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
09.03.01 Информатика и вычислительная техника,
утвержденной первым проректором РУТ (МИИТ)
Тимониным В.С.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Кроссплатформенная разработка приложений

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность (профиль): Технологии разработки программного обеспечения

Форма обучения: Очная

Рабочая программа дисциплины (модуля) в виде электронного документа выгружена из единой корпоративной информационной системы управления университетом и соответствует оригиналу

Простая электронная подпись, выданная РУТ (МИИТ)
ID подписи: 5665
Подписал: заведующий кафедрой Нутович Вероника Евгеньевна
Дата: 01.09.2026

1. Общие сведения о дисциплине (модуле).

Дисциплина формирует компетенции проектирования и разработки мультиплатформенных программных продуктов промышленного уровня. Студенты осваивают парадигмы декларативного построения пользовательских интерфейсов и выделения общей бизнес-логики в кроссплатформенное ядро. Практический фокус направлен на создание гибридных архитектур, интегрирующихся с нативными функциями мобильных и настольных операционных систем через платформенные каналы. Особое внимание уделяется использованию отечественного программного обеспечения и открытых технологических стеков, что обеспечивает подготовку инженеров к решению задач технологического суверенитета и последующей миграции решений под специализированные мобильные платформы.

Целью освоения дисциплины является формирование у обучающихся системных знаний и практических навыков проектирования, реализации и тестирования кроссплатформенных приложений с применением современных декларативных фреймворков и механизмов абстрагирования нативного кода для обеспечения их работоспособности на множестве целевых платформ.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности: Анализировать требования к мультиплатформенным системам и выбирать оптимальные архитектурные паттерны. Студенты осваивают механизмы управления асинхронным состоянием, построения отказоустойчивых сетевых клиентов и организации локального персистентного хранилища. Также решаются задачи интеграции кроссплатформенного ядра со специфичными аппаратными и программными интерфейсами целевых операционных систем и обеспечения промышленного качества кода посредством автоматизированного тестирования и профилирования производительности.

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.

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

Знать:

- парадигмы кроссплатформенной разработки и критерии выбора между подходами совместного использования интерфейса и совместного использования ядра;
- фазы жизненного цикла мобильного приложения и их влияние на архитектуру сохранения состояния;
- синтаксис, систему типов и модель асинхронного программирования языка Dart;
- принципы декларативного пользовательского интерфейса и внутреннее устройство деревьев виджетов, элементов и рендер-объектов;
- иерархию и классификацию виджетов, механизмы их перерисовки и оптимизации производительности;
- архитектурные паттерны управления состоянием, разделение слоев и обеспечение тестируемости бизнес-логики;
- механизмы декларативной маршрутизации, генерации маршрутов и обработки внешних ссылок;
- философию и архитектуру выделения общих модулей, иерархию исходных наборов и механизмы публикации артефактов;
- принцип платформенно-зависимых реализаций и стратегии абстрагирования нативных интерфейсов;
- модель асинхронного программирования, потоки данных и механизмы интеграции с жизненным циклом платформ;
- архитектуру сетевого взаимодействия, сериализацию данных и стратегии обработки ошибок транспортного уровня;
- паттерн репозитория с единым источником истины и стратегии кэширования для архитектуры offline-first;
- принцип типобезопасных запросов, генерацию кода из декларативных схем и управление миграциями баз данных;
- классификацию каналов взаимодействия, кодирование сообщений и асинхронную передачу данных между слоями;
- нативные политики разрешений, лучшие практики их запроса и ограничения на выполнение фоновых задач;
- пирамиду тестирования, фреймворки для автоматизации проверок и метрики покрытия кода;
- метрики качества кода, принципы рефакторинга и техники идентификации проблемных участков в мультиязычных проектах;

- стандарты оформления технической документации и структуру записей архитектурных решений;

- принципы импортозамещения, критерии выбора отечественного программного обеспечения и стратегии миграции;

- архитектурные особенности специализированных отечественных мобильных операционных систем и механизмы портирования решений.

