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)

Разработка под ОС Аврора

Направление подготовки: 09.03.01 Информатика и вычислительная техника

Направленность (профиль): Технологии разработки программного обеспечения

Форма обучения: Очная

1. Общие сведения о дисциплине (модуле).

Дисциплина «Разработка под ОС Аврора» направлена на формирование у студентов компетенций в области создания мобильных приложений для отечественной операционной системы, критически востребованной в государственном и корпоративном секторах в рамках политики импортозамещения. Студенты освоят нативный стек технологий на базе Qt и C++, научатся проектировать безопасные интерфейсы с использованием QML и фреймворка Silica, а также интегрировать переиспользуемую бизнес-логику, написанную на кроссплатформенных фреймворках Flutter и Kotlin Multiplatform. Практическая часть курса построена вокруг разработки полноценного корпоративного приложения с нуля – от настройки изолированной среды и работы с системными сервисами до криптографической подписи и подготовки дистрибутива для публикации в российских магазинах приложений.

Целью освоения дисциплины является формирование у обучающихся системных знаний и практических навыков проектирования, разработки и дистрибуции защищенных мобильных приложений под ОС «Аврора» с применением гибридных архитектурных подходов и нативных средств платформы.

Для достижения поставленной цели в рамках дисциплины решается комплекс задач, направленных на формирование у обучающихся способности: разрабатывать декларативные пользовательские интерфейсы в соответствии с гайдлайнами ОС «Аврора», реализовывать серверную логику и системные вызовы на языке C++ с учетом ограничений мобильной среды, настраивать безопасное взаимодействие с аппаратным обеспечением и системными сервисами, интегрировать кроссплатформенные модули бизнес-логики посредством нативных мостов, а также осуществлять сборку, тестирование и подготовку дистрибутивов к публикации в отечественных экосистемах.

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:

- архитектуру и компонентную модель операционной системы «Аврора» на базе ядра Linux и прослойки Sailfish;
- принципы изоляции приложений (sandboxing), модель разрешений и механизмы защиты данных в экосистеме ОС «Аврора»;
- жизненный цикл мобильного приложения в ОС «Аврора» и механизмы управления состояниями;
- синтаксис, декларативную природу и систему свойств языка QML для построения пользовательских интерфейсов;
- архитектуру и компоненты фреймворка Silica UI, специфика гайдлайнов UX/UI и принципы адаптивной верстки для ОС «Аврора»;
- механизмы связывания данных, обработки событий и аппаратно-ускоренной анимации в декларативных интерфейсах QML;
- особенности управления памятью, система смарт-указателей и иерархия объектов в фреймворке Qt;
- модель многопоточности, механизмы сигналов и слотов, а также обработка асинхронных событий в Qt Core;
- принципы взаимодействия декларативного слоя (QML) и императивного бэкенда (C++) через механизм контекста и экспорт классов;
- архитектуру встраиваемых СУБД и специфика работы с локальными базами данных в условиях ограниченных ресурсов мобильного устройства;
- механизмы объектно-реляционного отображения и асинхронных запросов к данным с использованием модуля Qt SQL;
- архитектурные паттерны и механизмы изоляции при интеграции кроссплатформенных фреймворков в нативную среду;
- протоколы межъязыкового взаимодействия для обмена данными между C++/QML и модулями Flutter/KMP;
- API системных сервисов ОС «Аврора» – механизмы работы с Push-уведомлениями, амбассадорами и фоновыми задачами;
- принципы безопасного взаимодействия с аппаратным обеспечением через системные интерфейсы и D-Bus в ОС «Аврора»;
- методы профилирования производительности, анализа потребления энергии и поиска утечек памяти в мобильных приложениях на базе Qt;
- архитектуру пакетного менеджера RPM, спецификация файла .spec и процесс криптографической подписи дистрибутивов;
- требования к ассетам, метаданным и процессу валидации приложений при публикации в отечественных магазинах;

- структуру, правила оформления и состав комплектов документов Единой системы программной документации (ЕСПД) и действующих стандартов ГОСТ.

