**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ**

**«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Образовательная программа бакалавриата «Программная инженерия»

|  |  |
| --- | --- |
| СОГЛАСОВАНО  Научный руководитель,  приглашённый преподаватель ОП «Программная инженерия» департамента программной инженерии факультета компьютерных наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Г.М. Сосновский  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. | УТВЕРЖДАЮ  Академический руководитель  ОП «Программная инженерия»  профессор департамента программной инженерии, канд. техн. наук  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ В. В. Шилов  «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_ 2023 г. |

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. Инв. № |  |
| Подп. и дата |  |
| Инв. № подл. | **RU.17701729.02.11-01 81 01-1-ЛУ** |

**АНДРОИД ПРИЛОЖЕНИЕ ДЛЯ ВИРТУАЛЬНОГО ПРОСМОТРА**

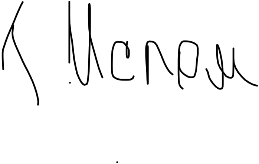
**ДОСТОПРИМЕЧАТЕЛЬНОСТЕЙ**

**Пояснительная записка**

**ЛИСТ УТВЕРЖДЕНИЯ**

**RU.17701729.02.11-01 81 01-1-ЛУ**

Исполнитель

студент группы БПИ204

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ / Туракулов И. У. /

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2023г.

УТВЕРЖДЕН

RU.17701729.02.11-01 81 01-1-ЛУ

|  |  |
| --- | --- |
| Подп. и дата |  |
| Инв. № дубл. |  |
| Взам. Инв. № |  |
| Подп. и дата |  |
| Инв. № подл. | **RU.17701729.02.11-01 81 01-1-ЛУ** |

**АНДРОИД ПРИЛОЖЕНИЕ ДЛЯ ВИРТУАЛЬНОГО ПРОСМОТРА**

**ДОСТОПРИМЕЧАТЕЛЬНОСТЕЙ**

**Пояснительная записка**

**RU.17701729.02.11-01 81 01–1**

**Листов 31**

АННОТАЦИЯ

В данном программном документе приведена пояснительная записка к программе «Андроид приложение для виртуального просмотра достопримечательностей»

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы:

1. Постановка задачи на разработку программы;
2. Описание функционирования программы с обоснованием выбора схемы алгоритма решения задачи и возможные взаимодействия программы с другими программами
3. Описание и обоснование выбора метода организации входных и выходных данных;
4. Описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами и аналогами.

Настоящий документ разработан в соответствии с требованиями:

1. ГОСТ 19.101–77 Виды программ и программных документов [1];
2. ГОСТ 19.102–77 Стадии разработки [2];
3. ГОСТ 19.103–77 Обозначения программ и программных документов [3];
4. ГОСТ 19.104–78 Основные надписи [4];
5. ГОСТ 19.105–78 Общие требования к программным документам [5];
6. ГОСТ 19.106–78 Требования к программным документам, выполненным печатным способом [6];
7. ГОСТ 19.404–79. Пояснительная записка. Требования к содержанию и оформлению.

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603–78 [8], ГОСТ 19.604–78-[9].

**СОДЕРЖАНИЕ**

[**1. ВВЕДЕНИЕ** 5](#_Toc131890285)

[**1.1. Наименование программы** 5](#_Toc131890286)

[**1.2. Краткая характеристика и область назначения** 5](#_Toc131890287)

[**2. НАЗНАЧЕНИЕ РАЗРАБОТКИ** 6](#_Toc131890288)

[**2.1. Функциональное назначение** 6](#_Toc131890289)

[**2.2. Эксплуатационное назначение** 6](#_Toc131890290)

[**3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ** 7](#_Toc131890291)

[**3.1. Постановка задачи на разработку программы** 7](#_Toc131890292)

[**3.2. Описание алгоритма и функционирования программы** 7](#_Toc131890293)

[**3.3. Описание и обоснование выбора состава технических и программных средств** 18](#_Toc131890294)

[**4. ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ** 19](#_Toc131890295)

[**4.1. Ориентировочная экономическая эффективность** 19](#_Toc131890296)

[**4.2. Предполагаемая потребность** 19](#_Toc131890297)

[**4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами** 19](#_Toc131890298)

[**ПРИЛОЖЕНИЕ 1** 21](#_Toc131890299)

[**ПРИЛОЖЕНИЕ 2** 22](#_Toc131890300)

[**ПРИЛОЖЕНИЕ 3** 23](#_Toc131890301)

[**ПРИЛОЖЕНИЕ 4** 24](#_Toc131890302)

# **ВВЕДЕНИЕ**

## **Наименование программы**

Наименование программы – «Андроид приложение для виртуального просмотра достопримечательностей».

Наименование программы на английском языке – «Android application for virtual sightseeing».

Наименование программы для пользователя – «Путешествие и точка».

## **Краткая характеристика и область назначения**

«Андроид приложение для виртуального просмотра достопримечательностей» — это приложение, позволяющее пользователям виртуально просматривать достопримечательности с помощью мобильных устройств на платформе Android. Оно может включать в себя функции, такие как виртуальные туры, интерактивные карты, фото и видеоресурсы о достопримечательностях, информацию об истории и фактах о достопримечательностях. Это приложение может быть использовано туристами, туристическими гидами или любыми людьми, желающими узнать больше о достопримечательностях.

# **НАЗНАЧЕНИЕ РАЗРАБОТКИ**

## **Функциональное назначение**

Функциональным назначением приложения «Путешествие и точка» предоставление пользователям виртуально просматривать достопримечательности с помощью мобильных устройств на платформе Android. Предоставляя пользователям возможности виртуального просмотра достопримечательностей и доступа к информации о них с использованием мобильных устройств на платформе Android. Это может включать в себя функции, такие как:

1. Виртуальные туры
2. Интерактивные карты
3. Фото и видео ресурсы о достопримечательностях
4. Информацию об истории и фактах о достопримечательностях.

