# РЕФЕРАТ

МОДУЛЬ "НАВИГАЦИЯ С ПОМОЩЬЮ КАМЕРЫ" В МОБИЛЬНОМ ПРИЛОЖЕНИИ "НАВИГАТОР БНТУ"

Объектом разработки является модуль для мобильного приложения “навигатор БНТУ”.

Цель проекта – разработать приложение, которое будет сообщать информацию о занятиях студентам, сканирующим с помощью приложения QR-код, висящий на аудитории.

В процессе работы выполнены следующие разработки: автоматизация процесса взаимодействия с расписанием.

Элементами научной новизны (практической значимости) полученных

результатов являются:

– легкое масштабирование приложения;

– простой и современный дизайн приложения.

В ходе дипломного проектирования прошли апробацию такое предложение как сканирование QR-кода для интерактивного взаимодействия с расписанием.

Результатами внедрения явились: удобство поиска необходимой аудитории при помощи камеры.

Студент-дипломник подтверждает, что приведенный в дипломном проекте

расчетно-аналитический материал объективно отражает состояние разрабатываемого объекта, все заимствованные из литературных и других источников теоретические и методологические положения и концепции сопровождаются ссылками на их авторов.

Дипломный проект: 71 с., 31 рис., 23 табл., 13 источников.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| № строки | Формат | Обозначение | | | Наименование | | Кол. листов | Примечание | |
| 1 |  |  | | | Документация общая | |  |  | |
| 2 |  |  | | |  | |  |  | |
| 3 | А4 | - | | | Задание по дипломному проекту | | 1 |  | |
| 4 | А4 | ДП–10702116/16–2020–РПЗ | | | Расчётно-пояснительная записка | | 71 |  | |
| 5 | А4 | ДП–10702116/16–2020-01 | | | Диаграмма классов | | 1 |  | |
| 6 | А4 | ДП–10702116/16–2020–02 | | | Схема программы | | 1 |  | |
| 7 | А4 | ДП–10702116/16–2020–03 | | | Диаграмма последовательности | | 1 |  | |
| 8 | А4 | ДП–10702116/16–2020–04 | | | Схема базы данных | | 1 |  | |
| 9 | А4 | ДП–10702116/16–2020–05 | | | Запрос приложением прав на использование камеры | | 1 |  | |
| 10 | А4 | ДП–10702116/16–2020–06 | | | Сканирование QR-кода приложением | | 1 |  | |
| 11 | А4 | ДП–10702116/16–2020–07 | | | Поведение приложения при сканировании не валидного QR-кода | | 1 |  | |
| 12 | А4 | ДП–10702116/16–2020–08 | | | Поведение приложения, когда кабинете не проводится занятий | | 1 |  | |
| 13 | А4 | ДП–10702116/16–2020–09 | | | Стандартное поведение приложения | | 1 |  | |
| 14 | А4 | ДП–10702116/16–2020–10 | | | Ответ приложения на нажатие кнопки «занятия в аудитории» | | 1 |  | |
|  |  |  | | |  | |  |  | |
|  |  |  |  |  | ДП–10702116/16–2020–РПЗ | | | | |
|  |  |  |  |  |
| Изм | Лист | № документа | Подпись | Дата |
| Разраб. | | Рыбак |  |  | Ведомость объема дипломного проекта | Лит. | Лист | | Листов |
| Руковод. | | Белова |  |  | У |  | |  |
| Консульт. | | Белова |  |  | 1-40 05 01  БНТУ, г. Минск | | | |
| Н.контр. | | Домаренко |  |  |
| Зав.каф. | | Полозков |  |  |

**ОГЛАВЛЕНИЕ**

[ВВЕДЕНИЕ 7](#_Toc41408905)

[1 АНАЛИЗ МЕТОДОВ ОРИЕНТИРОВАНИЯ ВНУТРИ ЗДАНИЙ 8](#_Toc41408906)

[1.1Анализ методов решения задачи помощи в ориентировании 8](#_Toc41408907)

[Вывод 11](#_Toc41408908)

[2 АНАЛИТИЧЕСКИЙ ОБЗОР ПРОГРАММНО-ТЕХНИЧЕСКИХ СРЕДСТВ 12](#_Toc41408909)

[2.1Требования к программно-информационной системе 12](#_Toc41408910)

[2.2Анализ существующих мобильных платформ 12](#_Toc41408911)

[2.3Выбор технологии 18](#_Toc41408912)

[Вывод 19](#_Toc41408913)

[3 ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ОБЕСПЕЧЕНИЯ ПОМОЩИ СТУДЕНТАМ ПО ОРИЕНТАЦИИ ВНУТРИ ЗДАНИЙ 20](#_Toc41408914)

[3.1 Архитектура программного средства 20](#_Toc41408915)

[3.2 Выбор архитектурного шаблона 22](#_Toc41408916)

[3.3 Проектирование базы данных 22](#_Toc41408918)

[3.4 Проектирование и разработка приложения 26](#_Toc41408919)

[4 ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММНОГО СРЕДСТВА 46](#_Toc41408920)

[4.1 Критическое тестирование 46](#_Toc41408921)

[4.2 Углубленное тестирование 47](#_Toc41408922)

[4.3 Результаты сканирования 47](#_Toc41408923)

[5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ 48](#_Toc41408924)

[6 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРММНОГО ПРОДУКТА 50](#_Toc41408925)

[6.1 Определение единовременных затрат на создание программного продукта 50](#_Toc41408926)

[6.2. Расчет ожидаемого прироста прибыли в результате внедрения программного продукта 56](#_Toc41408927)

[6.3. Расчёт показателей эффективности использования программного продукта 59](#_Toc41408928)

[7 ОХРАНА ТРУДА 62](#_Toc41408929)

[7.1 Производственная санитария,техника безопасности и пожарная профилактика 65](#_Toc41408930)

[7.2 Требования к помещениям для работы с ВДТ, ЭВМ и ПЭВМ 67](#_Toc41408931)

[ЗАКЛЮЧЕНИЕ 70](#_Toc41408932)

[СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ 71](#_Toc41408933)

ГРАФИЧЕСКАЯ ЧАСТЬ

**ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ, СИМВОЛОВ И ТЕРМИНОВ**

База данных – совокупность взаимосвязанных данных, организованных в соответствии со схемой базы данных таким образом, чтобы с ними мог работать пользователь.

Программный интерфейс приложения – набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением для использования во внешних программных продуктах.

Операционная система – совокупность системных программ, предназначенная для обеспечения определенного уровня эффективности системы обработки информации за счет автоматизированного управления ее работой и предоставляемого пользователю определенного набора услуг.

Программа – данные, предназначенные для управления конкретными компонентами системы обработки информации в целях реализации определенного алгоритма.

Программное обеспечение – программы, процедуры, правила и любая соответствующая документация, относящиеся к работе вычислительной системы.

Программное средство – объект, состоящий из программ, процедур, правил, а также, если предусмотрено, сопутствующих им документации и данных, относящихся к функционированию системы обработки информации.

Система управления базами данных – совокупность программ и языковых средств, предназначенных для управления данными в базе данных, ведения базы данных и обеспечения взаимодействия ее с прикладными программами.

Смартфон – это класс современных мобильных телефонов. Они отличаются от обычных телефонов более мощными аппаратными возможностями и многофункциональными операционными системами.

Язык программирования – формальная знаковая система, предназначенная для записи компьютерных программ.

БД – база данных.

МУ – мобильное устройство.

ОС – операционная система.

ПО – программное обеспечение.

# ВВЕДЕНИЕ

В XXI веке количество людей и скорость их перемещений значительно выросли, архитектурные объекты и их территории растут пропорционально. В условиях постоянного урбанистического роста человеку требуется помощь в ориентировании как на улице, так и внутри общественных зданий.

БНТУ один из крупнейших белорусских университетов, его кампус насчитывает 17 учебных корпусов, и более 15 административных. Некоторые из корпусов находятся в шаговой доступности относительно друг друга, до других же приходится добираться на транспорте в течении продолжительного времени, что порождает неудобства, так как немногие студенты знают, где находится тот или иной корпус, и не имеют ни малейшего представления о внутреннем его устройстве, что приводит к значительным затратам времени на ориентирование. В связи с этим становится очевидным вопрос о необходимости помощника, некого проводника, который сможет сообщить, где находится то или иное здание, указать к нему путь, подсказать что происходит в той или иной аудитории.

Во время технологического роста и автоматизации большинства сфер жизни человека, людям стало куда привычнее пользоваться телефонами для повседневных нужд. МУ (мобильное устройство) – это устройство способное решать множество задач, и навигация с помощью мобильного устройства уже является обыденностью. Намного удобнее использовать смартфон для навигации, нежели использовать бумажную карту, которая не всегда под рукой, и шанс ошибиться у среднестатистического человека весьма велик, поэтому необходимость программного решения очевидна.

Целью дипломного проектирования является разработка мобильного приложения для помощи студентам в ориентировании внутри корпусов университета.

Данное программное средство предоставит пользователям возможность получить информацию о происходящем в аудитории в реальном времени, не прибегая к помощи расписания.

# АНАЛИЗ МЕТОДОВ ОРИЕНТИРОВАНИЯ ВНУТРИ ЗДАНИЙ

Системы позиционирования сыграли важную роль в жизни людей с тех пор, как спутниковая технология GPS стала общедоступной в конце 90-х годов. Сегодня почти у каждого есть устройство с возможностями позиционирования, такое как смартфон, планшет, устройство слежения GPS или часы со встроенным GPS.

GPS революционизировал наружное позиционирование. Теперь, спустя годы, технология позиционирования уходит в помещение и появляются новые возможности для использования внутри помещений. Спутниковое позиционирование не работает внутри помещений, но появились и другие технологические стандарты, которые делают возможным размещение внутри помещений. Позиционирование в помещении сложнее, чем позиционирование на улице с использованием GPS, потому что в помещении должна быть определенная инфраструктура. Сигналы GPS не работают в помещении или на узких улицах, поскольку они имеют тенденцию ослаблять и рассеивать по крышам и стенам.

## Анализ методов решения задачи помощи в ориентировании

Навигация с помощью камеры в помещении достаточно нетривиальная задача и, на данный момент, единого подхода к решению данного вопроса нет. Существует несколько вариантов для решения подобной задачи: Bluetooth маяки, RSSI fingerprinting, навигация с помощью компьютерного зрения, авторские решения. У каждого из них собственный способ определения своего местоположения, рассмотрим их подробнее.

### Bluetooth маяки

Основная принцип работы в том, что Маяки размещены на нескольких позициях в здании. Через Bluetooth они отправляют сигналы на мобильные устройства. Таким образом, можно непрерывно определять их положение и передавать его во внутреннюю навигационную систему. Приложение, установленное на смартфоне, интерпретирует сигналы. Срок службы батареи маяка составляет от нескольких дней до нескольких лет, и они могут охватывать до 30 метров внутри зданий [1]. Маяки могут работать от батарей, поэтому их очень легко установить. Система управления маяками облегчает техническое обслуживание. Маяки также могут быть подключены к электросети, что усложняет установку. Также возможно интегрировать их в лампы, чтобы они могли использовать сеть питания ламп.

Данное решение неудобно тем, что требует денежных трат на покупку маяков, их установку, постоянное поддерживание питания, а также возможный ремонт в случае неисправности.

### RSSI fingerprinting

В методе на основе RSSI используется точка доступа WiFi, установленная в непосредственной близости, а местоположение рассчитывается с использованием свойства, в котором интенсивность сигнала постоянно уменьшается с расстоянием. Метод на основе Bluetooth использует маяк с низким энергопотреблением для расчета местоположения вместо WiFi. Метод требует метки RFID (идентификатор радио-волн) и считывателя RFID в качестве метода идентификации пользователя с помощью радиоволн. В настоящее время методы, основанные на Bluetooth и WIFI, имеют ограничения, поскольку требуют дополнительной установки. В частности, использование WIFI ограничено дальностью устройства передачи сигнала. Но для распознавания местоположения в помещении с использованием WiFi можно использовать точку доступа WiFi, установленную в существующем здании, и, следовательно, нет необходимости создавать дополнительную среду, и большинство современных устройств имеют преимущество в поддержке WiFi [2].

Данный метод хорош для помещений с полным покрытием WiFi сигналом, что требует либо маленькой площади, либо большого количества передатчиков, что так же является его основным недостатком и для решения нашей задачи не подходит.

### Метод компьютерного зрения

Метод компьютерного зрения единственный метод, не требующий дополнительных устройств для определения собственного местоположения, что является, несомненно, плюсом по сравнению с другими способами. Данная технология использует дополненную реальность для отображения направления движения навигации. Существуют две основных технологии ArCore – для андройд устройств, ArKit – для устройств компании Apple. Оба набора разработки появились значительно позже нежели предыдущие способы навигации, и устоявшихся решений для навигации внутри зданий пока нет, что является значительным недостатком данных решений.

ARCore – это набор для разработки программного обеспечения, разработанный Google, который позволяет создавать приложения дополненной реальности. Релиз технологии был сделан 1 марта 2018 года. По сути, ARCore делает две вещи: отслеживает положение мобильного устройства во время его движения и формирует собственное понимание реального мира. Технология отслеживания движения ARCore использует камеру телефона для определения интересных точек, называемых функциями, и отслеживает, как эти точки перемещаются с течением времени. Благодаря сочетанию движения этих точек и показаний с инерционных датчиков телефона ARCore определяет положение и ориентацию телефона при его перемещении в пространстве. В дополнение к определению ключевых точек, ARCore может обнаруживать плоские поверхности, такие как стол или пол, а также оценивать среднее освещение в области вокруг него. Эти возможности в совокупности позволяют ARCore создавать собственное понимание окружающего мира.

Набор инструментов ARKit, представленный разработчикам на WWDC 2017, представляет собой новый взгляд на технологии дополненной реальности. Разработка способна распознавать размеры окружающего пространства и учитывать условия освещения для максимально надежной интеграции виртуальных объектов в реальную жизнь. А совместимость с подавляющим большинством современных устройств iOS делает ARKit самой массовой платформой дополненной реальности в мире.

ARKit сочетает в себе отслеживание движения устройства, захват сцены камеры, расширенную обработку сцен и удобство отображения, чтобы упростить задачу создания AR-приложения. С помощью этих технологий вы можете создавать различные виды AR, используя переднюю или заднюю камеру устройства iOS.

Существуют готовые решения для навигации с дополненной реальностью от отдельных компаний: MapBox – кроссплатформенное решение, Placenote – для iOS. Благодаря кроссплатформенности MapBox кажется отличным решением, однако на стадии разработки выяснилось, что документация не дает полной картины использования инструментария и является достаточно сырой, что значительно усложняет задачу.

Главным достоинством метода компьютерного зрения является то, что он не требует дополнительных финансовых и материальных вложений, кроме непосредственной разработки ПО, однако является самым ненадежным и недоработанным вариантом, так как дополненная реальность поддерживается на относительно малом количестве мобильных устройств и решения для оптимального использования навигации внутри помещения пока не разработано, что делает этот вариант тяжело поддерживаемым и слабо расширяемым.

