МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
Факультет компьютерных технологий и прикладной математики

УТВЕРЖДАЮ:
Проректор по учебной работе,
качеству образования – первый
проректор
_____ Хагуров Т.А.
подпись
« 29 » августа 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Б1. О.34 Микросервисная архитектура

Направление подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Профиль Современные методы машинного обучения и компьютерного зрения

Форма обучения очная

Квалификация бакалавр

Краснодар 2025

Рабочая программа дисциплины «Микросервисная архитектура» составлена в соответствии с федеральным государственным образовательным стандартом высшего образования (ФГОС ВО) по направлению подготовки 02.03.02 Фундаментальная информатика и информационные технологии

Программу составил(а):
Приходько Татьяна Александровна, доцент, к. т. н.
Ф.И.О., должность, ученая степень, ученое звание — подпись

Рабочая программа дисциплины «Микросервисная архитектура» утверждена на заседании кафедры Вычислительных технологий
протокол № 1 « 26 » _августа_ 2025 г.

И.о. заведующего кафедрой (разработчика) Приходько Т.А

Утверждена на заседании учебно-методической комиссии факультета Компьютерных Технологий и Прикладной Математики
протокол № 1 « 28 » _августа_ 2025 г.
Председатель УМК факультета Коваленко А.В.
фамилия, инициалы — подпись

Рецензенты:

Мостовой Евгений Викторович, генеральный директор ООО «Портал-Юг», e-mail: mostovoy@portal-yug.ru

Луценко Евгений Вениаминович, доктор экономических наук, кандидат технических наук, профессор кафедры компьютерных технологий и систем Федерального государственного бюджетное образовательное учреждение высшего образования «Кубанский государственный аграрный университет имени И.Т. Трубилина», e-mail: prof.lutsenko@gmail.com

1. ЦЕЛИ И ЗАДАЧИ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1.1 Цель освоения дисциплины

Цели дисциплины: формирование у студентов общих компетенций, формирующих способность решать задачи профессиональной деятельности в области проектирования, развертывания, эксплуатации и администрирования надежных, масштабируемых и эволюционирующих распределенных систем, основанных на микросервисной архитектуре.

1.2 Задачи дисциплины

- **Привить студентам понимание эволюции архитектуры:** От монолитов к сервисно-ориентированной архитектуре (SOA) и, наконец, к микросервисам.
- **Умение проводить сравнительный анализ:** Умение объективно оценивать преимущества и недостатки микросервисов по сравнению с монолитной архитектурой. Понимание, когда микросервисы оправданы, а когда нет (избегание "архитектурного хайпа").
- **Изучение принципов:** Освоение ключевых принципов, таких как слабая связанность, сильная связность сервисов, независимость развертывания и автономность команд.

Приобретение практических навыков проектирования и разработки:

- Декомпозиция: Научить правильно разбивать бизнес-домен на отдельные, слабосвязанные сервисы на основе методологий типа Domain-Driven Design (DDD), определения ограниченных контекстов (Bounded Context) и выделения сервисов вокруг бизнес-возможностей.
- Проектирование API и взаимодействия: Научить проектировать эффективные API (REST, gRPC, GraphQL) и организовывать взаимодействие между сервисами (синхронное vs асинхронное, с использованием messaging брокеров как RabbitMQ или Kafka).
- Управление данными: Понимание принципов децентрализованного управления данными, когда каждый сервис владеет своей собственной базой данных и общается с другими через четко определенные API.
- Независимое развертывание: **Научить настраивать CI/CD (Continuous Integration / Continuous Deployment) пайплайны для независимого развертывания отдельных сервисов.**
- Контейнеризация и оркестрация: **Дать понимание технологий контейнеризации (Docker) и оркестрации (Kubernetes), которые являются фундаментом для управления жизненным циклом микросервисов.**

1. Место дисциплины (модуля) в структуре образовательной программы

Дисциплина «Микросервисная архитектура» относится к базовой части блока Б1 Дисциплины (модули).

Для изучения дисциплины студент должен владеть знаниями, умениями и навыками по дисциплинам: Дискретная математика, Конструирование алгоритмов и структур данных, Организация вычислительных систем, Алгоритмы и структуры данных, Теория вероятностей и математическая статистика, с которыми дисциплина связана логически и содержательно-методически.

Дисциплина в значительной степени взаимодействует для формирования компетенций с дисциплинами: Объектно-ориентированное программирование и шаблоны проектирования

Операционные системы; Web-разработка; Компьютерные сети; DevOps; Анализ и проектирование информационных систем; Разработка мобильных приложений.

1.4 Профессиональные роли в структуре образовательной программы

Роль 1: Data Engineer (Инженер по данным)
Задачи:

1. Проектирование и построение ETL-процессов
2. Создание и оптимизация хранилищ данных
3. Обеспечение качества и доступности данных
4. Настройка инфраструктуры для обработки больших данных
5. Интеграция разрозненных источников данных
6. Работа с данными в области природопользования, медицины, связи и телекоммуникаций

Роль 2: ML Engineer (Инженер МО)
Задачи:

1. Реализация ML-моделей в продуктивных системах
1. Оптимизация производительности и масштабирование моделей
1. Разработка ML-пайплайнов и автоматизация процессов
1. Мониторинг качества моделей в продуктиве
1. Интеграция ML-решений с бизнес-приложениями

Роль 3: MLOps (Специалист по эксплуатации ИИ)
Задачи:

1. Автоматизация процессов обучения и развертывания моделей
2. Мониторинг производительности ML-систем
3. Управление версиями моделей и данных
4. Обеспечение CI/CD для ML-проектов
5. Оптимизация вычислительных ресурсов

1.5 Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы

Изучение данной учебной дисциплины направлено на формирование у обучающихся следующих компетенций:

ОПК-3	***Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям***
ОПК-3.1	Аргументировано применяет методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности
ОПК-3.2	Использует инструментальные, программные и аппаратные средства измерений для оценки качества программного обеспечения
ОПК-4	***Способен участвовать в разработке технической документации программных продуктов и комплексов с использованием стандартов, норм и***

правил, а также в управлении проектами создания информационных систем на стадиях жизненного цикла

ОПК-4.4 Имеет практический опыт анализа и интерпретации информационных систем

ОПК-5 ***Способен инсталлировать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности***

ОПК-5.3 Имеет практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий

ОПК-6 ***Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности***

ОПК-6.1 Аргументировано применяет современные информационные технологии, в том числе отечественные, при создании программных продуктов и программных комплексов различного назначения

ОПК-6.2 Ориентируется в современных положениях и концепциях прикладного и системного программного обеспечения, архитектуры компьютеров и сетей (в том числе и глобальных), технологии создания и сопровождения программных продуктов и программных комплексов

LC-3 Способен проектировать и поддерживать архитектуру систем искусственного интеллекта

LC-3.1 Создает и развивает архитектуры системы ИИ на всех этапах жизненного цикла

PL-1 ***Способен применять язык программирования Python для решения задач в области ИИ***

PL-1.1 Разрабатывает и отлаживает прикладные решения разной сложности и для разного круга конечных пользователей с использованием языка программирования Python, тестирует, испытывает и оценивает качество таких решений

PL-1.2 Осуществляет выбор инструментов разработки на Python, приемлемых для создания прикладной системы обработки научных данных, машинного обучения и визуализации с заданными требованиями

PL-1.3 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности

PL-2 ***Способен применять JVM-совместимые языки программирования для решения задач в области ИИ***

PL-2.1 Разрабатывает и отлаживает прикладные решения разного уровня сложности и для широкого круга конечных пользователей с использованием JVM-совместимых языков программирования, тестирует, испытывает и оценивает качество таких решений

PL-2.2 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности

2. Структура и содержание дисциплины

2.1 Распределение трудоёмкости дисциплины по видам работ

Общая трудоёмкость дисциплины составляет 2 зач. ед. (72 часа), их распределение по видам работ представлено в таблице

Вид учебной работы	**Всего часов**	**Семестры (часы)**

			7	
Контактная работа, в том числе:		**54,2**	**54,2**	
Аудиторные занятия (всего):		**50**	**50**	
Занятия лекционного типа		**16**	16	
Лабораторные занятия		**34**	34	
Занятия семинарского типа (семинары, практические занятия)				
Иная контактная работа:		**2,2**	**2,2**	
Контроль самостоятельной работы (КСР)		**2**	2	
Промежуточная аттестация (ИКР)		**0,2**	0,2	
Самостоятельная работа, в том числе:		**19,8**	**19,8**	
Курсовая работа				
Проработка учебного (теоретического) материала		**7.8**	7.8	
Выполнение индивидуальных заданий (подготовка сообщений, презентаций)		**12**	12	
Реферат				
Подготовка к текущему контролю				
Контроль:		**0**	**0**	
Подготовка к экзамену		**0**	**0**	
Общая трудоемкость	**час.**	**72**	**72**	
	в том числе контактная работа	**54,2**	**54,2**	
	зач. ед	**2**	**2**	

2.2 Структура дисциплины

Распределение видов учебной работы и их трудоемкости по разделам дисциплины.

