МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра Прикладная математика и информатика
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Бизнес-информатика
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему Разработка чат-бота, поддерживающего анализ и выражение эмоций

Студент С.В. Шульпин
(И.О. Фамилия) (личная подпись)

Руководитель А.П. Тонких
(ученая степень, звание, И.О. Фамилия)

Тольятти 2020

Аннотация

Тема бакалаврской работы: «Разработка чат-бота, поддерживающего анализ и выражение эмоций».

В данной бакалаврской работе исследуются практические аспекты реализации чат-бота, способного анализировать и выражать эмоции.

В бакалаврской работе проводится исследование видов, типов эмоций, разбираются способы их определения, составляется теоретический проект чат-бота с выбором модели среди различных вариантов, рассчитывается экономическое обоснование создания чат-бота. В конечном итоге производится практическая реализация данного чат-бота с указанием исходного кода в приложении к данной работе.

Структура бакалаврской работы представлена введением, тремя главами, заключением, списком литературы, приложением.

Во введении описывается актуальность проводимого исследования, дается характеристика разрабатываемому чат-боту.

В первой главе производится анализ и определяются особенности выражения эмоций, психологические аспекты эмоций, рассматриваются техники автоматизации определения эмоций. Во второй главе производится процесс проектирования. Выбираются технологии и сервис для реализации чат-бота, рассматриваются концептуальные аспекты функционирования системы, описывается логическая модель работы проектируемого чат-бота. В третьей главе происходит процесс реализации чат-бота.

В работе использовано 8 таблиц, 40 рисунков, список литературы содержит 62 литературных источника. Общий объем выпускной квалификационной работы составляет 107 страниц.

Оглавление

Введение

На современном этапе развития цивилизации глобальная сеть Интернет представляет собой многофункциональную сеть для коммуникаций, развлечений, обучения и работы. Коммуникации через Интернет – неотъемлемая часть жизни каждого человека. Сейчас в обществе имеется немало способов, форм и средств коммуникации, при этом большая часть из них взаимосвязана с новыми средствами, которые основаны на интернет-технологиях.

Компьютерная сеть Интернет, помимо обширного источника различной информации, является главным инструментом для виртуального общения. Поддержание связи с родственниками и просто близкими людьми, контакты с рабочими партнерами, новые знакомства – все это важная и неотъемлемая часть повседневной жизни каждого человека, к тому же подбор более практичных методов интернет-общения стал довольно разнообразным [17].

Интернет и компьютерные сети играют важную роль в процессах развития бизнеса, важной частью которого является обратная связь компаний и потребителей. Сетевые технологии обеспечивают каналы взаимосвязи между участниками процессов. В связи с этим очень важна обработка данных, которые поступают от пользователей. Процесс обработки данных является трудоемким и затратным по времени. Усовершенствовать его можно за счет автоматизации обработки текстовых массивов данных. Потребность в понимании эмоционального оттенка и смыслов текстов приводит к увеличению заинтересованности владельцев информационных систем к автоматизации выделения эмоций из текста, что связано с большей необходимостью в способах выявления реакций на действия пользователей. Процессы автоматизации обработки и понимания текстов интересуют преимущественно корпоративный и государственный сектор, которым важно адекватно откликаться на реакции пользователей системы на акции, события, реформы, законы.

Для того чтобы обеспечить выявление эмоциональной окраски текста, необходимо провести сентимент-анализ (анализ тональности текста). Сентимент-анализ, или анализ тональности текста, – это развивающееся направление компьютерной лингвистики, основной задачей которого является выявление в документе эмоционально окрашенной лексики и эмоциональной оценки объектов автором. Такой анализ является одним из основных направлений компьютерной лингвистики. Важно отметить также, что компьютерная лингвистика сейчас находится на пике своего развития. Главная задача компьютерной лингвистики заключается в том, чтобы определить в тексте фрагменты, характеризующиеся эмоциональным окрасом слов и словосочетаний, а также оценкой изучаемых объектов или предметов. Системы анализа текстов и эмоций также применяются в приложениях взаимодействия между людьми. Самым простым примером реализации системы в рамках информационных технологий является база данных, содержащая в себе вопросы и ответы. В первую очередь требуется решить проблему по определению содержания базы данных, в том числе реализовать на практике «программы-интерпретаторы». Содержание представляет собой комплекс вопросов и допустимых к ним вариантов ответов. Также база данных содержит в себе историю бесед перед вопросами и название соответствующей темы общения. Это является описанием программы, которую называют чат-ботами. Чат-бот – это компьютерная программа, которая ведет диалог с поддержкой разных способов ввода-вывода. Чат-боты, или виртуальные собеседники, чаще всего применяются для различных практических задач, в том числе для предоставления услуг пользователям или сбора информации. Отдельные чат-боты применяют непростые алгоритмы обрабатывания разговорной речи, однако, в то же время имеется множество элементарных алгоритмов распознавания слов-ключей на входе и выбора ответа. Этот выбор больше всего схож с шаблоном формулировки из базы данных. Чат-боты способны решать

множество разных задач одновременно – от общения до рекомендаций по бизнесу, медицинских и юридических консультаций, онлайн-заказов товаров и услуг [36].

Таким образом, актуальность исследования проблематики анализа эмоционального направления массивов данных обусловливается тем, что онлайн-общение становится все более популярным, а также главным способом поддержания связи между пользователями в современных условиях развития информационных технологий. Это отражается в распространении программ обмена сообщениями, среди которых специализированные решения узкой направленности, а также различные социальные сети. В связи с этим целью работы является разработка чат-бота, поддерживающего анализ и выражение эмоций. Для достижения цели необходимо решить следующие задачи:

– исследовать особенности анализа и выражения эмоций;

– изучить состояние вопроса анализа и выражения эмоций в современных программных системах;

– определить методы и способы анализа и выражения эмоций;

– изучить способы определения эмоционального направления текстового содержимого;

– провести логическое проектирование чат-бота, выбрать технологии реализации проекта;

– реализовать проект чат-бота, поддерживающего анализ и выражение эмоций.

Объектом исследования выступают эмоции людей, которые можно распознать в тексте. Предмет исследования – чат-бот, поддерживающий анализ и выражение эмоций.

Структура работы обусловлена поставленной целью и задачами. Бакалаврская работа содержит введение, три главы, заключение, список использованной литературы, включающий 62 источника, заключение, приложение.

В первой главе производится анализ и определяются особенности выражения эмоций, психологические аспекты эмоций, рассматриваются техники автоматизации определения эмоций. Также определяются методы и способы анализа выражения эмоций, изучаются особенности определения эмоционального направления текстового содержимого.

Во второй главе производится процесс проектирования. Выбираются технологии и сервис для реализации чат-бота, рассматриваются концептуальные аспекты функционирования системы, описывается логическая модель работы проектируемого чат-бота.

В третьей главе происходит процесс реализации чат-бота, который сопровождается физическим моделированием чат-бота. Также разрабатывается руководство пользователя и производится расчет затрат на выполнение проекта с учетом затрат на заработную плату исполнителям, затрат на закупку или аренду оборудования, затрат на организацию рабочих мест и затрат на накладные расходы.

Глава 1 Исследование особенностей анализа и выражения эмоций

1.1 Анализ и выражение эмоций в современных программных системах

1.1.1 Психологические аспекты эмоций

Психологии эмоций посвящено огромное количество исследований, в этом пункте рассмотрим некоторые из них. Так, первое исследование, которое обращает на себя внимание – труд К. Отли, в котором высказывается следующая гипотеза: чувства возникают в связи с тем, что в человеке происходят когнитивные процессы; любое чувство содержит в себе, в некотором варианте, определенную задачу; с целью свершения данной задачи формируются определенные планы; исполнение планов находится в зависимости от существующих в наличии возможностей, а также от окружающей среды: чувства предполагают определенный уровень общения с персонами вокруг человека. В соответствии с данной концепцией, «счастливые», позитивные чувства появляются, если «планы согласованы между собой, а также внезапные преграды устранены» [49]. Негативные чувства появляются, если отсутствует координирование системы действий (плана), либо не получается реализовать задуманные действия, либо если появляется задача, на решение которой недостаточно существующих ресурсов.

Исследование чувств как позиций, несомненно имеющих в собственной базе некую задачу, является разумным для этих чувств, так же, как и сожаление либо удовольствие. На самом деле, данные чувства появляются в роли следствия, неудачи либо осуществления задачи [1]. Непосредственно, разочарованность имеет возможность появляться, если человек никак не может достигнуть поставленной задачи либо другими способами обманывать собственные ожидания, но ощущение удовлетворенности возникает при противоположных действиях. Однако к большинству эмоций анализ неприменим. В то время как

эмоции типа разочарование действительно имеют в своей основе цель, эмоции типа грусть могут быть мотивированы и многими другими факторами, причем невозможность достичь своей цели не является даже наиболее типичным из них. Более того, грусть может часто возникать и немотивированно.

Для возникновения немотивированных эмоций Отли предлагает следующее объяснение: они возникают в результате того, что человек, испытывающий ту или иную эмоцию, находится в определенном настроении, предрасполагающем к ее возникновению, например, человек, которому грустно, находится в депрессии. Представляется, что, в целом, описание эмоций через цели слишком ограниченно и дает хорошие результаты только для небольшого количества эмоций. Если приложить схему [1] к таким явно не предполагающим никакой цели эмоциям, как страх, грусть, отвращение и т.д., описания станут неестественными и противоречащими интуиции.

Анализ таких эмоций, как счастье и грусть, хотя и далеко не исчерпывающий, можно, в целом, признать приемлемым. В самом деле, некоторые ситуации, в которых эти эмоции возникают, подпадают под определение Отли. Однако описание эмоций страх, гнев и отвращение представляется абсолютно неадекватным. Гнев, как правило, возникает в результате того, что объект эмоции делает что-то, что субъекту кажется очень плохим. Поступок объекта может разрушить планы субъекта, но это вовсе не является обязательным [1].

Анализ такой эмоции, как страх, возникающей, когда «цель самосохранения» находится под угрозой, предполагает постоянное наличие у человека такой цели. Однако, когда человеку ничто не угрожает, это означает, что «цель самосохранения» достигнута и, следовательно, не существует. Авторы не разделяют чувства на позитивные и негативные, а разделяют их исходя из отличий в когнитивном строении, собственно, они разделяются на виды в связи с тем, что пребывает в центре внимания: событие, категория или объект [1].

Представленный подход является крайне действенным для отображения наиболее разных видов чувств. Событие и объект представлены составляющей долей по факту любой человеческой эмоции, и подобным образом, данные характеристики (в отличии от вида задачи) имеют все шансы быть наиболее универсальными при применении. Другое достоинства этого подхода состоят в том, что описываются не отдельные эмоции, а типы эмоций.

Типом эмоции называется группа «однородных» эмоций, т.е. таких, которые возникают в более или менее одинаковых ситуациях. Описание типов эмоций более привлекательно, чем описание отдельных эмоций, в том смысле, что первое предоставляет больше возможностей для создания общей картины широкого круга эмоций. Кроме того, такой подход к описанию может быть более эффективен при сравнении нескольких разных культур (языков), поскольку он не привязан к какому-то одному конкретному языку. Рассматривая отдельные эмоции, входящие в этот тип в каждой из культур, можно понять, каким образом соответствующая эмоция в каждой из этих культур концептуализируется.

Необходимо, однако, отметить, что достоинства, изложенного выше подхода, в некоторых случаях оборачиваются недостатками. Во-первых, универсальность описания оборачивается иногда недостаточностью различительной силы. Так, не всегда возможно, пользуясь только теми параметрами, что предложены исследователем, описать разницу между такими типами эмоций, как горе и грусть Оба типа описываются как вызванные нежелательным событием, а разница между ними – как сводящаяся к различной степени нежелательности (более высокой в случае горя). Однако в действительности различие является гораздо более глубоким, а именно, горе предполагает какую-то серьезную потерю, обычно потерю близкого человека, в то время как грусть может быть вызвана почти любым нежелательным событием. Еще более наглядным примером недостаточности этого подхода может служить сравнение эмоций типа «страх» и ментальных состояний типа «подозрение». И те,

и другие описываются как «недовольство при мысли о возможности неприятного события». Различает эти два типа психологических состояний эмоциональный компонент, присутствующий в «страхе» и отсутствующий в «подозрении».

Во-вторых, при всей универсальности этого подхода такие эмоции как меланхолия и тоска не связанные ни с каким событием, агенсом или объектом, не находят в нем места. Практически все немотивированные эмоции не подходят под предложенное в этой книге описание, так как оно предполагает, что все эмоции мотивированы.

Отдельно стоит отметить форму объяснения названий чувств А. Вежбицкой. Профессор создала модель объяснения наименований эмоций в разных языках в основе которых лежат универсальные семантические примитивы, то есть определенные, инстинктивно понятные. Согласно суждению А. Вежбицкой, «врожденные, а также многоцелевые определения обязаны находиться в описании множества языков планеты [4].

Представленные в ее трудах объяснения предполагают собой своего рода исходные модели действий либо сценарии, определяющие порядок мыслей, желаний и эмоций. Тем не менее данные модификации действий можно анализировать как формулы, предполагающие жесткое разделение требуемых, а также необходимых обстоятельств (никак не с целью чувств, однако с целью психологических определений), а также данные формулы не теряют из вида размытости пределов среди суждений. А. Вежбицкая [4] систематизирует наименования чувств следующим способом:

– эмоции, которые связаны с неблагоприятными «плохими вещами»;

– эмоции, которые связаны с «хорошими вещами»;

– эмоции, которые связаны с людьми, которые осуществили малоприятные поступки, и провоцируют отрицательный контекст;

– эмоции, которые связаны с размышлениями о собственной личности и самооценке;

– эмоции, которые связаны с отношениями людей друг с другом.

Подобным образом находились варианты подходов определения чувств, установлен наиболее результативный аспект для определения большего спектра чувств, а собственно – таких, которые обладают хорошей мотивацией. Проанализированный подход имеет ценность только в рамках межъязыкового сопоставления, которое может быть обнаружено в любом слоге.

1.1.2 Автоматическое определение эмоций

Методы автоматизированного определения чувств по голосу и речи человека нужны для формирования удачного интерактивного диалога. В современных условиях формируются компьютерные проекты, разрешающие определять чувства по речи и тексту человека. Учеными подтверждено присутствие глубокой взаимосвязи между эмоциями человека и отличительными чертами его речи. Системы, различающие эмоциональное состояние человека, применимы в интерактивном телевидении, онлайн-обучении, при изучении патологий функций мозга, также могут стать полезны людям, обладающим какими-либо речевыми отклонениями. Восприятие чувств иного человека важно, как для общения между людьми, так и при взаимодействии человека с системами искусственного разума [16].

1.2 Методы и способы анализа и выражения эмоций

1.2.1 Выделение оценочных суждений

Данный параграф посвящен изучению понятия «оценочное суждение» и его составляющим. Поскольку оценочное мнение состоит из таких компонентов, как объект мнения или высказывания, субъект и валентность (тон, либо отрицательные эмоции, либо положительные эмоции), то его принято назвать «тройкой». Первый компонент – объект, представляется лицом либо индивидом, к которому обращена эмоциональная фраза. Второй компонент – это субъект. Как

правило, это автор данной фразы, выражающий свое отношение к объекту в прямой либо в косвенной форме. И наконец, третий компонент – валентность, представляет собой уровень эмоционального отношения автора непосредственно к конкретному предмету [30].

С целью детального рассмотрения составляющих «тройки», отметим следующие аспекты оцениваемых предложений. Чтобы отделить текст, имеющий в составе предложения, содержащие описание данных, от остального текста чаще всего применяют систематизированные методы и средства, среди которых следующие: графовый метод, байесовский классификатор или методика разбора различных сочетаний слов. При этом похожесть и родственность каких-либо конкретных сочетаний слов обуславливаются словарями одного типа. Важно отметить, что определяющим показателем в данном случае может выступать присутствие либо отсутствие текстов эмотивного характера или же какой-либо части речи [17].

К широко используемой методике, выделяющей слова с высокоэмоциональным и чувственным окрасом, можно отнести анализ словосочетаний [28]. Для того, чтобы применить эту методику, требуется синтаксическая разметка текста. В итоге, учитывая общепринятые стандарты будет происходить закрепление данных словосочетаний, которые, как правило, будут оценивающими. Вместе с тем прилагательное либо наречие будут отвечать за субъектность, а прочие части данного сочетания слов будут способствовать определению характера оценки, которая, в свою очередь, определяет субъект это или объект. Непосредственно в результате этих действий начинает формироваться эмоциональная валентность, всегда учитывающая родственный контекст прилагательного или наречия, относительно таких понятий, как хорошо или плохо. На этапе завершения необходимо представить уже отношения, которые были получены ими и, в то же время, которые совместимы между собой [14]. Помимо обычных слов, демонстрирующих чувства, бывают сочетания слов,

содержащие в себе оценку с психологической позиции. Для определения этих состояний требуется провести подробный анализ всех сочетаний слов (биграммы, триграммы), происходит оценивание их «точности». Точность «n-словесной цепи» (n – количество слов) – численность индивидуальных параметров цепи, поделенное на единое количество применения существующей цепи. Применение «n-словесной» цепи предполагает индивидуальность в той ситуации, в которой каждое слово существующей фразы располагается в субъективном фрагменте. Можно осуществлять анализ любого сочетания слов отдельно, прибегая к обычной методике систематизации, кроме этого, можно обратиться и к более ограниченному набору слов для того, чтобы повысить слаженное взаимодействие и сосуществование компонентов «n-словесной цепи». Также необходимо действовать, определяя валентность: существует высказывание, что две похожих фразы или два словосочетания имеют одинаковый уровень валентности, однако не определено какое значение он имеет: положительное или отрицательное [17, 32].

1.2.2 Определение тональности

Как правило, исследовательская деятельность, имеющая отношение к психологической окраске текста, осуществляется с использованием методик автоматизированного образования, в частности, прибегая к стандартному байесовскому классификатору, установке опорных векторов и EM-алгоритму. Биграммы могут играть главную роль базовых признаков. При байесовской вероятности есть возможность использовать скрытое распространение Дирихле (LDA), в соответствии с которым охватывается небольшое количество тематик, где может произойти прохождение слов из словаря. Зарождение слова обуславливается нулевой вероятностью, учитывая тематику. Для каждого отдельного документа свойственен свой тематический вектор вероятности. Тем не менее, составление слова в нем может осуществляться исключительно с одной тематикой [27].

Грамматические классы, к которым относятся части речи, структурные компоненты, знаки препинания и пр., также можно использовать.

В наибольшей степени тексты, отзывы, обсуждения и статьи на просторах интернета преисполнены ненужной информацией. Чаще всего структуры для реферирования используют накопленную информацию в данной области или же анализируют лингвистически новые данные. Данные операции можно отнести к максимально трудоемким и времязатратным процедурам. Свойства правильно составленного предложения выглядят следующим образом [35]:

1. в начало предложение всегда ставится то слово, применение которого допустимо в начале;
2. в конец предложения всегда ставится то слово, применение которого допустимо в конце;
3. обязательное соблюдение порядка в сочетании с правилами синтаксиса.

Стандартно текст или реферат составляется по определенным критериям с правильной расстановкой последовательности предложений, абзацев, разделов, слов.

1.2.3 Адаптация к предметной области

Особо важной считается адаптация классификаторов к современным предметным сферам. На сегодняшний день есть несколько разновидностей адаптации.

1. Адаптация при помощи использования комплексов классификаторов: классификаторы разного рода можно сгруппировать. В определенный такт алгоритма каждый из классификаторов имеет свое определенное значение, участвующее при подведении конечных результатов. Для корректного использования полученных данных применяются различные способы мета-классификации. Впоследствии происходит калибровка показаний, полученных из фрагментов классификации.

2. Для различных доменов возможно применение принципа разделения. Изъясняясь простыми словами, для каждого домена формируют свой собственный словарь. Благодаря использованию данного способа количество особых выражений стремится к нулю, при этом использованные эмоционально окрашенные тексты остаются неизменными во всех сферах.

3. Мгновенное обучение классификатора на всех возможных пакетах информации служит оптимальным вариантом. Конечный результат при использовании данного способа недостаточно высок, в отличии от классификатора отдельных доменов. Как правило, данный метод выбирается в качестве фундамента для остальных способов.

