Приложение
к ОПОП СПО 09.02.11 Разработка и управление программным обеспечением

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ ДАГЕСТАН

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ РЕСПУБЛИКИ ДАГЕСТАН «ТЕХНИЧЕСКИЙ КОЛЛЕДЖ ИМЕНИ Р.Н. АШУРАЛИЕВА»

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

междисциплинарного курса МДК 02.02 Осуществление интеграции программных модулей

Специальность 09.02.11 Разработка и управление программным обеспечением

Квалификация программист

Махачкала, 2026г.

ОДОБРЕНО

предметной (цикловой) комиссией УГС 09.00.00. Информатика и вычислительная техника и 10.00.00 Информационная безопасность

Председатель П(Ц)К

 Ш.М. Мусаева

Протокол № 8 от 30 апреля 2026 г.

Фонд оценочных средств междисциплинарного курса МДК.02.02 Осуществление интеграции программных модулей разработан на основе требований:

- Приказа Минпросвещения России от 24.02.2025 N 138 "Об утверждении федерального государственного образовательного стандарта среднего профессионального образования по специальности 09.02.11 Разработка и управление программным обеспечением" (Зарегистрировано в Минюсте России 31.03.2025 N 81696);
- Рабочей программы междисциплинарного курса МДК.02.02 Осуществление интеграции программных модулей.

с учетом:

- Примерной образовательной программы «Профессионалитет» по специальности 09.02.11 Разработка и управление программным обеспечением, утвержденная протоколом ФУМО в системе СПО по УГПС 09.00.00 Информатика и вычислительная техника от 29.08.2025 № 6/2025, зарегистрирована в государственном реестре примерных образовательных программ – регистрационный номер 18/2025. Реквизиты утверждающего документа –приказ ФГБОУ ДПО ИРПО № 01-09-581/2025 от 13.10.2025;

Разработчик:

- Азизова Лилия Насруллаховна, к.п.н., преподаватель ГБПОУ РД «Технический колледж имени Р.Н. Ашуралиева»

© Азизова Лилия Насруллаховна, 2026

© ГБПОУ РД «Технический колледж имени Р.Н. Ашуралиева», 2026

СОДЕРЖАНИЕ

1. Общие положения....4
2. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ МЕЖДИСЦИПЛИНАРНОГО КУРСА....4
3. Формы и методы контроля и оценивания....9
4. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ МЕЖДИСЦИПЛИНАРНОГО КУРСА....11
5. ОЦЕНОЧНЫЕ СРЕДСТВА текущего контроля УСПЕВАЕМОСТИ....14
5.1. Типовые задания для оценки знаний и умений....14
5.2. Критерии оценивания....36
5.2.1. Критерии оценивания устного ответа....36
5.2.2. Критерии оценивания выполнения заданий на лабораторных и практических занятиях....36
5.2.3. Критерии оценивания тестовых заданий....36
5.2.4. Общая классификация ошибок....37
6. ОЦЕНОЧНЫЕ СРЕДСТВА ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ....37
6.1. Оценочные средства курсового проекта по междисциплинарному курсу МДК 02.01. Технология разработки программного обеспечения....37
6.1.1.Примерная тематика курсовых проектов:....37
6.1.2.Критерии оценивания курсового проекта....38
7. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ....40

1. ОБЩИЕ ПОЛОЖЕНИЯ

Фонд оценочных средств (далее – ФОС) междисциплинарного курса МДК. 02.02 Осуществление интеграции программных модулей является частью программы подготовки специалистов среднего звена специальности 09.02.11 Разработка и управление программным обеспечением.

ФОС позволяет оценить достижение, запланированных по междисциплинарному курсу, результатов обучения.

ФОС включают оценочные средства для проведения текущего контроля успеваемости и промежуточной аттестации по междисциплинарному курсу.

Текущий контроль успеваемости осуществляется с целью регулярного наблюдения за ходом поэтапного освоения обучающимися междисциплинарного курса, оптимизации управления образовательной деятельностью обучающихся, своевременной корректировки персональных образовательных результатов обучающихся педагогическими средствами.

Текущему контролю успеваемости подлежат все обучающиеся, осваивающие междисциплинарный курс.

Текущий контроль проводится в пределах учебного времени, отведенного на изучение междисциплинарного курса традиционными и инновационными методами с использованием современных технологий.

Результаты текущего контроля успеваемости обучающихся в виде оценки в балльном выражении («5», «4», «3», «2») записываются в журнале учебных занятий.

Текущий контроль освоения обучающимися программного материала междисциплинарного курса может иметь следующие виды: оперативный и рубежный контроль.

Оперативный контроль проводится с целью объективной оценки качества освоения программы междисциплинарного курса, а также стимулирования учебной работы обучающихся, мониторинга результатов образовательной деятельности, подготовки к промежуточной аттестации и обеспечения максимальной эффективности учебно-воспитательного процесса.

Рубежный контроль является контрольной точкой и проводится с целью комплексной оценки уровня освоения программного материала.

Промежуточная аттестация по междисциплинарному курсу проводится с целью оценки уровня освоения теоретических знаний, умений, приобретенного практического опыта.

Формы и периодичность промежуточной аттестации по междисциплинарному курсу определяются учебным планом образовательной программы: дифференцированные зачеты в 6 и 7 семестрах.

2. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОСВОЕНИЯ МЕЖДИСЦИПЛИНАРНОГО КУРСА

ПМ.02 Разработка и интеграция модулей программного обеспечения направлен на формирование общих и профессиональных компетенций.

Освоение междисциплинарного курса должно способствовать формированию общих компетенций:

– ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам.

– ОК 02. Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности.

– ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке Российской Федерации с учетом особенностей социального и культурного контекста.

– ОК 09. Пользоваться профессиональной документацией на государственном и

иностранном языках.

Освоение междисциплинарного курса должно способствовать формированию профессиональных компетенций:

– ПК 2.1. Проектировать модули программного обеспечения.

– ПК 2.3. Выполнять интеграцию модулей и компонентов программного обеспечения.

– ПК 2.5. Осуществлять документирование программных модулей программного обеспечения.

В результате освоения междисциплинарного курса обучающийся должен овладеть навыками:

– Проектировать модули, соответствующие бизнес-задачам.

– Создавать архитектурные диаграммы и документацию.

– Определять структуру и интерфейсы модулей.

– Анализировать требования к модулю и 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.

– 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.

– 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 отдельно.

– Включать в документацию особенности модулей, такие как ограничения, уязвимости или оптимальные настройки.

– Проводить регулярное обновление документации при изменении модулей или добавлении нового функционала.

В результате освоения междисциплинарного курса обучающийся должен **уметь:**

Код умения	**Название умения**
У1	Проектировать модули, соответствующие бизнес-задачам.
У2	Создавать архитектурные диаграммы и документацию.

У3	Определять структуру и интерфейсы модулей.
У4	Анализировать требования к модулю и определять его функциональность.
У5	Проектировать архитектуру модуля, включая выбор подходящих паттернов проектирования и структуры данных.
У6	Создавать диаграммы классов, последовательностей и прочих диаграмм для визуализации проектируемого модуля.
У7	Выбирать подходящие языки программирования и технологии для реализации модуля.
У8	Проектировать интерфейсы программного обеспечения для взаимодействия с другими модулями и системами.
У9	Учитывать требования к масштабируемости, производительности и безопасности при проектировании модуля.
У10	Проводить анализ и оптимизацию проектируемого модуля для повышения его эффективности и качества.
У11	Интегрировать модули и компоненты, обеспечивая их взаимодействие.
У12	Работать с API и устанавливать соединения между компонентами.
У13	Отслеживать и устранять конфликты и ошибки интеграции.
У14	Анализировать и определять зависимости между модулями и компонентами.
У15	Работать с различными форматами данных и протоколами передачи данных.
У16	Описывать функциональность модулей в документации.
У17	Создавать диаграммы для иллюстрации работы модулей.
У18	Программировать с использованием комментариев для документирования кода.
У19	Использовать специальные метки/теги для отметки важных частей кода в документации.
У20	Вести журнал изменений и фиксировать обновления программных модулей.
У21	Разбивать модули на логические блоки и описывать каждый блок отдельно.
У22	Включать в документацию особенности модулей, такие как ограничения, уязвимости или оптимальные настройки.
У23	Проводить регулярное обновление документации при изменении модулей или добавлении нового функционала

В результате освоения междисциплинарного курса обучающийся должен **знать:**

Код знания	**Название знания**
З1	Основные принципы проектирования модулей программного обеспечения.
З2	Языки программирования и технологии для реализации модулей.
З3	Паттерны проектирования и структуры данных для создания эффективных и масштабируемых модулей.
З4	Методы анализа требований и способов определения функциональности модуля.
З5	Принципы создания интерфейсов для взаимодействия с другими модулями и системами.
З6	Принципы обеспечения безопасности, производительности и масштабируемости при проектировании модулей.
З7	Методы анализа и оптимизации проектируемых модулей для повышения их эффективности и качества.
З8	Общих принципов функционирования аппаратных, программных и программно-аппаратных средств администрируемой информационно-коммуникационной системы.
З9	Международных стандартов локальных вычислительных сетей.
З10	Методы и подходы к интеграции модулей и компонентов.
З11	Принципы версионирования и управления изменениями при интеграции.
З12	Принципы безопасности при интеграции модулей и компонентов.
З13	Стандарты технической документации.
З14	Принципы документирования программного обеспечения.
З15	Инструменты для создания технической документации и комментирования кода

Общие компетенции:

