**«Санкт-Петербургский государственный электротехнический университет**

**«ЛЭТИ» им. В.И.Ульянова (Ленина)»**

**(СПбГЭТУ «ЛЭТИ»)**

|  |  |  |
| --- | --- | --- |
| **Направление** | 01.03.02 – Прикладная математика и информатика | |
| **Профиль** | Математическое обеспечение программно-информационных систем | |
| **Факультет** | КТИ | |
| **Кафедра** | МОЭВМ | |
| *К защите допустить* |  | |
| Зав. кафедрой |  | А.А. Лисс |

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

БАКАЛАВРА

Тема: РАЗРАБОТКА ОБРАЗОВАТЕЛЬНОГО ЧАТ-БОТА

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент |  |  |  | А.С. Петров |
|  |  | *подпись* |  |  |
| Руководитель | к.т.н., доцент |  |  | М.М. Заславский |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |
| Консультанты | к.т.н., доцент |  |  | К.В. Брюханов |
|  | *(Уч. степень, уч. звание)* | *подпись* |  |  |

Санкт-Петербург

2025

**ЗАДАНИЕ**

**на выпускную квалификационную работу**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой МО ЭВМ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г. |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Студент | Петров А.С. | | | |  | Группа | 1381 | |
| Тема работы: Разработка образовательного чат-бота | | | | | | | | |
| Место выполнения ВКР: СПбГЭТУ «ЛЭТИ» кафедра МО ЭВМ | | | | | | | | |
| Исходные данные (технические требования):  Необходимо разработать чат-бота для обучения студентов по учебным материалам, загруженным преподавателями. | | | | | | | | |
| Содержание ВКР:  Введение, Обзор предметной области, Формулировка требований к решению и постановка задачи, Архитектура программной реализации, Программная реализация проекта, Заключение | | | | | | | | |
| Перечень отчетных материалов: пояснительная записка, иллюстративный материал | | | | | | | | |
| Дополнительные разделы: экономическое обоснование ВКР | | | | | | | | |
|  | | | | | | | | |
| Дата выдачи задания | | | Дата представления ВКР к защите | | | | | |
| «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_ г. | | | | | |
|  | | |  | | | | | |
| Студент | |  | | А.С. Петров | | | |  |
| Руководитель *к.т.н., доцент* | |  | | М.М. Заславский | | | |  |
| *(Уч. степень, уч. звание)* | |  | |  | | | |  |
| Консультант *к.т.н., доцент* | |  | | К.В. Брюханов | | | |  |
| *(Уч. степень, уч. звание)* | |  | |  | | | |  |

**календарный план выполнения**

**выпускной квалификационной работы**

|  |  |
| --- | --- |
|  | Утверждаю |
|  | Зав. кафедрой МО ЭВМ |
|  | \_\_\_\_\_\_\_\_\_\_\_\_ А.А. Лисс |
|  | «\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2025 г. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | Петров А.С. |  | Группа | 1381 |
| Тема работы: Разработка образовательного чат-бота | | | | |

|  |  |  |
| --- | --- | --- |
| № п/п | Наименование работ | Срок выполнения |
| 1 | Обзор литературы по теме работы | 01.04 – 10.04 |
| 2 | Обзор предметной области | 10.04 – 13.04 |
| 3 | Формулировка требований к решению и постановка задачи | 13.04 – 16.04 |
| 4 | Программная реализация продукта | 16.04 – 25.04 |
| 5 | Экономическое обоснование ВКР | 25.04 – 05.05 |
| 5 | Оформление пояснительной записки | 05.05 – 20.05 |
| 6 | Оформление иллюстративного материала | 20.05 – 30.05 |
| 7 | Предзащита | 24.05 |

|  |  |  |
| --- | --- | --- |
| Студент |  | А.С. Петров |
| Руководитель *к.т.н., доцент* |  | М.М. Заславский |
| *(Уч. степень, уч. звание)* |  |  |

**РЕФЕРАТ**

Пояснительная записка 65 стр., 19 рис., 16 табл., 24 ист.

ДИСТАНЦИОННОЕ ОБУЧЕНИЕ, TELEGRAM-БОТ, БОТ, МЕССЕНДЖЕРЫ, TELEGRAM.

**Объектом исследования** является процесс дистанционного обучения студентов с использованием мессенджеров.

**Предметом исследования** является удобство, возможность и качество дистанционного обучения студентов с использованием мессенджеров.

**Цель работы:** разработка образовательного чат-бота для обучения студентов в мессенджере Telegram.

При выполнении этой работы были изучены существующие образовательные технологии, реализованные преимущественно в популярных мессенджерах. Затем был произведен сравнительный анализ этих технологий и возможности улучшения и расширения функционала существующих решений. Путем сравнения были выявлены характеристики, которыми должно обладать полученное решение. По результатам анализа было выполнено проектирование и разработка образовательного бота для мессенджера Telegram.

**ABSTRACT**

During this work, existing educational technologies were studied, implemented mainly in popular messengers. Then, a comparative analysis of these technologies and the possibility of improving and expanding the functionality of existing solutions was carried out. By comparison, the characteristics that the resulting solution should have were identified. Based on the results of the analysis, the design and development of an educational bot for the Telegram messenger was carried out.

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 9](#_Toc197963738)

[1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ 11](#_Toc197963739)

[1.1 Современные подходы к дистанционному обучению 11](#_Toc197963740)

[1.1.1 Дистанционное обучение: понятия и тенденции 11](#_Toc197963741)

[1.1.2 Особенности обучения студентов в онлайн-сфере 11](#_Toc197963742)

[1.1.3 Преимущества и ограничения цифровых платформ 12](#_Toc197963743)

[1.2 Мессенджеры как платформа для образовательных решений 13](#_Toc197963744)

[1.2.1 Рост популярности мессенджеров у молодежи 13](#_Toc197963745)

[1.2.2 Виды взаимодействия с пользователями в мессенджерах 13](#_Toc197963746)

[1.2.3 Преимущества и недостатки использования мессенджеров в образовании 13](#_Toc197963747)

[1.3 Сравнительный анализ мессенджеров для разработки ботов 14](#_Toc197963748)

[1.3.1 Обзор доступных мессенджеров 14](#_Toc197963749)

[1.3.2 Telegram 14](#_Toc197963750)

[1.3.3 WhatsApp 15](#_Toc197963751)

[1.3.4 Viber 15](#_Toc197963752)

[1.3.5 Вконтакте 15](#_Toc197963753)

[1.3.6 Facebook 16](#_Toc197963754)

[1.3.7 Discord 16](#_Toc197963755)

[1.3.8 Сравнительная таблица и выводы 17](#_Toc197963756)

[1.4 Telegram как образовательная платформа 18](#_Toc197963757)

[1.4.1 Основные возможности Telegram 18](#_Toc197963758)

[1.4.2 Особенности Telegram-ботов 19](#_Toc197963759)

[1.4.3 Причины выбора Telegram в рамках проекта 20](#_Toc197963760)

[1.5 Выводы по разделу обзор предметной области 21](#_Toc197963761)

[2 ФОРМУЛИРОВКА ТРЕБОВАНИЙ К РЕШЕНИЮ И ПОСТАНОВКА ЗАДАЧИ 23](#_Toc197963762)

[2.1 Постановка задачи 23](#_Toc197963763)

[2.2 Требования к решению 23](#_Toc197963764)

[2.3 Обоснование требований к решению 23](#_Toc197963765)

[2.4 Обоснование выбора метода решения 26](#_Toc197963766)

[3 АРХИТЕКТУРА ПРОГРАММНОЙ РЕАЛИЗАЦИИ 28](#_Toc197963767)

[3.1 Выбор методологии разработки ПО 28](#_Toc197963768)

[3.1.1 Общий обзор существующих методологий разработки ПО 28](#_Toc197963769)

[3.1.2 Каскадная модель (Waterfall) 28](#_Toc197963770)

[3.1.3 Agile-методология 29](#_Toc197963771)

[3.1.4 Scrum 30](#_Toc197963772)

[3.1.5 Kanban 30](#_Toc197963773)

[3.2 Сравнительный анализ подходов к разработке ПО 31](#_Toc197963774)

[3.3 Обоснование выбранного подхода к разработке проекта 32](#_Toc197963775)

[3.4 Особенности реализации методологии разработки в проекте 32](#_Toc197963776)

[3.5 Сценарии использования разработанного бота 33](#_Toc197963777)

[3.5.1 Пользователь – ученик 33](#_Toc197963778)

[3.5.2 Администратор – преподаватель 35](#_Toc197963779)

[3.6 Выбор базы данных 36](#_Toc197963780)

[3.7 Используемые технологии 38](#_Toc197963781)

[4 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРОЕКТА 39](#_Toc197963782)

[4.1 Регистрация чат-бота в Telegram 39](#_Toc197963783)

[4.2 Проектирование и реализация БД для чат-бота 40](#_Toc197963784)

[4.2.1 Общая схема БД 40](#_Toc197963785)

[4.2.2 Описание и назначение реализованных коллекций 41](#_Toc197963786)

[4.3 Служебные файлы и исполняемый файл 45](#_Toc197963787)

[4.3.1 Служебный файл config.py 45](#_Toc197963788)

[4.3.2 main.py 45](#_Toc197963789)

[4.3 Реализация классов для работы бота. 46](#_Toc197963790)

[4.3.1 Класс Database 46](#_Toc197963791)

[4.3.2 Класс User 49](#_Toc197963792)

[4.4 Функции, реализующие функционал работы бота 50](#_Toc197963793)

[4.4.1 Пользовательские функции 50](#_Toc197963794)

[4.4.2 Администраторские функции 53](#_Toc197963795)

[4.4.3 Остальные немаловажные функции 56](#_Toc197963796)

[ЗАКЛЮЧЕНИЕ 58](#_Toc197963797)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 60](#_Toc197963798)

[ПРИЛОЖЕНИЕ А 64](#_Toc197963799)

**ОПРЕДЕЛЕНИЯ, ОБОЗНАЧЕНИЯ И СОКРАЩЕНИЯ**

ЯП – язык программирования;

API – Application Programming Interface;

ПО – программное обеспечение;

БД – база данных;

СУБД – система управления базами данных;

ER-модель – модель данных, описывающая концептуальные схемы предметной области.