## **Эксплуатационное назначение**

Данное приложение является удобным и доступным способом исследования достопримечательностей с помощью смартфона и интернет-соединения, вместо традиционных путешествий. Это позволяет людям расширять свой кругозор и получать новые знания и впечатления о достопримечательностях в любой точке планеты.

# **ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ**

## **Постановка задачи на разработку программы**

Целью разработки данной программы является создание мобильного приложения «Путешествие и точка» для платформы Android, предоставляющего пользователям возможность виртуального просмотра и изучения туристических достопримечательностей с использованием различных функций и инструментов.

* + 1. **Задачи работы**

1. Разработка архитектуры приложения, определение основных компонентов, их взаимодействия и взаимосвязи.
2. Реализация функций авторизации, обеспечивая безопасный вход пользователей в систему.
3. Разработка функционала для выбора туристических мест, включая демонстрацию возможных туристических мест, использование поисковой строки и фильтрации по категориям.
4. Создание интерактивных элементов скипетров, таких как просмотр видео-контента, аудио-контента, чтение контента и 3D-просмотрщика для взаимодействия с карточками моделей достопримечательностей.
5. Разработка функционала для построения маршрута на карте, выбора начальной и конечной точки, а также подбора рекомендаций по траектории пути движения по карте.
6. Создание блоков с информацией о достопримечательностях, включая просмотр отдельных туристических наборов определенного региона и возможность добавления временных туристических событий через сканирование QR-кода.
7. Реализация пользовательского интерфейса, основанного на предоставленных требованиях к составу интерфейса и прототипе.
8. Тестирование и отладка приложения, обеспечивая его стабильность и корректную работу всех функций.

## **Описание алгоритма и функционирования программы**

* + 1. Постановка задачи на разработку программы

#### **Структура мобильного приложения**

1. **Загрузочная страница** – отображает промежуточный warm up приложения к взаимодействию с пользователем.

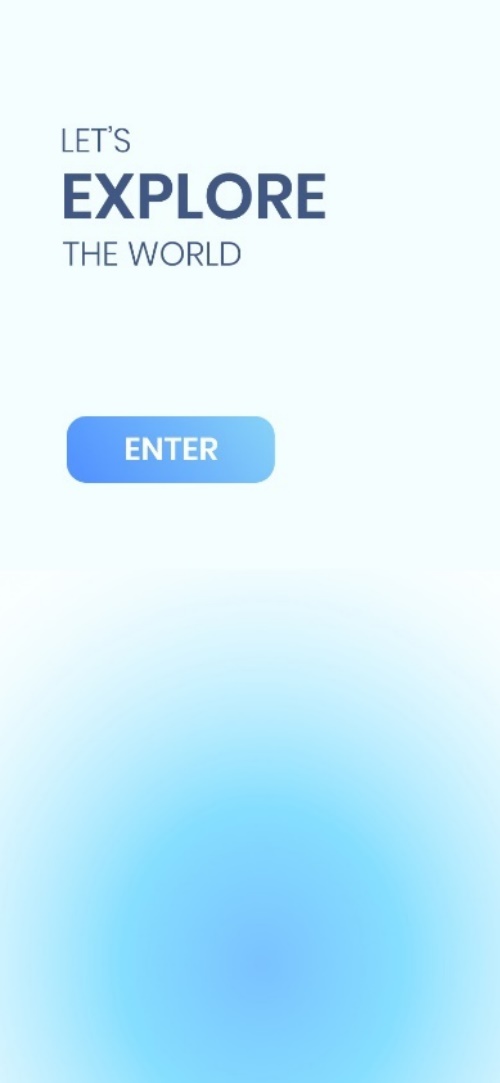


Рисунок 1 Загрузочная страница

При первом запуске приложения пользователю отображается загрузочная страница с переходом на авторизацию. Данные сессии проверяются через сессионный ключ, который хранится в защищенном хранилище Android телефона Keystore (<https://developer.android.com/training/articles/keystore>).

@HiltAndroidApp  
class MainApp: Application()

1. **Главный экран** - отображает поисковую строку, список туристических достопримечательностей, рекомендательные баннеры и блок рекламного опыта.

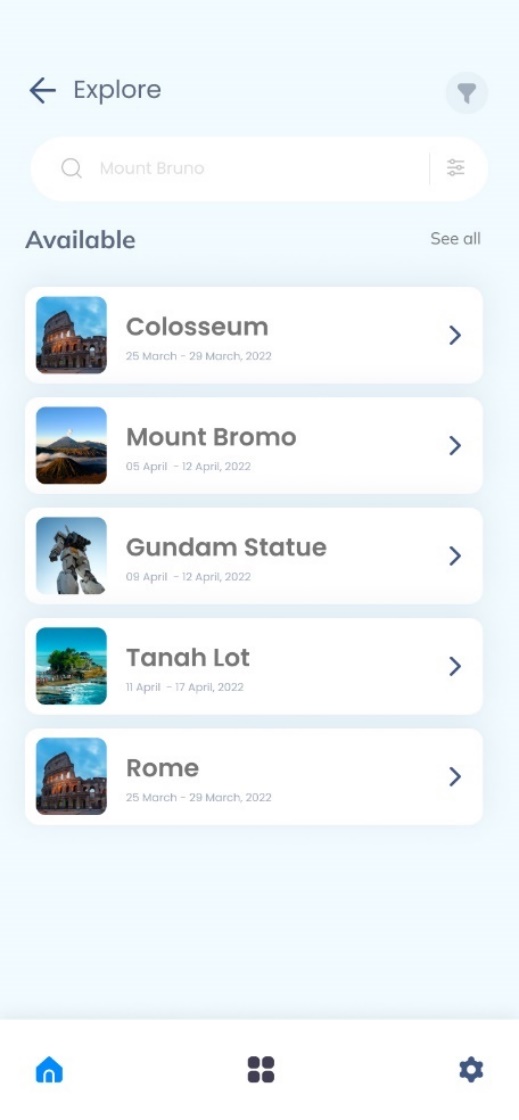


Рисунок 2 Главная страница

Главная страница приложения содержит один Recylerview с SearchBar’ом. Внутри Recyclerview заложена анимация шиммера при подзагрузке данных. Вызовы данных происходят на альтернативном потоке благодаря корутинам и Flow переменным.