Уметь:

- настраивать среду разработки, компиляторы и эмулятор при помощи Aurora IDE и Aurora SDK в условиях необходимости эмуляции специфичной аппаратной архитектуры и системных ограничений ОС «Аврора»;
- проектировать и верстать адаптивные пользовательские интерфейсы с использованием декларативного языка QML и нативных компонентов Silica в строгом соответствии с официальными гайдлайнами UX/UI;
- реализовывать серверную логику приложения и системные вызовы на языке C++ с применением фреймворка Qt Core с учетом жестких ограничений по управлению памятью;
- организовывать локальное хранение, кэширование и миграцию данных посредством СУБД SQLite и модуля Qt SQL в условиях работы внутри изолированного файлового пространства приложения;
- интегрировать переиспользуемые кроссплатформенные модули бизнес-логики на базе Flutter или KMP посредством настройки нативных мостов для бесшовного обмена данными;
- подключать и конфигурировать специфичные аврора-сервисы через Aurora API с обязательной программной реализацией запроса и обработки пользовательских разрешений;
- проводить профилирование производительности, поиск утечек памяти и отладку асинхронных процессов инструментами встроенного профайлера Qt Creator и Aurora SDK;
- собирать, криптографически подписывать и упаковывать итоговый дистрибутив приложения в формат RPM-пакета с соблюдением жестких требований валидации;
- оформлять техническую, эксплуатационную и пользовательскую документацию в отечественном офисном пакете в строгом соответствии с требованиями ЕСПД и ГОСТ.

Владеть:

- навыками работы с инструментарием Aurora SDK для сборки, отладки и эмуляции мобильных приложений;
- методами интеграции кроссплатформенных фреймворков в нативную среду ОС «Аврора» через FFI и Platform Channels;
- навыками оформления технической документации и подготовки дистрибутивов к публикации в отечественных витринах.

3. Объем дисциплины (модуля).

3.1. Общая трудоемкость дисциплины (модуля).

Общая трудоемкость дисциплины (модуля) составляет 4 з.е. (144 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 №7
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 80 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 и экосистема ОС «Аврора» Рассматриваемые вопросы: - компонентная модель операционной системы «Аврора» на базе ядра Linux и прослойки Sailfish;

№ п/п	Тематика лекционных занятий / краткое содержание
	- специфика мобильного применения в корпоративном и государственном секторах; - принципы изоляции приложений (sandboxing) и модель системных разрешений.
2	Жизненный цикл и многозадачность мобильных приложений Рассматриваемые вопросы: - состояния жизненного цикла приложения в ОС «Аврора» (активное, фоновое, приостановленное, уничтоженное); - механизмы управления состояниями и сохранения контекста; - особенности многозадачности и энергопотребления в мобильной среде.
3	Декларативный язык QML: синтаксис и базовые компоненты Рассматриваемые вопросы: - декларативная природа и система свойств языка QML; - структура QML-документов и иерархия визуальных элементов; - базовые компоненты и механизмы обработки пользовательских событий.
4	Фреймворк Silica UI и гайдлайны UX/UI для ОС «Аврора» Рассматриваемые вопросы: - архитектура и нативные компоненты фреймворка Silica UI; - официальные гайдлайны UX/UI и принципы адаптивной верстки; - паттерны навигации и специфика взаимодействия с сенсорным экраном.
5	Продвинутые интерфейсы на QML: связывание данных и анимация Рассматриваемые вопросы: - механизмы связывания данных (data binding) между компонентами; - создание аппаратно-ускоренной анимации и переходов; - оптимизация отрисовки декларативных интерфейсов.
6	Серверная логика на C++ и Qt Core: управление памятью Рассматриваемые вопросы: - иерархия объектов (parent-child) и система смарт-указателей в фреймворке Qt; - особенности управления памятью в императивном бэкенде; - обработка ошибок и исключительных ситуаций в C++.
7	Асинхронное программирование и многопоточность в Qt Рассматриваемые вопросы: - модель многопоточности и классы QThread, QtConcurrent; - механизмы сигналов и слотов для межпоточного взаимодействия; - обработка асинхронных событий и неблокирующие операции ввода-вывода.
8	Интеграция QML и C++: контексты и экспорт свойств Рассматриваемые вопросы: - принципы взаимодействия декларативного слоя (QML) и императивного бэкенда (C++); - механизм контекста QML и регистрация C++ классов; - экспорт свойств (Q_PROPERTY) и методов (Q_INVOKABLE) в декларативную среду.
9	Локальное хранение данных: SQLite в мобильной среде Рассматриваемые вопросы: - архитектура встраиваемых СУБД и специфика SQLite; - работа с локальными базами данных в условиях ограниченных ресурсов; - проектирование схемы данных и миграции в изолированном файловом пространстве.
10	Асинхронная работа с базами данных через Qt SQL Рассматриваемые вопросы: - механизмы объектно-реляционного отображения и модели данных в Qt; - выполнение асинхронных запросов и транзакций с использованием модуля Qt SQL; - кэширование данных и оптимизация производительности запросов.
11	Сетевое взаимодействие и потребление REST API Рассматриваемые вопросы:

№ п/п	Тематика лекционных занятий / краткое содержание
	- архитектура сетевых модулей и асинхронные HTTP-запросы в Qt; - механизмы сериализации и десериализации JSON-ответов в объекты C++; - обработка сетевых таймаутов и стратегии синхронизации данных с локальным кэшем.
12	Архитектура кроссплатформенной интеграции: Flutter и KMP Рассматриваемые вопросы: - архитектурные паттерны интеграции кроссплатформенных фреймворков в нативную среду; - механизмы изоляции Dart VM и Kotlin Native при работе под ОС «Аврора»; - стратегии переиспользования бизнес-логики из смежных дисциплин.
13	Межъязыковое взаимодействие: Platform Channels и FFI Рассматриваемые вопросы: - протоколы обмена данными между C++/QML и модулями Flutter; - использование Foreign Function Interface (FFI) для интеграции с Kotlin Multiplatform; - обработка асинхронных вызовов и сериализация данных через мосты
14	Системные сервисы ОС «Аврора»: Push-уведомления и фоновые задачи Рассматриваемые вопросы: - API системных сервисов и механизмы работы с Push-уведомлениями; - конфигурирование амбассадоров и планирование фоновых задач; - программная реализация запроса и обработки пользовательских разрешений.
15	Взаимодействие с аппаратным обеспечением и D-Bus Рассматриваемые вопросы: - принципы безопасного взаимодействия с камерой, геолокацией и сенсорами; - использование системных интерфейсов и шины сообщений D-Bus в ОС «Аврора»; - обработка аппаратных прерываний и специфика работы с периферией.
16	Дистрибуция, упаковка RPM и стандартизация документации Рассматриваемые вопросы: - архитектура пакетного менеджера RPM и спецификация файла .spec; - процесс криптографической подписи и требования к публикации в RuStore; - структура и правила оформления комплектов документов ЕСПД и ГОСТ.

4.2. Занятия семинарского типа.

Лабораторные работы

№ п/п	Наименование лабораторных работ / краткое содержание
1	Настройка среды разработки и эмулятора ОС «Аврора» Студент загружает и инсталлирует пакет Aurora SDK на рабочую станцию. Далее выполняется конфигурация виртуального устройства и запуск базового шаблонного приложения в эмуляторе. В процессе работы производится анализ логов компиляции и проверка корректности развертывания среды.
2	Проектирование базового пользовательского интерфейса на QML Обучающийся создает декларативную разметку интерфейса с использованием нативных компонентов фреймворка Silica. Производится настройка адаптивного поведения элементов верстки при изменении ориентации экрана эмулятора. Завершающим этапом является обработка базовых сенсорных событий и визуальная отладка отклика интерфейса.
3	Архитектурное разделение и структура проекта Студент формирует файловую структуру проекта курсовой работы с выделением отдельных директорий для графического и серверного слоев. Осуществляется настройка конфигурационных файлов сборщика для корректного разрешения внутренних зависимостей. В конце занятия

