

**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное автономное образовательное учреждение высшего  
образования**

**«Национальный исследовательский Нижегородский государственный университет им.  
Н. И. Лобачевского»  
(ННГУ)**

**Институт информационных технологий, математики и механики  
Кафедра математического обеспечения и суперкомпьютерных технологий**

**УЧЕБНЫЙ КУРС**

**«Проектирование и архитектура программных систем»**

**для подготовки по направлению 09.03.04 «Программная инженерия»**

**КОНЦЕПЦИЯ ПРОЕКТА**

**«QA-система на основе ИИ»**

**Выполнили** студенты группы 3822Б1ПР1  
Ворошилов Виталий Александрович, Крылов Михаил Георгиевич,  
Моисеев Артём Владимирович, Морозов Егор Алексеевич,  
Рамс Сергей Никитич

**Проверил** к.т.н., доцент  
Лебедев Илья Геннадьевич

Нижегород  
2025

## Содержание

1. Необходимость проекта .....	3
1.1. Обоснование необходимости .....	3
1.2. Видение проекта .....	3
1.3. Анализ выгод .....	3
2. Концепция решения .....	4
2.1. Цели и задачи .....	4
2.2. Предположения и ограничения .....	4
2.2.1. Предположения .....	4
2.3. Анализ использования .....	5
2.3.1. Пользователи .....	5
2.3.2. Сценарии использования .....	5
2.4. Требования .....	5
2.4.1. Требования пользователей .....	5
2.4.2. Системные требования .....	6
3. Рамки .....	7
3.1. Функциональность решения .....	7
3.2. За рамками решения .....	7
3.3. Критерии одобрения решения .....	8
4. Стратегии дизайна решения .....	9
4.1. Стратегия архитектурного дизайна .....	9
4.2. Стратегия технологического дизайна .....	9

# **1. Необходимость проекта**

## **1.1. Обоснование необходимости**

Пользователи тратят много времени на поиск нужной информации в больших текстах. Проект решает эту проблему с помощью нейросети, которая быстро и точно находит ответы на вопросы через Telegram-бота и веб-приложение.

## **1.2. Видение проекта**

Проект направлен на создание удобной платформы для быстрого получения ответов из текстовых источников через Telegram-бота и веб-интерфейс. Решение должно обеспечивать высокую точность и скорость обработки запросов, быть доступным и понятным пользователю. Разработка планируется к реализации в течение 12 месяцев и ориентирована на применение в образовательной, аналитической и профессиональной сферах.

## **1.3. Анализ выгод**

- Для пользователей: повышается скорость и удобство работы с информацией, снижаются трудозатраты на поиск нужных данных, повышается точность и достоверность получаемых ответов. Это способствует росту эффективности обучения, аналитической и исследовательской деятельности.
- Для заказчика: проект обеспечивает конкурентное преимущество за счёт внедрения современного инструмента обработки текстовых данных, сокращает время обслуживания запросов клиентов или сотрудников, а также способствует оптимизации внутренних процессов и снижению затрат.
- Для разработчиков: реализация проекта позволяет углубить компетенции в области обработки естественного языка, развить собственные технологические решения, повысить профессиональный опыт и потенциал дальнейшего коммерческого или исследовательского использования системы.

## **2. Концепция решения**

### **2.1. Цели и задачи**

1. Подготовка и дообучение (fine-tuning) базовой модели
  1. Выбор базовой модели
  2. Подготовка датасета для дообучения
  3. Дообучение модели
  4. Валидация качества модели
2. Разработка контроллера вычислительного узла
  1. Проектирование API
3. Разработка серверной части
  1. Проектирование структуры базы данных
  2. Проектирование API
  3. Интеграция с контроллером вычислительного узла
  4. Оптимизация и проверка надежности
  5. Контейнеризация и развертывание
4. Разработка веб-интерфейса
  1. Проектирование и дизайн UI/UX
  2. Имплементация
  3. Интеграция и тестирование
5. Разработка Telegram-бота
  1. Дизайн функционала бота
  2. Имплементация
6. Тестирование и мониторинг системы
  1. Комплексное тестирование
  2. Мониторинг потребления ресурсов
  3. Анализ использования

### **2.2. Предположения и ограничения**

#### **2.2.1. Предположения**

1. Предполагается доступ к GPU-серверам с объемом памяти не менее 8ГБ на этапе дообучения модели
2. Пользователи будут формулировать вопросы только на русском и английском языках
3. В среднем один пользователь будет производить от 5 до 10 запросов в день

## 2.3. Анализ использования

### 2.3.1. Пользователи

Решение не разделяет пользователей на группы.