Разделы (темы) дисциплины, изучаемые в 5 семестре

№	Наименование разделов (тем)	Количество часов				
		Всего	Аудиторная работа			Внеаудиторная работа
			Л	ПЗ	ЛР	СРС
1	2	3	4	5	6	7
1.	Введение в Микросервисную архитектуру. От Монолита к Микросервисам	8	2		4	2
2.	**Декомпозиция и проектирование сервисов. Антипаттерны**	8	2		4	2
3.	**Взаимодействие между микросервисами. Синхронные и асинхронные паттерны**	8	2		4	2
4.	Управление данными в распределенной системе	8	2		4	2
5.	Обнаружение сервисов, конфигурирование и безопасность	8	2		4	2
6.	Наблюдаемость (Observability) и мониторинг	8	2		4	2
7.	Развертывание и оркестрация микросервисов	10	2		4	4
8.	Тестирование и надежность микросервисных систем	11,8	2		6	3,8
ИТОГО по разделам дисциплины		**69,8**	**16**		**34**	**19,8**
Контроль самостоятельной работы (КСР)		2				
Промежуточная аттестация (ИКР)		0,2				

№	Наименование разделов (тем)	Количество часов				
		Всего	Аудиторная работа			Внеаудиторная работа
			Л	ПЗ	ЛР	СРС
1	2	3	4	5	6	7
Подготовка к текущему контролю		0				
Общая трудоемкость по дисциплине		**72**				

Примечание: Л – лекции, ПЗ – практические занятия/семинары, ЛР – лабораторные занятия, СРС – самостоятельная работа студента

2.3 Содержание разделов (тем) дисциплины

2.3.1 Занятия лекционного типа

№	Наименование раздела (темы)	Содержание раздела (темы)	Форма текущего контроля
1	2	3	4
1.	Введение в Микросервисную архитектуру. От Монолита к Микросервисам	Что такое монолитная архитектура? Её плюсы и минусы. Эволюция: Монолит -> Модульный монолит -> SOA -> Микросервисы. Определение микросервисной архитектуры (MSA). Сравнение "Монолит vs Микросервисы": сложность разработки, масштабируемость, надежность, скорость доставки. Организационный аспект: Команды по сервисам (Conway's Law). **Практический пример:** Разбор кейса компании (например, Netflix, Amazon), которая успешно перешла с монолита на микросервисы. **Домашнее задание:** Проанализировать существующее монолитное приложение и предложить, как его можно было бы декомпозировать на микросервисы.	ЛР
2.	Декомпозиция и Проектирование Сервисов. Антипаттерны	Принципы декомпозиции (Как "ломать" монолит?). Паттерн "Доменно-ориентированное проектирование" (Domain-Driven Design, DDD): - Ограниченные контексты (Bounded Contexts). - Сущности (Entities), Объекты-Значения (Value Objects), Агрегаты (Aggregates). Принцип единственной ответственности (Single Responsibility Principle) для сервисов. Типы связей между сервисами: слабая связанность, сильное сцепление. Антипаттерны: "Распределенный монолит". Практический пример: Декомпозиция домена "Интернет-магазин" (Заказы, Пользователи, Каталог, Доставка). Домашнее задание: Описать ограниченные контексты и агрегаты для выбранного домена (например, "Социальная сеть" или "Такси").	ЛР
3.	Взаимодействие между	Синхронное взаимодействие (REST, gRPC). - Плюсы и минусы.	ЛР

№	Наименование раздела (темы)	Содержание раздела (темы)	Форма текущего контроля
1	2	3	4
	Микросервисами. Синхронные и Асинхронные паттерны	- Проблемы каскадных отказов. Асинхронное взаимодействие (Message Brokers: RabbitMQ, Apache Kafka). Очереди сообщений и публикация/подписка (Pub/Sub). **Паттерны:** - API Gateway. - Сага (Saga) для управления распределенными транзакциями. - CQRS (Command Query Responsibility Segregation) и Event Sourcing. **Практический пример:** Реализация сценария "Оформление заказа" с использованием Saga и обменом сообщениями через брокер. **Домашнее задание:** Спроектировать схему взаимодействия сервисов для процесса "Регистрация пользователя" с использованием асинхронных сообщений.	
4.	Управление Данными в Распределенной Системе	Принцип "База данных на сервис" (Database per Service). - Проблемы согласованности данных (CAP-теорема, eventual consistency). - Способы обмена данными между сервисами: ▪ Асинхронные события. ▪ Дублирование данных и их синхронизация. - Паттерны: Materialized View для чтения данных из нескольких сервисов. **Практический пример:** Как сервис "Поиска" получает данные от сервиса "Каталога" через события. **Домашнее задание:** Описать, какие данные будут дублироваться в разных сервисах интернет-магазина и как будет поддерживаться их согласованность.	ЛР
5.	Обнаружение Сервисов, Конфигурирование и Безопасность	**Service Discovery:** Как сервисы находят друг друга в динамическом окружении? (Eureka, Consul, Nacos). - **Configuration Management:** Внешнее хранение конфигураций (Spring Cloud Config, Consul KV). - **Безопасность (Security):** Аутентификация и авторизация в распределенной системе.Паттерн API Gateway как единая точка входа. Токены доступа (JWT), OAuth 2.0. **Практический пример:** Настройка Service Registry и получение конфигурации из центрального хранилища. **Домашнее задание:** Спроектировать схему аутентификации для набора микросервисов.	ЛР
6.	Наблюдаемость (Observability) и Мониторинг	Чем observability отличается от мониторинга? Три столпа наблюдаемости:	ЛР

№	Наименование раздела (темы)	Содержание раздела (темы)	Форма текущего контроля
1	2	3	4
		1. **Логи (Logging):** Централизованный сбор логов (ELK Stack, Loki). 2. **Метрики (Metrics):** Сбор и визуализация метрик (Prometheus, Grafana). 3. **Трассировка (Tracing):** Отслеживание запроса по всем сервисам (Jaeger, Zipkin). - Сбор бизнес-метрик. - Создание дашбордов для отслеживания здоровья системы. **Практический пример:** Настройка трассировки для цепочки вызовов и просмотр трейса в Jaeger UI. **Домашнее задание:** Предложить ключевые метрики для мониторинга здоровья сервиса "Заказы"	
7.	Развертывание и оркестрация микросервисов	Проблемы ручного развертывания множества сервисов. - **Контейнеризация:** Docker как стандарт упаковки сервиса. - **Оркестрация:** Kubernetes как платформа для управления жизненным циклом контейнеров. Основные понятия: Pod, Deployment, Service, Ingress. - Стратегии развертывания: Blue-Green, Canary-релизы. - Инфраструктура как код (IaC): Terraform, Ansible. **Практический пример:** Написание Dockerfile и простого Helm-чарта для развертывания сервиса в Kubernetes. **Домашнее задание:** Описать манифест Kubernetes Deployment для типичного микросервиса	ЛР
8.	Тестирование и надежность микросервисных систем	Пирамида тестирования для микросервисов. - **Тестирование изолированного сервиса:** Модульные и интеграционные тесты. - **Тестирование взаимодействия:** Контрактное тестирование (Pact), тестирование на уровне сервиса. - **Тестирование всей системы:** E2E-тесты. - **Паттерны для обеспечения надежности:** Circuit Breaker (Предохранитель). Retry, Fallback, Bulkhead (Разделение отсеков). **Chaos Engineering:** Введение сбоев для проверки устойчивости системы. **Практический пример:** Написание контракта с помощью Pact и использование Circuit Breaker в коде. **Итоговое задание / Обзор:** Создание эскизного проекта микросервисной системы, включая декомпозицию, взаимодействие, данные и стратегию развертывания.	ЛР

Примечание: ЛР – отчет/защита лабораторной работы, КП - выполнение курсового проекта, КР - курсовой работы, РГЗ - расчетно-графического задания, Р - написание реферата, Э - эссе, К - коллоквиум, Т – тестирование, РЗ – решение задач.