При использовании данных внутри домена, которые не отмечены, нужно для малых частей ранее размеченных данных установить определенные параметры для наилучшего распознавания байесовским классификатором, применяющим алгоритм EM.

При анализе тональности текстов применимы методики статистики благодаря их всеобщей доступности и простоте использования. Изъяном данного метода является необходимость размещения дорогостоящего корпуса. Методы, основанные на выявлении тональностей и эмоций, получают базу данных из тональных словарей. Качество работы напрямую зависит от полноты лексикона используемых словарей. Исходя из вышеперечисленного, можно подытожить, что выбор и использование подхода напрямую зависит от сопутствующих факторов и баз данных.

1.2.4 Компьютерная лингвистика

Под понятием «компьютерная лингвистика» принято понимать междисциплинарную область, возникшую при столкновении следующих наук – лингвистика, математика, информатика и искусственный интеллект. По мере своего развития компьютерная лингвистика использует (в случае необходимости подстраиваясь) созданные в данных научных дисциплинах методики и средства.

В связи с тем, что в компьютерной лингвистике объектом обработки служат тексты на естественном языке, ее развитие не может осуществляться без наличия базовых знаний в области общей лингвистики (знание языка). В соответствии с составляющими в компьютерной лингвистике исследуются общие законы и правила естественного языка – его конструкция и функциональность. Также она содержит в себе следующие фрагменты:

– лексикография – отображает лексикон определенного естественного языка, то есть его отдельные слова, их грамматические и семантические качества, и конечно же методики формирования словарей;

– морфология – изучает внутреннее устройство и внешнюю модель словесной речи, в том числе части речи и их группы;

– семантика и прагматика – это две сферы которые взаимосвязаны между собой. Семантика – исследует смысл слов или предложений, а также остальные единицы речи, прагматика – изучает особенные свойства выражения данного смысла, связанные с определенными задачами общения;

– синтаксис – исследует конструктивную часть предложений, принципы сочетаемости и порядок следования слов в данном предложении, кроме этого, его общие качества и особенности, как единицы языка;

– фонология – исследует звуковую часть речи, т.е. роль звуков, используемых в языке.

Сфера разнообразных приложений лингвистики для ПК периодически расширяется. Дадим характеристику наиболее популярным прикладным задачам и инструментам. Для решения заданий экономической и производственной аналитики, зачастую необходима установка Text Mining (получение сведений из текста). Для того, чтобы выполнить задачу, нужно выделить в тексте естественного языка конкретные предметы, обозначающие сущность (географические наименования, имена различных персонажей и т.д.) их связи, и события, относящиеся к ним. Как правило, это происходит на базе неполного

синтаксического анализа текста, способного осуществлять обработку достаточно больших объемов текстов. Выделенная информация имеет свойство структурироваться.

С Text Mining связаны еще несколько однотипных задач – анализ тональности текстов и выделение конкретных мнений. На данный момент эти задачи, из-за своей популярности, привлекают к себе большое число разных исследователей. Рассмотрим первую задачу – она подобие обычной задачи контент-анализа текстов многочисленной коммуникации. Одна из главных целей этой задачи, оценить комплексность тональности текста в общем и в отдельном рассуждении. Рассмотрим вторую задачу – это поиск (на страницах, или в блогах) высказываний покупателей о продукции, сервисе и других объектах. На основе этих данных осуществляется глубинный анализ.

Поддержка диалога на естественном языке – прикладная задача, которая начала активно развиваться вместе с появлением интернета. Она позволяет использовать упрощенные методы анализа вопросов, а ответы создавать по шаблонам. В современном мире широко распространено применение чат-ботов во всемирной информационной сети [44].

1.3 Определение эмоционального направления текстового содержимого

1.3.1 Роль эмоций в процессе текстообразования

Эмоциональное направление текста можно рассмотреть с двух сторон: содержание и выражение. Содержание, в свою очередь, делится на когнитивное содержание текста и прагматическую стратегию автора. Если рассматривать выражение, то в нем эмотивность линейна, а в тексте выражена большим набором языковых и текстовых маркеров.

Функционально семантическое поле в тексте представлено несколькими позициями: эмотивная направленность, эмотивный фон, эмотивная окраска,

эмотивная тональность. В разных текстах своя особенность эмотивного содержания и выражается она в эмотивной окраске [30].

Каким же текстам присуща эмотивность? Практически всем функциональным стилям – научный, публицистический, официально-деловой и художественный. Именно этим стилям присуща психологическая эмоциональность. Специфика эмотивности текста регулируется функционально стилевыми нормами, потому что определяется сопоставлением эмотивного фона, тональности и окраски.

Основным вопросом лингвистики является проблема текстообразования. Обусловлено это актуальностью текста, как объекта изучения, а также отсутствием четко разработанной теории текстопостроения. Самым эффективным путем к изучению текста в современное время является исследование речевых произведений определенного вида. Эмотивный аспект текста, является самым сложным и важным из числа похожих исследований.

Субъективная оценочность – фундамент плана содержания эмотивности. Основными ресурсами эмотивности текста являются собственно эмотивные языковые средства. Совокупность эмоций в тексте – специфическое динамическое множество, которое меняется по ходу развития сюжета. Эмоции в полной мере отражают внутреннее мироощущение персонажа в отношениях с другими персонажами и в разнообразных обстоятельствах. При этом в эмоциональной сфере субъекта преобладает эмоциональное состояние, свойство, доминирующее над остальными [35].

1.3.2 Эмотивная составляющая в тексте

В эпоху современных информационных технологий обширно используются системы обработки коммуникационной и метаязыковой функций коммуникации. При этом появляется потребность обработки и других функций: фатической, эмотивной, апеллятивной, оценочной. При постоянном взаимодействии человека с компьютером применяется информационная функция коммуникации. Ее

используют при необходимости получить или уточнить любую информацию. Эмотивную функцию коммуникации стараются применять в автоматических системах оценки и сравнения объектов [23].

Эмоциональная составляющая коммуникации пока не так активно применяется в механизмах обработки текста. Это обуславливается проблемой разделения «необходимых» слов эмоционального характера в текстах и хитростью определения непосредственно эмотивного пространства, количества и состава его измерений. Но в новейших механизмах автоопределения эмоциональной оценки текста, зачастую применяют одномерное эмотивное пространство – это негатив и позитив, или другими словами, плохо или хорошо, так как нынешняя теория эмоций в лингвистике еще слабо развита.

1.3.3 Понятие лексической тональности и тональности предложения

Термин «тональность» можно расшифровать как эмоциональная оценка, проявление которой происходит непосредственно в тексте. Если говорить о лексической тональности, то она выступает в качестве эмоциональной составляющей, проявление которой происходит в контексте на уровне коммуникативного фактора и лексемы. По своей сути тональность текста в общем обуславливается лексической тональностью компонентов ее единиц и правилами их сочетания.

Истинная дефиниция тональности текста подразумевает разделение фрагментов текста, которые отражают отрицательную и положительную эмоциональность касаемо объекта эмоциональной оценки (объект тональности). В роли данного объекта может выступать имя собственное, название товара, предприятия, услуги, профессии и пр., относительно которого осуществляется анализ текста. Вместе с тем объект эмоциональной оценки можно установить в предложениях в качестве любого имени собственного либо нарицательного, а также он может быть установлен в форме одного целого для текста (учитывая при

этом его синонимические и анафорические употребления). Из этого следует, что тональность текста определяется тремя основными факторами:

1) оценка тональности (позитивная/нейтральная/ негативная);
2) объект тональности;
3) субъект тональности.

Под тональной оценкой подразумевается эмоциональное отношение автора к такому объекту. Объект тональности – тот, о ком он высказывается. А под субъектом тональности подразумевается автор [35].

1.3.4 Методы определения тональности текста

К основным методам определения тональности относятся следующие.

1. Изучение эмотивной лексики в тексте по ранее заданным словарям тональности (перечисления паттернов), применяя лингвистический анализ. В сочетании с имеющейся эмотивной лексикой текст можно оценить по шкале, отображающей число, как и негативной так и позитивной лексики. Этот метод должен использовать не только определенные перечисления паттернов, которые вполне могут быть постоянными выражениями, но и применять очередность воссоединения тональной лексики внутри этого предложения.

2. Оценивание текста, используя методики векторного анализа, и сопоставление с заранее распределенным эталонным корпусом по избранной мере приближенности и перенесения (классификация) текста к негативному или позитивному, отталкиваясь от приобретенных сравнительных результатов.

3. Смешанная методика (комбинация первой и второй методики).

Первая методика достаточно сложная при формировании тональных словарей (либо для получения списка тональных паттернов), однако при совместном использовании с синтаксическим и морфологическим анализом она достаточно гибкая: благодаря ей можно показать цепочки тональной лексики, и установить синтаксически корректные эмоциональные выражения. В случае надлежащего наполнения тональных словарных списков данная методика даст

прекрасную возможность достичь хорошей полноты (покрытия эмотивной лексики).

Отрицательной стороной данной методики выступает то, что, применяя ее, непросто совершить количественную оценку негативности-позитивности текста. Во избежание недостатков первой и второй методики, применяют смешанную методику.

Вторая методика функционирует наиболее действенно, однако требует присутствия заблаговременно распределенного эталонного корпуса, благодаря которому осуществляется обучение алгоритму сравнения. Основными отрицательными аспектами данной методики выступают рост трудоемкости и ограничение разнородности корпуса (другими словами, неполнота лексического покрытия), что способствует в результате утрате точности. Более того, методика не дает возможность выполнить глубокий анализ текста [30].

1.3.5 Определение тональности с использованием тональных словарей и лингвистического анализа

Тональность текста подлежит обязательному анализу, который выполняется в четыре стадии.

1) В соответствии с первой стадией анализа происходит проработка отдельного лингвистического устройства. Данное устройство автоматическим образом выполняет морфологическую оценку текста, лемматизацию всей лексики текста, определяет часть речи каждого слова, его морфологические параметры, а также обозначает данное слово в предложении и определяет его тип.

2) Вторая стадия включает в себя распределение всех слов и словосочетаний по заданным каталогам слов тональной лексики. Любое слово характеризуется двумя признаками – его тональность и уровень силы. При условии, что слово не обнаружено в имеющихся каталогах, его именуют как нейтральное слово.

3) Третья стадия включает в себя исходный анализ синтаксиса. Другими словами, осуществляется соединение слова и сочетания слов в тональном ряду, а в тексте предложения отмечается субъект, предикат и объект. Идентифицируются причастные/деепричастные обороты, подчинительные предложения, анафорические взаимосвязи и т.д. Безусловно, нельзя сказать, что каждое предложение в русском языке можно выразить как триаду субъект-предикат-объект. Во внимание принимаются и безличные, неопределенно-личные и обобщенно-личные предложения, предложения, которые имеют нулевую форму глагола, сказуемые, которые выражены неглагольной формой.

4) В конце анализа в предложении уже выделяется объект тональности и указывается его сентимент, обусловленный расположением и значимостью этого объекта [14].

1.3.6 Подходы для автоматического определения тональности текста

Под понятием «автоматическая обработка языка» принято понимать изменение текста на естественном либо искусственном языке за счет применения высоких компьютерных технологий. Для оптимального функционирования системы автоматизированной обработки, где обязателен к исполнению ряд заданий, требуется четкое определение и построение специального языка, на котором система с нами общается. Более того, важно не нарушать ряд утвержденных правил по набору символов: набор символов нужно совершать в определенной последовательности таким образом, чтобы в последующем система могла распознать, что ей требуется выполнить.

Учитывая ряд определенных моделей, система может оказать помощь в разрешении определенных проблем, в числе которых лексемизация либо токенизация (разбор данных на лексемы), коррекция орфографических ошибок, получение данных, грамматический или синтаксический разбор предложения, выявление смысла, ответы на вопросы, анализ эмоциональной окраски высказывания, автоматизированный перевод и пр.

Важно отметить, что автоматическая обработка языка выступает достаточно популярным направлением существующих сегодня технологий: ее применяют в технологиях информационного поиска (в случае проведения индексирования текстов), в машинном переводе, в том числе в словарной работе [14].

Для того, чтобы автоматически определить тональность текста необходимо применить ряд ниже представленных методик [41]:

1) гибридная методика. Она заключается в сочетании всех либо нескольких из выше представленных принципов и в том, что использует классификатор, который основывается на определенной последовательности;

2) обучение с помощью компьютера, без присутствия педагога (unsupervised learning). Эта методика берет за основу концепцию, в которой важнейшую роль играют термины, чаще всего встречающиеся в данном тексте и вместе с тем есть в определенном объеме во всем собрании. Отметив эти термины и установив их тональность, можно определить тональность всего текста;

3) обучение с помощью компьютера в присутствии педагога (supervised learning). Данная методика требует присутствия обучающей коллекции распределенных в границах сенсетивного пространства текстов, на основе которой формируется статистический либо вероятностный классификатор (к примеру, байесовский);

4) методика, которая опирается на правила с применением шаблонов (rule-based with patterns). Методика сводится к разработке правил, благодаря которым можно будет установить тональность текста. С этой целью текст надо разбить на слова либо последовательность слов (N-grams). Далее используют данные, полученные для того, чтобы выделить наиболее применяемые шаблоны, которые оцениваются либо положительно, либо отрицательно. Отмеченные шаблоны используют в процессе разработки правил в следующем виде – «ЕСЛИ условие, ТО заключение».

1.4 Постановка задачи

В результате изучения теоретических вопросов в направлении особенностей определения эмоциональной окраски текста были определены следующие задачи, решение которые позволит создать чат-бот, который поддерживает анализ и выражение эмоций:

- выбрать технологии реализации чат-бота;
- определить сервис для поставки служб для чат-бота;
- разработать концептуальную модель чат-бота;
- разработать логическую модель чат-бота;
- осуществить физическое моделирование чат-бота;
- проверить работоспособность чат-бота.

В разработанном приложении предлагаются различные возможности: инициация начального диалога, определение эмоционального наполнения сообщения, формирование ответов с учетом эмоциональной составляющей, поддержка разговора, защита (шифрование) данных пользователей, сохранение диалога.

Выводы по главе 1

В результате выполнения первой главы работы был проведен анализ и определены особенности выражения эмоций, психологические аспекты эмоций, рассмотрены техники автоматизации определения эмоций. Также были определены методы и способы анализа выражения эмоций, изучены особенности определения эмоционального направления текстового содержимого.

Глава 2 Логическое проектирование чат-бота, поддерживающего анализ и выражение эмоций

Чат-бот – это программное обеспечение, имеющее такую функциональную возможность как имитация диалога с пользователем посредством естественного языка и способов обмена смс, web-сайтов либо мобильных программных продуктов. Главное достоинство подобных программ выражается в их высокой эффективности, в частности: чат-боты выполняют функцию объединения этапов сложных процессов с целью проведения оптимизации и автоматизации ежедневных, типичных и повторяющихся задач за счет ряда обычных текстовых запросов, снижая при этом временной промежуток исполнения и увеличивая уровень эффективности задачи. Кроме того, онлайн-пользователи имеют возможность быть развернутыми на платформах, где уже имеется ряд потенциальных пользователей, таких как Facebook, Twitter, Вконтакте, Telegram. В связи с этим есть возможность выйти на связь с пользователями в типичной для них среде, что способствует повышению благоприятного восприятия учителя (психолога) в процессе, а сам процесс делает наиболее удобным, быстрым, понятным и привычным. Также, чат-боты могут упростить и ускорить процесс общения с мобильного устройства, браузера или другой платформы, которую поддерживает целевая платформа. Данные сервисы осуществляют поддержку контекста и управления диалогом, динамическим образом осуществляя настройки ответов на основании диалога.

В независимости от того, каким образом разработан механизм программы-собеседника и с помощью какой платформы, участие в данном процессе человека играет определяющую роль в контексте настройки, обучения и оптимизации всей системы. Чат-бот сегодня рассматривается в качестве самого оптимального средства, осуществляющего взаимодействие человека с компьютером. Со стороны технологического аспекта подобный программный продукт – вполне естественное

эволюционное развитие механизмов вопрос-ответ, который применяет обработку разговорного языка.

Для автоматизации распознавания эмоций в тексте посредством чат-бота выбран мессенджер Telegram. Приложение Telegram разработано для популярных платформ (Android, Windows, Mac, Linux), а также имеет веб-версию. Таким образом, его использование возможно практически везде, где есть интернет. Кроме того, Telegram поставляется с удобным API и позволяет создать чат-боты на своей основе.

2.1 Выбор языка программирования реализации чат-бота

Для реализации чат-бота необходим ряд средств, которые обеспечат процесс разработки. Они включают в себя язык программирования, среду разработки, систему управления базой данных, средства проектирования приложения и базы данных, поставщиков сервисов функционала чат-ботов, сервер для локальной разработки.

В качестве языка программирования для создания серверной части приложения (чат-бота) выбран PHP. Этот язык программирования предоставляет возможность создавать программные продукты, в том числе и чат-боты.

Действующий на настоящий момент вариант языка PHP предусмотрен для того, чтобы разработать web-приложения, используемые для различных целей. Сегодня PHP доминирует в использовании его хостинг-провайдерами, поскольку он популярен среди огромного числа разработчиков по всему миру.

Первоначально язык PHP разрабатывался в качестве языка программирования, который должен был использоваться при разработке web-приложений (сценариев), реализуемых на web-сервере. Данный факт является главным его достоинством, так как входящие в него инструменты дают

возможность достаточно быстро и результативно разрабатывать полноценные многофункциональные web-приложения.

Еще одно преимущество PHP выражается в наличии функции по интеграции собственных сценариев в HTML-код web-страниц, что заметно облегчает задачи по разработке динамических web-сайтов. За счет PHP у программистов есть возможность динамическим образом корректировать HTML-код страниц в зависимости от того, какие действия во время посещения сайта производит пользователь.

Данный язык предоставляет возможность посылать интернет-браузерам копии файлов, которые запрашиваются, а также совершать запуск небольших программ для того, чтобы выполнять различные задачи и функции, так называемые PHP-скрипты. Сравнительная характеристика PHP с другими популярными серверными языками программирования показана в таблице 1.

Таблица 1 – Сравнение языков программирования

	PHP	Ruby	Python
Предназначение	Создание динамических веб-приложений	Создание веб-приложений и компьютерах программ	Создание динамических веб-приложений и компьютерных программ
Сложность освоения	6	4	5
ООП	Да	Да	Да
Инструкция break	Да	Да	Да
Многомерные массивы	Да	Да	Да
Цикл foreach	Да	Да	Да
Множественное наследование	Нет	Нет	Да
Макросы	Нет	+/-	Нет
Именованные параметры	Нет	Да	Да
Наличие библиотек для работы с графикой и мультимедиа	Да	Да	Да
Работа с MySQL	Да	Да	Да
Наличие удобных и доступных средств разработки	Да (NetBeans)	Нет	Да (Visual Studio)
Поддержка популярными веб-серверами	Да (по умолчанию)	Нет (дополнительные службы)	Нет (дополнительные службы)

Как видно из таблицы, рассмотренные языки программирования сходятся по ряду показателей, при этом PHP имеет большое сообщество пользователей, активно развивается, в связи с этим в качестве языка программирования для создания серверной части чат-бота будет использован PHP.

Для разработки веб-приложений на PHP мы выбрали среду разработки PhpStorm. Данный программный продукт представляет собой интегрированную среду разработки, имеющую интеллектуальный редактор, отлично распознающий код, обеспечивающий автоматическое дополнение кода, рефакторинг, в том числе, предотвращающий ошибки в функционировании и поддерживающий сочетание языков.

В PhpStorm реализована поддержка инновационных технологий web-разработки, в том числе PHP, HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMA Script Harmony, шаблоны Jade, ZenCoding, Emmet, JavaScript.

