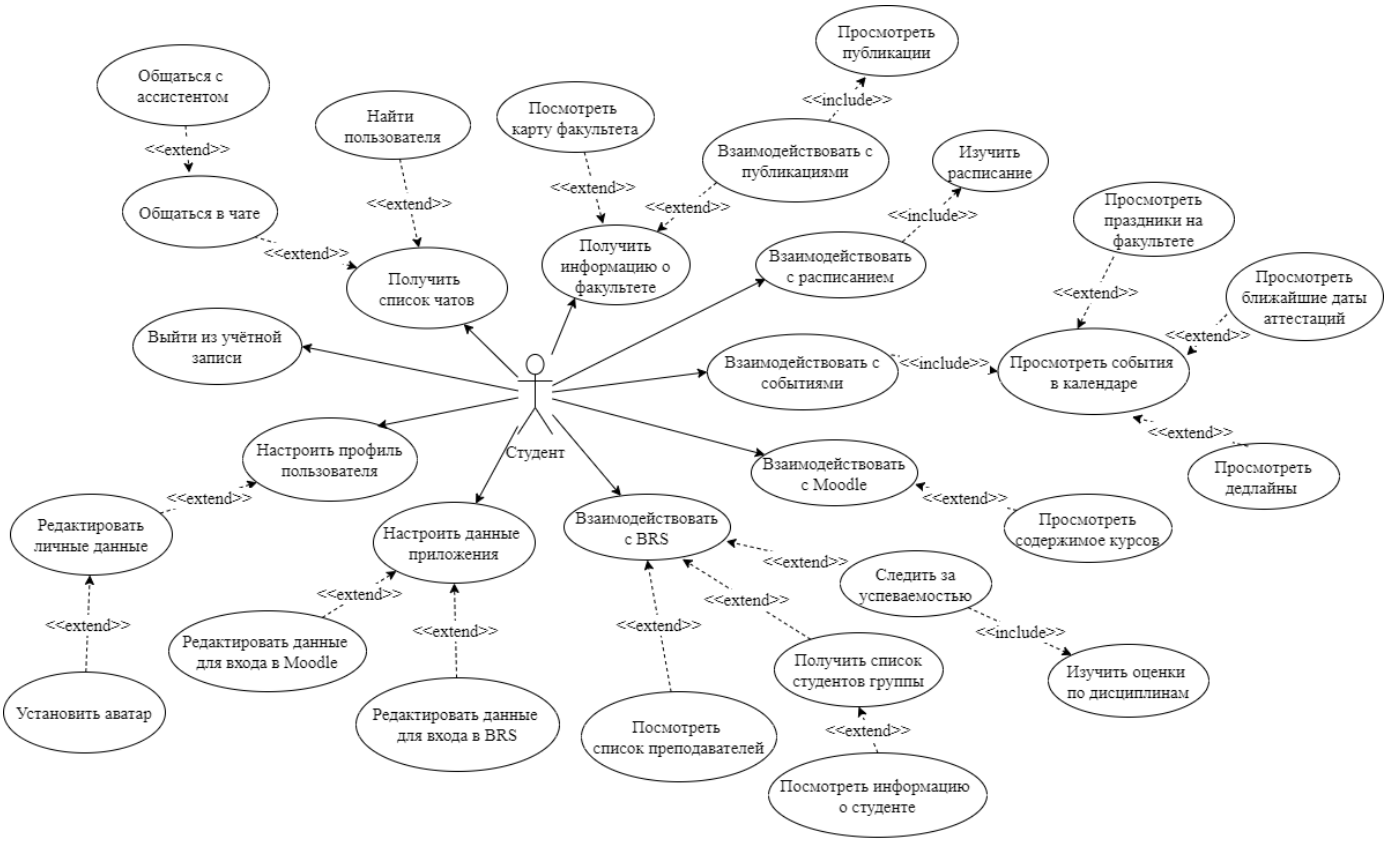

Рисунок 11 - Диаграмма прецедентов (student)

На данной диаграмме представлены все действия студента их возможные расширения и включения:

— Настройка профиля пользователя;

— Получение информации о факультете;

— Получение списка чатов;

— Взаимодействие с расписанием, событиями, moodle, BRS;

— Настройка приложения;

— Выход из учётной записи.

4.3.3 Диаграмма прецедентов (teacher)

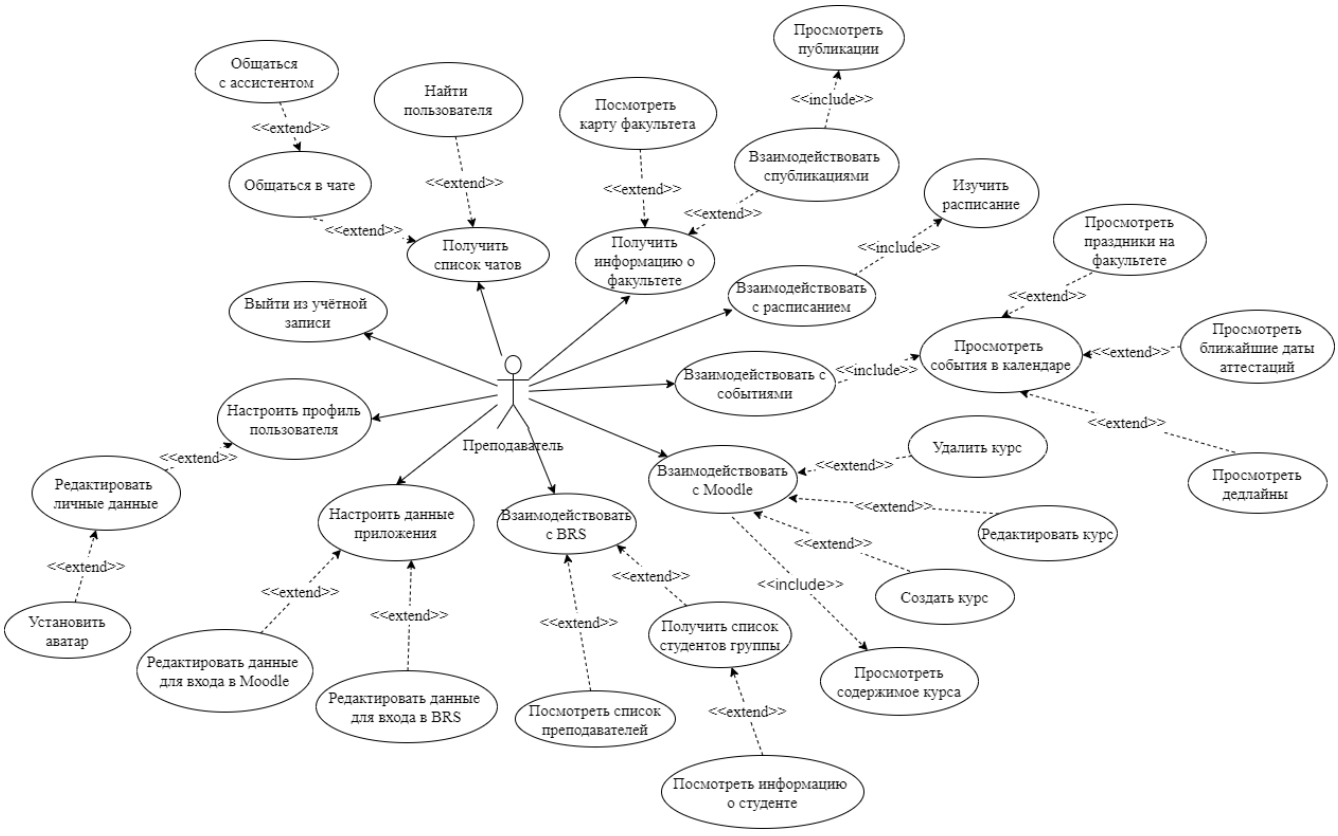

Рисунок 12 - Диаграмма прецедентов (teacher)

На данной диаграмме представлены все действия преподавателя их возможные расширения и включения:

— Получение списка чатов;

— Получение информации о факультете;

— Взаимодействие с расписанием, событиями, moodle, BRS;

— Настройка данных в приложении;

— Настройка профиля пользователя;

— Выход из учётной записи.

4.3.4 Диаграмма прецедентов (unauthorized user)

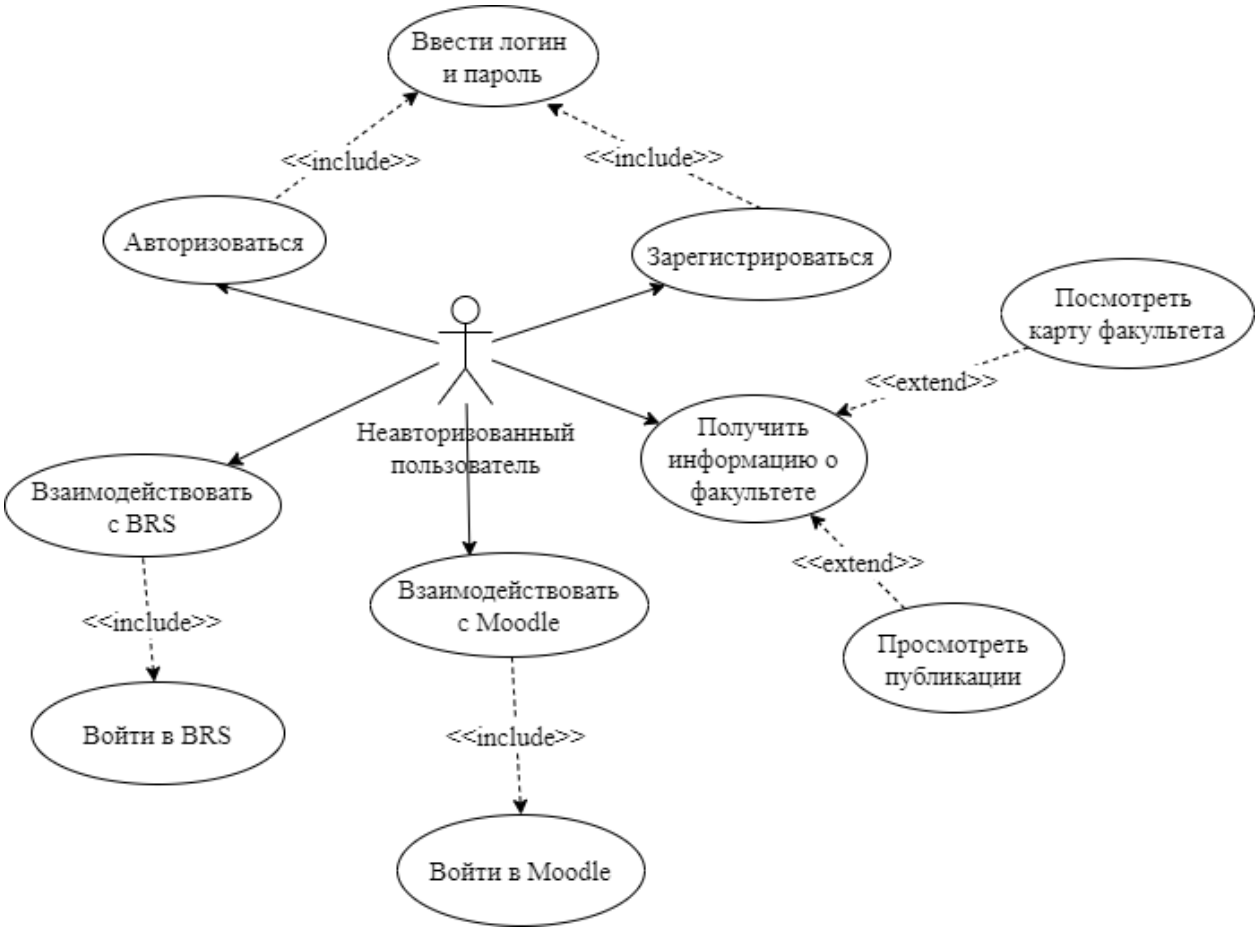

Рисунок 13 - Диаграмма прецедентов (unauthorized user)

На данной диаграмме представлены все действия преподавателя их возможные расширения и включения:

— Авторизация;

— Регистрация;

— Получение информации о факультете;

— Взаимодействие с moodle и BRS.