### Авторский метод решения

Основой для авторского метода решено взять сканирование QR кодов. QR-код это двумерная версия штрих-кода, известная по упаковке продуктов в супермаркете. Первоначально разработанный для оптимизации процессов в логистике автомобильной промышленности, QR-код нашел свое применение в мобильном маркетинге благодаря широкому распространению смартфонов. «QR» означает «Быстрый ответ», что означает мгновенный доступ к информации, скрытой в коде. QR-коды набирают популярность, потому что технология «с открытым исходным кодом», то есть доступна для всех. Существенными преимуществами QR-кодов по сравнению с обычными штрих-кодами являются большая емкость данных и высокая отказоустойчивость. В одном коде можно ввести до 7089 цифр или 4296 символов, включая знаки пунктуации и специальные символы. В дополнение к цифрам и символам можно также кодировать слова и фразы (например, адреса в Интернете). Чем больше данных добавляется в QR-код, тем больше увеличивается размер кода, и структура кода становится более сложной. Создание QR кода занимает немного времени, в интернете существует множество генераторов, которые позволяют создать и сохранить ваше изображение.

На основе информации полученной из QR кода, студентам будет сообщено где они находятся, какая пара проходит в кабинете, на котором отсканирован код, кто преподаватель и какая группа сейчас там находится, что значительно упростит задачу поиска аудитории для студентов, так как аудитория сама им сообщит, что сейчас в ней происходит.

Преимуществом авторского метода является простота расширения приложения, так как для добавления нового функционала достаточно добавить логику обработки новых данных, не придется разрабатывать приложение с нуля, так же программисту не потребуется изучать отдельную независимую технологию для поддержки данного ПО, с точки зрения мобильных устройств приложение будет доступно каждому современному устройству, по тому как не требует поддержки узко ориентированных технологий.

## Вывод

В ходе анализа было решено разрабатывать мобильное приложение, так как на данный момент около 96 процентов людей пользуются телефонами, 81 процент из которых – это смартфоны.

В данном разделе были рассмотрены различные средства реализации ориентирования внутри зданий, в ходе анализа предметной области было выяснено, что существующие решения в данный момент требуют затрат финансовых ресурсов или же достаточно сложных систем со сложной поддержкой и достаточно узкой направленностью, поэтому оптимальным выбором будет разработка собственного решения.

# АНАЛИТИЧЕСКИЙ ОБЗОР ПРОГРАММНО-ТЕХНИЧЕСКИХ СРЕДСТВ

## Требования к программно-информационной системе

Зачем нам нужна система позиционирования в помещении? Ответ лежит в больших зданиях, таких как корпуса БНТУ, где может быть несколько этажей и крыльев корпуса, и поиск необходимой аудитории может занимать продолжительное время, а если студент не помнит какая аудитория ему нужна и расписания под рукой не оказалось? Из этого следует, что необходимо разработать приложение способное, помочь студенту с поисками необходимой аудитории.

Приложение должно соответствовать данным требованиям:

1. Приложение должно поддерживаться на большом количестве устройств: система без проблем устанавливается на большинство мобильных устройств, что бы возможность ее использования была максимальной;
2. Приложение должно быть расширяемым: система должна быть спроектирована таким образом, чтобы добавление нового функционала не требовало переписывания всего функционала, а позволяло добавить логику обработки новых данных, при этом не нарушая работу всего приложения;
3. Приложение должно быть легко поддерживаемым с точки зрения разработки: система должна иметь понятную структуру и соответствовать стандартам разработки;
4. Приложение должно использовать камеру в качестве средства навигации: система должна уметь взаимодействовать с камерой, обладать функционалом для обработки QR-кодов;
5. Приложение должно сообщать вспомогательную информацию: на основе данных полученных из QR-кода система должна сообщить пользователю информацию о расписании занятий в аудитории.

## Анализ существующих мобильных платформ

Мобильная ОС обычно запускается при включении устройства, представляя экран с иконками или плитками, которые представляют информацию и обеспечивают доступ к приложению. Мобильные операционные системы также управляют подключением к сотовой и беспроводной сети, а также доступом к телефону. На сегодняшний день существует множество операционных систем для мобильных устройств, и три из наиболее распространенных – это операционная система для iPhone, Apple iOS и открытая ОС Chrome, Google Android. Эти две мобильные ОС используют разные подходы к мобильной операционной системе.

### Android ОС

Android является мощной операционной системой и поддерживает большое количество приложений в смартфонах. У Android есть миллионы доступных приложений для различных аспектов повседневной жизни, и он доступен по низкой цене на рынке, поэтому Android очень популярен [3]. По состоянию на начало 2016 года более миллиарда человек активно используют Android, как мобильную операционную систему, подобную Unix, которая работает на телефонах, планшетах, смарт-телевизорах, умных часах и множестве других устройств. Хотя это разработано и поддерживается Google, это бесплатное программное обеспечение с открытым исходным кодом, которое может использовать любой пользователь. В результате устройства Android производятся несколькими крупными производителями устройств, такими как Samsung, LG, Sony и HTC. Одним из основных факторов, определяющих успех Android, является Google Play, являющийся платформой для распространения приложений Android, пользователи которой могут использовать ее для загрузки и установки приложений на свои устройства. С более чем миллионом приложений и миллиардами загрузок, Google Play имеет больше трафика, чем аналогичные платформы для iOS и Windows 10.

Разработка под Android возможна при помощи следующих языков: Java, C++, Kotlin и других. Java является основным языком для Android, язык кроссплатформенный, что дает свободу в выборе платформы для разработки. Также Java является одним из самых популярных языков программирования, что предоставляет большое количество готовых решений и большое сообщество программистов, которые вероятнее всего уже сталкивались с теми проблемами, которые разработчик может встретить в процессе, что делает Android одной из самых доступных мобильных платформ. На устройствах под управлением Android 5.0 или более поздней версии по умолчанию используется управляемая среда выполнения Android Runtime (ART). Для создания Android приложений большинство разработчиков выбирают Android Studio, официальную студию разработки Android IDE, разработанную Google. Android Studio бесплатна и работает на Windows, OS X и большинстве дистрибутивов Linux.

Android Studio 3.6, текущий стабильный выпуск, имеет большое количество функций, упрощающих все аспекты разработки приложений для Android, таких как мощный редактор кода, визуальные макеты и редакторы тем, студии векторной графики и графических ресурсов ,а так же мгновенный запуск и профилировщик графического процессора, для дальнейшего улучшения процесса разработки приложений.

Android Studio также поставляется с реалистичным эмулятором, который вы можете использовать, если у вас нет Android-устройства. Поскольку системные образы для него находятся в свободном доступе, вы можете загрузить и запустить любую версию Android на нем.

Одной из негативных сторон Android можно выделить слабую поддержку устаревших устройств, в отличие от iOS-устройств Apple, большинство Android-устройств редко получают обновления программного обеспечение, если они вообще появляются. Android 10, последняя версия системы, работает на 8,2% всех Android-устройств сегодня, хотя она была выпущена около полугода назад.

Это означает, что, если вы решили разработать приложение, вы должны поддерживать более старые версии Android, такие как KitKat и Lollipop. Более того, если ваша основная база пользователей из развивающихся стран, рекомендуется поддерживать более старые версии вплоть до Android Gingerbread. Однако это обычно легко сделать благодаря не только библиотеке поддержки Android , но и ряду сторонних библиотек с открытым исходным кодом, так же если приложение не включает в себя функциональностей последних версий, то разработка может вестись под старую версию, так как более новые версии системы, будут поддерживать его автоматически.

Обработка различных размеров экрана и плотности: в первые годы существования Android было только небольшое количество устройств, которые могли работать на Android, и почти все они были телефонами. Однако с годами, когда все больше и больше производителей устройств переходят на эту платформу, размеры экранов устройств и плотность пикселей стали сильно различаться. С выпуском Android Honeycomb Android также начал поддерживать устройства с большим экраном, такие как планшеты и умные телевизоры. Следовательно, сегодня разработчики должны использовать изображения и макеты разных размеров для поддержки всех этих разных экранов. Однако благодаря автоматизации и роста технологии данная задача перекладывается на среду разработки, разработчику в свою очередь нужно только настроить компоненты отображения. Платформа Android постоянно развивается, и разработчики приложений должны идти в ногу с последними тенденциями, чтобы создавать приложения, соответствующие ожиданиям пользователей. После того, как вы разработали приложение, вы можете опубликовать его в интернете, чтобы люди во всем мире могли скачать и использовать его. Существует множество торговых площадок для Android-приложений, на которых вы можете опубликовать свое приложение, но сегодня наиболее популярным является Google Play, поскольку он уже предустановлен на большинстве устройств Android.

Рассмотрим операционную систему с точки зрения требований:

1. Аудитория: смартфоны с данной операционной системой занимают лидирующее место на мировом рынке, на Android приходится более 86% всех мобильных устройств [3];
2. Расширяемость: разработка продуктов под Android поддерживает множество шаблонов проектирования: Singleton, One Responsibility Of Truth и др., что позволяет построить гибкую и расширяемую систему;
3. Поддержка: Java занимает лидирующее место среди языков программирования и имеет огромное сообщество [5], благодаря универсальности языка любой Java-программист способен быстро понять работу приложения, так как язык использует общие механизмы работы для различных отраслей использования;
4. Камера: система позволяет работать с камерой напрямую, без каких любо дополнительных средств;
5. Сообщение информации пользователю: система оснащена множеством средств для взаимодействия с пользователем: activity, toast и др.

### Chrome ОС

Chrome ОС это операционная система на основе ядра Linux, разработанная Google. Он основан на бесплатном программном обеспечении Chromium OS и в качестве основного пользовательского интерфейса использует веб-браузер Google Chrome. В Chrome OS встроен медиаплеер и файловый менеджер. Он поддерживает приложения Chrome, которые напоминают собственные приложения, а также удаленный доступ к рабочему столу. Первоначально прием был скептическим: некоторые наблюдатели утверждали, что браузер, работающий в любой операционной системе, был функционально эквивалентен. По мере того, как все больше компьютеров с Chrome OS выходят на рынок, операционная система теперь редко оценивается отдельно от оборудования, на котором она работает. На Chrome ОС поддерживает Android приложения, которые возможно загрузить из Google Play.

Chrome ОС доступна только на аппаратном уровне для партнеров-производителей Google, но существуют неофициальные методы, позволяющие устанавливать ее на другом оборудовании. Ранее Google сообщил о возможности свободной разработки для Chrome OS, но не выпустил техническое описание, что делает разработку под данную систему обычной разработкой под Android.

### Операционная система iOs

IOS является синонимом операционной системы iPhone. Система поддерживает Objective-C, C, C ++ и язык программирования Swift. Она основана на Macintosh OS X. iPhone, iPod touch и iPad поставляются с IOS. Таким образом, нацелены на умные носимые вещи, такие как умные часы. Следовательно, платформа iOS в тренде

iOS – это операционная система, разработанная под продукцию компании Apple: планшетов, музыкальных плееров и смартфонов, которая устанавливается только на такие устройства, как iPhone, iPod Touch и iPad. IOS нельзя установить на другие устройства. Система является закрытой, что увеличивает ее надежность, iOS является одной из самых защищенных операционных систем.

Разработка на платформе iOS быстрее, проще и дешевле. Время разработки приложения для андроид телефона на 30-40% больше, чем для iOS. Одна из причин, по которой IOS легче разрабатывать, заключается в коде. Приложения Android обычно написаны на Java, языке, который предполагает написание большего количества кода. Тем не менее, официальным языком программирования Apple является Swift. Swift легко написать и выполнить. Соответственно, это занимает меньше времени, чем любая другая платформа. Android – это платформа с открытым исходным кодом. Таким образом, отсутствие стандартизации означает больше устройств, компонентов и фрагментацию программного обеспечения для учета магазина игр для Android. Закрытая экосистема iOS означает, что она разрабатывается для нескольких стандартизированных устройств и операционных систем. Таким образом, в магазине приложений iOS существуют строгие правила и ожидания качества, и более длительный процесс проверки, что может занять больше времени для одобрения приложений. Кроме того, ваше приложение может быть отклонено, даже если оно не соответствует стандартам Apple.

Swift - мощный язык программирования для macOS, iOS, watchOS, tvOS и не только. Swift является результатом последних исследований языков программирования в сочетании с многолетним опытом создания платформ Apple, отлично оптимизирован под аппаратные средства Apple. Язык является мультипарадигменным и реализует в себе множество нетривиальных решений, что делает его максимально гибким для реализации современных проектов.

Архитектура iOS – это многоуровневая архитектура. На верхнем уровне iOS действует как посредник между базовым оборудованием и приложениями. Однако приложения не взаимодействуют напрямую с базовым оборудованием. Точно так же приложения общаются с оборудованием через набор четко определенных систем интерфейсов. Эти интерфейсы облегчают написание приложений, которые работают последовательно на устройствах с различными аппаратными возможностями. Нижние уровни предоставляют базовые услуги для всех приложений. Кроме того, он основан на уровне верхнего уровня и предоставляет сложный графический интерфейс и связанные с ним услуги. Apple предоставляет большинство своих системных интерфейсов в специальных пакетах, называемых фреймами. Фрейм – это каталог, который содержит динамическую совместно используемую библиотеку, такую как заголовочные файлы, связанные изображения и вспомогательные приложения, необходимые для поддержки ресурсов этой библиотеки. Таким образом, каждый уровень имеет набор структур, которые разработчик использует для создания приложений.

Cocoa Touch – это структура пользовательского интерфейса. Он используется для создания программ для запуска на iOS. Он обеспечивает уровень абстракции iOS, операционной системы для iPhone, iPod Touch и iPad. Точно так же Cocoa Touch основан на наборе инструментов MacOS API, так как он написан в основном на языке Objective-C. Таким образом, Cocoa Touch позволяет использовать аппаратные средства и функции, присутствующие на компьютерах Mac OS, и, следовательно, являются преимуществом для ряда устройств iOS. Кроме того, Cocoa Touch следует архитектуре программного обеспечения Model-View-Controller (MVC). Инструменты для разработки приложений на основе приложений Cocoa Touch включены в iOS SDK платформы iOS.

EventKit Frame – чтобы драйвера могли отображать стандартные системные интерфейсы для просмотра и изменения событий, связанных с календарем

GameKit Marco – реализует поддержку Game Center, что позволяет пользователям обмениваться информацией, связанной с игрой, онлайн

Marco Twitter – поддерживает пользовательский интерфейс для создания твитов и поддерживает создание URL-адресов для доступа к службе Twitter.

UIKit Framework – дает жизненно важное значение для реализации графических приложений, основанных на событиях в инфраструктуре IOS. Некоторые важные функции документации User Interface Kit:

Graphics Core Framework – это механизм рисования для собственных приложений IOS и поддерживает пользовательские 2D и рендеринг на основе векторов изображений.

Аппаратное обеспечение устройств способствует очень высокой производительности для своих работ рендеринга графики и сложных вычислений, так как предоставляет очень быстрый доступ к графическому процессору, зачёт направленности компонентов устройства конкретно под программные решения iOS.

Поддержка многозадачности, управление приложениями и базовая инфраструктура, пользовательский интерфейс управления, поддержка событий касания и движения это и многое другое предоставляет iOS по-умолчанию, что делает систему очень привлекательной для разработчиков.

Рассмотрим операционную систему с точки зрения требований:

1. Аудитория: смартфоны с данной операционной системой занимают лишь второе место на мировом рынке, на iOS приходится более 13% всех мобильных устройств [3];
2. Расширяемость: разработка продуктов под iOS поддерживает множество шаблонов проектирования: Singleton, Facade и др., что позволяет построить гибкую и расширяемую систему;
3. Поддержка: Swift имеет достаточно большое сообщество разработчиков, что связано с его прогрессивностью и актуальностью. Однако для возможности поддержки проекта под iOS необходимо иметь устройство с последней версией MacOS [6];
4. Камера: система позволяет работать с камерой напрямую, без каких любо дополнительных средств;
5. Сообщение информации пользователю: система оснащена множеством средств для взаимодействия с пользователем: UIDisplay, UIWinfow и др.