Код компетенции	Формулировка компетенции	Знания, умения
ОК 01	Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам	**Умения:** распознавать задачу и/или проблему в профессиональном и/или социальном контексте, анализировать и выделять ее составные части; определять этапы решения задачи, составлять план действия, реализовывать составленный план, определять необходимые ресурсы; выявлять и эффективно искать информацию, необходимую для решения задачи и/или проблемы; владеть актуальными методами работы в профессиональной и смежных сферах; оценивать результат и последствия своих действий (самостоятельно или с помощью наставника)
		Знания: актуальный профессиональный и социальный контекст, в котором приходится работать и жить; структура плана для решения задач, алгоритмы выполнения работ в профессиональной и смежных областях; основные источники информации и ресурсы для решения задач и/или проблем в профессиональном и/или социальном контексте; методы работы в профессиональной и смежных сферах; 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	**Умения:** определять задачи для поиска информации, планировать процесс поиска, выбирать необходимые источники информации; выделять наиболее значимое в перечне информации, структурировать получаемую информацию, оформлять результаты поиска; оценивать практическую значимость результатов поиска; применять средства информационных технологий для решения профессиональных задач; использовать современное программное обеспечение в профессиональной деятельности; использовать различные цифровые средства для решения профессиональных задач
		Знания: номенклатура информационных источников, применяемых в профессиональной деятельности; приемы структурирования информации; формат оформления результатов поиска информации; современные средства и устройства информатизации, порядок их применения; программное обеспечение в профессиональной деятельности, в том числе цифровые средства; психологические основы деятельности коллектива
ОК 05	Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста.	**Умения:** грамотно излагать свои мысли и оформлять документы по профессиональной тематике на государственном языке проявлять толерантность в рабочем коллективе
		Знания: правила построения устных сообщений; особенности социального и культурного контекста
ОК 09	Пользоваться профессиональной документацией на государственном и иностранном языках	**Умения:** понимать общий смысл четко произнесенных высказываний на известные темы (профессиональные и бытовые), понимать тексты на базовые профессиональные темы участвовать в диалогах на знакомые общие и профессиональные темы; строить простые высказывания о себе и о своей профессиональной деятельности; кратко обосновывать и объяснять свои действия (текущие и планируемые) писать простые связные сообщения на знакомые или интересующие профессиональные темы

		Знания: 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

Профессиональные компетенции:

Код и формулировка компетенции	**Показатели освоения компетенции**
ПК 2.1. Проектировать модули программного обеспечения	**Навыки:** Проектирования модулей ПО с учетом требований заказчика. Создания архитектурных диаграмм и спецификаций модулей. Определения интерфейсов и взаимодействия модулей в системе.
	Умения: Проектировать модули, соответствующие бизнес-задачам. Создавать архитектурные диаграммы и документацию. Определять структуру и интерфейсы модулей. Анализировать требования к модулю и 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.
	Знания: Основные принципы проектирования модулей программного обеспечения. Языки программирования и технологии для реализации модулей. Паттерны проектирования и структуры данных для создания эффективных и 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 2.3. Выполнять интеграцию модулей и компонентов программного обеспечения	**Навыки:** 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.
	Умения: 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.
	Знания: 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.
ПК 2.5. Осуществлять документирование программных модулей программного обеспечения.	**Навыки:** Создания технической документации для модулей. Документирования кода, API и интерфейсов. Работы со специализированным ПО по документированию программного кода.
	Умения: 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 отдельно. Включать в документацию особенности модулей, такие как ограничения, уязвимости или оптимальные настройки. Проводить регулярное обновление документации при изменении модулей или добавлении нового функционала.
	Знания: Стандарты технической документации. Принципы документирования программного обеспечения. Инструменты для создания технической документации и комментирования кода.

3. ФОРМЫ И МЕТОДЫ КОНТРОЛЯ И ОЦЕНИВАНИЯ

Формы текущего контроля по междисциплинарному курсу:

- устный опрос (фронтальный, индивидуальный, комбинированный);
- тестирование (письменное или компьютерное);
- письменная проверка (ответы на вопросы, решение задач и примеров, составление тезисов, рефератов, выполнение схем, выполнение заданий для самостоятельной работы и др.);
- практическая проверка (при проведении практических и лабораторных занятий, выполнении и защите курсовых проектов (работ));
- самоконтроль и взаимопроверка.

Возможны и другие формы текущего контроля успеваемости, в том числе инновационные на основе информационно-коммуникационных технологий.

Преподаватель на одном учебном занятии может использовать одну или несколько форм текущего контроля.

Код и наименование профессиональных и общих компетенций, формируемых в рамках модуля	**Критерии оценки результата (показатели освоенности компетенций)**	**Формы контроля и методы оценки**
ПК 2.1. Проектировать модули программного обеспечения	Анализ проектной и технической документации. Проектирования модулей ПО с учетом требований заказчика. создания архитектурных диаграмм и спецификаций модулей. определения интерфейсов и взаимодействия модулей в системе.	Экспертная оценка деятельности в ходе выполнения практических работ, практической подготовки, интерпретация результатов собеседования и наблюдения, решение производственных задач. Текущий контроль: - экспертная оценка выполнения практических заданий по МДК;
ПК 2.3. Выполнять интеграцию модулей и компонентов	Анализировать и определять зависимости между модулями и компонентами работать с различными форматами данных и протоколами передачи данных и оптимизации программного кода	

программного обеспечения		- защита отчетов по практическим работам; - оценка заданий для самостоятельной работы. Промежуточная аттестация: - экспертная оценка выполнения практических заданий по МДК Защита курсового проекта
ПК 2.5. Осуществлять документирование программных модулей программного обеспечения	Использование выбранной системы контроля версий. Использование методов для получения кода с заданной функциональностью и степенью качества. Выявление ошибок в системных компонентах на основе спецификаций.	
ОК 01. Выбирать способы решения задач профессиональной деятельности, применительно к различным контекстам	Обоснованность планирования учебной и профессиональной деятельности; соответствие результата выполнения профессиональных задач эталону (стандартам, образцам, алгоритму, условиям, требованиям или ожидаемому результату); степень точности выполнения поставленных задач.	
ОК 02. Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности	полнота охвата информационных источников; скорость нахождения и достоверность информации; обновляемость и пополняемость знаний, влияющих на результаты учебной и производственной деятельности.	
ОК 05. Осуществлять устную и письменную коммуникацию на государственном языке с учетом особенностей социального и культурного контекста	демонстрация навыков грамотно общения и оформление документации на государственном языке Российской Федерации, принимая во внимание особенности социального и культурного контекста	
ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках	демонстрация умений понимать тексты на базовые и профессиональные темы; - составлять необходимую документацию на государственном и иностранном языках	

4. ПАСПОРТ ФОНДА ОЦЕНОЧНЫХ СРЕДСТВ МЕЖДИСЦИПЛИНАРНОГО КУРСА

МДК 02.02 Осуществление интеграции программных модулей

№	Наименование темы	Результаты обучения (освоенныеумения и знаний)	ПК, ОК	Текущий контроль успеваемости	Промежуточная аттестация
	МДК 02.02 Осуществление интеграции программных модулей				
	Раздел 1. Инструментальные средства разработки программного обеспечения				
	Тема 2.1. Современные технологии и инструменты интеграции				
	Понятие репозитория проекта, структура проекта	З11, З13-З15	ОК 01, 05, 09 ПК 2.1, 2.3, 2.5	Устный опрос, тестирование	Диф-ый зачет
	Виды, цели и уровни интеграции программных модулей. Автоматизация бизнес-процессов	З1, З4, З10		Устный опрос, тестирование	
	Выбор источников и приемников данных, сопоставление объектов данных	З3, З4, З10, З12		Устный опрос, тестирование	
	Транспортные протоколы. Стандарты форматирования сообщений	З8, З9		Устный опрос, тестирование	
	Организация работы команды в системе контроля версий	З11, З13		Устный опрос, тестирование	
	Разработка модульной структуры проекта (диаграммы модулей)	У2, У6, У17, У21		Отчет по лабораторному занятию	
	Разработка перечня артефактов и протоколов проекта	У15, У16, У23		Отчет по лабораторному занятию	
	Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и др. параметров импорта в репозиторий)	У19, У20		Отчет по лабораторному занятию	
	Разработка и интеграция модулей проекта (командная работа)	У11, У12-У14		Отчет по лабораторному занятию	
	Отладка отдельных модулей программного проекта	У10		Отчет по лабораторному занятию	
	Организация обработки исключений	У13		Отчет по лабораторному занятию	
	Тема 2.2. Инструментарий тестирования и анализа качества программных средств				
	Отладка программных продуктов. Инструменты отладки. Отладочные классы	З3, З7, З8	ОК 01, 05, 09 ПК 2.1, 2.3, 2.5	Устный опрос, тестирование	
	Ручное и автоматизированное тестирование. Методы и средства организации тестирования	З5, З7		Устный опрос, тестирование	
	Обработка исключительных ситуаций. Методы и способы идентификации сбоев и ошибок	З7, З8, З12		Устный опрос, тестирование	
	Применение отладочных классов в проекте	У7		Отчет по лабораторному занятию	
	Отладка проекта	У10		Отчет по лабораторному занятию	
	Инспекция кода модулей проекта	У10		Отчет по лабораторному занятию	
	Тестирование интерфейса пользователя средствами инструментальной среды разработки	У8		Отчет по лабораторному занятию	
	Разработка тестовых модулей проекта для тестирования отдельных модулей	У5		Отчет по лабораторному занятию	
	Выполнение функционального тестирования	У10		Отчет по лабораторному занятию	
	Тестирование интеграции	У13		Отчет по лабораторному занятию	
	Документирование результатов тестирования	У16, У22		Отчет по лабораторному занятию	
	Тема 2.3. Основы интеграции программных модулей				

