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

Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.01 Информатика и вычислительная техника
Отделение школы (НОЦ) информационных технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Информационная система уведомления обучающихся в ТПУ на платформе Telegram

УДК 004.451:004.455.1:004.65

Студент

Группа	ФИО	Подпись	Дата
8ВМ81	Бокор Владимир Алексеевич		

Руководитель ВКР

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Фадеев Александр Сергеевич	к.т.н.		

КОНСУЛЬТАНТЫ ПО РАЗДЕЛАМ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Конотопский В.Ю.	к.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Горбенко М.В.	к.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

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

Томск – 2020 г.

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ ПО ООП

Код результата	Результат обучения (выпускник должен быть готов)
	Общепрофессиональные компетенции
Р1	Воспринимать и самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте.
Р2	Владеть и применять методы и средства получения, хранения, переработки и трансляции информации посредством современных компьютерных технологий, в том числе в глобальных компьютерных сетях.
Р3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений и высказываний, основанных на интерпретации данных, интегрированных из разных областей науки и техники, выносить суждения на основании неполных данных, анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями.
Р4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью и готовностью к саморегулированию дальнейшего образования и профессиональной мобильности. Владеть, по крайней мере, одним из иностранных языков на уровне социального и профессионального общения, применять специальную лексику и профессиональную терминологию языка.
	Профессиональные компетенции
Р5	Выполнять инновационные инженерные проекты по разработке аппаратных и программных средств автоматизированных систем различного назначения с использованием современных методов проектирования, систем автоматизированного проектирования, передового опыта разработки конкурентно способных изделий.
Р6	Планировать и проводить теоретические и экспериментальные исследования в области проектирования аппаратных и программных средств автоматизированных систем с использованием новейших достижений науки и техники, передового отечественного и зарубежного опыта. Критически оценивать полученные данные и делать выводы.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и эксплуатации аппаратных и программных средств автоматизированных систем различного назначения.
	Общекультурные компетенции
Р8	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.
Р9	Осуществлять коммуникации в профессиональной среде и в обществе в целом, активно владеть иностранным языком, разрабатывать документацию, презентовать и защищать результаты инновационной инженерной деятельности, в том числе на иностранном языке.
Р10	Совершенствовать и развивать свой интеллектуальный и общекультурный уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.
Р11	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.

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

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

УТВЕРЖДАЮ:
Руководитель ООП

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

магистерской диссертации

(бакалаврской работы, дипломного проекта/работы, магистерской диссертации)

Студенту:

Группа	ФИО
8ВМ81	Бокору Владимиру Алексеевичу

Тема работы:

Информационная система уведомления обучающихся в ТПУ на платформе Telegram	
Утверждена приказом директора (дата, номер)	№ 59-63/с от 28.02.2020 г.

Срок сдачи студентом выполненной работы:	01.06.2020 г.

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Объектом проектирования является информационная система уведомления обучающихся. Режим работы – непрерывный. Система должна предоставлять веб-интерфейс для работы администратора.
Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	1. Аналитический обзор для выбора платформы для информационной системы 2. Постановка задачи проектирования 3. Разработка информационной системы 4. Публикация программного комплекса 5. Заключение по работе
Перечень графического материала *(с точным указанием обязательных чертежей)*	1. Скрипт получения расписания на неделю 2. Диаграмма моделей классов серверной части

	3. Класс очереди сообщений в сервере 4. Класс BaseHandler 5. Dockerfile для сервера

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Конотопский Владимир Юрьевич, к.э.н., доцент отделения социально-гуманитарных наук
Социальная ответственность	Горбенко Михаил Владимирович, к.т.н, доцент отделения общетехнических дисциплин
Обязательное приложение на английском языке	Татьяна Валерьевна Сидоренко, к.п.н., доцент отделения иностранных языков

Названия разделов, которые должны быть написаны на русском и иностранном языках:

1 Выбор платформы для информационной системы
2 Проектирование информационной системы
3 Программная реализация
4 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение
5 Социальная ответственность
6 Information System Design

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	10.03.2020

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Фадеев Александр Сергеевич	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ81	Бокор Владимир Алексеевич		

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

Школа **Информационных технологий и робототехники**
Направление подготовки **09.04.01 Информатика и вычислительная техника**
Уровень образования **Магистратура**
Отделение школы (НОЦ) **Информационных технологий**
Период выполнения **(осенний/весенний семестр 2019/2020 учебного года)**

Форма представления работы:

магистерская диссертация

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

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	01.06.2020

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
18.04.2020	Проектирование информационной системы	20
30.05.2020	Программная реализация	45
30.05.2020	Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	15
30.05.2020	Социальная ответственность	10
30.05.2020	Information System Design	10
Итого:		100

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Фадеев А.С	к.т.н.		

СОГЛАСОВАНО:
Руководитель ООП

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Кочегурова Е.А	к.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ВМ81	Бокору Владимиру Алексеевичу

Школа	ИШИТР	**Отделение школы (НОЦ)**	Отделение информационных технологий
Уровень образования	Магистратура	**Направление/специальность**	09.04.01 Информатика и вычислительная техника

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих*	Использовать действующие ценники и договорные цены на потребленные материальные и информационные ресурсы, а также указанную в МУ величину тарифа на эл. энергию
2. *Нормы и нормативы расходования ресурсов*	—
3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	Действующие ставки единого социального налога и НДС, ставка дисконтирования = 0,1 (см. МУ)
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого и инновационного потенциала НТИ*	Дать характеристику существующих и потенциальных потребителей (покупателей) результатов ВКР, ожидаемых масштабов их использования
2. *Разработка устава научно-технического проекта*	Разработать проект такого устава в случае, если для реализации результатов ВКР необходимо создание отдельной организации или отдельного структурного подразделения внутри существующей организации
3. *Планирование процесса управления НТИ: структура и график проведения, бюджет, риски и организация закупок*	Построение плана-графика выполнения ВКР, составление соответствующей сметы затрат, расчет цены результата ВКР.
4. *Определение ресурсной, финансовой, экономической эффективности*	Оценка экономической эффективности использования результатов ВКР, характеристика других видов эффекта
Перечень графического материала *(с точным указанием обязательных чертежей):*	
1. *«Портрет» потребителя результатов НТИ* 2. *Сегментирование рынка* 3. *Оценка конкурентоспособности технических решений* 4. *Диаграмма FAST* 5. *Матрица SWOT* 6. *График проведения и бюджет НТИ* 7. *Оценка ресурсной, финансовой и экономической эффективности НТИ* 8. *Потенциальные риски*	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Конотопский Владимир Юрьевич	к. э. н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ81	Бокор Владимир Алексеевич		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8ВМ81	Бокору В.А.

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

Тема ВКР:

Информационная система уведомления обучающихся в ТПУ на платформе Telegram	
Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Рабочее место расположено в офисе работодателя ООО «Хэнд Мэйд Код». В офисе рабочей зоной является место за персональным компьютером. Целью работы является разработка информационной системы уведомления обучающихся в ТПУ. Основным оборудованием, на котором производится работа, является персональный компьютер с периферийными устройствами.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Правовые и организационные вопросы обеспечения безопасности: – специальные (характерные при эксплуатации объекта исследования, проектируемой рабочей зоны) правовые нормы трудового законодательства; – организационные мероприятия при компоновке рабочей зоны.	Взаимоотношения с работодателем регулируются трудовым кодексом Российской Федерации от 30.12.2001 N 197-ФЗ Рабочее место при выполнении работ в положении сидя должно соответствовать требованиям ГОСТ 12.2.032-78. Требования к организации оборудования рабочих мест с ПК регулируется в СанПиН 2.2.2/2.4.1340 – 03.
2. Производственная безопасность: 2.1. Анализ выявленных вредных и опасных факторов 2.2. Обоснование мероприятий по снижению воздействия	Анализ выявленных вредных факторов: • недостаточная освещённость рабочей зоны: отсутствие или недостаток естественного света; • отклонение показателей микроклимата; • повышенный уровень шума. Анализ выявленных опасных факторов: • электрический ток
3. Экологическая безопасность:	Воздействие на атмосферу и гидросферу оказывается только при утилизации использованных приборов
4. Безопасность в чрезвычайных ситуациях:	В офисном помещении возможно ЧС техногенного характера – пожар (возгорание).

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Горбенко Михаил Владимирович	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ВМ81	Бокор Владимир Алексеевич		

РЕФЕРАТ

Выпускная квалификационная работа содержит: 108 страниц, 18 рисунков, 11 таблиц, 89 источников, 9 приложений.

Ключевые слова: веб-сервис, чат-бот, проектирование, Telegram, ТПУ, Django.

Объектом исследование является система информирования.

Цель работы – проектирование и разработка информационной системы уведомления обучающихся в ТПУ на основе платформы Telegram**.**

В процессе исследования проводились анализ мессенджеров, анализ архитектур и инструментов для реализации информационной системы.

В результате исследования был спроектирована и разработана информационная система, состоящая из серверной части и клиентской в виде приложения телеграм-бота.

Степень внедрения: информационная система размещена на сервере в Интернете и находится на этапе тестирования.

Область применения: использование программного обеспечения позволит упростить работу студентов и сотрудников ТПУ при взаимодействии с различными сервисами (расписание, новостной блок, информация о сотрудниках).

В будущем планируется дальнейшая разработка и совершенствование информационной системы в целях расширения функционала.

ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ

API – Application Programming Interface (программный интерфейс приложения)

CSRF – Cross-Site Request Forgery (межсайтовая подделка запроса)

DRF – Django Rest Framework

GIL – Global Interpreter Lock (глобальная блокировка интерпретатора)

GUI – Graphical User Interface (графический интерфейс пользователя)

JSON – JavaScript object notation (объект в нотации JavaScript)

JVM – Java Virtual Machine (виртуальная машина Java)

ORM – Object-Relational Mapping (объектно-реляционное отображение)

PHP – Personal Home Page

POJO – Plain Old Java Object (старый добрый Java-объект)

Pub – Publisher (издатель)

REST – Representational State Transfer (передача состояния представления)

RoR – Ruby on Rails

SQS – Simple Queue Service (простая служба очереди)

STL – Standard Template Library (стандартная библиотека шаблонов)

SNS – Simple Notification Service (простая служба уведомлений)

Sub – Subscriber (подписчик)

UML – Unified Modeling Language (унифицированный язык моделирования)

XML – eXtensible Markup Language (расширяемый язык разметки)

XSS – Cross-Site Scripting (межсайтовый скриптинг)

YAML – Yet Another Markup Language (Ещё один язык разметки)

РАЭК – Российская ассоциация электронных коммуникаций

Рунет – Русскоязычный Интернет

ООП – Объектно-ориентированное программирование

СУБД – Система Управления Базой Данных

Оглавление

Введение

В настоящее время информационные технологии проникают во все сферы деятельности, и получение образования не исключение. Глобальная тенденция «информатизации жизни» и образования отмечена во всех ключевых документах социально-экономического развития России [1]. Одним из пунктов в формировании информационного пространства знаний отмечена необходимость использования и развития различных образовательных технологий, в том числе дистанционное и электронное обучение [2]. Кроме того, одной из основных задач применения информационных и коммуникационных технологий для развития социальной сферы ставится создание технологических платформ для дистанционного обучения с целью повышения доступности качественных образовательных услуг [2].

Соответственно, ближайшее десятилетие должно стать эпохой значительных перемен в высшем образовании – развитие цифровой экономики и реорганизации образовательного процесса [3]. Учитывая специфику молодого поколения, университеты все более активно используют в коммуникациях цифровые каналы.

Создание мощных информационных систем для учебных заведений, способных удовлетворить информационные потребности любых пользователей, вовлеченных в данный процесс, является задачей дорогостоящей и трудоемкой [4]. Такая система должна решать в том числе и проблему своевременного оповещения учащихся о каких-либо событиях или информирования о временном изменении расписания занятий.

Данный вопрос разрабатывается в разных учреждениях уже несколько лет. Так, например, участникам хакатона EdHack: ChatBots было предложено решить проблемы современного онлайн образования и использовать чат-боты для повышения эффективности обучения и качества взаимодействия с учебными материалами [5]. В Государственном университете управления был разработан чат-бот с новостями и расписанием в социальной сети «ВКонтакте» [6]. В

университете ИТМО был разработан бот, который работает в связке с онлайн–платформой Центра дистанционного обучения (ЦДО) вуза, и сейчас он рассылает пользователям информацию о начисленных баллах за выполненные учебные задачи, а также расписании [7].

Чат-бот – это программный инструмент, который взаимодействует с пользователями на определенную тему или в определенной области естественным, диалоговым способом, используя текст и голос.

Чат-боты как приложения существуют уже долгое время. Например, в 1996 году Брайан Маклафлин разработал чат-бота с названием Claude, который использует стандартное сопоставление с образцом, чтобы найти подходящий ответ [8]. Claude распознает данные, вводимые пользователем, затем создает ответ на основе этого ввода, используя ответы в своей базе данных.

Чат-боты также использовались и в образовании с начала 1970х, они известны как интеллектуальные системы репетиторства [9]. В последние годы все больше и больше организаций стараются развивать данную технологию, чтобы была возможность общения не только с помощью простого запроса информации, за которым следует запрограммированный ответ [10]. Так были созданы два образовательных чат-бота, на платформе Facebook Messenger, они помогают студентам в области бухгалтерского учета. Большинство (72% пользователей) выразили свое общее удовлетворение этими чат-роботами в качестве виртуальных преподавателей [11].

Чат-боты могут также служить и круглосуточной справочной системой, стоимость которой намного меньше, чем содержание большого количество дополнительных сотрудников. Университеты, которые создали чат-бот для ответов на запросы студентов, фактически создали систему эффективного обмена информацией. Студентам, как новичкам, так и старшекурсникам, нет необходимости тратить время на то, чтобы узнать, к кому обратиться со своим специфическим вопросом.

Разработка автоматизированной системы информирования позволит учебному учреждению:

- автоматически решать рутинные задачи;
- добиться централизации источника информации;
- повысить лояльность аудитории. Бот работает быстрее человека и моментально отвечает на сообщение, быстрее выдает информацию;
- сделать информацию доступной, мобильной и удобной;
- повысить эффективности работы сотрудников;
- упростить процесс информирования студентов.

Объект исследования: система информирования.

Предмет исследования: использования чат-ботов для информирования пользователей ТПУ

Цель работы: проектирование и разработка информационной системы уведомления обучающихся в ТПУ на основе платформы Telegram**.**

1 Выбор платформы для информационной системы

1.1 Анализ российской интернет аудитории

Интернет захватывает многие сферы жизнедеятельности людей. По данным РАЭК, аудитория Рунета на декабрь 2019 года составила почти 96 млн человек, при этом, согласно экспертам, в «онлайне» теперь 79.8% взрослого населения [12].

Согласно данным Яндекс.Метрики, представленным на рисунке 1, с каждым годом растет доля смартфонов, как основного устройства для выхода в интернет.

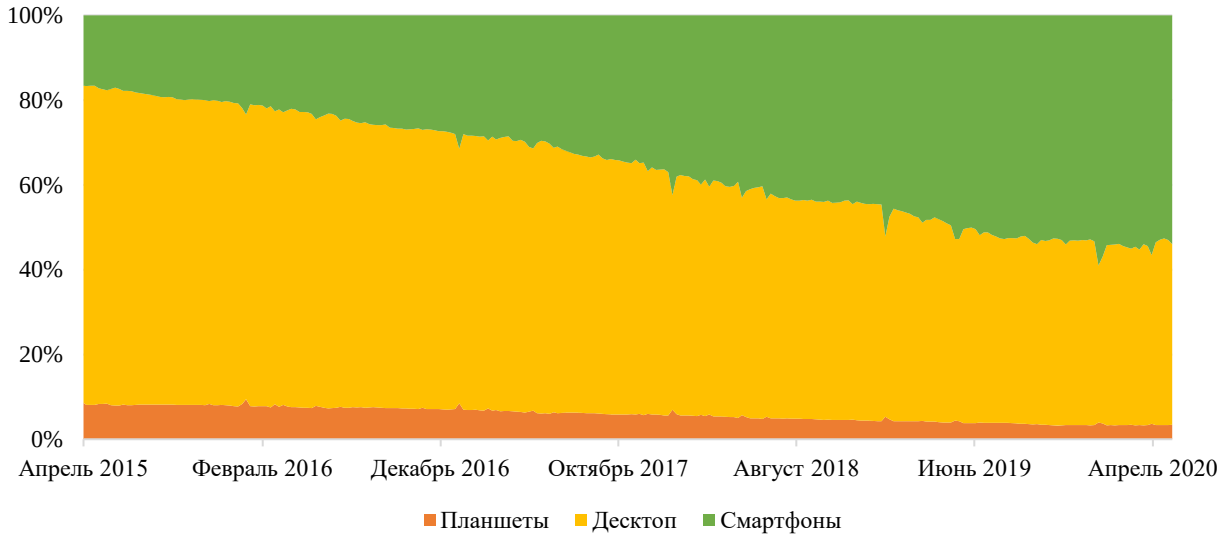


Рисунок 1 – Типы устройств в России за 2015-2020 гг. [13].

Так, в апреле 2015 года соотношение составляло 8,5% планшеты, 16,5% смартфоны и 75% десктоп, однако на май 2020 года соотношение следующее: 3,5% планшеты, 54% смартфоны и 42,5% десктоп.

Среди пользователей во всех возрастных группах время, проведенное онлайн через мобильные устройства, либо больше, либо примерно равно времени в интернете через стационарные компьютеры и ноутбуки [14].

При этом значительный рост продолжает показывать количество эксклюзивных мобильных пользователей интернета. В 2018 году этот показатель превышает аудиторию эксклюзивных десктопных пользователей, прирост за год

составил 20%. В настоящий момент 24% пользователей из всей России выходит в интернет только с мобильных устройств [15].

Среди эксклюзивной аудитории мобильных устройств, значимую долю составляют студенты и учащиеся [15].

Абсолютным лидером по времени, проводимому в сети, среди тематических групп являются социальные сети – 22% всего времени в десктопе и 24% – в мобайле. 52% россиян ежедневно заходят хотя бы в одну из социальных сетей [14].

Социальные сети, как и электронные образовательные системы, являются продуктами так называемого Веб 2.0 – этапа развития сети Интернет, при котором пользователи самостоятельно заполняют веб-сайты контентом, а разработчики предоставляют лишь сервис [16].

Согласно отчета директора по маркетингу Brand Analytics Василия Черного в Социальных медиа на ноябрь 2019 года публикуется 1.3 млрд сообщений в месяц от порядка 49 млн активных пользователей, при этом больше 40% сообщений приходится на социальную сеть «ВКонтакте» [17].

По данным за февраль 2019 года, наиболее популярными ресурсами по совокупной аудитории на мобильных и десктопных устройствах в Рунете являются: Яндекс - его среднесуточный охват среди населения 12-64 лет, проживающего в городах 100k+, составляет 47%, ВКонтакте – 43% и Google – 39%. На мобайле самая большая аудитория у WhatsApp. Этим мессенджером на мобильных устройствах пользуется 37% населения [14].

1.2 Анализ результатов опроса заинтересованности пользователей

Среди потенциальных пользователей создаваемой информационной системы был проведен опрос, с целью выявления заинтересованности в данной системе. В опросе приняло участие 94 человека. Распределение респондентов по уровням подготовки представлено на рисунке 2.

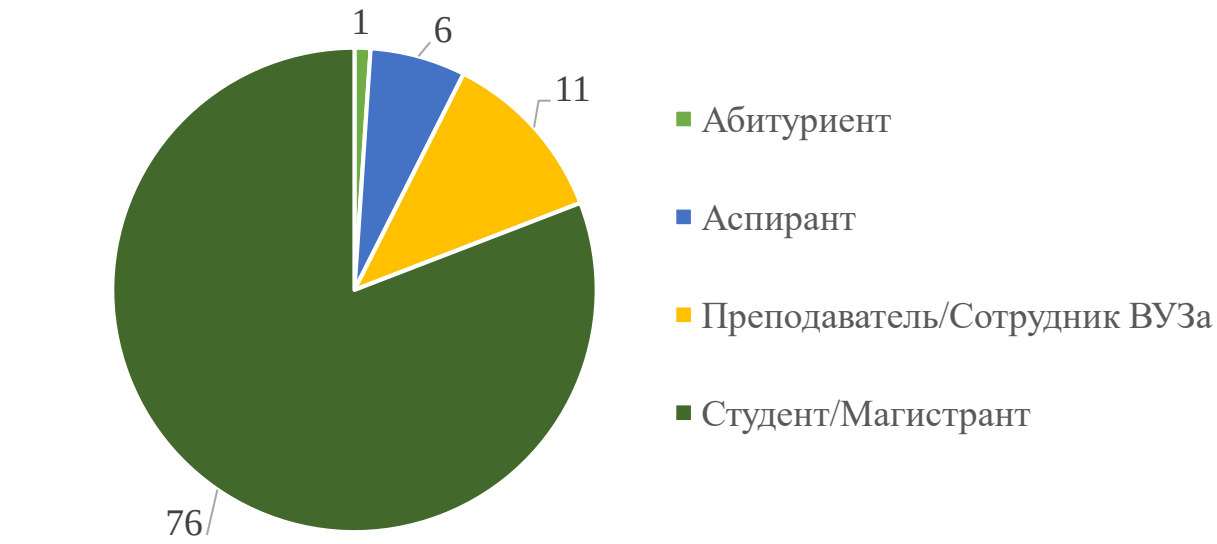


Рисунок 2 – Распределение респондентов по уровням подготовки

Согласно данным опроса, 92% высказали заинтересованность в данной системе. Кроме того, только 1 человек из 11 ответивших Преподавателей/Сотрудников ответил, что ему не интересна. Единственный ответивший абитуриент, также ответил отрицательно.

В опросе предлагалось так же выбрать, какими мессенджерами вы пользуетесь, результат представлен на рисунке 3.

