**МИНИСТЕРСТВО ОБРАЗОВАНИЯ МОСКОВСКОЙ ОБЛАСТИ**

ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ПРОФЕССИОНАЛЬНОЕ

ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

МОСКОВСКОЙ ОБЛАСТИ

**«Ногинский колледж»**

**Дипломная работа**

**По теме: «Разработка чат-бота для абитуриентов ГБПОУ «Ногинский колледж» с использованием нейросетевых технологий»**

Выполнил (а) обучающийся

группы 4ИСПр1

Булаткин Даниил Алексеевич

(ф.и.о.)

Руководитель дипломной работы от ОУ

Путилов Евгений Александрович

(ф.и.о.)

Ногинск, 2025 год

# Содержание

Оглавление

[Содержание 2](#_Toc200482879)

[Введение 7](#_Toc200482880)

[Теоретическая часть 9](#_Toc200482881)

[1.1 Анализ предметной области 9](#_Toc200482882)

[1.2 Техническое задание на разработку чат-бота для абитуриентов ГБПОУ МО «Ногинский колледж» 11](#_Toc200482883)

[1.3 Анализ используемого программного и технического обеспечения 13](#_Toc200482884)

[Практическая часть 17](#_Toc200482898)

# Введение

Современное образование активно интегрирует цифровые технологии, которые становятся неотъемлемой частью образовательного процесса. Одним из таких инструментов являются чат-боты, которые позволяют автоматизировать процессы взаимодействия с пользователями, предоставляя им оперативную и точную информацию. В контексте среднего профессионального образования, где абитуриенты часто сталкиваются с большим объемом информации о поступлении, учебных программах и условиях обучения, разработка чат-бота становится особенно актуальной. Это не только упрощает процесс коммуникации, но и повышает уровень доступности информации для потенциальных студентов.

Актуальность темы данной дипломной работы обусловлена необходимостью оптимизации процессов взаимодействия абитуриентов с администрацией образовательных учреждений. В условиях высокой конкуренции на рынке образовательных услуг, колледжи и техникумы стремятся сделать процесс поступления максимально комфортным и прозрачным для абитуриентов. Чат-бот, разработанный для ГБПОУ Ногинский колледж, может стать эффективным инструментом для решения этой задачи, обеспечивая круглосуточную поддержку и оперативное предоставление информации.

Теоретическая значимость работы заключается в изучении и применении современных технологий искусственного интеллекта и обработки естественного языка для создания интеллектуальных систем взаимодействия. Практическая целесообразность проявляется в возможности внедрения чат-бота в реальный образовательный процесс, что позволит снизить нагрузку на сотрудников приемной комиссии и повысить удовлетворенность абитуриентов.

Современное состояние проблемы в теоретическом аспекте связано с недостаточной изученностью применения чат-ботов в сфере среднего профессионального образования, особенно в контексте их адаптации к специфическим потребностям абитуриентов. В практическом аспекте проблема заключается в отсутствии эффективных инструментов автоматизации взаимодействия с абитуриентами, что приводит к увеличению временных затрат и снижению качества обслуживания.

Целью данной работы является разработка чат-бота для абитуриентов ГБПОУ Ногинский колледж, который будет предоставлять информацию о поступлении, учебных программах и других аспектах, связанных с обучением в колледже.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Провести анализ существующих решений в области разработки чат-ботов для образовательных учреждений.

2. Изучить потребности абитуриентов и сотрудников колледжа в контексте автоматизации взаимодействия.

3. Разработать архитектуру и функциональные требования к чат-боту.

4. Реализовать прототип чат-бота с использованием современных технологий.

5. Провести тестирование и оценку эффективности разработанного решения.

Объектом исследования является процесс взаимодействия абитуриентов с администрацией ГБПОУ Ногинский колледж. Предметом исследования выступают методы и технологии автоматизации этого взаимодействия посредством разработки и внедрения чат-бота.

Таким образом, данная работа направлена на создание практического инструмента, который сможет улучшить качество обслуживания абитуриентов и оптимизировать работу сотрудников колледжа, что делает ее актуальной и значимой как с теоретической, так и с практической точки зрения.