№ п/п	Наименование лабораторных работ / краткое содержание
	производится первичная компиляция пустого каркаса приложения для подтверждения готовности архитектуры.
4	Реализация серверной логики на C++ Обучающийся реализует классы бизнес-логики на языке C++ с применением механизмов автоматического управления памятью фреймворка Qt. Написанный код компилируется и проверяется на отсутствие критических ошибок через встроенный статический анализатор. Результатом работы является функционирующий императивный бэкенд, изолированный от графической оболочки.
5	Интеграция C++ бэкенда и QML фронтенда Студент регистрирует разработанные C++ классы в контексте QML и экспортирует их свойства для декларативного слоя. Выполняется привязка визуальных элементов интерфейса к вычислительным методам серверной логики. Производится тестирование двустороннего обмена данными между слоями в режиме реального времени.
6	Локальное хранение данных в изолированной среде (SQLite) Студент проектирует схему реляционной базы данных и инициализирует локальное хранилище SQLite внутри изолированного пространства приложения. Реализуются асинхронные функции для создания, чтения, обновления и удаления записей через модуль Qt SQL. В процессе работы проверяется корректность сохранения данных после принудительного перезапуска эмулятора.
7	Сетевое взаимодействие, потребление REST API и синхронизация данных Студент настраивает сетевой модуль приложения для асинхронного взаимодействия с внешним корпоративным REST API. В процессе работы реализуется механизм сериализации и десериализации JSON-ответов в объекты C++ с последующим кэшированием в локальную базу данных. Обучающийся программирует логику обработки сетевых таймаутов и состояний отсутствия соединения для обеспечения полноценного оффлайн-режима.
8	Асинхронные операции и многопоточность Обучающийся выносит ресурсоемкие сетевые запросы и операции с базой данных в отдельные потоки с использованием классов многопоточности Qt. Настраивается механизм асинхронной передачи результатов вычислений обратно в главный поток через сигналы и слоты. Студент проверяет отсутствие блокировок пользовательского интерфейса при выполнении фоновых задач.
9	Подготовка кроссплатформенного модуля (Flutter/KMP) Обучающийся адаптирует исходный код бизнес-логики на Flutter или Kotlin Multiplatform для целевой архитектуры ОС «Аврора». Выполняется кросс-компиляция модуля и генерация нативных библиотек для последующего встраивания. Завершается занятие проверкой работоспособности скомпилированных артефактов в изолированном тестовом стенде.
10	Настройка нативных мостов (Platform Channels / FFI) Студент настраивает протоколы межъязыкового взаимодействия для связки нативного C++ кода и кроссплатформенного модуля. Осуществляется инициализация каналов связи и определение контрактов данных для двустороннего обмена. Производится отладка процесса установления соединения между разнородными средами выполнения.
11	Передача данных и обработка асинхронных вызовов через мосты Обучающийся реализует сериализацию и десериализацию сложных структур данных при их передаче через настроенные нативные мосты. Выполняется обработка асинхронных колбэков и ошибок, возникающих на границе раздела сред. Студент проводит нагрузочное тестирование канала связи для выявления потенциальных задержек.
12	Интеграция Push-уведомлений и системных амбассадоров Студент интегрирует системные API ОС «Аврора» для отправки и приема локальных Push-уведомлений. В коде приложения реализуется программный запрос необходимых разрешений у пользователя с обработкой различных сценариев ответа. Производится верификация корректного отображения уведомлений в системном трее эмулятора.

№ п/п	Наименование лабораторных работ / краткое содержание
13	Работа с аппаратным обеспечением и D-Bus Обучающийся настраивает взаимодействие приложения с аппаратными сенсорами устройства через системную шину D-Bus. Реализуется безопасный доступ к данным геолокации или камеры с учетом строгих политик изоляции операционной системы. Результатом является успешное считывание и визуализация телеметрии в графическом интерфейсе.
14	Написание unit-тестов и UI-тестов Обучающийся покрывает критические участки серверной логики автоматизированными модульными тестами. Дополнительно создаются сценарные тесты для проверки реакций пользовательского интерфейса на симуляцию действий пользователя. Завершающим шагом является генерация отчета о проценте покрытия кода тестами.
15	Профилирование производительности и поиск утечек памяти Студент запускает встроенные инструменты профилирования для анализа потребления оперативной памяти и процессорного времени. Выявляются узкие места в производительности и потенциальные утечки ресурсов в серверном слое на C++. На основе полученных метрик вносятся оптимизации в алгоритмы и логику работы приложения.
16	Сборка, подпись и упаковка RPM-пакета Студент описывает спецификации и зависимости приложения в конфигурационном файле формата RPM. Выполняется сборка инсталляционного пакета и его криптографическая подпись с использованием предоставленных ключей разработчика. Готовый пакет устанавливается на чистый образ эмулятора для финальной верификации целостности.

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 нативного корпоративного мессенджера для ОС «Аврора» с использованием QML, многопоточной обработки сообщений и защищенного локального хранилища SQLite.