@HiltViewModel  
class MainViewModel @Inject constructor(  
 private val mainRepository: MainRepository,  
) : BindingViewModel() {

// the rest code

@get:Bindable  
 val architectureList: List<ArchitectureInfo>? by architectureListFlow.asBindingProperty(  
 viewModelScope,  
 null  
 )  
  
 private val \_archLiveData = MutableLiveData<List<ArchitectureInfo>?>()  
 val architectureLiveData: LiveData<List<ArchitectureInfo>?>  
 get() = \_archLiveData  
  
 init {  
 viewModelScope.launch {  
 mainRepository.updateArchitectureInfo().suspendOnSuccess {  
 \_archLiveData.value = data  
 mainRepository.insertArchitectureInfo(data)  
 }.suspendOnError {  
 Timber.d("error ${message()}")  
 }  
 }  
 }  
}

Происходит Long Polling вызовы для последующего переобновления секции. Если данных в Recycler view оказалось больше, то происходит разделение на чанки и интеграция пагинации.

fun getUsers(): Flow<PagingData<User>> {

return Pager(

config = PagingConfig(pageSize = 20, enablePlaceholders = false),

pagingSourceFactory = { UserPagingSource(apiService) }

).flow

}

На главной странице представлены следующие элементы:

1. Карусель изображений или слайдер, отображающий лучшие и наиболее популярные архитектурные объекты, доступные для виртуального просмотра. Это позволяет пользователям получить представление о масштабе и разнообразии предлагаемых мест.
2. Краткий список архитектурных достопримечательностей, сгруппированных по регионам, странам или городам, упрощая выбор и навигацию по интересующим объектам.
3. Фильтры и сортировки, позволяющие пользователям ограничить список архитектурных объектов в соответствии с их предпочтениями, такими как исторический период, стиль архитектуры, географическое расположение или рейтинг.
4. **Экран профиля** - содержит авторизацию пользователя, список избранных туристических достопримечательностей, справку, настройки и выход из профиля.

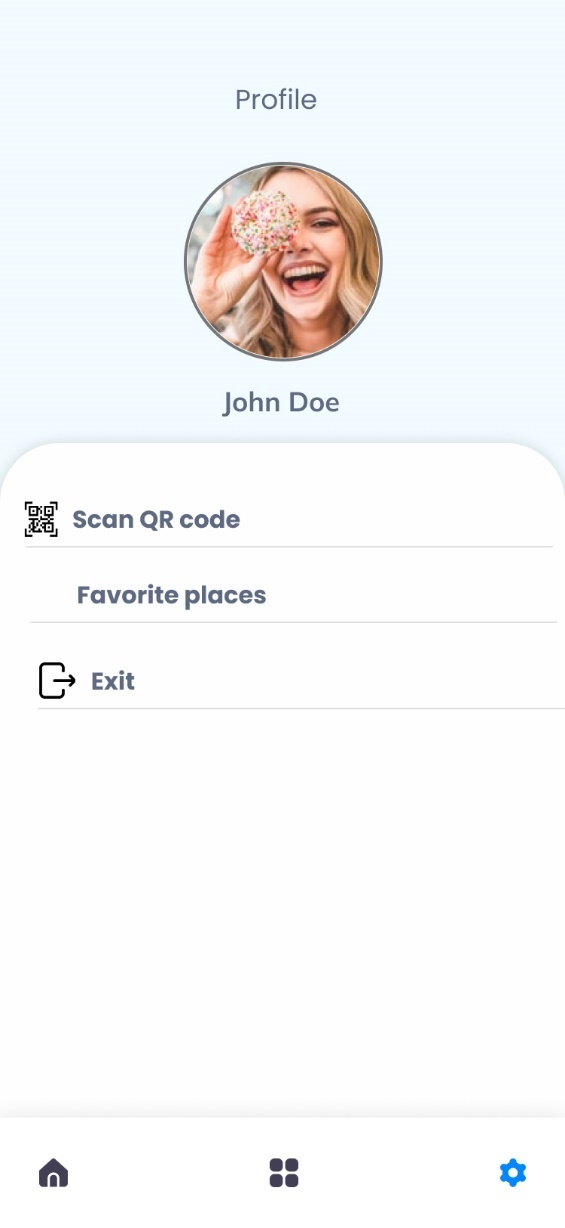


Рисунок 3 Профиль пользователя

Экран профиля предоставляет доступ к разнообразным опциям и настройкам, связанным с аккаунтом пользователя и взаимодействием с приложением. Проверяется наличие сессионного токена, при присутствии можно попытаться восстановить сессию через токен. Используется Firebase для регистрации/авторизации пользователя. Для отказоустойчивой работы приложения предлагается использовать защищенные протоколы общения во время взаимодействия с сервером. Отправляются токены в зашифрованном виде в Firebase каждые 5 минут для проверки статуса пользователя. Если пользователь находится вне сети, то отключается возможность авторизации и показывается сохраненные сниппеты с локальной базы данных Room Database. На данном экране можно найти следующие элементы:

1. Кнопка "Вход" – показывает панель авторизации
2. Кнопка "Избранное" – вывод списка предпочитаемых туристических объектов
3. Кнопка "Помощь" – предоставление информации о приложении
4. Кнопка "Сканировать QR код" – представление интерфейса для сканирования QR кода для добавления эксклюзивного набора.
5. Кнопка "Выйти" (для авторизованных пользователей)
6. **Экран каталога** - позволяет пользователю просматривать категории и популярные архитектурные места, а также использовать поисковую строку для навигации в выдаче.