2.3.2 Занятия семинарского типа

Не предусмотрены

2.3.3 Лабораторные занятия

Лабораторные занятие состоят в выполнении индивидуальных заданий и объединены в 16 заданий

№	Наименование раздела (темы)	Наименование лабораторных работ	Форма текущего контроля
1	2	3	4
1.	Введение в микросервисную архитектуру. От монолита к микросервисам	**Введение в проектирование веб-серверов** на примере ASP.NET Core 8/9 + Postgres Практикум:Задачи на создание и поднятие приложения ASP.NET Core + Postgres в docker envc N-м набором эндпоинтов. Выбрать предметную область проекта, который будет реализовываться группой студентов до конца семестра, спроектировать верхнеуровневую архитектуру будущего проекта. Описать домен.	ЛР
2.	Декомпозиция и проектирование сервисов. Антипаттерны	Декомпозиция и проектирование Сервисов. Выполнить декомпозицию сервисов согласно своему домену. RabbitMQ: level 0 Практикум: Поднимаем в докере RabbitMq + RabbitMq.AdminPanel, подключаем к проекту, публикуем первое сообщение в очередь. Подключаем консьюмера с автокоммитом (auto ack).	ЛР
3.	Взаимодействие между микросервисами. синхронные и асинхронные паттерны	**Освоить паттерны асинхронного взаимодействия через messaging.** Задачи: - Настройка RabbitMQ/Kafka - Реализация событий "OrderCreated", "PaymentProcessed" - Создание сервиса "Уведомления" как consumer событий Результат: Асинхронная обработка заказов через события.	ЛР

№	Наименование раздела (темы)	Наименование лабораторных работ	Форма текущего контроля
1	2	3	4
4	Взаимодействие между микросервисами. синхронные и асинхронные паттерны	**Масштабирование приложений** RabbitMQ: level 1 Практикум: Консьюмер с обычным коммитом, обработка ошибок/ Поднимаем несколько автоматических продьюсеров, замедляем консьюмеров. Смотрим на захлёбывание приложения - скейлим приложение.	ЛР
5.	Управление данными в распределенной системе	**Изучение настроек очередей** RabbitMQ: level 2 Практикум: играемся с настройками очередей, пробуем dead letter queue,делаем эндпоинт для отсылки реальных электронных писем друг другу (yandex/gmail)	ЛР
6.	Обнаружение сервисов, конфигурирование и безопасность	Настроить динамическое обнаружение сервисов и централизованное управление конфигурацией. **Задачи:** - Настройка Eureka Server (Service Discovery) - Регистрация всех сервисов в Eureka - Настройка Spring Cloud Config Server **Результат:** Динамическое обнаружение сервисов и внешние конфигурации.	
7.	Управление данными в распределенной системе	Apache Kafka: level 0 Практикум: Поднимаем в докере Kafka + Kafka UI, подключаем к проекту, публикуем первое сообщение в партицию, подключаем консьюмера	ЛР
8.	Взаимодействие между микросервисами. синхронные и асинхронные паттерны	Apache Kafka: level 1 Практикум: Играемся с настройками продьюсера, подключаем консьюмер-группу, балансируем	ЛР
9.	Наблюдаемость (Observability) и Мониторинг	Apache Kafka: level 2 Практикум: Играемся с настройками топиков, пишем батчевый консьюмер для увеличения пропускной способности	ЛР
10.	Обнаружение сервисов, конфигурирование и безопасность	Apache Kafka: level 3 Практикум: Пишем ретрайер, далее делаем dead letter queue руками, поднимаем новую консьюмер-группу для обработки событий из dead letter queue	ЛР
11.	Обнаружение сервисов, конфигурирование и безопасность	**Централизованная конфигурация** **Цель:** Вынести настройки из кода сервисов в центральное хранилище. **Задание:** 1. Создать сервис **Config-Server**.	ЛР

№	Наименование раздела (темы)	Наименование лабораторных работ	Форма текущего контроля
1	2	3	4
		2. Подключить его к Git-репозиторию с .yml/.properties файлами конфигурации. 3. Переделать Catalog-Service и Order-Service так, чтобы они брали конфигурацию с Config-Server при старте. **Результат:** Управление настройками всех сервисов из одного места.	
12.	Развертывание и оркестрация микросервисов	**Контейнеризация сервисов с помощью Docker** **Цель:** Упаковать каждый сервис в независимый контейнер. **Задание:** 1. Написать Dockerfile для каждого сервиса (Catalog, Order, Gateway и т.д.). 2. Собрать Docker-образы для всех сервисов. 3. Запустить систему вручную, запуская каждый контейнер с помощью docker run. **Результат:** Все сервисы запущены как Docker-контейнеры.	
13.	Развертывание и оркестрация микросервисов	**Оркестрация с помощью Docker Compose** **Цель:** Автоматизировать запуск всей системы одним командой. **Задание:** 1. Написать docker-compose.yml файл, который описывает все сервисы, их зависимости (например, order-service зависит от kafka и eureka), сети и переменные окружения. 2. Запустить всю инфраструктуру (БД, Kafka, Eureka, сервисы) одной командой docker-compose up. **Результат:** Полностью работающая микросервисная система, поднимаемая одной командой	
14.	Тестирование и надежность микросервисных систем Наблюдаемость (Observability) и Мониторинг	**Распределенная трассировка (Distributed Tracing)** **Цель:** Научиться отслеживать запрос, проходящий через несколько сервисов. **Задание:** 1. Интегрировать в сервисы (API-Gateway, Catalog, Order) библиотеку для трассировки (например, Spring Cloud Sleuth + Zipkin).	ЛР

№	Наименование раздела (темы)	Наименование лабораторных работ	Форма текущего контроля
1	2	3	4
		2. Настроить экспорт трасс в Zipkin. 3. Создать заказ через Gateway и найти полный trace этого запроса в UI Zipkin. **Результат:** Возможность визуализации пути запроса и анализа задержек.	
15.	Тестирование и надежность микросервисных систем	**Мониторинг и сбор логов (Prometheus + Grafana)** **Цель:** Настроить мониторинг метрик и визуализацию здоровья системы. **Задание:** 1. Интегрировать в сервисы метрики для Prometheus (используя Spring Boot Actuator и Micrometer). 2. Настроить Prometheus для сбора этих метрик. 3. В Grafana создать дашборд, отображающий: количество запросов, ошибки, задержки, статус Circuit Breaker'ов. **Результат:** Дашборд Grafana, показывающий ключевые метрики работы системы.	ЛР
16.	Подведение итогов курса	Разбор вопросов и задач	обсуждение

Примечание: ЛР – отчет/защита лабораторной работы, КП - выполнение курсового проекта, КР - курсовой работы, РГЗ - расчетно-графического задания, Р - написание реферата, Э - эссе, К - коллоквиум, Т – тестирование, РЗ – решение задач.

2.3.4 Примерная тематика курсовых работ (проектов)

Курсовая работа не предусмотрена. В качестве курсового проекта студенты защищают инфраструктуру проекта веб-приложения с использованием ML по заданию от индустриального партнера.

2.4 Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)

№	Вид СРС	Перечень учебно-методического обеспечения дисциплины по выполнению самостоятельной работы
1	2	3
1	Изучение теоретического материала	Методические указания по выполнению самостоятельной работы, утвержденные на заседании кафедры вычислительных технологий факультета компьютерных технологий и прикладной математики ФГБОУ ВО «КубГУ», протокол №7 от

		07.05.2025г. Источники основной и дополнительной литературы.
	Решение задач	Методические указания по выполнению самостоятельной работы, утвержденные на заседании кафедры вычислительных технологий факультета компьютерных технологий и прикладной математики ФГБОУ ВО «КубГУ», протокол №7 от 07.05.2025г. Источники основной и дополнительной литературы.

Учебно-методические материалы для самостоятельной работы обучающихся из числа инвалидов и лиц с ограниченными возможностями здоровья (ОВЗ) предоставляются в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

– в печатной форме увеличенным шрифтом,

– в форме электронного документа,

– в форме аудиофайла,

– в печатной форме на языке Брайля.

Для лиц с нарушениями слуха:

– в печатной форме,

– в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

– в печатной форме,

– в форме электронного документа,

– в форме аудиофайла.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

3. Образовательные технологии

В соответствии с требованиями ФГОС в программа дисциплины предусматривает использование в учебном процессе следующих образовательные технологии: чтение лекций с использованием мультимедийных технологий; метод малых групп, разбор практических задач и кейсов.

При обучении используются следующие образовательные технологии:

- Технология коммуникативного обучения – направлена на формирование коммуникативной компетентности студентов, которая является базовой, необходимой для адаптации к современным условиям межкультурной коммуникации.

- Технология разноуровневого (дифференцированного) обучения – предполагает осуществление познавательной деятельности студентов с учётом их индивидуальных способностей, возможностей и интересов, поощряя их реализовывать свой творческий потенциал. Создание и использование диагностических тестов является неотъемлемой частью данной технологии.

- Технология модульного обучения – предусматривает деление содержания дисциплины на достаточно автономные разделы (модули), интегрированные в общий курс.

- Информационно-коммуникационные технологии (ИКТ) - расширяют рамки образовательного процесса, повышая его практическую направленность, способствуют интенсификации самостоятельной работы учащихся и повышению познавательной активности. В рамках ИКТ выделяются 2 вида технологий:

- Технология использования компьютерных программ – позволяет эффективно дополнить процесс обучения технологиям на всех уровнях.

- Интернет-технологии – предоставляют широкие возможности для поиска информации, разработки научных проектов, ведения научных исследований.

- Технология индивидуализации обучения – помогает реализовывать личностно-ориентированный подход, учитывая индивидуальные особенности и потребности учащихся.

- Проектная технология – ориентирована на моделирование социального взаимодействия учащихся с целью решения задачи, которая определяется в рамках профессиональной подготовки, выделяя ту или иную предметную область.

- Технология обучения в сотрудничестве – реализует идею взаимного обучения, осуществляя как индивидуальную, так и коллективную ответственность за решение учебных задач.

- Игровая технология – позволяет развивать навыки рассмотрения ряда возможных способов решения проблем, активизируя мышление студентов и раскрывая личностный потенциал каждого учащегося.

- Технология развития критического мышления – способствует формированию разносторонней личности, способной критически относиться к информации, умению отбирать информацию для решения поставленной задачи.