## Выбор технологии

В ходе анализа мы имеем две равноценные мобильные операционные системы: Android и iOS, которые предоставляют полный функционал для разработки программного решения, и выбор не кажется настолько однозначным. Для этого придется рассмотреть дополнительные аспекты как со стороны операционных систем, так и рынка мобильных устройств, что даст нам представление о пользователях, покупаемости товара с операционной системой, под которую будет разрабатываться программное решение. Во внимание возьмем следующие аспекты: количество устройств на каждой операционной системе, размер магазинов приложений, сообщество разработчиков, средства разработки.

1. Популярность системы: по результатам статистического анализа на данный момент более 70% мобильных устройств используют Android в качестве операционной системы и только около 29% iOS [3];
2. Магазины приложений: по данным на март 2020 года в официальном магазине Android – Play Store находится 2,87 миллиона доступных приложений [4]. Против 2,2 миллионов в Apple App Store [5]. В расчет берутся только основные площадки. Хотя у Android имеются и сторонние;
3. Сообщество разработчиков: по данным опроса за 2019 год, одного из самых авторитетных ресурсов для разработчиков Stackoverflow, который ежегодно опрашивает пользователей собирая всевозможную статистику, на языке Java, технологии на котором ведется разработка под андройд, работает более 40% программистов, когда на Swift, языке для разработки под iOS 6,6% разработчиков [6];
4. Средства разработки: Android имеет более широкое распространение среди пользователей и не требует устройства на с операционной системой от Apple, что сильно сужает число потенциальных разработчиков, так как более 77% пользователей в качестве основной операционной системы используют Windows OS, которая поддерживает разработку под Android и лишь 17% MacOS [7], так же не стоит забывать про аудиторию пользователей Linux, которая так же составляет достаточно весомый процент, которая так же поддерживается разработкой под Android систему.

## Вывод

Для разработки мобильного приложения была выбрана операционная система Android, поскольку имеет более низкий порог вхождения: не требует специализированных под систему устройств, является системой с открытым исходным кодом, а также пользуется большим спросом среди покупателей смартфонов, как на самой ОС, так и на Chrome ОС, что также расширяет потенциальную аудиторию пользовательских устройств.

# ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ПРОГРАММЫ ДЛЯ ОБЕСПЕЧЕНИЯ ПОМОЩИ СТУДЕНТАМ ПО ОРИЕНТАЦИИ ВНУТРИ ЗДАНИЙ

## Архитектура программного средства

Архитектура – это набор значимых решений по поводу организации системы программного обеспечения, набор структурных элементов и их интерфейсов, при помощи которых компонуется система, вместе с их поведением, определяемым во взаимодействии между этими элементами, компоновка элементов в постепенно укрупняющиеся подсистемы, а также стиль архитектуры который направляет эту организацию – элементы и их интерфейсы, взаимодействия и компоновку [8].

### Обзор архитектурных шаблонов

Для удовлетворения проектируемой системы атрибутам качества применяются различные архитектурные шаблоны [9].

Примеры архитектурных шаблонов:

1. Mediator – посредник;
2. Observer – наблюдатель;
3. MVC – модель представление контроллер;
4. Layered – многоуровневая архитектура;
5. MVP – модель- представление презентер;
6. MVVM – модель-представление-modelView.

Посредник – шаблон, который применяется для структуризации распределенных систем с несвязными компонентами. Данный шаблон используется, если в системе присутствует большое количество модулей, и их прямое взаимодействие друг с другом становится слишком сложным. Для решения проблемы вводится посредник (например, шина данных), по которой модули общаются друг с другом. Таким образом, повышается функциональная совместимость модулей системы. Все недостатки вытекают из наличия посредника: он понижает производительность, его недоступность может сделать недоступной всю систему, он может стать объектом атак и узким местом системы.

Наблюдатель – шаблон, суть которого заключается в том, что если пользователь вносит изменения в модель посредством одного представления, то эти изменения отражаются и во всех остальных представлениях. В этом случае представление выполняет роль «наблюдателя» за моделью: как только модель будет изменена, представление генерирует соответствующее событие и все остальные представления обновляют свое содержимое. Недостатком данного шаблона является то, что реализовать шаблон в многопоточном, не имеющем состояний слое служб весьма затруднительно.

Многоуровневая архитектура – архитектура, в которой систему можно разбить на уровни, находящиеся на определенных уровнях абстракции. На диаграмме уровни изображаются один над другим. Каждый уровень может вызывать только уровень на один ниже него. Таким образом, разработку каждого уровня можно вести относительно независимо, что повышает модифицируемость системы. Недостатками данного подхода являются усложнение системы и снижение производительности.

MVC – архитектурный шаблон, который создан для разделения приложения на урони, которые при необходимости собственного изменения не потребуют изменения других уровней.

Модель – уровень данных, отвечающий за управление бизнес-логикой и управление сетью или базой данных API.

Представление – уровень пользовательского интерфейса - визуализация данных из модели.

Контроллер – логический уровень, получает уведомления о поведении пользователя и обновляет модель по мере необходимости.

Таким образом, это означает, что и контроллер, и представление зависят от модели: контроллер для обновления данных, представление для получения данных. Но самое важное, что модель отделена и может тестироваться независимо от пользовательского интерфейса [10].

MVP – это архитектурный шаблон пользовательского интерфейса, разработанный для облегчения автоматического модульного тестирования и улучшения разделения проблем в логике представления:

Модель представляет собой интерфейс определения данных, которые будут отображаться или иным образом действовал на в пользовательском интерфейсе.

Представление представляет собой пассивный интерфейс, который отображает данные (модель) и направляет пользовательские команды (события) презентатору для воздействия на эти данные.

Перзентор действует на модели и представление. Он извлекает данные из репозиториев (модель) и форматирует их для отображения в представлении.

Обычно реализация представления создает конкретный объект презентатора, предоставляя ссылку на себя [11].

MVVM – расшифровывается как Model, View, ViewModel.

Модель: содержит данные приложения. Он не может напрямую общаться с View. Как правило, рекомендуется выставлять данные в ViewModel через Observables.

Представление: представляет пользовательский интерфейс приложения, лишенный какой-либо логики приложения. Это наблюдает за ViewModel.

ViewModel: действует как связь между моделью и представлением. Он отвечает за преобразование данных из модели. Он предоставляет потоки данных для просмотра. Он также использует обратные вызовы для обновления представления. Он будет запрашивать данные из модели.

## Выбор архитектурного шаблона

В качестве архитектуры была выбрана многослойная архитектура.

Архитектура заключается в том, чтобы поставить классы в рамки уровней и настроить общение через них.

Уровень пользовательского интерфейса: здесь мы можем поместить действия, фрагменты, адаптеры и любые другие классы, связанные с пользовательским интерфейсом.

Слой домена: здесь мы можем разместить бизнес-логику нашего приложения. Код, который является эксклюзивным для нашего приложения.

Уровень данных: связанные с базой данных классы и любые другие системные службы.

Цель многоуровневой архитектуры – создать границы, в которых один слой взаимодействует только со смежным, и это происходит в одном направлении. Каждый слой взаимодействует с соседним слоем, и поток идет в одном направлении.

Это означает, что уровень пользовательского интерфейса не может напрямую взаимодействовать с уровнем данных. Эта изоляция помогает нам минимизировать влияние изменений. Хотя бывают случаи, когда у нас может быть открытый слой, который можно обойти, но мы более или менее следуем взаимодействию со смежным слоем в одном направлении. Классическое представление многоуровневой архитектуры приводится на рисунке 3.1.

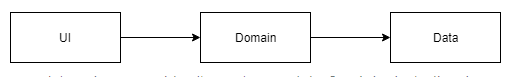


Рисунок 3.1 – Схема классического понимания многоуровневой архитектуры

## Проектирование базы данных

Представление данных в программном средстве будет осуществляться на уровне базы данных. Для представления данных в приложении необходимо спроектировать логическую модель базы данных.

База данных – совокупность данных, хранимых в соответствии со схемой данных, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных. Таким образом, база данных – совокупность сущностей предметной области, выраженных в таблицах базы данных, а также связи между ними. В данном проекте будет использоваться реляционная база данных в лице SQLite.

Что из себя представляет реляционная модель, она организует данные в одну или несколько таблиц (или «отношений») столбцов и строк с уникальным ключом, идентифицирующим каждую строку. Строки также называются записями или кортежами [13] Столбцы также называются атрибутами. Как правило, каждая таблица или отношение представляет один «тип объекта» (например, клиент или продукт). Строки представляют экземпляры объекта такого типа, а столбцы представляют значения, приписанные этому экземпляру.

SQLite — это внутри процессная библиотека, которая реализует автономный, без серверный, транзакционный механизм базы данных SQL с нулевой конфигурацией. Это база данных с нулевой конфигурацией, что означает, что, как и другие базы данных, вам не нужно настраивать ее в своей системе. Движок SQLite не является автономным процессом, как другие базы данных, вы можете связать его статически или динамически в соответствии с вашими требованиями к вашему приложению. SQLite получает доступ к своим файлам хранения напрямую.

Плюсы использования SQLite:

1. SQLite не требует отдельного серверного процесса или системы для работы (без сервера);
2. SQLite поставляется с нулевой конфигурацией, что означает отсутствие необходимости в настройке или администрировании;
3. Полная база данных SQLite хранится в одном межплатформенном файле на диске;
4. SQLite очень маленький и легкий, менее 400 КБ полностью настроен или менее 250 КБ с опущенными дополнительными функциями;
5. SQLite является автономным, что означает отсутствие внешних зависимостей;
6. Транзакции SQLite полностью совместимы с ACID, что обеспечивает безопасный доступ из нескольких процессов или потоков;
7. SQLite поддерживает большинство функций языка запросов, представленных в стандарте SQL92 (SQL2);
8. SQLite написан на языке ANSI-C и предоставляет простой и удобный API;
9. SQLite доступен в UNIX, в частности Android.

Рассмотрим таблицы, которые будут представлены в базе данных для данного дипломного проекта. Таблицы представлены в таблице 3.1.

Таблица 3.1 – Таблицы базы данных

|  |  |
| --- | --- |
| Имя таблицы | Описание |
| Building | Таблица, представляющая корпуса БНТУ. |
| Time | Таблица представляет собой набор временных отрезков, которые отражают временные  отрезки университетских занятий. |
| Work\_week | Таблица, содержащая в себе названия дней недели. |
| Calendar | Таблица, содержащая в себе дни месяца, разделенные на четные и нечетные по неделям. |
| Schedule | Таблица, содержащая расписание занятий университета. |

Схема базы данных представлена на рисунке 3.2. Она состоит из 5 таблиц, отражающие в себе отдельные независимые части системы.

A screenshot of a cell phone

Description automatically generated

Рисунок 3.2 – Схема базы данных приложения

Наиболее важной таблицей в данной базе является таблица «schedule», то есть таблица расписания, рассмотрим ее атрибуты, которые представлены в таблице 3.2 .

Таблица 3.2 – Атрибуты таблицы «schedule»

|  |  |  |
| --- | --- | --- |
| Атрибут | Тип атрибута | Описание |
| id | Integer | Уникальный идентификатор записи, являющийся первичным ключом |
| day | Integer | Ссылка на день недели в которому относится пара. |
| place | Integer | Ссылка на место проведения пары: корпус и аудитория. |

Продолжение таблицы 3.2

|  |  |  |
| --- | --- | --- |
| 1 | 2 | 3 |
| teacher | Text | Преподаватель ведущий занятие. |
| lesson | Text | Название предмета. Занятия, по которому проводятся в данной аудитории. |
| group | Integer | Группа, для которой проводится занятие. |
| week\_number | Integer | Порядок недели: четная, нечетная. По-умолчанию имеет значение «0», но может иметь значения «1» в зависимости от порядка недели. |
| time | Integer | Ссылка на временной промежуток проведения пары. |

Рассмотрим связи, которые имеет данная таблица. Таблица имеет три внешних ключа, все они ссылаются на другие таблицы, которые создают полноту представления о занятии. На рисунке видно, что поле «time» является внешним ключом к таблице «time», «place» внешний ключ к таблице «building», поле «day» в свою очередь является внешним ключом к таблице «work\_week».SQL-скрипт создания базы данных и остальных таблиц продемонстрирован на рисунке 3.4.

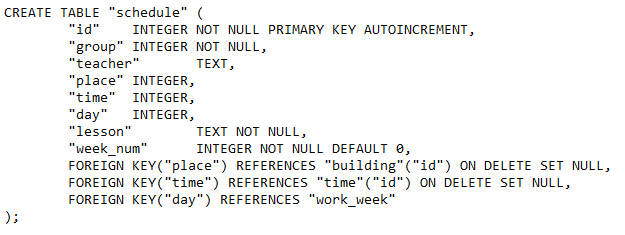


Рисунок 3.3 – SQL-скрипт создания таблицы «schedule»

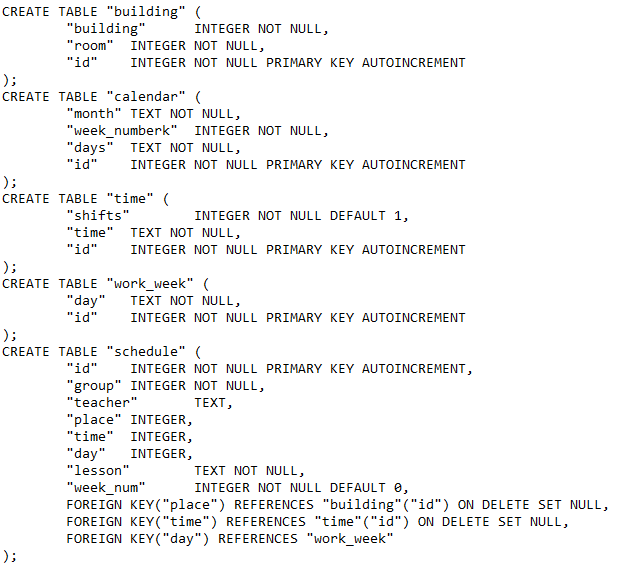


Рисунок 3.4 – SQL-скрипт создания базы данных и ее таблиц

## Проектирование и разработка приложения

Для того что бы начать разрабатывать задачу, основанную на QR-сканнере, необходимо создать собственную реализацию QR-сканнера или использовать уже готовую.

В данном дипломном проекте будет использоваться «barcodescanner:zxing» – это небольшая библиотека, для сканирования BAR и QR кодов. В приложение библиотека будет добавлена по средством вставки новой записи в build.gradle файл. Так же добавим в проект зависимость для работы с JSON файлами. JSON будет использоваться как формат в котором будет храниться информация в QR-коде, это обосновано тем, что формат достаточно прост в использовании и обработка данных этого формата автоматизирована и удобна. Зависимости, использованные для данного ДП продемонстрированы на рисунке 3.5.



Рисунок 3.5 -Зависимости для QR-сканнера и работы с JSON

Логическое моделирование позволяет определить черты, поведение и функционал проектируемой программы. Диаграмма последовательности отражает основные этапы работы приложения, что позволяет при разработке приложения понять какие задачи будет решать приложение и из каких подзадач оно состоит, диаграмма последовательности изображена на рисунке 3.6.

A screenshot of a cell phone

Description automatically generated

Рисунок 3.6 – Диаграмма последовательности

### Структура проекта

Рассмотрим структуру проекта, какие пакеты в нем содержаться и за что каждый из них отвечает, структура изображена на рисунке 3.7.