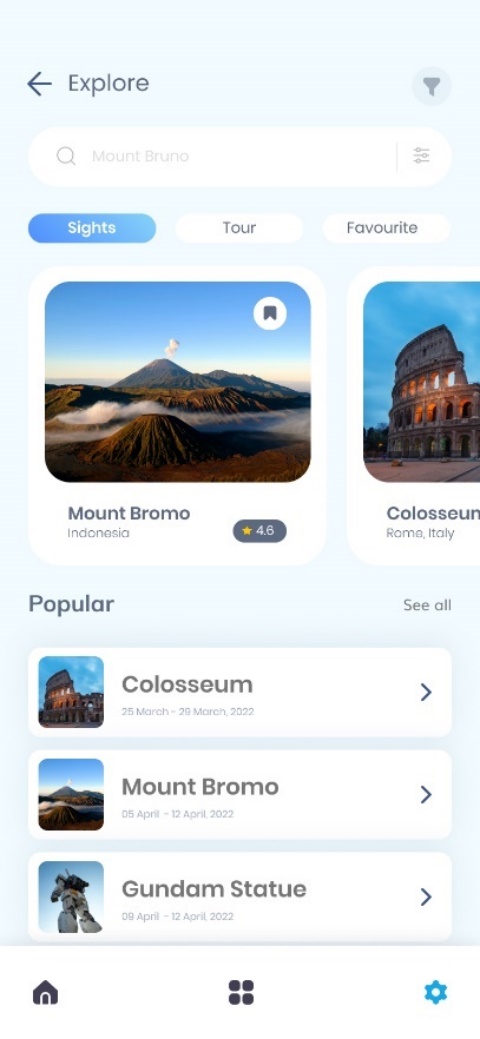
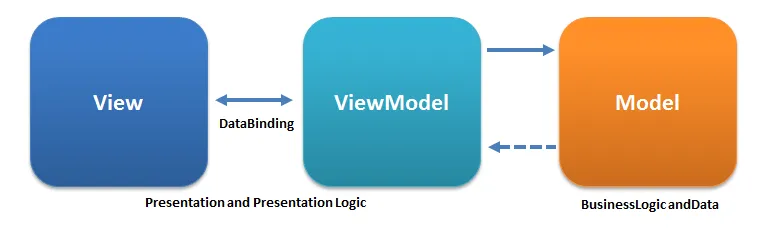


Рисунок 4 Каталог туристических мест

Каталог мобильного приложения содержит разнообразные категории туристических объектов, облегчая пользователям навигацию и поиск интересующих их мест. Эти категории включают:

1. Популярные места – объекты, которые наиболее часто посещаются или исследуются пользователями приложения.
2. Места, добавленные в избранное – те достопримечательности, которые пользователи отметили, как особенно интересные или значимые для них.
3. Туристические места – обширная категория, включающая разнообразные объекты, такие как парки, музеи, памятники и т.д.
4. Архитектурные места – объекты с выдающимися архитектурными особенностями, такие как здания, мосты, храмы и т.п.

Архитектура MVVM позволяет эффективно управлять отображением данных и обработкой событий, связанных с навигацией по каталогу. Это обеспечивает гибкость и модульность приложения, а также упрощает его расширение и модификацию.



При использовании поисковой панели приложение предлагает пользователю подсказки, основанные на популярных запросах и релевантных результатах. Это позволяет ускорить процесс поиска и обеспечить более точные результаты.

Нулевой сниппет с ViewPager (<https://developer.android.com/reference/androidx/viewpager/widget/ViewPager>) и тремя вкладками обеспечивает удобное и интуитивно понятное взаимодействие пользователя с приложением. Он предоставляет быстрый доступ к основным категориям туристических объектов, что позволяет сэкономить время при выборе интересующих мест.

При переходе на страницу с подробной информацией о выбранном объекте пользователь получает доступ к разнообразным данным, таким как история места, фотографии, видеоматериалы, а также отзывы и рейтинги других пользователей. Это обогащает пользовательский опыт и позволяет получить максимум информации о туристическом объекте перед принятием решения о его посещении.

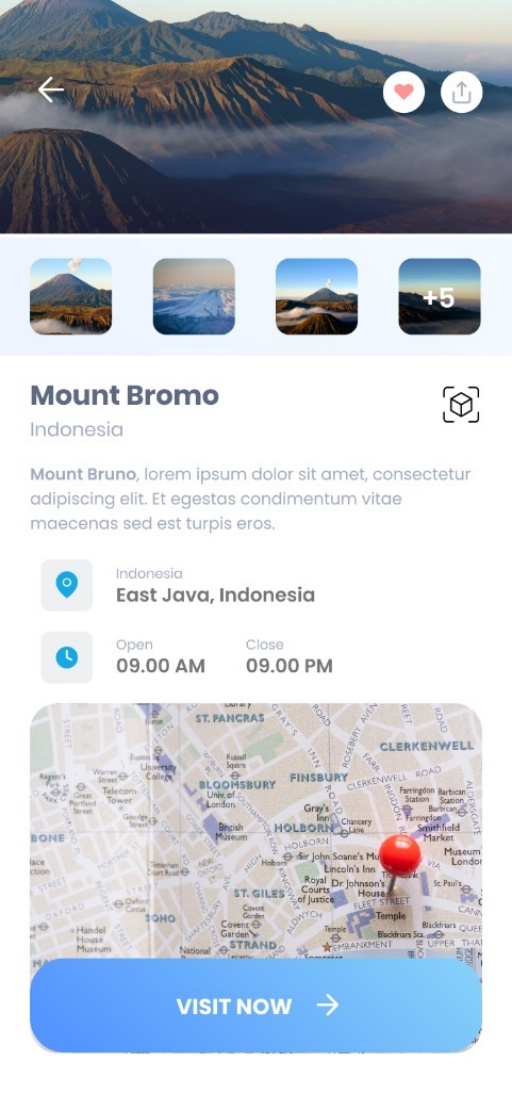


Рисунок 5 Информация о сниппете

Раздел информации включает описание, название достопримечательности и видеоматериалы с изображениями. Также имеется кнопка для просмотра указанной достопримечательности в режиме дополненной реальности (AR). При нажатии на эту кнопку пользователь переходит на сайт, где отображается соответствующая модель.