Комплексное использование в учебном процессе всех вышеназванных технологий стимулируют личностную, интеллектуальную активность, развивают познавательные процессы, способствуют формированию компетенций, которыми должен обладать будущий специалист.

Основные виды интерактивных образовательных технологий включают в себя:

- работа в малых группах (команде) - совместная деятельность студентов в группе под руководством лидера, направленная на решение общей задачи путём творческого сложения результатов индивидуальной работы членов команды с делением полномочий и ответственности;

- проектная технология - индивидуальная или коллективная деятельность по отбору, распределению и систематизации материала по определенной теме, в результате которой составляется проект;

- анализ конкретных ситуаций - анализ реальных проблемных ситуаций, имевших место в соответствующей области профессиональной деятельности, и поиск вариантов лучших решений;

- развитие критического мышления – образовательная деятельность, направленная на развитие у студентов разумного, рефлексивного мышления, способного выдвинуть новые идеи и увидеть новые возможности.

Подход разбора конкретных задач и ситуаций широко используется как преподавателем, так и студентами во время лекций, лабораторных занятий и анализа результатов самостоятельной работы. Это обусловлено тем, что при исследовании и решении каждой конкретной задачи имеется, как правило, несколько методов, а это требует разбора и оценки целой совокупности конкретных ситуаций.

Семестр	Вид занятия	Используемые интерактивные образовательные технологии	количество интерактивных часов
7	**Л**	Мультимедийные лекции – изложение теоретического материала, демонстрация примеров	**16**
7	ЛР	Практические занятия в режимах взаимодействия «преподаватель – студент» и «студент – студент»	34
Итого			**50**

Примечание: Л – лекции, ПЗ – практические занятия/семинары, ЛР – лабораторные занятия, СРС – самостоятельная работа студента

Темы, задания и вопросы для самостоятельной работы призваны сформировать навыки поиска информации, умения самостоятельно расширять и углублять знания, полученные в ходе лекционных и практических занятий.

Подход разбора конкретных ситуаций широко используется как преподавателем, так и студентами при проведении анализа результатов самостоятельной работы.

Для лиц с ограниченными возможностями здоровья предусмотрена организация консультаций с использованием электронной почты.

Для лиц с нарушениями зрения:

– в печатной форме увеличенным шрифтом,

– в форме электронного документа.

Для лиц с нарушениями слуха:

– в печатной форме,

– в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

– в печатной форме,

– в форме электронного документа.

Для лиц с ограниченными возможностями здоровья предусмотрена организация консультаций с использованием электронной почты.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

4. Оценочные и методические материалы

4.1 Оценочные средства для текущего контроля успеваемости и промежуточной аттестации

Оценочные средства предназначены для контроля и оценки образовательных достижений обучающихся, освоивших программу учебной дисциплины «название дисциплины».

Оценочные средства включает контрольные материалы для проведения **текущего контроля** в форме оценки лабораторных работ и проекта к **зачету**.

Оценочные средства для инвалидов и лиц с ограниченными возможностями здоровья выбираются с учетом их индивидуальных психофизических особенностей.

– при необходимости инвалидам и лицам с ограниченными возможностями здоровья предоставляется дополнительное время для подготовки ответа на экзамене;

– при проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья предусматривается использование технических средств, необходимых им в связи с их индивидуальными особенностями;

– при необходимости для обучающихся с ограниченными возможностями здоровья и инвалидов процедура оценивания результатов обучения по дисциплине может проводиться в несколько этапов.

Процедура оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья по дисциплине (модулю) предусматривает предоставление информации в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

– в печатной форме увеличенным шрифтом,

– в форме электронного документа.

Для лиц с нарушениями слуха:

– в печатной форме,

– в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

– в печатной форме,

– в форме электронного документа.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

Структура оценочных средств для текущей и промежуточной аттестации

ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2

№ п/п	Контролируемые разделы (темы) дисциплины*	Код контролируемой компетенции (или ее части)	Наименование оценочного средства	
			Текущий контроль	Промежуточная аттестация
	Введение в Микросервисную архитектуру. От Монолита к микросервисам	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №1*	*Вопросы к зачету*
	Декомпозиция и проектирование сервисов. Антипаттерны	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №2*	*Вопросы к зачету*
	Взаимодействие между микросервисами. Синхронные и асинхронные паттерны	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №3,4,8*	*Вопросы к зачету*
	Управление данными в распределенной системе	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №5,7*	*Вопросы к зачету*
	Обнаружение сервисов, конфигурирование и безопасность	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №6,10,11*	*Вопросы к зачету*
	Наблюдаемость (Observability) и мониторинг	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №9,14*	*Вопросы к зачету*

	Развертывание и оркестрация микросервисов	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №12,13*	*Вопросы к зачету*
	Тестирование и надежность микросервисных систем	ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2	*Лабораторная работа №14,15*	*Вопросы к зачету*

Показатели, критерии и шкала оценки сформированных компетенций

Соответствие **пороговому уровню** освоения компетенций планируемым результатам обучения и критериям их оценивания (оценка: **зачтено**):

ОПК-3 ***Способен к разработке алгоритмических и программных решений в области системного и прикладного программирования, математических, информационных и имитационных моделей, созданию информационных ресурсов глобальных сетей, образовательного контента, прикладных баз данных, тестов и средств тестирования систем и средств на соответствие стандартам и исходным требованиям***

ОПК-3.1 Аргументировано применяет методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности

Знает принципы микросервисной архитектуры (MSA) и их отличия от монолитной и сервис-ориентированной архитектур (SOA), ключевые паттерны проектирования микросервисов (например, Server Discovery, API Gateway, Circuit Breaker, Saga, CQRS), знает методы декомпозиции предметной области (Domain-Driven Design, Subdomain, Bounded Context) для перехода от монолита к микросервисам.

Знает принципы и протоколы межсервисного взаимодействия (REST/gRPC, Messaging/Message Brokers), стратегии и инструменты для непрерывной интеграции и непрерывного развертывания (CI/CD) в контексте MSA.

Умеет применить эти знания при разработке программных продуктов и программных комплексов в различных областях человеческой деятельности.

Владеет навыком разработки и реализации отдельного микросервиса на выбранном технологическом стеке (например, Spring Boot, Node.js, .NET), настройки межсервисного взаимодействия с использованием REST API, gRPC или брокеров сообщений (RabbitMQ, Kafka), навыком контейнеризации микросервисов с использованием Docker и описания многоконтейнерных приложений с помощью Docker Compose, навыком работы с инструментами для регистрации и агрегации логов, мониторинга и трассировки запросов в распределенной системе.

ОПК-3.2 Использует инструментальные, программные и аппаратные средства измерений для оценки качества программного обеспечения

Знает метрики и критерии качества для микросервисных систем (надежность, отказоустойчивость, масштабируемость, время отклика), инструменты и методы для мониторинга состояния микросервисов (Prometheus, Grafana) и трассировки распределенных запросов (Jaeger, Zipkin). Знает подходы к обеспечению отказоустойчивости (Resilience) и методы тестирования на устойчивость (Chaos Engineering).

Умеет выбирать и настраивать инструменты мониторинга для сбора ключевых метрик (метрики приложения, системные метрики, метрики бизнес-логики), анализировать данные мониторинга и трассировки для выявления "узких мест" (bottlenecks) и ошибок в работе распределенной системы.Уметь оценивать качество программного кода микросервисов с использованием статических анализаторов кода (SonarQube).
Владеет навыками настройки панелей мониторинга (Dashboards) в Grafana для визуализации состояния системы, написания и выполнения контрактных тестов (Pact, Spring Cloud Contract) для проверки взаимодействия между микросервисами, навыками использования инструментов для трассировки (Jaeger) для отслеживания пути запроса через несколько сервисов, навыками проведения нагрузочного тестирования отдельных сервисов и системы в целом с использованием инструментов (например, JMeter).

ОПК-4 ***Способен участвовать в разработке технической документации программных продуктов и комплексов с использованием стандартов, норм и правил, а также в управлении проектами создания информационных систем на стадиях жизненного цикла***

ОПК-4.4 Имеет практический опыт анализа и интерпретации информационных систем

Знает *правила* анализа и интерпретации информационных систем

Умеет аргументировано применять методы проектирования, разработки и реализации программных продуктов и программных комплексов в различных областях человеческой деятельности. **Владеет** навыками использования инструментальных, программных и аппаратных средств измерений для оценки качества программного обеспечения

ОПК-5 ***Способен инсталлировать и сопровождать программное обеспечение информационных систем и баз данных, в том числе отечественного происхождения, с учетом информационной безопасности***

ОПК-5.3 Имеет практические навыки установки и инсталляции программных комплексов, применения основ сетевых технологий.

Знает принципы разработки и функционирования ИС и БД.
Умеет документировать и сопровождать ИС и БД.
Владеет практическими навыками установки и инсталляции программных комплексов, применения основ сетевых технологий.

ОПК-6 ***Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности***

ОПК-6.1 Аргументировано применяет современные информационные технологии, в том числе отечественные, при создании программных продуктов и программных комплексов различного назначения.

