МИНОБРНАУКИ РОССИИ
Глазовский инженерно-экономический институт (филиал)
федерального государственного бюджетного образовательного
учреждения высшего образования
«Ижевский государственный технический университет имени М.Т.Калашникова»
(ГИЭИ (филиал) ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»)

УТВЕРЖДАЮ
И.о. директора
_____ А.В. Золотов
11 марта 20 26 г.

ДОКУМЕНТ ПОДПИСАН
ЭЛЕКТРОННОЙ ПОДПИСЬЮ

Сертификат: 3DA273AA38C240BA869FA062FFFB332C
Владелец: Золотов Алексей Васильевич
Действителен: с 29.07.2026 до 22.10.2027

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

МДК.02.02 «Осуществление и интеграция программных модулей»

09.02.11 Разработка и управление программным обеспечением

Глазов 2026

Фонд оценочных средств разработан на основе Федерального государственного образовательного стандарта по специальности среднего профессионального образования 09.02.11 "Разработка и управление программным обеспечением", утвержденного приказом Министерства просвещения Российской Федерации 24 февраля 2025 г. № 138 (зарегистрирован 31.03.2025 № 81696).

Организация разработчик: ГИЭИ (филиал) ФГБОУ ВО «ИжГТУ имени М.Т. Калашникова»

Разработчик: Горбушин Денис Шарибзянович, преподаватель СПО

Утверждено: Протокол Ученого совета филиала № 7, от 11 марта 2026 г.

Руководитель образовательной программы

_____ Т.А. Савельева

13 марта 2026 г.

Согласовано: Начальник отдела по учебно-методической работе

_____ И.Ф. Яковлева

13 марта 2026 г.

ПАСПОРТ
ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ
по дисциплине «Разработка и интеграция модулей программного обеспечения»

Результаты освоения дисциплины соотносятся с планируемыми результатами освоения образовательной программы, представленными в матрице компетенций выпускника.

В результате освоения дисциплины обучающийся должен:

Код ОК, ПК	**Уметь**	**Знать**	**Владеть навыками**
ОК.01	Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам	актуальный профессиональный и социальный контекст, в котором приходится работать и жить; основные источники информации и ресурсы для решения задач и проблем в профессиональном и/или социальном контексте; алгоритмы выполнения работ в профессиональной и смежных областях; методы работы в профессиональной и смежных сферах; структуру плана для решения задач; порядок оценки результатов решения задач профессиональной деятельности	-
ОК.02	Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности	номенклатура информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации, современные средства и устройства информатизации; порядок их применения и программное обеспечение в профессиональной деятельности в том числе с использованием цифровых средств	-
ОК.03	Планировать и	содержание актуальной	-

	реализовывать собственное профессиональное и личностное развитие, предпринимательскую деятельность в профессиональной сфере, использовать знания по правовой и финансовой грамотности в различных жизненных ситуациях	нормативно-правовой документации; современная научная и профессиональная терминология; возможные траектории профессионального развития и самообразования; основы предпринимательской деятельности; основы финансовой грамотности; правила разработки бизнес-планов; порядок выстраивания презентации; кредитные банковские продукты	
ОК.04	Эффективно взаимодействовать и работать в коллективе и команде	психологические основы деятельности коллектива, психологические особенности личности; основы проектной деятельности	-
ОК.05	Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста	особенности социального и культурного контекста; правила оформления документов и построения устных сообщений	-
ОК.06	Проявлять гражданско-патриотическую позицию, демонстрировать осознанное поведение на основе традиционных российских духовно-нравственных ценностей, в том числе с учетом гармонизации межнациональных и межрелигиозных отношений, применять стандарты антикоррупционного поведения	сущность гражданско-патриотической позиции, общечеловеческих ценностей; значимость профессиональной деятельности по специальности; стандарты антикоррупционного поведения и последствия его нарушения	-
ОК.07	Содействовать сохранению окружающей среды, ресурсосбережению, применять знания об изменении климата,	правила экологической безопасности при ведении профессиональной деятельности; основные ресурсы,	-

