

МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования
Гомельский государственный технический университет имени П. О. Сухого

Факультет автоматизированных и информационных систем

Кафедра «Информатика»

специальность 1-40 04 01 «Информатика и технологии программирования»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломной работе
на тему

Мобильное приложение «Абитуриент ГГТУ»

Разработал студент гр. ИП-41	_____	<u>Процкая М. А.</u>
	(подпись)	(Ф.И.О.)

Руководитель работы	_____	<u>доцент, к.т.н., Прокопенко Д. В.</u>
	(подпись)	(ученое звание, ученая степень, Ф.И.О.)

Консультант по экономической части	_____	<u>профессор, д.э.н., Голуб В. А.</u>
	(подпись)	(ученое звание, ученая степень, Ф.И.О.)

Консультант по охране труда и технике безопасности	_____	<u>профессор, д.т.н., Кудин В. П.</u>
	(подпись)	(ученое звание, ученая степень, Ф.И.О.)

Нормоконтроль	_____	<u>Самовендюк Н. В.</u>
	(подпись)	(ученое звание, ученая степень, Ф.И.О.)

Рецензент	_____	_____
	(подпись)	(ученое звание, ученая степень, должность, организация, Ф.И.О.)

Дипломная работа (_____ с.) допущена к защите
в Государственной экзаменационной комиссии.

Зав. кафедрой «Информатика»	_____	<u>доцент, к.т.н., Трохова Т. А.</u>
	(подпись)	(ученое звание, ученая степень, Ф.И.О.)

Гомель 2021

Реферат

МОБИЛЬНОЕ ПРИЛОЖЕНИЕ «АБИТУРИЕНТ ГГТУ»: дипломная работа / М. А. Процкая. – Гомель : ГГТУ им. П. О. Сухого, 2021. – Дипломная работа: 81 страница, 54 рисунка, 5 таблиц, 12 источников, 6 приложений.

Ключевые слова: мобильное приложение, абитуриент, университет, приемная кампания, личный кабинет, вступительные испытания.

Объектом разработки является программный продукт, направленный на оптимизацию работы приемной кампании университета.

Цель работы: разработка информационно-справочного мобильного приложения, разработка которого приведет к повышению эффективности процесса взаимодействия абитуриентов и студентов и Гомельского государственного технического университета.

В процессе работы было сделано следующее: выполнен анализ существующего программного обеспечения, создана база данных, разработан программный продукт.

Основными функциями разрабатываемого программного комплекса являются:

- создание личного кабинета абитуриента;
- календарь со свежими и актуальными новостями и информацией о планирующихся событиях и мероприятиях;
- обеспечение доступа к сайтам и аккаунтам ГГТУ в социальных сетях;
- общая и необходимая информация об университете: контакты, адреса, местоположения на карте, руководство университета;
- просмотр информации о ходе приемной кампании;
- подсчет общего балла по введенным баллам вступительных испытаний и аттестата для получения информации о том, на какие специальности пользователь может пройти на основе прошлогодних проходных баллов.

Элементом новизны является разработка непосредственно для университета, возможность подсчета общего проходного балла, доступность основного функционала для всех пользователей.

Областью практического применения является применение этого программного обеспечения в Гомельском государственном техническом университете как информационно-справочный ресурс.

Студент-дипломник подтверждает, что дипломная работа выполнена самостоятельно, приведенный в дипломной работе материал объективно отражает состояние разрабатываемого объекта, пояснительная записка проверена в системе «Антиплагиат» «АО «Антиплагиат»» (режим доступа: <https://www.antiplagiat.ru/>). Процент оригинальности составляет 81,18%. Все заимствованные из литературных и других источников, теоретические и методологические положения и концепции сопровождаются ссылками на источники, указанные в «Списке использованных источников».

Лист задания

Резюме

Тема работы: «Мобильное приложение “Абитуриент ГГТУ”».

Объектом исследования является программный продукт, направленный на оптимизацию работы приемной кампании университета.

Цель работы: разработка информационно-справочного мобильного приложения, разработка которого приведет к повышению эффективности процесса взаимодействия абитуриентов и студентов и Гомельского государственного технического университета.

Основным результатом работы является создание мобильного приложения, позволяющего осуществлять предварительную регистрацию абитуриента, вход в личный кабинет, использование перечня сервисов.

Рэзюмэ

Тэма працы: «Мабільнае прыкладанне “Абітурыент ГДТУ”».

Аб’ектам даследвання з’яўляецца праграмны прадукт, накіраваны на аптымізацыю працы прыёмнай кампаніі ўніверсітэта.

Мэта працы: распрацоўка інфармацыйна-даведачнага мабільнага прыкладання, распрацоўка якога прывядзе да павышэння эфектыўнасці працэсу ўзаемадзеяння абітурыентаў і студэнтаў і Гомельскага дзяржаўнага тэхнічнага ўніверсітэта.

Асноўным вынікам работы з’яўляецца стварэнне мабільнае прыкладанне, якое дазваляе ажыццяўляць папярэднюю рэгістрацыю абітурыента, уваход у асабісты кабінет, выкарыстанне пераліку сэрвісаў.

Summary

The theme is «Mobile application “GSTU entrant”».

The object of the research is a software product aimed at optimizing the work of the university's admissions campaign.

Purpose of the work: development of an information and reference mobile application, the development of which will lead to an increase in the efficiency of the process of interaction between applicants and students and the Gomel State Technical University.

The main result of the work is the creation of a mobile application that allows pre-registration of an entrant, access to a personal account, and the use of a list of services.

СОДЕРЖАНИЕ

Введение.....	7
1 Аналитический обзор существующих методов и средств автоматизации	9
1.1 Обзор существующих систем автоматизации.....	9
1.2 Анализ современного языка программирования Java.....	12
1.3 Анализ используемых технологий для реализации поставленной задачи.....	18
1.4 Анализ инструментальных средств автоматизации разработки	19
1.5 Техническое задание для мобильного приложения «Абитуриент ГГТУ».....	21
2 Анализ предметной области и алгоритмы.....	22
2.1 Анализ предметной области.....	22
2.2 Алгоритм работы мобильного приложения	22
2.3 Функциональная модель мобильного приложения	23
2.4 Сущности и связи между ними.....	24
2.5 Архитектура мобильного приложения	26
2.6 Информационная модель мобильного приложения	29
2.7 Проектирование базы данных	31
3 Структура и основные алгоритмы программного обеспечения.....	34
3.1 Общая структура программного обеспечения	34
3.2 Основные алгоритмы программного обеспечения.....	38
4 Тестирование, верификация и валидация программного обеспечения.....	45
4.1 Ручное тестирование программного продукта	47
4.2 Автоматизированное тестирование программного продукта	52
5 Экономическое обоснование дипломной работы.....	55
5.1 Расчет общей трудоемкости разработки программного обеспечения.....	55
5.2 Расчет затрат на разработку программного продукта.....	58
5.3 Формирование цены при создании программного обеспечения	64
5.4 Расчет эффекта от внедрения программного обеспечения.....	66
6 Охрана труда и техника безопасности	68
6.1 Явление статического электричества	68
6.2 Электростатические заряды на производстве и их опасность	69
6.3 Причины возникновения статического электричества	70
6.4 Опасные и вредные факторы статического электричества	72

6.5 Защита от статического электричества.....	74
7 Ресурсо- и энергосбережение при внедрении программного обеспечения	77
7.1 Основные понятия в области ресурсо- и энергосбережения.....	77
7.2 Ресурсо- и энергосбережение в Республике Беларусь	77
7.3 Расчет показателей ресурсо- и энергосбережения	78
Заключение	80
Список использованных источников	81
Приложение А Исходный текст программного продукта	82
Приложение Б Каталог функций программного обеспечения	156
Приложение В Расчет общей трудоемкости разработки программного обеспечения.....	157
Приложение Г Параметры для расчета производственных затрат на разработку программного обеспечения	158
Приложение Д Расчет суммарных затрат на разработку программного обеспечения.....	159
Приложение Е Техничко-экономические показатели проекта	160

ВВЕДЕНИЕ

В современном обществе из-за огромного непрерывного потока информации различного характера возникают сложности с ее обработкой. Поэтому возникает необходимость прибегать к специализированным программам и средствам для осуществления поиска, анализа и обработки информации.

Одними из этих средств являются мобильные приложения.

Смартфоны и прочее мобильные устройства не только стали частью нашей повседневной жизни, они – полноценное предложение нас. С помощью мобильных телефонов мы не только общаемся друг с другом, но и заказываем товары из магазинов, покупаем билеты, бронируем жилье, вызываем такси, используем телефоны как навигаторы, фото- и видеокамеры, читалки, онлайн банки, и просто как способ развлечься и скоротать время. Согласно статистике, опубликованной в *Datareportal*, 67% взрослых людей во всем мире используют смартфоны ежедневно, а это почти 5,19 млрд человек (при общем количестве населения в 7,75 млрд). Тенденция к переходу с простых мобильных устройств на многофункциональные смартфоны с каждым годом только увеличивается.

В понятие разработки мобильных приложений для смартфонов, планшетов и прочих мобильных устройств входит написание программного кода с целью создания программ, которые будут работать на определенных мобильных платформах (на сегодняшний день существует 2 основные платформы мобильных операционных систем – *Android* и *iOS*, и менее популярные *Windows Phone* и *Symbian* [6]). Эти программы и приложения могут предварительно устанавливаться на мобильные телефоны, персональные цифровые помощники, корпоративные цифровые ассистенты, смартфоны и прочие мобильные устройства до того, как устройства попадут в руки пользователю, либо загружаться пользователями в устройство непосредственно в процессе использования.

Пользовательский интерфейс мобильных приложений играет очень важную роль в процессе создания приложения, ведь интерфейс является соединяющим звеном между аппаратным и программным обеспечением мобильного устройства и фокусом пользовательского взаимодействия.

В процессе разработки мобильного приложения разработчики должны всегда учитывать, насколько внимание пользователей ограничено размером экрана, как сократить количество нажатий клавиш, и как наиболее компактно вместить в приложение необходимый набор функций. Поэтому для мобильных разработчиков процесс разработки мобильных приложений часто не ограничивается одним только написанием кода по поставленному заданию, разработка мобильных приложений включает в себя более широкий и творческий спектр деятельности.

Главная особенность в разработке мобильных приложений заключается в форм-факторе самих девайсов, под которые мы пишем программы, потому что это – смартфоны, планшеты, которые имеют маленький либо небольшой экран, либо нестандартный (квадратный, прямоугольный) экран. Кроме того, если мы возьмем смарт-часы, то это экран 2х2 см, или вообще круглый экран. Поэтому

очень важным моментом создания мобильного приложения является так называемый *User Experience*, то есть взаимодействие с пользователем.

Следующим важным моментом в разработке мобильных приложений является большое разнообразие устройств, под которые мы хотим создать приложение. То есть, когда создается приложение, оно может рассчитываться только на смартфоны, или на смартфоны/планшеты/часы, на различные устройства дополненной реальности, или может рассчитываться на взаимодействие между несколькими экранами одновременно.

Еще одной важной особенностью является постоянное взаимодействие с интернетом. Исторически сложилось, что мобильные приложения – это достаточно легкие приложения, которые не несут в себе большой вычислительной логики. Изначально мобильные устройства располагали достаточно скромной вычислительной мощностью, но сейчас превосходят любой компьютер десятилетней давности. Но даже, учитывая современные многоядерные процессоры и огромное количество оперативной памяти, все равно производительность остается главным фактором в создании мобильных приложений. Поэтому большая часть вычислительной логики, которая есть в приложении, является взаимодействием с интернетом и с облачными сервисами [4].

Целью дипломной работы является повышения эффективности процесса взаимодействия абитуриентов и студентов и Гомельского государственного технического университета посредством разработки и, в последствии, ввода в эксплуатацию, информационно-справочного мобильного приложения для управления информированностью абитуриентов и студентов ГГТУ.

Задачами дипломной работы являются:

- произвести анализ взаимодействия абитуриентов с университетом;
- спроектировать мобильное приложение;
- выбрать наиболее подходящие способы и средства реализации мобильного приложения;
- разработать мобильное приложение, начать процесс его ввода в эксплуатацию.

1 АНАЛИТИЧЕСКИЙ ОБЗОР СУЩЕСТВУЮЩИХ МЕТОДОВ И СРЕДСТВ АВТОМАТИЗАЦИИ

1.1 Обзор существующих систем автоматизации

В ходе аналитического обзора был произведен поиск имеющихся аналогов и сформирован следующий вывод: единственным найденным полным аналогом является недавно опубликованное приложение «Абитуриент ГГУ», в остальном же обнаруженные приложения не являются разработанными непосредственно для абитуриентов университетов, только для студентов в целом (например, приложения с расписанием занятий). Также были выявлены преимущества «Абитуриент ГГУ» перед всеми аналогами. На рисунке 1.1 представлен интерфейс приложения-аналога «Мой универ».

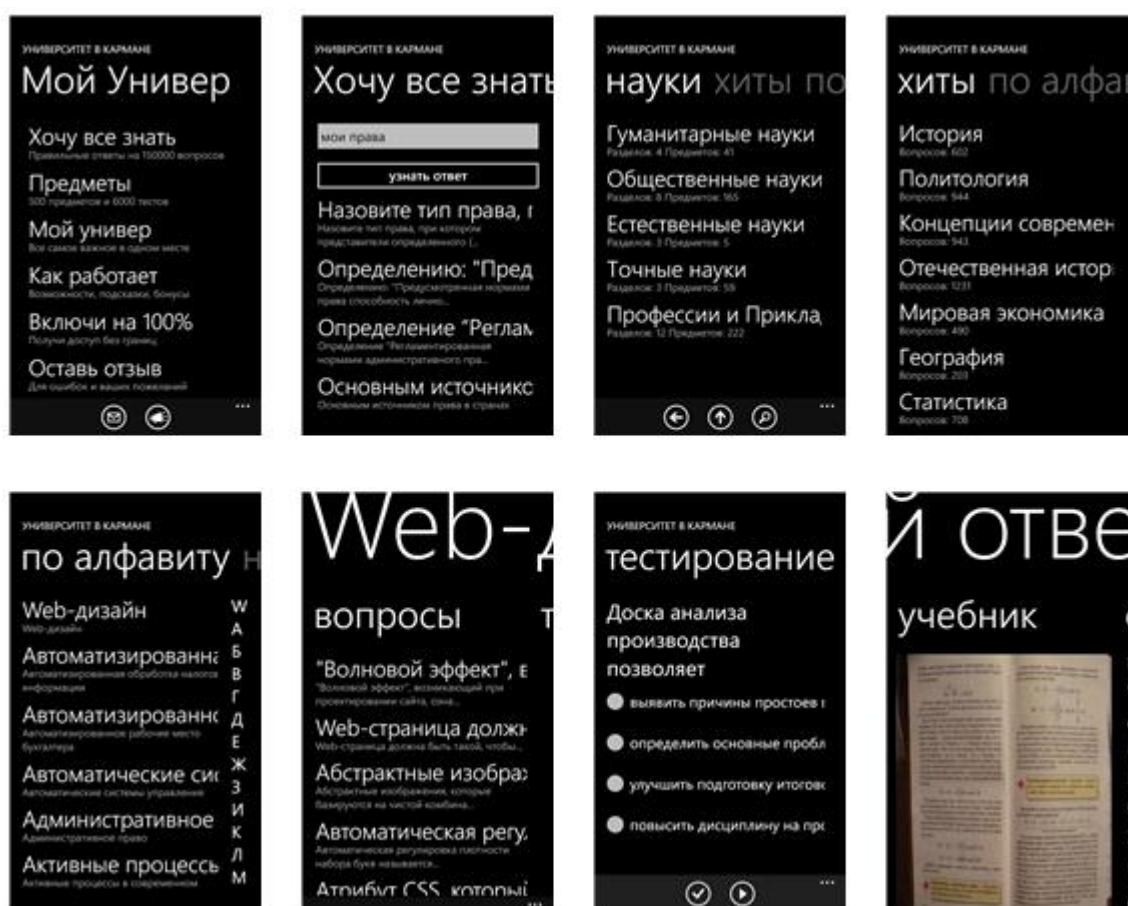


Рисунок 1.1 – Приложение «Мой Универ»

На рисунке 1.2 представлен интерфейс приложения-аналога для студентов медицинского университета.

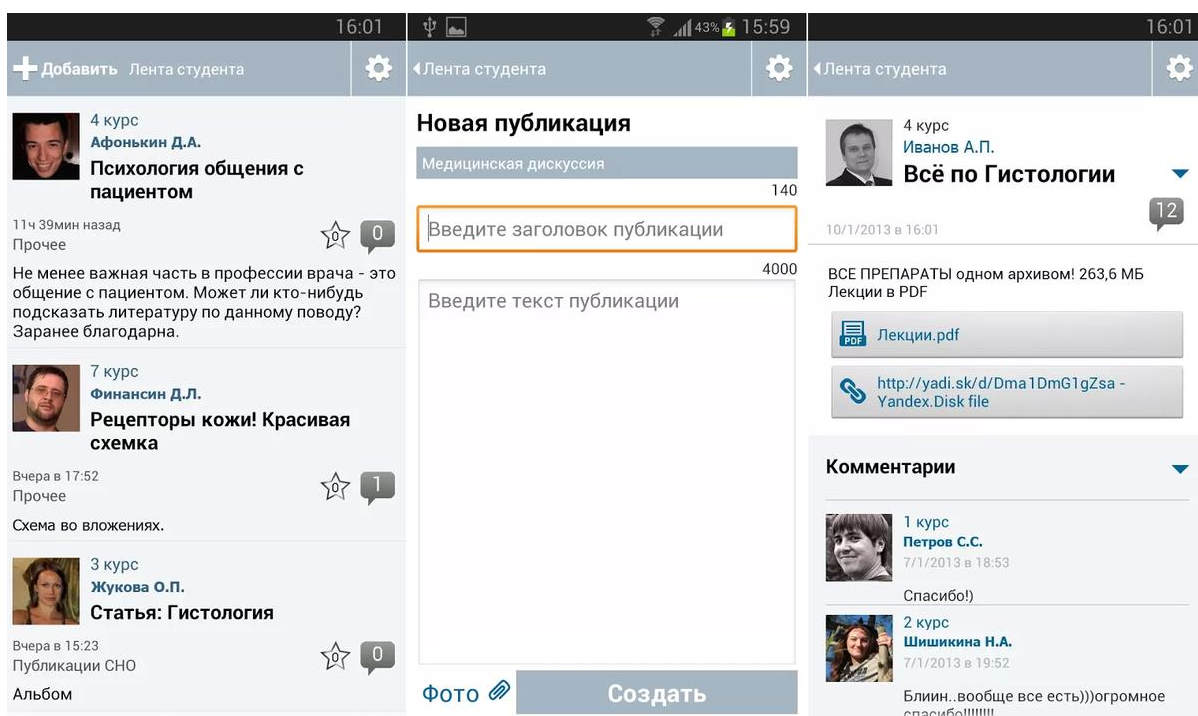


Рисунок 1.2 – Приложение для студентов медицинского университета

На рисунке 1.3 представлен интерфейс приложения-аналога для студентов СДУ.

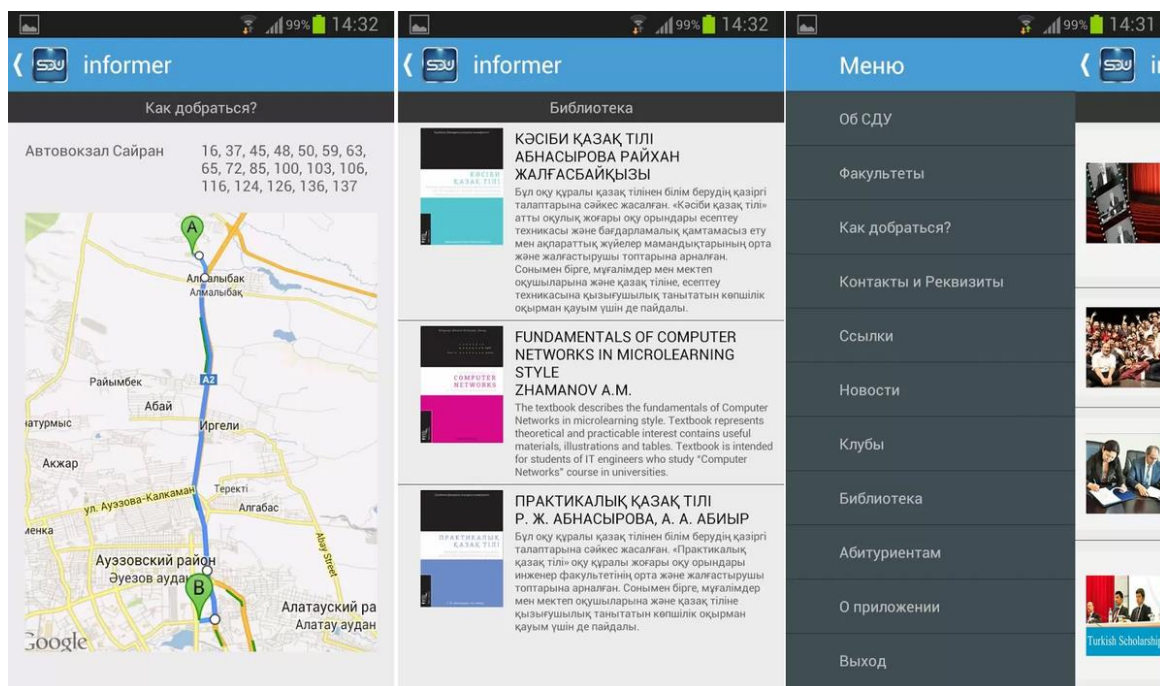


Рисунок 1.3 – Приложение для студентов СДУ

На рисунке 1.4 представлен интерфейс приложения-аналога для студентов университета Анджело.

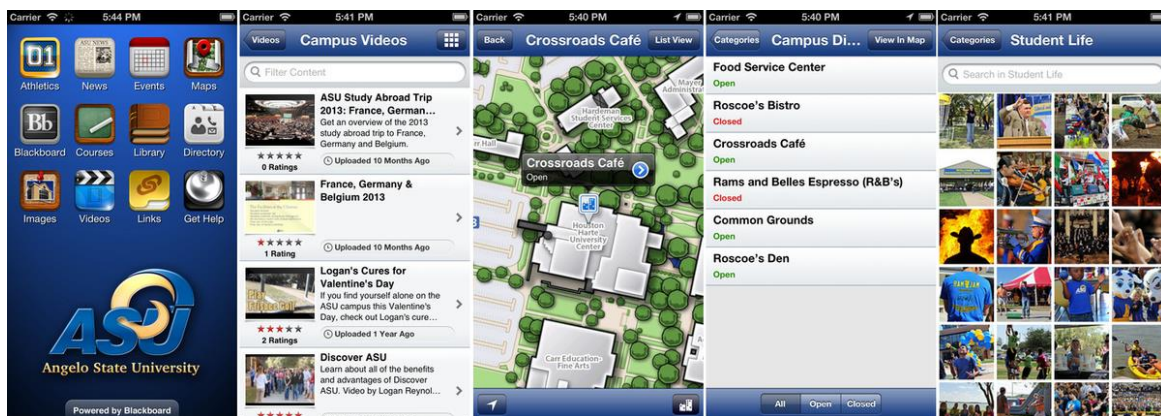


Рисунок 1.4 – Приложение университета Анджело

На рисунке 1.2 представлен интерфейс приложения-аналога «Абитуриент ГГУ».

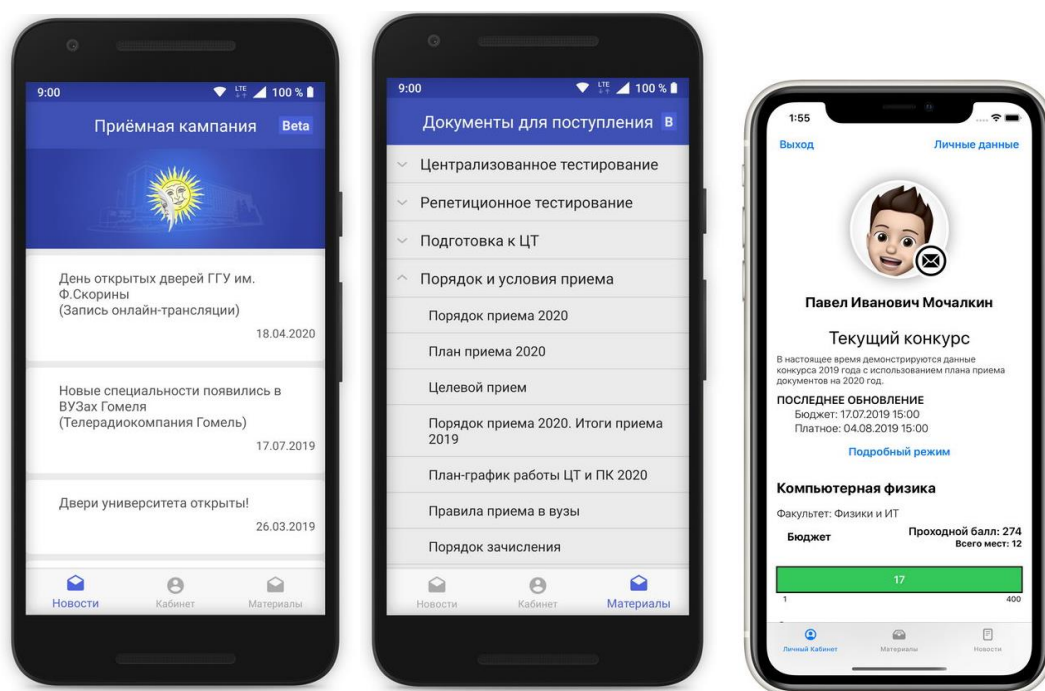


Рисунок 1.5 – Приложение «Абитуриент ГГУ»

Преимуществами разрабатываемого приложения являются:

- возрастание эффективности взаимодействия студентов и абитуриентов с вузом;
- повышение общего уровня информированности, а, вследствие, и обучения, так как доступ к информации является важнейшим фактором ее изучения;
- бесплатное;
- разрабатывается непосредственно для ГГТУ.

1.2 Анализ современного языка программирования Java

Технология *Java* протестирована, усовершенствована, расширена и проверена участниками сообщества разработчиков *Java*, архитекторов и энтузиастов. *Java* позволяет разрабатывать высокопроизводительные портативные приложения практически на всех компьютерных платформах. Доступность приложений в разнородных средах позволяет компаниям предоставлять более широкий спектр услуг, способствует повышению производительности, уровня взаимодействия и совместной работы конечных пользователей и существенному снижению стоимости совместного владения корпоративными и потребительскими приложениями. *Java* стала незаменимым инструментом для разработчиков и открыла для них следующие возможности:

- написание программного обеспечения на одной платформе и его запуск практически на любой другой платформе;
- создание программ, работающих в веб-браузере и имеющих доступ к веб-службам;
- разработка приложений на стороне сервера для форумов в Интернете, магазинов, опросов, обработки форм HTML и много другого;
- объединение приложений или служб с использованием языка *Java* для создания высокоспециализированных приложений или служб;
- создание многофункциональных и эффективных приложений для мобильных телефонов, удаленных процессоров, микроконтроллеров, беспроводных модулей, датчиков, шлюзов, потребительских продуктов и практически любых других категорий электронных устройств.

Java можно найти везде. Это основной язык разработки для *Android* [5]. Он используется в веб-приложениях, правительственных веб-сайтах и технологиях обработки больших данных, таких как *Hadoop* и *Apache Storm*. *Java* подходит и для научных проектов, особенно в области обработки естественного языка. Язык *Java* преобладал и в программировании для мобильных устройств, задолго до появления смартфонов – первые мобильные игры в начале 2000-х годов были написаны на *Java*. *Java*, благодаря своей долгой истории, заработал свое место в Зале славы программирования. Индекс *TIOBE*, один из самых авторитетных индексов популярности программ в мире, при составлении рейтинга использует результаты поисковой выдачи. Несмотря на растущую популярность *Go* и *Python*, *Java* остается на вершине списка уже более десятилетия.

Java – уже не единственный официально поддерживаемый язык для разработки на *Android*. *Java* далеко не единственный выбор в веб-программировании. Тем не менее, *Java* идет в ногу со временем. Рассмотрим преимущества *Java*.

Java включает в себя объектно-ориентированное программирование (ООП) – концепцию, в которой вы не только определяете тип данных и его структуру, но и набор функций, применяемых к нему. Таким образом, структура данных становится объектом, которым можно управлять для создания отношений между различными объектами.

При другом подходе – процедурном программировании – нужно следовать четким инструкциям, использовать переменные и функции. При ООП можно группировать эти переменные и функции посредством контекста, маркировать их и ссылаться на функции в контексте каждого конкретного объекта. Сравнение подходов программирования представлено на рисунке 1.6.

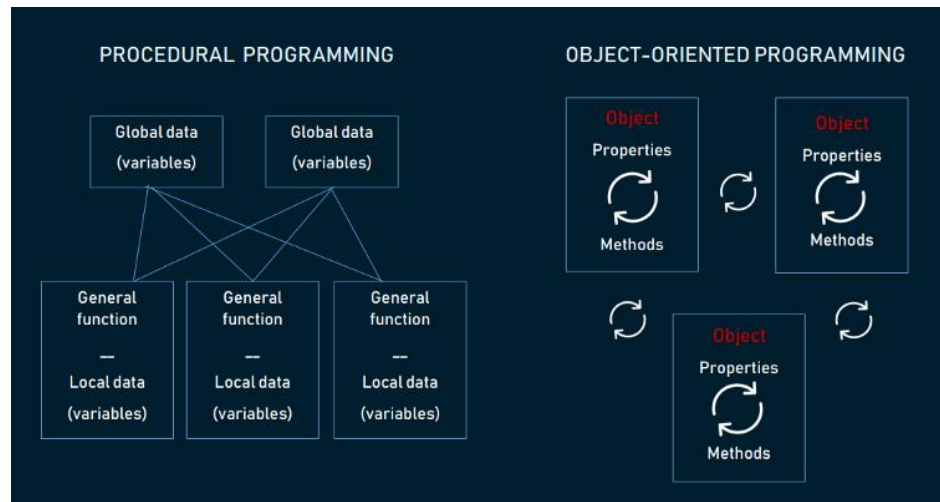


Рисунок 1.6 – Сравнение подходов программирования

Плюсы применения ООП:

- можно повторно использовать объекты в других программах;
- ООП предотвращает ошибки, так как объекты скрывают информацию, к которой не должно быть доступа;
- ООП эффективно организует структуру программ, в том числе больших;
- ООП упрощает обслуживание и модернизацию старого кода.

Java – язык высокого уровня, он похож на человеческий язык, в отличие от языков низкого уровня, напоминающих машинный код.

Языки высокого уровня преобразуются с помощью компиляторов или интерпретаторов, что упрощает разработку, делая язык более легким для написания, чтения и обслуживания. Ниже на рисунке 1.7 представлен пример программы на *Java*.

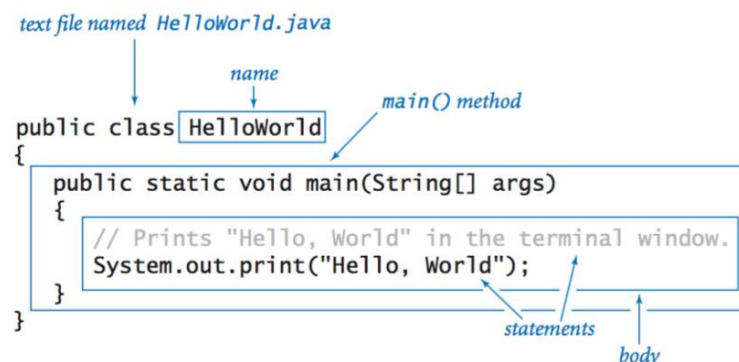


Рисунок 1.7 – Пример программы на *Java*

Синтаксис *Java* основан на *C++*, поэтому *Java* похож на *C*. Тем не менее, он проще, что позволяет новичкам быстрее учиться и эффективнее использовать код для достижения конкретных результатов.

Java не так дружелюбен к новичкам, как *Python*, однако довольно прост для любого разработчика с базовым пониманием фреймворков, пакетов, классов и объектов. Он прост, типизирован и предсказуем, что позволяет учиться мыслить в правильном направлении.

Корпоративные приложения – главное преимущество *Java* с 90-х годов, когда организации начали искать надежные инструменты программирования не на *C*. *Java* поддерживает множество библиотек – строительных блоков любой корпоративной системы. Библиотеки помогают разработчикам создавать любые функции, которые могут понадобиться компании. *Java* широко распространен – это язык, который преподают в рамках введения в программирование в большинстве школ и университетов. Возможности интеграции *Java* впечатляют: большинство хостинг-провайдеров поддерживают *Java*. Более того, *Java* – язык, дешевый в обслуживании: работать с *Java* можно с любого компьютера, вне зависимости от конкретной аппаратной инфраструктуры.

Существует мнение, что *Java* – безопасный язык, однако это не совсем так. Сам язык не защищает от уязвимостей, но некоторые его функции устраняют распространенные уязвимости. Во-первых, в отличие от *C*, в *Java* нет указателей. Указатель – это объект, который сохраняет адрес ячейки памяти другого значения, что может вызвать несанкционированный доступ к памяти. Во-вторых, в *Java* есть *Security Manager*, созданная для каждого приложения политика безопасности, в которой можно указать правила доступа. Это позволяет запускать приложения *Java* в «песочнице» и устранять таким образом уязвимости.

«Написать один раз и использовать везде» (*WORA*) – популярная в *IT*-сфере фраза, с помощью которой *Sun Microsystems* описывает кроссплатформенные возможности *Java*. Можно создать *Java*-приложение на *Windows*, скомпилировать его в байт-код и запустить его на любой другой платформе, поддерживающей виртуальную машину *Java* (*JVM*). Таким образом, *JVM* служит уровнем абстракции между кодом и оборудованием. Механизм работы *WORA* представлен на рисунке 1.8.

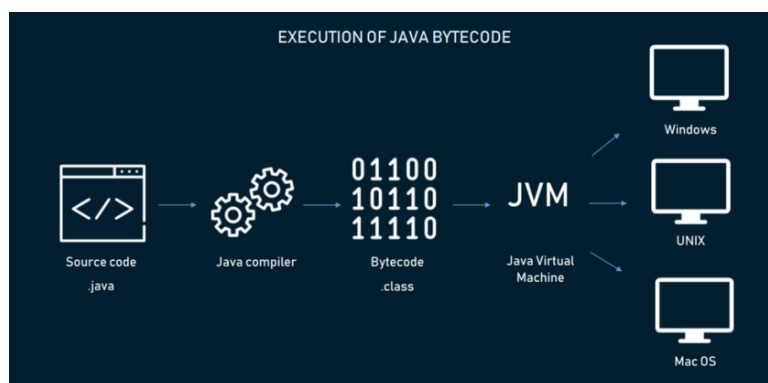


Рисунок 1.8 – Механизм работы *WORA*

Все основные операционные системы, включая *Windows*, *Mac OS* и *Linux*, поддерживают *JVM*. Если программа не опирается на специфичные для платформы функции и пользовательский интерфейс, ее можно с легкостью перенести: по крайней мере, большую ее часть.