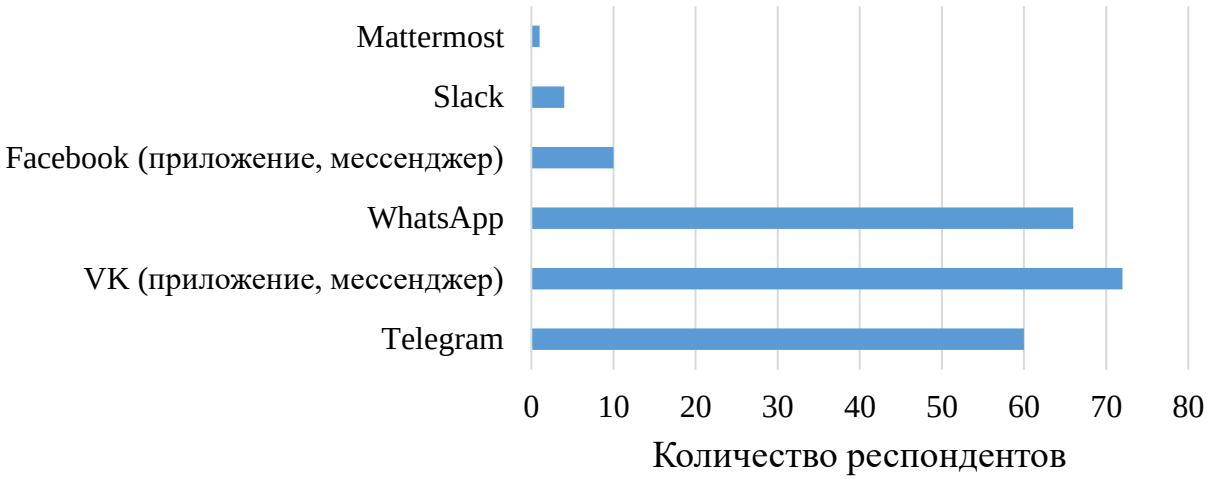


Рисунок 3 – Мессенджеры, которые используются респондентами

Как видно из рисунка 3, наиболее частыми используемыми мессенджерами и приложениями являются VK, WhatsApp и Telegram. Кроме

того, всего один человек из опрошенных пользуется только Telegram, два только VK. Таким образом, все остальные участники пользуются двумя и более различными мессенджерами.

Среди того, что хотелось бы видеть в качестве основного функционала можно выделить следующие пункты (можно было выбрать несколько вариантов), в скобках указан процент от количества опрошенных:

- быстрое получение расписания на неделю (68%);
- получение актуальных новостей (65%);
- поиск персональных сайтов сотрудников (58%);
- оставления заявок на техническую поддержку (30%);
- общение с единым деканатом (4%).
- уведомления из электронных курсов (1%).

1.3 Анализ мессенджеров

Образовательные стратегии для мессенджеров

Вот некоторые основные стратегии, которые преподаватели могут использовать, при общении со студентами в мессенджерах:

- возможность создания групповых чатов для учебных групп;
- возможность создавать аудио/видео уроки и отправлять непосредственно студентам;
- возможность оставаться на связи со студентами за пределами аудитории;
- возможность отправлять вопросы или задания студентам, даже если они не в аудитории;
- возможность отправлять медиа-контент, например, изображения или диаграммы.

1.3.1 WhatsApp

WhatsApp – это приложение для обмена мгновенными сообщениями, которое обеспечивает связь между владельцами разных платформ (разные мобильные телефоны с разным аппаратным и программным обеспечением) [18].

Число пользователей WhatsApp растет с каждым днем: с более 200 миллионов активных пользователей в апреле 2013 года до 700 миллионов в январе 2015 года и 2 миллиардами пользователей в 180 странах в 2020 году [19].

Пользователи могут общаться как индивидуально, так и создавать группы, с помощью которых общение может происходить более чем с одним контактом одновременно. Использование приложения – бесплатно.

Однако, для того, чтобы создать бота в WhatsApp необходимо подать заявку на участие в программе WhatsApp Business API [20], которая в настоящее время ориентирована на средний и крупный бизнес. Начать использовать WhatsApp API можно только после предварительно регистрации бизнес-аккаунта, после того как заявка будет рассмотрена и одобрена.

С помощью чат-бота в WhatsApp можно решать такие же задачи, как и в других мессенджерах (Viber, Telegram, Facebook Messenger). Бот может отправлять ссылки, emoji, аудиофайлы, изображения, геолокацию.

Пользователи могут использовать готовые кнопки ответов или общаться на естественном языке. В отличие от других мессенджеров, в WhatsApp нельзя делиться стикерами, а так же делать специальные – ссылки в тексте ответа, при нажатии на которые идёт отправка фразы с текстом как будто сам пользователь написал и отправил этот текст [21].

1.3.2 VK

ВКонтакте – одна из самых известных и высокотехнологичных российских компаний, крупнейшая социальная сеть в России и странах СНГ, с аудиторией 97 млн пользователей в месяц [22].

ВКонтакте предлагает привычную для многих пользователей среду для общения. В нем сочетаются как преимущества WhatsApp, так и преимущества

Telegram по наличию API в открытом доступе и возможность создания чат-ботов от имени сообщества [23].

Для того чтобы создать достаточно создать сообщество, от имени которого бот будет общаться с пользователями ВКонтакте. Кроме того, командой разработчиков представлены примеры реализации и подробное описание интеграции [23].

В качестве основных преимуществ платформы можно выделить следующие [24] пункты.

- **Мультимедиа**. У ВКонтакте своя библиотека медиаконтента, который можно использовать для создания подборок фотографий, видео или аудиозаписей.
- **Кроссплатформенность.** Сообщения сообществ работают в полной и мобильной версиях, а также во всех официальных клиентах. Это значит, что пользователь может взаимодействовать с ботом на любой платформе.
- **Доступный интерфейс.**
- **Хорошо структурированная документация.**
- **Поддержка разработчиков.**

Используя публичный API ВКонтакте разработчик соглашается с Правилами [25] платформы, кроме того ВКонтакте оставляет за собой право без уведомления полностью или частично прекратить доступ приложения к публичному API (в том числе путём блокировки приложения) в случае нарушений.

1.3.3 Telegram

Telegram – это приложение для обмена сообщениями, которое фокусируется на скорости и безопасности [26].

В отличие от WhatsApp, Telegram представляет собой облачный мессенджер с синхронизацией [26]. Таким образом можно получить доступ к сообщениям и медиафайлам сразу с нескольких устройств.

Telegram предоставляет открытое API для разработчиков для создания собственных клиентов. Кроме того, есть Bot API [27], платформа для разработчиков, которая позволяет любому легко создавать специализированные инструменты для Telegram, интегрировать любые сервисы и даже принимать платежи от пользователей по всему миру [26].

В апреле 2020 года Telegram объявил о том, что достиг отметки в 400 млн активных пользователей, что на 100 миллионов больше, чем год назад [28]. Каждый день в Telegram регистрируется не менее 1.5 миллиона человек. Папки, облачное хранилище, поддержка работы с компьютеров – все это делает Telegram незаменимым инструментом для работы и учебы [28].

Развитие Telegram привело к созданию большого количества научно-популярных каналов. Любой специалист может вести собственный канал о своей деятельности, новостях и тенденциях.

Можно сделать вывод, что Telegram является удобным бесплатным средством коммуникации студента и преподавателя, который ускорит и упростит взаимодействие между ними. Также использование данного мессенджера в электронном обучении не ново [29, 30].

1.3.4 Facebook Messenger

Социальная сеть Facebook в апреле 2016 года открыла свою платформу Messenger, с тех пор она насчитывает более 1.3 миллиарда пользователей в месяц [31], больше чем 300 тысяч аккаунтов с чат-ботами и 8 миллиардами сообщений в день [32].

Основным преимуществом использования чат-ботов на платформе Facebook Messenger является низкий барьер для входа как для создателя, так и для целевой аудитории. Кроме того, по состоянию на октябрь 2019 года, данная платформа находится на втором месте по количеству активных пользователей в месяц [33], что говорит о его популярности.

Преимуществами для пользователей является знакомый интерфейс, нет необходимости скачивать и устанавливать дополнительные приложения. Для

разработчиков чат-ботов можно выделить следующие преимущества: использование существующей инфраструктуры социальной платформы, персонализация разговора, платформа есть и на мобильном телефоне, и в вебе [10].

1.4 Вывод

Таким образом, исходя из описанных выше разделов, можно выделить двух лидеров, которые предоставляют сравнительно равные возможности – Telegram и ВКонтакте. Они оба представлены, как в мобильном, так и десктопом сегменте. Среди потенциальных пользователей будущей информационной системы они также примерно равны (небольшое преимущество за ВКонтакте). Обе платформы предоставляют открытый доступ к своему API. Для разработки был выбран Telegram, в следствие относительного опыта работы с данной платформой, однако, возможна дальнейшая реализация бота и на платформе VK.

2 Проектирование информационной системы

Перед началом разработки было проведено проектирование системы. Первым делом определены функциональные требования [34] и построены диаграммы вариантов использования в нотации UML.

На рисунках 4, 5 представлены диаграммы вариантов использования. Было выделено три роли: администратор, зарегистрированный и незарегистрированный пользователь.

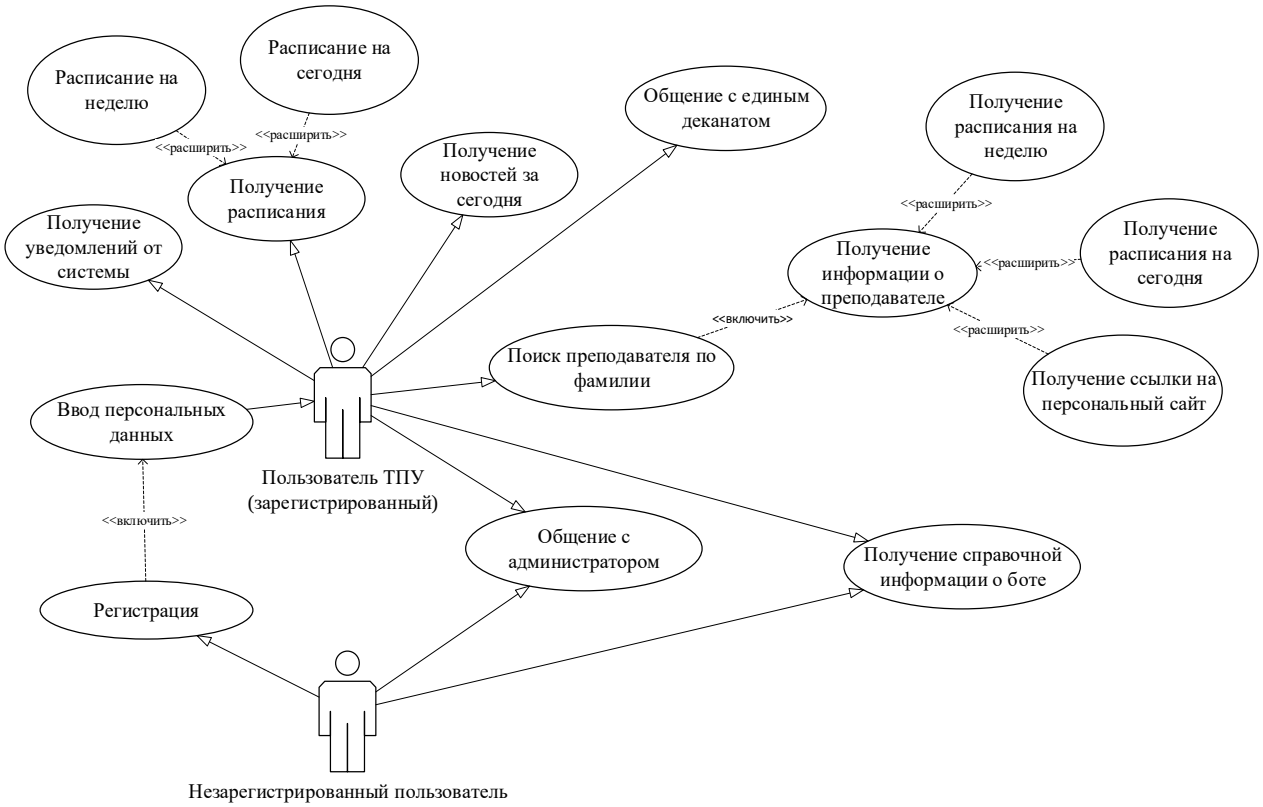


Рисунок 4 – Диаграмма вариантов использования для незарегистрированного и зарегистрированного пользователя

Очевидно, что для полноценного использования веб-приложения необходимо быть зарегистрированным пользователем. Незарегистрированные пользователи могут только общаться с администратором и получить справочную информацию о клиенте, который они используют.

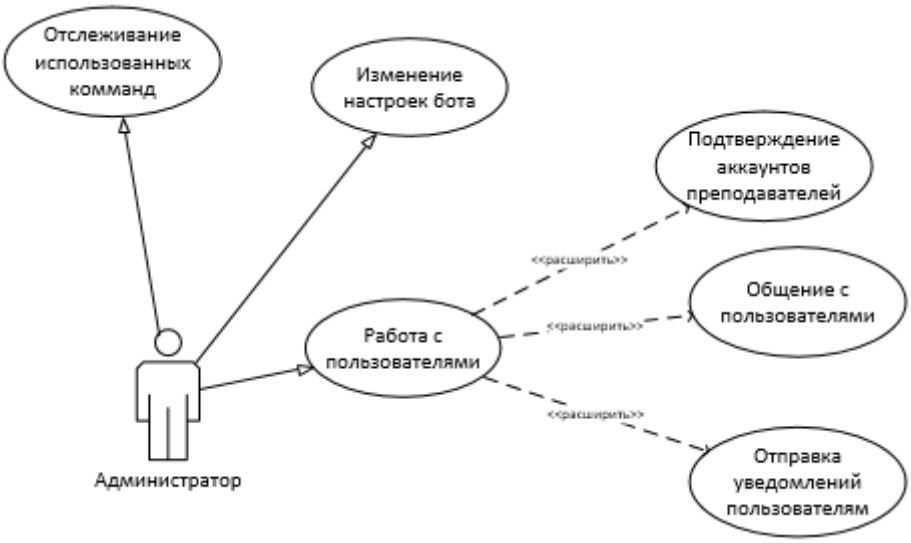


Рисунок 5 – Диаграмма вариантов использования для администратора

2.1 Архитектура информационной системы

Функциональность приложения разделена на две главные части: клиент и сервер. Такое разделение интересов позволяет как клиенту, так и серверу развиваться независимо, так как требуется только то, чтобы интерфейс оставался прежним.

Сервер **–** является ядром системы и предоставляет клиентам определенный функционал. Сервер, обычно, не является инициатором «общения», а ожидает запросы от внешних клиентов и отвечает на них. Связь может быть установлена по-разному, в зависимости от выбранного интерфейса. Сколько запросов одновременно обрабатывает сервер можно ограничивать, если такого требует политика разработки. Для сервера используют выделенную инфраструктуру, которая должна легко масштабироваться в случае возрастания нагрузки.

Клиент – это связь между пользователем и сервером. Он отправляет запросы на сервер и отображает ответ в удобном для пользователя виде. Клиенты могут быть разного типа, от простого терминала до сложных веб и GUI интерфейсов.

Описанную архитектуру называют двухуровневой.

Логику приложения можно по-разному разделять между сервером и клиентами. Но стоит отметить, что чем больше специализированный получается

клиент, тем менее гибким выходит вся система, так как уменьшается общий функционал, который достигается за счет центрального места – сервера. Поэтому при использовании такой архитектуры стараются строить так называемый «тонкий клиент».

Однако у данного подхода также есть и недостатки. Архитектура клиент/сервер оказывается парализованной, когда недоступен сервер (вследствие сбоя программы или планового обслуживания): ни один клиент не сможет работать, пока сервер не заработает снова. По этой причине системы, организованные по принципу клиент/сервер, обычно требуют наличия администратора, который обеспечивает их бесперебойную работу [35].

На рисунке 6 графически изображена архитектура создаваемой информационной системы.

Рисунок 6 – Архитектура информационной системы

В работе используется REST, для которого можно выделить следующие принципы построения [36].

- **Отсутствие состояния**. Вся информация, необходимая для выполнения запроса, полностью содержится в коммуникации между клиентом и сервером. Состояние полностью хранится на стороне клиента, а на сервере сеанса нет. Если необходима аутентификация – клиент должен делать это при каждом запросе.

- **Кэшируемость**. Клиент, сервер и любые промежуточные компоненты могут кэшировать ресурсы для повышения производительности.
- **Единый интерфейс**. Существуют единые правила для общения между компонентами, что упрощает архитектуру.
- **Многоуровневость**. Отдельные компоненты не могут видеть за пределами непосредственного уровня, с которым они взаимодействуют. Это позволяет компонентам быть независимыми и, следовательно, легко заменяемыми или расширяемыми.

2.2 Обоснование выбора программных средств разработки

На первом этапе, при проектировании информационной системы, был также поставлен вопрос о технологиях, которые будут использоваться при разработке.

2.2.1 Выбор технологий для сервера

В мире существует большое количество различных фреймворков, которые позволяют ускорить процесс разработки, поэтому часто разработчики выбирают технологию, написанную на уже знакомом языке. Выучить новый язык несложно, однако это занимает время. Ниже рассматриваются преимущества и недостатки некоторых из популярных фреймворков для бэкенд разработки [37].

2.2.1.1 Django

Django **–** это высокоуровневый фреймворк, который использует MVT (модель, представление, шаблон), написанный на Python. Это бесплатная среда с открытым исходным кодом, которая способствует быстрой разработке и помогает разработчикам создавать эффективный и чистый код [38]. Это довольно мощная инфраструктура и многие крупнейшие компании используют ее, например, Pinterest, Instagram, Udemy и другие [38].

Можно выделить следующие преимущества Django.

- **Быстрая и недорогая разработка MVP.** Django имеет четко установленную структуру, благодаря которой можно легко повторно использовать код из одного проекта в другом. Кроме того, существует множество готовых библиотек, которые помогут создать прототип продукта.
- **Явный, логический синтаксис.** В основе Django лежит Python, который считается одним из самых простых языков для изучения [40].
- **Открытая экосистема.** Django open-source проект, а значит доступно большое количество как бесплатных так и платных библиотек.
- **Django Rest Framework.**
- **Панель администратора.** Встроенная панель является удобным инструментом для пользовательского управления.

К главному недостатку Django можно отнести то, что это монолитная система. Инфраструктура фреймворка представляет собой единый блок, в котором все компоненты связаны вместе, поэтому нельзя выбирать, какие части необходимо подключить в конкретном продукте. Кроме того, Django держит разработчиков в определенных рамках и шаблонах.

2.2.1.2 Spring

Spring Framework предоставляет комплексную модель программирования и конфигурации для современных корпоративных приложений на основе Java на любой платформе развертывания [41].

Большинство разработчиков по всему миру используют Spring Framework для создания высокопроизводительных веб-приложений. Кроме того, он помогает создавать простые, быстрые, гибкие и переносимые решения [37].

К преимуществам Spring Framework относят следующие пункты.

- **Plain Old Java Object (POJO).** Простой Java-объект, не унаследованный от какого-то специфического объекта и не реализующий никаких служебных интерфейсов сверх тех, которые нужны для бизнес-модели [42].
- **Гибкие конфигурации.**

- **Аспектно-ориентированное программирование.** Разработчики могут иметь разные модули компиляции или отдельный загрузчик классов.
- **Легко тестировать.**

Недостатки Spring Framework:

- **Сложность и высокий порог входа.** Его стоит использовать только в том случае, если есть опыт работы с этой платформой ранее [43].
- **Отсутствие руководств.** Например, в документации Spring ничего не говорится о таких угрозах, как XSS или CSRF.
- **Параллелилизм.** Несмотря на то, что это является также и плюсом фреймворка, однако разработчику нужно четкое понимание, чтобы знать какие методы или классы могут быть полезны в конкретном случае [44].

2.2.1.3 Ruby on Rails

Ruby on Rails **–** также является фреймворком с открытым исходным кодом. Это позволяет разработчикам использовать готовые решения, помогая им сэкономить время на процессах программирования [45]. Rails в своем стэке используют такие компании как Airbnb, Twitter, Github [46]

Основанный на языке Ruby, Ruby on Rails унаследовал логику и простоту своего родителя. По сути, Rails – это слой поверх Ruby, который помогает разработчикам создавать веб-приложения.

В качестве полноценной среды он «из коробки» предлагает систему объектно-реляционного отображения (ORM) для бизнес-данных и логики, управления приложениями и маршрутизацию [45].

Можно выделить следующие преимущества ***Ruby on Rails***.

- **Компонентная структура на основе плагинов и гемов.** Плагины (уровень приложения) и гемы (уровень системы), позволяют опытным разработчикам RoR быстро создавать эффективные приложения с меньшим количеством кода. Плагины хорошо документированы и просты в использовании. Новые гемы постоянно добавляются в общедоступные

репозитории, такие как популярный ресурс RubyGems, который в настоящее время содержит для загрузки более 150 000 гемов [47].

- **Легко переносить и модифицировать.**
- **Разнообразие сохранённых наборов настроек и инструментов.** Множество функций, которые уже предварительно настроены, что ускоряет процесс разработки.

Существуют также и недостатки.

- **Быстро растущая сложность и технический долг.** Имеющаяся гибкость приводит к тому, что существует много способов сделать одно и тоже, код может стать трудным для чтения и потребует дополнительной доработки в дальнейшем [48].
- **Сложнее создать API.** Нет такого инструмента, как DRF.
- **Качество документаций.** Достаточно сложно найти хорошую документацию, особенно для менее популярных гемов.

2.2.1.4 Laravel

Фреймворк Laravel полезен для создания быстрых серверных веб-приложений, основан на PHP [49]. В своем стэке Laravel используют такие компании как 9GAG, MasterCard, E-Commerce [50].

Можно выделить следующие преимущества Laravel.

- **Поддержка ORM.**
- **Легковесность**.
- **Управление очередями.** Laravel предоставляет абстракцию над необязательными задачами и постановкой их в очередь, что значительно ускоряет время отклика пользователя [51].
- **Быстрое выполнение веб-приложения [51].**

Одновременно и преимуществом, и недостатком является то, что фреймворк достаточно новый (2011 года), а значит в нем еще встречаются не устранённые проблемы, кроме того, сообщество разработчиков меньше, чем в других представленных фреймворках.