## ВВЕДЕНИЕ

Современное общество все чаще сталкивается с необходимостью быстрой и доступной передачи знаний в условиях дистанционного взаимодействия. Это обусловлено активной цифровизацией всех сфер жизни и повсеместным внедрением интернет-технологий. Одним из перспективных направлений в этой области является использование популярных мессенджеров, таких как Telegram, который благодаря своей открытой архитектуре, безопасности, высокой скорости работы и широкому функционалу занимает лидирующие позиции.

Объектом исследования данной работы является процесс дистанционного обучения студентов с использованием мессенджеров.

Предметом исследования данной работы является удобство, эффективность и качество дистанционного обучения студентов посредством чат-ботов в мессенджере Telegram.

Целью данной работы является разработка Telegram-бота, обеспечивающего структурированную подачу учебных материалов, автоматизированную проверку усвоения знаний и хранение данных о результатах обучения.

Задачи работы:

Проведение анализ современных подходов к дистанционному обучению и возможностей мессенджеров для образовательных целей.

Определение требований к функционалу образовательного чат-бота.

Разработка архитектуры программного решения, включая выбор технологий и базы данных.

Реализация образовательного Telegram-бота с возможностью администрирования учебных материалов и тестирования студентов.

Практическая значимость проекта заключается в возможности его использования в образовательных учреждениях, онлайн-курсах и других сферах, требующих автоматизации подачи и контроля учебных материалов. Разработанный бот способен упростить процесс обучения для студентов и администрирования для преподавателей, что особенно актуально в условиях роста спроса на дистанционное образование.

## 1 ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

### 1.1 Современные подходы к дистанционному обучению

#### 1.1.1 Дистанционное обучение: понятия и тенденции

Дистанционное обучение – достаточно современная технология, позволяющая упростить и сделать более доступным повышение грамотности населения. Связано это с развитием IT технологий и сети Интернет. [1] Дистанционное обучение позволяет получать образование в условиях возможных ограничений, как например пандемия коронавируса в 2020 году. Также дистанционное обучение открывает возможности для получения образования для людей с ограниченными возможностями, которые физически не имеют возможности перемещаться для получения образования.

Тенденция на получение образования, не выходя из дома получили широкое развитие во времена уже упомянутой пандемии 2020 года. Тогда людям нельзя было запрещено выходить из дома по причине введенных властями ограничений. Ограничения были введены для предотвращения распространения заболевания. И это дало толчок на быстрое развитие дистанционных технологий в различных сферах жизни людей.

В 2020 году получили широкое распространение онлайн-сервисы доставки, образования из-за возникновения острой необходимости в них в силу внешних обстоятельств.

#### 1.1.2 Особенности обучения студентов в онлайн-сфере

Современные студенты являются активными пользователями сети Интернет и не могут самостоятельно решать возникающие жизненные сложности без доступа к интернету. Этот факт создает новые возможности при организации и планировании учебного процесса в сети Интернет.

Одной из ключевых особенностей является высокая онлайн грамотность и привычка работать с интернетом, мессенджерами и социальными сетями, где они проводят достаточно много времени из своей жизни. При этом обычно особенно у студентов младших курсов отсутствует само дисциплинированность, структуризация, планирование и организованность, что и усложняет процесс самообучения без внешнего контроля. [2]

Потребление контента среди молодежи, как правило происходит в формате коротких и насыщенных структурированных блоков информации. Чему подтверждением является популярность вышедшего в 2017 году сервиса для создания и просмотра коротких видеороликов под названием «TikTok». Поэтому традиционные лекции теряют внимание аудитории молодых студентов. Эффективность обучения требует адаптации содержания под привычки восприятия современной молодежи: использование коротких и понятных уроков и четко структурированной модульной архитектуры. [3]

Среди плюсов дистанционного обучения следует выделить тот факт, что обучающиеся ценят гибкость учебного процесса и возможность учиться в любое удобное для них время и из любого удобного места.

Важным фактором также является легкая доступность качественных учебных материалов, возможность повторного прохождения уроков и тестов, а также широкий выбор учебных курсов.

Одновременно с вышеперечисленными преимуществами можно выделить и недостатки такого формата. К ним следует отнести:

1. дефицит живого общения между обучающимися
2. сложность совместной работы
3. усиленные требования к самоорганизации.

По итогу вышеизложенного производится вывод о том, что дистанционное обучение требует учета психологических и поведенческих факторов обучающихся. Полученное решение должно быть адаптировано под стиль жизни современной молодежи, то есть быть интересным, доступным и максимально простым в использовании. [4]

#### 1.1.3 Преимущества и ограничения цифровых платформ

Дистанционные образовательные платформы стали неотъемлемой частью современного образовательного процесса. Они используются и в ВУЗах, и в школах, и в системах дополнительного образования. Под дистанционными образовательными платформами понимаются онлайн-системы, которые хранят, и распространяют учебные материалы, и сопровождают образовательный процесс.

### 1.2 Мессенджеры как платформа для образовательных решений

#### 1.2.1 Рост популярности мессенджеров у молодежи

Мессенджеры занимают центральное место в общении современной молодежи. Для обучающихся по программам среднего и высшего образования они являются привычной и удобной формой взаимодействия как в повседневной жизни, так и в учебной среде. Популярность мессенджеров объясняется как простотой их использования, так и высокой скоростью обмена сообщениями. [5]

#### 1.2.2 Виды взаимодействия с пользователями в мессенджерах

Мессенджеры предлагают разные форматы общения для пользователей. Во многих популярных мессенджерах присутствует поддержка текстовой переписки, видеозвонков, голосовых сообщений, стикеров, эмодзи, отправка фотографий, видеосообщений, отправка видеофайлов, текстовых файлов и т. д. Но основным и наиболее используемым инструментом общения являются текстовые сообщения.

#### 1.2.3 Преимущества и недостатки использования мессенджеров в образовании

Использование мессенджеров в сфере образования набирает популярность благодаря их доступности и привычности для обучающихся. Одним из главных преимуществ мессенджеров является скорость и простота коммуникации. В отличие от традиционных образовательных платформ, мессенджеры не требуют отдельной регистрации, по причине наличия с целью использования в повседневной жизни. Также не требуется обучение пользователя для работы с интерфейсом, поскольку он уже умеет пользоваться мессенджером из-за его частого использования.

Однако у такого подхода есть и область ограничений. В первую очередь сюда следует отнести отсутствие четкого разделения между личным и учебным пространством, что может вызывать у обучающегося ощущение перегрузки. Да и к тому же не все преподаватели в силу своих личных качеств, возраста и жизненной позиции не готовы переходить на такую форму взаимодействия со своими учениками.

Таким образом, мессенджеры могут эффективно дополнять основной образовательный процесс, особенно если их использование будет уместно с учетом специфики аудитории и выполняемых задач. Главная цель – подбор удобного формата и отсутствие перегрузки в этом мессенджере обучающегося информацией, не касающейся учебного процесса.

### 1.3 Сравнительный анализ мессенджеров для разработки ботов

#### 1.3.1 Обзор доступных мессенджеров

Создание образовательного чат-бота требует очень внимательного выбора платформы. На практике используются различные мессенджеры, однако все они отличаются возможностями своих API и доступными библиотеками, на основе которых и планируется разработка бота. Также все мессенджеры имеют разный уровень распространенности среди пользователей, а в частности студентов, для которых и разрабатывается данный чат-бот. Ниже рассмотрены наиболее популярные мессенджеры, которые подходят для разработки бота с целью его использования в образовательных целях.

#### 1.3.2 Telegram

Telegram [6] считается одной из самых удобных платформ для разработки чат-бота. Он имеет открытое API и Bot API [7], которые поддерживаются многими языками программирования. Боты, написанные для использования в мессенджере Telegram могут иметь функционал взаимодействия с пользователями через текстовые сообщения, кнопки, меню, формы и встроенные в бота команды. Важным преимуществом является отсутствие необходимости одобрения разработанного продукта: разработчик может самостоятельно запустить собственный продукт своими силами. Также мессенджер Telegram не имеет ограничений на количество сообщений, что также немаловажно для использования образовательного бота.

#### 1.3.3 WhatsApp

Мессенджер WhatsApp [8] предлагает ограниченные возможности для разработки ботов через Business API [9]. Данный интерфейс рассчитан в первую очередь на корпоративные сценарии такие как: автоматизация поддержки клиентов, уведомления и маркетинг. Следует также отметить, что доступ к API платный и требует одобрения, да и работает в рамках бизнес-аккаунта. Боты, написанные для данного мессенджера в полном объеме, не подходят образовательного процесса, поскольку существуют ограничения по времени и цене вопроса.

#### 1.3.4 Viber

Мессенджер Viber [10] предоставляет Bot API [11], который позволяет создавать публичные боты. Имеется поддержка основных функций, таких как обмен сообщениями, кнопочки и т. д. Однако мессенджер имеет ограниченный интерфейс настройки, а предоставляемая документация уступает по полноте решению, которое предоставляет мессенджер Telegram. Для запуска бота также требуется регистрация аккаунта через платформу Viber Admin Panel.

Следует также учитывать, что данный мессенджер не имеет большой популярности среди молодежи, в особенности если сравнивать с мессенджером Telegram. Но несмотря на это, в регионах он еще используется, что делает его возможным выбором для реализации образовательного бота.

#### 1.3.5 Вконтакте

Мессенджер Вконтакте [12] имеет обширные возможности для создания ботов через платформу VK Mini Apps [13]. Разработчики могут реализовать полноценные сценарии взаимодействия бота с его пользователями в рамках сообществ, использовать кнопки, меню, отправку опросов и медиа. Платформа поддерживает широкие интеграции, но ориентирована в первую очередь на внутриэкосистемные приложения.

Использование внутренних приложений помимо мессенджера безусловно добавляет сложности в разработке и снижает охват аудитории, поскольку многие новодобавленные функциин могут не работать на старых версиях мессенджера. Использование старых версий необходимо принять во внимание, поскольку далеко не факт, что обучающиеся имеют современные устройства. Среди молодежи мессенджер Вконтакте остается достаточно популярным, особенно в рамках стран СНГ. Но ограничение на привязку к сообществу и необходимость прохождения модерации разработанного продукта бесусловно снижают гибкость продукта в сравнении с мессенджером Telegram.

#### 1.3.6 Facebook

Мессенджер Facebook [14] предоставляет Messenger Platform API [15]. Данная библиотека достаточно функционально развита и имеет функционал, используемый для работы с текстом, кнопками, шаблонами, файлами и т. д. Однако недавно Facebook ужесточил условия доступа и модерации. Для запуска бота необходимо создание бизнес-страницы, прохождение проверки и соблюдение строгих правил по приватности.

Необходимо также принять во внимание заметное снижение популярности мессенджера Facebook среди молодежи на просторах стран СНГ. Это делает его непривлекательным для создания образовательного бота.

#### 1.3.7 Discord

Платформа Discord [16] изначально задумывалась для общения людей, вовлеченных в игры, но сейчас достаточно широко используется образовательными сообществами для коммуникации преподавателей и обучающихся. Discord предлагает библиотеку discord.py [17], разработанную для популярного и многофункционального языка программирования Python [18]. Возможности этой библиотеки обширны: поддерживается обработка событий, взаимодействие с пользователями в голосовых и текстовых каналах, управление ролями, а также интеграция с внешними сервисами.

Платформа достаточно удобна для организации учебных серверов, где бот может производить модерацию чатов, выдачу учебных материалов, проводить опросы или тесты. Но здесь же возникает и недостаток, а именно, что платформа требует от пользователя регистрации и привычки к ней, если ранее у пользователя не было опыта взаимодействия с ней.

#### 1.3.8 Сравнительная таблица и выводы

По результатам проведенного сравнительного анализа мессенджеров для разработки чат-ботов были получены результаты, которые отображены в таблице 1.

Таблица 1 – Сравнительный анализ мессенджеров.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Мессенджер | Открытый API | Простота запуска | Интерактивность | Популярность | Ограничения |
| Telegram | Да | Очень высокая | Высокая | Очень высокая | Практически отсутствуют |
| WhatsApp | Частично | Низкая | Низкая | Средняя | Платный доступ, ограничения |
| Viber | Да | Средняя | Средняя | Средняя | Урезанный функционал |
| Вконтакте | Да | Средняя | Средняя | Высокая | Привязка к сообществу |
| Facebook | Да | Сложная | Средняя | Низкая | Сложная модерация |
| Discord | Да | Средняя | Очень высокая | Высокая | Требует отдельный сервер |

По результатам проведенного анализа было принято решение проводить разработку образовательного бота под мессенджер Telegram. Этот мессенджер сочетает в себе простоту разработки, мощный функционал, высокую популярность среди молодой целевой аудитории разрабатываемого продукта и минимальные возможные ограничения как со стороны разработки, так и со стороны пользователей данного приложения.

### 1.4 Telegram как образовательная платформа

#### 1.4.1 Основные возможности Telegram

Telegram – это современный мессенджер, сочетающий в себе инструменты для общения, обмена контентом. Его функциональность значительно превосходит базовые текстовые чаты, что делает этот мессенджер привлекательным для различных сфер жизни людей.

Одним из ключевых преимуществ мессенджера Telegram является поддержка чат-ботов, которые могут быть интегрированы в беседы, чаты и даже каналы. Эти боты способны выполнять заранее заданные своей функциональностью команды, отправлять сообщения, обрабатывать формы, проводить опросы и тестирования, а также выдавать справочную информацию или напоминания.

Платформа поддерживает встроенные кнопки, как в виде клавиатуры над строкой ввода, так и в виде inline-кнопок под сообщением от бота. Это реализует функционал загрузки дополнительного контента без необходимости использования других приложений. Через данные элементы пользователь может, например, выбрать вариант ответа перейти на веб-страничку, проголосовать и т. д.

Необходимо также отметить наличие бота-менеджера всех ботов @BotFather [19], созданных на платформе. Через этот бот происходит настройка и создание всех ботов на платформе.

Кроме этого важно заметить наличие системы уведомлений и подписок через каналы. Telegram обеспечивает целостную и быструю доставку сообщений, а благодаря облачной архитектуре происходит моментальная синхронизация между устройствами.

#### 1.4.2 Особенности Telegram-ботов

Telegram-боты представляют собой виртуальные приложения, встроенные в мессенджер Telegram и способные выполнять автоматические действия по заранее заданному сценарию. Они взаимодействуют с пользователями через стандартный интерфейс мессенджера и выстраивают диалог с пользователем, который направлен на конкретную цель.

Одной из ключевых особенностей Telegram-ботов является гибкость сценариев взаимодействия с пользователем. Разработчик сам задает структуру и стиль общения в виде последовательных шагов:

1) пользователь вводит команду, текст или нажимает на кнопки

2) бот производит проверку полученных данных

3) бот выдает пользователю ответ, который предусмотрен сценарием использования этого бота.

Боты в мессенджере Telegram не требуют установки отдельного ПО – пользователю достаточно лишь перейти по ссылке или найти этого бота в поиске мессенджера Telegram. Это безусловно снижает порог входа для пользователей данного продукта.

Важно заметить, что мессенджер Telegram поддерживает всевозможные типы входных и выходных данных. Это предоставляет расширенные возможности для разработчиков для получения насыщенных и интерактивных форматов взаимодействия пользователя и бота.

Отличительной чертой мессенджера Telegram следует отметить объединение визуального интерфейса с программируемой логикой поведения бота. С помощью inline-кнопок и встроенных меню можно управлять поведением бота без необходимости ввода текста пользователем, что очень удобно для пользователей с недостаточным уровнем компьютерной грамотности. Этот факт особенно важен в сфере получения образования, где простота взаимодействия напрямую влияет на вовлеченность обучающихся.

С технической стороны вопроса реализации Telegram-боты работают через API и возможна их интеграция с внешними базами данных, платежными системами и другими онлайн сервисами. Это дает возможность к созданию полноценных образовательных систем с учетом персонализации обучающегося и ведения статистики для ее дальнейшего использования преподавателем для оценки своих пробелов или успехов и аттестации обучающихся.

#### 1.4.3 Причины выбора Telegram в рамках проекта

При выборе платформы для реализации образовательного чат-бота необходимо учитывать, как технические возможности, так и поведенческие особенности целевой аудитории этого продукта. В рамках данного проекта был выбран мессенджер Telegram, что обосновано рядом объективных причин и прикладных факторов.

Во-первых, Telegram отличается высокой популярностью среди студентов и в принципе молодежи. По данным различных аналитических исследований, этот мессенджер занимает лидирующие позиции по количеству активных пользователей в возрастной группе от 18 до 30 лет. Это означает, что большинство потенциальных пользователей данного продукта уже имеют установленный мессенджер и активно им пользуются в повседневной жизни, а значит для доступа к разработанному боту им не потребуется совершать дополнительных действий.

Во-вторых, Telegram предоставляет открытые и хорошо документированные инструменты для разработки и реализации бота. Bot API позволяет гибко настраивать сценарии взаимодействия, использовать кнопки, меню, обработку полученных от пользователя данных. Это снижает технические барьеры при создании сложных логик поведения и дает свободу разработчику в проектировании продукта.

В совокупности все вышеперечисленные причины делают выбор нереляционной базы данных обоснованным с точки зрения функциональных, архитектурных и эксплуатационных требований. Данный подход способствует достижению целей разрабатываемого проекта – созданию масштабируемой и гибкой образовательной системы, которая адаптирована под динамическую структуру хранимых данных и высокую нагрузку при активном использовании.

### 1.5 Выводы по разделу обзор предметной области

В результате проведенного обзора предметной области была рассмотрена актуальность использования цифровых инструментов в образовательной среде, в частности использования Telegram-ботов в мессенджере Telegram. Современное дистанционное обучение требует гибких и доступных решений, которые способны обеспечить непрерывную коммуникацию и поддержку образовательного процесса. Мессенджеры благодаря своей популярности и удобству, а также активному их использованию в повседневной жизни становятся эффективной средой для внедрения в них образовательных технологий.

Особое внимание в проведенном анализе было уделено мессенджеру Telegram, который сочетает в себе простоту пользовательского интерфейса и широкие технические возможности для разработки чат-ботов. Проведенный анализ показал, что Telegram превосходит ряд альтернативных решений по ключевым параметрам, таким как API, независимость от коммерческих ограничений других мессенджеров и популярность среди молодежной аудитории.

Таким образом, выбор мессенджера Telegram в качестве основной платформы для реализации бота представляется обоснованным и рациональным. Полученные в результате исследования и анализа данные создают необходимую теоретическую базу для проектирования и разработки программного решения, адаптированного под цели и потребности современного дистанционного обучения.

## 2 ФОРМУЛИРОВКА ТРЕБОВАНИЙ К РЕШЕНИЮ И ПОСТАНОВКА ЗАДАЧИ

### 2.1 Постановка задачи

Необходимо разработать образовательного телеграм-бота, ориентированного на обучение студентов материалами, загруженными преподавателями в бота. Также после изучения какого-либо учебного материала студент проходит проверку усвоения полученных знаний путем прохождения тестирования от бота.

### 2.2 Требования к решению

Результат должен обладать нижеперечисленному набору свойств:

1. Порционная подача учебного материала обучающемуся, чтобы он усваивал новый материал, не пугаясь большого объема предоставленных материалов.
2. Реализована система прохождения проверки усвоения выданных ботом материалов в виде тестов с заранее заданными вариантами ответов, либо со свободным ответом обучающегося.
3. Реализовано сохранение данных о пользователе разработанного бота в базе данных для их дальнейшего использования в служебных целях.
4. Реализована система администрирования бота, предназначенная для использования администраторами бота, по большей части ими будут являться преподаватели, то есть авторы загруженных курсов.
5. Сохранение учебных курсов и тестов в базе данных.
6. Разделение учебного функционала чат-бота на дисциплины, курсы, темы, уроки с последующими тестами для структуризации доступных материалов и упрощения доступа к ним.
7. Сохранение ответов обучающихся в базе данных для дальнейшего их использования в статистике разработанного бота.

### 2.3 Обоснование требований к решению

Бот должен обеспечивать порционную подачу учебных материалов, чтобы обучающийся не пугался сразу большому объему учебных материалов. Этот пункт из требований к решению основан на подходе с постепенным представлением информации, который в свою же очередь базируется на принципах когнитивной психологии, где в свою очередь установлено, что получение избыточного объема данных одномоментно может приводить к когнитивной перегрузке пользователя. А вот поэтапное предоставление учебного материала способствует формированию устойчивых знаний, снижет уровень стресса и тревожности у обучающегося. Это очень важно учитывать в условиях удаленного взаимодействия с образовательной системой.

Реализована система прохождения проверки усвоения выданных ботом материалов в виде тестов с заранее заданными вариантами ответов, либо со свободным ответом обучающегося. Требование на наличие тестов для обучающегося обосновано на основании необходимости контроля усвоения учебных материалов как для проверки самим преподавателем качества загруженного материала на основании ответов обучающегося, так и для контроля уровня полученных знаний обучающимися в результате пройденного ими обучения.

Реализовано сохранение данных о пользователе разработанного бота в базе данных для их дальнейшего использования в служебных целях. Данное требование обеспечивает технические возможности для реализации различных сценариев взаимодействия пользователя с ботом. Сюда следует отнести, например, такие сценарии как:

Отслеживание статистики всех пользователей

Отслеживание успехов в освоении учебных материалов конкретного пользователя.

Администрирование разработанного бота

Реализована система администрирования бота, предназначенная для использования администраторами бота, по большей части ими будут являться преподаватели, то есть авторы загруженных курсов. Данное требование объясняется необходимостью разграничения пользователей на роли, которыми они обладают. Преподаватель или администратор, обладая расширенными правами должен иметь возможность осуществления контроля за обучающимися, добавления или удаления учебных дисциплин, курсов, тем, уроков и тестов. Данное требование делает описываемый проект масштабируемым и устойчивым к возможному дальнейшему расширению функционала.

Сохранение учебных курсов и тестов в базе данных. Данное требование необходимо для обеспечения целостности, доступности и многократного использования учебных материалов. Поскольку в данном случае база данных выступает централизованным хранилищем данных. Также использование базы данных позволяет изменять содержимое учебного и проверочного контента из панели администратора без переписывания исходного кода продукта, а, например, использованием одной команды в мессенджере Telegram и далее отправкой текстового сообщения.

Разделение учебного функционала чат-бота на дисциплины, курсы, темы, уроки с последующими тестами для структуризации доступных материалов и упрощения доступа к ним. Данное требование обосновано упрощением навигации по продукту с точки зрения конечного пользователя этого продукта и делает восприятие информации более целостным. Вышеописанная структура приближает дистанционный формат подачи учебного материала к классическим образовательным моделям, которые являются общепринятыми и используются повсеместно в академической среде. Согласно этому требованию открывается возможность для реализации метода последовательного прохождения обучающих курсов с сохранением логических связей в учебном материале.

Сохранение ответов обучающихся в базе данных. Сохранение прогресса пользователей в базу данных позволяет в дальнейшем использовать эти данные для отображения всевозможной статистики, которая будет полезна администраторам этого бота, то есть преподавателям, которым эти данные необходимы, например, для проверки качества своих материалов и проведения контроля усвоения материалов обучающимися.

### 2.4 Обоснование выбора метода решения

В процессе анализа информации и планировании возможной реализации программного решения в рамках поставленной задачи особое внимание было уделено выбору метода, который обеспечивает эффективность, масштабируемость и пользовательскую доступность разработанного продукта одновременно. Учтя специфику продукта, направленного на предоставление учебных материалов через чат-бот в мессенджере, был произведен выбор на использование принципов модульности разрабатываемого продукта и адаптивного и эффективного взаимодействия с пользователем.

Изначальный анализ предметной области показал, что наиболее значимой особенность образовательных систем является необходимость последовательной и понятной подачи учебной информации, а также необходимость наличия механизма контроля усваиваемости учебных материалов. Для достижения вышеперечисленных целей был использован подход, реализующий порционную подачу учебного материала, который достаточно вероятно обеспечит постепенное вовлечение обучающихся в образовательный процесс. Вышеизложенный метод минимизирует информационную нагрузку и способствует качественному усвоению новых знаний, что важно заметить в формате взаимодействия обучающегося с чат-ботом, где отсутствует преподаватель.

Одним из важных ключевых элементов является выбор метода интеграции системы тестирования, которая реализована в виде двух возможных вариантов: тесты с заранее заданными ответами и вопросы, требующие развернутый ответ. Данное решение позволяет повышать точность оценки усвоения учебных материалов и представления преподавателям гибкого инструмента для формирования тестовых заданий, заранее учтя возможность самостоятельного выбора сложности вопросом самим преподавателем в зависимости от целей курса или темы, разработанном им самим же.

## 3 АРХИТЕКТУРА ПРОГРАММНОЙ РЕАЛИЗАЦИИ

### 3.1 Выбор методологии разработки ПО

#### 3.1.1 Общий обзор существующих методологий разработки ПО

Успешная реализация программного продукта в значительной степени зависит не только от технических решений, но и от самой организации процесса разработки ПО.

Современная практика разработки программного обеспечения располагает большим выбором методологий разработки. Среди наиболее распространенных методологий следует сразу выделить:

* Каскадную модель разработки (Waterfall) или водопад [20]
* Инкрементальную модель разработки [21]
* Быструю разработку приложений (RAD) [22]
* Гибкие методологии, например, такие как Agile, Scrum, Kanban [23-25]

Каждая из вышеперечисленных методологий обладает своими преимуществами и ограничениями. Так, например, каскадная модель предполагает строгое следование этапам разработки – от постановки задачи до сопровождения системы – и подходит для проектов с четко определенными и неизменными требованиями. В свою же очередь, гибкие подходы ориентированы на итеративную разработку, где каждый последующий шаг уточняется и корректируется на основе взаимодействия с заказчиком или лидером проекта в виде обратной связи между этими сторонами. При этом взаимодействие производится после получения промежуточных результатов на промежуточных итерациях. В следующих пунктах представлено описание каждой из упомянутых методологий разработки.

#### 3.1.2 Каскадная модель (Waterfall)

Одной из наиболее ранних и формализованных моделей разработки ПО является каскадная методология. Ее основная идея заключается в последовательном прохождении всех этапов жизненного цикла проекта: от формальной постановки задачи, анализа требований заказчика и в последующем формировании технического задания до тестирования и дальнейшего сопровождения разработанного проекта. Для начала разработки по данной методологии требуется строгий план, составлением которого занимается аналитический отдел компании-разработчика, либо привлекается стороняя организация, называемая подрядчиком, которая реализует необходимый план. Каждый этап начинается строго только после завершения предыдущего, что исключает возможность возврата к предыдущим стадиям без пересмотра всего изначально разработанного плана.

Данная модель предполагает наличие полностью определенных требований на момент старта проекта и сводит к минимуму возможную необходимость взаимодействия между заказчиком и разработчиками в процессе работы над проектом. Основным достоинством методологии Waterfall является высокий уровень предсказуемости бюджетов проекта, сроков его реализации и разработки и результат разработки известен заранее. Однако существуют и отрицательные стороны данной методологии: при изменении внешних условий или требований данный подход становится неустойчивым. Поэтому данная методология целесообразна только для проектов с фиксированной спецификацией и стабильной предметной областью.

#### 

#### 3.1.3 Agile-методология

В противоположность каскадной модели Waterfall, методология Agile не требует жесткого планирования, ибо ориентирована на гибкость, быструю адаптацию под изменения в начальных условиях и постоянную обратную связь. В этой методологии особое внимание уделяется взаимодействия между участниками проекта, скорости реакции на изменения и регулярной демонстрации заказчику работающего продукта, хоть и не готового в полной степени.

Этот подход предполагает за собой итерационную разработку проекта, когда на каждом цикле разработки создается рабочий фрагмент системы. Данный подход способствует выявлению недочетов на ранних этапах, что снижает риск возможных проблем в виде масштабных переработок уже реализованного кода. Agile не диктует конкретную реализацию и может быть адаптирован под особенности конкретного проекта, что делает его популярным как в крупных командах разработчиков, так и в индивидуальной разработке.

#### 

#### 3.1.4 Scrum

Методология Scrum представляет собой конкретную реализацию идей Agile методологии и фокусируется на командной работе в рамках четко определенных временных промежутков – спринтов, которые обычно длятся от одной до четырех недель. В рамках каждого спринта команда разработчиков должна реализовать заранее согласованный набор функциональных требований к продукту. Основой этой методологии являюся такие артефакты как Product Backlog (список задач), Sptint Backlog (задачи спринта) и Daily Stand-ups (ежедневные созвоны).

Отличительной чертой данной методологии является высокая степень вовлеченности всех участников, задействованных в реализации проекта, а именно: самого заказчика, разработчиков, системных аналитиков, бизнес аналитиков, архитектора, и т. д. Заказчик проекта может вносить изменения в список задач по мере продвижения команды по итерациям спринтов. Методология хорошо подходит для проектов, где требования переменчивы, а проект разрабатывается поэтапно.

#### 

#### 3.1.5 Kanban

Kanban – еще одна Agile-совместимая методология, делающая акцент на непрерывном потоке задач, визуализации процессов и гибкой постановке приоритетов. В отличие от методологии Scrum, в Kanban нет понятия спринтов, а задачи возникают и добавляются по мере необходимости. Каждая задача проходит через конкретный набор стадий, например, сюда можно включить следующие стадии: запланировано, в работе, на тестировании, выполнено. Возможно отображение всех задач с их состоянием, историей их изменений и комментариями на специальной доске.

Данная методология эффективна для небольших команд по количеству работающих в них сотрудников или даже для одиночных разработчиков, поскольку позволяет оперативно реагировать на изменения, не разрушая структуру происходящего процесса разработки. Методика хорошо применяется в тех проектах, где работа ведется непрерывно и не требует жесткой дисциплины сотрудников в виде спринтов, как это, например, необходимо в случае использования методологии Scrum.

#### 3.2 Сравнительный анализ подходов к разработке ПО

Учитывая особенности разрабатываемого проекта – чат-бота, который предназначается для проведения обучения в онлайн формате, последующей проверки усвоения учебных материалов тоже в онлайн формате можно произвести выделение нескольких ключевых требований к выбору методологии разработки:

* Возможность поэтапной реализации функционала
* Гибкость при изменении требований к продукту
* Регулярная проверка работоспособности промежуточных версий
* Поддержка работы в малой команде или даже в индивидуальном порядке

На основании этих требований был произведен подробный анализ вышеизложенных методологий разработки ПО. Результаты этого анализа представлены в виде таблицы и отражены в таблице 2.

Таблица 2 – Сравнение методологий разработки ПО.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Методология | Гибкость | Удобство для малых команд | Обратная связь | Поддержка изменений |
| Waterfall | Низкая | Не подходит | После завершения проекта | Плохо поддерживается |
| Scrum | Высокая | Подходит | Каждые 1-2 недели | Отлично подходит |
| Kanban | Средняя | Подходит отлично | Постоянная | Хорошо поддерживается |
| Agile | Высокая | Подходит | Быстрая | Отлично |

#### 3.3 Обоснование выбранного подхода к разработке проекта

Проведя учет вышеуказанных критериев для полноценно-успешной реализации проекта была выбрана гибкая методология Agile с элементами методологий Scrum и Kanban. Этот выбор обусловлен недетерминированностью требований, характерной для образовательных проектов. Преподаватели, которые в этом проекте являются заказчиками его разработки в случае успешной реализации будут администраторами бота, должны иметь возможность внесения изменений в структуру курсов, формат текстов или внешний вид интерфейсов. Следовательно, строгое следование заранее определенному плану такому как например Waterfall в данном случае неприменимо.

Кроме того, разработка ведется последовательно, с добавлением функциональных блоков: изначально были реализованы базовые сценарии использования бота его конечными пользователями – обучающимися, затем была добавлена система администрирования, а в завершении разработки был добавлен аналитический блок для получения статистики пользователей в боте. Такой поэтапный подход позволяет своевременно выявлять и устранять потенциальные недостатки и проводить ручное тестирование как отдельных модулей, так и целого продукта даже еще не полнофункционального и на основании всего этого своевременно вносить изменения и улучшения без необходимости полной переработки архитектуры разрабатываемого продукта.

#### 3.4 Особенности реализации методологии разработки в проекте

Реализация выбранного подхода осуществлялась с учетом специфики: разработка велась в индивидуальном порядке и коротких итерациях или спринтах примерно длительностью по неделе. По завершению каждой итерации проводилась ручная проверка работоспособности проекта и согласование с руководителем этого проекта. Полученные замечания от руководителя учитывались при формировании следующего этапа. Таким образом была реализована обратная связь, которая свойственна методологии Scrum, даже в условиях отсутствия полноценной команды разработки. Также в рамках проектирования было применено своевременное сохранение разработанного решения на различных носителях информации во избежание утраты разработанной части продукта.

В результате этого был выбран гибкий подход к управлению разработкой, который позволил обеспечить высокий уровень адаптивности, последовательность разработки, последовательное наращивание функционала и стабильность разрабатываемого решения на всех этапах жизненного цикла программного продукта. Выбор методологии Agile оказался наиболее оправданным с точки зрения динамики требований, ресурсных ограничений и характера проекта.

### 3.5 Сценарии использования разработанного бота

#### 3.5.1 Пользователь – ученик

Описание процесса производится с самого начала взаимодействия пользователя с разработанным ботом.

Действующее лицо: обучающийся

Сценарий использования бота:

1. Обучающийся проходит регистрацию в боте, указывая боту данные о себе, а именно:

* Фамилия
* Имя
* Возраст
* Город проживания

В базу данных происходит запись этих данных и помимо этого еще туда пишется дата и время регистрации и роль пользователя.

1. Следующим шагом загружается меню бота со следующими кнопочками:

* Хочу учиться
* Помощь
* Остановка

3.1) При выборе пункта помощь открываются кнопочки с заданными в исходном коде вопросами. При выборе любого из вопросов бот дает заранее заданный ответ.

3.2) Выбор пункта «Остановка» останавливает бота, и пользователь при загрузке бота снова должен будет ввести команду «/menu» или «/start» для продолжения работы с ботом.

4) При выборе сценария «Хочу учиться» пользователю загружается из базы данных список доступных дисциплин для изучения. Пользователь выбирает любую дисциплину из доступных и дальше возможно 2 варианта развития событий.

5.1) Выбранная дисциплина содержит учебные курсы и тогда пользователь увидит доступные курсы в виде кнопочек с названиями этих курсов.

5.2) Выбранная дисциплина не содержит доступных учебных курсов и тогда пользователь увидит текстовое сообщение от бота в формате «Дисциплина «Название дисциплины» не содержит учебных курсов. Выберите другую дисциплину!».

6) Пользователь, то есть обучающийся, выбирает учебный курс и если в нем существуют доступные темы, то ему отображаются эти темы в виде кнопочек с названиями тем.

6.2) Выбранный учебный курс не содержит доступных тем и тогда пользователь увидит текстовое сообщение от бота в формате «Курс «Название курса» не содержит доступных тем. Выберите другой курс!»

7.1) Ученик выбирает учебную тему и если в ней существуют доступные уроки, то ему отображаются эти уроки в виде кнопочек с названиями этих уроков.

7.2) Выбранная учебная тема не содержит доступных уроков, тогда пользователь увидит текстовое сообщение от бота в формате «Тема «Название темы» не содержит доступных уроков. Выберите другую тему!»

8) Пользователь, нажимая на кнопочку с названием урока приступает к обучению и видит обучающий урок в формате текстовых сообщений от бота. Также после текста урока ему отображается кнопочка «Дальше», если еще существуют порции изучаемого урока. В случае если отображенная порция является последней, то отображается кнопочка с названием «Перейти к тесту».

9.1) В случае если тесты для урока были загружены в бота, то бот переходит к тестированию и задает вопросы обучающемуся в виде текстовых сообщений. В зависимости от типа вопроса, он просит либо нажатие на кнопочку с правильным ответом, либо ввод текстового ответа.

9.2) Если для выбранного урока тестов не существует, то бот отображает пользователю сообщение «Тест к уроку «Название урока» не найден» с кнопочкой «Вернуться к списку доступных уроков».

10) После прохождения тестового вопроса бот отображает обучающемуся в виде текстового сообщения правильный ли он ответ выбрал или написал в зависимости от типа вопроса (с данными вариантами ответов или с открытым ответом)

11) В конце прохождения теста бот отображает пользователю его результат по всем вопросам, относящимся к уроку. Также отображается кнопочка с названием «Вернуться к списку уроков».

#### 3.5.2 Администратор – преподаватель

Администратор может использовать полностью весь функционал бота, доступный пользователю, но отличительной особенностью администраторской роли является доступ к командам, к которым нет доступа у обычных пользователей бота. Новых администраторов могут назначать только действующие администраторы бота, поэтому первым администратором является разработчик бота, имеющий навык работы с базами данных. Назначением администраторской роли обладают только администраторы данного бота.

Ниже предоставлен функционал команд, доступных администратору:

* Изменение роли пользователя бота
* Добавление учебной дисциплины
* Удаление учебной дисциплины
* Добавление учебного курса
* Удаление учебного курса
* Добавление учебной темы
* Удаление учебной темы
* Добавление учебного урока
* Удаление учебного урока
* Добавление теста к уроку
* Удаление теста из урока
* Отображение статистики обучающихся в боте
* Получение информации о пользователях бота
* Получение информации о дисциплинах
* Получение информации о курсах
* Получение информации о темах
* Получение информации об уроках
* Получение информации о тестированиях

#### 3.6 Выбор базы данных

При проектировании архитектуры проекта особенное внимание было уделено выбору подходящей под проект модели хранения и управления данными [26]. Анализ структуры и характера информации, с которой необходимо работать боту показал, что предпочтительным является использование нереляционной БД [27]. Решение было принято на основе ряда факторов, связанных с особенностями функционала и требований к масштабируемости проекта.

Следует обратить особое внимание на то, что структура данных в системе отличается гибкостью и иерархичностью. Учебные материалы, тесты и ответы от пользователей представляют из себя вложенные структуры и плохо поддаются нормализации в рамках реляционной модели БД [28]. Нереляционная модель в свою же очередь представляет собой естественный способ сохранения этих данных в формате документов, что в свою очередь снижает сложность разработки.

Вторым важным фактором является требование к масштабируемости и высокой скорости отклика при увеличении числа пользователей. Нереляционные СУБД спроектированы в отличие от традиционных реляционных СУБД для горизонтального масштабирования, что делает их более устойчивыми в условиях увеличения объема данных или числа запросов к БД. Это важно учесть в случае разработки чат-бота, у которого может быть большое количество одновременных сессий.

Третьим фактором является учет того, что в данном проекте нет сложных транзакционных зависимостей, которые характерны для финансовых и логистических систем.

И последним аргументом в пользу нереляционного подхода реализации является стремление к минимизации накладных расходов на реализацию связей между сущностями. В случае реляционной модели связи между сущностями, структурирующими интерфейс потребовали бы сложных JOIN операций, что тоже снижает производительность. В отличие от реляционных моделей нереляционные позволяют служебную информацию хранить в документе, обеспечивая тем самым быстрый доступ без дополнительных запросов к БД.

#### 3.7 Используемые технологии

Для реализации программного решения в виде Telegram бота-учителя, главная цель использования которого – обучение студентов по заранее загруженным материалам был произведен выбор языка программирования Python, по причине его простоты с точки зрения разработчика и наличия большого количества готовых библиотек в том числе для работы с ботами в мессенджере Telegram.

В качестве основной библиотеки для взаимодействия мессенджера Telegram с разрабатываемым ботом выбор был сделан в пользу python-telegram-bot [29].

В качестве базы данных для хранения необходимой для работы бота информации было принято решение использовать нереляционную модель базы данных, а именно такую СУБД как mongoDB [30].

## 

## 4 ПРОГРАММНАЯ РЕАЛИЗАЦИЯ ПРОЕКТА

### 4.1 Регистрация чат-бота в Telegram

В первую очередь при реализации проекта была выполнена регистрация разработанного чат-бота в мессенджере Telegram. Регистрация выполнялась в уже существующем боте @BotFather в мессенджере Telegram с использованием команды /newbot. Все доступные команды этого бота представлены в таблице 3.

Таблица 3 – доступные команды у Telegram бота @BotFather

|  |  |
| --- | --- |
| Команда | Описание |
| /newbot | Производит создание нового бота в системе |
| /mybots | Отображает информацию об уже созданных ботах |
| /setcommands | Отображает список доступных команд |
| /setabouttext | Присваивает текст в поле «О чат-боте». |
| /setuserpic | Устанавливает загружаемую владельцем картинку в качестве аватарки в бота |
| /setuserpic | Создание списка доступных в боте команд |
| /deletebot | Удаляет выбранного бота |

Следующим шагом после создания бота была выполнена его настройка с использованием команды /setabouttext для того, чтобы бот давал новым пользователям информацию о том, что он делает. А с использованием команды /setserpic было загружено в качестве аватара изображение СПбГЭТУ «ЛЭТИ». Данное изображение представлено на рис. 1.



Рисунок 1 – изображение, использованное в качестве аватарки для разработанного бота.