Java создавался как язык для распределенного программирования: он имеет встроенный механизм совместного использования данных и программ несколькими компьютерами, что повышает производительность и эффективность труда. На рисунке 1.9 представлено сравнение видов программирования.

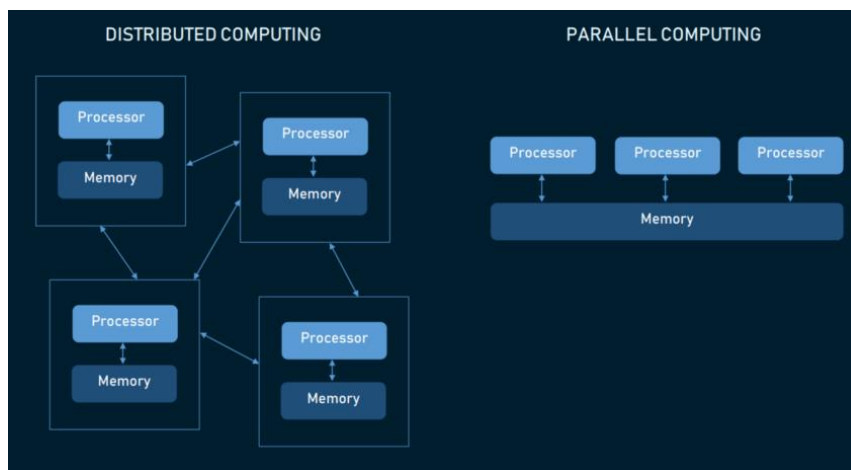


Рисунок 1.9 – Сравнение распределенного и параллельного программирования

В других языках нужно использовать внешний *API* для дистрибуции. В *Java* эта технология встроена. Специфическая для *Java* методология распределенных вычислений называется *Remote Method Invocation (RMI)*. *RMI* позволяет использовать все преимущества *Java*: безопасность, независимость от платформы и объектно-ориентированное программирование для распределенных вычислений. Кроме того, *Java* также поддерживает программирование сокетов и методологию распределения *CORBA* для обмена объектами между программами, написанными на разных языках.

Разработчикам *Java* не нужно вручную писать код для управления памятью благодаря автоматическому управлению памятью (*АММ*). *АММ* также используется в языке программирования *Swift* и при очистке памяти приложениями, которые автоматически обрабатывают распределение и освобождение памяти. Что именно это означает?

Эффективность программы напрямую связана с памятью. При этом объем памяти ограничен. При написании приложения на языках с ручным управлением памятью, разработчики рискуют забыть выделить память, что приведет к увеличению объема занимаемой приложением памяти и проблемам с производительностью. Программы очистки памяти ищут объекты, которые больше не используются программой, и удаляют их. Это влияет на работу процессора, однако умная оптимизация и настройка позволяют снизить это влияние.

Поток – наименьшая единица обработки в программировании. Чтобы максимально эффективно использовать время процессора, *Java* позволяет запускать потоки одновременно, что называется многопоточностью.

Потоки используют одну и ту же область памяти, поэтому между ними можно быстро переключаться. Потоки независимы друг от друга: один поток не влияет на работу других потоков. Это особенно полезно в играх и программах с большим объемом анимации. На рисунке 1.10 изображен пример многопоточного выполнения.



Рисунок 1.10 – Пример многопоточного выполнения.

Уже много лет развитию *Java* способствуют сообщество, поддержка *Oracle* и изобилие приложений и языков на *JVM*. Кроме того, постоянно выпускаются новые версии *Java* с новыми интересными функциями.

Сообщество разработчиков *Java* не имеет себе равных. Около 45% респондентов опроса *StackOverflow* 2018 используют *Java*. У *Java* чрезвычайно большая экосистема хорошо протестированных библиотек и фреймворков для любых задач. Начинающий разработчик, скорее всего, выберет *Java*: на тему *Java*-программирования существует более 1000 курсов на *Udemy* и более 300 на *Coursera*.

Кроме всех вышеперечисленных плюсов программирования на *Java*, существует и ряд минусов.

Недавно *Oracle* объявила, что с 2019 года компания начнет взимать плату за использование *Java Standard Edition* 8 в «коммерческих целях». За все новые обновления и исправления ошибок придется заплатить. Плата зависит от количества пользователей или компьютеров.

Текущая версия *Java* бесплатна для простого использования. Таким образом, каждая использующая *Java* компания должна оценить, насколько эффективно она использует *Java*. Компания должна понять, что выгоднее: искать альтернативное решение или продолжать пользоваться *Java*.

У любого языка высокого уровня довольно низкая производительность из-за компиляции и абстракции с помощью виртуальной машины. Однако это не единственная причина низкой скорости *Java*. Например, приложение очистки памяти: это полезная функция, которая, к сожалению, приводит к значительным проблемам с производительностью, если требует больше 20 процентов времени

процессора. Плохая настройка кэширования может вызвать чрезмерное использование памяти. Существует также взаимная блокировка потоков: так происходит, когда несколько потоков пытаются получить доступ к одному и тому же ресурсу. В этом случае происходит кошмар каждого *Java*-разработчика – ошибка из-за нехватки памяти. Тем не менее умелое планирование может решить все эти проблемы.

Для создания графического интерфейса пользователя (GUI) разработчики используют различные инструменты, ориентированные для конкретного языка. Для *Android*-приложений есть *Android Studio*, которая помогает создавать приложения с нативным дизайном. Однако, когда дело доходит до пользовательского интерфейса на ПК, *Java*-инструмента для создания нативного дизайна нет.

Есть несколько инструментов для разработки GUI для *Java*: самые популярные из них – *Swing*, *SWT*, *JavaFX*, *JSF*. Библиотека *Swing* – это старый, но надежный кроссплатформенный инструмент, интегрированный в различные *Java-IDE*, в том числе *Eclipse* и *NetBeans*. Однако, если вы не используете шаблоны, вы заметите несоответствия интерфейса. *SWT* использует собственные компоненты, но не подходит для сложного интерфейса. *JavaFX* – лаконичный и современный, но слишком новый. В целом, перед созданием GUI на *Java* нужно подробнее изучить инструменты.

Многословность кода может показаться преимуществом, которое поможет при изучении языка. Однако, длинные, чрезмерно сложные предложения затрудняют чтение и просмотр кода. Как и естественные языки, многие языки программирования высокого уровня содержат лишнюю информацию. *Java* – это более легкая версия неприступного *C++*, которая вынуждает программистов прописывать свои действия словами из английского языка. Это делает язык более понятным для неспециалистов, но менее компактным.

Сравнивая *Java* и *Python*, можно заметить, в чем преимущество лаконичного кода *Python*. В *Python* не используются точка с запятой, круглые и фигурные скобки. Вместо «и», «или» и «нет» в качестве операторов используются «&&», «||» и «!».

Большинство организаций так или иначе используют *Java*. Широкий спектр вариантов использования *Java* делает ее практически незаметной в использовании: поэтому часто возникает вопрос «где используется *Java*?».

Несмотря на активный рост *Kotlin*, *Java* по-прежнему остается де-факто основным языком *Android*-приложений. Таким образом, все разработчики *Java* очень легко могут стать *Android*-программистами. Хотя *Android* использует *Android SDK* вместо *JDK*, тем не менее, код написан на *Java*.

Помимо уже упомянутых *Hadoop* и *Apache Storm*, *Java* использовалась для создания *Eclipse*, *OpenOffice*, *Gmail*, *Atlassian* и других.

Java – один из самых востребованных языков в финансовой отрасли. Он используется для создания надежных, быстрых и простых веб-сайтов как на стороне сервера, так и на стороне клиента. *Java* также используется для моделирования данных.

Многие компании используют *Java* для создания систем *PoS*, поскольку их создание обычно требует кроссплатформенности и обширного штата специалистов.

На *Java* написана *Murex*, популярная программа управления банками для фронтальной и обратной связи.

Hadoop написан на *Java*. *Scala*, *Kafka* и *Spark* используют *JVM*. Кроме того, *Java* предоставляет доступ к множеству проверенных библиотек, инструментов отладки и мониторинга.

1.3 Анализ используемых технологий для реализации поставленной задачи

В настоящее время наиболее популярными являются две операционные системы: *Android* и *iOS*, однако на территории Республики Беларусь более популярна система *Android* (рисунок 1.11), исходя из этого для разработки была выбрана именно она.

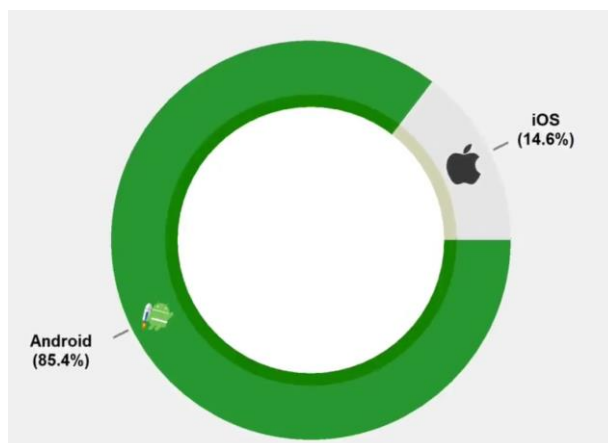


Рисунок 1.11 – Распространенность операционных систем

Актуальность разработки мобильных приложений растет не то, что с каждым годом, но и с каждым месяцем. Сотни новых мобильных приложений выходят на онлайн площадках каждый день. Казалось бы, сложно придумать что-то новое, но хорошие разработки в коллаборации с хорошим маркетингом могут принести много денег создателю.

Существует 2 метода разработки мобильных приложений:

- нативная разработка;
- кроссплатформенная, или гибридная разработка (*ReactNative*, *Flutter*, *Xamarin*).

Нативная разработка подразумевает создание программы для мобильного устройства на конкретном языке под конкретную платформу. Нативные приложения достаточно производительны и не имеют ограничений в разработке (*Java* и *Kotlin* – для *Android*, а *Swift* – для *iOS*). К плюсам такой разработки можно отнести достаточно быструю реакцию на действия пользователя, возможность

иметь прямой доступ к аппаратной части и разработать наиболее привычный для пользователя конкретной платформы интерфейс. К недостаткам можно отнести достаточно высокую стоимость разработки и поддержки, и длительное время, требуемое на разработку.

Разрабатывать одно и то же приложение под разные платформы (*iOS/Android*) долго и дорого, к тому же их тестирование также займет достаточно длительное время (к примеру, компиляция обновлений для Андроид занимает около 30 секунд), поэтому, если нужно создать простое приложение сразу для двух платформ, прибегают к гибриднему способу разработки. Кроссплатформенная разработка производится с помощью *web*-технологий – *HTML*, *CSS* и *JavaScript* – которые позволяют разработать приложение сразу на несколько платформ. Но для того, чтобы приложение работало в соответствии со своей платформой, его нужно «перевести» на понятный платформе язык, или добавить промежуточное звено-переводчик. К достоинствам можно отнести низкую стоимость разработки, ведь для этого иногда достаточно будет задействовать всего одного специалиста. А вот к недостаткам можно отнести возможные трудности в работе всех функций и задержки в реакции на действие пользователя (приложение может быть медлительным), также интерфейс будет достаточно простым и нужно будет его дополнительно дорабатывать.

В качестве языков программирования могут применяться *Java*, *Kotlin*, *Dart*, *C++*, *Python*, *C#* и другие. Однако для разработки «Абитуриент ГГТУ» был выбран именно *Java*, как наиболее распространенный и используемый.

1.4 Анализ инструментальных средств автоматизации разработки

Создание современных информационных систем представляет собой сложнейшую задачу, решение которой требует применения специальных методик и инструментов. Неудивительно, что в последнее время среди системных аналитиков и разработчиков значительно вырос интерес к *CASE*-технологиям и инструментальным *CASE*-средствам, позволяющим максимально систематизировать и автоматизировать все этапы разработки программного обеспечения.

Моделирование предметной области, как правило, выполняется с помощью *CASE*-средств. К таким средствам относятся *BPwin* (*AllFusion Process Modeler*), *Oracle Designer* (*Oracle*), *Rational Rose* (*Rational Software*) и остальные.

При разработке или закупке программного обеспечения модели бизнес-процессов служат прекрасным средством документирования потребностей, помогая обеспечить высокую эффективность инвестиций в сферу *IT*. В руках же системных аналитиков и разработчиков *BPwin* – еще и мощное средство моделирования процессов при создании корпоративных информационных систем (КИС).

BPwin предоставляет средства для изучения операций и управления операциями на различных уровнях детализации. Например, иногда бывает важно сосредоточиться на определенной части бизнеса организации. *BPwin* позволяет вам разделить сложный процесс на множество управляемых частей, обеспечивая

группам разработчиков модели возможность сосредоточиться на интересующих их аспектах. В итоге, эти различные аспекты могут быть согласованы и объединены, чтобы составить единый, целостный взгляд на предприятие. *BPwin* позволяет вам объединить отдельные модели в единую согласованную модель и достигнуть согласования проекта. *BPwin* помогает вам понять общее влияние изменений на существующие бизнес-процессы, обеспечивая быструю и эффективную адаптацию.

BPwin поддерживает три методологии моделирования: функциональное моделирование (*IDEF0*) – верхнеуровневое; описание бизнес-процессов (*IDEF3*) – поток работ и диаграммы потоков данных (*DFD*). Чаще всего применяется для создания функциональной модели предметной области на начальных этапах проектирования информационной системы, а также для анализа существующей или проектируемой ИС.

Функциональная модель включает в себя:

- поименованные процессы, функции или задачи, которые должны выполняться в системе;
- взаимодействия этих процессов, функций, задач с внешним миром и между собой.

На рисунке 1.12 представлен пример диаграммы, построенной в программе *AllFusion*.

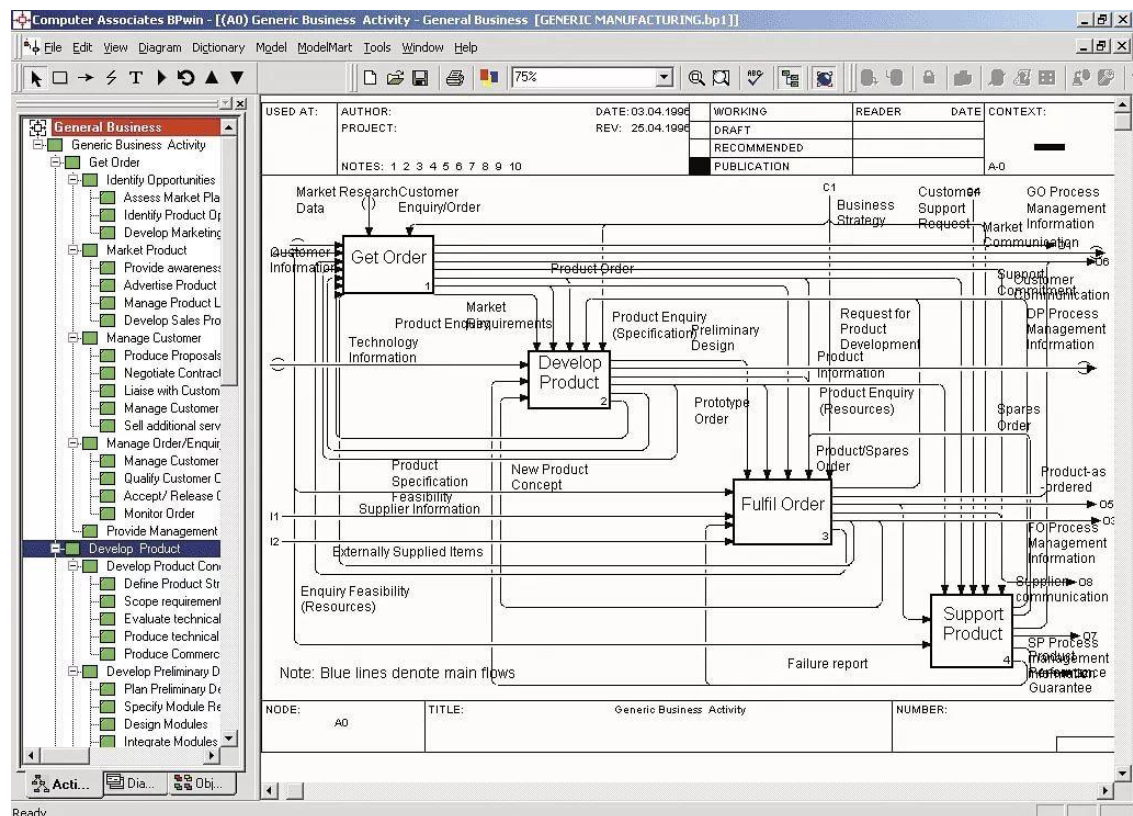


Рисунок 1.12 – Диаграмма в программе *AllFusion*

1.5 Техническое задание для мобильного приложения «Абитуриент ГГТУ»

Целью данной работы является повышения эффективности процесса взаимодействия абитуриентов и студентов и Гомельского государственного технического университета посредством разработки и, в последствии, ввода в эксплуатацию, информационно-справочного мобильного приложения для управления информированностью абитуриентов и студентов ГГТУ.

Задачами работы являются:

- произвести анализ взаимодействия абитуриентов с университетом;
- проанализировать существующие разработки и обоснование выбора технологии проектирования;
- написать техническое задание;
- спроектировать мобильное приложение;
- разработать функциональную и организационную структуру мобильного приложения;
- выбрать наиболее подходящие способы и средства реализации мобильного приложения;
- разработать мобильное приложение;
- протестировать работоспособность мобильного приложения;
- начать процесс ввода мобильного приложения в эксплуатацию.

Приложение должно иметь следующий функционал:

- свежие и актуальные новости и информация о планирующихся событиях и мероприятиях, чтобы постоянно держать пользователя в курсе и активно вовлекать его в жизнь университета;
- обеспечение доступа к сайтам и аккаунтам ГГТУ в социальных сетях;
- общая и необходимая информация об университете: контакты, адреса, местоположения на карте, руководство университета;
- регистрация;
- создание личного аккаунта внутри приложения;
- просмотр информации о ходе приемной кампании;
- калькулятор баллов: подсчет общего балла по введенным баллам вступительных испытаний и аттестата для получения информации о том, на какие специальности пользователь может пройти на основе прошлогодних проходных баллов.

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ И АЛГОРИТМЫ

2.1 Анализ предметной области

Программа предназначена для облегчения труда и уменьшения ошибок приемной комиссии ВУЗа. Она позволяет хранить данные о всех поступающих. Исходя из поданных документов определяет какие льготы обладает абитуриент, так же по завершению приема можно автоматически рассчитать кто попал в институт и на какую форму обучения.

Программное обеспечение (ПО) для автоматизации деятельности работников приемной комиссии высшего учебного заведения относится к такому типу прикладного ПО, как информационные системы (ИС), неотъемлемой частью которых является база данных и средства ее обработки. Таким образом, при создании приложения необходимо спроектировать и реализовать базу данных (БД) и разработать удобный интерфейс для взаимодействия с ней, включающий в себя функции добавления, удаления, фильтрации, а также средства редактирования данных.

В процессе своей деятельности сотрудники приемной комиссии выполняют такие действия, как прием от абитуриентов заявления и копии документов, с присвоением им идентификационного номера, выполняют формирование списков на вступительные испытания и формирование по итогам экзаменов списков на зачисление абитуриентов на обучение на бюджетной или коммерческой основе.

В качестве модели представления данных была выбрана реляционная модель данных предметной области, построенная на взаимосвязи отношений. В настоящее время реляционная модель данных является фактическим стандартом, на который ориентируются практически все современные коммерческие системы управления базами данных (СУБД).

Анализ предметной области позволил выделить сущности, их атрибуты и связи между сущностями. В качестве инструмента моделирования базы данных используется *CASE (computer-aided software engineering)* система *AllFusion ERwin Data Modeler*.

2.2 Алгоритм работы мобильного приложения

Идея работы алгоритма программы состоит в помощи абитуриентам в ходе приемной кампании. Абитуриент при входе в приложение может зарегистрироваться либо войти, зайти в личный кабинет и осуществить ввод баллов по вступительным испытаниям, просмотреть информацию о ходе приемной кампании.

Также абитуриент должен получать новости о приемной кампании, новости университета и уведомления о важных событиях.

Помимо всего прочего, абитуриент может воспользоваться так называемым калькулятором баллов: ввести баллы вступительных испытаний и аттестата

и, получив подсчитанный общий балл, получить информацию о том, на какие специальности он может пройти на основе проходных баллов прошлого года.

Для реализации данного алгоритма необходимо составить функциональную и информационную модели программного комплекса. Описание моделей представлено ниже.

2.3 Функциональная модель мобильного приложения

Под функциональным моделированием понимается процесс построения функциональных моделей объекта автоматизации, либо отдельных процессов. Функциональная модель – суть ориентированный граф, вершинами которого являются выполняемые функции, а дугами либо элементы подсистемы, либо потоки информации, вещества или энергии.

IDEF0 – Function Modeling – методология функционального моделирования, предназначенная для формализации и описания бизнес-процессов. Отличительной особенностью IDEF0 является ее акцент на соподчиненность объектов. В IDEF0 рассматриваются логические отношения между работами, а не их временная последовательность:

- функция на диаграмме представлена блоком, имеющим 3 входа (снизу, слева, сверху) и один выход (справа);
- потоки информации об управлении или ограничениях входят в блок сверху;
- информация, которая подвергается обработке, показана с левой стороны блока;
- результаты выхода показаны с правой стороны;
- механизм (человек или автоматизированная система), который осуществляет операцию, представляется дугой, входящей в блок снизу [1].

Проектируемое мобильное приложение, разрабатываемое для абитуриентов, должно отвечать определенным функциональным требованиям.

Концептуальное моделирование предметной области – это структуризация предметной области, для которой разрабатывается приложение.

Целью концептуального моделирования предметной области является выявление перечня и иерархии предметов, объектов, факторов и явлений, полный набор которых позволяет реализовать поставленные перед приложением цели и задачи.

Концептуальная модель (содержательная модель) – это абстрактная модель, определяющая состав и структуру объекта, свойства элементов и причинно-следственные связи, присущие анализируемому объекту и существенные для достижения целей моделирования. В концептуальной модели обычно в словесной форме приводятся сведения о природе и параметрах (характеристиках) элементарных явлений исследуемого объекта, о виде и степени взаимодействия между ними, о месте и значении каждого элементарного явления в общем процессе функционирования объекта.

Концептуальное моделирование будет выполняться при помощи графического языка для визуализации, спецификации конструирования и документирования систем, в котором большая роль принадлежит, программному обеспечению: *UML (Unified Modeling Language, UML)*. При помощи *UML* можно разработать детальный план будущей системы, в которой будут содержаться не только ее концептуальные элементы, такие как 10 системные функции и бизнес-процессы, но и особенности, такие как классы, написанные на специальных языках программирования, схемы базы данных.

UML представляет собой графическую нотацию, которая предназначена для моделирования и описания всех процессов, протекающих в процессе разработки. Основу *UML* представляют диаграммы, которые различаются по типам и предназначены для моделирования различных аспектов разработки.

Для моделирования бизнес-процессов информационно-аналитической системы используется инструмент для моделирования, анализа документирования и оптимизации бизнес-процессов *CASE (computer-aided software engineering)* система *AllFusion ERwin Data Modeler*.

AllFusion позволяет проводить описание, анализ и моделирование бизнес-процессов, а также строить систему классификации и кодирования.

Были описаны основные понятия концептуального и функционального моделирования, далее перейдем к определению целей модели.

На рисунке 2.1 изображена контекстная диаграмма функциональной модели мобильного приложения.



Рисунок 2.1 – Контекстная диаграмма функциональной модели мобильного приложения

2.4 Сущности и связи между ними

Исходя из анализа задачи, стало понятно, что достаточно будет выделить в программе две роли: абитуриент и студент, для которого нет необходимости проводить авторизацию.

На рисунке 2.2 представлена диаграмма вариантов использования приложения.

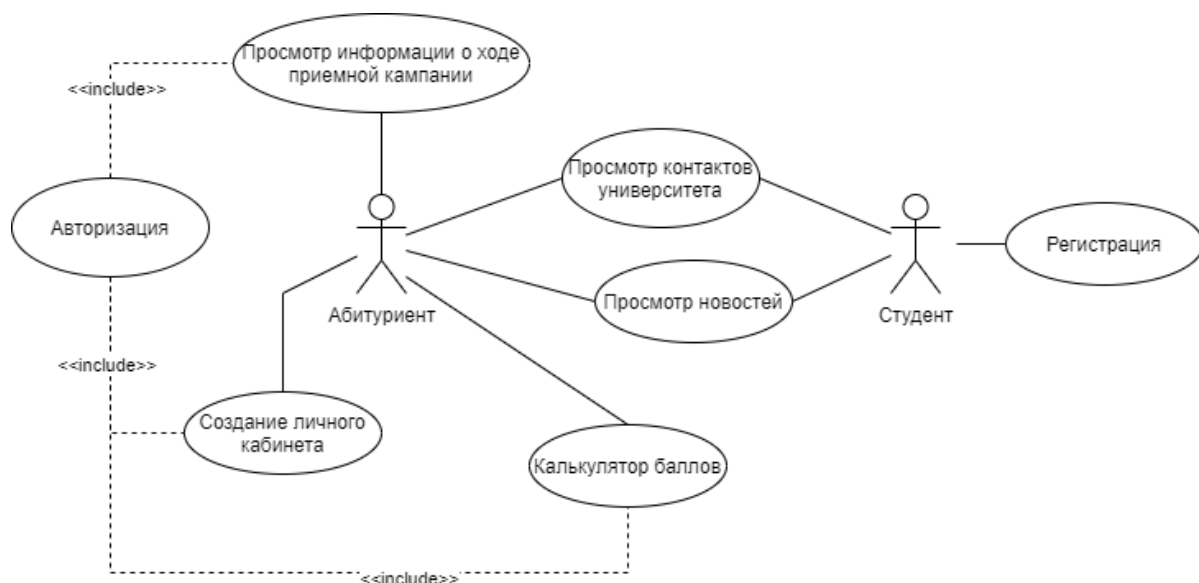


Рисунок 2.2 – Диаграмма вариантов использования

Прецедент «Авторизация» реализует механизм авторизации в приложении, после чего пользователь пройдет аутентификацию как абитуриент.

Прецедент «Просмотр информации о ходе приемной кампании» реализует просмотр информации, в результате которого пользователь получит данные о ходе приемной кампании.

Прецедент «Создание личного кабинета» реализует возможность ввести данные, необходимые для создания заявки на подачу документов.

Прецедент «Просмотр контактов университета» реализует просмотр основной информации об университете: контакты, данные об администрации.

Прецедент «Просмотр новостей» позволяет просматривать свежие новости о жизни университета.

Прецедент «Калькулятор баллов» реализует возможность ввести баллы вступительных испытаний и на их основе подсчитать общий балл, чтобы, на основе прошлогодней информации, абитуриент мог понять, на какие специальности может пройти.

Прецедент «Регистрация» позволяет любому пользователю зарегистрироваться в приложении как студент [2].

Список прецедентов представлен в таблице 2.1

Таблица 2.1 – Список прецедентов

№	Прецедент	Действие исполнителя	Ответ системы
1	2	3	4
1	Авторизация	Осуществляет вход в систему как абитуриент	Авторизация пользователя в приложении

Продолжение таблицы 2.1

1	2	3	4
2	Просмотр информации о ходе приемной кампании	Просматривает информацию о ходе приемной кампании	Открытие информации о ходе приемной кампании
3	Создание личного кабинета	Создает личный кабинет, вводит баллы вступительных испытаний	Изменяет данные в базе данных
4	Просмотр контактов университета	Просматривает информацию об университете	Открывает информацию
5	Просмотр новостей	Просматривает новости университета	Открывает информацию
6	Калькулятор баллов	Вводит баллы вступительных испытаний	Подсчитывает и выводит информацию о полученном балле абитуриента
7	Регистрация	Вводит данные для регистрации в приложении	Изменяет данные в базе данных

Таким образом, были выделены основные функции и прецеденты разрабатываемого мобильного приложения.

2.5 Архитектура мобильного приложения

Разрабатываемое мобильное приложение является приложением с двухзвенной архитектурой «клиент-сервер».

Архитектура «клиент-сервер» определяет общие принципы организации взаимодействия в сети, где имеются серверы, узлы-поставщики некоторых специфичных функций (сервисов) и клиенты (потребители этих функций). Практические реализации такой архитектуры называются клиент-серверными технологиями.

Двухзвенная архитектура – распределение трех базовых компонентов между двумя узлами (клиентом и сервером). Двухзвенная архитектура используется в клиент-серверных системах, где сервер отвечает на клиентские запросы напрямую и в полном объеме (рисунок 2.3).

Расположение компонентов на стороне клиента или сервера определяет следующие основные модели их взаимодействия в рамках двухзвенной архитектуры:

- сервер терминалов – распределенное представление данных;
- файл-сервер – доступ к удаленной базе данных и файловым ресурсам;
- сервер базы данных – удаленное представление данных;

– сервер приложений – удаленное приложение.

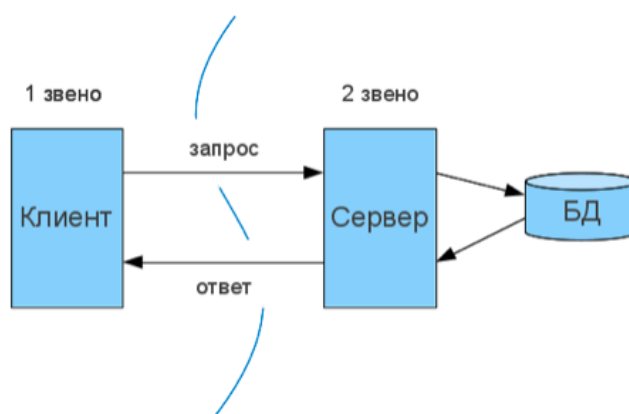


Рисунок 2.3 – Двухзвенная архитектура

Перечисленные модели с вариациями представлены на рисунке 2.4.

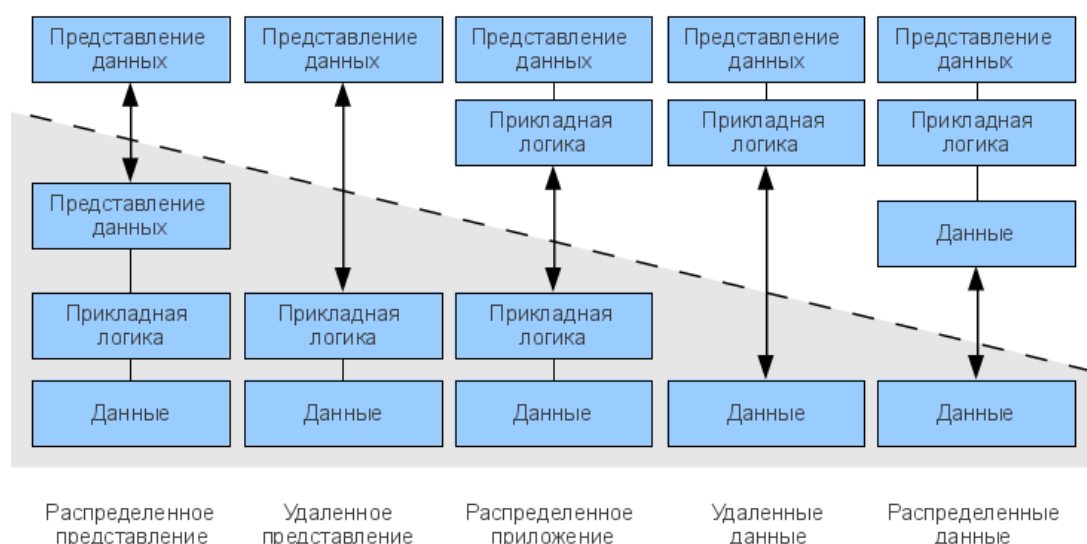


Рисунок 2.4 – Модели клиент-серверного взаимодействия

Исторически первой появилась модель распределенного представления данных (модель сервер терминалов). Она реализовывалась на универсальной ЭВМ (мэйнфрейме), выступавшей в роли сервера, с подключенными к ней алфавитно-цифровыми терминалами. Пользователи выполняли ввод данных с клавиатуры терминала, которые затем передавались на мэйнфрейм и там выполнялась их обработка, включая формирование «картинки» с результатами. Эта «картинка» и возвращалась пользователю на экран терминала.

С появлением персональных компьютеров и локальных сетей, была реализована модель файлового сервера, представлявшего доступ файловым ресурсам, в т.ч. и к удаленной базе данных. В этом случае выделенный узел сети является

файловым сервером, на котором размещены файлы базы данных. На клиентах выполняются приложения, в которых совмещены компонент представления и прикладной компонент (СУБД и прикладная программа), использующие подключенную удаленную базу как локальный файл. Протоколы обмена при этом представляют набор низкоуровневых вызовов операций файловой системы.

Такая модель показала свою неэффективность ввиду того, что при активной работе с таблицами БД возникает большая нагрузка на сеть. Частичным решением является поддержка тиражирования (репликации) таблиц и запросов. В этом случае, например при изменении данных, обновляется не вся таблица, а только модифицированная ее часть.

С появлением специализированных СУБД появилась возможность реализации другой модели доступа к удаленной базе данных – модели сервера баз данных. В этом случае ядро СУБД функционирует на сервере, прикладная программа на клиенте, а протокол обмена обеспечивается с помощью языка *SQL*. Такой подход по сравнению с файловым сервером ведет к уменьшению загрузки сети и унификации интерфейса «клиент-сервер». Однако, сетевой трафик остается достаточно высоким, кроме того, по-прежнему невозможно удовлетворительное администрирование приложений, поскольку в одной программе совмещаются различные функции.

С разработкой и внедрением на уровне серверов баз данных механизма хранимых процедур появилась концепция активного сервера БД. В этом случае часть функций прикладного компонента реализованы в виде хранимых процедур, выполняемых на стороне сервера. Остальная прикладная логика выполняется на клиентской стороне. Протокол взаимодействия – соответствующий диалект языка *SQL*.

Преимущества такого подхода очевидны:

- возможно централизованное администрирование прикладных функций;
- снижение стоимости владения системой (*TOC, total cost of ownership*) за счет аренды сервера, а не его покупки;
- значительное снижение сетевого трафика (т.к. передаются не *SQL*-запросы, а вызовы хранимых процедур).

Основной недостаток – ограниченность средств разработки хранимых процедур по сравнению с языками высокого уровня.

Реализация прикладного компонента на стороне сервера представляет следующую модель – сервер приложений. Перенос функций прикладного компонента на сервер снижает требования к конфигурации клиентов и упрощает администрирование, но представляет повышенные требования к производительности, безопасности и надежности сервера.