Кроме того, легковесность тоже может быть недостатком, так как будет необходимость в интеграции сторонних инструментов для решения нетривиальных задач, что может быть утомительно и сложно.

2.2.1.5 Вывод

Таким образом, исходя из преимуществ и недостатков фреймворков, описанных выше, в качестве технологии для разработки выбран **Django**. Обусловлено это необходимостью создания прототипа в сжатые сроки одним человеком, наличию удобной библиотеки для построения REST API, знакомству с языком и фреймворком в целом, кроме того в нем сочетаются преимущества большинства из описанных решений, а недостатки проявляются только на поздних стадиях разработки.

2.2.2 Выбор технологий для клиента

2.2.2.1 Python

Python – это быстрый, простой в использовании и простой в развертывании язык программирования, который широко используется для разработки масштабируемых веб-приложений [53].

Выделяются следующие преимущества этого языка программирования.

- **Прост в изучении.**
- **Поддержка ООП.**
- Идеально подходит для **создания прототипов** и быстрого тестирования идей.
- **Большое сообщество** и обширная поддержка различных библиотек.

К недостаткам можно отнести следующие пункты.

- **Интерпретируемый**. В следствие этого по скорости проигрывает компилируемым, несмотря на развитие технологий.
- **Однопоточность**. В Python есть GIL – только один поток управляет интерпретатором, а значит только один поток выполняется в конкретный момент времени [54].

2.2.2.2 Java

Java – это язык программирования общего назначения, который следует парадигме объектно-ориентированного программирования и подходу Write Once Run Anywhere. Java используется для настольных, веб, мобильных и корпоративных приложений [55]. Известно, что Java чрезвычайно стабильна, поэтому многие крупные предприятия приняли ее и используют в приложениях начиная от сайтов электронной коммерции и заканчивая крупными финансовыми приложений, таких как электронные торговые системы [56].

К преимуществам этого языка программирования относятся следующие пункты.

- **Библиотеки с открытым кодом.**
- **ООП парадигма**.
- **JVM**. Высокая независимость от платформы.
- **Поддержка многопоточности.**

В качестве недостатков можно выделить.

- **Порог вхождения.**
- **Ошибки в JVM** и библиотеках [57].
- **Требования к памяти**. По сравнению с другими компилируемыми языками, такими как C и C ++, Java потребляет больше памяти и значительно снижает производительность [58].

2.2.2.3 C++

C++ – это язык программирования общего назначения, ссозданный как расширение языка программирования C, или «C with Classes». Язык значительно расширился со временем, и современный C ++ теперь обладает объектно-ориентированными, универсальными и функциональными функциями в дополнение к средствам низкоуровневой манипуляции с памятью [59].

Почти все низкоуровневые системы, такие как операционные системы, файловые системы и т. д, написаны на C / C ++.

C++ также широко используется благодаря тому, что он чрезвычайно быстрый и стабильный. C++ также предоставляет STL (стандартная библиотека шаблонов). STL – это список готовых к использованию библиотек для различных структур данных, арифметических операций и алгоритмов [60].

Преимущества C++:

- более быстрое выполнение программы;
- множество компиляторов и библиотек;
- низкий уровень абстракции;
- широкий спектр применения;
- мультиплатформенность.

Недостатки C++:

- сложный синтаксис;
- менее эффективная ООП по сравнению с другими языками [61];
- нет сборки мусора или динамического выделения памяти;
- проблемы переполнения буфера и повреждения памяти.

2.2.2.4 Вывод

В связи с вышесказанным, для разработки первой версии клиента для Telegram был выбран Python, как один из самых быстрых способов создать работающий продукт. Кроме того, существует большое количество готовых библиотек по работе с Telegram Bot API, таких как Telepot [62], что позволит сосредоточиться на написании бизнес-логики и функционала приложения, а не на коммуникациях.

3 Программная реализация

В соответствии с описанной функциональностью в главе 2, необходимо было получить данные, для предоставления конечному пользователю. Данные получены с помощью веб-парсинга – автоматизированного процесса извлечения информации с веб-сайта. Для этих целей используется парсер для синтаксического разбора файлов Beautiful Soup [62].

Часть информации, такая как соответствие тривиального названия группы и идентификатор группы в системе ТПУ, информация о преподавателях, была извлечена заранее и сохранена в локальной базе данных с помощью fixtures. Fixtures – это коллекция данных, которая может быть загружена в БД, они могут быть написаны в форматах JSON, XML или YAML [64] и использоваться в качестве инициализирующих значений при запуске приложения. В качестве СУБД используется PostgreSQL [65].

Часть информации, как получение расписания или свежих новостей извлекается во время работы сервера по запросу, при этом она не сохраняется, а отдается в качестве ответа клиенту.

Пример скрипта для получения расписания на неделю представлен в приложении А. Расписание извлекается с сайта **https://rasp.tpu.ru,** для того, чтобы сделать скрипт универсальным передает идентификатор: для пользователей начинающийся с “user_”, для группы с “gruppa_”. В ТПУ используется своя система нумерации недель, возникает необходимость синхронизировать календарную неделю с неделями в ТПУ.

Расписание на сайте формируется с помощью JavaScript, а Beautiful Soup его не выполняет. Для того чтобы имитировать действие пользователя, была так же использована библиотека dryscrape [66].

Расписание на сайте представлено в виде таблицы, в которой дни расположены по столбцам, однако при парсинге таблица считывается построчно. Полученная таблица была транспонирована, за исключением первого элемента каждой строки (в котором содержится время о начале пары). Кроме того,

необходимо учесть дни, которые являются выходными. Такие дни отображаются растянутой ячейкой сквозь всю таблицу, а значит извлекаются только в первой строке. Было принято решение извлечь их до транспонирования, получив индексы элементов (дни недели), а затем на эти места вставить «пустой день».

3.1 Модуль Server

Как следует из предыдущей главы в качестве фреймворка для сервера был выбран Django. Для построения серверной части использовался так же Django REST Framework (DRF) [67]. DRF – это мощный и гибкий инструментарий для создания веб-API.

Django предоставляет встроенную панель администрирования, которая позволяет управлять всем проектом целиком, в том числе это удобный интерфейс по работе с базой данных. У панели администрирования собственная форма аутентификации, которая разрешает работу только суперпользователям или с правами на вход в данную систему.

DRF в свою очередь ограничивает права доступа к предоставляемому API с помощью «разрешений» [68], на тех эндпоинтах, где содержится личная информация установлена проверка ***IsAdminUser***, которая отклонит все запросы, если пользователь не является доверенным администратором. Telegram боту предоставляется специальный токен, по которому его запросы классифицируются как запросы администратора.

В терминологии Django внутри проекта было создано несколько приложений, отдельных модулей, предоставляющих определенный функционал. Приложение может быть повторно использоваться в других проектах и содержит в себе набор моделей, URL-ов, сериалайзеров и т.д. Выделены следующие отдельные модули (рисунок 7):

- профили пользователей;
- ядро системы;
- модуль менеджмента;
- функционал;

- аналитика.

Рисунок 7 – Панель администрирования серверной части информационной системы

Так информационная система может быть использована не только студентами, но и сотрудниками университета, были выделены две модели **Employee** и **Student**, расширяющие модель **TpuUser**. Данные модели используются как основные для управления пользователями. Диаграмма моделей классов для все модулей представлена в приложении Б.

В блоке «ядро» расположены настройки для Telegram бота, а также команды, которые могут быть расширены только на стороне сервера, без необходимости добавлять их в клиентском приложении. Это могут быть простые информационные команды, не требующие дополнительных действий. Например, команда ***«/help»***, по отправке которой в бот, придет справочное сообщение, написанное в панели администратора, рисунок 8.

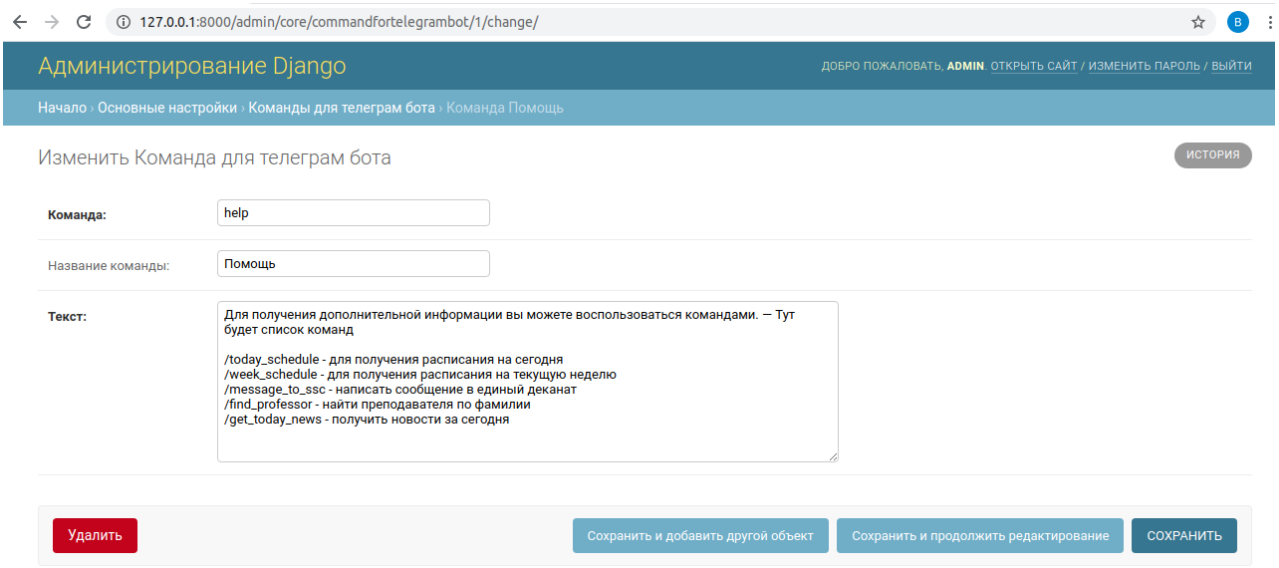


Рисунок 8 – Добавление команды для Telegram бота в панели администратора

Приложение Аналитика создано для того, чтобы исследовать работу клиентских приложений. На данный момент оно представлена моделью **CommandUsage**, в которую записывается какой пользователь и какую команду выполнял в боте, рисунок 9. Таким образом можно узнать наиболее востребованный функционал и продолжать развивать его.

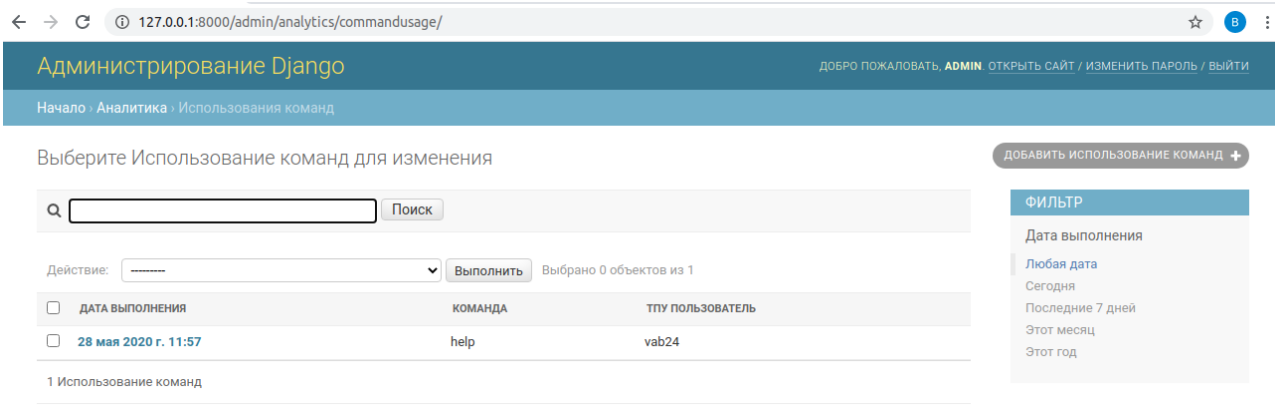


Рисунок 9 – Использование команд в Telegram боте

В модуле функционал содержится вся вспомогательная информация, такая как номера групп, название школ ТПУ, информация о сотрудниках. Кроме того, именно этот модуль предоставляет API для получения расписания и новостей.

3.1.1 Уведомления клиентов с помощью сервера

Одной из особенностей информационной системы является возможность уведомлять одного или нескольких пользователей через систему оповещения. В панели администратора создается событие, в котором заполняется текст сообщения и выбираются пользователи, пример такого заполнения представлен на рисунке 10.

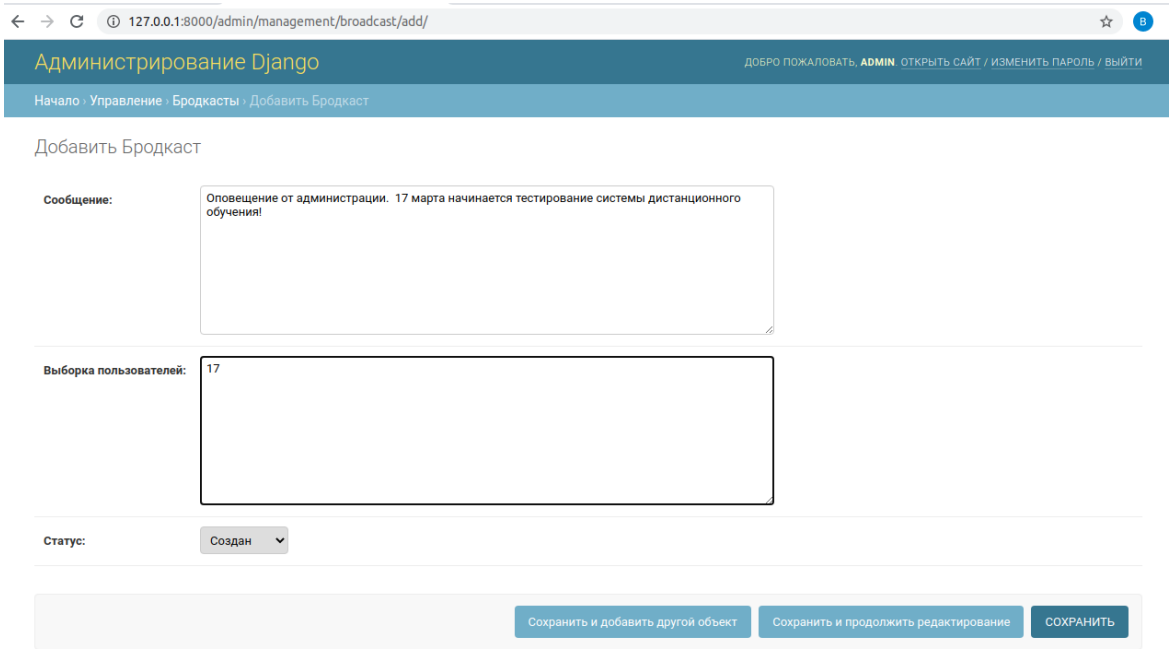


Рисунок 10 – Создание оповещение в панели администратора

Однако, возникает необходимость в отправке данного запроса с уведомлением каждому из подключенных клиентов. На стороне клиента нет сервера, чтобы он принимал запросы, а соединение между Сервером и Клиентом не держится постоянно.

Чтобы решить эту задачу был выбран сервис по отправке сообщений Amazon Simple Notification Service (SNS), который работает по модели «издатель – подписчик» (Pub/Sub). Коммуникация между издателями и подписчиками происходит асинхронно. Сервис Amazon SNS использует хранилище сообщений в различных зонах доступности, чтобы обеспечить повышенную сохранность сообщений [69].

Потребление подписчиками (веб-серверы, клиенты и так далее) сообщений происходит по одному или нескольким поддерживаемых

протоколов, в частности с помощью электронной почты или Amazon SQS). Если подписанный адрес недоступен, сервис Amazon SNS применяет политики повторной отправки недоставленных сообщений. Кроме того, сервис может отправлять сообщения в очереди сообщений, которые не могут быть обработаны. Подписчики получают все сообщения, опубликованные в темах, на которые они подписаны, и все подписчики на темы получают одинаковые сообщения. Таким образом, одно сообщение попадает во все клиенты и может быть обработано по-своему.

В качестве протокола подписки используется упомянутый выше Amazon SQS. Сервис Amazon Simple Queue Service (Amazon SQS) предлагает безопасную, надежную и доступную очередь. Также у Amazon SQS есть очереди недоставленных сообщений [70]. На рисунке 11 представлена диаграмма общения.

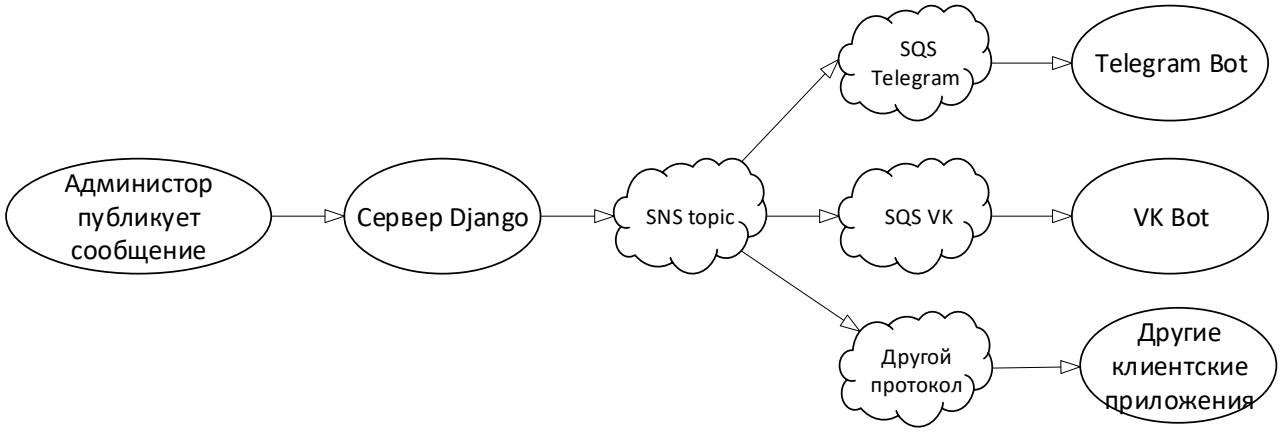


Рисунок 11 – Схема общения через SNS-SQS стэк Сервера и Клиентов

Таким образом, Amazon SQS и Amazon SNS – это сервисы для передачи сообщений в очередях и по темам, они легко масштабируются, просты в использовании и не требуют настройки посредников.

Класс очередей для сообщений представлен в приложении В. Для взаимодействия с Amazon используется библиотека boto3 [71]. При запуске сервера проверяется доступность соединений с AWS с помощью метода **connect**, который запрашивает все темы, доступные для текущего клиента и, если тема совпадает с выставленной в настройках, устанавливается соединение. Когда

возникает необходимость отправки сообщений в очередь, используется метод **send_message** который публикует информацию в AWS SNS.

3.2 Модуль Telegram Bot Client

Клиентское приложение будут использовать большое количество пользователей одновременно, поэтому важно, чтобы оно умело работать асинхронно. Асинхронность в программировании – это выполнение действия в неблокирующем режиме, позволяя основному потоку продолжать свою работу [72]. В Python работа с асинхронностью реализованы через библиотеку asyncio, и синтаксиса **async/await** [74].

Для того чтобы сосредоточиться на написании бизнес-логики, а не коммуникациях между приложением и Telegram Bot API был использован фреймворк Telepot [62]. В нем также есть асинхронная версия, которая позволяет делать асинхронные HTTP запросы.

Основным классом является **Bot**, который управляет всеми процессами и содержит информацию о подключаемых модулях. Так же отправка сообщений происходит через этот класс, который в свою очередь вызывает соответствующие методы у Telepot. Так как события происходят асинхронно и бот не должен завершать свою работу после обработки одного сообщения, необходимо использовать бесконечный цикл, который будет управлять остальными событиями, а также отлавливать все возникающие исключения.

Технические сообщения, содержащие информацию о возникающих ошибках, отправляемые в чат администратора сложно отличать друг от друга. Когда происходит задержка в отправке, например, под высокой нагрузкой, формируется пул сообщений, которые отправляются уже после восстановления нормального режима работы. Возникает необходимость идентифицировать такие сообщения. Для решения этой проблемы ко всем сообщениям, которые отправляются в чат администратора, добавляется так же время отправки в текущей временной зоне.

При отправке пользователям большого количества сообщений возникает еще одно ограничение: API Telegram запрещает отправку более 30 сообщений в секунду, в случае если данный лимит превышается, будет возникать ошибка [27]. Было найдено [73] и адаптировано решение, основанное на алгоритме Leaky bucket. Когда количество сообщений превышает допустимое количество, устанавливается дополнительная задержка на отправку (поток отправляется в сон), которая увеличивается с количеством сообщений. Таким образом «корзина» блокируется пока не будут отправлены предыдущие сообщения в очереди.

Сообщения от пользователей обрабатываются с помощью классов **ChatPerUser** (в случае обычного сообщения) и **CallbackQueryHandler** (в случае сообщения в формате callback). Callback запрос возникает в том случае, когда пользователь кликает на кнопку содержащую информацию, необходимую для дальнейшей обработки действий.

ChatPerUser наследуется от класса **ChatHandler**, определенного в библиотеке Telepot, переопределяя его поведение. На каждое открытие происходит проверка, является ли чат персональным или нет. В дальнейшем, возможно расширение функционала и реализации возможности работы в групповых чатах.

Так как соединение с пользователем не держится, и чат закрывается по прошествии определенного периода времени, необходимо отличать длительные процессы, такие как регистрация, от коротких команд. Для этого было решено добавлять переменную класса для каждого такого процесса. Так на каждое новое сообщение проверяется запущен ли процесс регистрации и если да, то запускается следующий шаг вопроса (подробнее в 3.2.1.1).

Для того, чтобы унифицировать работу с сообщениями и добавить вспомогательные методы, был введен класс **Request**, который хранит информацию о сообщении и чате, из которого оно было отправлено.

