

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Б1.В.ДВ.03.01

(индекс дисциплины)

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Разработка веб-сервисов с интеграцией искусственного интеллекта

(наименование дисциплины)

по направлению подготовки
09.03.04 Программная инженерия

направленность (профиль)
Программная инженерия с применением ИИ-технологий

Форма обучения: заочная

Год набора: 2025

Общая трудоемкость: 5 ЗЕ

Распределение часов дисциплины по семестрам

Семестр	7	Итого
	экзамен	
Вид занятий	Форма контроля	
Лекции	6	6
Лабораторные		
Практические		
Руководство: курсовые работы (проекты) / РГР		
Промежуточная аттестация	0,35	0,35
Контактная работа	6,35	6,35
Самостоятельная работа	165	165
Контроль	8,65	8,65
Итого	180	180

Рабочую программу составил(и):

Доцент института цифровых технологий, к.п.н., Ерофеева Е.А.

(должность, ученое звание, степень, Фамилия И.О.)

Рецензирование рабочей программы дисциплины:



Отсутствует



Рецензент

(должность, ученое звание, степень, Фамилия И.О.)

Рабочая программа дисциплины составлена на основании ФГОС ВО и учебного плана направления подготовки

09.03.04 Программная инженерия

Срок действия рабочей программы дисциплины до «31» августа 2031 г.

УТВЕРЖДЕНО

На заседании института цифровых технологий

(протокол заседания № 1 от «05» сентября 2025 г.).

1. Цель освоения дисциплины

Цель освоения дисциплины – формирование у обучающихся профессиональных компетенций в области проектирования, разработки и интеграции веб-сервисов, использующих методы искусственного интеллекта; освоение технологий создания серверных и клиентских компонентов, взаимодействия с моделями машинного обучения, обеспечения корректности, надёжности и производительности веб-решений с ИИ-функциональностью.

2. Место дисциплины в структуре ОПОП ВО

Дисциплины и практики, на освоении которых базируется данная дисциплина: Тестирование и верификация систем с ИИ-компонентами, Инженерия программного обеспечения, Системы искусственного интеллекта, Архитектура информационных систем и методы интеграции.

Дисциплины и практики, для которых освоение данной дисциплины необходимо как предшествующее: Производственная практика (преддипломная практика), Выпускная квалификационная работа.

3. Планируемые результаты обучения

Формируемые и контролируемые компетенции (код и наименование)	Индикаторы достижения компетенций (код и наименование)	Планируемые результаты обучения
ПК-7 Способен разрабатывать сервисы на основе аналитики больших данных	ПК-7.1. Понимает принципы разработки сервисов, приемы аналитики больших данных	Знать: принципы разработки сервисов; методы аналитики (дескриптивная, диагностическая, предиктивная, предписывающая аналитика). Уметь: проектировать сервисно-ориентированную архитектуру. Владеть: навыками проектирования API для аналитических сервисов.
	ПК-7.2. Умеет разрабатывать сервисы на основе аналитики больших данных	Знать: как работать с потоковыми данными. Уметь: разрабатывать микросервисы, которые предоставляют аналитические функции (например, рекомендации, прогнозирование). Владеть: практическими навыками создания веб-сервисов с интеграцией аналитических библиотек.
	ПК-7.3. Умеет интегрировать модели искусственного интеллекта в веб-сервисы и обеспечивать их корректную работу в	Знать: принципы интеграции моделей ИИ в веб-архитектуру; особенности взаимодействия backend-сервисов с ML-модулями (API, RPC, контейнеризация, очереди); механизмы валидации и

Формируемые и контролируемые компетенции (код и наименование)	Индикаторы достижения компетенций (код и наименование)	Планируемые результаты обучения
	составе программной системы.	предобработки данных при работе с моделями. Уметь: разрабатывать API, взаимодействующие с ИИ-моделями (REST, gRPC); организовывать обмен данными между сервисом и моделью (сериализация, очереди, брокеры сообщений); внедрять модели ИИ в микросервисную архитектуру с учётом требований производительности и отказоустойчивости. Владеть: навыками деплоя и конфигурирования ИИ-компонентов в составе веб-сервиса (Docker, контейнеры, сервис-оркестраторы); инструментами мониторинга качества и доступности ИИ-модулей в продакшене; практическими методами отладки и эксплуатационной поддержки ИИ-интеграций (логирование, трассировка, профилирование).

4. Структура и содержание дисциплины

Модуль (раздел)	Вид учебной работы	Наименование тем занятий (учебной работы)	Семестр	Объем, ч.	Баллы	Интерактив, ч.	Формы текущего контроля (наименование оценочного средства)
Модуль 1. Архитектура веб-сервисов и основы интеграции ИИ	Лек 1	Тема 1. Архитектура современного веб-приложения: клиент, сервер, API, микросервисы	7	0,5		–	
	Лек 2	Тема 2. Принципы проектирования API. REST, gRPC, веб-хуки	7	0,5		–	
	Лек 3	Тема 3. Инфраструктура для веб-сервисов: контейнеризация, Docker, оркестраторы	7	0,5	4	–	Отчет по практической работе 1
	Лек 4	Тема 4. Способы интеграции ИИ-моделей в веб-сервисы: прямой вызов, сервис-модель, ML-node	7	0,5	4	–	Отчет по практической работе 2
	Пр 1	ПР1. Проектирование структуры веб-сервиса: модули, конфигурация, маршруты	7				
	Пр 2	ПР2. Создание минимального REST-сервиса (FastAPI / Flask / Spring)	7				
	Пр 3	ПР3. Работа с маршрутизацией и обработкой запросов	7				
	Пр 4	ПР4. Создание контейнера Docker для веб-сервиса	7				
Модуль 2. Интеграция	Лек 5	Тема 5. Модели ИИ как сервисы: REST-модели, RPC-модели, модельные контейнеры	7	0,5		–	