Уметь:

- проектировать многоуровневую архитектуру кроссплатформенного приложения при помощи метода записи архитектурных решений в условиях необходимости обоснованного выбора между парадигмами совместного использования интерфейса и ядра;

- реализовывать декларативный пользовательский интерфейс при помощи стека виджетов фреймворка Flutter в условиях необходимости адаптации дизайна под нативные гайдлайны конкретных операционных систем;

- управлять асинхронным состоянием приложения при помощи архитектурных паттернов BLoC и Riverpod в условиях обработки реактивных потоков данных от сетевых запросов и нативных сенсоров;

- разрабатывать общие модули бизнес-логики и данных при помощи Kotlin Multiplatform в условиях строгой изоляции общего кода от платформенно-зависимых реализаций через механизм контрактов;

- реализовывать сетевое взаимодействие с серверными сервисами при помощи асинхронного HTTP-клиента Ktor в условиях обеспечения отказоустойчивости и работы при нестабильном соединении;

- проектировать локальное персистентное хранилище при помощи генератора типобезопасных запросов SQLDelight в условиях реализации архитектуры offline-first с фоновой синхронизацией;

- обеспечивать двустороннюю связь между кроссплатформенным ядром и нативным кодом при помощи платформенных каналов в условиях соблюдения нативных жизненных циклов и строгих политик запроса разрешений;

- покрывать автоматическими тестами бизнес-логику и компоненты интерфейса при помощи профильных фреймворков в условиях необходимости достижения заданного показателя покрытия и обеспечения регрессионной устойчивости;

- оформлять проектную и архитектурную документацию при помощи отечественного офисного пакета в условиях соблюдения требований государственных стандартов и корпоративных норм.

Владеть:

- навыками настройки среды разработки и автоматизированной сборки кроссплатформенных проектов под множество целевых платформ;
- приемами профилирования производительности, выявления утечек памяти и оптимизации рендеринга высокочастотной телеметрии;
- методами интеграции сторонних библиотек и настройки сборочных скриптов для управления зависимостями общего ядра;
- практиками использования систем контроля версий и организации процессов непрерывной интеграции для командной разработки;
- способами отладки взаимодействия между управляемым и нативным кодом с использованием специализированных инструментов разработчика.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 3 з.е. (108 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 №6
Контактная работа при проведении учебных занятий (всего):	64	64
В том числе:		
Занятия лекционного типа	32	32
Занятия семинарского типа	32	32

