МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА»
(РУТ (МИИТ)

Рабочая программа дисциплины (модуля),
как компонент образовательной программы
базового высшего образования
по направлению подготовки
09.03.02 Информационные системы и технологии,
утвержденной первым проректором РУТ (МИИТ)
Тимониным В.С.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Разработка мобильных приложений

Направление подготовки: 09.03.02 Информационные системы и технологии

Направленность (профиль): Технологии искусственного интеллекта в транспортных системах

Форма обучения: Очная

1. Общие сведения о дисциплине (модуле).

Дисциплина формирует фундаментальные и прикладные компетенции в области нативной мобильной разработки под платформу Android с использованием языка Kotlin. Студенты изучают современный декларативный подход к созданию пользовательских интерфейсов, передовые архитектурные паттерны проектирования и механизмы работы с локальными и удаленными источниками данных. Особое внимание уделяется созданию отказоустойчивых приложений, способных функционировать в условиях нестабильного сетевого соединения по принципу offline-first. В рамках курса моделируется полноценный инженерный цикл предприятия, включающий декомпозицию требований, настройку внедрения зависимостей, написание многоуровневых тестов и подготовку кодовой базы к релизу. Выпускник приобретает навыки, востребованные ведущими компаниями финтех-сектора и ИТ-экосистем.

Целью освоения дисциплины является формирование у студентов системных инженерных компетенций для самостоятельного проектирования, разработки и тестирования масштабируемых нативных мобильных приложений под платформу Android с применением современных декларативных фреймворков и принципов чистой архитектуры.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности: Анализировать предметную область и проектировать архитектуру клиентских приложений с разделением на изолированные слои. Разрабатывать адаптивные и доступные декларативные пользовательские интерфейсы с оптимизацией производительности. Организовывать безопасное сетевое взаимодействие и локальное кэширование данных для обеспечения работы без подключения к интернету. Настраивать фоновые процессы с учетом системных ограничений мобильной операционной среды. Покрывать критическую бизнес-логику и пользовательские сценарии автоматизированными тестами различных уровней.

2. Планируемые результаты обучения по дисциплине (модулю).

Перечень формируемых результатов освоения образовательной программы (компетенций) в результате обучения по дисциплине (модулю):

ПК-2 - Способен разрабатывать программные продукты с применением различных языков, технологических стеков и платформенных решений;

ПК-3 - Способен проводить многоуровневое тестирование программных продуктов с обеспечением заданных показателей качества.

Обучение по дисциплине (модулю) предполагает, что по его результатам обучающийся будет:

Знать:

- архитектуру операционной системы Android, компоненты приложения и механизмы управления их жизненным циклом в условиях нехватки системных ресурсов;
- современные идиомы и концепции языка Kotlin, применяемые в мобильной разработке (делегаты, sealed-классы, функции-расширения, инлайн-функции);
- модель многопоточности и реактивного программирования в Kotlin (Coroutines, Flow, Channels, Dispatchers, обработка исключений и отмена корутин);
- принципы проектирования SOLID и эволюцию архитектурных паттернов построения клиентских приложений (MVP, MVVM, MVI, Clean Architecture);
- концепцию инверсии управления (IoC) и механизмы внедрения зависимостей (DI) с применением библиотеки Hilt и управления скоупами компонентов;
- паттерны навигации в мобильных приложениях (Navigation Component, глубокие ссылки, управление состоянием бэкстека и передача аргументов);
- парадигму декларативного построения пользовательских интерфейсов в Jetpack Compose, принцип «UI как функция состояния» и работу композабл-функций;
- механизм рекомпозиции в Jetpack Compose, оптимизацию производительности UI-дерева и стратегии минимизации лишних перерисовок;
- стратегии управления состоянием (State Hoisting, remember, mutableStateOf) и обработку побочных эффектов (LaunchedEffect, DisposableEffect, SideEffect) в Compose;
- навигацию в Jetpack Compose (Navigation Compose, типы навигации, передача аргументов, глубокие ссылки) и интеграцию с ViewModel;
- стандарты Material Design 3 в Compose (MaterialTheme, цветовые схемы, типографика), принципы адаптивной верстки и обеспечение доступности (Accessibility/TalkBack);
- кастомные макеты (Custom Layouts), цепочки модификаторов (Modifier chain) и продвинутые анимации (animate*AsState, Animatable, Transition) в Jetpack Compose;