4.4 Диаграмма развёртывания

Далее приведена диаграмма развёртывания. Она используется для моделирования архитектуры системы, ее компонентов и их размещения на устройствах. Она показывает физическую структуру системы, то есть как ее компоненты располагаются на устройствах и как они взаимодействуют между собой.

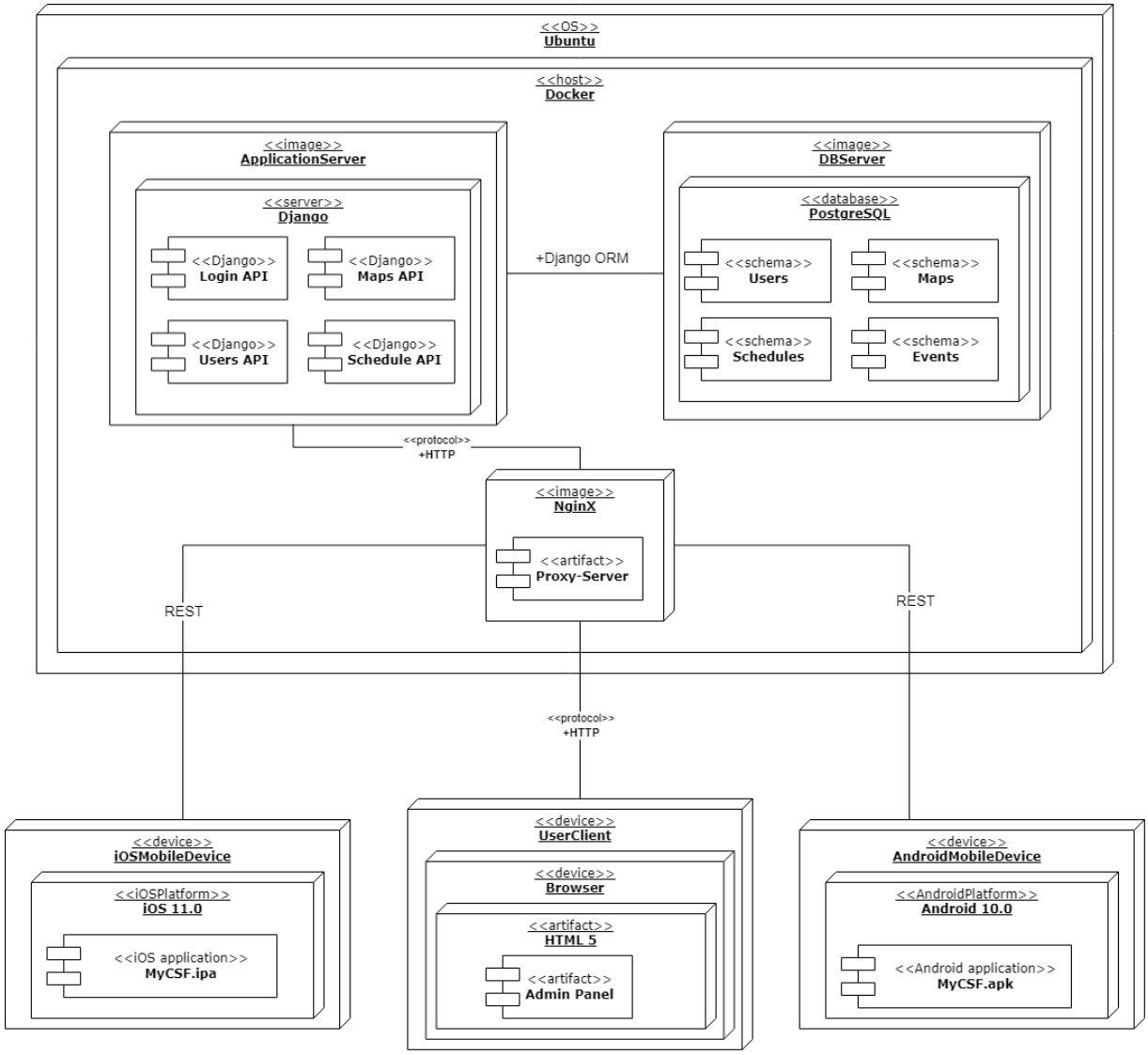

Рисунок 14 - Диаграмма развёртывания

4.5 Диаграммы состояний

Далее приведены диаграммы состояний. Они используются для моделирования поведения системы в различных состояниях. Эти диаграммы показывают, как система реагирует на внешние события и как она изменяет свое состояние в ответ на эти события.

4.5.1 Диаграмма состояний (mobile app)

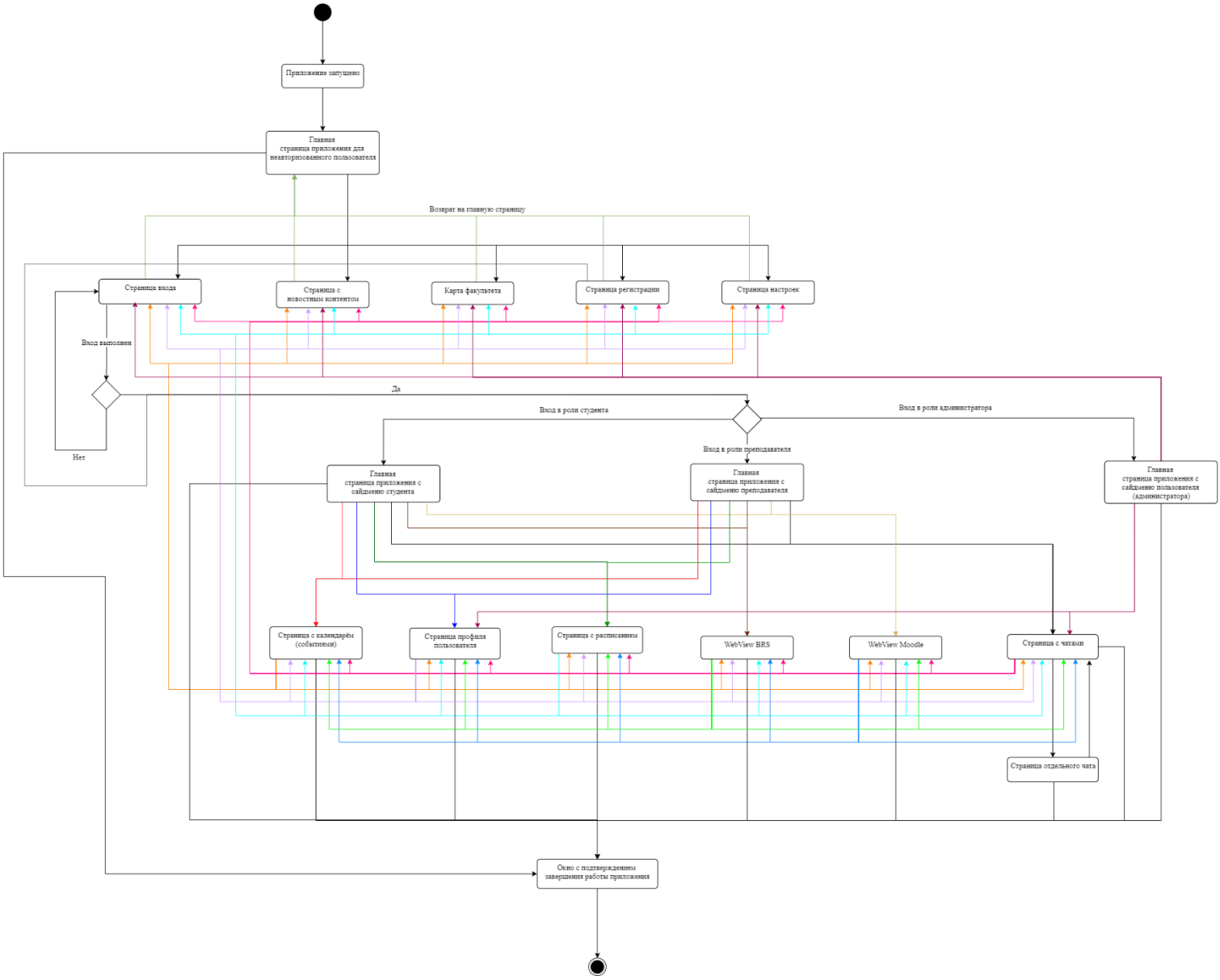

Рисунок 15 - Диаграмма состояний (mobile app)

4.5.2 Диаграмма состояний (user)