Проектирование и реализация мобильного клиента системы электронного документооборота (СЭД) с адаптивным интерфейсом на базе Silica UI и асинхронной синхронизацией с REST API.

Создание приложения для полевых инспекторов с оффлайн-режимом, локальным кэшированием данных в SQLite и программным запросом разрешений на доступ к геолокации и камере.

Разработка безопасного хранилища корпоративных учетных данных с учетом модели изоляции (sandboxing) и строгих политик разрешений ОС «Аврора».

Мобильное приложение для инвентаризации и складского учета с визуализацией данных в QML, обработкой бизнес-логики на C++ и генерацией QR-кодов.

Разработка сервиса фоновых Push-уведомлений и амбассадоров для логистического приложения курьерской доставки в экосистеме ОС «Аврора».

Приложение для мониторинга телеметрии промышленного оборудования с безопасным доступом к аппаратным сенсорам через системную шину D-Bus.

Клиентское приложение для системы контроля доступа (СКУД) с интеграцией системных сервисов, фоновыми задачами и аппаратно-ускоренной анимацией переходов.

Медицинское приложение для сбора и визуализации анамнеза с использованием локальной БД, асинхронных запросов Qt SQL и строгим соблюдением гайдлайнов UX/UI.

Интеграция модуля бизнес-логики на Flutter в нативное приложение ОС «Аврора» посредством настройки Platform Channels и сериализации данных.

Разработка банковского клиента с переиспользуемой бизнес-логикой на Kotlin Multiplatform (KMP) и связью с нативным C++ бэкендом через FFI.

Кроссплатформенное приложение-агрегатор корпоративных сервисов с архитектурой микромодулей, изоляцией Dart VM и оптимизацией межъязыкового взаимодействия.

Адаптивный корпоративный дашборд для аналитики данных с использованием продвинутых механизмов связывания данных (data binding) и оптимизацией отрисовки в QML.

Приложение для интерактивной навигации по территории кампуса с локальными векторными картами, обработкой жестов и профилированием потребления памяти.

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

№ п/п	Библиографическое описание	Место доступа
1	Программирование на языке C++ в среде Qt Creator : учебное пособие / Е. Р. Алексеев, Г. Г. Злобин, Д. А. Костюк, О. В. Чеснокова. — 2-е изд. — Москва : ИНТУИТ, 2016. — 715 с. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/100414 (дата обращения: 22.06.2026)

2	Разработка графического интерфейса пользователя информационной системы с использованием библиотеки Qt : учебное пособие / Ю. В. Минин, А. И. Елисеев, В. В. Алексеев, Ю. А. Губсков. — Тамбов : ТГТУ, 2021. — 84 с. — ISBN 978-5-8265-2397-1. — Текст : электронный	Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/320516 (дата обращения: 22.06.2026)

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

Официальный портал разработчиков ОС «Аврора» (документация, API, инструкции SDK) – https://developer.auroraos.ru/doc

Официальная документация фреймворка Qt (QML, C++, Qt Quick) – https://doc.qt.io/

Официальная документация Flutter (Dart, виджеты, интеграция) – https://docs.flutter.dev/

Официальная документация Kotlin Multiplatform – https://kotlinlang.org/docs/multiplatform/

Документация разработчика RuStore (требования к публикации, SDK) – https://www.rustore.ru/help

Электронная библиотечная система «Лань» – https://e.lanbook.com/

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

Операционные системы – отечественные дистрибутивы Linux из единого реестра РФ (Astra Linux, РЕД ОС, Альт Linux).

Офисные пакеты – российские решения для совместной работы с документами и подготовки отчетов по ГОСТ (Р7-Офис, МойОфис).

Среды разработки и SDK – Aurora SDK, Aurora IDE, специализированные редакторы для кроссплатформенных фреймворков (IntelliJ IDEA, VS Code).

Технологический стек – фреймворки для нативной и кроссплатформенной мобиль разработки (Qt 6, Flutter, KMP), встраиваемые реляционные СУБД (SQLite), серверные СУБД из реестра РФ (Postgres Pro).

Инструменты тестирования API – открытые среды для отладки сетевых запросов и консольные утилиты (Hoppscotch, Bruno, curl).

Системы контроля версий – распределенные системы управления кодом и отечественные платформы хостинга репозиториев (Git, GitFlic).

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 7 семестре.

Курсовой проект в 7 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