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 44 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	Парадигмы кроссплатформенной разработки и выбор технологического стека Рассматриваемые вопросы: - классификация подходов к кроссплатформенной разработке и их компромиссы; - критерии выбора парадигмы и технологического стека под конкретные бизнес-требования и ограничения; - обзор экосистем Flutter и Kotlin Multiplatform в контексте современных рыночных трендов.
2	Жизненный цикл мобильного приложения и основы архитектуры Рассматриваемые вопросы: - фазы создания, приостановки, восстановления и завершения работы мобильного приложения; - особенности обработки событий жизненного цикла в кроссплатформенных фреймворках; - влияние жизненного цикла на архитектуру сохранения состояния и управления ресурсами.
3	Язык Dart и модель асинхронного программирования Рассматриваемые вопросы: - синтаксис, система типов и ключевые особенности языка Dart; - механизм изолятов и многопоточность в Dart; - модели Future и Stream, конструкции async/await для неблокирующих операций в потоке интерфейса.
4	Принципы декларативного UI и компонентная модель Flutter Рассматриваемые вопросы: - концепция интерфейса как функции состояния и ее отличие от императивного подхода; - внутреннее устройство Flutter, включая деревья виджетов, элементов и рендер-объектов; - механизмы инкрементального обновления интерфейса и реактивного рендеринга.
5	Иерархия виджетов Flutter и оптимизация производительности Рассматриваемые вопросы: - различия между stateless, stateful и inherited виджетами; - механизмы перерисовки виджетов и управление их жизненным циклом; - оптимизация производительности через const-конструкторы и использование ключей.
6	Архитектурные паттерны управления состоянием во Flutter Рассматриваемые вопросы: - концептуальные основы и сценарии применения паттернов Provider, BLoC и Riverpod; - разделение слоев presentation, domain и data в архитектуре приложения; - обеспечение тестируемости бизнес-логики и работа с реактивными потоками данных.
7	Механизмы маршрутизации, навигации и deep linking Рассматриваемые вопросы: - декларативный подход к навигации с использованием Navigator 2.0 и генерация маршрутов; - реализация deep linking и обработка внешних ссылок на внутренние экраны приложения; - обработка системных кнопок навигации и интеграция с платформенными навигационными стеками.
8	Философия и архитектура Kotlin Multiplatform Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- концепция shared-модулей и выделение общей бизнес-логики из платформенного кода; - иерархия исходных наборов и настройка сборочных скриптов Gradle; - механизмы публикации артефактов и интеграция общих модулей в нативные проекты.
9	Механизм контрактов expect/actual и абстрагирование нативных API Рассматриваемые вопросы: - принцип платформенно-зависимых реализаций и правила объявления ожидаемых деклараций; - стратегии абстрагирования нативных API для предотвращения утечки платформенных типов в общий код; - лучшие практики проектирования интерфейсов shared-модулей для взаимодействия с нативным слоем.
10	Асинхронное программирование в Kotlin: Coroutines и Flow Рассматриваемые вопросы: - модель Coroutines, контексты выполнения и механизмы отмены асинхронных задач; - потоки данных Flow и Channels для обработки последовательностей асинхронных событий; - интеграция корутин с жизненным циклом мобильных платформ.
11	Архитектура сетевого взаимодействия на Ktor Client Рассматриваемые вопросы: - построение кроссплатформенного HTTP-клиента и настройка интерсепторов; - сериализация и десериализация данных с использованием профильных библиотек Kotlin; - стратегии обработки ошибок транспортного уровня, таймауты и повторные запросы.
12	Архитектура offline-first приложений и паттерн Repository Рассматриваемые вопросы: - паттерн репозитория с единым источником истины; - стратегии кэширования данных, включая Cache-First и Network-First; - механизмы фоновой синхронизации и разрешения конфликтов данных между клиентом и сервером.
13	Локальное персистентное хранилище и технология SQLDelight Рассматриваемые вопросы: - принцип типобезопасных SQL-запросов и генерация Kotlin-кода из декларативных файлов; - проектирование схем баз данных и управление миграциями в кроссплатформенной среде; - интеграция SQLDelight с Coroutines Flow для реализации реактивных выборок данных.
14	Механизмы Platform Channels во Flutter и нативные вызовы Рассматриваемые вопросы: - классификация каналов взаимодействия, включая MethodChannel и EventChannel; - кодирование сообщений и асинхронная передача данных между слоями; - обработка ошибок и обеспечение потокобезопасности при взаимодействии между Dart и нативным слоем.
15	Нативные политики разрешений, жизненные циклы и фоновые задачи Рассматриваемые вопросы: - модель Runtime Permissions в Android и Privacy Descriptions в iOS; - лучшие практики запроса разрешений и обработка отказов пользователя; - ограничения операционных систем на выполнение фоновых задач и стратегии работы в background-режиме.
16	Обеспечение качества, импортозамещение и миграция под ОС Аврора Рассматриваемые вопросы: - пирамида тестирования кроссплатформенных приложений и измерение test coverage для shared-модулей; - принципы импортозамещения в мобильной разработке и критерии выбора отечественного ПО; - архитектурные особенности ОС Аврора и роль platform channels как моста для будущей миграции.

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Настройка среды разработки и создание первого кроссплатформенного проекта Студент устанавливает комплект разработки Flutter и настраивает интегрированную среду для программирования. Затем он создает базовый проект и выполняет его успешную сборку и запуск на эмуляторе Android и симуляторе iOS. В завершение работы осуществляется проверка корректности развертывания приложения на физическом мобильном устройстве.
2	Построение декларативного пользовательского интерфейса и навигации Студент реализует основные экраны прототипа курсового проекта с использованием декларативных виджетов Flutter. В процессе работы настраивается маршрутизация между экранами и применяется стилизация в соответствии с гайдлайнами Material Design. Результатом является работающий пользовательский интерфейс с навигацией и базовым оформлением.
3	Асинхронная обработка данных и управление состоянием через Provider Студент подключает к пользовательскому интерфейсу источники тестовых данных и реализует асинхронную загрузку информации. Для управления реактивным состоянием приложения внедряется архитектурный паттерн Provider. Выполняется проверка корректного обновления интерфейса при изменении внутренних данных без перезагрузки экранов.
4	Архитектурное разделение слоев и реактивные потоки в паттерне BLoC Студент проводит рефакторинг презентационного слоя прототипа путем внедрения паттерна BLoC. Осуществляется строгое разделение бизнес-логики и компонентов пользовательского интерфейса с использованием реактивных потоков. В конце занятия тестируется обработка сложных пользовательских взаимодействий через созданные блоки.
5	Создание библиотеки компонентов и адаптация под платформенные гайдлайны Студент разрабатывает библиотеку переиспользуемых пользовательских виджетов для нужд курсового проекта. Реализуются адаптивные макеты, корректно отображающиеся на устройствах с различными размерами экрана и ориентацией. Дополнительно применяется стилизация Cupertino для обеспечения нативного внешнего вида интерфейса на платформе iOS.
6	Инициализация Kotlin Multiplatform и настройка сборочных скриптов Студент инициализирует модуль Kotlin Multiplatform внутри существующей структуры курсового проекта. Осуществляется настройка наборов исходного кода для общих, мобильных и настольных целевых платформ в сборочных скриптах Gradle. Завершающим этапом является успешная компиляция общего кода и подключение полученной библиотеки к нативным проектам.
7	Разработка сетевого слоя и сериализация данных на Ktor Client Студент реализует кроссплатформенный клиент HTTP для взаимодействия с серверной частью приложения. Настраивается сериализация и десериализация объектов JSON с использованием профильных библиотек Kotlin. Созданный сетевой модуль интегрируется с общим слоем бизнес-логики для обеспечения загрузки данных из сети.
8	Проектирование локального хранилища и реализация offline-first архитектуры Студент проектирует схему локальной реляционной базы данных с использованием декларативных файлов SQL. Генерируется типобезопасный программный интерфейс для выполнения операций чтения и записи на всех целевых платформах. Реализуется паттерн репозитория для кэширования сетевых ответов и обеспечения работы приложения без подключения к интернету.
9	Платформенные контракты и механизм expect/actual Студент объявляет ожидаемые интерфейсы в общем модуле для доступа к специфичным функциям устройства. Пишутся фактические реализации этих интерфейсов на языках Kotlin для Android и Swift для iOS. Осуществляется подключение созданных платформенных реализаций к общей доменной логике курсового проекта.