- архитектуру клиент-серверного взаимодействия по протоколу REST, контракты обмена данными (DTO) и стратегии сериализации (JSON, Kotlinx.serialization);

- инструментарий библиотек Retrofit и OkHttp для организации сетевых запросов, применение перехватчиков (Interceptors) и обработку сетевых ошибок;

- архитектуру локального хранения данных с использованием библиотеки Room (сущности, DAO, миграции) и реализацию паттерна Repository;

- концепцию Offline-First, стратегии кэширования и алгоритмы синхронизации данных между локальным хранилищем и удаленным сервером;

- механизмы безопасного хранения чувствительных данных на устройстве (Android Keystore, EncryptedSharedPreferences, шифрование файлов);

- подсистему планирования фоновых задач WorkManager, типы воркеров, гарантии выполнения и обработку системных ограничений (Constraints);

- модель системных разрешений (Permissions) в Android, декларативный запрос разрешений и взаимодействие с аппаратными компонентами устройства (Location, Camera);

- пирамиду тестирования мобильных приложений, методологию написания Unit-тестов с использованием JUnit и фреймворка мокирования Mockk;

- инструментарий для автоматизированного UI-тестирования декларативных интерфейсов (Compose Testing API, Semantics tree, тестовые правила);

- архитектуру системы сборки Gradle, настройку вариантов сборки (build types, product flavors), управление графами зависимостей и инструменты обфускации байт-кода (R8, ProGuard).

Уметь:

- проектировать архитектуру мобильного приложения, применяя принципы Clean Architecture и паттерн MVVM, при условии строгого разделения слоев Domain, Data и Presentation для обеспечения тестируемости бизнес-логики и ее независимости от Android SDK;

- разрабатывать декларативные пользовательские интерфейсы, используя Jetpack Compose и гайдлайны Material Design 3, при условии обеспечения адаптивности верстки, поддержки доступности и оптимизации лишних рекомпозиций UI-дерева;

- реализовывать локальное хранение и кэширование данных, применяя библиотеку Room и паттерн Repository, в условиях необходимости обеспечения работы приложения в режиме offline-first с гарантированной сохранностью пользовательских сессий и критических бизнес-сущностей;

- организовывать асинхронное сетевое взаимодействие с REST API, используя Retrofit, OkHttp и Kotlin Coroutines/Flow, при условии обработки сетевых ошибок, таймаутов, авторизации по токенам и безопасного маппинга DTO в доменные модели;

- настраивать внедрение зависимостей и управление скоупами компонентов, используя библиотеку Hilt, в условиях модульной структуры проекта и соблюдения принципов инверсии управления для устранения жестких связей между классами;

- проектировать и реализовывать многоуровневое тестирование, применяя JUnit 5, Mockk и Compose Testing API, при условии покрытия критических бизнес-сценариев и пользовательских путей для обеспечения заданных показателей качества;

- настраивать процессы фоновой синхронизации и выполнения отложенных задач, используя WorkManager, в условиях нестабильного сетевого соединения и строгих ограничений операционной системы Android на фоновую активность и расход батареи;

- подготавливать приложение к публикации и автоматизировать сборку, применяя инструменты обфускации R8/ProGuard и настройки CI/CD пайплайнов, при условии обеспечения безопасности хранения ключей подписи и оптимизации размера итогового артефакта.

Владеть:

- навыками декомпозиции сложных бизнес-требований на атомарные, тестируемые компоненты в рамках чистой архитектуры;

- методами отладки и поиска корневых причин ошибок в многопоточной среде и асинхронных цепочках Kotlin Coroutines;

- инструментарием профилирования и устранения утечек памяти в жизненном цикле мобильных компонентов;

- практиками написания самодокументируемого кода и ведения технической документации с использованием отечественного офисного программного обеспечения.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 6 з.е. (216 академических часа(ов).

