

Введение

В условиях активного развития цифровых технологий и расширения онлайн-взаимодействия всё большее значение приобретают навыки эффективной коммуникации и ведения переговоров. Способность грамотно формулировать свои мысли, аргументировать позицию и находить взаимовыгодные решения становится важным элементом в учебной и профессиональной деятельности. Однако традиционные формы обучения данным навыкам часто требуют значительных временных и организационных затрат, что делает актуальным поиск гибких и доступных инструментов практики.

Целью проекта является разработка чат-бота в мессенджере Telegram, который позволит пользователям тренировать навыки ведения переговоров и диалогов в интерактивной форме. Бот будет предоставлять теоретические материалы, демонстрационные примеры, а также сценарии для практических тренировок с автоматической обратной связью. Такой формат обучения удобен, доступен с любого устройства и не требует специальной подготовки. Далее приведено краткое описание разделов пояснительной записки.

Первый раздел носит название «Анализ задачи». В нем можно ознакомиться с постановкой задачи, которая включает в себя: исследование предметной области поставленной задачи, инструменты разработки (будет рассмотрена среда, в которой создается данный проект), выбор модели жизненного цикла программного обеспечения. Также в этом разделе можно узнать о том, как данная задача решается в настоящее время. Все входные и выходные данные тоже будут описаны в первом разделе.

В разделе «Проектирование задачи» будут рассмотрены основные аспекты разработки ресурса. Здесь можно узнать об организации данных в контексте среды разработки. В данном разделе будет описан пользовательский интерфейс, составлены алгоритмы процесса обработки информации.

«Реализация» – это третий раздел отчета, в котором описываются все элементы и объекты, которые будут использованы при реализации данного telegram-бота.

Четвёртый раздел – «Тестирование». В нем будет описано полное и функциональное тестирование. Будут смоделированы все возможные действия пользователя при работе с чат ботом.

В разделе «Руководство пользователя» будет описано назначение, область применения, среда функционирования данного программного продукта.

«Заключение» будет содержать краткую формулировку задачи, результаты проделанной работы, описание использованных методов и средств.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		3

В разделе «Список использованных источников» будет приведён список используемых при разработке источников.

В приложениях к пояснительной записке будут приведены UX проектирование и тест-кейсы.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
						4
Изм	Лист	№ докум.	Подпис	Дат		

1 Анализ задачи

1.1 Постановка задачи

1. Организационно-экономическая сущность задачи.

Темой данного проекта является telegram-бота «Переговорный репетитор».

Целью разработки является создание программного продукта — чат-бота для мессенджера Telegram, предназначенного для интерактивного обучения пользователя основам техник ведения переговоров.

Программный продукт предназначен для индивидуальных пользователей, желающих развить коммуникативные и переговорные навыки; студентов, изучающих дисциплины, связанные с деловыми коммуникациями, менеджментом или психологией; корпоративных сотрудников, участвующих в деловых переговорах, продажах, переговорах с клиентами и партнёрами.

Периодичность использования чат-бота предполагается по мере необходимости — для подготовки к конкретным переговорам или освежения знаний.

Источниками данных выступают обучающие материалы, расположенные в сети Интернет.

Эти связи обеспечат централизованный сбор статистики, анализ результатов обучения и интеграцию в существующую образовательную инфраструктуру.

На рынке существуют следующие решения, частично пересекающиеся по функционалу:

- Skillbox Bot, Stepik Bot, Coursera Bot — чат-боты для дистанционного обучения, предоставляющие материалы и тесты, но не ориентированные именно на переговоры.
- Soft Skills Coach Bot — тематический бот для развития коммуникационных навыков, но с ограниченным контентом и без адаптации под пользователя.

2. Функциональные требования

Чат-бот для Telegram должен выполнять следующие функции:

- Навигация по разделам
- Быстрые команды и кнопки для удобного перехода.
- Обучающий модуль (Теоретический блок)
- Представление структурированных материалов по темам.
- Возможность последовательного или выборочного изучения тем.
- Отображение примеров и кейсов.
- Короткие практические упражнения с выбором вариантов ответов.
- Автоматическая проверка.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		5

- Оценка уровня знаний.
- Диалоговые сценарии, имитирующие переговоры с виртуальным собеседником.
- Выдача оценки (баллов) и комментариев по результатам.

3. Описание входной (исходной) информации.

Данные, поступающие от пользователя, включают идентификационные данные (Telegram ID, имя пользователя/username), инициализирующие команды (/start), навигационные запросы (выбор пунктов меню клавиатуры), ответы на вопросы тестирования и практических заданий (выбор вариантов ответов), текстовые сообщения для обратной связи и файлы подтверждения оплаты (изображения/скриншоты).

Со стороны администратора входной информацией являются команды управления контентом, тексты обновленных лекций и медиафайлы для кейсов, а также команды подтверждения платежей.

Формы представления:

Текстовые сообщения, нажатия на кнопки (Callback/Reply клавиатуры), мультимедийные файлы (изображения).

4. Описание результатной (выходной) информации.

Включает теоретические блоки по техникам переговоров, последовательности тестовых вопросов, мгновенную обратную связь (реакцию) на выбранный вариант ответа с пояснением правильности решения, итоговые результаты тестирования (количество набранных баллов), статус доступа к функционалу (платный/бесплатный), ответы администратора на вопросы пользователей.

Формы представления:

Текстовые сообщения (форматированный HTML), экранные меню и кнопки, мультимедийные материалы (фото/видео кейсы).

5. Описание условно-постоянной информации.

Справочники техник переговоров: структурированные описания методов («Три да», «Маленькие ходы», «Принципиал», «Игра в одни ворота», «Если вместо нет»).

База тестовых вопросов: набор вопросов с вариантами ответов и указанием правильного ключа для блока теоретического тестирования.

Библиотека сценариев (скриптов): разветвленные диалоговые структуры для режима «Отработка техник», содержащие реплики бота, варианты ответов пользователя и соответствующие реакции (feedback).

Конфигурационные данные: список ID администраторов, токены доступа, платежные реквизиты.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	№докум.	Подпис	Дат		6

Файлы хранения состояний: JSON-файлы, содержащие базу пользователей (users.json), архив обратной связи (feedback.json) и динамический контент (content.json).

6. Нефункциональные (эксплуатационные) требования.

Требования к применению (интерфейсу):

Качество взаимодействия обеспечивается интуитивно понятным меню навигации в чате Telegram. Минимальное количество действий для доступа к обучающему контенту (не более 2-3 кликов от главного меню). Обеспечение защиты от некорректного ввода путем использования кнопочных интерфейсов (ReplyKeyboardMarkup) вместо ручного ввода текста в режимах тестирования.

Требования к производительности:

Время реакции бота на команду пользователя не должно превышать 2 секунд при стабильном интернет-соединении. Система должна корректно обрабатывать одновременные запросы от нескольких пользователей (обеспечивается многопоточной архитектурой библиотеки telebot). Надежное сохранение данных пользователей (прогресс, оплата) в JSON-файлы в реальном времени для предотвращения потери данных при перезапуске.

Требования к реализации:

Технические стандарты включают:

Язык программирования: Python (рекомендуемая версия 3.8 и выше).

Основная библиотека взаимодействия с API Telegram: pyTelegramBotAPI (telebot).

Система хранения данных: файловая система (формат JSON) для обеспечения переносимости и простоты структуры без использования тяжеловесных СУБД.