	Разработка REST API. Клиент-серверное взаимодействие. Особенности передачи информации по HTTP протоколу. Структура HTTP запроса. HTTP методы: GET, POST, DELETE, PUT, PATCH. HTTP заголовки. Тело запроса.	32, 33, 35, 311	ОК 01, 05, 09 ПК 2.1, 2.3, 2.5	Устный опрос, тестирование	
	Маршрутизация запросов. Группировка маршрутов. Статические ресурсы.	32, 33, 35		Устный опрос, тестирование	
	Обработка запросов пользователя. Path, Query параметры. Обработка содержимого body: raw, objects, forms, multipart. Валидация данных.	33, 34, 35		Устный опрос, тестирование	
	Формирование и отправка ответов: object, file. Параметры ответов: статус код, тип содержимого, заголовки, cookies. Перенаправления. Сериализация/десериализация объектов.	33, 34, 35		Устный опрос, тестирование	
	Создание и управление фоновыми задачами.	33, 34, 38		Устный опрос, тестирование	
	Аутентификация и авторизация. oAuth, JWT, forms. Сессии. Ролевое разграничение доступа к ресурсам.	35, 36, 312		Устный опрос, тестирование	
	Разработка WebSocket API. Взаимодействие клиента и сервера по WebSocket протоколу. Настройки соединения. Открытие и закрытие соединения. Передача сообщения серверу.	32, 33, 35, 38		Устный опрос, тестирование	
	Разработка микросервисов. Микросервисная и монолитная архитектура.	31, 32, 33, 36		Устный опрос, тестирование	
	Синхронное (REST, gRPC) и асинхронное (брокеры сообщений) взаимодействие между микросервисами.	33, 36, 39, 310		Устный опрос, тестирование	
	Создание клиентского приложения для работы с публичным API	У12		Отчет по лабораторному занятию	
	Создание REST API приложения с реализацией: добавления, удаления, изменения и создания данных (от 3 - 4 сущностей)	У1, У5		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: работа с удаленным источником данных	У15		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: работа со статическими изображениями (ресурсами) - загрузка, передача, удаление.	У15		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: обработка path и query параметров	У3		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: обработка ошибок, передача сообщений об ошибке пользователю	У13		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: валидация полученных данных	У4		Отчет по лабораторному занятию	
	Расширение функционала REST API приложения: добавление фоновых задач	У5		Отчет по лабораторному занятию	
	Тема 2.4. Управление и мониторинг интегрированной системы				
	Настройка конфигурации и сборки приложения	37, 38, 314	ОК 01, 05, 09 ПК 2.1, 2.3, 2.5	Устный опрос, тестирование	
	Логирование событий. Конфигурация логирования. Уровни логирования	37, 38, 314		Устный опрос, тестирование	
	Логирование в файлы различного формат	38, 314, 315		Устный опрос, тестирование	
	Мониторинг приложения: нагрузка, ошибки, сбор статистики. Внедрение сборщика метрик	36, 37, 38		Устный опрос, тестирование	

	Инструменты контейнеризации. Контейнеризация приложения. Средства доставки и средства развертывания решения	З2, З6, З8		Устный опрос, тестирование	
	Настроить конфигурацию rest api приложения (порт, хост, данные для подключения к источнику данных, приватные ключи).	У9		Отчет по лабораторному занятию	
	Внедрить логирование в rest api приложение	У10		Отчет по лабораторному занятию	
	Упаковка rest api приложения в контейнер и доставка на другое устройство	У7		Отчет по лабораторному занятию	
	Самостоятельная работа обучающихся:				
	Изучить теоретический материал и составить тезисы (краткий конспект) по следующим темам:				
	Протоколы с использованием безопасного соединения: HTTPS, WSS	З5, З8, З9	ОК 01-09 ПК 2.1, 2.3, 2.5	Конспект, устный опрос	
	Предотвращение угроз безопасности: SQL инъекции, CSRF, XSS. Хеширование сенситивных данных, применение алгоритмов хеширования паролей с солью.	З6, З12, З4			
	Анализ уязвимостей. Регулярные аудиты безопасности. Применение лучших практик защиты информации.	З7, З10, З13, З14			
	Масштабирование интегрированных решений. Горизонтальное и вертикальное масштабирование.	З1, З3, З6			
	Оптимизации производительности. Кэширование данных. Оптимизация запросов к базам данных.	З2, З7, З3, З11, З15			
	Профилирование кода. Уменьшение времени отклика	З7, З8, З10			
	Курсовое проектирование		ОК 01-09 ПК 2.1, 2.3, 2.5		
	Анализ требований и формирование технического задания (ТЗ) на модуль	З1-З15, У1-У23		Отчет	
	Проектирование архитектуры модуля и выбор стека технологий			Отчет	
	Проектирование структуры базы данных модуля			Отчет	
	Реализация бизнес-логики и RESTful API модуля			Отчет	
	Разработка пользовательского интерфейса (UI/UX) модуля			Отчет	
	Реализация механизмов безопасности и аутентификации			Отчет	
	Интеграционное взаимодействие: обмен данными с внешними системами			Отчет	
	Тестирование модуля и отладка интеграции			Отчет	
	Документирование кода, API и подготовка инструкций			Отчет	
	Развертывание модуля на тестовом стенде, интеграционные испытания и ввод в опытную эксплуатацию			Отчет	

5. ОЦЕНОЧНЫЕ СРЕДСТВА ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ

МДК 02.02 Осуществление интеграции программных модулей

5.1.Типовые задания для оценки знаний и умений

Раздел 2. Инструментальные средства разработки программных модулей

Тема 2.1. Современные технологии и инструменты интеграции

1. Задание для устного опроса по темам

1. Структура хранения**.** Почему важно разделять код, конфигурации и документацию?
2. В чем принципиальное различие между монорепозиторием и полирепозиторием? Приведите примеры бизнес-кейсов, где предпочтителен каждый из подходов.
3. Как в структуре проекта обычно организуется хранение конфигураций для разных сред (dev, test, prod)? Какие существуют подходы к управлению секретами (ключами, паролями) внутри репозитория и за его пределами?
4. Перечислите основные уровни интеграции программных модулей (уровень данных, уровень приложений (API), уровень процессов, уровень пользовательского интерфейса). Для каждого уровня приведите примеры используемых технологий.
5. Сравните синхронный и асинхронный типы взаимодействия модулей. В каких сценариях бизнес-логики применение асинхронной интеграции критически важно, а в каких – избыточно?
6. Сформулируйте критерии выбора бизнес-процесса для автоматизации. Какие метрики (ROI, время выполнения, частота ошибок) используются для оценки эффективности внедрения интеграционного решения?
7. Какие факторы влияют на выбор между реляционной (SQL) и нереляционной (NoSQL) базой данных в качестве источника или приемника данных в корпоративной системе?
8. Что такое процесс трансформации и сопоставления данных (ETL/ELT)? Опишите алгоритм действий при несовпадении структур данных (схем) в источнике и приемнике (например, разные типы полей или разные уровни нормализации).
9. Как вы будете обрабатывать невалидные данные (дубли, NULL-значения, невалидные форматы) на этапе выбора источника, чтобы интеграция прошла успешно?
10. Сравните HTTP/HTTPS, AMQP (например, RabbitMQ) и Kafka Protocol с точки зрения надежности доставки (гарантии Exactly-once/At-least-once) и сценариев использования.
11. Перечислите преимущества и недостатки JSON, XML и Protocol Buffers (Protobuf) при построении интеграционных шин. Как выбор формата влияет на производительность системы и человекочитаемость логов?
12. Какие стратегии управления версиями форматов сообщений (версионирование) существуют в распределенных системах для обеспечения обратной совместимости при обновлении модулей?
13. Опишите модель ветвления Git Flow (или GitHub Flow). Какие типы веток существуют и какова их роль в жизненном цикле разработки?
14. Опишите алгоритм действий разработчика при возникновении конфликта слияния (merge conflict) в Git. Какие подходы к коммуникации в команде помогают минимизировать количество таких конфликтов?
15. Как в команде организован процесс контроля качества перед слиянием Pull Request (PR)? Опишите роль CI/CD (непрерывной интеграции), код-ревью и политик защиты веток (branch protection rules) в этом процессе.

Тестовые вопросы

Тема2.1. Современные технологии и инструменты интеграции

Вариант 1.

1. Что из перечисленного является характерным признаком монорепозитория (monorepo)?

А) Каждый проект хранится в отдельном репозитории со своей историей

Б) Все проекты и модули компании хранятся в одном репозитории

В) Репозиторий содержит только один исполняемый файл

Г) В репозитории запрещено хранить конфигурации

2. Какой файл в корне репозитория обычно содержит информацию о зависимостях для Python-проекта?

А) package.json

Б) pom.xml

В) requirements.txt

Г) go.mod

3. Для хранения конфигураций, различающихся для сред разработки, тестирования и продакшена, чаще всего используют:

А) одну общую папку config с файлами dev.env, test.env, prod.env

Б) отдельные репозитории для каждой среды

В) жесткое кодирование параметров в исходном коде

Г) хранение всех конфигураций в базе данных

4. Какой уровень интеграции предполагает обмен данными между базами данных разных систем без участия бизнес-логики?

А) Уровень пользовательского интерфейса

Б) Уровень процессов

В) Уровень приложений (API)

Г) Уровень данных

5. В каком сценарии применение асинхронной интеграции является наиболее оправданным?

А) Операция списания денег с банковской карты в реальном времени

Б) Отправка массовых email-уведомлений по расписанию

В) Авторизация пользователя через внешний SSO-провайдер

Г) Запрос текущего курса валют для отображения на сайте

6. Какой показатель НЕ является прямым критерием для выбора бизнес-процесса для автоматизации?

А) Высокая частота выполнения операции

Б) Значительное количество ручных ошибок при выполнении

В) Сложность алгоритма, реализованного в процессе

Г) Объем занимаемого дискового пространства логами

7. При выборе между SQL и NoSQL базой данных в качестве источника для интеграции решающим фактором в пользу NoSQL чаще всего является:

А) строгая схема данных и транзакционная целостность

Б) необходимость хранить большие объемы неструктурированных или полуструктурированных данных

В) поддержка сложных JOIN-запросов

Г) наличие стандартного языка запросов (SQL)

8. Что понимается под сопоставлением (mapping) объектов данных в процессе интеграции?
А) Установление соответствия между полями источника и приемника
Б) Шифрование данных при передаче
В) Сжатие данных для уменьшения трафика
Г) Выбор транспортного протокола

9. Какая стратегия очистки данных применяется для замены пропущенных (NULL) значений в числовых полях?
А) Удаление всей записи с NULL
Б) Замена на среднее арифметическое или медиану по столбцу
В) Замена на константу «0» без анализа
Г) Пропуск таких записей без обработки

10. Какой протокол обеспечивает гарантию доставки «ровно один раз» (exactly-once) в типовой конфигурации?
А) HTTP без повторных попыток
Б) AMQP с подтверждением (ack) и механизмом повторов
В) Kafka при правильной настройке идемпотентного продюсера
Г) UDP с контрольной суммой

11. Какой формат сериализации обеспечивает наилучшую производительность (скорость и размер сообщения) среди перечисленных?
А) JSON
Б) XML
В) YAML
Г) Protocol Buffers (Protobuf)

12. Какая стратегия версионирования API позволяет клиентам постепенно переходить на новую версию без нарушения работы?
А) Одновременное отключение старой версии
Б) Использование заголовка Accept-Version или параметра ?version=
В) Хранение всех версий в одном эндпоинте без маркировки
Г) Обновление всех клиентов принудительно в одну ночь

13. В модели ветвления Git Flow ветка release **предназначена для:**
А) разработки новых функций
Б) исправления критических багов в продакшене
В) подготовки к релизу: финальных правок, тестирования и обновления версий
Г) постоянной интеграции изменений от всех разработчиков

14. Какой из перечисленных шагов является корректным при возникновении конфликта слияния (merge conflict)?
А) Отменить слияние и попросить коллегу вручную исправить его код
Б) Вручную разрешить конфликт в файлах, затем выполнить git add и git commit
В) Перезаписать свою ветку чужой командой git push --force
Г) Игнорировать конфликт и продолжить слияние с флагом --no-verify

15. Какой инструмент или практика в Git позволяет автоматически запускать сборку и тесты перед принятием Pull Request?
А) Git hooks (pre-commit)
Б) Branch protection rules с настройкой CI-проверок
В) Команда git stash
Г) Использование тегов (tags)

Вариант 2

1. Что из перечисленного является преимуществом полирепозиторной стратегии перед монорепозиторием?