Знает: принципы микросервисной архитектуры (MSA): слабая связанность, сильная когерентность, независимое развертывание. • Ключевые технологические паттерны MSA: API Gateway, Service Discovery, Circuit Breaker, внешний конфигурационный сервер.• Современные протоколы взаимодействия сервисов (REST/gRPC, асинхронная коммуникация через брокеры сообщений - RabbitMQ, Kafka).

Умеет: Выбирать технологический стек (язык, фреймворк, БД) для каждого сервиса исходя из его задач, а не из единого стандарта.• Реализовывать сервисы на современных фреймворках (например, Spring Boot, FastAPI,) и контейнеризировать их с помощью Docker.• Настраивать межсервисное взаимодействие, выбирая тип коммуникации (синхронный/асинхронный) в зависимости от требований к согласованности и производительности.

Владеет: **Навыками** проектирования и развертывания отказоустойчивых микросервисных систем с использованием оркестратора (Kubernetes). • **Методами** обоснования выбора технологий (включая отечественные) для конкретного проекта, оценки компромиссов (производительность/сложность).

ОПК-6.2 Ориентируется в современных положениях и концепциях прикладного и системного программного обеспечения, архитектуры компьютеров и сетей (в том числе и глобальных), технологии создания и сопровождения программных продуктов и программных комплексов

Знает: Принципы работы сетей: модель OSI/TCP-IP, DNS, HTTP/• Принципы мониторинга и observability в распределенных системах (логи, метрики, трейсинг - ELK Stack, Prometheus, Grafana, Zipkin).

Умеет: Проектировать сетевое взаимодействие между сервисами, понимая последствия выбора портов, протоколов и политик безопасности. • Настраивать пайплайн CI/CD для отдельного микросервиса, обеспечивая его автоматизированное тестирование и развертывание. • Настраивать централизованный сбор логов и мониторинг ключевых метрик (health, latency, traffic) для группы сервисов.• Анализировать проблемы в распределенной системе, используя логи и трейсы для идентификации "больного" сервиса.

Владеет: **Методами** диагностики и решения проблем, характерных для распределенных систем (сетевые задержки, частичные отказы, проблемы согласованности данных).**Системным мышлением** для понимания полного жизненного цикла микросервисного приложения — от разработки и до эксплуатации и масштабирования в глобальной сети.

LC-3 ***Способен проектировать и поддерживать архитектуру систем искусственного интеллекта***

LC-3.1 Создает и развивает архитектуры системы ИИ на всех этапах жизненного цикла

Знает архитектуры системы ИИ.
Умеет определять архитектуру ИИ, подходы к проектированию, выбор подходящего стека технологий ИИ для RnD и промышленной разработки.
Владеет способностью применять различные принципы и паттерны при проектировании архитектуры систем ИИ.

PL-1 ***Способен применять язык программирования Python для решения задач в области ИИ***

PL-1.1 Разрабатывает и отлаживает прикладные решения разной сложности и для разного круга конечных пользователей с использованием языка программирования Python, тестирует, испытывает и оценивает качество таких решений.
Знает основы синтаксиса языка, пишет небольшие скрипты для автоматизации ручной работы по обработке небольших объемов данных с помощью встроенных модулей и внешних библиотек (csv, json, requests).

Владеет основными библиотеками для выполнения большинства рутинных задач в крупных проектах: ввод-вывод, серверное программирование (FastAPI, Flask, Django REST Framework), применение многопоточности (модуль threading). **Умеет** разрабатывать серверные приложения и поддерживать их. Использует особенности виртуальной машины Python (например, GIL), разрабатывает библиотечный код общего пользования, а также документацию к нему. Профилирует и оптимизирует приложения на Python, используя встроенные инструменты (например, cPython).

PL-1.2 Осуществляет выбор инструментов разработки на Python, приемлемых для создания прикладной системы обработки научных данных, машинного обучения и визуализации с заданными требованиями

Знает и применяет основные библиотеки для научных вычислений, такие как NumPy, SciPy и Pandas. **Знает** и применяет основные библиотеки для визуализации данных, например, Matplotlib и Seaborn

Умеет разрабатывать собственные компоненты для библиотек машинного обучения с учётом интеграции с ними.

Владеет приемами оптимизации кода с использованием библиотек для научных вычислений. **Владеет** навыками применения библиотек машинного обучения, в том числе глубокого обучения, такие как scikit-learn, PyTorch и TensorFlow

PL-1.3 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности

Знает принципы работы систем обработки больших данных различной степени сложности.

Умеет разрабатывать и поддерживать простейшие ETL-скрипты в пайплайнах обработки данных, применять основные функции фреймворка PySpark, может самостоятельно построить процесс обработки больших данных с использованием Airflow. **Владеет** инструментами профилирования и оптимизации ETL процессы для обработки больших данных в рамках Spark/Mapreduce фреймворка. Самостоятельно поддерживает инфрастуктуру обработки больших данных.

PL-2 Способен применять JVM-совместимые языки программирования для решения задач в области ИИ

PL-2.1 Разрабатывает и отлаживает прикладные решения разного уровня сложности и для широкого круга конечных пользователей с использованием JVM-совместимых языков программирования, тестирует, испытывает и оценивает качество таких решений.

Знает и понимает модель памяти Java и способен поддерживать приложения с высоким параллелизмом и конкуренцией. Понимает алгоритмы сборки мусора и способен оптимизировать сборку мусора. **Умеет** применять основные библиотеки для решения рутинных задач в серверном программировании: ввод-вывод. **Владеет** навыками применения простейших примитивов многопоточного программирования, интеграция с базами данных.

PL-2.2 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности.

Знает принципы работы систем обработки больших данных различной степени сложности.

Умеет разрабатывать и поддерживать простые ETL алгоритмы в пайплайнах обработки данных. Эффективно применяет фреймворки для пакетной обработки

данных (Spark, Mapreduce) и адаптирует алгоритмы для вычислений на малых объемах данных под большие данные. **Владеет** инструментами организации потоковых вычислений (Kafka, Flink) и интеграции с аналитическими БД (ClickHouse, Greenplum)

Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

Примеры лабораторных работ
кратко описаны в разделе 2.3.3

Приблизительные темы на индивидуальные проекты (можно предложить свою)

1. Система управления заказами - Ozon
2. Логистика и ПВЗ - Wildberries
3. Управление банковскими транзакцями - Сбербанк
4. Управление базой авиабилетов - Aviasales
5. Билетная система и снабжение поездов - РЖД
6. Кэшбэк на покупки - АльфаБанк
7. Кредитная система - Тинкофф-Банк
8. Бронирование отелей - Яндекс Путешествия
9. Учет покупок в супермаректах - Тандер
10. Ценообразование на полках - X5 Group
11. Учет топлива в сети заправок - ЛукОйл
12. Газообеспечение ИЖС и учет - Газпром
13. Управление электросетями - Россети
14. Телефония(звонки/sms/мобильный интернет) - Билайн
15. Складкской учет и мерчендайзинг - Metro C&C
16. Управление почтовой логистикой - Почта России
17. Билетная система проездных - Московский Метрополитен
18. Система учета показаний счетчиков МКД - ГИС ЖКХ
19. Агрегатор такси: матчинг водителей и клиентов - Яндекс Такси
20. Социальная сеть: модуль друзей и сообщений - VK
21. Аналитика и активность видеохостинга - Rutube
22. Сервис по поиску работы: резюме, вакансии, отклики - HeadHunter
23. Автоматизированния система орошения полей - РусАгро
24. Быстрая доставка еды и теневые склады - Яндекс Лавка
25. Управление базой недвижимости - Циан
26. Управление базой автомобилей - АвтоРУ
27. Цифровое распространение компьютерных игр - VK Play
28. Отслеживание умных платежей цифрового рубля - ЦБ РФ
29. Обменный курс цифровой валюты на реальную - Криптобиржа Bybit
30. Отслеживание индекса потребительских цен в режиме реального времени - РоссСтат

Зачетно-экзаменационные материалы для промежуточной аттестации
Примеры вопросов для подготовки к зачету

1. Различие между стандартными синхронными вызовами и асинхронным межсервисным взаимодействием.
2. Синхронные протоколы: SOAP, WSDL, RPC, HTTP1.1/REST, HTTP2/GRPC
3. Асинхронные протоколы: TCP/AMQP/MQTT
4. RabbitMQ История, для чего придумано
5 Очереди, обменники, пуш модель, гарантии доставки, библиотека RabbitMq.Client
6 Роутинг, балансировка, приоритезация событий
7 ack/nack/reject событий, повторная обработка
8 ttl, max queue length, dead letter queue
9 Apache Kafka: История, для чего придумано, сравнение с RabbitMQ
10 Модель с внешним координатором, новая модель с распределенным консенсусом
11 Топики, партиции, продьюсеры, консьюмеры, оффсеты
12 Библиотека от Confluent, разбор API, пример подключения к проекту
13 Распределение событий по партициям, коммиты/автокоммиты, консьюмергруппы, ребалансировка
14 Гибкие настройки партиций: retention, segment, cleanup-policy
15 Ошибки при обработке, повторная обработка событий, dead letter queue

Перечень компетенций (части компетенции), проверяемых оценочным средством - ОПК-3; ОПК-4; ОПК-5; ОПК-6; LC-3; PL-1; PL-2

4.2 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

Методические рекомендации, определяющие процедуры оценивания на зачете:

Процедура промежуточной аттестации проходит в соответствии с Положением о текущем контроле и промежуточной аттестации обучающихся ФГБОУ ВО «КубГУ».