	принципы бережливого производства, эффективно действовать в чрезвычайных ситуациях	задействованные в профессиональной деятельности; пути обеспечения ресурсосбережения; принципы бережливого производства; основные направления изменения климатических условий региона	
ОК.08	Использовать средства физической культуры для сохранения и укрепления здоровья в процессе профессиональной деятельности и поддержания необходимого уровня физической подготовленности	роль физической культуры в общекультурном, профессиональном и социальном развитии человека; основы здорового образа жизни; условия профессиональной деятельности и зоны риска физического здоровья для специальности; средства профилактики перенапряжения	-
ОК.09	Пользоваться профессиональной документацией на государственном и иностранном языках	правила построения простых и сложных предложений на профессиональные темы; основные общеупотребительные глаголы (бытовая и профессиональная лексика); лексический минимум, относящийся к описанию предметов, средств и процессов профессиональной деятельности; особенности произношения; правила чтения текстов профессиональной направленности	-
ПК 2.1	− проектировать модули, соответствующие бизнес-задачам; − создавать архитектурные диаграммы и документацию; − определять структуру и интерфейсы модулей; − анализировать требования к модулю и	− основные принципы проектирования модулей программного обеспечения; − языки программирования и технологии для реализации модулей; − паттерны проектирования и структуры данных для создания эффективных и	− проектирования модулей ПО с учетом требований заказчика; − создания архитектурных диаграмм и спецификаций модулей; определения интерфейсов и взаимодействия модулей в системе.

	определять его функциональность; − проектировать архитектуру модуля, включая выбор подходящих паттернов проектирования и структуры данных; − создавать диаграммы классов, последовательностей и прочих диаграмм для визуализации проектируемого модуля; − выбирать подходящие языки программирования и технологии для реализации модуля; − проектировать интерфейсы программного обеспечения для взаимодействия с другими модулями и системами; − учитывать требования к масштабируемости, производительности и безопасности при проектировании модуля; проводить анализ и оптимизацию проектируемого модуля для повышения его эффективности и качества	масштабируемых модулей; − методы анализа требований и способов определения функциональности модуля; − принципы создания интерфейсов для взаимодействия с другими модулями и системами; − принципы обеспечения безопасности, производительности и масштабируемости при проектировании модулей; методы анализа и оптимизации проектируемых модулей для повышения их эффективности и качества.	
ПК 2.2	− разрабатывать модули программного обеспечения с использованием различных языков программирования и технологий; − применять паттерны проектирования и структуры данных для создания эффективных и масштабируемых модулей; − анализировать требования и определять функциональность модуля; − создавать интерфейсы для взаимодействия с	− язык программирования, основные конструкции, синтаксис; − паттерны проектирования; − структуры данных; − принципы создания интерфейсов для взаимодействия с другими модулями и системами, таких как REST API, SOAP; − работу с инструментальным программным обеспечением; − методы оптимизации кода и алгоритмов;	− создания модулей программного обеспечения на различных языках программирования; − отладки и тестирования разработанных модулей; − применения структурного и объектно-ориентированного программирования; − оптимизации кода и алгоритмов программных модулей для увеличения производительности; мониторинга и анализа