А) Упрощение управления зависимостями между проектами
Б) Лучшая изоляция и независимость команд, работающих над разными проектами
В) Единая история изменений для всех модулей
Г) Легкость атомарных коммитов сразу в несколько проектов

2. Какой файл в корне репозитория описывает цель проекта, инструкцию по установке и примеры использования?

А) CHANGELOG.md
Б) README.md
В) .gitignore
Г) LICENSE

3. Как в структуре проекта обычно организуют хранение секретов (паролей, ключей API)?

А) Помещают их в отдельный файл .env, который добавляют в .gitignore
Б) Хранят прямо в коде в зашифрованном виде
В) Сохраняют в файле config.json внутри репозитория
Г) Включают в название ветки

4. Какой уровень интеграции отвечает за оркестровку бизнес-процессов, затрагивающих несколько сервисов?

А) Уровень данных
Б) Уровень приложений
В) Уровень процессов
Г) Уровень пользовательского интерфейса

5. Для какого типа операций синхронное взаимодействие между модулями является обязательным требованием?

А) Ежедневная выгрузка отчетов в файловое хранилище
Б) Онлайн-проверка платежа и мгновенный ответ клиенту
В) Массовое обновление поискового индекса
Г) Отправка аналитических событий в систему сбора логов

6. Что из перечисленного является критерием эффективности автоматизации бизнес-процесса?

А) Уменьшение времени выполнения процесса (среднее время)
Б) Увеличение количества сотрудников, задействованных в процессе
В) Рост числа ручных операций
Г) Снижение частоты выполнения процесса

7. При выборе источника данных для интеграции решающим фактором в пользу реляционной СУБД (SQL) является:

А) высокая скорость записи неструктурированных логов
Б) необходимость поддержки сложных транзакций с ACID
В) масштабирование на сотни серверов без шардирования
Г) гибкая схема, позволяющая добавлять поля в любой момент

8. В процессе ETL трансформация данных выполняется:

А) после загрузки данных в приемник

Б) перед загрузкой данных в приемник (или во время)
В) только на стороне источника
Г) не выполняется вообще, данные передаются «как есть»

9. Как обычно обрабатываются дублирующиеся записи в процессе интеграции?
А) Все дубли сохраняются как есть
Б) Выполняется дедупликация по ключевым полям с выбором последней или наиболее полной записи
В) Система генерирует ошибку и останавливает процесс
Г) Дубли заменяются на случайные значения

10. Какой протокол чаще всего используется для передачи больших объемов потоковых данных с возможностью воспроизведения (replay) сообщений?
А) HTTP/1.1
Б) WebSocket
В) Kafka (протокол поверх TCP)
Г) SMTP

11. Какое преимущество формата XML перед JSON в корпоративных интеграциях?
А) Более компактный размер сообщений
Б) Поддержка пространств имен (namespaces) и строгих схем (XSD) для валидации
В) Лучшая читаемость для человека
Г) Более высокая скорость парсинга

12. Какой подход к версионированию сообщений позволяет одновременно поддерживать несколько версий в рамках одного сервиса?
А) Использование одной неизменяемой модели данных
Б) Версионирование через заголовки и использование адаптеров для преобразования старых сообщений в новые
В) Создание отдельного сервиса для каждой версии API
Г) Запрет на изменение формата сообщений после первого релиза

13. В модели GitHub Flow основная ветка (main) всегда должна быть в состоянии, готовом к релизу. Что из этого является обязательным шагом перед слиянием feature-ветки?
А) Создание отдельной ветки release
Б) Обновление версии в файле package.json
В) Прохождение всех CI-проверок и получение одобрения код-ревью
Г) Обязательное подписание коммитов GPG-ключом

14. Какая команда Git используется для переноса изменений из одной ветки в другую с созданием нового коммита слияния?
А) git rebase
Б) git merge --no-ff
В) git cherry-pick
Г) git pull --rebase

15. Что такое «политика защиты ветки» (branch protection rule) в системах Git (например, GitHub, GitLab)?
А) Правило, запрещающее создавать новые ветки
Б) Набор условий, которые должны быть выполнены для разрешения пуша или слияния в защищенную ветку
В) Шифрование содержимого ветки
Г) Автоматическое удаление ветки после слияния

КЛЮЧ К ТЕСТУ

Вариант 1		Вариант 1	
№ вопроса	Правильный ответ	№ вопроса	Правильный ответ
1	Б	1	Б
2	В	2	Б
3	А	3	А
4	Г	4	В
5	Б	5	Б
6	Г	6	А
7	Б	7	Б
8	А	8	Б
9	Б	9	Б
10	В	10	В
11	Г	11	Б
12	Б	12	Б
13	В	13	В
14	Б	14	Б
15	Б	15	Б

Лабораторные занятия № 1-6:

1. Разработка модульной структуры проекта (диаграммы модулей)
2. Разработка перечня артефактов и протоколов проекта
3. Настройка работы системы контроля версий (типов импортируемых файлов, путей, фильтров и др. параметров импорта в репозиторий)
4. Разработка и интеграция модулей проекта (командная работа)
5. Отладка отдельных модулей программного проекта
6. Организация обработки исключений

Тема 2.2. Инструментарий тестирования и анализа качества программных средств

1. Задание для устного опроса по темам

1. Перечислите основные типы инструментов отладки (статический анализ, трассировка, интерактивные отладчики, профилировщики). Для каждого типа приведите примеры популярных инструментов и опишите сценарии их использования.
2. В чем разница между простой точкой останова, условной точкой останова и точкой останова с действием (например, логгированием)? Как они помогают при локализации трудно воспроизводимых ошибок?
3. Что такое отладочный класс (например, Debug в .NET или логирующие декораторы в Python)? Как организовать условную компиляцию отладочного кода (директивы препроцессора, флаги сборки) и почему это важно для production-окружения?
4. Назовите сильные и слабые стороны ручного и автоматизированного тестирования. В каких случаях предпочтительнее ручное тестирование, а в каких – автоматизация? Приведите примеры.
5. Перечислите основные уровни тестирования (модульное, интеграционное, системное, приемочное). Для каждого уровня укажите цель, типичные инструменты и кто обычно выполняет эти тесты.
6. Опишите методы «белого ящика» (покрытие операторов, ветвей, условий) и «черного ящика» (эквивалентное разбиение, анализ граничных значений). Какой метод эффективнее для поиска ошибок в бизнес-логике, а какой – для проверки надежности?

7. Как организуется автоматическое выполнение тестов в конвейере непрерывной интеграции? Опишите роль тестовых отчетов, метрик покрытия и требований к времени выполнения тестового набора.
8. Объясните принцип иерархии классов исключений (на примере любого языка программирования). Почему важно перехватывать конкретные типы исключений, а не использовать «голый» catch (Exception)? Приведите примеры антипаттернов.
9. Какие методы используются для идентификации ошибок в работающей системе (логирование, трейсинг, метрики)? Опишите связь между уровнем логирования (ERROR, WARN, INFO, DEBUG) и последующим анализом инцидентов.
10. В чем разница между «предотвращением» исключения (проверка предусловий) и «обработкой» исключения (try-catch-finally)? Приведите сценарий, где лучше использовать проверку, а где – перехват с повторными попытками (retry) и задержками (backoff).

Тестовые вопросы

Тема 2.2. Инструментарий тестирования и анализа качества программных средств

Вариант 1

1. Какой инструмент относится к статическому анализу кода?
А) отладчик GDB
Б) линтер (например, pylint)
В) профилировщик
Г) трассировщик

2. Что такое условная точка останова (conditional breakpoint)?
А) точка, которая останавливает выполнение всегда
Б) точка, которая останавливает выполнение только при выполнении заданного условия
В) точка, которая записывает значения переменных в лог
Г) точка, которая игнорируется при сборке Release

3. Какой класс в .NET обычно используется для отладочного вывода информации?
А) Trace
Б) Debug
В) Console
Г) Log

4. Какая директива препроцессора в C# позволяет включать отладочный код только при сборке в режиме Debug?
А) #if DEBUG
Б) #define RELEASE
В) #pragma warning
Г) #region

5. Какой тип отладки позволяет в процессе выполнения остановиться и просмотреть стек вызовов и значения переменных?
А) трассировка
Б) профилирование
В) интерактивная отладка (с использованием отладчика)
Г) статический анализ

6. Какое тестирование выполняется разработчиками для проверки отдельных компонентов (модулей) программы?