Итоговой формой контроля сформированности компетенций у обучающихся по дисциплине является зачет. Студенты обязаны сдать зачет в соответствии с расписанием и учебным планом.

ФОС промежуточной аттестации состоит из вопросов к зачету и результатов текущего контроля.

Зачет по дисциплине преследует цель оценить работу студента за курс, получение теоретических знаний, их прочность, развитие творческого мышления, приобретение навыков самостоятельной работы, умение применять полученные знания для решения практических задач.

Форма проведения зачета: устно.

Результат сдачи зачета заноситься преподавателем в зачетную ведомость и зачетную книжку.

Оценивание уровня освоения дисциплины основывается на качестве выполнения студентом заданий текущего контроля и ответов на вопросы зачета.

Критерии оценки:

1. Оценка ответов на вопросы к зачету (50% итоговой оценки)

Зачет

Полные, развернутые ответы с демонстрацией глубокого понимания темы.
Ответы содержат основные идеи, но без углубленного анализа.
Использование примеров, формул, корректных терминов.
Возможны небольшие ошибки в деталях или формулировках.
Умение анализировать и сравнивать методы.
% выполнения: 60–100% (допускаются незначительные неточности).

Незачет

Отсутствие понимания ключевых концепций.
Грубые ошибки или неспособность ответить на большую часть вопросов.
% выполнения: <60%.

2. Оценка выполнения практических кейсов и лабораторных работ (50% итоговой оценки)

Зачет

Полное выполнение всех этапов кейса с инновационными решениями.
Четкая документация кода и анализ результатов.
% выполнения: 60–100%.

или

Выполнены основные задачи, но без дополнительной оптимизации.

или

Решены базовые задачи, но с критическими ошибками.
Низкое качество кода или отсутствие анализа.

Незачет

Невыполнение ключевых этапов.
Код нерабочий или отсутствует.
% выполнения: <60%.

Методические рекомендации, определяющие процедуры оценивания лабораторных работ:

Процедура оценивания лабораторных работ проходит в соответствии с Положением о текущем контроле и промежуточной аттестации обучающихся ФГБОУ ВО «КубГУ».

По каждой лабораторной работе оформляется отчет. Отчеты сдаются на проверку руководителю в течение курса по мере их выполнения, и защищаются студентами в установленном порядке.

При защите отчета студенту могут быть заданы вопросы и дополнительные задания по сути лабораторной работы, в том числе из списка контрольных вопросов к данной лабораторной работе. При неудовлетворительной оценке знаний студента по теме данного отчета, студент возвращается к повторному изучению соответствующих материалов, после чего допускается к повторной защите. Неудовлетворительно выполненный отчет также возвращается на доработку.

Отчет должен содержать заголовок, тему лабораторной работы, цель, задание, индивидуальную тему, описание хода выполнения работы, необходимые прикладные материалы (схемы, макеты документов и т.п.), в соответствии с требованиями к содержанию, и выводы по работе.

Оценочные средства для инвалидов и лиц с ограниченными возможностями здоровья выбираются с учетом их индивидуальных психофизических особенностей.

– при необходимости инвалидам и лицам с ограниченными возможностями здоровья предоставляется дополнительное время для подготовки ответа на зачете;

– при проведении процедуры оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья предусматривается использование технических средств, необходимых им в связи с их индивидуальными особенностями;

– при необходимости для обучающихся с ограниченными возможностями здоровья и инвалидов процедура оценивания результатов обучения по дисциплине может проводиться в несколько этапов.

Процедура оценивания результатов обучения инвалидов и лиц с ограниченными возможностями здоровья по дисциплине предусматривает предоставление информации в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

– в печатной форме увеличенным шрифтом,

– в форме электронного документа.

Для лиц с нарушениями слуха:

– в печатной форме,

– в форме электронного документа.

Для лиц с нарушениями опорно-двигательного аппарата:

– в печатной форме,

– в форме электронного документа.

Данный перечень может быть конкретизирован в зависимости от контингента обучающихся.

4.3. Методические указания по организации вычислительной инфраструктуры

Требования к аппаратному и программному обеспечению рабочих мест

Аппаратные требования.

Для выполнения лабораторных работ по дисциплине «Мкросервисная архитектура» студентам и преподавателю необходим стационарный компьютер или ноутбук с современной конфигурацией. Рекомендуется многопроцессорный CPU, например Intel Core i3/i5/i7 не ниже 4-го поколения или аналогичный AMD Ryzen, с поддержкой многопоточности и оперативной памятью не менее 8 ГБ. Важно: Система должна быть 64-битной. Жесткий диск (Storage): Быстрый SSD-накопитель: Минимум: 256 ГБ свободного пространства. Образы контейнеров и виртуальные машины занимают значительный объем. Компьютер должен иметь стабильное подключение к сети Интернет со скоростью не ниже 5–10 Мбит/с для скачивания SDK, библиотек и обновлений программного обеспечения.

Программные требования

Система виртуализации: Docker Desktop (для Windows/macOS) или Docker Engine + Docker Compose (для Linux).

- Для Windows/macOS: Обязательно включить в настройках Docker Desktop использование WSL 2 (на Windows) для повышения производительности.
- Alternativa/VirtualBox или VMware Workstation Player (могут потребоваться для специфичных заданий).

Интегрированная среда разработки (IDE): На выбор:

- Visual Studio Code (рекомендуется) с расширениями: Docker, Dev Containers, Kubernetes, Python/Java/Go (в зависимости от стека технологий).
- IntelliJ IDEA Ultimate (для Java/Spring) или PyCharm Professional (для Python).
- Lite-версии: Sublime Text, Notepad++.

Система контроля версий: Git. Клиент должен быть установлен и доступен из командной строки. Git GUI-клиент (GitKraken, SourceTree, Fork).

Командная строка/Терминал:

- Windows: PowerShell (рекомендуется) или Command Prompt. Желательно: Установить Windows Terminal и WSL 2 (Windows Subsystem for Linux) с дистрибутивом Ubuntu.
- macOS/Linux: Стандартный терминал (Bash, Zsh).

Специализированное ПО для микросервисов:

Orchestrator (на выбор, в зависимости от учебного плана):Kubernetes: Локальная установка: Minikube (рекомендуется для начала) или kind (Kubernetes in Docker) либо в составе Docker.

Инструменты для работы с API:

Postman или **Insomnia** — для тестирования REST/gRPC API микросервисов.

Языки программирования и среды выполнения (в зависимости от выбранного стека):

- **Java:** JDK 11, 17 или 21 (рекомендуется LTS-версия, например, от OpenJDK).
- **Python:** Python 3.8+ (рекомендуется 3.10+). Менеджер пакетов **pip**.
- **Node.js:** Node.js 16+ (LTS). Менеджер пакетов **npm** или **yarn**.
- **Go:** Go 1.19+.
- **.NET:** .NET 6.0+ SDK.

Дополнительные рекомендации и замечания

- **Стабильное интернет-соединение:** Требуется для загрузки образов Docker, установки пакетов, работы с системами контроля версий.
- **Административные права:** На время установки ПО студентам могут потребоваться права администратора на своих рабочих машинах.
- **Виртуальная лаборатория (альтернатива):** Если невозможно обеспечить все локальные рабочие места требуемой мощностью, рекомендуется развернуть виртуальную лабораторную среду на серверах ВУЗа (например, на базе VMware vSphere или Proxmox VE), где студенты будут получать доступ к заранее подготовленным виртуальным машинам с установленным всем необходимым ПО.

5. Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля)

5.1. Основная литература

1. Долженко, А. И. Разработка и сопровождение программных систем. Технологии Microsoft.NET для разработки приложений : лабораторный практикум / А. И. Долженко, С. А. Глушенко. - Ростов-на-Дону : Издательско-полиграфический комплекс РГЭУ (РИНХ), 2019. - 140 с. - ISBN 978-5-7972-2626-0. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2211949 (дата обращения: 06.06.2025). – Режим доступа: по подписке.
2. Ананченко И.В., Зудилова Т.В., Иванов С.Е Контейнеризатор приложений docker — установка, настройка, основы управления приложениями в средах с поддержкой контейнеризации: учебно-методическое пособие. - Санкт-Петербург: Университет ИТМО, 2023. - 54 с.
3. Баркович, А. А. Веб-проектирование : учебное пособие / А.А. Баркович, Т.А. Филимонова. — Москва : ИНФРА-М, 2025. — 231 с. — (Высшее образование). — DOI 10.12737/2116156. - ISBN 978-5-16-019399-1. - Текст : электронный. - URL: https://znanium.ru/catalog/product/2116156 (дата обращения: 06.06.2025). – Режим доступа: по подписке..
4. Мол, Д. Создание облачных, мобильных и веб-приложений на F# : практическое руководство / Д. Мол ; пер. с англ. А. Н. Киселёва. - 2-е изд. - Москва : ДМК Пресс, 2023. - 209 с. - ISBN 978-5-89818-584-8. - Текст : электронный. - URL:

https://znanium.com/catalog/product/2107948 (дата обращения: 06.06.2025). – Режим доступа: по подписке.
5. Скотт, Д. Kafka в действии : практическое руководство / Д. Скотт, В. Гамов, Д. Клейн ; пер. с англ. А. Н. Киселева. - Москва : ДМК Пресс, 2022. - 310 с. - ISBN 978-5-93700-118-4. - Текст : электронный. - URL: https://znanium.com/catalog/product/2109488 (дата обращения: 06.06.2025). – Режим доступа: по подписке.