PhpStorm выполняет следующие функции:

– интеллектуальный редактор PHP кода с подсветкой синтаксиса, автодополнением кода, расширенными настройками форматирования кода, предотвращением ошибок на лету;

– поддержка PHP 7.0, 5.6, 5.5, 5.4 и 5.3, генераторов, сопрограмм;

– PHP рефакторинг, code (re)arranger, детектор дублируемого кода;

– поддержка Vagrant, Composer, встроенный REST клиент, Command Line Tools, SSH консоль;

– поддержка фреймворков (MVC view для Symfony2, Yii) и специализированных плагинов для ведущих PHP фреймворков (Symfony, Magento, Drupal, Yii, CakePHP и многие другие);

– визуальный отладчик для PHP приложений, валидация конфигурации отладчика, PHPUnit с покрытием кода (поддержка PHPUnit 5), а также интеграция с профилировщиком;

– HTML, CSS, JavaScript редактор; отладка и модульное тестирование для JS; поддержка HTML5, CSS, Sass, SCSS, Less, Stylus, Compass, CoffeeScript, TypeScript, ECMAScript Harmony, Emmet и других технологий веб-разработки;

– полный набор инструментов для фронтенд-разработки;

– поддержка стилей кода, встроенные стили PSR1/PSR2, Symfony2, Zend, Drupal и другие;

– интеграция с системами управления версиями, включая унифицированный интерфейс;

– удалённое развёртывание приложений и автоматическая синхронизация с использованием FTP, SFTP, FTPS и др.;

– LiveEdit: изменения в коде можно мгновенно просмотреть в браузере без перезагрузки страницы;

– PHP UML;

– интеграция с баг-трекерами;

– инструменты работы с базами данных, SQL редактор;

– кроссплатформенность (Windows, Mac OS X, Linux).

PhpStorm 10 обеспечивает расширенную поддержку PHP 7, включая инспекции совместимости. Обеспечивает возможность работы с:

− безымянными классами;

− декларациями типов возвращаемого значения;

− операторами объединения со значением «ноль»;

− декларациями группового использования;

− лексическими анализаторами, учитывающими контекст, а также другие возможности версии PHP 7.

Для работы проектируемой системы (чат-бота) требуется база данных. К базе данных и сопровождающим СУБД, согласно которым нужно соблюдать ряд условий, позволяющих создать безопасный чат-бот, который надежно защищает данные пользователей.

1) Несложный процесс по обновлению данных. Обновление данных, в свою очередь, определяется как добавление, удаление или их изменение.

2) Высокий уровень реагирования на запросы. Понятие «время отклика» интерпретируется как промежуток времени от момента, когда был создан, точнее, отправлен запрос на базу данных, и до реального получения данных. Немного похож на данное понятие термин «время доступа», определение которого выглядит как промежуток времени между пуском команды записи (считывания) и реальным получением данных. Понятие «доступ» характеризуется как процесс поиска, чтения информации либо ее записи.

3) Независимость данных. Под понятием «независимость данных» принято понимать функцию по преобразованию логической и физической структуры базы данных, не изменяя при этом пользовательские данные. Независимость данных предусматривает инвариантность в отношении хранения информации, ПО и техсредств. Независимость данных отвечает за обеспечение минимального количества преобразований структуры базы данных в случае модернизации стратегии доступа к информации и структуры непосредственно имеющихся исходных данных.

4) Совокупное применение данных большим числом пользователей.

5) Безопасность данных. Данное понятие представляет собой защиту внесенных данных от преднамеренного/непреднамеренного несоблюдения конфиденциальности, искажения либо разрушения. Безопасность данных отвечает за их целостную сохранность (целостность) и защиту. Под целостностью данных понимается устойчивость охраняемых сведений к разрушению и уничтожению, которая взаимосвязана с поломкой технических средств, ошибками в системе и неверными действиями со стороны пользователей.

База данных предусматривает следующие аспекты:

– отсутствует возможность неверно внесённых данных либо 2-х аналогичных записей об одном и том же явлении и т.п.;

– блокировка от возникновения ошибок во время процессов обновления базы данных;

– отсутствует возможность удалить информацию, внесенную в связанные таблицы;

– данные в процессе работы в режиме, когда много пользователей, и в дифференцированных базах данных не искажаются;

– безопасность данных в случаях технических сбоев и т.п. (восстановление данных).

Обеспечение целостного состояния осуществляется за счет триггеров целостности, а именно, специально созданными приложениями, которые функционируют в рамках утвержденных условий. Организация защиты данных от взлома в обязательном порядке предусматривает ограничение доступа к конфиденциальной информации.

Для того, чтобы данные находились в безопасности, необходимо выполнить следующие мероприятия:

– подключение системы паролей;

– дифференциация доступа к базе данных посредством использования специально разработанных настроек, которые могут производится только администратором базы данных;

– генерация видов – специально составленных таблиц, которые опираются на исходные, т.е. те таблицы, что уже существуют в базе данных, и предназначены для определенных пользователей;

– подведение под стандарты формирования и пользования базой данных. Таким образом, процесс стандартизации гарантирует преемственность поколений СУБД, заметно облегчает процессы взаимосвязи базы данных одного поколения СУБД с одними и теми же или разными моделями данных;

– соответствие отображения информации по надлежащему предмету;

– интуитивно-понятный пользовательский интерфейс.

Для того, чтобы реализовать систему, выбрана база данных, которая имеет реляционную модель данных, позволяющую благоприятным образом производить сохранение введенных и формируемых данных. Процесс выбора подходящей СУБД для того, чтобы реализовать ИС, является достаточно трудоемким мероприятием, требующим определенных знаний, умений, навыков и интереса со стороны программиста. Грамотный выбор СУБД даст возможность не только сэкономить время, но и обеспечит высокий уровень безопасности ИС. К популярным СУБД на бесплатной основе, которые используются для создания веб-приложений, относятся MySQL и PostgreSQL. В таблице 2 представлен сравнительный анализ данных систем управления базами данных [25] MySQL и PostgreSQL, которые конкурируют на рынке ПО согласно основным показателям.

Таблица 2 – Сравнение СУБД

Параметры	MySQL	PostgreSQL
Краткое описание	Широко используемая свободная реляционная система управления БД	Широко используемая свободная реляционная система управления БД
Основная модель хранения данных	Реляционная БД	Реляционная БД
Дополнительная модель хранения данных	БД типа Key/Value, документно-ориентированная БД	БД типа Key/Value, документно-ориентированная БД
Разработчик	Oracle	PostgreSQL Global Development Group
Лицензия	Открытое ПО	Открытое ПО
Язык реализации	C++, C	C
Поддерживаемые операционные системы сервера	FreeBSD, Linux, Solaris, OS X, Windows	FreeBSD, Linux, Solaris, OS X, Windows, NetBSD, OpenBSD, HP-UX, Unix
Схема данных	Да	Да
Типизация	Да	Да
Поддержка XML	Да	Да
Поддержка вторичных индексов	Да	Да
SQL	Да	Да

Продолжение таблицы 2

API и другие методы доступа	Проприетарное нативное API, ADO.NET, JDBC, ODBC	Нативная C библиотека, потоковое API для больших объектов, ADO.NET, JDBC, ODBC
Поддерживаемые языки программирования	Ada, C, C#, C++, D, Delphi, Eiffel, Erlang, Haskell, Java, JavaScript (Node.js), Objective-C, OCaml, Perl, PHP, Python, Ruby, Scheme, Tcl	.Net, C, C++, Delphi, Java, Perl, PHP, Python, Tcl
Язык написания скриптов на стороне сервера	Да	Функции, которые определены пользователем
Триггеры	Да	Да
Методы разбиения	Горизонтальное разбиение, шардинг с MySQL Cluster или MySQL Fabric	декларативное разбиение (по диапазону или списку) начиная с PostgerSQL 10.0
Методы репликаций	Master-Master, Master-Slave	Master-Slave
MapReduce	Нет	Нет
Концепции согласования	Немедленное согласование	Немедленное согласованиее
Параллелизм	Да	Да
Возможность хранения только в памяти	Да	Нет
Контроль доступа пользователей	Концепт пользователей с детальной авторизацией	Детальные права доступа в соответствии с SQL стандартом

Как видно из таблицы, рассматриваемые СУБД схожи по множеству параметров, при этом важно отметить, что СУБД MySQL более популярна в среде PHP-разработчиков, и соответственно поддержка ПО, которое использует СУБД MySQL будет дешевле. Таким образом, для проекта чат-бота будет использована СУБД MySQL.

Для проектирования базы данных CRM-системы будет использована среда MySQL Workbench. Функции, которые присутствуют в программе:

– есть возможность на примере представить модель базы данных в графической форме;

– интуитивно-понятная функциональная система, отвечающая за установление взаимосвязи между таблицами, а также взаимосвязи «многие ко многим» с функцией по формированию таблиц связей;

– функция, отвечающая за восстановление структуры таблиц, которая уже присутствует на сервере базы данных (восстановление связей происходит посредством InnoDB совместно с MyISAM, где в дальнейшем установку связей необходимо производить ручным способом);

– доступный и понятный редактор SQL-запросов, который дает возможность мгновенно совершать отправку их на сервер и получать ответ в форме таблицы;

– функция редактирования информации в таблице в режиме визуального моделирования.

Взаимодействие составных частей чат-бота (веб-сервера, клиента, сервисов) будет осуществляться с помощью HTTP-запросов и REST API. REST представляет собой стиль архитектурной организации ПО, созданный для того, чтобы создать распределенные масштабируемые web-сервисы, применяемые HTTP-запросы [9].

Протокол передачи гипертекста – HTTP является протоколом стека TCP/IP. Первоначально его разработали для того, чтобы публиковать и получать HTML-страницы. Однако, на сегодняшний день он применяется для работы распределенных ИС. Применяют HTTP в интернет-сети для передачи данных, и это сегодня самый популярный прикладной протокол.

В основе HTTP лежит технология «клиент-сервер». Например, клиент web-браузер осуществляет отправку запроса на сервер, HTTP-протокол устанавливает типы сообщений, которые используются клиентом с целью запроса web-страницы, включая типы сообщений, используемые сервером с целью предоставления ответа. Самые популярные типы сообщений – это GET, POST и PUT.

POST и PUT можно применять для отправки сообщения, они же и загружают данные на web-сервер. В случае, когда пользователь вносит информацию в форму, встроенную в web-страницу, POST интегрирует эти сведения в сообщение, которое отправляет на сервер. PUT отвечает за загрузку ресурсов либо контента непосредственно на web-сервер.

Несмотря на то, что HTTP имеет крайне гибкий характер, данный протокол недостаточно защищен. Сообщения POST загружают данные на сервер в форме обычного текста, который может быть вскрыт и прочитан. Также и ответы сервера, в основном HTML-страницы, не имеют шифровку.

Для того чтобы организовать безопасное соединение через Интернет, применяется безопасный HTTP-протокол (HTTPS) с целью получения доступа либо публикации сведений на web-сервере. Для HTTPS допускается применение аутентификации и шифрования с целью обеспечения защиты данных, в случае, когда происходит их движение непосредственно от клиента к серверу и наоборот. Также HTTPS имеет возможность установить новый ряд правил для того, чтобы данные могли пройти между прикладным и транспортным уровнями.

У REST есть возможность установки архитектурных принципов проектирования web-сервисов, которые направлены на системообразующие ресурсы. Также REST владеет методами обработки и передачи состояний ресурсов по HTTP различными клиентами (приложениями), которые были созданы с помощью разных языков программирования. На сегодняшний день в большинстве случаев применяется REST, поскольку она признается оптимальной и лучшей моделью разработки web-сервисов. Более того, можно сказать, что REST колоссально повлияла на Web, практически вытеснив дизайн интерфейса, который был создан на SOAP и WSDL. Данный феномен обусловлен существенно простым стилем разработки.

В чистом виде (в каком она привлекает столь пристальное внимание) конкретная реализация web-сервисов REST следует четырем базовым принципам проектирования:

– открытое применение HTTP-методов;

– возможность не сохранять состояние;

– возможность предоставить URI, которые идентичны структуре каталогов;

– передача данных в XML, JavaScript Object Notation (JSON) или в обоих форматах.

Определяющей характеристикой web-сервиса RESTful выступает использование HTTP-методик согласно протоколу, установленному в RFC 2616. В частности, установка HTTP GET используется как метод генерирования формирования данных, применяющихся клиентским приложением для того, чтобы извлечь ресурс, получить информацию с сервера или реализовать запрос. Данный запрос способствует тому, что web-сервер определяет и исполняет возврат набора необходимых в данном случае ресурсов.

В контексте REST следует использовать HTTP-методику при установке протокола. Это основной принцип системы проектирования REST, который устанавливает согласованность между операциями create, read, update и delete (CRUD) и HTTP-методиками. В соответствии с этим принципом можно выполнить следующее:

– для создания ресурса на сервере используют POST;

– для извлечения ресурса используют GET;

– с целью изменения или обновления состояния ресурса используют PUT;

– с целью удаления ресурса используются DELETE.

Общепринятым подходом, соответствующим рекомендациям REST по явному применению HTTP-методов, является использование в URI имен существительных вместо глаголов. В web-сервисе RESTful глаголы POST, GET, PUT и DELETE уже определены протоколом. В идеале для реализации обобщенного интерфейса и явного вызова операций клиентскими приложениями web-сервис не должен определять дополнительные команды или удаленные процедуры, например, /adduser или /updateuser. Этот общий принцип применим также к телу HTTP-запроса, которое предназначено для передачи состояния ресурса, а не имени вызываемых удаленного метода или удаленной процедуры.

Для того, чтобы удовлетворить регулярно возникающие требования к работоспособности web-сервиса необходимо, чтобы REST был масштабируемым. Чтобы сформировать топологию сервисов, дающую возможность в случае надобности выполнить перенаправление запроса с одного сервера на другой для уменьшения времени реагирования на вызов web-сервиса, как правило, используют серверные кластеры с возможностью дифференциации нагрузки и аварийного переключения на резерв, а также прокси-серверы и шлюзы. Обращение к промежуточным серверам для улучшения масштабируемости необходимо, чтобы клиенты web-сервисов REST осуществляли отправку полноценных запросов, которые содержат все необходимые для их осуществления данные. Цель таких запросов – осуществление элементами на промежуточных серверах перенаправления, маршрутизации и распределения нагрузки при отсутствии локальной возможности сохранить состояние между запросами.

Когда происходит обработка цельного запроса, у сервера нет необходимости в извлечении состояния либо контекста приложения. Приложение web-сервиса REST включает HTTP-заголовок и в самой структуре запроса параметры, контекст и сведения, которые требуются серверному элементу для генерирования ответа. С этой точки зрения отсутствие возможности сохранить состояние (statelessness) способствует улучшению работоспособности web-сервиса, упрощению дизайна и серверных компонентов, так как при отсутствии состояния на сервере нет надобности синхронизировать данные сеанса с внешним приложением.

Состояния, которые не сохраняют элементы сервера, отличаются простотой при моделировании, создании и распределении между серверами в соответствии с равной нагрузкой. Сервис такого типа, как правило, демонстрирует лучшую производительность, а также делегирует основную долю ответственности за сохранение состояния на приложение клиента. В web-сервисе RESTful сервер выполняет функцию составления ответов и имеет интерфейс, который дает возможность клиентскому приложению самостоятельно выполнять сохранение

своего состояния. В частности, делая запрос многостраничного набора результатов, необходимо, чтобы клиентское приложение в запросе отражало номер определенной страницы, а не какой-либо следующей страницы.

Web-сервис, имеющий возможность несохранения состояния, формирует ответ, который отражает ссылку на номер следующей страницы в наборе, и позволяет клиенту самостоятельным образом сохранить данное значение. В целом модель web-сервисов RESTful можно разделить на две сферы ответственности – сервер и клиентское приложение, объясняющие суть функционирования не сохраняющего состояние сервиса. Работа клиентского приложения в сочетании с сервисом играет важную роль с точки зрения отказа от сохранения состояния в web-сервисах RESTful. Конечный результат в данном случае представляется как повышение уровня производительности посредством сокращения трафика и минимизации состояния приложения на сервере.

Со стороны применения ресурсов из клиентского приложения предлагаемые URI устанавливают, в достаточной ли степени будет интуитивно-понятным web-сервис REST и будет ли он применяться также, как запланировал создатель. Еще один отличительный параметр web-сервиса RESTful целиком и полностью относится к URI-адресам.

Простыми и довольно легкими представляются URI-адреса web-сервиса REST. Механизм URI должен быть достаточно удобным и понятным. Наиболее простой способ дойти до максимально удобного уровня применения – это конструирование URI по образцу каталогов. Такие URI считаются иерархическими, имеющими единый корневой путь, а их ветви показывают главные характеристики сервиса. Исходя из такого определения, можно сделать выводы, что URI – это не обычная строка, имеющая слэши, как свои разделители, оно больше походит на дерево с лежащими достаточно высоко и низко ветками, которые соединяются в узлы.

Образность ресурса зачастую показывает имеющееся положение этого ресурса (его инструментов) на период запроса клиентским приложение. Ресурсное представление в этот момент считается обычным кадром состояния. Ресурсное представление обязано быть упрощенным, так же, как и представление в информационной базе, складывающееся из отображения посередине имен столбцов и фрагментов «JSON», где обозначения фрагментов в «JSON» соответствуют обозначению строк. В том случае, когда механизм содержит модель данных, то, опираясь на это значение, представление ресурса считается кадром состояния инструментов одного из объектов модели данных. Web-сервис REST должен обслуживать данные объекты.

Конечный набор ограничений очень плотно связан с дизайном web-сервисов RESTful и касается формата данных, которыми в свою очередь обмениваются приложения и сервис во время процесса в режиме «запрос-ответ» либо в качестве HTTP-запроса. В этом случае ценится легкость, ясность и простота.

Чаще всего объекты модели данных связаны между собой, и эти связи нужно отражать в способности их представлять для отдачи клиентскому приложению. Информация во время обмена уходит и приходит в формате JSON – это самый обычный и простой формат для обмена данными, он вполне удобен во вовремя чтения и заполнения и для человека, и для ПК. Свою основу он берет от языка программирования «JavaScript», установленного стандартом (ECMA-262 3rd Edition, December 1999). Текстовый формат «JSON» не имеет зависимости от языка реализации и управляется соглашениями. Такие особенности показывают, что «JSON» лучший язык для обмена данными. Свое начало «JSON» берет от двух структурных данных:

– собрание пар (ключ/значение). В разнообразных языках эта система осуществлена как объект, механизм, хеш и т.д.;

– организованный каталог значений. Во многих языках это имеет реализацию в качестве массива, каталога, вектора или конкретной последовательности.

По сути, это многофункциональные механизмы данных. Практически каждый язык поддерживает их в определенном формате. Предположим, что формат данных, который не зависит от языка программирования, должен опираться на эти механизмы.

В нотации JSON показано именно так: объект – это совершенно хаотичный набор пар (ключ-значение). Начало объекта – это открывающаяся фигурная скобка, конец – закрывающаяся фигурная скобка. Все имена без исключения идут вместе с двоеточием (ключ-значение) и разделяются запятой; массив – организованный список значений. Массив имеет свое начало с открывающейся квадратной скобкой и конец – закрывающаяся квадратная скобка. А значения разделяются запятой.

Также значение может определяться строкой в двойных кавычках, числом (true, false, null), объектом или же массивом. Механизмы могут быть и вложенными.

Строка – это определенный набор символов в конкретной кодировке «Unicode» и нуль-терминатора, имеющая двойные кавычки, применяющая обратный слеш в виде символа экрана. Символ характеризуется как одна символьная строка. Число характеризуется в десятичном механизме счисления. Трудности могут создавать любые лексемы.

SOCKS 4/5 – это распространённое обозначение двух версий сетевого протокола SOCKS 4 и SOCKS 5. Основным преимуществом SOCKS (от. англ. «sockets» – гнёзда) является возможность работы клиент-серверных приложений за границами межсетевого экранирования, то есть SOCKS прокси может принять запрос от клиента, находящегося за фаерволом, просмотреть его права доступа и передать запрос на внешний сервер.

Кроме представленных выше средств для реализации чат-бота необходим семантический разбор естественного языка. Для облегчения разбора разрабатываются шаблоны семантического анализа – определенные алгоритмы, которые, при применении их в текст, будут возвращать наличие или отсутствие в нем определенных признаков.