А) системное тестирование
Б) интеграционное тестирование
В) модульное тестирование
Г) приемочное тестирование

7. Какой метод проектирования тестов относится к стратегии «белого ящика»?
А) эквивалентное разбиение
Б) анализ граничных значений
В) покрытие ветвей (branch coverage)
Г) тестирование сценариев использования

8. В каком случае автоматизированное тестирование является предпочтительным по сравнению с ручным?
А) при однократном выполнении тестов
Б) при необходимости исследовательского тестирования
В) при частых регрессионных проверках и больших объемах тестов
Г) когда пользовательский интерфейс часто меняется

9. Какой инструмент чаще всего используется для модульного тестирования в экосистеме Java?
А) Selenium
Б) JUnit
В) Postman
Г) Jenkins

10. Что обычно подразумевается под требованием к времени выполнения тестов в CI/CD-конвейере?
А) тесты должны выполняться не более заданного времени (например, 10 минут)
Б) тесты должны покрывать 100% кода
В) количество тестов должно быть минимальным
Г) тесты должны запускаться каждый час

11. Какая конструкция в Java предназначена для перехвата и обработки исключений?
А) try-catch-finally
Б) using
В) with
Г) ensure

12. Почему не рекомендуется перехватывать общее исключение Exception **(или** Throwable**) в большинстве случаев?**
А) это снижает производительность
Б) это скрывает специфические ошибки и затрудняет диагностику
В) это запрещено синтаксисом языка
Г) это приводит к утечкам памяти

13. Какой уровень логирования следует использовать для сообщений о критических сбоях, требующих немедленного вмешательства администратора?
А) INFO
Б) DEBUG
В) WARN
Г) ERROR

14. Какая стратегия повторных попыток (retry) предполагает увеличение задержки между попытками по экспоненте?
А) exponential backoff
Б) circuit breaker
В) retry with fixed delay
Г) fallback

15. Какой подход помогает предотвратить возникновение исключения, а не обрабатывать его уже после возникновения?
А) проверка предусловий (например, if перед операцией)
Б) использование try-catch
В) логирование ошибок
Г) использование блока finally

Вариант 2

1. Какой инструмент используется для пошагового выполнения кода и инспекции состояния программы во время работы?
А) статический анализатор
Б) отладчик (например, GDB)
В) линтер
Г) компилятор

2. Что такое точка останова с действием (action breakpoint)?
А) точка, которая останавливает выполнение и выполняет пользовательский код (например, вывод сообщения в лог)
Б) точка, которая только останавливает выполнение
В) точка, которая пропускает выполнение нескольких строк
Г) точка, которая изменяет значения переменных

3. Какой модуль в Python чаще всего используется для отладочного вывода (логирования) в продакшене?
А) logging
Б) pdb
В) sys
Г) os

4. Как в C++ принято условно включать отладочный код (например, дополнительные проверки) только для Debug-сборки?
А) #ifdef _DEBUG
Б) using namespace std;
В) #pragma once
Г) __declspec

5. Что такое профилирование программы?
А) поиск утечек памяти и узких мест производительности
Б) проверка синтаксиса и стиля кода
В) автоматическое выполнение тестов
Г) сборка проекта из исходников

6. Какой уровень тестирования проверяет корректность взаимодействия между несколькими компонентами или сервисами?
А) модульное тестирование
Б) интеграционное тестирование
В) системное тестирование
Г) приемочное тестирование

7. Какой метод тестирования основан на знании внутреннего устройства и кода программы?
А) черный ящик
Б) белый ящик
В) серый ящик
Г) функциональное тестирование

8. В каком случае ручное тестирование предпочтительнее автоматизированного?
А) при тестировании пользовательского интерфейса, который часто меняется и требует субъективной оценки
Б) при регрессионном тестировании больших объемов
В) при нагрузочном тестировании
Г) при тестировании API с фиксированной спецификацией

9. Какая среда или фреймворк часто используется для автоматизации тестирования веб-приложений?
А) JUnit
Б) Selenium WebDriver
В) Pytest (для модульных тестов)
Г) TestNG

10. Что показывает метрика «покрытие кода» (code coverage)?
А) процент операторов (или ветвей), выполненных во время тестов
Б) общее количество тестов
В) время выполнения тестового набора
Г) число найденных дефектов

11. Какое ключевое слово в C# используется для определения блока, который гарантированно выполняется независимо от того, возникло исключение или нет?
А) catch
Б) finally
В) throw
Г) try

12. Какой подход позволяет централизованно обрабатывать все необработанные исключения на уровне приложения?
А) перехват исключений в каждом методе
Б) использование глобального обработчика исключений
В) использование проверок if перед каждой операцией
Г) игнорирование исключений

13. Какой уровень логирования предназначен для вывода детальной отладочной информации, которая обычно не включается в продакшен-окружении?
А) ERROR
Б) WARN
В) DEBUG
Г) INFO

14. Что представляет собой паттерн «Circuit Breaker» в контексте обработки сбоев?

А) повторное выполнение операции с увеличивающейся задержкой

Б) разрыв цепи вызовов при высоком проценте ошибок, чтобы дать системе восстановиться

В) преобразование одного типа исключения в другой

Г) логирование всех ошибок в отдельный файл

15. Какую информацию обычно содержит стек-трейс (stack trace) исключения?

А) последовательность вызовов методов от точки входа до места возникновения исключения

Б) значения всех локальных переменных

В) время выполнения каждого метода

Г) версию сборки и имя пользователя

Ключ к тестам

Вариант 1		Вариант 2	
№ вопроса	Правильный ответ	№ вопроса	Правильный ответ
1	Б	1	Б
2	Б	2	А
3	Б	3	А
4	А	4	А
5	В	5	А
6	В	6	Б
7	В	7	Б
8	В	8	А
9	Б	9	Б
10	А	10	А
11	А	11	Б
12	Б	12	Б
13	Г	13	В
14	А	14	Б
15	А	15	А

Лабораторные занятия № 7-14:

1. Применение отладочных классов в проекте
2. Отладка проекта
3. Инспекция кода модулей проекта
4. Тестирование интерфейса пользователя средствами инструментальной среды разработки
5. Разработка тестовых модулей проекта для тестирования отдельных модулей
6. Выполнение функционального тестирования
7. Тестирование интеграции
8. Документирование результатов тестирования

Тема 2.3. Основы интеграции программных модулей

1. Задание для устного опроса по темам

1. Опишите полную структуру HTTP-запроса (стартовая строка, заголовки, тело). Перечислите основные HTTP-методы (GET, POST, PUT, DELETE, PATCH) с пояснением их идемпотентности, безопасности и типичных сценариев использования. Приведите примеры, когда каждый метод применяется.
2. Какие заголовки HTTP используются для управления кэшированием, типом содержимого, аутентификацией и согласованием формата? Опишите назначение заголовков Content-Type, Accept, Authorization, Cache-Control, Set-Cookie и приведите примеры их значений.

3. Что такое маршрутизация в веб-фреймворках? Как организовать группировку маршрутов (например, по версиям API или по функциональным модулям)? Как обрабатываются статические ресурсы (CSS, JS, изображения) и чем их обработка отличается от обработки динамических запросов?
4. В чем разница между параметрами пути (/users/{id}) и параметрами запроса (?page=2&limit=10)? Приведите примеры, когда предпочтительно использовать каждый тип. Как они извлекаются на серверной стороне?
5. Какие форматы данных могут передаваться в теле HTTP-запроса? Опишите application/json, application/x-www-form-urlencoded, multipart/form-data. В каких случаях используется каждый из них? Какие сложности возникают при обработке больших файлов через multipart?
6. Какими способами можно организовать валидацию данных, поступающих в REST API (например, через DTO, аннотации, схемы JSON)? Как следует обрабатывать ошибки валидации и возвращать понятные сообщения клиенту?
7. Опишите структуру HTTP-ответа. Перечислите основные группы статус-кодов (2xx, 3xx, 4xx, 5xx) с примерами наиболее употребительных кодов (200, 201, 204, 301, 400, 401, 403, 404, 500). Какие заголовки часто используются в ответах (например, Location, Retry-After)?
8. Что такое сериализация и десериализация при передаче данных по HTTP? Приведите примеры форматов (JSON, XML) и опишите, как происходит преобразование объектов в эти форматы и обратно. Какие проблемы могут возникнуть (циклические ссылки, даты, большие объемы)?
9. Как реализуется перенаправление (редирект) в HTTP? Какие статус-коды для этого используются (301, 302, 307, 308) и в чем их отличие? Что такое cookie, как они устанавливаются через заголовок Set-Cookie и для чего применяются в веб-приложениях?
10. Какие подходы используются для выполнения длительных или отложенных операций в веб-приложениях (фоновые задачи)? Опишите использование очередей сообщений (RabbitMQ, Redis), планировщиков (cron, Hangfire) и механизмов асинхронной обработки. Как они интегрируются с REST API (например, возврат статуса «принято» 202)?
11. Проведите сравнение механизмов аутентификации на основе сессий, JWT (JSON Web Token) и протокола OAuth 2.0. Каковы преимущества и недостатки каждого? В каких сценариях какой подход предпочтительнее?
12. Как организовать контроль доступа к ресурсам на основе ролей (RBAC) в REST API? Опишите архитектурные решения (например, использование фильтров, посредников, аннотаций). Как проверить права пользователя при каждом запросе?
13. Что такое WebSocket и в чем его ключевые отличия от HTTP? Опишите процесс установления соединения (handshake), обмена сообщениями и закрытия. Для каких задач (чат, онлайн-игры, биржевые котировки) WebSocket подходит лучше, чем REST?
14. Какие настройки соединения можно задавать при использовании WebSocket (таймауты, размеры сообщений, поддержка подпротоколов)? Как сервер может инициировать закрытие соединения? Какие методы обработки сообщений на сервере и клиенте существуют?
15. Сравните монолитную и микросервисную архитектуру: назовите основные преимущества и недостатки каждой, критерии выбора. Опишите синхронные (REST, gRPC) и асинхронные (брокеры сообщений, например, RabbitMQ, Apache Kafka) способы коммуникации между микросервисами. Какие факторы (задержки, согласованность, надежность) влияют на выбор способа взаимодействия?

Тестовые вопросы

Тема 2.3. Основы интеграции программных модулей

Вариант 1

1. Какой HTTP-метод считается одновременно безопасным (не изменяет состояние сервера) и идемпотентным?
А) POST
Б) PUT
В) DELETE
Г) GET

2. Какой заголовок HTTP используется для указания формата тела запроса (например, JSON или XML)?
А) Accept
Б) Content-Type
В) Authorization
Г) Cache-Control

3. Какой статус-код HTTP возвращается при успешном создании нового ресурса (например, при POST-запросе)?
А) 200 OK
Б) 201 Created
В) 204 No Content
Г) 202 Accepted

4. Для передачи файлов через форму в HTTP-запросе обычно используется:
А) application/json
Б) application/x-www-form-urlencoded
В) multipart/form-data
Г) text/plain

5. В REST API параметр, который входит в шаблон пути (например, /users/{userId}**), называется:**
А) Query-параметр
Б) Path-параметр
В) Body-параметр
Г) Header-параметр

6. Процесс преобразования объекта в формат, пригодный для передачи по сети (JSON, XML), называется:
А) десериализацией
Б) сериализацией
В) валидацией
Г) маршрутизацией

7. Какой формат используется для хранения JSON Web Token (JWT)?
А) XML
Б) JSON
В) YAML
Г) Protocol Buffers

8. Какой протокол аутентификации позволяет пользователю авторизоваться через стороннего провайдера (Google, Facebook) без передачи пароля сервису?