Операционная среда: кроссплатформенная совместимость (Windows/Linux/macOS), возможность развертывания на облачных серверах.

Используемые модули: json (сериализация данных), os (работа с файловой системой), random (генерация вариантов ответов), time (логирование).

1.2 Диаграмма вариантов использования

Диаграмма вариантов использования описывает функциональное назначение системы и взаимодействие актеров с прецедентами использования. Разработанная модель включает трех актеров: Гостя, Пользователя и Администратора, каждый из которых обладает своим набором прав и доступных действий.

Взаимодействие с системой начинается с роли «Гость» — пользователя, который еще не оплатил доступ к полному функционалу. Ему предоставляется

возможность получить приветственное сообщение при запуске бота, использовать навигацию главного меню, просматривать базовые уроки по переговорам (разделы теории и кейсов), а также инициировать процесс оплаты для расширения прав доступа. Роль «Пользователь» присваивается авторизованным участникам после подтверждения платежа. Помимо базовых функций, Пользователь получает доступ к разделу «Практикум», включающему прохождение тестов по теории с просмотром результатов и интерактивную отработку техник, таких как метод принципиала, техника «Если вместо нет», «Игра в одни ворота», «Маленькие ходы» и «Три да».

Управление системой осуществляет актер «Администратор», имеющий доступ к специальной панели управления. В его полномочия входит проверка и подтверждение оплат пользователей, работа с входящими сообщениями (ответ на вопросы и удаление переписки), а также редактирование образовательного контента — изменение текстов теоретических блоков и материалов кейсов без необходимости вмешательства в исходный код программы. Такая структура обеспечивает четкое разделение зон ответственности и гибкость образовательного процесса.

1.3 Инструменты разработки

Для реализации программного продукта — Telegram-бота «Репетитор по переговорам» — был выбран язык программирования Python, так как он обладает развитой экосистемой модулей для работы с API мессенджеров, лаконичным синтаксисом и обеспечивает высокую скорость разработки. В качестве интегрированной среды разработки (IDE) использовался редактор кода VS Code, предоставляющий удобные средства отладки и подсветки синтаксиса. Основным инструментом взаимодействия с Telegram Bot API выступила библиотека pyTelegramBotAPI (telebot), которая обеспечивает обработку входящих сообщений и управление интерфейсом. Хранение данных пользователей и контента организовано с помощью встроенного модуля JSON, что позволило создать легковесную файловую базу данных.

Для проектирования архитектуры системы и построения диаграмм использовался веб-ресурс Draw.io, а оформление программной документации выполнялось в текстовом редакторе Microsoft Word. Разработка и тестирование программного продукта велись на персональном компьютере под управлением операционной системы Windows 11, оснащенный процессором уровня AMD Ryzen 5 и 16 ГБ оперативной памяти.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		8

1.4 Разработка плана работы над проектом

Для планирования этапов создания бота и контроля сроков выполнения задач было использовано табличное приложение Microsoft Office Excel. Это позволило структурировать процесс разработки и визуализировать ключевые вехи проекта.

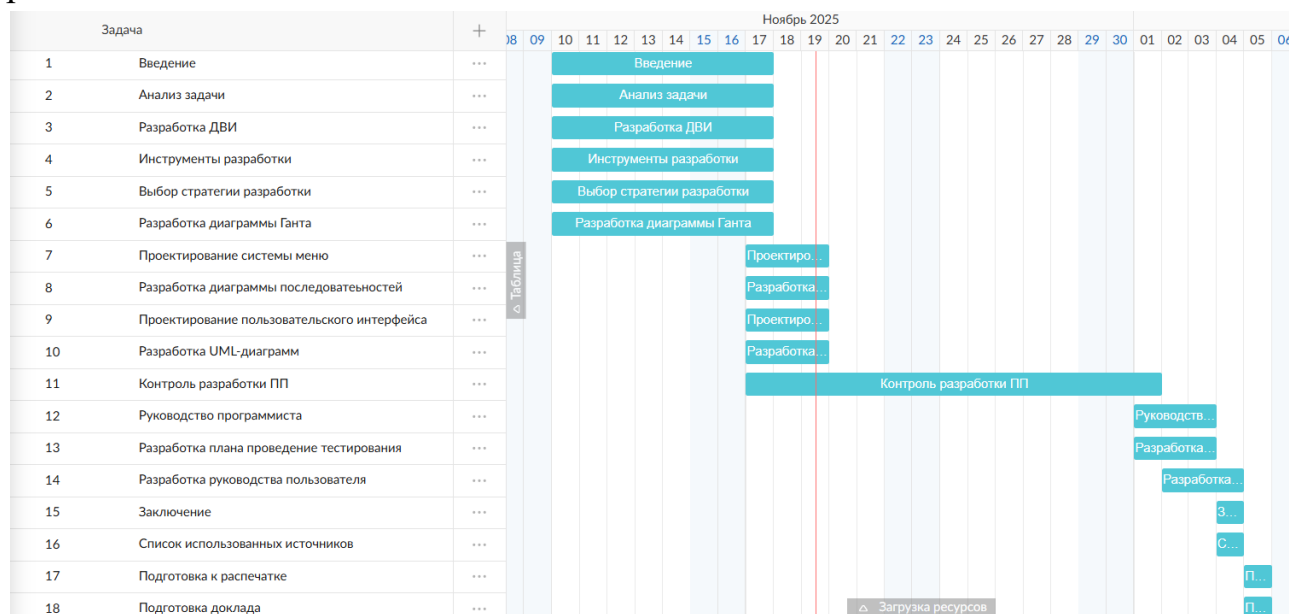


Рисунок 1.1 – диаграмма Ганта

1.5 Выбор стратегии разработки и модели жизненного цикла

Выбор стратегии разработки был осуществлен с помощью таблиц:

Таблица 1.1 – Выбор модели жизненного цикла на основе характеристик требований

№ критерия	Критерии категории требований	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Являются ли требования к проекту легко определяемыми и реализуемыми?	Да	Да	Да	Нет	Нет	Нет
2.	Могут ли требования быть сформулированы в начале ЖЦ?	Да	Да	Да	Да	Нет	Нет
3.	Часто ли будут изменяться требования на протяжении ЖЦ?	Нет	Нет	Нет	Нет	Да	Да
4.	Нужно ли демонстрировать	Нет	Нет	Да	Нет	Да	Да

	требования с целью их определения?						
5.	Требуется ли проверка концепции программного средства или системы?	<u>Нет</u>	<u>Нет</u>	Да	<u>Нет</u>	Да	Да
6.	Будут ли требования изменяться или уточняться с ростом сложности системы (программного средства) в ЖЦ?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да	Да
7.	Нужно ли реализовать основные требования на ранних этапах разработки?	Нет	Нет	<u>Да</u>	<u>Да</u>	<u>Да</u>	<u>Да</u>

Вычисления: 7 за каскадную, 6 за V-образную, 6 за RAD, 3 за инкрементную, 1 за быстрого прототипирования и 1 за эволюционную.