override fun onCreate(savedInstanceState: Bundle?) {  
 onTransformationStartContainer()  
 super.onCreate(savedInstanceState)  
 binding {  
 lifecycleOwner = this@DetailActivity  
 component = this@DetailActivity.architectureInfo  
 wikiBtn.setOnClickListener {  
 WebViewActivity.startActivity(  
 view = root,  
 url = architectureInfo.wikipediaUrl,  
 title = architectureInfo.name  
 )  
 }  
 arBtn.setOnClickListener {  
 startActivity(Intent(Intent.ACTION\_VIEW, Uri.parse(architectureInfo.arUrl)))  
 }  
 }  
 connectivityObserver.observe().onEach { status ->  
 when (status.name) {  
 "Available" -> {  
 setActivationBtn(true)  
 }  
 "Unavailable" -> {  
 setActivationBtn(false)  
 }  
 else -> {  
 setActivationBtn(false)  
 }  
 }  
 }.launchIn(lifecycleScope)  
}

В информационном блоке также представлена карта, облегчающая навигацию к местоположению архитектурных объектов и позволяющая планировать маршруты с использованием разнообразных видов транспорта или пешеходных путей. Для этого реализована интеграция с Map SDK (<https://developers.google.com/maps/documentation/android-sdk/start?hl=ru>).

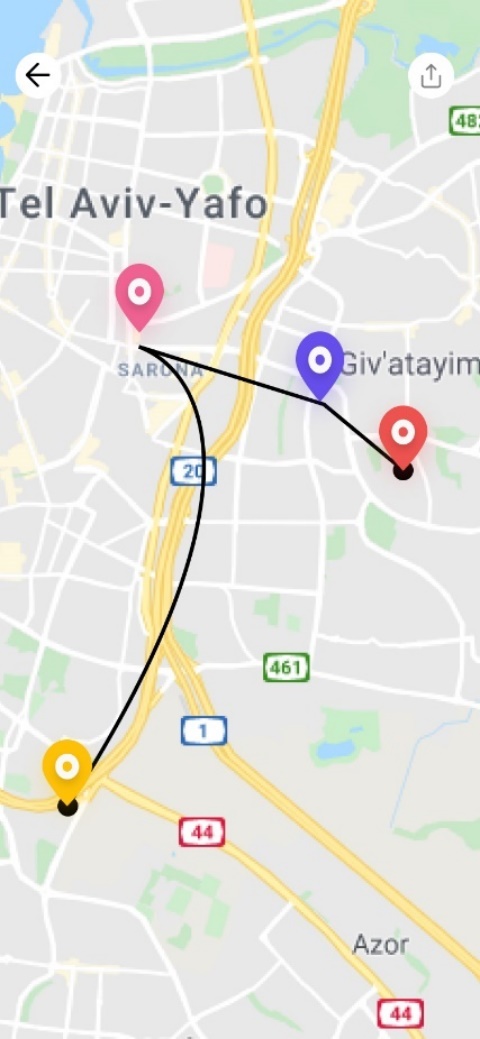


Рисунок 6 Путь до достопримечательности

#### **Шифрование данных**

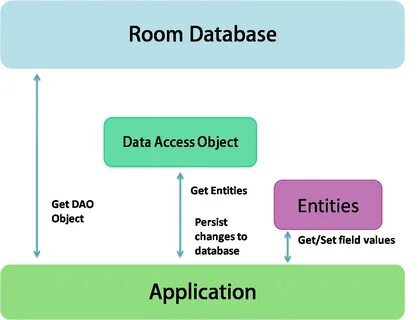
Для обеспечения безопасности пользовательских данных и конфиденциальности, приложение будет использовать современные методы шифрования и алгоритмы, такие как SSL/TLS (<https://developer.android.com/training/articles/security-ssl>), для передачи данных между клиентом и сервером. Это позволит защитить данные от несанкционированного доступа и утечки.

#### **Взаимодействие и обновление токенов**

Приложение будет использовать механизмы аутентификации и авторизации на основе токенов, такие как OAuth 2.0, для взаимодействия с внешними сервисами, такими как Google, Yandex и Facebook. Это обеспечит безопасный вход пользователя в систему. Приложение будет периодически обновлять токены для поддержания безопасности сессии.

#### **Получение данных из локального хранилища**

Приложение будет использовать локальное хранилище (Room) для хранения пользовательских данных, таких как избранные достопримечательности, настройки и кэшированный контент. Это позволит оптимизировать производительность приложения и обеспечит быстрый доступ к данным, даже при отсутствии интернет-соединения. В случае необходимости, приложение будет обновлять данные из локального хранилища, синхронизируя их с сервером.