А) OAuth 2.0
Б) Базовая аутентификация (Basic Auth)
В) Аутентификация по сессии
Г) JWT

9. Что из перечисленного является характеристикой cookie, установленной сервером?
А) Cookie хранятся только на сервере
Б) Cookie передаются в заголовке Set-Cookie от сервера к клиенту
В) Cookie не могут содержать срок истечения
Г) Cookie шифруются автоматически

10. Какой механизм выполнения фоновых задач подразумевает использование брокера сообщений (например, RabbitMQ)?
А) Планировщик cron
Б) Отложенное выполнение через очередь задач
В) Создание отдельного потока внутри веб-запроса
Г) Использование Task.Run в коде контроллера

11. В чем основное отличие WebSocket от HTTP в контексте клиент-серверного взаимодействия?
А) WebSocket использует другой синтаксис URL
Б) WebSocket поддерживает полноценную двустороннюю связь без переустановки соединения
В) WebSocket не поддерживает шифрование
Г) WebSocket работает только на порту 8080

12. Какой статус-код HTTP используется для постоянного перенаправления (редиректа)?
А) 301
Б) 302
В) 307
Г) 308

13. Какой способ взаимодействия между микросервисами является асинхронным?
А) REST API через HTTP
Б) gRPC с блокирующими вызовами
В) Обмен сообщениями через брокер (Kafka, RabbitMQ)
Г) Прямой вызов методов через RPC

14. Какой протокол часто используют для синхронного высокопроизводительного обмена в микросервисах, основанный на HTTP/2 и бинарной сериализации?
А) SOAP
Б) REST
В) gRPC
Г) GraphQL

15. Для реализации ролевого разграничения доступа (RBAC) в REST API обычно используется:
А) проверка прав в каждом методе через атрибуты или фильтры
Б) отдельный сервер базы данных
В) изменение порта сервера для каждой роли
Г) передача ролей в заголовке User-Agent

Вариант 2

1. Какой HTTP-метод не является идемпотентным (повторение запроса может дать разные результаты)?
А) PUT
Б) DELETE
В) GET
Г) POST

2. Какой заголовок HTTP указывает серверу, какие форматы данных клиент готов принять в ответе?
А) Content-Type
Б) Accept
В) Accept-Encoding
Г) Connection

3. Какой статус-код HTTP означает, что запрос принят в обработку, но еще не завершен (используется для асинхронных операций)?
А) 200 OK
Б) 201 Created
В) 202 Accepted
Г) 204 No Content

4. Какой Content-Type используется для отправки данных HTML-формы в виде пар ключ-значение (без файлов)?
А) application/json
Б) multipart/form-data
В) application/x-www-form-urlencoded
Г) text/html

5. В REST API параметры, которые передаются в строке запроса после знака «?» (например, ?page=2&limit=10**), называются:**
А) Path-параметрами
Б) Query-параметрами
В) Body-параметрами
Г) Route-параметрами

6. Обратный процесс – преобразование полученного JSON/XML в объект программы – называется:
А) десериализацией
Б) сериализацией
В) валидацией
Г) деплоем

7. Какой механизм аутентификации требует хранения состояния на сервере (обычно в виде идентификатора сессии)?
А) JWT (JSON Web Token)
Б) Базовая аутентификация
В) Сессионная аутентификация (с использованием cookie)
Г) OAuth 2.0 без состояния

8. Какое утверждение о JWT верно?
А) JWT содержит только заголовок и полезную нагрузку, подпись отсутствует
Б) JWT может быть использован для безсессионной аутентификации, так как содержит информацию о пользователе и подпись

В) JWT передается только в теле запроса
Г) JWT не может содержать срок действия

9. Какие данные обычно хранятся в cookie на стороне клиента?
А) пароли в открытом виде
Б) идентификатор сессии или токен, а также настройки пользователя
В) весь кэш страниц
Г) исходный код приложения

10. Для организации периодических фоновых задач (например, ежедневная рассылка) чаще всего используется:
А) очередь сообщений с отложенной доставкой
Б) планировщик задач (cron, Hangfire, Quartz)
В) запуск задачи внутри каждого HTTP-запроса
Г) триггеры базы данных

11. Какой шаг обязателен при установке WebSocket-соединения?
А) HTTP-запрос с заголовком Upgrade: websocket
Б) отправка сообщения с ключом subscribe
В) закрытие всех HTTP-соединений
Г) использование порта 8080

12. Какой статус-код HTTP означает «временное перенаправление» (редирект без изменения метода запроса)?
А) 301
Б) 302
В) 307
Г) 308

13. Какое из следующих утверждений верно для синхронного взаимодействия микросервисов через REST?
А) Клиент не блокируется в ожидании ответа
Б) Взаимодействие основано на обмене сообщениями через брокер
В) Клиент отправляет запрос и ожидает ответа, соединение может быть открыто длительное время
Г) Такое взаимодействие не поддерживает протокол HTTP

14. Какой тип архитектуры предполагает развертывание всех компонентов в едином процессе и общей базе данных?
А) Микросервисная
Б) Монолитная
В) Серверлесс
Г) Событийно-ориентированная

15. Для ограничения доступа к определенным эндпоинтам в зависимости от роли пользователя в REST API обычно применяют:
А) проверку роли с использованием middleware или фильтров авторизации
Б) изменение DNS-имени сервера
В) создание отдельных портов для каждой роли
Г) хранение прав в имени пользователя

Ключ к тестам:

Ключ к варианту 1

Вариант 1		**Вариант 2**	
№ вопроса	Правильный ответ	№ вопроса	Правильный ответ
1	Г	1	Г
2	Б	2	Б
3	Б	3	В
4	В	4	В
5	Б	5	Б
6	Б	6	А
7	Б	7	В
8	А	8	Б
9	Б	9	Б
10	Б	10	Б
11	Б	11	А
12	А	12	В
13	В	13	В
14	В	14	Б
15	А	15	А

Лабораторные занятия № 15-22:

1. Создание клиентского приложения для работы с публичным API
2. Создание REST API приложения с реализацией: добавления, удаления, изменения и создания данных (от 3 - 4 сущностей)
3. Расширение функционала REST API приложения: работа с удаленным источником данных
4. Расширение функционала REST API приложения: работа со статическими изображениями (ресурсами) - загрузка, передача, удаление.
5. Расширение функционала REST API приложения: обработка path и query параметров
6. Расширение функционала REST API приложения: обработка ошибок, передача сообщений об ошибке пользователю
7. Расширение функционала REST API приложения: валидация полученных данных
8. Расширение функционала REST API приложения: добавление фоновых задач

Тема 2.4. Управление и мониторинг интегрированной системы

1. Задание для устного опроса по темам

1. Опишите подходы к управлению конфигурацией приложения для сред разработки, тестирования и производства. Какие форматы (.env, JSON, YAML, XML) и инструменты (например, Spring Cloud Config, Consul, переменные окружения) вы используете? Как обеспечить безопасное хранение секретов (паролей, ключей API) без их жесткого кодирования?
2. В чем отличие сборки (build) от компиляции? Опишите типовой процесс сборки современного приложения на примере одного из стеков (Java + Maven/Gradle, .NET + MSBuild, Node.js + npm). Как управляются внешние зависимости, их версии и как обеспечивается воспроизводимость сборки?
3. Перечислите стандартные уровни логирования (TRACE, DEBUG, INFO, WARN, ERROR, FATAL). Для каждого уровня приведите реальные сценарии использования в коде. Как конфигурируется логирование (через файлы logback.xml, appsettings.json, переменные окружения) и как динамически менять уровень без перезапуска приложения?

4. В чем преимущества структурированного логирования (например, в формате JSON) по сравнению с текстовыми логами? Как настроить запись логов в файлы с автоматической ротацией (по размеру, по времени) и сжатием архивов? Приведите пример конфигурации для популярных библиотек (Serilog, Log4j, Python logging).
5. Какие метрики жизненно важны для мониторинга работоспособности приложения (CPU, память, количество запросов, время отклика, частота ошибок, количество активных соединений)? Опишите принцип работы сборщика метрик (например, Micrometer, Prometheus client) и как внедрить его в код приложения.
6. Как организовать сбор логов и метрик в единую систему? Опишите архитектуру стека ELK (Elasticsearch, Logstash, Kibana) и связки Prometheus + Grafana. Какие данные они обрабатывают и как помогают в обнаружении проблем?
7. Что такое контейнеризация и чем она отличается от виртуализации? Опишите структуру Dockerfile: назначение инструкций FROM, RUN, COPY, ADD, CMD, ENTRYPOINT, EXPOSE, ENV. Как правильно передавать конфигурацию в контейнер (переменные окружения, монтирование файлов, ConfigMap)?
8. Для каких целей используется docker-compose.yml? Опишите, как в нем описываются сервисы, сети, тома и зависимости. Приведите пример, как поднять многокомпонентное приложение (веб-сервис + база данных + кэш) с помощью Compose.
9. Опишите конвейер CI/CD: этапы сборки, тестирования, упаковки в образ, публикации в реестр и развертывания. Какие инструменты используются (Jenkins, GitLab CI, GitHub Actions, TeamCity)? Как автоматизировать процесс и обеспечить откат при неудачном деплое?
10. Какие задачи решает Kubernetes (или Docker Swarm) при развертывании микросервисов? Объясните концепции: **Pod**, **Service**, **Deployment**, **ConfigMap**, **Secret**. Как с их помощью управлять конфигурацией, обновлениями (rolling update) и масштабированием приложения?

Тестовые вопросы

Тема 2.4. Управление и мониторинг интегрированной системы

Вариант 1

1. Какой способ передачи конфигурации в приложение является наиболее безопасным для хранения секретов (паролей, ключей API)?

А) Хранение в файле application.properties внутри репозитория

Б) Передача через переменные окружения или внешний секретный менеджер (Vault)

В) Жесткое кодирование в исходном коде

Г) Хранение в файле config.yml в папке сборки

2. Какой инструмент управления зависимостями используется в экосистеме Java для сборки проектов наряду с Maven?