Итог: На основе результатов заполнения табл. 3 подходящей является RAD модель, каскадная и v-образная

Таблица 1.2 – Выбор модели жизненного цикла на основе характеристик команды разработчиков

№ критерия	Критерии категории команды разработчиков проекта	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Являются ли проблемы предметной области проекта новыми для большинства разработчиков?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да
2.	Являются ли инструментальные средства, используемые в проекте, новыми для большинства разработчиков?	Да	Да	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да
3.	Изменяются ли роли участников проекта на протяжении ЖЦ?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да	Да
4.	Является ли структура процесса разработки более значимой для разработчиков, чем гибкость?	Да	Да	<u>Нет</u>	Да	<u>Нет</u>	<u>Нет</u>
5.	Важна ли легкость распределения человеческих ресурсов проекта?	Да	Да	Да	Да	<u>Нет</u>	<u>Нет</u>

6.	Приемлет ли команда разработчиков оценки, проверки, стадии разработки?	<u>Да</u>	<u>Да</u>	Нет	<u>Да</u>	<u>Да</u>	<u>Да</u>
----	--	-----------	-----------	-----	-----------	-----------	-----------

Вычисления: 3 за каскадную, 3 за V-образную, 4 за RAD, 3 за инкрементную, 4 за быстрого прототипирования и 3 за эволюционную.

Итог: На основе результатов заполнения табл. 4 подходящими являются быстрого прототипирования и RAD.

Таблица 1.3 – Выбор модели жизненного цикла на основе характеристик коллектива пользователей

№ критерия	Критерии категории коллектива пользователей	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Будет ли присутствие пользователей ограничено в ЖЦ разработки?	<u>Да</u>	<u>Да</u>	<u>Нет</u>	<u>Да</u>	<u>Нет</u>	<u>Да</u>
2.	Будут ли пользователи оценивать текущее состояние программного продукта (системы) в процессе разработки?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Да</u>	<u>Да</u>	<u>Да</u>
3.	Будут ли пользователи вовлечены во все фазы ЖЦ разработки?	<u>Нет</u>	<u>Нет</u>	<u>Да</u>	<u>Нет</u>	<u>Да</u>	<u>Нет</u>
4.	Будет ли заказчик отслеживать ход выполнения проекта?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Да</u>	<u>Да</u>

Вычисления: 3 за каскадную, 3 за V-образную, 3 за RAD, 2 за инкрементную, 1 за быстрого прототипирования и 1 за эволюционную.

Итог: На основе результатов заполнения табл. 5 подходящей является каскадная и RAD.

Таблица 1.4 – Выбор модели жизненного цикла на основе характеристик типа проектов и рисков

№ критерия	Критерии категории типов проекта и рисков	Каскадная	V-образная	RAD	Инкрементная	Быстрого прототипирования	Эволюционная
1.	Разрабатывается ли в проекте продукт нового для организации	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Да</u>	<u>Да</u>	<u>Да</u>

	направления?						
2.	Будет ли проект являться расширением существующей системы?	Да	Да	Да	Да	<u>Нет</u>	<u>Нет</u>
3.	Будет ли проект крупно- или среднemasштабным?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да	Да
4.	Ожидается ли длительная эксплуатация продукта?	Да	Да	<u>Нет</u>	Да	<u>Нет</u>	Да
5.	Необходим ли высокий уровень надежности продукта проекта?	<u>Нет</u>	Да	<u>Нет</u>	Да	<u>Нет</u>	Да
6.	Предполагается ли эволюция продукта проекта в течение ЖЦ?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да	Да
7.	Велика ли вероятность изменения системы (продукта) на этапе сопровождения?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да	Да
8.	Является ли график сжатым?	Нет	Нет	<u>Да</u>	<u>Да</u>	<u>Да</u>	<u>Да</u>
9.	Предполагается ли повторное использование компонентов?	<u>Нет</u>	<u>Нет</u>	Да	Да	Да	Да
10.	Являются ли достаточными ресурсы (время, деньги, инструменты, персонал)?	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	<u>Нет</u>	Да	Да

Вычисления: 8 за каскадную, 6 за V-образную, 7 за RAD, 2 за инкрементную, 3 за быстрого прототипирования и 2 за эволюционную.

Итог: На основе результатов заполнения табл. 6 подходящей является инкрементная и эволюционная модели.

Общий итог: в итоге заполнения табл. 3 – 6 наиболее подходящей является каскадная модель, но в будущем я выберу RAD-модель по причине её удобства в разработке проекта.

2 Проектирование задачи

2.1 Диаграмма последовательности

Диаграмма последовательности системы «Репетитор по техникам ведения переговоров» иллюстрирует автоматизированный процесс взаимодействия пользователя с Telegram-ботом при получении образовательного контента и прохождении проверки знаний. На диаграмме отражены три ключевых участника: Пользователь, сам Telegram-бот и Файл с данными, выполняющий роль базы знаний.

Процесс взаимодействия инициируется пользователем через выбор интересующего раздела меню. Взаимодействие строится таким образом, что бот выступает активным посредником: при запросе разделов «Изучение переговоров», «Практикум» или «Отработка техник» система обращается к файловому хранилищу, извлекает необходимые списки кейсов, сценариев или теоретических блоков и передает их пользователю.

Отдельный цикл взаимодействия предусмотрен для режима тестирования. Диаграмма демонстрирует, как при выборе «Тест по теории» бот запрашивает пул вопросов, транслирует их пользователю, а затем принимает ответы для проверки. Завершается процесс возвратом итоговых результатов тестирования. Диаграмма наглядно показывает, как система обеспечивает последовательную выдачу материалов и обработку пользовательских реакций, с четким разделением интерфейсной части и логики хранения данных. Диаграмму можно увидеть на рисунке 2.1.

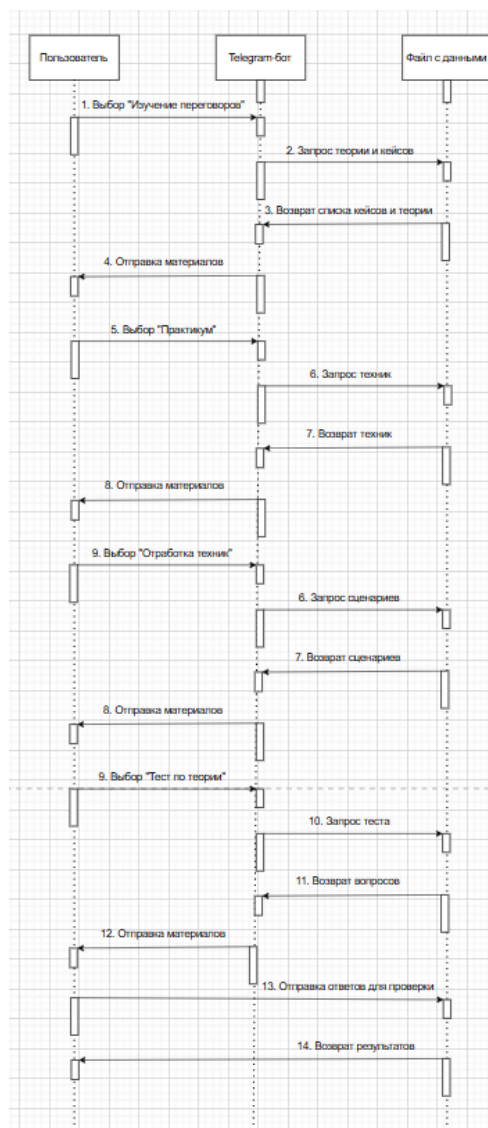


Рисунок 2.1 – диаграмма последовательности

2.2 Проектирование системы меню для чат-бота

Главное меню спроектировано с целью обеспечения максимального удобства пользователя. Меню для пользователей не оплативших и оплативших курс имеет определённые отличия (рисунок 2.2)