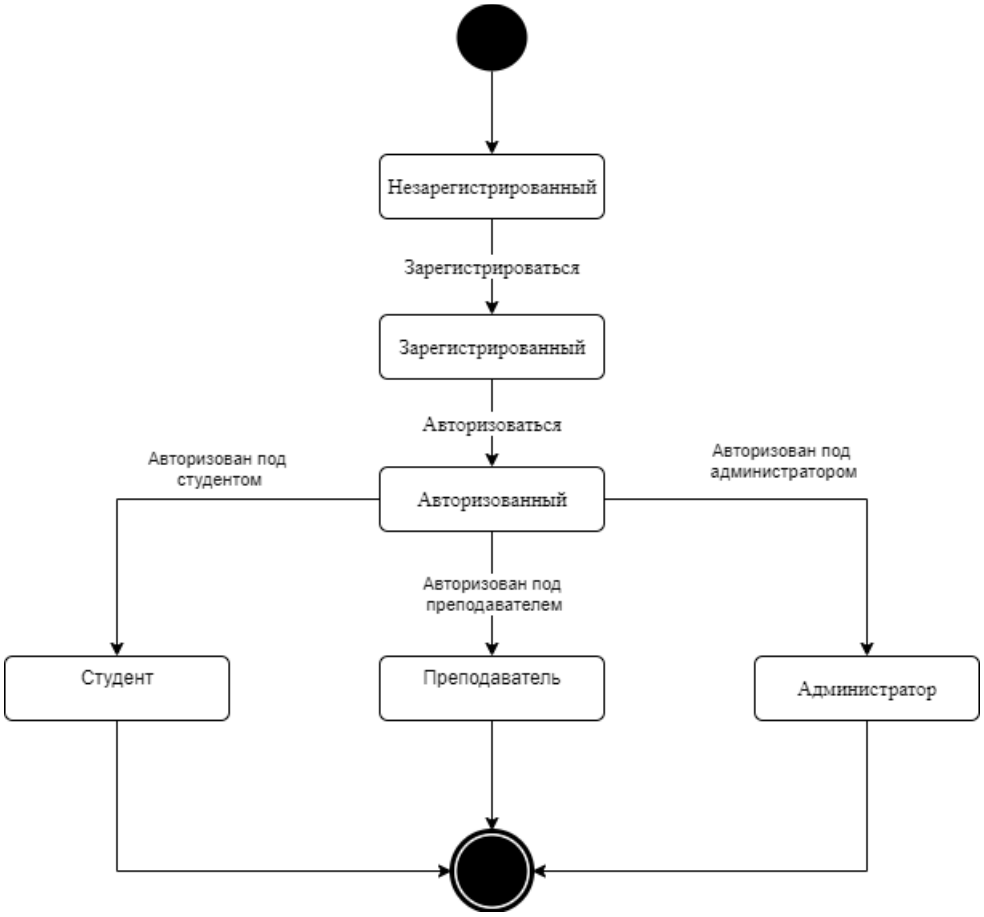

Рисунок 16 - Диаграмма состояний (user)

4.6 Диаграммы сотрудничества

Далее будут приведены диаграммы сотрудничества. Они используются для моделирования взаимодействия между объектами в системе и показывают, как объекты обмениваются сообщениями и как они взаимодействуют друг с другом для выполнения определенной функции.

Рисунок 17 - Авторизация пользователя

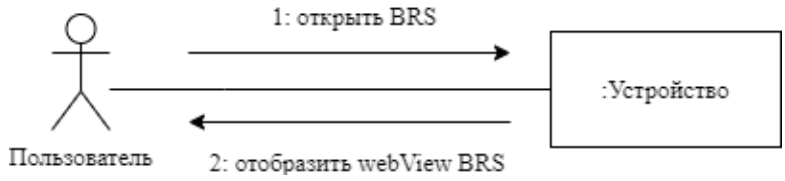

Рисунок 18 - Взаимодействие с BRS

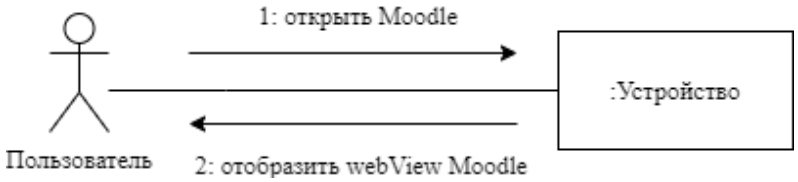

Рисунок 19 - Взаимодействие с Moodle

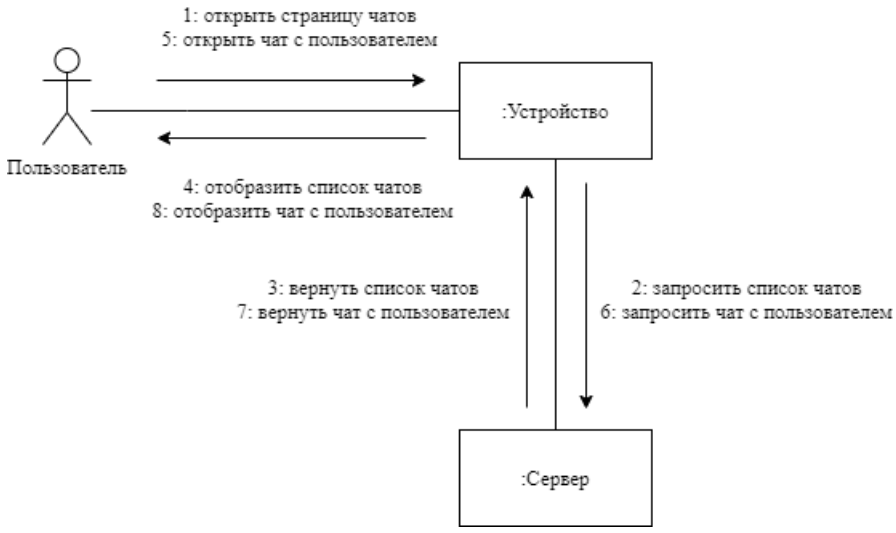

Рисунок 20 - Общение с пользователями

Рисунок 21 - Просмотр карты факультета

Рисунок 22 - Просмотр расписания

Рисунок 23 - Просмотр события

Рисунок 24 - Регистрация пользователя

Рисунок 25 - Редактирование профиля пользователя

4.7 Диаграмма последовательности

Далее приведена диаграмма последовательности, позволяющая описать последовательность происходящих событий при взаимодействии пользователя с системой.

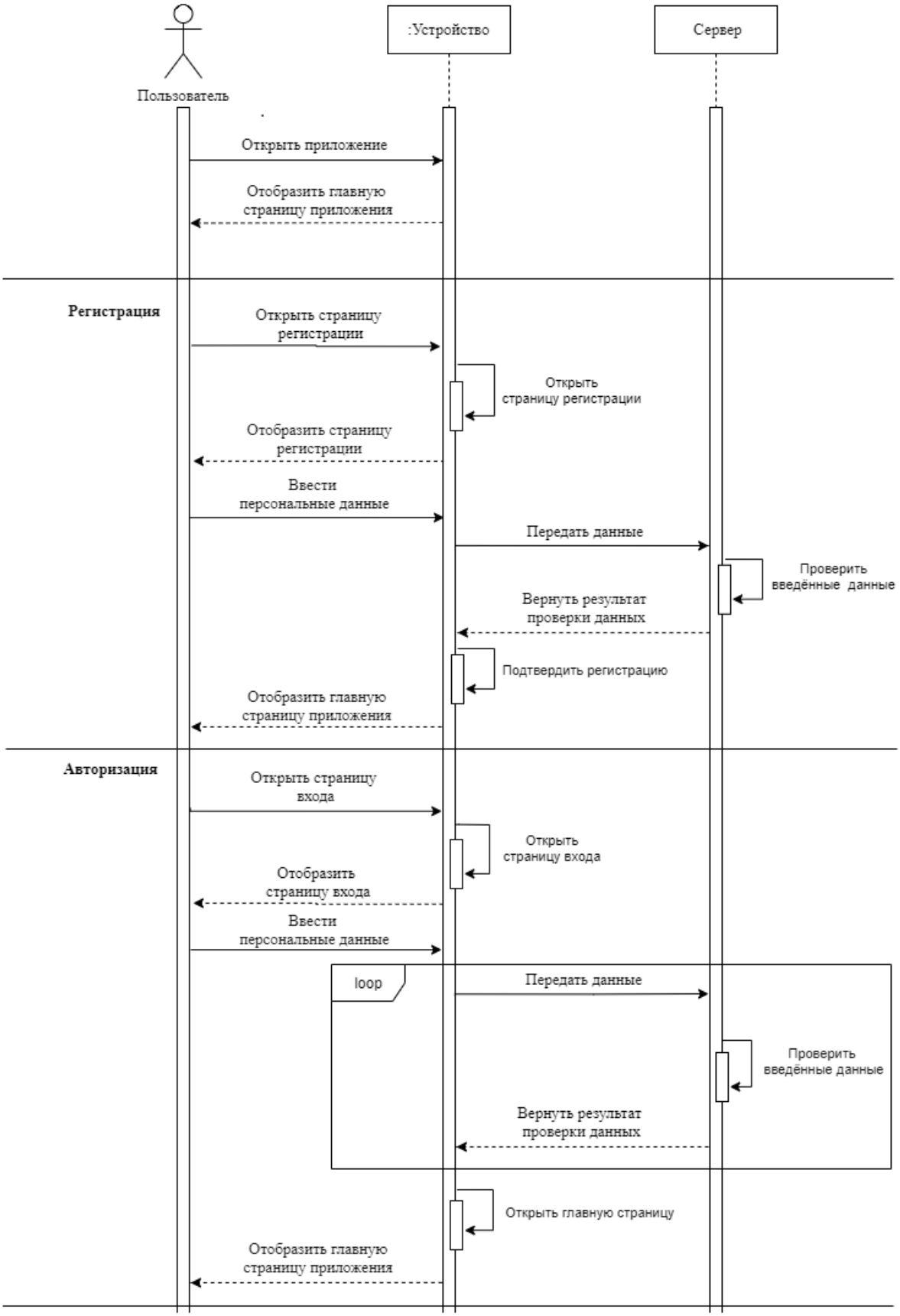

Рисунок 26 - Диаграмма последовательности (часть 1)