Модуль (раздел)	Вид учебной работы	Наименование тем занятий (учебной работы)	Семестр	Объем, ч.	Баллы	Интерактив, ч.	Формы текущего контроля (наименование оценочного средства)
моделей ИИ в веб-приложения	Лек 6	Тема 6. Предобработка данных и форматирование входов/выходов моделей	7	0,5			
	Лек 7	Тема 7. Схемы взаимодействия backend ↔ модель (Batch, realtime, streaming)	7	0,5			
	Лек 8	Тема 8. Паттерны интеграции ИИ- модуля в микросервисную архитектуру	7	0,5		–	
	Пр 5	ПР5. Подключение готовой ML-модели к веб-API	7		4	–	
	Пр 6	ПР6. Реализация эндпоинта /predict с использованием модели	7		4	–	
	Пр 7	ПР7. Предобработка данных: валидация, нормализация, сериализация	7		4	–	Отчет по практической работе 3
	Пр 8	ПР8. Интеграция модели в отдельный контейнер ML-node	7		6	–	Отчет по практической работе 4
Модуль 3. Хранение данных, очереди, асинхронность	Лек 9	Тема 9. Базы данных для веб-сервисов: реляционные и NoSQL	7	0,5		–	
	Лек 10	Тема 10. Асинхронные веб-сервисы, event-loop, background-tasks	7	0,5		–	
	Лек 11	Тема 11. Системы очередей и брокеры сообщений (RabbitMQ, Kafka)	7	0,5		–	
	Лек 12	Тема 12. Логирование, мониторинг, метрики и трассировка	7	0,5		–	

Модуль (раздел)	Вид учебной работы	Наименование тем занятий (учебной работы)	Семестр	Объем, ч.	Баллы	Интерактив, ч.	Формы текущего контроля (наименование оценочного средства)
	Пр 9	ПР9. Подключение базы данных, создание схемы и CRUD-операций	7		6	—	Отчет по практической работе 7
	Пр 10	ПР10. Асинхронный API: реализация фоновых задач	7		6	—	Отчет по практической работе 8
	Пр 11	ПР11. Связка сервис → очередь → модель	7		6	—	Отчет по практической работе 9
	Пр 12	ПР12. Логирование и мониторинг веб-сервиса	7		6	—	Отчет по практической работе 10
Модуль 4. Тестирование, безопасность и деплой веб-сервисов с ИИ	Лек 13	Тема 13. Тестирование API и ИИ-эндпоинтов: unit, integration, e2e	7			—	
	Лек 14	Тема 14. Проверка корректности моделей в составе сервиса	7			—	
	Лек 15	Тема 15. Безопасность веб-сервисов с ИИ, защита API, rate limiting	7			—	
	Лек 16	Тема 16. Деплой, CI/CD, автоматизация обновлений модели	7			—	
	Пр 13	ПР13. Тестирование API (Postman/pytest)	7		6	—	Отчет по практической работе 11
	Пр 14	ПР14. Проверка корректности модели в продакшен-сценариях	7		6	—	Отчет по практической работе 12
	Пр 15	ПР15. Реализация базовых механизмов безопасности (JWT, токены)	7		6	—	Отчет по практической работе 13
	Пр 16	ПР16. Деплой веб-сервиса и модели (Docker Compose / Kubernetes)	7		6	—	Отчет по практической работе 14

Модуль (раздел)	Вид учебной работы	Наименование тем занятий (учебной работы)	Семестр	Объем, ч.	Баллы	Интерактив, ч.	Формы текущего контроля (наименование оценочного средства)
	СР	Самостоятельное изучение методических рекомендаций при подготовке к практическим работам.	7	165	-		
	ПА	Промежуточная аттестация	7	0,35		—	
	Псц.		7		10		
	Контроль	экзамен	7	8,65	100		Итоговый тест
Итого:				180			

Схема расчета итогового балла: по накопительному рейтингу
Текущий рейтинг + Результат итогового теста и все делится на 2

5. Образовательные технологии

При изучении дисциплины используются следующие образовательные технологии:

- **технологии традиционного обучения** в форме лекций, практических работ и самостоятельной работы обучающихся;
- **технология проектного обучения**, предусматривающая разработку фрагментов веб-сервисов, интеграцию ИИ-модулей, защиту результатов практических работ и демонстрацию работоспособности созданных решений;
- **практико-ориентированное обучение**, основанное на выполнении реальных задач по разработке API, интеграции моделей машинного обучения, работе с контейнерами, очередями и базами данных.

Технологии традиционного обучения включают объяснительно-иллюстративный формат, применяемый во всех модулях курса при изучении архитектуры веб-сервисов, принципов интеграции ИИ и моделей взаимодействия компонентов.

Организация учебного процесса предполагает активную позицию обучающихся при выполнении практических заданий: разработку серверных модулей, реализацию ИИ-функциональности в составе веб-сервисов, настройку инфраструктуры, анализ корректности интеграции и представление полученных результатов.

На практических занятиях обучающиеся:

- демонстрируют работоспособность разработанных веб-сервисов и ИИ-модулей;
- обосновывают принятые архитектурные решения;
- обсуждают возникающие технические проблемы интеграции и пути их решения;
- проводят мини-защиту выполненной работы.

Итоговая аттестация проводится в форме **экзамена**, включающего оценку теоретических знаний и практических навыков разработки веб-сервисов с интеграцией искусственного интеллекта.

6. Методические указания по освоению дисциплины

6.1. Рекомендации по подготовке к практическим занятиям

Обучающимся следует:

- при подготовке к практическим занятиям использовать не только материалы лекций и учебную литературу, но и официальную документацию по используемым технологиям (фреймворки для веб-разработки, библиотеки ИИ, инструменты контейнеризации, системы управления БД);
- заранее познакомиться с примерами исходного кода, структурой API и форматами данных, применяемыми в задании;
- в начале занятия уточнить у преподавателя моменты, вызвавшие затруднения, чтобы избежать ошибок при реализации серверной логики или интеграции моделей ИИ;
- на занятии доводить каждое задание до работоспособного состояния, демонстрировать корректную работу разработанного сервиса, уметь пояснить принятые архитектурные решения.

Для того чтобы практические занятия приносили максимальную пользу, необходимо помнить, что выполнение заданий тесно связано с лекционным материалом: архитектурой веб-сервисов, методами интеграции моделей искусственного интеллекта, принципами обработки данных, методами деплоя и обеспечением работоспособности сервиса. Только после усвоения этих основ будет возможна качественная реализация серверных модулей, маршрутов API и ИИ-функциональности в практических работах.

При самостоятельной работе над заданиями необходимо:

- обосновывать каждый этап разработки (выбор архитектуры, схемы API, способ интеграции модели, структуру хранения данных);

- анализировать несколько вариантов решения (например, прямой вызов модели или вынесение её в отдельный сервис) и выбирать наиболее рациональный с точки зрения производительности, устойчивости и удобства поддержки;
- предварительно составлять план разработки: структуру каталогов проекта, точки входа, взаимодействующие компоненты, последовательность выполнения задач;
- сопровождать решения комментариями, диаграммами, схемами вызовов или архитектурными эскизами, если требуется представить логику работы сервиса.

Каждая практическая работа должна завершаться получением функционирующего результата — корректно работающего веб-сервиса или его модуля, готового к интеграции, тестированию или деплою. Полученный результат обучающийся должен проверить самостоятельно: протестировать основные маршруты API, корректность взаимодействия модуля ИИ, поведение сервиса при ошибочных данных.

Развитие устойчивых навыков разработки достигается регулярной самостоятельной практикой, анализом примеров, экспериментированием с архитектурой и изучением профессиональной документации.

6.2. Рекомендации по подготовке к экзамену

Подготовка к экзамену направлена на систематизацию, закрепление и обобщение теоретических знаний и практических навыков, полученных в ходе изучения дисциплины. Экзамен проверяет сформированность элементов компетенций, связанных с разработкой веб-сервисов, проектированием архитектуры, интеграцией моделей искусственного интеллекта и обеспечением работоспособности сервисов.

Для успешной подготовки обучающимся рекомендуется:
последовательно повторить ключевые темы лекционного курса:

- архитектура веб-приложений и принципы проектирования API;
- методы интеграции моделей искусственного интеллекта в веб-сервисы;
- основы контейнеризации и деплоя;
- работа с базами данных, очередями сообщений и асинхронными компонентами;
- обеспечение безопасности, логирование и мониторинг веб-сервисов;

проанализировать выполненные практические работы, уделив внимание:

- корректности структуры серверного приложения;
- способам передачи данных и форматам запросов/ответов;
- работе API-эндпоинтов и взаимодействию с ИИ-модулем;
- работе с контейнерами, очередями и базой данных;
- настройке окружения и использованию документации;

пересмотреть примеры интеграции ИИ-моделей, изученные в курсе:

- особенности предобработки данных;
- схемы взаимодействия сервиса и модели;
- возможные ошибки интеграции и способы их устранения;

систематизировать знания по архитектурным решениям, применяемым при построении веб-сервисов:

- проектирование микросервисов;
- разделение функциональности;
- типовые точки отказа и методы повышения устойчивости.

Подготовка к экзамену должна проводиться последовательно на протяжении всего семестра. Рекомендуется активно участвовать в обсуждениях на занятиях, выполнять все практические задания, анализировать примеры реализации и регулярно обращаться к документации используемых технологий. Такая работа формирует глубокое понимание материала и обеспечивает успешное прохождение экзамена.

7. Оценочные средства

7.1. Паспорт оценочных средств

Семестр	Код контролируемой компетенции (или ее части)	Наименование оценочного средства
7	ПК-7	Тестовые задания 1-450 Вопросы к экзамену 1-80 Отчеты по практическим работам 1-15

7.2. Типовые задания или иные материалы, необходимые для текущего контроля

7.2.1. Типовые тестовые материалы

(наименование оценочного средства)

Типовые примеры заданий

1. Какова основная роль API в веб-сервисе?

- A. Хранение данных на сервере
- B. Организация обмена данными между клиентом и сервером
- C. Оптимизация работы процессора
- D. Ускорение обучения модели

Правильный ответ: B

2. Что является ключевым преимуществом архитектуры микросервисов?

- A. Отсутствие необходимости в сервере
- B. Возможность независимого масштабирования компонентов
- C. Отказ от использования баз данных
- D. Гарантированная скорость работы сервиса

Правильный ответ: B

3. Какой формат чаще всего используется для передачи данных через REST-API?

- A. XML-RPC
- B. YAML
- C. JSON
- D. CSV

Правильный ответ: C

4. Какой компонент отвечает за контейнеризацию приложения?

- A. Git
- B. Docker
- C. Selenium
- D. Webpack

Правильный ответ: B

5. Какой способ интеграции ИИ-модели наиболее устойчив при высокой нагрузке?

- A. Встраивание модели прямо в фронтенд
- B. Хранение модели в виде текстового файла
- C. Выделение модели в отдельный сервис (ML-node)
- D. Запуск модели в браузере пользователя

Правильный ответ: C

6. Какая операция выполняется при предобработке данных для ИИ-модели?

- A. Компиляция приложения
- B. Сериализация объектов
- C. Нормализация и очистка входных данных
- D. Рендеринг интерфейса

Правильный ответ: C

7. Какой статус HTTP соответствует успешному выполнению запроса?

- A. 200
- B. 301
- C. 404
- D. 500

Правильный ответ: A

8. Что обеспечивает брокер сообщений (Kafka, RabbitMQ) в веб-сервисе?

- A. Автоматическую генерацию пользовательского интерфейса
- B. Асинхронный обмен данными между сервисами
- C. Хранение моделей машинного обучения
- D. Тестирование API

Правильный ответ: B

9. Какой инструмент используется для документирования REST-API?

- A. Swagger / OpenAPI
- B. LaTeX
- C. Figma
- D. Blender

Правильный ответ: A

10. Что означает масштабирование «по горизонтали»?

- A. Увеличение мощности одного сервера
- B. Оптимизацию SQL-запросов
- C. Добавление новых экземпляров сервисов
- D. Уменьшение размера Docker-образа

Правильный ответ: C

11. Что является преимуществом использования Docker Compose?

- A. Ускорение обучения модели
- B. Управление несколькими контейнерами как единой системой
- C. Автоматическая генерация API
- D. Создание интерфейса веб-приложения

Правильный ответ: B

12. Что проверяет end-to-end тестирование в веб-сервисе?

- A. Производительность процессора
- B. Корректность работы модели на GPU
- C. Взаимодействие всех компонентов системы целиком
- D. Работу CSS-стилей

Правильный ответ: C

13. Что обеспечивает JWT в веб-сервисе?

- A. Обучение нейронных сетей
- B. Контейнеризацию сервисов

- C. Маршрутизацию HTTP-запросов
- D. Авторизацию и безопасность

Правильный ответ: D

14. Что является типичной точкой отказа при интеграции ИИ-модели?

- A. Ошибки в CSS
- B. Несоответствие формата входных данных
- C. Неправильный выбор шрифта
- D. Наличие изображений большого размера

Правильный ответ: B

15. Какой подход чаще всего используется для деплоя веб-сервисов с ИИ-моделями?

- A. Ручная загрузка файлов на сервер
- B. CI/CD пайплайн с автоматическим обновлением сервисов
- C. Использование ZIP-архива
- D. Деплой только на локальную машину

Правильный ответ: B

7.2.2. Пример практической работы

Практическая работа 1. Проектирование структуры веб-сервиса: модули, конфигурация, маршруты

Цель работы

Освоение базовых принципов проектирования архитектуры веб-сервиса, включая:

- выделение основных модулей приложения;
- формирование структуры конфигурации;
- проектирование API и маршрутов;
- понимание логики взаимодействия компонентов веб-приложения.

Порядок выполнения работы

Определение тематики и назначения веб-сервиса

Необходимо указать:

- какое назначение имеет разрабатываемый веб-сервис;
- кто является предполагаемым пользователем;
- какие данные сервис принимает и возвращает.

Формирование архитектуры проекта

Следует выделить ключевые модули веб-приложения. Примерный набор модулей:

Модуль	Назначение
config	параметры конфигурации приложения
routes	обработчики HTTP-запросов
services	бизнес-логика
models	схемы данных запросов и ответов
ml	взаимодействие с моделью ИИ (при наличии)
db	работа с базой данных (если используется)
utils	вспомогательные функции

Каждый модуль необходимо кратко описать.

Проектирование API и маршрутов

Необходимо разработать **не менее 5 API-маршрутов**. Для каждого маршрута указываются:

Маршрут	Метод	Входные параметры	Выходные данные	Назначение
---------	-------	-------------------	-----------------	------------

Примеры маршрутов:

- GET /api/health — проверка доступности сервиса;
- GET /api/items — получение списка объектов;
- GET /api/items/{id} — получение объекта по идентификатору;
- POST /api/items — добавление объекта;
- POST /api/predict — выполнение предсказания (если планируется ИИ-функциональность).

Описание конфигурации приложения

В конфигурации требуется указать:

- порт приложения;
- используемое окружение: dev, prod;
- путь к модели ИИ (если используется);
- строку подключения к базе данных;
- переменные окружения.

Пример:

```
APP_PORT=8000
ENV=development
ML_MODEL_PATH=models/model.pkl
DB_URL=postgresql://user:pass@localhost:5432/mydb
```

Представление архитектурной схемы

Архитектура может быть описана в текстовой или графической форме.

Пример текстовой схемы:

```
Клиент → Маршрутизатор → Обработчик → Сервисный слой
                                   → (База данных)
                                   → (Модель ИИ)
```

Состав отчёта

В отчёт включаются:

1. название работы;
2. цель работы;
3. функциональное описание сервиса;
4. таблица модулей приложения;
5. таблица маршрутов API;
6. описание конфигурации;
7. архитектурная схема;
8. выводы (3–5 пунктов).

Вопросы к защите

1. Что включает архитектура веб-сервиса?
2. Чем отличается модуль маршрутизации от модуля бизнес-логики?
3. Зачем необходимо разделение проекта на модули?
4. Какие обязательные элементы содержит API-маршрут?
5. Чем отличаются path-параметры от query-параметров?
6. Какие параметры обычно входят в конфигурацию приложения?
7. Каковы преимущества единообразной структуры проекта?
8. Какова последовательность обработки HTTP-запроса?
9. Почему важно заранее проектировать структуру данных?

10. Какие проблемы возникают при отсутствии архитектуры на ранних этапах разработки?
11. Какое назначение имеет маршрут /api/health?
12. Что следует учитывать при проектировании API для будущей интеграции модели ИИ или базы данных?

Требования к оформлению

Отчет должен содержать подробное описание (включая иллюстрации). Отчёт по практическому занятию выполняется на страницах формата А4 в электронном виде.

При оформлении отчёта используется сквозная нумерация страниц, считая титульный лист первой страницей. Номер страницы на титульном листе не ставится. Номера страницы ставятся по центру сверху.

При оформлении отчёта соблюдать следующие требования:

- Для заголовков: полужирный шрифт, 14 пт, центрированный.
- Для основного текста: нежирный шрифт, 14 пт, выравнивание по ширине.
- Во всех случаях тип шрифта – Times New Roman, отступ абзаца 1.25 см, полуторный междустрочный интервал.
- Поля: левое – 2 см, правое, верхнее и нижнее – 1 см.

Процедура оценивания

Оценка выполненной практической работы проводится по следующим критериям:

1. Наличие всей существенной информации по работе
2. Точность и полнота предоставляемых сведений
3. Непротиворечивость приводимой информации
4. Правильность интерпретаций и выводов, которые сделаны по результатам работы
5. Степень достижения обучающимся поставленной цели
6. Обоснованность применяемого решения
7. Грамотность (содержательная) используемых формулировок

Критерии оценки за отчеты по практическим работам:

Формы текущего контроля	Критерии и нормы оценки
Отчеты по практическим работам 1, 2, 3	4 балла – задание выполнено в полном объёме без замечаний 3 балла – задание выполнено в полном объёме, присутствуют замечания 2 баллов – задание выполнено в объёме 70%, замечаний нет. 1 балл – задание выполнено в объёме менее 50%, присутствуют замечания. 0 баллов – задание не выполнено.
Отчеты по практическим работам 4 - 16	6 баллов – задание выполнено в полном объёме без замечаний 5 баллов – задание выполнено в полном объёме, присутствуют замечания 4 балла – задание выполнено в объёме 70%, замечаний нет. 3 балла – задание выполнено в объёме 50%, замечаний нет. 2 балла - задание выполнено в объёме менее 50%, замечаний нет. 1 балл – задание выполнено в объёме менее 50%, присутствуют замечания.

	0 баллов – задание не выполнено.
--	----------------------------------

Типовые задания для итогового теста

1. Какой компонент веб-сервиса отвечает за обработку входящего HTTP-запроса?

- A. Frontend
- B. API-маршрут (endpoint)
- C. База данных
- D. Очередь сообщений

Правильный ответ: B

2. Какие из перечисленных относятся к методам интеграции модели ИИ в веб-сервис?

- A. Встраивание модели прямо в backend
- B. Выделение модели в отдельный сервис (ML-node)
- C. Запуск модели на стороне пользователя
- D. Контейнеризация модели

Правильные ответы: A, B, D

3. Соотнесите HTTP-метод и его назначение

- 1 — GET
- 2 — POST
- 3 — PUT
- 4 — DELETE
- A — Создание нового ресурса
- B — Удаление ресурса
- C — Обновление существующего ресурса
- D — Получение данных

Правильный ответ:

1–D, 2–A, 3–C, 4–B

4. Упорядочьте этапы обработки запроса веб-сервисом

- 1 — Обработка маршрутизатором
- 2 — Формирование ответа
- 3 — Валидация входных данных
- 4 — Вызов бизнес-логики / модели

Правильная последовательность:

1 → 3 → 4 → 2

5. Какой формат данных чаще всего используется в REST-API?

- A. XML
- B. CSV
- C. JSON
- D. PDF

Правильный ответ: C

6. Для чего используется контейнеризация (Docker) в веб-разработке?

- A. Для создания дизайна интерфейса
- B. Для изоляции окружения и переноса приложения
- C. Для ускорения обучения модели
- D. Для анализа логов

Правильный ответ: B

7. Какие из перечисленных являются преимуществами микросервисной архитектуры?

- A. Независимое масштабирование компонентов
- B. Уменьшение числа используемых технологий
- C. Изоляция сбоев
- D. Упрощение CI/CD

Правильные ответы: A, C, D

8. Какой статус HTTP говорит об ошибке сервера?

- A. 200
- B. 301
- C. 404
- D. 500

Правильный ответ: D

9. Соотнесите компонент системы и его назначение

- 1 — Брокер сообщений (RabbitMQ, Kafka)
- 2 — База данных
- 3 — ML-сервис
- 4 — API-сервер
- A — Хранение структурированных данных
- B — Асинхронный обмен событиями
- C — Выполнение предсказаний модели
- D — Приём и обработка HTTP-запросов

Правильный ответ:

1–B, 2–A, 3–C, 4–D

10. Что обеспечивает JWT-токен в веб-сервисе?

- A. Хеширование пароля
- B. Авторизацию и подтверждение личности
- C. Генерацию логов
- D. Сжатие данных

Правильный ответ: B

11. Какие параметры относятся к конфигурации веб-сервиса?

- A. Порт запуска приложения
- B. URL базы данных
- C. Размер шрифта интерфейса
- D. Путь к ML-модели

Правильные ответы: A, B, D

12. Упорядочьте этапы деплоя веб-сервиса

- 1 — Сборка Docker-образа
- 2 — Настройка переменных окружения
- 3 — Запуск контейнера
- 4 — Проверка работоспособности сервиса

Правильная последовательность:

1 → 2 → 3 → 4

13. Что является обязательным элементом документации API?

- A. Цветовая схема сайта
- B. Примеры запросов и ответов
- C. Логотип сервиса
- D. История изменений интерфейса

Правильный ответ: B

14. Какие ошибки чаще всего возникают при интеграции модели ИИ в сервис?

- A. Неверный формат входных данных
- B. Неправильная компоновка CSS
- C. Конфликт маршрутов на фронтенде
- D. Несоответствие типов данных в предобработке

Правильные ответы: A, D

15. Соотнесите тип хранения данных и пример его использования

- 1 — Реляционная БД
- 2 — NoSQL (документная)
- 3 — In-мемогу хранилище
- 4 — Файловое хранилище
- A — Хранение конфигураций, логов модели
- B — Хранение больших JSON-документов
- C — Хранение пользовательских записей и транзакций
- D — Кэширование запросов

Правильный ответ:

1–C, 2–B, 3–D, 4–A

7.3. Оценочные средства для промежуточной аттестации по итогам освоения дисциплины

7.3.1. Вопросы к промежуточной аттестации

Семестр 7

№	Вопросы к экзамену
1.	Дайте определение Jakarta EE и перечислите ключевые спецификации, используемые при разработке веб-сервисов.
2.	Объясните назначение контейнера сервлетов и его роль в архитектуре Jakarta EE.
3.	Перечислите основные компоненты веб-приложения Jakarta EE и охарактеризуйте их.
4.	Опишите жизненный цикл сервлета и его влияние на обработку HTTP-запросов.
5.	Объясните назначение слоистой архитектуры (Controller → Service → Repository) в Jakarta EE.
6.	Раскройте понятие "enterprise-уровня" приложений и их особенности по сравнению с обычными веб-сервисами.

№	Вопросы к экзамену
7.	Охарактеризуйте роль Jakarta RESTful Web Services (JAX-RS) при создании API.
8.	Перечислите способы маршрутизации HTTP-запросов в JAX-RS и принципы их работы.
9.	Объясните работу аннотаций JAX-RS (@GET, @POST, @Path, @Produces, @Consumes).
10.	Охарактеризуйте роль Jakarta Servlet API в работе веб-приложения.
11.	Объясните разницу между Jakarta JSON-B и Jakarta JSON-P и сферы их применения.
12.	Опишите процесс сериализации и десериализации объектов в Jakarta JSON-B.
13.	Объясните назначение DTO и их роль в веб-приложениях.
14.	Раскройте важность валидации входных данных и перечислите механизмы Jakarta Bean Validation.
15.	Объясните, что такое схема данных API (API contract) и для чего она применяется.
16.	Интеграция моделей ИИ в Jakarta-приложения
17.	Перечислите подходы к интеграции внешних ML/AI-моделей через HTTP в Jakarta EE.
18.	Объясните особенности реализации клиентских вызовов к ML-модулю с использованием JAX-RS Client API.
19.	Опишите этапы подготовки входных данных в Jakarta-сервисе перед передачей модели.
20.	Перечислите типичные ошибки взаимодействия Jakarta-бэкенда с внешним ML-сервисом.
21.	Объясните назначение выделенного ML-сервиса и преимущества его использования.
22.	Охарактеризуйте работу Jakarta Concurrency API при фоновой обработке AI-задач.
23.	Опишите архитектурные схемы вызова модели ИИ из Jakarta-приложения.
24.	Объясните необходимость обработки ошибок модели на стороне веб-сервиса.
25.	Раскройте требования к форматам данных при интеграции Jakarta-сервиса с ML-моделью.
26.	Дайте определение Jakarta Persistence (JPA) и перечислите основные компоненты ORM.
27.	Объясните назначение EntityManager и его роль в работе с базой данных.
28.	Перечислите типичные шаблоны работы репозитория в Jakarta EE.
29.	Объясните принципы транзакционности и роль Jakarta Transaction API.
30.	Охарактеризуйте использование JPQL и его отличия от SQL.
31.	Опишите подходы к оптимизации JPA-запросов в веб-сервисах.
32.	Объясните назначение Jakarta Messaging (JMS) и его использование в распределённых системах.
33.	Перечислите основные модели доставки сообщений JMS.
34.	Опишите архитектуру веб-приложения, использующего JMS для обработки AI-задач.
35.	Раскройте роль Message-driven beans (MDB) при работе с очередями.
36.	Объясните преимущества использования брокера сообщений для интеграции ИИ.
37.	Перечислите основные механизмы Jakarta Security.

№	Вопросы к экзамену
38.	Объясните принципы работы фильтров безопасности в веб-приложениях.
39.	Охарактеризуйте роли и принципы ролевой модели безопасности в Jakarta EE.
40.	Объясните назначение HTTP-аутентификации и методов её реализации.
41.	Раскройте понятие CORS и опишите его конфигурацию в приложении Jakarta EE.
42.	Объясните, как защищать REST-эндпоинты от несанкционированного доступа.
43.	Охарактеризуйте методы защиты от атак SQL-injection и XSS в Java-приложениях.
44.	Дайте определение Docker-контейнеру и опишите его преимущества в разработке Jakarta-приложений.
45.	Опишите структуру Dockerfile для Java/Jakarta-сервиса.
46.	Перечислите типичные ошибки при контейнеризации Jakarta-приложений.
47.	Объясните назначение Docker Compose при работе с микросервисной системой.
48.	Раскройте особенности деплоя Jakarta-сервисов в Kubernetes.
49.	Объясните подходы к контейнеризации ML-модулей и их интеграции с Jakarta-бэкендом.
50.	Объясните назначение журналирования с использованием Jakarta Logging.
51.	Перечислите подходы к централизованному сбору логов Jakarta-сервисов.
52.	Охарактеризуйте принципы работы систем APM (Application Performance Monitoring).
53.	Опишите виды метрик, используемых для мониторинга работы веб-сервиса.
54.	Объясните назначение трассировки запросов в распределённых сервисах.
55.	Перечислите подходы к оптимизации производительности Jakarta-приложений.
56.	Объясните разницу между горизонтальным и вертикальным масштабированием.
57.	Охарактеризуйте факторы, влияющие на время ответа JAX-RS-эндпоинтов.
58.	Опишите способы оптимизации сетевых вызовов к внешним ML-сервисам.
59.	Объясните назначение кэширования результатов модели и его варианты.
60.	Оцените влияние асинхронности на производительность веб-сервиса.
61.	Дайте определение CI/CD в контексте Jakarta EE.
62.	Объясните этапы автоматической сборки Jakarta-приложения.
63.	Перечислите инструменты для CI/CD в Java/Jakarta-экосистеме.
64.	Опишите разницу между dev, staging и production окружениями.
65.	Приведите этапы деплоя Jakarta-приложения в контейнеризированной среде.
66.	Объясните назначение health-checks в оркестрации сервисов.
67.	Объясните роль спецификации OpenAPI при разработке Jakarta-REST-сервисов.
68.	Перечислите требования к качественной документации API.
69.	Опишите этапы тестирования REST-эндпоинтов в Jakarta EE.
70.	Охарактеризуйте назначение интеграционных тестов при работе Jakarta-сервиса с ML-модулем.

№	Вопросы к экзамену
71.	Оцените важность контрактного тестирования при взаимодействии микросервисов.
72.	Раскройте принципы чистой архитектуры (Clean Architecture) применительно к Jakarta-сервисам.
73.	Перечислите правила разделения ответственности между слоями приложения.
74.	Объясните понятие “точка отказа” в архитектуре и способы её устранения.
75.	Опишите подходы к проектированию API для взаимодействия с ИИ-модулем.
76.	Раскройте преимущества использования CDI (Contexts and Dependency Injection).
77.	Объясните, как внедрение зависимостей упрощает архитектуру веб-приложения.
78.	Опишите жизненный цикл HTTP-запроса в приложении Jakarta EE.
79.	Объясните назначение CORS в сценариях фронтенд ↔ backend.

7.3.2. Критерии и нормы оценки

Семестр	Форма проведения промежуточной аттестации	Критерии и нормы оценки	
7	Экзамен (по накопительному рейтингу)	«отлично»	рейтинговый балл 85-100
		«хорошо»	рейтинговый балл 70-84
		«удовлетворительно»	рейтинговый балл 55-69
		«неудовлетворительно»	рейтинговый балл 0-54

8. Учебно-методическое и информационное обеспечение дисциплины

8.1. Обязательная литература

№ п/п	Авторы, составители	Заглавие (заголовок)	Тип (учебник, учебное пособие, учебно-методическое пособие, практикум, др.)	Год издания	Количество в научной библиотеке / Наименование ЭБС
1.	Аншина М. Л.	Аншина М. Л. Архитектура приложений и данных [Электронный ресурс] : учебное пособие / М. Л. Аншина. — М. : МИРЭА — Российский технологический университет, 2024.	учебное пособие	2025	ЭБС «Лань»
2.	Турнецкая, Е. Л.	Тестирование и контроль качества программного обеспечения : учебное пособие / Е. Л. Турнецкая, А. В. Аграновский, А. А. Сенцов. — Санкт-Петербург : ГУАП, 2023. — 118 с. — ISBN 978-5-8088-1891-0. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/461498 (дата обращения: 28.11.2025)	учебное пособие	2023	ЭБС «Лань»
3.	Ю. А. Антохина, М. Л. Кричевский, Ю. А. Мартынова, А. А. Оводенко	Искусственный интеллект. Инноватика : учебное пособие / Ю. А. Антохина, М. Л. Кричевский, Ю. А. Мартынова, А. А. Оводенко. — Санкт-Петербург : ГУАП, 2023. — 320 с. — ISBN 978-5-8088-1830-9. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/341003 (дата обращения: 28.11.2025).	учебное пособие	2023	ЭБС «Лань»
4.	Баланов, А. Н.	Машинное обучение и искусственный интеллект : учебное пособие для вузов / А. Н. Баланов. — 2-е изд., стер. — Санкт-Петербург : Лань, 2025. — 172 с. — ISBN 978-5-507-52891-2. — Текст : электронный // Лань : электронно-библиотечная	Учебное пособие	2025	ЭБС «Лань»

№ п/п	Авторы, составители	Заглавие (заголовок)	Тип (учебник, учебное пособие, учебно- методическое пособие, практикум, др.)	Год издания	Количество в научной библиотеке / Наименование ЭБС
		система. — URL: https://e.lanbook.com/book/462248 (дата обращения: 28.11.2025).			
5.	Маран, М. М.	Маран, М. М. Программная инженерия : Учебное пособие для вузов / М. М. Маран. — 3-е изд., стер. — Санкт-Петербург : Лань, 2022. — 196 с. — ISBN 978-5-8114-9323-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/189470 (дата обращения: 28.11.2025).	Учебное пособие	2022	ЭБС «Лань»

8.2. Дополнительная литература

№ п/п	Авторы, составители	Заглавие (заголовок)	Тип (учебник, учебное пособие, учебно- методическое пособие, практикум, др.)	Год издания	Количество в научной библиотеке / Наименование ЭБС
1.	Бубнов, А. А.	Бубнов, А. А. Тестирование программного обеспечения : учебное пособие / А. А. Бубнов, С. А. Бубнов, В. В. Тишкина. — Рязань : РГРТУ, 2024. — 164 с. — ISBN 978-5-7722-0421-4. — Текст : электронный // Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/494540 (дата обращения: 28.11.2025).	учебное пособие	2024	ЭБС «IPRbooks»
2.	Турнецкая, Е. Л.	Турнецкая, Е. Л. Программная инженерия. Тестирование и контроль	учебное пособие	2025	ЭБС «IPRbooks»

№ п/п	Авторы, составители	Заглавие (заголовок)	Тип (учебник, учебное пособие, учебно- методическое пособие, практикум, др.)	Год издания	Количество в научной библиотеке / Наименование ЭБС
		качества программного обеспечения : учебное пособие для вузов / Е. Л. Турнецкая, А. В. Аграновский. — Санкт- Петербург : Лань, 2025. — 172 с. — ISBN 978-5-507-51677-3. — Текст : электронный // Лань : электронно- библиотечная система. — URL: https://e.lanbook.com/book/455672 (дата обращения: 28.11.2025).			
3.	Романов, Е. Л.	Романов, Е. Л. Программная инженерия : учебное пособие / Е. Л. Романов. — Новосибирск : НГТУ, 2017. — 395 с. — ISBN 978-5-7782-3455-0. — Текст : электронный // Лань : электронно- библиотечная система. — URL: https://e.lanbook.com/book/118221 (дата обращения: 28.11.2025).	учебное пособие	2017	ЭБС «IPRbooks»

8.3. Перечень профессиональных баз данных и информационных справочных систем

№	Наименование	Ссылка
1	Jakarta EE Specifications & API – официальные спецификации, документация, примеры (полностью бесплатно)	https://jakarta.ee/specifications/
2	Eclipse Foundation – репозитории и документация по серверам Jakarta (GlassFish, Payara, WildFly)	https://www.eclipse.org/
3	GitHub – открытые проекты по Java, Jakarta EE, REST-API, ML-сервисам, интеграции моделей	https://github.com/
4	GitHub Awesome Lists – curated-подборки инструментов по Java, AI, microservices	https://github.com/sindresorhus/awesome
5	OpenAPI Specification – официальная документация и примеры описания REST-API	https://swagger.io/specification/
6	OWASP – бесплатные стандарты и рекомендации по безопасности веб-приложений и API	https://owasp.org/
7	Docker Hub – публичные контейнеры для Jakarta, Java, ML-моделей, СУБД	https://hub.docker.com/
8	Kaggle Datasets – полностью бесплатные датасеты для интеграции в ML-сервисы	https://www.kaggle.com/datasets
9	Papers With Code – бесплатные ML-модели, REST-примеры, бенчмарки	https://paperswithcode.com/
10	DevDocs – бесплатная оффлайн/онлайн документация по Java, HTTP, JSON, Docker и др.	https://devdocs.io/

8.4. Перечень программного обеспечения

№ п/п	Наименование ПО	Реквизиты лицензии / договора
1	OpenJDK 17 / 21 (реализация Java)	Лицензия: GNU GPL v2 with Classpath Exception (свободное ПО)
2	Jakarta EE (GlassFish / Payara / WildFly) — сервер приложений	GlassFish: CDDL + GPL (свободное ПО); Payara Community Edition: Apache License 2.0; WildFly: LGPL v2.1
3	Maven (система сборки)	Лицензия: Apache License 2.0 (свободное ПО)
4	Postman (инструмент для тестирования REST-API)	Бесплатная версия для образовательного и личного использования
5	Docker Desktop / Docker Engine (контейнеризация)	Лицензия: Apache License 2.0, бесплатная версия для обучения
6	Docker Compose	Лицензия: Apache License 2.0 (свободное ПО)
7	Git (система контроля версий)	Лицензия: GNU GPL v2 (свободное ПО)
8	GitHub (web-интерфейс)	Бесплатный доступ для образовательного использования (Free Tier)

№ п/п	Наименование ПО	Реквизиты лицензии / договора
9	H2 / PostgreSQL (БД для разработки и тестирования)	H2 — MPL 2.0 (free), PostgreSQL — PostgreSQL License (свободное ПО)
10	OpenAPI / Swagger UI (генерация и документация API)	Лицензия: Apache License 2.0 (свободное ПО)
11	curl / httpie (утилиты для работы с HTTP-запросами)	curl — MIT License; httpie — BSD License
12	VS Code / IntelliJ IDEA Community Edition (IDE)	VS Code — MIT License; IntelliJ CE — Apache License 2.0

8.5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине

№ п/п	Наименование оборудованных учебных кабинетов, лабораторий, мастерских и др. объектов для проведения практических и лабораторных занятий, помещений для самостоятельной работы обучающихся (номер аудитории)	Перечень основного оборудования
1	Компьютерный класс. Учебная аудитория для проведения занятий лекционного типа. Учебная аудитория для проведения занятий семинарского типа. Учебная аудитория для проведения лабораторных работ. Учебная аудитория для курсового проектирования (выполнения курсовых работ). Учебная аудитория для проведения групповых и индивидуальных консультаций. Учебная аудитория для проведения занятий текущего контроля и промежуточной аттестации. (УЛК-401).	Компьютер (монитор 19", системный блок Pentium (R) Dual-Core E5500 2,8 GHz / 4 Gb / 500 Gb), столы ученические, столы компьютерные, стол преподавательский, стулья, доска аудиторная (меловая).
2	Помещение для самостоятельной работы обучающихся (УЛК-105).	Столы, стулья, стеллажи (в т.ч. выставочные) с книгами, компьютеры, мобильные рабочие места.
3	Помещение для самостоятельной работы обучающихся (УЛК-406).	Столы компьютерные, стулья, микрокомпьютеры raspberry pi 32 bit.