Для разработки предметно-ориентированной системы знаний чат-бота можно использовать язык разметки для искусственного интеллекта AIML (Artificial Intelligence Markup Language), собственный набор знаний в виде размеченных специальным образом файлов. Эти файлы будут содержать различные шаблоны, которые относятся к конкретной предметной области и помогут чат-боту фокусироваться на конкретной теме общения. В случае реализации задания шаблоны для проекта должны содержать конструкции, которые относятся к сфере определения эмоциональной окраски сообщений.

Говоря о языке разметки AIML, отметим, что он представляет собой модифицированную вариацию языка XML (Extensible Markup Language). Главная цель создания языка AIML заключается в том, чтобы работали функции, которые требуются для извлечения и обработки данных и знаний, соответствующих шаблону, в том числе для формирования выходных сигналов в соответствии со схемой «стимул-реакция». С помощью AIML можно выполнять команды, которые написаны на других языках программирования, а это, в свою очередь, способствует значительному расширению области использования программы и дает возможность интегрировать ее в различные приложения. Ключами в языке выступают category, pattern и template:

```
<aiml>
    <category>
        <pattern>Шаблон вопроса?</pattern>
        <template>
            <random>
                <li>Шаблон ответа.</li>
                <li>Шаблон ответа?</li> </random>
        </template>
```

```
    </category>
</aiml>
```

Тег category представляет собой родительский тег pattern и template, который хранит шаблоны вопроса и ответов. Тег random дает возможность выбирать более одного ответа на вопрос. Данный выбор осуществляется программой случайно. Определенные теги pattern дают возможность выполнить описание разных вариаций вопросов, соответствующих данной категории, и на которые должны представляться идентичные версии ответов.

2.2 Выбор сервиса для создания чат-бота

Для реализации чат-бота выбран сервис Telegram. Телеграм (Telegram) это – кроссплатформенный мессенджер для смартфонов, позволяющий обмениваться сообщениями и медиафайлами разных форматов - фото, аудио, видео, можно даже отправлять голосовые и видео сообщения. На сегодняшний день мессенджеры пользуются большим спросом, это связано с изменением в области мобильного интернета: высокие скорости, низкая цена и широкое распространение смартфонов [1]. Прогрессивность мессенджеров можно сравнить с явлением десятилетней давности – взрывом социальных медиа.

Telegram подходит для реализации чат-бота. Чат-бот – это некий помощник, который ведет диалог с пользователями посредством сообщений и обладает множеством специфичных функций. Чат-бота можно использовать как для рассылки информации, так и для ее сбора.

В зависимости от того, как запрограммированы конкретные боты, их можно разделить на две большие группы: работающие по заранее заготовленным командам и обучающиеся.

Боты, работающие по командам, опираются на заранее написанные ключевые слова, которые они понимают. Каждая из таких команд должна быть написана разработчиком отдельно с использованием регулярных выражений или

других форм анализа строк. Если пользователь задал вопрос, не использовав ни одного ключевого слова, робот не может понять его и, как правило, отвечает сообщениями вроде «простите, я не понял».

Несмотря на то, что функциональные возможности подобных ботов ограничены, они могут быть эффективными в ситуациях, когда пользователю необходимо выбрать несколько пунктов из предложенных.

Обучающиеся боты опираются на искусственный интеллект, чтобы общаться с пользователями. Вместо заранее подготовленных ответов робот отвечает адекватными предложениями по теме. К тому же, все слова, сказанные боту и ботом, записываются для последующей обработки.

Так как сегодня существует достаточно большой выбор всевозможных фреймворков и сервисов, которые могут помочь в создании чат-бота и обучить его некоторым разговорным навыкам, нет необходимости создавать своих чат-ботов с нуля.

Telegram – уникальный по своей структуре мессенджер, являющийся кроссплатформенным приложением. Помимо стандартного обмена сообщениями в диалогах и группах, в мессенджере можно хранить неограниченное количество файлов, вести каналы (микроблоги), создавать и использовать ботов. Посредством специально разработанного API у сторонних программистов появляется возможность разрабатывать боты, узконаправленные и специализированные аккаунты, которые могут управляться непосредственно программами. Обычные чат-боты могут отвечать на спецкоманды в частных, а также в групповых чатах, более того, они могут выполнять поиск в интернет-сети и ряд других задач, они могут также использоваться для развлекательных целей и в бизнес-деятельности [2].

Telegram-боты – разновидность чат-ботов. Их суть заключается в реакции на определенные сообщения от пользователей. Таким образом, сфера их применения безгранична. Множество отечественных и зарубежных организаций используют

Telegram-ботов для упрощения и автоматизации внутренних рабочих процессов. При регистрации бота выдается уникальный ключ, с помощью которого в дальнейшем и будет происходить связь между клиентом и сервером. Такая схема исключает необходимость дополнительных настроек клиент-серверной архитектуры, так как все происходит автоматически и занимает несколько строчек кода.

Таким образом Telegram был выбран, как самая удобная оболочка для разрабатываемого приложения. Для реализации приложения необходимо создание и настройка бота в Telegram. Эта возможность реализована внутри самого мессенджера. Так же для организации работы приложения с Telegram необходим API, позволяющий производить различные действия с этим сервисом. Для работы с Telegram был выбран Telegram.Bot API, так как он есть для языка программирования PHP и имеет множество обучающих материалов.

Телеграм-боты – это сторонние приложения, которые запускаются внутри Telegram. Пользователи могут взаимодействовать с ботами, отправляя им сообщения, команды и встроенные запросы. Пользователи управляют своими ботами, используя HTTPS-запросы к Telegram API ботам.

Этот API позволяет подключать ботов к системе Telegram. Telegram Bots представляют собой специально сформированные учетные записи, отличительной чертой которых выступает то, что для их создания не требуется номер телефона. Эти учетные записи в то же время служат интерфейсом для кода, реализуемого на клиентском сервере.

Промежуточный сервер Telegram сам может выполнить обработку шифрования и связи с Telegram API. Также отметим, что за счет простейшего HTTPS-интерфейса, благодаря которому существует облегченная версия Telegram API, клиентский сервер может взаимодействовать с Telegram-сервером.

В протоколах Telegram Bot API особо отмечаются два метода для проведения обновлений:

– применение периодических запросов;

– установка вебхуков.

Хранение входящих обновлений происходит до того момента, пока программа не будет обновлена, однако не дольше 24 часов. Независимо от методов обновления в результате будет получен объект Update, который отправляется в JSON-формате.

К самому несложному варианту можно отнести систематический опрос серверов Telegram на тему присутствия новых данных. Это можно выполнить посредством т.н. Long Polling, иными словами, производится открытие соединения на недолгий период времени и каждое имеющееся обновление мгновенно отправляются боту. Работа вебхуков осуществляются по-другому. Установка вебхука предполагает, что в случае, когда в чат поступает сообщение, приложение само оповещает об этом. Теперь нет необходимости, постоянно обращаться к серверу. Тем не менее, это вынуждает расплачиваться необходимостью установки полноценного web-сервера на ту машину, где будут запускать ботов. Кроме этого, для реализации нормальной работы требуется наличие собственного SSL-сертификата (Secure Sockets Layer), т. к. вебхуки в Telegram работают только по HTTPS [12]. В документации Telegram Bot API описаны все методы и передаваемые параметры, которые необходимы для работы чат-бота. Все ответы приходят в JSON-формате. Список методов для создания чат-бота:

– методика getUpdates применяется для того, чтобы получить обновления посредством Long Polling. Возврат ответа осуществляется в форме массива объектов Update;

– методика setWebhook. Требуется для того, чтобы дать задание URL web-хука, куда с бота будет происходить отправка обновления. Всегда в процессе обновления на данный адрес будет происходить отправка HTTPS POST с сериализованным в JSON объектом Update. В случае неблагоприятного запроса к серверу попытка отправки повторится неоднократно, но, согласно определенным

промежуткам, неопределенное количество раз. Для того, чтобы сделать оптимальный уровень безопасности будет целесообразным включить токен в URL web-хука, к примеру, такой, как: https://yourwebhookserver.com/<token>. Поскольку никто посторонний не имеет представление о токене, можно быть уверенным, что запросы к web-хуку отсылает именно Telegram;

– метод getWebhookInfo содержит информацию о текущем состоянии web-хука;

– метод sendMessage используется для отправки сообщений;

– метод sendPhoto используется для отправки фото;

– методика editMessageText зачастую применяется для того, чтобы отредактировать текстовые сообщения, отправка которых была сделана ботом либо посредством бота;

– объект типа User отвечает за предоставление данных о боте либо самом пользователе Telegram;

– объект типа Chat представляет собой информацию о чате;

– объект типа Message представляет собой информацию о сообщении;

– объект типа KeyboardButton – это, по факту, одна кнопка на клавиатуре ответа. Для типичных и несложных текстовых кнопок данный объект может заменяться на строку, которая содержит текст на кнопке;

– объект типа InlineKeyboardMarkup – это уже установленная клавиатура, всплывающая при написании сообщения;

– объект типа InlineKeyboardButton – это одна кнопка на клавиатуре ответа. Обязательное условие заключается в том, чтобы подключить ровно одно опциональное поле.

Разнообразие методов создания чат-ботов способствуют появлению чат-ботов с разными функциями.

2.3 Концептуальная модель чат-бота

Разрабатываемый чат-бот состоит из нескольких частей, которые при взаимодействии друг с другом обеспечивают корректную работу всего приложения: сама программа, которая выполняет все заявленные требования; база данных, являющаяся хранилищем необходимой информации; платформа, на которой будет работать программа (бот), и сервер, на который будет выложен код программы и откуда будет осуществляться ее запуск. Модель проектируемого чат-бота показана на рисунке 1.

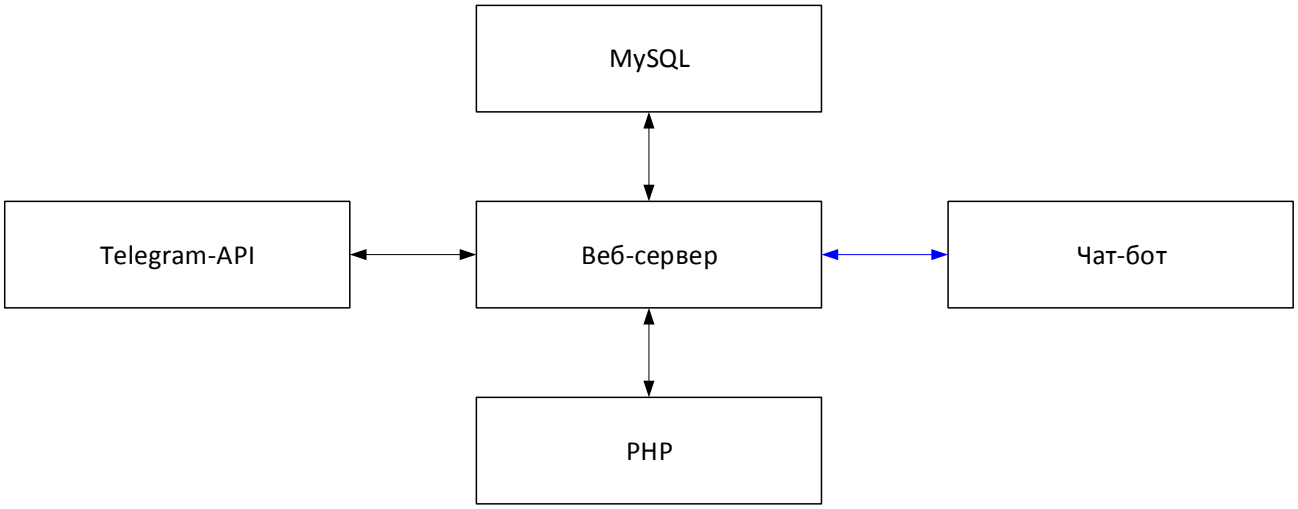


Рисунок 1 – Модель разрабатываемого проекта чат-бота

С концептуальной точки зрения работа приложения может быть описана следующим образом: когда пользователь взаимодействует с чат-ботом в Telegram, API-интерфейс осуществляет отправку данных о взаимодействии в код по HTTP-запросу, в результате чего код также делает отправку сведений, которыми обозначает, каким образом нужно реагировать. В результате Bot API можно представить в качестве посредника, осуществляющего взаимодействие бота в

Telegram с логикой приложения. Bot API включает следующие компоненты: обновления и методы. К разработчику поступают обновления, которые показывают, как пользователь взаимодействовал с ботом. При этом методы вызова требуются с той целью, чтобы бот мог исполнять ряд действий, среди которых отправление сообщений пользователям.

Основной ряд требований, который предъявляется по отношению к чат-боту:

– диалог пользователя и виртуального собеседника проводится только на естественном языке;

– необходимо, чтобы у виртуального собеседника в обязательном порядке был доступ к базе данных для определения эмоционального состояния собеседника;

– при окончании диалога пользователя с виртуальным собеседником, требуется установить психоэмоциональное состояние и предоставить надлежащие (позитивные/нейтральные) ответы;

– необходимо, чтобы беседа проводилась максимально легко и естественно;

– необходимо, чтобы у пользователя была возможность осуществлять выбор режима для ведения диалога, времени, длительности и объема запросов. Если возникнет потребность в прекращении диалога, то это можно легко выполнить. Запись о прекращенных диалогах вносится в базу данных с соответствующей пометкой;

– необходимо, чтобы виртуальный собеседник умел принимать решения, будучи в состоянии неопределенности из-за нехватки сведений. Притом пользователь может получить дополнительные вопросы;

– необходимо, чтобы система имела базу данных собеседников и пользователей, которая будет автоматическим образом пополняться виртуальным собеседником, опираясь на информацию, предоставленную ему клиентом в ходе

беседы. В базе данных, наряду с их идентификационной информацией и историей запросов, должна также сохраняться дополнительная информация (возраст, класс, проблемные направления). В дальнейшем данная информация может быть использована как для персонификации диалога, так и в процессе определения проблематики при выдаче рекомендаций или формирования соответствующих выводов. По составленной базе данных программа должна выдавать сведения как об отдельно выбранном собеседнике, так и статистические данные по всем собеседникам, которые могут быть использованы при принятии решений для направлений социально-психологических мероприятий;

– база знаний интеллектуального собеседника должна пополняться в ручном или полуавтоматическом режиме, т.е. программа должна иметь способность обучения и переобучения;

– у виртуального собеседника должна быть визуальная оболочка, позволяющая передавать также эмоциональную составляющую диалога.

2.4 Логическое моделирование чат-бота

Для того, чтобы выполнить логическое моделирование системы, применялась диаграмма вариантов пользования, выступающая исходным концептуальным представлением системы в ходе ее создания. Эта диаграмма включает в себя актеров, варианты применения и их взаимоотношения. При построении диаграммы могут использоваться также общие элементы нотации: примечания и механизмы расширения.

Суть данной диаграммы состоит в следующем: проектируемая система представляется в виде множества актеров, взаимодействующих с системой с помощью так называемых вариантов использования. При этом актером (действующим лицом, актантом, актером) называется любой объект, субъект или система, взаимодействующая с моделируемой системой извне. В свою очередь

вариант использования – это спецификация сервисов (функций), которые система предоставляет актеру. Иначе говоря, каждый вариант применения может определять определенный набор действий, которые выполняются системой во время общения с актером. Притом в модели никоим образом нельзя увидеть то, как будет выполнен данный набор действий. В данной модели показаны актеры с различными вариантами взаимодействия (рисунок 2).

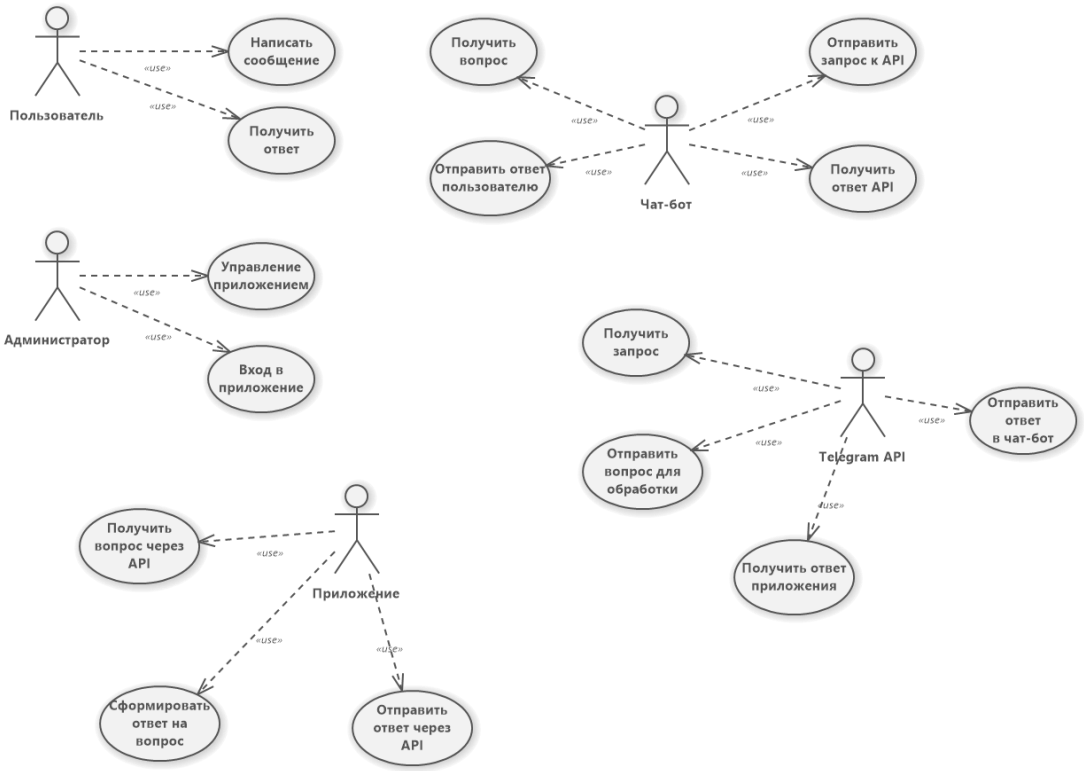


Рисунок 2 – Диаграмма вариантов использования чат-бота

Диаграмма последовательности (рисунок 3) используется для уточнения диаграмм прецедентов, т.е. для более детального описания логики сценариев использования. Диаграммы последовательностей обычно содержат объекты, которые взаимодействуют в рамках сценария, сообщения, которыми они обмениваются, и возвращаемые результаты, связанные с сообщениями. Впрочем,

часто возвращаемые результаты обозначают лишь в том случае, если это не очевидно из контекста.

Объекты обозначаются прямоугольниками с подчеркнутыми именами (чтобы отличить их от классов). Сообщения (вызовы методов) – линиями со стрелками. Возвращаемые результаты – пунктирными линиями со стрелками. Прямоугольники на вертикальных линиях под каждым из объектов показывают “время жизни” (фокус) объектов.

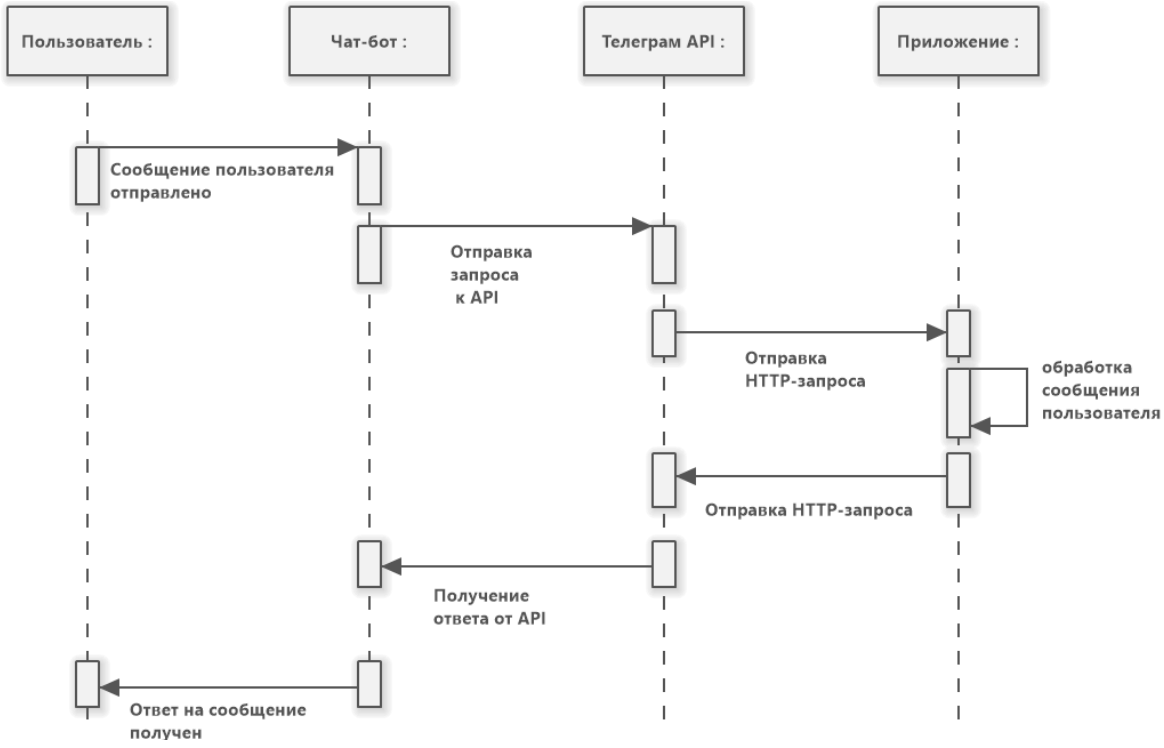


Рисунок 3 – Диаграмма последовательности, описывающая ответ чат-бота на входящее сообщение пользователя

Диаграмма классов показана на рисунке 4.