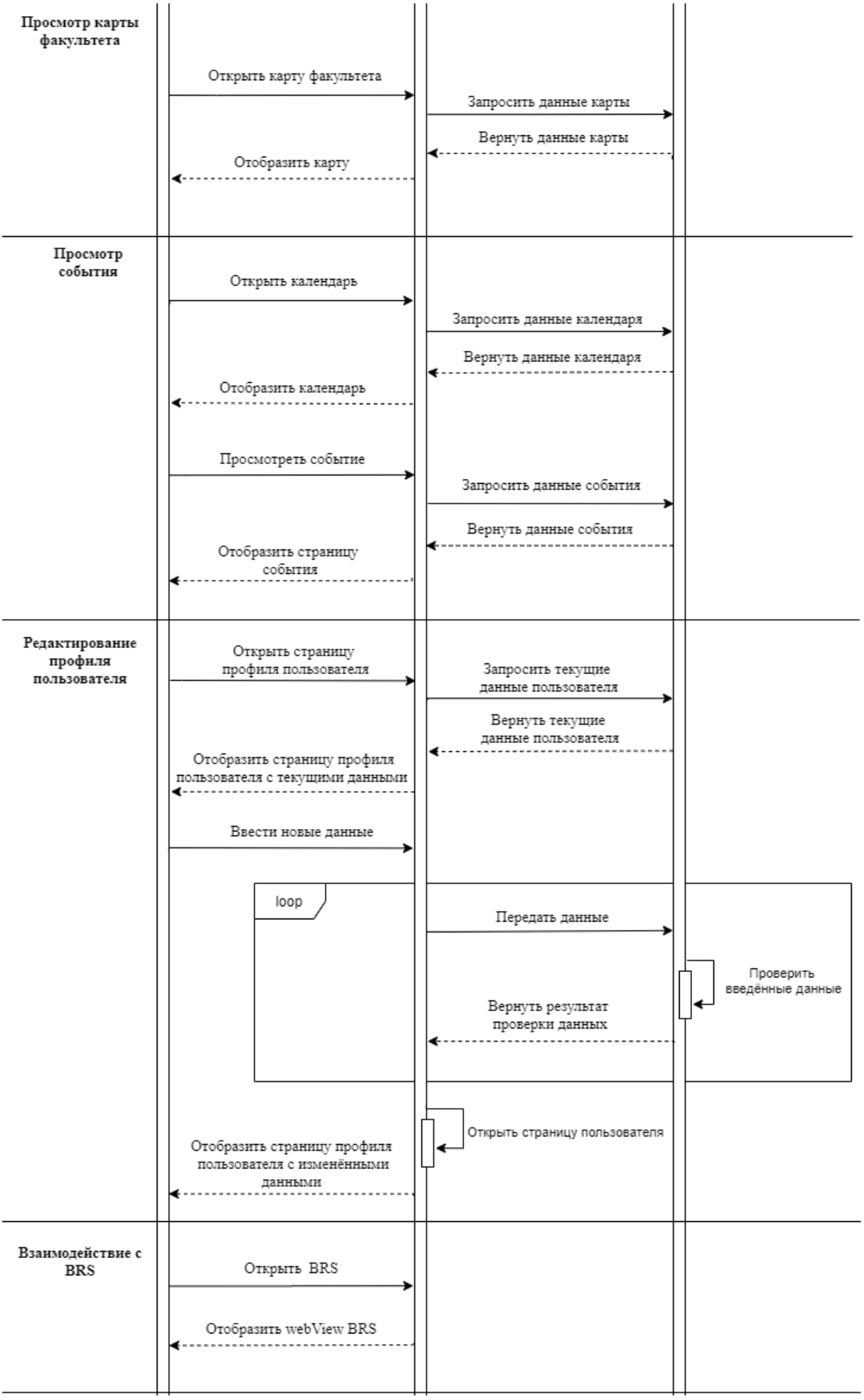

Рисунок 27 - Диаграмма последовательности (часть 2)

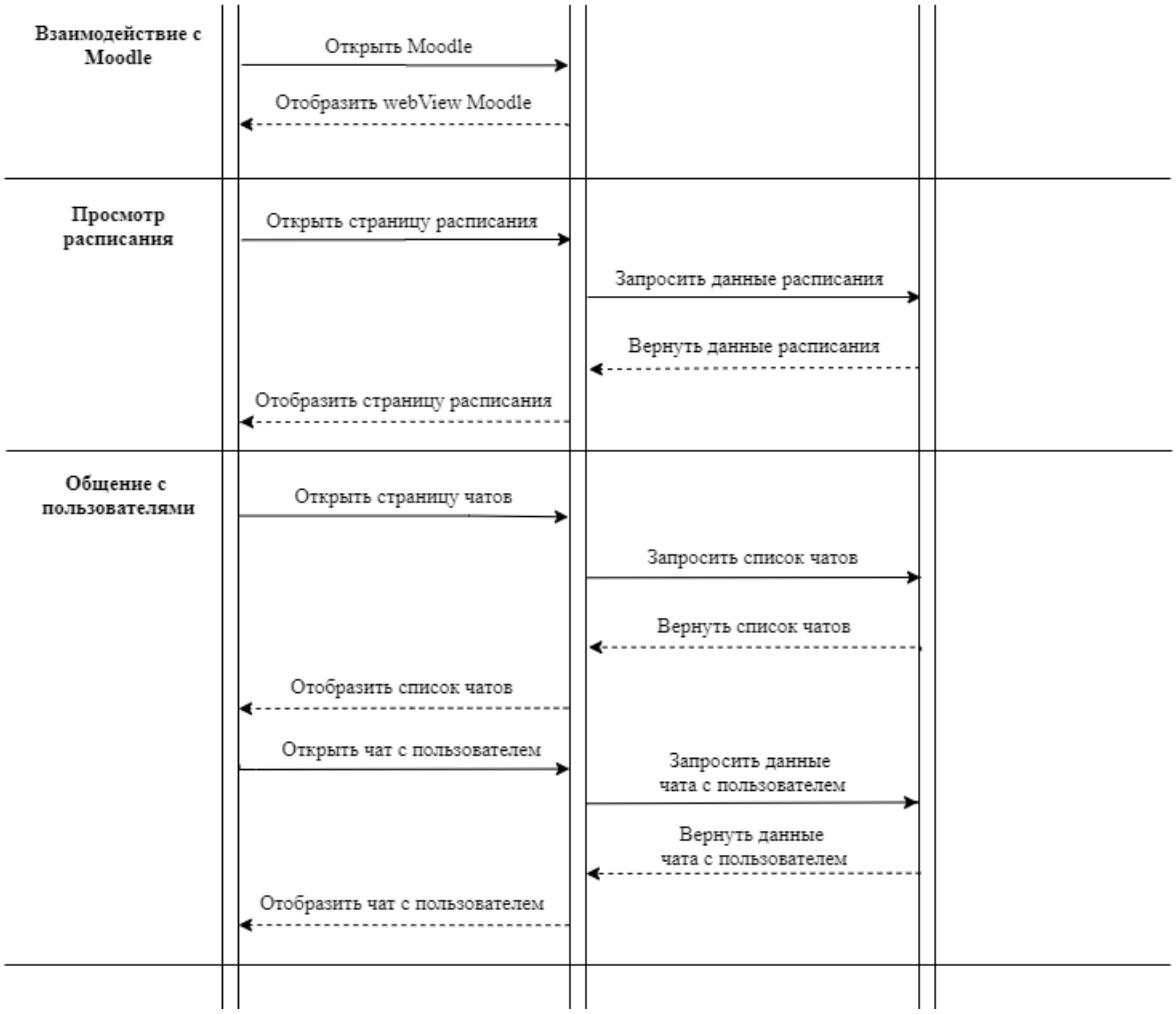

Рисунок 28 - Диаграмма последовательности (часть 3)

При запуске приложения открывается главная страница.

При регистрации происходит открытие окна регистрации, затем введённые данные передаются на сервер, после чего отображается ответ сервера.

При авторизации открывается страница ввода, после чего введённые в неё данные отправляются на сервер, который осуществляет их проверку. Если проверка успешно пройдена, то отображается страница приложения. А если данные оказались некорректными, то происходит повторная попытка ввода.

При запросе карты факультета происходит её отображение.

При просмотре события сначала запрашиваются данные календаря на сервере, после чего календарь отображается. Затем на сервере запрашиваются

данные конкретного события, которые отображаются после получения их с сервера.

При редактировании профиля пользователя текущие данные пользователя запрашиваются на сервере, после чего отображаются на странице профиля. Каждый раз новые данные отправляются на сервер, который, после их обновления, присылает соответствующий ответ. Затем отображается страница профиля с обновлёнными данными.

При взаимодействии с moodle и BRS система возвращает соответственные webView.

При просмотре расписания соответствующие данные запрашиваются на сервере, после чего они отображаются на странице расписания.

При осуществлении общения с пользователями на сервере запрашиваются данные о чатах, после чего они отображаются на соответствующей странице. Затем, при открытии конкретного чата, его данные также запрашиваются на сервере и затем отображаются на странице чата.

4.8 Диаграмма классов

Диаграмма классов используется для моделирования структуры классов и их отношений между собой в системе.

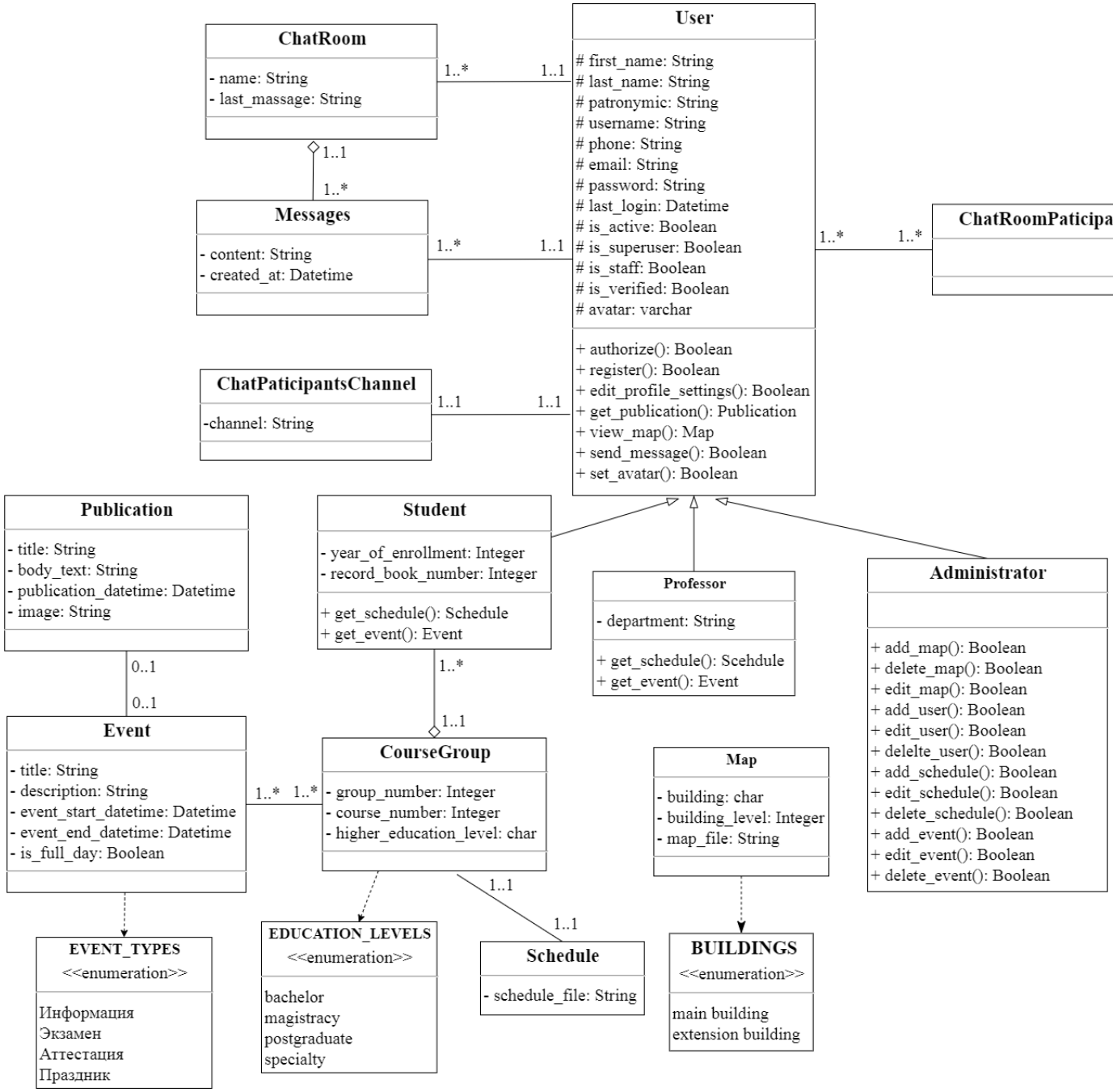

Рисунок 29 - Диаграмма классов

4.9 Диаграмма объектов

Диаграмма объектов используется для моделирования конкретных объектов и их отношений в системе. Она показывает экземпляры классов, их атрибуты и связи между ними.