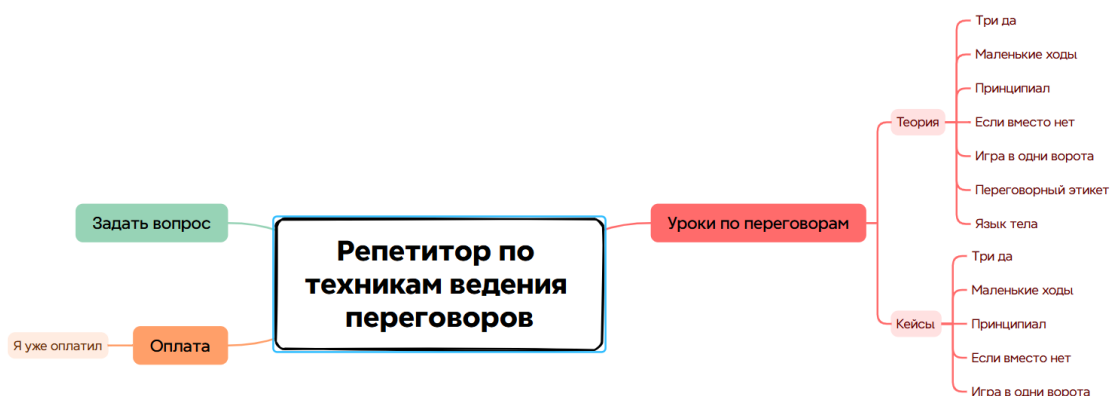


Рисунок 2.2 – структура меню не оплатившего пользователя

С покупкой платного курса пользователю открывается доступ к тестовой части, тренировке по определению техник и своего поведения в разных ситуациях согласно разным техникам (рисунок 2.3).



Рисунок 2.3 – структура меню для оплатившего пользователя

Администратору доступно редактирование теоретической части курса, ответы на вопросы пользователей и подтверждение оплаты.

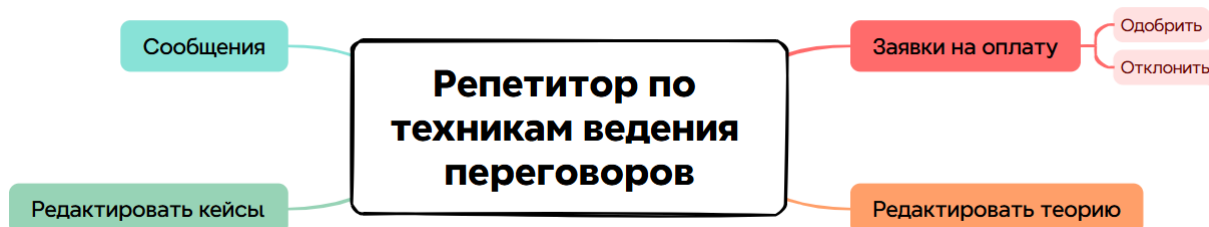


Рисунок 2.4 – структура меню для администратора

2.3 UX-проектирование пользовательского интерфейса

В проекте UX-прототип показывает как должен себя вести чат бот в разных ситуациях. Он предназначен для детальной визуализации архитектуры диалогов и проверки логической связности сценариев взаимодействия до начала программной реализации. Прототип позволяет смоделировать пользовательский путь (User Journey), заранее выявить потенциальные «тупики» в навигации и оптимизировать структуру меню для достижения максимальной эргономичности. Кроме того, он служит эталоном для разработчика, четко регламентируя реакции системы.

Ниже можно наблюдать изображения прототипов:

На рисунке 2.4 показано приветствие бота и сообщение, которое видит пользователь после команды «/start».

Рисунок 2.5 иллюстрирует процесс начала теста.

Разделы с теорией и кейсами представлены на рисунке 2.6.

Разделы с отработкой техник и их определением на рисунке 2.7.

Рисунок 2.8 демонстрирует процесс задания вопроса администратору.

Процесс оплаты показан на рисунке 2.9.



Рисунок 2.4 – приветствие бота



Рисунок 2.5 – начало теста

Изм	Пуст	Недокум.	Подпис	Дат

УП ТРПО 5-04-0612-02.42.06.25 ПЗ

Пуст

16

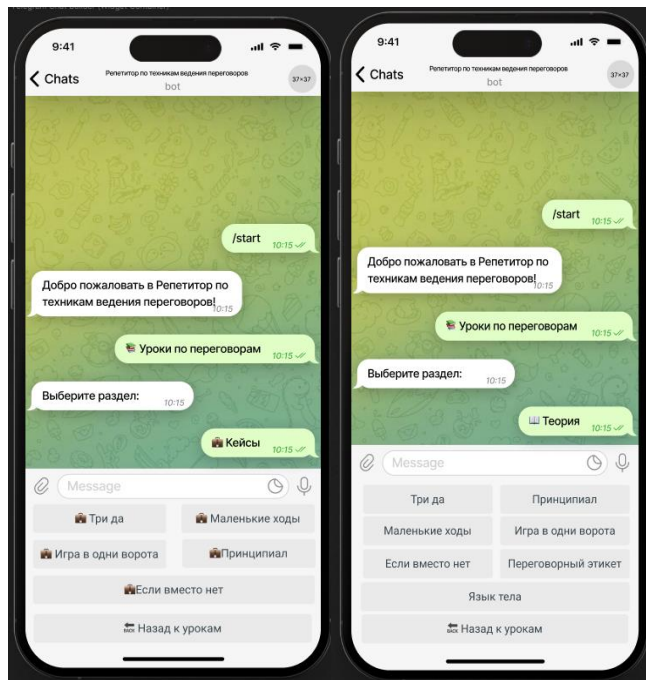


Рисунок 2.6 – теоретические разделы

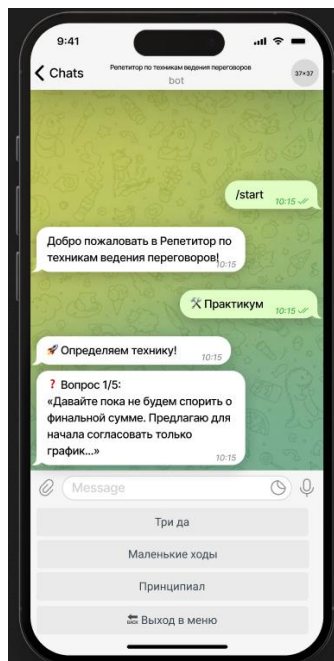


Рисунок 2.7 – определение техник



Рисунок 2.8 – процесс задания вопроса администратору



Рисунок 2.9 – процесс оплаты

3. Реализация

3.1 Руководство программиста

Telegram-бот «Репетитор по техникам ведения переговоров» представляет собой программный комплекс, реализованный на языке Python с использованием библиотеки pyTelegramBotAPI (telebot). Система построена по клиент-серверной архитектуре, где серверная часть выполняет функции обработки запросов, управления данными и бизнес-логикой, а клиентская часть реализована в рамках интерфейса мессенджера Telegram. Для взаимодействия с Telegram API используются основные компоненты библиотеки: класс TeleBot для управления ботом, классы клавиатур для создания интерактивного меню, а также специальные декораторы для обработки команд, текстовых сообщений и callback-запросов.