@Database(  
 entities = [ArchitectureInfo::class],  
 version = 1,  
 exportSchema = true  
)  
abstract class AppDatabase : RoomDatabase() {  
  
 abstract fun architectureInfoDao(): ArchitectureInfoDao  
}  
  
  
@Dao  
interface ArchitectureInfoDao {  
  
 @Insert(onConflict = OnConflictStrategy.REPLACE)  
 suspend fun insertArchitectureInfo(architectureInfo: ArchitectureInfo)  
  
 @Query("SELECT \* FROM ArchitectureInfo WHERE id = :id\_")  
 suspend fun getArchitecture(id\_: Int): ArchitectureInfo?  
  
 @Query("SELECT \* FROM ArchitectureInfo ORDER BY id ASC")  
 fun getArchitectureInfo(): Flow<List<ArchitectureInfo>>  
}

## **Описание и обоснование выбора состава технических и программных средств**

### **Состав технических и программных средств**

Мобильное приложение с операционной системой Android 9.0 или выше.

### **Обоснование выбора технических и программных средств**

Приложение разрабатывалось под оптимальную рекомендованную версию операционной системы Android 9.0 «Pie» API 28. Так как, исходя из официального сайта API Levels (на 08.04.2023) процент пользователей, кто использует версию 9.0 и выше составляют 89.4%.

# **ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ**

## **Ориентировочная экономическая эффективность**

В рамках данной работы расчёт экономической эффективности не предусмотрен.

## **Предполагаемая потребность**

В отличие от обычных путешествий, связанных с тратами на авиаперелеты, проживание и транспорт, данное приложение дает возможность пользователям осуществлять виртуальные экскурсии из дома, сэкономив средства на путешественческих расходах. Более того, приложение способно предложить более обширный ассортимент туров и культурных впечатлений за малую часть стоимости реального путешествия в каждую из точек назначения. Это делает виртуальные экскурсии привлекательным вариантом для туристов с ограниченным бюджетом и тех, кто физически неспособен отправиться в поездку.

## **Экономические преимущества разработки по сравнению с отечественными и зарубежными аналогами**

В рамках анализа конкурентов приложения были выделены следующие основные характеристики для сравнения:

* + 1. Функциональность – разные ключевые функции приложения.
    2. Кроссплатформенность – приложение портируется на разные платформы.
    3. Удобство использования – целевой аудитории удобно взаимодействовать с данным приложением.
    4. Безопасность данных – защищённость персональных данных.
    5. Пользовательский опыт – приложение может иметь удобный интерфейс, высококачественную графику и интерактивные функции, которые делают виртуальный тур увлекательным и приятным.
    6. Доступность информации – приложение может предоставлять обширную и подробную информацию о достопримечательностях, включая их историю, факты и мультимедийные ресурсы, такие как фотографии и видео.
    7. Интерактивность – приложение может позволить пользователям взаимодействовать с виртуальными картами и совершать виртуальные туры, что может улучшить общее впечатление.
    8. Целевая аудитория – приложение может быть специально ориентировано на туристов, гидов или всех, кто заинтересован узнать больше о достопримечательностях.

Процесс поиска аналогов в сети Интернет позволил выявить следующих прямых конкурентов:

Далее приведена таблица анализа указанных конкурентов по описанным характеристикам. По каждому критерию выставляется оценка по пятибалльной шкале.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Критерии / Название приложения | Яндекс Карты [10] | Art Station [11] | Google Карты [12] | The Secret Door [13] | Redigo [14] | Путешествие и точка |
| Функциональность | 5 | 3 | 5 | 3 | 2 | 4 |
| Кроссплатформенность | 5 | 3 | 5 | 0 | 4 | 2 |
| Удобство использования | 5 | 4 | 4 | 3 | 2 | 4 |
| Интерактивность | 4 | 3 | 4 | 3 | 2 | 4 |
| Целевая аудитория | 4 | 4 | 4 | 3 | 2 | 3 |
| Оценка: | **4,6** | **3,4** | **4,4** | **2,4** | **2,6** | **3,4** |

# **ПРИЛОЖЕНИЕ 1**

1. **СПИСОК ИСПОЛЬЗУЕМОЙ ЛИТЕРАТУРЫ**
2. ГОСТ 19.101–77 Виды программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.102–77 Стадии разработки. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.103–77 Обозначения программ и программных документов. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.104–78 Основные надписи. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.105–78 Общие требования к программным документам. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.106–78 Требования к программным документам, выполненным печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.404–79 Пояснительная записка. Требования к содержанию и оформлению. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.603–78 Общие правила внесения изменений. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 19.604–78 Правила внесения изменений в программные документы, выполненные печатным способом. // Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
11. Яндекс Карты // Yandex Maps [Электронный ресурс]. Режим доступа: <https://yandex.ru/maps> (Дата обращения 03.04.2023).
12. Art Station [Электронный ресурс]. Режим доступа: <https://www.artstation.com/> (Дата обращения 03.02.2023)
13. Google Карты // Google Maps [Электронный ресурс]. Режим доступа: <https://www.google.com/maps/> (Дата обращения 03.04.2023)
14. The Secret Door // Потайная дверь [Электронный ресурс]. Режим доступа: <http://secretdoor.notepadwebdevelopment.com/> (Дата обращения 04.04.2023)
15. Redigo [Электронный ресурс]. Режим доступа: <https://redigo-ru.livejournal.com/> (Дата обращения (07.04.2023)
16. API levels and Android versions [Электронный ресурс] // URL: <https://apilevels.com/> (Дата обращения 06.04.2023, режим доступа свободный)

# **ПРИЛОЖЕНИЕ** **2**

**ТЕРМИНОЛОГИЯ**

1. **Activity** – является отдельным экраном в Android.
2. **Android SDK** – средство разработки моб. приложений.
3. **Fragment** – является частью модуля Activity.
4. **AR ­–** дополненная реальность
5. **DI –** внедрение зависимостей
6. **Cниппет –** элемент пользовательского интерфейса
7. **3D-Просмотрщик –** ПО, которое позволяет пользователю видеть и исследовать трехмерные объекты или сцены в виртуальной реальности.