В настоящее время намечается тенденция возврата к тому, с чего начиналась клиент-серверная архитектура – к централизации вычислений на основе модели терминал-сервера. В современной реинкарнации терминалы отличаются от своих алфавитно-цифровых предков тем, что имея минимум программных и аппаратных средств, представляют мультимедийные возможности (в т.ч. графический пользовательский интерфейс). Работу терминалов обеспечивает

высокопроизводительный сервер, куда вынесено все, вплоть до виртуальных драйверов устройств, включая драйверы видеоподсистемы.

2.6 Информационная модель мобильного приложения

Большинство автоматизированных систем тем или иным образом используют структурированную информацию. В современных приложениях такие упорядоченные данные принято хранить в базах данных – особых файлах, использование которых вместе со специальными программными средствами позволяет пользователю, как просматривать необходимую информацию, так и манипулировать ею.

База данных – совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных.

База данных нужна для долгого хранения большого количества данных, которые не пропадут после закрытия приложения. Кроме того, база данных на основе *SQL* позволяет производить различные манипуляции по выборке.

Облачные технологии – это концепция обработки данных, в которой вычисление и хранение данных осуществляется в облаке – удаленном дата-центре. Все функции по предоставлению и управлению низкоуровневой инфраструктурой берет на себя «облако», полностью скрывая эти детали от пользователя. Среди плюсов организации таких вычислений можно отметить следующие:

- доступность. Услуга предоставляется через Интернет и является доступной круглосуточно при условии функционирования сервера-провайдера;
- использование технологии виртуализации, позволяющая сделать вычислительные ресурсы автономными и взаимно независимыми;
- расширяемость, масштабируемость. Обработка и хранение данных осуществляется на серверах удаленных дата-центров и распределена по физическим машинам, предоставляя ровно такие мощности, какие необходимы пользователю;
- условие оплаты по подписке и за фактический уровень потребленных ресурсов («*pay as you go*») – потребитель платит ровно за то количество работы вычислительных мощностей, сколько им было использовано. Кроме того, такая организация оплаты существенно сокращает уровень капитальных издержек, что критично важно при выборе технологии для стартапа;
- легкость в использовании. Пользователю не нужно заботиться об инфраструктуре и сопровождении процессов обработки и хранения данных.

Предоставление в пользование вычислительных мощностей и баз данных дата-центра может осуществляться в нескольких вариантах: *SaaS*, *PaaS*, *HaaS*, *IaaS*, *CaaS* – *as a Service* – в качестве Интернет-сервиса. *S*, *P*, *H*, *I*, *C* – *Software*, *Platform*, *Hardware*, *Infrastructure*, *Communication* – соответственно, программное обеспечение, платформа, аппаратное обеспечение, инфраструктура, коммуникации. Выбор варианта использования вычислительных мощностей напрямую зависит от целей деятельности.

Одной из составляющих облачных технологий, предоставляемых компанией *Microsoft*, является услуга по предоставлению функционала облачной базы данных под названием *SQL Azure*.

SQL Azure – проекция традиционного *SQL Server* на облако, предоставляющая возможности для работы с базой данных посредством интернет-сервисов. Эта технология позволяет хранить структурированную и неструктурированную информацию, исполнять реляционные запросы, а также предоставляет функционал для осуществления поиска, создания аналитических отчетов, интеграции и синхронизации данных. На данный момент *SQL Azure* поддерживает сервис реляционных баз данных, имеющий название *SQL Azure Database*.

SQL Azure Database – облачная платформа реляционной базы данных, построенная на технологиях *SQL Server*. При использовании этой платформы можно легко построить в облаке проект реляционной базы данных со всеми преимуществами, предоставляемыми любой облачной технологией. Кроме того, *SQL Azure* предоставляет высокий уровень безопасности со встроенной защитой данных, самовосстановлением и системой резервного копирования.

Двумя главными нововведениями в *SQL Azure* являются переход к реляционной структуре и использование *T-SQL*. Эти новшества предоставляют разработчикам целый набор преимуществ. Во-первых, преимущества реляционной структуры данных (удобство в хранении и обработки). Во-вторых, использование привычного языка запросов, что позволяет использовать навыки работы с *SQL Server*, а также работать с библиотеками *ADO.NET* и интерфейсами *ODBC* и *OleDB*.

Благодаря переходу на *T-SQL*, пользователи имеют возможность не только использовать стандартный функционал *REST*-протокола (получение/добавление/удаление единицы информации), но также и все преимущества *T-SQL* (создание/удаление таблиц/связей, выполнение разнообразных запросов и проч.). Однако разработчики все же оставили возможность доступа по протоколу *REST* в *SQL Azure*, которую затем можно реализовать через «привязку» к *ADO.NET*. Подключение к базам данных может осуществляться как из веб (например *Azure*), так и из локального приложения, а управление данными предоставляется функционалом *SQL Server Management Studio*. Таким образом, как и все концепции *SQL Azure*, неизменным осталось поддержка высокого уровня масштабируемости, постоянной доступности (99,9% ежемесячной доступности согласно соглашению с пользователем), самовосстановления, множественное зеркалирование и многое другое.

Если в предыдущих версиях не осуществлялась поддержка схемы данных, была представлена минимальная реляционная функциональность, то теперь пользователи имеют возможность работать с традиционной *RDBMS* моделью данных. Что касается работы с нереляционной структурой данных, то теперь ее можно осуществить в хранилище *Windows Azure storage* (с более низким уровнем оплаты, всего 15 центов за 1 Гб и 1 цент за каждые 10 тысяч транзакций по сравнению с минимальным уровнем оплаты *SQL Azure* 9,99 \$ за 1 Гб). Однако следует помнить, что *SQL Azure Database* имеет некоторые преимущества по сравнению

с обычной базой данных в облаке: в *SQL Azure Database* предоставляется также инфраструктура по настройке, установке и управлению базами данных.

Таким образом, в наиболее общем случае использование облачной базы данных *SQL Azure* может основываться на уже имеющихся навыках работы с локальной БД, поэтому перенос и последующее использование БД в облаке не требует особенных дополнительных средств и усилий [3].

В базе данных будут храниться все данные абитуриентов, данные о факультетах и специальностях, формах обучения.

2.7 Проектирование базы данных

В ходе анализа предметной области были выделены следующие сущности:

- пользователь (*User*);
- абитуриент (*Enrollee*);
- факультет (*Faculty*);
- специальность (*Speciality*);
- форма обучения (*Education form*);
- приоритет (*Priority*).

Сущность «Пользователь» должна хранить основную информацию о пользователе, прошедшем регистрацию в приложении. Атрибуты сущности:

- *ID*;
- электронная почта;
- пароль.

Сущность «Форма обучения» должна хранить основную информацию о формах обучения. Атрибуты сущности:

- *ID*;
- название формы обучения.

Сущность «Абитуриент» должна хранить основную информацию об абитуриенте, участвующем в приемной кампании. Атрибуты сущности:

- *ID*;
- *ID* пользователя;
- фамилия;
- имя;
- отчество;
- номер паспорта;
- адрес;
- номер телефона;
- форма обучения;
- участие на условиях целевой подготовки;
- участие без вступительных испытаний;
- участие вне конкурса;
- участие в конкурсе на бюджетную форму обучения;
- балл вступительного испытания по первому профильному предмету;

- балл вступительного испытания по второму профильному предмету;
- балл вступительного испытания по третьему предмету (при наличии);
- средний балл документа об образовании, умноженный на 10;
- согласие (несогласие) на участие в конкурсе на платную форму обучения.

Сущность «Факультет» должна хранить информацию о факультете, на который осуществляется прием документов. В ней должны присутствовать следующие характеристики:

- *ID*;
- полное наименование факультета;
- краткое наименование факультета.

Сущность «Специальность» должна хранить информацию о специальности факультета, на которую осуществляется прием документов. В ней должны присутствовать следующие характеристики:

- *ID*;
- *ID* факультета, к которому данная специальность относится;
- полное наименование специальности;
- краткое наименование специальности;
- план приема на бюджетную форму обучения;
- план приема на платную форму обучения;
- план приема на условиях целевой подготовки.

Сущность «Приоритета» должна хранить информацию о специальности, на которую абитуриент желает быть зачисленным в указанном им порядке. В ней должны присутствовать следующие характеристики:

- *ID*;
- *ID* абитуриента;
- *ID* специальности;
- порядковый номер.

Были выделены связи между сущностями.

Связь между сущностями «Пользователь» и «Абитуриент» – один-к-одному. Данная связь необходима для того, чтобы отделить данные для входа в приложение от личных данных абитуриента, так как в приложение имеют право входить и обычные студенты.

Связь между сущностями «Абитуриент» и «Приоритет» – один-ко-многим. Связь означает, что абитуриент может иметь несколько единиц приоритета, в то время как одна единица приоритета принадлежит только одному абитуриенту.

Связь между сущностями «Абитуриент» и «Форма обучения» – один-к-одному, так как одновременно абитуриент может проходить по конкурсу только на одну форму обучения.

Связь между сущностями «Факультет» и «Специальность» – один-ко-многим. Это означает, что каждому факультету ставится в соответствие несколько специальностей, вместе с тем каждая специальность может относиться только к

одному факультету.

На рисунке 2.5 представлена логическая модель базы данных.

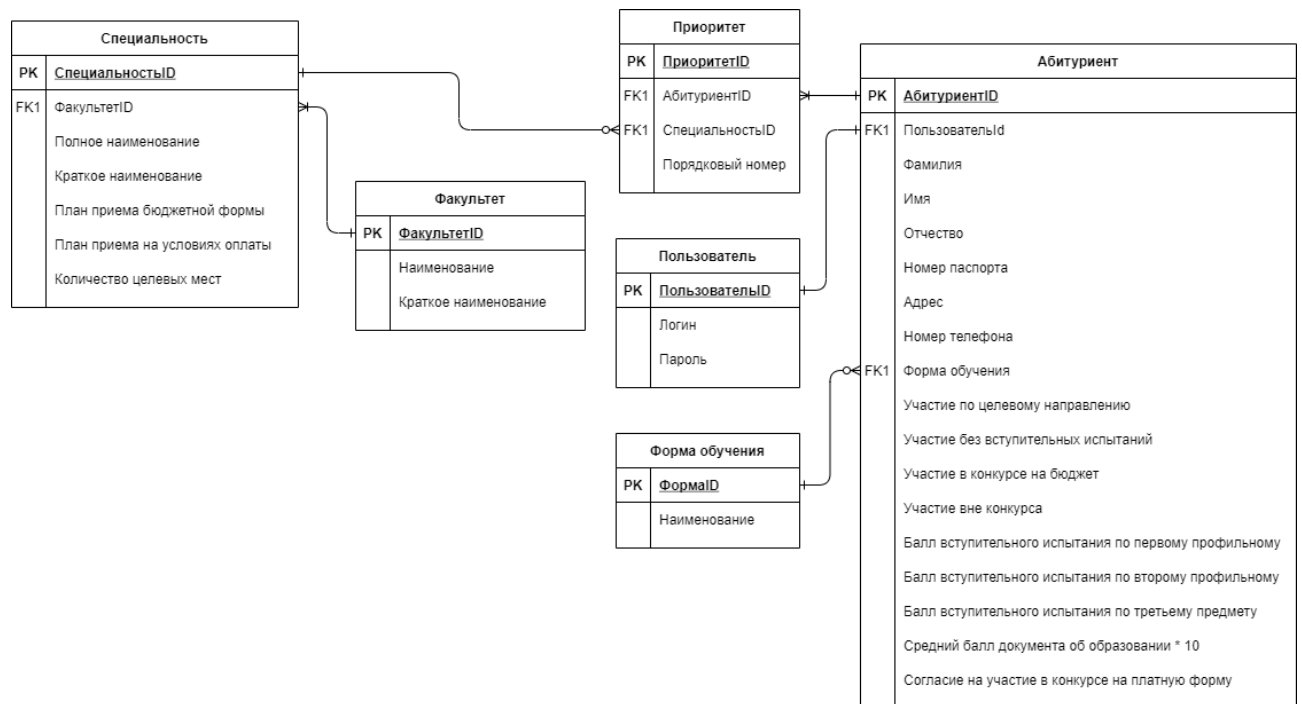


Рисунок 2.5 – Логическая модель базы данных

3 СТРУКТУРА И ОСНОВНЫЕ АЛГОРИТМЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

3.1 Общая структура программного обеспечения

Программное обеспечение состоит из двух основных частей:

- клиентская;
- серверная.

Клиентская часть программного обеспечения представлена непосредственно мобильным приложением. Основным файлом в данном проекте является файл *AndroidManifest.xml*. Этот файл описывает конфигурацию приложения, его разрешения. Также в данном файле находятся описания *Activity* (Активностей). Файл *AndroidManifest.xml* представлен на рисунке 3.1.

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.diplomaproject">
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />

    <uses-feature android:name="android.hardware.type.watch" />

    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>

    <application...>

</manifest>
```

Рисунок 3.1 – Файл *AndroidManifest.xml*

Приложению необходимо иметь доступ к интернет-соединению, следовательно, в файле манифеста прописаны следующие разрешения:

- `<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />`;
- `<uses-permission android:name="android.permission.INTERNET" />`.

Также в приложении поддерживается функция скачивания файлов, для сохранения которых необходимо разрешение `<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>`.

Приложение состоит из *Activity*, которые в обязательном порядке регистрируются в файле манифеста (рисунок 3.2).

```

<activity
    android:name=".businesslogic.account.ViewAccountActivity"
    android:label="Просмотр аккаунта"
    android:parentActivityName=".MainActivity" />
<activity
    android:name=".businesslogic.LoginActivity"
    android:label="Вход в личный кабинет"
    android:parentActivityName=".MainActivity" />
<activity
    android:name=".businesslogic.account.CreateAccountActivity"
    android:label="Создание аккаунта"
    android:parentActivityName=".MainActivity" />
<activity
    android:name=".MainActivity"
    android:label="Абитуриент ГГТУ"
    android:theme="@style/Theme.DiplomaProject.NoActionBar">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>

```

Рисунок 3.2 – Регистрация *Activity* в *AndroidManifest.xml*

Клиентская часть состоит из трех логических частей:

- слой представления;
- слой бизнес-логики;
- слой доступа к данным.

Слой представления включает в себя все окна пользовательского интерфейса и находится в папке *layout* (рисунок 3.3).

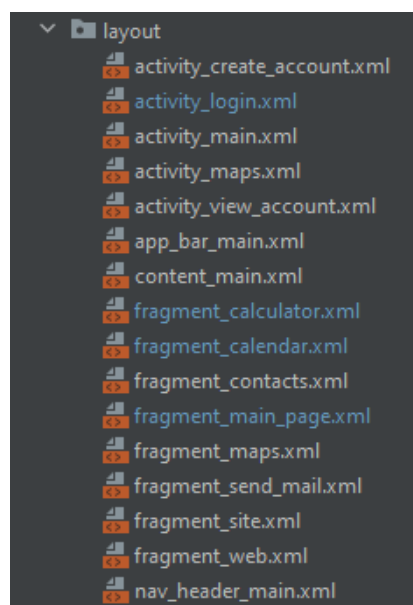


Рисунок 3.3 – Слой представления

Слой бизнес-логики включает классы, отвечающие за логику работы всех окон пользовательского интерфейса, обрабатывающие полученные от них данные, а также механизмы реагирования на действия пользователя. Слой бизнес-логики представлен на рисунке 3.4.

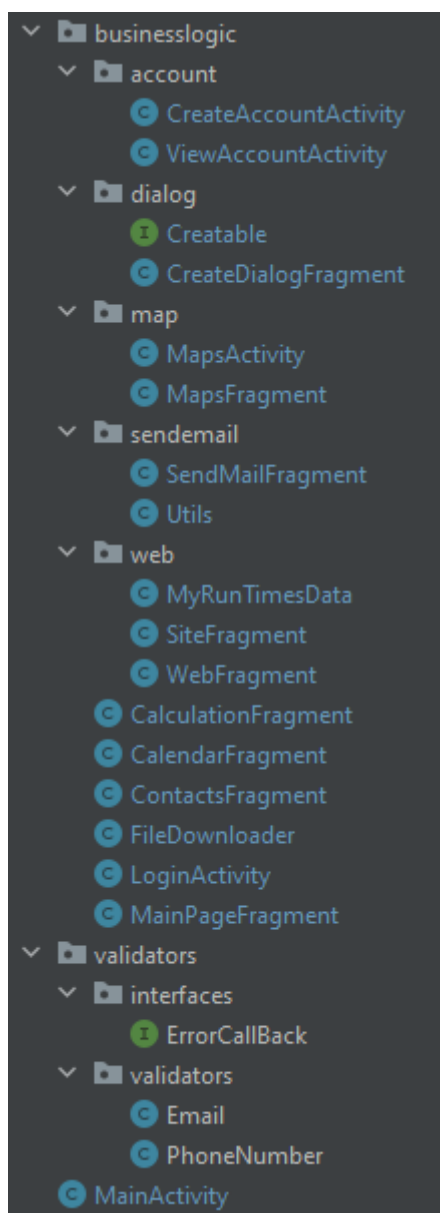


Рисунок 3.4 – Слой бизнес-логики

Слой доступа к данным включает в себя основные классы для подключения и работы с базой данных (рисунок 3.5).

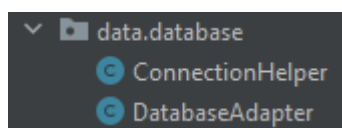


Рисунок 3.5 – Слой доступа к данным

Серверная часть экосистемы проекта представлена облачной базой данных *SQL Azure*, функционал которой является одной из составляющих облачных технологий, предоставляемых компанией *Microsoft*. Для подключения к базе данных использовался *JDBC* – стандарт взаимодействия *Java*-приложений и различных СУБД. Он основан на концепции драйверов, позволяющих получить соединение с базой данных по *URL*. Пример подключения к базе данных представлен на рисунке 3.6.

```
public Connection connection(){
    host = "gstuenrollee.database.windows.net";
    database = "EnrolleeDB";
    username = "enrolleeadmin";
    pass = "12345678Mm";
    port = "1433";

    StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
    StrictMode.setThreadPolicy(policy);
    Connection connection = null;
    String ConnectionURL = null;

    try{
        Class.forName("net.sourceforge.jtds.jdbc.Driver");
        ConnectionURL= "jdbc:jtds:sqlserver://" + host + ":" + port + ";" + "databasename="+
            database + ";user="+username+";password="+pass+";";
        connection = DriverManager.getConnection(ConnectionURL);
    }
    catch (Exception exception){
        Log.e( tag: "Error", exception.getMessage());
    }
    return connection;
}
```

Рисунок 3.6 – Пример подключения к базе данных

Также в экосистеме программного продукта реализован *Python*-скрипт, который реализует поставку данных с сайта в базу данных при помощи пакета *Beautiful Soup*. Пример полученных в приложении данных с сайта представлен на рисунке 3.7.

```
C:\Users\masha\CalendarScrapperService\venv\Scripts\python.exe "C:\Program Files\JetBrains\PyCharm Community Edition 2020.3.3\plugins\pyth
Connected to pydev debugger (build 203.7148.72)
[{'event_date': '2021-05-04', 'event_title': 'Регистрируемся на Централизованное тестирование!', 'event_url': '/news/2021-05-04-000000'}]
```

Рисунок 3.7 – Полученные с сайта данные

3.2 Основные алгоритмы программного обеспечения

В ходе разработки приложения было принято решение сделать его доступным для всех желающих, в то время как исключительно абитуриент может войти в свой личный кабинет.

При запуске приложения пользователю открывается главная страница, оформленная в виде плиточного интерфейса (рисунок 3.8).



Рисунок 3.8 – Главная страница

На главной странице пользователь имеет возможность перейти по интересующему его пункту и попасть на встроенную в приложение версию соответствующего сайта, данные с которого при этом загружаются асинхронно для ускорения работы приложения (рисунок 3.9).

Алгоритм загрузки сайта в приложение реализован при помощи компонента *WebView*. Преимуществом данного способа является быстрая реализация и рациональность решения для создания мобильной версии сайта.

Также в приложение есть боковое меню, реализованное при помощи компонента *DrawerLayout* и виджета *NavigationView* (рисунок 3.10). Элементы бокового меню являются различными сервисами, которыми может воспользоваться любой пользователь приложения, такими как:

- просмотр необходимой информации об университете: контакты, адрес, руководство университета;
- местоположение университета на карте;

- открытие сайта университета внутри приложения;
- связь с разработчиком приложения;
- просмотр важных для абитуриентов новостей, реализованный в виде календаря;
- калькулятор баллов: вычисление балла пользователя на основе подсчета по введенным данным.

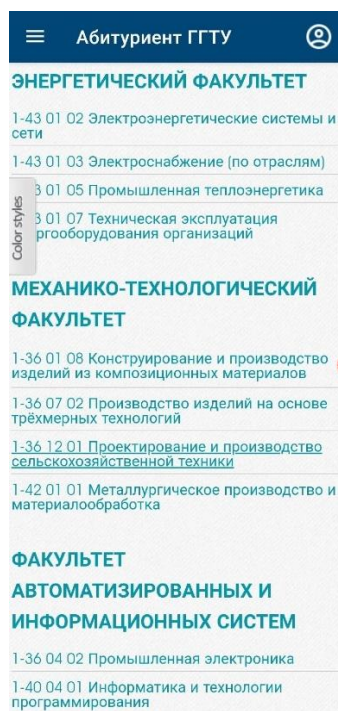


Рисунок 3.9 – Загрузка сайта внутри приложения

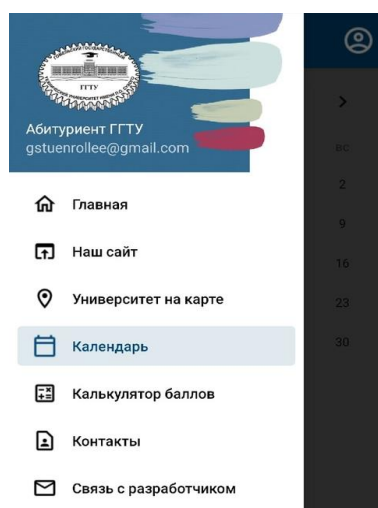


Рисунок 3.10 – Боковое меню

Помимо всех вышеперечисленных функций, в приложении присутствует функционал, предназначенный непосредственно для абитуриента: создание личного кабинета (либо вход в него), заполнение данных, отправляемых в приемную

комиссию, а также просмотр этих данных. При нажатии на кнопку перехода в личный кабинет, пользователь попадает на страницу авторизации и регистрации (рисунок 3.11).

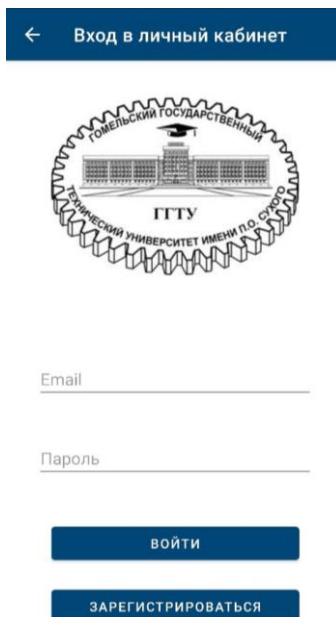


Рисунок 3.11 – Страница авторизации и регистрации

Окно создания личного кабинета представлено на рисунке 3.12, окно просмотра данных абитуриента – на рисунке 3.13.




Рисунок 3.12 – Создание личного кабинета

Просмотр аккаунта	
Фамилия	Процкая
Имя	Мария
Отчество	Александровна
Серия и номер паспорта	НВ1234567
Адрес	Гомель, ул. Ленина, 3/6
Дата рождения	2001-10-04
Номер телефона	+375251234567
Форма обучения	Очная
Участие в конкурсе по целевому направлению	<input type="checkbox"/>
Участие в конкурсе без вступительных экзаменов	<input type="checkbox"/>
Участие в конкурсе на бюджет	<input checked="" type="checkbox"/>
Участие вне конкурса	<input type="checkbox"/>
Балл по первому профильному предмету	98
Балл по второму профильному предмету	67
Балл по третьему	

Рисунок 3.13 – Просмотр данных абитуриента

Окна создания личного кабинета и просмотра данных созданы при помощи элемента *ScrollView*, так как содержат большое количество информации, которую нужно поместить на экране, из-за чего приходится использовать полосы прокрутки. *ScrollView* является контейнерным элементом и наследуется от *ViewGroup*. На панели инструментов компоненты можно найти в разделе *Containers*. В контейнер *ScrollView* можно размещать только один дочерний элемент (обычно *LinearLayout*), который в свою очередь может быть контейнером для других элементов.

Окно просмотра контактов университета представлено на рисунке 3.14.

☰

Контакты

👤

Учреждение образования «Гомельский государственный технический университет имени П.О.Сухого» (ГГТУ им. П.О. Сухого)

Пр-т Октября, 48, 246746, г. Гомель, Республика Беларусь

Телефон: [\(+375 232\) 22-46-36](tel:+375232224636)

Факс: [\(+375 232\) 26-02-87](tel:+375232260287)

E-mail: rector@gstu.by

<http://www.gstu.by>

Режим работы: понедельник - пятница
8.00 - 12.00; 13.00 - 17.00

Рисунок 3.14 – Просмотр контактов университета

На рисунке 3.15 представлено отображение университета на карте.



Рисунок 3.15 – Университет на карте

На рисунке 3.16 представлено окно для связи с разработчиком.

A screenshot of a mobile application window titled "Связь с разработчик..." with a back arrow and a user icon. The form contains a "Тема:" label followed by a text input field with the placeholder "Введите тему письма". Below this is a large text area with the placeholder "Введите текст письма". At the bottom right, there is a dark blue button labeled "ОТПРАВИТЬ".

Рисунок 3.16 – Окно для связи с разработчиком

На рисунке 3.17 представлено окно просмотра важных новостей в виде календаря.



Рисунок 3.17 – Окно календаря

На рисунке 3.18 представлен калькулятор баллов.

The screenshot shows a web interface for a calculator. At the top, there is a dark blue header with a hamburger menu icon, the text "Калькулятор баллов", and a user profile icon. Below the header, the text "Введите ваши баллы" is displayed. There are four input fields with labels and values: "Первый профильный предмет" with value 55, "Второй профильный предмет" with value 59, "Третий предмет" with value 63, and "Балл документа об образовании" with value 48. Below these fields, the word "РЕЗУЛЬТАТ" is displayed, followed by the large number "225". At the bottom, there is a dark blue button with the text "ПОДСЧИТАТЬ".

Рисунок 3.18 – Калькулятор баллов

Рассмотрим механизм загрузки данных в календарь. Сайт университета не имеет публичного *API*, поэтому возникла необходимость автоматизированного сбора данных с *html*-страницы. Для этого был создан отдельный *Python*-скрипт, являющийся частью разрабатываемой экосистемы. В нем при помощи пакета *Beautiful Soup* создается дерево синтаксического анализа для анализируемых страниц, при помощи которого извлекаются данные (дата события и информация о нем) с сайта, которые, в последствии, загружаются в словарь, который, в свою очередь, загружается в базу данных. Далее данные из базы данных загружаются в приложение и отображаются в виде информации о событии, появляющейся при нажатии на отмеченный день, и отметки в календаре в месте соответствующей даты. Информация о событии содержит в себе ссылку на сайт с соответствующей новостью.

Для реализации отправки письма разработчикам необходимо ввести тему и содержание письма и нажать на кнопку отправить. Данные письма как экстра-содержимое вместе с действием «*android.content.Intent.ACTION_SEND*» передаются в специальный механизм для описания операции – *Intent* (Намерение). *Intent* в свою очередь запускает отправку письма.

Приложение является дружелюбным и интуитивно понятным для пользователя: в случае возникновения исключительных ситуаций (например, проблемы с подключением к базе данных или ввод неверных данных) на экране сразу же появляется соответствующее уведомление. Примеры уведомлений представлены на рисунках 3.19-3.20.

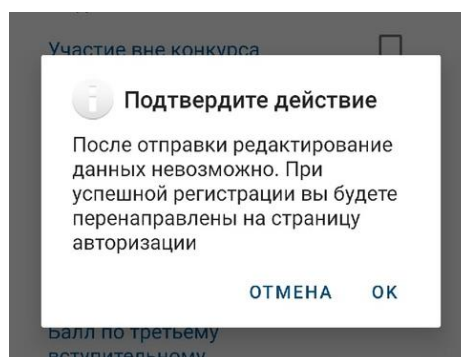


Рисунок 3.19 – Уведомление для подтверждения действия при создании личного кабинета

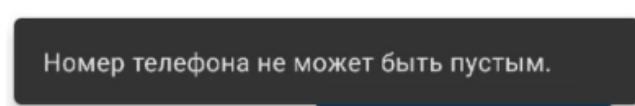


Рисунок 3.20 – Уведомление о вводе неверного значения

4 ТЕСТИРОВАНИЕ, ВЕРИФИКАЦИЯ И ВАЛИДАЦИЯ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Тестирование программного обеспечения – процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и ее ожидаемым поведением на конечном наборе тестов, выбранных определенным образом.

В разное время и в различных источниках тестированию давались различные определения, в том числе:

- процесс выполнения программы с целью нахождения ошибок;
- интеллектуальная дисциплина, имеющая целью получение надежного программного обеспечения без излишних усилий на его проверку;
- техническое исследование программы для получения информации о ее качестве с точки зрения определенного круга заинтересованных лиц;
- проверка соответствия между реальным поведением программы и ее ожидаемым поведением на конечном наборе тестов, выполненных определенным образом;
- процесс наблюдения за выполнением программы в специальных условиях и вынесения на этой основе оценки каких-либо аспектов ее работы;
- процесс, имеющий целью выявление ситуаций, в которых поведение программы является неправильным, нежелательным или не соответствующим спецификации;
- процесс, содержащий в себе все активности жизненного цикла, как динамические, так и статические, касающиеся планирования, подготовки и оценки программного продукта и связанных с этим результатов работ с целью определить, что они соответствуют описанным требованиям, показать, что они подходят для заявленных целей и для определения дефектов.

Первые программные системы разрабатывались в рамках научных исследований или программ для нужд министерств обороны. Тестирование таких продуктов проводилось строго формализовано с записью всех тестовых процедур, тестовых данных, полученных результатов. Тестирование выделялось в отдельный процесс, который начинался после завершения кодирования, но при этом, как правило, выполнялось тем же персоналом.

В 1960-х много внимания уделялось «исчерпывающему» тестированию, которое должно проводиться с использованием всех путей в коде или всех возможных входных данных. Было отмечено, что в этих условиях полное тестирование программного обеспечения невозможно, потому что, во-первых, количество возможных входных данных очень велико, во-вторых, существует множество путей, в-третьих, сложно найти проблемы в архитектуре и спецификациях. По этим причинам «исчерпывающее» тестирование было отклонено и признано теоретически невозможным.

В начале 1970-х годов тестирование программного обеспечения обозначалось как «процесс, направленный на демонстрацию корректности продукта» или как «деятельность по подтверждению правильности работы программного

обеспечения». В зарождавшейся программной инженерии верификация ПО значилась как «доказательство правильности». Хотя концепция была теоретически перспективной, на практике она требовала много времени и была недостаточно всеобъемлющей. Было решено, что доказательство правильности – неэффективный метод тестирования программного обеспечения. Однако, в некоторых случаях демонстрация правильной работы используется и в наши дни, например, приемо-сдаточные испытания. Во второй половине 1970-х тестирование представлялось как выполнение программы с намерением найти ошибки, а не доказать, что она работает. Успешный тест – это тест, который обнаруживает ранее неизвестные проблемы. Данный подход прямо противоположен предыдущему. Указанные два определения представляют собой «парадокс тестирования», в основе которого лежат два противоположных утверждения: с одной стороны, тестирование позволяет убедиться, что продукт работает хорошо, а с другой – выявляет ошибки в программах, показывая, что продукт не работает. Вторая цель тестирования является более продуктивной с точки зрения улучшения качества, так как не позволяет игнорировать недостатки программного обеспечения.

В 1980-е годы тестирование расширилось таким понятием, как предупреждение дефектов. Проектирование тестов – наиболее эффективный из известных методов предупреждения ошибок. В это же время стали высказываться мысли, что необходима методология тестирования, в частности, что тестирование должно включать проверки на всем протяжении цикла разработки, и это должен быть управляемый процесс. В ходе тестирования надо проверить не только собранную программу, но и требования, код, архитектуру, сами тесты. «Традиционное» тестирование, существовавшее до начала 1980-х, относилось только к скомпилированной, готовой системе (сейчас это обычно называется системное тестирование), но в дальнейшем тестировщики стали вовлекаться во все аспекты жизненного цикла разработки. Это позволяло раньше находить проблемы в требованиях и архитектуре и тем самым сокращать сроки и бюджет разработки. В середине 1980-х появились первые инструменты для автоматизированного тестирования. Предполагалось, что компьютер сможет выполнить больше тестов, чем человек, и сделает это более надежно. Поначалу эти инструменты были крайне простыми и не имели возможности написания сценариев на скриптовых языках.

В начале 1990-х годов в понятие «тестирование» стали включать планирование, проектирование, создание, поддержку и выполнение тестов и тестовых окружений, и это означало переход от тестирования к обеспечению качества, охватывающего весь цикл разработки программного обеспечения. В это время начинают появляться различные программные инструменты для поддержки процесса тестирования: более продвинутые среды для автоматизации с возможностью создания скриптов и генерации отчетов, системы управления тестами, ПО для проведения нагрузочного тестирования. В середине 1990-х годов с развитием Интернета и разработкой большого количества веб-приложений особую популярность стало получать «гибкое тестирование» (по аналогии с гибкими методологиями программирования).

Существует несколько признаков, по которым принято производить классификацию видов тестирования. Обычно выделяют следующие:

- а) по объекту тестирования:
 - 1) функциональное тестирование;
 - 2) нагрузочное тестирование;
 - 3) стресс-тестирование;
 - 4) тестирование стабильности;
 - 5) конфигурационное тестирование;
 - 6) юзабилити-тестирование;
 - 7) тестирование безопасности;
 - 8) тестирование локализации;
 - 9) тестирование совместимости;
- б) по знанию внутреннего строения системы:
 - 1) тестирование черного ящика;
 - 2) тестирование белого ящика;
 - 3) тестирование серого ящика;
- в) по степени автоматизации:
 - 1) ручное тестирование;
 - 2) автоматизированное тестирование;
- г) по степени изолированности:
 - 1) тестирование компонентов;
 - 2) интеграционное тестирование;
 - 3) системное тестирование;
- д) по времени проведения тестирования:
 - 1) альфа-тестирование;
 - 2) бета-тестирование;
- е) по признаку позитивности сценариев:
 - 1) позитивное тестирование;
 - 2) негативное тестирование.

4.1 Ручное тестирование программного продукта

При входе в приложение пользователь (вне зависимости от того, является он абитуриентом или нет) попадает на главную страницу и имеет доступ к общему функционалу. Далее он может либо авторизоваться/зарегистрироваться, либо воспользоваться одной из общих функций.

Если пользователь желает продолжить работу как абитуриент, ему необходимо нажать на иконку пользователя в правом верхнем углу главного окна (рисунок 4.1) и перейти на окно входа в систему (рисунок 4.2). В окне входа в систему пользователю необходимо ввести логин (электронную почту) и пароль и нажать кнопку «Войти». Если пользователь желает зарегистрироваться как новый пользователь, необходимо нажать кнопку «Зарегистрироваться».

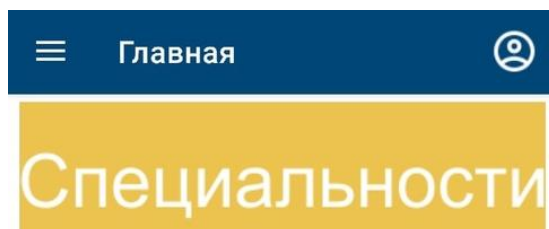


Рисунок 4.1 – Фрагмент окна главной страницы

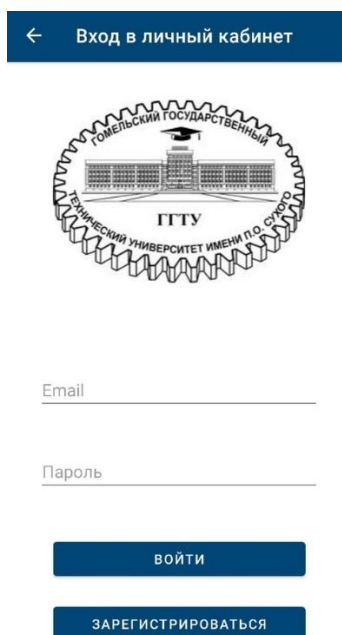


Рисунок 4.2 – Окно входа в систему

Если при входе в систему пользователь ввел несуществующие данные, на экране отобразится сообщение о том, что такой пользователь не найден (рисунок 4.3).

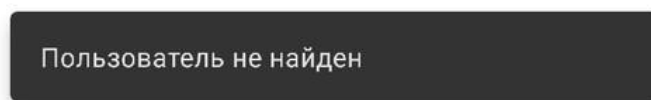
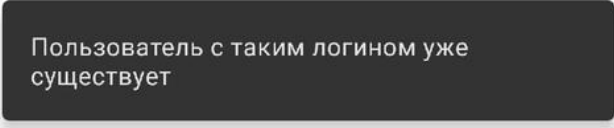


Рисунок 4.3 – Сообщение о вводе данных несуществующего пользователя

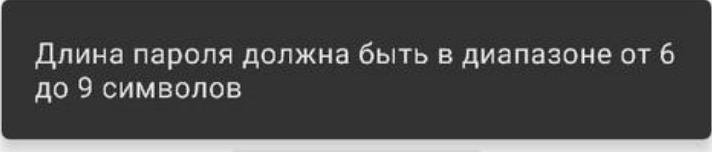
Если при регистрации пользователь ввел уже существующие данные, на экране отобразится сообщение о существовании пользователя с такими данными (рисунок 4.4).



Пользователь с таким логином уже существует

Рисунок 4.4 – Сообщение о вводе уже существующих данных

Если при регистрации пользователь ввел пароль с длиной, не входящей в необходимый интервал, он получит сообщение о вводе неверного пароля (рисунок 4.5).



Длина пароля должна быть в диапазоне от 6 до 9 символов

Рисунок 4.5 – Сообщение о вводе пароля неверной длины

Для проверки введенного пароля был создан специальный класс *Email*, являющийся классом-валидатором для введенного поля (рисунок 4.6).

