Документ подписан простой электронной подписью
Информация о владельце:
ФИО: Гриб Владислав Валерьевич
Должность: Ректор
Дата подписания: 06.11.2026 17:39:38
Уникальный программный ключ:
637517d24e103c3db032acf37e839d98e4c5bb2f5eb89c29abfcd7f43985447

Образовательное частное учреждение высшего образования
«МОСКОВСКИЙ УНИВЕРСИТЕТ ИМЕНИ А.С. ГРИБОЕДОВА»
(ИМПЭ им. А.С. Грибоедова)
ФАКУЛЬТЕТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
(МЕЖДУНАРОДНЫЙ ДИПЛОМАТИЧЕСКИЙ КОЛЛЕДЖ)

УТВЕРЖДАЮ
Директор Международного
дипломатического колледжа
_____ Н.И. Платонова
«16» февраля 2026г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по учебной дисциплине

Основы проектирования информационных систем

Специальность
09.02.13 Интеграция решений с применением технологий искусственного интеллекта

Квалификация выпускника:
специалист по работе с искусственным интеллектом

на базе основного общего образования

Форма обучения: очная

Москва

Фонд оценочных средств для дисциплины «Основы проектирования информационных систем». Специальность 09.02.13 Интеграция решений с применением технологий искусственного интеллекта квалификация: специалист по работе с искусственным интеллектом /– М.: ИМПЭ им. А. С. Грибоедова.

Фонд оценочных средств является неотъемлемой частью рабочей программы дисциплины.

Организация-разработчик:	Образовательное частное учреждение высшего образования «Московский университет имени А. С. Грибоедова»

Директор Международного
дипломатического колледжа_____/Н.И. Платонова

1. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ТЕКУЩЕГО КОНТРОЛЯ УСПЕВАЕМОСТИ, ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ ОБУЧАЮЩИХСЯ ПО ДИСЦИПЛИНЕ

Настоящий Фонд оценочных средств (ФОС) по дисциплине «Основы проектирования информационных систем» является неотъемлемым приложением к рабочей программе дисциплины (РПД) «Основы проектирования информационных систем». На данный ФОС распространяются все реквизиты утверждения, представленные в РПД по данной дисциплине.

2. ПЕРЕЧЕНЬ ОЦЕНОЧНЫХ СРЕДСТВ

Для определения качества освоения обучающимися учебного материала по дисциплине используются следующие оценочные средства:

№ п/п	Оценочное средство	Краткая характеристика оценочного средства	Представление оценочного средства в ФОС
1	Тестирование	Вид контроля, позволяющий оценить изученный теоретический материал.	Вопросы для проведения тестирования
2	Практические задания	Вид контроля, позволяющий оценить умение обучающегося применять осваиваемую компетенцию в практических ситуациях и при решении производственных задач	Задания к практическому занятию
3	Контрольная работа	Вид контроля, позволяющий определить результат освоения компетенций по дисциплине в рамках рассматриваемой темы, оцениваемый с помощью соответствующих индикаторов достижения компетенций	Задания контрольной работы
4	Самостоятельная работа	Вид контроля, позволяющий оценить проработку теоретического материала, изучение рекомендуемой литературы, выполнение практико-ориентированных заданий (заполнение таблиц, проведение сравнительного анализа, составление схем и др.), решение практических задач, создание презентаций, написание рефератов, подборку нормативного и иного материала и выполнение других заданий	Задания самостоятельной работы
5.	Зачет/Зачет с оценкой/Экзамен	Вид контроля, позволяющий выявить степень овладения знаниями, умениями и навыками, необходимыми для дальнейшего освоения образовательной программы подготовки	Вопросы для подготовки к зачету/зачету с оценкой/экзамену

3.ТИПОВЫЕ КОНТРОЛЬНЫЕ ЗАДАНИЯ ИЛИ ИНЫЕ МАТЕРИАЛЫ, НЕОБХОДИМЫЕ ДЛЯ ОЦЕНКИ РЕЗУЛЬТАТОВ ОБУЧЕНИЯ ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

Темы для подготовки к промежуточной аттестации

1. Информационная система: определение, классификация, основные компоненты. Примеры ИС с интегрированными компонентами ИИ.
2. Жизненный цикл программного обеспечения: основные модели (каскадная, Vобразная, итеративная, спиральная). Их преимущества, недостатки и область применения.
3. Гибкие методологии разработки (Agile): манифест, принципы, фреймворки (Scrum, Kanban). Особенности применения в проектах с элементами ИИ.
4. Особенности жизненного цикла AI-насыщенных систем. Концепция MLOps и ее отличие от классического DevOps.
5. Стандарты в области проектирования ИС: ISO/IEC/IEEE 12207, ГОСТы серии 34 (обзор, назначение).
6. Системный и бизнес-анализ в проектировании ИС. Роль аналитика. Стейкхолдеры и источники требований.
7. Виды требований: функциональные, нефункциональные (атрибуты качества). Особенности формулировки требований к AI-компонентам (данные, метрики, этика).
8. Техники сбора и документирования требований (интервью, мозговой штурм, пользовательские истории, сценарии).
9. Язык моделирования UML: назначение, виды диаграмм. Диаграмма вариантов использования (Use Case Diagram): элементы, нотация, назначение.
10. Диаграммы поведения в UML: диаграммы последовательности (Sequence Diagram) и активности (Activity Diagram). Их роль в описании алгоритмов взаимодействия модулей, включая AI-сервисы.
11. Диаграммы структуры в UML: диаграммы классов (Class Diagram) и компонентов (Component Diagram). Применение для проектирования архитектуры.
12. Нотация моделирования бизнес-процессов BPMN 2.0 (основные элементы). Моделирование процесса с этапом принятия решения AI-моделью.
13. Концептуальное моделирование данных: сущность, атрибут, связь. Построение ERдиаграмм (нотация Чена или «воронья лапка»).
14. Понятие архитектуры программного обеспечения. Архитектурные стили: многослойная (N-звенная) архитектура, клиент-сервер, микросервисная архитектура.
15. Событийно-ориентированная архитектура (EDA). Ее преимущества для систем, обрабатывающих потоковые данные и события от AI-моделей.
16. Принципы интеграции AI-сервисов в информационные системы. Паттерны: AI-as-aService, встраивание, batch-обработка.
17. Проектирование интерфейсов взаимодействия: API (REST, gRPC), сообщения (message queues). Контракты API для AI-сервисов.
18. Проектирование баз данных: этапы (концептуальный, логический, физический). Нормализация отношений: цели и основные нормальные формы (1NF, 2NF, 3NF).

19. Проектирование для больших данных: отличия хранилища данных (Data Warehouse) и озера данных (Data Lake). Концепция Feature Store для машинного обучения.
20. Системы контроля версий: назначение, базовые принципы. Git: базовые понятия (репозиторий, коммит, ветка, удаленный репозиторий). Основные команды (git init, git add, git commit, git push/pull).
21. Инструментальные средства проектирования: CASE-средства, средства визуального моделирования (примеры: Draw.io, PlantUML, Enterprise Architect).
22. Понятие и уровни тестирования программного обеспечения (модульное, интеграционное, системное). Особенности тестирования систем с AI (валидация модели, мониторинг дрейфа).
23. Виды тестовой документации: тест-план, чек-лист, тест-кейс. Составление тестовых сценариев для проверки интеграции компонентов.
24. Управление IT-проектом: основные роли (менеджер проекта, аналитик, архитектор, разработчик). Артефакты проектирования и их место в жизненном цикле.
25. Риски в проектах по интеграции ИИ: технические (качество данных, интерпретируемость моделей), организационные, этические. Способы минимизации.

Темы для самостоятельной работы обучающихся

1. Сравнение каскадной и Agile-методологий: Критерии выбора для проектов с высоким уровнем неопределённости в требованиях к AI-модели.
2. Модель CRISP-DM: Анализ этапов кросс-индустриального стандартного процесса добычи данных и его сопоставление с этапами проектирования ИС.
3. Концепция MLOps: Исследование отличий от классического DevOps и новых ролей в проектной команде.
4. Гибкие фреймворки (Scrum vs Kanban): Сравнение применительно к этапу проектирования архитектуры AI-системы.
5. Анализ рисков на ранних этапах жизненного цикла AI-проекта: технические, данныевые, этические.
6. Модель зрелости ML-систем (MLOps Maturity Model): Изучение уровней и критериев оценки зрелости процессов.
7. Техники сбора требований для AI-систем: Специфика интервью с экспертами предметной области и data scientist-ами.
8. Нефункциональные требования к AI-компонентам: Детальный анализ требований к производительности (latency, throughput), масштабируемости и интерпретируемости.
9. Функциональные требования к системе мониторинга дрейфа данных (data drift) и концептуального дрейфа (concept drift).
10. Моделирование домена с помощью UML Class Diagram: Разработка диаграммы классов для системы, включающей сущности «Модель», «Эксперимент», «Набор данных».
11. Диаграмма последовательности (UML Sequence Diagram) для процесса инференса: Моделирование взаимодействия между клиентом, API-шлюзом, AI-сервисом и базой данных.

12. BPMN-модель бизнес-процесса с автоматическим принятием решений: Моделирование процесса кредитного скоринга с этапом автоматического утверждения/отказа на основе прогноза модели.
13. Сравнение монолитной и микросервисной архитектуры с точки зрения эксплуатации ML-моделей в продакшене.
14. Паттерны интеграции ML-моделей: Глубокий анализ паттернов «Сервисориентированный» (ML-as-a-Service) и «Встроенный» (Embedded).
15. Event-Driven Architecture (EDA) для AI: Исследование применения шин событий (Kafka) для обработки потоковых данных и запуска ML-пайплайнов.
16. Проектирование API для AI-сервисов: Сравнение REST, gRPC и GraphQL для задач передачи данных в/из модели.
17. Концепция Feature Store: Изучение назначения, архитектуры и преимуществ для обеспечения согласованности данных между обучением и обслуживанием модели.
18. Архитектурные шаблоны отказоустойчивости (Circuit Breaker, Retry) при вызовах внешних AI-сервисов.
19. Проектирование схемы данных для хранения метаданных ML-экспериментов: Анализ необходимых атрибутов и связей.
20. Data Warehouse vs Data Lake vs Data Lakehouse: Сравнительный анализ парадигм хранения данных для целей машинного обучения.
21. Принципы и инструменты версионирования данных (на примере DVC — Data Version Control).
22. Контейнеризация ML-приложений: Изучение практик создания Docker-образов для воспроизводимости экспериментов и развертывания моделей.
23. Оркестрация ML-пайплайнов: Обзор возможностей Apache Airflow, Kubeflow Pipelines для автоматизации обучения и переобучения моделей.
24. Инфраструктура как код (IaC) для ML-стендов: Знакомство с использованием Terraform для описания облачных ресурсов под ML-нагрузку.
25. Обзор современных инструментов визуального проектирования архитектуры (например, Draw.io, Lucidchart, архитектурные диаграммы в Mermaid).
26. Принципы тестопригодности (Testability) архитектуры: Как проектировать систему, чтобы ML-компоненты было легко тестировать.
27. A/B-тестирование и канареечные релизы (Canary Releases) AI-моделей: Методология и инструменты для безопасного развертывания.
28. Проектная документация для AI-проектов: Анализ состава документации сверх стандартного ТЗ (Model Card, FactSheet).
29. Этические руководства (Ethical Guidelines) и их отражение в проектных решениях: Принципы FAIR (Findable, Accessible, Interoperable, Reusable) для данных и моделей.
30. Анализ кейса провала внедрения AI-системы: Исследование реального примера (на основе публикаций) и выявление ошибок, допущенных на этапе проектирования.

Примерные варианты тестовых оценочных заданий (ТОЗ) для контрольного рубежа в рамках текущего контроля

Тестовое задание	Оцениваемые компетенции

ВОПРОСЫ ОТКРЫТОГО ТИПА	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Модель жизненного цикла ПО, в которой следующий этап начинается только после полного завершения предыдущего, называется _____ моделью. +: каскадной	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Манифест, провозглашающий ценности: "люди и взаимодействие важнее процессов и инструментов", "работающий продукт важнее исчерпывающей документации", называется _____. +: Agile	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Совокупность практик, направленных на объединение процессов разработки ML-моделей и их эксплуатации в продакшене, называется _____. +: MLOps	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Временной интервал в Scrum, в течение которого создается рабочий инкремент продукта, называется _____. +: спринт	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Международный стандарт, определяющий процессы жизненного цикла программного обеспечения, имеет номер _____. +: ISO/IEC 12207	ОК 02.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Организация, разрабатывающая и поддерживающая спецификации UML и BPMN, называется _____. +: OMG (Object Management Group)	ОК 02.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Диаграмма UML, показывающая взаимодействие между объектами во времени и использующая вертикальные линии жизни, называется диаграммой _____. +: последовательности	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Диаграмма UML, описывающая функциональность системы с точки зрения акторов и вариантов их взаимодействия, называется диаграммой _____. +: вариантов использования	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Нотация для моделирования бизнес-процессов, использующая элементы "пул", "дорожка", "шлюз", называется _____. +: BPNM	ПК 1.1.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Графическое представление сущностей предметной области, их атрибутов и связей между ними называется _____-диаграммой. +: ER(Entity Relationship)	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОВТЕТ) Архитектурный стиль, предполагающий разбиение приложения на небольшие независимые сервисы, называется _____ архитектурой. +: микросервисной	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Архитектурный стиль, в котором компоненты реагируют на события, генерируемые другими компонентами, называется _____ архитектурой. +: событийно-ориентированной	ОК 01.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Централизованное хранилище для управления и версионирования признаков, используемых ML-моделями, называется _____. +: Feature Store	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Протокол взаимодействия микросервисов, использующий HTTP и принципы работы с ресурсами (GET, POST, PUT, DELETE), называется _____. +: REST	ПК 1.1.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ)	ПК 1.1.

Архитектурный паттерн, обеспечивающий отказоустойчивость при вызовах внешних сервисов путем автоматического отключения проблемного компонента, называется _____. +: Circuit Breaker	
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Команда Git для сохранения изменений в локальном репозитории называется _____. +: commit	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Команда Git для отправки изменений из локального репозитория в удаленный называется _____. +: push	ПК 1.4.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Независимая линия разработки в Git, позволяющая работать над новым функционалом изолированно, называется _____. +: ветка	ПК 1.4
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Инструмент для версионирования больших файлов (датасетов, моделей) в Git-репозиториях называется _____. +: Git LFS	ПК 1.4.

(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Вид тестирования, проверяющий взаимодействие между отдельными модулями или сервисами, называется _____ тестированием. +: интеграционным	ПК 1.7.
(ВПИСАТЬ ПРАВИЛЬНЫЙ ОТВЕТ) Метод развертывания новой версии модели, при котором трафик постепенно переключается на нее для небольшой части пользователей, называется _____ развертыванием. +: канареечным	ПК 1.7.
ВОПРОСЫ ЗАКРЫТОГО ТИПА	
Модель жизненного цикла ПО, наиболее предпочтительная для проектов с высокой неопределённостью в требованиях к AIкомпонентам на начальном этапе: Каскадная (водопадная) V-образная +:Итеративная/Agile Спиральная	ОК 01.
Какой из перечисленных ресурсов является ОФИЦИАЛЬНЫМ источником спецификаций языка моделирования UML? Блог на Habr Книга "Приемы объектно-ориентированного проектирования" +: Сайт консорциума Object Management Group (OMG) Документация Microsoft	ОК 02.
Какая из перечисленных диаграмм UML наилучшим образом подходит для описания последовательности шагов во времени при взаимодействии пользователя, фронтенда и AI-микросервиса? Диаграмма вариантов использования (Use Case) Диаграмма классов (Class Diagram) +: Диаграмма последовательности (Sequence Diagram) Диаграмма развертывания (Deployment Diagram)	ПК 1.1.
Требование к системе: "Точность (Accuracy) модели классификации текста на этапе приемочного тестирования должна составлять не менее 92%". К какому виду требований оно относится? Функциональное требование +: Нефункциональное требование (атрибут качества) Бизнес-требование Ограничение проекта	ПК 1.3.
Какой архитектурный стиль предполагает разбиение приложения на небольшие, слабосвязанные сервисы, которые общаются по стандартным протоколам (HTTP, gRPC) и могут развертываться независимо? Монолитная архитектура +: Микросервисная архитектура Многослойная архитектура (n-звенная) Архитектура "клиент-сервер"	ОК 01.

Концепция Feature Store в MLOps-архитектуре в первую очередь предназначена для: Хранения исходного кода модели Оркестрации Docker-контейнеров +: Централизованного управления, версионирования и обслуживания признаков (features) для обучения и инференса моделей Мониторинга потребления ресурсов серверов	ПК 1.1.
Какая последовательность команд Git является КОРРЕКТНОЙ для того, чтобы сохранить изменения в локальном репозитории и отправить их в удаленный репозиторий с именем origin в ветку main? git pull -> git add . -> git commit git commit -m "msg" -> git add . -> git push +: git add . -> git commit -m "msg" -> git push origin main git push origin main -> git add . -> git commit -m "msg"	ПК 1.4.
Инструмент Git LFS (Large File Storage) используется в основном для: Управления большим количеством мелких текстовых конфигурационных файлов Создания резервных копий всей базы кода +: Эффективной работы с бинарными файлами большого размера (датасеты, модели, видео) в Git-репозитории Визуализации истории коммитов	ПК 1.4.
Какой тип тестирования проверяет взаимодействие между двумя или более отдельными модулями или сервисами (например, между вебприложением и AI-API)? Модульное (юнит-тестирование) +: Интеграционное тестирование Нагрузочное тестирование Приемочное тестирование	ПК 1.7.

Шкала оценивания тестовых заданий

Данный вид контроля, рассчитан на выявление уровня усвоения теоретического и практического материала в рамках изучения дисциплин. Критерии оценивания теста: 1 вопрос = 1 балл.

Оценка «5»	30-27 Баллов
Оценка «4»	26-23 Баллов
Оценка «3»	22-18 Баллов
Оценка «2»	0-17 Баллов