	другими модулями и системами; − обеспечивать безопасность, производительность и масштабируемость при разработке модулей; − оптимизировать проектируемые модули для повышения их эффективности и качества; − работать с системой контроля версий; − улучшать производительность модулей, выявляя и устраняя узкие места; − проводить анализ и мониторинг производительности приложений; применять инструменты для рефакторинга и оптимизации программного кода.	− эффективные алгоритмы и структуры данных для повышения производительности; − многопоточность в программных модулях; − методы оптимизации сетевых протоколов для ускорения обмена данными; − кэширование данных; − управление памятью; техники повышения производительности программного обеспечения	производительности приложений.
ПК 2.3	− интегрировать модули и компоненты, обеспечивая их взаимодействие; − работать с API и устанавливать соединения между компонентами; − отслеживать и устранять конфликты и ошибки интеграции; − анализировать и определять зависимости между модулями и компонентами; работать с различными форматами данных и протоколами передачи данных	− общие принципы функционирования аппаратных, программных и программно-аппаратных средств администрируемой информационно-коммуникационной системы; − международные стандарты локальных вычислительных сетей; − методы и подходы к интеграции модулей и компонентов; − принципы версионирования и управления изменениями при интеграции; принципы безопасности при интеграции модулей и компонентов	− интеграции программных модулей и компонентов в единое программное решение; − работы с API и веб-сервисами для взаимодействия между модулями; − работы с интеграционными платформами и инструментами; обеспечения совместимости и стабильности системы
ПК 2.4	− анализировать требования к программному обеспечению и составлять планы тестирования; − создавать тестовые	− принципы и методы тестирования программного обеспечения; − основы программирования и архитектуры	− отладки программного обеспечения на уровне программных модулей; − тестирования программного обеспечения; − формирования

	сценарии и тест-кейсы для проверки функциональности и соответствия требованиям; − выполнять тестирование программного обеспечения вручную и автоматизировать процесс тестирования; − анализировать результаты тестирования и документировать найденные ошибки; − разрабатывать стратегии отладки и исправлять ошибки в программном обеспечении; − выполнять модульные тесты с использованием инструментов тестирования, в том числе автоматизированного тестирования; − использовать системы контроля дефектов ПО; составлять отчет о выполнении тестирования ПО	программного обеспечения; − основы баз данных и SQL-запросов; − инструменты для автоматизации тестирования; − основы разработки и отладки программного обеспечения на разных языках программирования; − понятие дефекта программного обеспечения; − критерии качества ПО; − виды и типы тестирования ПО; − техники ручного тестирования; − техники автоматизированного тестирования; − жизненный цикл дефекта ПО; − принципы работы в системе контроля дефектов; основные понятия о качестве ПО	тестовых сценариев; − подготовки тестовых платформ (установка операционной системы, дополнительного ПО и другого по необходимости); − оценки объема тестирования ПО с целью определения необходимых ресурсов для его выполнения; − настройки тестовой среды и аппаратных средств для выполнения тестирования ПО в соответствии с заданием на тестирование в пределах своей компетенции; − формирования и представления отчетности о подготовке к выполнению задания на тестирование ПО в соответствии с установленными регламентами; выполнения тестовых процедур на тестовых данных
ПК 2.5	− описывать функциональность модулей в документации; − создавать диаграммы для иллюстрации работы модулей; − программировать с использованием комментариев для документирования кода; − использовать специальные метки/теги для отметки важных частей кода в документации; − вести журнал изменений и фиксировать обновления программных модулей; − разбивать модули на логические блоки и описывать каждый блок отдельн;	− стандарты технической документации; − принципы документирования программного обеспечения; инструменты для создания технической документации и комментирования кода	− создания технической документации для модулей; − документирования кода, API и интерфейсов; работы со специализированным ПО по документированию программного кода

	− включать в документацию особенности модулей, такие как ограничения, уязвимости или оптимальные настройки; проводить регулярное обновление документации при изменении модулей или добавлении нового функционала.		

Раздел 1. Зачетно – экзаменационные материалы

Зачет: вопросы с ответами

Вопрос 1. Что понимается под модулем в смысловом контексте разработки ПО? Ответ: автономная функциональная единица программы с явно заданным интерфейсом (API), которая может быть реализована, протестирована и развёрнута независимо от других модулей.

Вопрос 2. Что такое контрактное тестирование? Ответ: метод тестирования, при котором взаимодействие между клиентом и сервисом проверяется по заранее согласованному контракту (описанию интерфейса, формату данных, ожидаемым ответам), чтобы обеспечить совместимость между компонентами.