```
public HashMap<String, Boolean> build() {
    boolean success;
    emailValidationResponse = new HashMap<>();
    if (this.isRequired) {
        emailValidationResponse.put(IS_REQUIRED, true);
    } else {
        emailValidationResponse.put(IS_REQUIRED, false);
    }
    if (this.value != null && !this.value.isEmpty()) {
        emailValidationResponse.put(EMPTY, false);
        Pattern pattern = Patterns.EMAIL_ADDRESS;
        if (pattern.matcher(this.value).matches()) {
            success = true;
            emailValidationResponse.put(IS_EMAIL, true);
        } else {
            success = false;
            emailValidationResponse.put(IS_EMAIL, false);
        }
    } else {
        success = false;
        emailValidationResponse.put(EMPTY, true);
        emailValidationResponse.put(IS_EMAIL, false);
    }
    emailValidationResponse.put(SUCCESS, success);
    return emailValidationResponse;
}
```

Рисунок 4.6 – Проверка *email* в классе-валидаторе

Если пользователь введет в поле для ввода *email* неверные данные, он получит соответствующее уведомление. Пример уведомления представлен на рисунке 4.7.



Рисунок 4.7 – Сообщение о вводе неверного *email*

Если пользователь при регистрации вводит верные данные, он попадает на страницу заполнения информации (рисунок 4.8).

Рисунок 4.8 – Страница заполнения информации

При вводе неверных данных пользователь получает соответствующее уведомление (рисунок 4.9).

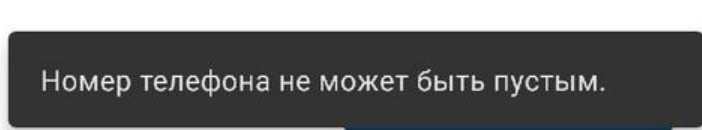


Рисунок 4.9 – Сообщение о вводе неверных данных

Для валидации поля номера телефона, как и для поля *email*, был создан отдельный класс-валидатор: *PhoneNumber* (рисунок 4.10).

```
public HashMap<String, Boolean> build() {
    HashMap<String, Boolean> phoneNumberValidatorResp = new HashMap<>();

    if (this.isRequired) {
        phoneNumberValidatorResp.put(IS_REQUIRED, true);
    } else {
        phoneNumberValidatorResp.put(IS_REQUIRED, false);
    }

    if (this.value != null && !this.value.isEmpty()) {
        phoneNumberValidatorResp.put(EMPTY, false);
        if (this.value.length() >= minLength && this.value.length() <= maxLength) {
            phoneNumberValidatorResp.put(SUCCESS, true);
            phoneNumberValidatorResp.put(MAX_LENGTH, true);
            phoneNumberValidatorResp.put(MIN_LENGTH, true);
        } else {
            phoneNumberValidatorResp.put(SUCCESS, false);
            phoneNumberValidatorResp.put(MAX_LENGTH, false);
            phoneNumberValidatorResp.put(MIN_LENGTH, false);
        }
    } else {
        phoneNumberValidatorResp.put(SUCCESS, false);
        phoneNumberValidatorResp.put(EMPTY, true);
        phoneNumberValidatorResp.put(MAX_LENGTH, false);
        phoneNumberValidatorResp.put(MIN_LENGTH, false);
    }

    return phoneNumberValidatorResp;
}
```

Рисунок 4.10 – Проверка номера телефона в классе-валидаторе

Если пользователь ввел верные данные и нажал на кнопку «Добавить данные», открывается диалоговое окно подтверждения действия (рисунок 4.11).

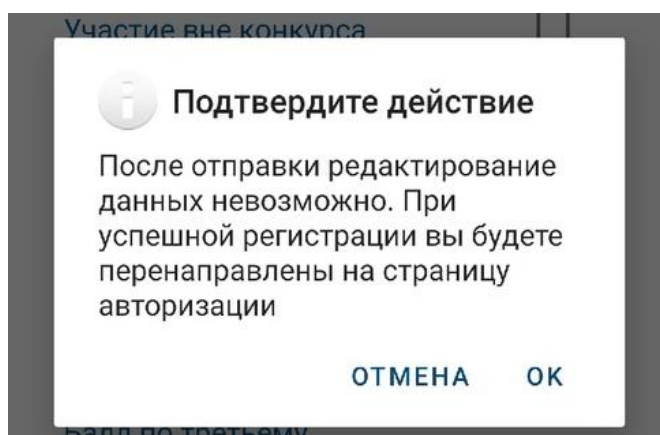


Рисунок 4.11 – Окно подтверждения действия

Подобный набор проверок на корректность введенных данных представлен и в остальных окнах приложения.

В таблице 4.1 приведена информация об основных окнах приложения и их функционале.

Таблица 4.1 – Функционал основных окон приложения

Название окна	Выполняемый окном функционал
CreateAccount	Заполнение данных абитуриента и создание записи в базе данных
ViewAccount	Загрузка данных абитуриента из базы данных в приложение и отображение в окне просмотра
Map	Отображение карты с меткой «университет» и информацией о вариантах проезда
SendEmail	Отправка письма разработчику приложения
Calculation	Ввод абитуриентом баллов вступительных испытаний и подсчет общего проходного балла
Calendar	Просмотр важных новостей университета с возможностью перехода по ссылке интересующей новости
Contacts	Отображение основной информации об университете: контакты, адрес, электронная почта, сайт
Login	Предоставление набора функционала программного обеспечения в зависимости от роли пользователя, обеспечение безопасности
MainPage	Отображение главной страницы, оформленной в виде плиточного меню с возможностью перехода по каждому пункту на соответствующий сайт, открывающийся внутри приложения
Main	Главное окно, стартовая страница приложения
Web	Отображение сайта внутри приложения

4.2 Автоматизированное тестирование программного продукта

Для автоматизированного тестирования выберем тестирование *Python*-скрипта.

Тестирование проводилось при помощи модуля *pytest*.

Pytest – это основанная на *Python* среда тестирования, которая используется для написания и выполнения тестовых кодов. В настоящее время службы *REST pytest* в основном используются для тестирования *API*, однако возможно использовать *pytest* и для написания простых и сложных тестов, то есть можно воспользоваться *pytest* для тестирования *API*, базы данных, пользовательского интерфейса и т. д.

Рассмотрим основные преимущества *pytest*:

- *pytest* может выполнять несколько тестов параллельно, что сокращает время выполнения набора тестов;

- у *pytest* есть собственный способ автоматического определения тестового файла и тестовых функций, если это не указано явно;
 - *pytest* позволяет нам пропустить подмножество тестов во время выполнения;
 - *pytest* позволяет нам запускать подмножество всего набора тестов;
 - *pytest* является бесплатным и открытым исходным кодом;
 - благодаря простому синтаксису, *pytest* очень прост для запуска.
- Для установки *pytest* необходимо выполнить команду, представленную на рисунке 4.12.

```
pip install pytest
```

Рисунок 4.12 – Команда для установки *pytest*

Для написания тестов был создан отдельный файл – *test_main.py*. В нем определена функция *test_scrape_events* (рисунок 4.13).

```
def test_scrape_events(mocked, monkeypatch):
    mocked.patch('requests.get', MockRequest)
    monkeypatch.setattr(BeautifulSoup, '__init__', result_value=None)

    assert 'test_date' == scrape_events()[0]['event_date']
```

Рисунок 4.13 – Функция *test_scrape_events*

Чтобы запустить тест, необходимо в терминале прописать команду «*pytest*» (рисунок 4.14).

```
(venv) C:\Users\masha\CalendarScrapperService>pytest
```

Рисунок 4.14 – Команда для выполнения тестов

При успешном выполнении теста на экран выведется соответствующая информация (рисунок 4.15):

```
(venv) C:\Users\masha\CalendarScrapperService>pytest
===== test session starts =====
platform win32 -- Python 3.8.7, pytest-6.2.4, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\masha\CalendarScrapperService
plugins: cov-2.12.0, mock-3.6.1
collected 1 item

test_main.py
PASSED
===== 1 passed in 0.03 seconds =====
```

Рисунок 4.15 – Уведомление о том, что тесты пройдены

Рассмотрим подробнее параметры функции *test_scrape_events*. Их два: *mock* и *monkeypatch*. Данные параметры необходимы для осуществления возможности подмены методов и значений атрибутов классов программы во время ее выполнения. Такая технология используется во многих динамических языках программирования. В данном случае эта подмена необходима для создания функции-заглушки.

Для осуществления подмены необходимо было создать классы, которые будут использоваться в качестве заглушек. В них прописаны необходимые заменяемые функции и возвращаемые значения (рисунок 4.16):

```
class MockRequest:
    def __init__(self, url):
        self.encoding = ""
        self.text = "Test text"

class MockSoupResult:
    def __init__(self, id, attrs):
        self.value = {'data-date': 'test_date'}

    def __getitem__(self, item):
        if isinstance(item, str):
            return self.value[item]
        else:
            return 0

    @staticmethod
    def find(name, href):
        return MockDescription

class MockDescription:
    def __init__(self):
        self.value = {'href': 'test_href'}

    def __getitem__(self, item):
        return self.value[item]