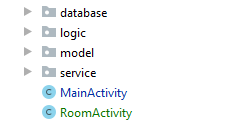


Рисунок 3.7 – Структура проекта

Так как проект строится по многоуровневой архитектуре, то в свою очередь имеет следующие слои:

1. Слой «database» отвечает за соединение с базой данных и запросы для получения данных;
2. Слой «logic» содержит в себе основную логику приложения, отвечающую за его особенности;
3. Слой «model» содержит в себе классы, описывающие упрощенную модель сущностей реального мира, для которых разрабатывается приложение;
4. Слой «service» слой, отвечающий за вызов необходимой логики для обработки запроса вышестоящего уровня;
5. Слой «view» неявный слой, отвечающий за представление и работу интерфейса, представленный классами MainAcivity и RoomActivity;

Направление зависимостей между слоями идёт от слоя представления к слою доступа к данным. В идеальной ситуации каждый слой зависит только от следующего слоя: слой представления зависит от сервисного слоя (например, представление зависит от контроллера), сервисный слой зависит от слоя бизнес-логики (например, контроллер зависит от бизнес-сервиса), а слой бизнес-логики — от слоя доступа к данным (например, бизнес-сервис зависит от репозитория). При этом компоненты бизнес-слоя могут зависеть от других компонентов бизнес-слоя, тогда как в других слоях аналогичные зависимости нежелательны (например, зависимость одного репозитория от другого). Так же нежелательны зависимости в обратном направлении (бизнес-слой не должен зависеть от сервисного слоя) и зависимости между слоями, не являющимися соседними (сервисный слой не должен зависеть от слоя доступа к данным, например).

Схема классов приложения представлена на рисунке 3.8.

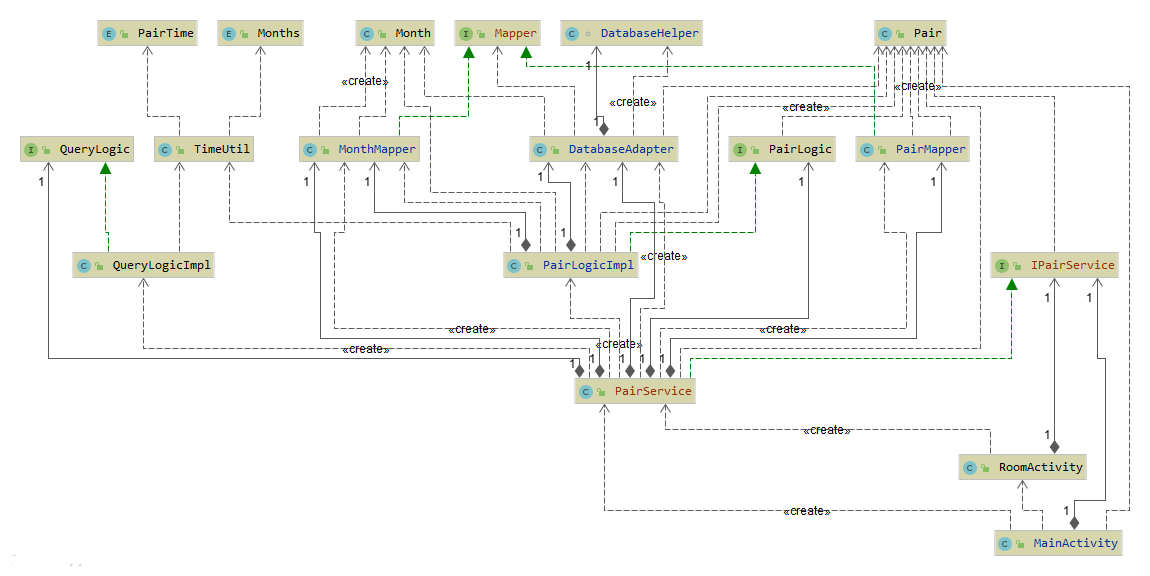


Рисунок 3.8 – Диаграмма классов приложения.

### Слой database

Для того что бы использовать уже созданную базу под данный проект необходимо создать в проекте папку «assets», в которую поместим файл базы данных «scheduled.db» как показано на рисунке 3.9.

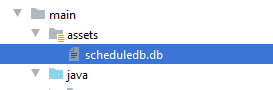


Рисунок 3.9 – Ресурсная папка assets

#### DatabaseHelper

Для упрощения работы с базами данных SQLite в Android нередко применяется класс SQLiteOpenHelper. Для использования необходимо создать класса-наследник от SQLiteOpenHelper, переопределив как минимум два его метода:

1. onCreate: вызывается при попытке доступа к базе данных, но когда еще эта база данных не создана
2. onUpgrade. вызывается, при обновлении схемы базы данных.

Рассмотрим класс DatabaseHelper, который отвечает за то, что бы приложение могло подключиться к базе. Поля данного класса проиллюстрированы на рисунке 3.10. Описание полей приведено в таблице 3.3.

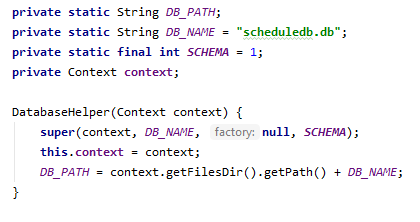


Рисунок 3.10 – Поля класса DatabaseHelper и его конструктор

Таблица 3.3 – Поля класса DatabaseHelper

|  |  |
| --- | --- |
| Имя поля | Назначение поля |
| String DB\_PATH | Строковое поле отвечает за полный путь к базе данных. |
| String DB\_NAME | Строковое поле хранит имя файла базы данных, к которому приложение будет обращаться.. |
| Int SCHEMA | Целочисленное поле отражает версию базы данных. |

По умолчанию база данных будет размещаться во внешнем хранилище, выделяемом для приложения в папке /data/data/[название\_пакета]/databases/, и чтобы получить полный путь к базе данных в конструкторе используется выражение: context.getFilesDir().getPath() + DB\_NAME. Методы класса отражены в таблице 3.4.

Таблица 3.4 – Методы класса DatabaseHelper

|  |  |
| --- | --- |
| Метод класса | Описание |
| onCreate(SQLiteDatabase db) | Стандартный метод, отвечающий за создание базы данных, так как в проекте используется уже готовая база, метод не получает реализации |
| onUpgrade(SQLiteDatabase db, int oldVersion, int newVersion) | Стандартный метод, отвечающий за обновление базы данных, так как база не должна обновляться в силу особенности задачи то метод не получает имплементации. |
| Create\_db() | Дополнительный метод, которой побайтово копирует базу данных из папки assets в то  место, которое указано в переменной DB\_PATH. |
| SQLiteDatabase open() throws SQLException | Дополнительный метод, цель которого вернуть объект базы данных с флагом OPEN\_READONLY. |

#### DatabaseAdapter

Класс DatabaseAdapter представляет собой более высокоуровневый класс, отвечающий за запросы к базе данных. Описание полей приведено в таблице 3.5. Строковое поля класса содержат в себе SQL-скрипты для взаимодействия с таблицами базы данных. Содержимое полей представлено на рисунке 3.11.

Таблица 3.5 – Поля класса databaseAdapter

|  |  |
| --- | --- |
| Имя поля | Описание |
| DatabaseHelher dbHelper | Поле, представляющее собой ссылку на объект класса Databasehelper и доступа к его функциональности. |
| SQLiteDatabase database | Поле, представляющее собой ссылку на объект базы данных. |
| String GET\_PAIR | Строковое поле, представляющее SQL запрос к базе данных на выбоку занятия с подходящими параметрами. |
| String GET\_ROOM\_PAIRS | Строковое поле, представляющее SQL запрос к базе данных на выбоку занятий для конкретной аудитории. |
| String GET\_MONTH | Строковое поле, представляющее SQL запрос к базе данных на выбоку дней месяца. |

Рассмотрим поле GET\_PAIR подробнее. Данный SQL-запрос выбирает все поля из таблицы schedule, building, time и work\_week объединяя их в общую таблицу по средствам join, далее идет проверка на соблюдение условий, что бы данные, которые были вставлены вместо знака «?», это будет сделано автоматически, когда строку будут использовать в качестве запроса, соответствовали полям таблиц, результирующий список этого запроса будет возвращен базой данных. Каждый строковый запрос будет вызван в методах класса. Методы класса отражены в таблице 3.6.

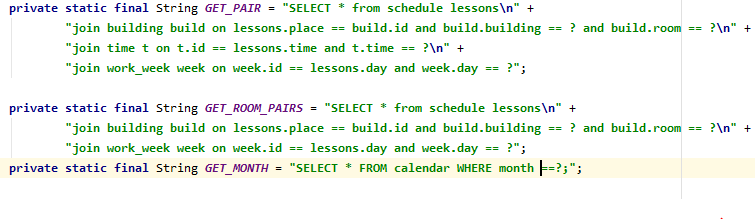


Рисунок 3.11 – Строковые поля содержащие запросы к базе

Таблица 3.6 – Методы класса DatabaseАдаптерr

|  |  |
| --- | --- |
| Метод класса | Описание |
| List<Pair> getPair (Mapper pairMapper, int | Метод возвращает список объектов Pair при |
| building, int room, String time, String day) | запросе на пару в данный момент врени. |
| List<Pair> getRoomPair(Mapper pairMapper, int building, int room, String day) | Метод возвращает список объектов Pair при запросе на занятия в конкретной аудитории. |
| List<Month> getMonth(Mapper monthMapper, String month) | Возвращает список объектов типа месяц |
| List<Object> handleRawData(String[] params, Mapper mapper, String query) | Приватный метод. обрабатывающий данные полученные из базы и приводя их к понятному для приложения виду. |

### Слой model

Уровень, описывающий основные классы манипуляций программы, то есть объекты реального мира в программном представлении и вспомогательные объекты для работы с ними. Классы данного уровня представлены в таблице 3.7. Данный уровень не содержит в себе никакой бизнес логики.

Таблица 3.7 – Классы уровня model

|  |  |
| --- | --- |
| Класс | Описание |
| Pair | Класс, описывающий программную модель университетской пары. |
| Month | Класс, описывающий программную модель месяца, содержащую дни и их порядок относительно чередования четных и нечетных недель. |
| PairTime | Класс-перечисление, представляющий  ограниченное количество временных промежутков, соответствующих времени проведения занятий. |
| Months | Класс-перечисление, представляющий месяца года и их названия на русском и английском языке, существует для единообразие работы приложения как на устройствах с русским языком системы, так и английским. |
| Mapper | Интерфейс, задающий поведение для сборки набора данных в единый объект. |
| PairMapper | Класс-наследник Mapper, который собирает разделенные данные о занятии в единый объект, для дальнейшего удобства работы с ними. |
| MonthMapper | Класс-наследник Mapper, который собирает разделенные данные о месяце в единый объект, для дальнейшего удобства работы с ними. |

#### Класс Pair

Университетское занятие основной элемент манипуляции в данном приложении, оно представлено данным классом.

Таблица 3.8 – Поля класса Pair

|  |  |
| --- | --- |
| Имя поля | Описание |
| Int room | Целочисленное поле, содержащее информацию об аудитории, в которой проводится занятие. |
| String day | Строковое значение, содержащее информацию о дне проведения занятия |
| String time | Строковое значение, содержащее время проведение занятия |
| String pair | Строковое поле. Содержащее информацию о названии предмета. |
| String teacher | Строковое поле, содержащее информацию о преподавателе. |
| Int group | Целочисленное поле. Содержащее информацию о группе, с которой проводится занятие. |
| Int weekNumber | Целочисленное поле, отражающее порядок недели, в которую данное занятие проводится. |

Класс так же содержит шаблонный код по заполнению полей, который не несет в себе никакой бизнес логики.

#### Класс Month

Класс, описывающий программную модель месяца, содержащую дни и их порядок относительно чередования четных и нечетных недель.

Таблица 3.9 – Поля класса Month

|  |  |
| --- | --- |
| Имя поля | Описание |
| String name | Строковое поле представляющее имя месяца. |
| List<String> days | Поле спискового типа, которое содержит дни месяца. Относящиеся к одном из типов недель: четному или нечетному. |
| Int order | Целочисленное поле представляющие порядок недели: 0 -нечетная,1-четная. |

Класс так же содержит шаблонный код по заполнению полей, который не несет в себе никакой бизнес логики.

#### Класс-перечисление PairTime

Классы перечисления существуют для хранения ограниченной коллекции данных, в данном случае времени занятий.

Данный класс существует для того, чтобы приложение умело ориентироваться в каком временном отрезке находится пользователь, он содержит в себе ограниченный набор значений, которые соответствуют всем временным отрезкам занятий в университете. Содержимое класса отображено на рисунке 3.12. Каждое значение, которое содержит этот класс изначально проинициализировано и не может быть изменено в процессе работы программы, это сделано для того, чтобы получаемые значения всегда оставались корректными. Поля данного класса отражены в таблице 3.13.

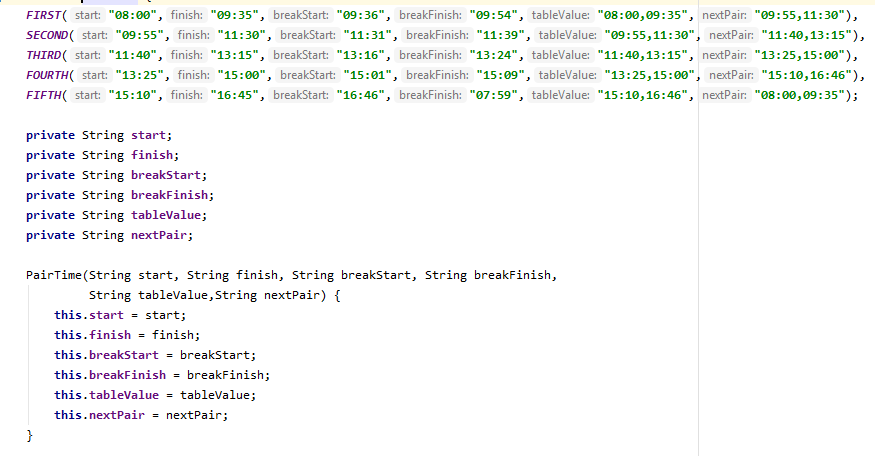


Рисунок 3.12 -Класс-перечисление PairTime

Таблица 3.10 – Поля класса PairTime

|  |  |
| --- | --- |
| Имя поля | Описание |
| String start | Строковое поле, содержащее время начала пары. |
| String finish | Строковое поле, содержащее время конца пары. |
| String breakStart | Строковое поле, содержащее время начала перерыва. |
| String breakFinish | Строковое поле, содержащее время конца перерыва. |
| Sting tableValue | Строковое поле, содержащее значение, соответствующее табличному значению  данного временного промежутка |
| String nextPair | Строковое поле, содержащее время начала следующей пары. |

#### Класс Months

Так как приложение работает с устройством пользователя, то разработчик не может предсказать на каком языке работает система пользователя. Поэтому был предусмотрен вариант второго языка, а именно английского, для этого и необходим данный класс, если приложение получит данные с устройства вне зависимости от того русскоязычная система или англоязычная, оно сможет понять что за данные к нему пришли. Содержимое Класса изображено на рисунке 3.13.

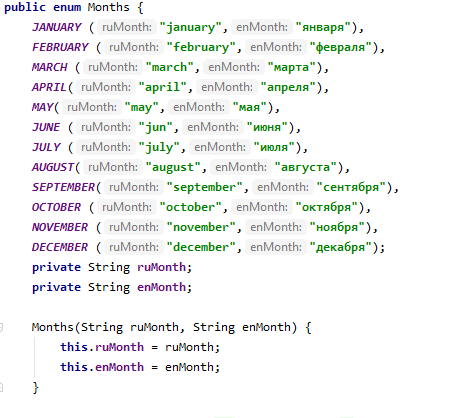


Рисунок 3.13 – Класс Months

#### Интерфейс Mapper и его реализация в классе PairMapper

Данные классы необходимы приложению для преобразования данных, полученных из базы, в объекты для удобства взаимодействия с ними в программе.