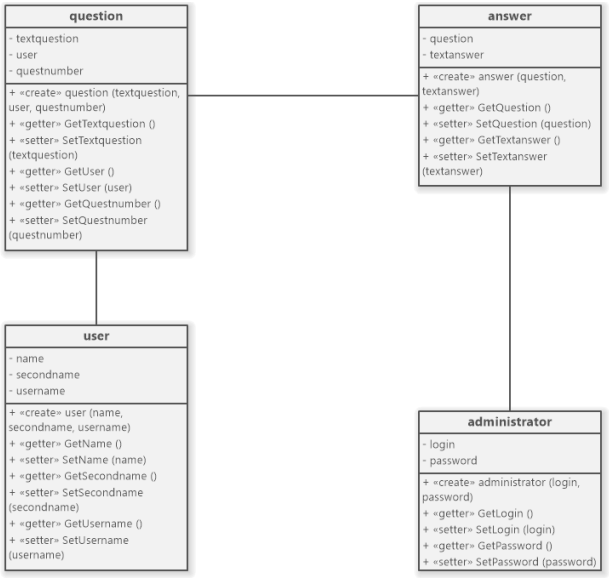


Рисунок 4 – Диаграмма классов

Диаграмма классов занимает важное место в проектировании объектно-ориентированной системы. Система классов используется на разных этапах проектирования. Данная система может быть в разной степени детализирована.

Выводы по главе 2

В результате выполнения второй главы выбраны технологии и сервис для реализации чат-бота, рассмотрены концептуальные аспекты функционирования системы, описана логическая модель работы проектируемого чат-бота. В качестве сервиса для чат-бота выбран Telegram. Telegram-боты – разновидность чат-ботов. Их суть заключается в реакции на определенные сообщения от пользователей. При регистрации бота выдается уникальный ключ, с помощью которого в дальнейшем и будет происходить связь между клиентом и сервером. Разрабатываемый чат-бот состоит из нескольких частей, которые при взаимодействии друг с другом обеспечивают корректную работу всего приложения.

Глава 3 Физическое проектирование чат-бота. Оценка и обоснование экономической эффективности проекта

3.1 Физическое моделирования чат-бота

Для реализации чат-бота требуется наличие протокола HTTPS, который является обязательным требованием поставщика сервиса для чат-бота Telegram API. Без этого протокола чат-бот не будет работать. Для этих целей использован сервис Cloudflare, который предоставляет эмулированный HTTPS для домена, к которому будет обращаться чат-бот. Вид панели Cloudflare показан на рисунке 5.

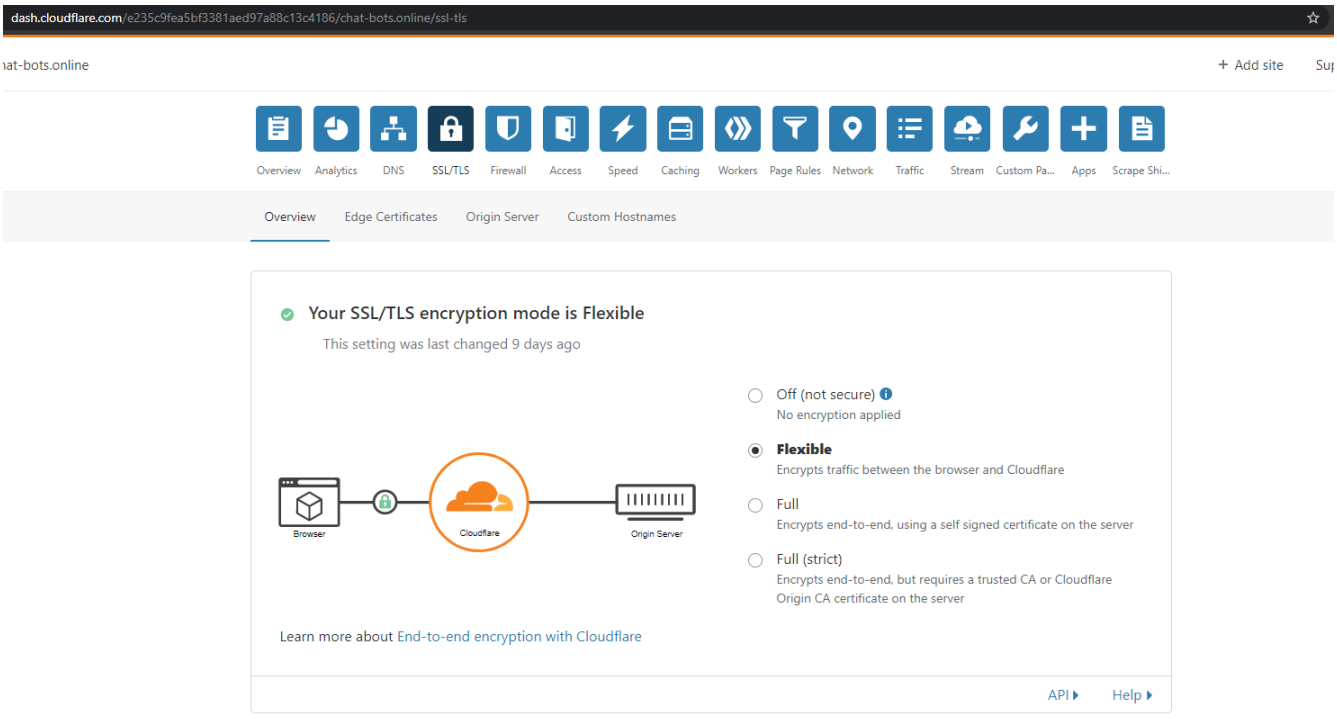


Рисунок 5 – Окно с настройками Cloudflare

Для размещения приложения использован внешний хостинг, процесс работы с которым показан на рисунках 6 – 9. В качестве хостинга выбран сервис Hostinger.ru, который предоставляет удобные инструменты управления доменами и проектами.

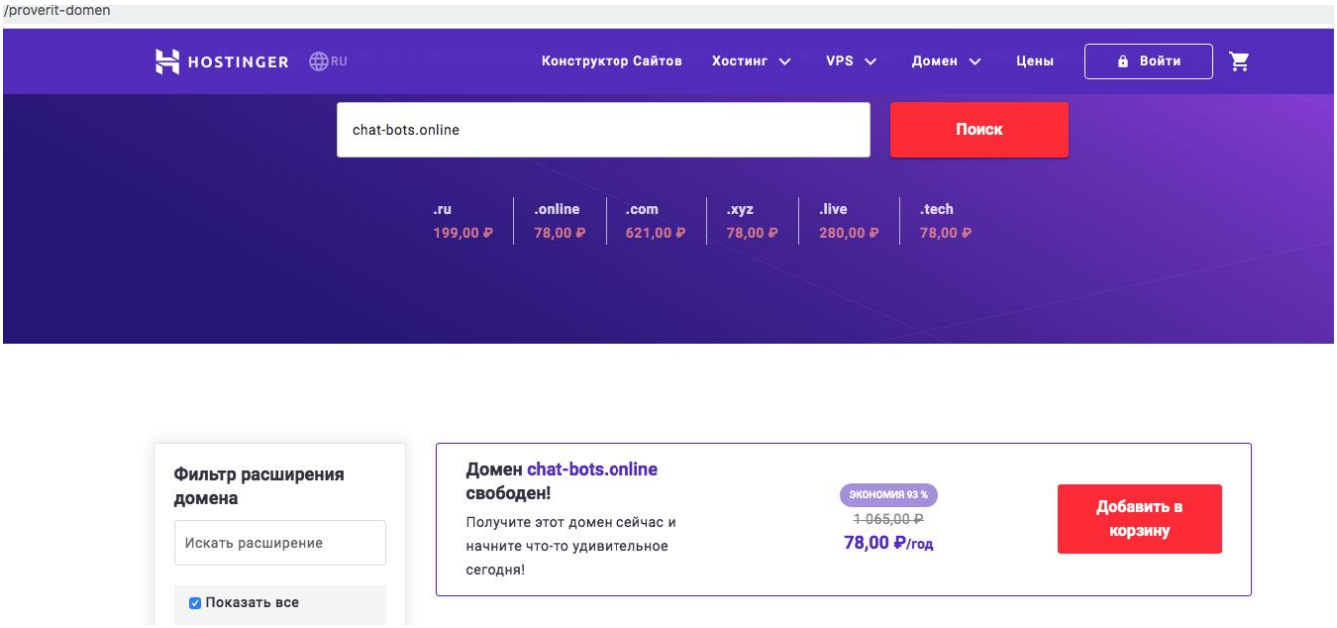


Рисунок 6 – Подбор домена для чат-бота

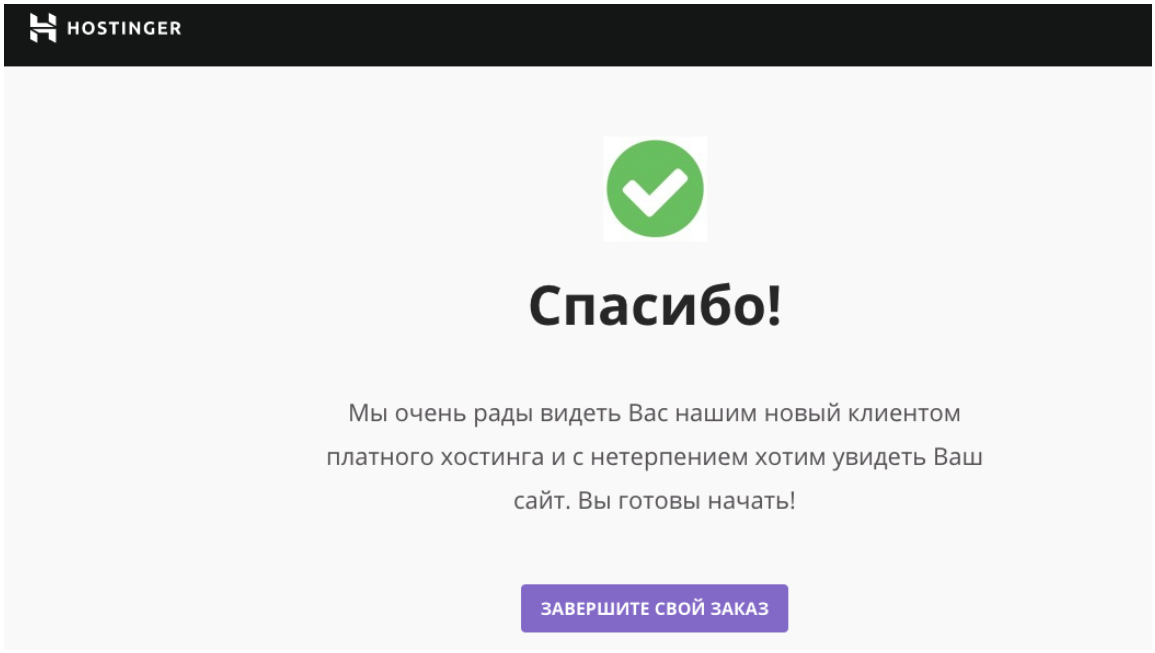


Рисунок 7 – Заказ хостинга для чат-бота

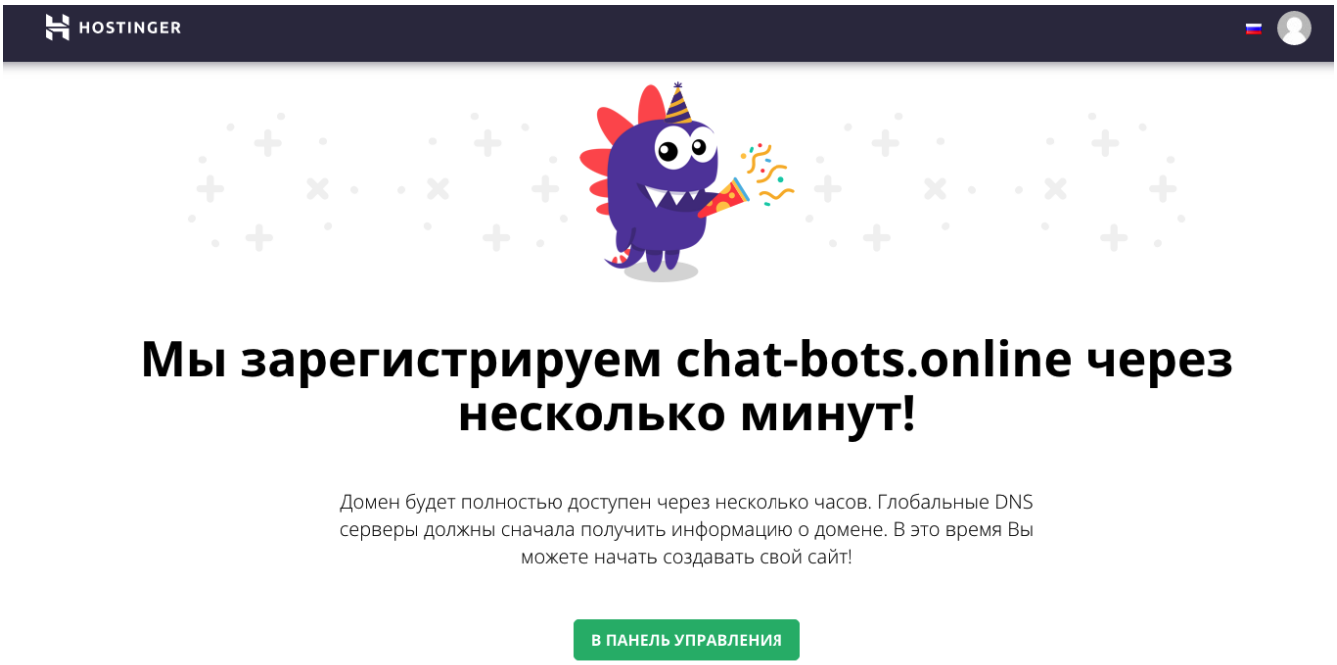


Рисунок 8 – Регистрация домена для чат-бота

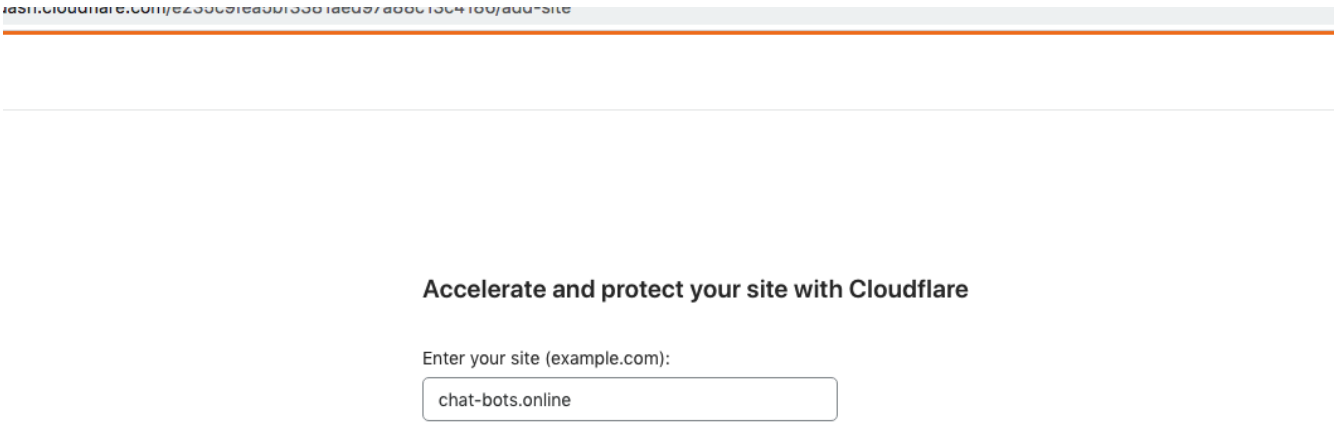


Рисунок 9 – Проверка домена на доступность регистрации

После завершения процесса регистрации, как было указано выше проведена настройка домена (рисунок 10) и привязка его к сервису для подключения защищенного протокола HTTPS.

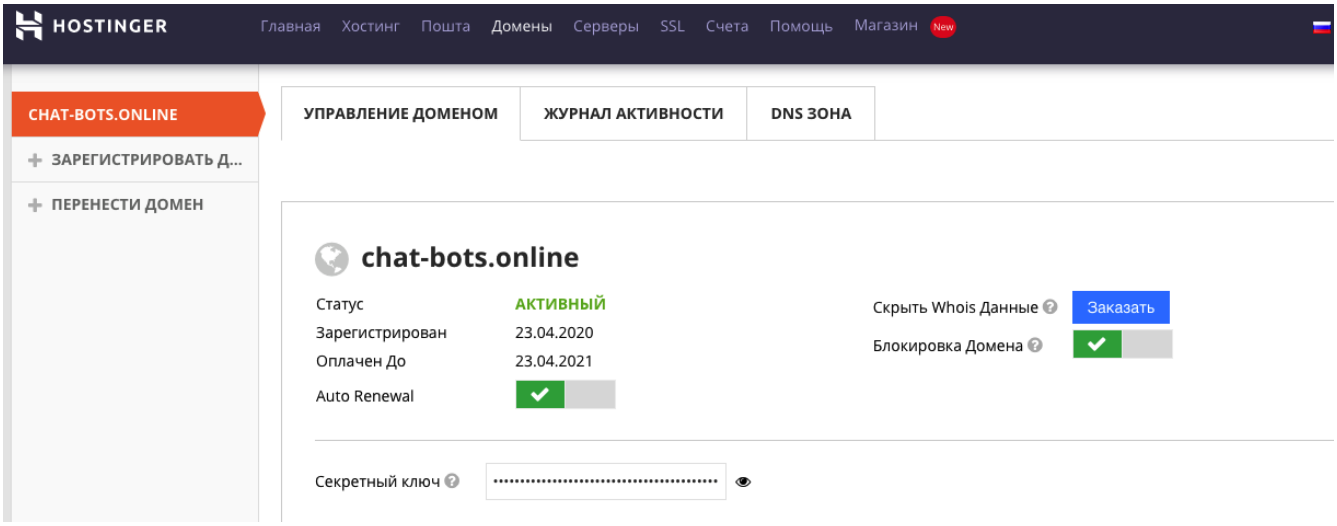


Рисунок 10 – Панель управления доменом

После регистрации домена необходимо добавить сайт, который будет обеспечивать функционал чат-бота (рисунок 11).