    @staticmethod
    def text():
        return 'test_text'
```

Рисунок 4.16 – Классы-заглушки

В результате проведения ручного и автоматизированного тестирования программного продукта можно сделать вывод о том, что весь необходимый функционал был выполнен в полном объеме, ручное и автоматизированное тестирование пройдено успешно.

5 ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ ДИПЛОМНОЙ РАБОТЫ

5.1 Расчет общей трудоемкости разработки программного обеспечения

Общий объем трудоемкости разработки системы по автоматизации учета лабораторных испытаний на промышленном предприятии (V_o) определяется исходя из количества и объема функций, реализуемых программой, по каталогу функций ПО в соответствии с таблицей 1.1 приложения 1 источника [7] по формуле (5.1):

$$V_o = \sum_{i=1}^n V_i, \quad (5.1)$$

где V_i – объем отдельной функции ПО;
 n – общее число функций.

$$V_o = 130 + 490 + 1970 + 7860 + 4720 + 5240 + 1540 + 1680 + 470 \\ = 24100$$

Анализируя разработанную программу, уточненный объем ПО (V_y) определяем по формуле (5.2):

$$V_y = \sum_{i=1}^n V_{yi}, \quad (5.2)$$

где V_{yi} – уточненный объем отдельной функции ПО в строках исходного кода (LOC).

$$V_y = 70 + 190 + 1830 + 195 + 350 + 35 + 80 + 50 + 300 = 3100$$

Сравнение исходного и уточненных объемов строк исходного кода представлены в таблице Б.1 приложения Б.

Разработанное в ходе выполнения дипломной работы приложение относится к третьей категории сложности.

На основании принятого к расчету (уточненного) объема (V_y) и категории сложности ПО определяется нормативная трудоемкость ПО (T_n) выполняемых работ, которая приведена в таблице 5.1.

Таблица 5.1 – Нормативная трудоемкость на разработку ПО (T_n)

Уточненный объем, V_y	3-я категория сложности ПО	Номер нормы
3100	134	39

Дополнительные затраты труда, связанные с повышением сложности разрабатываемого ПО, учитываются посредством коэффициента повышения сложности ПО (K_c). K_c рассчитывается по формуле (5.3):

$$K_c = 1 + \sum_{i=1}^n K_i, \quad (5.3)$$

где K_i – коэффициент, соответствующий степени повышения сложности;

n – количество учитываемых характеристик.

Таким образом:

$$K_c = 1 + 0,08 + 0,06 = 1,14.$$

Новизна разработанного ПО определяется путем экспертной оценки данных, полученных при сравнении характеристик разрабатываемого ПО с имеющимися аналогами. Влияние фактора новизны на трудоемкость учитывается путем умножения нормативной трудоемкости на соответствующий коэффициент, учитывающий новизну ПО (K_n). Разработанная программа обладает категорией новизны В, а значение $K_n = 0,63$.

Современные технологии разработки компьютерных программ предусматривают широкое использование коробочных продуктов (пакетов, модулей, объектов). Степень использования в разрабатываемом ПО стандартных модулей определяется их удельным весом в общем объеме ПО.

В данном программном комплексе используется до 20% стандартных модулей, что соответствует значению коэффициента $K_t = 0,9$.

Приложение разработано на языке *Java*, что соответствует коэффициенту функционирования в локальных сетях, учитывающему средства разработки ПО, $K_{yp} = 1,2$.

Значения коэффициентов удельных весов трудоемкости стадий разработки ПО в общей трудоемкости ПО определяются с учетом установленной категории новизны ПО согласно таблице 2.5 [7].

При этом сумма значений коэффициентов удельных весов всех стадий в общей трудоемкости равна единице. Значения коэффициентов приведены в таблице 5.2.

Таблица 5.2 – Значения коэффициентов удельных весов трудоемкости стадий разработки ПО в общей трудоемкости ПО

Категория новизны ПО	Без применения CASE-технологии				
	Стадии разработки ПО				
	ТЗ	ЭП	ТП	РП	ВН
	Значения коэффициентов				
	$K_{ТЗ}$	$K_{ЭП}$	$K_{ТП}$	$K_{РП}$	$K_{ВН}$
В	0,08	0,19	0,28	0,34	0,11

Нормативная трудоемкость ПО (T_H) выполняемых работ по стадиям разработки корректируется с учетом коэффициентов: повышения сложности ПО (T_C), учитывающих новизну ПО (K_H), учитывающих степень использования стандартных модулей (K_T), средства разработки ПО ($K_{ур}$) и определяются по формулам:

– для стадии ТЗ по формуле (5.4):

$$T_{y.тз} = T_H \cdot K_{тз} \cdot K_C \cdot K_H \cdot K_{ур}, \quad (5.4)$$

– для стадии ЭП по формуле (5.5):

$$T_{y.эп} = T_H \cdot K_{эп} \cdot K_C \cdot K_H \cdot K_{ур}, \quad (5.5)$$

– для стадии ТП по формуле (5.6):

$$T_{y.тп} = T_H \cdot K_{тп} \cdot K_C \cdot K_H \cdot K_{ур}, \quad (5.6)$$

– для стадии РП по формуле (5.7):

$$T_{y.рп} = T_H \cdot K_{рп} \cdot K_C \cdot K_H \cdot K_T \cdot K_{ур}, \quad (5.7)$$

– для стадии ВН по формуле (5.8):

$$T_{y.вн} = T_H \cdot K_{вн} \cdot K_C \cdot K_H \cdot K_{ур}, \quad (5.8)$$

Коэффициенты K_C , K_H , $K_{ур}$ вводятся на всех стадиях разработки, а коэффициент K_T вводится только на стадии РП.

Таким образом:

$$T_{тз} = 134 \cdot 1,14 \cdot 0,63 \cdot 1,2 \cdot 0,08 \approx 9$$

$$T_{эп} = 134 \cdot 1,14 \cdot 0,63 \cdot 1,2 \cdot 0,19 \approx 22$$

$$T_{тп} = 134 \cdot 1,14 \cdot 0,63 \cdot 1,2 \cdot 0,28 \approx 32$$

$$T_{рп} = 134 \cdot 1,14 \cdot 0,63 \cdot 1,2 \cdot 0,9 \cdot 0,34 \approx 35$$

$$T_{вн} = 134 \cdot 1,14 \cdot 0,63 \cdot 1,2 \cdot 0,11 \approx 13$$

Общая трудоемкость разработки ПО (T_0) определяется суммированием нормативной (скорректированной) трудоемкости ПО по стадиям разработки по формуле (5.9):

$$T_0 = \sum_{i=1}^n T_{yi}, \quad (5.9)$$

где T_{yi} – нормативная (скорректированная) трудоемкость разработки ПО на i -й стадии (чел/дней);

n – количество стадий разработки.

Таким образом:

$$T_o = 9 + 22 + 32 + 35 + 13 = 111 \text{ чел/дней.}$$

Результаты расчетов по определению нормативной и скорректированной трудоемкости ПО по стадиям разработки и общую трудоемкость разработки ПО (T_o) представлены в таблице В.1 приложения В.

5.2 Расчет затрат на разработку программного продукта

В состав затрат на разработку системы по автоматизации учета лабораторных испытаний промышленного предприятия входят следующие статьи расходов:

- затраты труда на создание программного продукта (затраты по основной, дополнительной заработной плате и соответствующие отчисления) ($Z_{тр}$);
- затраты на изготовление эталонного экземпляра ($Z_{эт}$);
- затраты на технологию (затраты на приобретение и освоение программных средств, используемых при разработке программного продукта; затраты на ПО, используемое как эталон) ($Z_{тех}$);
- затраты на машинное время (расходы на содержание и эксплуатацию технических средств разработки, эксплуатации и сопровождения) ($Z_{мв}$);
- затраты на материалы (информационные носители) ($Z_{мат}$);
- затраты на энергию, на использование каналов связи (для отдельных видов);
- общепроизводственные расходы (затраты на управленческий персонал, на содержание помещений) ($Z_{общ.пр}$);
- непроизводственные (коммерческие) расходы (затраты, связанные с рекламой, поиском заказчиков, поставками конкретных экземпляров) ($Z_{непр}$).

В таблице Г.1 приложения Г приведены значения основных параметров, необходимых для расчета затрат на разработку программного продукта.

Суммарные затраты на разработку ПО (Z_p) определяются по формуле (5.10):

$$Z_p = Z_{тр} + Z_{эт} + Z_{тех} + Z_{мв} + Z_{мт} + Z_{общ.пр} + Z_{непр} \quad (5.10)$$

Расходы на оплату труда разработчиков с отчислениями определяются по формуле (5.11):

$$Z_{тр} = Z_{П_{осн}} + Z_{П_{доп}} + ОТЧ_{зп}, \quad (5.11)$$

где $ЗП_{осн}$ – основная заработная плата разработчиков, руб.;

$ЗП_{доп}$ – дополнительная заработная плата разработчиков, руб.;

$ОТЧ_{зп}$ – сумма отчислений от заработной платы (социальные нужды, страхование от несчастных случаев), руб.

Основная заработная плата разработчиков рассчитывается по формуле (5.12):

$$ЗП_{осн} = C_{ср\,час} \cdot T_o \cdot K_{ув}, \quad (5.12)$$

где $C_{ср\,час}$ – средняя часовая тарифная ставка;

T_o – общая трудоемкость разработки, чел-час;

$K_{ув}$ – коэффициент, учитывающий доплаты стимулирующего характера.

Средняя часовая тарифная ставка определяется по формуле (5.13):

$$C_{ср\,час} = \frac{\sum_i C_{чи} \cdot n_i}{\sum_i n_i}, \quad (5.13)$$

где $C_{чи}$ – часовая тарифная ставка разработчика i – й категории;

n_i – количество разработчиков i -й категории.

Часовая тарифная ставка разработчика i -й категории определяется по формуле (5.14):

$$C_{ч} = T_{ст} \cdot k, \quad (5.14)$$

где $T_{ст}$ – базовая ставка;

k – тарифный коэффициент.

Таким образом:

$$C_{ср\,час} = C_{ч} = \frac{195 \cdot 2,5}{168} = 2,9 \text{ руб.}$$

$$ЗП_{осн} = 2,9 \cdot 111 \cdot 8 \cdot 1,5 = 3862,8 \text{ руб.}$$

Дополнительная заработная плата определяется по формуле (5.15):

$$ЗП_{доп} = ЗП_{осн} \cdot \frac{Н_{доп}}{100\%}, \quad (5.15)$$

где $Н_{доп}$ – норматив отчислений на дополнительную заработную плату разработчиков.

Таким образом:

$$ЗП_{доп}^{мес} = 3862,8 \cdot 0,15 = 579,42 \text{ руб.}$$

$$ЗП^{мес} = 3862,8 + 579,42 = 4442,22 \text{ руб.}$$

Отчисления от основной и дополнительной заработной платы (отчисления на социальные нужды и обязательное страхование) рассчитываются по формуле (5.16):

$$\text{ОТЧ}_{\text{CH}} = (\text{ЗП}_{\text{осн}} + \text{ЗП}_{\text{доп}}) \cdot \frac{H_{\text{зп}}}{100\%}, \quad (5.16)$$

где $H_{\text{зп}}$ – процент отчислений на социальные нужды и обязательное страхование от суммы основной и дополнительной заработной платы ($H_{\text{зп}} = 34,6 \%$).

$$\text{ОТЧ}_{\text{CH}} = 4449,22 \cdot 0,346 = 1539,43 \text{ руб.}$$

$$\text{З}_{\text{тр}} = 3862,8 + 579,42 + 1539,43 = 5981,65 \text{ руб.}$$

Затраты машинного времени определяются по формуле (5.17):

$$\text{З}_{\text{МВ}} = C_{\text{ч}} \cdot K_{\text{Т}} \cdot t_{\text{ЭВМ}}, \quad (5.17)$$

где $C_{\text{ч}}$ – стоимость 1 часа машинного времени (руб./ч.);

$K_{\text{Т}}$ – коэффициент мультипрограммности, показывающий распределение времени работы ЭВМ в зависимости от количества пользователей ЭВМ;

$K_{\text{Т}}=1$;

$t_{\text{ЭВМ}}$ – машинное время ЭВМ, необходимое для разработки и отладки проекта (ч.).

Стоимость машино-часа определяется по формуле (5.18):

$$C_{\text{ч}} = \frac{\text{ЗП}_{\text{обсл}} + \text{З}_{\text{АР}} + \text{З}_{\text{АМ}} + \text{З}_{\text{ЭП}} + \text{З}_{\text{ВМ}} + \text{З}_{\text{ТР}} + \text{З}_{\text{ПР}}}{F_{\text{ЭВМ}}}, \quad (5.18)$$

где $\text{ЗП}_{\text{обсл}}$ – затраты на заработную плату обслуживающего персонала с учетом всех отчислений, (руб. в год);

$\text{З}_{\text{АР}}$ – стоимость аренды помещения под размещение вычислительной техники, (руб. в год);

$\text{З}_{\text{АМ}}$ – амортизационные отчисления за год, (руб. в год);

$\text{З}_{\text{ЭП}}$ – затраты на электроэнергию, (руб. в год);

$\text{З}_{\text{ВМ}}$ – затраты на материалы, необходимые для обеспечения нормальной работы ПЭВМ (вспомогательные), (руб. в год);

$\text{З}_{\text{ТР}}$ – затраты на текущий и профилактический ремонт ЭВМ (руб. в год);

$\text{З}_{\text{ПР}}$ – прочие затраты, связанные с эксплуатацией ПЭВМ (руб. в год);

$F_{\text{ЭВМ}}$ – действительный фонд времени работы ЭВМ (час/год).

Все статьи затрат формируются в расчете на единицу ПЭВМ.

Затраты на заработную плату обслуживающего персонала ($\text{ЗП}_{\text{обсл}}$) определяются по формуле (5.19):

$$ЗП_{\text{обсл}} = \frac{ЗП_{\text{осн}} + ЗП_{\text{доп}} + ОТЧ_{\text{зп}}}{Q_{\text{ЭВМ}}}, \quad (5.19)$$

$$ЗП_{\text{осн}} = 12 \sum_{i=1}^n T_{ci},$$

$$ЗП_{\text{доп}} = ЗП_{\text{осн}} \cdot \frac{Н_{\text{доп}}}{100\%},$$

$$ОТЧ_{\text{зп}} = (ЗП_{\text{осн}} + ЗП_{\text{доп}}) \cdot \frac{Н_{\text{зп}}}{100\%},$$

где $ЗП_{\text{осн}}$ – основная заработная плата обслуживающего персонала, руб.;

$ЗП_{\text{доп}}$ – дополнительная заработная плата обслуживающего персонала, руб.;

$ОТЧ_{\text{зп}}$ – сумма отчислений от заработной платы (социальные нужды, страхование от несчастных случаев), руб.;

$Q_{\text{ЭВМ}}$ – количество обслуживаемых ПЭВМ, шт.;

T_{ci} – месячная тарифная ставка i -го работника, руб.;

n – численность обслуживающего персонала, чел.;

$Н_{\text{доп}}$ – процент дополнительной заработной платы обслуживающего персонала от основной;

$Н_{\text{зп}}$ – процент отчислений на социальные нужды и обязательное страхование от суммы основной и дополнительной заработной платы.

Тарифная ставка 5-го разряда обслуживающего персонала:

$$T_{c5} = 195 \cdot 1,29 = 251,55 \text{ руб.}$$

$$ЗП_{\text{осн}} = 12 \cdot 251,55 = 3018,6 \text{ руб.}$$

$$ЗП_{\text{доп}} = 452,79 \text{ руб.}$$

$$ЗП = ЗП_{\text{осн}} + ЗП_{\text{доп}} = 3018,6 + 452,79 = 3471,39 \text{ руб.}$$

$$ОТЧ_{\text{зп}} = 3471,39 \cdot 0,346 = 1201,1 \text{ руб.}$$

$$ЗП_{\text{обсл}} = 3471,39 + 1201,1 = 4672,49 \text{ руб.}$$

Годовые затраты на аренду помещения ($З_{\text{АР}}$) определяются по формуле (5.20):

$$З_{\text{АР}} = \frac{C_{\text{АР}} \cdot S}{Q_{\text{ЭВМ}}}, \quad (5.20)$$

где $C_{\text{АР}}$ – средняя годовая ставка арендных платежей, руб./м²;

S – площадь помещения, м²;

$Q_{ЭВМ}$ – количество ПЭВМ, шт.

$$З_{АР} = 202,8 \cdot 10 = 2028 \text{ руб.}$$

Сумма годовых амортизационных отчислений ($З_{АМ}$) определяется по формуле (5.21):

$$З_{АМ} = З_{\text{приобр}} \cdot (1 + K_{\text{доп}}) \cdot Н_{АМ}, \quad (5.21)$$

где $З_{\text{приобр}}$ – затраты на приобретение (стоимость) единицы ПЭВМ, руб.;

$K_{\text{доп}}$ – коэффициент, характеризующий дополнительные затраты, связанные с доставкой, монтажом и наладкой оборудования, $K_{\text{доп}} = 12 - 13\%$ от $З_{\text{приобр}}$;

$З_{\text{приобр}} \cdot (1 + K_{\text{доп}})$ – балансовая стоимость ЭВМ, руб.;

$Н_{АМ}$ – норма амортизации, %.

$$З_{АМ} = 2500 \cdot (1 + 0,12) \cdot 0,125 = 350 \text{ руб.}$$

Стоимость электроэнергии, потребляемой за год, ($З_{ЭП}$) определяется по формуле (5.22):

$$З_{ЭП} = M \cdot F_{ЭВМ} \cdot C_{эл} \cdot A, \quad (5.22)$$

где M – паспортная мощность ПЭВМ, (кВт), $M = 1$ кВт;

$C_{эл}$ – стоимость одного кВт-часа электроэнергии, руб.;

$F_{ЭВМ}$ – действительный годовой фонд времени работы ПЭВМ, $F_{ЭВМ} = 2056$ ч., согласно производственному календарю на 2021 год.

$$З_{ЭП} = 1 \cdot 2056 \cdot 0,2867 \cdot 0,9 = 530 \text{ руб.}$$

Затраты на материалы ($З_{ВМ}$), необходимые для обеспечения нормальной работы ПЭВМ составляют около 1% от балансовой стоимости ЭВМ и определяются формулой (5.23):

$$З_{ВМ} = З_{\text{приобр}} \cdot (1 + K_{\text{доп}}) \cdot K_{МЗ}, \quad (5.23)$$

где $З_{\text{приобр}}$ – затраты на приобретение (стоимость) ЭВМ, руб.;

$K_{\text{доп}}$ – коэффициент, характеризующий дополнительные затраты, связанные с доставкой, монтажом и наладкой оборудования, $K_{\text{доп}} = 12 - 13\%$ от $З_{\text{приобр}}$;

$K_{МЗ}$ – коэффициент, характеризующий затраты на вспомогательные материалы ($K_{МЗ} = 0,01$).

$$З_{ВМ} = 2500 \cdot (1 + 0,12) \cdot 0,01 = 28 \text{ руб.}$$

Затраты на текущий и профилактический ремонт ($Z_{тр}$) принимаются равными 5% от балансовой стоимости ЭВМ и рассчитываются по формуле (5.24):

$$Z_{тр} = Z_{приобр} \cdot (1 + K_{доп}) \cdot K_{тр}, \quad (5.24)$$

где $K_{тр}$ – коэффициент, характеризующий затраты на текущий и профилактический ремонт ($K_{мз} = 0,05$).

$$Z_{тр} = 2500 \cdot (1 + 0,12) \cdot 0,05 = 140 \text{ руб.}$$

Прочие затраты, связанные с эксплуатацией ЭВМ ($Z_{пр}$), состоят из амортизационных отчислений на здания, стоимости услуг сторонних организаций, составляют 5 % от балансовой стоимости и рассчитываются по формуле (5.25):

$$Z_{пр} = Z_{приобр} \cdot (1 + K_{доп}) \cdot K_{пр}, \quad (5.25)$$

где $K_{пр}$ – коэффициент, характеризующий размет прочих затрат, связанных с эксплуатацией ЭВМ ($K_{пр} = 0,05$).

$$Z_{пр} = 2500 \cdot (1 + 0,12) \cdot 0,05 = 140 \text{ руб.}$$

Для расчета машинного времени ЭВМ ($t_{эвм}$ в часах), необходимого для разработки и отладки проекта, следует использовать формулу (5.26):

$$t_{эвм} = (t_{рп} + t_{вн}) \cdot F_{см} \cdot K_{см}, \quad (5.26)$$

где $t_{рп}$ – срок реализации стадии «Рабочий проект» (РП), 8 дней;
 $t_{вн}$ – срок реализации стадии «Ввод в действие» (ВП), 2 дня;
 $F_{см}$ – продолжительность рабочей смены, (ч.), $F_{см} = 8$ ч.;
 $K_{см}$ – количество рабочих смен, $K_{см} = 1$.

$$t_{эвм} = (8 + 2) \cdot 8 \cdot 1 = 80 \text{ ч;}$$

$$\begin{aligned} Cч &= \frac{4672,49 + 2028 + 350 + 530 + 28 + 140 + 140}{2056} = \\ &= \frac{7888}{2056} = 3,84 \text{ руб/ч;} \end{aligned}$$

$$Z_{мв} = 3,84 \cdot 1 \cdot 80 = 307,2 \text{ руб.}$$

При написании дипломной работы были использованы среда разработки *Android Studio* и облачный сервис *Microsoft Azure SQL Databases*, поэтому

затраты на технологию ($Z_{\text{тех}}$) и изготовление эталонного экземпляра ($Z_{\text{эт}}$) будут нулевыми.

Затраты на материалы (носители информации и пр.), необходимые для обеспечения нормальной работы ПЭВМ рассчитываются по формуле (5.27):

$$Z_{\text{мт}} = Z_{\text{приобр}} \cdot (1 + K_{\text{доп}}) \cdot K_{\text{мз}}, \quad (5.27)$$

где $Z_{\text{приобр}}$ – затраты на приобретение ЭВМ, руб.;

$K_{\text{доп}}$ – коэффициент, характеризующий дополнительные затраты, связанные с доставкой, монтажом и наладкой оборудования, $K_{\text{доп}} = 12 - 13 \%$ от $Z_{\text{приобр}}$;

$K_{\text{мз}}$ – коэффициент, характеризующий затраты материалы ($K_{\text{мз}} = 0,01$).

Таким образом:

$$Z_{\text{мт}} = 2500 \cdot (1 + 0,12) \cdot 0,01 = 28 \text{ руб.}$$

Общепроизводственные затраты рассчитываются по формуле (5.28):

$$Z_{\text{общ пр}} = Z_{\text{Посн}} \cdot \frac{N_{\text{доп}}}{100\%}, \quad (5.28)$$

где $N_{\text{доп}}$ – норматив общепроизводственных затрат.

$$Z_{\text{общ пр}} = 3018,6 \cdot 0,05 = 150,93 \text{ руб.}$$

Непроизводственные затраты рассчитываются по формуле (5.29):

$$Z_{\text{непр}} = Z_{\text{Посн}} \cdot \frac{N_{\text{непр}}}{100\%}, \quad (5.29)$$

где $N_{\text{непр}}$ – норматив непроизводственных затрат.

$$Z_{\text{непр}} = 3018,6 \cdot 0,05 = 150,93 \text{ руб.}$$

Итого получаем суммарные затраты на разработку:

$$Z_p = 5981,65 + 0 + 0 + 307,2 + 28 + 150,93 + 150,93 = 6618,71 \text{ руб.}$$

Результаты расчетов приведены в таблице Д.1 приложения Д.

5.3 Формирование цены при создании программного обеспечения

Оптовая цена ПО ($C_{\text{опт}}$) определяется формулой (5.30):

$$C_{\text{опт}} = Z_p + P_p, \quad (5.30)$$

$$П_p = \frac{З_p \cdot У_p}{100},$$

где $З_p$ – себестоимость ПО, руб.;

$П_p$ – прибыль от реализации ПО, руб.;

$У_p$ – уровень рентабельности ПО, % ($У_p = 30\%$).

$$П_p = \frac{6618,71 \cdot 30}{100} = 1985,61 \text{ руб.}$$

$$Ц_{\text{опт}} = 6618,71 + 1985,61 = 8604,32 \text{ руб.}$$

Прогнозируемая отпускная цена ПО с НДС рассчитывается по формуле (5.31):

$$Ц_{\text{отп}} = З_p + П_p + Р_{\text{ндс}}, \quad (5.31)$$

Налог на добавленную стоимость ($Р_{\text{ндс}}$) рассчитывается по формуле (5.32):

$$Р_{\text{ндс}} = (З_p + П_p) \cdot \frac{Н_{\text{ндс}}}{100}, \quad (5.32)$$

где $Н_{\text{ндс}}$ – ставка налога на добавленную стоимость, %, $Н_{\text{ндс}} = 20\%$.

$$Р_{\text{ндс}} = (6618,71 + 1985,61) \cdot 0,2 = 1720,86 \text{ руб.}$$

$$Ц_{\text{отп}} = 6618,71 + 1985,61 + 1720,86 = 10325,18 \text{ руб.}$$

Розничную цену на программный продукт ($Ц_{\text{розн}}$) можно определить по формуле (5.33):

$$Ц_{\text{розн}} = Ц_{\text{отп}} + Т_n \quad (5.33)$$

где $Т_n$ – торговая наценка при реализации программного обеспечения через специализированные магазины (торговых посредников), ее значение принимается равным 10%.

$$Ц_{\text{розн}} = 10325,18 \cdot 1,1 = 11357,7$$

Результаты расчетов формирования цены на разработку программы приведены в таблице 5.3.

Таблица 5.3 – Расчет формирования цены на разработку программы

Наименование статьи расходов	Значение
Суммарные затраты на разработку ПО	6618,71
Полная себестоимость	6618,71
Прибыль от реализации ПО	1985,61
Отпускная цена ПО без НДС	8604,32
Налог на добавленную стоимость	1720,86
Отпускная цена ПО с НДС	10325,18
Розничная цена	11357,7

5.4 Расчет эффекта от внедрения программного обеспечения

Для того, чтобы рассчитать годовой экономический эффект от использования нового ПО необходимо такие параметры как заработная плата специалиста, работающего с программой, стоимость 1 часа работы этого специалиста и время, сэкономленное при использовании программы. Все параметры и расчеты приведены в таблице 5.4.

Эффект (прибыль) может просчитываться по формуле (5.34):

$$\mathcal{E} = \mathcal{Z}_{\text{баз}} - \mathcal{Z}, \quad (5.34)$$

где $\mathcal{Z}_{\text{баз}}$ – текущие и инвестиционные затраты по базовому варианту, включающие затраты на приобретение продукта (цену), его эксплуатацию;

\mathcal{Z} – текущие и инвестиционные затраты по варианту, предложенному студентом-дипломником.

По результатам изучения рыночных цен программных продуктов схожего функционального назначения было установлено, что средняя стоимость аналога составляет 24575 руб.

Таким образом, эффект:

$$\mathcal{E} = 24575 - 11357,7 = 13217,3 \text{ руб.}$$

На основе рассчитанного эффекта от разработки программного обеспечения следует рассчитать следующие итоговые показатели, характеризующие экономическую эффективность проекта:

– рентабельность затрат (\mathcal{P}) или инвестиций (\mathcal{I}) на новую информационную технологию, программный продукт рассчитывается по формуле (5.35):

$$\mathcal{P} = \frac{\mathcal{E}(\Pi)}{\mathcal{Z}(\mathcal{I})} \cdot 100\% \quad (5.35)$$

Таким образом, рентабельность:

$$P = \frac{13217,3}{11357,7} \cdot 100\% = 116,4\%$$

– срок окупаемости затрат (инвестиций) рассчитывается по формуле (5.36):

$$T = \frac{Z(I)}{Э(П)} \quad (5.36)$$

Таким образом, срок окупаемости затрат:

$$T = \frac{11357,7}{13217,7} = 0,86 \text{ года.}$$

Т.к. срок окупаемости составляет меньше одного календарного года, то проведение динамической оценки (расчет динамических показателей эффективности) не целесообразно [7].

Годовой экономический эффект определяется по формуле (5.37):

$$ГЭЭ = Э(П) - P_{\text{баз}} \cdot Z(I), \quad (5.37)$$

где $P_{\text{баз}}$ – рентабельность затрат (инвестиций) базового варианта, 25%.

Таким образом, годовой экономический эффект:

$$ГЭЭ = 13217,3 - 0,25 \cdot 11357,7 = 10377,88$$

На основании выполненных расчетов была сформирована таблица технико-экономических показателей проекта (таблица Е.1 приложения Е). После оценки технико-экономических показателей проектного программного обеспечения можно сделать вывод о том, что реализация проекта является обоснованной и экономически целесообразной, так как срок окупаемости проекта меньше года при размере годового экономического эффекта 10377,88 руб. с уровнем рентабельности 116,4%.

6 ОХРАНА ТРУДА И ТЕХНИКА БЕЗОПАСНОСТИ

Согласно Закону Республики Беларусь «Об охране труда», охрана труда – система обеспечения безопасности жизни и здоровья работающих в процессе трудовой деятельности, включающая правовые, социально-экономические, организационные, технические, психофизиологические, санитарно-гигиенические, лечебно-профилактические, реабилитационные и иные мероприятия и средства, где работающие – граждане Республики Беларусь, иностранные граждане и лица без гражданства, работающие по трудовым договорам и (или) гражданско-правовым договорам, на основе членства (участия) в юридических лицах любых организационно-правовых форм, а также привлекаемые к выполнению работ (оказанию услуг) юридическими лицами в порядке и на условиях, установленных законодательством [8].

6.1 Явление статического электричества

Существование человека в любой среде связано с воздействием на него и среду обитания электромагнитных полей. В случае неподвижных электрических зарядов мы имеем дело с электростатическими полями.

Наряду с естественными статическими электрическими полями в условиях техносферы и в быту человек подвергается воздействию искусственных статических электрических полей.

Электрические поля от избыточных зарядов на предметах, одежде, теле человека оказывают большую нагрузку на нервную систему человека, также чувствительна к электростатическим электрическим полям и сердечно-сосудистая система организма.

Статическое электричество – это совокупность явлений, связанных с возникновением, сохранением и релаксацией свободного электрического заряда на поверхности и в объеме диэлектрических и полупроводниковых веществ, материалов, изделий или на изолированных проводниках.

Возникновение зарядов статического электричества происходит при деформации, дроблении веществ, относительном перемещении двух находящихся в контакте тел, слоев жидких и сыпучих материалов, при интенсивном перемешивании, кристаллизации, а также вследствие индукции.

Наиболее чувствительны к электростатическим полям нервная, сердечно-сосудистая, нейрогуморальная и другие системы организма. Это вызывает необходимость гигиенического нормирования предельно допустимой интенсивности электростатического поля.

Электростатическое поле характеризуется напряженностью, определяемой отношением силы, действующей в поле на точечный электрический заряд, к величине этого заряда. Единицей измерения напряженности является вольт на метр. Допустимый уровень напряженности электростатических полей – $60 \frac{\text{кВ}}{\text{м}}$. В случае, если напряженность поля превышает это значение, должны применяться соответствующие средства защиты.

6.2 Электростатические заряды на производстве и их опасность

В некоторых отраслях промышленного производства, связанных с обработкой диэлектрических материалов, нефтеперерабатывающей, текстильной, бумажной, и т.д. наблюдаются явления электризации тел – статическое электричество.

По определению ГОСТ 31613-2012 «Электростатическая искробезопасность. Общие технические требования и методы испытаний» термин «статическое электричество» означает совокупность явлений, связанных с возникновением, сохранением и релаксацией свободного электрического заряда на поверхности и в объеме диэлектриков и полупроводников, изделий на изолированных (в том числе диспергированных (лат. *dispergo* – рассеивать; порошки, эмульсии) в диэлектрической среде) проводниках [9].

Электризация материалов часто препятствует нормальному ходу технологических процессов производства, а также создает дополнительную пожарную опасность вследствие искрообразования при разрядах при наличии в помещениях, резервуарах и ангарах горючих паро- и газовоздушных смесей.

Этот же ГОСТ дает определение понятия электростатической искробезопасности (ЭСИБ) как состояние объекта, при котором исключена возможность взрыва и пожара от статического электричества. Электростатическая искробезопасность должна обеспечиваться путем устранения разрядов статического электричества, способных стать источником зажигания огнеопасных веществ (материалов, смесей, изделий, продукции и т.д.)

В ряде случаев статическая электризация тела человека и затем последующий разряд с человека на землю или заземленное производственное оборудование, а также электрический разряд с незаземленного оборудования через тело человека могут вызвать болевые и нервные ощущения и быть причиной непроизвольного резкого движения, в результате которого человек может получить травму (падения, ушибы и т.д.).

Согласно гипотезе о статической электризации тел при соприкосновении двух разноразрядных веществ из-за неравновесности атомных и молекулярных сил на их поверхности происходит перераспределение электронов (в жидкостях и газах еще и ионов) с образованием двойного электрического слоя с противоположными знаками электрических зарядов. Таким образом, между соприкасающимися телами, особенно при их трении, возникает контактная разность потенциалов, значение которой зависит от ряда факторов – диэлектрических свойств материалов, значения их взаимного давления при соприкосновении, влажности и температуры поверхностей этих тел, климатических условий.

При последующем разделении этих тел каждое из них сохраняет свой электрический заряд, а с увеличением расстояния между ними (при уменьшении электрической емкости системы) за счет совершаемой работы по разделению зарядов, разность потенциалов возрастает и может достигнуть десятков и сотен киловольт.

При одинаковых значениях диэлектрической постоянной ϵ соприкасающихся материалов электростатические заряды не возникают.

При статической электризации во время технологических процессов, сопровождающихся трением, размельчением твердых частиц, пересыпанием сыпучих материалов, переливанием диэлектрических жидкостей (нефтепродуктов и т.п.) на изолированных от земли металлических частях оборудования возникают, относительно земли, напряжения порядка десятков киловольт. Так, например, при движении резиновой ленты транспортера и в устройствах ременной передачи на ленте (ремне) и на роликах транспортера (шкивах) из-за некоторой пробуксовки возникают заряды противоположных знаков и большого значения, а разность потенциалов достигает 45 кВ. Аналогично происходит электризация при сматывании (наматывании) тканей, бумаги, полиэтиленовой пленки и др.

При относительной влажности воздуха 85% и более разрядов статического электричества практически не возникает. В аэрозолях электрические заряды возникают от трения частиц вещества друг о друга и о воздух во время движения.

Применяемое в электроустановках минеральное масло, в процессе его переливания, например, слив трансформаторного масла в бак, также подвергается электризации.

Электрические заряды, образующиеся на частях производственного оборудования и изделиях, могут взаимно нейтрализовываться вследствие некоторой электропроводности влажного воздуха, а также стекать в землю по поверхности оборудования, но в некоторых случаях, когда заряды велики и разность потенциалов также велика, то (при малой влажности воздуха) может произойти быстрый искровой разряд между наэлектризованными частями оборудования или на землю. Энергия такой искры может оказаться достаточной для воспламенения горючей или взрывоопасной смеси. Например, для многих паро- и газозоодушных взрывоопасных смесей требуется небольшая энергия ($0,1 \cdot 10^{-3}$ Втс). Практически при напряжении 3 кВ искровой разряд вызывает воспламенение паро- и газозоодушных взрывоопасных смесей, а при 5 кВ – большей части горючих пылей и волокон.

6.3 Причины возникновения статического электричества

Электростатические заряды возникают на поверхностях некоторых материалов, как жидких, так и твердых, в результате сложного процесса контактной электролизаии.

Электролизаии возникает при трении двух диэлектрических или диэлектрического и проводящего материалов, если последний изолирован. При разделении двух диэлектрических материалов происходит разделение электрических зарядов, причем материал, имеющий большую диэлектрическую проницаемость, заряжается положительно, а меньшую – отрицательно. Чем больше различаются диэлектрические свойства материалов, тем интенсивнее происходит разделение и накопление зарядов. На соприкасающихся материалах с одинаковыми

диэлектрическими свойствами (диэлектрической проницаемостью) зарядов не образуется [10].

Интенсивность образования электрических зарядов определяется различием электрических свойств материалов в материалах электрических свойств, а также силой и скоростью трения. Чем больше сила и скорость трения и больше различие электрических свойств, тем интенсивнее происходит образование электрических зарядов.

Например, электростатические заряды образуются на кузове движущегося в сухую погоду автомобиля, если резина колес обладает хорошими изолирующими свойствами. В результате между кузовом и землей возникает электрическое напряжение, которое может достигнуть 10 кВ (киловольт) и привести к возникновению искры при выходе человека из автомобиля – разряд через человека на землю.

Заряды могут возникнуть при измельчении, пересыпании и пневмотранспортировке твердых материалов, при переливании, перекачивании по трубопроводам, перевозке в цистернах диэлектрических жидкостей (бензина, керосина), при обработке диэлектрических материалов (эбонита, оргстекла), при сматывании тканей, бумаги, пленки (например, полиэтиленовой). При пробуксовывании резиновой ленты транспортера относительно роликов или ремня ременной передачи относительно шкива могут возникнуть электрические заряды с потенциалом до 45 кВ.

Кроме трения, причиной образования статических зарядов является электрическая индукция, в результате которой изолированные от земли тела во внешнем электрическом поле приобретают электрический заряд. Особенно велика индукционная электролизация электропроводящих объектов. Например, на металлических предметах (автомобиль и т.п.), изолированных от земли, в сухую погоду под действием электрического поля высоковольтных линий электропередач или грозových облаков могут образовываться значительные электрические заряды.

На экранах мониторов и телевизоров положительные заряды накапливаются под действием электронного пучка, создаваемого электроннолучевой трубкой.

В радиоэлектронной промышленности статическое электричество образуется при изготовлении, испытании, транспортировке и хранении полупроводниковых приборов и интегральных микросхем, в помещениях вычислительных центров, на участках множительной техники, а также в ряде других процессов, где применяются диэлектрические материалы, являясь побочным нежелательным фактором.

В химической промышленности при производстве пластических материалов и изделий из них также происходит образование электростатических зарядов и полей напряженностью $240-250 \frac{\text{кВ}}{\text{м}}$ [11].

6.4 Опасные и вредные факторы статического электричества

При прикосновении человека к предмету, несущему электрический заряд, происходит разряд последнего через тело человека. Величины возникающих при разрядке токов небольшие и они очень кратковременны. Поэтому электротравм не возникает. Однако разряд, как правило, вызывает рефлекторное движение человека, что в ряде случаев может привести к резкому движению, падению человека с высоты.

Линии электропередачи, электрооборудование, различные электроприборы – все технические системы, генерирующие, передающие и использующие электромагнитную энергию, создают в окружающей среде электромагнитные поля (переменные электрические и неразрывно связанные с ними переменные магнитные поля).

Действие на организм человека электромагнитных полей определяется частотой излучения, его интенсивностью, продолжительностью и характером действия, индивидуальными особенностями организма. Спектр электромагнитных полей включает низкие частоты до 3 Гц, промышленные частоты – от 3 Гц до 300 Гц, радиочастоты – от 30 Гц до 300 МГц, а также относящиеся к радиочастотам ультравысокие частоты (УВЧ) – от 300 МГц до 300 ГГц.

Электромагнитное излучение радиочастот широко используется в связи, телерадиовещании, в медицине, радиолокации, радионавигации и т.д.

Электромагнитные поля оказывают на организм человека тепловое и биологическое воздействие. Переменное электрическое поле вызывает нагрев диэлектриков (хрящей, сухожилий и др.) за счет токов проводимости и за счет переменной поляризации. Выделение теплоты может приводить к перегреванию, особенно тех органов и тканей, которые недостаточно хорошо снабжены кровеносными сосудами (хрусталик глаза, желчный пузырь, мочевой пузырь). Наиболее чувствительны к биологическому воздействию радиоволн центральная нервная и сердечно-сосудистая системы. При длительном действии радиоволн не слишком большой интенсивности (порядка $10 \frac{\text{Вт}}{\text{м}^2}$) появляются головные боли, быстрая утомляемость, изменение давления и пульса, нервно-психические расстройства. Может наблюдаться похудение, выпадение волос, изменение в составе крови.

Воздействие СВЧ-излучения интенсивностью более $100 \frac{\text{Вт}}{\text{м}^2}$ может привести к помутнению хрусталика глаза и потере зрения, тот же результат может дать длительное облучение умеренной интенсивности (порядка $10 \frac{\text{Вт}}{\text{м}^2}$), при этом возможны нарушения со стороны эндокринной системы, изменения углеводного и жирового обмена, сопровождающиеся похудением, повышение возбудимости, изменение ритма сердечной деятельности, изменения в крови (уменьшение количества лейкоцитов).

Действию электромагнитных полей промышленной частоты человек подвергается в производственной, городской и бытовой зонах. Санитарными нормами установлены предельно допустимые уровни напряженности

электрического поля внутри жилых зданий, на территории жилой зоны. Люди, страдающие от нарушений сна и головных болей, должны перед сном убирать или отключать электрические приборы, генерирующие электрические поля.

Воздействие электромагнитных полей может быть изолированным – от одного источника, сочетанным – от двух и более источников одного частотного диапазона, смешанным – от двух и более источников электромагнитных полей различных частотных диапазонов, и комбинированным – в случае одновременного действия какого-либо другого неблагоприятного фактора.

Воздействие может быть постоянным или прерывистым, общим (облучается все тело) или местным (облучается часть тела). В зависимости от места нахождения человека относительно источника излучения он может подвергаться воздействию электрической или магнитной составляющих поля или их сочетанию, а в случае пребывания в волновой зоне – воздействию сформированной электромагнитной волны. Контроль уровней электрического поля осуществляется по значению напряженности электрического поля, выраженной в $\frac{В}{м}$. Контроль уровней магнитного поля осуществляется по значению напряженности магнитного поля, выраженной в $\frac{А}{м}$.

Энергетическим показателем для волновой зоны излучения является плотность потока энергии, или интенсивность, – энергия, проходящая через единицу поверхности, перпендикулярной к направлению, распространения электромагнитной волны за одну секунду. Измеряется в $\frac{Вт}{м^2}$. Нормирование уровней в соответствии с ГОСТ 12.1.006-84.

Длительное действие электрических полей может вызывать головную боль в височной и затылочной области, ощущение вялости, расстройство сна, ухудшение памяти, депрессию, апатию, раздражительность, боли в области сердца. Для персонала ограничивается время пребывания в электрическом поле в зависимости от напряженности поля (180 минут в сутки при напряженности $10 \frac{кВ}{м}$, 10 минут в сутки при напряженности $20 \frac{кВ}{м}$).

У людей, работающих в зоне воздействия электростатического поля, встречаются разнообразные жалобы: на раздражительность, головную боль, нарушение сна, снижение аппетита и др. Характерны своеобразные «фобии», обусловленные страхом ожидаемого разряда. Склонность к «фобиям» обычно сочетается с повышенной эмоциональной возбудимостью [10].

Установлено также благотворное влияние на самочувствие снятия избыточного электростатического заряда с тела человека (заземление, хождение босиком).

Наибольшая опасность электростатических зарядов заключается в том, что искровой разряд может обладать энергией, достаточной для воспламенения горючей или взрывоопасной смеси. Искра, возникающая при разрядке электростатических зарядов, является частой причиной пожаров и взрывов.

Так, удаление из помещения пыли из диэлектрического материала с помощью вытяжной вентиляции может привести к накоплению в газоходах

электростатических зарядов и отложений пыли. Появление искрового разряда в этом случае может привести к воспламенению или взрыву пыли. Известны случаи очень серьезных аварий на предприятиях в результате взрывов в системах вентиляции.

При перевозке легковоспламеняющихся жидкостей, при их перекачке по трубопроводам, сливе из цистерны или за счет плескания жидкости накапливаются электростатические заряды, и может возникнуть искра, которая воспламенит жидкость.

Наибольшую опасность статическое электричество представляет на производстве и на транспорте, особенно при наличии пожаро-взрывоопасных смесей, пылей и паров легковоспламеняющихся жидкостей.

В бытовых условиях (например, при хождении по ковру) накапливаются небольшие заряды, и энергии возникших искровых разрядов недостаточно для инициирования пожара в обычных условиях быта.

6.5 Защита от статического электричества

Допустимые уровни напряженности электростатических полей установлены в ГОСТ 12.1.045-84. «Система стандартов безопасности труда. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля» Они зависят от времени пребывания на рабочих местах. Предельно допустимый уровень напряженности электростатических полей равен $60 \frac{\text{кВ}}{\text{м}}$ в 1 ч [12].

Применение средств защиты работающих обязательно в тех случаях, когда фактические уровни напряженности электростатических полей на рабочих местах превышают $60 \frac{\text{кВ}}{\text{м}}$.

При выборе средств защиты от статического электричества должны учитываться особенности технологических процессов, физико-химические свойства обрабатываемого материала, микроклимат помещений и др., что определяет дифференцированный подход при разработке защитных мероприятий.

Защита от статического электричества осуществляется двумя путями:

- уменьшением интенсивности образования электрических зарядов;
- устранением образовавшихся зарядов статического электричества.

Уменьшение интенсивности образования электрических зарядов достигается за счет снижения скорости и силы трения, различия в диэлектрических свойствах материалов и повышения их электропроводимости. Уменьшение силы трения достигается смазкой, снижением шероховатости и площади контакта взаимодействующих поверхностей. Скорости трения ограничивают за счет снижения скоростей обработки и транспортировки материалов.

Так как заряды статического электричества образуются при плескании, распылении и разбрызгивании диэлектрических жидкостей, желательно эти процессы устранять или, по крайней мере, их ограничивать. Например, наполнение диэлектрическими жидкостями резервуаров свободно падающей струей не

допускается. Сливной шланг необходимо опустить под уровень жидкости или, в крайнем случае, струю направить вдоль стенки, чтобы не было брызг [10].

Поскольку интенсивность образования зарядов тем выше, чем меньше электропроводность материала, то желательно применять по возможности материалы с большей электропроводностью или повышать их электропроводность путем введения электропроводных (антистатических) присадок. Так, для покрытия полов нужно использовать антистатический линолеум, желательно периодически проводить антистатическую обработку ковров, ковровых материалов, синтетических тканей и материалов с использованием препаратов бытовой химии.

Соприкасающиеся предметы и вещества предпочтительнее изготавливать из одного и того же материала, так как в этом случае не будет происходить контактной электролизаии. Например, полиэтиленовый порошок желательно хранить в полиэтиленовых бочках, а пересыпать и транспортировать по полиэтиленовым шлангам и трубопроводам. Если сделать это не представляется возможным, то применяют материалы, близкие по своим диэлектрическим свойствам. Например, электризация в паре фторопласт-полиэтилен меньше, нежели в паре фторопласт-эбонит.

Таким образом, для защиты от статического электричества необходимо применять слабоэлектризующиеся или неэлектризующиеся материалы, устранять или ограничивать трение, распыление, разбрызгивание, плескание диэлектрических жидкостей.

Устранение зарядов статического электричества достигается прежде всего заземлением корпусов оборудования. Заземление для отвода статического электричества можно объединять с защитным заземлением электрооборудования. Если заземление используется только для снятия статического электричества, то его электрическое сопротивление может быть существенно больше, чем для защитного сопротивления электрооборудования (до 100 Ом). Достаточно даже тонкого провода, чтобы электрические заряды постоянно стекали в землю [10].

Для снятия статического электричества с кузова автомобиля применяют электропроводную полосу – «антистатик», прикрепленную к днищу автомобиля. Если при выходе из автомобиля вы заметили, что кузов «искрит», разрядите кузов, прикоснувшись к нему металлическим предметом, например, ключом зажигания. Для человека это не опасно. Обязательно сделайте это, если собираетесь заправить машину бензином.

Самолеты снабжены металлическими тросиками, закрепленными на шасси и днищах фюзеляжа, что позволяет при посадке снимать с корпуса статические заряды, образовавшиеся в полете.

Для снятия электрических зарядов заземляются защитные экраны мониторов компьютеров. Бензозаправщики снабжаются заземлителями в виде цепей, постоянно контактирующих с землей при движении автомобиля. При сливе бензина в цистерны на бензозаправочной станции автомобиль-заправщик и система слива бензина обязательно заземляются дополнительно.

Влажный воздух имеет достаточную электропроводность, чтобы образующиеся электрические заряды стекали через него. Поэтому во влажной воздушной

среде электростатических зарядов практически не образуется, и увлажнение воздуха является одним из наиболее простых и распространенных методов борьбы со статическим электричеством.

Еще один распространенный метод устранения электростатических зарядов – ионизация воздуха. Образующиеся при работе ионизатора ионы нейтрализуют заряды статического электричества. Таким образом, бытовые ионизаторы воздуха не только улучшают аэроионный состав воздушной среды в помещении, но и устраняют электростатические заряды, образующиеся в сухой воздушной среде на коврах, ковровых синтетических покрытиях, одежде. На производстве используют специальные мощные ионизаторы воздуха различных конструкций, но наиболее распространены электрические ионизаторы.

В качестве индивидуальных средств защиты могут применяться антистатическая обувь, антистатические халаты, заземляющие браслеты для защиты рук и другие средства, обеспечивающие электростатическое заземление тела человека.

При грозе, во время ударов молнии в различные промышленные, транспортные и другие объекты, находящиеся вдали от производственных зданий и сооружений, возможно проникновение (занос) электростатических потенциалов в здание по внешним металлическим сооружениям и коммуникациям – эстакадам, монорельсам и канатам подвесных дорог, по трубопроводам, оболочкам кабелей и т.д.

Для приема электрического разряда молнии и отвода ее в землю применяют устройства, называемые молниеотводами. Молниеотвод состоит из несущей части – опоры (которой может служить само здание или сооружение), молниеприемника, токоотвода и заземления. Наиболее распространенные стержневые и Тросовые молниеотводы.

При выполнении молниезащиты зданий и сооружений для повышения безопасности людей и животных необходимо заземлители молниеотводов (кроме углубленных) размещать в редко посещаемых местах, в удалении на 5 метров и более от грунтовых, проезжих и пешеходных дорог.

Для защиты от проявления электростатической индукции в зданиях и сооружениях, присоединяют металлические корпуса всего оборудования, установленного в защищаемом здании, к специальному заземлителю или к защитному заземлению местной электросети; отдельно стоящие неизолированные тросовые и стержневые молниеотводы, наложением молниеприемной сети на плоскую неметаллическую кровлю.

7 РЕСУРСО- И ЭНЕРГОСБЕРЕЖЕНИЕ ПРИ ВНЕДРЕНИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

7.1 Основные понятия в области ресурсо- и энергосбережения

Ресурсы – ценности, запасы, возможности, источники дохода в государственном бюджете. В общем виде ресурсы делятся на природные и экономические (трудовые, финансовые, материальные).

Ресурсосбережение – деятельность (организационная, экономическая, техническая, научная, практическая, информационная), совокупность организационно-технических мер и мероприятий, которые сопровождают все стадии жизненного цикла объекта и ориентированы на рациональное использование и обоснованное расходование ресурсов. Различают энергосбережение и материалосбережение.

Ресурсосбережение обеспечивается посредством:

- использования ресурсосберегающих и энергосберегающих технологий;
- снижения фондоемкости и материалоемкости продукции;
- повышения производительности труда;
- сокращения затрат живого и овеществленного труда.

Энергосбережение (экономия электроэнергии) – реализация правовых, организационных, научных, производственных, технических и экономических мер, направленных на рациональное использование (и экономное расходование) энергетических ресурсов и на вовлечение в хозяйственный оборот возобновляемых источников энергии.

Эффекты от мероприятий энергосбережения рассчитывают:

- как стоимость сэкономленных энергоресурсов или доля стоимости от потребляемых энергоресурсов, в том числе на единицу продукции;
- как количество тонн условного топлива сэкономленных энергоресурсов или доля от величины потребляемых энергоресурсов;
- в натуральном выражении;
- как снижение доли энергоресурсов в валовом внутреннем продукте в стоимостном выражении, либо в натуральных единицах на 1 руб. валового внутреннего продукта.

7.2 Ресурсо- и энергосбережение в Республике Беларусь

Республика Беларусь не относится к странам, имеющим достаточное количество собственных энергетических ресурсов (например, Дания, Швейцария, Япония). Однако их опыт показывает, что экономика может динамично развиваться вследствие эффективного использования топливно-энергетических ресурсов, проведения энергосберегающих мероприятий, освоения энергоэффективных технологий, уменьшения уровня издержек производства.

В республике в течение последних двадцати пяти лет наблюдается прогресс в сфере энергосбережения и оптимизации топливно-энергетического

баланса. В результате, за период 1996-2020 гг. значительно снизилась энергоёмкость валового внутреннего продукта. Динамика отображена на рисунке 7.1.

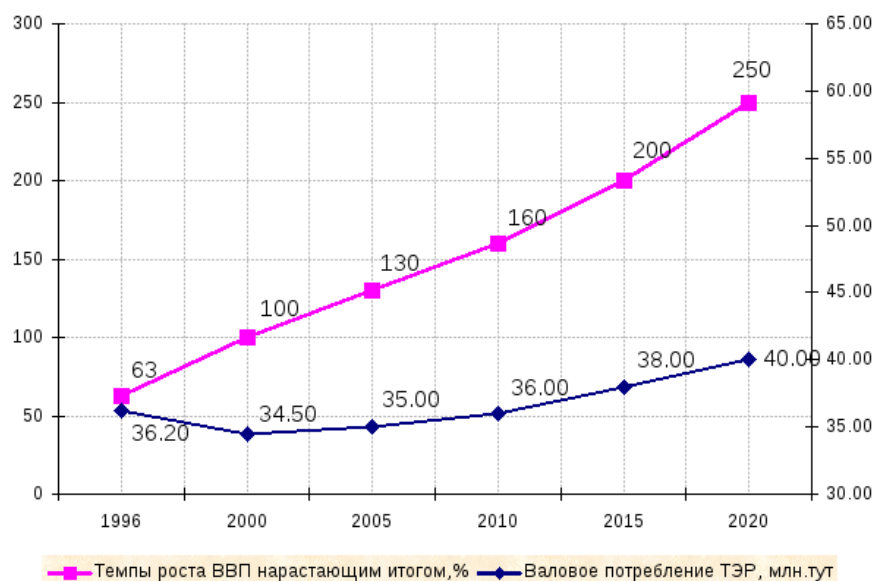


Рисунок 7.1 – Динамика валового потребления ТЭР по отношению к ВВП

Такие значительные результаты в повышении эффективности энергоиспользования в народном хозяйстве достигнуты в большой степени благодаря сложившемуся в республике системному подходу в работе по энергосбережению. На основании анализа состояния и эффективности энергоиспользования в народном хозяйстве, имеющихся резервов экономии топливно-энергетических ресурсов разрабатываются целевые показатели по энергосбережению и в установленном порядке доводятся в виде заданий министерствам, ведомствам и регионам.

Ресурсосбережение и энергосбережение способствует росту эффективности экономики и повышению ее конкурентоспособности.

Экономия ресурсов, связанная с внедрением разработанного программного обеспечения, заключается в сокращении трудоемкости, а также канцелярских принадлежностей.

Согласно результатам данной работы, можно сказать, что программное обеспечение призвано упростить процесс взаимодействия абитуриентов и студентов с информационными ресурсами университета, а также максимально оптимизировать работу приемной кампании. Таким образом, становится возможным значительно уменьшить количество времени, требуемое для реализации поставленной задачи.

7.3 Расчет показателей ресурсо- и энергосбережения

Расчет экономии расходов происходит по формуле (7.1):

$$\mathcal{E} = K \cdot C_{\text{зп}} \cdot n \cdot 12 \cdot O_p, \quad (7.1)$$

где K – коэффициент сокращения остальных расходов;
 $C_{\text{зп}}$ – среднемесячная заработная плата служащего, руб.;
 n – количество рабочих;
 O_p – остальные расходы, %.

Среднемесячная заработная плата служащего составляет 640 рублей. Остальные расходы принимаем равными 3%. Коэффициент сокращения остальных расходов $K = 0,5$. Количество рабочих $n = 1$.

Таким образом:

$$\mathcal{E} = 0,5 \cdot 640 \cdot 1 \cdot 12 \cdot 0,03 = 115,2 \text{ руб.}$$

По результатам проведенной оценки было установлено, что внедрение нового программного продукта является целесообразным, так как есть возможность сэкономить на других расходах.

Далее по формуле (7.2) рассчитаем экономию электроэнергии при работе мобильного приложения:

$$\mathcal{E} = (T_p - T_a) \cdot P \cdot C_{\text{эл}} \cdot K_{\text{и}}, \quad (7.2)$$

где T_p – трудоемкость работы вручную, часов;
 T_a – трудоемкость работы с помощью программного продукта, часов;
 P – паспортная мощность персонального компьютера, кВт;
 $C_{\text{эл}}$ – стоимость одного $\frac{\text{кВт}}{\text{ч}}$ электроэнергии, руб;
 $K_{\text{и}}$ – коэффициент использования персонального компьютера.

Для проведения всего цикла работы от реализации входных данных до получения желаемых результатов с использованием компьютера тратится около нуля целых пяти десятых человека-часа рабочего времени, для этой же операции с применением разработанного программного продукта требуется порядка нуля целых одной десятой человека-часа. По состоянию на 1 января 2021 года, стоимость одного $\frac{\text{кВт}}{\text{ч}}$ электроэнергии в Республике Беларусь равна 0,2867 рублей. Примем паспортную мощность персонального компьютера равной 0,47 кВт и рассчитаем стоимость сэкономленной электроэнергии за один месяц при пятидневной рабочей неделе и восьмичасовом рабочем дне:

$$\mathcal{E} = (0,5 - 0,1) \cdot 0,47 \cdot 0,2867 \cdot (5 \cdot 8 \cdot 4) \cdot 0,5 = 4,31 \text{ руб.}$$

Данный расчет показывает, что при внедрении разработанного программного продукта можно сэкономить на электроэнергии 4,31 рубля в месяц, соответственно, на 51,72 рубль в год.

ЗАКЛЮЧЕНИЕ

Основываясь на постановке задачи дипломной работы, было разработано мобильное приложение для абитуриентов, имеющее ряд преимуществ по сравнению с аналогами. Приложение является интуитивно понятным, простое в использовании, имеет дружелюбный для пользователя интерфейс с современным дизайном.

В ходе выполнения данной дипломной работы были пройдены следующие важные этапы по созданию мобильного приложения:

- аналитический обзор существующих аналогов, современных технологий и сред программирования, который позволил получить необходимую информацию для постановки задачи на проектирование, выделить разработанный продукт среди аналогов;
- анализ предметной области и разработка алгоритмов работы мобильного приложения позволили определить необходимые инструменты и технологии в работе, способ хранения данных, архитектуру приложения, его функционал;
- программная реализация приложения, в результате завершения которой был создан программный продукт, состоящий из мобильного приложения и *Python*-скрипта для получения данных с сайта;
- тестирование приложения, в результате была проверена корректность и стабильность работы программного обеспечения в процессе его использования.

В результате выполнения работы было создано уникальное на данный момент программное обеспечение, реализующее функционал регистрации абитуриента, входа в личный кабинет, использования перечня сервисов.

Использование данного мобильного приложения позволит сократить время абитуриентов и приемной комиссии во время проведения приемной кампании, повысит популярность информационных ресурсов университета в глазах абитуриентов и студентов.

В дальнейшем планируется выполнять поддержку и улучшение функционала разработанного программного обеспечения, также возможно интегрирование в систему ВУЗа.

Цели данной дипломной работы были полностью достигнуты путем выполнения всех поставленных задач.

Список использованных источников

1. Черемных, С.В. Структурный анализ систем: IDEF-технологии / С.В. Черемных, И.О. Семенов, В.С. Ручкин. – Москва : Финансы и статистика. 2015. – 208 с.
2. Диаграмма прецедентов – Электрон. данные. : Режим доступа: http://it.samgtu.ru/sites/it.samgtu.ru/files/diagrammy_precedentov.pdf. – Дата доступа: 14.04.2021.
3. SQL Azure – Электрон. данные. : Режим доступа: <https://habr.com/ru/post/66815/>. – Дата доступа: 29.04.2021.
4. Медникс З. Программирование под Android / З. Медникс, Л. Дорнин. – Санкт-Петербург : Издательский Дом Питер, 2012. – 560 с.
5. Дейтел П. Android для программистов: создаем приложения. – Санкт-Петербург : Издательский Дом Питер. 2012. – 560 с.
6. Онлайн-портал по разработке мобильных приложений RA Solution: Основные мобильные платформы: Android, Windows Phone, Symbian и iOS – Электрон. данные. – Режим доступа: http://rasolution.ru/ru/articles/mobile_platform/. – Дата доступа: 14.04.2021.
7. Кожевников, Е. А. Расчет экономической эффективности разработки программных продуктов : методические указания по подготовке организационно-экономического раздела дипломных работ для студентов специальности 1-40-01-02 «Информационные системы и технологии (по направлениям)» дневной формы обучения / Е. А. Кожевников, Н. В. Ермалинская. – Гомель : ГГТУ им. П. О. Сухого, 2012. – 68 с.
8. Об охране труда: Закон Республики Беларусь №356-3: подписан Президентом Республики Беларусь 23 июня 2008 г.: рег. № 61-3 от 12 июля 2013 г.: текст по состоянию на 18 декабря 2019 г.
9. Электростатическая искробезопасность. Общие технические требования и методы испытаний: ГОСТ 31613-2012. – Введ. 15.02.2013. – Москва : Стандартинформ, 2013. – 16 с.
10. Безопасность жизнедеятельности / под ред. Э. А. Арустамова. – Москва : Дашков и К, 2000. – 678 с.
11. Бутиков, Е. И. Физика для поступающих в вузы / Е. И. Бутиков, А. А. Быков, А. С. Кондратьев. – Москва : Наука, 1991. – 640 с.
12. Система стандартов безопасности труда. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля: ГОСТ 12.1.045-84. – Введ. 01.07.1985. – Москва : Стандартинформ, 2006. – 2 с.

ПРИЛОЖЕНИЕ А

(обязательное)

Исходный текст программного продукта

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    package="com.example.diplomaproject">
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.WAKE_LOCK" />

    <uses-feature android:name="android.hardware.type.watch" />

    <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"
/>
    <uses-permission android:name="android.permission.INTERNET" />
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
    <uses-permission      android:name="android.permission.WRITE_EXTERNAL_STOR-
AGE"/>

    <application
        android:allowBackup="true"
        android:icon="@drawable/gstu"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.DiplomaProject">
        <activity
            android:name=".businesslogic.account.ViewAccountActivity"
            android:label="Просмотр аккаунта"
            android:parentActivityName=".MainActivity" />
        <activity
            android:name=".businesslogic.LoginActivity"
            android:label="Вход в личный кабинет"
            android:parentActivityName=".MainActivity" />
        <activity
            android:name=".businesslogic.account.CreateAccountActivity"
            android:label="Создание аккаунта"
            android:parentActivityName=".MainActivity" />
        <activity
            android:name=".MainActivity"
            android:label="Абитуриент ГГТУ"
            android:theme="@style/Theme.DiplomaProject.NoActionBar">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
```

```

</activity>

<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="AIzaSyDgDz0Lyg97G_-Youb2HILsCwuaa9IFouY" />

<activity
    android:name=".businesslogic.map.MapsActivity"
    android:label="Университет на карте"
    android:parentActivityName=".MainActivity" />

<uses-library
    android:name="com.google.android.maps"
    android:required="false" />

<meta-data
    android:name="com.google.android.gms.version"
    android:value="12451000"
    tools:replace="android:value" />
<meta-data
    android:name="com.google.android.wearable.standalone"
    android:value="true" />
</application>

</manifest>