Интерфейс Mapper содержит в себе 1 метод «collectObject» общего типа, метод специально имеет общий тип что бы не дублировать один и тот же код вызова метода на разных объектах, интерфейс изображен на рисунке 3.14.

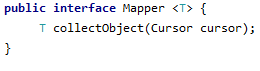


Рисунок 3.14 – Интерфейс Mapper

Рассмотрим одну из реализаций интерфейса на примере класса PairMapper. Реализация метода «collectObject» изображена на рисунке 3.15.

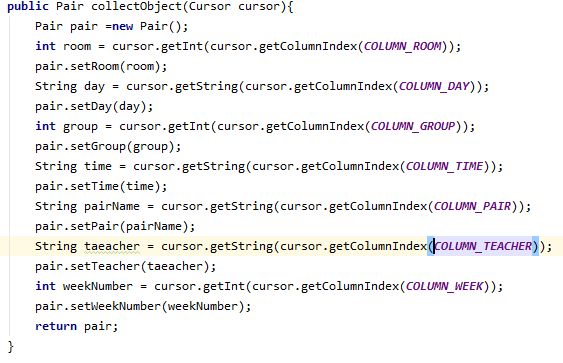


Рисунок 3.15 – метод collectObject в классе PairMapper

Метод принимает на вход объект типа Cursor, этот объект приходит в качестве ответа БД и содержит в себе поля, полученные в результате выборки. В начале метода создается новый пустой объект Pair и после из объекта coursor, который возвращается после запроса к базе данных, данный объект все поля полученные при выборке из таблиц, по имени поля извлекаются значения и добавляются в объект, поле того, как все поля объекта были заполнены, он возвращается методом. Названия полей изображены на рисунке 3.16.

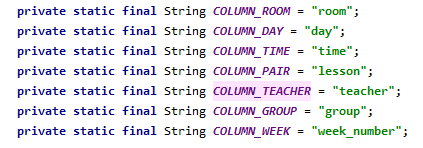


Рисунок 3.16 – Поля содержащие названия колонок базы данных

### Слой logic

Данный слой содержит в себе логику работы со временем и помогает с организацией запросов. Он необходим приложению для ориентирования во времени, чтобы определить в каком промежутке находится устройство во время сканирования QR-кода. Классы слоя представлены в таблице 3.11.

Таблица 3.11 – Классы уровня logic

|  |  |
| --- | --- |
| Класс | Описание |
| PairLogic | Интерфейс, содержащий метод, который должен сделать выбор в случае, когда база данных вернула несколько занятий в случае, когда ожидается одно |
| PairLogicImpl | Класс-реализация интерфейса PairLogic. |
| QueryLogic | Интерфейс содержащий метод для формирования из информации, полученной после сканирования, полного набора параметров запроса |
| QueryLogicImpl | Класс реализация интерфейса QueryLogic |
| TimeUtil | Класс, содержащий методы по работе со временем устройства. |

#### Интерфейс PairLogic и его реализация PairLogicImpl

Данный интерфейс и его реализация существуют для разрешения нестандартных ситуаций. На данном этапе у приложения существует одна нестандартная ситуация, когда после запроса к базе данных вернется несколько записей, тогда придется решать какая из них корректна для того момента времени, но создание интерфейса объясняется тем, что возможно при дальнейшей модификации и расширении приложения будет принято решение о создании новой логики обработки в особенных случаях и понадобится несколько вариантов обработки, или же добавлении новых поведений. Интерфейс обладает одним методом getCurrentPair, данный метод изображен на рисунке 3.17.

Данный метод рассматривает три ситуации, которые могут возникнуть:

1. Когда длина списка занятий равна единице, это значит, что в кабинете в данный день и в данное время проводится одна пара и ее можно вернуть;
2. Когда длина списка занятий равна нулю, это значит, что в данный момент ни одного занятия в аудитории не проводится, метод вернет пустой объект типа Pair;
3. Когда список состоит из двух пар, данная ситуация возникнет, если в кабинете в четную и нечетную неделю ведутся разные занятия, в этом случае метод сделает дополнительный запрос в базу данных и получит список дат на текущий месяц, разделенных на два списка: список с четными днями и список с нечетными днями, после перебирая каждый список будет сравнивать с датой дня, в который происходит сканирование, поле обнаружения совпадений метод получает идентификатор, указывающий к какой неделе относится день, после сравнивает с идентификатором занятий и возвращает одно из двух.

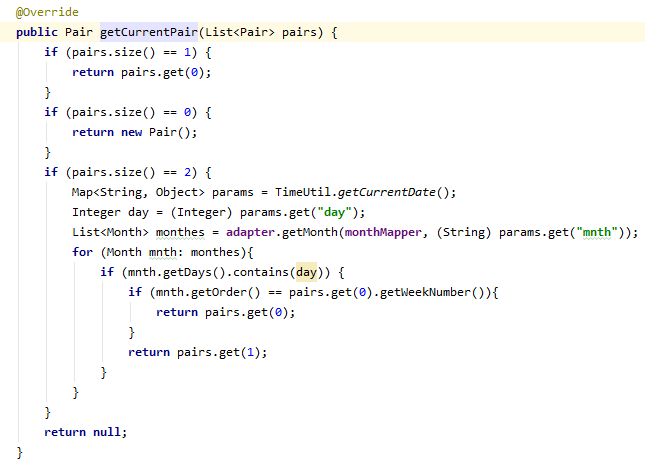


Рисунок 3.17 – Реализация метода getCurrentPair

#### Интерфейс QueryLogic и его реализация QuerylogicImpl

Данный интерфейс создан для того, чтобы задать поведение по сборке параметров запроса в единую коллекцию. В дальнейшем при расширении программы понадобятся новые поведения или новые реализации старого поведения для этого существует данный интерфейс. На данном этапе он имеет один метод collectQueryParams, который обрабатывает начальные данные, полученные при сканировании QR-кода, метод оборачивает данные в JSON формат, для большего удобства, и извлекает необходимые поля room и building. Следующим шагом запрашивает дополнительные данные из класса TimeUtil дополняя запрос необходимыми параметрами, после оборачивает их в коллекцию ключ-значение и возвращает ее. Реализация метода в классе QueryLogicImpl продемонстрирована на рисунке 3.18.

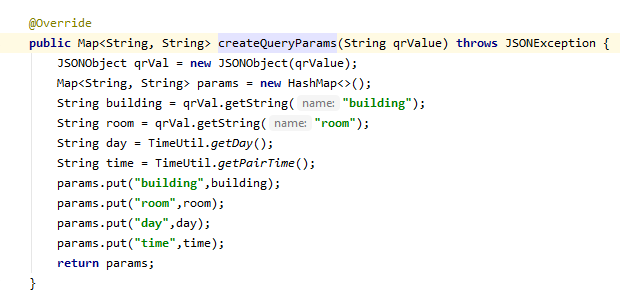


Рисунок 3.18 – реализация метода createQueryParams в классе QueryLogicImpl

#### Класс TimeUtil

Данный класс является основным средством манипуляций со временем именно он содержит методы, которые возвращают необходимые данные: время, месяц, день недели. Методы данного класса представлены в таблице 3.12.

Таблица 3.12 - Методы класса TimeUtil

|  |  |
| --- | --- |
| Имя метода | Описание |
| getDay | Возвращает день недели |
| getCurrentTime | Возвращает время с устройства  пользователя |
| getPairTime | Возвращает время в формате «час:минута» |
| getCurrentDate | Возвращает имя месяца и день в числовом варианте |
| getMonthForInt | Возвращает численное значения месяца |

Рассмотрим ключевые метод: getPairTime один из основных методов данного класса. Метод getPairTime изначально задает формат времени, после получает время в данный момент и приводит к нужному формату, после поля класса PairTime, который содержит в себе временные отрезки всех занятий, сравниваются с текущим значением времени и если условие, при котором момент сканирования стоит позже начала занятия и раньше его конца, соблюдается, тогда приложение узнает в каком конкретном промежутке сейчас находится пользователь. Метод использует приватный метод getCurrentTime для получения необходимых значений времени, но так как данный метод возвращает время в неподходящем формате , его реализация была отделена от метода getPairTime, так как если бы метод объединял в себе два правило один метод-одна ответственность было бы нарушено. Метод показан на рисунке 3.19.

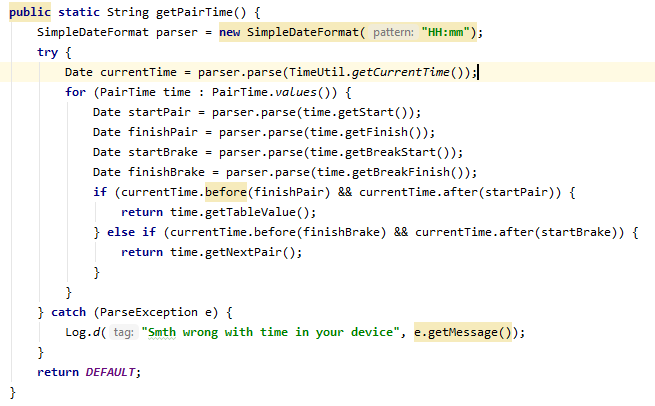


Рисунок 3.19 – Метод getPairTime класса TimeUtil

Второй основной метод это метод getCurrentDate, данный метод существует для того что бы определить занятие на четной и нечетной неделе, он вернет день и название месяца в момент использования приложение, эти данные будут сопоставлены с данными полученными из таблицы и отталкиваясь от этого будет приниматься решение. Метод получает численно значение календарного дня, после запрашивает месяц у устройства, использующего приложение, и сопоставляет с полями класса Months, который содержит меня всех месяцев на двух языках, и когда находится совпадающее значение, то метод возвращает коллекцию, содержащую дату и месяц. Метод использует приватный метод класса getMonthForInt чтобы получить числовое значение месяца, а по нему получить строковое, в силу особенностей языка разработки получение имени месяца пришлось разбить на эти 2 этапа. Метод проиллюстрирован на рисунке 3.20.

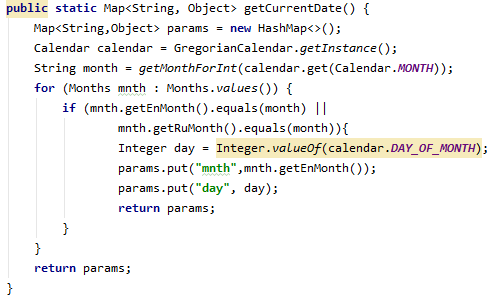


Рисунок 3.20 – Метод getCurrentDate класса TimeUtil

### Слой service

Данный слой служит для того, что б связывать всю систему воедино и упорядочивает вызовы классов, выступая управляющим звеном. Он содержит один интерфейс и один класс: интерфейс PairService и его реализация PairServiceImpl. Сервис содержит два метода getPair и getPairView.

Метод getPair предназначен для сообщения прямого результата сканирования кода, он запускает всю цепочку вызовов классов. Метод изображен на рисунке 3.21.

Изначально метод вызывает объект класса QueryLogic, который возвращает набор параметров запроса, после вытаскиваются необходимые параметры по имени, далее объекту класса DatabaseAdapter передается весь набор параметров и получается список занятий, в конце мы передаем список параметров объекту класса PairLogic, который выбирает результирующее занятие из списка

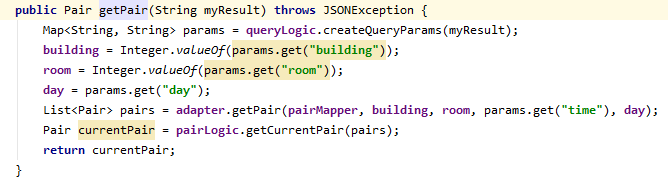


Рисунок 3.21 – Метод getPair класса PairServiceImpl

Метод getPairView служит для того, чтобы вернуть список занятий в аудитории за весь день. Метод изображен на рисунке 3.22.

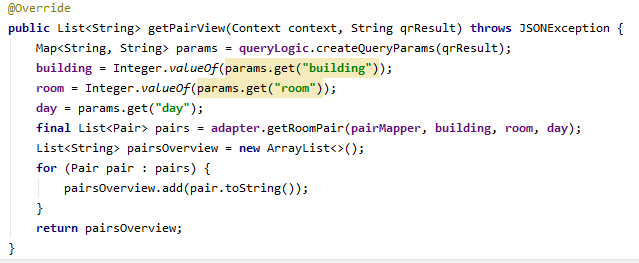


Рисунок 3.22 – метод getPairView класса pairServiceImpl

Метод работает схожим образом с методом getPair, основное отличие данного метода, что приложение не передает конкретное время, а лишь день и место проведения занятий, после чего в результирующем списке мы получаем все занятия проходящие в этой аудитории за день, после чего переводим каждый объект Pair в строковое представление и добавляем в лист строк , который является результатом работы данного метода.

### Слой view

Классы слоя view отвечает за отображение пользовательского интерфейса. В данном слое находится два класса:

1. Класс MainActivity, отвечающий за отображение сканера и вывод результата сканаирования на экран;
2. Класс RoomActivity, отвечающий за отображение всех занятий на экране по нажатию кнопки.

Рассмотрим основные методы класса MainActivity:

1. onCreate – метод отвечающий за инициализацию основных компонентов работы приложения после его старта, так что он проверяет даны ли права на использование камеры приложению;
2. onResume – метод отвечающий за проверку прав приложения на использования камеры после того как приложение возобновило работу после паузы (блокирока экрана и его последующая разблокировка);
3. onRequestPermissionsResult – метод отвечающий за реакцию приложение на разрешение или отклонение прав приложения на использование камеры;
4. handleResult – переопределенный метод обработки результатов сканирования кода, который запускает сервис на обработку запроса.

Метод onCreate представлен на рисунке 3.23.

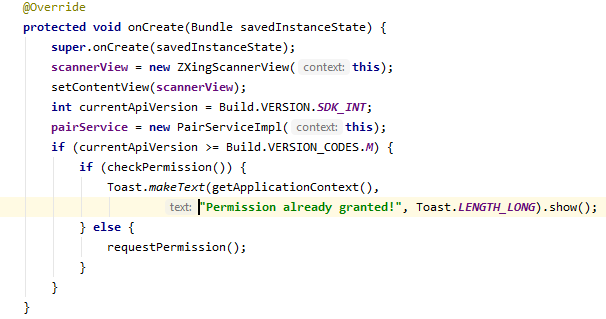


Рисунок 3.23 – Метод onCreate класса mainActivity

Метод onCreate вызывается при создании или перезапуска активности. Система может запускать и останавливать текущие окна в зависимости от происходящих событий. Внутри данного метода настраивают статический интерфейс активности. Инициализирует статические данные активности, связывают данные со списками и так далее. Связывает с необходимыми данными и ресурсами. Задаёт внешний вид через метод setContentView. Первоначально метод создает новый объект класса ZXingScannerView, данный класс отображает рамку сканнера на экране поле того как его установят как объект отображения, строчка setContentView(scannerView) выполняет данную функцию, результат отражает рисунок 3.24. Далее метод проверяет были ли приложению предоставлены права, если нет он запросит права у пользователя.