### 2.3.2. Сценарии использования



Рисунок 2.3.2.1 – Диаграмма сценариев использования (Use Case)

## 2.4. Требования

### 2.4.1. Требования пользователей

- UR-1. Прием ввода контекста. Описание требования: Система позволяет пользователю ввести текст контекста.
- UR-2. Ввод вопроса и получение ответа. Описание требования: Пользователь может задать вопрос к загруженному контенту и получить ответ в виде фрагмента текста.
- UR-3. Поддержка двух интерфейсов. Описание требования: Система позволяет использовать веб-интерфейс и Telegram-бот с эквивалентной базовой функциональностью (ввод контекста, ввод вопроса, получение ответа).
- UR-4. Повторный запрос к контексту. Описание требования: Взаимодействие системы с пользователем происходит в виде чата, к загруженному контексту имеется возможность задавать вопросы неограниченное число раз.

- UR-5. История и сессии. Описание требования: Пользователь может просматривать историю своих запросов в рамках сессии.
- UR-6. Удобство интерфейса. Описание требования: простая форма ввода, кнопка «Отправить», область для вывода ответа.
- UR-7. Логирование запросов. Описание требования: Система логирует запросы и ответы в целях отладки и анализа; персональные данные не сохраняются.

#### **2.4.2. Системные требования**

- SR-1. Аппаратные требования для обучения/дообучения. Описание требования: Для этапа *fine-tuning* необходимы GPU-серверы с минимум 8 GB видеопамати (рекомендация —  $\text{GPU} \geq 12 \text{ GB}$  для комфортной работы). Диск и оперативная память выбираются по объёму данных (рекомендовано: 100 GB диск, 32 GB RAM для сервера обучения).
- SR-2. Набор API. Описание требования: Необходим минимальный набор API для интеграции.
- SR-3. Telegram-интеграция. Описание требования: Поддержка webhook/long-polling для бота; устойчивость к повторным сообщениям и перегрузкам.
- SR-4. Лимиты ответов. Описание требования: Максимальная длина генерируемого ответа — 2500–4000 символов (в качестве конфигурируемого параметра).
- SR-5. Точность модели. Описание требования: Целевые метрики на тестовой выборке: Exact Match (EM)  $\approx 0.4$ , F1-score  $\approx 0.7$ .

## **3. Рамки**

### **3.1. Функциональность решения**

В рамках разрабатываемого решения будут реализованы следующие основные функции и возможности:

1. Прием входных данных от пользователя
  1. Ввод текста (контекста) для анализа
  2. Ввод вопроса, на который требуется получить ответ из текста
  3. Возможность повторного запроса благодаря организации взаимодействия в виде чата
2. Обработка запроса нейронной сетью
  1. Передача текста и вопроса сервером на другой сервер - контроллер вычислительного узла - сервер, на котором запущена предобученная QA модель
  2. Получение ответа от модели в виде фрагмента текста
3. Отображение результата
  1. Вывод ответа пользователю в понятном виде
  2. Отображение вопросов к контексту в виде чата
4. Поддержка двух интерфейсов
  1. Веб-интерфейс:
    - Форма для ввода текста, открывающая чат
    - Кнопка "отправить"
    - Область чата
  2. Telegram-бот
    - Команда /start - приветствие и инструкции
    - Поддержка диалогового взаимодействия: бот запрашивает текст, каждое последующее сообщение интерпретируется как вопрос, на который бот возвращает ответ
    - Команда /newchat - инициирование запроса ботом нового контекста у пользователя
5. Логирование (базовое)
  - Запись в обезличенном виде запросов и ответов в лог для отладки и анализа качества модели

### **3.2. За рамками решения**

Планируемые к добавлению функции:

- Мультиязычность. Модель обучена на двух языках, расширение требует переобучения или замены модели.
- Голосовой ввод текста или вопроса. Требуется интеграции с Text-To-Speech (TTS) сервисом.
- Оценка ответа пользователем (лайк/дизлайк). Полезно для улучшения модели.
- Поддержка извлечения текста из фотографий. Требуется внедрения OCR-технологий.