```

CreateAccountActivity.java

```

package com.example.diplomaproject.businesslogic.account;

import android.annotation.SuppressLint;
import android.app.DatePickerDialog;
import android.content.Intent;
import android.os.Bundle;
import android.text.format.DateUtils;
import android.view.View;
import android.view.inputmethod.InputMethodManager;
import android.widget.AdapterView;
import android.widget.CheckBox;
import android.widget.EditText;
import android.widget.Spinner;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;

import com.example.diplomaproject.R;
import com.example.diplomaproject.data.database.ConnectionHelper;
import com.example.diplomaproject.businesslogic.LoginActivity;
import com.example.diplomaproject.businesslogic.dialog.Creatable;
import com.example.diplomaproject.businesslogic.dialog.CreateDialogFragment;
import com.example.diplomaproject.validators.interfaces.ErrorCallBack;

```

```

import com.example.diplomaproject.validators.validators.PhoneNumber;
import com.google.android.material.snackbar.Snackbar;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import java.util.HashMap;

public class CreateAccountActivity extends AppCompatActivity implements Creatable, ErrorCallBack {

    Spinner spinner;
    String ConnectionResult;
    Connection connection;
    ConstraintLayout CLCreate;
    int USER_ID;
    final String OLD_FORMAT = "dd MMM yyyy r.";
    final String NEW_FORMAT = "yyyy-MM-dd";
    HashMap<String, Boolean> phoneNumber;
    TextView currentDate;
    Calendar date = Calendar.getInstance();

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_create_account);
        spinner = findViewById(R.id.EducationForm);
        CLCreate = findViewById(R.id.CLCreate);
        currentDate = findViewById(R.id.DateOfBirth);
        setInitialDate();
        fillSpinner();

        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        Intent intent = getIntent();
        USER_ID = intent.getIntExtra("userId", 0);
    }

    public void fillSpinner() {
        try {
            ConnectionHelper connectionHelper = new ConnectionHelper();
            connection = connectionHelper.connection();
            if (connection != null) {
                String query = "SELECT Name FROM EDUCATIONFORM";
                Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery(query);
                if (resultSet != null) {
                    ArrayList<String> data = new ArrayList<>();
                    while (resultSet.next()) {

```

```

        String name = resultSet.getString("Name");
        data.add(name);
    }
    ArrayAdapter<String> arrayAdapter = new ArrayAdapter<>(this,
        android.R.layout.simple_spinner_item, data);
    spinner.setAdapter(arrayAdapter);
}
} else {
    ConnectionResult = "Check Connection";
}
} catch (Exception exception) {
    ConnectionResult = exception.getMessage();
}
}

public void onCreateModelButtonClick(View view) {
    CreateDialogFragment dialog = new CreateDialogFragment();
    dialog.show(getSupportFragmentManager(), "createDialog");
}

public int getFormId(String formName) {
    int id = 0;
    try {
        ConnectionHelper connectionHelper = new ConnectionHelper();
        connection = connectionHelper.connection();
        if (connection != null) {
            String query = "SELECT * FROM EDUCATIONFORM WHERE Name=N'" +
formName + "'";
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(query);
            if (resultSet != null) {
                while (resultSet.next()) {
                    id = resultSet.getInt("FormId");
                }
            }
        } else {
            ConnectionResult = "Check Connection";
        }
    } catch (Exception exception) {
        ConnectionResult = exception.getMessage();
    }
    return id;
}

public int convertBooleanToInt(boolean boolValue) {
    return boolValue ? 1 : 0;
}

public void onSetDateBtnClick(View view) {
    new DatePickerDialog(CreateAccountActivity.this, d,
        date.get(Calendar.YEAR),

```

```

        date.get(Calendar.MONTH),
        date.get(Calendar.DAY_OF_MONTH))
        .show();
    }

    private void setInitialDate() {

        currentDate.setText(DateUtils.formatDateTime(this,
            date.getTimeInMillis(),
            DateUtils.FORMAT_SHOW_DATE | DateUtils.FORMAT_SHOW_YEAR));
    }

    // установка обработчика выбора даты
    DatePickerDialog.OnDateSetListener d = (view, year, monthOfYear, dayOfMonth) -> {
        date.set(Calendar.YEAR, year);
        date.set(Calendar.MONTH, monthOfYear);
        date.set(Calendar.DAY_OF_MONTH, dayOfMonth);
        setInitialDate();
    };

    @SuppressWarnings("CutPasteId")
    @Override
    public void create() {
        phoneNumber = new PhoneNumber.PhoneNumberBuilder(((EditText) findViewById(R.id.PhoneNumber)).getText().toString())
            .setRequired(true)
            .setMinLength(12)
            .setMaxLength(12)
            .build();
        PhoneNumber number = new PhoneNumber(CreateAccountActivity.this);
        if (number.isValid(phoneNumber)) {
            try {
                //databaseAdapter = new DatabaseAdapter(this.getApplicationContext());
                String surname = ((EditText) findViewById(R.id.Surname)).getText().toString();
                String name = ((EditText) findViewById(R.id.Name)).getText().toString();
                String patronymic = ((EditText) findViewById(R.id.Patronymic)).getText().toString();
                String passport = ((EditText) findViewById(R.id.Passport)).getText().toString();
                String address = ((EditText) findViewById(R.id.Address)).getText().toString();

                String oldDateString = ((TextView) findViewById(R.id.DateOfBirth)).getText().toString();
                String newDateString;

                @SuppressWarnings("SimpleDateFormat")
                SimpleDateFormat simpleDateFormat = new SimpleDateFormat(OLD_FORMAT);

                Date date = simpleDateFormat.parse(oldDateString);
                simpleDateFormat.applyPattern(NEW_FORMAT);
                newDateString = simpleDateFormat.format(date);
                Date newDate = simpleDateFormat.parse(newDateString);
                java.sql.Date sqlDate = new java.sql.Date(newDate.getTime());
            }
        }
    }

```

```

        String phone = ((EditText) findViewById(R.id.PhoneNumber)).getText().toString();

        String educationForm = spinner.getSelectedItem().toString();

        boolean isAimed = ((CheckBox) findViewById(R.id.IsAimed)).isChecked();
        boolean isWithoutExams = ((CheckBox) findViewById(R.id.IsWithoutExams)).isChecked();

        boolean isBudget = ((CheckBox) findViewById(R.id.IsBudget)).isChecked();
        boolean isOutOfCompetitions = ((CheckBox) findViewById(R.id.IsOutOfCompetitions)).isChecked();

        String profSubj1EE = ((EditText) findViewById(R.id.ProfSubj1EE)).getText().toString();
        String profSubj2EE = ((EditText) findViewById(R.id.ProfSubj2EE)).getText().toString();
        String subj3EE = ((EditText) findViewById(R.id.Subj3EE)).getText().toString();
        String certificateScoreX10 = ((EditText) findViewById(R.id.CertificateScoreX10)).getText().toString();

        boolean isAgreeForPayment = ((CheckBox) findViewById(R.id.IsAgreeForPayment)).isChecked();

        boolean checkIntValue = checkInt(Integer.parseInt(profSubj1EE),
            Integer.parseInt(profSubj2EE), Integer.parseInt(subj3EE),
            Integer.parseInt(certificateScoreX10));

        if(checkIntValue) {
            try {
                ConnectionHelper connectionHelper = new ConnectionHelper();
                connection = connectionHelper.connection();
                if (connection != null) {
                    String query = "INSERT INTO ENROLLEE VALUES (" + USER_ID + ", "
                        + N"" + surname + "", N"" +
                            name + "", N"" + patronymic + "", N"" + passport + "", N"" + address +
                            "", "" + sqlDate + "", "" + phone + "", " + getFormId(educationForm) +
                            "", " + convertBooleanToInt(isAimed) + ", " +
                            convertBooleanToInt(isWithoutExams) + ", " +
                            convertBooleanToInt(isBudget) + ", " +
                            convertBooleanToInt(isOutOfCompetitions) + ", " +
                            Integer.parseInt(profSubj1EE) + ", " + Integer.parseInt(profSubj2EE)
                        +
                            ", " + Integer.parseInt(subj3EE)
                        + ", " + Integer.parseInt(certificateScoreX10) + ", " +
                            convertBooleanToInt(isAgreeForPayment) + ")";
                    Statement statement = connection.createStatement();
                    statement.executeUpdate(query);

                } else {
                    ConnectionResult = "Check Connection";
                }
            } catch (Exception exception) {

```

```

        showSnackBar("Вы ввели неверные данные");
    }
}
else
{
    showSnackBar("Вы ввели баллы в неверном диапазоне");
}
} catch (Exception exception) {
}
Intent intent = new Intent(this, LoginActivity.class);
startActivity(intent);
}
}

private void showSnackBar(String msg) {
    Snackbar snackbar = Snackbar
        .make(CLCreat, msg, Snackbar.LENGTH_LONG);
    snackbar.show();
}

@Override
public void onValidationError(String s) {
    hideKeyboard();
    showSnackBar(s);
}

private void hideKeyboard() {
    InputMethodManager inputMethodManager = (InputMethodManager) getSystemService(CreateAccountActivity.INPUT_METHOD_SERVICE);
    if (inputMethodManager != null) {
        inputMethodManager.hideSoftInputFromWindow(getWindow().getDecorView().getWindowToken(), 0);
    }
}

public boolean checkInt(int first, int second, int third, int fourth) {
    return first > 0 && first < 101 && second > 0 && second < 101
        && third > 0 && third < 101 && fourth > 0 && fourth < 101;
}
}

```

ViewAccountActivity.java

```

package com.example.diplomaproject.businesslogic.account;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.widget.CheckBox;
import android.widget.TextView;

```



```

import androidx.appcompat.app.AppCompatActivity;

import com.example.diplomaproject.R;
import com.example.diplomaproject.data.database.ConnectionHelper;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;

public class ViewAccountActivity extends AppCompatActivity {

    String ConnectionResult = "";
    Connection connection;
    int USER_ID;
    TextView surnameT;
    TextView nameT;
    TextView patronymicT;
    TextView passportT;
    TextView addressT;
    TextView dateOfBirthT;
    TextView phoneNumberT;
    TextView educationFormT;
    CheckBox isAimedC;
    CheckBox isWithoutExamsC;
    CheckBox isBudgetC;
    CheckBox isOutOfCompetitionsC;
    TextView profSubj1T;
    TextView profSubj2T;
    TextView subj3T;
    TextView certificateT;
    CheckBox isAgreeForPaymentC;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_view_account);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        Intent intent = getIntent();
        USER_ID = intent.getIntExtra("userId", 0);
        surnameT = findViewById(R.id.Surname);
        nameT = findViewById(R.id.Name);
        patronymicT = findViewById(R.id.Patronymic);
        passportT = findViewById(R.id.Passport);
        addressT = findViewById(R.id.Address);
        dateOfBirthT = findViewById(R.id.DateOfBirth);
        phoneNumberT = findViewById(R.id.PhoneNumber);
        educationFormT = findViewById(R.id.EducationForm);
        isAimedC = findViewById(R.id.IsAimed);
        isWithoutExamsC = findViewById(R.id.IsWithoutExams);
        isBudgetC = findViewById(R.id.IsBudget);
        isOutOfCompetitionsC = findViewById(R.id.IsOutOfCompetitions);
        profSubj1T = findViewById(R.id.ProfSubj1EE);
    }
}

```

```

        profSubj2T = findViewById(R.id.ProfSubj2EE);
        subj3T = findViewById(R.id.Subj3EE);
        certificateT = findViewById(R.id.CertificateScoreX10);
        isAgreeForPaymentC = findViewById(R.id.IsAgreeForPayment);
        viewInfo();
    }

    @SuppressWarnings("SetTextI18n")
    public void viewInfo() {
        String surname, name, patronymic, passport, address, dateOfBirth, phoneNumber, educationForm;
        boolean isAimed, isWithoutExams, isBudget, isOutOfCompetitions, isAgreeForPayment;
        int profSubj1, profSubj2, subj3, certificate;
        try {
            ConnectionHelper connectionHelper = new ConnectionHelper();
            connection = connectionHelper.connection();
            if (connection != null) {
                String query = "SELECT e.Surname, e.Name, e.Patronymic, e.Passport, e.Address, " +
                    "e.DateOfBirth, e.PhoneNumber, f.Name, e.IsAimed, e.IsWithoutEntranceExams," +
                    "e.IsBudget, e.IsOutOfCompetition, e.ProfSubj1EE, e.ProfSubj2EE, e.Subj3EE, " +
                    "e.CertificateScoreX10, e.IsAgreeForPayment FROM ENROLLEE e JOIN " +
                    "EDUCATIONFORM f ON f.FormId = e.FormID WHERE e.UserID=" +
                    USER_ID + """;
                Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery(query);
                if (resultSet != null) {
                    while (resultSet.next()) {
                        surname = resultSet.getString(1);
                        name = resultSet.getString(2);
                        patronymic = resultSet.getString(3);
                        passport = resultSet.getString(4);
                        address = resultSet.getString(5);
                        dateOfBirth = resultSet.getDate(6).toString();
                        phoneNumber = resultSet.getString(7);
                        educationForm = resultSet.getString(8);
                        isAimed = convertIntToBoolean(resultSet.getInt(9));
                        isWithoutExams = convertIntToBoolean(resultSet.getInt(10));
                        isBudget = convertIntToBoolean(resultSet.getInt(11));
                        isOutOfCompetitions = convertIntToBoolean(resultSet.getInt(12));
                        profSubj1 = resultSet.getInt(13);
                        profSubj2 = resultSet.getInt(14);
                        subj3 = resultSet.getInt(15);
                        certificate = resultSet.getInt(16);
                        isAgreeForPayment = convertIntToBoolean(resultSet.getInt(16));
                        setData(surname, name, patronymic, passport, address, dateOfBirth,
                            phoneNumber, educationForm, isAimed, isWithoutExams, isBudget,

```

```

        isOutOfCompetitions, profSubj1, profSubj2, subj3,
        certificate, isAgreeForPayment);
    }
} else {
    ConnectionResult = "Check Connection";
}
} catch (Exception exception) {
    ConnectionResult = exception.getMessage();
}
}

@SuppressLint("SetTextI18n")
public void setData(String surname, String name, String patronymic, String passport,
    String address, String dateOfBirth, String phoneNumber,
    String educationForm, boolean isAimed, boolean isWithoutExams,
    boolean isBudget, boolean isOutOfCompetitions, int profSubj1, int profSubj2,
    int subj3, int certificate, boolean isAgreeForPayment) {
    surnameT.setText(surname);
    nameT.setText(name);
    patronymicT.setText(patronymic);
    passportT.setText(passport);
    addressT.setText(address);
    dateOfBirthT.setText(dateOfBirth);
    phoneNumberT.setText(phoneNumber);
    educationFormT.setText(educationForm);
    isAimedC.setChecked(isAimed);
    isWithoutExamsC.setChecked(isWithoutExams);
    isBudgetC.setChecked(isBudget);
    isOutOfCompetitionsC.setChecked(isOutOfCompetitions);
    profSubj1T.setText(Integer.toString(profSubj1));
    profSubj2T.setText(Integer.toString(profSubj2));
    subj3T.setText(Integer.toString(subj3));
    certificateT.setText(Integer.toString(certificate));
    isAgreeForPaymentC.setChecked(isAgreeForPayment);
}

public boolean convertIntToBoolean(int intValue) {
    return intValue > 0;
}
}

```

Creatable.java

```

package com.example.diplomaproject.businesslogic.dialog;

public interface Creatable {
    void create ();
}

```

CreateDialogFragment.java

```

package com.example.diplomaproject.businesslogic.dialog;

import android.app.AlertDialog;
import android.app.Dialog;
import android.content.Context;
import android.os.Bundle;

import androidx.annotation.NonNull;
import androidx.fragment.app.DialogFragment;

public class CreateDialogFragment extends DialogFragment {
    private Creatable creatable;

    @Override
    public void onAttach(Context context){
        super.onAttach(context);
        creatable = (Creatable) context;
    }
    @NonNull
    public Dialog onCreateDialog(Bundle savedInstanceState) {

        AlertDialog.Builder builder=new AlertDialog.Builder(getActivity());
        return builder
            .setTitle("Подтвердите действие")
            .setIcon(android.R.drawable.ic_dialog_info)
            .setMessage("После отправки редактирование данных невозможно. При
успешной регистрации вы будете перенаправлены на страницу авторизации")
            .setPositiveButton("OK", (dialog, which) -> {
                creatable.create();
            })
            .setNegativeButton("Отмена", null)
            .create();
    }
}

```

MapsFragment.java

```

package com.example.diplomaproject.businesslogic.map;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;

```

```

public class MapsFragment extends Fragment {

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState){

        View root = inflater.inflate(R.layout.fragment_maps, container, false);
        Intent intent = new Intent(getActivity(), MapsActivity.class);
        startActivity(intent);
        return root;
    }
}

```

MapsActivity.java

```

package com.example.diplomaproject.businesslogic.map;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

import com.example.diplomaproject.R;
import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;

public class MapsActivity extends AppCompatActivity implements OnMapReadyCallback {

    private GoogleMap mMap;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_maps);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        LatLng gstu = new LatLng(52.4068247, 30.9370456);
        googleMap.addMarker(new MarkerOptions().position(gstu)

```

```

        .title("ГГТУ им. П. О. Сухого").snippet("троллейбусы № 2, 19, 20, автобусы №
16, 17, 20, 21"));
        mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(gstu, 15));
    }
}

```

SendMailFragment.java

```

package com.example.diplomaproject.businesslogic.sendemail;

import android.content.Intent;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;

import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;

public class SendMailFragment extends Fragment implements View.OnClickListener{

    EditText editTextSubject, editTextMessage;
    Button send;

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View root = inflater.inflate(R.layout.fragment_send_mail, container, false);

        editTextSubject = root.findViewById(R.id.subjEditText);
        editTextMessage = root.findViewById(R.id.messageEditText);
        send = root.findViewById(R.id.sendEmailButton);

        send.setOnClickListener(this);

        return root;
    }

    @Override
    public void onClick(View v) {
        String subject = editTextSubject.getText().toString();
        String message = editTextMessage.getText().toString();

        Intent email = new Intent(Intent.ACTION_SEND);
        email.putExtra(Intent.EXTRA_EMAIL, new String[]{ Utils.EMAIL });
        email.putExtra(Intent.EXTRA_SUBJECT, subject);
        email.putExtra(Intent.EXTRA_TEXT, message);

        //need this to prompts email client only
    }
}

```

```

        email.setType("message/rfc822");

        startActivity(Intent.createChooser(email, "Выберите почтовый клиент:"));
        editTextMessage.setText("");
        editTextSubject.setText("");
    }
}

```

Utils.java

```

package com.example.diplomaproject.businesslogic.sendemail;

public class Utils {

    public static final String EMAIL = "gstuenrollee@gmail.com";

}

```

MyRunTimesData.java

```

package com.example.diplomaproject.businesslogic.web;

public class MyRunTimesData {
    private static String myStringData;

    public static String getMyStringData() {
        return myStringData;
    }

    public static void setMyStringData(String myStringData) {
        MyRunTimesData.myStringData = myStringData;
    }
}

```

SiteFragment.java

```

package com.example.diplomaproject.businesslogic.web;

import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.TextView;

import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;

```

```

public class SiteFragment extends Fragment {

    TextView contentView;
    String contentText = null;
    WebView webView;
    String url = "https://www.gstu.by/";

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View root = inflater.inflate(R.layout.fragment_site, container, false);
        contentView = root.findViewById(R.id.content);
        webView = root.findViewById(R.id.webView);
        webView.setWebViewClient(new WebViewClient(){
            @Override
            public void onReceivedError(WebView view, int errorCode,
                                       String description, String failingUrl) {
                Log.d("WEB_VIEW_TEST", "error code:" + errorCode + " - " + description);
            }
        });
        webView.loadUrl(url);
        if (contentText != null) {
            webView.loadData(contentText, "text/html; charset=utf-8", "utf-8");
        }
        return root;
    }
}

```

WebFragment.java

```

package com.example.diplomaproject.businesslogic.web;

import android.content.Context;
import android.os.Bundle;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.webkit.WebView;
import android.webkit.WebViewClient;
import android.widget.TextView;

import androidx.fragment.app.Fragment;

import com.example.diplomaproject.MainActivity;
import com.example.diplomaproject.R;

public class WebFragment extends Fragment {
    TextView contentView;

```



```

String contentText = null;
WebView webView;
String url = MyRunTimesData.getMyStringData();

@Override
public View onCreateView(LayoutInflater inflater, ViewGroup container,
                          Bundle savedInstanceState) {
    View root = inflater.inflate(R.layout.fragment_web, container, false);
    contentView = root.findViewById(R.id.content);
    webView = root.findViewById(R.id.webView);
    webView.setWebViewClient(new WebViewClient(){
        @Override
        public void onReceivedError(WebView view, int errorCode,
                                    String description, String failingUrl) {
            Log.d("WEB_VIEW_TEST", "error code:" + errorCode + " - " + description);
        }
    });
    webView.loadUrl(url);
    if (contentText != null) {
        webView.loadData(contentText, "text/html; charset=utf-8", "utf-8");
    }
    return root;
}

public void onAttach(Context context) {
    super.onAttach(context);
    ((MainActivity)context).getSupportActionBar().setTitle("Абитуриент ГГТУ");
}
}

```

CalculationFragment.java

```

package com.example.diplomaproject.businesslogic;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.constraintlayout.widget.ConstraintLayout;
import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;
import com.google.android.material.snackbar.Snackbar;

public class CalculationFragment extends Fragment implements View.OnClickListener{

```

```

Button calculateBtn;
EditText subj1;
EditText subj2;
EditText subj3;
EditText certificate;
TextView result;
ConstraintLayout CLCalculation;

public View onCreateView(@NonNull LayoutInflater inflater,
                        ViewGroup container, Bundle savedInstanceState){

    View root = inflater.inflate(R.layout.fragment_calculator, container, false);

    subj1 = root.findViewById(R.id.subj1Edit);
    subj2 = root.findViewById(R.id.subj2Edit);
    subj3 = root.findViewById(R.id.subj3Edit);
    certificate = root.findViewById(R.id.certificateEdit);
    result = root.findViewById(R.id.resultText);
    calculateBtn = root.findViewById(R.id.calculateBtn);
    CLCalculation = root.findViewById(R.id.calculateFragment);
    calculateBtn.setOnClickListener(this);
    return root;
}

@SuppressLint({ "NonConstantResourceId", "SetTextI18n" })
@Override
public void onClick(View v) {
    if (v.getId() == R.id.calculateBtn) {
        if(Integer.parseInt(subj1.getText().toString()) > 0 &&
            Integer.parseInt(subj1.getText().toString()) < 101 &&
            Integer.parseInt(subj2.getText().toString()) > 0 &&
            Integer.parseInt(subj2.getText().toString()) < 101 &&
            Integer.parseInt(subj3.getText().toString()) > 0 &&
            Integer.parseInt(subj3.getText().toString()) < 101 &&
            Integer.parseInt(certificate.getText().toString()) > 0 &&
            Integer.parseInt(certificate.getText().toString()) < 101)
        {
            int res = Integer.parseInt(subj1.getText().toString()) +
                Integer.parseInt(subj2.getText().toString()) +
                Integer.parseInt(subj3.getText().toString()) +
                Integer.parseInt(certificate.getText().toString());
            result.setText(Integer.toString(res));
        }
        else {
            showSnackBar("Вы ввели неверные значения. Диапазон: от 1 до 100.");
        }
    }
}

private void showSnackBar(String msg) {
    Snackbar snackbar = Snackbar
        .make(CLCalculation, msg, Snackbar.LENGTH_LONG);

```

```

        snackbar.show();
    }
}

```

CalendarFragment.java

```

package com.example.diplomaproject.businesslogic;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.net.Uri;
import android.os.Bundle;
import android.text.method.LinkMovementMethod;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.CalendarView;
import android.widget.TextView;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;
import com.example.diplomaproject.data.database.ConnectionHelper;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Objects;

public class CalendarFragment extends Fragment{

    Connection connection;
    CalendarView calendarView;
    String ConnectionResult;
    TextView eventTitle;
    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState){

        View root = inflater.inflate(R.layout.fragment_calendar, container, false);
        calendarView = root.findViewById(R.id.calendarView);
        eventTitle = root.findViewById(R.id.eventTitle);

        fillDates();

        calendarView.setOnDateChangeListener((view, year, month, dayOfMonth) -> {
            // display the selected date by using a toast
            fillCalendar(year + "-0" + (month + 1) + "-0" + dayOfMonth);
        });
    }
}

```

```

    });

    return root;
}

public void fillDates(){
    String date;
    ArrayList<String> dates = new ArrayList<>();
    try {
        ConnectionHelper connectionHelper = new ConnectionHelper();
        connection = connectionHelper.connection();
        if (connection != null) {
            String query = "SELECT EventDate FROM EVENT";
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(query);
            if(resultSet != null) {
                while (resultSet.next()) {
                    date = resultSet.getString("EventDate");
                    dates.add(date);
                }
                setDate(dates);
            }
        } else {
            ConnectionResult = "Check Connection";
        }
    } catch (Exception exception){
        ConnectionResult = exception.getMessage();
    }
}

@SuppressLint("SimpleDateFormat")
public void setDate(ArrayList<String> dates) throws ParseException {
    for (String date: dates) {
        calendarView.setDate(Objects.requireNonNull(new SimpleDateFormat("yyyy-MM-
dd")

                .parse(date)).getTime(), true, true);
    }
}

public void fillCalendar(String date){
    String title;
    String url;
    try {
        ConnectionHelper connectionHelper = new ConnectionHelper();
        connection = connectionHelper.connection();
        if (connection != null) {
            String query = "SELECT EventTitle, EventUrl FROM EVENT WHERE EventDate
= ""

                + date + """;
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(query);
            if(resultSet != null) {

```

```

        while (resultSet.next()) {
            title = resultSet.getString("EventTitle");
            url = resultSet.getString("EventUrl");
            setEvent(title, url);
        }
    } else {
        ConnectionResult = "Check Connection";
    }
} catch (Exception exception){
    ConnectionResult = exception.getMessage();
}
}

public void setEvent(String title, String url){
    eventTitle.setText(title);
    eventTitle.setOnClickListener(v -> {
        Intent browserIntent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
        startActivity(browserIntent);
    });
    eventTitle.setMovementMethod(LinkMovementMethod.getInstance());
}
}

```

ContactsFragment.java

```

package com.example.diplomaproject.businesslogic;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;

import com.example.diplomaproject.R;

public class ContactsFragment extends Fragment {

    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState){

        View root = inflater.inflate(R.layout.fragment_contacts, container, false);

        return root;
    }
}

```

FileDownloader.java

```
package com.example.diplomaproject.businesslogic;

import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
public class FileDownloader {
    private static final int MEGABYTE = 1024 * 1024;

    public static void downloadFile(String fileUrl, File directory){
        try {

            URL url = new URL(fileUrl);
            HttpURLConnection urlConnection = (HttpURLConnection)url.openConnection();
            //urlConnection.setRequestMethod("GET");
            //urlConnection.setDoOutput(true);
            urlConnection.connect();

            InputStream inputStream = urlConnection.getInputStream();
            FileOutputStream fileOutputStream = new FileOutputStream(directory);
            int totalSize = urlConnection.getContentLength();

            byte[] buffer = new byte[MEGABYTE];
            int bufferLength = 0;
            while((bufferLength = inputStream.read(buffer))>0 ){
                fileOutputStream.write(buffer, 0, bufferLength);
            }
            fileOutputStream.close();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

LoginActivity.java

```
package com.example.diplomaproject.businesslogic;

import android.app.Activity;
import android.content.Intent;
import android.os.Bundle;
import android.view.View;
```