Если сообщение не относится к процессу регистрации, создав экземпляр Request, он передается в **Router.** Сообщения в Telegram имеют различный тип и

должны быть обработаны в соответствии с ним. Так, например, сообщения, отправленные из чата, могут быть голосовыми, текстовыми, аудио, видео и т.д. Данный класс на основе информации определяет, какой именно обработчик должен продолжить работу с данным сообщением.

Каждый новый обработчик наследуется от общего класса **BaseHandler**, пример которого приведен в приложении Г. В нем определен базовый функционал обработки и извлечения информации из переданного сообщения. Так каждое входящее событие может обработано только этим классом. Базовая обработка включает в себя проверку существует ли пользователь, отправивший сообщение, а если нет, то создает его.

С ростом функциональности возникает потребность в использовании Callback запросов, в которых содержится дополнительная информация. Для ее извлечения используется специальный метод **glance_callback_data**.

При обработке сообщения может возникнуть необходимость передать его другому обработчику, для этого предоставлен специальный интерфейс **delegate,** который принимает класс нового обработчика, и вызывает его методы.

Некоторые команды, должны быть записаны в аналитику, чтобы смотреть на частоту их использования. Было решено добавить дополнительный обработчик **CommandHandler**, который в свою очередь наследуется от **BaseHandler**. В нем переопределен вызов handle так, что он дополнительно записывает использование команд, а также получает команды из интерфейса администратора (см. 3.1).

Для того чтобы определить однозначное сопоставление и методы для каждой сущность, было решено ввести модели, которые подключаются к главному классу Bot через **ModelsManager.** В моделях определены методы и поля, в соответствие с теми, которые определены на сервере.

Для того, чтобы получать данные от сервера, для каждой модели определен так же свой API, который позволяет взаимодействовать с сервером. Каждое конкретная реализация наследуется от общего класса **ApiClient**, в котором определены основные методы отправки основных http запросов. Все

запросы также являются асинхронными и выполняются с помощью библиотеки **aiohttp**, что полностью соответствует выбранной концепции.

Таким образом, описанное выше можно схематично представить в виде диаграммы, приведенной на рисунке 12.

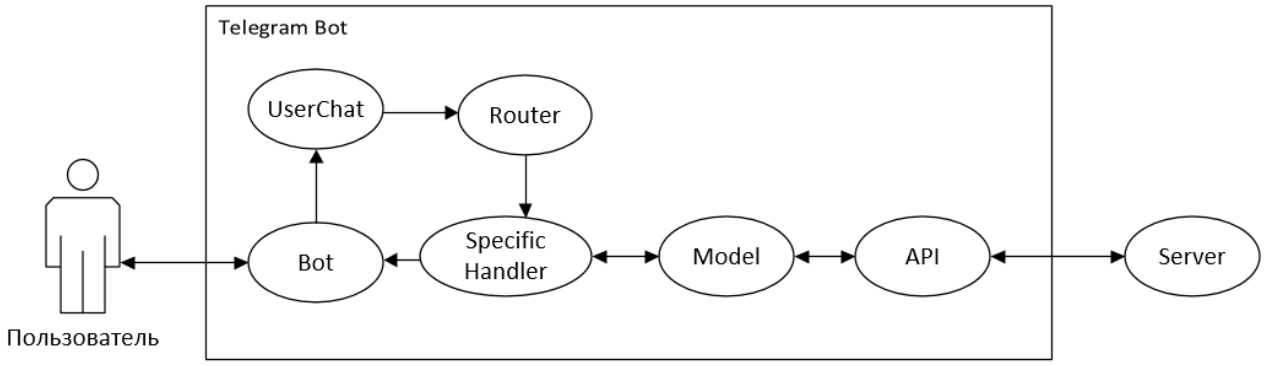


Рисунок 12 – Диаграмма процессов в Telegram Bot

3.2.1 Реализованный функционал

В соответствии с результатами опроса, приведенными в части 1.2 были выбраны и реализованы некоторые функции, которые описаны далее в этом разделе.

3.2.1.1 Регистрация

Процесс регистрации реализован как отдельный блок в модуле **RegistrationProcess.**

Как было сказано в главе 3.2, чат по прошествии определенного времени закрывается, соответственно необходимо сохранять информацию о том, какой именно шаг регистрации проходит пользователь. Для этого были введены две переменные класса ***_current_step*** и ***_is_running.*** На основании их определяется следующий блок.

Каждый блок вопросов представляет собой отдельный класс, в котором реализованы функции отправки вопроса и сохранения результата. Такая реализация позволяет расширять блок вопросов при необходимости и видоизменять подтверждение данных, так проверка корректности для

открытых текстовых вопросов будет отличаться от проверки отправленного телефона.

Регистрация для студентов и сотрудников отличается. Общими запрашиваемыми данными являются: Род занятий (студент или сотрудник), пользовательское имя в системе ТПУ, а также номер мобильного телефона. Для студентов спрашивается так же тривиальный номер группы, а сотрудников – хотят ли они получать сообщения от студентов (для дальнейшей реализации). Процесс регистрации представлен на рисунке 13.

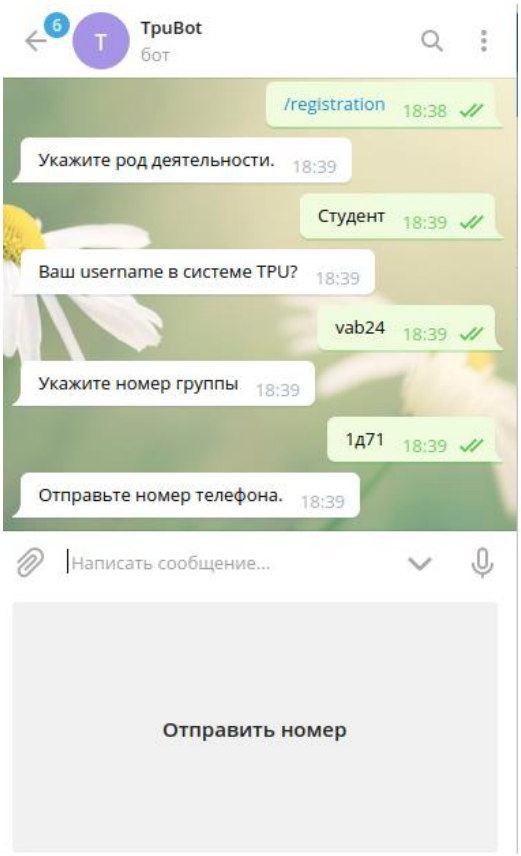


Рисунок 13 – Процесс регистрации студента в Telegram Bot.

Все закрытые вопросы сопровождаются кнопками быстрого ответа. Также все вопросы проходят стадию подтверждения, например, если пользователь введет группу, которой нет в базе данных, его попросят повторить ввод.

3.2.1.2 Расписание

Пользователь может получить два вида расписания: на сегодняшний день или на текущую неделю. После того, как пользователь вводит команду для его получения: **«/today_schedule»** или **«/week_schedule»**, сообщение по цепочке, описанной в главе 3.2 через Router попадает в **TodayScheduleHandler или WeekScheduleHandler** соответственно. Далее рассматривается расписание на сегодня.

Так как расписание предоставляется только зарегистрированному пользователю, необходимо осуществить данную проверку, вызвав обработку из суперкласса. Если пользователь существует, запрашивается метод модели ***get_today_schedule_by_user_id***, который в свою очередь вызывает синонимичный метод в своем API, отправляя запрос на сервер.

В случае положительного ответа сервера, извлекается информация о расписании и формируется блок сообщений для пользователя. Они возвращаются в Обработчик и затем отправляются в виде отдельный коротких сообщений, с удобным выделением важной информации, рисунок 14.

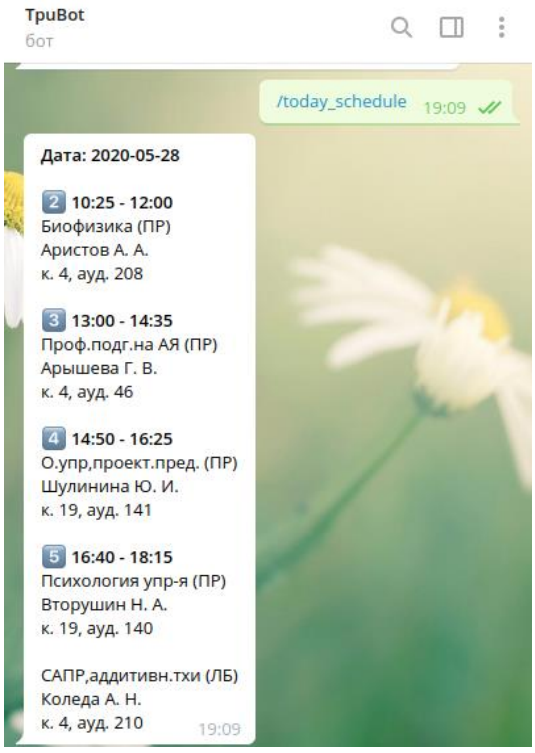


Рисунок 14 – Процесс получения расписания на сегодня для студента

Чтобы выделить важные фрагменты сообщения было решено использовать emoji, а также облегченный режим текстовой разметки Markdown.

Если пользователь является студентом, то ему придет расписание для группы, выбранной при регистрации, если пользователь является сотрудником, то предоставляется расписание для этого сотрудника.

3.2.1.3 Поиск преподавателя

Пользователь может осуществить поиск в базе данных по фамилии преподавателя. Поиск — это двухэтапный процесс взаимодействия, сначала боту необходимо определить какая именно команда, а затем уже осуществить сам поиск. Возможно было сделать процесс поиска, как процесс регистрации (см. 3.2.1.1), однако было принято решение реализовать его следующим образом.

По аналогии с 3.2.1.2 после ввода пользователем команды «**/find_professor»** сообщение попадает в **FindProfessorHandler,** который только отправляет ответное сообщение с просьбой написать фамилию преподавателя, но «заставляя» ответить на это сообщение. Таким образом, тип отправляемого сообщения меняется на ***reply,*** и оно обрабатывается **ReplyMessageHandler.**

В вышеуказанном обработчике определяется, на что именно отвечает пользователь и определяется дальнейший путь сообщения исходя из соответствия шаблону. В данном случае происходит делегация сообщения обработчику **FindProfessorByLastNameHandler.**

Здесь извлекается информация о фамилии преподавателя, через модель запрашивается API и возвращается список преподавателей. Если список пустой, было принято решение отправлять предупреждающее сообщение, что преподаватели с такой фамилией не найдены.

В случае успешного поиска ему в ответ присылается информация, содержащая всех найденных преподавателей. Пример полного успешного запроса представлен на рисунке 15.

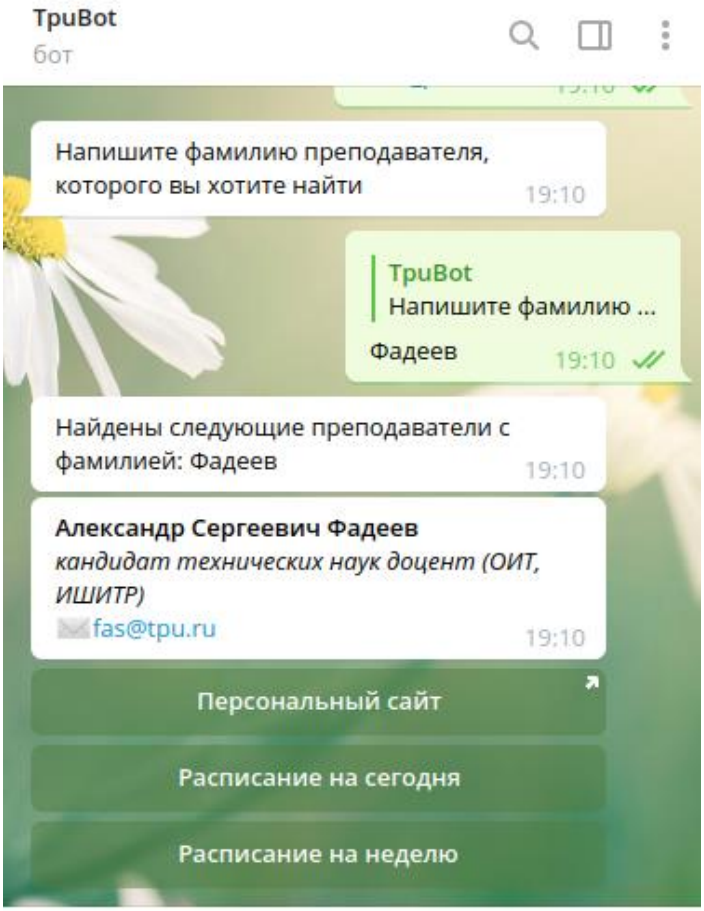


Рисунок 15 – Поиск преподавателя по фамилии в Telegram Bot

Как видно из рисунка выше, информация приходит в виде карточки, в котором заполнена ФИО преподавателя, дополнительная информация (если имеется), адрес электронной почты. Кроме того, ниже прикрепляются активные кнопки, по которым можно: получить расписание на сегодня/неделю, а также перейти на персональный сайт преподавателя.

Для того, чтобы не дублировать запросы на сервер по поиску информации о преподавателе, было решено класть в кнопки типа «расписания» callback информацию – идентификатор пользователя и название route.

Через **CallbackQueryHandler** и **Route** сообщение с запросом попадает в **TodayLectorScheduleHandler/WeekLectorScheduleHandler** соответственно,

в котором извлекается идентификатор, дальнейшая обработка происходит также как и расписание текущего пользователя, описанного в пункте 3.2.1.2.

3.2.1.4 Получение новостей

Пользователь может запросить новости за сегодняшний день с помощью специальной команды, пример представлен на рисунке 16. Получение новостей по принципу схоже с поиском преподавателя, описанного в главе 3.2.1.3. Обработка сообщения происходит в классе **GetTodayNewsHandler.**

Новости также предоставляются в формате карточки, с выделенным заголовком и кратким описанием, ниже предоставляется ссылка на полную версию новостей.

Рисунок 16 – Получение новостей за текущее число

3.2.1.5 Общение с единым деканатом

Для общения с единым деканатом была создана специальная группа в Telegram, в которую может вступить ответственный сотрудник, для ответов на вопросы пользователей. Процесс взаимодействия также является двух и более этапным, поэтому есть две возможные реализации: как процесс расписания (3.2.1.2) или как поиск преподавателя (3.2.1.3). Было принято решение реализовывать по второму методу.

На команду **«/message_to_ssc»** сообщение попадает в **WriteMessageToSscHandler**. Так как сообщение может быть прочитано и отвечено не сразу, необходимо предупредить пользователя об этом. Для этого отправляется сообщение с соответствующим текстом. Здесь, как и в части 3.2.1.3 было решено использовать механизм изменения типа сообщения на reply.

Получив сообщение в **ReplyMessageHandler**, и определив, что тип шаблона соответствует ожидаемому, оно переадресовывается в ранее созданную группу, с пояснением от кого отправлено сообщение. Процесс общения представлен на рисунке 17.

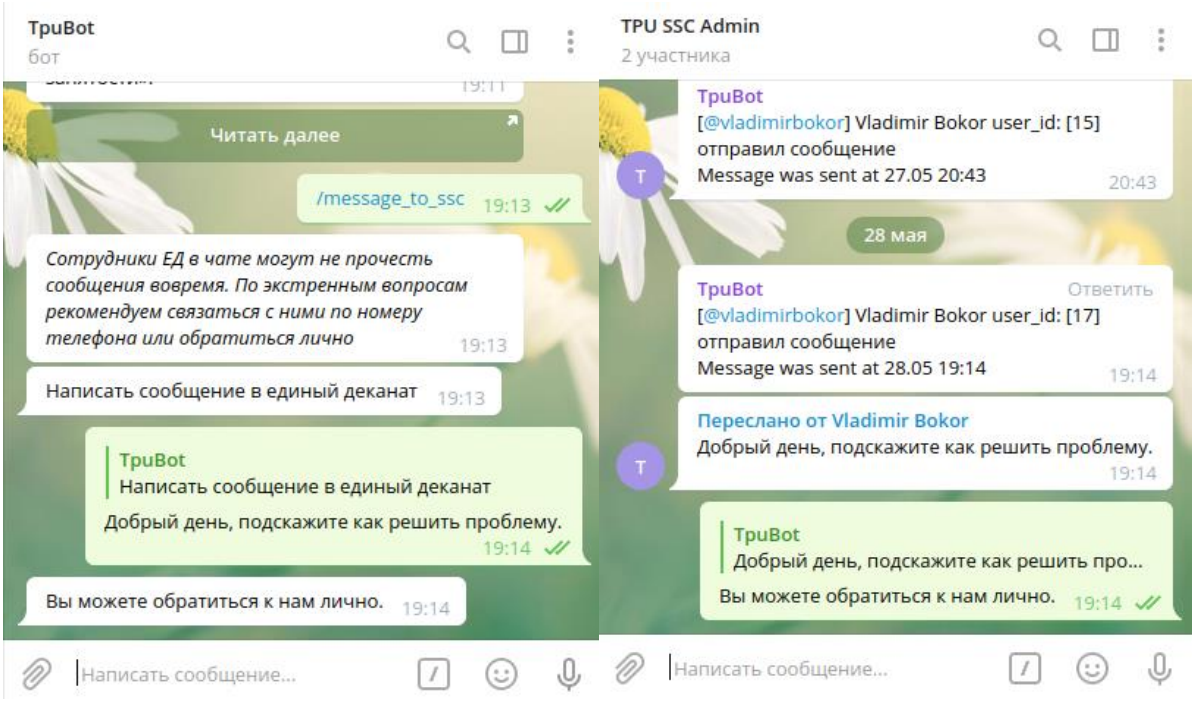


Рисунок 17 – Процесс общения с группой единого деканата

После этого любой человек из группы может ответить на него и бот отправит ответ анонимно. Таким образом, студент не будет знать кто именно ему ответил, но получит необходимую информацию.

Возможность ответа реализована с помощью проверки текста отправляемого сообщения. Группа зарегистрирована в боте через **AdminGroupChat,** в котором определены действия на сообщения из группы. Если отправленное сообщение является текстовым ответом на сообщение бота, тогда оно переадресовывается пользователю, как представлено на рисунке 17.

3.3 Публикация системы

Для возможности доступа пользователей и администраторов в любое время суток к клиенту и серверу, система была опубликована на популярном сервисе **DigitalOcean**, который предоставляет облачные услуги для разработчиков, дает возможность развертывать и масштабировать приложения [75]. В этом сервисе арендуется виртуальный сервер (1 ГБ ОЗУ / 25 ГБ ПЗУ) с предустановленным образом «Ubuntu Docker 5:19.03.1~3 on 18.04», в котором также установлен Docker [76] для развертывания системы. Разработанные части проекта клонируются с сервера GitLab [77] на сервер DigitalOcean. Результат запуска сервера показан на рисунке 18.

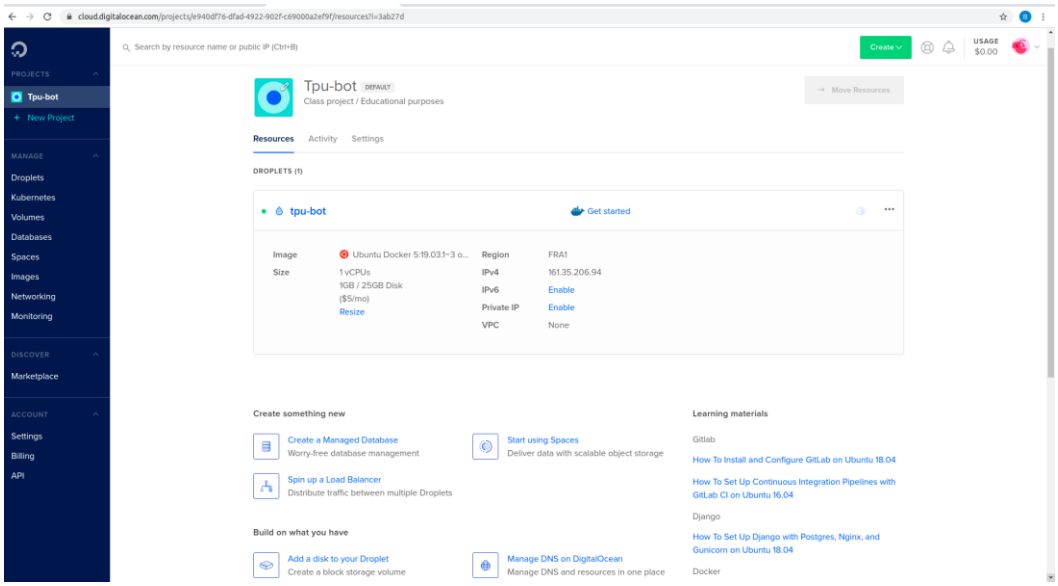


Рисунок 18 – Запущенный сервер с информационной системе в сервисе DigitalOcean.

Как упоминалось выше, для обоих подпроектов (сервер и Telegram клиент) был описан Dockerfile, который необходим для быстрого развертывания готовых программных продуктов с гарантированным сохранением стабильной работы. Основной принцип работы Docker – контейнеризация приложений. Этот тип виртуализации позволяет упаковывать программное обеспечение по изолированным средам – контейнерам. Каждый из этих виртуальных блоков содержит все нужные элементы для работы приложения. Это дает возможность одновременного запуска большого количества контейнеров на одном хосте [78].

Пример описания Dockerfile для серверной части приведен в приложении Д. В нем описываются те команды, которые необходимо выполнить для создания работоспособного окружения. Для того, чтобы работала библиотека dryscrape [66] необходимо установить дополнительные пакеты, которых нет в выбранном основном изображении. Здесь же описываются команды по запуску и установке всех библиотек. Для Telegram бота Dockerfile создается идентично.

Так как в серверной части используется также PostgreSQL [65], было принято решение воспользоваться инструментом Docker Compose, который позволяет создавать и запускать многоконтейнерные приложения, настроив docker-compose.yaml. В нем определяются имена контейнеров, команда для запуска, а также указываются какие разделы необходимо смонтировать в виртуальной среде.

4 Финансовый менеджмент, ресурсоэффективность и ресурсосбережение

Цель раздела – комплексное описание и анализ финансово-экономических аспектов выполненной работы. Проведена оценка полных денежных затрат на проект, а также дана приближенная оценка результатов его внедрения.