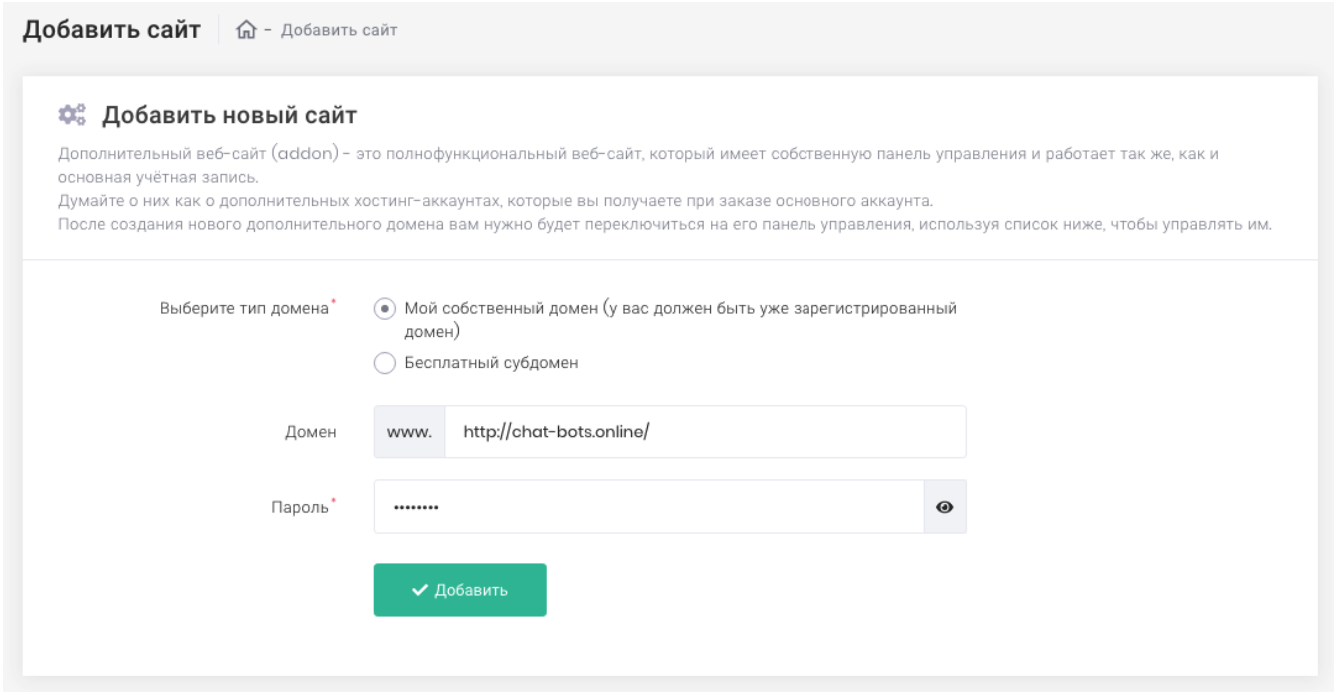


Рисунок 11 – Управление сайтами в личном кабинете пользователя

Рисунки 12 – 15 отображают процесс настройки NS-серверов.

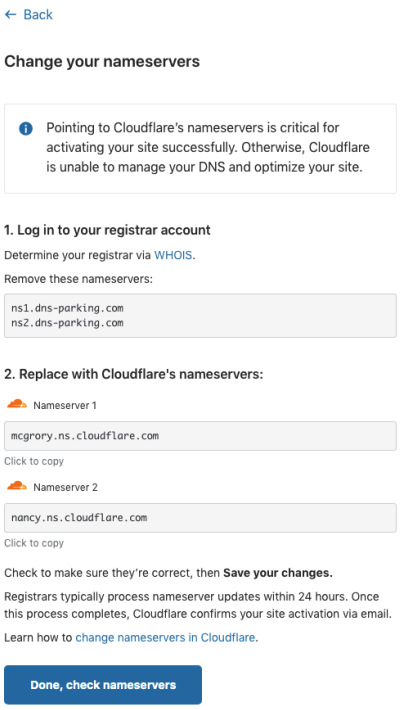


Рисунок 12 – Привязка NS-серверов хостинга к сервису Cloudflare

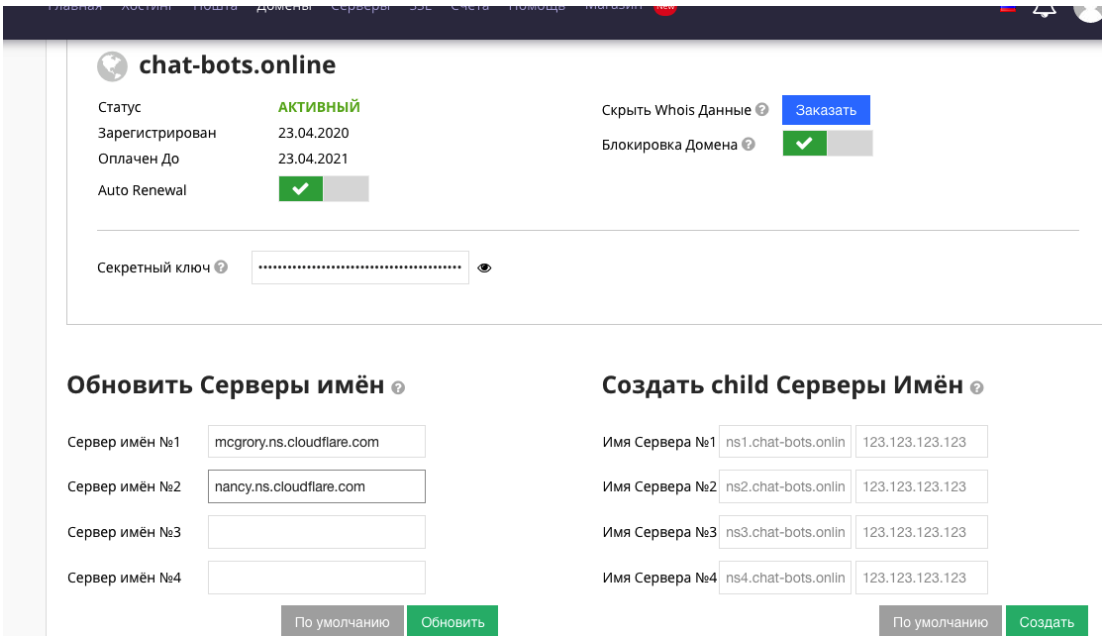


Рисунок 13 – Изменение параметров NS-серверов

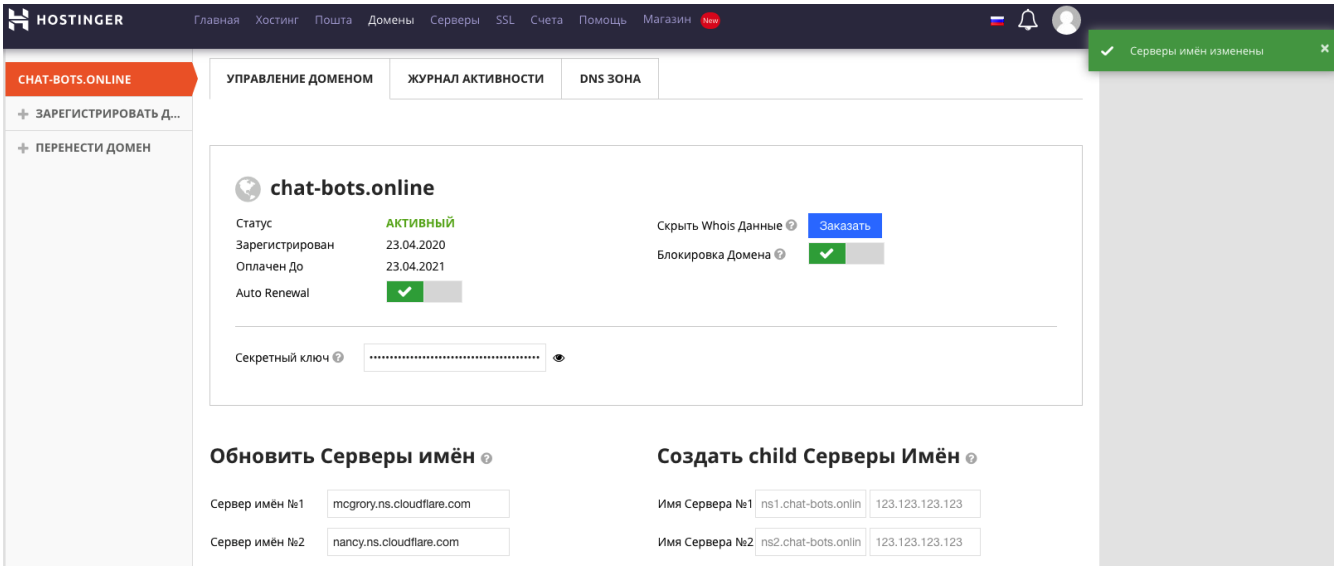


Рисунок 14 – Панель управления доменов

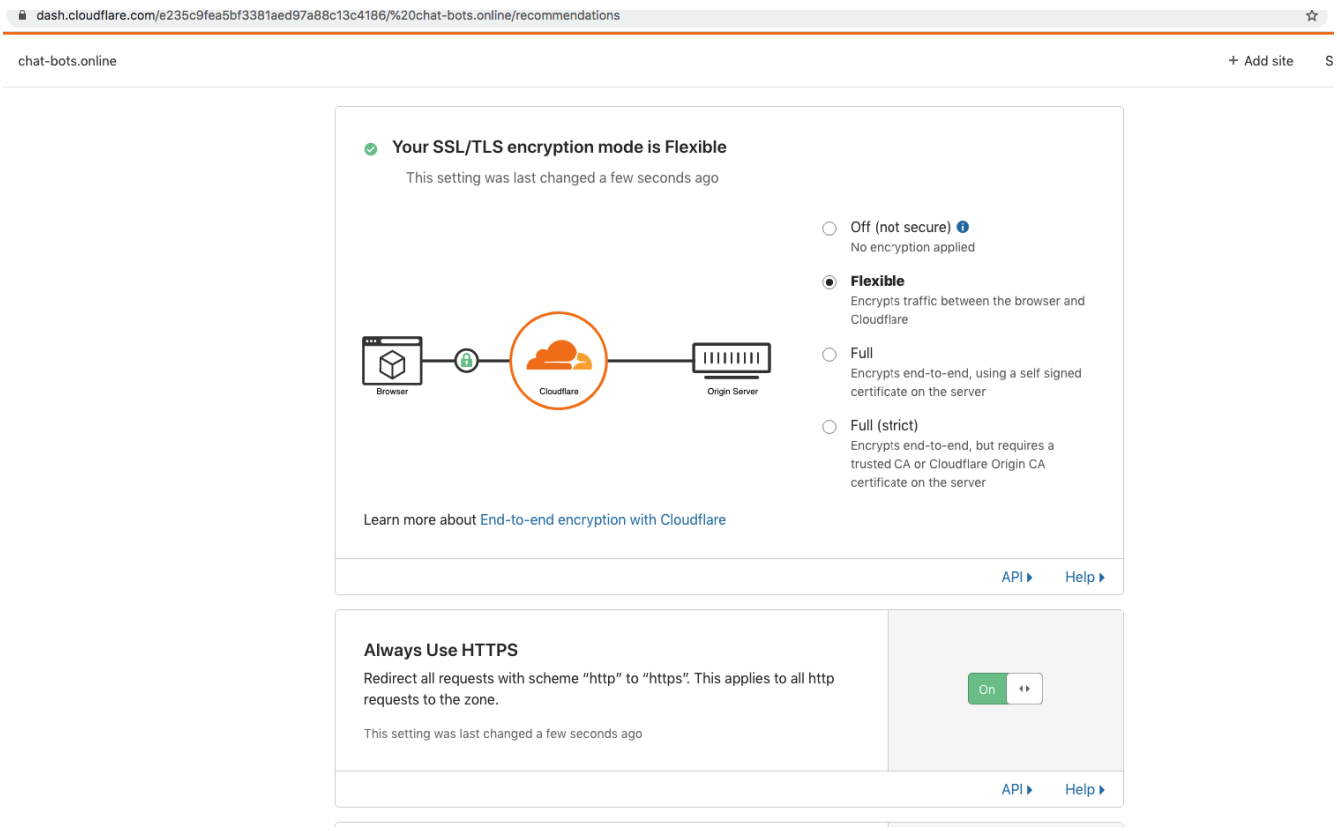


Рисунок 15 – Активация HTTPS в кабинете Cloudflare

После настройки доступа к хостингу с использованием HTTPS-протокола необходимо зарегистрировать новый бот в Botfather (рисунок 16), который предоставляет элементы управления и создает чат-боты, которые работают в пределах мессенджера Telegram (рисунки 17 –18).

Рисунок 16 – Чат-бот Botfather

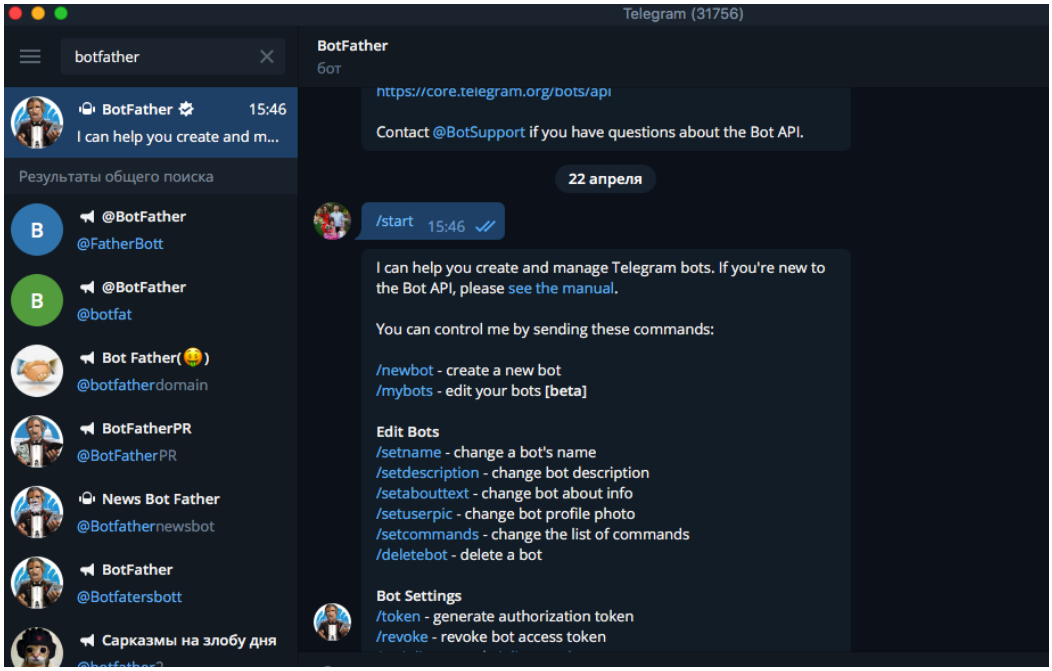


Рисунок 17 – Настройка чат-бота

Командой для создания чат-бота является – /newbot. После ее выполнения Botfather выводит рекомендации, как работать с ними.

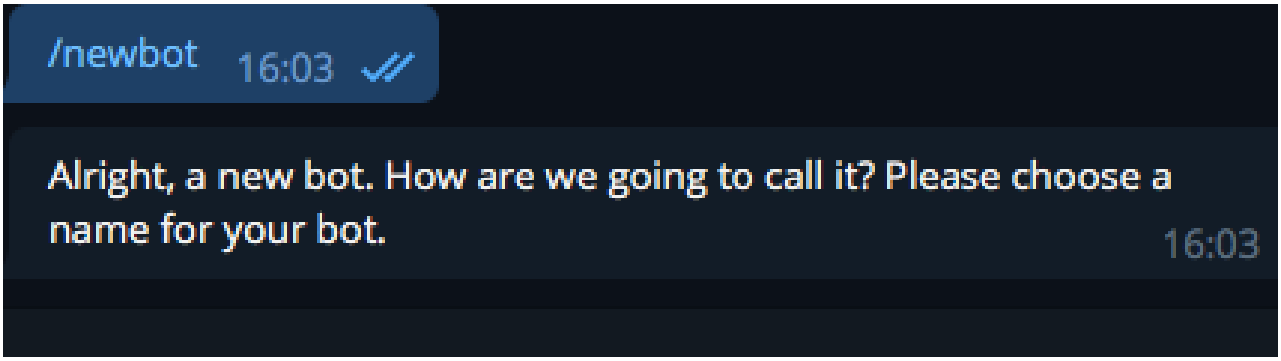


Рисунок 18 – Создание чат-бота командой /newsbot

Создание чат-бота с названием «Emo-bot» (рисунок 19).

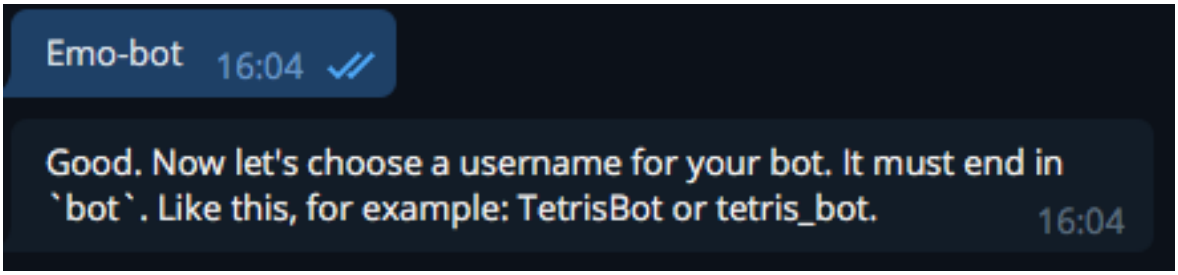


Рисунок 19 – Созданный чат-бот

Следующим этапом является создание токена и подключение API-телеграмма для работы чат-бота (рисунок 20).

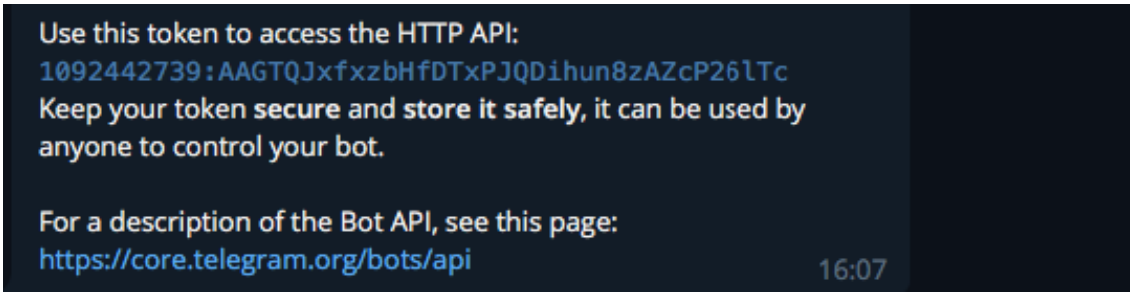


Рисунок 20 – Создание токена для работы с API

Процесс настройки чат-бота показан на рисунках 21 – 22.

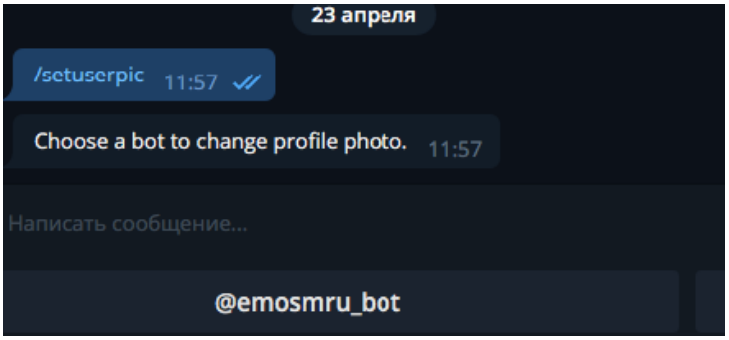


Рисунок 21 – Установка изображения для чат-бота

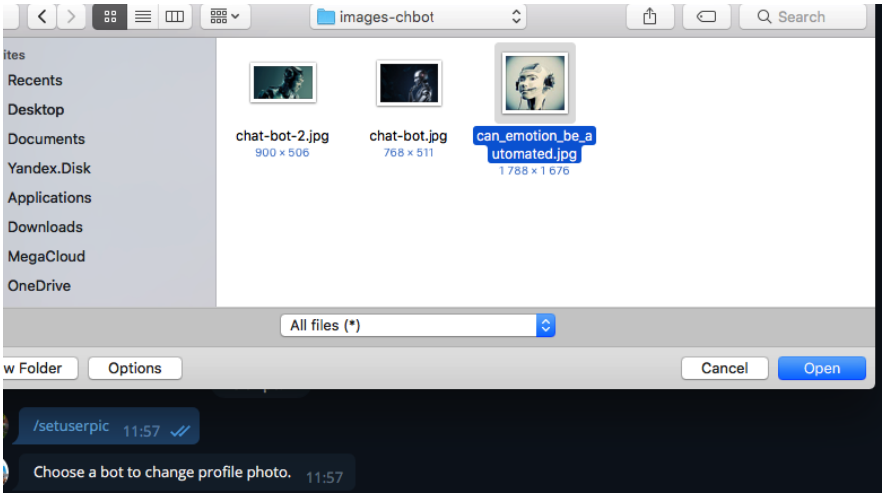


Рисунок 22 – Выбор изображения для установки в профиль чат-бота

После установки изображения чат-бота (рисунок 23) необходимо настроить параметры безопасности чат-бота, которые показаны на рисунке 24.