```

import android.view.inputmethod.InputMethodManager;
import android.widget.EditText;

import androidx.appcompat.app.AppCompatActivity;
import androidx.constraintlayout.widget.ConstraintLayout;

import com.example.diplomaproject.R;
import com.example.diplomaproject.data.database.ConnectionHelper;
import com.example.diplomaproject.data.database.DatabaseAdapter;
import com.example.diplomaproject.businesslogic.account.CreateAccountActivity;
import com.example.diplomaproject.businesslogic.account.ViewAccountActivity;
import com.example.diplomaproject.validators.interfaces.ErrorCallBack;
import com.example.diplomaproject.validators.validators.Email;
import com.google.android.material.snackbar.Snackbar;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.HashMap;

public class LoginActivity extends AppCompatActivity implements ErrorCallBack {

    HashMap<String, Boolean> emailValidation;
    HashMap<String, Object> passwordValidation;
    String ConnectionResult = "";
    Connection connection;
    ConstraintLayout CLLogin;
    EditText emailEdit;
    EditText passEdit;
    DatabaseAdapter databaseAdapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_login);
        getSupportActionBar().setDisplayHomeAsUpEnabled(true);
        CLLogin = findViewById(R.id.LoginLayout);
        emailEdit = findViewById(R.id.editTextEmail);
        passEdit = findViewById(R.id.editTextPassword);
    }

    public void onEnterBtnClick(View view) throws SQLException {
        String email = emailEdit.getText().toString();
        String pass = passEdit.getText().toString();

        emailValidation = new Email.EmailBuilder(emailEdit.getText().toString())
            .setRequired(true)
            .build();

        Email emailField = new Email(LoginActivity.this);
    }

```

```

        if (emailField.isValid(emailValidation)) {
            /*
            boolean check = databaseAdapter.Authorization(email, pass);
            if(check){
                //Открыть личный кабинет в режиме просмотра
                Intent intent = new Intent(getBaseContext(), CreateAccountActivity.class);
                startActivity(intent);
            }
            else{
                Toast toast = Toast.makeText(this, "Проблема с подключением",
                Toast.LENGTH_SHORT);
                toast.setGravity(Gravity.CENTER, 0, 0);
                toast.show();
            }*/
            int checkExist = checkExisting(email);
            if (checkExist != 0) {
                try {
                    ConnectionHelper connectionHelper = new ConnectionHelper();
                    connection = connectionHelper.connection();
                    int id = 0;
                    if (connection != null) {
                        String query = "SELECT UserID FROM ENROLLEEUSER WHERE
                        Login=N'" + email +
                        "' AND Password=N'" + pass + "'";
                        Statement statement = connection.createStatement();
                        ResultSet resultSet = statement.executeQuery(query);
                        if (resultSet != null) {
                            while (resultSet.next()) {
                                id = resultSet.getInt("UserID");
                            }
                            Intent intent = new Intent(getBaseContext(), ViewAccountActivity.class);
                            intent.putExtra("userId", id);
                            startActivity(intent);
                        }
                    } else {
                        ConnectionResult = "Check Connection";
                    }

                } catch (Exception exception) {
                    ConnectionResult = exception.getMessage();
                }
            } else {
                showSnackBar("Пользователь не найден");
            }
        }
    }

    public void onRegisterBtnClick(View view) throws SQLException {
        String email = emailEdit.getText().toString();
        String pass = passEdit.getText().toString();
    }

```



```

databaseAdapter = new DatabaseAdapter();
emailValidation = new Email.EmailBuilder(emailEdit.getText().toString())
    .setRequired(true)
    .build();

Email emailField = new Email(LoginActivity.this);

if (emailField.isValid(emailValidation)) {
    if (pass.length() > 5 && pass.length() < 10) {
        int checkExist = checkExisting(email);
        if (checkExist == 0) {

            try {
                ConnectionHelper connectionHelper = new ConnectionHelper();
                connection = connectionHelper.connection();
                int id = 0;
                if (connection != null) {
                    String query = "INSERT INTO ENROLLEEUSER VALUES ('" +
                        email + "', N'" + pass + "')";
                    Statement statement = connection.createStatement();
                    statement.executeUpdate(query);

                    String selectQuery = "SELECT UserID FROM ENROLLEEUSER
WHERE Login=N'" + email +
                        "' AND Password=N'" + pass + "'";
                    ResultSet resultSet = statement.executeQuery(selectQuery);
                    if (resultSet != null) {
                        while (resultSet.next()) {
                            id = resultSet.getInt("UserID");
                        }
                        Intent intent = new Intent(getBaseContext(), CreateAccountActiv-
ity.class);

                        intent.putExtra("userId", id);
                        startActivity(intent);
                    }
                } else {
                    ConnectionResult = "Check Connection";
                }
            } catch (Exception exception) {
                ConnectionResult = exception.getMessage();
            }
        } else {
            showSnackBar("Пользователь с таким логином уже существует");
        }

    } else {
        showSnackBar("Длина пароля должна быть в диапазоне от 6 до 9 символов");
    }
}
}

```

```

private void showSnackBar(String msg) {
    Snackbar snackbar = Snackbar
        .make(CLLLogin, msg, Snackbar.LENGTH_LONG);
    snackbar.show();
}

private void hideKeyboard() {
    InputMethodManager inputMethodManager = (InputMethodManager) getSystemService(Activity.INPUT_METHOD_SERVICE);
    if (inputMethodManager != null) {
        inputMethodManager.hideSoftInputFromWindow(getWindow().getDecorView().getWindowToken(), 0);
    }
}

@Override
public void onValidationError(String s) {
    hideKeyboard();
    showSnackBar(s);
}

public int checkExisting(String login) {
    int res = 0;
    try {
        ConnectionHelper connectionHelper = new ConnectionHelper();
        connection = connectionHelper.connection();
        if (connection != null) {

            String selectQuery = "SELECT UserID FROM ENROLLEEUSER WHERE Login=N'" + login + "'";
            Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery(selectQuery);
            if (resultSet != null) {
                while (resultSet.next()) {
                    res++;
                }
            }
            } else {
                ConnectionResult = "Check Connection";
            }
        } catch (Exception exception) {
            ConnectionResult = exception.getMessage();
        }
        return res;
    }
}

```

MainPageFragment.java

```

package com.example.diplomaproject.businesslogic;

```

```

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ImageButton;
import android.widget.Toast;

import androidx.annotation.NonNull;
import androidx.fragment.app.Fragment;
import androidx.fragment.app.FragmentTransaction;

import com.example.diplomaproject.R;
import com.example.diplomaproject.businesslogic.web.MyRunTimesData;
import com.example.diplomaproject.businesslogic.web.WebFragment;

public class MainPageFragment extends Fragment implements View.OnClickListener {

    ImageButton speciality;
    ImageButton campaign;
    ImageButton regulations;
    ImageButton sokr;
    ImageButton excursia;
    ImageButton admission;
    View root;
    public View onCreateView(@NonNull LayoutInflater inflater,
                             ViewGroup container, Bundle savedInstanceState) {

        root = inflater.inflate(R.layout.fragment_main_page, container, false);

        speciality = root.findViewById(R.id.speciality);
        campaign = root.findViewById(R.id.campaign);
        regulations = root.findViewById(R.id.regulations);
        sokr = root.findViewById(R.id.sokr);
        excursia = root.findViewById(R.id.excursia);
        admission = root.findViewById(R.id.admission);

        speciality.setOnClickListener(this);
        campaign.setOnClickListener(this);
        regulations.setOnClickListener(this);
        sokr.setOnClickListener(this);
        excursia.setOnClickListener(this);
        admission.setOnClickListener(this);
        return root;
    }

    @SuppressWarnings("NonConstantResourceId")
    @Override
    public void onClick(View v) {

```

```

switch (v.getId()){
    case R.id.speciality:
        try {
            MyRunTimesData.setMyStringData("https://abiturient.gstu.by/specialties");
            Fragment fragment = new WebFragment();
            FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

            transaction.replace(R.id.fragment_main_page, fragment);
            transaction.addToBackStack(null);
            transaction.commit();
        }
        catch (Exception e) {
            Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                Toast.LENGTH_SHORT).show();
        }
        break;

    case R.id.campaign:
        try {
            MyRunTimesData.setMyStringData("https://abiturient.gstu.by/course-of-documents-acceptance");
            Fragment fragment = new WebFragment();
            FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

            transaction.replace(R.id.fragment_main_page, fragment);
            transaction.addToBackStack(null);
            transaction.commit();
        }
        catch (Exception e) {
            Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                Toast.LENGTH_SHORT).show();
        }
        break;

    case R.id.regulations:
        try {
            MyRunTimesData.setMyStringData("https://abiturient.gstu.by/regulations");
            Fragment fragment = new WebFragment();
            FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

            transaction.replace(R.id.fragment_main_page, fragment);
            transaction.addToBackStack(null);
            transaction.commit();
        }
        catch (Exception e) {
            Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                Toast.LENGTH_SHORT).show();
        }
        break;
}

```

```

        case R.id.sokr:
            try {
                MyRunTimesData.setMyStringData("https://abiturient.gstu.by/higher-education-in-reduced-terms-of-training");
                Fragment fragment = new WebFragment();
                FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

                transaction.replace(R.id.fragment_main_page, fragment);
                transaction.addToBackStack(null);
                transaction.commit();
            }
            catch (Exception e) {
                Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                    Toast.LENGTH_SHORT).show();
            }
            break;

        case R.id.excursia:
            try {
                MyRunTimesData.setMyStringData("https://abiturient.gstu.by/excursion");
                Fragment fragment = new WebFragment();
                FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

                transaction.replace(R.id.fragment_main_page, fragment);
                transaction.addToBackStack(null);
                transaction.commit();
            }
            catch (Exception e) {
                Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                    Toast.LENGTH_SHORT).show();
            }
            break;

        case R.id.admission:
            try {
                MyRunTimesData.setMyStringData("https://abiturient.gstu.by/how-to-enter-the-GSTU-PO-Sukhoi");
                Fragment fragment = new WebFragment();
                FragmentTransaction transaction = getChildFragmentManager().beginTransaction();

                transaction.replace(R.id.fragment_main_page, fragment);
                transaction.addToBackStack(null);
                transaction.commit();
            }
            catch (Exception e) {
                Toast.makeText(getActivity().getApplicationContext(), "Can't connect to Internet",
                    Toast.LENGTH_SHORT).show();
            }
    }

```

```

        break;
    }
}
}

```

ConnectionHelper.java

```

package com.example.diplomaproject.data.database;

import android.annotation.SuppressLint;
import android.os.StrictMode;
import android.util.Log;

import java.sql.Connection;
import java.sql.DriverManager;

public class ConnectionHelper {

    String username, pass, host, port, database;
    @SuppressWarnings("NewApi")
    public Connection connection(){
        host = "gstuenrollee.database.windows.net";
        database = "EnrolleeDB";
        username = "enrolleeadmin";
        pass = "12345678Mm";
        port = "1433";

        StrictMode.ThreadPolicy policy = new StrictMode.ThreadPolicy.Builder().permitAll().build();
        StrictMode.setThreadPolicy(policy);
        Connection connection = null;
        String ConnectionURL = null;

        try{
            Class.forName("net.sourceforge.jtds.jdbc.Driver");
            ConnectionURL= "jdbc:jtds:sqlserver://" + host + ":" + port + ";" + "databasename="+
                database + ";user="+username + ";password="+pass + ";";
            connection = DriverManager.getConnection(ConnectionURL);
        }
        catch (Exception exception){
            Log.e("Error", exception.getMessage());
        }
        return connection;
    }
}

```

ErrorCallBack.java

```

package com.example.diplomaproject.validators.interfaces;

public interface ErrorCallBack {

```

```

    void onValidationError(String s);
}

```

Email.java

```

package com.example.diplomaproject.validators.validators;

import android.provider.ContactsContract;
import android.util.Patterns;

import com.example.diplomaproject.validators.interfaces.ErrorCallBack;

import org.jetbrains.annotations.NotNull;

import java.util.HashMap;
import java.util.regex.Pattern;

public class Email {
    public static String TAG = ContactsContract.CommonDataKinds.Email.class.getSimpleName();
    private String value = ""; //This is important, so we'll pass it to the constructor.
    private boolean isRequired = false;
    //public static String emailPattern = "[a-zA-Z0-9._-]+@[a-z]+\\.+[a-z]+";

    //Static Response Code.
    public static String SUCCESS = "Success";
    public static String IS_EMAIL = "IsEmail";
    public static String IS_REQUIRED = "IsRequired";
    public static String EMPTY = "Empty";
    public HashMap<String, Boolean> emailValidationResponse;
    private ErrorCallBack errorCallBack;

    public String getValue() {
        return value;
    }

    public boolean isRequired() {
        return isRequired;
    }

    public static class EmailBuilder {
        private String value; //This is important, so we'll pass it to the constructor.
        private boolean isRequired = false;
        public HashMap<String, Boolean> emailValidationResponse;

        public EmailBuilder(String value) {
            this.value = value;
        }

        public EmailBuilder setRequired(boolean isRequired) {

```

```

        this.isRequired = isRequired;
        return this;
    }

    public HashMap<String, Boolean> build() {
        boolean success;
        emailValidationResponse = new HashMap<>();
        if (this.isRequired) {
            emailValidationResponse.put(IS_REQUIRED, true);
        } else {
            emailValidationResponse.put(IS_REQUIRED, false);
        }
        if (this.value != null && !this.value.isEmpty()) {
            emailValidationResponse.put(EMPTY, false);
            Pattern pattern = Patterns.EMAIL_ADDRESS;
            if (pattern.matcher(this.value).matches()) {
                success = true;
                emailValidationResponse.put(IS_EMAIL, true);
            } else {
                success = false;
                emailValidationResponse.put(IS_EMAIL, false);
            }
        } else {
            success = false;
            emailValidationResponse.put(EMPTY, true);
            emailValidationResponse.put(IS_EMAIL, false);
        }
        emailValidationResponse.put(SUCCESS, success);
        return emailValidationResponse;
    }
}

public boolean isValid(@NotNull HashMap<String, Boolean> emailValidation) {
    if (emailValidation.get(Email.SUCCESS)) {
        return true;
    } else {
        if (emailValidation.get(Email.IS_REQUIRED)) {
            if (!emailValidation.get(Email.EMPTY)) {
                if (!emailValidation.get(Email.IS_EMAIL)) {
                    errorCallback.onValidationError("Неверный Email");
                    return false;
                }
            } else {
                errorCallback.onValidationError("Поле email не должно быть пустым");
                return false;
            }
        } else {
            if (!emailValidation.get(Email.EMPTY)) {
                if (!emailValidation.get(Email.IS_EMAIL)) {
                    errorCallback.onValidationError("Неверный Email");

```



```

        return false;
    }
}
}
return true;
}

public Email(ErrorCallBack c) {
    this.errorCallBack = c;
}

private Email(EmailBuilder emailBuilder) {
    this.value = emailBuilder.value;
    this.isRequired = emailBuilder.isRequired;
    this.emailValidationResponse = emailBuilder.emailValidationResponse;
}
}

```

PhoneNumber.java

```

package com.example.diplomaproject.validators.validators;

import android.widget.EditText;

import com.example.diplomaproject.validators.interfaces.ErrorCallBack;

import java.util.HashMap;

public class PhoneNumber {

    public static String TAG = PhoneNumber.class.getSimpleName();
    private String value; //This is important, so we'll pass it to the constructor.
    private boolean isRequired = false;
    private int maxLength = 12;
    private EditText mEditText;
    private boolean checkInputType = false;

    //Static Response Code.
    public static String SUCCESS = "Success";
    public static String MAX_LENGTH = "MaxLength";
    public static String MIN_LENGTH = "MinLength";
    public static String IS_REQUIRED = "IsRequired";
    public static String EMPTY = "Empty";

    private ErrorCallBack errorCallBack;

    public String getValue() {
        return value;
    }
}

```

```

public boolean isRequired() {
    return isRequired;
}

public int getMaxLength() {
    return maxLength;
}

public EditText getEditText() {
    return mEditText;
}

public Boolean checkInputType() {
    return checkInputType;
}

public static class PhoneNumberBuilder {
    private String value; //This is important, so we'll pass it to the constructor.
    private boolean isRequired = false;
    private int maxLength = 10;
    private int minLength = 10;
    private EditText mEditText;
    private boolean checkInputType = false;

    public PhoneNumberBuilder(String value) {
        this.value = value;
    }

    public PhoneNumberBuilder setRequired(boolean isRequired) {
        this.isRequired = isRequired;
        return this;
    }

    public PhoneNumberBuilder setMaxLength(int maxLength) {
        this.maxLength = maxLength;
        return this;
    }

    public PhoneNumberBuilder setMinLength(int minLength) {
        this.minLength = minLength;
        return this;
    }

    public PhoneNumberBuilder setEditText(EditText mEditText) {
        this.mEditText = mEditText;
        return this;
    }

    public PhoneNumberBuilder checkInputType(boolean checkInputType) {
        this.checkInputType = checkInputType;
        return this;
    }
}

```

```

    }

    public HashMap<String, Boolean> build() {
        HashMap<String, Boolean> phoneNumberValidatorResp = new HashMap<String,
Boolean>();

        if (this.isRequired) {
            phoneNumberValidatorResp.put(IS_REQUIRED, true);
        } else {
            phoneNumberValidatorResp.put(IS_REQUIRED, false);
        }
        if (this.value != null && !this.value.isEmpty()) {
            phoneNumberValidatorResp.put(EMPTY, false);
            if (this.value.length() >= minLength && this.value.length() <= maxLength) {
                phoneNumberValidatorResp.put(SUCCESS, true);
                phoneNumberValidatorResp.put(MAX_LENGTH, true);
                phoneNumberValidatorResp.put(MIN_LENGTH, true);
            } else {
                phoneNumberValidatorResp.put(SUCCESS, false);
                phoneNumberValidatorResp.put(MAX_LENGTH, false);
                phoneNumberValidatorResp.put(MIN_LENGTH, false);
            }
        } else {
            phoneNumberValidatorResp.put(SUCCESS, false);
            phoneNumberValidatorResp.put(EMPTY, true);
            phoneNumberValidatorResp.put(MAX_LENGTH, false);
            phoneNumberValidatorResp.put(MIN_LENGTH, false);
        }
        return phoneNumberValidatorResp;
    }
}

public PhoneNumber(ErrorCallBack back) {
    this.errorCallBack = back;
}

public boolean isValid(HashMap<String, Boolean> hashMap) {

    if (hashMap.get(PhoneNumber.SUCCESS)) {
        return true;
    } else {
        if (hashMap.get(PhoneNumber.IS_REQUIRED)) {
            if (!hashMap.get(PhoneNumber.EMPTY)) {
                if (!hashMap.get(PhoneNumber.MIN_LENGTH) && !hashMap.get(Pho-
neNumber.MAX_LENGTH)) {
                    errorCallBack.onValidationError("Неверный номер телефона");
                    return false;
                }
            } else {
                errorCallBack.onValidationError("Номер телефона не может быть пустым.");
                return false;
            }
        }
    }
}

```

```

        }
    } else {
        if (!hashMap.get(PhoneNumber.EMPTY)) {
            if (!hashMap.get(PhoneNumber.MIN_LENGTH) && !hashMap.get(PhoneNumber.MAX_LENGTH)) {
                errorCallback.onValidationError("Неверный номер телефона");
                return false;
            }
        }
    }
}
return true;
}

private PhoneNumber(PhoneNumberBuilder phoneNumberBuilder) {
    this.value = phoneNumberBuilder.value;
    this.isRequired = phoneNumberBuilder.isRequired;
    this.maxLength = phoneNumberBuilder.maxLength;
    this.mEditText = phoneNumberBuilder.mEditText;
    this.checkInputType = phoneNumberBuilder.checkInputType;
}
}

```

MainActivity.java

```

package com.example.diplomaproject;

import android.content.Intent;
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuItem;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;
import androidx.appcompat.widget.Toolbar;
import androidx.drawerlayout.widget.DrawerLayout;
import androidx.navigation.NavController;
import androidx.navigation.Navigation;
import androidx.navigation.ui.AppBarConfiguration;
import androidx.navigation.ui.NavigationUI;

import com.example.diplomaproject.businesslogic.LoginActivity;
import com.google.android.material.navigation.NavigationView;

public class MainActivity extends AppCompatActivity {

    private AppBarConfiguration mAppBarConfiguration;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
}

```

```

        Toolbar toolbar = findViewById(R.id.toolbar);
        setSupportActionBar(toolbar);

        DrawerLayout drawer = findViewById(R.id.drawer_layout);
        NavigationView navigationView = findViewById(R.id.nav_view);
        mAppBarConfiguration = new AppBarConfiguration.Builder(R.id.nav_main_page,
            R.id.nav_site, R.id.nav_map, R.id.nav_calendar, R.id.nav_calc,
            R.id.nav_contacts)
            .setDrawerLayout(drawer)
            .build();
        NavController navController = Navigation.findNavController(this, R.id.nav_host_frag-
ment);
        NavigationUI.setupActionBarWithNavController(this, navController, mAppBarConfig-
uration);
        NavigationUI.setupWithNavController(navigationView, navController);
    }

    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate the menu; this adds items to the action bar if it is present.
        getMenuInflater().inflate(R.menu.main, menu);
        return true;
    }

    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        int id = item.getItemId();
        TextView headerView = (TextView) findViewById(R.id.selectedMenuItem);
        switch(id){
            case R.id.account :
                Intent intent = new Intent(getBaseContext(), LoginActivity.class);
                startActivity(intent);
                headerView.setText("Настройки");
                return true;
            }
        //headerView.setText(item.getTitle());
        return super.onOptionsItemSelected(item);
    }

    @Override
    public boolean onSupportNavigateUp() {
        NavController navController = Navigation.findNavController(this, R.id.nav_host_frag-
ment);
        return NavigationUI.navigateUp(navController, mAppBarConfiguration)
            || super.onSupportNavigateUp();
    }
}

```

activity_create_account.xml

```

<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/scroll"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        android:id="@+id/CLCreate"
        tools:context=".businesslogic.account.CreateAccountActivity">
        <TextView
            android:id="@+id/MainLabel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="20dp"
            android:ems="10"
            android:text="Введите ваши данные"
            android:textColor="@color/purple_200"
            android:inputType="textMultiLine"

            android:textSize="20sp"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintEnd_toEndOf="parent"
            app:layout_constraintTop_toTopOf="parent" />

        <TextView
            android:id="@+id/SurnameLabel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginStart="30dp"
            android:layout_marginTop="24dp"
            android:ems="10"
            android:inputType="textMultiLine"
            android:text="Фамилия"
            android:textColor="@color/purple_200"

            android:textSize="16sp"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toBottomOf="@+id/MainLabel" />

        <EditText
            android:id="@+id/Surname"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="15dp"
            android:ems="10"
            android:hint="Фамилия"
            android:inputType="text"

```

```

        android:textSize="16sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@id/MainLabel" />
<TextView
    android:id="@+id/NameLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Имя"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/SurnameLabel" />

<EditText
    android:id="@+id/Name"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:ems="10"
    android:hint="Имя"
    android:inputType="text"
    app:layout_constraintEnd_toEndOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Surname" />

<TextView
    android:id="@+id/PatronymicLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Отчество"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/NameLabel" />

<EditText
    android:id="@+id/Patronymic"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:ems="10"

```

```

        android:hint="Отчество"
        android:inputType="text"
        app:layout_constraintEnd_toEndOf="parent"
        android:textSize="16sp"
        app:layout_constraintTop_toBottomOf="@+id/Name" />

<TextView
    android:id="@+id/PassportLabel"
    android:layout_width="133dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Серия и номер паспорта"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/PatronymicLabel" />

<EditText
    android:id="@+id/Passport"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:ems="10"
    android:hint="Серия и номер паспорта"
    android:inputType="textMultiLine"
    app:layout_constraintEnd_toEndOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Patronymic" />

<TextView
    android:id="@+id/AddressLabel"
    android:layout_width="133dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Адрес"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/PassportLabel" />

<EditText
    android:id="@+id/Address"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"

```



```

        android:layout_marginTop="5dp"
        android:ems="10"
        android:hint="Адрес"
        android:inputType="textMultiLine"
        app:layout_constraintEnd_toEndOf="parent"
        android:textSize="16sp"
        app:layout_constraintTop_toBottomOf="@+id/Passport" />

<TextView
    android:id="@+id/DateOfBirthLabel"
    android:layout_width="133dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Дата рождения"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Address" />

<TextView
    android:id="@+id/DateOfBirth"
    android:layout_width="187dp"
    android:layout_height="36dp"
    android:layout_marginStart="30dp"
    android:layout_marginTop="32dp"
    android:ems="10"
    android:hint="Дата рождения"
    android:inputType="date"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/DateOfBirthLabel"
    tools:ignore="MissingConstraints" />

<Button
    android:id="@+id/dateBtn"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginRight="50dp"
    android:layout_marginTop="20dp"
    android:onClick="onSetDateBtnClick"
    android:text="Изменить"
    app:layout_constraintRight_toRightOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/DateOfBirthLabel" />

<TextView
    android:id="@+id/PhoneNumberLabel"
    android:layout_width="133dp"
    android:layout_height="wrap_content"

```

```

android:layout_marginStart="30dp"
android:layout_marginTop="26dp"
android:ems="10"
android:inputType="textMultiLine"
android:text="Номер телефона"
android:textColor="@color/purple_200"

android:textSize="16sp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/DateOfBirth" />

```

```

<EditText
    android:id="@+id/PhoneNumber"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Номер телефона"
    android:inputType="phone"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/dateBtn"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/FormTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="20dp"
    android:text="Выберите форму обучения"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/PhoneNumber"
    tools:ignore="MissingConstraints" />

```

```

<Spinner
    android:id="@+id/EducationForm"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="5dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Форма обучения"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/PhoneNumber"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/IsAimedTextView"
    android:layout_width="204dp"

```

```

android:layout_height="wrap_content"
android:layout_marginStart="30dp"
android:layout_marginTop="20dp"
android:text="Участие в конкурсе по целевому направлению"
android:textColor="@color/purple_200"
android:textSize="16sp"
app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/EducationForm"
tools:ignore="MissingConstraints" />

```

```

<CheckBox
    android:id="@+id/IsAimed"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="28dp"
    android:layout_marginTop="27dp"
    android:layout_marginEnd="50dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsAimedTextView"
    app:layout_constraintTop_toBottomOf="@+id/EducationForm"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/IsWithoutExamsTextView"
    android:layout_width="208dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="15dp"
    android:text="Участие в конкурсе без вступительных экзаменов"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/IsAimedTextView"
    tools:ignore="MissingConstraints" />

```

```

<CheckBox
    android:id="@+id/IsWithoutExams"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="28dp"
    android:layout_marginTop="35dp"
    android:layout_marginEnd="50dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsWithoutExamsTextView"
    app:layout_constraintTop_toBottomOf="@+id/IsAimed"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/IsBudgetsTextView"
    android:layout_width="210dp"

```

```

android:layout_height="wrap_content"
android:layout_marginStart="30dp"
android:layout_marginTop="15dp"
android:text="Участие в конкурсе на бюджет"
android:textColor="@color/purple_200"
android:textSize="16sp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/IsWithoutExamsTextView"
tools:ignore="MissingConstraints" />

```

```

<CheckBox
    android:id="@+id/IsBudget"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="148dp"
    android:layout_marginTop="35dp"
    android:layout_marginEnd="50dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsBudgetsTextView"
    app:layout_constraintTop_toBottomOf="@+id/IsWithoutExams"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/IsOutOfCompetitionsTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="15dp"
    android:text="Участие вне конкурса"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/IsBudgetsTextView"
    tools:ignore="MissingConstraints" />

```

```

<CheckBox
    android:id="@+id/IsOutOfCompetitions"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="216dp"
    android:layout_marginTop="25dp"
    android:layout_marginEnd="50dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsOutOfCompetitionsTextView"
    app:layout_constraintTop_toBottomOf="@+id/IsBudget"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/ProfSubj1EELabel"
    android:layout_width="wrap_content"

```

```

        android:layout_height="wrap_content"
        android:layout_marginStart="30dp"
        android:layout_marginTop="26dp"
        android:ems="10"
        android:inputType="textMultiLine"
        android:text="Балл по первому вступительному испытанию"
        android:textColor="@color/purple_200"

        android:textSize="16sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/IsOutOfCompetitionsTextView" />

<EditText
    android:id="@+id/ProfSubj1EE"
    android:layout_width="70dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="30dp"
    android:layout_marginEnd="40dp"

    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/IsOutOfCompetitions" />

<TextView
    android:id="@+id/ProfSubj2EELabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Балл по второму вступительному испытанию"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj1EELabel" />

<EditText
    android:id="@+id/ProfSubj2EE"
    android:layout_width="70dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="45dp"
    android:layout_marginEnd="45dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj1EE" />

<TextView
    android:id="@+id/Subj3EELabel"
    android:layout_width="wrap_content"

```

```

        android:layout_height="wrap_content"
        android:layout_marginStart="30dp"
        android:layout_marginTop="26dp"
        android:ems="10"
        android:inputType="textMultiLine"
        android:text="Балл по третьему вступительному испытанию"
        android:textColor="@color/purple_200"

        android:textSize="16sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/ProfSubj2EELabel" />

<EditText
    android:id="@+id/Subj3EE"
    android:layout_width="70dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="45dp"
    android:layout_marginEnd="40dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj2EE" />

<TextView
    android:id="@+id/CertificateScoreX10Label"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="26dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Средний балл документа об образовании"
    android:textColor="@color/purple_200"

    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Subj3EELabel" />
<EditText
    android:id="@+id/CertificateScoreX10"
    android:layout_width="70dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="45dp"
    android:layout_marginEnd="40dp"
    android:ems="10"
    android:inputType="number"
    app:layout_constraintEnd_toEndOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Subj3EE" />

<TextView
    android:id="@+id/IsAgreeForPaymentTextView"

```

```

android:layout_width="186dp"
android:layout_height="wrap_content"
android:layout_marginStart="30dp"

android:layout_marginTop="20dp"
android:inputType="textMultiLine"
android:text="Согласие на участие в конкурсе на платную форму обучения"
android:textColor="@color/purple_200"
android:textSize="16sp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/CertificateScoreX10"
tools:ignore="MissingConstraints" />

```

<CheckBox

```

android:id="@+id/IsAgreeForPayment"
android:layout_width="30dp"
android:layout_height="21dp"
android:layout_marginLeft="76dp"
android:layout_marginTop="25dp"
android:layout_marginEnd="50dp"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintLeft_toRightOf="@+id/IsAgreeForPaymentTextView"
app:layout_constraintTop_toBottomOf="@+id/CertificateScoreX10"
tools:ignore="MissingConstraints" />

```

<Button

```

android:id="@+id/CreateButton"
style="@style/Widget.AppCompat.Button.Colored"
android:layout_width="wrap_content"
android:layout_height="wrap_content"
android:layout_marginTop="20dp"
android:layout_marginEnd="20dp"
android:enabled="true"
android:onClick="onCreateModelButtonClick"
android:text="Добавить данные"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintTop_toBottomOf="@+id/IsAgreeForPaymentTextView" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>

```

activity_login.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"

```

```
android:layout_height="match_parent"
tools:context=".businesslogic.LoginActivity"
android:id="@+id/LoginLayout">
```

```
<Button
    android:id="@+id/enterBtn"
    android:layout_width="264dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="75dp"
    android:layout_marginTop="40dp"
    android:layout_marginEnd="75dp"
    android:layout_marginBottom="10dp"
    android:onClick="onEnterBtnClick"
    android:text="Войти"
    app:layout_constraintBottom_toTopOf="@+id/registerBtn"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/editTextPassword" />
```

```
<Button
    android:id="@+id/registerBtn"
    android:layout_width="264dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="75dp"
    android:layout_marginTop="20dp"
    android:layout_marginEnd="75dp"
    android:layout_marginBottom="10dp"
    android:onClick="onRegisterBtnClick"
    android:text="Зарегистрироваться"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/enterBtn" />
```

```
<EditText
    android:id="@+id/editTextEmail"
    android:layout_width="295dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="58dp"
    android:layout_marginTop="90dp"
    android:layout_marginEnd="58dp"
    android:layout_marginBottom="20dp"
    android:ems="10"
    android:hint="Email"
    android:inputType="textEmailAddress"
    app:layout_constraintBottom_toTopOf="@+id/editTextPassword"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/imageView2" />
```

```
<EditText
```



```

        android:id="@+id/editTextPassword"
        android:layout_width="295dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="58dp"
        android:layout_marginTop="20dp"
        android:layout_marginEnd="58dp"
        android:layout_marginBottom="5dp"
        android:ems="10"
        android:hint="Пароль"
        android:inputType="textPassword"
        app:layout_constraintBottom_toTopOf="@+id/enterBtn"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"

        app:layout_constraintTop_toBottomOf="@+id/editTextEmail" />

```

```

<ImageView
    android:id="@+id/imageView2"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"

```

```

    app:layout_constraintTop_toTopOf="parent"
    app:srcCompat="@drawable/unnamed" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>

```

activity_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.drawerlayout.widget.DrawerLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/drawer_layout"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:fitsSystemWindows="true"
    tools:openDrawer="start">
    <TextView
        android:id="@+id/selectedMenuItem"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="28sp"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <include
        layout="@layout/app_bar_main"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

```

```

        <com.google.android.material.navigation.NavigationView
            android:id="@+id/nav_view"
            android:layout_width="wrap_content"
            android:layout_height="match_parent"
            android:layout_gravity="start"
            android:fitsSystemWindows="true"
            app:headerLayout="@layout/nav_header_main"
            app:menu="@menu/activity_main_drawer" />
    </androidx.drawerlayout.widget.DrawerLayout>

```

activity_maps.xml

```

<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".businesslogic.map.MapsActivity" />

```

activity_view_account.xml

```

<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/scroll"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <androidx.constraintlayout.widget.ConstraintLayout
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:orientation="vertical"
        tools:context=".businesslogic.account.ViewAccountActivity">

        <TextView
            android:id="@+id/SurnameLabel"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_marginTop="5dp"
            android:layout_marginStart="30dp"
            android:ems="10"
            android:textColor="@color/purple_200"
            android:text="Фамилия"
            android:inputType="textMultiLine"
            android:textSize="16sp"
            app:layout_constraintStart_toStartOf="parent"
            app:layout_constraintTop_toTopOf="parent" />
    </androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>

```

```

<TextView
    android:id="@+id/Surname"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="5dp"
    android:ems="10"
    android:hint="Фамилия"
    android:inputType="textMultiLine"

    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

```

```

<TextView
    android:id="@+id/NameLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:layout_marginStart="30dp"
    android:ems="10"
    android:text="Имя"
    android:inputType="textMultiLine"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/SurnameLabel" />

```

```

<TextView
    android:id="@+id/Name"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Имя"
    android:inputType="textMultiLine"

    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Surname" />

```

```

<TextView
    android:id="@+id/PatronymicLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:layout_marginStart="30dp"
    android:text="Отчество"
    android:inputType="textMultiLine"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    android:textSize="16sp"

```

```

app:layout_constraintTop_toBottomOf="@+id/NameLabel" />

<TextView
    android:id="@+id/Patronymic"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Отчество"
    android:inputType="textMultiLine"
    app:layout_constraintEnd_toEndOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Name" />

<TextView
    android:id="@+id/PassportLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="8"
    android:text="Серия и номер паспорта"
    android:inputType="textMultiLine"
    android:layout_marginStart="30dp"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/PatronymicLabel" />

<TextView
    android:id="@+id/Passport"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Серия и номер паспорта"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Patronymic" />