### 4.2 Проектирование и реализация БД для чат-бота

#### 4.2.1 Общая схема БД

В процессе создания Telegram-бота учителя, ориентация которого направлена на дистанционное взаимодействие с пользователями, которых необходимо обучать, важной задачей стало проектирование системы хранения данных, которая обеспечит как целостность загруженной в бота информации, так и гибкость в работе с разными структурами учебного контента. Для реализации этих требований была спроектирована и внедрена БД. Итоговая диаграмма БД представлена на рис. 2.

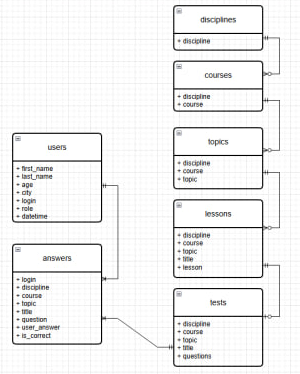


Рисунок 2 – ER-модель БД, используемой в боте.

#### 4.2.2 Описание и назначение реализованных коллекций

Описание производится в порядке реализации и использования этих коллекций (сущностей) в боте.

1. **users**

Коллекция users предназначена для хранения информации о каждом пользователе разработанного бота. Описание полей и их предназначения находятся ниже:

* first\_name – имя пользователя. Указывается самим пользователем при регистрации. Возможное использование – при реализации личного кабинета пользователя в боте и для администраторов в случае необходимости идентификации пользователя при неправильном указании каких-либо других данных.
* last\_name – фамилия пользователя. Указывается самим пользователем при регистрации. Возможное использование – при реализации личного кабинета пользователя в боте и для администраторов в случае необходимости идентификации пользователя при неправильном указании каких-либо других данных.
* age – возраст пользователя. Указывается самим пользователем при регистрации. Потенциальное использование – при сборе статистики администраторами.
* city – город проживания пользователя. Указывается самим пользователем при регистрации. Потенциальное использование возможно администраторами бота при получении статистики бота.
* login – telegram-login пользователя. Используется как первичный ключ в структуре связей в БД. В случае если у пользователя не указан login в настройках своей учетной записи telegram, то бот выводит сообщение с просьбой указания логина в настройках.
* role – роль пользователя в боте. Используется для автоматического определения сценария использования бота. По умолчанию всем новым пользователям предоставляется значение «user».
* datetime – дата и время регистрации пользователя в боте. Использование возможно в служебных целях и предназначено для администраторов.

1. **disciplines**

Коллекция disciplines предназначена для хранения информации, а точнее названия, о доступных в боте учебных дисциплинах. Каждая запись имеет одно поле – название дисциплины.

1. **Courses**

Коллекция courses предназначена для хранения информации о доступных в боте учебных курсах. Каждая запись имеет 2 поля:

* discipline – название учебной дисциплины.
* сourse – название учебного курса.

1. **Topics**

Коллекция topics предназначена для хранения информации о доступных в боте темах обучения. Каждая такая запись имеет 3 поля:

* discipline – название учебной дисциплины.
* course – название учебного курса.
* topic – название учебной темы.

1. **Lessons**

Коллекция lessons предназначена для хранения информации о доступных в боте обучающих уроках. Каждая такая запись имеет 5 полей:

* discipline – название учебной дисциплины.
* course – название учебного курса.
* topic – название темы.
* title – название урока.
* lesson – текст самого урока.

1. **Tests**

Коллекция lessons предназначена для хранения информации о доступных в боте тестах, проходимых пользователями после изучения уроков, к которым эти тесты относятся. Каждая такая запись относится к одному из уроков. Но у урока может не быть тестов, но запись может существовать без вопросов в ней. Структура каждой из таких записей представлена ниже:

* discipline – название учебной дисциплины.
* Course – название учебного курса.
* topic – название учебной темы.
* title – название обучающего урока.
* questions – массив из объектов со следующей структурой:
  + type – тип вопроса: multiple\_choice означает тестовый вопрос с готовыми вариантами ответов. Пользователю необходимо выбрать правильный. Тип text\_answer означает что вопрос требует открытого ответа, то есть в виде сообщения от пользователя.
  + text – Текст задаваемого ботом вопроса.
  + correct\_answer – правильный ответ, заложенный создателем этого вопроса в бота.
  + options – массив, содержащий варианты ответов для пользователя. Используется только в вопросах с заранее заготовленными вариантами ответов.

1. **answers**

Коллекция answers предназначена для реализации сохранения всех ответов пользователей на тестовые вопросы. Используется для получения статистики бота администраторами. Новая запись возникает после каждого прохождения теста пользователем. Структура каждой из записей представлена ниже:

* login – telegram-логин пользователя бота. Используется для идентификации автора решения.
* discipline – название учебной дисциплины.
* course – название учебного курса.
* topic – название учебной темы.
* title – название обучающего урока.
* question – текст вопроса.
* user\_answer – ответ пользователя на вопрос.
* correct\_answer – правильный ответ на вопрос.
* is\_correct – правильно ли ответил пользователь (true; false)

Многократное дублирование данных в данной реализации бота объясняется тем, что бот не рассчитан на большую аудиторию, а на студентов среднестатистического ВУЗа, в котором учится предположим 1000 человек. Зато такая реализация обладает преимуществом в виде избежания возможных ошибок распределения учебных материалов и потери ответов на них.

### 4.3 Служебные файлы и исполняемый файл

#### 4.3.1 Служебный файл config.py

Данный файл был реализован с целью изоляции и хранения служебных переменных. Они определены в этом файле и используются в других файлах через import из Python. Ниже представлены переменные, которые были заданы в конфигурационном файле, и их назначение:

* BOT\_TOKEN – уникальный токен созданного бота. Используется в программе для взаимодействия мессенджера Telegram с ботом.
* MONGO\_URL – адрес на котором развернут сервер БД MongoDB. Используется для подключения к этой самой БД.
* DB\_NAME – имя базы данных, с которой взаимодействует бот.
* COLLECTION\_NAME\_DISCIPLINES – название коллекции, к которой происходит подключение при работе с коллекцией «disciplines».
* COLLECTION\_NAME\_COURSES – название коллекции, к которой происходит подключение при работе с коллекцией «courses».
* COLLECTION\_NAME\_TOPICS – название коллекции, к которой происходит подключение при работе с коллекцией «topics».
* COLLECTION\_NAME\_LESSONS – название коллекции, к которой происходит подключение при работе с коллекцией «lessons».
* COLLECTION\_NAME\_TESTS – название коллекции, к которой происходит подключение при работе с коллекцией «tests».
* COLLECTION\_NAME\_ANSWERS – название коллекции, к которой происходит подключение при работе с коллекцией «answers».

#### 4.3.2 main.py

Данный файл реализован с целью центрального управления работой бота. Здесь установлены все обработчики кнопок, команд, которые реагируют на действия пользователей бота и делают конкретные действия.

Один из модулей работы с ботом тоже реализован в этом файле. Этим модулем является модуль регистрации нового пользователя в боте. Основан он на системе состояний. Изначально происходит проверка на существования у нового пользователя заданного логина. Если логина нет, то ему бот выдает сообщение о том, что «Произошла ошибка. Установите логин в настройках приложения Telegram». Если же логин установлен, то тогда пользователь будет авторизован в боте, если запись содержащая его логин существует в коллекции «users» в БД. Проверка регистрации пользователя реализована в виде функции «check\_user\_db», которая делает запрос к БД и понимает по возвращенному ответу что делать дальше. Если пользователь зарегистрирован, то он видит меню бота. Если не зарегистрирован, то ему загружается регистрация. Регистрация пользователя реализована следующис образом: начинается регистрацию пользователя с приветственного сообщения. Затем в формате диалога получает в виде сообщений информацию от пользователя о пользователе: имя, фамилию, возраст, город проживания. На каждом этапе диалога бот проверяет полученную от пользователя информацию с помощью регулярных выражений для защиты от возможных NoSQL инъекций и корректности предоставляемой боту информации.

### 4.3 Реализация классов для работы бота.

#### 4.3.1 Класс Database

Данный класс реализован с целью взаимодействия с базой данных MongoDB из бота. Ниже представлены поля и методы этого класса и объяснение с какой целью это реализовано:

* self.collection\_disciplines – коллекция дисциплин.
* self.collection\_courses – коллекция курсов.
* self.collection\_topics – коллекция тем.
* self.collection\_lessons – коллекция уроков.
* self.collection\_tests – коллекция вопросов.
* self.collection\_users – коллекция пользователей.
* self.collection\_answers – коллекция с ответами.
* add\_user – добавление нового пользователя в БД. Используется при успешной регистрации нового пользователя.
* get\_disciplines – получение списка дисциплин из БД. Используется при отображении доступных дисциплин при входе пользователя в обучение.
* get\_courses – получение списка курсов, доступных по выбранной дисциплине. Используется при отображении доступных курсов в выбранной дисциплине.
* get\_topics – получение списка тем, доступных в выбранном курсе. Используется при отображении доступных для изучения тем в выбранном курсе.
* get\_lessons – получение списка уроков, доступных в выбранной теме. Используется при отображении доступных для просмотра уроков в выбранной теме.
* get\_lesson – получение одного урока. Загружает учебный материал из БД, чтобы бот затем отобразил его в виде порций ученику.
* get\_lesson\_test – получение тестов, доступных по конкретному уроку.
* is\_user – проверяет наличие информации о пользователе в БД. Используется и при добавлении нового пользователя и при различных проверках на существование пользователя.
* get\_users – получение информации из БД о всех пользователях бота. Используется администраторами для просмотра информации об учениках.
* admin\_get\_disciplines – получение всей информации о дисциплинах для администраторов бота.
* admin\_get\_courses – получение всей информации о курсах для администраторов бота.
* admin\_get\_topics – получение всей информации о темах для администраторов бота.
* admin\_get\_lessons – получение всей информации об уроках для администраторов бота.
* get\_tests – получение всех тестов, доступных в боте. Используется для предоставления информации администраторам бота.
* is\_admin – проверка является ли пользователь администратором. Используется в служебных целях.
* add\_admin – изменение роли пользователя до администратора. Доступ только у администраторов.
* del\_admin – изменение роли администратора до пользователя. Доступ только у администраторов.
* del\_user – удаление пользователя из списка пользователей бота. Доступ только у администраторов.
* is\_discipline – проверка существования заданной дисциплины в боте. Служебное использование.
* add\_discipline – добавление дисциплины в БД. Доступ только у администраторов.
* del\_discipline – удаление дисциплины из БД. Доступ только у администраторов.
* is\_course – проверка на существование курса в БД. Доступ только у администраторов.
* add\_course – добавление нового курса в БД. Доступ только у администраторов.
* del\_course – удаление курса из БД. Доступ только у администраторов.
* is\_topic – проверка на существование темы в БД. Доступ только у администраторов.
* add\_topic – добавление новой темы в БД. Доступ только у администраторов.
* del\_topic – удаление конкретной темы из БД. Доступ только у администраторов.
* is\_lesson – проверка на существование урока в БД. Доступ только у администраторов.
* add\_lesson – добавление урока в БД. Доступ только у администраторов.
* del\_lesson – удаление урока из БД. Доступ только у администраторов.
* is\_test – проверка на существование теста в БД. Служебное использование.
* is\_question – проверка на существование в БД вопроса. Служебное использование.
* add\_test – добавление нового теста в конкретный урок в БД. Доступ только у администраторов.
* del\_test – удаление конкретного теста из конкретного урока из БД. Доступ только у администраторов.
* add\_result\_test – выгрузка всех успехов пользователей из БД по прохождению всех тестов. Доступ только у администраторов.

#### 4.3.2 Класс User

Класс пользователя был реализован с целью хранения информации о пользователе во время его работы с ботом. Здесь хранится вся информация о пользователе бота. Предназначен для правильной и безошибочной работы бота. Реализованы нижепредставленные поля класса:

* self.login – telegram-логин пользователя.
* self.first\_name – имя пользователя.
* self.last\_name – фамилия пользователя.
* self.age – возраст пользователя.
* self.city – город происхождения пользователя.
* self.discipline – текущая дисциплина пользователя.
* self.topic – текущая выбранная тема пользователя.
* self.course – текущий курс пользователя.
* self.lesson – текущий урок, изучаемый пользователем.

Вышеизложенная реализация класса позволяет четко управлять работой разработанного бота без ошибок.

### 4.4 Функции, реализующие функционал работы бота

Разработка Telegram-бота предполагает реализацию набора функций, каждая из которых отвечает за определённый аспект взаимодействия с пользователем и обработку поступающих данных. Для обеспечения стабильной и предсказуемой работы чат-бота была реализована модульная структура кода, при которой логика отдельных процессов вынесена в специализированные функции. Это позволило упростить поддержку проекта, повысить читаемость кода и обеспечить масштабируемость функционала в будущем.

Функции в рамках программной реализации можно условно разделить на несколько категорий: обработка пользовательских команд, управление обучающим процессом, реализация тестирования, административные возможности, а также функции, отвечающие за взаимодействие с базой данных и обработку ошибок. Ниже приводится описание ключевых функций, участвующих в обеспечении логики работы программного решения.

#### 4.4.1 Пользовательские функции

Ниже представлены в виде списка разработанные функции, с которыми работает пользователь:

* menu – отображение главного меню бота из которого пользователь начинает свои дальнейшие действия.
* teacher – функция, вызывающаяся после нажатия пользователем на кнопку с названием «Хочу учиться». Если в БД есть какие-либо дисциплины, то пользователю выдадутся в виде кнопочек доступные дисциплины, в противном случае – сообщение об отсутствии дисциплин в БД. Конкретный результат представлен на рис. 3.

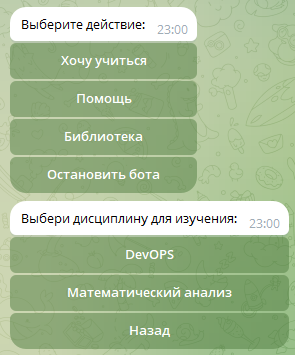


Рисунок 3 – Главное меню бота.

* handle\_teacher\_discipline – функция, обрабатывающая нажатие на кнопку с названием дисциплины. Далее она получает из БД доступные курсы по выбранной пользователем дисциплине. В случае, если по выбранной дисциплине не существует курсов, то пользователь получает сообщение об ошибке, что курсы в выбранной им дисциплине не найдены.
* handle\_course\_selection – функция, обрабатывающая нажатие пользователя на кнопку с названием курса. Далее она получает из БД доступные темы по выбранному пользователем курсу и отображает их пользователю в виде кнопочек. В случае, если по выбранному курсу нет тем, то пользователь получает сообщение от бота об ошибке, что в выбранном курсе темы не найдены.
* handle\_topic\_selection – функция, обрабатывающая нажатие на кнопку с названием темы. Далее она получает из БД доступные уроки по выбранной пользователем теме и отображает названия загруженных уроков пользователю в виде кнопочек с названиями этих уроков. В противном случае, если в выбранной пользователем теме не существует уроков, то бот возвращает пользователю сообщение об ошибке, что в выбранной теме уроки не найдены.
* handle\_lesson\_selection – функция, обрабатывающая нажатие на кнопку с названием урока. Далее она получает из БД сам урок и сразу происходит загрузка проверочных вопросов, если они конечно же существуют в БД. В противном случае пользователь проходит порционный тест и ему отображается кнопка «Вернуться к списку уроков». Если тесты есть, то кнопка будет «Перейти к тесту».
* handle\_next\_part – функция, обрабатывающая нажатие на кнопку далее во время прохождения тестирования пользователем.
* send\_lesson\_part – функция, реализующая показ следующей части урока, если он еще не закончился.
* finish\_lesson – функция, реализующая завершение урока, отображение кнопки «Начать тест» или «Вернуться к списку уроков»
* handle\_text\_answer – функция, обрабатывающая текстовый ответ от пользователя на тестовый вопрос.
* finish\_test – функция, отображающая пользователю результат тестирования в виде текстового сообщения от бота.
* show\_lessons – функция, загружающая и обрабатывающая уроки из БД.
* show\_courses – функция, загружающая учебные курсы из БД и обрабатывающая их.
* show\_topics - функция, загружающая учебные темы из БД и обрабатывающая их.
* handle\_back\_buttons – функция, обрабатывающая нажатия на кнопки «Назад» в разделе обучения.
* help – функция, реализующая раздел «Помощь» в главном меню.
* button\_handler\_help – функция, обрабатывающая и дающая ответы пользователю на заданные заранее вопросы по организации учебного бота.

#### 4.4.2 Администраторские функции

Ниже представлены в виде списка разработанные функции, с которыми работает администратор бота:

* admin\_panel – функция вызывающаяся при обработке команды «/admin» от администратора. Если ученик вводит эту команду, то он получает сообщение об ошибке прав доступа. Функция отправляет сообщение от бота обо всех возможных администраторских командах и их необходимых параметрах.
* add\_admin – функция, обрабатывающая команду «/add\_admin» <логин>. Результатом вызова этой команды является добавление пользователя с указанным в параметре команды логином в список администраторов бота. Команда доступна только администраторам бота.
* del\_admin – функция, обрабатывающая команду «/del\_admin» <логин>. Результатом вызова этой команды является удаление пользователя с указанным в параметре команды логином из списка администраторов бота. Команда доступна только администраторам бота.
* del\_user – функция, обрабатывающая команду «/del\_user» <логин>. Результатом вызова этой команды является удаление информации о пользователе из бота. Команда доступна только администраторам бота.
* get\_users – функция, обрабатывающая команду «/get\_users». Результатом вызова этой команды является получение и отображение в виде сообщений от бота всей информации о пользователях бота. Команда доступна только администраторам бота.
* get\_disciplines – функция, обрабатывающая команду «/get\_disciplines». Результатом вызова этой команды является получение и отображение дисциплин из бота. Команда доступна только администраторам бота.
* get\_courses – функция, обрабатывающая команду «/get\_courses». Результатом вызова этой команды является получение и отображение информации об учебных курсах в формате текстовых сообщений от бота. Команда доступна только администраторам бота.
* get\_topics – функция, обрабатывающая команду «/get\_topics». Результатом вызова этой команды является получение и отображение информации об учебных дисциплинах в формате текстовых сообщений от бота. Команда доступна только администраторам бота.
* get\_lessons – функция, обрабатывающая команду «/get\_lessons». Результатом вызова этой команды является получение и отображение информации об уроках в формате текстовых сообщений от бота. Команда доступна только администраторам бота.
* get\_tests – функция, обрабатывающая команду «/get\_tests». Результатом вызова этой команды является получение и отображение информации обо всех тестах в формате текстовых сообщений от бота. Команда доступна только администраторам бота.
* add\_discipline – функция, обрабатывающая команду «/add\_discipline» <Название дисциплины>. Результатом вызова этой команды является добавление новой дисциплины в разработанную образовательную систему. Команда доступна только администраторам бота.
* del\_discipline – функция, обрабатывающая команду «/del\_discipline» <Название дисциплины>. Результатом вызова этой команды является удаление из системы введенной дисциплины. Команда доступна только администраторам бота.
* add\_course - функция, обрабатывающая команду «/add\_course» <Название дисциплины> <Название курса>. Результатом вызова этой команды является добавление нового курса в разработанную образовательную систему. Команда доступна только администраторам бота.
* del\_course - функция, обрабатывающая команду «/del\_course» <Название дисциплины> <Название курса>. Результатом вызова этой команды является удаление курса из разработанной образовательной системы. Команда доступна только администраторам бота.
* add\_topic - функция, обрабатывающая команду «/add\_topic» <Название дисциплины> <Название курса> <Название темы>. Результатом вызова этой команды является добавление новой темы в разработанную образовательную систему. Команда доступна только администраторам бота.
* del\_topic - функция, обрабатывающая команду «/del\_topic» <Название дисциплины> <Название курса> <Название темы>. Результатом вызова этой команды является удаление заданной темы из разработанной образовательной системы. Команда доступна только администраторам бота.
* add\_lesson – функция, обрабатывающая команду «/add\_lesson» <Название дисциплины> <Название курса> <Название темы> <Название урока>. Результатом вызова этой команды является добавление нового урока в разработанную образовательную систему. Затем вызывается функция обработки текстового сообщения от администратора для ввода текста урока. Команда доступна только администраторам бота.
* del\_lesson – функция, обрабатывающая команду «/del\_lesson» <Название дисциплины> <Название курса> <Название темы> <Название урока>. Результатом вызова этой команды является удаление урока из разработанной образовательную систему. Команда доступна только администраторам бота.
* add\_test – функция, обрабатывающая команду «/add\_test» <Название дисциплины> <Название курса> <Название темы> <Название урока>. Функция добавляет в бота новый вопрос посредством диалога с администратором. Функция доступна только для администраторов.
* del\_test – функция, обрабатывающая команду «/del\_test» <Название дисциплины> <Название курса> <Название темы> <Название урока>. Функция удаляет тест из бота. Функция доступна только для администраторов.
* get\_all\_user\_results – функция, обрабатывающая команду «/all\_results». Функция выгружает все ответы пользователей на проверочные вопросы и собирает всю информацию в виде .csv файла и бот отправляет этот файл администратору. Функция доступна только для администраторов бота.