3.2. Объем дисциплины (модуля) в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении учебных занятий:

Тип учебных занятий	Количество часов	
	Всего	Семестр №5
Контактная работа при проведении учебных занятий (всего):	80	80
В том числе:		
Занятия лекционного типа	32	32
Занятия семинарского типа	48	48

3.3. Объем дисциплины (модуля) в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации составляет 136 академических часа (ов).

3.4. При обучении по индивидуальному учебному плану, в том числе при ускоренном обучении, объем дисциплины (модуля) может быть реализован полностью в форме самостоятельной работы обучающихся, а также в форме контактной работы обучающихся с педагогическими работниками и (или) лицами, привлекаемыми к реализации образовательной программы на иных условиях, при проведении промежуточной аттестации.

4. Содержание дисциплины (модуля).

4.1. Занятия лекционного типа.

№ п/п	Тематика лекционных занятий / краткое содержание
1	Архитектура ОС Android и управление жизненным циклом компонентов Рассматриваемые вопросы: - архитектура операционной системы Android и основные компоненты приложения; - механизмы управления жизненным циклом компонентов в условиях нехватки системных ресурсов; - обработка конфигурационных изменений и сохранение состояния UI.
2	Современные идиомы языка Kotlin в мобильной разработке Рассматриваемые вопросы: - делегаты свойств и их применение для управления состоянием;

№ п/п	Тематика лекционных занятий / краткое содержание
	- sealed-классы и алгебраические типы данных для моделирования бизнес-логики; - функции-расширения и инлайн-функции для оптимизации производительности.
3	Kotlin Coroutines и Kotlin Flow: многопоточность и реактивное программирование Рассматриваемые вопросы: - модель многопоточности в Kotlin: Coroutines, Dispatchers, обработка исключений и отмена корутин; - реактивное программирование с использованием Flow и Channels; - стратегии обработки побочных эффектов и интеграция с жизненным циклом компонентов.
4	Принципы SOLID и архитектурные паттерны: от MVP до Clean Architecture Рассматриваемые вопросы: - принципы проектирования SOLID и их применение в мобильной разработке; - эволюция архитектурных паттернов: MVP, MVVM, MVI и их сравнительный анализ; - Clean Architecture: разделение слоёв Domain, Data и Presentation для обеспечения тестируемости.
5	Внедрение зависимостей: IoC, DI и библиотека Hilt Рассматриваемые вопросы: - концепция инверсии управления (IoC) и механизмы внедрения зависимостей; - архитектура библиотеки Hilt и управление скоупами компонентов; - применение DI в модульной структуре проекта для устранения жестких связей.
6	Jetpack Compose: декларативный UI и принцип «UI как функция состояния» Рассматриваемые вопросы: - парадигма декларативного построения пользовательских интерфейсов и отличие от императивного XML-подхода; - работа композабл-функций, аннотация @Composable и правила их создания; - базовые компоновщики (Column, Row, Box, LazyColumn) и структура UI-дерева.
7	Jetpack Compose: рекомпозиция, управление состоянием и побочные эффекты Рассматриваемые вопросы: - механизм рекомпозиции и оптимизация лишних перерисовок UI-дерева; - стратегии управления состоянием: State Hoisting, remember, mutableStateOf; - обработка побочных эффектов через LaunchedEffect, DisposableEffect и SideEffect.
8	Jetpack Compose: навигация, Material Design 3 и доступность Рассматриваемые вопросы: - декларативная навигация через Navigation Compose, передача аргументов и глубокие ссылки; - стандарты Material Design 3: MaterialTheme, цветовые схемы и типографика; - обеспечение доступности (Accessibility/TalkBack) и поддержка динамических шрифтов.
9	Jetpack Compose: кастомные макеты, модификаторы и анимации Рассматриваемые вопросы: - создание кастомных макетов (Custom Layouts) и работа с Placement; - цепочки модификаторов (Modifier chain) и их порядок применения; - продвинутые анимации: animate*AsState, Animatable, Transition API.
10	REST API: архитектура взаимодействия, контракты данных и сериализация Рассматриваемые вопросы: - архитектура клиент-серверного взаимодействия по протоколу REST; - контракты обмена данными (DTO) и стратегии версионирования API; - сериализация данных: JSON и Kotlinx.serialization.
11	Retrofit и OkHttp: сетевые запросы, перехватчики и обработка ошибок Рассматриваемые вопросы: - инструментарий библиотек Retrofit и OkHttp для организации сетевых запросов; - применение перехватчиков (Interceptors) для авторизации и логирования; - обработка сетевых ошибок, таймаутов и повторных попыток запросов.