№ п/п	Наименование лабораторных работ / краткое содержание
10	Двусторонняя связь с нативным кодом через MethodChannel Студент настраивает канал методов для вызова нативных функций платформы из слоя Dart. Реализуется передача структурированных аргументов и обработка асинхронных ответов от нативного кода. Проверяется корректность двустороннего обмена данными и обновление пользовательского интерфейса Flutter на основе полученных результатов.
11	Потоковая передача телеметрии от нативных сенсоров через EventChannel Студент реализует канал событий для получения непрерывного потока телеметрических данных от нативных сенсоров устройства. Осуществляется обработка входящих сообщений и их преобразование в формат, понятный кроссплатформенному слою. Реализованные данные визуализируются в реальном времени на информационной панели приложения.
12	Интеграция службы уведомлений и обработка фоновых задач Студент интегрирует службу доставки уведомлений для отправки сообщений на мобильные устройства. Настраиваются обработчики фоновых задач, срабатывающие при закрытом приложении. Проверяется механизм обновления локальной базы данных при получении скрытых сигналов от сервера.
13	Управление системными разрешениями и жизненным циклом приложения Студент внедряет механизм запроса разрешений на доступ к геолокации и камере в реальном времени. Реализуется обработка сценариев отказа пользователя и повторного запроса системных привилегий. Осуществляется привязка ресурсоемких операций к событиям жизненного цикла приложения для предотвращения утечек памяти.
14	Модульное тестирование общей бизнес-логики на Kotlin Студент пишет модульные тесты для общей доменной логики и репозиториев данных на языке Kotlin. Создаются заглушки для сетевых и базовых зависимостей с целью изоляции тестируемых компонентов. Генерируется отчет о покрытии кода тестами для подтверждения соответствия общих модулей требованиям курсового проекта.
15	Виджет-тестирование и сквозная интеграционная проверка интерфейса Студент разрабатывает автоматизированные тесты для критически важных компонентов пользовательского интерфейса Flutter. Симулируются действия пользователя для проверки корректности маршрутизации и отображения данных. Пишется интеграционный тест, запускающий приложение и валидирующий сквозной пользовательский сценарий от старта до финального экрана.
16	Профилирование производительности и оптимизация рендеринга телеметрии Студент подключает инструменты профилирования для анализа потребления памяти и времени отрисовки кадров при поступлении высокочастотной телеметрии. В процессе работы выявляются узкие места, вызывающие задержки рендеринга интерфейса и утечки памяти на стыке общего кода и слоя представления. Осуществляется рефакторинг проблемных участков с выносом ресурсоемких вычислений в изоляты и оптимизацией перерисовки виджетов. Завершающим этапом является повторное профилирование для подтверждения стабильной частоты кадров и отсутствия деградации производительности.