Вопрос 3. Разница между синхронной и асинхронной интеграцией? Ответ: синхронная — вызов клиента к удалённому сервису с немедленным ответом; асинхронная — обмен сообщениями через очередь/паблишер-слушатель, без немедленного ожидания ответа.

Вопрос 4. Что такое API-first дизайн? Ответ: подход, при котором контракт API определяется и согласуется до реализации модулей, чтобы обеспечить совместимость и параллельную разработку.

Вопрос 5. Что такое контрактный интерфейс и чем он полезен для интеграции? Ответ: формальное соглашение о способах взаимодействия модулей (методы, параметры, форматы данных, ошибки). Полезен для независимой разработки и тестирования, снижения конфликтов версий.

Вопрос 6. Что такое idempotent операция и зачем она нужна в интеграции? Ответ: операция — такая, что повторное выполнение даёт тот же эффект, что и одно выполнение. Нужно для устойчивости к повторным вызовам при сетевых сбоях и повторных отправках сообщений.

Вопрос 7. Что такое SLA, SLO и SLI в контексте интеграции? Ответ: SLA — договор об уровне сервиса; SLO — целевой показатель уровня сервиса; SLI — реально измеряемый показатель, по которому оценивают достижение SLO.

Вопрос 8. Что такое транзитивные зависимости и зачем они опасны для интеграций? Ответ: зависимость одного атрибута через другой атрибут (A зависит от B, B зависит от C). Могут вызывать сложности при изменении интерфейсов и усложнять эволюцию модулей.

Вопрос 9. Какие виды тестирования обычно применяются при Ost–интеграции? Ответ: модульное (unit), интеграционное (integration), контрактное (contract), end-to-end, регрессионное.

Вопрос 10. Что такое распределённая трассировка (distributed tracing) и зачем она нужна? Ответ: сбор и отображение путей запросов через несколько

сервисов (trace ID, spans) для поиска узких мест и анализа задержек.

Вопрос 11. Что такое логирование структурированное и почему оно важно в интеграции? Ответ: логирование в формате структурированных данных (ключ-значение), что облегчает фильтрацию и корреляцию по трассам, пользователям и операциям.

Вопрос 12. Что такое версионирование API и зачем оно нужно? Ответ: присвоение версии контракту API (например, v1, v2) для управления несовместимыми изменениями и плавного перехода клиентов.

Вопрос 13. Какие принципы безопасности важны при интеграции между модулями? Ответ: аутентификация и авторизация, валидация входящих данных, безопасное хранение секретов, ограничение доступа по принципу минимальных привилегий, аудит и мониторинг.

Вопрос 14. Что такое ORM и зачем он может быть полезен в реализации модуля? Ответ: Object-Relational Mapping — абстракция доступа к базе данных через объекты. Ускоряет разработку, но может скрывать реальные запросы и влиять на производительность; важно знать, какие запросы формируются.

Вопрос 15. Что такое сигнализация об ошибках и возврат статус-кодов в API? Ответ: системное оповещение об ошибках через коды статуса HTTP/протоколов и единый формат ошибок, чтобы клиенты корректно отработали ошибки и предприняли повторные действия.

Вопрос 16. Что такое оркестрация и координация между сервисами? Ответ: оркестрация управляет взаимодействиями между сервисами (например, последовательность вызовов, транзакционная логика), координация — распределение задач и обмен сообщениями без жёсткой централизации.

Вопрос 17. Какие существуют подходы к миграциям схемы в интегрированных системах? Ответ: планирование версий схем, безболезненная миграция данных, откат, тестирование миграций на стейджинге, обратная совместимость.

Вопрос 18. Что такое тест-драйвер интеграции (test harness)? Ответ: набор инструментов и скриптов, который запускает совместно тестируемые модули, обеспечивает подготовку данных и сбор результатов тестов.

