МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«РОССИЙСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ – МСХА имени К.А. ТИМИРЯЗЕВА»
(ФГБОУ ВО РГАУ - МСХА имени К.А. Тимирязева)

Институт экономики и управления АПК
Кафедра прикладной информатики

УТВЕРЖДАЮ:
Директор института
экономики и управления АПК
_____ Л.И. Хоружий
“__28__” ____08_____ 2025 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Б1.В.11 API-технологии

для подготовки бакалавров

ФГОС ВО

Направление: 09.03.03 Прикладная информатика
Направленность: Программные решения для бизнеса

Курс 2
Семестр 3

Форма обучения: очная

Год начала подготовки: 2025

Москва, 2025

Разработчик (и): Лапшин М. С., ассистент
(ФИО, ученая степень, ученое звание) (подпись)

(ФИО, ученая степень, ученое звание) (подпись)

« 28 » августа 2025 г.

Рецензент: Ивашова О.Н., к.с.-х.н., доцент
(ФИО, ученая степень, ученое звание) (подпись)

« 28 » августа 2025 г.

Программа составлена в соответствии с требованиями ФГОС ВО, профессионального стандарта и учебного плана по направлению подготовки 09.03.03 «Прикладная информатика»

Программа обсуждена на заседании кафедры прикладной информатики протокол №1 от « 28 » августа 2025 г.

И.о. зав. кафедрой
прикладной информатики Худякова Е.В., д.э.н., профессор
(ФИО, ученая степень, ученое звание) (подпись)

« 28 » августа 2025 г.

Согласовано:

Председатель учебно-методической комиссии
института экономики и управления АПК
Гупалова Т.Н., к.э.н., доцент
(ФИО, ученая степень, ученое звание) (подпись)

« 28 » августа 2025 г.

И.о. заведующего выпускающей кафедрой
прикладной информатики Худякова Е.В., д.э.н., профессор
(ФИО, ученая степень, ученое звание) (подпись)

« 28 » августа 2025 г.

Заведующий отделом комплектования ЦНБ

СОДЕРЖАНИЕ

Аннотация

рабочей программы учебной дисциплины
Б1.В.11 «API-технологии»
для подготовки бакалавров по направлению 09.03.03 «Прикладная информатика», направленность «Программные решения для бизнеса»

Цель освоения дисциплины: Целью освоения дисциплины «API-технологии» является формирование у обучающихся компетенций, обеспечивающих способность к проектированию, реализации и сопровождению API-сервисов и прикладных систем на языке Python в составе решений на основе искусственного интеллекта, включая интеграцию модулей обработки данных и моделей машинного обучения, обеспечение качества и производительности сервисов, выбор и обоснование стека используемых библиотек и инструментов для решения задач обработки данных и построения ИИ-сервисов.

Место дисциплины в учебном плане: дисциплина включена в формируемую участниками образовательных отношений часть учебного плана по направлению подготовки09.03.03 «Прикладная информатика».

Требования к результатам освоения дисциплины: в результате освоения дисциплины формируются следующие компетенции (индикаторы) их достижения: ПК-8 (PL-1).1; ПК-8 (PL-1).2; ПК-8 (PL-1).3..

Краткое содержание дисциплины: Дисциплина посвящена проектированию и разработке API-сервисов на языке Python для компонентов систем искусственного интеллекта и обработки данных. Рассматриваются основы архитектуры API-ориентированных приложений, подходы к проектированию REST и gRPC-интерфейсов, документирование контрактов (OpenAPI/Proto), обработка ошибок и обеспечение надёжности. Изучаются фреймворки для разработки API на Python (например, FastAPI, Flask, Django REST Framework), интеграция с библиотеками обработки данных и машинного обучения, организация пайплайнов инференса и подготовки данных. Отдельное внимание уделяется вопросам тестирования, профилирования и оптимизации Python-приложений, выбору инструментов и библиотек с учётом требований к производительности и сопровождаемости, а также организации простейших конвейеров обработки больших данных и интеграции API-сервисов в состав таких систем.

Общая трудоемкость дисциплины/в т.ч. практическая подготовка: 108/4 (часы/зач. ед.)

Промежуточный контроль: зачет.

1. Цель освоения дисциплины

Целью освоения дисциплины «API-технологии» является формирование у обучающихся компетенций, обеспечивающих способность к проектированию, реализации и сопровождению API-интерфейсов и сервисов взаимодействия компонентов искусственного интеллекта на языке Python, включая разработку прикладных API-сервисов и программных модулей в различных парадигмах программирования, применение средств конкурентной обработки запросов (асинхронное программирование, параллельные вычисления и очереди задач),

интеграцию библиотек обработки данных и машинного обучения, а также анализ и интерпретацию результатов тестирования, профилирования и нагрузочных испытаний при решении задач ИИ.

Значимость формирования цифровых и алгоритмических компетенций в процессе профессиональной подготовки специалистов в области ИТ обусловлена требованиями цифровой трансформации экономики, а также приоритетами государственной политики Российской Федерации в области внедрения искусственного интеллекта и интеллектуальных технологий. Освоение дисциплины направлено на обеспечение готовности выпускников к практико-ориентированной деятельности в условиях цифрового общества и глобализированных рынков данных.

Дисциплина реализуется в контексте участия образовательных программ в федеральной инициативе подготовки топ-специалистов в области информационных технологий и искусственного интеллекта (ТОП ИИ), осуществляемой с 2025 года в рамках федеральных проектов «Искусственный интеллект» и «Кадры для цифровой трансформации» национального проекта «Экономика данных и цифровая трансформация государства». Программа ориентирована на развитие у студентов продвинутых компетенций в области проектирования и внедрения ИТ-решений и ИИ-моделей, в тесной связи с индустриальными партнёрами, участвующими в образовательном процессе, в том числе в формате проектной деятельности и софинансирования.

2. Место дисциплины в учебном процессе

Дисциплина «API-технологии» относится к части, формируемой участниками образовательных отношений, Блока 1 «Дисциплины (модули)» учебного плана и реализуется в 3 семестре в соответствии с требованиями ФГОС ВО, ОПОП ВО и Учебного плана по направлению подготовки 09.03.03 «Прикладная информатика», направленность (профиль) «Программные решения для бизнеса» (уровень образования — бакалавриат).

Содержательно дисциплина занимает важное место в подготовке бакалавра, поскольку обеспечивает переход от базовых знаний программирования, математического аппарата и основ ИИ к инженерной практике построения прикладных API-сервисов в составе ИИ-решений: проектирование контрактов и интерфейсов (REST/gRPC), разработка API-сервисов на Python, организация конкурентной обработки запросов (асинхронность, параллельные вычисления и очереди задач), интеграция модулей обработки данных и моделей машинного обучения, тестирование и профилирование, а также оценка качества решений по метрикам (задержка, пропускная способность, устойчивость). Новизна и значимость дисциплины в учебном процессе определяется её практико-ориентированной направленностью на разработку и эксплуатацию API-компонентов ИИ-решений с измеримыми требованиями к качеству, что соответствует профилю «Программные решения для бизнеса» и задачам внедрения ИИ в прикладных информационных системах.

Предшествующими курсами, на которых непосредственно базируется дисциплина «API-технологии», являются: «Программирование на языке Python»

(Б1.В.20.03), «Математическая статистика» (Б1.О.07), «Линейная алгебра» (Б1.О.22), «Теоретические основы информатики» (Б1.О.23), «Основы ИИ в АПК» (Б1.В.21). Дисциплина изучается параллельно с дисциплинами «Технологии обработки больших данных в АПК» (Б1.В.13) и «Базы данных» (Б1.О.20.02), что обеспечивает связность формирования компетенций по построению API-сервисов, работающих с данными, моделями и инфраструктурой хранения.

Дисциплина «API-технологии» является основополагающей для изучения следующих дисциплин и практик: «Информационные системы и технологии» (Б1.О.20.03), «ИТ-инфраструктура организации АПК» (Б1.В.03), «Управление информационными системами в АПК» (Б1.В.04), «Разработка геоинформационных систем для предприятий АПК» (Б1.В.05), «Разработка распределенных систем» (Б1.В.09), «Программирование в 1С» (Б1.В.10), «Технологии работы с открытыми данными» (Б1.В.14), «Компьютерное зрение» (Б1.В.15), «AIoT-технологии и средства автоматизации в АПК» (Б1.В.16), «Машинное обучение» (Б1.В.17), «Анализ пространственно-временных данных на основе машинного обучения» (Б1.В.18), «Глубокое обучение» (Б1.В.19), «Средства работы в команде» (Б1.В.ДВ.02.01).

Рабочая программа дисциплины «API-технологии» для инвалидов и лиц с ограниченными возможностями здоровья разрабатывается индивидуально с учетом особенностей психофизического развития, индивидуальных возможностей и состояния здоровья таких обучающихся.

3. Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

Образовательные результаты освоения дисциплины обучающимся, представлены в таблице 1.

4. Структура и содержание дисциплины

4.1 Распределение трудоёмкости дисциплины по видам работ

по семестрам

Общая трудоёмкость дисциплины составляет 108/4 часов, их распределение по видам работ семестрам представлено в таблице 2.