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 кроссплатформенного приложения для сбора и визуализации телеметрии с нативных IoT-сенсоров с использованием EventChannel

Проектирование мобильного клиента для полевых изысканий с архитектурой offline-first и фоновой синхронизацией на базе SQLDelight

Адаптация корпоративного мессенджера для специализированных отечественных ОС (ОС Аврора) с использованием платформенных каналов

Создание фитнес-трекера с общим ядром на Kotlin Multiplatform и декларативным UI на Flutter, интегрирующимся с нативными шагомерами

Разработка системы курьерской доставки с офлайн-кэшированием маршрутов и обработкой фоновых push-уведомлений

Мобильное приложение для складского учета с использованием нативной камеры для сканирования QR-кодов и типобезопасным локальным хранилищем

Панель управления умным домом с потоковой передачей данных от устройств и адаптивным интерфейсом на Flutter

Клиентское приложение для банка с типобезопасным сетевым взаимодействием на Ktor и биометрической аутентификацией через нативные контракты

Аудиоплеер с фоновым воспроизведением, управлением из системного notification-центра и локальной базой данных плейлистов

Навигатор для закрытых территорий с загрузкой офлайн-карт и корректной обработкой отказов в геолокации согласно политикам Runtime Permissions

Корпоративный портал с единым ядром бизнес-логики на KMP и нативной адаптацией UI под гайдлайны различных операционных систем

Бортовой журнал водителя с автоматическим трекингом поездок, привязкой к жизненному циклу приложения и оптимизацией потребления памяти

Офлайн-платформа для просмотра учебных курсов с прогрессивной загрузкой медиа и кэшированием состояния прохождения через паттерн Repository

Агрономический дневник с интеграцией нативных API камеры для фиксации данных и асинхронной синхронизацией отчетов при появлении сети

Приложение для инспекторов безопасности с генерацией типобезопасных SQL-отчетов и экспортом данных в файловую систему через MethodChannel

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

№ п/п	Библиографическое описание	Место доступа
1	Синицын, И. В. Разработка мобильных приложений : учебное пособие / И. В. Синицын, Е. А. Чернов, Ю. А. Воронцов. — Москва : РТУ МИРЭА, 2023 — Часть 1 — 2023. — 162 с. — ISBN 978-5-7339-1799-3. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/368735 (дата обращения: 22.06.2026)
2	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: 22.06.2026)
3	Калгина, И. С. Разработка мобильных приложений : учебное пособие / И. С. Калгина. — Чита : ЗабГУ, 2022. — 163 с. — ISBN 978-5-9293-3137-4. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/363323 (дата обращения: 22.06.2026)
4	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: 22.06.2026)
5	Заметти, Ф. Flutter на практике : руководство / Ф. Заметти ; перевод с английского А. С. Тищенко. — Москва : ДМК Пресс, 2020. — 328 с. — ISBN 978-5-97060-808-1. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/179464 (дата обращения: 22.06.2026)
6	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: 22.06.2026)

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

Официальная документация фреймворка Flutter – сборник руководств и справочных материалов по построению декларативных интерфейсов (https://docs.flutter.dev/).

Руководства по языку Dart – документация по синтаксису, стандартным библиотекам и моделям асинхронного программирования (https://dart.dev/guides).

Документация Kotlin Multiplatform – справочные материалы по архитектуре shared-модулей и настройке сборочных скриптов (https://kotlinlang.org/docs/multiplatform.html).

Официальная документация Ktor – руководства по созданию асинхронных кроссплатформенных HTTP-клиентов (https://ktor.io/docs).

Справочная система SQLDelight – документация по генерации типобезопасных API из декларативных SQL-схем (https://cashapp.github.io/sqldelight/).

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

Операционные системы: отечественные дистрибутивы на базе ядра Linux, включая защищенные решения из реестра отечественного программного обеспечения.

Офисные пакеты: отечественные решения для оформления технической и проектной документации согласно требованиям государственных стандартов.

Среды разработки и редакторы: кроссплатформенные интегрированные среды и легковесные редакторы кода с поддержкой целевых языков программирования.

Технологический стек: открытые фреймворки и языки для декларативного построения интерфейсов, выделения общей бизнес-логики и асинхронного сетевого взаимодействия.

Системы контроля версий и автоматизации: распределенные системы управления кодом и платформы непрерывной интеграции, совместимые с отечественными операционными системами.

Системы управления базами данных: отечественные и открытые реляционные СУБД для серверной части, а также встраиваемые решения для локального кэширования.

Инструменты тестирования и анализа: встроенные фреймворки для модульного и виджет-тестирования, а также статические анализаторы качества кода.

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 лабораторных занятий – наличие персональных компьютеров вычислительного класса.

9. Форма промежуточной аттестации:

Зачет в 6 семестре.
Курсовой проект в 6 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