А при вызове метода onResume activity переходит в состояние Resumed, а пользователь может с ней взаимодействовать. И, собственно, activity остается в этом состоянии, пока она не потеряет фокус, например, вследствие переключения на другую activity или просто из-за выключения экрана устройства. Метод onResume существует для ситуаций, когда работа приложения была приостановлена, но не завершена, а после восстановления он проверит не изменилось ли окружение системы в отношении данного приложения. Конкретно он проверит предоставлены ли права и в случае положительного ответа запустит камеру снова, в противном случае запросит права повторно. Метод изображен на рисунке 3.25.

A screen shot of a computer

Description automatically generated

Рисунок 3.24 – Отображение интерфейса для захвата QR-кода.

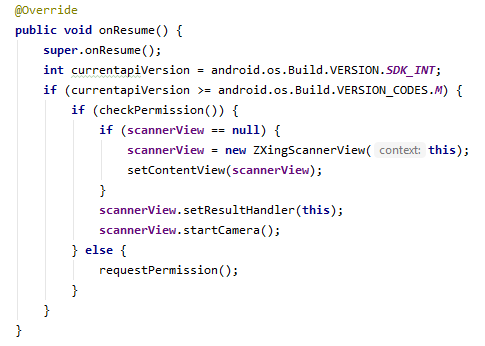


Рисунок 3.25 – Метод onResume класса MainAcivity

Метод handleResult предназначен для обработки информации, которая была отсканирована. В данном проекте он создает диалоговое окно, в котором отображается результат сканирования и решает, что будет отображено в нем. Первая часть метода, создает объект конструктора диалоговых окон, устанавливая заголовок диалогового окна, и две кнопки. Кнопка «Занятия в аудитории» вызывает другой элемент пользовательского отображения и инициирует его старт, кнопка «Ок» возвращает пользователя на главный экран - рисунок 3.26.

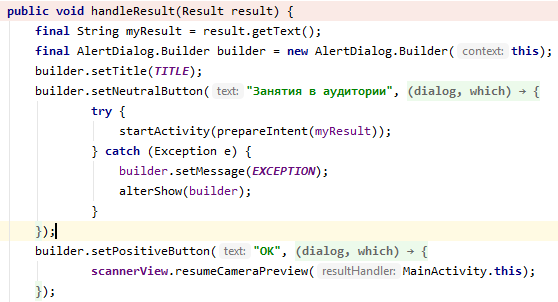


Рисунок 3.26 – Первая часть метода handleResult класса MainActivity

Вторая часть вызывает метод объекта PairService и в зависимости от результата выведет разные сообщения: при условии получения пустого объекта Pair, в диалоговое окно будет выведено сообщение о инвалидности отсканированного кода, если же объект Pair будет заполнен, то в диалоговое окно будет выведено строковое представление занятия. Вторая часть метода изображена на рисунке 3.27.

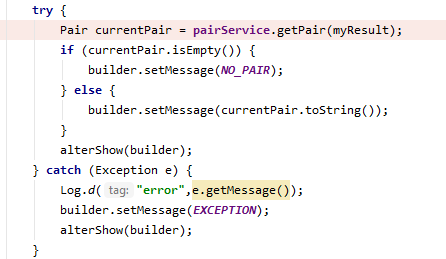


Рисунок 3.27 – Вторая часть handleResult класса MainActivity

Рассмотрим методы класса RoomActivity:

1. Метод onCreate - метод отвечающий за инициализацию основных компонентов работы данной части пользовательского интерфейса после его старта, так же он вызывает метод на объекте PairService , запрашивая все занятия в аудитории в данный день, отображая их для пользователя. Метод приведен на рисунке 3.27.



Рисунок 3.27 – Метод onCreate класса RoomActivity.

# ТЕСТИРОВАНИЕ И ОТЛАДКА ПРОГРАММНОГО СРЕДСТВА

## Критическое тестирование

При разработке приложения было проведено критическое тестирование, то есть. ошибки при стандартной работе приложения. Описание тестовых случаев и их результаты, представлены в таблице 4.1.

Таблица 4.1 – Протокол тестовых испытаний разработанного приложения, критическим тестированием

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название модуля/экрана | Описание тестового случая | Ожидаемые результаты | Тестовый случай пройден?  Да/Нет | Комментарии |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | MainActivity | 1. Открыть приложение; 2. Отклонить запрос на использование камеры. | 1. Сообщение о необходимости прав 2. Повторный запрос прав | Да |  |
| 2 | MainActivity | 1. Открыть приложение; 2. Принять запрос на использование камеры. | Вывод сообщения о предоставлении прав | Да |  |
| 3 | MainActivity | 1. Отсканировать валидный QR-код; 2. В базе данных нет информации о занятиях на момент сканирования. | Вывод диалогового окна с сообщением о том что занятий в данный момент не проводится. | Да |  |
| 4 | MainActiivity | 1. Отсканировать валидный QR-код; 2. В базе данных есть информация о проводимых занятиях на момент сканирования. | Вывод диалогового окна со списком занятий в аудитории. | Да |  |
| 5 | MainActivity | 1. Отсканировать любой QR-код; 2. Получить результат сканирования 3. Нажать «Ок» | Диалоговое окно исчезает, экран отображает QR-сканнер. | Да |  |
| 6 | RoomActivity | 1. Отсканировать валидный QR-код; 2. В базе данных нет информации о занятиях на момент сканирования; | Вывод на экране пустого списка занятий. | Да |  |

Продолжение таблицы 4.1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  | 1. Нажать кнопку «занятия в аудитории» |  |  |  |
| 7 | RoomActivity | 1. Отсканировать валидный QR-код; 2. В базе данных есть информации о занятиях на момент сканирования; 3. Нажать кнопку «занятия в аудитории» | Вывод на экран списка занятий в аудитории за весь день, в который происходит сканирование. | Да |  |

## Углубленное тестирование

Было произведено углубленное тестирование, то есть проверка работоспособности в нестандартных ситуациях. Описание тестовых случаев и их результаты, представлены в таблице 4.2.

Таблица 4.2 – Протокол тестовых испытаний разработанного приложения, углубленным тестированием

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| № | Название модуля/экрана | Описание тестового случая | Ожидаемые результаты | Тестовый случай пройден?  Да/Нет | Комментарии |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | MainActivity | Сканирование не валидного QR-кода. | Вывод диалогового окна с сообщением «QR-код содержит некорректные данные». | Да |  |
| 2 | MainActivity | 1. Сканирование не валидного QR-кода; 2. Нажатие в диалоговом окне «Занятия в аудитории» | Повторный вывод диалогового окна с сообщением «QR-код содержит некорректные данные». | Да |  |

## Результаты сканирования

В результате сканирования ошибок выявлено не было, приложение работает стабильно и без неожиданных поведений.

# РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ

Работа приложения построена на сканировании QR-кода, для корректной работы приложения было решено задать формат, в котором информация будет храниться в коде, JSON прекрасно подходит для этой задачи, так как он прост в понимании и обработке. Для того что бы создать понятный приложению QR-код необходимо задать два параметра в формате JSON: building, room. Оба параметра соответствуют значениям в базе данных, пример QR-кода приведен на рисунке 5.1.

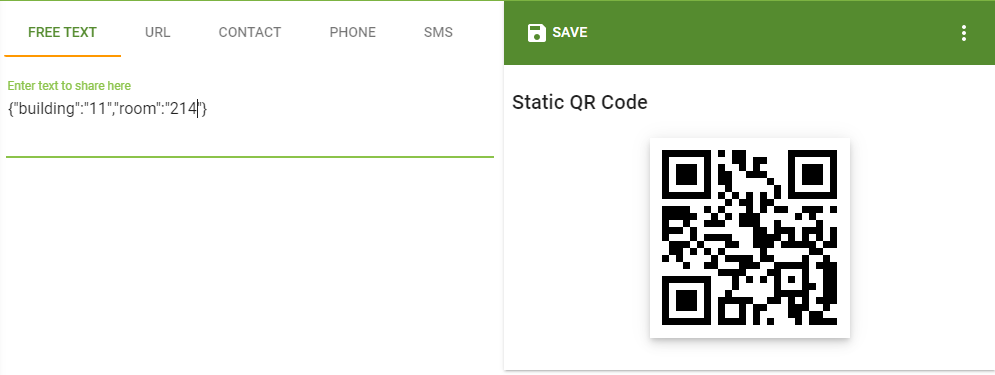


Рисунок 5.1 – Пример QR-кода и его содержания

Для того что бы узнать информацию о проходящих в аудитории занятиях необходимо открыть приложение и навести камеру на QR-код, таким образом, что б он попадал в рамки зеленого прямоугольника, пример изображен на рисунке 5.2.

A picture containing object, clock

Description automatically generated

Рисунок 5.2 – Сканирование QR-кода

После сканирования приложение сообщит пользователю о приводящихся в аудитории занятиях рисунок 5.3, так же пользователь может посмотреть полный список занятий в этот день, нажав на кнопку «Занятия в аудитории» в диалоговом окне рисунок 5.4.

A screenshot of a computer screen

Description automatically generated

Рисунок 5.3 – Информация о занятии в аудитории 214

A screenshot of a cell phone

Description automatically generated

Рисунок 5.4 – Полный список занятий

# 6 ТЕХНИКО-ЭКОНОМИЧЕСКОЕ ОБОСНОВАНИЕ РАЗРАБОТКИ ПРОГРММНОГО ПРОДУКТА

## 6.1 Определение единовременных затрат на создание программного продукта

При расчете экономической эффективности разработки программного обеспечения необходимо сопоставить затраты на решение задачи при ручном методе ее решения с затратами, связанными с ее автоматизацией. В том случае, если разрабатываемая задача внедряется взамен уже функционирующей или она представляет собой модификацию существующей задачи, необходимо осуществить сравнение затрат на создание и функционирование старой и новой задачи.

Единовременные капитальные затраты представляют собой цену программного продукта (ПП) или модели. Различают оптовую и отпускную цены. Все расчеты между покупателем и продавцом продукции, к числу которой относят и программные продукты (модели), производятся на основе отпускных цен. В настоящее время в соответствии с законодательством РБ в отпускную цену наряду с оптовой ценой включается налог на добавленную стоимость [1].

Определяющим фактором оптовой цены разработки является трудоёмкость создания ПП.

### 6.1.1 Определение трудоемкости разработки программного продукта

Предусмотрено четыре степени новизны разрабатываемых задач:

* – разработка задач, предусматривающая применение принципиально новых методов разработки, проведение научно-исследовательских работ;
* – разработка типовых проектных решений, оригинальных задач и систем, не имеющих аналогов;
* – разработка проекта с использованием типовых проектных решений при условии их изменения; разработка проектов, имеющих аналогичные решения;

Г – привязка типовых проектных решений.

Сложность алгоритма представлена тремя группами:

1. алгоритмы оптимизации и моделирования систем и объектов;
2. алгоритмы учета, отчетности, статистики и поиска;
3. алгоритмы, реализующие стандартные методы решения, а также не предусматривающие применения сложных численных и логических методов.

В данном дипломном проекте ПП имеет степень новизны В и сложность алгоритма 2.

Трудоемкость разработки ПП может быть определена на основе типовых норм времени для программирования задач на мобильнои устройстве. Она включает время на постановку задачи и время на программирование задачи и определяется по формуле:

Трз = Тоа + Тбс + Тп + Тотл + Тдр + Тдо, (6.1)

где Тоа - трудоемкость подготовки описания задачи и исследования алгоритма решения;

Тбс – трудоемкость разработки блок-схемы алгоритма;

Тп – трудоемкость программирования по готовой блок-схеме;

Тотл – трудоемкость отладки программы на мобильном устройстве;

Тдр – трудоемкость подготовки документации по задаче в рукописи;

Тдо – трудоемкость редактирования, печати и оформления документации по задаче.

Составляющие приведенной формулы определяются, в свою очередь, через условное число операторов *Q* в разрабатываемом ПП по формуле:

*Q* = *q* ∙ *C* ∙ (1 + *p*), (6.2)

где *q* – число операторов в программе (принят 1174);

*C* – коэффициент сложности программы (принят 1,4);

*p* – коэффициент коррекции программы в ходе её разработки (принят 0,2).

Коэффициент сложности программы *С* характеризует относительную сложность программ задачи по отношению к так называемой типовой задаче, сложность которой принята за единицу. Значение коэффициента определяется на базе экспертных оценок.

Коэффициент коррекции программ *p* характеризует увеличение объема работ за счет внесения изменений в алгоритм и программу, изменения состава и структуры информации, а также уточнений, вносимых разработчиком программы для улучшения её качества без изменения постановки задачи. Значение *p* может быть принято равным 0,15…0,5 (принято 0,2).

*Q* = 1174 ∙ 1,4 ∙ (1 + 0,2) = 1972 (операторов).

Составляющие трудоемкости разработки программы определятся по формулам:

Тоа, (6.3)

Тбс, (6.4)

Тп, (6.5)

Тотл, (6.6)

Тдр, (6.7)

Тдо0,75 ∙ Тдр (6.8)

где *W* – коэффициент увеличения затрат труда вследствие недостаточного или некачественного описания задачи (*W* = 1,2...1,5);

*K* – коэффициент квалификации разработчика алгоритмов и программ (при стаже работы до двух лет К = 0,8, при стаже от двух до трех лет К = 1,0, при стаже от трех до пяти лет К = 1,1...1,2, при стаже от пяти до семи лет К = 1,3...1,4, при стаже свыше семи лет К = 1,5...1,6).

Тоа22,27 (ч),

Тбс63,10(ч),

Тп63,10 (ч),

Тотл315,52(ч),

Тдр87,64(ч),

Тдо0,75 ∙ 87,64 65,73(ч).

Исходя из полученных данных можно найти трудоёмкость разработки программного продукта, используя формулу (7.1):

Трз = 22,27 + 63,10 + 63,10 + 315,52+ 87,64 + 65,73 = 617,36 (ч).

### 6.1.2 Определение себестоимости создания программного продукта

Для определения себестоимости создания программного продукта необходимо

определить затраты на заработную плату разработчика по формуле:

Зрз = Трз ∙ *t*чр ∙ (1 + *k*пр) ∙ (1 + *a*) ∙ (1 + *b*), (6.9)

где Трз – трудоемкость разработки программного продукта, чел-ч.;

*t*чр – среднечасовая ставка работника, осуществлявшего разработку программного продукта;

*k*пр – коэффициент, учитывающий процент премий в организации-разработчике (принят 0,4);

*а* – коэффициент, учитывающий дополнительную заработную плату (принят 0,15);

*b* – коэффициент, учитывающий отчисления от фонда заработной платы (отчисления в фонд социальной защиты населения и отчисления на обязательное медицинское страхование от несчастных случаев) (*b* = 0,346).

Среднечасовая ставка работника определяется исходя из Единой тарифной системы оплаты труда в Республике Беларусь по следующей формуле:

*t*др, (6.10)

где – среднемесячная заработная плата работника первого разряда (принято 41 руб.) [2].

*k*т – тарифный коэффициент работника соответствующего разряда (принято 2,97).

167 – среднее нормативное количество рабочих часов в месяце для 2020 года.

*t*чр 0,729 (руб.).

Теперь можно определить затраты на заработную плату разработчиков программы:

Зрз = 360,65 ∙ 0,729 ∙ (1 + 0,4) ∙ (1 + 0,15) ∙ (1 + 0,346) = 569,75(руб.).

В себестоимость разработки ПП включаются также затраты на отладку ПП в процессе его создания. Для определения их величины необходимо рассчитать стоимость машино-часа работы мобильного устройства, на которой осуществлялась отладка. Даная величина соответствует величине арендной платы за один час работы мобильного устройства.

Затраты на отладку программы определяются по формуле:

Зот =Тотл ∙ *S*мч, (6.11)

где Tотл – трудоемкость отладки программы (160 час.).