### 3.3. Критерии одобрения решения

- Пользователь должен иметь возможность, согласно требованиям:
  - Ввести текст через веб-интерфейс и иметь возможность задавать вопросы в виде чата
  - Получить ответ от модели в течение 10 секунд
  - Задавать неограниченное число вопросов к исходному тексту
- Ответ должен быть корректным с точки зрения содержания, на основе текстовых примеров
- Интерфейсы должны работать и отображаться корректно, как на десктопных, так и на мобильных устройствах
- Технические критерии:
  - Два интерфейса: веб-интерфейс и Telegram-бот
  - Модель должна соответствовать следующим минимальным требованиям:
    - Метрика Exact Match (EM)  $\approx 0.4$  (на тестовой выборке). Метрика оценивает полное совпадение ответа нейронной сети с эталонным ответом (с учетом регистра и пробелов)
    - Метрика F1-мера (F1-score)  $\approx 0.7$  (на тестовой выборке). F1 учитывает частичное совпадение между сгенерированным и эталонным ответом. Она особенно полезна, когда ответ почти правильный, но не идеален.
- Логгирование: запись запросов пользователей и ответов модели

## 4. Стратегии дизайна решения

### 4.1. Стратегия архитектурного дизайна

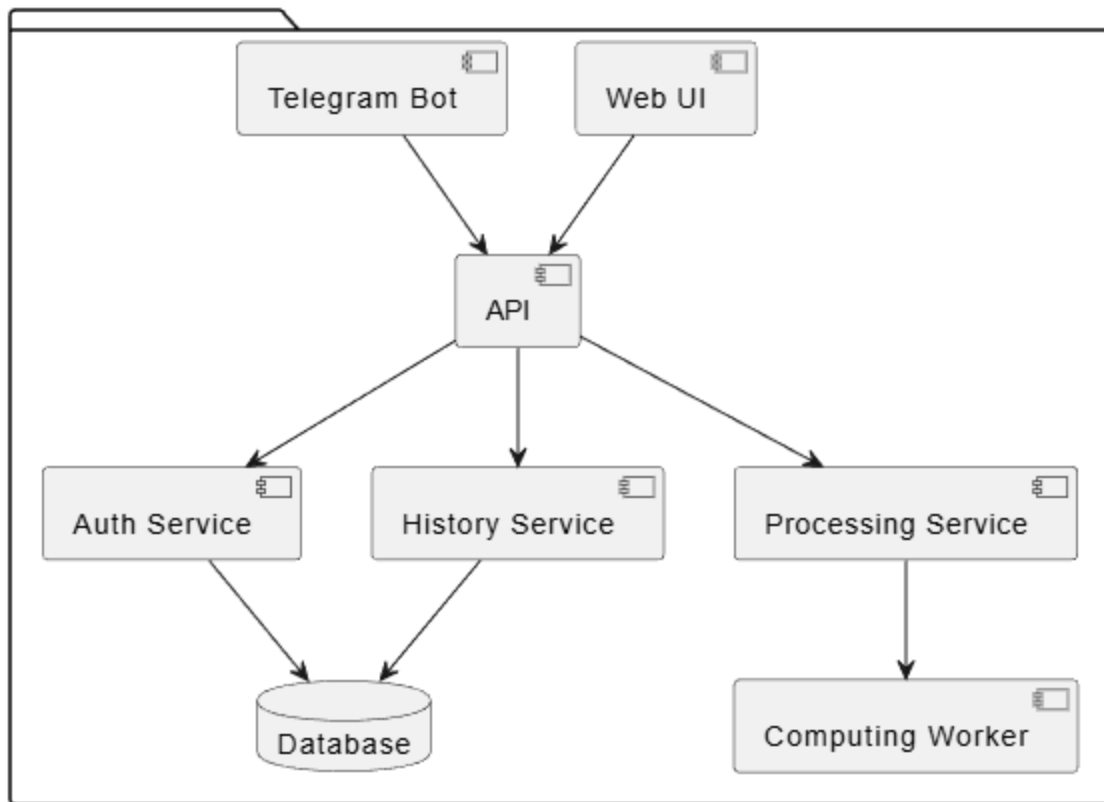


Рисунок 4.1.1 – Стратегия архитектурного дизайна

### 4.2. Стратегия технологического дизайна

- Серверная часть:
  - Go - язык разработки
  - gin - HTTP веб-фреймворк
  - gorm - ORM для взаимодействия с БД
  - gRPC - шлюз взаимодействия с контроллером вычислительного узла (серверная часть выступает в роли клиента)
- Контроллер вычислительного узла:
  - Python - язык разработки
  - gRPC - шлюз взаимодействия с основной серверной частью (контроллер вычислительного узла выступает в роли сервера)
  - loguru - библиотека для организации логгирования
- Веб-интерфейс:
  - TypeScript - язык разработки

- Vite - система сборки
- Vue.js - UI-фреймворк
- Naive UI - библиотека компонентов
- База данных: PostgreSQL - СУБД