Для хранения данных используется файловая система на основе формата JSON, что обеспечивает легковесность и переносимость решения без необходимости развертывания сложных СУБД. Система хранения включает три основных файла: users.json, который содержит профили пользователей, включая статус оплаты, имя пользователя и флаг запроса на оплату; feedback.json, хранящий историю обращений пользователей к администратору и метаданные сообщений; и content.json — динамическую базу данных образовательного контента (теоретические блоки и кейсы), доступную для редактирования через административную панель.

Ключевой алгоритм системы — конечный автомат (FSM), реализованный через глобальный словарь состояний. Он отслеживает текущий режим работы пользователя (тестирование, практикум, отработка техник), индекс текущего вопроса и накопленные баллы. Это позволяет системе поддерживать контекст диалога и корректно обрабатывать последовательные шаги в обучающих сценариях. Проект включает два основных модуля: пользовательский и административный. Пользовательский модуль обеспечивает навигацию по урокам, прохождение тестирования с подсчетом баллов и интерактивную отработку переговорных сценариев. Административный модуль предоставляет инструменты для модерации оплат, обработки обратной связи и редактирования контента «на лету» через интерфейс чата.

Запуск бота осуществляется методом, который обеспечивает непрерывный цикл получения обновлений от серверов Telegram, автоматически обрабатывая сетевые исключения и восстанавливая соединение при разрывах.

3.1.1 Структура и описание процедур и функций

Таблица 3.1 – основные функции и процедуры

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		19

Имя функции 1	Модуль 2	Назначение 3
load_json(filename, default)	База данных	Загрузка данных из JSON-файла с обработкой исключений и возвратом значения по умолчанию при отсутствии файла.
save_json(filename, data)	База данных	Сериализация и сохранение структур данных (словарей/списков) в файл формата JSON.
get_main_keyboard(user_id)	Интерфейс	Генерация главного меню с динамическим изменением кнопок в зависимости от статуса оплаты пользователя (платный/бесплатный).
get_lessons_keyboard()	Интерфейс	Формирование меню навигации по разделу «Уроки» (выбор между теорией и кейсами).
get_theory_keyboard()	Интерфейс	Создание списка кнопок с темами теоретических занятий на основе файла контента.
get_cases_keyboard()	Интерфейс	Генерация клавиатуры для выбора практических кейсов из доступного списка.
get_training_menu_keyboard()	Интерфейс	Генерация меню выбора сценариев для режима «Отработка техник».
get_options_keyboard(options)	Интерфейс	Динамическое создание кнопок с вариантами ответов для тестирования и тренажера.
send_question(chat_id, ...)	Тестирование	Универсальная функция отображения вопроса, вариантов ответов и кнопок для режимов тестирования и практикума.
send_welcome(message)	Основная логика	Обработчик команды /start: инициализация нового пользователя в базе данных и вывод приветственного сообщения.
handle_message(message)	Основная логика	Основной маршрутизатор логики: обрабатывает текстовый ввод, управляет состояниями FSM (тесты, диалоги) и навигацией меню.
admin_panel(message)	Администрирование	Обработчик команды /admin: инициализация и отображение панели управления для администратора.
handle_callbacks(call)	Обработка событий	Обработка нажатий инлайн-кнопок: подтверждение оплат, управление сообщениями обратной связи и редактирование контента.

3.1.2 Описание использованных компонентов и библиотек

Для создания проекта использовались следующие компоненты и библиотеки, описанные в таблице 3.2:

Таблица 3.2 – использованные компоненты и библиотек

Компонент/библиотека	Назначение
telebot.TeleBot	Основной класс для инициализации бота и взаимодействия с Telegram Bot API
telebot.types.ReplyKeyboardMarkup	Создание пользовательских клавиатур (меню) в нижней части экрана

telebot.types.InlineKeyboardMarkup	Создание интерактивных кнопок, прикрепленных к сообщениям (для админ-панели)
decorator @bot.message_handler	Обработка входящих текстовых сообщений и команд (фильтрация по типу контента)
decorator @bot.callback_query_handler	Перехват и обработка событий нажатия на инлайн-кнопки (Callback Query)
json	Встроенный модуль для сериализации и десериализации данных (чтение/запись JSON-файлов)
os.path	Работа с файловой системой (проверка наличия файлов базы данных перед загрузкой)
random	Генерация случайных чисел и перемешивание вариантов ответов в тестах
time	Работа со временем (генерация уникальных ID сообщений, временные метки)

3.2 Спецификация проекта

Проект telegram-бота “Репетитор по техникам ведения переговоров” имеет следующую файловую структуру, представленную в таблице 3.3.

Таблица 3.3 – спецификация программы

Имя файла компонента	Назначение и содержание
bot.py	Основной исполняемый файл, содержащий код инициализации и логику бота
users.json	Файл базы данных пользователей (ID, оплата, имя)
content.json	Файл с текстами лекций и медиа-контентом кейсов
feedback.json	Файл для хранения сообщений обратной связи
requirements.txt	Список зависимостей для развертывания (pyTelegramBotAPI)

4 Тестирование

В период с 28 по 30 ноября происходил процесс тестирования Telegram-бота «Репетитор по техникам ведения переговоров». Успешно было пройдено 100% тест-кейсов крайне высокого и высокого приоритета, 100% тест-кейсов среднего приоритета и 100% тест-кейсов низкого приоритета.

Команда тестировщиков описана в таблице 4.1.

Таблица 4.1 – Команда тестировщиков

ФИО	Должность	Роль
Колб Роман Иосифович	Разработчик, тестировщик	Ответственный за тестирование чат-бота

Тестирование производилось под ОС Windows 11 x64. Все тест-кейсы выполнялись вручную. Тестирование показало среднюю стабильность функциональности на начальном этапе (обнаружено 5 дефектов разной важности). Повторное тестирование показало прирост качества (исправлено 100% обнаруженных ранее дефектов). Расписание проведения тестирования представлено в таблице 4.2.

Таблица 4.2 – Расписание проведения тестирования

ФИО	Дата	Длительность	Продолжительность
Колб Роман Иосифович	28.11.2025	Разработка тест-кейсов	3
Колб Роман Иосифович	29.11.2025	Тестирование приложения	4
Колб Роман Иосифович	29.11.2025	Анализ выполнения тест-кейсов	1
Колб Роман Иосифович	30.11.2025	Проведение регрессивного тестирования	3
Колб Роман Иосифович	30.11.2025	Составление отчёта о результатах тестирования	2

Разработанные тест-кейсы и результаты тестирования представлены в приложении В. Статистика по разработанным тест-кейсам представлена в таблице 4.3.

Таблица 4.3 – Статистика по разработанным тест-кейсам

Статус	Количество тест-кейсов	Приоритет				
		Крайне низкий	Низкий	Средний	Высокий	Крайне высокий
Пройдено	14	0	3	6	4	1
Не пройдено	0	0	0	0	0	0
Заблокировано	0	0	0	0	0	0
Не протестировано	0	0	0	0	0	0

В таблице 4.4 представлена статистика по выявленным и исправленным ошибкам.

Таблица 4.4 – Статистика по выявленным ошибкам

Статус	Количество ошибок	Важность			
		Низкая	Средняя	Высокая	Критическая
Найдено	5	1	2	1	1
Исправлено	5	1	2	1	1
Проверено	5	1	2	1	1
Открыто заново	0	0	0	0	0
Отклонено	0	0	0	0	0

Список найденных дефектов предоставлен в таблице 4.5.