Вопрос 19. Что такое задержка (latency) и пропускная способность (throughput) в контексте интеграции? Ответ: latency — время отклика; throughput — объем обработанных операций за единицу времени. Важно мониторить оба параметра.

Вопрос 20. Что такое "мягкий отказ" (graceful degradation) в интеграции? Ответ: при сбоях часть функциональности продолжается работать в

ограниченном виде, чтобы снизить влияние на пользователей.

Вопрос 21. Что означает принцип "слой архитектуры" в контексте интеграции? Ответ: разделение системы на уровни (клиентский слой, сервисный слой, доступ к данным) для снижения связности и упрощения тестирования.

Вопрос 22. Какой подход к мониторингу выбрать: push-модели или pull-модели? Ответ: обычно используют гибрид: push-метрики отправляются в систему мониторинга; pull-метрики позволяют агентам извлекать данные. Вокруг этого строят алерты и дашборды.

Вопрос 23. Что такое контрактная версия (versioning contract) и зачем она нужна? Ответ: версия контракта может существовать независимо от версии реализации; позволяет потребителю понять совместим ли сервис и когда переход на новую версию необходим.

Вопрос 24. Что означает принцип "idempotent retries" в сетевых интеграциях? Ответ: повторные попытки выполнения идентичной операции не меняют результат; удобно для устойчивости к сбоям сети.

Вопрос 25. Какие документы обычно сопровождают процесс интеграции модулей? Ответ: спецификация API (OpenAPI/Swagger), документация по контрактам, руководство по развертыванию, план миграции, тестовый план и результаты тестов, политика мониторинга и аварийного отказа.

Лабораторные работы (примерный набор)

Цель общий: развить практические навыки организации разработки, интеграции модулей, тестирования и развёртывания. Предположим стек нейтральный (любой язык/фреймворк).

Лабораторная 1. Организация проекта и управление версиями
Цель: настроить репозиторий, окружение, сборку и тесты.
Задачи:
Создать репозиторий проекта и базовую структуру модулей.
Настроить сборку/пакетирование и тестовый прогон (Unit-тесты).
Настроить CI (например, GitHub Actions) для сборки и тестов на каждую пушу.
Deliverables: структура репозитория, конфигурации CI, простые unit-тесты; инструкции по запуску.
Критерии оценки: корректность конфигурации, автоматизация тестов, читаемость документации.

Лабораторная 2. Реализация сервиса-модуля и интерфейса
Цель: реализовать простой сервис/модуль с явным интерфейсом (REST или RPC).
Задачи:

Реализовать модуль A, который предоставляет API (например, управление сущностью "задача" или "пользователь").
Определить контракт на уровне API (формат запросов/ответов, валидация).
Написать модуль-клиент, который вызывает этот сервис.
Deliverables: исходный код модулей, документация по интерфейсу, тесты на модуль.
Критерии оценки: качество дизайна интерфейса, валидация входящих данных, наличие тестов.

Лабораторная 3. Контрактное тестирование
Цель: закрепить контрактное тестирование.
Задания:
Определить OpenAPI-спецификацию для сервиса A.
Реализовать набор контрактных тестов, проверяющих соответствие клиента и сервиса контракту.
Добавить тесты на негативные сценарии (ошибки валидации, неверные форматы).
Deliverables: OpenAPI-спецификация, набор контрактных тестов, отчёт об исполнении.
Критерии оценки: полнота контрактов, устойчивость тестов к изменениям, читаемость.

Лабораторная 4. Интеграция через сообщение/асинхронная коммуникация
Цель: реализовать интеграцию через очередь/сообщения.
Задания:
Внедрить простой издатель-подписчик (publisher-subscriber) механизм (например, через встроенную симуляцию очереди или реальную очередь).
Реализовать обработчик, который принимает сообщение и выполняет действия.
Написать тесты на корректность передачи и обработку ошибок.
Deliverables: код интеграции, тесты, инструкции по развёртыванию очереди (если используется реальная система).
Критерии оценки: корректность обмена сообщениями, устойчивость к повторным передачам, обработка ошибок.