*S*мч – стоимость машино-часа работы мобильного устройства (0,3 руб./час.).

Зот =160∙ 0,3 = 48 (руб.).

Себестоимость разработки ПП определяется по формуле:

Спр =Зрз ∙ *F* + Зот, (6.12)

где *F* – коэффициент накладных расходов проектной организации без учета эксплуатации мобильного устройства (принят 1,17).

Таким образом

Спр =569,75 ∙ 1,17 + 48 = 714,60 (руб.).

### 6.1.3. Определение оптовой и отпускной цены программного продукта

Оптовая цена складывается из себестоимости создания программного продукта и плановой прибыли на программу.

Оптовая цена ПП определяется по формуле:

Цо =Спр + Пр, (6.13)

где Пр – плановая прибыль на программу, руб.

Плановая прибыль на программу определяется по формуле:

Пр =Спр ∙ Нп, (6.14)

где Спр – себестоимость программы;

Нп – норма прибыли проектной организации (принята Нп = 0,3).

Таким образом

Пр =714,60 ∙ 0,3 = 214,38 (руб.)

Тогда

Цо =714,60 + 214,38= 928,98 (руб)

Отпускная цена программы определяется по формуле:

Цпр =Цо +Цо ∙ Кндс, (6.15)

где Цо – оптовая цена программы, руб;

Кндс – ставка налога на добавленную стоимость (принято 20%) [1].

Из этого следует, что

Цпр =928,98 +928,98 ∙ 0,2 = 1114,776 (руб.).

### 6.1.4. Определение стоимости машино-часа работы мобильного устройства

Стоимость машино-часа работы мобильного устройства определяется по формуле:

*S*мч = Сэ + , (6.16)

где Сэ – расходы на электроэнергию за час работы мобильного устройства, руб;

Аму – годовая величина амортизационных отчислений на реновацию мобильного устройства;

Рму – годовые затраты на ремонт и техническое обслуживание мобильного устройства, руб;

Расходы на электроэнергию за час работы мобильного устройства определяются по формуле:

Сэ =*N*э ∙*k*ис ∙ Цэ, (6.17)

где *N*э – установленная мощность электродвигателя мобильного устройства (принято 0,025 кВт),

*k*ис – коэффициент использования энергоустановок по мощности (*k*ис = 0,9),

Цэ – стоимость 1 кВт-часа электроэнергии (принято 0,19 руб.) [3],

Чэл – среднечасовое потребление электроэнергии мобильного устройства, кВт.

Сэ =0,025∙0,9 ∙ 0,19 = 0,0042 (руб.).

Годовая величина амортизационных отчислений на реновацию мобильного устройства определяется по формуле:

Аму =Цму ∙*k*у ∙ = ∙ , (6.18)

где Цму – цена мобильного устройства на момент его выпуска (350 руб.),

*k*у – коэффициент удорожания мобильного устройства (принят *k*у = 1),

– норма амортизационных отчислений на мобильное устройство, % ( = 10%);

– балансовая стоимость мобильное устройство, руб.

Из формулы 4.18 получим:

Аму =350 ∙ 1 ∙ 1,05 ∙ = 36,75 (руб.)

Годовые затраты на ремонт и техническое обслуживание мобильного устройства укрупненно могут быть определены по формуле:

Рму = ∙ , (6.19)

где *k*ро – коэффициент, учитывающий затраты на ремонт и техническое обслуживание мобильного устройства, в том числе затраты на запчасти, зарплату ремонтного персонала и др. (*k*ро = 0,13).

Тогда

Рму = ∙ .

Таким образом, получим стоимость машино-часа работы мобильного устройства:

*S*мч = 0,0042 + =0,046(руб.).

## 6.2. Расчет ожидаемого прироста прибыли в результате внедрения программного продукта

Внедрение ПП может обеспечить пользователю ожидаемый прирост прибыли за счет сокращения трудоемкости решения задачи, являющейся предметом автоматизации и, как результат, снижения текущих затрат, связанных с решением данной задачи.

В том случае, если внедряемый ПП заменяет ручной труд, то производится сопоставление текущих затрат, связанных с решением задачи в ручном режиме и автоматизированном. В том случае, если разрабатываемая задача внедряется взамен уже функционирующей или она представляет собой модификацию существующей задачи, необходимо осуществить сравнение затрат на создание и функционирование старой и новой задачи.

### 6.2.1 Определение годовых эксплуатационных расходов при работе со схожими технологиями

Годовые эксплуатационные расходы при ручной обработке информации (ручном решении задачи) определяются по формуле:

Зр = *T*р ∙ к ∙ *t*чр ∙ (1 + *q*) ∙ (1 + *a*) ∙ (1 + *b*), (6.20)

где Тр – трудоемкость разового решения задачи вручную;

к – периодичность решения задачи в течение года;

*t*чр – среднечасовая ставка работника, осуществляющего ручной расчет задачи (0,601 руб.);

*q* – коэффициент, учитывающий процент премий (принят 0,4);

а – коэффициент, учитывающий дополнительную заработную плату (принят 0,15);

*b* – коэффициент, учитывающий отчисления от фонда заработной платы (принят 0,346).

Трудоемкость разового ручного решения задачи определяется по формуле:

Тр = , (6.21)

Для начала выполнения задачи необходимо включить компьютер и подключиться к интернету: *Tp*1 = 0,2 ч. Затем, найти необходимый день недели и расписание для него: *Tp*2 = 0,2 ч. Далее, необходимо необходимо непосредственно прибыть на место и найти нужную аудиторию: *Tp*3 = 0,4 ч.

Периодичность решения задачи в течение года зависит непосредственно от популярности технологии. Сейчас сервисами, предоставляющими интерактивное расписание, пользуются большое количество предприятий. Можно говорить о том, что периодичность решения задачи в год составляет порядка 10000 раз.

Тр = 0,2 + 0,2 + 0,4 = 0,8 (чел. – ч).

Зр = 0,8∙ 10000 ∙ 0,601 ∙ (1 + 0,4) ∙ (1 + 0,15) ∙ (1 + 0,346) = 10419,22 (руб.).

### 6.2.2. Определение годовых затрат, связанных с эксплуатацией ПП

Для расчета годовых текущих затрат, связанных с эксплуатацией ПП, необходимо определить время решения данной задачи на мобильном устройстве.

Время решения задачи на мобильном устройстве определяется по формуле:

Тз = (Твв + Тр + Твыв)∙ , (6.22)

где Твв – время ввода в мобильное устройство исходных данных, необходимых для решения задачи, мин (принято 0.3 мин.);

Тр – время вычислений, мин (принято 0.01 мин.);

Твыв – время вывода результатов решения задачи, мин; (принято 0,01 мин.);

*d*пз – коэффициент, учитывающий подготовительно-заключительное время (принято *d*пз=0,1).

Тз = (0,3+0,01+ 0,01) ∙ = 0,00586(чел. – ч).

На основе рассчитанного времени решения задачи может быть определена заработная плата пользователя данного ПП. Затраты на заработную плату пользователя ПП определяются по формуле:

Зп = *T*з ∙ к ∙ *t*чр ∙ (1 + *q*) ∙ (1 + *a*) ∙ (1 + *b*), (6.23)

где Тз – время решения задачи на мобильном устройстве, час;

*t*чп – среднечасовая ставка пользователя программы, руб. (определяется аналогично ставке работника, осуществляющего ручной расчет).

Зр = 0,00586∙ 10000 ∙ 0,601∙ (1 + 0,4) ∙ (1 + 0,15) ∙ (1 + 0,346) = 76,32 (руб.).

В состав затрат, связанных с решением задачи, включаются также затраты, связанные с эксплуатацией мобильного устройства.

Затраты на оплату вспомогательных средств (листы с распечатанными QR-кодами) для решения задачи определяются по следующей формуле:

Звс = *Sл*∙к ∙ *S*мч, (6.24)

где *S*мч – стоимость одного машино-часа работы мобильного устройства,

*Sл* – стоимость листа бумаги и распечатка (принята 0,9 руб.),

к – количество учебных аудиторий (принято 500).

Отсюда

Звс = 0.9 ∙ 500 ∙ 0,046= 20,7 (руб.).

Годовые текущие затраты, связанные с эксплуатацией задачи, определяются по формуле:

Зт = Зп + Звс, (6.25)

где Зп – затраты на заработную плату пользователя программы;

За – затраты на оплату аренды мобильного устройства при решении задачи.

Зт = 75,43 + 20,7 = 96,13 (руб.).

### 6.2.3. Определение ожидаемого прироста прибыли в результате внедрения программного продукта

Ожидаемый прирост прибыли в результате внедрения задачи взамен ручного ее расчета укрупнено может быть определен по формуле:

Пу = (Зр – Зт) ∙ (1 – Снп), (6.26)

где Снп – ставка налога на прибыль (18%) [4].

Пу = (10419,22 – 96,13) ∙ (1 – 0,18) = 8464,93 (руб.).

## 6.3. Расчёт показателей эффективности использования программного продукта

Для определения годового экономического эффекта от разработанной программы необходимо определить суммарные капитальные затраты на разработку и внедрения программы по формуле:

Ко = Кз + Цпр, (6.27)

где Кз – капитальные и приравненные к ним затраты;

Цпр – отпускная цена программы.

Капитальные и приравненные к ним затраты определяются, если необходимо приобретение нового МУ для решения комплекса задач, в который входит рассматриваемая, по формуле:

(6.28)

где – балансовая стоимость комплекта вычислительной техники, необходимого для решения задачи (принято за 350 руб.).

1114,776 = 1125,03 (руб.).

Годовой экономический эффект от сокращения ручного труда при обработке информации определяется по формуле:

ЭФ = Пу – Е ∙ Ко = Пу – Е ∙ (Кз + Цпр), (6.29)

где Е – коэффициент эффективности, равный ставке за кредиты на рынке долгосрочных кредитов (Е = 0,2).

ЭФ = 8464,93 – 0,2 ∙ 1125,03 = 8239,92(руб.).

Срок возврата инвестиций определяется по формуле:

, (6.30)

= 0,13 (г.).

Таким образом, разрабатываемый в дипломном проекте "Модуль "Навигация с помощью камеры" в мобильном приложении "Навигатор БНТУ"" модуль с отпускной ценой в 1114,77руб. будет иметь ожидаемый прирост прибыли в 8464,93 руб. Годовой экономический эффект составит 8239,92 руб., а срок возврата инвестиции составит 0,13 года, что указывает на экономическую целесообразность разработки.

В таблице 6.1 приведены технико-экономические показатели проекта.

Таблица 6.1 – Технико-экономические показатели проекта

|  |  |  |  |
| --- | --- | --- | --- |
| № | Наименование показателя | Варианты | |
| Базовый | Проектный |
| 1 | Трудоемкость решения задачи, час | 1 | 0,00586 |
| 2 | Периодичность решения задачи, раз/год | 10000 | 10000 |
| 3 | Годовые текущие затраты, связанные с решением задачи, руб. | 96,13 | 76,32 |
| 4 | Отпускная цена программы, руб |  | 1114,77 |
| 5 | Степень новизны программы |  | В |
| 6 | Группа сложности алгоритма |  | 2 |
| 7 | Прирост условной прибыли, руб |  | 8464,93 |
| 8 | Годовой экономический эффект, руб. |  | 8239,92 |
| 9 | Срок возврата инвестиций, лет |  | 0,13 |

# ОХРАНА ТРУДА

## Производственная санитария, техника безопасности и пожарная профилактика

При работе с ПЭВМ пользователи могут подвергаться воздействию опасных и вредных производственных факторов, таких как повышенные уровни: электромагнитного, ультрафиолетового и инфракрасного излучения; статического электричества; запыленность воздуха рабочей зоны; повышенное или пониженное содержание аэроионов в воздухе рабочей зоны; повышенный или пониженный уровень освещенности рабочей зоны, содержание в воздухе рабочей зоны вредных веществ (оксид углерода, озон, аммиак, фенол, формальдегид); напряжение зрения, памяти, внимания; длительное статическое напряжение; монотонность труда; нерациональная организация рабочего места; эмоциональные перегрузки.

Работа пользователей ПЭВМ проводится в соответствии с Санитарными нормами и правилами «Требования при работе с видеодисплейными терминалами и электронно-вычислительными машинами» и Гигиеническим нормативом «Предельно-допустимые уровни нормируемых параметров при работе с видеодисплейными терминалами и электронно-вычислительными машинами», утвержденными постановлением Министерства здравоохранения от 28.06.2013 г. № 59 и Типовой инструкцией по охране труда при работе с персональными ЭВМ, утвержденной постановлением Министерства труда и социальной защиты от 24.12.2013 № 130.

Площадь помещения на одного пользователя ПЭВМ на базе плоских дискретных экранов (жидкокристаллические, плазменные) составляет не менее 4,5 м2.

### Метеоусловия

В производственных помещениях, в которых работа с использованием ПЭВМ является основной (операторские, расчетные, посты управления, залы вычислительной техники), обеспечиваются оптимальные параметры микроклимата для категории работ 1а и 1б данные приведены в таблице 5.1.

.

Таблица 7.1. – Оптимальные параметры микроклимата для помещений с ВДТ, ЭВМ и ПЭВМ

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Период года | Категория работ | Температура воздуха, оС, не более | Относительная влажность воздуха, % | Скорость движения воздуха, м/с |
| Холодный | легкая-1а | 22-24 | 40-60 | 0,1 |
| легкая-1б | 21-23 | 40-60 | 0,1 |
| Теплый | легкая-la | 23-25 | 40-60 | 0,1 |
| легкая-1б | 22-24 | 40-60 | 0,2 |

Работа с компьютером относится к категории 1а (работы, производимые сидя и сопровождающиеся незначительным физическим напряжением, при которых расход энергии составляет до 120 ккал/ч, т.е. до 139 Вт).

Интенсивность теплового излучения работающих от нагретых поверхностей технологического оборудования, осветительных приборов, инсоляции на постоянных рабочих местах не превышает значений, указанных в табл. 7.2.

Таблица 7.2 – Предельно допустимые уровни интенсивности излучения в инфракрасном и видимом диапазоне излучения на расстоянии 0,5 м со стороны экрана ВДТ, ЭВМ и ПЭВМ

|  |  |  |  |
| --- | --- | --- | --- |
| Диапазоны длин волн | 400-760 нм | 760-1050 нм | свыше 1050 нм |
| Предельно допустимые уровни | 0,1 Вт/м2 | 0,05 Вт/м2 | 4,0 Вт/м2 |

Для создания нормальных метеорологических условий наиболее целесообразно уменьшить тепловыделения от самого источника — монитора, что предусматривается при разработке его конструкции.

В производственных помещениях для обеспечения необходимых показателей микроклимата предусмотрены системы отопления, вентиляции и кондиционирования воздуха.

### Вентиляция и отопление

Воздух рабочей зоны помещения соответствует санитарно-гигиеническим требованиям по содержанию вредных веществ и частиц пыли, приведенным в Санитарных нормах и правилах «Требованию к контролю воздуха рабочей зоны», Гигиеническом нормативе «Предельно допустимые концентрации вредных веществ в воздухе рабочей зоны», утв. пост. Министерства здравоохранения от 10.10.2017 г. № 92.

В помещениях проводится ежедневная влажная уборка и систематическое проветривание после каждого часа работы.

Уровни положительных и отрицательных аэроионов, а также коэффициент униполярности в воздухе всех помещений, где расположены ПЭВМ, соответствуют значениям, указанным в табл. 7.3.