**ПРИЛОЖЕНИЕ 3**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ КЛАССОВ**

Таблица 3.1

Описание и функциональное назначение классов мобильной части

| Класс | Назначение |
| --- | --- |
| NetworkModule | Часть DI |
| PersistenceModule | Часть DI |
| RecyclerViewBinding | Часть DI |
| RepositoryModule | Часть DI |
| ViewBinding | Часть DI |
| Contstants | Содержатся константные поля |
| ArchitectureInfo | Содержит информацию об сниппете |
| ArchitectureClient | Клиент для взаимодействия с API получения списка данных |
| ArchitectureService | API для взаимодействия с domain слоем |
| ConnectivityObserver | Интерфейс, в котором содержится функциональная возможность проверки интернет-соединения |
| HttpRequestInterceptor | Класс – логгирует данные с API |
| NetworkConnectivityObserver | Класс – наблюдатель для проверки наличии интернет-соединения |
| AppDatabase | Абстрактный класс для сохранения данных в БД |
| ArchitectureInfo | Интерфейс для взаимодействия с БД |
| MainActivity | Класс – точка входа приложения |
| DetailFragment | Класс-фрагмент для показа информации о сниппете |
| CatalogFragment | Класс-фрагмент для показа нескольких блоков информации |
| ProfileFragment | Класс-фрагмент для показа информации о пользователе |
| WebViewActivity | Класс-который навигируется в веб страницу через приложение |
| ArchitectureAdapter | Класс-адаптер для обработки компактного списка данных и отображения в UI |
| ChipsAdapter | Класс-адаптер для обработки большого списка данных и отображения в UI |
| MainRepository | Класс-репозиторий отвечает за получение и обновление списка архитектурных объектов приложении |
| MainViewModel | ViewModel который предоставляет данные и управлет логикой пользовательского интерфейса для экрана приложения. Он использует MainRepository. Для получения и обновления списка объектов |
| App | DI |