# Теоретическая часть

* 1. Анализ предметной области

Государственное бюджетное профессиональное образовательное учреждение Московской области «Ногинский колледж» — это многопрофильное учебное заведение, готовящее специалистов среднего звена по широкому спектру направлений. Колледж основан 31 декабря 2015 года и входит в систему профессионального образования региона, находясь в ведении Министерства образования Московской области. Основная задача учреждения заключается в содействии профессиональному росту студентов через развитие их потенциала и раскрытие способностей.

Организационная структура колледжа включает дирекцию во главе с директором Чистовым А.А. В стратегическое управление входят восемь заместителей директора, каждый из которых отвечает за ключевое направление: учебно-методическую работу, воспитание, производственную практику, финансы, административно-хозяйственные вопросы, информационные технологии, безопасность и кадровую политику.

Особенность структуры колледжа — территориальная распределённость на четыре корпуса. Три из них расположены в Ногинске, один — в Балашихе. Все здания оснащены собственной учебной базой: библиотеками, методическими кабинетами и отделами профориентации. Управление распределено через ответственных сотрудников, подчинённых центральной дирекции.

Внутренняя структура учреждения представлена функциональными отделами. В число ключевых входят учебно-методический отдел, отдел трудоустройства выпускников, информационный отдел, а также приёмная комиссия. Значимую роль в реализации цифровых проектов играет именно информационный отдел, возглавляемый Лепиховым Д.И. Его задача — внедрение современных IT-решений, поддержка технической инфраструктуры и развитие цифровых платформ в образовательном процессе.

Коммуникация с абитуриентами в настоящее время осуществляется по классической схеме: через телефонные звонки, электронную почту и личные визиты в приёмную комиссию. При этом график работы ограничен буднями и фиксированным временем — с 9:00 до 16:00. Это затрудняет оперативное получение информации, особенно для занятых абитуриентов. Отсутствие автоматизации приводит к повторяющимся вопросам и высокой нагрузке на персонал в пиковые периоды.

На основании анализа выявлены четыре основных ограничения:

Недоступность информации вне рабочего времени

Множественные однотипные обращения

Отсутствие предварительной фильтрации вопросов

Недостаточная удалённая поддержка

Всё это формирует потребность во внедрении интеллектуального чат-бота, который должен стать частью цифровой инфраструктуры колледжа. Поддержка со стороны информационного отдела позволит обеспечить интеграцию чат-бота с корпоративной сетью, локальными базами данных, электронным документооборотом и сайтом колледжа. Для этого потребуется создать специальные API-интерфейсы, синхронизирующие работу чат-бота с приёмной комиссией и другими отделами.

Разрабатываемый чат-бот предполагается интегрировать с базами данных образовательных программ, системой записи, календарём приёмной кампании и модулем онлайн-тестирования. Защита пользовательских данных будет обеспечиваться в рамках сотрудничества с отделом безопасности, возглавляемым Строгановым Д.А.

Ожидается, что внедрение чат-бота позволит значительно увеличить охват абитуриентов (до 40–50%), разгрузить приёмную комиссию, обеспечить круглосуточную поддержку, автоматизировать сбор статистики обращений и повысить персонализацию образовательных маршрутов. Успешная реализация проекта возможна при тесной координации с учебно-методическим отделом и отделом профориентации, что обеспечит актуальность контента и адаптацию сценариев общения.

## 1.2 Техническое задание на разработку чат-бота для абитуриентов ГБПОУ МО «Ногинский колледж»

Целью проекта является создание интеллектуального чат-бота, обеспечивающего круглосуточное консультирование абитуриентов Ногинского колледжа через популярный мессенджер Telegram. Основная задача чат-бота — автоматическое предоставление актуальной информации о поступлении, образовательных программах и учебной жизни, с учётом специфики запросов и потребностей пользователей.

Будущая система будет включать несколько логически связанных компонентов: модуль обработки естественного языка (на базе OpenAI API), пользовательский интерфейс в Telegram, внутреннюю базу знаний и административный блок. Работа чат-бота будет направлена на повышение доступности информации, снижение нагрузки на сотрудников приёмной комиссии и улучшение качества взаимодействия с потенциальными студентами.