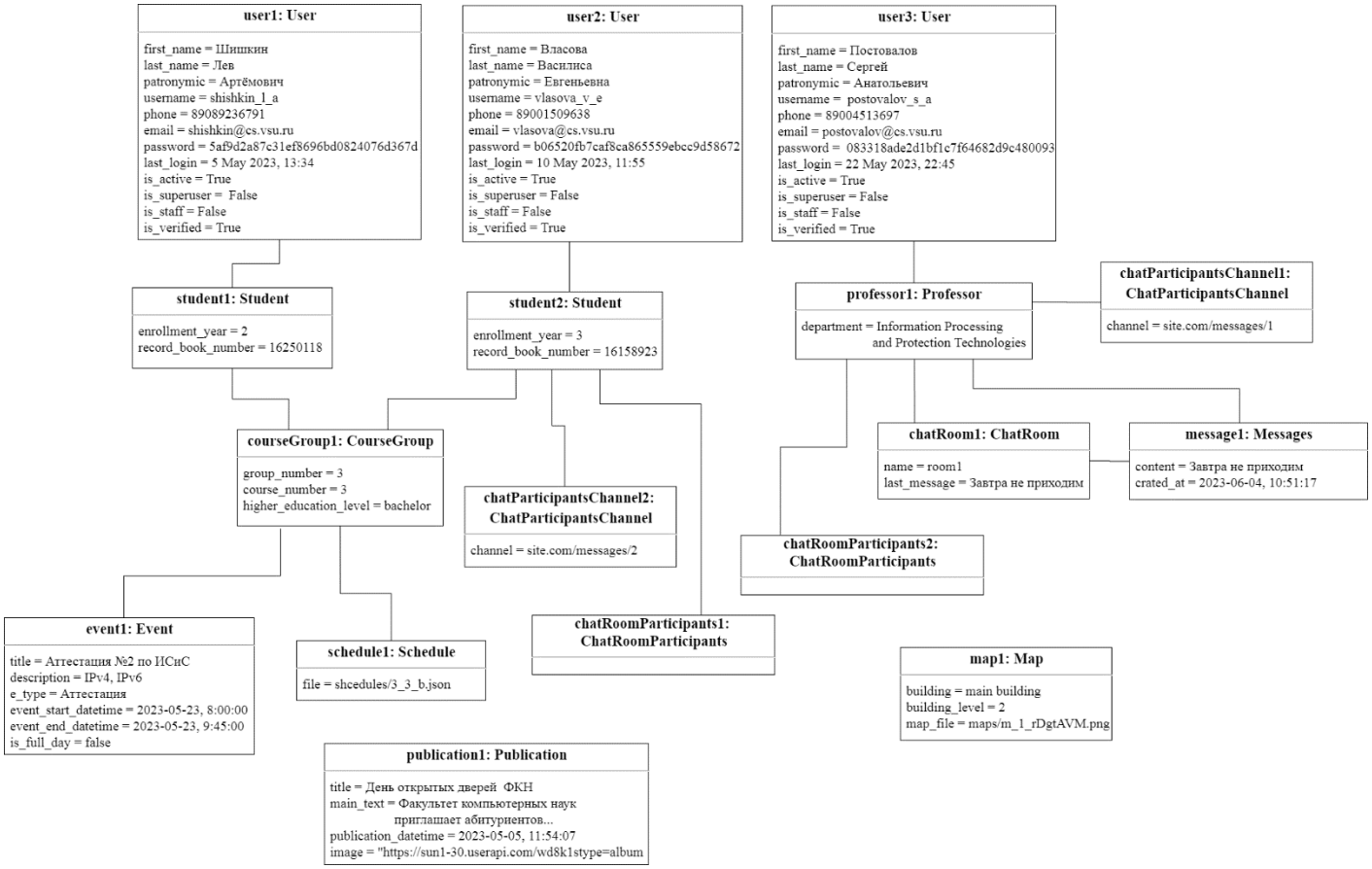

Рисунок 30 - Диаграмма объектов

5 Реализация

5.1 Средства реализации

Приложение должно соответствовать архитектуре клиент-сервер. Для реализации серверной части были выбраны следующие средства:

— ОС Windows 10 / Ubuntu 22.10;

— Фреймворк Django 4.2;

— ЯП Python 3;

— СУБД PostgreSQL 13;

— Система контроля версий Git 2.40.0.

Для реализации серверной части выбран фреймворк Django и язык Python, поскольку эта связка обладает рядом преимуществ:

— Встроенная панель администратора Django позволяет рационально решать задачи администратора, особенно при большом количестве данных.

— На языке Python написано большое количество разнообразных библиотек, предоставляющих при необходимости готовые решения.

— Наличие объектно-реляционного отображения, которое позволяет удобно взаимодействовать с базами данных [6][7].

В качестве средств реализации клиентской части были выбраны:

— Android Studio;

— Фреймворк Flutter 3.7.8;

— ЯП Dart 2.9.15;

— Система контроля версий Git 2.40.0.

Фреймворк Flutter и язык программирования Dart были выбраны для реализации клиентской части, потому что:

— Dart предоставляет возможность «горячей перезагрузки», то есть мгновенного применения изменений в коде без необходимости полного перезапуска приложения, что удобно при разработке;

— Движок Flutter написан на языке C++, поддерживает низкоуровневый рендеринг и обладает высокой производительностью (до 120 кадров в секунду).

— Flutter – кроссплатформенный фреймворк, что позволяет его использовать в разных ОС.

5.2 Реализация базы данных

Далее представлена ER-диаграмма используемой базы данных (рис. 31). ER (Entity-Relationship) диаграмма - это графическое представление структуры базы данных, которое используется для моделирования сущностей, их атрибутов и отношений между ними.

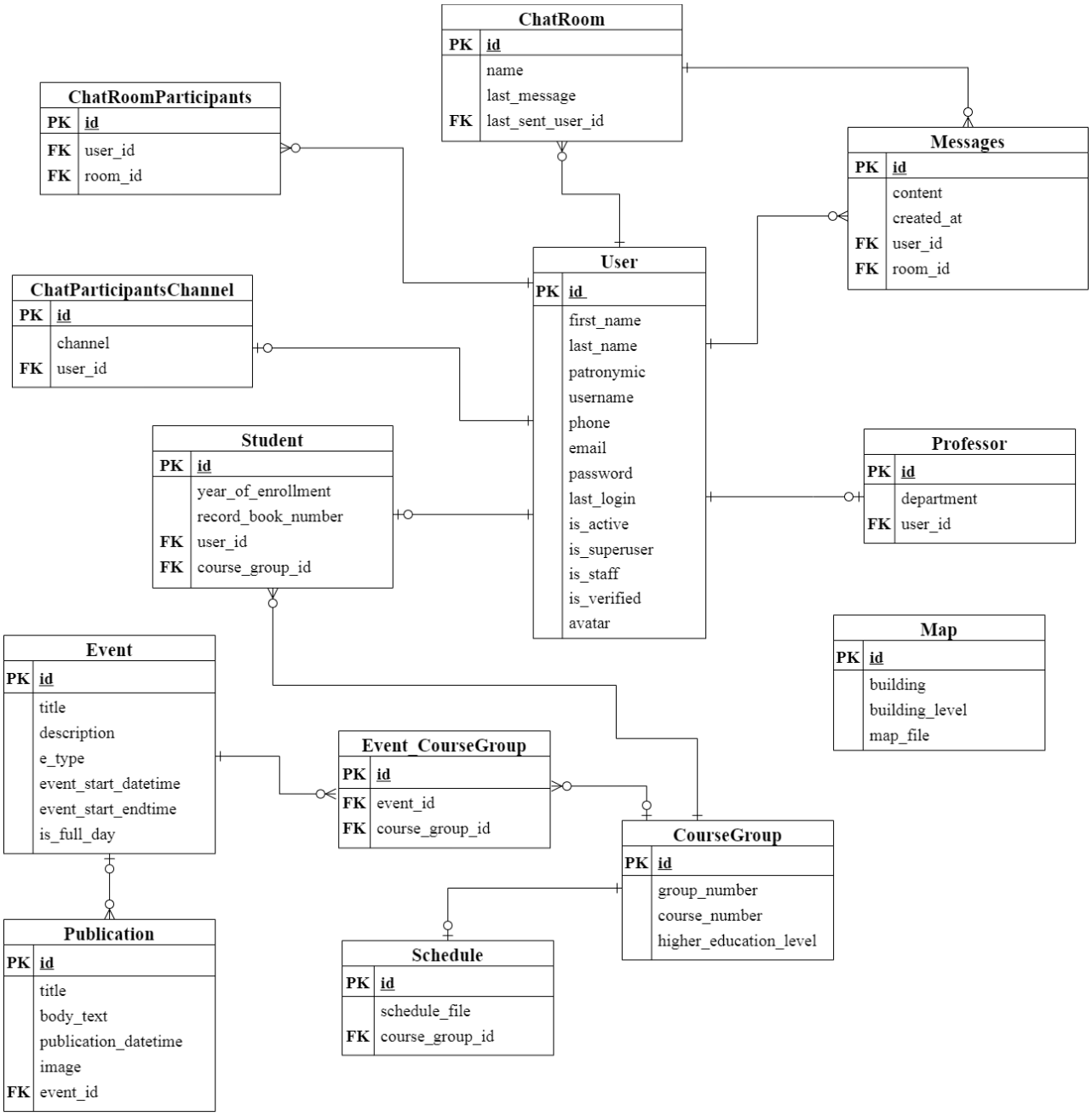

Рисунок 31 - ER-диаграмма

Как видно, база данных содержит большое количество сущностей. Далее будет приведена краткая информация о них и связях между ними.