4.1 Организация и планирование работ

При организации процесса реализации конкретного проекта необходимо рационально планировать занятость каждого из его участников и сроки проведения отдельных работ. В данном пункте составлен полный перечень проводимых работ по разработке информационной системы уведомления обучающихся в ТПУ, определены их исполнители и рациональная продолжительность.

Перечень работ и продолжительность их выполнения приведены в таблице 1.

Таблица 1 – Перечень работ и продолжительность их выполнения

Этапы работы	**Исполнители**	**Загрузка исполнителей**
Постановка целей и задач, получение данных	НР	НР – 100%
Составление и утверждение ТЗ	НР, И	НР – 100% И – 10%
Подбор и изучение литературы и аналогов по тематике	НР, И	НР – 30% И – 100%
Разработка календарного плана	НР, И	НР – 100% И – 10%
Проектирование информационной системы	НР, И	НР – 30% И – 100%
Получение базы данных	НР, И	НР – 100% И – 10%
Анализ полученных данных	И	И – 100%
Разработка информационной системы	НР, И	НР – 10% И – 100%

Продолжение таблицы 1

Тестирование информационной системы	НР, И	НР – 30% И – 100%
Оформление расчетно-пояснительной записки	И	И – 100%
Подготовка к защите ВКР	НР, И	НР – 30% И – 100%

4.1.1 Продолжительность этапов работ

Расчет продолжительности этапов работ проведен опытно-статическим методом с помощью экспертного способа.

Для определения вероятных (ожидаемых) значений продолжительности работ применена следующая формула:

$$t_{\text{ож}} = \frac{3 \cdot t_{min} + 3 \cdot t_{min}}{5},$$

где t_{min} – минимальная продолжительность работы, дн.;

t_{max} – максимальная продолжительность работы, дн.

Для построения линейного графика была рассчитана длительность этапов в рабочих днях, а затем переведена в календарные дни. Расчет продолжительности выполнения каждого этапа в рабочих днях ($T_{\text{РД}}$) проводился по формуле:

$$T_{\text{РД}} = \frac{t_{\text{ож}}}{K_{\text{ВН}}} \cdot K_{\text{д}},$$

где $t_{\text{ож}}$ – продолжительность работы, дн.;

$K_{\text{ВН}}$ – коэффициент выполнения работ, учитывающий влияние внешних факторов на соблюдение предварительно определенных длительностей, в частности, возможно $K_{\text{ВН}}$ = 1;

$K_{\text{д}}$ – коэффициент, учитывающий дополнительное время на компенсацию непредвиденных задержек и согласование работ ($K_{\text{д}}$ = 1–1,2; в этих границах конкретное значение принимает сам исполнитель).

Расчет продолжительности этапа в календарных днях проводился по формуле:

$$T_{КД} = T_{РД} \cdot T_{К}$$

где $T_{РД}$ – продолжительность выполнения этапа в календарных днях;

$T_{К}$ – коэффициент календарности, позволяющий перейти от длительности работ в рабочих днях к их аналогам в календарных днях, рассчитываемый по формуле для шестидневной рабочей недели:

$$T_{К} = \frac{T_{КАЛ}}{T_{КАЛ} - T_{ВД} - T_{ПД}} = \frac{365}{365 - 52 - 10} = 1{,}205,$$

где $T_{КАЛ}$ – календарные дни ($T_{КАЛ}$= 365);

$T_{ВД}$ – выходные дни ($T_{ВД}$ = 52);

$T_{ПД}$ – праздничные дни ($T_{ПД}$ = 10).

В таблице 2 приведен результат расчета определения продолжительности этапов работ и их трудоемкости по исполнителям, занятым на каждом этапе.

Таблица 2 – Трудозатраты на выполнение проекта

Этап	Исполнители	Продолжительность работ, дни			Трудоемкость работ по исполнителям чел.- дн.			
					$T_{РД}$		$T_{КД}$	
		t_{min}	t_{max}	$t_{ож}$	НР	И	НР	И
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Постановка целей и задач, получение данных	НР	3	5	3,8	4,56	–	5,49	–
Составление и утверждение ТЗ	НР, И	2	3	2,4	2,88	0,29	3,47	0,35
Подбор и изучение литературы и аналогов по тематике	НР, И	10	15	12	4,32	14,40	5,21	17,35
Разработка календарного плана	НР, И	2	4	2,8	3,36	0,34	4,05	0,40
Проектирование информационной системы	НР, И	5	7	5,8	2,09	6,96	2,52	8,39
Получение базы данных	НР, И	7	14	9,8	11,76	1,18	14,17	1,42
Анализ полученных данных	И	5	7	5,8	–	6,96	–	8,39
Разработка информационной системы	НР, И	30	45	36	4,32	43,20	5,21	52,06
Тестирование информационной системы	НР, И	7	14	9,8	3,53	11,76	4,25	14,17
Оформление расчетно-пояснительной записки	И	15	20	17	–	20,40	–	24,58

Продолжение таблицы 2

Подготовка к защите ВКР	НР, И	7	14	9,8	3,53	1,76	4,25	4,17
Итого:				**115**	**40,34**	**117,24**	**48,61**	**141,27**

Величины трудоемкости этапов по исполнителям $T_{КД}$ (данные столбцов 8 и 9) позволяют построить линейный график осуществления проекта – таблица 3.

Таблица 3 – Линейный график работ

Этап	НР	И	Январь			Февраль			Март			Апрель			Май			Июнь	
			10	20	30	40	50	60	70	80	90	100	110	120	130	140	150	160	170
1	5,49	–																	
2	3,47	0,35																	
3	5,21	17,35																	
4	4,05	0,40																	
5	2,52	8,39																	
6	14,17	1,42																	
7	–	8,39																	
8	5,21	52,06																	
9	4,25	14,17																	
10	–	24,58																	
11	4,25	4,17																	

НР – ; И –

4.2 Расчет сметы затрат на выполнение проекта

В состав затрат на создание проекта включается величина всех расходов, необходимых для реализации комплекса работ, составляющих содержание данной разработки. Расчет сметной стоимости ее выполнения производится по следующим статьям затрат:

- материалы и покупные изделия;
- заработная плата;
- социальный налог;
- расходы на электроэнергию (без освещения);
- амортизационные отчисления;
- арендная плата за пользование имуществом;
- прочие (накладные расходы) расходы.

4.2.1 Расчет затрат на материалы

К данной статье расходов относится стоимость материалов, покупных изделий, полуфабрикатов и других материальных ценностей, расходуемых непосредственно в процессе выполнения работ над объектом проектирования. Сюда же относятся специально приобретенное оборудование, инструменты и прочие объекты, относимые к основным средствам, стоимостью до 40 000 руб. включительно.

Сюда же включаются расходы на совершение сделки купли-продажи (т.н. транзакции). Приближенно они оцениваются в процентах к отпускной цене закупаемых материалов, как правило, это 5 ÷ 20 %. Расчет затрат приведен в таблице 4.

Таблица 4 – Расчет затрат на материалы

Наименование материалов	Цена за ед., руб.	Кол-во	Сумма, руб.
Бумага для принтера формата А4	240	1 уп.	240
Катридж для принтера Samsung SCX-3400	1300	1 шт.	1300
Лицензия Jet Brains (студентам)	0	1 шт.	0
Итого:			**1540**

Допустим, что ТЗР составляют 5 % от отпускной цены материалов, тогда расходы на материалы с учетом ТЗР равны $С_{мат}$ = 1540 * 1,05 = 1617 руб.

4.2.2 Расчет заработной платы

Данная статья расходов включает заработную плату научного руководителя и инженера (в его роли выступает исполнитель проекта), а также премии, входящие в фонд заработной платы. Расчет основной заработной платы выполняется на основе трудоемкости выполнения каждого этапа и величины месячного оклада исполнителя.

В 2020 году 366 календарных дней, при шестидневной рабочей недели количество дней отдыха равно 66, следовательно, в месяце среднем 25 рабочих дней. Среднедневная тарифная заработная плата ($ЗП_{дн-т}$) рассчитывается по формуле:

$$ЗП_{дн-т} = \frac{МО}{25}$$

Расчета затрат на полную заработную плату приведен в таблице 5. Затраты времени по каждому исполнителю в рабочих днях с округлением до целого взяты из таблицы 2 . Для учета в ее составе премий, дополнительной зарплаты и районной надбавки используется следующий ряд коэффициентов: $К_{ПР}$ = 1,1; $К_{доп.ЗП}$ = 1,188; $К_{р}$ = 1,3. Таким образом, для перехода от тарифной (базовой) суммы заработка исполнителя, связанной с участием в проекте, к соответствующему полному заработку (зарплатной части сметы) необходимо первую умножить на интегральный коэффициент $К_{и}$ = 1,1* 1,188*1,3 = 1,699.

Таблица 5 – Затраты на заработную плату

Исполнитель	**Оклад, руб./мес.**	**Среднедневная ставка, руб./раб.день**	**Затраты времени, раб.дни**	**Коэффициент**	**Фонд з/платы, руб.**
НР	33 664	1346,56	40	1,699	91 512,28
И	15 470	618,8	117	1,699	123 007
Итого:					214 519,28

4.2.3 Расчет затрат на социальный налог

Затраты на единый социальный налог (ЕСН), включающий в себя отчисления в пенсионный фонд, на социальное и медицинское страхование, составляют 30 % от полной заработной платы по проекту, т.е.

$$C_{\text{соц}} = C_{\text{ЗП}} \cdot 0{,}3 = 214519{,}28 * 0{,}3 = 64355{,}78$$

4.2.4 Расчет затрат на электроэнергию

Данный вид расходов включает в себя затраты на электроэнергию, потраченную в ходе выполнения проекта на работу используемого оборудования, рассчитываемые по формуле:

$$\text{С}_{\text{эл.об}} = P_{\text{об}} \cdot t_{\text{об}} \cdot \text{С}_{\text{э}},$$

где $P_{\text{об}}$ – мощность, потребляемая оборудованием, кВт;

$\text{С}_{\text{э}}$ – тариф на 1 кВт·час;

$t_{\text{об}}$ – время работы оборудования, час.

Для ТПУ $\text{С}_{\text{э}}$ = 6,59 руб./квт·час (с НДС).

Время работы оборудования вычисляется на основе итоговых данных таблицы 2 для инженера ($T_{\text{РД}}$) из расчета, что продолжительность рабочего дня равна 8 часов.

$$t_{\text{об}} = T_{\text{РД}} \cdot K_t,$$

где $K_t \leq 1$ – коэффициент использования оборудования по времени, равный отношению времени его работы в процессе выполнения проекта к $T_{\text{РД}}$, определяется исполнителем самостоятельно.

Мощность, потребляемая оборудованием, определяется по формуле:

$$P_{\text{об}} = P_{\text{НОМ}} \cdot K_C,$$

где $P_{\text{НОМ.}}$ – номинальная мощность оборудования, кВт;

$K_C \leq 1$ – коэффициент загрузки, зависящий от средней степени использования номинальной мощности.

Определим время работы персонального компьютера, принимая коэффициент использования равным 1, так как все работы выполняются за ним:

$$t_{\text{об}} = (117 \cdot 8) \cdot 1 = 936\,,$$

Расчет затрат на электроэнергию для технологических целей приведен в таблице 6.

Таблица 6 – Затраты на технологическую электроэнергию

Наименование оборудования	**Время работы оборудования $t_{об}$, час**	**Потребляемая мощность $P_{об}$, кВт**	**Затраты $С_{эл.об}$, руб.**
Персональный компьютер	936	0,44	2714,03
Лазерный принтер	1	0,31	2,04
Итого:			**2716,07**

4.2.5 Расчет амортизационных расходов

В данной статье представлен расчёт амортизации используемого оборудования за время выполнения проекта по следующей формуле:

$$C_{\text{АМ}} = \frac{H_A \cdot t_{\text{об}} \cdot C_{\text{об}} \cdot n}{F_{\text{Д}}},$$

где H_A – годовая норма амортизации единицы оборудования;

$C_{\text{об}}$ – балансовая стоимость единицы оборудования с учетом ТЗР;

$F_{\text{Д}}$ – действительный годовой фонд времени работы соответствующего оборудования, берется из специальных справочников или фактического режима его использования в текущем календарном году ($F_{\text{Д}} = 300 * 8 = 2400$ ч);

$t_{\text{об}}$ – фактическое время работы оборудования в ходе выполнения проекта, учитывается исполнителем проекта;

n – число задействованных однотипных единиц оборудования.

Расчет для ПК:

$$C_{\text{АМ}} = \frac{0{,}4 \cdot 45000 \cdot 936}{2400} = 7020 \text{ руб}$$

Расчет для принтера:

$$C_{\text{АМ}} = \frac{0{,}5 \cdot 4690 \cdot 1}{300} = 7{,}8 \text{ руб}$$

Итого начислено амортизации 7027,8 руб.

4.2.6 Расчет расходов, учитываемых непосредственно на основе платежных (расчетных) документов (кроме суточных)

Оплата выделенного сервера минимальной мощности для хостинга – 5$ в месяц. Хостинг осуществляется непосредственно в месяц перед защитой. Оплата по текущему курсу доллара равна – 75 * 5 = 375 р/месяц.

4.2.7 Расчет прочих расходов

В статье «Прочие расходы» отражены расходы на выполнение проекта, которые не учтены в предыдущих статьях, их следует принять равными 10% от суммы всех предыдущих расходов, т.е.:

$$C_{\text{проч}} = \left(C_{\text{мат}} + C_{\text{зп}} + C_{\text{соц}} + C_{\text{эл.об}} + C_{\text{ам}} + C_{\text{нп.р}} \right) \cdot 0{,}1$$

$$C_{\text{проч}} = 0{,}1 * (1617 + 214519{,}28 + 64355{,}75 + 2716{,}07 + 7027{,}8 + 375) = 29260{,}76$$

4.2.8 Расчет общей стоимости разработки

Проведя расчет по всем статьям сметы затрат на разработку, можно определить общую себестоимость проекта «Информационная система уведомления обучающихся в ТПУ на платформе Telegram» – таблица 7.

Таблица 7 – Смета затрат на разработку проекта

Статья затрат	**Условное обозначение**	**Сумма, руб.**
Материалы и покупные изделия	$C_{\text{мат}}$	1617
Основная заработная плата	$C_{\text{зп}}$	214 519,28
Отчисления в социальные фонды	$C_{\text{соц}}$	64355,78
Расходы на электроэнергию	$C_{\text{эл.}}$	2716,07
Амортизационные отчисления	$C_{\text{ам}}$	7027,8
Непосредственно учитываемые расходы	$C_{\text{нр}}$	375
Прочие расходы	$C_{\text{проч}}$	29061,09
Итого:		**319672,02**

4.2.9 Расчет прибыли

Прибыль отсутствует, так как результат не подлежит коммерциализации.

4.2.10 Расчет НДС

НДС составляет 20% от суммы затрат на разработку.

$$\text{НДС} = C \cdot 0{,}2 = 319672{,}02 \cdot 0{,}2 = 63934{,}4$$

4.2.11 Цена разработки НИР

Цена равна сумме полной себестоимости и НДС:

$$Ц_{НИР} = C + \text{НДС} = 319672{,}02 + 63934{,}4 = 383606{,}42$$

4.3 Оценка экономической эффективности проекта

Так как проект направлен не на получение прибыли, а на упрощение взаимодействия студентов/преподавателей с различными системами ТПУ – прибыль от внедрения отсутствует.

Для того, чтобы оценить экономический эффект от внедрения данной разработки, необходимо провести отдельное сложное исследование, которое выходит за рамки представленной работы.

5 Социальная ответственность

Выполняемая работа заключалась в проектировании и разработке информационной системы уведомления обучающихся в ТПУ, таким образом, работу можно классифицировать как работу разработчика программного обеспечения.

Разработка осуществлялась в офисе, на территории работодателя (ООО «Хэнд Мейд КОД»), за настольным персональным компьютером. Далее будут рассмотрены факторы рабочей зоны и рабочего места, влияющие на состояние сотрудника.

5.1 Правовые и организационные вопросы обеспечения безопасности

Правовое регулирование трудовых отношений между работодателем, работником и государством регулируется Трудовым кодексом Российской Федерации от 30.12.2001 N 197-ФЗ. В ТК РФ в соответствии с Конституцией РФ, признаются свобода труда, выбор и согласие на него, а также выбор профессии и деятельности [79].

В соответствии со ст. 91 ТК РФ, Нормальная продолжительность рабочего времени не может превышать 40 часов в неделю

В соответствии со ст. 111 ТК РФ Всем работникам предоставляются выходные дни (еженедельный непрерывный отдых). При пятидневной рабочей неделе работникам предоставляются два выходных дня в неделю, при шестидневной рабочей неделе - один выходной день.

В соответствии со ст. 142 ТК РФ, в случае задержки выплаты заработной платы на срок более 15 дней работник имеет право, известив работодателя в письменной форме, приостановить работу на весь период до выплаты задержанной суммы, кроме ряда перечисленных случаев.

В соответствии со ст. 212 ТК РФ, работодатель обязан обеспечить безопасные условия труда, а также обязательное социальное страхование

работников от несчастных случаев на производстве и профессиональных заболеваний.

Для комфортного времяпрепровождения на территории рабочего места, организации необходимо соблюдать ряд правил для офисного помещения и персональных компьютеров, изложенных в трудовом праве российской федерации.

Предъявляемые требования к расположению и компоновке рабочего места:

Высота рабочей поверхности стола для взрослых пользователей должна регулироваться в пределах (680-800) мм, при отсутствии такой возможности высота рабочей поверхности стола должна составлять 725 мм [79].

Модульными размерами рабочей поверхности стола для ПК, на основании которых должны рассчитываться конструктивные размеры, следует считать: ширину 800, 1000, 1200 и 1400 мм, глубину 800 и 1000 мм при нерегулируемой его высоте, равной 725 мм [79].

Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной – не менее 500 мм, глубиной на уровне колен – не менее 450мм и на уровне вытянутых ног – не менее 650 мм [79].

Конструкция рабочего стула должна обеспечивать:

- ширину и глубину поверхности сиденья не менее 400 мм;
- поверхность сиденья с закругленным передним краем;
- регулировку высоты поверхности сиденья в пределах (400 550) мм и углам наклона вперед до 15 град, и назад до 5 град.;
- высоту опорной поверхности спинки (300±20) мм, ширину – не менее 380 мм и радиус кривизны горизонтальной плоскости –400 мм;
- угол наклона спинки в вертикальной плоскости в пределах ±30 градусов;
- регулировку расстояния спинки от переднего края сиденья в пределах (260 400) мм;

- стационарные или съемные подлокотники длиной не менее 250мм и шириной – (50 70) мм;
- регулировку подлокотников по высоте над сиденьем в пределах (230±30) мм и внутреннего расстояния между подлокотниками в пределах (350 500) мм [79].

При выборе компьютеров для сотрудников важным является возможность монитора компьютера изменять положение в различных плоскостях (горизонтальные или вертикальные), с возможной устойчивой фиксацией в положении, которая удобна пользователю. Экран монитора должен содержать регулировку яркости и контрастности, что каждый работник мог установить нужный режим, которые будет соответствовать чувствительности глаз.

- Освещенность на поверхности стола в зоне размещения рабочего документа должна быть 300-500 лк. Освещение не должно создавать бликов на поверхности экрана. Освещенность поверхности экрана не должна быть более 300 лк [79].
- Рабочие столы следует размещать таким образом, чтобы видеодисплейные терминалы были ориентированы боковой стороной к световым проемам, чтобы естественный свет падал преимущественно слева [79].
- Конструкция рабочего стула (кресла) должна обеспечивать поддержание рациональной рабочей позы при работе на ПЭВМ, позволять изменять позу с целью снижения статического напряжения мышц шейноплечевой области и спины для предупреждения развития утомления. Тип рабочего стула (кресла) следует выбирать с учетом роста пользователя, характера и продолжительности работы с ПЭВМ [79].
- Контролируемыми гигиеническими параметрами персональных цифровых ЭВМ (в т.ч. портативных) являются: уровни электромагнитных полей (ЭМП), акустического шума, концентрация вредных веществ в воздухе, визуальные показатели ВДТ, мягкое рентгеновское излучение[79].

Помещение должно быть оборудовано системами вентиляции, кондиционирования и отопления.

5.2 Производственная безопасность

В данном пункте анализируются вредные и опасные факторы, которые могут возникать при проведении исследований в лаборатории, при разработке или эксплуатации проектируемого решения.

5.2.1 Анализ опасных и вредных производственных факторов

Согласно ГОСТ 12.0.003-2015 [81] в таблице 8 представлены возможные вредные и опасные факторы. Работа по разработке программного обеспечения делится на три основных этапа: проектирование, разработка и эксплуатация.

Таблица 8 – Возможные опасные и вредные факторы

Факторы (ГОСТ 12.0.003-2015)	Этапы работ			Нормативные документы
	Проектирование	Разработка	Эксплуатация	
1. Недостаточная освещённость рабочей зоны: отсутствие или недостаток естественного света;	+	+	+	СП 52.13330.2011 СанПиН 2.2.1/2.1.1.1278–03
2. Отклонение показателей микроклимата	+	+	+	СанПиН 2.2.4.548–96 СанПиН 2.2.2/2.4.1340-03
3. Повышенный уровень шума	+	+	+	ГОСТ 12.1.003-2014 ССБТ ГОСТ 12.1.029-80
4. Электрический ток	+	+	+	ГОСТ 12.1.038-82

5.2.2 Недостаточная освещённость рабочей зоны; отсутствие или недостаток естественного света

Освещение рабочего места специалиста складывается из естественного и

искусственного освещения. Естественное освещение достигается установкой оконных проемов с коэффициентом естественного освещения КЕО не ниже 1,2 % в зонах с устойчивым снежным покровом и не ниже 1,5 % на остальной территории [82].

Нормируемые показатели естественного, искусственного и совмещенного освещения в соответствии с СанПиН 2.2.1/2.1.1.1278-03 указаны в таблице 9 [83].