Таблица 4.5 – Список найденных дефектов

Идентификатор	Важность	Описание	Статус
D_01	Критическая	При вводе команды /admin пользователем без прав администратора бот не выдавал сообщение об отказе.	Исправлено
D_02	Высокая	В файле users.json статус оплаты (paid) не обновлялся мгновенно после подтверждения заявки.	Исправлено
D_03	Средняя	В режиме «Отработка техник» при нажатии кнопки «Назад» не сбрасывалось состояние (заикливание).	Исправлено
D_04	Средняя	Некорректное отображение жирного шрифта в описании кейса №2.	Исправлено
D_05	Низкая	Опечатка в приветственном сообщении при регистрации нового пользователя.	Исправлено

5 Руководство пользователя

5.1 Общие сведения о программном продукте

Telegram-бот «Репетитор по техникам ведения переговоров» представляет собой программную систему для автоматизированного обучения и тренировки коммуникативных навыков. Назначение программы — повышение квалификации пользователей в области деловых и межличностных переговоров посредством прохождения теоретических тестов, решения практических кейсов и отработки интерактивных сценариев.

Основные возможности включают: доступ к библиотеке теоретических материалов и разбору реальных кейсов, прохождение тестирования для проверки знаний, интерактивную отработку техник переговоров («Три да», «Маленькие ходы», «Принципиал» и др.) в режиме диалога, а также систему обратной связи с администратором. Программа функционирует в среде мессенджера Telegram и не предъявляет специфических требований к аппаратному обеспечению пользователя — достаточно устройства с установленным приложением Telegram и доступом в интернет. Использование системы носит образовательный характер: обучение выполняется в удобном для пользователя темпе.

5.2 Руководство системного программиста

Установка программного продукта требует предварительной подготовки: наличия интерпретатора языка программирования Python версии 3.8+, аккаунта в Telegram и созданного бота через @BotFather (для получения API-токена).

Процесс инсталляции включает: установку необходимых зависимостей через менеджер пакетов командой `pip install pyTelegramBotAPI`, размещение исполняемого файла `main.py` в рабочей директории и настройку токена доступа в коде программы (переменная `TOKEN`). При первом запуске система автоматически инициализирует необходимые файлы базы данных (`users.json`, `content.json`, `feedback.json`).

Запуск осуществляется командой `python main.py`. При успешной установке система выводит в консоль сообщение: «Бот запущен...». Возможные сообщения об ошибках включают проблемы с подключением к серверам Telegram (`NetworkError`) или ошибки аутентификации (неверный токен). После установки необходимо проверить работоспособность, отправив боту команду `/start` и убедившись в корректном отображении главного меню.

5.3 Руководство пользователя

5.3.1 Запуск программы

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		24

Запуск программы осуществляется через мессенджер Telegram тремя способами: через глобальный поиск по username бота, по прямой ссылке формата https://t.me/peregovorrep_bot. После открытия диалога с ботом необходимо нажать кнопку «Запустить» или «Start» в нижней части экрана для начала работы и инициализации главного меню.

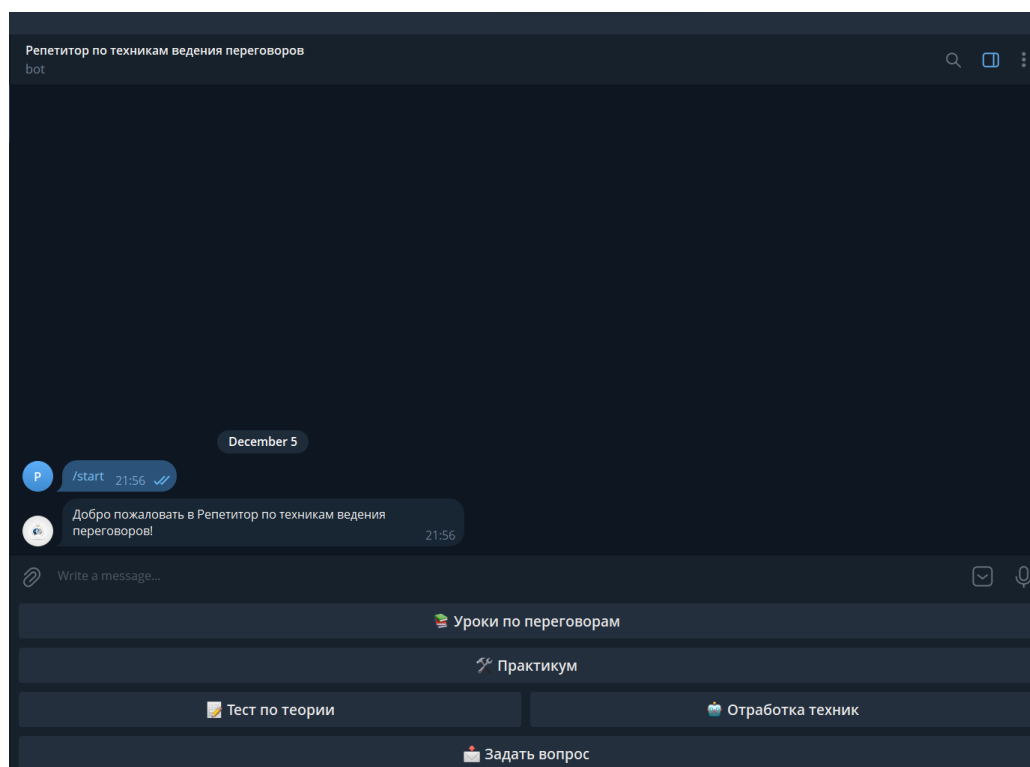


Рисунок 5.1 – Запуск бота

5.3.2 Инструкции по работе с программой

5.3.2.1 Инструкции по работе с программой для пользователя

Управление программой осуществляется через текстовые команды и интерактивные кнопки.

Основные команды и кнопки:

- /start — инициализация бота, регистрация пользователя в системе и вызов главного меню.
- /admin — вызов панели администратора (доступно только пользователям с соответствующими правами).
- Уроки по переговорам — переход к библиотеке теоретических материалов и кейсов.
- Оплата — инициация процесса оплаты для получения полного доступа к функционалу.
- Тест по теории — запуск режима проверки знаний (доступно после оплаты).

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
Изм	Лист	Недокум.	Подпис	Дат		25

- Отработка техник — переход к выбору интерактивных сценариев диалога (доступно после оплаты).
- Практикум — режим решения практических задач (доступно после оплаты).
- Задать вопрос — форма обратной связи для отправки сообщения администратору.

Процесс обучения и тестирования состоит из нескольких этапов. На рисунке 5.1 можно увидеть, что работа начинается с выбора режима в главном меню. При выборе кнопки «Тест по теории» система автоматически формирует пул из 10 вопросов, представленных на рисунке 5.2. Вопросы подаются последовательно, выбор варианта ответа осуществляется нажатием на соответствующую кнопку. После завершения теста система вычисляет итоговый балл и выводит результат, как показано на рисунке 5.3.