User – это пользователь, который может быть либо студентом (Student), либо преподавателем (Professor). Данная сущность позволяет агрегировать студентов, преподавателей, администраторов и т.д. в единую таблицу. Поскольку каждый пользователь может являться студентом или преподавателем (а может и не являться), а каждый студент или преподаватель может быть только одним пользователем (и при этом они всегда являются пользователями), то сущность User связана с сущностями Student и Professor связями 1:0,1 (Запись “0,1” означает необязательный класс принадлежности. Далее будет использована аналогичная нотация. Если 0 нет, класс принадлежности полагается обязательным).

Message – это сообщения, отправленные пользователями. Каждый пользователь может отправить одни или несколько сообщений (а может не отправить ни одного), а каждое сообщение может быть отправлено одним и только одним пользователем. Поэтому сущности User и Message связаны отношением 1:0,N.

Event – событие, происходящее на факультете. Каждому событию соответствуют несколько публикаций (Publication) или не соответствует ни одной. В то же время каждая публикация может быть или не привязана к событию, или привязана, но только к одному (одна и та же публикация не может соответствовать нескольким разным событиям). Поэтому сущности Event и Publication имеют связь 0,1:0,N.

CourseGroup – это сущность, указывающая на курс и группу. Каждое событие может быть привязано (или не привязано) к одной или нескольким сущностям Event_CourseGroup. Таким образом, Event связано с Envent_CourseGroup связью 1:0,N, которая в свою очередь связана 0,N:0,1 с сущностью CourseGroup.

Schedule – это расписание. Каждое расписание привязано к определённому курсу и группе, но гипотетически расписание какой-то конкретной группы по различным причинам может ещё не существовать. Поэтому CourseGroup связано с Shedule связью 1:0,1.

Так же отдельно в базе данных хранится сущность Map – карта факультета. Она не связана никакими связями с другими сущностями.

На рисунке ниже представлена физическая модель базы данных (рис. 32).

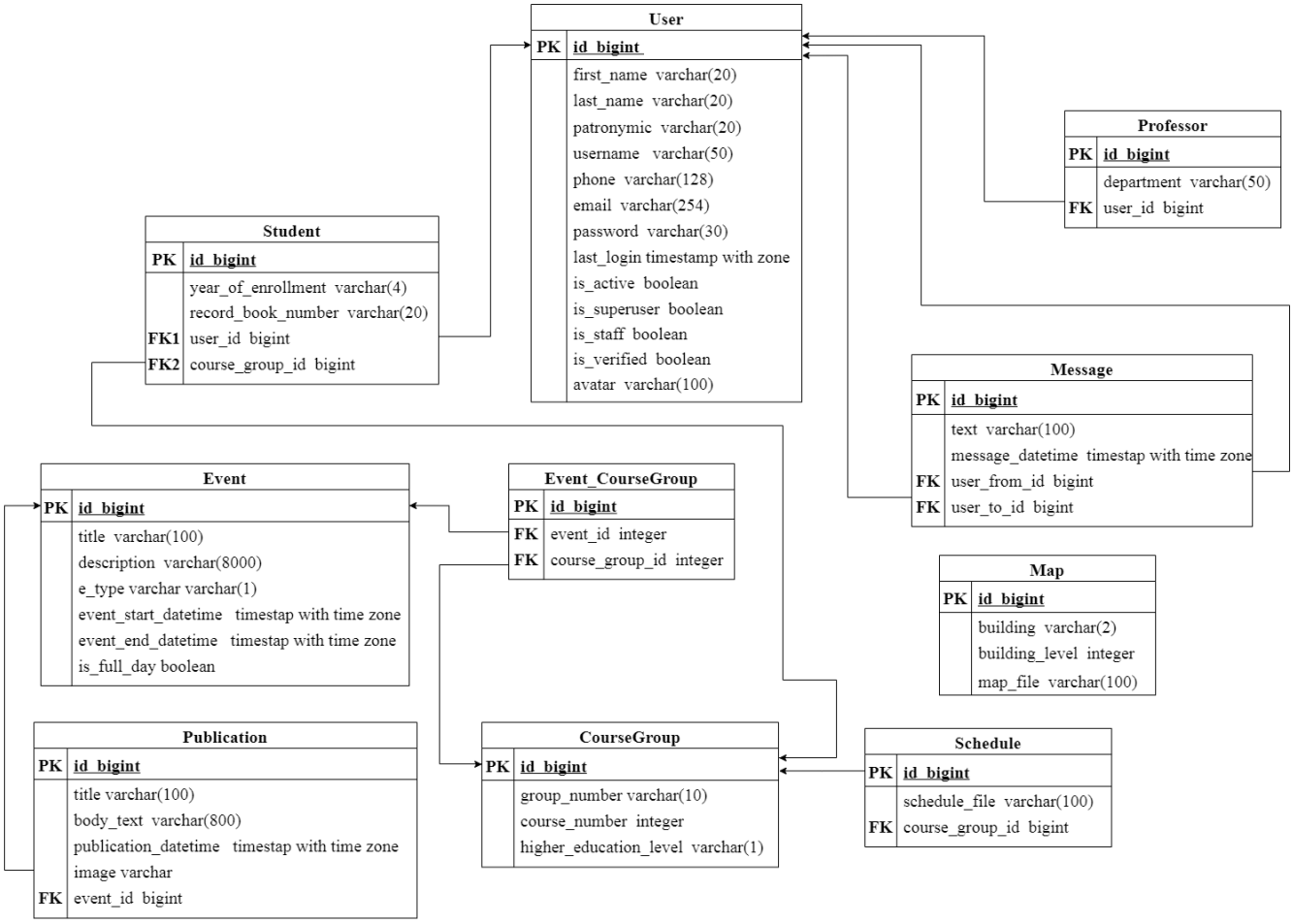

Рисунок 32 - Физическая модель базы данных

5.3 Реализация серверной части приложения

Серверная часть, помимо описанной выше базы данных, состоит из следующих компонентов:

— Файл Settings.py, содержащий настройки базы данных, настройки для подключения к ней, валидации полей, настройки JWT-токенов, секретное слово, список разрешённых хостов и т.д.

— Определены некоторые сервисы, в частности, NewsParser, schedule_utilities.

— Сериализаторы для подготовки данных к отправке и наоборот, для загрузки полученных данных на сервер.

— Модели данных, с которыми будет работать Django ORM. Также определены некоторые дополнительные поля (например, для генерации пути к изображению).

— Файл маршрутизации (urls.py) – определяет пути к конечным точкам API. Также определены контроллеры для конечных точек.

— Файл permissions.py – содержит классы, с помощью которых осуществляется разрешение или запрет на доступ к определённым данным.

5.4 Реализация клиентской части приложения

5.4.1 Общая информация

Клиентская часть приложения отвечает за пользовательский интерфейс и взаимодействие пользователя с данными.

Клиентская часть содержит:

— Файл home.dart – основной файл, определяющий навигацию и перемещение между экранами приложения.

— Каталог файлов views, содержащий файлы .view, хранящие в себе страницы приложения.

— Папка api, содержащая сервисы, контроллеры и модели.

— Библиотеки flutter_native_splash, view_flutter, file_picker.

5.4.2 Графический интерфейс

Далее будут представлены экраны приложения и приведена информация о них.

Приветственная страница содержит логотип приложения.

Рисунок 33 - Приветственная страница

При входе в приложение пользователю предлагается зайти или зарегистрироваться. Для этого служат экраны регистрации и авторизации.

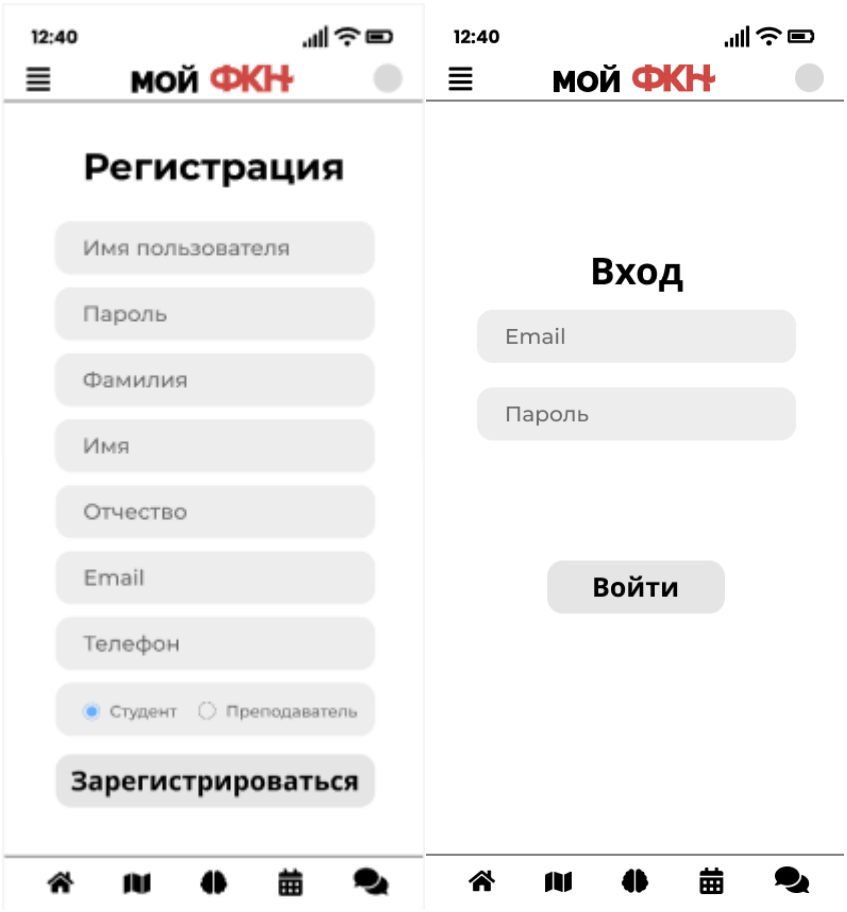

Рисунок 34 - Экраны регистрации и авторизации

Далее идут страницы онбординга. На них содержатся подсказки о использовании приложения. Однако они появляются лишь при первом запуске приложения.