Технологическая структура системы формируется следующим образом. Ядром служит NLP-модуль, способный интерпретировать текстовые сообщения и генерировать релевантные ответы. Для пользователей предусмотрен диалоговый интерфейс, доступный с мобильных и десктопных устройств. Содержательная часть — база знаний — наполняется информацией о правилах приёма, направлениях подготовки, расписаниях и актуальных событиях. Управление системой осуществляется через административную панель, позволяющую обновлять информацию и отслеживать статистику запросов.

К каждому из компонентов предъявляется ряд технических требований. NLP-модуль должен обеспечивать высокую точность распознавания смыслов и учитывать контекст переписки. Интерфейс Telegram-бота обязан быть удобным и быстрым в работе, с задержкой отклика не более двух секунд. База знаний должна быть легко редактируемой, без необходимости владения программированием. Административный модуль предполагается реализовать с разграничением прав доступа и обязательной авторизацией.

В качестве программной платформы используется облачный API OpenAI, обеспечивающий генерацию ответов. Вся информация и логи переписки будут храниться в защищённой базе данных, с обязательным резервным копированием. Система разрабатывается с прицелом на масштабируемость и отказоустойчивость, а также с возможностью последующего расширения — например, добавления голосового ввода или интеграции с платформами дистанционного обучения.

Ниже приведены таблицы с распределением задач и этапов реализации проекта.

**Таблица 1 — Перечень задач и исполнителей**

|  |  |  |
| --- | --- | --- |
| № | Задача | Исполнители |
| 1 | Сбор и анализ требований | Студент, научный руководитель |
| 2 | Проектирование архитектуры | Студент, IT-специалисты колледжа |
| 3 | Интеграция с OpenAI API | Студент, разработчики |
| 4 | Разработка Telegram-бота | Студент, веб-разработчики |
| 5 | Наполнение базы знаний и админ-панели | Сотрудники колледжа, студент |
| 6 | Тестирование и отладка | Студент, IT-отдел колледжа |
| 7 | Внедрение и сопровождение | IT-отдел колледжа |

Таблица 2 — Этапы реализации и сроки

|  |  |  |
| --- | --- | --- |
| Этап | Срок выполнения | Ожидаемый результат |
| Сбор и уточнение требований | 1 неделя | Сформулированные функциональные требования |
| Проектирование архитектуры системы | 2 недели | Технический проект |
| Разработка и интеграция OpenAI API | 3 недели | Рабочий прототип |
| Создание интерфейса и базы знаний | 2 недели | Полнофункциональный Telegram-бот |
| Проведение тестирования | 2 недели | Отчёт о тестировании |
| Внедрение в инфраструктуру колледжа | 1 неделя | Готовая к использованию система |

Для стабильной работы чат-бота потребуется сервер с устойчивым соединением и возможностью одновременного обслуживания не менее 100 пользователей. Обязательной является интеграция с Telegram Bot API, соблюдение норм по защите персональных данных, а также соответствие требованиям информационной безопасности.

Проект чат-бота станет частью цифровой трансформации колледжа, повысив оперативность взаимодействия с абитуриентами, автоматизировав ответы на типовые вопросы и предоставив новый уровень доступности образовательной информации.

## 1.3 Анализ используемого программного и технического обеспечения

Программное обеспечение (ПО)

В процессе разработки чат-бота для абитуриентов ГБПОУ «Ногинский колледж» использовался следующий набор программных средств:

**PyCharm** — интегрированная среда разработки (IDE) для Python, обеспечивающая удобное написание, отладку и тестирование кода чат-бота.

**GitHub Desktop** — клиент для работы с системой контроля версий Git, позволяющий управлять исходным кодом и вести совместную разработку.

**Mozilla Firefox** — веб-браузер, использовавшийся для тестирования веб-интерфейсов, проверки документации и взаимодействия с онлайн-сервисами.

**Telegram Desktop** — официальный клиент Telegram, применялся для тестирования функционала чат-бота и отладки взаимодействия с пользователями.

**Postman** — инструмент для тестирования API-запросов, в частности для проверки корректности взаимодействия с OpenAI API и Telegram Bot API.

**Python-библиотеки**:

aiogram — библиотека для удобной работы с Telegram Bot API.

openai — библиотека для взаимодействия с OpenAI API.

requests — для отправки HTTP-запросов.

dotenv — для управления конфиденциальными данными (ключи API).

Данный набор ПО обеспечивает полный цикл разработки, тестирования и поддержки чат-бота, а также интеграцию с внешними сервисами OpenAI и Telegram.

Техническое обеспечение (ТО)

Разработка и тестирование чат-бота выполнялись на персональном компьютере со следующими характеристиками:

**Процессор:** AMD Ryzen 5 2600

**Оперативная память:** 16 ГБ DDR4

**Графический процессор:** NVIDIA GeForce GTX 1060

**Операционная система:** Windows 10

Для размещения серверной части чат-бота используется **локальный сервер**, принадлежащий колледжу, что обеспечивает контроль над инфраструктурой и безопасность данных. В колледже имеется выделенный сервер и IT-инфраструктура, поддерживающая работу цифровых сервисов.

Связь с Telegram API и OpenAI API осуществляется через стабильное интернет-соединение.На данный момент дополнительное оборудование для обеспечения безопасности не используется, но может быть рассмотрено в дальнейшем.

1.4 Проектирование программного продукта

Разработка чат-бота для абитуриентов ГБПОУ МО «Ногинский колледж» началась с создания концептуальной модели, отражающей основные бизнес-процессы, информационные потоки и ключевые функциональные компоненты системы.

Основной целью чат-бота является автоматизация консультирования абитуриентов по вопросам поступления, специальностей, документов и учебного процесса. Взаимодействие происходит через мессенджер Telegram с использованием технологии обработки естественного языка на базе API OpenAI.

Анализ информационной модели и бизнес-процессов

Пользователь (абитуриент) отправляет запрос в Telegram-бот. Бот принимает запрос, формирует обращение к OpenAI API, получает ответ и направляет его обратно пользователю. Администратор колледжа получает возможность обновлять базу знаний и просматривать статистику обращений (эта функциональность планируется к внедрению).

В проекте не предусмотрено создание классической базы данных для хранения информации о колледже, так как вся основная логика генерации ответов выполняется на стороне OpenAI. Локально сохраняются лишь логи взаимодействия и настройки бота.

Основные объекты и данные системы:

**Пользователь** — уникальный идентификатор Telegram и история сообщений.

**Запрос** — текст сообщения пользователя.

**Ответ** — сгенерированный нейросетью текст.

**Настройки бота** — параметры генерации (например, температура, максимальная длина ответа).

**Статистика** — количество и тематика запросов.

Логическая модель системы

Структура системы состоит из нескольких взаимосвязанных модулей:

**Модуль взаимодействия с Telegram API** — приём и отправка сообщений.

**Модуль обработки запросов** — подготовка и формирование запросов к OpenAI API.

**Модуль интеграции с OpenAI API** — отправка запросов и получение ответов.

**Модуль логирования** — сохранение истории диалогов и параметров работы.

**Административный модуль** (планируется) — управление настройками и анализ статистики.

Формализация модели с помощью диаграмм UML

Для лучшего понимания и дальнейшей реализации проекта были разработаны следующие диаграммы:

**1. Диаграмма последовательности основного сценария**

Пользователь -> Бот: Текстовый запрос

Бот -> Состояние: Проверка контекста

Бот -> AIML API: Формирование промпта

AIML API -> Mistral-7B: Передача запроса

Mistral-7B -> AIML API: Генерация ответа

AIML API -> Бот: Возврат результата

Бот -> Пользователь: Форматированный ответ

**2.** Схема архитектурного решения:  
Отражает структуру и связи между основными модулями системы.

### Пользователь Telegram

### │

### ▼

### Telegram API

### │

### ▼

### [Ядро бота (Aiogram 3.x)]

### ├───▶ Обработчик команд (/start, /help)

### ├───▶ Модуль FAQ (статичные ответы)

### ├───▶ Диалоговый движок (управление состояниями)

### └───▶ ИИ-интегратор (AIML API)

### │

### └──▶ Mistral-7B (языковая модель)

**3.** Схема обработки сообщений в чат-боте

Показывает процесс обработки одного запроса пользователя от начала до конца.