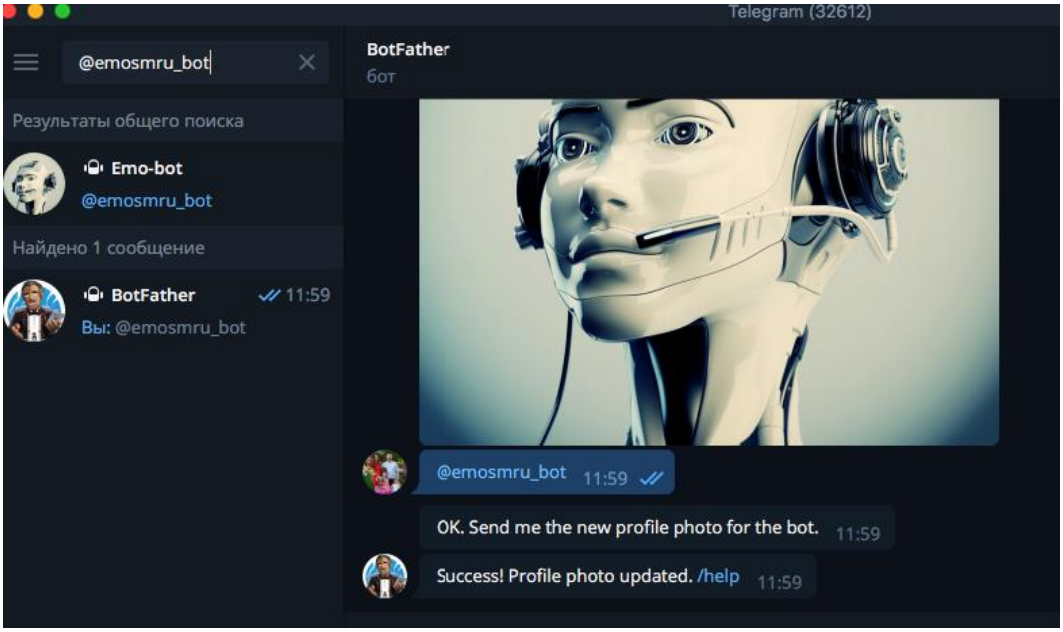


Рисунок 23 – Профиль чат-бота с изображением

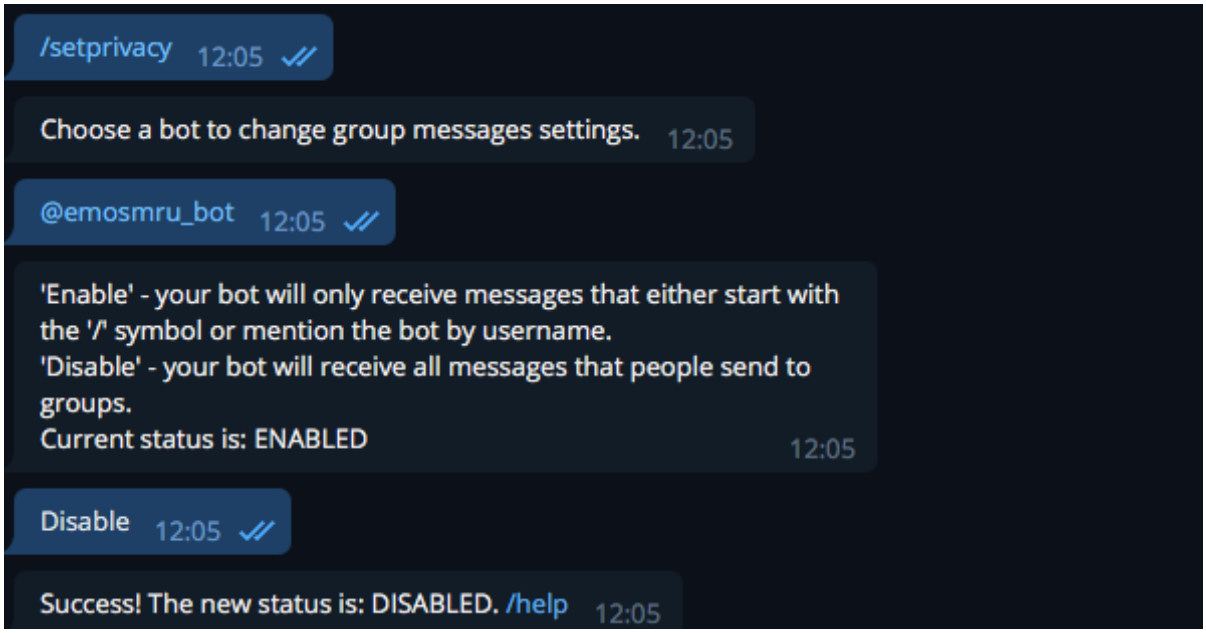


Рисунок 24 – Установка параметров безопасности чат-бота

Следующим этапом было настройка ftp-доступа для размещения файлов на сервере. Процесс показан на рисунке 25.

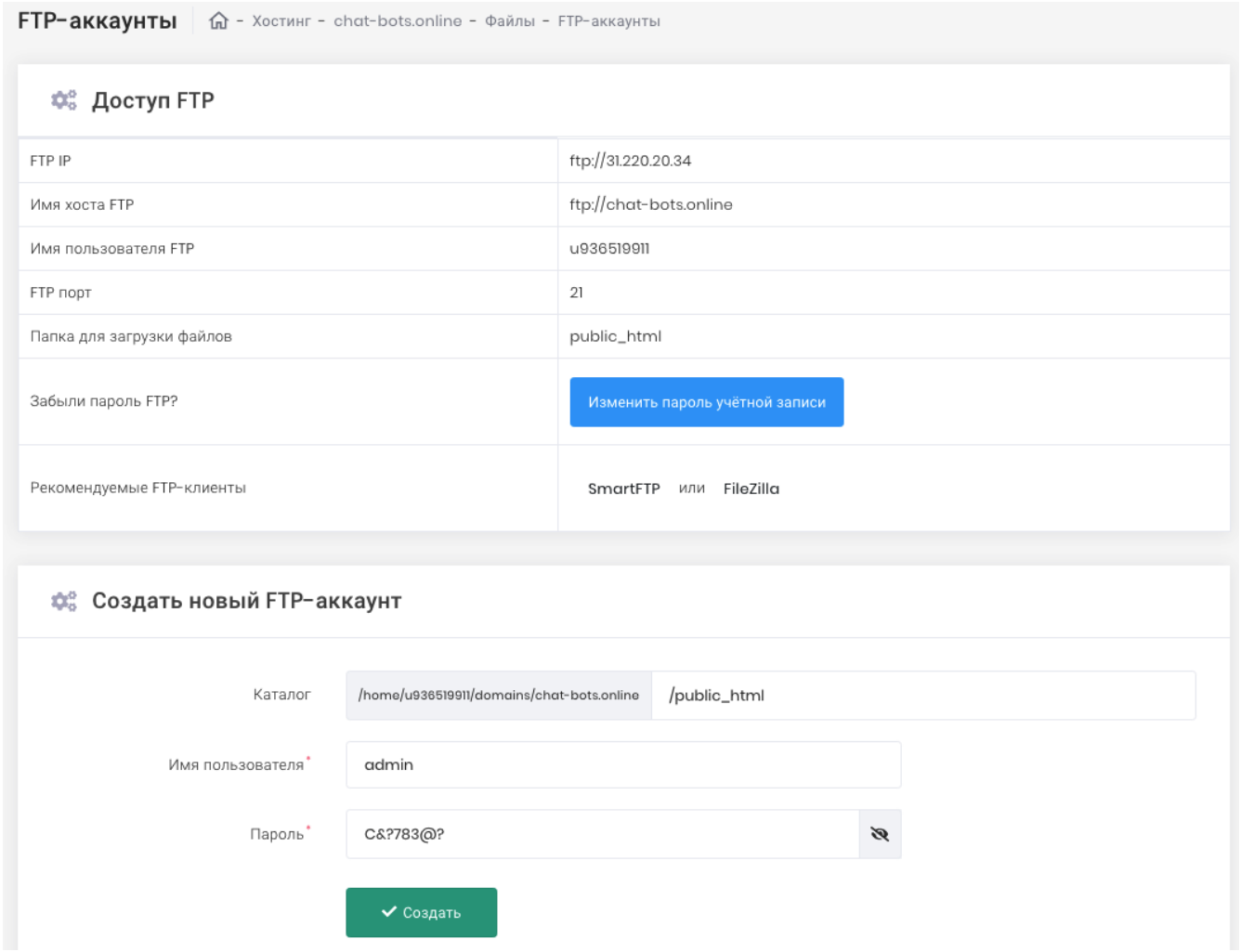


Рисунок 25 – Настройка ftp-доступа

Для доступа на ftp-сервер используются параметры из таблицы 3, аккаунт для авторизации: логин – admin, пароль – 12345.

Таблица 3 – Параметры ftp

FTP IP	ftp://31.220.20.34
Имя хоста FTP	ftp://chat-bots.online

Продолжение таблицы 3

Имя пользователя FTP	u936519911
FTP порт	21
Папка для загрузки файлов	public_html

Следующим этапом было создание структуры проекта и реализация его функционала (рисунок 26).

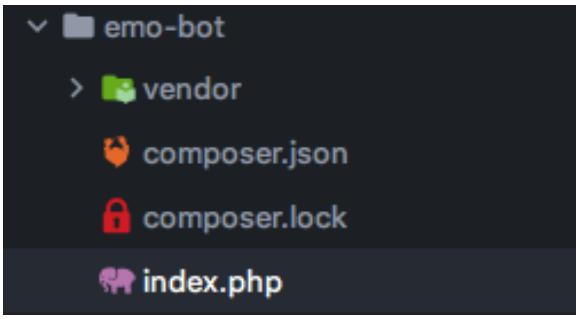


Рисунок 26 – Создание файлов проекта

Создание базы данных показано на рисунке 27.

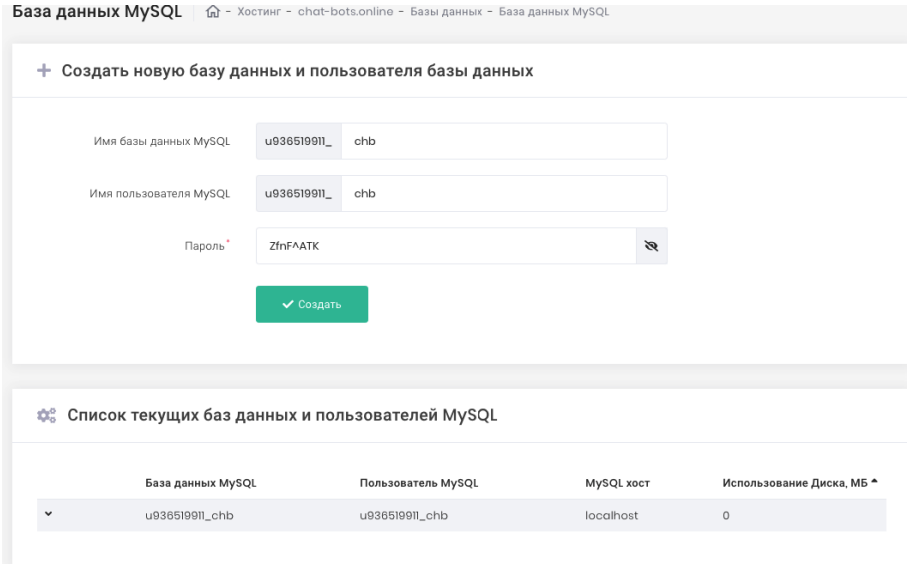


Рисунок 27 – Создание базы данных для проекта

Доступ к созданной базе данных осуществляется с помощью учётных данных: пароль – ZfnFATK, логин – u936519911_adm. Подключение среды

разработки PHPStorm к базе данных на сервере показано на рисунке 28. После этого были созданы файлы реализации функционала чат-бота (рисунки 29 – 31).