Рисунок 35 - Экраны онбординга (1)

Рисунок 36 - Экраны онбординга (2)

Приложение открывается на главной странице. Она содержит новости факультета, а в её нижней части располагаются кнопки для навигации по приложению.

Рисунок 37 - Главная страница

Свайпом слева направо пользователь может открыть боковое меню. В зависимости от того, авторизован пользователь или нет, его вид бдет намного различаться. Если пользователь авторизован, то в верхней части бокового меню будет отображаться его аватар, фамилия и инициалы, а также роль.

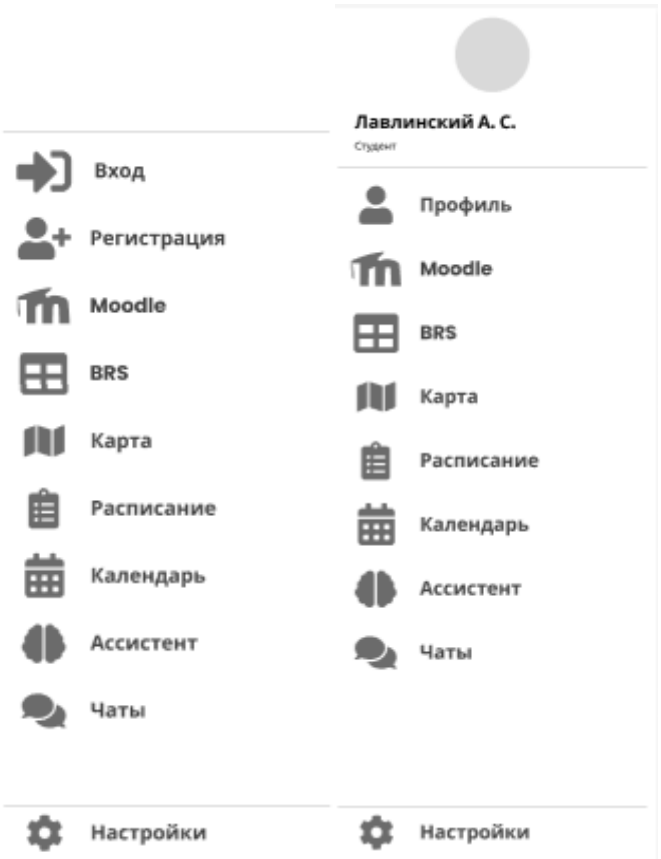

Рисунок 38 - Боковое меню

Пользователь имеет возможность отредактировать данные профиля. Для этого служит страница редактирования профиля, содержащая соответствующие поля ввода.

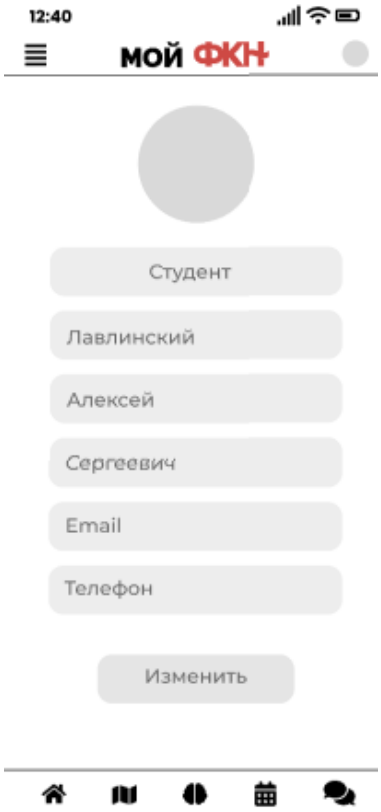

Рисунок 39 - Страница редактирования профиля

Приложение осуществляет интеграцию с сервисами moodle и BRS. Для просмотра данных из них существуют специальные страницы.

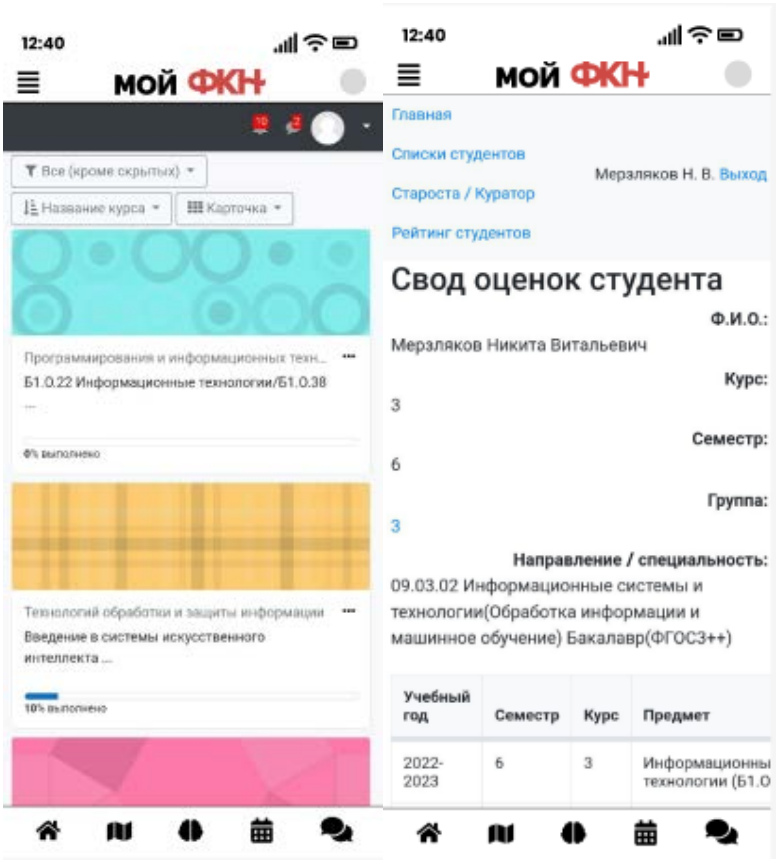

Рисунок 40 - Страницы интеграции с moodle и BRS

Приложение имеет карту факультета, страницу которой можно открыть, нажав соответствующую кнопку в боковом меню или в меню навигации в нижней части экрана. Для переключения этажей на карте служит кнопка слоёв, расположенная в правом нижнем углу карты.

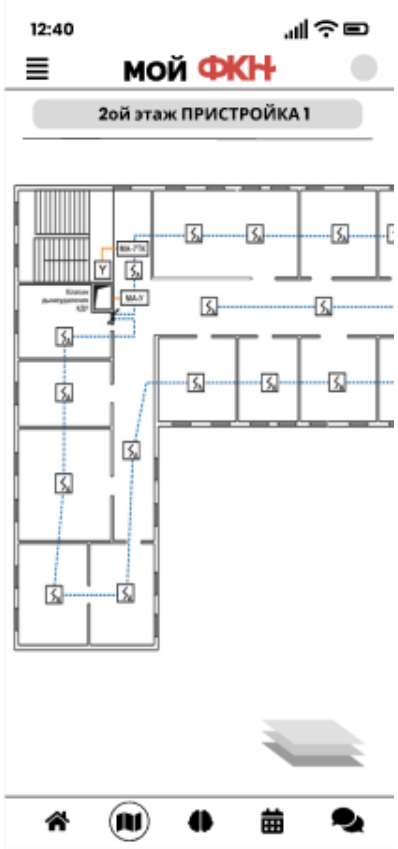

Рисунок 41 - Карта факультета

Для реализации чатов существует специальная страница чатов. Для того, чтобы ориентироваться в чатах и осуществлять переход между ними, используется соответствующая страница выбора чатов, доступ к которой осуществляется из бокового меню или меню выбора чатов.

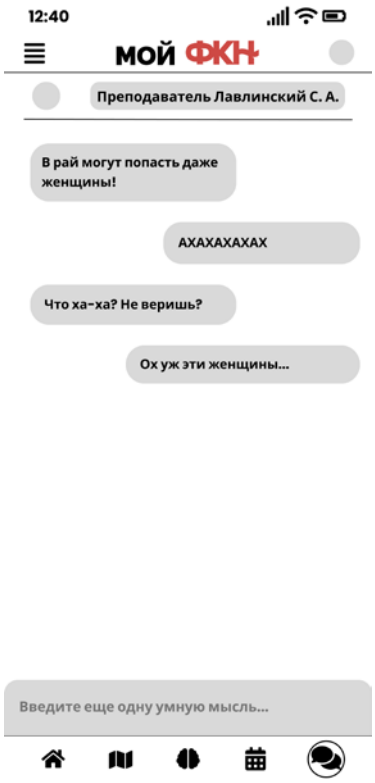

Рисунок 42 - Страница чатов.

Рисунок 43 - Страница списка чатов.

Также существует отдельный доступ к чату с чат-ботом.

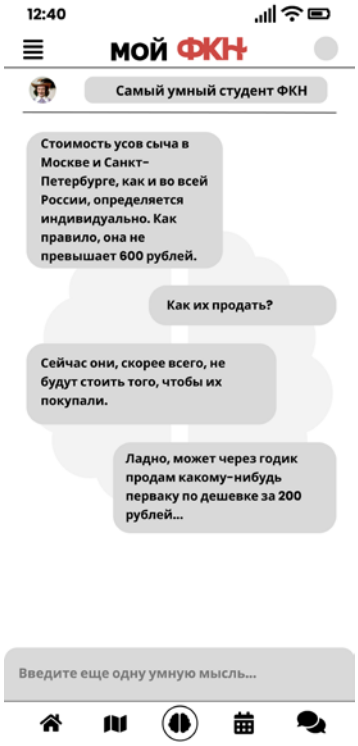

Рисунок 44 - Страница чата с чат-ботом.

Для осуществления интеграции с moodle и BRS необходимо ввести соответствующие данные для входа. Для этого служит соответствующая страница.

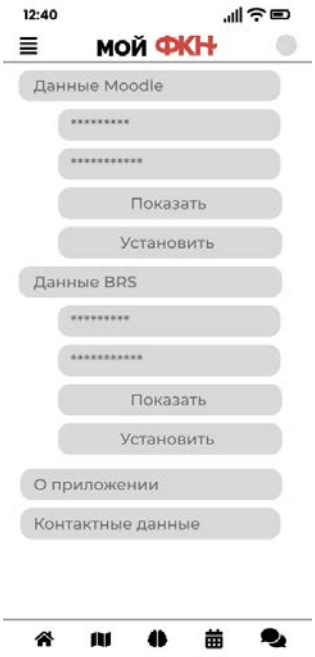

Рисунок 45 - Страница ввода данных для moodle и BRS.

Существуют окна информации о приложении, служащие для просмотра информации о приложении, его текущей версии и команды разработчиков.