А) npm

Б) Gradle

В) pip

Г) NuGet

3. Какой уровень логирования следует использовать для детальной отладочной информации, которая не должна попадать в продакшен?

А) INFO
Б) ERROR
В) DEBUG
Г) WARN

4. Какой файл конфигурации обычно используется для настройки логирования в приложениях на Java (Logback)?
А) appsettings.json
Б) logback.xml
В) logging.conf
Г) log4j2.yaml

5. Что из перечисленного является преимуществом структурированного логирования в формате JSON?
А) Меньший размер логов по сравнению с текстовыми
Б) Лучшая интеграция с системами поиска и аналитики (например, Elasticsearch)
В) Возможность записывать бинарные данные
Г) Отсутствие необходимости в уровнях логирования

6. Какая метрика относится к категории «нагрузка» (load) при мониторинге веб-приложения?
А) Количество свободной оперативной памяти
Б) Количество активных HTTP-соединений или запросов в секунду
В) Температура процессора
Г) Версия операционной системы

7. Какой сборщик метрик чаще всего используется в связке с системой визуализации Grafana для хранения временных рядов?
А) Elasticsearch
Б) Micrometer
В) Prometheus
Г) Logstash

8. Что делает инструкция COPY **в Dockerfile?**
А) Выполняет команду в контейнере
Б) Копирует файлы и папки с хоста в образ
В) Устанавливает рабочую директорию
Г) Определяет команду по умолчанию

9. В чем главное отличие контейнеризации от виртуализации?
А) Контейнеры используют общее ядро хостовой ОС, а виртуальные машины имеют собственное ядро
Б) Контейнеры требуют гипервизор
В) Виртуальные машины более легковесны
Г) Контейнеры не поддерживают изоляцию процессов

10. Какой файл в проекте Docker Compose описывает сервисы, сети и тома?
А) Dockerfile
Б) docker-compose.yml
В) compose.json
Г) docker.service

11. Какой этап не относится к типовому конвейеру CI/CD?
А) Сборка и тестирование кода
Б) Упаковка приложения в контейнер
В) Публикация образа в реестре
Г) Написание технической документации

12. Что из перечисленного позволяет автоматически развернуть приложение на кластер и управлять его жизненным циклом?
А) Docker
Б) Ansible
В) Kubernetes
Г) Terraform

13. Как называется объект Kubernetes, который определяет желаемое состояние (количество реплик, стратегию обновления) для пода?
А) Pod
Б) Service
В) Deployment
Г) ConfigMap

14. Какой объект Kubernetes используется для хранения конфигурационных данных в виде пар ключ-значение и передачи их в поды?
А) Secret
Б) PersistentVolume
В) ConfigMap
Г) Ingress

15. Что такое Rolling Update в Kubernetes?
А) Полная остановка всех старых подов перед созданием новых
Б) Постепенное обновление подов новой версией без прерывания обслуживания
В) Создание резервной копии всего кластера
Г) Удаление всех подов и их пересоздание

Вариант 2

1. Какой подход к конфигурации позволяет изменять параметры без перекомпиляции и без перезапуска приложения в некоторых реализациях?
А) Хранение в коде с использованием #define
Б) Внешние файлы конфигурации, которые перечитываются при изменении
В) Использование только переменных окружения (изменение требует перезапуска)
Г) Встраивание конфигурации в базу данных

2. Какой инструмент сборки использует файл pom.xml **для описания проекта?**
А) Gradle
Б) Maven
В) Ant
Г) Make

3. Какой уровень логирования предназначен для записи информационных сообщений о ходе работы приложения (например, успешные операции)?
А) DEBUG
Б) ERROR

В) WARN
Г) INFO

4. В конфигурации логирования .NET Core (Serilog) параметры настройки часто задаются в файле:
А) web.config
Б) log4net.config
В) appsettings.json
Г) logging.properties

5. Что означает ротация (rotation) логов?
А) Шифрование старых логов
Б) Автоматическое переключение на новый файл лога по достижении размера или времени, с архивированием старых
В) Преобразование текстовых логов в JSON
Г) Отправка логов в централизованную систему

6. Какая метрика показывает среднее время обработки запроса и используется для оценки производительности?
А) CPU usage
Б) Request duration (или latency)
В) Memory usage
Г) Number of threads

7. Какой компонент в стеке ELK отвечает за визуализацию и построение дашбордов?
А) Elasticsearch
Б) Logstash
В) Kibana
Г) Beats

8. Какая инструкция Dockerfile устанавливает рабочую директорию для последующих команд?
А) RUN
Б) CMD
В) WORKDIR
Г) ENV

9. Что делает инструкция EXPOSE **в Dockerfile?**
А) Открывает порт на хосте для внешнего доступа
Б) Указывает порты, которые контейнер будет прослушивать во время выполнения (носит документальный характер и используется для связывания)
В) Устанавливает переменные окружения
В) Запускает сервис на указанном порту

10. Для чего используется docker-compose up -d**?**
А) Для сборки образа и запуска контейнеров в фоновом режиме
Б) Для остановки всех контейнеров
В) Для удаления всех образов
Г) Для просмотра логов всех сервисов

11. Какой инструмент CI/CD использует файл .gitlab-ci.yml **для описания пайплайна?**
А) Jenkins
Б) GitHub Actions

В) GitLab CI
Г) TeamCity

12. Какой из перечисленных инструментов является платформой оркестрации контейнеров?
А) Docker Engine
Б) Kubernetes
В) Ansible
Г) Terraform

13. Какой объект Kubernetes предоставляет стабильный IP-адрес и DNS-имя для доступа к группе подов?
А) Pod
Б) Service
В) Deployment
Г) Ingress

14. Какой объект Kubernetes предназначен для хранения конфиденциальной информации (пароли, токены) в закодированном виде?
А) ConfigMap
Б) Secret
В) PersistentVolume
Г) NetworkPolicy

15. Какой механизм Kubernetes позволяет автоматически увеличивать или уменьшать количество реплик подов на основе метрик (например, загрузка CPU)?
А) Rolling Update
Б) Horizontal Pod Autoscaler (HPA)
В) Vertical Pod Autoscaler
Г) Cluster Autoscaler

Ключ к тестам:

Вариант 1		**Вариант 2**	
№ вопроса	Правильный ответ	№ вопроса	Правильный ответ
1	Б	1	Б
2	Б	2	Б
3	В	3	Г
4	Б	4	В
5	Б	5	Б
6	Б	6	Б
7	В	7	В
8	Б	8	В
9	А	9	Б
10	Б	10	А
11	Г	11	В
12	В	12	Б
13	В	13	Б
14	В	14	Б
15	Б	15	Б

Лабораторные занятия № 23-25:

1. Настроить конфигурацию rest api приложения (порт, хост, данные для подключения к

источнику данных, приватные ключи).
2. Внедрить логирование в rest api приложение
3. Упаковка rest api приложения в контейнер и доставка на другое устройство

5.2. Критерии оценивания

5.2.1. Критерии оценивания устного ответа

При оценке устного ответа, обучающегося учитывается:

1) полнота и правильность ответа;
2) степень осознанности, понимания изученного;

Отметка «5»: ответ правильный, полный в соответствии с изученным материалом; материал изложен в определенной логической последовательности, литературным языком.

Отметка «4»: ответ правильный, полный в соответствии с изученным материалом; материал изложен в определенной логической последовательности; возможны отдельные затруднения в формулировке выводов.

Отметка «3»: ответ, в котором в основном правильно, но схематично или с отклонениями от последовательности изложения раскрыт материал или неполный, несвязный ответ, изложенный нелогично

Отметка «2»: при ответе обнаружено непонимание обучающимся основного содержания учебного материала, неумение его анализировать допущены существенные ошибки, которые обучающийся не смог исправить при наводящих вопросах преподавателя, отсутствует логика в изложении материала, нет необходимых обобщений и самостоятельной оценки фактов; недостаточно сформированы навыки устной речи.

5.2.2. Критерии оценивания выполнения заданий на лабораторных и практических занятиях

Отметка «5»: работа выполнена полностью и правильно; сделаны правильные выводы.

Отметка «4»: работа выполнена правильно с учетом 1-2 несущественных ошибок, исправленных самостоятельно по требованию преподавателя.

Отметка «3»: работа выполнена правильно не менее чем на половину или допущены 3-4 существенные ошибки.

Отметка «2»: допущены 5 и более существенные ошибки в ходе работы, которые обучающийся не может исправить даже по требованию преподавателя.

Задания к лабораторным занятиям представлены в методических указаниях к лабораторным занятиям по МДК.02.02 осуществление интеграции программных модулей

Методы оценки:

– Защита отчетов по выполненному заданию на лабораторных и практических занятиях.

– Интерпретация результатов наблюдений за деятельностью обучающегося на лабораторных занятиях.

5.2.3. Критерии оценивания тестовых заданий

Оценка в баллах	Степень выполнения задания
Неуд.	Выполнено от 0 до 49,9 % предложенных заданий

Удов.	Выполнено от 50 до 69,9% предложенных заданий
Хор.	Выполнено от 70 до 89,9% предложенных заданий
Отл.	Выполнено от 90 до 100% предложенных заданий

5.2.4. Общая классификация ошибок

При оценке знаний и умений учитываются ошибки и недочеты в работе.

Грубыми считаются ошибки:

- незнание определения основных понятий, законов;
- неумение выделить в ответе главное;
- обобщить результаты изучения;
- неумение применить знания для решения задач;
- неумение использовать полученные данные для выводов;
- неумение пользоваться первоисточниками, учебником, справочником;
- нарушение техники безопасности, небрежное отношение к оборудованию

Негрубыми считаются ошибки:

- неточность формулировок, определений, понятий, законов, вызванная неполнотой охвата основных признаков определяемого понятия или заменой 1-3 из этих признаков второстепенными;
- ошибки, вызванные несоблюдением условий проведения лабораторных занятий;
- недостаточно продуманный план устного ответа (нарушение логики изложения, подмена отдельных основных вопросов второстепенными);
- нерациональные методы работы со справочной литературой;
- неумение выполнять лабораторные задания в общем виде.