5.2 Дополнительная литература:

1. Бабушкина, Ирина Анатольевна. Практикум по объектно-ориентированному программированию / И. Бабушкина, С. Окулов. - 2-е изд. - М. : БИНОМ. Лаборатория знаний, 2009. - 366 с. : ил. - Библиогр. : с. 358. - ISBN 9785996302192 : 189.75..
2. Орлов, С. А. Технологии разработки программного обеспечения [Текст] : учебник С.А. Орлов. - СПб. : ПИТЕР, 2002. - 463с. - (Учебник для вузов). - Библиогр.:с.454-457 . - Алф. указ.: с. 458-463. (37 экз. в библиотеке КубГУ).
3. Громов Ю.Ю. , Иванова О.Г. , Беляев М.П. , Минин Ю.В. Технология программирования. - Министерство образования и науки Российской Федерации, Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
4. «Тамбовский государственный технический университет». - Тамбов : Издательство ФГБОУ ВПО «ТГТУ», 2013. - 173 с. [Электронный ресурс]. - URL: http://biblioclub.ru/index.php?page=book&id=277802.
5. Хорев П.Б. Технологии объектно-ориентированного программирования: Учебное пособие для студентов вузов. / П.Б. Хорев. – М.: Академия, 2004. – 448с. (51 экз. в библиотеке КубГУ)

5.3. Интернет-ресурсы, в том числе современные профессиональные базы данных, информационные справочные системы и конференции

Конференции А*:

1. https://openreview.net/forum?id=FMMF1a9ifL
2. https://openreview.net/forum?id=ElUrNM9U8c#discussion
3. https://openreview.net/forum?id=JoO6mtCLHD
4. https://aclanthology.org/2024.findings-emnlp.760/
5. https://aclanthology.org/2020.coling-main.588/
6. https://link.springer.com/chapter/10.1007/978-3-030-72113-8_30
7. https://link.springer.com/chapter/10.1007/978-3-031-42448-9_10
8. https://aclanthology.org/2024.findings-naacl.288/

Электронно-библиотечные системы (ЭБС):

1. ЭБС «ЮРАЙТ» https://urait.ru/
2. ЭБС «УНИВЕРСИТЕТСКАЯ БИБЛИОТЕКА ОНЛАЙН» http://www.biblioclub.ru/
3. ЭБС «BOOK.ru» https://www.book.ru
4. ЭБС «ZNANIUM.COM» www.znanium.com
5. ЭБС «ЛАНЬ» https://e.lanbook.com

Профессиональные базы данных

1. Scopus http://www.scopus.com/

2. ScienceDirect https://www.sciencedirect.com/
3. Журналы издательства Wiley https://onlinelibrary.wiley.com/
4. Научная электронная библиотека (НЭБ) http://www.elibrary.ru/
5. Полнотекстовые архивы ведущих западных научных журналов на Российской платформе научных журналов НЭИКОН http://archive.neicon.ru
6. Национальная электронная библиотека (доступ к Электронной библиотеке диссертаций Российской государственной библиотеки (РГБ) https://rusneb.ru/
7. Президентская библиотека им. Б.Н. Ельцина https://www.prlib.ru/
8. База данных CSD Кембриджского центра кристаллографических данных (CCDC) https://www.ccdc.cam.ac.uk/structures/
9. Springer Journals: https://link.springer.com/
10. Springer Journals Archive: https://link.springer.com/
11. Nature Journals: https://www.nature.com/
12. Springer Nature Protocols and Methods: https://experiments.springernature.com/sources/springer-protocols
13. Springer Materials: http://materials.springer.com/
14. Nano Database: https://nano.nature.com/
15. Springer eBooks (i.e. 2020 eBook collections): https://link.springer.com/
16. "Лекториум ТВ" http://www.lektorium.tv/
17. Университетская информационная система РОССИЯ http://uisrussia.msu.ru

Информационные справочные системы

1. **Консультант Плюс** - справочная правовая система (доступ по локальной сети с компьютеров библиотеки)

Ресурсы свободного доступа

1. КиберЛенинка http://cyberleninka.ru/;
2. Американская патентная база данных http://www.uspto.gov/patft/
3. Министерство науки и высшего образования Российской Федерации https://www.minobrnauki.gov.ru/;
4. Федеральный портал "Российское образование" http://www.edu.ru/;
5. Информационная система "Единое окно доступа к образовательным ресурсам" http://window.edu.ru/;
6. Единая коллекция цифровых образовательных ресурсов http://school-collection.edu.ru/;
7. Проект Государственного института русского языка имени А.С. Пушкина "Образование на русском" https://pushkininstitute.ru/;
8. Справочно-информационный портал "Русский язык" http://gramota.ru/;
9. Служба тематических толковых словарей http://www.glossary.ru/;
10. Словари и энциклопедии http://dic.academic.ru/;
11. Образовательный портал "Учеба" http://www.ucheba.com/;
12. Законопроект "Об образовании в Российской Федерации". Вопросы и ответы http://xn--273--84d1f.xn--p1ai/voprosy_i_otvety.

Собственные электронные образовательные и информационные ресурсы КубГУ

1. Электронный каталог Научной библиотеки КубГУ http://megapro.kubsu.ru/MegaPro/Web
2. Электронная библиотека трудов ученых КубГУ http://megapro.kubsu.ru/MegaPro/UserEntry?Action=ToDb&idb=6
3. Среда модульного динамического обучения http://moodle.kubsu.ru
4. База учебных планов, учебно-методических комплексов, публикаций и конференций http://infoneeds.kubsu.ru/

5. Библиотека информационных ресурсов кафедры информационных образовательных технологий http://mschool.kubsu.ru;
6. Электронный архив документов КубГУ http://docspace.kubsu.ru/
7. Электронные образовательные ресурсы кафедры информационных систем и технологий в образовании КубГУ и научно-методического журнала "ШКОЛЬНЫЕ ГОДЫ" http://icdau.kubsu.ru/

5.4 Публикации конференций A*

1. Farzana Ahamed Bhuiyan and Akond Rahman. 2021. Characterizing co-located insecure coding patterns in infrastructure as code scripts. In Proceedings of the 35th IEEE/ACM International Conference on Automated Software Engineering (ASE '20). Association for Computing Machinery, New York, NY, USA, 27–32. **https://doi.org/10.1145/3417113.3422154**

2. Michael Hilton, Timothy Tunnell, Kai Huang, Darko Marinov, and Danny Dig. 2016. Usage, costs, and benefits of continuous integration in open-source projects. In Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering (ASE '16). Association for Computing Machinery, New York, NY, USA, 426–437. https://doi.org/10.1145/2970276.2970358

3. IEEE Transactions on Big Data – научные статьи по обработке больших данных.
4. Journal of Big Data (SpringerOpen) – открытый журнал с исследованиями в области Big Data.
5. Big Data Research (Elsevier) – публикации по анализу, управлению и визуализации данных.
6. Data Science Journal (CODATA) – междисциплинарные исследования данных.
7. ACM Transactions on Knowledge Discovery from Data (TKDD) – методы извлечения знаний из больших данных.
8. https://openreview.net/forum?id=FMMF1a9ifL
9. https://openreview.net/forum?id=ElUrNM9U8c#discussion
10. https://openreview.net/forum?id=JoO6mtCLHD
11. https://aclanthology.org/2024.findings-emnlp.760/
12. https://aclanthology.org/2020.coling-main.588/
13. https://link.springer.com/chapter/10.1007/978-3-030-72113-8_30
14. https://link.springer.com/chapter/10.1007/978-3-031-42448-9_10
15. https://aclanthology.org/2024.findings-naacl.288/

6. Методические указания для обучающихся по освоению дисциплины (модуля)

По курсу предусмотрено проведение лекционных занятий, на которых дается систематизированный материал по Микросервисной архитектуре. В ходе лекций рассматриваются методы декомпозиции систем и ключевые концепции работы механизмов брокеров, очередей, средств удаленного вызова процедур и объектов. После каждой лекции рекомендуется выполнение практических заданий для закрепления ключевых понятий и методов.

Самостоятельная работа является ключевым компонентом успешного освоения дисциплины «Микросервисная архитектура». Её цель — сформировать у обучающихся способность к самостоятельному изучению, экспериментированию и решению практических задач в области распределённых систем без прямого руководства преподавателя.

Основные задачи самостоятельной работы:

- Закрепление и углубление знаний, полученных на лекциях;
- Формирование практических навыков разработки и развёртывания микросервисов;
- Развитие способности к самообучению и работе с технической документацией;
- Подготовка к выполнению лабораторных работ.