Таблица 9 – Нормируемые показатели естественного, искусственного и совмещенного освещения в соответствии с СанПиН 2.2.1/2.1.1.1278-03

Помещения	Рабочая поверхность и плоскость нормирования КЕО и освещенности (Г – горизонтальная, В – вертикальная) и высота плоскости над полом, м	Естественное освещение		Совмещенное освещение		Искусственное освещение				
		КЕО, %		КЕО, %		Освещенность, лк				
		При верхнем или комбинированном освещении	При боковом освещении	При верхнем или комбинированном освещении	При боковом освещении	При комбинированном освещении		При общем освещении	Показатель дискомфорта, М, не более	Коэффициент пульсации освещенности, %, не более
						Всего	От общего			
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Кабинеты, рабочие комнаты, офисы	Г – 0,8	3,0	1,0	1,8	0,6	400	200	300	40	15
Помещения для работы с дисплеями и видеотерминалами, залы ЭВМ	Г – 0,8 Экран монитора: В – 1,2	3,5 -	1,2 -	2,1 -	0,7 -	500 -	300	400 200	15 -	10

Для искусственного освещения помещений с персональными компьютерами следует применять светильники типа ЛПО36. Допускается применять светильники прямого света, преимущественно отраженного света типа ЛПО13, ЛПО5, ЛСО4, ЛПО34, ЛПО31 с люминесцентными лампами типа ЛБ.

В утреннее и вечернее время вводится общее искусственное освещение. Основными источниками искусственного освещения являются лампы белого света ЛБ-20. Для обеспечения нормируемых значений освещенности по СанПиН 2.2.1/2.1.1.1278-03 в помещениях для работы за ПК следует проводить чистку стекол оконных рам и светильников не реже двух раз в год и проводить своевременную замену перегоревших ламп

Выполним расчет естественного освещения для офисного помещения ООО «Хэнд Мейд КОД». Расчет производится согласно СНиП 23.05-95 «Естественное и искусственное освещение». Помещение имеет размеры 6 х 5 х 2,5 м, в котором установлено окно размером 1,5 х 2 м. Освещение боковое, одностороннее, выделение пыли и других аэрозолей допустимо с концентрацией не более 5 мг/м^3.

Эквивалентная площадь световых проемов может быть рассчитана по формуле:

$$S_{\text{экв}} = N \cdot S_{\text{окн}} = 2 \cdot 1{,}5 = 3 \text{ м}^2.$$

Площадь помещения равна 30 м^2.

Далее также будут применены следующие величины [83, 84]:

а) $n_0 = 9$ – световая характеристика окна, зависящая от глубины помещения, выступа окна и соотношения длин сторон;

б) $K_{\text{ЗД}} = 1{,}2$ – коэффициент, учитывающий уменьшение КЕО от затемнения противостоящим зданием;

в) $r_1 = 3$ – коэффициент, учитывающий повышение КЕО при боковом освещении благодаря свету, отраженному от внутренних поверхностей;

г) t_0 – общий коэффициент светопропускания, вычисляющийся как [85]

$$t_0 = t_1 \cdot t_2 \cdot t_3 \cdot t_4 = 0{,}75 \cdot 0{,}9 \cdot 1 \cdot 1 = 0{,}675,$$

где t_1 – коэффициент светопропускания материала, определяемый по таблице из приложения Е. Для тройного оконного стекла равен 0,75;

t_2 – коэффициент, учитывающий потери света в переплетах, определяемый по таблице из приложения Е, для одинарного металлического переплета равен 0.9;

t_3 – коэффициент, учитывающий потери света в несущих конструкциях, определяемый по таблице из приложения Ж (при боковом освещении $t_3 = 1$);

t_4 – коэффициент, учитывающий потери света в солнцезащитных устройствах, определяемый по таблице из приложения Ж. Для регулируемых жалюзи равен 1.

Рассчитаем фактический коэффициент естественного освещения (КЕО) по формуле:

$$КЕО_ф = \frac{S_{экв} \cdot t_0 \cdot r_1 \cdot 100}{S \cdot n_0 \cdot K_{ЗД}} = \frac{3 \cdot 0{,}675 \cdot 3 \cdot 100}{30 \cdot 9 \cdot 1{,}2} = 1{,}875.$$

Получили, что фактический коэффициент естественного освещения соответствует норме для офисов и для помещений для работы с дисплеями.

Рассчитаем фактическое искусственное освещение. На рассматриваемом рабочем месте основными источниками искусственного освещения являются лампы белого света ЛБ-20 в количестве N = 16 шт. Световой поток одной лампы F = 1180 лм. Коэффициент запаса примем равным k = 1,1, а коэффициент минимальной освещённости z = 1,1.

Индекс помещения определяется по формуле:

$$i = \frac{S}{h \cdot (a + b)},$$

где S – площадь помещения;

a и b –длина и ширина помещения;

h – расчётная высота подвеса светильников над рабочей поверхностью, определяемая по формуле:

$$h = H - h_р - h_с,$$

где H – высота потолка в помещении, м;

$h_р$ – расстояние от пола до рабочей поверхности стола, 1 м;

$h_с$ – расстояние от потолка до светильника, 0,2 м.

Индекс помещения равен:

$$i = \frac{30}{(2{,}5 - 1 - 0{,}2) \cdot (5 + 6)} = 2{,}09.$$

Зная индекс помещения, найдем коэффициент использования светового потока. Примем коэффициенты отражения от стен ρ_c=70%, потолка $\rho_п$=50%, пола $\rho_р$=30%. Для ЛБ20 тип кривой силы света – косинусный, выбираем из приложения И или [86] n = 0.83.

Фактическое искусственное освещение рассчитывается по формуле:

$$E = \frac{F \cdot N \cdot n}{S \cdot z \cdot k} = \frac{1180 \cdot 16 \cdot 0{,}83}{30 \cdot 1{,}1 \cdot 1{,}1} = 431{,}7 \text{ лк}.$$

Полученное значение снова удовлетворяет как нормам для кабинетов и офисов (300 лк), так и нормам для помещений для работы с дисплеями (400 лк).

5.2.3 Отклонение показателей микроклимата

Показатели микроклимата должны обеспечивать сохранение теплового баланса человека с окружающей средой и поддержание оптимального или допустимого теплового состояния организма. Отклонение показателей от нормы в пределах допустимых величин могут вызвать локальные ощущения теплового дискомфорта и напряжение механизмов терморегуляции.

Согласно СанПиН 2.2.4.548-96 [87], санитарные правила устанавливают гигиенические требования к показателям микроклимата рабочих мест производственных помещений с учетом интенсивности энерготрат работающих, времени выполнения работы, периодов года и содержат требования к методам измерения и контроля микроклиматических условий.

Оптимальные микроклиматические условия установлены по критериям оптимального теплового и функционального состояния человека. Они обеспечивают общее и локальное ощущение теплового комфорта в течение 8-часовой рабочей смены при минимальном напряжении механизмов терморегуляции, не вызывают отклонений в состоянии здоровья, создают предпосылки для высокого уровня работоспособности и являются предпочтительными на рабочих местах. Категория работ определена, как

наименее энергозатратная, т.е. не сопровождающаяся какой-либо физической нагрузкой (сидячий вид деятельности, умственный труд) – 1а. Оптимальные величины показателей микроклимата представлены в таблице 10.

Таблица 10 – Оптимальные величины показателей микроклимата на рабочих местах производственных помещений

Период года	Кат. работ	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Теплый	1а	22-24	21-25	60-40	0,1
Холодный	1а	23-25	22-26	60-40	0,1

Для определения температуры воздуха в помещении использовался настенный термометр. Температуры воздуха составила около 24 °С. Для измерения относительной влажности воздуха использовался гигрометр механического типа. Показания гигрометра на момент измерения составили 60%.

Можно сделать вывод, что микроклимат на рабочем месте является оптимальным для работы в данное время года.

5.2.4 Повышенный уровень шума

Превышение уровня шума является распространенным вредным фактором на рабочем месте. Его нарушение влечет за собой негативное воздействие не только на органы слуха, но и на весь организм человека в целом, через центральную нервную систему.

При выполнении работ, описанных выше, специалист может оказаться под шумовым воздействием со стороны оборудования, находящегося в рабочем помещении: персональные компьютеры, осветительные приборы дневного света, а также шумы, проникающие извне.

Выполняемые работы оцениваются как научная деятельность, конструирование и проектирование, программирование, следовательно,

согласно СН2.2.4/2.1.8.562-96 эквивалентный уровень шума в рабочем помещении недолжен превышать 50дБА (таблица 11).

Таблица 11 – Эквивалентные уровни звука для проектно-конструкторских бюро, лабораторий для теоретических работ

Вид трудовой деятельности, рабочее место	Эквивалентные уровни шума, дБА
Творческая деятельность, руководящая работа с повышенными требованиями, научная деятельность, конструирование и проектирование, программирование, преподавание и обучение, врачебная деятельность. Рабочие места в помещениях дирекции, проектно-конструкторских бюро, расчетчиков, программистов вычислительных машин, в лабораториях для теоретических работ и обработки данных, приема больных в здравпунктах	50

Снижению уровня шума способствует установка звукопоглощающих материалов (плиты, панели), подвесных акустических потолков, а также установка малошумного оборудования.

5.2.5 Электрический ток

Среди распространенных опасностей в рабочей зоне находится и поражение электрическим током. Опасность поражения определяется величиной тока проходящего через тело человека I или напряжением прикосновения U. Напряжение считается безопасным при напряжении прикосновения U < 42 В.

При получении человеком разряда электрического тока могут быть получены электротравмы, электрические удары и даже летальный исход (согласно ГОСТ 12.1.009-2009 [88]).

Для защиты от поражения электрическим током следует выполнить следующие пункты:

- обеспечить недоступность токоведущих частей от случайных прикосновений;
- электрическое разделение цепей;
- устранить опасность поражения при появлении напряжения на разных частях.

Токи статического электричества, наведенные в процессе работы компьютера на корпусах монитора, системного блока и клавиатуры, могут приводить к разрядам при прикосновении к этим элементам. Такие разряды опасности для человека не представляют, но могут привести к выходу из строя вышеописанного оборудования.

На рабочем месте пользователя размещены дисплей, клавиатура и системный блок. Перед началом работы следует убедиться в отсутствии свешивающихся со стола или висящих под столом проводов электропитания, в целостности вилки и провода электропитания, в отсутствии видимых повреждений аппаратуры и рабочей мебели, в отсутствии повреждений и наличии заземления приэкранного фильтра.

Методы защиты от воздействия статического электричества:

- влажная уборка, чтобы уменьшить количество пылинок в воздухе и на предметах офиса;
- использование увлажнителей воздуха;
- защитное заземление;
- применение средств индивидуальной защиты, таких как
- антистатические спреи и браслеты.

5.3 Экологическая безопасность

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

В случае выхода из строя ПК, они списываются и отправляются на специальный склад, который при необходимости принимает меры по утилизации списанной техники и комплектующих [89].

В нормативном документе СанПиН 2.2.2/2.4.1340-03 [79], даются следующие общие рекомендации по снижению опасности для окружающей среды, исходящей от компьютерной техники:

- применять оборудование, соответствующее санитарным нормам и стандартам экологической безопасности;
- применять расходные материалы с высоким коэффициентом использования и возможностью их полной или частичной регенерации;
- отходы в виде компьютерного лома утилизировать;
- использовать экономичные режимы работы оборудования.

Одним из самых распространенных источников ртутного загрязнения являются вышедшие из эксплуатации люминесцентные лампы. Каждая такая лампа, кроме стекла и алюминия, содержит около 60 мг ртути. Поэтому отслужившие свой срок люминесцентные лампы, а также другие приборы, содержащие ртуть, представляют собой опасный источник токсичных веществ [89].

5.4 Безопасность при чрезвычайных ситуациях

5.4.1 Пожарная безопасность

Самым распространенным чрезвычайным обстоятельством в офисе является пожар. Такое рабочее место относится к категории “В” (пожароопасные), так как в данном помещении присутствует пыль, вещества и материалы, способные при взаимодействии с воздухом гореть. Возникновение пожара может произойти по нескольким факторам:

- возникновением короткого замыкания в электропроводке вследствие неисправности самой проводки или электросоединений и электрораспределительных щитов;
- возгоранием устройств вычислительной аппаратуры вследствие нарушения изоляции или неисправности самой аппаратуры;
- возгоранием мебели или пола по причине нарушения правил пожарной безопасности, а также неправильного использования дополнительных бытовых электроприборов и электроустановок;
- возгоранием устройств искусственного освещения.

Для локализации или ликвидации загорания на начальной стадии используются первичные средства пожаротушения:

- огнетушащие вещества (вода, песок, земля);
- огнетушащие материалы (грубошерстные куски материи – кошмы, асбестовые полотна, металлические сетки с малыми ячейками ит. п.);
- немеханизированный ручной пожарный инструмент (багры, крюки, ломы, лопаты и т. п.);
- пожарный инвентарь (бочки и чаны с водой, пожарные ведра, ящики и песочницы с песком);
- пожарные краны на внутреннем водопроводе противопожарного водоснабжения в сборе с пожарным стволом и пожарным рукавом;
- огнетушители [82].

5.5 Выводы

В ходе выполнения раздела были оценены различные вопросы безопасности (правовая, производственная, экологическая, безопасность ЧС) проведен анализ опасных и вредных производственных факторов, в частности расчет освещенности на рабочем месте. Выявлено, что основные показатели соответствуют стандартам.

Заключение

В данной работе поставлен вопрос о создании информационной системы обучающихся ТПУ. Спроектирована и разработана архитектура программного комплекса, состоящего из Серверной части и Клиентской части. Описаны и разработаны ключевые модули для сервера с использованием фреймворка Django и Django Rest Framework, СУБД – PostgreSQL. Клиент для мессенджера Telegram разработан с использованием языка программирования Python и фреймворка Telepot.

В связи с тем, что данные, которые можно показать конечному пользователю, не были предоставлены, были описаны несколько скриптов для веб-парсинга. Часть извлеченных данных была сохранена заранее, часть скриптов работает в «онлайн» режиме.

Информационная система опубликована в Интернете.

В качестве дальнейшего развития данной системы возможны следующие шаги реализации.

- Интеграция с платформой онлайн обучения Moodle. Разработанная система позволяет отправлять студенту лекции, а также присылать уведомления. Также пользователь может загружать документы, которые будут пересланы в Moodle.
- Реализация клиентов на других платформах. Одним из самых лучших вариантов для дальнейшего развития является ВКонтакте.
- Предоставления новых возможностей для пользователей после интеграции разработанной системы с основной базой данных ТПУ.
- Ускорение отклика на команды, за счет интеграции с основной базой данных ТПУ.
- Реализация созданной системы в других ВУЗах.

Список публикаций студента

1. В.А. Бокор. Чат-бот студента ТПУ // XVII Международная научно-практическая конференция студентов, аспирантов и молодых ученых «Молодежь и современные информационные технологии», 17-20 февраля 2020 г., ТПУ, Томск.

2. В.А. Бокор. Чат-бот студента ТПУ // VIII региональная научно-практическая конференция «Наука и практика: проектная деятельность от идеи до внедрения», ноябрь 2019 г., ТУСУР, Томск

Список использованных источников

1. Концепция развития единой информационной образовательной среды в Российской Федерации [Электронный ресурс]. – Режим доступа: http://minobr.govmurman.ru/opencms/export/sites/minobr/.content/docs/GLAS/koncepciya_eios.pdf (дата обращения: 20.05.2020).

2. Указ П. Р. Ф. О Стратегии развития информационного общества в Российской Федерации на 2017-2030 годы» от 09. 05.2017 г. № 203 [Электронный ресурс] //Режим доступа: http://www. kremlin. ru/acts/bank/41919 (дата обращения: 20.05.2020).

3. Попова О. И. Трансформация высшего образования в условиях цифровой экономики //Вопросы управления. – 2018. – №. 5 (35).

4. Аббакумов А. А., Сидоров Д. П., Егунова А. И. Использование мессенджеров для информирования слушателей учебных заведений //Образовательные технологии и общество. – 2018. – Т. 21. – №. 3.

5. «Мой Универ» стал платформой для образовательных чат-ботов на EdHack: ChatBots [Электронный ресурс]. – Режим доступа: http://sk.ru/news/b/news/archive/2016/09/13/moy-univer-vystupila-platformoy-dlya-obrazovatelnyh-chatbotov-na-edhack-chatbots.aspx (Дата обращения: 01.10.2019).

6. Чат-бот с новостями и расписанием появился в Государственном университете управления [Электронный ресурс]. – Режим доступа: https://guu.ru/news_ru/55084/ (Дата обращения: 02.10.2019).

7. Бот расскажет студентам Университета ИТМО о начисленных баллах, расписании и личном рейтинге [Электронный ресурс]. – Режим доступа: http://news.ifmo.ru/ru/education/students/news/7276/(Дата обращения: 02.10.2019).

8. Laven, S. (1996). Claude - by brian mclaughlin. Retrieved from The Simon Laven Page: https://www.simonlaven.com/claude.htm.

9. Laurillard D. Rethinking university teaching: A conversational framework for the effective use of learning technologies. – Routledge, 2013.

10. Smutny P., Schreiberova P. Chatbots for learning: A review of educational chatbots for the Facebook Messenger //Computers & Education. – 2020. – С. 103862.

11. Schmulian A., Coetzee S. A. The development of Messenger bots for teaching and learning and accounting students' experience of the use thereof //British Journal of Educational Technology. – 2019. – Т. 50. – №. 5. – С. 2751-2777.

12. Рунет подвел итоги года: объем экономики Рунета составил 4,7 трлн рублей [Электронный ресурс]. – Режим доступа: https://raec.ru/live/raec-news/11400/ (дата обращения: 21.05.2020).

13. Типы устройств в России [Электронный ресурс]. – Режим доступа: https://radar.yandex.ru/device_categories (дата обращения: 22.05.2020).

14. Аудитория интернета в России выросла на 7% за три года [Электронный ресурс] / 2019. – Режим доступа: https://mediascope.net/news/1035826/?sphrase_id=203276 (Дата обращения: 03.10.2019).

15. Аудитория интернета в России выросла на 4% [Электронный ресурс] / 2018. – Режим доступа: https://mediascope.net/news/805084/https://mediascope.net/news/1035826/?sphrase_id=203276 (Дата обращения: 03.10.2019).

16. O'Reilly T. What Is Web 2.0 [Электронный ресурс]. – Режим доступа: http://www.oreilly.com/pub/a/web2/archive/what-is-web-20.html (Дата обращения: 03.10.2019).

17. Социальные сети сегодня, осень 2019. Цифры и тренды [Электронный ресурс]. – Режим доступа: http://files.runet-id.com/2019/itogi/itogi19-2--cherniy.pdf (Дата обращения: 23.05.2020).

18. Официальный сайт WhatsApp [Электронный ресурс] / – Режим доступа: https://www.whatsapp.com/ (Дата обращения: 04.10.2019).

19. Number of monthly active WhatsApp users worldwide from April 2013 to March 2020 [Электронный ресурс]. – Режим доступа: https://www.statista.com/statistics/260819/number-of-monthly-active-whatsapp-users/ (Дата обращения: 23.05.2020).

20. WhatsApp Business API [Электронный ресурс] / – Режим доступа: https://www.whatsapp.com/business/api (Дата обращения: 04.10.2019).

21. В России появились легальные боты для WhatsApp [Электронный ресурс]. – Режим доступа: http://tdaily.ru/news/2019/07/30/v-rossii-poyavilis-legalnye-boty-dlya-whatsapp (Дата обращения: 23.05.2020).

22. Официальный сайт VK [Электронный ресурс] / – Режим доступа: https://vk.com/about (Дата обращения: 04.10.2019).

23. API для чат-ботов [Электронный ресурс] / – Режим доступа: https://vk.com/dev/bots_docs (Дата обращения: 04.10.2019).

24. Боты для сообществ [Электронный ресурс] / – Режим доступа: https://vk.com/dev/bots (Дата обращения: 04.10.2019).

25. Правила платформы [Электронный ресурс] / – Режим доступа: https://vk.com/dev/rules (Дата обращения: 04.10.2019).

26. Официальный сайт Telegram [Электронный ресурс] / – Режим доступа: https://telegram.org/(Дата обращения: 04.10.2019).

27. Официальный сайт документации Telegram [Электронный ресурс] / – Режим доступа: https://core.telegram.org/ (Дата обращения: 04.10.2019).

28. 400 миллионов пользователей, 20 000 стикеров, улучшенные опросы и €400K для авторов образовательных тестов [Электронный ресурс] / – Режим доступа: https://telegram.org/blog/400-million/ru?ln=a (Дата обращения: 25.05.2020).

29. Коноплев Д.И. Telegram как новая среда коммуникации в СМИ и соцсетях // Знак: проблемное поле медиаобразования. 2017. №3 (25). С.198-200.

30. Иванов А.Д. Чат-бот в Telegram и ВКонтакте как новый канал распространения новостей // Вестник ВУиТ. 2016. №3. С.126-132.

31. Constine J. 2.5 billion people use at least one of Facebook’s apps //Techcrunch. http://social.techcrunch.com/2018/07/25/facebook-2-5-billion-people. – 2018.

32. Facebook Messenger passes 300,000 bots [Электронный ресурс] / – Режим доступа: https://venturebeat.com/2018/05/01/facebook-messenger-passes-300000-bots/ (Дата обращения: 25.05.2020).

33. Most popular mobile messaging apps worldwide as of October 2019, based on number of monthly active users [Электронный ресурс] / – Режим доступа: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/ (Дата обращения: 26.05.2020).

34. Alexander M. G., Nganga G. Introduction: Importance of marine concrete structures and durability design //Marine Concrete Structures. – Woodhead Publishing, 2016. – С. 1-13.

35. Гудлиф П. Ремесло программиста. Практика написания хорошего кода //Пер. с англ.-СПб.: Символ Плюс. – 2009.

36. REST principles [Электронный ресурс] / – Режим доступа: https://ninenines.eu/docs/en/cowboy/2.7/guide/rest_principles/ (Дата обращения: 30.04.2020).

37. 10 Most Popular Web Frameworks in 2020 https://medium.com/front-end-weekly/10-most-popular-web-frameworks-in-2020-167b9103e08a (Дата обращения: 30.04.2020).