Таблица 1

Требования к результатам освоения учебной дисциплины

№ п/п	Код компетенции	Содержание компетенции (или её части)	Индикаторы компетенций	В результате изучения учебной дисциплины обучающиеся осваивают следующий уровень:		
				знать	уметь	владеть
	ПК-8 (PL-1)	Способен применять язык программирования Python для решения задач в области ИИ	ПК-8 (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 Уровень: Экспертный Уровень освоения индикатора: 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).	основы построения web- и API-приложений на Python; архитектурные подходы к построению сервисов (клиент–сервер, микросервисы, REST/gRPC); особенности выполнения кода в виртуальной машине Python (в том числе влияние GIL на параллелизм и производительность); базовые принципы организации тестирования и профилирования программ на Python/C++.	разрабатывать и отлаживать API-сервисы на Python (например, на основе FastAPI / Flask / Django REST Framework), реализовывать обработку запросов и ошибок, писать модульные и интеграционные тесты (pytest или аналог), применять средства профилирования (cProfile, line_profiler и др.) для оценки качества и производительности решений.	практическими приёмами организации структуры проекта на Python, ведения документации к API (docstring, OpenAPI-спецификация), использования инструментов тестирования и профилирования кода, приёмами поиска и исправления типичных ошибок (исключения, некорректная работа с ресурсами, неэффективные участки кода).

			ПК-8 (PL-1).2 Осуществляет выбор инструментов разработки на Python, приемлимых для создания прикладной системы обработки научных данных, машинного обучения и визуализации с заданными требованиями Уровень: Экспертный Уровень освоения индикатора: Умеет разрабатывать собственные компоненты для библиотек машинного обучения с учётом интеграции с ними	экосистему инструментов и библиотек Python для разработки API-сервисов и ИИ-решений (FastAPI, Flask, Django REST Framework, библиотеки для работы с HTTP-запросами, библиотеки для работы с данными и моделями — NumPy, pandas, scikit-learn, PyTorch / TensorFlow и др.); критерии выбора фреймворков и библиотек с учётом требований к функционалу, производительности и и сопровождаемости.	подбирать и обосновывать набор библиотек и фреймворков для реализации API-сервиса в контексте конкретной задачи (обработка данных, инференс модели, визуализация результатов, интеграция с хранилищами и внешними сервисами); конфигурировать выбранные инструменты и интегрировать их в единый программный комплекс.	навыками практического использования выбранных библиотек в небольших проектах: настройка и запуск API-сервиса, подключение модулей обработки данных и моделей, организация конфигурации и управления зависимостями (requirements.txt, poetry, virtualenv и др.).
			ПК-8 (PL-1).3 Разрабатывает и поддерживает системы обработки больших данных различной степени сложности Уровень: Экспертный Уровень освоения индикатора:	общие принципы организации обработки больших данных и построения ETL-/ELT-процессов; подходы к	проектировать и реализовывать фрагменты систем обработки данных с Python-компонентами, доступными через API (например, сервис	практическими приёмами оптимизации простых ETL-процессов и API-оболочек вокруг них (выбор форматов данных, пакетная

			Владеет инструментами профилирования и оптимизации ETL процессы для обработки больших данных в рамках Spark/Mapreduce фреймворка.	масштабированию Python-решений (распределённая обработка, очереди задач, использование фреймворков Spark / Dask / Airflow и др.); основные источники потерь производительности в таких системах.	подготовки данных или запуска расчётов), настраивать простейшие конвейеры обработки (загрузка → преобразование → выгрузка), использовать средства мониторинга и логирования для оценки работы системы.	обработка, кэширование допустимых результатов), базовыми навыками работы с инструментами профилирования и анализа производительности процессов обработки данных.

Таблица 2а

Распределение трудоёмкости дисциплины по видам работ по семестрам

Вид учебной работы	Трудоёмкость	
	час. всего/*	В т.ч. по семестрам №3
Общая трудоёмкость дисциплины по учебному плану	108/4	108/4
1. Контактная работа:	54,25/4	54,25/4
Аудиторная работа	54,25/4	54,25/4
в том числе:		
лекции (Л)	18	18
практические занятия (ПЗ)	36/4	36/4
лабораторные работы (ЛР)	0	0
курсовая работа (проект) (КР/КП) (консультация, защита)	0	0
консультации перед экзаменом	0	0
контактная работа на промежуточном контроле (КРА)	0,25	0,25
2. Самостоятельная работа (СРС)	53,75	53,75
реферат/эссе (подготовка)	0	0
курсовая работа/проект (КР/КП) (подготовка)	0	0
расчётно-графическая работа (РГР) (подготовка)	0	0
контрольная работа	0	0
самостоятельное изучение разделов, самоподготовка (проработка и повторение лекционного материала и материала учебников и учебных пособий, подготовка к лабораторным и практическим занятиям, коллоквиумам и т.д.)	46,75	46,75
Подготовка к экзамену (контроль)	0	0
Подготовка к зачёту/ зачёту с оценкой (контроль)	0	0
Вид промежуточного контроля:	зачет	

* в том числе практическая подготовка.(см учебный план)

4.2 Содержание дисциплины

Таблица 3а

Тематический план учебной дисциплины

Наименование разделов и тем дисциплин (укрупнённо)	Всего	Аудиторная работа				Внеаудиторная работа СР
		Л	ПЗ/С всего/*	ЛР всего/*	ПК-Р	
Раздел 1 «Основы API в ИИ-системах»	20	4	8	-	-	8
Раздел 2 «Реализация API на Pythpn и интеграция инференса»	24	4	8	-	-	12
Раздел 3 «Производительные и устойчивые API-сервисы на Python для ИИ»	26	4	8/2	-	-	14
Раздел 4. «Оптимизация и развертывание ИИ-API-сервисов (платформы и ускорители)»	24	4	8/2	-	-	12
Раздел 5. «Качество и эксплуатация API ИИ-сервисов»	13,75	2	4	-	-	7,75

Наименование разделов и тем дисциплин (укрупнённо)	Всего	Аудиторная работа				Внеаудиторная работа СР
		Л	ПЗ/С всего/*	ЛР всего/*	ПК-Р	
Всего за 3 семестр	108	18	36/4	0	0,25	53,75
Итого по дисциплине	108	18	36/4	0	0,25	53,75

* в том числе практическая подготовка

Раздел 1. Основы API в ИИ-системах

Тема 1. Введение в API-технологии для ИИ: роль API, типовые архитектуры, требования (latency, throughput, безопасность).

Роль API в жизненном цикле ИИ-систем (подготовка данных, обучение, инференс, получение и интерпретация результатов). Типовые архитектуры ИИ-решений в прикладных информационных системах: монолит, микросервисы, API gateway, отдельный сервис инференса, сервис данных. Синхронные и асинхронные взаимодействия, очереди задач и событийная интеграция. Ключевые требования к API: задержка ответа, пропускная способность, устойчивость, стоимость эксплуатации. Базовые риски и требования безопасности для API ИИ-сервисов: контроль доступа к данным и модели, ограничение частоты запросов, защита от некорректных входов и злоупотребления ресурсами. Обзор интерфейсов для инференса и интеграции: REST и gRPC, потоковые и пакетные режимы, применение API в прикладных сценариях бизнеса.

Тема 2. Проектирование API-контрактов для ИИ-сервисов: REST/gRPC, модели данных, ошибки, версионирование.

Принципы проектирования ресурсов и методов REST и RPC-методов gRPC под ИИ-сценарии. Модели данных запросов и ответов: входные признаки, параметры вызова, метаданные, результаты и их интерпретация. Схемы и валидация: обязательные поля, ограничения по типам и размерам, допустимые форматы. Обработка ошибок и коды ответов, классификация ошибок (ошибки клиента, ошибки сервера, ошибки модели, перегрузка, таймауты). Версионирование API и версионирование моделей (api version, model_version), принципы обратной совместимости. Идемпотентность, пагинация и фильтрация для сервисов данных и справочников. Документирование контракта (OpenAPI для REST, Proto для gRPC) и подготовка примеров запросов и ответов.

Раздел 2. Реализация API на Python и интеграция инференса

Тема 3. Реализация API на Python: REST (FastAPI/Flask/Django REST Framework) и/или gRPC Python, сериализация JSON/Protobuf.

Структура Python API-сервиса: маршрутизация, обработчики запросов, слои приложения, конфигурация и зависимости. Реализация REST-endpoint’ов на FastAPI/Flask/Django REST Framework и/или RPC-интерфейса на gRPC для Python. Сериализация и десериализация JSON и Protobuf, контроль схем и типов данных, использование моделей валидации (например, Pydantic для REST-контрактов). Валидация входных данных и нормализация параметров. Единый формат ошибок API и обработка исключений. Логирование и корреляция

запросов (request_id), журналирование ошибок и ключевых событий. Базовые меры защиты на уровне API: TLS (на уровне развёртывания), токены и ключи доступа, ограничения на размеры запросов и таймауты.

Тема 4. Интеграция AI/ML-компонента за API: пайплайн, препроцессинг/постпроцессинг, очереди, батчинг.
Организация пайплайна инференса: подготовка входа, препроцессинг, выполнение модели, постпроцессинг и формирование ответа. Форматирование входов и выходов модели, контроль размеров и типов данных, обработка отсутствующих и некорректных значений. Интеграция библиотек машинного обучения и инференса в Python-сервис (например, scikit-learn, PyTorch/TensorFlow, ONNX Runtime), управление загрузкой модели и прогрев. Очереди задач и диспетчеризация вычислений, применение фоновых задач и брокеров задач (например, Celery/RQ по возможности), обратное давление (backpressure) при перегрузке. Батчинг запросов и компромисс между задержкой и пропускной способностью. Кэширование допустимых результатов и дедупликация запросов. Таймауты, отмена выполнения и сценарии деградации сервиса.

Раздел 3. Производительные API-сервисы для ИИ на Python
Тема 5. Конкурентная обработка запросов в API ИИ-сервисах: асинхронность, очереди задач, параллельные вычисления, потокобезопасность общих ресурсов.
Модели конкурентности API-сервера в Python: синхронная обработка, асинхронная обработка запросов, выделение вычислительных задач в отдельные процессы, применение очередей задач. Влияние особенностей выполнения Python-кода на производительность и масштабирование (в том числе практические ограничения потоков и способы их обхода с помощью процессов и очередей). Проектирование очередей задач для инференса и правила управления нагрузкой. Безопасный доступ к общим ресурсам API: кэш, метрики, пул соединений к базе данных, ограничители частоты запросов. Типовые ошибки конкурентности и способы предотвращения. Проверка корректности поведения сервиса при параллельной нагрузке и воспроизведение проблем.

Тема 6. Производительность и профилирование API: метрики p95/p99, поиск узких мест, оптимизация сериализации, вычислений и работы с памятью.
Метрики и целевые показатели API (p95/p99 задержки, пропускная способность, доля ошибок). Профилирование обработки запросов и пайплайна инференса (время CPU, память, операции ввода-вывода), выявление узких мест в препроцессинге, сериализации, работе с базой данных и кэшем. Причины деградации: избыточные аллокации, лишние преобразования и копирования, неэффективные структуры данных, блокировки на общих ресурсах, очередь задач и ожидание. Оптимизация форматов обмена данными и сериализации, снижение накладных расходов. Нагрузочное тестирование и сравнение конфигураций (число воркеров, асинхронность, батчинг, кэширование), интерпретация результатов.

Раздел 4. Оптимизация под платформы и ускорители

Тема 7. Платформенная оптимизация и развертывание: ограничения, выбор библиотек и рантаймов, контейнеризация, измерения на целевой платформе.

Анализ ограничений платформы: ресурсы CPU и памяти, требования к времени ответа, особенности ОС и доступных сред выполнения. Выбор Python-стека и рантаймов инференса под платформу и требования бизнеса, оценка компромиссов между простотой сопровождения и производительностью. Практики развертывания API-сервисов: конфигурация окружения, управление зависимостями, контейнеризация (например, Docker), настройка параметров сервера приложений (workers, timeouts). Методы оптимизации моделей и вычислений в прикладной эксплуатации: квантование, сжатие, оптимизация графа, упрощение препроцессинга и постпроцессинга. Измерения на целевой платформе: задержка, память, стабильность, выявление ограничений и выбор мер по оптимизации.

Тема 8. Использование GPU-ускорения в ИИ-системе: готовые инструменты оптимизации, профилирование, анализ эффекта.

Сценарии применения ускорения в прикладной системе: ускорение инференса, ускорение части препроцессинга, пакетная обработка. Применение готовых средств оптимизации и развертывания моделей (например, TensorRT для NVIDIA GPU, а также возможности ONNX Runtime и фреймворков машинного обучения по использованию GPU), настройка режимов точности FP16/INT8 при наличии и контроль влияния на качество. Организация вычислений и данных при работе с GPU, снижение накладных расходов передачи данных, выбор стратегии батчинга. Профилирование ускоренного контура и поиск узких мест, анализ эффекта ускорения и компромиссы по качеству, задержке, стоимости и сложности сопровождения.

Раздел 5. Качество и эксплуатация API ИИ-сервисов

Тема 9. Тестирование, документирование и эксплуатация: модульное, интеграционное и контрактное тестирование, мониторинг и трассировка, итоговый мини-проект.

Тестирование API: модульное, интеграционное, контрактное, негативные сценарии и проверка устойчивости при ошибках входных данных, перегрузке и таймаутах. Проверка корректности инференса на контрольных наборах и регресс-проверки (на уровне подхода). Документирование API (OpenAPI/Proto), подготовка примеров запросов и ответов и сценариев использования в прикладной системе. Мониторинг и логирование: задержка, количество запросов, доля ошибок, ресурсные показатели, анализ событий. Трассировка запросов и диагностика деградаций, базовые уведомления о сбоях. Итоговый мини-проект: разработка прототипа Python API-сервиса для ИИ-задачи,

4.3 Лекции/лабораторные/практические/ занятия

Таблица 4а

Содержание лекций/лабораторного практикума/практических занятий и контрольные мероприятия

№ п/п	Название раздела, темы	№ и название лекций/лабораторных/практических/семинарских занятий	Формируемые компетенции	Вид контрольного мероприятия	Кол-во часов / из них практическая подготовка
	Раздел 1. Основы API в ИИ-системах.				
1	Тема 1. Введение в API-технологии для ИИ: роль API, типовые архитектуры, требования (latency/throughput/безопасность)	Лекция №1. API в ИИ-системах: архитектуры, контуры инференса, требования к latency/throughput и безопасности	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	–	2
		Практическая работа №1. Проектирование архитектуры Python API-сервиса инференса и потоков данных	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	Защита работы	2
		Практическая работа №2. Метрики и SLO для API инференса: p95/p99 latency, throughput, error rate, лимитирование и backpressure	ПК-8 (PL-1).1.	Защита работы	2
2	Тема 2. Проектирование API-контрактов для ИИ-сервисов: REST/gRPC, модели данных, ошибки, версионирование	Лекция №2. Контракты API для инференса: REST и gRPC, схемы данных, коды ошибок, идемпотентность, версионирование API и моделей	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	–	2

		Практическая работа №3. Проектирование REST API инференса и оформление спецификации OpenAPI	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	Защита работы	2
		Практическая работа №4. Проектирование gRPC интерфейса инференса и оформление proto-описания	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	Защита работы	2
	Раздел 2. Реализация API на Python и интеграция инференса.				
3	Тема 3. Реализация API на Python: REST (FastAPI/Flask/DRF) и/или gRPC Python, сериализация JSON/Protobuf	Лекция №3. Реализация API на Python: структура сервиса, валидация, сериализация JSON/Protobuf, обработка ошибок, логирование, request_id	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	–	2
		Практическая работа №5. Реализация базового REST API на Python (health + infer), валидация и единый формат ошибок	ПК-8 (PL-1).1.	Защита работы	2
		Практическая работа №6. Реализация API по контракту: gRPC Python или расширение REST, таймауты и ограничения входных данных	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	Защита работы	2
4	Тема 4. Интеграция AI/ML-компонента за API: пайплайн, препроцессинг/постпроцессинг, очереди, батчинг	Лекция №4. Инференс за API: пайплайн pre/infer/post, интеграция библиотек ML, очереди задач,	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	–	2

		батчинг, кэширование, таймауты			
		Практическая работа №7. Интеграция модели в API: подключение библиотеки инференса, прогрев, измерение базовой latency	ПК-8 (PL-1).2.	Защита работы	2
		Практическая работа №8. Очередь задач и батчинг инференса: backpressure и сравнение latency/throughput	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	Защита работы	2
	Раздел 3. Производительные API-сервисы для ИИ на Python.				
5	Тема 5. Конкурентная обработка запросов: асинхронность, процессы/очереди задач, потокобезопасность	Лекция №5. Конкурентность в Python API: async/await, workers, очереди задач, общие ресурсы и типовые ошибки	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	–	2
		Практическая работа №9. Настройка конкурентной обработки: workers/async, выделение вычислительных задач, параметры сервера приложений	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	Защита работы	2/1
		Практическая работа №10. Потокобезопасность общих ресурсов: кэш и метрики, лимитирование, корректность под параллельной нагрузкой	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	Защита работы	2
6	Тема 6. Производительность и	Лекция №6. Метрики API и	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	–	2

	профилирование API: p95/p99, bottleneck’и, оптимизация	профилирование Python: p95/p99, сериализация, I/O, работа с памятью, инструменты профилирования			
		Практическая работа №11. Нагрузочное тестирование API: сценарии нагрузки, сбор p95/p99, throughput, error rate	ПК-8 (PL-1).3.	Защита работы	2/1
		Практическая работа №12. Профилирование и оптимизация API: выявление узких мест и подтверждение эффекта “до/после”	ПК-8 (PL-1).1. ПК-8 (PL-1).3.	Защита работы	2
	Раздел 4. Оптимизация под платформы и ускорители.				
7	Тема 7. Платформенная оптимизация и развертывание: окружение, контейнеризация, измерения	Лекция №7. Развертывание и ограничения платформ: зависимости, Docker, настройка сервера приложений, оптимизация модели и пайплайна	ПК-8 (PL-1).2. ПК-8 (PL-1).3.	–	2
		Практическая работа №13. Развертывание API-сервиса: управление зависимостями, контейнеризация и запуск в целевом окружении	ПК-8 (PL-1).2.	Защита работы	2/1
		Практическая работа №14. Оптимизация модели и вычислений:	ПК-8 (PL-1).2. ПК-8 (PL-1).3.	Защита работы	2

		квантование/оптимизация графа, сравнение latency/памяти/качества			
8	Тема 8. Использование GPU-ускорения: инструменты, профилирование, эффект	Лекция №8. GPU-инференс в Python: режимы выполнения, батчинг, накладные расходы, инструменты оптимизации и профилирования (обзор)	ПК-8 (PL-1).2. ПК-8 (PL-1).3.	–	2
		Практическая работа №15. Настройка ускоренного инференса (при наличии GPU): подключение провайдера/фреймворка, сравнение с CPU	ПК-8 (PL-1).2. ПК-8 (PL-1).3.	Защита работы	2/1
		Практическая работа №16. Профилирование ускоренного контура: анализ bottleneck’ов и настройка батчинга/передачи данных	ПК-8 (PL-1).3.	Защита работы	2
	Раздел 5. Качество и эксплуатация API ИИ-сервисов.				
9	Тема 9. Тестирование, документирование и эксплуатация	Лекция №9. Качество API: unit/integration/contract тестирование, документация, мониторинг и трассировка, регресс производительности	ПК-8 (PL-1).1. ПК-8 (PL-1).2. ПК-8 (PL-1).3.	–	2
		Практическая работа №17. Контрактные и интеграционные тесты API, обновление	ПК-8 (PL-1).1. ПК-8 (PL-1).2.	Защита работы	2

		спецификации и негативные сценарии			
		Практическая работа №18. Итоговый мини-проект: Python API-сервис для ИИ-задачи, измерения, профилирование, отчёт и защита	ПК-8 (PL-1).1. ПК-8 (PL-1).2. ПК-8 (PL-1).3.	Защита проекта	2

Таблица 5а[1]

Перечень вопросов для самостоятельного изучения дисциплины

№ п/п	Название раздела, темы	Перечень рассматриваемых вопросов для самостоятельного изучения
Раздел 1		
1.	Тема 1 Введение в API-технологии для ИИ: роль API, типовые архитектуры, требования	Роль API в контурах ИИ (данные → препроцессинг → инференс → постпроцессинг → ответ). Типовые архитектуры API для ИИ: монолит/микросервисы, API gateway, inference-service, сервисы данных. Синхронные и асинхронные сценарии (очереди, события, callbacks). Показатели качества API для ИИ: latency, p95/p99, throughput, error rate, доступность, стоимость. Базовые риски безопасности API ИИ-сервиса (доступ к модели/данным, вредоносные входы, злоупотребление ресурсами). Компетенции: ПК-16 (PL-3).1. ПК-16 (PL-3).2.
2.	Тема 2. Проектирование API-контрактов для ИИ-сервисов: REST/gRPC, модели данных, ошибки, версионирование	REST vs gRPC для инференса: когда какой подход оправдан. Контракт запрос/ответ для модели: входные поля, параметры, метаданные, формат результата. Схемы данных и правила валидации (ограничения, типы, размеры, допустимые значения). Единая модель ошибок для AI API (ошибки клиента, сервера, модели, таймауты, перегрузка), коды/статусы и сообщения. Версионирование API и модели (api_version/model_version), совместимость и эволюция схем. Идемпотентность, дедупликация запросов, форматирование ответов и диагностика. Компетенции: ПК-16 (PL-3).1.
Раздел 2		
3	Тема 3. Реализация API на C++: HTTP (Boost.Asio/Beast) и/или gRPC C++, сериализация JSON/Protobuf	Архитектура C++ API-сервиса: обработчики, маршрутизация, middleware, конфигурация. Сборка проекта и управление зависимостями (CMake, структуры модулей). Сериализация и её влияние на производительность: JSON vs Protobuf, контроль схем и версий сообщений. Валидация входных данных и формирование единых ошибок API. Логирование и корреляция запросов (request-id), уровни логов и формат. Настройка таймаутов, keep-alive/соединений, базовые принципы TLS (на уровне понимания). Компетенции: ПК-16 (PL-3).1.

[1] Таблица 5а заполняется для очной формы обучения

№ п/п	Название раздела, темы	Перечень рассматриваемых вопросов для самостоятельного изучения
4	Тема 4. Интеграция AI/ML-компонента за API: пайплайн, препроцессинг/постпроцессинг, очереди, батчинг	Организация инференс-пайплайна: препроцессинг, инференс, постпроцессинг; где возникают задержки. Выбор и подключение рантайма инференса на C++ (ONNX Runtime/OpenVINO/TensorRT и др. — по стеку). Форматы входов/выходов и контроль размеров/типов тензоров, работа с памятью. Очереди задач, backpressure, стратегии отказа при перегрузке. Батчинг: правила формирования батча, компромисс latency vs throughput. Таймауты, отмена, “теплый старт” (warmup), кэширование допустимых результатов. Компетенции: ПК-16 (PL-3).1. ПК-16 (PL-3).2.
Раздел 3		
5	Тема 5. Многопоточность в API ИИ-сервисах: thread pool, синхронизация, атомики/блокировки, потокобезопасные структуры	Типовые конкурентные ошибки в API ИИ-сервиса: гонки на кэше, очередях, пулах соединений, метриках, контекстах модели. Thread pool и модели обработки запросов (producer–consumer), выбор размера пула. Когда использовать атомарные операции, а когда мьютексы/разделяемые блокировки; примеры для счётчиков и общих структур. Причины deadlock/livelock и базовые способы предотвращения (порядок захвата, scoped_lock, минимизация критических секций). Потокобезопасные структуры для API-контура: очередь задач, кэш, rate limiter. Компетенции: ПК-16 (PL-3).1.
6	Тема 6. Производительность и профилирование API: метрики p95/p99, поиск узких мест, оптимизация сериализации/копирований/памяти	Как корректно измерять latency/throughput для AI API (прогрев, стабильные входы, повторяемость). Интерпретация p95/p99 и типовые причины “длинного хвоста” задержек. Основы нагрузочного тестирования API и сценарии нагрузок (ступеньки, всплески, долгий прогон). Профилирование CPU/памяти и поиск bottleneck’ов: блокировки, аллокации, копирования, сериализация, I/O. Подходы к оптимизации: уменьшение копирований, эффективные буферы, выбор формата данных, настройка пула потоков и батчинга. Компетенции: ПК-16 (PL-3).1.
Раздел 4		
7	Тема 7. Платформенная оптимизация и embedded: ограничения, выбор библиотек/рантаймов, кросс-сборка, измерения	Анализ ограничений целевой платформы: CPU/RAM, энергопотребление, ОС/драйверы, доступные ускорители. Выбор рантайма инференса и зависимостей под платформу, критерии выбора (скорость, память, поддержка ops, переносимость). Методы облегчения модели и вычислений: квантование (INT8/FP16), pruning/сжатие, оптимизация графа; оценка влияния на качество и скорость. Кросс-компиляция и развёртывание (ARM/x86), особенности зависимостей и сборки. Измерения на целевой платформе: latency/throughput/память, выявление узких мест и их устранение. Компетенции: ПК-16 (PL-3).2.
8	Тема 8. GPU/FPGA-ускорение в ИИ-системе: инструменты оптимизации, профилирование, анализ эффекта	Где в AI API-сервисе появляется ускорение: инференс, препроцессинг, постпроцессинг; влияние передачи данных CPU↔GPU/FPGA. Применение готовых инструментов оптимизации и развертывания (например, TensorRT; провайдеры ускорения в ONNX Runtime/OpenVINO — по стеку), режимы FP16/INT8 и калибровка. Организация асинхронности и пайплайнинга, выбор размеров батча. Профилирование ускоренного контура и поиск ограничителей

№ п/п	Название раздела, темы	Перечень рассматриваемых вопросов для самостоятельного изучения
		производительности (GPU/CPU, копирования, синхронизация, очереди). Критерии оценки эффекта ускорения и корректное сравнение конфигураций. Компетенции: ПК-16 (PL-3).3.
Раздел 5		
9	Тема 9. Тестирование, документирование и эксплуатация: unit/integration/contract, мониторинг/трассировка, мини-проект	Виды тестов для AI API: unit, integration, contract; негативные сценарии (таймауты, перегрузка, невалидные входы). Подходы к тестированию инференса: эталонные примеры, стабильность результатов, регресс по качеству и скорости. Поддержка документации API (OpenAPI/Proto), примеры запросов/ответов, политика версий. Мониторинг и диагностика: метрики latency/RPS/errors, логи с request-id, трассировка запросов и поиск “узких мест” в цепочке. Регресс производительности после изменений, базовые практики CI для сборки/тестов/проверок. Компетенции: ПК-16 (PL-3).1. ПК-16 (PL-3).2. ПК-16 (PL-3).3.

5. Образовательные технологии

Таблица 6

Применение активных и интерактивных образовательных технологий

№ п/п	Тема и форма занятия		Наименование используемых активных и интерактивных образовательных технологий
1.	Тема 1. Введение в API-технологии для ИИ	Л	Проблемная лекция с разбором кейсов прикладных ИИ-сервисов в бизнес-сценариях. Дискуссия и работа в малых группах по выбору архитектуры и требований к качеству (latency/throughput/устойчивость).
		ПЗ	Информационно-коммуникационные технологии (ИКТ): работа с учебным порталом, репозиторием, электронными ресурсами.
2	Тема 2 Проектирование API-контрактов для ИИ-сервисов	Л	Кейс-метод: проектирование контракта под заданный сценарий инференса. Проектная работа в группах с взаимной экспертизой (peer review) спецификаций OpenAPI/Proto.
		ПЗ	использование редакторов спецификаций и онлайн-валидаторов.
3	Тема 3. Реализация API на Python	Л	Мастер-класс/демонстрация (live coding) базового API-сервиса. Практикум в компьютерном классе: реализация endpoint’ов/методов и отладка.
		ПЗ	работа с системой контроля версий, сборкой (CMake), CI-шаблонами.
4	Тема 4. Интеграция AI/ML-компонента за API	Л	Кейс-стади: интеграция инференса и обработка ошибок/таймаутов.
		ПЗ	работа с датасетом примеров и инструментами измерений.
5	Тема 5. Конкурентная обработка запросов в Python API	Л	Проблемное обучение: разбор типовых гонок/дедлоков на примерах API-кэша/очереди.
		ПЗ	Практикум: командное решение задач по синхронизации (atomics/locks), code review

			решений. ИКТ: использование санитайзеров и
7	Тема 7. Развертывание и	Л	Кейс-метод: подбор платформы (по стека под возможности).
6	Тема 6. Производительность и профилирование	ПЗ	практика сборки/эмуляции) и сравнение конфигураций.
		ПЗ	инструментами профилирования (perf/VTune), документирование и защита
8	Тема 8. Использование	Л	Интерактивная лекция с разбором ускорения и профилирования.
	GPU-ускорения в ИИ-системе	ПЗ	применение готовых инструментов оптимизации (например, TensorRT) и профилирование (Nsight/аналог), обсуждение результатов в группе. ИКТ: работа с SDK ускорителя и профилировщиками.
9	Тема 9. Тестирование, документирование и эксплуатация API	Л	Проектное обучение: выполнение мини-проекта по разработке API-сервиса с документацией и тестами.
		ПЗ	Презентация/защита результатов и взаимная оценка. ИКТ: репозиторий, трекер задач, шаблоны отчётов, электронные ресурсы.

6. Текущий контроль успеваемости и промежуточная аттестация по итогам освоения дисциплины

6.1. Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений и навыков и (или) опыта деятельности

1) Примеры заданий практических работ

Практическая работа №1. Проектирование архитектуры API-сервиса для ИИ-инференса

Задание. Разработать архитектуру API-ориентированной ИИ-системы для выбранной задачи инференса на Python, определить состав компонентов и взаимодействий, сформулировать требования к качественным характеристикам.
Порядок выполнения работы.

1. Выбрать вариант ИИ-задачи (по заданию преподавателя) и определить входные данные (формат, объём, ограничения) и ожидаемый результат (структура ответа, показатели).
2. Сформулировать требования к времени ответа и пропускной способности, а также ограничения на размеры запросов и частоту обращений.
1. Определить состав компонентов решения: API-сервис на Python, модуль инференса, компоненты подготовки данных (при необходимости), кэш результатов, хранилище данных и/или очередь задач.
2. Описать взаимодействие компонентов и поток данных: путь запроса от клиента до результата, точки валидации, места хранения и передачи данных.
3. Описать последовательность обработки запроса для двух сценариев: корректный запрос; отказ (ошибка валидации, таймаут обработки, перегрузка сервиса).
4. Сформулировать не менее пяти требований к качеству и безопасности API: задержка ответа (в том числе p95/p99), пропускная способность, устойчивость, ограничения на входные данные, базовые требования к контролю доступа и журналированию.

Отчет. Архитектурная схема (компоненты и связи). Описание последовательности обработки запроса для двух сценариев. Перечень требований к качеству и безопасности.

Практическая работа №2. Метрики качества и целевые показатели API ИИ-сервиса

Задание. Определить метрики функционирования API ИИ-сервиса и установить целевые значения показателей качества (SLO), подготовить программу измерений.
Порядок выполнения работы.

1. Определить перечень метрик: среднее время ответа и процентильные значения (p95, p99), пропускная способность (запросов в секунду), доля ошибок, как минимум одна ресурсная метрика (CPU или память), а также метрика очереди/батча (при наличии).

2. Задать целевые значения для ключевых метрик и описать критерии деградации качества (какие значения считаются неприемлемыми и почему).
3. Сформировать профиль нагрузки, включающий базовый режим, режим ступенчатого роста и режим всплеска нагрузки; указать длительность каждого режима и число параллельных клиентов.
4. Описать методику измерений: прогрев сервиса, количество повторов, фиксированный набор входных данных, правила фиксации результатов и сравнения конфигураций.

Отчет. Таблица метрик и целевых значений. Описание критериев деградации. Программа измерений с профилем нагрузки и условиями эксперимента.

Практическая работа №3. Проектирование REST API и подготовка спецификации OpenAPI

Задание. Разработать контракт REST API сервиса инференса и оформить его в виде спецификации OpenAPI с примерами запросов и ответов.

Порядок выполнения работы.

1. Определить состав методов API: метод проверки работоспособности (health/ready) и метод инференса.
2. Описать структуру запроса инференса (входные данные, параметры, версия модели) и структуру ответа (результат, дополнительные показатели, метаданные, request_id).
3. Определить правила валидации входных данных (обязательные поля, допустимые диапазоны, ограничения на размер и формат).
4. Раработать единый формат ошибок API и перечень типовых ошибок (валидация, таймаут, перегрузка, внутренние ошибки).
5. Определить стратегию версионирования API и совместимости изменений, а также правила версионирования модели (model_version).
6. Сформировать спецификацию OpenAPI и подготовить примеры трёх сценариев: корректный запрос; запрос с ошибкой валидации; отказ по перегрузке или таймауту.

Отчет. Файл спецификации OpenAPI. Примеры запросов и ответов для трёх сценариев. Краткое описание стратегии версионирования.

Практическая работа №4. Проектирование gRPC интерфейса и подготовка proto-описания

Задание. Разработать gRPC интерфейс сервиса инференса и оформить его в виде proto-описания с правилами совместимости.

Порядок выполнения работы.

1. Определить состав gRPC-сервиса и перечень rpc-методов (не менее одного метода инференса).
2. Описать структуру сообщений запроса и ответа: входные данные, параметры вызова, версия модели, идентификатор запроса, результат и метаданные.

3. Определить правила обработки ошибок и соответствующие статусы (некорректные входные данные, таймаут, перегрузка).
4. Задать требования к совместимости схемы при развитии интерфейса (добавление полей, reserved, запрет переиспользования номеров и изменения смысла полей).
5. Оформить proto-описание и подготовить примеры сообщений запроса и ответа.

Отчет. Файл proto-описания. Краткое описание правил совместимости. Примеры запроса и ответа.

6.2. Описание показателей и критериев контроля успеваемости, описание шкал оценивания

Оценочные средства текущего контроля успеваемости и сформированности компетенций основана на подсчете баллов, «заработанных» студентом в течение семестра.

Успеваемость студента по дисциплине оценивается в баллах от 0 до 100.

Оценка знаний проводится по следующим критериям:

- посещение занятий – 10 баллов;
- выполнение практических заданий – 10 баллов;
- выполнение контрольной работы - 10 баллов;
- качество коллоквиума – 10 баллов;
- качество курсового проекта - 20 баллов;
- промежуточный контроль (зачет) – 20 баллов;
- промежуточный контроль (экзамен) – 20 баллов.

Соответствие балльной оценки общепринятой 4-х балльной шкале оценок приведено в таблице 7.

Таблица 7

Соответствие балльных оценок по 4-х балльной шкале

Балльная оценка	Оценка по 4хбалльной шкале	Оценка по шкале «Зачтено» / «Не зачтено»
0-59	Неудовлетворительно - 2	Не зачтено
60-69	Удовлетворительно - 3	Зачтено
70-89	Хорошо – 4	Зачтено
90-100	Отлично - 5	Зачтено

Критерии оценивания результатов обучения показаны в таблицах 8,9.

Таблица 8

Критерии оценивания по шкале «Зачтено» / «Не зачтено»

Оценка «Зачтено/Не зачтено»	Критерии оценивания
Зачтено	Оценка «**зачтено**» ставится, если студент показал глубокие систематизированные знания в объеме, необходимом для дальнейшей учебы и в предстоящей работе по профессии, владеет приемами рассуждения и сопоставления материала из разных источников: теорию связывает с практикой, другими темами данного курса, других изучаемых предметов; выполнил все практические задания, предоставив правильные и аргументированные выводы в соответствии с предъявленными требованиями.
Незачтено	Оценка «**не зачтено**» ставится, если студент в ответах не раскрыл основное содержание вопросов, носящих несистематизированный, отрывочный, поверхностный характер; студент не понимает существа излагаемых им вопросов, что свидетельствует о том, что студент не может дальше продолжать обучение или приступать к профессиональной деятельности без дополнительных занятий по соответствующей дисциплине; не выполнил практические задания в соответствии с предъявленными требованиями.

Таблица 9

Критерии оценивания результатов обучения (зачет)

Оценка	**Критерии оценивания**
Высокий уровень «5» (отлично)	оценку **«отлично»** заслуживает студент, освоивший знания, умения, компетенции и теоретический материал без пробелов; выполнивший все задания, предусмотренные учебным планом на высоком качественном уровне; практические навыки профессионального применения освоенных знаний сформированы. **Компетенции**, закреплённые за дисциплиной, **сформированы на уровне – высокий**.

Средний уровень «4» (хорошо)	оценку **«хорошо»** заслуживает студент, практически полностью освоивший знания, умения, компетенции и теоретический материал, учебные задания не оценены максимальным числом баллов, в основном сформировал практические навыки. **Компетенции**, закреплённые за дисциплиной, **сформированы на уровне – хороший (средний)**.
Пороговый уровень «3» (удовлетворительно)	оценку **«удовлетворительно»** заслуживает студент, частично с пробелами освоивший знания, умения, компетенции и теоретический материал, многие учебные задания либо не выполнил, либо они оценены числом баллов близким к минимальному, некоторые практические навыки не сформированы. **Компетенции**, закреплённые за дисциплиной, **сформированы на уровне – достаточный**.
Минимальный уровень «2» (неудовлетворительно)	оценку «**неудовлетворительно»** заслуживает студент, не освоивший знания, умения, компетенции и теоретический материал, учебные задания не выполнил, практические навыки не сформированы. **Компетенции**, закреплённые за дисциплиной, **не сформированы**.

7. Учебно-методическое и информационное обеспечение дисциплины

7.1. Основная литература

1. Jiayi Wang, Guoliang Li AOP: Automated and Interactive LLM Pipeline Orchestration for Answering Complex Queries URL: https://vldb.org/cidrdb/papers/2025/p32-wang.pdf (дата доступа 28 августа 2025 г.)
2. Kim, S., Yu, Y. & Seo, H. Artificial intelligence orchestration for text-based ultrasonic simulation via self-review by multi-large language model agents. Sci Rep 15, 12474 (2025).URL: https://doi.org/10.1038/s41598-025-97498-y (дата доступа 28 августа 2025 г.)
3. Промышленные API для распознавания изображений, речи, рекомендаций, используемые как backend для LLM-агентов (описание входных/выходных форматов, SLA). URL: https://orq.ai/blog/llm-orchestration

7.2. Дополнительная литература

1. Петрова Е.С. Модели tool calling в LLM через REST API: безопасность и масштабируемость // Научный результат. Информационные технологии. 2025. Т. 10. № 2.
2. Смирнов Д.Ю. Защита API от prompt-инъекций в сервисах генеративного ИИ // Искусственный интеллект и принятие решений. 2025. № 1. С. 112–130.

3. Лебедев М.А. gRPC vs REST для model-serving в ИИ-приложениях // Вестник компьютерных и информационных технологий. 2025. № 1. (2 уровень).
4. Федоров И.П. Аутентификация и rate limiting в multi-tenant AI API // Программные продукты и системы. 2024. № 3. (2 уровень, ВАК по ИС).
5. Григорьева О.Н. API-интерфейсы для vision-language моделей: протоколы и производительность // Искусственный интеллект и принятие решений. 2024. № 3. С. 78–95. (1 уровень).

7.3. Нормативные правовые акты

1. ГОСТ Р 59277 2020. Системы искусственного интеллекта. Классификация, термины и общие положения. – М.: Стандартинформ.
2. ГОСТ Р 59898 2021. Оценка качества систем искусственного интеллекта. Общие положения. – М.: Стандартинформ.
3. ГОСТ Р 71476 2024. Искусственный интеллект. Концепции и терминология искусственного интеллекта. – М.: Стандартинформ.
4. ГОСТ Р ИСО/МЭК 25010 2015. Системы и программная продукция. Модели качества. – М.: Стандартинформ.
5. ГОСТ Р ИСО/МЭК 12207 2010. Информационная технология. Процессы жизненного цикла программных средств. – М.: Стандартинформ.
6. ГОСТ 19.201 78. Единая система программной документации. Техническое задание. Требования к содержанию и оформлению. – М.: Изд во стандартов.
7. ГОСТ 19.502 78. Единая система программной документации. Описание применения. Требования к содержанию и оформлению. – М.: Изд во стандартов.
8. ГОСТ 34.602 89. Информационная технология. Комплекс стандартов на автоматизированные системы. Техническое задание на создание автоматизированной системы. – М.: Изд во стандартов.
9. ГОСТ Р 57580 2017. Защита информации финансовых организаций. Общие положения. – М.: Стандартинформ.
10. ГОСТ Р 56939 2016. Защита информации. Обеспечение безопасности персональных данных при их обработке в информационных системах. – М.: Стандартинформ.

8. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины

1. Hugging Face – платформа открытых моделей и датасетов, документация по Model Hub и Inference API: https://huggingface.co
2. GitHub – репозитории библиотек и примеров сервисов с ИИ-API (в т. ч. организации Hugging Face, OpenAI и др.): https://github.com
3. OpenAI – документация по API для работы с языковыми и мультимодальными моделями: https://platform.openai.com/docs
4. Google Cloud AI / Vertex AI – документация по API для NLP, Vision, Speech, табличных данных и генеративного ИИ: https://cloud.google.com/ai

5. Microsoft Azure AI (Azure AI Studio, Cognitive Services) – API-сервисы анализа текста, изображений, речи и генеративного ИИ: https://azure.microsoft.com/en-us/products/ai-services
6. Amazon Web Services (Amazon Bedrock и другие AI-сервисы) – API для генеративных и классических ML-моделей: https://aws.amazon.com/machine-learning
7. Платформы российских провайдеров генеративного ИИ и AI-API (GigaChat, YaGPT, др.) – официальные порталы с документацией по REST/SDK-интерфейсам.
8. Документация библиотеки Transformers (работа с моделями Hugging Face, примеры API-интеграции): https://huggingface.co/docs/transformers
9. Документация Hugging Face Hub / Inference Endpoints (управление репозиториями и деплой моделей как сервисов): https://huggingface.co/docs/hub
10. Документация по фреймворкам оркестрации LLM и агентным системам (chain-/agent-подходы, tool calling, вызов внешних API).
11. Документация web-фреймворков для создания REST/gRPC-сервисов (FastAPI, Django REST Framework, Flask, gRPC и др.).
12. Облачные и локальные среды для практики (Google Colab, Kaggle Notebooks, JupyterHub в вузе и др.).

9. Перечень программного обеспечения и информационных справочных систем

1. Базы данных Министерства сельского хозяйства Российской Федерации: www.mcx.ru.

2. Базы данных Федеральной службы государственной статистики: www.gks.ru.

3. Python 3.12+ – интерпретатор языка программирования для разработки API-сервисов и интеграции моделей ИИ (библиотеки requests, FastAPI, Flask).

4. FastAPI – фреймворк для создания высокопроизводительных REST/gRPC API с автоматической документацией (Swagger/OpenAPI).

5. Hugging Face Transformers – библиотека для загрузки, дообучения и инференса моделей ИИ (LLM, CV, speech) с примерами API-интеграции.

6. LangChain / LlamaIndex – фреймворки оркестрации LLM-агентов, tool calling и цепочек вызовов внешних API.

7. GitHub Copilot / Cursor AI – ИИ-ассистенты для автодополнения кода в IDE (VS Code, JetBrains), генерации API-эндпоинтов и тестов.

8. Docker / Podman – контейнеризация для деплоя ИИ-моделей как микросервисов с API (многоуровневая архитектура).

9. Google Colab / Kaggle Notebooks – облачные Jupyter-среды для прототипирования API-сервисов с моделями ИИ.

Таблица 9

Перечень программного обеспечения

№ п/п	Наименование раздела учебной дисциплины (модуля)	Наименование программы	Тип программы	Автор / организация	Год разработки / актуальной версии
1	APIтехнологии в ИИ. Основы разработки сервисов	Python 3.x	Язык программирования, среда выполнения	Python Software Foundation	2008 / актуальная версия
2	APIтехнологии в ИИ. Вебсервисы и REST	FastAPI	Фреймворк для разработки web и RESTAPI	Sebastián Ramírez и соавт.	2018 / актуальная версия
3	Модели ИИ и интеграция по API	Hugging Face Transformers	Библиотека для работы с моделями ИИ	Hugging Face Inc.	2018 / актуальная версия
4	Оркестрация LLM и вызов внешних API	LangChain	Фреймворк для построения LLMагентов и цепочек	LangChain Inc.	2022 / актуальная версия
5	Контейнеризация ИИсервисов с API	Docker Desktop / Docker Engine	Платформа контейнеризации	Docker Inc.	2013 / актуальная версия
6	Управление исходным кодом и проектами	Git / GitHub	Система контроля версий и хостинг репозиториев	Linus Torvalds, GitHub Inc.	2005 / актуальная версия
7	Проектирование и тестирование API	Postman	Среда тестирования и документации API	Postman Inc.	2014 / актуальная версия
8	Демонстрация и лабораторные работы по ИИ	Jupyter Notebook / JupyterLab	Интерактивная среда для выполнения кода	Project Jupyter	2015 / актуальная версия

Сведения об обеспеченности специализированными аудиториями, кабинетами, лабораториями

Наименование специальных* помещений и помещений для самостоятельной работы (№ учебного корпуса, № аудитории)	Оснащенность специальных помещений и помещений для самостоятельной работы**
1	2
Корпус 1, Аудитория 201 Количество рабочих мест: 24	Встроенные сетевые адаптеры (Intel I219-V или Realtek RTL8111H), интерфейс RJ-45, скорость 10/100/1000 Мбит/с. Точки доступа: Ubiquiti UniFi AP AC Pro, стандарты IEEE 802.11a/b/g/n/ac, частоты 2.4 ГГц (450 Мбит/с) и 5 ГГц (1300 Мбит/с), поддержка MU-MIMO, питание PoE.
Корпус 1, Аудитория 203 Количество рабочих мест: 18	Встроенные сетевые адаптеры (Intel I219-V или Realtek RTL8111H), интерфейс RJ-45, скорость 10/100/1000 Мбит/с. Точки доступа: Ubiquiti UniFi AP AC Pro, стандарты IEEE 802.11a/b/g/n/ac, частоты 2.4 ГГц (450 Мбит/с) и 5 ГГц (1300 Мбит/с), поддержка MU-MIMO, питание

	PoE.Структурное подразделение: Кафедра Цифровая кафедра
Корпус 1, Аудитория 206 Количество рабочих мест: 24	Встроенные сетевые адаптеры (Intel I219-V или Realtek RTL8111H), интерфейс RJ-45, скорость 10/100/1000 Мбит/с. Точки доступа: Ubiquiti UniFi AP AC Pro, стандарты IEEE 802.11a/b/g/n/ac, частоты 2.4 ГГц (450 Мбит/с) и 5 ГГц (1300 Мбит/с), поддержка MU-MIMO, питание PoE.
Центральная научная библиотека имени Н.И. Железнова	Читальные залы библиотеки
Студенческое общежитие	Комната для самоподготовки

11. Методические рекомендации обучающимся по освоению дисциплины

Освоение дисциплины «API-технологии» требует активной вовлечённости обучающихся в процесс решения прикладных задач, связанных с проектированием, разработкой и сопровождением API-сервисов в составе решений на основе искусственного интеллекта, включая интеграцию компонентов обработки данных и инференса моделей, обеспечение устойчивости и производительности, а также тестирование и оценку качества полученных решений. Дисциплина ориентирована на развитие проектного и исследовательского подхода с использованием кейсов, приближённых к практикам разработки программных решений для бизнеса (например, построение API для сервиса инференса, интеграция с источниками данных и информационными системами, обеспечение требований по задержке и пропускной способности, поддержка версионирования API и моделей).

Лекционные занятия направлены на формирование системного понимания принципов построения API в составе ИИ-решений, проектирования контрактов (REST/gRPC), организации обмена данными и обработки ошибок, а также разработки прикладных API-сервисов на языке Python с учётом требований к качеству, сопровождаемости и производительности. Лекции сопровождаются презентациями, схемами архитектур и потоков данных, примерами контрактов и фрагментами кода, проходят с использованием мультимедийного оборудования и электронных образовательных ресурсов (например, LMS, системы опроса и тестирования, репозитории с учебными примерами, средства совместной работы).

Обучающиеся обязаны вести тематический конспект, дополняя его материалами из рекомендованной и дополнительной литературы, в том числе отраслевыми источниками и официальной документацией стандартов и библиотек (Python, документация FastAPI/Flask/Django REST Framework, OpenAPI, gRPC/Protocol Buffers, документация библиотек машинного обучения и инференса, а также инструментов тестирования и профилирования). Перед каждой новой темой рекомендуется повторение ключевых концепций и самостоятельное выполнение мини-заданий на закрепление.

Практические занятия проводятся в компьютерных классах, оборудованных современным программным обеспечением, и строятся по

принципу: постановка инженерной задачи – разбор эталонного решения/шаблона – выполнение индивидуального варианта – анализ и обсуждение результатов. Каждое практическое задание направлено на развитие конкретных навыков, в том числе: проектирование и документирование API-контрактов для ИИ-сервисов (OpenAPI/Proto), включая версионирование и обработку ошибок; реализация API-сервисов на Python (REST и/или gRPC), интеграция с модулем инференса и компонентами обработки данных; организация конкурентной обработки запросов (асинхронное программирование, настройка числа воркеров, применение очередей задач и фоновых процессов при необходимости), обеспечение корректной работы общих ресурсов (кэш, метрики, лимитирование); измерение и анализ метрик качества API (p95/p99 задержки, пропускная способность, доля ошибок), нагрузочное тестирование и профилирование кода; оптимизация модели и пайплайна инференса на уровне прикладного Python-решения (прогрев, батчинг, кэширование, снижение накладных расходов сериализации и преобразований данных); подготовка к развертыванию и воспроизводимому запуску сервиса (управление зависимостями, конфигурация, контейнеризация). При наличии технической возможности рассматриваются сценарии использования ускорения вычислений (GPU) и оценка эффекта ускорения в контуре API.

Результаты работы оформляются в виде исходного кода и отчётных материалов (контракт API, инструкция запуска, результаты тестирования и измерений производительности) и размещаются в системе контроля версий и/или в LMS (например, Git, GitLab/GitHub, корпоративный репозиторий, электронный курс).

Наиболее трудоёмкие темы дисциплины:
• Тема 5. Конкурентная обработка запросов и устойчивость API-сервиса на Python (асинхронность, очереди задач, работа с общими ресурсами).
• Тема 6. Нагрузочное тестирование, профилирование и оптимизация производительности API.
• Тема 7–8. Развертывание, оптимизация модели и пайплайна, а также использование ускорения вычислений (при наличии условий).

Самостоятельная работа включает:
• изучение справочной и профессиональной документации по Python, API-фреймворкам (FastAPI/Flask/Django REST Framework), OpenAPI, gRPC/Protocol Buffers, библиотекам машинного обучения и инференса, инструментам тестирования и профилирования;
• выполнение заданий по проектированию контрактов API, разработке и тестированию API-сервиса, анализу метрик и профилированию;
• подготовку мини-проектов, сравнительных измерений и отчётов по принятым архитектурным и оптимизационным решениям.

Для полноценного освоения дисциплины обучающемуся необходимо:
• посещать все аудиторные формы занятий (лекции и практики);
• поддерживать личную систему организации разработки (структура проекта, управление зависимостями, конфигурации, контроль версий);
• использовать электронные ресурсы для хранения прогресса и материалов (Git, облачные хранилища, LMS);

• принимать участие в консультациях, включая онлайн-формат (через LMS, мессенджеры, e-mail);
• активно участвовать в коллективной обратной связи при разборе решений и защите практических работ.

Промежуточная и итоговая аттестация проводится на основе совокупности оценок за выполненные практические работы, участия в защите и анализа выбранного кейса. Итоговая форма зачёта — защита индивидуального (или парного) проекта, включающего разработку API-сервиса для ИИ-задачи на Python с демонстрацией работы, представлением контракта, результатами тестирования, нагрузочных измерений и профилирования, а также обоснованием выбранного стека и применённых оптимизаций.

Виды и формы отработки пропущенных занятий

Студент, пропустивший занятия обязан отработать:

Пропущенные лекции – предоставив преподавателю конспект лекции, ответив на вопросы устно, пройдя собеседование по пропущенной теме, пройти тестирование.

Пропущенные практические занятия – в форме выполненных заданий, устного опроса, посещения дополнительных занятий.

Защита индивидуальных заданий проводятся в часы в дни и часы, устанавливаемые преподавателем.

Пропуск занятия по документально подтвержденной дирекцией уважительной причине не является основанием для снижения оценки выполненной практической работы.

Методические рекомендации преподавателям по организации обучения по дисциплине

Преподавание курса «API-технологии» должно носить контекстный характер и обеспечивать формирование у обучающихся профессионально значимых компетенций, связанных с проектированием, разработкой и сопровождением API-сервисов в составе прикладных программных решений для бизнеса с использованием технологий искусственного интеллекта. В процессе обучения должна отчётливо прослеживаться целевая установка на развитие личности и инженерного мышления; интеграционное единство форм, методов и средств обучения; взаимодействие обучающихся и преподавателя; учёт индивидуального стиля учебной деятельности и педагогической деятельности.

Реализация технологий контекстного обучения в профессионально-образовательном процессе обеспечивается соблюдением следующих условий: мотивационное обеспечение обучающихся на основе включения в профессионально ориентированные задачи (проектирование контракта API для инференса, обеспечение требований по задержке и пропускной способности, выбор стека Python-инструментов и подхода к развертыванию сервиса); наличие диагностически заданной цели обучения, то есть измеримого представления об ожидаемом результате (достижение индикаторов ПК-8 (PL-1).1–ПК-8 (PL-1).3,

подтверждаемое результатами тестирования, профилирования и сравнительных измерений); представление учебного материала в виде системы познавательных и практических задач, ситуаций, заданий, проектов и упражнений (контракты REST/gRPC, реализация API на Python, интеграция библиотеки инференса, конкурентная обработка запросов, нагрузочное тестирование и оптимизация); описание способов взаимодействия субъектов образовательного процесса (обсуждение архитектурных решений, совместный разбор типовых ошибок, взаимная экспертиза контрактов, кода и отчётов); обозначение границ правилосообразной (алгоритмической) и творческой деятельности (обязательные требования к контракту, корректности и воспроизводимости измерений при допускаемом выборе фреймворков и вариантов оптимизации при обосновании); обеспечение открытости обучения профессиональному будущему, ориентированность на практики разработки и эксплуатации API-сервисов (наблюдаемость, тестируемость, безопасность, воспроизводимость экспериментов).

В результате изучения дисциплины обучающиеся получают знания и навыки, необходимые для разработки API-сервисов на языке Python в составе ИИ-решений и информационных систем, включая проектирование контрактов, организацию обработки запросов и ошибок, интеграцию модулей обработки данных и инференса, обеспечение качества и производительности, а также выбор и применение библиотек и инструментов, соответствующих задачам прикладной разработки. Существенное внимание уделяется анализу качества решений по метрикам (p95/p99 задержки, пропускная способность, доля ошибок), корректной постановке экспериментов и интерпретации результатов для принятия инженерных решений.

Методика преподавания дисциплины строится на сочетании лекций с практическими занятиями; групповыми и индивидуальными консультациями по отдельным разделам программы; внеаудиторной самостоятельной работой обучающихся (работа с учебниками и учебными пособиями, методическими указаниями и заданиями, изучение специализированной литературы, поиск необходимой информации в сети Интернет, работа с официальной документацией библиотек и SDK). Самостоятельная работа ориентирована на освоение теоретических положений, подготовку к практическим занятиям, а также оформление отчётных материалов по тестированию и измерениям производительности, принятым архитектурным решениям и выбору инструментов.

Лекционный курс должен быть логичным и последовательным. Каждая лекция начинается с актуализации знаний и постановки цели занятия и задач, которые должны быть достигнуты в ходе изучения материала. Проведение лекций рекомендуется осуществлять на основе проблемного метода обучения, стимулирующего самостоятельный поиск решений и аргументацию выбора технологий и архитектурных подходов для Python API-сервисов. Для повышения интереса и обеспечения наглядности используются мультимедийные средства (презентации, схемы архитектур и потоков данных, примеры контрактов и фрагменты кода), а также элементы интерактивного обучения (обсуждение вариантов контрактов, сравнительный анализ архитектур, разбор инженерных

компромиссов latency/throughput/стоимость и качество обслуживания). В дополнение к традиционной лекции целесообразно применять проблемные лекции, лекции-визуализации, бинарные лекции (например, совмещение вопросов API и вопросов интеграции инференса/обработки данных), дискуссии по выбору фреймворка и подходов к оптимизации.

Важная роль на лекциях отводится дискуссии: обучающиеся рассматриваются как участники профессионального диалога, способные предлагать решения, аргументировать выбор и критически анализировать альтернативы. Каждая лекция завершается подведением итогов и формулировкой выводов, а также указанием материалов для самостоятельного изучения и подготовки к практическим занятиям.

Практические занятия строятся по аналогичной структуре: актуализация знаний, постановка цели и задач, выполнение работы, анализ полученных результатов и формулирование выводов. Практические работы должны соответствовать принципам контекстного подхода и включать исследовательские задачи профессиональной направленности: проектирование и документирование REST/gRPC контрактов, реализацию API-сервисов на Python с интеграцией инференса и компонент обработки данных, организацию конкурентной обработки запросов (асинхронная модель, настройка числа воркеров, применение очередей задач при необходимости), проведение нагрузочного тестирования и профилирования, оптимизацию по результатам измерений, подготовку к развертыванию и сопровождению. На практических занятиях рекомендуется применять технологии дифференцированного обучения, включая поддержку обучающихся, испытывающих затруднения, и расширенные задания для обучающихся с более высоким уровнем подготовки.

Практические занятия проводятся под руководством преподавателя и предусматривают анализ типовых ошибок, допущенных при выполнении заданий, и разбор наиболее удачных решений. Обучающиеся привлекаются к сравнительному анализу предложенных вариантов, обсуждают достоинства и недостатки, приобретают навыки ведения дискуссии и обоснования инженерных решений. Успех закрепления знаний и умений обеспечивается системой текущего контроля, включающей проверку результатов практических работ, защиту выполненных заданий и оценку отчётных материалов (контракты API, результаты тестирования и нагрузочных измерений, выводы профилирования и оптимизации).

В процессе самостоятельной работы обучающиеся закрепляют теоретические положения, изучают примеры, рассмотренные на практических занятиях, и выполняют индивидуальные задания, которые при возможности соотносятся с научными интересами обучающегося или тематикой выпускной квалификационной работы. Существенное значение имеет работа с литературой и официальными источниками (документация Python, FastAPI/Flask/Django REST Framework, OpenAPI, gRPC/Protocol Buffers, библиотеки машинного обучения и инференса, руководства по тестированию и нагрузочному тестированию, материалы по профилированию и оптимизации), а также анализ актуальных практик разработки и эксплуатации API-сервисов в прикладных информационных системах.

Особенности методики преподавания данной дисциплины состоят в интенсификации теоретической, практической и самостоятельной работы обучающихся и широком применении активных и интерактивных форм и методов обучения, ориентированных на решение профессионально значимых задач разработки, тестирования и оптимизации API-сервисов на Python в составе прикладных ИИ-решений и программных систем для бизнеса.

Программу разработал:

Лапшин М. С., ассистент 

РЕЦЕНЗИЯ

на рабочую программу дисциплины «_____»
ОПОП ВО по направлению *шифр* _____, направленность _____
(квалификация выпускника – бакалавр/специалист/магистр)

ФИО, должность, место работы, ученая степень (далее по тексту рецензент), проведена рецензия рабочей программы дисциплины «_____» ОПОП ВО по направлению *шифр* – **«_____»,** направленность **«_____»** (уровень обучения) разработанной в ФГБОУ ВО «Российский государственный аграрный университет – МСХА имени К.А. Тимирязева», на кафедре _____ (разработчик – ФИО, должность, ученая степень.

Рассмотрев представленные на рецензирование материалы, рецензент пришел к следующим выводам:

1. Предъявленная рабочая программа дисциплины «_____» (далее по тексту Программа) *соответствует* требованиям ФГОС ВО по направлению 09.03.03 Прикладная информатика, компетентностно-ролевым моделям в сфере искусственного интеллекта. Программа *содержит* все основные разделы, *соответствует* требованиям к нормативно-методическим документам.

2. Представленная в Программе ***актуальность*** учебной дисциплины в рамках реализации ОПОП ВО *не подлежит сомнению* – дисциплина относится к обязательной/формируемой участниками образовательных отношений части учебного цикла – Б1.

3. Представленные в Программе ***цели*** дисциплины *соответствуют* требованиям ФГОС ВО 09.03.03 Прикладная информатика, компетентностно-ролевым моделям в сфере искусственного интеллекта.

4. В соответствии с учебным планом и компетентностно-ролевыми моделями в сфере искусственного интеллекта, Программой за дисциплиной «_____» закреплено _____ ***компетенций***. Дисциплина «_____» и представленная Программа *способна реализовать* их в объявленных требованиях. Дополнительная **(если есть)** компетенция в соответствии с (*указать профессиональный стандарт или иное*). Результаты обучения, представленные в Программе в категориях знать, уметь, владеть *соответствуют* специфике и содержанию дисциплины и *демонстрируют возможность* получения заявленных результатов.

5. Общая трудоёмкость дисциплины «_____» составляет ___ зачётных единицы (___ часов/из них практическая подготовка____).

6. Информация о взаимосвязи изучаемых дисциплин и вопросам исключения дублирования в содержании дисциплин *соответствует* действительности. Дисциплина «_____» взаимосвязана с другими дисциплинами ОПОП ВО и Учебного плана по направлению *шифр* – _____ и возможность дублирования в содержании отсутствует.

7. Представленная Программа предполагает использование современных образовательных технологий, используемые при реализации различных видов учебной работы. Формы образовательных технологий *соответствуют* специфике дисциплины.

8. Программа дисциплины «_____» предполагает _____ занятий в интерактивной форме.

9. Виды, содержание и трудоёмкость самостоятельной работы студентов, представленные в Программе, *соответствуют* требованиям к подготовке выпускников, содержащимся во ФГОС ВО направления *шифр* _____.

10. Представленные и описанные в Программе формы *текущей* оценки знаний (опрос, как в форме обсуждения отдельных вопросов, так и выступления и участие в дискуссиях, диспутах, круглых столах, мозговых штурмах и ролевых играх, выполнение эссе, участие в тестировании, коллоквиумах, работа над домашним заданием в форме игрового проектирования (в профессиональной области) и аудиторных заданиях - работа с историческими текстами), *соответствуют* специфике дисциплины и требованиям к выпускникам.

Форма промежуточного контроля знаний студентов, предусмотренная Программой, осуществляется в форме экзамена/зачета с оценкой/зачета/защиты КР/КП, что *соответствует* статусу дисциплины, как дисциплины обязательной/вариативной части учебного цикла – Б1 ФГОС ВО направления *шифр* _____**.**

11. Формы оценки знаний, представленные в Программе, *соответствуют* специфике дисциплины и требованиям к выпускникам.

12. Учебно-методическое обеспечение дисциплины представлено: основной литературой – ____ источник (базовый учебник), дополнительной литературой – ___ наименований, периодическими изданиями – _____ источников со ссылкой на электронные ресурсы, Интернет-ресурсы – ____ источника и *соответствует* требованиям ФГОС ВО направления *шифр* _____**.**

13. Материально-техническое обеспечение дисциплины соответствует специфике дисциплины «_____» и обеспечивает использование современных образовательных, в том числе интерактивных методов обучения.

14. Методические рекомендации студентам и методические рекомендации преподавателям по организации обучения по дисциплине дают представление о специфике обучения по дисциплине «_____».

ОБЩИЕ ВЫВОДЫ

На основании проведенного рецензирования можно сделать заключение, что характер, структура и содержание рабочей программы дисциплины «_____» ОПОП ВО по направлению *шифр* _____, направленность **«**_____**»** (квалификация выпускника – бакалавр/специалист/магистр), разработанная ФИО, должность, ученая степень соответствует требованиям ФГОС ВО, компетентностно-ролевых моделей в сфере искусственного интеллекта, современным требованиям экономики, рынка труда и позволит при её реализации успешно обеспечить формирование заявленных компетенций.

Рецензент: Ивашова О. Н.,
доцент кафедры и организации
производства ФГБОУ ВО
«Российский государственный
аграрный университет – МСХА
имени К.А. Тимирязева», к.э.н. _____
(подпись)

«28» августа 2025 г.

(подпись)

[2] Рецензия рассмотрена на заседании кафедры
экономики и организации производства
28.08.2025 Протокол №1 _____ Худякова Е. В.
(подпись)

[2] Только для внешних рецензентов

УТВЕРЖДАЮ:
Директор института *(наименование)*
_____
“____”_____202__ г.

Лист актуализации рабочей программы дисциплины[3]

«_____»

индекс по учебному плану, наименование

для подготовки бакалавров/ специалистов/ магистров
Направление: {шифр – название}_____
Направленность:_____
Форма обучения _____
Год начала подготовки[4]:_____

Курс ____
Семестр ___
[5]а) В рабочую программу не вносятся изменения. Программа актуализирована для 20__ г. начала подготовки.
б) В рабочую программу вносятся следующие изменения (указать на какой год начала подготовки):

1) ……..;
2) ……..;
3) ……

Разработчик (и): _____
(ФИО, ученая степень, ученое звание)
«__» _____ 202_г.
Рабочая программа пересмотрена и одобрена на заседании кафедры _____
_____ протокол № _____ от «__» _____202__г.
Заведующий кафедрой _____

Заведующий выпускающей кафедрой *(наименование)* _____«__» _____ 202_г.

[3] Рабочая программа дисциплины актуализируется ежегодно перед началом нового учебного года.
[4] Указывается год начала подготовки актуализируемой РПД
[5] Разработчик выбирает один из представленных вариантов.