№ п/п	Тематика лекционных занятий / краткое содержание
12	Room: архитектура локального хранения, миграции и паттерн Repository Рассматриваемые вопросы: - архитектура библиотеки Room: сущности, DAO и связи между таблицами; - стратегии миграции базы данных при изменении схемы; - реализация паттерна Repository для абстрагирования источников данных.
13	Offline-First: стратегии кэширования и алгоритмы синхронизации Рассматриваемые вопросы: - концепция Offline-First и её преимущества для мобильных приложений; - стратегии кэширования данных и управление актуальностью кэша; - алгоритмы синхронизации данных между локальным хранилищем и удаленным сервером.
14	Безопасность мобильного приложения: Android Keystore и шифрование данных Рассматриваемые вопросы: - механизмы безопасного хранения чувствительных данных (Android Keystore, EncryptedSharedPreferences); - шифрование файлов и защита от обратной разработки; - модель системных разрешений (Permissions) и декларативный запрос разрешений.
15	WorkManager: фоновые задачи и системные ограничения Android Рассматриваемые вопросы: - подсистема планирования фоновых задач WorkManager и типы воркеров; - гарантии выполнения задач и обработка системных ограничений (Constraints); - взаимодействие с аппаратными компонентами устройства (Location, Camera).
16	Инженерия качества: тестирование и система сборки Gradle Рассматриваемые вопросы: - пирамида тестирования и методология Unit-тестов с использованием JUnit и Mockk; - автоматизированное UI-тестирование декларативных интерфейсов (Compose Testing API, Semantics tree); - архитектура Gradle, варианты сборки (build types, product flavors) и инструменты обфускации (R8, ProGuard).

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Настройка базовой навигации и жизненного цикла Студент создает стартовую Activity и настраивает граф навигации с передачей аргументов между экранами. В ходе работы осуществляется обработка конфигурационных изменений и сохранение состояния UI при поворотах экрана.
2	Реализация базовых декларативных экранов Студент верстает стартовый экран и экран настроек с использованием базовых компоновщиков и цепочек модификаторов. Выполняется интеграция экранов с ViewModel для первичного отображения статических данных.
3	Создание переиспользуемых UI-компонентов и темизация Студент реализует кастомные элементы интерфейса и применяет стандарты Material Design 3 через MaterialTheme. Осуществляется настройка динамических цветов и поддержка адаптивной верстки для различных размеров экранов.
4	Обработка побочных эффектов и анимации Студент внедряет механизмы LaunchedEffect и DisposableEffect для корректного запуска