6. ОЦЕНОЧНЫЕ СРЕДСТВА ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО МЕЖДИСЦИПЛИНАРНОМУ КУРСУ МДК 02.02 Осуществление интеграции программных модулей

Промежуточная аттестация по междисциплинарному курсу МДК 02.01. Разработка программного обеспечения проводится в форме дифференцированного зачета в 6 и 7 семестрах.

6.1. Оценочные средства курсового проекта по междисциплинарному курсу МДК 02.02 Осуществление интеграции программных модулей

6.1.1. Примерная тематика курсовых проектов:

- Проектирование программного модуля «_____».
- Проектирование информационной системы «____».
- Проектирование приложения для автоматизации деятельности «____».
- Проектирование и разработка автоматизированного рабочего места специалиста «_____»
- Проектирование обучающего интерактивного пособия по дисциплине/теме «_____».
- Проектирование обучающей и контролирующей программы по дисциплине/теме «_____».

Порядок выполнения курсового проекта, структура и требования к содержанию курсового проекта, правила оформления, требования к представляемым документам, порядок защиты и критерии оценивания курсового проекта описываются в методических

указаниях по выполнению курсового проекта по междисциплинарному курсу МДК. 02.02 Осуществление интеграции программных модулей

6.1.2. Критерии оценивания курсового проекта

Оценка результатов курсового проекта проводиться по следующим критериям:

1. Умение выполнять постановку задачи, анализировать и исследовать задачу, выполнять построение модели.
2. Умение разрабатывать классы, осуществлять формализацию расчетов.
3. Качество выполненного функционального и инфологического проектирования.
4. Умение разрабатывать структурную схему интерфейса, схемы связности модулей, схему движения информационных потоков.
5. Оформление курсового проекта в соответствии со стандартными требованиями.

Пункты с 1 по 5 дают до 50% вклада в итоговую оценку студента.

6. Умение защищать результаты своего проекта, грамотное построение речи, использование при выступлении специальных терминов.
7. Способность кратко и наглядно изложить результаты проекта. Полнота и правильность ответов на вопросы.

Пункты 6, 7 дают до 35% вклада в итоговую оценку студента.

8. Уровень самостоятельности, творческой активности и оригинальности при выполнении проекта.

Пункт 8 дает до 15 % вклада в итоговую оценку студента.

Повышает ценность курсового проекта его практическое использование в учебном процессе или на предприятии/организации.

Курсовой проект оценивается по пятибалльной системе.

Оценка «отлично» ставится студенту, который в срок, в полном объеме и на высоком уровне выполнил курсовой проект по выбранной теме (например, развитие и интеграция HR-модуля, создание системы управления проектами, разработка платежного модуля или другого из предложенных направлений). При защите и написании проекта студент продемонстрировал глубокое владение технологиями и методологиями разработки. В рамках проекта выполнены следующие работы:

– проведен всесторонний анализ требований и сформировано техническое задание (ТЗ) на модуль, согласованное с заказчиком или руководителем, с четким описанием функциональных и нефункциональных требований;

– разработана архитектура модуля, обоснован выбор стека технологий с учетом масштабируемости, производительности и безопасности; представлены структурные схемы и описание взаимодействия компонентов;

– спроектирована структура базы данных модуля на логическом и физическом уровнях, построены ER-диаграммы, определены индексы и ограничения целостности;

– реализована бизнес-логика и RESTful API модуля в полном соответствии со спецификацией – все методы (включая валидацию, обработку ошибок, пагинацию) документированы через OpenAPI/Swagger;

– разработан пользовательский интерфейс (UI/UX) модуля: спроектированы экранные формы, диалоги, отчеты, обеспечена адаптивность и удобство работы;

– внедрены механизмы безопасности и аутентификации (JWT/OAuth, сессии, ролевое разграничение доступа) с защитой от распространенных уязвимостей;

– настроено интеграционное взаимодействие с внешними системами – реализован синхронный или асинхронный обмен данными (REST, gRPC, брокеры сообщений), обеспечена обработка сбоев и повторных попыток;

– проведено тестирование модуля и отладка интеграции: разработаны модульные, интеграционные и приемочные тесты, составлены тестовые сценарии, обеспечено покрытие кода;

– подготовлена полная документация: код прокомментирован, API описан (коллекция Postman), составлены инструкции по установке, настройке и эксплуатации;

– выполнено развертывание модуля на тестовом стенде, проведены интеграционные испытания, произведен ввод в опытную эксплуатацию, зафиксированы результаты.

Отчет подготовлен в соответствии с предъявляемыми требованиями, содержит все необходимые диаграммы, схемы и листинги. Отзыв руководителя – положительный, защита проведена на высоком уровне.

Оценка «хорошо» ставится студенту, который выполнил курсовой проект в срок, но допустил незначительные замечания по одному или нескольким этапам. При защите студент в целом ориентируется в проекте, но отвечает не на все вопросы. Все перечисленные выше работы выполнены, однако имеются следующие недочеты:

– в анализе требований и ТЗ не полностью учтены отдельные сценарии использования или второстепенные функции;

– архитектура и стек технологий обоснованы, но возможны упрощения – отсутствует детальное описание выбора конкретных библиотек или фреймворков;

– ER-диаграммы и структура БД построены корректно, но не в полной мере оптимизированы (не все индексы добавлены, не проработаны вопросы шардирования/репликации);

– бизнес-логика и API реализованы, но не обработаны некоторые граничные условия, недостаточно проработана валидация или документация методов;

– пользовательский интерфейс разработан, но имеются мелкие огрехи в юзабилити, не реализована часть адаптивности или отсутствуют всплывающие подсказки;

– механизмы безопасности внедрены, но отсутствует аудит логирования действий пользователей или не настроены CORS политики;

– интеграционное взаимодействие настроено, но не полностью покрыто тестами и нет документации по форматам обмена;

– тестирование проведено, но покрытие кода тестами ниже рекомендуемого уровня, не все интеграционные сценарии проверены;

– документация подготовлена, но содержит незначительные пропуски (отсутствует описание некоторых эндпоинтов или инструкция по развертыванию);

– развертывание выполнено, но при опытной эксплуатации выявлены единичные ошибки, не влияющие на основной функционал, либо не полностью заполнены журналы испытаний.

Пояснительный текст к моделям и диаграммам частично недоработан, отчет оформлен с небольшими отступлениями. Отзыв руководителя положительный, но содержит рекомендации.

Оценка «удовлетворительно» ставится студенту, который выполнил проект с существенными просчетами и ошибками, либо не в полном объеме реализовал отдельные этапы. При защите студент показывает слабое понимание принятых решений. Характерные недостатки:

– анализ требований проведен поверхностно, ТЗ составлено нечетко, отсутствует описание нефункциональных требований;

– архитектура выбрана без должного обоснования, выбор технологий не оптимален, структурные схемы неполны или противоречивы;

– структура БД спроектирована с ошибками – некорректные связи, отсутствие первичных ключей или индексов, не соблюдены принципы нормализации;

– бизнес-логика и API реализованы с нарушениями REST принципов, присутствуют ошибки в обработке запросов и валидации, методы не документированы;

– интерфейс пользователя разработан, но неудобен в использовании, отсутствует реакция на действия пользователя, формы не соответствуют требованиям;

– механизмы безопасности реализованы лишь частично – нет разграничения по ролям, отсутствует защита от атак (SQL-инъекции, XSS), не настроено шифрование;

– интеграционное взаимодействие выполнено, но нестабильно – нет обработки ошибок, повторных попыток, асинхронные взаимодействия не реализованы;

– тестирование проведено несистематически – большинство сценариев не проверены, отладка интеграции не выполнена;

– документация отсутствует или составлена формально, не позволяет установить и использовать модуль;

– развертывание на тестовом стенде выполнено, но интеграционные испытания не проведены или выявили серьезные проблемы, модуль не готов к эксплуатации.

Отчет содержит ошибки в оформлении, неполные диаграммы, отсутствует пояснительный текст. Отзыв руководителя содержит замечания. При защите студент допускает ошибки в понимании архитектуры и кода.

Оценка «неудовлетворительно» ставится студенту, который не выполнил курсовой проект в установленный срок, либо выполнил его с грубыми нарушениями требований. Признаки неудовлетворительной работы:

– анализ требований и ТЗ отсутствуют или не соответствуют предметной области;

– архитектура не разработана или скопирована без осмысления, выбор технологий не обоснован;

– структура БД не спроектирована или не соответствует заданию;

– бизнес-логика и API не реализованы или реализованы с критическими ошибками, делающими модуль неработоспособным;

– пользовательский интерфейс отсутствует или полностью нерабочий;

– механизмы безопасности и аутентификации не реализованы;

– интеграционное взаимодействие с внешними системами не выполнено;

– тестирование модуля и отладка интеграции не проводились;

– документация отсутствует;

– развертывание не выполнено, модуль не готов к эксплуатации.

Отчет не соответствует требованиям или не представлен вовсе, отсутствуют необходимые диаграммы и схемы. Отзыв руководителя отрицательный. Защита не состоялась или показала полное непонимание материала.

Студенты, не имеющие положительной оценки за курсовой проект к промежуточной аттестации по междисциплинарному курсу, не допускаются.

7. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ

Основные печатные издания

1. Емелина, Е. И., Поддержка и тестирование программных модулей: учебник / Е. И. Емелина. - Москва: КноРус, 2025. — 267 с.

2. Зубкова Т.М. Технология разработки программного обеспечения. – Санкт-Петербург: Лань, 2021. – 252 с.

3. Шитов, В. Н., Проектирование и разработка интерфейсов пользователя: учебное пособие / В. Н. Шитов, К. Е. Успенский. — Москва: КноРус, 2023. — 294 с.

Основные электронные издания:

1. Гагарина, Л. Г. Технология разработки программного обеспечения: учебное пособие / Л.Г. Гагарина, Е.В. Кокорева, Б.Д. Сидорова-Виснадул; под ред. Л.Г. Гагариной. — Москва: ФОРУМ: ИНФРА-М, 2022. — 400 с. — (Среднее профессиональное образование). - ISBN 978-5-8199-0812-9. - Текст: электронный. - URL: https://znanium.com/catalog/product/1794453

2. Зараменских, Е. П. Проектирование информационных систем: учебник и практикум для среднего профессионального образования / Е. П. Зараменских. — 2-е изд. — Москва: Издательство Юрайт, 2025. — 119 с. — (Профессиональное образование). — ISBN 978-5-534-21417-8. — Текст: электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/571330