Лабораторная 5. Мониторинг, журналирование и трассировка
Цель: обеспечить наблюдаемость интеграционных сценариев.
Задачи:
Добавить структурированное логирование и базовый трейсинг (идентификаторы traceId/spans).
Поведенческие тесты на задержки и пропускную способность.
Настроить простой дашборд/отчёт по ключевым метрикам (latency, error rate).
Deliverables: обновлённый код, примеры логов и трассировок, краткий отчёт по мониторингу.
Критерии оценки: корректность логирования, полнота трассировки, полезность метрик.

Практические работы (практикум-проекты)

Практическая работа 1. Интегрированная система заказов и оплаты (микросервисы)
Цель: спроектировать и реализовать минимальную архитектуру из двух модулей: заказ и платежи.
Задачи:
Спроектировать API двух модулей и контракт взаимодействия.
Реализовать оба сервиса и клиентскую часть.
Реализовать сценарий оплаты заказа через контракт и тесты на интеграцию.
Разработать простой пайплайн развёртывания (локально через Docker Compose).
Deliverables: код двух сервисов, тесты, инструкции по развёртыванию, документ по контрактам.
Оценка: корректность взаимодействия, покрытие тестами, качество кода и документации.

Практическая работа 2. Интеграция через асинхронные очереди
Цель: реализовать интеграцию по очереди между сервисами.
Задачи:
Реализовать публикацию событий и обработчики.
Написать тесты на устойчивость к повторным сообщениям и обработку ошибок.
Опционально добавить простой мониторинг очередей.
Deliverables: код, тесты, инструкции по развёртыванию.
Оценка: корректность передачи сообщений, устойчивость к сбоям, качество тестов.

Практическая работа 3. CI/CD для модульной интеграции
Цель: построить конвейер CI/CD для сборки, тестирования и развёртывания модулей.
Задачи:
Настроить сборку, тесты и создание артефактов.
Добавить шаги развёртывания в окружение staging/production (локально или в облаке).
Внедрить статический анализ кода и сканирование зависимостей.
Deliverables: файл конвейера (например, GitHub Actions), инструкции по развёртыванию, отчёты о тестах и анализе.
Оценка: полнота пайплайна, надёжность тестирования, сопровождаемость.

Тесты (пример набора вопросов, 25–40 вопросов)

Вопрос 1. Что понимается под модульной интеграцией? Варианты: a) интеграция через общую БД, b) совместная работа модулей через чётко определённые интерфейсы, c) копирование кода между модулями, d) развёртывание в одном контейнере. Ответ: b

Вопрос 2. Что такое контрактное тестирование и чем оно отличается от модульного тестирования? Варианты: a) контрактные тесты проверяют взаимодействие между компонентами по контракту; модульные тесты — поведение отдельных единиц кода. Ответ: b

Вопрос 3. Какие уровни тестирования существуют в контексте интеграции? Варианты: a) unit, integration, contract, E2E; b) smoke, sanity, acceptance; c) performance, stress, soak. Ответ: a

Вопрос 4. Что означает принцип idempotence в операциях интеграции? Варианты: a) операция выполняется один раз, b) повторное выполнение даёт тот же эффект, c) повторение приводит к ошибке. Ответ: b

Вопрос 5. Какой подход к версионированию API предпочтителен для поддержания совместимости? Варианты: a) без версий, b) семантическое версионирование в URL или заголовках, c) версия не нужна. Ответ: b

Вопрос 6. Что такое OpenAPI и для чего он используется? Варианты: a) формализация интерфейсов HTTP API; b) язык программирования; c) база данных. Ответ: a

Вопрос 7. Что такое соглашение об уровне обслуживания (SLA) применительно к интеграции? Варианты: a) документ о расписании митингов, b) набор целевых показателей доступности и производительности, c) план восстановления после сбоя. Ответ: b