38. Django documentation [Электронный ресурс] / – Режим доступа: https://docs.djangoproject.com/en/2.2/ (дата обращения: 17.05.2020).

39. Django. The Web framework for perfectionists with deadlines [Электронный ресурс] / – Режим доступа: https://stackshare.io/django (дата обращения: 17.05.2020).

40. Easiest Programming Languages to Learn in 2020 – 17 Easy Coding Languages [Электронный ресурс] / – Режим доступа: https://careerkarma.com/blog/easiest-programming-languages-to-learn/ (дата обращения: 17.05.2020).

41. Spring Framework [Электронный ресурс] / – Режим доступа: https://spring.io/projects/spring-framework (дата обращения: 18.05.2020).

42. What is a POJO Class? [Электронный ресурс] / – Режим доступа: https://www.baeldung.com/java-pojo-class (дата обращения: 18.05.2020).

43. Java spring framework – pros, cons, common mistakes [Электронный ресурс] / – Режим доступа: https://skywell.software/blog/java-spring-framework-pros-cons-mistakes/ (дата обращения: 18.05.2020).

44. Advantages of Spring Framework With Limitations [Электронный ресурс] / – Режим доступа: https://data-flair.training/blogs/advantages-of-spring/ (дата обращения: 18.05.2020).

45. Rails Official Site [Электронный ресурс] / – Режим доступа: https://rubyonrails.org/ (дата обращения: 20.05.2020).

46. Rails. Web development that doesn't hurt [Электронный ресурс] / – Режим доступа: https://stackshare.io/rails (дата обращения: 20.05.2020).

47. RubyGems [Электронный ресурс] / – Режим доступа: https://rubygems.org/stats (дата обращения: 20.05.2020).

48. Which Framework to Choose: Comparing Django and Ruby on Rails [Электронный ресурс] / – Режим доступа: https://medium.com/better-programming/which-framework-to-choose-comparing-django-vs-ruby-on-rails-21e40d92d981 (дата обращения: 21.05.2020).

49. Laravel Framework [Электронный ресурс] / – Режим доступа: https://laravel.com/docs/7.x (дата обращения: 21.05.2020).

50. Laravel. A PHP Framework For Web Artisans [Электронный ресурс] / – Режим доступа: https://stackshare.io/laravel (дата обращения: 21.05.2020).

51. Pros and Cons of Laravel [Электронный ресурс] / – Режим доступа: https://nimapinfotech.com/blog/pros-and-cons-of-laravel/ (дата обращения: 29.05.2020).

52. Advantages and Disadvantages of Laravel [Электронный ресурс] / – Режим доступа: https://medium.com/@saschathattil/advantages-and-disadvantages-of-laravel-224ecc09021a (дата обращения: 29.05.2020).

53. Python Official Site [Электронный ресурс] / – Режим доступа: https://www.python.org/about/ (дата обращения: 29.05.2020).

54. Зачем нужен Python Global Interpreter Lock и как он работает [Электронный ресурс] / – Режим доступа: https://tproger.ru/translations/global-interpreter-lock-guide/ (дата обращения: 27.05.2020).

55. Java Programming Language [Электронный ресурс] / – Режим доступа: https://docs.oracle.com/javase/7/docs/technotes/guides/language/ (дата обращения: 27.05.2020).

56. Where is Java used in Real World? [Электронный ресурс] / – Режим доступа: https://javarevisited.blogspot.com/2014/12/where-does-java-used-in-real-world.html (дата обращения: 27.05.2020).

57. Java Pros And Cons [Электронный ресурс] / – Режим доступа: https://wiki.c2.com/?JavaProsAndCons (дата обращения: 27.05.2020).

58. Pros and Cons of Java [Электронный ресурс] / – Режим доступа: https://www.prosancons.com/computer/pros-and-cons-of-java/ (дата обращения: 27.05.2020).

59. Stroustrup B. The C++ programming language. – Pearson Education, 2013.

60. Holzner S. C++ Programming Black Book+ With CD. – Dreamtech Press, 2001.

61. 10 Best Programming Languages to Learn in 2020 (for Job & Future) [Электронный ресурс] / – Режим доступа: https://hackr.io/blog/best-programming-languages-to-learn-2020-jobs-future (дата обращения: 22.05.2020).

62. Telepot Introduction [Электронный ресурс] / – Режим доступа: https://telepot.readthedocs.io/en/latest/ (дата обращения: 27.05.2020).

63. Beautiful Soup Documentation [Электронный ресурс] / – Режим доступа: https://www.crummy.com/software/BeautifulSoup/bs4/doc/ (дата обращения: 26.05.2020).

64. Providing initial data for models [Электронный ресурс] / – Режим доступа: https://docs.djangoproject.com/en/2.2/howto/initial-data/ (дата обращения: 29.05.2020).

65. PostgreSQL: The World's Most Advanced Open Source Relational Database [Электронный ресурс] / – Режим доступа: https://www.postgresql.org/ (дата обращения: 29.05.2020).

66. Dryscrape’s documentation [Электронный ресурс] / – Режим доступа: https://dryscrape.readthedocs.io/en/latest/ (дата обращения: 29.05.2020).

67. Django Rest Framework [Электронный ресурс] / – Режим доступа: https://www.django-rest-framework.org/ (дата обращения: 29.05.2020).

68. Django Rest Framework. Permissions [Электронный ресурс] / – Режим доступа: https://www.django-rest-framework.org/api-guide/permissions/#permissions (дата обращения: 27.05.2020).

69. What is Amazon SNS? [Электронный ресурс] / – Режим доступа: https://docs.aws.amazon.com/sns/latest/dg/welcome.html (дата обращения: 29.05.2020).

70. What is Amazon Simple Queue Service? [Электронный ресурс] / – Режим доступа: https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html (дата обращения: 29.05.2020).

71. Boto3 documentation [Электронный ресурс] / – Режим доступа: https://boto3.amazonaws.com/v1/documentation/api/latest/index.html (дата обращения: 27.05.2020).

72. Asynchronous Message Processing [Электронный ресурс] / – Режим доступа: https://docs.microsoft.com/en-us/previous-versions/7ch3stsw(v=vs.100)?redirectedfrom=MSDN (дата обращения: 14.05.2020).

73. Throttling Async Functions in Python Asyncio [Электронный ресурс] / – Режим доступа: https://stackoverflow.com/questions/45440900/throttling-async-functions-in-python-asyncio/45502319#45502319 (дата обращения: 14.05.2020).

74. Coroutines and Tasks [Электронный ресурс] / – Режим доступа: https://docs.python.org/3.7/library/asyncio-task.html (дата обращения: 14.05.2020).

75. Meet DigitalOcean [Электронный ресурс] / – Режим доступа: https://www.digitalocean.com/about/ (дата обращения: 29.05.2020).

76. Docker Official Site [Электронный ресурс] / – Режим доступа: https://www.docker.com/ (дата обращения: 29.05.2020).

77. Gitlab Official Site [Электронный ресурс] / – Режим доступа: https://gitlab.com/ (дата обращения: 29.05.2020).

78. Что такое Docker и как его использовать в разработке [Электронный ресурс] / – Режим доступа: https://eternalhost.net/blog/razrabotka/chto-takoe-docker (дата обращения: 29.05.2020).

79. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 24.04.2020).

80. СанПиН 2.2.2/2.4.1340 – 03. Санитарно-эпидемиологические правила и нормативы «Гигиенические требования к персональным электронно-вычислительным машинам и организации работы». – М.: Госкомсанэпиднадзор, 2003.

81. ГОСТ 12.0.003-2015 Система стандартов безопасности труда (ССБТ). Опасные и вредные производственные факторы. Классификация.

82. Безопасность жизнедеятельности. Учебник. Под ред. Э.А. Арустамова / 10-е изд., перераб. и доп. — М.: Изд-во «Дашков и К°», 2006. — 476 с.

83. СанПиН 2.2.1/2.1.1.1278-03 «Гигиенические требования к естественному, искусственному и совмещенному освещению жилых и общественных зданий».

84. СП 52.13330.2011 Свод правил. Естественное и искусственное освещение.

85. СП 23-102-2003 Естественное освещение жилых и общественных зданий.

86. Справочная книга для проектирования электрического освещения / Под ред. Г.М. Кнорринга.- М.: Энергия, 1976. - 384 с.

87. СанПиН 2.2.4.548–96. Гигиенические требования к микроклимату производственных помещений.

88. ГОСТ 12.1.009-2009. Система стандартов безопасности труда. Электробезопасность.

89. ГОСТ 17.4.3.04-85 «Охрана природы. Почвы. Общие требования к контролю и охране от загрязнения»

Приложение А
(обязательное)

Скрипт получения расписания на неделю

```
import datetime

import dryscrape
import requests
from bs4 import BeautifulSoup

BASE_URL = "https://rasp.tpu.ru"


def get_current_week_schedule(identifier):
    today = datetime.date.today()
    _, week_num, _ = today.isocalendar()
    tpu_week = week_num + 17  # TODO: Sync weeks
    dryscrape.start_xvfb()
    sess = dryscrape.Session()
    url = f"{BASE_URL}/{identifier}/2019/{tpu_week}/view.html"
    sess.visit(url)
    source = sess.body()
    soup = BeautifulSoup(source, 'lxml')
    data = get_week_data(soup)
    return data


def get_week_data(soup):
    times, schedule = get_schedule_and_times(soup)
    week_schedule = []
    today = datetime.date.today()
    dates = [today + datetime.timedelta(days=i) for i in
             range(0 - today.weekday(), 7 - today.weekday())]
    for i, day in enumerate(schedule):
        lessons = [elem.text.strip() for elem in day]
        week_schedule.append(
            {"date": dates[i],
             "lessons": lessons}
        )
    return {
        "times": times,
        "week_schedule": week_schedule
    }


def get_schedule_and_times(soup):
    table_body = soup.find('tbody')
    times, schedule = parse_rasp_table(table_body)
    parsed_times = [elem.get('title') for elem in times]
    return parsed_times, schedule


def parse_rasp_table(table_body):
    rows = table_body.find_all('tr')
    times = []
    columns = []
    free_day_indexes = []
    for i, row in enumerate(rows):
        cols = row.find_all('td')
        time = cols.pop(0)
        if i == 0:
```

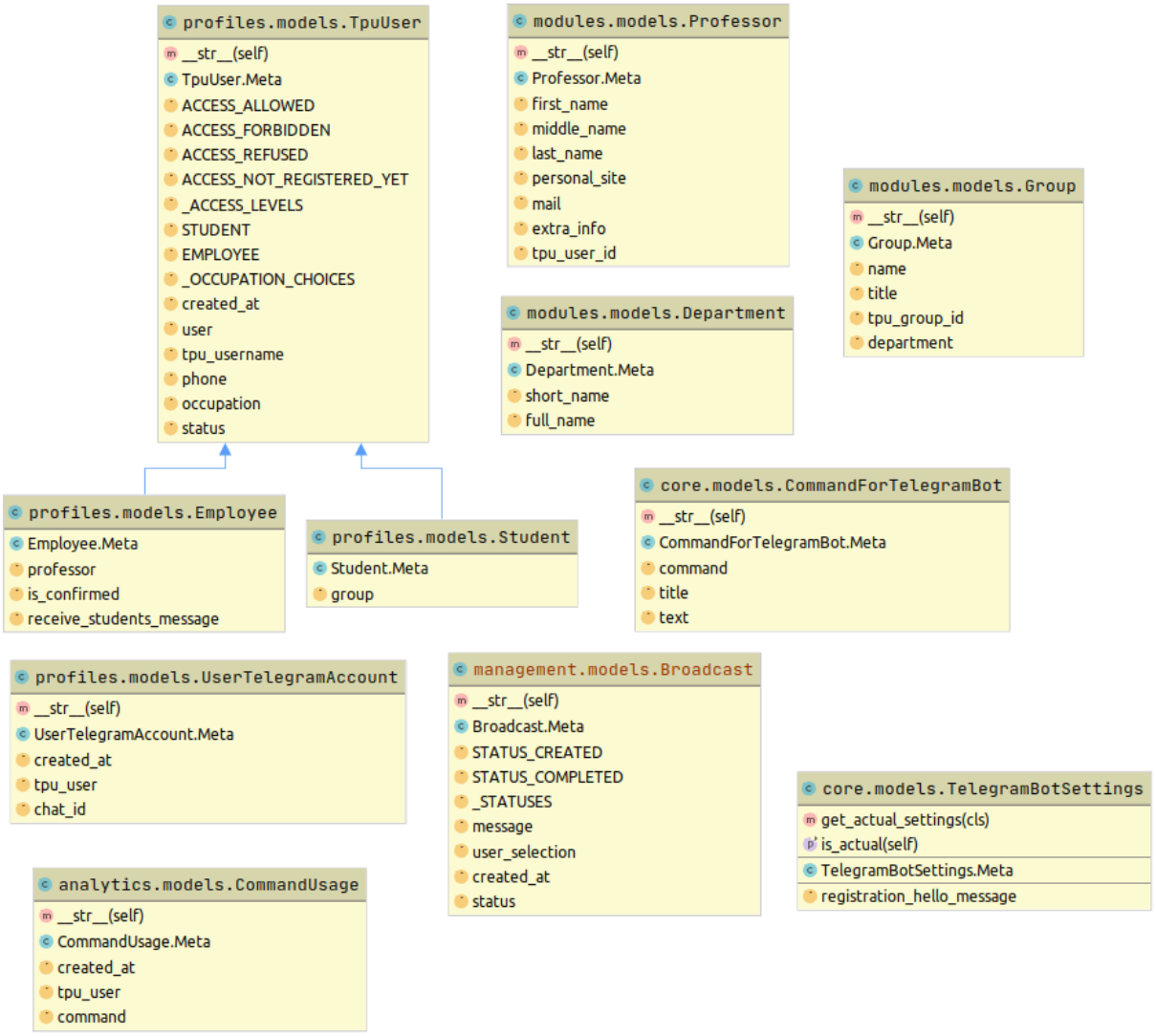
```
            free_day_indexes += [
                idx for idx, element in enumerate(cols) if is_free_day(element)
            ]
            for index in reversed(free_day_indexes):
                del cols[index]
        times.append(time)
        columns.append(cols)
    # transponse schedule to read by line
    schedule = list(map(list, zip(*columns)))
    if free_day_indexes:
        free_day = make_free_day()
        for index in free_day_indexes:
            schedule.insert(index, free_day)
    return times, schedule


def is_free_day(col):
    return "free-day" in col.get("class")


def make_free_day():
    FREE_DAY = '''
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>,
    <td class="cell hidden-xs"></td>'''
    soup = BeautifulSoup(FREE_DAY, 'lxml')
    free_day = soup.find_all("td")
    return free_day
```

Приложение Б
(обязательное)

Диаграмма моделей классов



Приложение В
(обязательное)

Класс очереди сообщений в сервере

```
import decimal
import json
import logging

import boto3
from django.conf import settings

logger = logging.getLogger(__name__)

ENABLE_MESSAGING = settings.ENABLE_MESSAGING
AWS_MESSAGING_REGION = settings.AWS_MESSAGING_REGION
SNS_TOPIC_NAME = settings.SNS_TOPIC_NAME

NOTIFICATION_TEST = 'notifications:test'
NEW_MANAGEMENT_BROADCAST = 'global:new_management_broadcast'

class MessagesQueue(object):
    def __init__(self):
        self.enabled = ENABLE_MESSAGING
        self.connected = False
        self.error = ''
        self.client = None
        self.topic_name = None

    def connect(self):
        if not self.enabled or self.connected:
            return True

        try:
            self.client = boto3.client('sns', region_name=AWS_MESSAGING_REGION)
            self.topic_name = self._get_topic_name(SNS_TOPIC_NAME)
            self.connected = True
            self.error = ''
            return self.connected
        except StopIteration as ex:
            message = \
                ('Cannot find topic name %s for AWS account in %s region.' +
                 'Create topic with name %s and try again.') % (
                    SNS_TOPIC_NAME, AWS_MESSAGING_REGION, SNS_TOPIC_NAME
                )
            self.error = message
            logger.error(message)
            return False
        except Exception as ex:
            self.error = 'Unknown error while connecting'
            logger.error(ex)
            return False

    def send_message(self, message):
        if not self.enabled or not self.connected:
            return None

        data = json.dumps(message) \
            if type(message) != str else message

        self.client.publish(TopicArn=self.topic_name, Message=data)
```

```
    def _get_topic_name(self, name):
        response = self.client.list_topics()
        topic_name = next(
            t['TopicArn'] for t in response['Topics']
            if name in t['TopicArn']
        )

        return topic_name


queue = MessagesQueue()


def create_new_broadcast_message(broadcast):
    from management.serializers import BroadcastSerializer
    serializer = BroadcastSerializer(broadcast)

    return {
        'type': NEW_MANAGEMENT_BROADCAST,
        'payload': serializer.data
    }
```

Приложение Г
(обязательное)

Класс BaseHandler

```
import telepot
import json


class Handler(object):
    create_user_if_not_exists = False
    user = None

    def __init__(self, request, *args, **kwargs):
        self.request = request
        self.message = request.message
        self.chat = request.chat
        self.admin_chat = self.chat.core_bot.admin_chat
        self.args = args
        self.kwargs = kwargs

        if self.chat.user:
            self.user = self.chat.user

        self.admin_message = self._create_admin_message()

    def _create_admin_message(self):
        if self.chat.chat_type == 'callback':
            chat = self.chat.open_message.get('from')
        else:
            chat = self.chat.open_message.get('chat')

        standart_prefix = f"[@{chat.get('username')}] " \
                          f"{chat.get('first_name')} {chat.get('last_name')}"

        extra_prefix = f"user_id: [{self.user.id}]" if self.user else ""

        return f'{standart_prefix} {extra_prefix} отправил сообщение'

    @classmethod
    def as_view(cls, **static_kwargs):
        async def view(request, *args, **kwargs):
            if static_kwargs and len(static_kwargs):
                kwargs['options'] = static_kwargs
            self = cls(request, *args, **kwargs)
            return await self.handle(*args, **kwargs)
        return view

    def glance_callback_data(self):
        return json.loads(self.message.get('data', "[]"))

    def get_message_text(self):
        return self.message.get('text')

    async def delegate(self, cls):
        handler = cls(self.request, self.args, self.kwargs)
        return await handler.handle()

    async def handle(self, *args, **kwargs):
        if not self.user and self.create_user_if_not_exists:
            chat = self.message['chat']
            chat_user_data = {
```

```
                'telegram_account': {
                    'chat_id': chat.get('id'),
                }
            }
            self.user = self.chat.core_bot.models.users.add_user_to_cache(
                chat_user_data
            )
            self.chat.user = self.user

    async def forward_message_to_admin_chat(self, admin_chat=None,
admin_message=None):
        if not admin_message:
            admin_message = self.admin_message
        if not admin_chat:
            admin_chat = self.admin_chat
        await admin_chat.send_message(admin_message)
        if self.message.get('message_id', None):
            await admin_chat.forward_message(self.chat.id,
self.message['message_id'])
        else:
            await admin_chat.send_message('Попытка проксировать несуществующее
сообщение')

    async def _delete_inline_message(self, notification_message=None):
        try:
            return await self.chat.editor.deleteMessage()
        except telepot.exception.TelegramError as error:
            message_id = telepot.message_identifier(
                self.request.message['message'])
            deleted = await self.chat.core_bot.delete_message(message_id)
            if not deleted and notification_message:
                await self._edit_inline_message_text(notification_message)
            return None

    async def _edit_inline_message_text(self, *args, **kwargs):
        try:
            return await self.chat.editor.editMessageText(*args, **kwargs)
        except telepot.exception.TelegramError:
            message_id = telepot.message_identifier(
                self.request.message['message'])
            return await self.chat.core_bot.edit_message_text(message_id, *args,
**kwargs)
```

Приложение Д
(обязательное)

Dockerfile для сервера

```
FROM python:3.6

ENV DB_NAME=${DB_NAME} \
  DB_USER=${DB_USER} \
  DB_PASS=${DB_PASS} \
  DB_HOST=${DB_HOST} \
  DB_PORT=${DB_PORT}

RUN apt-get update && apt-get install -y \
    qt5-default \
    libqt5webkit5-dev \
    build-essential \
    python-lxml \
    python-pip \
    xvfb

RUN mkdir /code
WORKDIR /code
COPY requirements.txt /code/

RUN pip install -r requirements.txt

COPY . /code/
```

Приложение Е
(справочное)

Значения коэффициентов t_1 , t_2 [85]

Вид светопропускающего материала	Значения t_1	Вид переплета	Значения t_2
Стекло оконное листовое:		Переплеты для окон и фонарей промышленных зданий:	
одинарное	**0,9**	деревянные:	
двойное	**0,8**	одинарные	**0,75**
тройное	**0,75**	спаренные	**0,7**
Стекло витринное толщиной 6-8 мм	**0,8**	двойные раздельные	**0,6**
Стекло листовое армированное	**0,6**	стальные:	
Стекло листовое узорчатое	**0,65**	одинарные открывающиеся	**0,75**
Стекло листовое со специальными свойствами:		одинарные глухие	**0,9**
Солнцезащитное	**0,65**	двойные открывающиеся	**0,6**
Контрастное	**0,75**	двойные глухие	**0,8**
Органическое стекло:		Переплеты для окон жилых, общественных и вспомогательных зданий:	
прозрачное	**0,9**	деревянные:	
молочное	**0,6**	одинарные	**0,8**
Пустотелые стеклянные блоки:		спаренные	**0,75**
светорассеивающие	**0,5**	двойные раздельные	**0,65**
светопрозрачные	**0,55**	с тройным остеклением	**0,5**
Стеклопакеты	**0,8**	Металлические:	
		одинарные	**0,9**
		спаренные	**0,85**
		двойные раздельные	**0,8**
		с тройным остеклением	**0,7**
		Стекложелезобетонные панели с пустотелыми стеклянными блоками при толщине шва:	
		20 мм и менее	**0,9**
		более 20 мм	**0,85**
Примечание – Значения коэффициентов t_1 , t_2 для светопропускающего материала и переплетов, не указанных в таблице, следует определять по ГОСТ 26602.4.			