#### 4.4.3 Остальные немаловажные функции

В данном разделе описаны немаловажные функции для корректной и стабильной работы разработанного Telegram-бота. Ниже в списке представлены названия этих функций и описано их значение для проекта:

* send\_large\_message – функция, необходимая для отправки длинных текстовых сообщений. Такие сообщения возможны, если объем данных начнет расти при увеличении количества учеников, уроков и тестов.
* handler\_add\_lesson – обработчик ввода текста для загрузки урока в бота. Необходим, поскольку в главной функции находится один один приемник текстовых сообщений от пользователя, и чтобы бот различал к чему эти сообщения относятся необходимо разделить логику обработки.
* handle\_text\_answers – функция, являющаяся главным обработчиком текстовых сообщений, получаемых от пользователей и администраторов бота. В зависимости от состояния пользователя функция вызывает разные функции. Используется при получении ответа от пользователя на тестовый вопрос или при загрузке учебного материала (урока или теста) в бота.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы была разработана образовательная система в виде Telegram-бота, предназначенного для дистанционного обучения студентов. Основной целью проекта являлось создание удобного и функционального инструмента, который позволяет структурированно подавать учебные материалы, проводить тестирование для проверки усвоения знаний и сохранять результаты для дальнейшего анализа.

Реализованный чат-бот обладает следующими ключевыми особенностями:

Порционная подача материала: Учебные материалы разделены на дисциплины, курсы, темы и уроки, что обеспечивает удобное и последовательное обучение.

Система тестирования: Бот поддерживает два типа вопросов — с вариантами ответов и открытые, что позволяет гибко оценивать знания студентов.

Административный функционал: Преподаватели могут добавлять, редактировать и удалять учебные материалы, а также получать статистику по успеваемости студентов.

Использование современных технологий: Для реализации бота были выбраны Python, библиотека python-telegram-bot и нереляционная СУБД MongoDB, что обеспечило гибкость, масштабируемость и высокую производительность системы.

Проведенный анализ предметной области показал, что мессенджер Telegram является удобной платформой для подобного решения благодаря своей популярности среди молодежи, открытому API и широким возможностям для разработки ботов. Сравнение с альтернативными мессенджерами подтвердило преимущества Telegram в контексте образовательных задач.

Разработанный бот успешно прошел этапы проектирования, реализации и тестирования. Его практическая значимость заключается в возможности применения в учебных заведениях для автоматизации процесса обучения и контроля знаний. Система может быть легко адаптирована для других дисциплин и курсов, что делает её универсальным инструментом для дистанционного образования.

В перспективе проект может быть расширен за счет интеграции с другими образовательными платформами, добавления мультимедийных материалов (видео, аудио) и реализации более сложных алгоритмов анализа успеваемости студентов. Таким образом, работа вносит вклад в развитие дистанционных образовательных технологий и демонстрирует эффективность использования чат-ботов в современном обучении.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Вознесенская Е. В. Дистанционное обучение – история развития и современные тенденции в образовательном пространстве. // Наука-образованию. URL: <https://cyberleninka.ru/article/n/distantsionnoe-obuchenie-istoriya-razvitiya-i-sovremennye-tendentsii-v-obrazovatelnom-prostranstve/viewer> (дата обращения: 01.04.25)
2. Еременко М. С. Анализ мотивационной структуры личности студентов и особенностей проявления их онлайн-активности // Психология личности. URL: <https://naukaru.ru/temp/e9a3b8f8466b16015db4bb1f3818675e.pdf> (дата обращения: 01.04.25)
3. Шукшина Л. В. Особенности обучения и мотивации студентов в условиях онлайн взаимодействия. // Современное педагоческое образование. URL: <https://cyberleninka.ru/article/n/osobennosti-obucheniya-i-motivatsiya-studentov-v-usloviyah-onlayn-vzaimodeystviya/viewer> (дата обращения: 02.04.25)
4. Кузьмина Л. В. Преимущества и недостатки дистанционного обучения. // Педагогические науки. URL: <https://cyberleninka.ru/article/n/preimuschestva-i-nedostatki-distantsionnogo-obucheniya/viewer> (дата обращения 03.04.25)
5. Начарова Л. И. Мессенджеры: новые медиа или эволюционный этап развития социальных сетей? // Мир науки, культуры, образования. URL: <https://cyberleninka.ru/article/n/messendzhery-novye-media-ili-evolyutsionnyy-etap-razvitiya-sotsialnyh-setey/viewer> (дата обращения: 05.04.25)
6. Мессенджер Telegram [Электронный ресурс]. URL: <https://core.telegram.org/> (дата обращения 10.04.25)
7. Документация Bot API [Электронный ресурс]. URL: <https://core.telegram.org/bots/api> (дата обращения 10.04.25)
8. Документация WhatsApp [Электронный ресурс]. URL: <https://wappi.pro/api-documentation> (дата обращения 13.04.25)
9. Документация WhatsApp Business API [Электронный ресурс]. URL: <https://docs.1msg.io/> (дата обращения 13.04.25)
10. Документация Viber [Электронный ресурс]. URL: <https://developers.viber.com/docs/> (дата обращения 13.04.25)
11. Документация Viber API [Электронный ресурс]. URL: <https://developers.viber.com/docs/api/> (дата обращения 14.04.25)
12. Мессенджер Вконтакте [Электронный ресурс]. URL: <https://vk.com/> (дата обращения 14.04.25)
13. Приложение VK Mini Apps [Электронный ресурс]. URL: <https://vk.com/mini-apps> (дата обращения 14.04.25)
14. Мессенджер Facebook [Электронный ресурс]. URL: <http://facebook.com.vn/> (дата обращения 14.04.25)
15. Документация для разработки приложений под месседжер Facebook [Электронный ресурс]. URL: <https://api-platform.com/docs/symfony/messenger/> (дата обращения 14.04.25)
16. Приложение Discord [Электронный ресурс]. URL: <https://discord.com> (дата обращения 15.04.25)
17. Документация по модулю discord.py [Электронный ресурс]. URL: <https://github.com/denisnumb/discord-py-guide/blob/main/discord-py.md> (дата обращения 15.04.25)
18. Документация по языку программирования Python [Электронный ресурс]. URL: <https://www.python.org/doc/> (дата обращения 15.04.25)
19. Прародитель всех telegram-ботов [Электронный ресурс]. URL: @BotFather (дата обращения 16.04.25)
20. Шахина Ирина Вячеславовна, Муллин Алексей Александрович, РАЗНИЦА МЕЖДУ МЕТОДОЛОГИЯМИ AGILE И WATERFALL [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/agile-vs-waterfall-raznitsa-mezhdu-metodologiyami/viewer> (дата обращения 18.04.25)
21. Н. Д. Андреев, Ф. А. Новиков, ИНКРЕМЕНТАЛЬНЫЙ ПРЕДМЕТНО-ОРИЕНТИРОВАННЫЙ ПРОЦЕСС РАЗРАБОТКИ ПРИКЛАДНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/inkrementalnyy-predmetno-orientirovannyy-protsess-razrabotki-prikladnogo-programmnogo-obespecheniya/viewer> (дата обращения 18.04.25)
22. А. П. Магутина, ИССЛЕДОВАНИЕ ПОТЕНЦИАЛА AR LOW-CODE ПЛАТФОРМ ДЛЯ БЫСТРОЙ РАЗРАБОТКИ ИММЕРСИВНЫХ ПРИЛОЖЕНИЙ [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/issledovanie-potentsiala-ar-low-code-platform-dlya-bystroy-razrabotki-immersivnyh-prilozheniy/viewer> (дата обращения 18.04.25)
23. Д. В. Карпов, ГИБКАЯ МЕТОДОЛОГИЯ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/gibkaya-metodologiya-razrabotki-programmnogo-obespecheniya/viewer> (дата обращения 18.04.25)
24. Арман А. Преимущества гибких методологий управления проектами при разработке софта [Электронный ресурс]. URL: <https://www.researchgate.net/publication/370399618_Preimusestva_gibkih_metodologij_upravlenia_proektami_pri_razrabotke_softa> (дата обращения: 18.04.25)
25. Мухамадеева Рената Маратовна, Брюхова Елена Михайловна, СОВРЕМЕННОЕ СОСТОЯНИЕ И ПРОБЛЕМЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ С ИСПОЛЬЗОВАНИЕМ ГИБКИХ МЕТОДОЛОГИЙ [Электронный ресурс]. URL: <https://www.rypravlenie.ru/wp-content/uploads/2022/03/09-Appendix_Mukhamadeeva_Bryukhova.pdf> (дата обращения: 18.04.25)
26. Скрябин Илья Александрович, ПОДХОД К ВЫБОРУ СИСТЕМЫ УПРАВЛЕНИЯ БАЗАМИ ДАННЫХ НА ПРЕДПРИЯТИИ [Электронный ресурс]. URL: <https://web.snauka.ru/issues/2021/11/97021> (дата обращения 22.04.25)
27. Галигузова Е. В., Илларионова Ю. Е., СРАВНЕНИЕ РЕЛЯЦИОННЫХ И НЕРЕЛЯЦИОННЫХ СУБД [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/sravnenie-relyatsionnyh-i-nerelyatsionnyh-subd/viewer> (дата обращения: 22.04.25)
28. Васильева К. Н., Хусаинова Г. Я., РЕЛЯЦИОННЫЕ БАЗЫ ДАННЫХ [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/relyatsionnye-bazy-dannyh/viewer> (дата обращения 22.04.25)
29. Документация по библиотеке python-telegram-bot для разработки ботов на языке Python [Электронный ресурс]. URL: <https://github.com/python-telegram-bot/python-telegram-bot> (дата обращения 23.04.25)
30. Документация по mongoDB [Электронный ресурс]. URL: <https://github.com/mongodb/docs> (дата обращения 24.04.25)

## ПРИЛОЖЕНИЕ А