Таблица 7.3 – Уровни ионизации и коэффициент униполярности воздуха помещений при работе с ВДТ, ЭВМ и ПЭВМ

|  |  |  |  |
| --- | --- | --- | --- |
| Уровни | Число ионов в 1 см3 воздуха | | Коэффициент униполярности (У) |
| n+ | n- |
| Минимально допустимые | 400 | 600 | 0,4 ≤ У < 1,0 |

Продолжение таблицы 7.3

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | 2 | 3 | 4 |
| Оптимальные | 1500-3000 | 3000-5000 |  |
| Максимально допустимые | 50000 | 50000 |

Одним из мероприятий по оздоровлению воздушной среды является устройство вентиляции и отопления. Задачей вентиляции является обеспечение чистоты воздуха и параметров метеорологических условий на рабочих местах. Чистота воздушной среды достигается удалением загрязненного или нагретого воздуха из помещения и подачей в него свежего воздуха. Для поддержания нормального микроклимата необходим достаточный объем вентиляции, для чего в вычислительном центре предусматривается кондиционирование воздуха, осуществляющее поддержание постоянных параметров микроклимата в помещении независимо от наружных условий.

Параметры микроклимата поддерживаются в холодный период года за счет системы водяного отопления с нагревом воды до 100°С, а в теплый - за счет кондиционирования, с параметрами отвечающими требованиям СНБ 4.02.01-03.

### Освещение

В помещении при использовании ПЭВМ предусмотрены естественное и искусственное освещение. Естественное освещение на рабочих местах осуществляется через световые проемы, ориентированные преимущественно на север, северо-восток, восток, запад или северо-запад и обеспечивает коэффициент естественной освещенности не ниже 1,5 %. Оконные проемы оборудованы регулируемыми устройствами типа жалюзи, занавесей.

Для внутренней отделки интерьера помещений используются материалы с коэффициентом отражения для потолка – 0,7- 0,8; для стен – 0,5- 0,6; для пола – 0,3- 0,5.

Искусственное освещение в помещениях осуществляется системой общего равномерного освещения. При работе с документами применяется система комбинированного освещения, а освещенность поверхности стола в зоне размещения рабочего документа составляет 300-500 люкс. Освещенность поверхности экрана не более 300 люкс. В качестве источников света применяем люминесцентные лампы типа ЛБ. Коэффициент запаса для осветительных установок общего освещения принимается равным 1,4, а коэффициент пульсации – не более 5 %.

### Шум

Основными источниками шума в помещениях, оборудованных ПЭВМ, являются принтеры, множительная техника и оборудование для кондиционирования воздуха, в самих ПЭВМ — вентиляторы систем охлаждения и трансформаторы.

В табл. 7.4 приведены допустимые уровни шума, которые обеспечиваются за счет использования малошумного оборудования, применения звукопоглощающих материалов для облицовки помещений, а также различных звукопоглощающих устройств (перегородки и т. п.).

Таблица 7.4 – Предельно-допустимые уровни звука, эквивалентные уровни звука и уровни звукового давления в октавных полосах частот при работе с ВДТ, ЭВМ и ПЭВМ и периферийными устройствами

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Категория нормы  шума | Уровни звукового давления, дБ, в октавных полосах  со среднегеометрическими частотами, Гц | | | | | | | | | Уровни звука и  эквивалентные уровни звука, дБА |
| 31.5 | 63 | 125 | 250 | 500 | 1000 | 2000 | 4000 | 8000 |
| I | 86 | 71 | 61 | 54 | 49 | 45 | 42 | 40 | 38 | 50 |
| II | 93 | 79 | 70 | 63 | 58 | 55 | 52 | 50 | 49 | 60 |
| III | 96 | 83 | 74 | 68 | 63 | 60 | 57 | 55 | 54 | 65 |
| IV | 103 | 91 | 83 | 77 | 73 | 70 | 68 | 66 | 64 | 75 |

### Электробезопасность

Помещение вычислительного центра по степени опасности поражения электрическим током относится к помещениям без повышенной опасности.

Основные меры защиты от поражения током:

1. изоляция и недоступность токоведущих частей;
2. защитное заземление (R3 = 4 Ом ГОСТ 12.1.030 - 81).

Первая помощь при поражениях электрическим током состоит из двух этапов: освобождение пострадавшего от действия тока и оказание ему доврачебной медицинской помощи. После освобождения пострадавшего от действия электрического тока необходимо оценить его состояние. Во всех случаях поражения электрическим током необходимо вызвать врача независимо от состояния пострадавшего.

### Излучение

При работе с дисплеем ПЭВМ могут возникать следующие опасные факторы: электромагнитные и электростатические поля, ультрафиолетовое и инфракрасное излучение.

Уровни физических факторов на рабочих местах пользователей, создаваемые ПЭВМ и периферийными устройствами, не превышают предельно-допустимые уровни: электромагнитных и электростатических полей таблица 7.5 и таблица 7.6, ультрафиолетового таблица 7.7.

Таблица 7.5 – Предельно допустимые уровни электромагнитных полей от экранов ВДТ, ЭВМ и ПЭВМ

|  |  |
| --- | --- |
| Наименование параметра | Предельно-допустимые уровни |
| Напряженность электрического поля в диапазоне частот: | не более 25,0 В/м  не более 2,5 В/м |
| 5 Гц-2 кГц  2-400 кГц |
| Плотность магнитного потока магнитного поля в диапазоне частот: | не более 250 нТл  не более 25 нТл |
| 5 Гц-2 кГц  2-400 кГц |
| Напряженность электростатического поля | не более 15 кВ/м |

Таблица 7.6 – Предельно допустимые уровни электромагнитных полей при работе с ВДТ, ЭВМ, ПЭВМ от клавиатуры, системного блока, манипулятора «мышь», беспроводных системам передачи информации и иных периферийных устройств

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Диапазоны частот | 0,3-300  кГц | 0,3-3  МГц | 3-30  МГц | 30-300  МГц | 0,3-300  ГГц |
| Предельно допустимые уровни | 25 В/м | 15 В/м | 10 В/м | 3 В/м | 10 мкВт/см2 |

Таблица 7.7 – Предельно допустимые уровни интенсивности излучения в ультрафиолетовом диапазоне на расстоянии 0,5 м со стороны экрана ВДТ, ЭВМ и ПЭВМ

|  |  |  |  |
| --- | --- | --- | --- |
| Диапазоны длин волн | 200-280 нм | 280-315 нм | 315-400 нм |
| Предельно допустимые уровни | не допускается | 0,0001 Вт/м2 | 0,1 Вт/м2 |

Наиболее эффективным и часто применяемым методом защиты от электромагнитных излучений является установка экранов. Экранируют либо источник излучения, либо рабочее место. Часто экран устанавливают непосредственно на монитор.

При работе монитора на экране кинескопа накапливается электростатический заряд, создающий электростатическое поле. При этом персонал, работающий с монитором, приобретают электростатический потенциал. Заметный вклад в общее электростатическое поле вносят электризующиеся от трения поверхности клавиатуры и мыши.

### Пожарная безопасность

По взрывопожарной и пожарной опасности помещения и здания для ЭВМ относятся к категории Д согласно ТКП 474-2013. Здания для ВЦ и части зданий другого назначения, в которых предусмотрено размещение ЭВМ, относятся к 2 степени огнестойкости согласно ТКП 45-2.02-315-2018.

Для предотвращения распространения огня во время пожара с одной части здания на другую устраивают противопожарные преграды в виде стен, перегородок, дверей, окон. Особое требование предъявляется к устройству и размещению кабельных коммуникаций.

Нормы первичных средств пожаротушения для вычислительных центров приведены в табл. 7.8.

Таблица 7.8 – Примерные нормы первичных средств пожаротушения для вычислительного центра

|  |  |  |  |
| --- | --- | --- | --- |
| Помещение | Площадь, м2 | Углекислотные огнетушители ручные | Порошковые огнетушители |
| Вычислительный центр | 100 | 1 | 1 |

Для ликвидации пожаров в начальной стадии применяются первичные средства пожаротушения: внутренние пожарные водопроводы, огнетушители типа ОВП-10, ОУ-2, асбестовые одеяла и др.

Эвакуация персонала вычислительного центра осуществляется через эвакуационные выходы. Количество и общая ширина эвакуационных выходов определяются в зависимости от максимального возможного числа эвакуирующихся через них людей и предельно допустимого расстояния от наиболее удаленного места возможного пребывания людей до ближайшего эвакуационного выхода согласно ТКП 45-2.02-315-2018.

Расчетное время эвакуации устанавливается по реальному расчету времени движения одного или нескольких потоков людей через эвакуационные выходы из наиболее удаленных мест размещения людей. Необходимое время эвакуации устанавливается на основе данных о критической продолжительности пожара с учетом степени огнестойкости здания, категории производства по взрывной и пожарной опасности.

## Требования к помещениям для работы с ВДТ, ЭВМ и ПЭВМ

Помещения для эксплуатации ЭВМ и ПЭВМ имеют естественное и искусственное освещение.

Запрещается выполнение основной работы с использованием ЭВМ и ПЭВМ на постоянных рабочих местах без естественного освещения, если это не обусловлено технологическим процессом.

Естественное освещение на рабочих местах с ЭВМ и ПЭВМ осуществляется через световые проемы, ориентированные преимущественно на север, северо-восток, восток, запад или северо-запад и обеспечивает коэффициент естественной освещенности не ниже 1,5 %. Оконные проемы оборудованы регулируемыми устройствами типа жалюзи, занавесей, внешних козырьков и другое.

Площадь одного рабочего места для пользователей ЭВМ и ПЭВМ на базе электронно-лучевой трубки составляет не менее 6 м2. Минимальная площадь одного рабочего места для взрослых пользователей и обучающихся учреждений образования с использованием ЭВМ или ПЭВМ на базе ЭЛТ может составлять не менее 4,5 м2 при условиях: отсутствия на рабочем месте периферийных устройств (принтер, сканер и другое); продолжительности работы – не более 4 часов в день.

Площадь одного рабочего места для пользователей ЭВМ и ПЭВМ на базе плоских дискретных экранов (жидкокристаллические, плазменные и другое) составляет не менее 4,5 м2.

Помещения, где размещаются рабочие места с ЭВМ и ПЭВМ, оборудованы защитным заземлением (защитным занулением) в соответствии с техническими требованиями по эксплуатации. Запрещается размещать рабочие места с ЭВМ и ПЭВМ на расстоянии менее 10 м от силовых кабелей, вводов и высоковольтных трансформаторов.

Помещения, в которых для работы используются преимущественно ЭВМ и ПЭВМ (операторские, расчетные и т.п.), не граничат с помещениями, в которых уровни шума и вибрации превышают нормируемые значения для данной категории проводимых в них работ и их типа (механические цеха, мастерские и т.п.). Звукоизоляция ограждающих конструкций помещений с ЭВМ и ПЭВМ обеспечивает нормируемые параметры шума в них.

Помещения с ЭВМ и ПЭВМ оборудованы системами отопления, кондиционирования воздуха или эффективной приточно-вытяжной вентиляцией.

Полимерные материалы, используемые для внутренней отделки интерьера помещений с ЭВМ и ПЭВМ, соответствуют требованиям Санитарных норм и правил, устанавливающих критерии гигиенической безопасности полимерных и полимеросодержащих материалов, изделий и конструкций, применяемых в промышленном и гражданском строительстве.

Поверхность пола в помещениях эксплуатации ЭВМ и ПЭВМ ровная, без выбоин, нескользкой, удобной для очистки и влажной уборки, обладает антистатическими свойствами.

Для внутренней отделки интерьера помещений, где расположены ЭВМ и ПЭВМ, использованы диффузно отражающие материалы с коэффициентом отражения для потолка – 0,7-0,8; для стен – 0,5-0,6; для пола – 0,3-0,5.

# ЗАКЛЮЧЕНИЕ

В данном дипломном проекте было разработано мобильное приложение, которое призвано помочь студентам с ориентированием по аудиториям и расписанию. Приложение может использоваться во всем университете, так как его логика разработана специально для его нужд.

Данное решение экономически выгодно и не требует больших затрат. Так же разработанная модель удобна тем, что распечатанные QR-коды не понадобится перепечатывать множество раз, так как логика приложения достаточно гибкая и при необходимости может корректироваться, без необходимости изменений структуры кодов.

Основным недостатком приложения является необходимость ручного заполнения базы данных, так как на данный момент нет возможности автоматизированного и единообразно структурированного заполнения расписания. Однако данная задача решается другими проектами, что позволит в дальнейшем справиться с этим недостатком. У приложения много перспектив и вариантов дальнейшего развития функциональностей, например поиск по расписанию или преподавателям.

Таким образом задачи, поставленные данным дипломным проектом были полностью выполнены.

# СПИСОК ИСПОЛЬЗОВАНЫХ ИСТОЧНИКОВ

1. Practical Fingerprinting Localization for Indoor Positioning System by Using Beacons. // Платформа для публикации https://www.hindawi.com [Электронный ресурс]. – Режим доступа: https://www.hindawi.com/journals/js/2017/9742170/ – Дата доступа: 17.03.2020.
2. Wireless and Mobile Technologies for the Internet of Things**. //** Платформа для публикации https://www.hindawi.com [Электронный ресурс]. – Режим доступа: <https://www.hindawi.com/journals/misy/2016/5032365/> – Дата доступа: 03.03.2020.
3. Mobile Operating System Market Share Worldwide. // Онлайн инструмент статистики <https://statcounter.com/> [Электронный ресурс]. – Режим доступа: <https://gs.statcounter.com/os-market-share/mobile/worldwide> – Дата доступа: 20.04.2020.
4. Number of available applications in the Google Play Store from December 2009 to March 2020. // Платформа бизнес-данных https://www.statista.com [Электронный ресурс]. – Режим доступа: <https://www.statista.com/statistics/266210/number-of-available-applications-in-the-google-play-store/> – Дата доступа: 16.04.2020.
5. Number of apps available in leading app stores as of 1st quarter 2020.// Платформа бизнес-данных https://www.statista.com [Электронный ресурс]. – Режим доступа: <https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/> – Дата доступа: 18.04.2020.
6. Most Popular Technologies**. //** Открытое сообщество разработчиков <https://stackoverflow.com/> [Электронный ресурс]. – Режим доступа: <https://insights.stackoverflow.com/survey/2019#technology> – Дата доступа: 06.04.2020.
7. Global market share held by operating systems for desktop PCs, from January 2013 to January 2020. **//** Платформа бизнес данных https://www.statista.com [Электронный ресурс]. – Режим доступа: <https://www.statista.com/statistics/218089/global-market-share-of-windows-7/> – Дата доступа: 06.04.2020.
8. Kruchten, P. The Rational Unified Process: An Introduction, Third Edition. / P. Kruchten, 2003 – 212 c.: ил.
9. Eric Freeman, Bert Bates, Kathy Sierra. Head First Design Patterns: A Brain-Friendly Guide 2015 – 126 c.: ил.
10. Model-View-Controller **//** Свободная энциклопедия<https://ru.wikipedia.org/> [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Controller> – Дата доступа: 25.04.2020.
11. Model-View-Presenter **//** Свободная энциклопедия<https://ru.wikipedia.org/> [Электронный ресурс]. – Режим доступа: <https://ru.wikipedia.org/wiki/Model-View-Presenter> – Дата доступа: 25.04.2020.
12. Android MVVM Design Pattern **//** Сайт о программированииhttps://www.journaldev.com [Электронный ресурс]. – Режим доступа: <https://www.journaldev.com/20292/android-mvvm-design-pattern> – Дата доступа: 27.04.2020.
13. Michael J. Hernandez Database Design (2013) – 73 c.: ил.