Вопрос 8. Что такое каскадное откатывание (cascade rollback) в транзакциях между модулями? Варианты: a) откат только одного модуля, b) откат всех изменений, связанных транзакцией, c) откат в очереди сообщений. Ответ: b

Вопрос 9. Что такое событие по архитектуре событийно-ориентированной интеграции? Варианты: a) запрос к сервису через REST, b) публикация события, которое подписчики обрабатывают асинхронно, c) обновление БД напрямую. Ответ: b

Вопрос 10. Что означает термин "идентификатор трассировки" (traceId) в распределённой трассировке? Ответ: уникальный идентификатор, связывающий все узлы запроса в рамках одного пользовательского сценария.

Вопрос 11. Какие практики безопасности важны при интеграции модулей? Варианты: a) шифрование данных на канале и в хранилище, b) ограничение привилегий, c) аудит и мониторинг, d) все перечисленное. Ответ: d

Вопрос 12. Что такое контрактное тестирование потребителя (consumer-driven contract testing)? Варианты: a) тестирование поставщика по контракту потребителя, b) тестирование потребителя по контракту поставщика, c) тестирование только интерфейсов без контрактов. Ответ: b

Вопрос 13. Какой паттерн можно использовать для устойчивости к сбоем внешних сервисов? Варианты: a) circuit breaker, b) observer, c) singleton. Ответ: a

Вопрос 14. Что такое мониторинг зависимостей и почему он важен? Варианты: a) мониторинг только приложений, b) отслеживание состояния зависимостей и внешних сервисов, c) только логирование. Ответ: b

Вопрос 15. Что такое "канонический формат" обмена данными (например, JSON, Protobuf) и почему он важен для интеграции? Ответ: согласованный формат сериализации данных, обеспечивающий совместимость между модулями и языками программирования.

Вопрос 16. Что такое миграции схемы и какие подводные камни они часто несут в интеграционных проектах? Ответ: изменение структуры данных без потери данных; риск несовместимости, потребности в ревизии клиентских контрактов и потенциальная простоя.

Вопрос 17. Чем отличается синхронная аутентификация между сервисами от асинхронной? Ответ: синхронная — происходит как часть запроса, асинхронная — через отдельный канал/сообщение.

Вопрос 18. Что такое "idempotent retries" и как их реализовать в REST/API? Ответ: повторные запросы не должны менять результат; реализуется через idempotent методы (GET, PUT) и/или уникальные idempotency keys.

Вопрос 19. Какие документы обычно сопровождают процесс интеграции модулей? Ответ: API контракт, документация по интерфейсам, тест-планы, сценарии интеграции, инструкции по развёртыванию, политика мониторинга.

Вопрос 20. Что такое "практики DevOps" в контексте осуществления и интеграции модулей? Ответ: автоматизация сборки, тестирования, развёртывания, мониторинга, управление конфигурациями и инфраструктурой как код.

Вопрос 21. Что такое "распределённая архитектура"? Ответ: архитектура, в которой функциональность разделена на независимые сервисы, взаимодействующие через сети и сообщения.

Вопрос 22. Что такое "semantic versioning" и когда он уместен? Ответ: подход к версионированию базовой функциональности, где MAJOR/MINOR/PATCH означают несовместимые изменения, новые возможности и исправления ошибок соответственно.

Вопрос 23. Что означает "права доступа и аутентификация между сервисами"? Ответ: сервисы должны аутентифицироваться и авторизоваться для доступа к чужим ресурсам; часто применяются JWT, mTLS, OAuth2.

Вопрос 24. Что такое "replay protection" в системах обмена сообщениями? Ответ: предотвращение повторной обработки идентичной передачи данных, чтобы избежать дублирования.

Вопрос 25. Что такое "test doubles" и зачем они нужны в тестировании модулей? Ответ: заглушки/моки/фейки, которые заменяют реальные зависимости для изоляции тестируемого модуля.