Приложение Ж
(справочное)

Значения коэффициентов t_3, t_4 [85]

Несущие конструкции покрытий	Коэффициент, учитывающий потери света в несущих конструкциях, t_3	Солнцезащитные устройства, изделия и материалы	Коэффициент, учитывающий потери света в солнцезащитных устройствах, t_4
Стальные фермы	**0,9**	Убирающиеся регулируемые жалюзи и шторы (межстекольные, внутренние, наружные)	**1**
Железобетонные и деревянные фермы и арки	**0,8**	Стационарные жалюзи и экраны с защитным углом не более 45° при расположении пластин жалюзи или экранов под углом 90° к плоскости окна	
Балки и рамы сплошные при высоте сечения:		горизонтальные	**0,65**
50 см и более	**0,8**	вертикальные	**0,75**
менее 50 см	**0,9**	Горизонтальные козырьки:	
		с защитным углом не более 30°	**0,8**
		с защитным углом от 15° до 45° (многоступенчатые)	**0,9-0,6**
		Балконы глубиной:	
		до 1,20 м	**0,9**
		1,50 м	**0,85**
		2,00 м	**0,78**
		3,00 м	**0,62**
		Лоджии глубиной:	
		до 1,20 м	**0,8**
		1,50 м	**0,70**
		2,00 м	**0,55**
		3,00 м	**0,22**

Приложение И
(справочное)

Коэффициенты использования светового потока светильников с типовыми кривыми силы света, излучаемого в нижнюю полусферу [86]

Типовая КСС	Равномерная М								Косинусная Д								Глубокая Г							
$\rho_п$, %	70				50		30	0	70				50		30	0	70				50		30	0
$\rho_с$, %	50		30		50	30	10	0	50		30		50	30	10	0	50		30		50	30	10	0
$\rho_р$, %	30	10	30	10	10		10	0	30	10	30	10	10		10	0	30	10	30	10	10		10	0
Значение $i_а$	Коэффициент использования, %																							
0,5	28	28	21	21	25	19	15	13	36	35	30	30	34	28	25	22	58	57	55	53	57	53	49	47
0,6	35	34	27	26	31	24	18	17	43	42	35	34	40	33	28	27	68	65	62	60	64	60	57	56
0,7	44	39	32	31	39	31	25	24	48	47	41	38	45	38	33	31	74	69	68	64	69	64	61	60
0,8	49	46	38	36	43	36	29	28	54	51	45	43	49	43	37	36	78	73	72	69	72	69	66	64
0,9	51	48	40	39	46	39	31	30	57	55	48	46	52	46	41	39	81	76	75	72	75	72	70	67
1,0	54	50	43	41	48	41	34	32	60	57	52	50	55	49	45	42	84	78	78	75	77	74	72	70
1 1	56	52	46	43	50	43	35	33	64	60	55	52	58	51	47	44	87	81	80	77	79	76	74	72
1,25	59	55	49	46	53	45	38	35	69	63	60	56	61	55	50	48	90	83	84	79	82	79	76	75
1,5	64	59	53	50	56	49	42	39	75	69	67	62	67	61	55	53	94	86	88	83	85	82	79	78
1 75	68	62	57	53	60	53	45	42	79	72	71	66	70	65	60	57	97	88	92	85	86	85	82	80
2 0	73	65	61	56	63	56	48	45	83	75	75	69	73	68	64	61	99	90	95	88	88	87	84	82
2,25	76	68	65	60	66	59	51	48	86	77	79	73	76	71	66	64	101	92	97	90	90	88	85	83
2,5	79	70	68	63	68	61	54	51	89	80	82	75	78	73	69	66	103	93	99	91	91	89	87	85
3,0	83	75	73	67	72	65	58	55	93	83	86	79	81	77	73	71	105	94	102	92	93	91	89	86
3,5	87	78	77	70	75	68	61	59	96	86	90	82	83	80	76	73	107	95	104	94	94	93	90	88
4,0	91	80	81	73	78	72	65	62	99	88	93	84	85	83	79	76	109	96	105	94	94	94	91	89
5,0	95	83	86	77	80	75	69	65	105	90	98	88	88	85	81	79	111	97	108	96	96	95	92	90

Приложение К
(справочное)

Chapter 2

Information System Design

Студент:

Группа	ФИО	Подпись	Дата
8ВМ81	Бокор Владимир Алексеевич		

Консультант ОИТ:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Фадеев Александр Сергеевич	к.т.н		

Консультант-лингвист отделения иностранных языков ШБИП:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИЯ	Сидоренко Татьяна Валерьевна	к.п.н		

2 Information System Design

Before starting the development of student information and notification system, a design was carried out. Initially, functional requirements needed to be defined. They specify the functions that the new system must be able to perform to meet the user requirements [34]. The common tool for identifying the functional requirements is the Use Case Diagram in the UML notation.

Figures 4, 5 show the use case diagrams where three roles were allocated: administrator, registered and unregistered user.

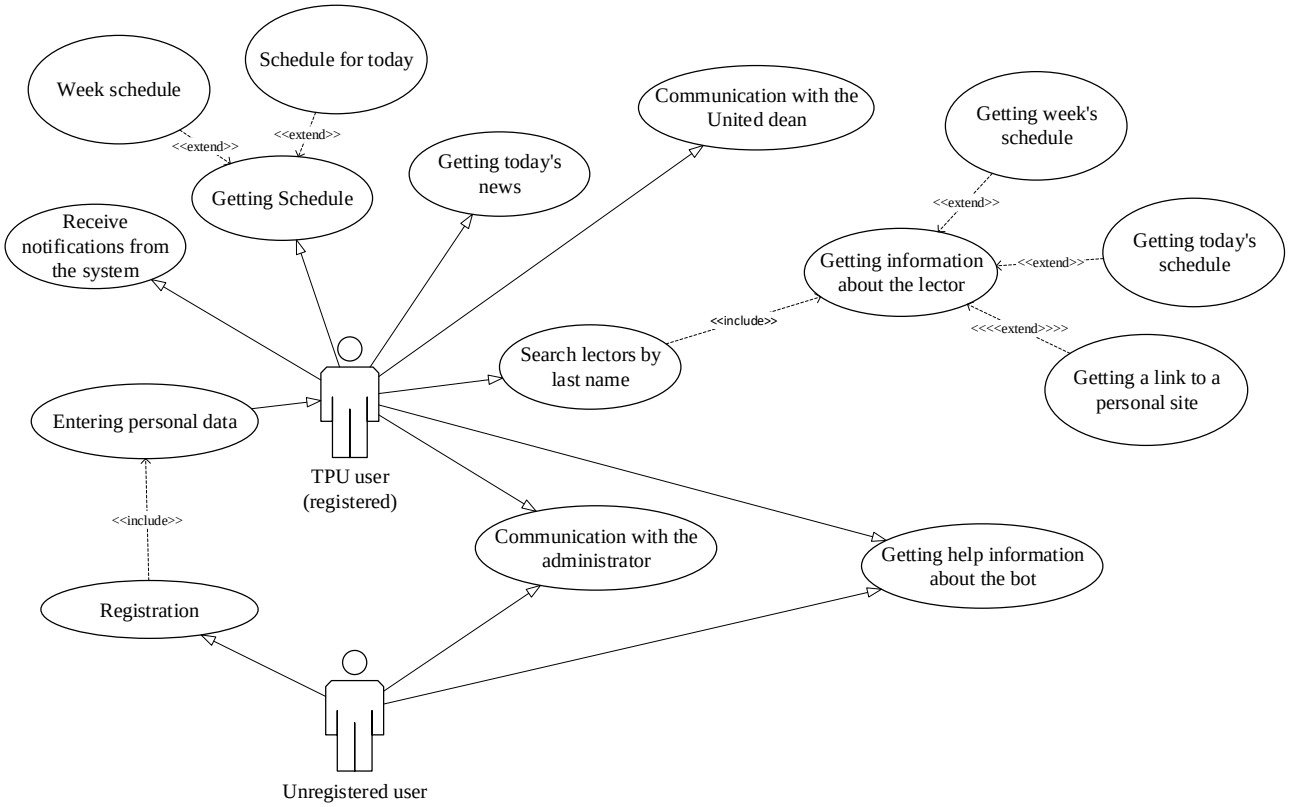


Figure 4. Use Case Diagram for unregistered and registered user

Obviously, user must be a registered to have a full usage of the web application. Unregistered users can only talk with an administrator and get the help information about a client they use.

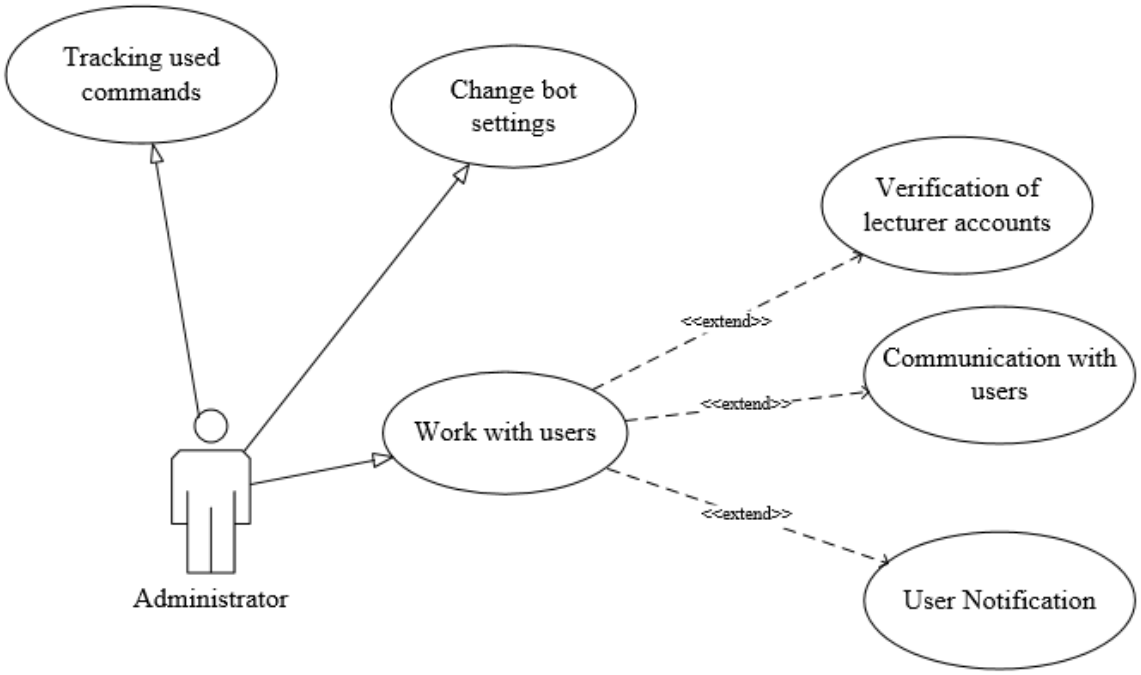


Figure 5. Use Case Diagram for administrator

2.1 Information System Architecture

The application functionality is divided into two main parts: a client and a server. This separation allows programmers develop both the client and the server independently at the same time, since it is only necessary that the interface remains constant.

Server. The server provides certain well-defined services to clients. It will generally be a powerful computer dedicated to providing specific functionality or to managing a resource (shared files, printers, a database, or pooled processing power). The server waits for requests from clients and responds to them. It may be able to handle any number of simultaneous client connections or might be limited to certain usage patterns [35].

Client. The client is a connection between a user and a server. It sends requests to the server and displays the response in a user-friendly way. Clients can be of various types, from a simple terminal to complex web and GUI interfaces.

Application logic can be shared between the server and clients in different ways. But it is worth noting that the more specialized the client is, the less flexible the

whole system is, as the overall functionality decreases, which is achieved due to the central place - the server. Therefore, when using such an architecture “thin client” tries to be built.

However, here is also and a drawbacks. The client/server design is crippled when the server is unavailable (through some software fault or routine maintenance): No client will be able to operate until the server comes back to life. For this reason, client/server installations generally require a designated administrator to keep all systems running smoothly [35].

Figure 6 graphically depicts the architecture of the information system.

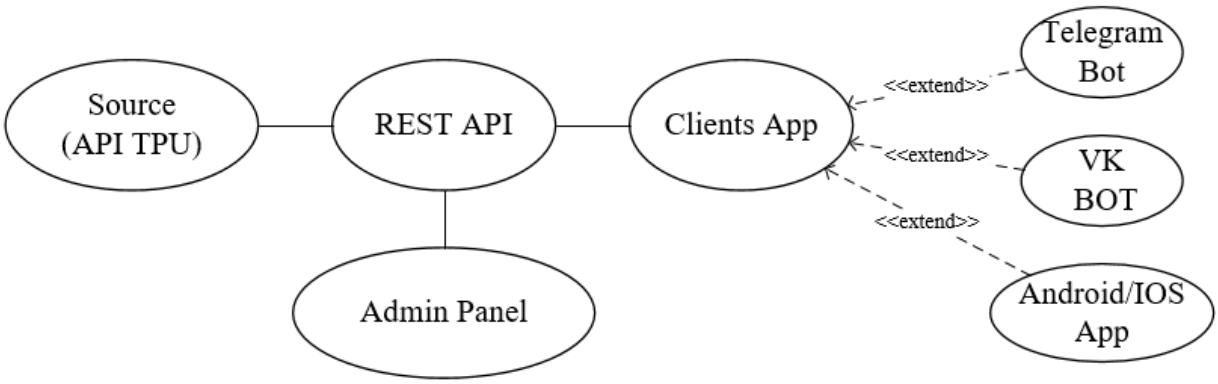


Figure 6. Information System Architecture

The work uses REST (Representational State Transfer). When building REST, the following features must be observed:

1. **Stateless**. That means the communication between the client and the server always contains all the information needed to perform the request. There is no session state in the server, it is kept entirely on the client's side. If access to a resource requires authentication, then the client needs to authenticate itself with every request [36].

2. **Cacheable**. The client, the server and any intermediary components can all cache resources in order to improve performance [36].

3. **Uniform interface**. There are uniform rules for communication between components, which simplifies the architecture.

4. **Layered system**. Individual components cannot see beyond the immediate layer with which they are interacting. This allows the components to be independent and therefore easily replaceable or expandable

2.2 Justification for the selection of software development tools

At the first stage, also was raised the question about the technologies that will be used in the development.

2.2.1 Choosing technologies for the server side

In the world there are a large number of different frameworks that can speed up the development process, so developers often choose a technology written in a familiar language. Learning a new language is not so hard, but it takes time. The following are the advantages and disadvantages of some of the popular frameworks for backend development [37].

2.2.1.1 Django

Django is a high-level Python Web framework that encourages rapid development and clean, pragmatic design [38]. This is a fairly powerful infrastructure and many of the largest companies use it, for instance, Pinterest, Instagram, Udemy and others [39].

The following advantages of Django can be distinguished.

1. **Rapid and low-cost prototype development**. Django has a flexible structure that allows reuse code from one project to another. Besides, there are plenty of out-of-the-box libraries that can help build a product prototype or an MVP fast.

2. **Explicit, logical syntax**. Django is based on python, which one of the simplest language for learning [40].

3. **An extensive open-source ecosystem.** Being an open-source ecosystem means that numerous tools and libraries, both free and paid, are available for everyone at any time.

4. **Django admin portal.** The integrated panel is a convenient tool for user management.

5. Django's REST framework.

The main disadvantage of Django is that it is a monolithic system. The framework infrastructure is a single unit in which all components are connected together, so you cannot choose which parts to add in a particular product. In addition, Django keeps developers within specific frames and patterns.

2.2.1.2 Spring

Spring is the most popular application development program for java enterprise. It allows developers to write backend web applications in java [41].

Most of the developers around the globe use the spring framework to create high-performance web apps. Moreover, it helps in creating simple, fast, flexible, and portable java based applications [36].

The advantages of the Spring Framework include the following points.

1. **Plain Old Java Object (POJO). POJO** is an ordinary Java object, not bound by any special restriction and not requiring any class path. [42].

2. Flexible Configurations. Developers have the option to choose either XML or Java-based annotations for configuration purposes. Having such an option makes the jobs of developers a lot simpler [43].

3. The AOP Module. Developers can have different compilation units or a separate class loader [43].

4. **Testing is Easier.** The Spring Dependency injection helps developers insert test data.

Disadvantages of Spring Framework.

1. **Complexity**. The Spring framework should only use it if you have an experienced team of developers who have used this framework before. The learning curve will be difficult [43].

2. **Parallel Mechanisms**. One of the biggest advantages of Spring is that it gives developers a wide array of options, but this could also be a disadvantage because it causes confusion. Developers have to know which features will be useful, and making the wrong decisions could lead to significant delays [44].

3. **No Specific Guidelines**. Within the Spring documentation, it says nothing about dealing with threats such as XSS or CSRF.

2.2.1.3 Ruby on Rails

Ruby on Rails is also an open source framework. This allows developers to use ready-made solutions, helping them save time on programming processes [45]. Rails is used by companies such as Airbnb, Twitter, Github [46].

Based on the Ruby language, Ruby on Rails inherited its parent's logic and simplicity. Basically, Rails is a layer on top of Ruby that helps developers build web applications.

As a fully-fledged framework, it offers an object-relational-mapping (ORM) system for business data and logic, application management, and routing out of the box [45].

The following benefits of Ruby on Rails are highlighted.

1. **Component structure based on plugins and gems**. Ruby on Rails' component structure, based on plugins (application level) and gems (system level), lets experienced RoR developers quickly put together efficient applications with less coding. The plugins are well documented and easy to use. New gems are constantly added to public repositories, such as the popular RubyGems resource, which currently contains more than 150,000 total gems for download [47].

2. Easy to migrate and modify.

3. Diversity of presets and tools. There are lots of must-have features that are already preconfigured.

There are also disadvantages.

1. **Faster complexity and tech-debt buildup.** Ruby on Rails' flexibility has a downside. Basically, with so many ways to code the same outcome, code can get difficult to read and may require a steeper learning curve as well as more rework later on [48].

2. **More difficult-to-create API.** Building an API with Ruby on Rails can be incredibly complex, as RoR has no equivalent to Django's REST framework.

3. **Documentation quality and standards vary**. With Ruby on Rails, it may be hard to find good documentation, especially for less popular gems.

2.2.1.4 Laravel

Laravel is a web application framework that provides an expressive and elegant syntax. Laravel is useful for creating fast server-side web applications based on PHP [49]. Companies like 9GAG, MasterCard, E-Commerce use Laravel in their stack [50].

The following advantages of Laravel can be distinguished.

1. **Object Relational Mapping support.**
2. **Easy to learn.**
3. **Queue management.** Laravel provides an excellent abstraction process in order to abstract the unnecessary tasks and getting them queued behind the scenes making the user response time much faster [51].
4. **Fast execution of the web application [52].**

The advantage and the disadvantage at the same time, is that the framework is quite new (2011), which means that there can be some bugs, as it is still being developed. In addition, the developer community is smaller than in other presented frameworks.

Also, Laravel is a lightweight framework so it has less inbuilt support when compared to Django and Ruby on Rails. This problem can be solved by integrating third-party tools but for large or custom websites, the tasks can become tedious and complicated.

2.2.1.5 Conclusion

Overall, based on the advantages and disadvantages of the frameworks described above, as the technology for development Django was chosen. This is due to the need to create a prototype in a short time by one person, to have a convenient library for building a REST API, to get to know the language and the framework as a whole. In addition, it combines the advantages of most of the described solutions, and the disadvantages appear only in the late stages of development.

2.2.2 Choosing technologies for the client side

2.2.2.1 Python

Python is fast, easy-to-use, and easy-to-deploy programming language that is being widely used to develop scalable web applications. Python provides excellent library support and has a large developer community [52].

The following advantages of this programming language are highlighted.

1. Easy to learn and use.

2. **OOP support**.

3. Ideal for building **prototypes** and testing out ideas faster.

4. Open-source with an ever-growing **community** support.

The disadvantages include the following points.

1. **Interpreted**. As a result of this, it loses compiled in speed, despite the development of technology.

2. **One-threaded.** Python has GIL (Global Interpreter Lock) – is a mechanism used in computer-language interpreters to synchronize the execution of threads so that only one native thread can execute at a time. [53].

2.2.2.2 Java

Java is a general-purpose programming language that follows the paradigm of object-oriented programming and the Write Once Run Anywhere approach. Java is used for desktop, web, mobile, and enterprise applications [54]. It is known that Java is extremely stable, so many large enterprises have adopted it and use it in applications ranging from e-commerce sites to large financial applications such as electronic trading systems [55].

The advantages of this programming language include the following points.

1. An abundance of open-source libraries

2. Follows the OOP paradigm.

3. **JVM**. A high degree of platform independence.

4. Supports multithreading.

As disadvantages can be identified as following

1. Entry threshold.

2. **Bugs in JVM** implementations (all JVMs are not created equal). This isn't Java's fault, but it has been known to limit Java's usefulness [56].

3. **Memory requirement**. Compared to other compiled programs like C and C++, Java consumes more memory and significantly slower in performance [57].

2.2.2.3 C++

C++ is a general-purpose programming language created as an extension of the C programming language, or "C with Classes". The language has expanded significantly over time, and modern C++ now has object-oriented, generic, and functional features in addition to facilities for low-level memory manipulation [58].

Almost all low-level systems such as operating systems, file systems, etc are written in C/C++.

C++ is also widely used by competitive programmers owing to the fact that it is extremely fast and stable. C++ also provides something called STL - Standard Template Library. STL is a pool of ready-to-use libraries for various data structures, arithmetic operations, and algorithms.

Advantages of C ++:

– faster execution of programs than most programming languages,

– a galore of compilers and libraries to work with,

– runs close to the system hardware and hence, offers a low level of abstraction,

– wide variety of application.

C ++ disadvantages:

– complex syntax,

– less efficient object-oriented system compared to other OOP-based programming languages,

– no garbage collection or dynamic memory allocation,

– plagued by the issues of buffer overflow and memory corruption.

2.2.2.4 Conclusion

Overall, Python was chosen as one of the fastest ways to create a working product for developing the first version of the client for Telegram. In addition, there is a large number of ready-made libraries to work with Telegram Bot API, such as Telepot [61], which will allow to focus on writing business logic and application functionality, rather than on communications.