Рисунок 46 - Страницы информации о приложении

Если на главной странице нажать на событие, то откроется окно этого приложения.

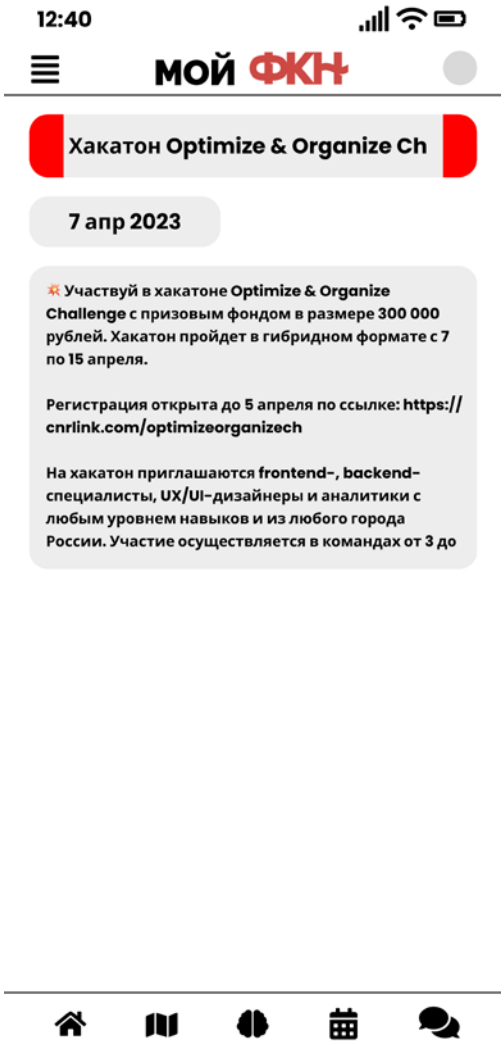

Рисунок 47 - Страница события

5.5 Методология разработки

При реализации проекта была использована методология Whaterfall (каскадная модель).

Следуя каскадной модели, разработчик переходит от одной стадии к другой строго последовательно. Сначала полностью завершается этап «определение требований», в результате чего получается список требований к ПО. После того как требования полностью определены, происходит переход к проектированию, в ходе которого создаются документы, подробно описывающие для программистов способ и план реализации указанных требований. После того как проектирование полностью выполнено,

программистами выполняется реализация полученного проекта. На следующей стадии процесса происходит интеграция отдельных компонентов, разрабатываемых различными командами программистов. После того как реализация и интеграция завершены, производится тестирование и отладка продукта; на этой стадии устраняются все недочёты, появившиеся на предыдущих стадиях разработки. После этого программный продукт внедряется и обеспечивается его поддержка — внесение новой функциональности и устранение ошибок [7] (рис. 48).

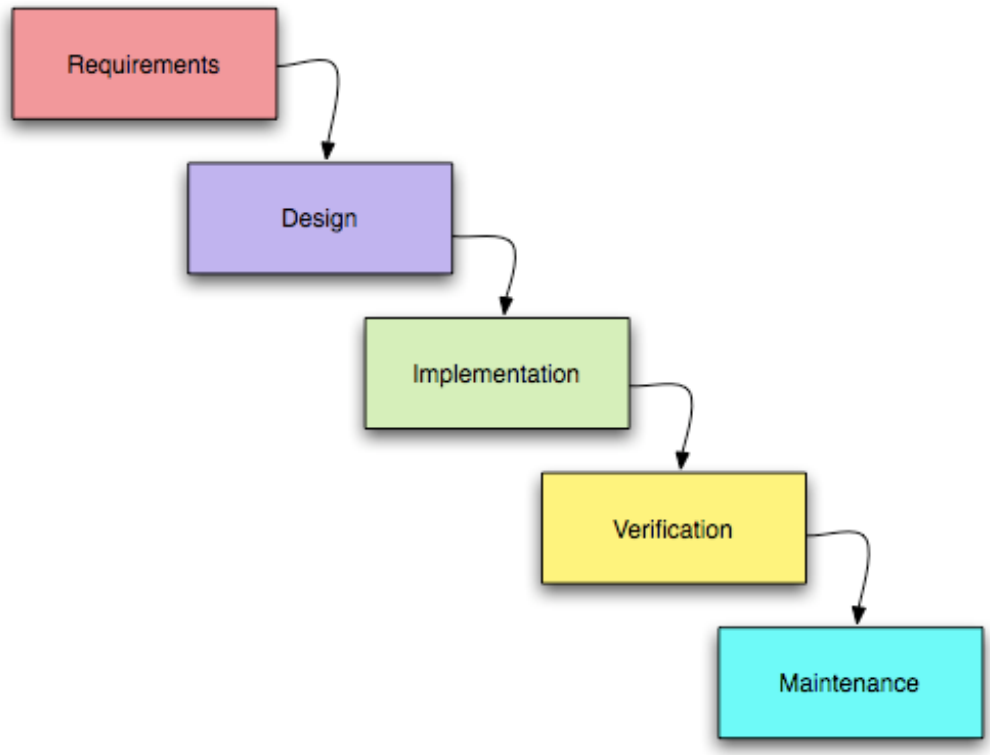

Рисунок 48 - Процесс разработки по методологии Whaterfall.

Данная методология была выбрана, поскольку она хорошо подходит для проектов, в которых требования и границы прозрачны и известны в начале разработки.

6 Тестирование

6.1 Дымовое тестирование

Для данного тестирования необходимо проверить работоспособность на следующие основные сценарии:

— Регистрация

— Авторизация

— Открытие бокового меню

— Просмотр карты факультета

— Открытие страницы Moodle

— Открытие страницы BRS

Далее в таблице представлены результаты тестирования. Тестирование производилось на Samsung Galaxy S20 (Android 11) и IPhone 12.

Таблица 2 - Результаты дымового тестирования

Сценарий	**Результат**
Регистрация	Пройден
Авторизация	Пройден
Открытие бокового меню	Пройден
Открытие карты факультета	Пройден
Открытие страницы Moodle	Пройден
Открытие страницы BRS	Пройден

Как можно видеть, приложение прошло тесты на все основные сценарии.

6.2 UI-тестирование

Далее в таблице приведены результаты UI-тестирования.

Таблица 3 – результаты UI-тестирования.

Шаги теста	Ожидаемый ответ	Результат
Нажатие на кнопку «войти» при корректных входных данных.	Открытие главной страницы	Пройден
Нажатие на кнопку «войти» при некорректных входных данных	Вход не удался	Пройден
Нажатие на кнопку регистрации при корректно введённых данных	Регистрация удалась, происходит открытие главной страницы	Пройден
Нажатие на кнопку регистрации при некорректно введённых входных данных	Регистрация не удалась, снова открыто окно регистрации	Пройден
Открытие бокового меню: пользователь не зарегистрирован	Откроется боковое меню незарегистрированного пользователя	Пройден
Открытие бокового меню: пользователь зарегистрирован	Откроется боковое меню зарегистрированного пользователя	Пройден
Открытие страницы Moodle: пользователь не ввёл данные для Moodle	Пользователю предложат ввести данные для Moodle	Пройден
Открытие страницы Moodle: пользователь уже вводил данные для Moodle	Страница Moodle открывается успешно	Пройден
Открытие страницы BRS: пользователь не ввёл данные для BRS	Пользователю предложат ввести данные для BRS	Пройден
Открытие страницы BRS:	Страница BRS открывается успешно	Пройден

пользователь уже вводил данные для BRS		
Нажатие на кнопку «главная страница»	Открывается главная страница	Пройден
Нажатие на кнопку «карта»	Открывается карта	Пройден
Нажатие на кнопку чат-бота»	Открывается страница чат-бота	Пройден
Нажатие на кнопку чатов	Открывается страница чатов	Пройден
Нажатие на кнопку чатов, нажатие на конкретный чат, набор сообщения, нажатие на кнопку отправку.	Сообщение будет отправлено получателю	Пройден

В итоге было получено заключение, что все сценарии UI-тестирования были пройдены успешно.

6.3 Юзабилити-тесты

Для проведения юзабилити-тестов было выбрано 4 человека, ранее не пользовавшихся приложением. В ходе тестирования была выполнена проверка некоторых сценариев взаимодействия с приложением.

Таблица 4 – результаты юзабилити-тестирования.

Сценарий	Пользователь 1	Пользователь 2	Пользователь 3	Пользователь 4
Регистрация	Пройден	Пройден	Пройден	Пройден
Авторизация	Пройден	Пройден	Пройден	Пройден
Открытие бокового меню	Пройден	Пройден	Пройден	Пройден
Открытие карты факультета	Пройден	Пройден	Пройден	Пройден

Открытие страницы Moodle	Пройден	Пройден	Пройден	Пройден
Открытие страницы BRS	Пройден	Пройден	Пройден	Пройден

В результате проведения юзабилити-тестирования было установлено, что приложение прошло успешно все проводимые тесты.

Заключение

В ходе реализации данного проекта были достигнуты поставленные задачи. Разработанное приложение выполняет поставленные задачи, а именно:

— Позволяет выполнять регистрацию и вход;

— Реализует различные роли для студента и преподавателя;

— Содержит панель администратора;

— Реализует интеграцию с Moodle;

— Реализует интеграцию с BRS;

— Позволяет просматривать карту факультета;

— Позволяет просматривать ленту событий;

— Позволяет хранить, просматривать и изменять данные профиля;

— Поддерживает систему чатов;

— Содержит встроенного чат-бота.

В результате проведённого тестирования было установлено, что готовое приложение успешно прошло все выполненные тесты на различные сценарии использования, тем самым подтвердив свою работоспособность.

Список использованных источников

1) Чем недовольны российские студенты [Электронный ресурс]. – Режим доступа: https://student-app.ru/blog9. – Заглавие с экрана. – (Дата обращения: 16.03.2023).

2) Официальный сайт FlipTable [Электронный ресурс] – Режим доступа: https://fliptable.ru/ . – (Дата обращения: 20.03.2023).

3) Официальный сайт BlackBoard [Электронный ресурс] – Режим доступа: https://www.blackboard.com/ . – (Дата обращения: 20.03.2023).

4) Университет в кармане [Электронный ресурс] – Режим доступа: http://moyuniver.ru/ . – Заглавие с экрана. – (Дата обращения: 20.03.2023).

5) What does «Django» mean, and how do you pronounce it? Дата обращения: 14 мая 2009. Архивировано 10 сентября 2017 года.

6) Официальный сайт Python [Электронный ресурс] – Режим доступа: https://www.python.org/ . – (Дата обращения: 20.04.2023).

7) Royce, Winston (1970), Managing the Development of Large Software Systems.