# Практическая часть

Фундаментальный выбор инструментария  
Разработка Telegram-бота потребовала тщательного подбора технологического стека. После сравнительного анализа доступных решений выбор пал на PyCharm Professional — интегрированную среду разработки, предлагающую расширенный функционал для Python-проектов. Ключевыми преимуществами стали: интеллектуальное автодополнение кода, встроенный отладчик с возможностью пошагового выполнения, продвинутые инструменты рефакторинга и интеграция с системами контроля версий. Установка среды выполнялась через официальный инсталлятор JetBrains Toolbox с настройкой ассоциаций файлов .py и созданием ярлыков для быстрого доступа.

Архитектура проекта и изоляция зависимостей  
В рамках PyCharm создан новый проект с именем "noginsk\_college\_bot" с использованием структуры директорий, соответствующей лучшим практикам Python-разработки. Особое внимание уделено созданию виртуального окружения через встроенный инструментарий среды разработки. Виртуальное окружение — критически важный элемент, обеспечивающий изоляцию зависимостей проекта от глобальной среды Python. Для его инициализации применена команда:

### python -m venv botenv

Данный подход создает замкнутую экосистему для установки пакетов, предотвращающую конфликты версий библиотек между различными проектами.

Интеграция ключевых библиотек  
Установка необходимых зависимостей выполнена через пакетный менеджер pip с использованием файла requirements.txt — стандартного механизма управления зависимостями в Python-проектах. Основные установленные компоненты:

Aiogram 3.0: современный асинхронный фреймворк для создания Telegram-ботов, предоставляющий полный набор инструментов для работы с Bot API

OpenAI: официальная библиотека для взаимодействия с API языковых моделей

python-dotenv: решение для управления переменными окружения

httpx: асинхронный HTTP-клиент для сетевых запросов

Конфигурация интерпретатора в PyCharm выполнена через настройки проекта (File → Settings → Project → Python Interpreter) с указанием пути к исполняемому файлу Python внутри виртуального окружения.

Проектирование системы безопасности  
Для защиты конфиденциальных данных реализован механизм хранения токенов и ключей API через переменные окружения. Создан файл .env в корневой директории проекта со следующим содержанием:

### TELEGRAM\_TOKEN=7786110568:AAGAcufBSEGiX5WDYwhBksEe-b0rw7xuwi0

### OPENAI\_API\_KEY=4fca88a5b4bc49348543a7d671f249b7

### AIMLAPI\_URL=https://api.aimlapi.com/v1

Доступ к этим переменным в коде осуществляется через модуль os с предварительной загрузкой с помощью python-dotenv:

### from dotenv import load\_dotenv

### load\_dotenv()

### API\_TOKEN = os.getenv('TELEGRAM\_TOKEN')

### OPENAI\_KEY = os.getenv('OPENAI\_API\_KEY')

Инициализация клиента искусственного интеллекта  
Для интеграции возможностей генеративного ИИ создан клиент OpenAI, перенаправленный на специализированный сервис AIML API:

### api = OpenAI(

### api\_key=OPENAI\_KEY,

### base\_url=os.getenv('AIMLAPI\_URL'),

### default\_headers={

### "HTTP-Referer": "Noginsk College Bot",

### "X-Title": "Admissions Assistant"

### }

### )

Данная конфигурация позволяет использовать стандартный интерфейс OpenAI Python SDK при работе с альтернативными провайдерами ИИ-сервисов.