# **ПРИЛОЖЕНИЕ 4**

**ОПИСАНИЕ И ФУНКЦИОНАЛЬНОЕ НАЗНАЧЕНИЕ ПОЛЕЙ И ФУНКЦИЙ**

Таблица 4.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методы |  |  |  |  |
| Класс | NetworkModule.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| provideOkHttpClient | public | Метод | - | Метод для внедрения OkHttp в зависимость |
| provideRetrofit | public | Метод | okHttpClient: OkHttpClient | Метод для внедрения Retrofit в зависимость |
| provideArchitectureService | public | Метод | retrofit: Retrofit | Метод, который предоставляет сервис для работы с API |
| provideArchitectureClient | public | Метод | architectureService: ArchitectureSer | Метод для предоставления работы с архитектурным сервисом API |
| Методы |  |  |  |  |
| Класс | PersistenceModule.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| provideMoshi | public | Метод | - | Создает и предоставляет экземпляр Moshi, который используется для сериализации и десериализации JSON. |
| provideAppDatabase | public | Метод | application: Application | Создает и предоставляет экземпляр базы данных AppDatabase с использованием библиотеки Room. |
| provideArchInfoDao | public | Метод | appDatabase: AppDatabase | Предоставляет экземпляр ArchitectureInfoDao из AppDatabase для доступа к операциям базы данных. |
| Методы |  |  |  |  |
| Класс | RecyclerViewBinding.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| bindAdapter | public | Метод | view: RecyclerView, adapter: RecyclerView.Adapter<\*> | Устанавливает адаптер для RecyclerView, задает политику восстановления состояния адаптера и применяет ее. |
| bindAdapterArchList | public | Метод | view: RecyclerView, archList: List<ArchitectureInfo> | Устанавливает список архитектурных объектов archList для адаптера ArchitectureAdapter, если список не пуст и адаптер является экземпляром ArchitectureAdapter. |
| Методы |  |  |  |  |
| Класс | RepositoryModule.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| provideDetailRepository | public | Метод | archClient: ArchitectureClient,  archInfoDao: ArchitectureInfoDao | Создает и предоставляет экземпляр MainRepository с использованием ArchitectureClient и ArchitectureInfoDao для обработки данных из API и локальной базы данных. |
| Методы |  |  |  |  |
| Класс | ViewBinding.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| bindToast | public | Метод | view: View, text: String? | Создает и показывает всплывающее сообщение (Toast) с заданным текстом на View. |
| bindLoadImagePalette | public | Метод | view: AppCompatImageView, url: String, paletteCard: MaterialCardView | Загружает изображение с помощью библиотеки Glide, а затем устанавливает доминирующий цвет изображения как фоновый цвет MaterialCardView. |
| bindLoadImage | public | Метод | view: AppCompatImageView, url: String | Загружает изображение с помощью библиотеки Glide и устанавливает его в AppCompatImageView. |
| bindGone | public | Метод | view: View, shouldBeGone: Boolean | Устанавливает видимость View на основе значения shouldBeGone. Если значение shouldBeGone равно true, View становится невидимым (View.GONE), иначе становится видимым (View.VISIBLE). |
| Свойства |  |  |  |  |
| Класс | Constants.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| BASE\_URL | public | String | get | Ссылка для получения замоканных данных |
| Свойства |  |  |  |  |
| Класс | ArchitectureInfo.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| Id | public | Int | get, set | Идентификатор архитектурного объекта; является первичным ключом в базе данных. |
| Name | public | String | get, set | Название архитектурного объекта. |
| Description | public | String | get, set | Описание архитектурного объекта. |
| imageUrl | public | String | get, set | URL-адрес изображения архитектурного объекта. |
| WikipediaUrl | public | String | get, set | URL-адрес страницы Википедии, посвященной архитектурному объекту. |
| arUrl | public | String | get, set | URL-адрес ресурса с дополненной реальностью (AR), связанного с архитектурным объектом. |
| Методы |  |  |  |  |
| Класс | ArchitectureClient.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| fetchArchitectureInfo | public | Suspend метод | - | Получает список архитектурных объектов с помощью ArchitectureService и возвращает ApiResponse. |
| Методы |  |  |  |  |
| Класс | ArchitectureService.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| fetchArchitectureInfo | public | Suspend метод | - | Получает список архитектурных объектов с помощью ArchitectureService и возвращает ApiResponse. |
| Методы |  |  |  |  |
| Класс | ConnectivityObserver.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| observe | public | Метод | - | Возвращает поток Flow<Status> для наблюдения за изменениями статуса сетевого подключения. |
| Свойства |  |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| Status | public | Enum | get | Available Сетевое подключение доступно. Unavailable Сетевое подключение недоступно. Losing Сетевое подключение теряется (например, когда качество подключения ухудшается). Lost |
| Методы |  |  |  |  |
| Класс | HttpRequestInterceptor.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| intercept | public | Метод | Chain: Interceptor.Chain | Логгирование |
| Методы |  |  |  |  |
| Класс | NetworkConnectivityObserver.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| observe | public | Метод | - | Возвращает поток Flow<ConnectivityObserver.Status> для наблюдения за изменениями статуса сетевого подключения. |
| Свойства |  |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| context | public | Context | get | - |
| connectivityManager | private | SystemService | get | - |
| Методы |  |  |  |  |
| Класс | AppDatabase.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| architectureInfoDao | abstract | Метод | - | Абстрактный метод, который предоставляет доступ к объекту ArchitectureInfoDao. В реализации Room базы данных этот метод будет автоматически реализован, и в результате будет возвращать объект ArchitectureInfoDao для работы с архитектурными объектами. |
| Методы |  |  |  |  |
| Класс | ArchitectureInfoDao.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| insertArchitectureInfo | Public | Метод | architectureInfo: ArchitectureInfo | Вставляет объект ArchitectureInfo в базу данных с использованием стратегии замещения при конфликте (REPLACE). |
| getArchitecture | Public | Метод | id\_: Int | Получает архитектурный объект по его идентификатору (id). Если объект не найден, возвращает null. |
| getArchitectureInfo | public | Метод | - | Возвращает поток Flow<List<ArchitectureInfo>> со всеми архитектурными объектами, отсортированными в порядке возрастания идентификатора. |
| Методы |  |  |  |  |
| Класс | ArchitectureAdapter.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| setArchList | public | Метод | archList: List<ArchitectureInfo> | Устанавливает новый список архитектурных объектов и обновляет RecyclerView. |
| onCreateViewHolder | public | Метод | parent: ViewGroup, viewType: Int | Создает новый элемент списка архитектурных объектов с использованием связывания данных (Data Binding). |
| onBindViewHolder | Public | Метод | holder: ArchViewHolder, position: Int | Связывает элемент списка архитектурных объектов с данными, используя связывание данных (Data Binding). |
| getItemCount | public | Метод |  | Возвращает количество элементов в списке архитектурных объектов. |
| Методы |  |  |  |  |
| Класс | MainRepository.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| getArchitectureList | public | Метод | onStart: () -> Unit,  onError: (String?) -> Unit | Возвращает список архитектурных объектов из базы данных в виде потока Flow<List<ArchitectureInfo>>. В случае ошибки вызывает onError(). |
| updateArchitectureInfo | public | Метод |  | Запрашивает список архитектурных объектов с сервера и возвращает его. |
| insertArchitectureInfo | public | Метод | architectureInfo: List<ArchitectureInfo> | Вставляет список архитектурных объектов в базу данных, если их там нет. |
| Свойства |  |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| architectureClient | private | ArchitectureClient | get | - |
| architectureInfoDao | private | ArchitectureInfoDao | get | - |
| Методы |  |  |  |  |
| Класс | MainViewModel.kt |  |  |  |
| Имя | Модификатор доступа | Тип | Аргументы | Назначение |
| init | public |  |  | Инициализационный блок, запускает корутину для обновления архитектурных данных и загрузки их в локальное хранилище, а также устанавливает значение для \_archLiveData. |
| Свойства |  |  |  |  |
| Имя | Модификатор доступа | Тип | Доступ | Назначение |
| errorMessage | public | String? | get, set | Свойство хранит сообщение об ошибке. |
| isLoading | public | Boolean | get, set | Свойство указывает на то, загружаются ли данные. |
| architectureListFlow | private | Flow<List<ArchitectureInfo>> | get, set | Приватный поток Flow для получения списка архитектурных объектов из MainRepository. |
| architectureList | public | List<ArchitectureInfo>? | get, set | Свойство, связанное с архитектурным списком, полученным из architectureListFlow. |
| mainRepository | private | MainRepository | get | LiveData для наблюдения за списком архитектурных объектов. |

**ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Лист регистрации изменений | | | | | | | | | |
| Номера листов (страниц) | | | | | Всего листов (страниц в докум.) | № документа | Входящий № сопроводительного докум. и дата | Подп. | Дата |
| Изм. | Измененных | Замененных | Новых | Аннулированных |
| 1 | 26 |  | 26 |  | 26 | 1 | C:\Users\User\Downloads\IMG_5546.png |  | 08.04.2023 |
| 2 | 8 | 5 | 2 |  | 28 | 2 |  | C:\Users\User\Downloads\IMG_5546.png | 08.04.2023 |
| 3 | 5 | 6 | 2 | 2 | 30 | 3 | C:\Users\User\Downloads\IMG_5546.png | C:\Users\User\Downloads\IMG_5546.png | 08.04.2023 |
| 4 | 3 | - | 3 |  | 33 | 4 |  |  | 10.04.2023 |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |