🤖 ИИ-ассистент / Chatbot

Замечания и контекст

ИИ-ассистент должен представлять собой полноценное решение с продуманным пользовательским опытом, а не простую обертку над API языковой модели. Проект должен демонстрировать понимание принципов работы с LLM, prompt engineering, и современных подходов к построению диалоговых систем.

Важно: Выбор предметной области и типа ассистента (текстовый, голосовой, мультимодальный) остается за студентом, но решение должно быть обоснованным и соответствовать выбранной задаче.

📋 МИНИМАЛЬНЫЕ ТРЕБОВАНИЯ (на 5 баллов)

1. Документация

Обязательные разделы:

- **Описание ассистента:** цель, целевая аудитория, основные сценарии использования
- **Архитектура решения:** схема взаимодействия компонентов, выбор модели и обоснование
- **Prompt Engineering документация:**
  - Системный промпт с объяснением каждой части
  - Шаблоны промптов для различных сценариев
  - Примеры типичных диалогов
- **API документация:** описание endpoints, форматы запросов/ответов
- **Ограничения системы:** что ассистент НЕ умеет делать
- **Инструкция пользователя:** как взаимодействовать с ботом

2. Проектирование

Архитектура:

- Использование современной LLM через API (OpenAI, Anthropic, Google, open-source)
- Многослойная архитектура: UI → Backend → LLM API
- Разделение бизнес-логики и интеграции с моделью

Prompt Engineering:

- Продуманный системный промпт, определяющий поведение ассистента
- Минимум 3 различных шаблона промптов для разных сценариев
- Обработка edge cases (пустой ввод, слишком длинные сообщения, неподходящий контент)

UX/UI:

- Интуитивный интерфейс чата (веб или мессенджер)
- Индикация состояний: печатает, обрабатывает, ошибка
- Возможность начать новый диалог

3. Реализация

Основной функционал:

- Обработка текстовых запросов пользователя
- Генерация релевантных ответов через API модели
- Базовая валидация входных данных (длина, формат)
- Обработка ошибок API (таймауты, лимиты, недоступность)
- Логирование запросов и ответов

Безопасность:

- API ключи хранятся в переменных окружения
- Базовая защита от prompt injection (фильтрация опасных паттернов)
- Rate limiting для предотвращения злоупотреблений

4. Тестовые данные и сценарии

- Минимум 10 примеров диалогов, демонстрирующих возможности
- Тестовые сценарии для различных типов запросов
- Примеры обработки некорректных запросов

5. Использование

- Время ответа не более 10 секунд для стандартных запросов
- Поддержка минимум 10 одновременных пользователей
- Корректная обработка кириллицы (если применимо)
- Сохранение истории в рамках сессии

6. Ограничения

Что НЕЛЬЗЯ делать:

- ❌ Простая прокси без обработки к ChatGPT/Claude API

- ❌ Отсутствие промпт-инжиниринга (raw передача вопросов)
- ❌ Хардкод ответов или fake AI
- ❌ Хранение API ключей в коде
- ❌ Игнорирование ошибок API
- ❌ Отсутствие валидации входных данных

🚀 МАКСИМАЛЬНЫЕ ТРЕБОВАНИЯ (на 10 баллов)

Все минимальные требования ПЛЮС:

1. Продвинутая архитектура

Множественные модели:

- Использование 2+ различных моделей для разных задач
- Интеллектуальный выбор модели based on запрос
- Fallback механизм при недоступности основной модели
- Оптимизация costs через выбор подходящей модели

RAG (Retrieval-Augmented Generation):

- Интеграция с векторной БД (Pinecone, Weaviate, pgvector, ChromaDB)
- Загрузка и индексация документов (PDF, TXT, MD)
- Семантический поиск по knowledge base
- Минимум 100 документов/страниц в базе знаний
- Метрики качества retrieval (precision, recall)

MCP (Model Context Protocol) или Tool Use:

- Интеграция внешних инструментов (калькулятор, поиск, API)
- Определение когда использовать tools
- Минимум 3 различных tool/function

2. Продвинутая обработка

Препроцессинг:

- Определение языка и intent запроса
- Классификация типа запроса
- Проверка на токсичность/inappropriate content
- Автоматическое исправление опечаток

Постпроцессинг:

- Форматирование ответов (markdown, списки, таблицы)
- Fact-checking критически важной информации
- Добавление источников/ссылок к ответам
- Структурированный вывод (JSON для определенных запросов)

3. Управление контекстом

История диалогов:

- Полноценное управление контекстом беседы
- Сохранение истории между сессиями
- Умная компрессия контекста при превышении лимитов
- Возможность ссылаться на предыдущие сообщения

Персонализация:

- Профили пользователей с предпочтениями
- Адаптация стиля ответов под пользователя
- Запоминание важной информации о пользователе
- Настройки: формальность, длина ответов, технический уровень

4. Мультимодальность

Минимум один дополнительный тип input/output:

- **Изображения:** анализ загруженных изображений, генерация изображений
- **Аудио:** speech-to-text, text-to-speech интеграция
- **Документы:** обработка загруженных PDF/DOCX
- **Структурированные данные:** работа с таблицами, CSV

5. Мониторинг и аналитика

Метрики:

- Dashboard с метриками использования
- Среднее время ответа, количество запросов
- Популярные темы/вопросы
- User satisfaction metrics (если есть feedback)

Логирование:

- Структурированные логи (JSON)
- Трассировка запросов (correlation ID)
- Мониторинг затрат на API

- Alerts при критических ошибках

6. Оптимизация

Performance:

- Кэширование частых запросов
- Streaming ответов (печать по мере генерации)
- Асинхронная обработка тяжелых запросов
- Load balancing при высокой нагрузке

Cost optimization:

- Автоматический выбор между моделями по cost/quality
- Батчинг запросов где возможно
- Использование embeddings cache для RAG

7. Документация для разработчиков

- OpenAPI/Swagger спецификация
- Гайд по добавлению новых промптов
- Документация по расширению функционала
- Performance benchmarks

⚠️ ОСНОВНЫЕ ОШИБКИ И НЕДОЧЕТЫ

Критические ошибки в безопасности

1. **Хранение API ключей в коде или репозитории**
   Пример: `api_key = "sk-xxxxx"` прямо в коде
   Минус 2 балла
2. **Отсутствие защиты от prompt injection**
   Пример: Прямая передача пользовательского ввода без санитизации
   Минус 1 балл
3. **Логирование sensitive данных**
   Пример: Полные данные пользователей в логах
   Минус 1 балл

Ошибки в архитектуре

4. **Отсутствие обработки ошибок API**
   Пример: Приложение падает при 429 (rate limit) или timeout
   Минус 1 балл

5. **Простая proxy без добавленной ценности**
   Пример: `response = openai.complete(user_input)` и всё
   Минус 2 балла
6. **Отсутствие промпт-инжиниринга**
   Пример: Нет системного промпта, контекста, инструкций
   Минус 1 балл

Ошибки в UX/функциональности

7. **Нет индикации состояния обработки**
   Пример: Пользователь не понимает, работает ли бот
   Минус 1 балл
8. **Потеря контекста в рамках сессии**
   Пример: Бот забывает о чем говорили 2 сообщения назад
   Минус 1 балл
9. **Отсутствие валидации входных данных**
   Пример: Принимаются сообщения любой длины, крашится на emoji
   Минус 1 балл

Ошибки в реализации

10. **Синхронная блокирующая обработка**
    Пример: Весь сервер ждет ответа от OpenAI для одного пользователя
    Минус 1 балл
11. **Отсутствие логирования**
    Пример: Невозможно отследить проблемы в production
    Минус 1 балл
12. **Хардкод конфигураций**
    Пример: URL, модели, параметры зашиты в код
    Минус 1 балл

Ошибки в документации

13. **Отсутствие документации промптов**
    Пример: Непонятно какие промпты используются и почему
    Минус 1 балл
14. **Нет описания ограничений системы**
    Пример: Не указано что бот НЕ умеет делать
    Минус 1 балл

Ошибки в развертывании

15. **Приложение не развернуто или недоступно**
    Пример: Работает только localhost
    Минус 2 балла

💡 Рекомендации

Для успешной защиты:

- Выберите конкретную предметную область и сделайте ассистента экспертом
- Продемонстрируйте понимание работы LLM через грамотный prompt engineering
- Покажите несколько интересных use cases, не только Q&A
- Обязательно обработайте edge cases и ошибки
- Добавьте хотя бы одну "изюминку" из максимальных требований

Примеры хороших тем:

- 🏥 Медицинский ассистент с RAG по медицинским справочникам
- 📚 Образовательный тьютор с персонализацией под уровень студента
- 💼 HR-ассистент с интеграцией в корпоративные системы
- 🛒 E-commerce консультант с поиском по каталогу товаров
- 🎮 Game master для текстовых RPG с управлением состоянием мира