№ п/п	Наименование лабораторных работ / краткое содержание
	асинхронных операций из UI. Реализуются анимации переходов между состояниями элементов списка с использованием API Animatable.
5	Интеграция REST API и сериализация данных Студент настраивает клиент Retrofit с конвертером Kotlinx.serialization и реализует методы для получения списка маршрутов. Выполняется обработка различных HTTP-кодов ответов и преобразование сетевых исключений в пользовательские сообщения.
6	Реализация пагинации и реактивных потоков Студент интегрирует библиотеку Paging 3 для порционной загрузки длинных списков документов из сети. Осуществляется связывание потока данных с UI через StateFlow во ViewModel с автоматической отменой запросов при уничтожении экрана.
7	Настройка Room и реализация DAO Студент описывает сущности базы данных и пишет SQL-запросы через аннотации Room для фильтрации и сортировки маршрутов. Осуществляется настройка автоматических миграций для безопасного обновления схемы базы данных между релизами.
8	Реализация паттерна Repository и единого источника истины Студент объединяет сетевой и локальный источники данных в рамках единого репозитория. Реализуется стратегия кэширования, при которой UI сначала получает данные из Room и фоном инициирует обновление из сети.
9	Реализация отложенных задач через WorkManager Студент создает CoroutineWorker для фоновой отправки черновиков документов на сервер. Настраиваются уникальные цепочки задач с политикой сохранения существующей работы и механизмом повторных попыток при сетевых сбоях.
10	Уведомления и мониторинг статуса синхронизации Студент настраивает создание каналов уведомлений для информирования пользователя об успешной или неудачной синхронизации. Интегрируется Flow для отображения прогресса фоновой загрузки непосредственно на главном экране приложения.
11	Работа с геолокацией и построение маршрутов Студент подключает FusedLocationProviderClient для получения актуальных координат устройства и запрашивает разрешения в рантайме. Осуществляется интеграция с SDK карт для отображения текущего местоположения и точек маршрута.
12	Интеграция камеры и сканирование штрихкодов Студент настраивает CameraX для захвата изображений и запускает анализатор кадров для распознавания QR-кодов. Реализуется сохранение полученных снимков во внутреннее хранилище приложения с привязкой к конкретному заданию.
13	Написание Unit-тестов для бизнес-логики Студент покрывает тестами ViewModel и интеракторы с использованием JUnit и Mockk для изоляции от сетевых запросов. Проверяются сценарии успешной загрузки, обработки ошибок и корректной трансформации доменных моделей.
14	Автоматизированное UI-тестирование в Compose Студент пишет тесты на пользовательские сценарии с использованием ComposeTestRule и Semantics tree. Осуществляется проверка корректности отображения состояний загрузки, ошибок и успешного ввода данных в формы.
15	Настройка Gradle и обфускация кода Студент настраивает подписывание релизных сборок и активирует инструменты R8 для уменьшения размера APK и защиты от декомпиляции. Создаются правила proguard-rules.pro для сохранения необходимых классов при работе с рефлекссией.
16	Формирование релизного артефакта и анализ утечек памяти Студент генерирует финальный Android App Bundle и проводит аудит приложения с помощью

№ п/п	Наименование лабораторных работ / краткое содержание
	LeakCanary. Устраняются выявленные утечки памяти в жизненном цикле компонентов и оптимизируется потребление ресурсов.

Практические занятия

№ п/п	Тематика практических занятий/краткое содержание
1	Проектирование архитектуры и инициализация проекта Студент анализирует предметную область приложения и рисует схему Clean Architecture с разделением на слои Domain, Data и Presentation. Затем проектируется базовая структура Gradle-модулей и конфигурируется внедрение зависимостей для изоляции бизнес-логики. Результатом является утвержденная архитектурная схема и спецификация DI-контейнера.
2	Проектирование компонентной базы UI и управление состоянием Студент разрабатывает дизайн-систему приложения и описывает стратегии State Hoisting для переиспользуемых декларативных компонентов. Проводится анализ сценариев обработки побочных эффектов при загрузке данных и взаимодействии с пользователем. Итогом служит спецификация UI-компонентов и карта состояний для ключевых экранов.
3	Проектирование сетевого слоя и контрактов API Студент изучает документацию внешнего сервера и формирует набор DTO-моделей для обмена данными с применением алгебраических типов. Разрабатывается структура перехватчиков для автоматического прикрепления токенов авторизации и логирования. Результатом является спроектированный интерфейс клиента и схема маппинга сетевых ответов в доменные ошибки.
4	Проектирование локальной базы данных и стратегии Offline-First Студент строит ER-диаграмму локального хранилища и описывает связи между сущностями маршрутных заданий и пользовательских отчетов. Формируется алгоритм разрешения конфликтов при синхронизации устаревших локальных данных с ответом сервера. Итогом выступает схема миграций и описание паттерна Repository с единым источником истины.
5	Проектирование фоновой синхронизации и системных ограничений Студент анализирует сценарии потери сети и формирует требования к отложенной отправке созданных черновиков документов на сервер. Разрабатывается конфигурация ограничений для запуска синхронизации только при наличии Wi-Fi и достаточного заряда батареи. Результатом является схема очереди фоновых задач и алгоритм обработки сетевых сбоев.
6	Проектирование интеграции с геолокацией и аппаратным обеспечением Студент формирует матрицу требуемых системных разрешений и сценарии обработки отказов пользователя в их предоставлении. Разрабатывается алгоритм фонового трекинга координат с учетом ограничений энергосбережения операционной системы. Итогом является спецификация взаимодействия с системными API и схема сохранения треков в локальную БД.
7	Проектирование стратегии многоуровневого тестирования Студент выделяет критические бизнес-сценарии синхронизации и формирует матрицу тестовых покрытий для ViewModel и UseCases. Разрабатываются моки внешних зависимостей и описываются ожидаемые состояния UI при различных сетевых ошибках. Результатом является план тестирования и набор тест-кейсов для последующей автоматизации.
8	Проектирование вариантов сборки и конвейера CI/CD Студент анализирует требования к разделению серверов разработки и продакшена и настраивает product flavors в конфигурации сборщика. Формируется скрипт автоматической сборки, линтинга и прогона тестов при каждом коммите в репозиторий. Итогом является конфигурация вариантов сборки и схема пайплайна непрерывной интеграции.

4.3. Самостоятельная работа обучающихся.

№ п/п	Вид самостоятельной работы
1	Изучение рекомендованной литературы.
2	Подготовка к лабораторным работам.
3	Выполнение курсового проекта.
4	Подготовка к промежуточной аттестации.
5	Подготовка к текущему контролю.

4.4. Примерный перечень тем курсовых проектов

Разработка мобильного приложения для курьерской службы с поддержкой offline-маршрутизации и фоновой синхронизации статусов доставки.

Проектирование и реализация мобильного терминала для инвентаризации склада с интеграцией камеры для сканирования штрихкодов и отложенной выгрузкой отчетов.

Разработка корпоративного мобильного портала с offline-доступом к базе знаний и безопасным хранением пользовательских сессий.

Создание мобильного клиента для мониторинга транспорта с фоновым трекингом геолокации и оптимизацией расхода батареи устройства.

Разработка приложения для выездных инженеров с локальным кэшированием медиафайлов и очередью фоновой загрузки данных на сервер.

Проектирование мобильного дневника пациента для телемедицины с шифрованием локальной базы данных и реактивным обновлением интерфейса.

Разработка мобильного приложения для агентов недвижимости с offline-картами, кэшированием изображений и пагинацией списков объектов.

Создание системы мобильного аудита для торговых представителей с формированием локальных отчетов и автоматической фоновой синхронизацией.

Разработка мобильного гида с локальной базой данных экспонатов и триггерными уведомлениями на основе геозон.

Проектирование приложения для учета личных финансов с двухсторонней синхронизацией данных и обеспечением безопасности локального хранилища.

Разработка мобильного клиента для проведения полевых социологических опросов с гарантированной очередью отправки ответов при нестабильной сети.

Создание приложения для управления локальной сетью IoT-устройств с кэшированием состояний и обработкой разрывов соединения.

Разработка мобильного клиента для библиотечной системы с offline-поиском по каталогу и фоновой загрузкой метаданных книг.

Проектирование приложения для волонтеров поисково-спасательного отряда с поддержкой offline-карт и обменом координатами в фоновом режиме.

Разработка мобильного образовательного тренажера с локальным хранением прогресса, адаптивным UI и фоновой загрузкой контентных модулей.

5. Перечень изданий, которые рекомендуется использовать при освоении дисциплины (модуля).

№ п/п	Библиографическое описание	Место доступа
1	Петросян, Л. Э. Разработка мобильных приложений на языке Kotlin : учебное пособие для вузов / Л. Э. Петросян, К. В. Гусев. — Санкт-Петербург : Лань, 2024. — 104 с. — ISBN 978-5-507-49186-5. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/414971 (дата обращения: 19.06.2026)
2	Коузен, К. Kotlin. Сборник рецептов / К. Коузен ; перевод с английского А. Н. Киселева. — Москва : ДМК Пресс, 2021. — 220 с. — ISBN 978-5-97060-883-8. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/241007 (дата обращения: 19.06.2026)
3	Митяков, Е. С. Современная разработка мобильных приложений на языке Kotlin: Практикум : учебное пособие / Е. С. Митяков, М. С. Поздняков. — Москва : РТУ МИРЭА, 2025. — 117 с. — ISBN 978-5-7339-2726-8. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/513041 (дата обращения: 19.06.2026)
4	Степанов, П. В. Разработка мобильных приложений под Android на языке Kotlin : учебное пособие / П. В. Степанов, Н. А. Приходько, А. И. Запорожских. — Москва : РТУ МИРЭА, 2025. — 889 с. — ISBN 978-5-7339-2596-7. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/504855 (дата обращения: 19.06.2026)
5	Воронцов, Ю. А. Платформы разработки мобильных приложений : учебное пособие / Ю. А. Воронцов, М. А. Овчинников, Е. А. Чернов. — Москва : РТУ МИРЭА, 2023. — 172 с. — ISBN 978-5-7339-1857-0. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/382436 (дата обращения: 19.06.2026)

6. Перечень современных профессиональных баз данных и информационных справочных систем, которые могут использоваться при освоении дисциплины (модуля).

официальная документация языка Kotlin (JetBrains) – https://kotlinlang.org/docs/android-overview.html;

официальная документация Android Developers (Google) – https://developer.android.com/kotlin;

документация и гайдлайны Jetpack Compose – https://developer.android.com/develop/ui/compose/tutorial;

спецификации и компоненты Material Design 3 – https://developer.android.com/develop/ui/compose/components;

архитектурные гайдлайны и Open-Source репозитории Google (Now in Android) – https://github.com/android/nowinandroid;

интерактивные руководства Google Codelabs по Android и Kotlin – https://developer.android.com/codelabs.

7. Перечень лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства, необходимого для освоения дисциплины (модуля).

отечественные операционные системы семейства Linux: Astra Linux, РЕД ОС и аналоги из реестра отечественного ПО;

офисные пакеты для подготовки отчетов и презентаций по ГОСТ: Р7-Офис, МойОфис;

интегрированные среды разработки и эмуляторы: Android Studio, AOSP Emulator, Waydroid;

системы сборки и управления зависимостями: Gradle, Kotlin DSL;

инструменты непрерывной интеграции и статического анализа кода: GitLab CI, Detekt, Android Lint;

системы управления локальными базами данных и тестирования API: DB Browser for SQLite, Hoppscotch, Postman;

библиотеки профилирования и поиска утечек памяти: LeakCanary, Android Profiler.

8. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю).

Учебные аудитории для проведения учебных занятий, оснащенные компьютерной техникой и наборами демонстрационного оборудования.

Для практических и лабораторных занятий – наличие персональных компьютеров вычислительного класса.

9. Форма промежуточной аттестации:

Курсовой проект в 5 семестре.
Экзамен в 5 семестре.

10. Оценочные материалы.

Оценочные материалы, применяемые при проведении промежуточной аттестации, разрабатываются в соответствии с локальным нормативным актом РУТ (МИИТ).

Авторы:

старший преподаватель кафедры
«Цифровые технологии управления
транспортными процессами» И.С. Разживайкин

Согласовано:

Заведующий кафедрой ЦТУТП В.Е. Нутович

Председатель учебно-методической
комиссии Н.А. Андриянова