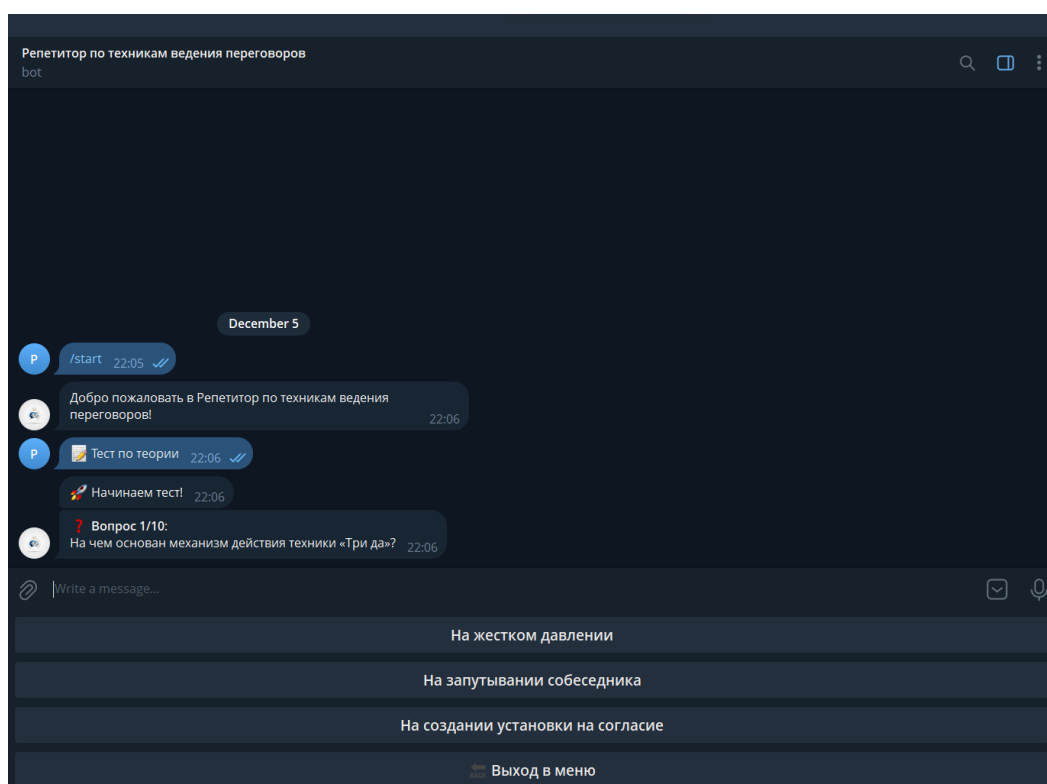


Рисунок 5.2 – Начало теста

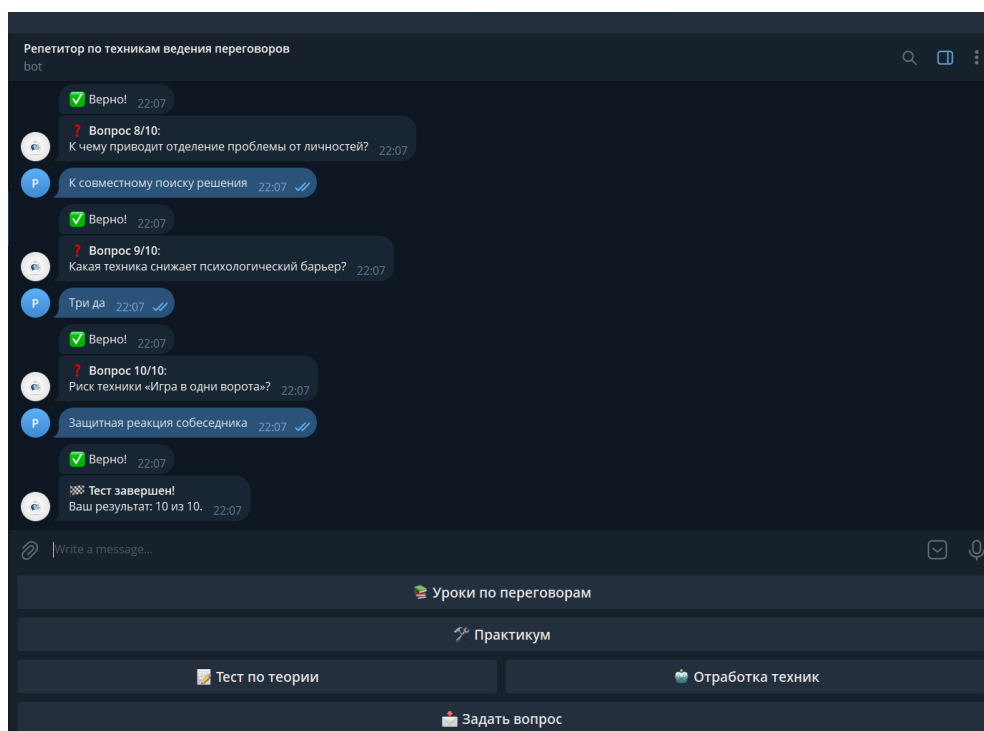


Рисунок 5.3 – Окончание теста

Для отработки практических навыков используется режим «Отработка техник». На рисунке 5.4 отображено меню выбора конкретной техники («Три да», «Маленькие ходы» и др.). В процессе диалога, представленном на рисунке 5.5, бот моделирует реальные ситуации: пользователь выбирает реплику, а система дает мгновенную обратную связь (Feedback), поясняя, была ли выбранная стратегия верной.

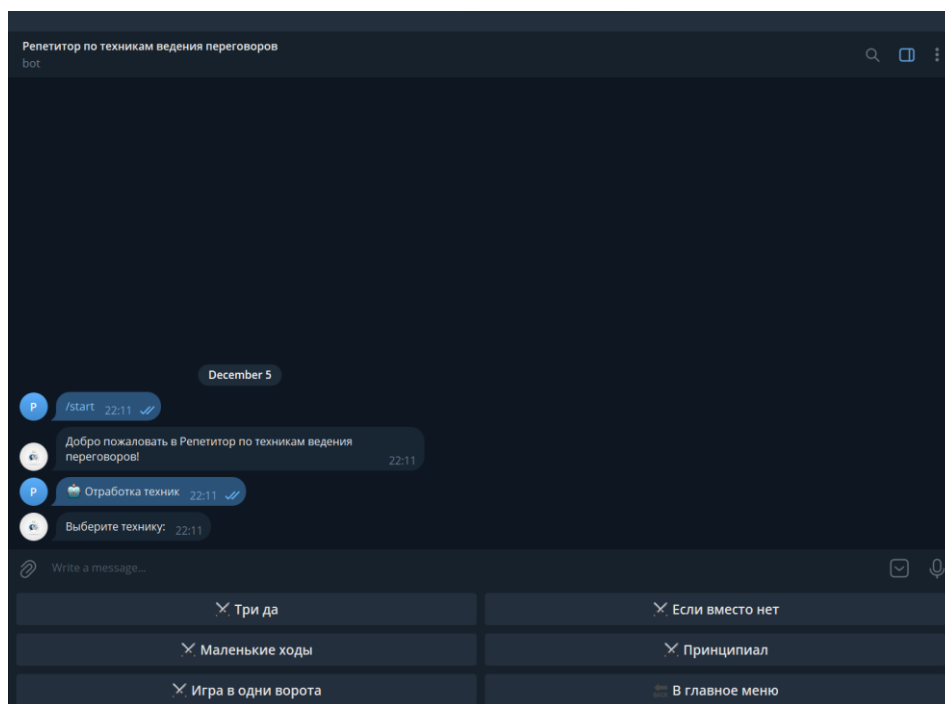


Рисунок 5.4 – Меню выбора техник

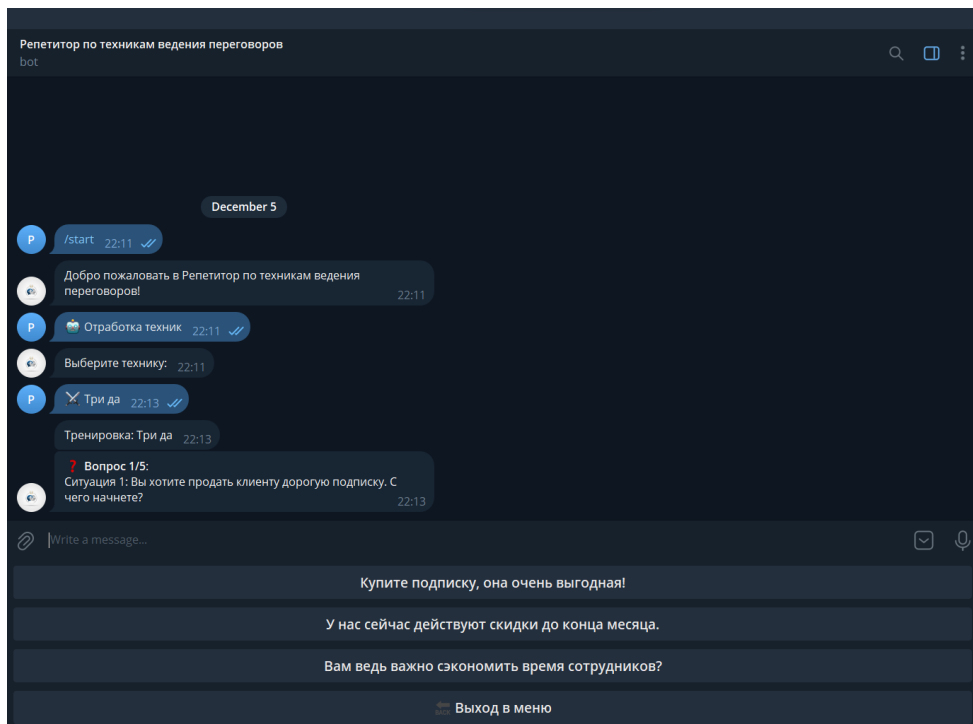


Рисунок 5.5 – Моделирование ситуации

В случае возникновения вопросов пользователь может воспользоваться функцией обратной связи, показанной на рисунке 5.6.

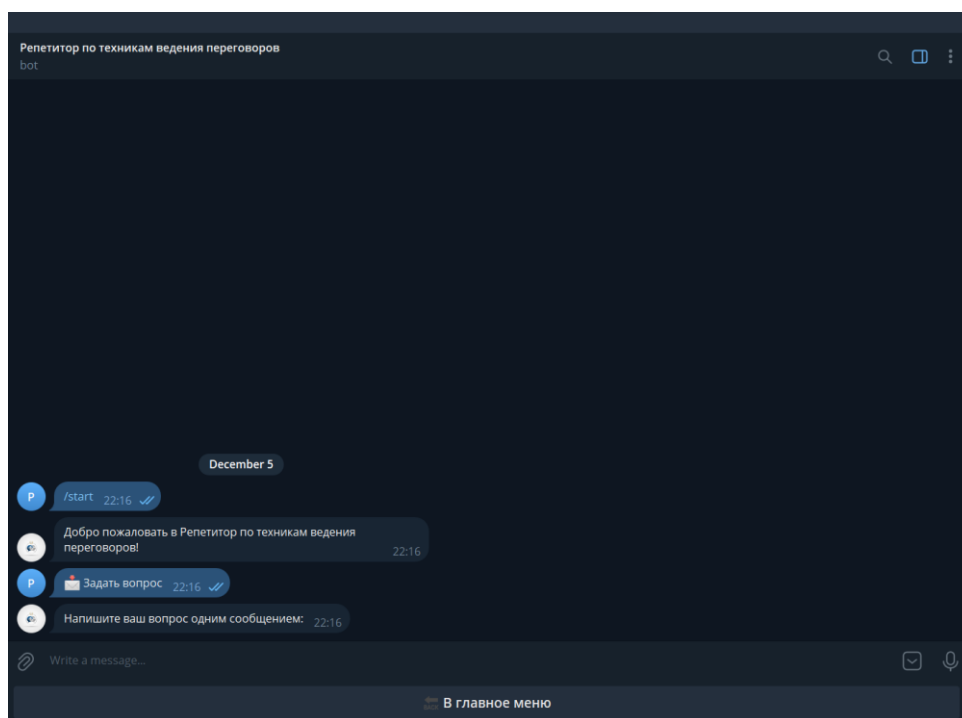


Рисунок 5.6 – Кнопка «Задать вопрос»

5.3.3 Завершение работы с программой

Программа не требует явного завершения – пользователь может закрыть чат, а все данные будут сохранены.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
						29
Изм	Лист	№докум.	Подпис	Дат		

Заключение

В ходе выполнения работы была достигнута поставленная цель — разработан и реализован программный продукт в виде чат-бота для мессенджера Telegram, предназначенный для обучения пользователей навыкам ведения переговоров.

В результате проектирования и программирования были решены следующие задачи:

Реализована архитектура приложения на языке Python с использованием библиотеки pyTelegramBotAPI, что обеспечило стабильное взаимодействие с серверами Telegram.

Разработана структура данных, позволяющая хранить теоретические материалы, кейсы и разветвленные сценарии диалогов.

Внедрена система разграничения прав доступа, включающая бесплатный ознакомительный режим и платный функционал с ручной верификацией платежей администратором.

Создан интерактивный тренажер, позволяющий пользователям отрабатывать переговорные техники в режиме симуляции реального диалога с мгновенной обратной связью.

Тестирование показало, что бот корректно обрабатывает запросы пользователей, сохраняет прогресс прохождения тестов и устойчив к ошибкам ввода. Использование JSON-файла для хранения данных обеспечило простоту развертывания системы без необходимости использования СУБД.

Практическая значимость работы заключается в создании доступного инструмента обучения, который может быть использован для самообразования.

В качестве перспективных направлений развития и улучшения функционала чат-бота можно выделить: масштабирование системы хранения данных: Переход от файловой структуры (JSON) к использованию полноценной реляционной системы управления базами данных (например, PostgreSQL или MySQL). Это обеспечит более высокую надежность, целостность данных и скорость обработки запросов при увеличении числа активных пользователей. Также можно выделить автоматизацию приема платежей в виде интеграции с платежными шлюзами (YooKassa или Telegram Payments) для исключения ручной проверки чеков администратором и предоставления мгновенного доступа к материалам сразу после оплаты.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
						30
Изм	Лист	Недокум.	Подпис	Дат		

Список использованных источников

1. Документация библиотеки pyTelegramBotAPI [Электронный ресурс]. — Режим доступа: <https://pypi.org/project/pyTelegramBotAPI/>.
2. Фишер, Р. Переговоры без поражения. Гарвардский метод / Р. Фишер, У. Юри, Б. Паттон. — М.: Манн, Иванов и Фербер, 2021. — 272 с.
3. Шебанов, А. Разработка чат-ботов на Python. Полное руководство / А. Шебанов. — М.: БХВ-Петербург, 2023. — 320 с.
4. Чалдини, Р. Психология влияния. Убеждай, воздействуй, защищайся / Р. Чалдини. — СПб.: Питер, 2022. — 480 с.

					УП ТРПО 5-04-0612-02.42.06.25 ПЗ	Лист
						31
Изм	Лист	Недокум.	Подпис	Дат		