<TextView
    android:id="@+id/AddressLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    app:layout_constraintStart_toStartOf="parent"
    android:text="Адрес"
    android:inputType="textMultiLine"
    android:textColor="@color/purple_200"
    android:layout_marginStart="30dp"
    android:textSize="16sp"

```

```

app:layout_constraintTop_toBottomOf="@+id/PassportLabel" />

<TextView
    android:id="@+id/Address"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="36dp"
    android:ems="10"
    android:hint="Адрес"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Passport" />

<TextView
    android:id="@+id/DateOfBirthLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:text="Дата рождения"
    android:layout_marginStart="30dp"
    app:layout_constraintStart_toStartOf="parent"
    android:inputType="date"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Address" />

<TextView
    android:id="@+id/DateOfBirth"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Дата рождения"
    app:layout_constraintEnd_toEndOf="parent"
    android:inputType="date"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/Address" />

<TextView
    android:id="@+id/PhoneNumberLabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:text="Номер телефона"
    app:layout_constraintStart_toStartOf="parent"
    android:layout_marginStart="30dp"
    android:inputType="phone"
    android:textColor="@color/purple_200"
    android:textSize="16sp"

```

```
app:layout_constraintTop_toBottomOf="@+id/DateOfBirthLabel"
tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/PhoneNumber"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:hint="Номер телефона"
    app:layout_constraintEnd_toEndOf="parent"
    android:inputType="phone"
    android:textSize="16sp"
    app:layout_constraintTop_toBottomOf="@+id/DateOfBirth"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/FormTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="20dp"
    android:text="Форма обучения"
    android:textSize="16sp"
    android:ems="10"
    android:layout_marginStart="30dp"
    app:layout_constraintStart_toStartOf="parent"
    android:textColor="@color/purple_200"
    app:layout_constraintTop_toBottomOf="@+id/PhoneNumberLabel"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/EducationForm"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="168dp"
    android:layout_marginTop="20dp"
    android:textSize="16sp"
    android:hint="Форма обучения"
    app:layout_constraintEnd_toEndOf="parent"
    android:ems="10"
    app:layout_constraintTop_toBottomOf="@+id/PhoneNumber"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/IsAimedTextView"
    android:layout_width="218dp"
    android:layout_height="wrap_content"
    android:layout_marginTop="25dp"
    android:text="Участие в конкурсе по целевому направлению"
    android:textSize="16sp"
    android:layout_marginStart="30dp"
    app:layout_constraintStart_toStartOf="parent"
```

```
android:textColor="@color/purple_200"  
app:layout_constraintTop_toBottomOf="@+id/FormTextView"  
tools:ignore="MissingConstraints" />
```

```
<CheckBox  
    android:id="@+id/IsAimed"  
    android:layout_width="30dp"  
    android:layout_height="21dp"  
    android:layout_marginLeft="28dp"  
    android:layout_marginTop="32dp"  
    android:layout_marginEnd="40dp"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintLeft_toRightOf="@+id/IsAimedTextView"  
    app:layout_constraintTop_toBottomOf="@+id/EducationForm"  
    tools:ignore="MissingConstraints" />
```

```
<TextView  
    android:id="@+id/IsWithoutExamsTextView"  
    android:layout_width="238dp"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="30dp"  
    android:layout_marginTop="20dp"  
    android:text="Участие в конкурсе без вступительных экзаменов"  
    android:textSize="16sp"  
    android:textColor="@color/purple_200"  
    app:layout_constraintStart_toStartOf="parent"  
    app:layout_constraintTop_toBottomOf="@+id/IsAimedTextView"  
    tools:ignore="MissingConstraints" />
```

```
<CheckBox  
    android:id="@+id/IsWithoutExams"  
    android:layout_width="30dp"  
    android:layout_height="21dp"  
    android:layout_marginLeft="28dp"  
    android:layout_marginTop="42dp"  
    android:layout_marginEnd="40dp"  
    app:layout_constraintEnd_toEndOf="parent"  
    app:layout_constraintLeft_toRightOf="@+id/IsWithoutExamsTextView"  
    app:layout_constraintTop_toBottomOf="@+id/IsAimed"  
    tools:ignore="MissingConstraints" />
```

```
<TextView  
    android:id="@+id/IsBudgetsTextView"  
    android:layout_width="wrap_content"  
    android:layout_height="wrap_content"  
    android:layout_marginStart="30dp"  
    android:layout_marginTop="20dp"  
    android:text="Участие в конкурсе на бюджет"  
    android:textSize="16sp"  
    android:textColor="@color/purple_200"  
    app:layout_constraintStart_toStartOf="parent"
```

```
app:layout_constraintTop_toBottomOf="@+id/IsWithoutExamsTextView"
tools:ignore="MissingConstraints" />
```

```
<CheckBox
    android:id="@+id/IsBudget"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="148dp"
    android:layout_marginTop="35dp"
    android:layout_marginEnd="40dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsBudgetsTextView"
    app:layout_constraintTop_toBottomOf="@+id/IsWithoutExams"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/IsOutOfCompetitionsTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="20dp"
    android:text="Участие вне конкурса"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/IsBudgetsTextView"
    tools:ignore="MissingConstraints" />
```

```
<CheckBox
    android:id="@+id/IsOutOfCompetitions"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="216dp"
    android:layout_marginTop="22dp"
    android:layout_marginEnd="40dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsOutOfCompetitionsTextView"
    app:layout_constraintTop_toBottomOf="@+id/IsBudget"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/ProfSubj1EELabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="25dp"
    android:ems="10"
    android:layout_marginStart="30dp"
    android:text="Балл по первому профильному предмету"
    android:textColor="@color/purple_200"
    android:inputType="textMultiLine"
```



```

        android:textSize="16sp"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/IsOutOfCompetitionsTextView" />

<TextView
    android:id="@+id/ProfSubj1EE"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="30dp"
    android:layout_marginEnd="50dp"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/IsOutOfCompetitions" />

<TextView
    android:id="@+id/ProfSubj2EELabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Балл по второму профильному предмету"
    android:textColor="@color/purple_200"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj1EELabel" />

<TextView
    android:id="@+id/ProfSubj2EE"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:layout_marginEnd="50dp"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/ProfSubj1EE"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj1EE" />

<TextView
    android:id="@+id/Subj3EELabel"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Балл по третьему предмету"
    android:textColor="@color/purple_200"
    android:textSize="16sp"

```

```

app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/ProfSubj2EELabel" />

<TextView
    android:id="@+id/Subj3EE"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"

    android:layout_marginEnd="50dp"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/ProfSubj2EE" />

<TextView
    android:id="@+id/CertificateScoreX10Label"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginLeft="30dp"
    android:layout_marginTop="20dp"
    android:ems="10"
    android:inputType="textMultiLine"
    android:text="Средний балл документа об образовании * 10"
    android:textSize="16sp"
    app:layout_constraintStart_toStartOf="parent"
    android:textColor="@color/purple_200"
    app:layout_constraintTop_toBottomOf="@+id/Subj3EELabel"
    tools:ignore="MissingConstraints" />

<TextView
    android:id="@+id/CertificateScoreX10"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginTop="40dp"
    android:layout_marginEnd="50dp"
    android:inputType="textMultiLine"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/Subj3EE" />

<TextView
    android:id="@+id/IsAgreeForPaymentTextView"
    android:layout_width="249dp"
    android:layout_height="wrap_content"
    android:layout_marginStart="30dp"
    android:layout_marginTop="20dp"
    android:inputType="textMultiLine"
    android:text="Согласие на участие в конкурсе на платную форму обучения"
    android:textSize="16sp"
    android:layout_marginBottom="30dp"
    android:textColor="@color/purple_200"

```

```

app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/CertificateScoreX10Label"
tools:ignore="MissingConstraints" />

```

```

<CheckBox
    android:id="@+id/IsAgreeForPayment"
    android:layout_width="30dp"
    android:layout_height="21dp"
    android:layout_marginLeft="76dp"
    android:layout_marginTop="27dp"
    android:layout_marginEnd="40dp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintLeft_toRightOf="@+id/IsAgreeForPaymentTextView"
    app:layout_constraintTop_toBottomOf="@+id/CertificateScoreX10Label"
    tools:ignore="MissingConstraints" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>
</ScrollView>

```

app_bar_main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.coordinatorlayout.widget.CoordinatorLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">

    <com.google.android.material.appbar.AppBarLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:theme="@style/Theme.DiplomaProject.AppBarOverlay">

        <androidx.appcompat.widget.Toolbar
            android:id="@+id/toolbar"
            android:layout_width="match_parent"
            android:layout_height="?attr/actionBarSize"
            android:background="?attr/colorPrimary"
            app:popupTheme="@style/Theme.DiplomaProject.PopupOverlay" />

    </com.google.android.material.appbar.AppBarLayout>

    <include layout="@layout/content_main" />

</androidx.coordinatorlayout.widget.CoordinatorLayout>

```

content_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout      xmlns:android="http://schemas.an-
droid.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    app:layout_behavior="@string/appbar_scrolling_view_behavior"
    tools:showIn="@layout/app_bar_main">

    <fragment
        android:id="@+id/nav_host_fragment"
        android:name="androidx.navigation.fragment.NavHostFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        app:defaultNavHost="true"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:navGraph="@navigation/mobile_navigation" />
</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_calculator.xml

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout      xmlns:android="http://schemas.an-
droid.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:id="@+id/calculateFragment"
    tools:context=".businesslogic.CalculationFragment">

    <TextView

        android:id="@+id/textView5"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="10dp"
        android:gravity="center_horizontal"
        android:text="Введите ваши баллы"
        android:textSize="25sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

    <TextView
        android:id="@+id/textView2"
```

```
android:layout_width="193dp"
android:layout_height="48dp"
android:layout_marginStart="10dp"
android:layout_marginTop="28dp"
android:gravity="center_horizontal"
android:text="Первый профильный предмет"
android:textSize="18sp"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/textView5" />
```

```
<TextView
    android:id="@+id/textView3"
    android:layout_width="193dp"
    android:layout_height="48dp"
    android:layout_marginStart="10dp"
    android:layout_marginTop="28dp"
    android:gravity="center_horizontal"
    android:text="Второй профильный предмет"
    android:textSize="18sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView2" />
```

```
<TextView
    android:id="@+id/textView4"
    android:layout_width="193dp"
    android:layout_height="48dp"
    android:layout_marginStart="10dp"
    android:layout_marginTop="28dp"
    android:gravity="center_horizontal"
    android:text="Третий предмет"
    android:textSize="18sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView3" />
```

```
<TextView
    android:id="@+id/textView6"
    android:layout_width="193dp"
    android:layout_height="48dp"
    android:layout_marginStart="10dp"
    android:layout_marginTop="28dp"
    android:gravity="center_horizontal"
    android:text="Балл документа об образовании"
    android:textSize="18sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/textView4" />
```

```
<EditText
    android:id="@+id/subj1Edit"
    android:layout_width="100dp"
    android:layout_height="48dp"
    android:layout_marginStart="57dp"
    android:layout_marginTop="28dp"
```

```

android:layout_marginEnd="33dp"
android:ems="10"
android:inputType="number"
android:textSize="16sp"
app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintStart_toEndOf="@+id/textView2"
app:layout_constraintTop_toBottomOf="@id/textView5" />

```

```

<EditText
    android:id="@+id/subj2Edit"
    android:layout_width="100dp"
    android:layout_height="48dp"
    android:layout_marginStart="57dp"
    android:layout_marginTop="28dp"
    android:layout_marginEnd="33dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/textView3"
    app:layout_constraintTop_toBottomOf="@+id/subj1Edit" />

```

```

<EditText
    android:id="@+id/subj3Edit"
    android:layout_width="100dp"
    android:layout_height="48dp"
    android:layout_marginStart="55dp"
    android:layout_marginTop="28dp"
    android:layout_marginEnd="35dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/textView4"
    app:layout_constraintTop_toBottomOf="@+id/subj2Edit" />

```

```

<EditText
    android:id="@+id/certificateEdit"
    android:layout_width="100dp"
    android:layout_height="48dp"
    android:layout_marginStart="55dp"
    android:layout_marginTop="28dp"
    android:layout_marginEnd="35dp"
    android:ems="10"
    android:inputType="number"
    android:textSize="16sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/textView6"
    app:layout_constraintTop_toBottomOf="@+id/subj3Edit" />

```

```

<View
    android:layout_width="match_parent"

```

```

        android:layout_height="10dp"
        android:layout_marginTop="20dp"
        android:background="@android:drawable/dialog_holo_light_frame"
        app:layout_constraintEnd_toStartOf="@id/resultTextView"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/textView6" />

```

```

<TextView
    android:id="@+id/resultTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="48dp"
    android:layout_marginTop="56dp"
    android:text="РЕЗУЛЬТАТ"
    android:textSize="20sp"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/certificateEdit"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/resultText"
    android:layout_width="wrap_content"
    android:layout_height="67dp"
    android:layout_marginTop="20dp"
    android:inputType="number"
    android:textSize="50sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintHorizontal_bias="0.498"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/resultTextView" />

```

```

<Button
    android:id="@+id/calculateBtn"
    android:layout_width="193dp"
    android:layout_height="45dp"
    android:layout_marginTop="36dp"
    android:text="Подсчитать"
    android:textSize="19sp"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/resultText" />

```

```

</androidx.constraintlayout.widget.ConstraintLayout>

```

fragment_calendar.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"

```

```
xmlns:app="http://schemas.android.com/apk/res-auto"  
tools:context=".businesslogic.CalendarFragment">
```

```
<CalendarView  
    android:id="@+id/calendarView"  
    android:layout_width="match_parent"  
    android:layout_height="wrap_content"  
    android:selectedWeekBackgroundColor="#ff0000"  
    android:weekNumberColor="#0000ff"  
    android:weekSeparatorLineColor="#00ff00"  
    tools:ignore="MissingConstraints" />  
  
<TextView  
    android:id="@+id/eventTitle"  
    android:layout_width="match_parent"  
    android:layout_height="50dp"  
    android:layout_marginTop="20dp"  
    android:textColor="@color/purple_200"  
    android:textAlignment="center"  
    android:textSize="20dp"  
    app:layout_constraintTop_toBottomOf="@+id/calendarView"/>
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_contacts.xml

```
<?xml version="1.0" encoding="utf-8"?>  
<androidx.constraintlayout.widget.ConstraintLayout  
    xmlns:android="http://schemas.android.com/apk/res/android"  
    xmlns:app="http://schemas.android.com/apk/res-auto"  
    xmlns:tools="http://schemas.android.com/tools"  
    android:layout_width="match_parent"  
    android:layout_height="match_parent">  
  
    <TextView  
        android:id="@+id/mainTextView"  
        android:layout_width="321dp"  
        android:layout_height="63dp"  
        android:layout_marginStart="45dp"  
        android:layout_marginTop="86dp"  
        android:layout_marginEnd="45dp"  
        android:inputType="textMultiLine"  
        android:text="Учреждение образования «Гомельский государственный техниче-  
ский университет имени П.О.Сухого» (ГГТУ им. П.О. Сухого)"  
        android:textAlignment="center"  
        android:textColor="@color/purple_200"  
        android:textStyle="bold"  
        app:layout_constraintEnd_toEndOf="parent"  
        app:layout_constraintStart_toStartOf="parent"  
        app:layout_constraintTop_toTopOf="parent"
```



```

tools:ignore="MissingConstraints" />

<TextView
    android:id="@+id/addressTextView"
    android:layout_width="314dp"
    android:layout_height="51dp"
    android:layout_marginStart="16dp"
    android:layout_marginTop="22dp"
    android:layout_marginEnd="16dp"
    android:inputType="textMultiLine"
    android:text="Пр-т Октября, 48, 246746, г. Гомель, Республика Беларусь"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/mainTextView"
    android:textAlignment="center"
    tools:ignore="MissingConstraints" />

<TextView
    android:id="@+id/phoneTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="76dp"
    android:layout_marginTop="53dp"
    android:inputType="textMultiLine"
    android:text="Телефон:"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/addressTextView"
    tools:ignore="MissingConstraints" />

<TextView
    android:id="@+id/phoneValueTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="55dp"
    android:layout_marginTop="53dp"
    android:layout_marginEnd="91dp"
    android:inputType="phone"
    android:text="+375 232) 22-46-36"
    android:autoLink="phone"
    android:linksClickable="true"
    android:textIsSelectable="true"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/phoneTextView"
    app:layout_constraintTop_toBottomOf="@+id/addressTextView"
    tools:ignore="MissingConstraints" />

<TextView
    android:id="@+id/faxTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="103dp"

```

```

android:layout_marginTop="29dp"
android:inputType="textMultiLine"
android:text="Факс"
android:textColor="@color/purple_200"
app:layout_constraintStart_toStartOf="parent"
app:layout_constraintTop_toBottomOf="@+id/phoneTextView"
tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/faxValueTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="55dp"
    android:layout_marginTop="29dp"
    android:layout_marginEnd="91dp"
    android:inputType="phone"
    android:text="( +375 232) 26-02-87"
    android:textIsSelectable="true"
    android:autoLink="phone"
    android:linksClickable="true"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/faxTextView"
    app:layout_constraintTop_toBottomOf="@+id/phoneValueTextView"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/emailGstuTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="99dp"
    android:layout_marginTop="29dp"
    android:inputType="textMultiLine"
    android:text="E-mail"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/faxTextView"
    tools:ignore="MissingConstraints" />

```

```

<TextView
    android:id="@+id/emailGstuValueTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="55dp"
    android:layout_marginTop="29dp"
    android:layout_marginEnd="123dp"
    android:inputType="textEmailAddress"
    android:text="rector@gstu.by"
    android:autoLink="email"
    android:linksClickable="true"
    android:textIsSelectable="true"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/emailGstuTextView"

```

```
app:layout_constraintTop_toBottomOf="@+id/faxValueTextView"
tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/siteValueTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="146dp"
    android:layout_marginTop="29dp"
    android:layout_marginEnd="146dp"
    android:inputType="textWebEditText"
    android:text="http://www.gstu.by"
    android:textIsSelectable="true"
    android:autoLink="web"
    android:linksClickable="true"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/emailGstuValueTextView"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/workTimeTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_marginStart="39dp"
    android:layout_marginTop="29dp"
    android:inputType="textMultiLine"
    android:text="Режим работы"
    android:textColor="@color/purple_200"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/siteValueTextView"
    tools:ignore="MissingConstraints" />
```

```
<TextView
    android:id="@+id/workTimeValueTextView"
    android:layout_width="176dp"
    android:layout_height="41dp"
    android:layout_marginStart="55dp"
    android:layout_marginTop="29dp"
    android:layout_marginEnd="45dp"
    android:inputType="textMultiLine"
    android:text="понедельник - пятница 8.00 - 12.00; 13.00 - 17.00"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toEndOf="@+id/workTimeTextView"
    app:layout_constraintTop_toBottomOf="@+id/siteValueTextView"
    tools:ignore="MissingConstraints" />
```

```
</androidx.constraintlayout.widget.ConstraintLayout>
```

fragment_main_page.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".businesslogic.MainPageFragment"
    android:id="@+id/fragment_main_page">

    <ImageButton
        android:id="@+id/admission"
        android:layout_width="170dp"
        android:layout_height="170dp"
        android:layout_marginTop="5dp"
        android:src="@drawable/admission"
        app:layout_constraintRight_toRightOf="@+id/sokr"
        app:layout_constraintTop_toBottomOf="@+id/sokr"
        tools:ignore="MissingConstraints" />

    <ImageButton
        android:id="@+id/campaign"
        android:layout_width="170dp"
        android:layout_height="170dp"
        android:layout_marginTop="5dp"
        android:src="@drawable/campaign"
        app:layout_constraintLeft_toLeftOf="@+id/speciality"
        app:layout_constraintTop_toBottomOf="@+id/speciality"
        tools:ignore="MissingConstraints" />

    <ImageButton
        android:id="@+id/speciality"
        android:layout_width="345dp"
        android:layout_height="170dp"
        android:layout_marginStart="5dp"
        android:layout_marginTop="5dp"
        android:layout_marginEnd="5dp"
        android:src="@drawable/specialnosti"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:ignore="MissingConstraints" />

    <ImageButton
        android:id="@+id/regulations"
        android:layout_width="170dp"
        android:layout_height="170dp"
        android:layout_marginTop="5dp"
        android:src="@drawable/regulations"
        app:layout_constraintRight_toRightOf="@+id/speciality"
        app:layout_constraintTop_toBottomOf="@+id/speciality"

```

```

tools:ignore="MissingConstraints" />

<ImageButton
    android:id="@+id/sokr"
    android:layout_width="345dp"
    android:layout_height="170dp"
    android:layout_marginTop="5dp"
    android:src="@drawable/sokr"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toBottomOf="@+id/campaign"
    tools:ignore="MissingConstraints" />

<ImageButton
    android:id="@+id/excursia"
    android:layout_width="170dp"
    android:layout_height="170dp"
    android:layout_marginTop="5dp"
    android:src="@drawable/excursia"
    app:layout_constraintLeft_toLeftOf="@id/sokr"
    app:layout_constraintTop_toBottomOf="@+id/sokr"
    tools:ignore="MissingConstraints" />
</androidx.constraintlayout.widget.ConstraintLayout>

```

fragment_maps.xml

```

<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    </androidx.constraintlayout.widget.ConstraintLayout>

```

fragment_send_mail.xml

```

<RelativeLayout
    xmlns:tools="http://schemas.android.com/tools"
    xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout_width="match_parent"
    tools:context=".businesslogic.sendemail.SendMailFragment"
    android:layout_height="match_parent"
    android:id="@+id/sendMailFragment"
    tools:ignore="NamespaceTypo">

    <EditText
        android:id="@+id/subjEditText"
        android:layout_width="248dp"
        android:layout_height="51dp"
        android:layout_alignParentTop="true"

```

```
android:layout_alignParentRight="true"
android:layout_marginTop="37dp"
android:layout_marginRight="14dp"
android:background="@drawable/edit_text_style"
android:elevation="5dp"
android:ems="10"
android:gravity="top|left"
android:hint="Введите тему письма">
```

```
</EditText>
```

```
<EditText
    android:id="@+id/messageEditText"
    android:layout_width="wrap_content"
    android:layout_height="447dp"
    android:layout_below="@+id/subjEditText"
    android:layout_alignLeft="@+id/subjTextView"
    android:layout_alignRight="@+id/subjEditText"
    android:layout_marginLeft="0dp"
    android:layout_marginTop="48dp"
    android:layout_marginRight="4dp"
    android:background="@drawable/edit_text_style"
    android:elevation="5dp"
    android:ems="10"
    android:gravity="top|left"
    android:hint="Введите текст письма"
    android:inputType="textMultiLine" />
```

```
<TextView
    android:id="@+id/subjTextView"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_alignBaseline="@+id/subjEditText"
    android:layout_alignBottom="@+id/subjEditText"
    android:layout_alignParentLeft="true"
    android:layout_marginLeft="20dp"
    android:textSize="18sp"
    android:text="Тема:" />
```

```
<Button
    android:id="@+id/sendEmailButton"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_below="@+id/messageEditText"
    android:layout_alignRight="@+id/messageEditText"
    android:layout_marginTop="50dp"
    android:text="Отправить" />
```

```
</RelativeLayout>
```

fragment_site.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    tools:context="businesslogic.web.SiteFragment">

    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="773dp"
        android:layout_weight="1" />
    <ScrollView
        android:layout_weight="1"
        android:layout_width="1dp"
        android:layout_height="0dp">

        <TextView android:id="@+id/content"
            android:layout_width="0dp"
            android:layout_height="0dp" />
    </ScrollView>

</LinearLayout>
```

fragment_web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    xmlns:tools="http://schemas.android.com/tools"
    android:orientation="vertical"
    tools:context="businesslogic.web.WebFragment">

    <WebView
        android:id="@+id/webView"
        android:layout_width="match_parent"
        android:layout_height="773dp"
        android:layout_weight="1" />
    <ScrollView
        android:layout_weight="1"
        android:layout_width="1dp"
        android:layout_height="0dp">

        <TextView android:id="@+id/content"
            android:layout_width="0dp"
            android:layout_height="0dp" />

    </ScrollView>

</LinearLayout>
```

</ScrollView>

</LinearLayout>

nav_header_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    android:layout_width="match_parent"
    android:layout_height="200dp"
    android:background="@drawable/maxresdefault"
    android:gravity="bottom"
    android:orientation="vertical"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:theme="@style/ThemeOverlay.AppCompat.Dark">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="120dp"
        android:layout_height="78dp"
        android:contentDescription="@string/nav_header_desc"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        app:srcCompat="@drawable/gstu" />

    <TextView
        android:layout_width="151dp"
        android:layout_height="wrap_content"
        android:paddingTop="@dimen/nav_header_vertical_spacing"
        android:text="Абитуриент ГГТУ"
        android:textAppearance="@style/TextAppearance.AppCompat.Body1" />

    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="gstuenrollee@gmail.com" />
</LinearLayout>
```

activity_main_drawer.xml

```
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    tools:showIn="navigation_view">

    <group android:checkableBehavior="single">
        <item
            android:id="@+id/nav_main_page"
```



```

        android:icon="@drawable/home"
        android:title="Главная" />
    <item
        android:id="@+id/nav_site"
        android:icon="@drawable/browser"
        android:title="Наш сайт" />
    <item
        android:id="@+id/nav_map"
        android:icon="@drawable/location"
        android:title="Университет на карте" />
    <item
        android:id="@+id/nav_calendar"
        android:icon="@drawable/calendar"
        android:title="Календарь"/>
    <item
        android:id="@+id/nav_calc"
        android:icon="@drawable/calculate"
        android:title="Калькулятор баллов" />
    <item
        android:id="@+id/nav_contacts"
        android:icon="@drawable/contact"
        android:title="Контакты" />
    <item
        android:id="@+id/nav_mail"
        android:icon="@drawable/mail"
        android:title="Связь с разработчиком" />
</group>
</menu>

```

main.xml

```

<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto">
    <item
        android:id="@+id/account"
        android:orderInCategory="100"
        android:title="Войти в личный кабинет"
        android:icon="@drawable/account"
        app:showAsAction="always" />
</menu>

```

mobile_navigation.xml

```

<?xml version="1.0" encoding="utf-8"?>
<navigation xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/mobile_navigation"
    app:startDestination="@+id/nav_main_page">

    <fragment

```

```

        android:id="@+id/nav_main_page"
        android:name="com.example.diplomaproject.businesslogic.MainPageFragment"
        android:label="Главная"
        tools:layout="@layout/fragment_main_page" />

<fragment
    android:id="@+id/nav_site"
    android:name="com.example.diplomaproject.businesslogic.web.SiteFragment"
    android:label="Наш сайт"
    tools:layout="@layout/fragment_site" />

<fragment
    android:id="@+id/nav_map"
    android:name="com.example.diplomaproject.businesslogic.map.MapsFragment"
    android:label="Университет на карте"
    tools:layout="@layout/fragment_maps" />

<fragment
    android:id="@+id/nav_calendar"
    android:name="com.example.diplomaproject.businesslogic.CalendarFragment"
    android:label="Календарь"
    tools:layout="@layout/fragment_calendar" />

<fragment
    android:id="@+id/nav_calc"
    android:name="com.example.diplomaproject.businesslogic.CalculationFragment"
    android:label="Калькулятор баллов"
    tools:layout="@layout/fragment_calculator" />

<fragment
    android:id="@+id/nav_contacts"
    android:name="com.example.diplomaproject.businesslogic.ContactsFragment"
    android:label="Контакты"
    tools:layout="@layout/fragment_contacts" />
<fragment
    android:id="@+id/nav_mail"
    android:name="com.example.diplomaproject.businesslogic.sendemail.SendMailFrag-
ment"
    android:label="Связь с разработчиком"
    tools:layout="@layout/fragment_send_mail" />
</navigation>

```

strings.xml

```

<resources>
    <string name="app_name">Абитуриент ГГТУ</string>
    <string name="menu_home">Home</string>
    <string name="menu_gallery">Gallery</string>
    <string name="menu_slideshow">Slideshow</string>
    <string name="title_activity_login">Authorization</string>
    <string name="prompt_email">Email</string>

```

```

    <string name="prompt_password">Password</string>
    <string name="action_sign_in">Sign in or register</string>
    <string name="action_sign_in_short">Sign in</string>
    <string name="welcome">"Welcome !"</string>
    <string name="invalid_username">Not a valid username</string>
    <string name="invalid_password">Password must be >5 characters</string>
    <string name="login_failed">"Login failed"</string>
    <string name="title_activity_maps">Map</string>
    <string name="title_activity_main">MainActivity</string>
    <string name="fais_link"><a href="https://fais.gstu.by/">Факультет автоматизирован-
ных информационных систем </a></string>
    <string name="mtf_link"><a href="https://mtf.gstu.by/">Механико-технологический
факультет </a></string>
    <string name="msf_link"><a href="https://msf.gstu.by/">Машиностроительный
факультет </a></string>
    <string name="gef_link"><a href="https://gef.gstu.by/">Гуманитарно-экономический
факультет </a></string>

    <string name="ef_link"><a href="https://ef.gstu.by/">Энергетический факультет
</a></string>
    <!-- TODO: Remove or change this placeholder text -->
    <string name="hello_blank_fragment">Hello blank fragment</string>

</resources>

```

ПРИЛОЖЕНИЕ Б

(справочное)

Каталог функций программного обеспечения

Таблица Б.1

Код функций	Наименование (содержание) функций	Объем функции строк исходного кода (LOC)	
		По каталогу (V ₀)	уточненный (V _y)
Ввод, анализ входной информации			
101	Организация ввода информации	130	70
102	Контроль, предварительная обработка и ввод информации	490	190
109	Управление вводом-выводом	1970	1830
Формирование, ведение и обслуживание базы данных			
206	Манипулирование данными	7860	195
207	Организация поиска и поиск в базе данных	4720	350
Формирование и обработка файлов			
304	Управление файлами	5240	35
Управление ПО, компонентами ПО и внешними устройствами			
506	Обработка ошибочных сбойных ситуаций	1540	80
507	Обеспечение интерфейса между компонентами	1680	50
Тестирование, проведение тестовых испытаний прикладных программ, вспомогательные программные функции			
602	Вспомогательные и сервисные программы	470	300
	Итого:	24100	3100

ПРИЛОЖЕНИЕ В

(справочное)

Расчет общей трудоемкости разработки программного обеспечения

Таблица В.1

Показатели	Стадии разработки					Итого
	ТЗ	ЭП	ТП	РП	ВН	
Общий объем ПО (V_o), кол-во строк LOC	-	-	-	-	-	24100
Общий уточненный объем ПО (V_y), кол-во строк LOC	-	-	-	-	-	3100
Категория сложности разрабатываемого ПО	-	-	-	-	-	3
Нормативная трудоемкость разработки ПО (T_n), чел./дн.	-	-	-	-	-	134
Коэффициент повышения сложности ПО (K_c)	1,14	1,14	1,14	1,14	1,14	-
Коэффициент, учитывающий новизну ПО (K_n)	0,63	0,63	0,63	0,63	0,63	-
Коэффициент, учитывающий степень использования стандартных модулей (K_t)	-	-	-	0,9	-	0,9
Коэффициент, учитывающий средства разработки ПО (K_{yp})	1,2	1,2	1,2	1,2	1,2	-
Коэффициенты удельных весов трудоемкости стадий разработки ПО ($K_{тз}$), $K_{эп}$, $K_{тп}$, $K_{рп}$, $K_{вн}$)	0,08	0,19	0,28	0,34	0,11	1,0
Распределение скорректированной (с учетом K_c , K_n , K_t , K_{yp}) трудоемкости ПО по стадиям, чел./дн.	9	22	32	35	13	-
Общая трудоемкость разработки ПО (T_o), чел./дн.	-	-	-	-	-	111

ПРИЛОЖЕНИЕ Г

(справочное)

Параметры для расчета производственных затрат на разработку программного обеспечения

Таблица Г.1

Параметр	Единица измерения	Значение
Базовая ставка	руб.	195
Разряд разработчика	—	1
Тарифный коэффициент 5-го разряда	—	1,29
Коэффициент $K_{ув}$	—	1,79
Норматив отчислений на доп. зарплату разработчиков ($H_{доп}$)	%	10
Численность обслуживающего персонала	чел.	1
Разряд обслуживающего персонала	—	5
Базовая ставка 5-го разряда	—	159,63
Коэффициент $K_{ув}$	—	2,48
Средняя годовая ставка арендных платежей ($C_{АР}$)	руб./м ²	195.8
Площадь помещения (S)	м ²	9
Количество ПЭВМ ($Q_{ЭВМ}$)	шт.	1
Затраты на приобретение единицы ПЭВМ	руб.	1600
Стоимость одного кВт-часа электроэнергии ($C_{эл}$)	руб.	0,2909
Затраты на технологию ($Z_{тех}$)	руб.	—
Норматив общепроизводственных затрат ($H_{доп}$)	%	5
Норматив непроизводственных затрат ($H_{непр}$)	%	5

ПРИЛОЖЕНИЕ Д
(справочное)

Расчет суммарных затрат на разработку программного обеспечения

Таблица Д.1

Статья затрат	Итого
Затраты на оплату труда разработчиков ($Z_{тр}$), руб.	5981,65
Затраты машинного времени ($Z_{мв}$), руб.	307,2
Стоимость машино-часа, руб/ч	3,84
Сумма годовых амортизационных отчислений, руб.	350
Действительный годовой фонд времени работы ПЭВМ, дн.	2056
Затраты на текущий и профилактический ремонт, руб.	140
Прочие затраты, связанные с эксплуатацией ЭВМ, руб.	140
Машинное время ЭВМ, ч.	80
Затраты на изготовление эталонного экземпляра ($Z_{эт}$), руб.	0
Затраты на технологию ($Z_{тех}$), руб.	0
Общепроизводственные затраты ($Z_{общ,пр}$)	150,93
Непроизводственные (коммерческие) затраты ($Z_{непр}$)	150,93
Затраты на материалы ($Z_{мат}$), руб.	28
Суммарные затраты на разработку ПО (Z_p)	6618,71

ПРИЛОЖЕНИЕ Е

(справочное)

Технико-экономические показатели проекта

Таблица Е.1

№ п/п	Наименование показателя	Единица измерения	Базовый вариант	Проектный вариант
Показатели затрат на разработки				
1	Общая трудоемкость разработки ПО	чел.-дн.	×	111
2	Затраты на разработку ПО	руб.	×	6618,71
2.1	Затраты на оплату труда разработчиков	руб.	×	5981,65
2.2	Затраты машинного времени	руб.	×	307,2
2.3	Затраты на материалы	руб.	×	28
2.4	Общепроизводственные затраты	руб.	×	150,93
2.5	Непроизводственные затраты	руб.	×	150,93
Показатели стоимости				
3	Отпускная цена ПП с НДС	руб.	×	10325,18
4	Розничная цена ПП	руб.	24575	11357,7
Показатели экономической эффективности				
5	Рентабельность затрат	%	×	116,4
6	Простой срок окупаемости проекта	лет	×	0,86
7	Годовой экономический эффект	руб.	×	10377,88