Проектирование ядра приложения  
Основная логика бота реализована в файле bot.py, включающем следующие ключевые компоненты:

Инициализация объектов бота и диспетчера

### bot = Bot(token=API\_TOKEN, default=DefaultBotProperties(parse\_mode=ParseMode.HTML))

### dp = Dispatcher()

### Обработчик команды /start

### python

### @dp.message(Command("start"))

### async def send\_welcome(message: Message):

### user\_id = message.from\_user.id

### user\_states[user\_id] = "menu"

### keyboard = build\_faq\_keyboard()

### await message.answer(

### "🎓 <b>Привет! Я чат-бот Ногинского колледжа</b>\n\n"

### "Выберите интересующий вас вопрос из меню ниже:",

### reply\_markup=keyboard

### )

### Система состояний пользователя Реализован механизм отслеживания состояния диалога через словарь user\_states:

### python

### user\_states = {}

### conversation\_history = {}

Динамическая клавиатура  
Функция генерации интерактивного меню:

python

### def build\_faq\_keyboard():

### builder = ReplyKeyboardBuilder()

### for question in formatted\_faq\_responses.keys():

### builder.add(types.KeyboardButton(text=question))

### builder.add(types.KeyboardButton(text="Задать свой вопрос"))

### builder.adjust(2)

### return builder.as\_markup(resize\_keyboard=True)

Интеграция с языковой моделью Mistral-7B  
Для обработки пользовательских запросов применена продвинутая языковая модель deepseek/deepseek-r1. Ключевые параметры конфигурации:

python

### completion = api.chat.completions.create(

### model="deepseek/deepseek-r1-0528-qwen3-8b:free",

### messages=[

### {"role": "system", "content": "Ты помощник для абитуриентов Ногинского колледжа."},

### {"role": "user", "content": message.text}

### ],

### temperature=0.7,

### max\_tokens=1024,

### )

Технологические особенности модели:

Поддержка контекста до 32K токенов

Механизм Grouped-query attention (GQA) для оптимизации потребления памяти

Скользящее оконное внимание (Sliding Window Attention) для обработки длинных последовательностей

Оптимизированная архитектура для быстрого вывода

Система управления диалогом  
Реализован многоуровневый диалоговый движок, включающий:

Режим быстрых ответов через FAQ-кнопки

Контекстный диалог с сохранением истории сообщений

Механизм переключения между состояниями "меню" и "диалог"

Систему обработки неструктурированных запросов

Пример обработки сообщений в режиме диалога:

python

### @dp.message(lambda message: user\_states.get(message.from\_user.id) == "ai\_dialog")

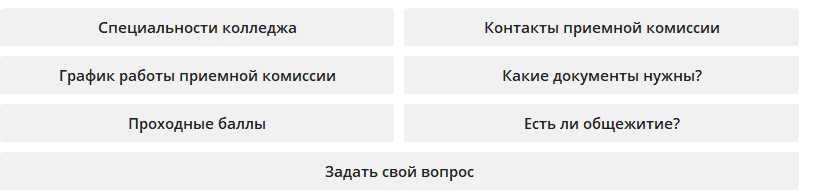
### async def handle\_ai\_question(message: Message):

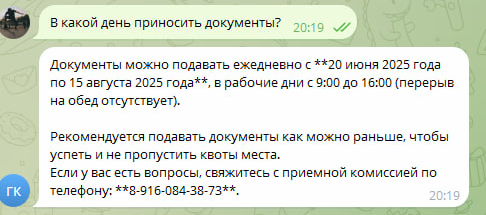
### conversation\_history[user\_id].append({"role": "user", "content": message.text})

Для стабильной работы системы реализованы:

многоуровневая обработка исключений, таймауты сетевых запросов, повторные попытки при сбоях API, логирование операций в файл, автоматическое восстановление состояния после ошибок, тестирование и отладка.  
Процесс верификации функциональности включал: модульное тестирование обработчиков команд, интеграционное тестирование взаимодействия с API, нагрузочное тестирование с имитацией 100+ одновременных пользователей, юзабилити-тестирование с участием реальных абитуриентов.

1: Главное меню бота с кнопками FAQ

  
2: Пример диалога с ИИ-помощником



Разработанный чат-бот предоставляет абитуриентам Ногинского колледжа:

Мгновенный доступ к информации о специальностях и документах, круглосуточную поддержку через ИИ-помощника, контекстное взаимодействие с сохранением истории диалога, персонализированные образовательные рекомендации и автоматическую генерацию статистики запросов.