При самостоятельной работе студентам необходимо изучать рекомендованную литературу, в том числе в виде официальной документации к используемым открытым программным продуктам и облачным платформам.

Для студентов с ограниченными возможностями здоровья предусмотрены дополнительные индивидуальные консультации, на которых преподаватель разъясняет сложные аспекты дисциплины, помогает адаптировать задания и обеспечивает специальные условия для освоения практических навыков работы с многопоточными, векторными и GPU-вычислениями. Индивидуальный подход позволяет таким студентам полноценно участвовать в учебном процессе и достигать требуемых результатов обучения.

Важнейшим компонентом курса является самостоятельная проектная работа, в ходе которой студент разрабатывает законченное решение с уровнем технологической готовности (УТГ) 5-9 с применением CI/CD/CT для решения задач (кейсов) индустриальных партнеров. Допускается выполнение проектов в командах.

Кейсы ПАО «Сбербанк»

1. НЛП-анализ жалоб клиентов в свободной форме

Описание:

В рамках клиентского сервиса Сбербанк обрабатывает обращения из чатов, мобильного приложения и жалобной формы. Требуется построить модель семантического анализа, выделяющую суть обращения, определяющую тональность и потенциальную серьёзность инцидента.

Цель:

Автоматизировать классификацию обращений для ускорения маршрутизации и выявления повторяющихся болевых точек в продуктах и процессах.

Ожидаемый результат:

Прототип модели, автоматически выделяющей темы жалоб (например, «ошибка в приложении», «двойное списание»), их эмоциональную окраску и критичность.

2. Генерация сценариев фишинговых писем для обучения сотрудников

Описание:

Банк проводит киберучения, включая рассылку тестовых фишинговых писем сотрудникам для повышения их устойчивости к социальным атакам. Проект предполагает использование генеративной модели для создания реалистичных фишинговых писем различных типов (поддельные счета, HR-запросы, ИТ-поддержка).

Цель:

Создать генератор, способный на основе заданных параметров (тема, стиль, уровень угрозы) создавать тексты фишинга для тренировок.

Ожидаемый результат:

Набор разнообразных примеров фишинга и оценка их эффективности по реакции сотрудников, а также классификация моделей угроз.

3. Объяснимость и контроль генеративных моделей в банковском ИИ

Описание:

Банк активно использует LLM и NLP-сервисы (в чат-ботах, генерации шаблонов ответов, автоответах на e-mail), однако встает вопрос: как объяснять и контролировать поведение таких моделей, особенно в юридически значимых коммуникациях?

Цель:

Исследовать подходы к трассировке решений LLM (например, через логирование reasoning chain, пост-фильтрацию ответов, встроенные правила).

Ожидаемый результат:

Концепция системы explainability + compliance-модуля, обеспечивающего соответствие генерации стандартам банка и регулятора.

4. Модель анализа инвестиционной привлекательности малого бизнеса
Описание:
Банк активно развивает кредитование и инвестиционные инструменты для малого и среднего предпринимательства (МСП). Требуется создать модель, которая на основе открытых и банковских данных (выручка, расходы, тип деятельности, отзывы, онлайн-активность) оценивает инвестиционную привлекательность МСП.
Цель:
Разработать систему рейтинговой оценки компаний малого бизнеса с возможностью визуализации факторов и динамики показателей.
Ожидаемый результат:
Модель, присваивающая компании инвестиционный рейтинг (например, A–E), объясняющая ключевые параметры и дающая рекомендации для инвестора.

5. Сравнение text2video / text2img моделей
Описание:
Сбербанк заинтересован в сравнении text2video / text2img моделей (открытые модели, особенно китайские). Задача требует применения облачных ресурсов партнера для машинного обучения. От студентов требуется навык запуска открытых моделей, планирования, структурирования и логирования экспериментов, совместной работы. Задача может быть распараллелена для сравнения множества моделей независимо в группе студентов.
Цель:
Провести сравнение работы актуальных открытых моделей text2video / text2img.
Ожидаемый результат:
Таблица с результатами экспериментов модель / репозиторий / функционал / требования / оценка производительности / X примеров генераций (было/стало), human_eval по принципу арены (какая лучше)

Кейсы от «АВАЛАБ»
1. LLM и RAG для BI-системы Fastboard
Описание:
Для разрабатываемой компанией BI-системы Fastboard требуется разработать интерфейс на естественном языке для построения отчетов на больших массивах данных в ClickHouse. С помощью LLM необходимо классифицировать запросы пользователей на естественном языке и извлекать фактические параметры для дальнейшего вызова веб-сервиса отчетов.
Цель:
Разработать промпты для классификации и обработки запросов пользователей LLM и преобразования их к вызовам типовых отчетов с фактическими параметрами, извлекаемыми из запроса.
Ожидаемый результат:
Инструмент на основе LLM, позволяющий запрашивать данные о продажах.
2. Анализ обращений клиентов и CRM-переписки
Описание:
В службе клиентского сервиса застройщика ежедневно обрабатываются десятки обращений (e-mail, звонки, мессенджеры). Требуется реализовать систему семантического анализа и классификации NLU: выявлять суть обращений, уровень удовлетворенности, отслеживать повторяющиеся запросы.
Цель:
Автоматизировать первичный разбор и маршрутизацию запросов по тематике (сдача объекта, отделка, документы, жалоба и т.д.).

Ожидаемый результат:
Прототип, который выделяет суть обращений и формирует дашборд по текущим «болям» клиентов.

3. Мультимодальный агент для анализа строительных площадок

Описание:
ООО «АВА ЛАБ» разрабатывает систему для мониторинга строительных объектов. Требуется создать прототип мультимодального ИИ-агента, способного анализировать изображения со стройплощадки (видео/фото), а также принимать голосовые и текстовые запросы (например, «проверь монтаж перекрытия на 5 этаже»).

Цель:
Объединить возможности компьютерного зрения (распознавание стадии строительства, техники, нарушений) и НЛП (понимание запросов, отчётов).

Ожидаемый результат:
Интерактивный агент, который на запрос специалиста может показать нужный участок, прокомментировать прогресс, зафиксировать нарушения.

Для студентов с ограниченными возможностями здоровья предусмотрены дополнительные индивидуальные консультации, на которых преподаватель подробно разъясняет сложные аспекты дисциплины, помогает адаптировать практические задания и обеспечивает специальные условия для освоения методов работы с системами искусственного интеллекта. Индивидуальный подход позволяет таким студентам полноценно участвовать в учебном процессе и достигать требуемых результатов обучения.

7. Материально-техническое обеспечение по дисциплине (модулю)

1. Облачные платформы и сервисы
cloud.ru,YandexCloud,AWS/GCP/Azure– облачные вычисления

2. Системы управления версиями и коллаборации
Git/GitHub/GitLab – контроль версий кода и совместная разработка
. Свободное ПО (Open Source)
GitLab, GIT, Docker, Kubernetes

Виртуальные машины, кластер Managed Kubernetes и ресурсы GPU в облаке предоставляется индустриальным партнером ПАО «Сбербанк»:

№	Продукт	Параметры продукта	Кол-во	Кол-во конфигураций	Ед. изм.
1	Виртуальная машина	Виртуальная машина 10% vCPU 2 vCPU 4 RAM	1	60	Шт
		ОС Ubuntu 22.04	1		Шт
		Системный диск SSD	1		Шт
			10		Гб
		Аренда публичного IP	1		Шт
2	Виртуальная машина с GPU	Виртуальная машина с GPU NVIDIA® Tesla® V100 2 GPU 8 vCPU 128 ГБ RAM	1	1	Шт

		ОС Ubuntu_24.04	1		Шт
		Системный диск SSD	1		Шт
			2000		Гб
		Диск SSD	2		Шт
			4096		Гб
		Аренда публичного IP	1		Шт
3	K8S	Master node 8 vCPU 16 RAM	1	1	Шт
		Worker node 10% доля 4 vCPU 32 RAM	5		Шт
		Worker node SSD-NVME	64		Гб

Дополнительные облачные ресурсы предоставляются технологическим партнером Yandex Cloud.

№	Вид работ	Наименование учебной аудитории, ее оснащенность оборудованием и техническими средствами обучения
1	Лекционные занятия	Аудитория, укомплектованная специализированной мебелью и техническими средствами обучения
2	Лабораторные занятия	Аудитория, укомплектованная специализированной мебелью и техническими средствами обучения, компьютерами, проектором, программным обеспечением
3	Практические занятия	Аудитория, укомплектованная специализированной мебелью и техническими средствами обучения
4	Групповые (индивидуальные) консультации	Аудитория, укомплектованная специализированной мебелью и техническими средствами обучения, компьютерами, программным обеспечением
5	Текущий контроль, промежуточная аттестация	Аудитория, укомплектованная специализированной мебелью и техническими средствами обучения, компьютерами, программным обеспечением
6	Самостоятельная работа	Кабинет для самостоятельной работы, оснащенный компьютерной техникой с возможностью подключения к сети «Интернет», программой экранного увеличения и обеспеченный доступом в электронную информационно-образовательную среду университета.

Примечание: Конкретизация аудиторий и их оснащение определяется ОПОП.