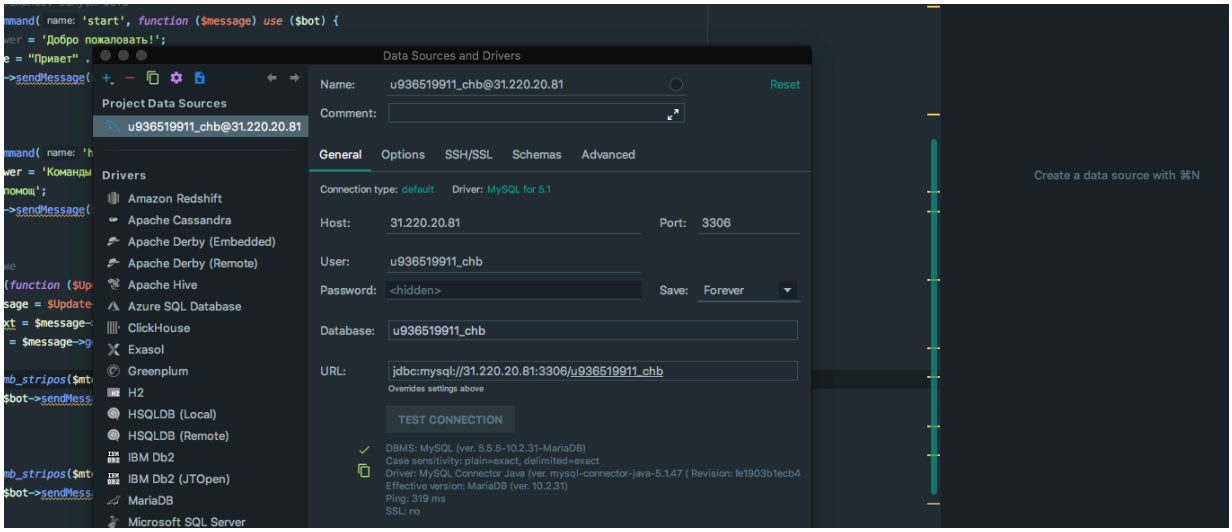


Рисунок 28 – Подключение базы данных к среде разработки

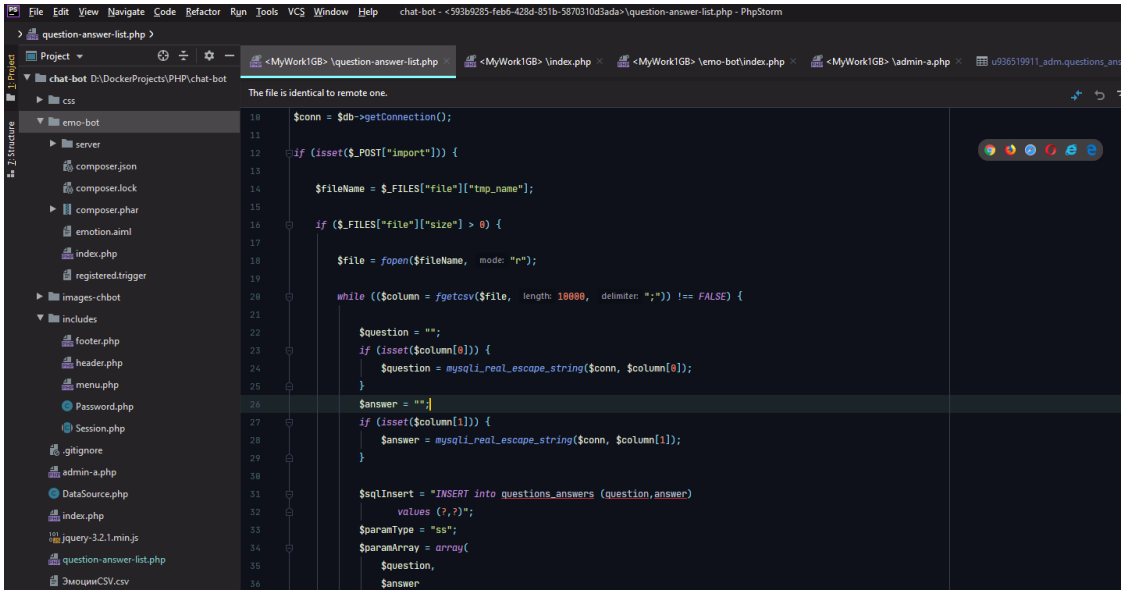


Рисунок 29 – Реализация списка данных и импорта файла

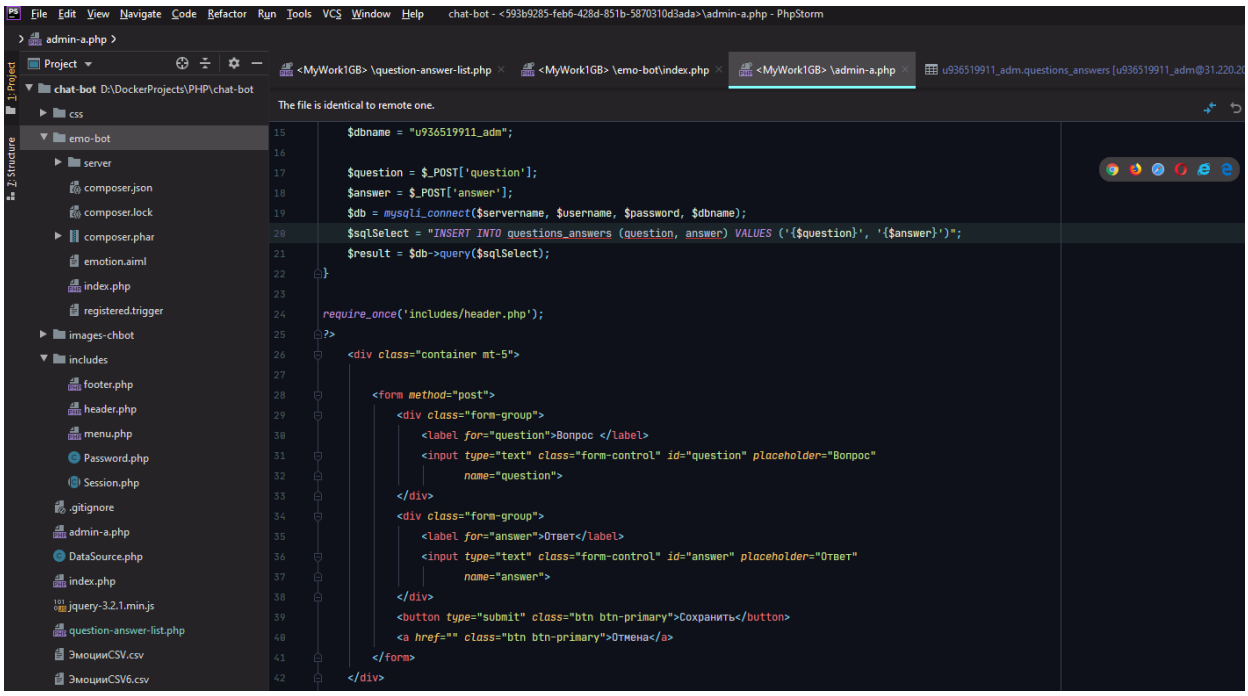


Рисунок 30 – Форма добавления данных

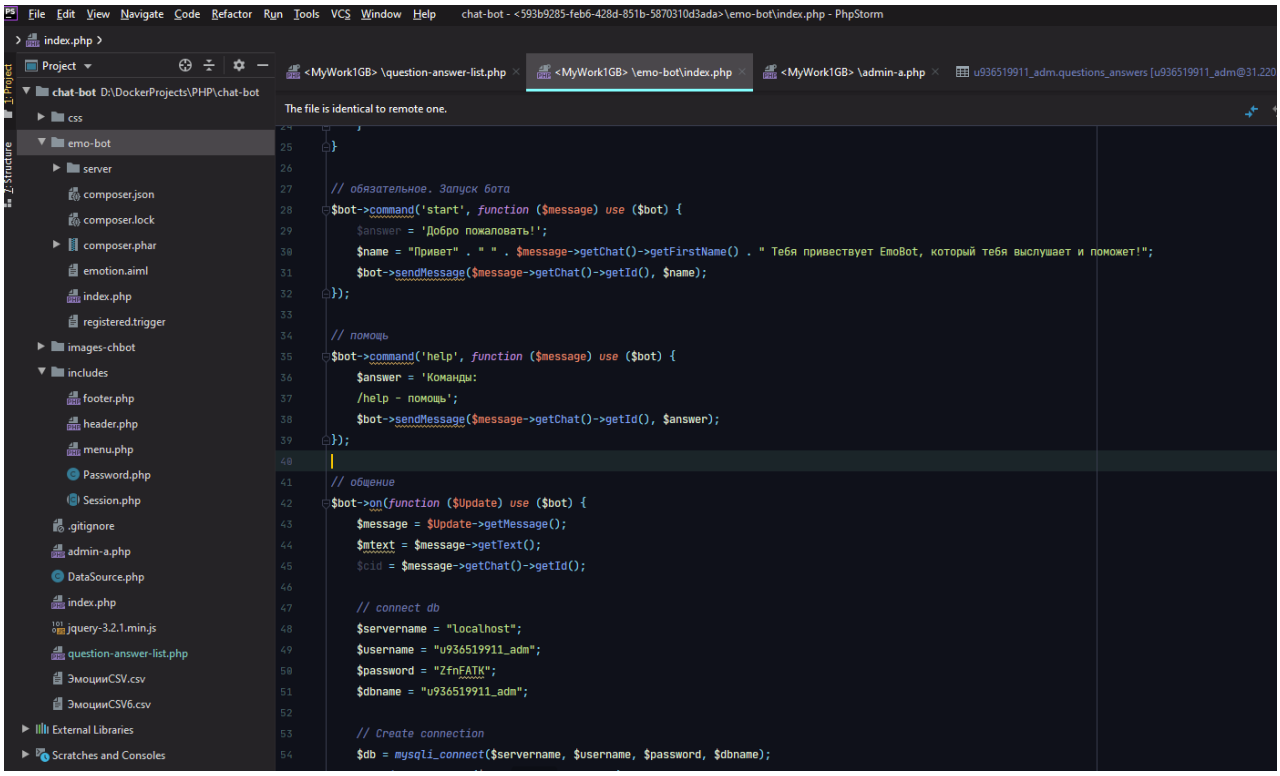


Рисунок 31 – Реализация чат-бота

3.2 Руководство пользователя

1. Для запуска чат-бота нужно перейти в приложение Telegram и в поиске ввести Emo-bot (рисунок 32).

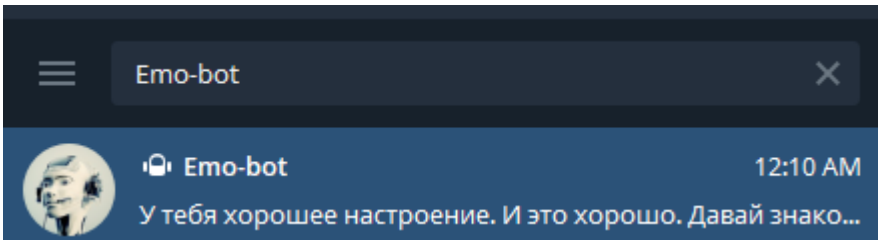


Рисунок 32 – Поиск чат-бота

2. Для запуска чат-бота нужно ввести команду /start (рисунок 33).

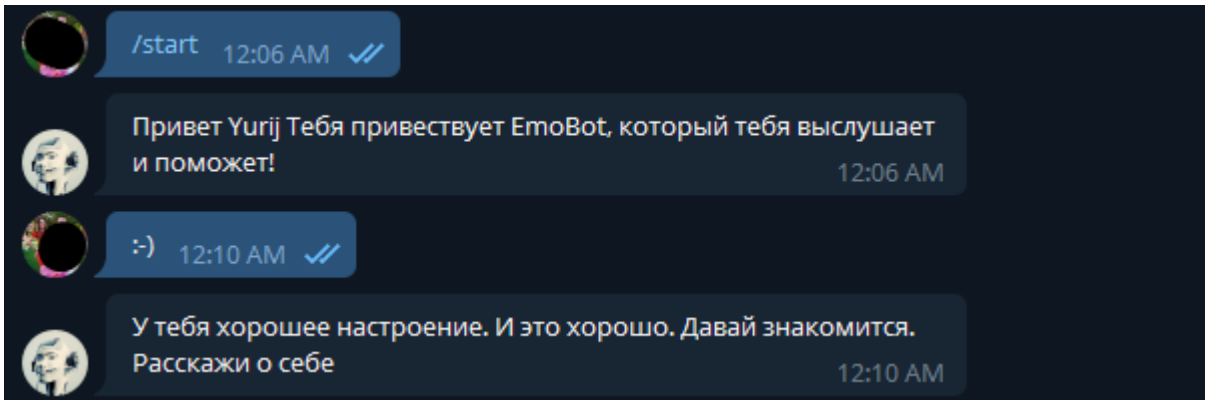


Рисунок 33 – Запуск чат-бота. Реакция по положительные эмоции

3. Чат-бот реагирует на текстовые сообщения и смайлы. После отправки сообщений чат-бот возвращает ответы, которые учитывают эмоциональный окрас сообщений, отправленные пользователем (рисунки 34 – 37).

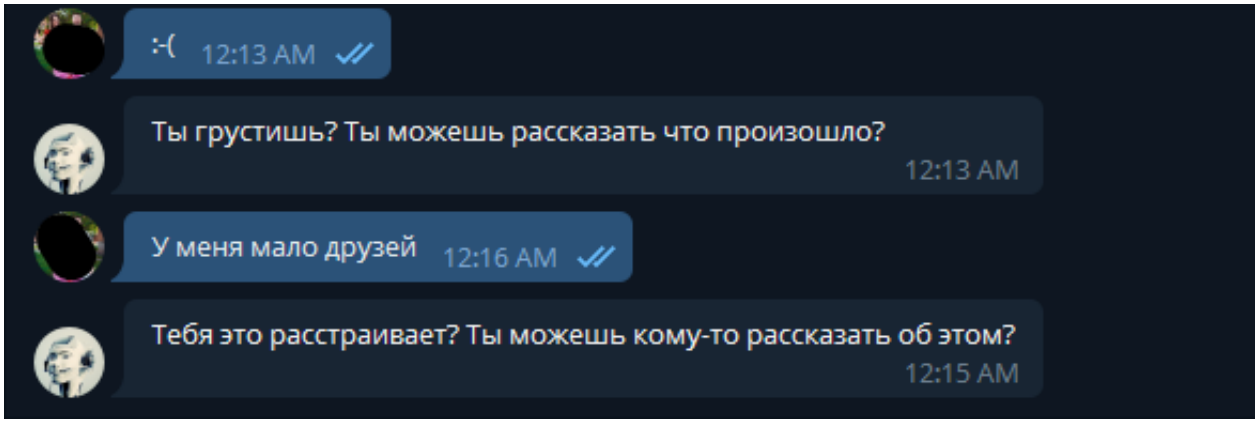


Рисунок 34 – Реакция на эмоции грусти

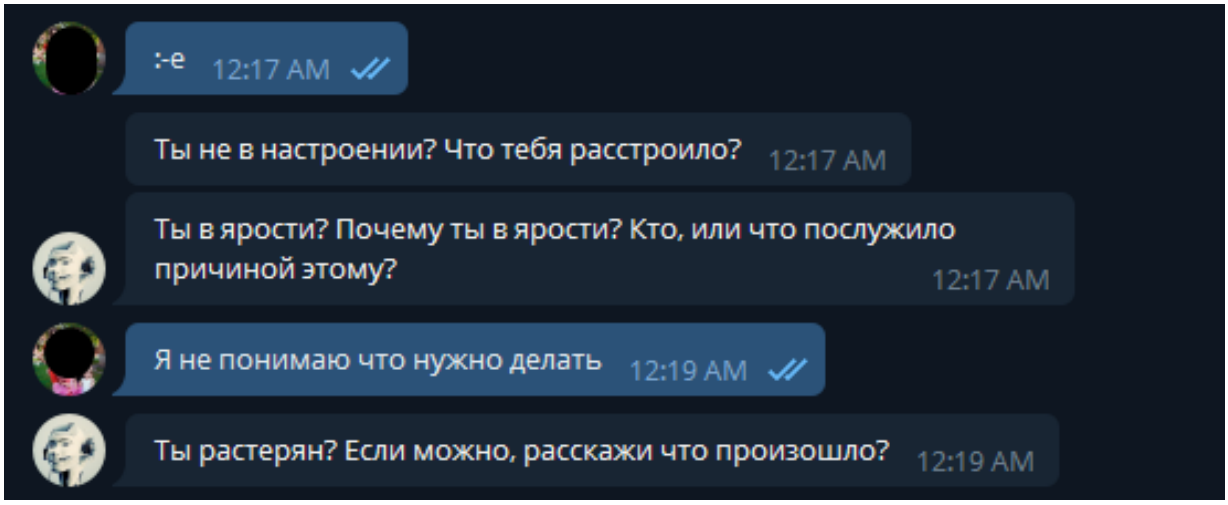


Рисунок 35 – Реакция на эмоции плохого настроения

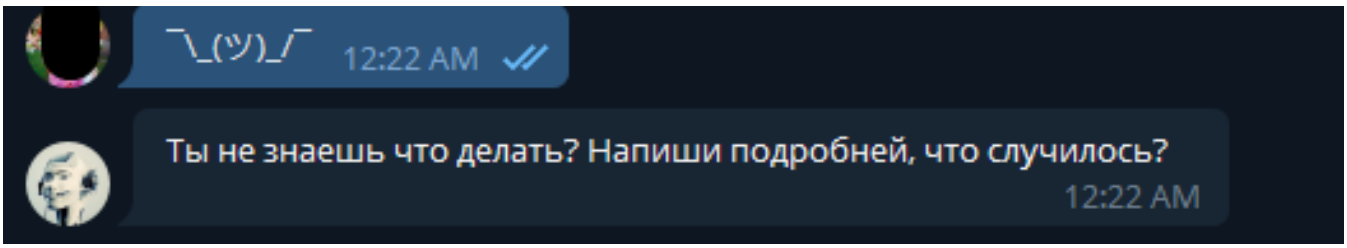


Рисунок 36 – Реакция на эмоции неопределенности

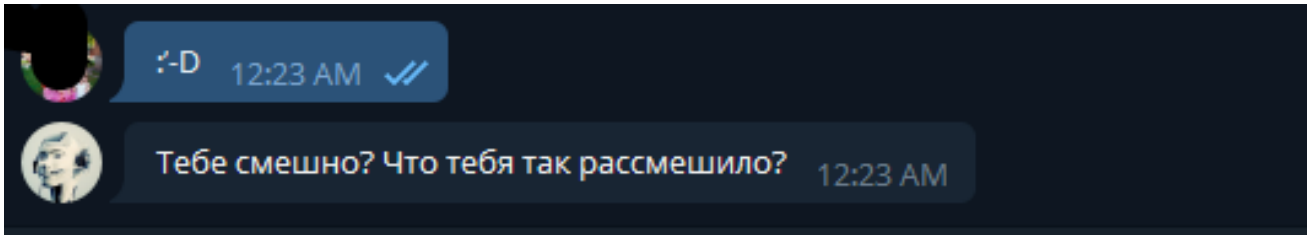
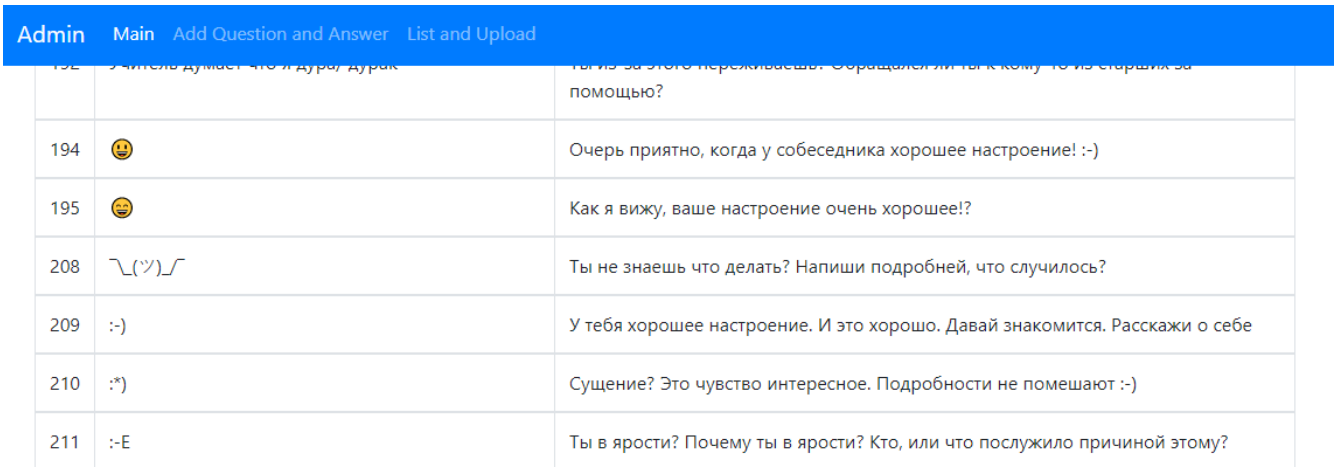


Рисунок 37 – Реакция на эмоции веселья

4. Базу вопросов чат-бота можно пополнять за счет не сложной административной панели, в которую можно вносить варианты диалогов пользователей в форме «вопрос-ответ» (рисунок 38), а также путем импорта файла со списком элементов возможных диалогов между пользователями и чат-ботом.



		помощью?
194	😃	Очерь приятно, когда у собеседника хорошее настроение! :-)
195	😄	Как я вижу, ваше настроение очень хорошее!?
208	¯_(ツ)_/¯	Ты не знаешь что делать? Напиши подробней, что случилось?
209	:-)	У тебя хорошее настроение. И это хорошо. Давай знакомится. Расскажи о себе
210	:*)	Сущение? Это чувство интересное. Подробности не помешают :-)
211	:-E	Ты в ярости? Почему ты в ярости? Кто, или что послужило причиной этому?

Рисунок 38 – База вопросов-ответов для чат-бота

5. Для добавления комбинации вопроса-ответа нужно перейти в пункт меню «Add Question and Answer», после этого откроется форма для ввода данных (рисунок 39).

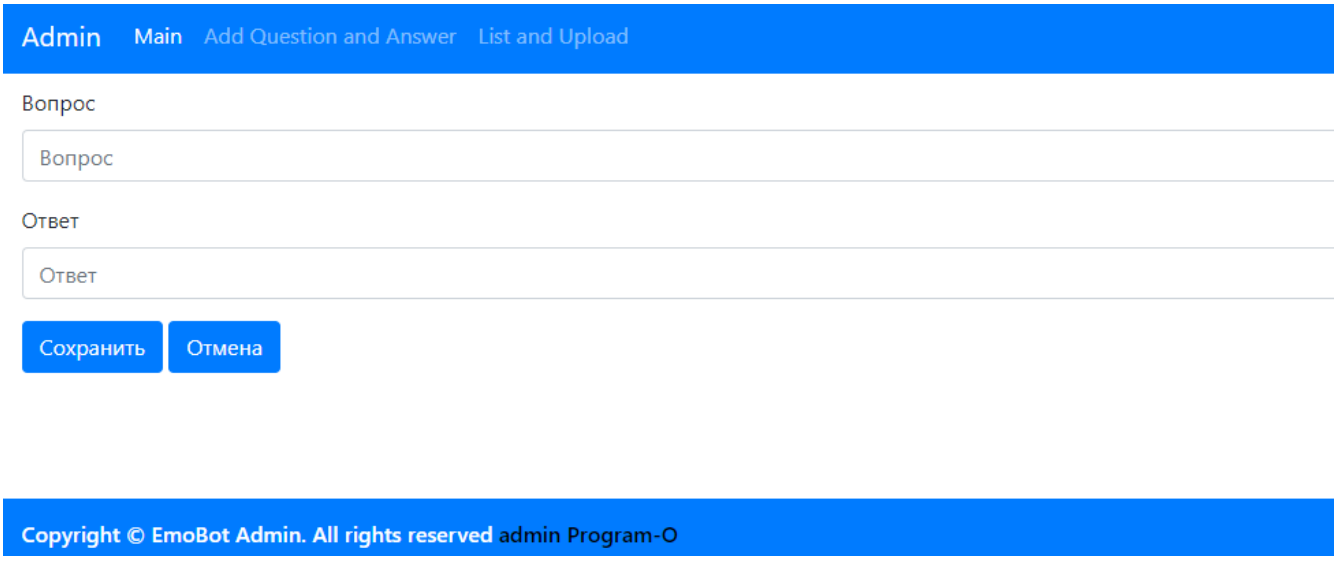


Рисунок 39 – Добавление данных для чат-бота

6. Добавление данных в базу чат-бота возможно с помощью файла импорта, в котором находятся строки текста (рисунок 40).

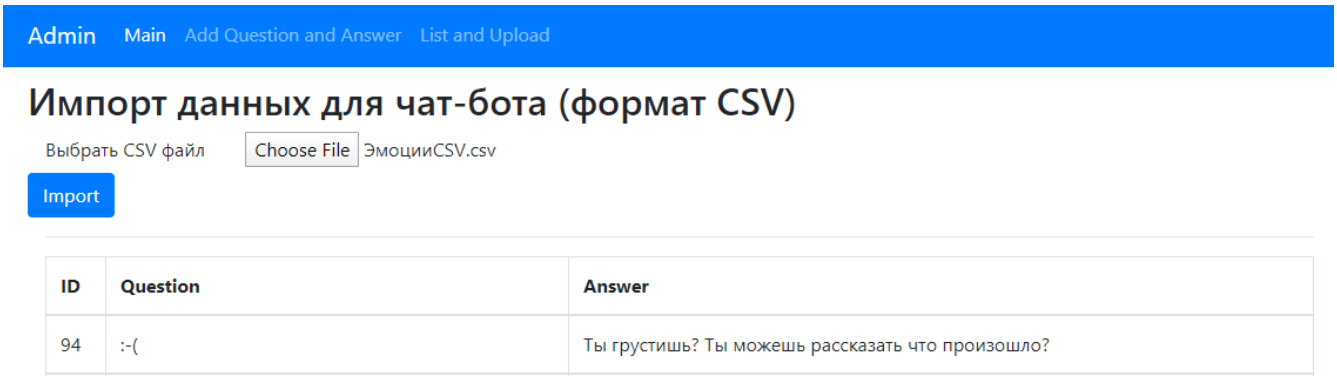


Рисунок 40 – Импорт данных с файла

После выбора и добавления данных они записываются в базу данных и чат-бот может их использовать.

3.3 Экономическое обоснование создания чат-бота

Программное изделие разработано с помощью среды разработки PHPStorm, в качестве хранилища данных будет использована СУБД MySQL. Затраты труда на разработку программного изделия принимаются в соответствии с исходными данными таблицы 4.

Таблица 4 – Этапы проектирования и разработки программного продукта

Этап, t_i	№ работы	Содержание работы	Продолжительность работ (дни)
Планирование	1	Изучение научно-технической литературы. Определение среды разработки и языка программирования	3
	2	Предварительный выбор методов проектирования	2
	3	Определение стадий, этапов и сроков разработки и документации на неё	1
Технический проект	1	Проектирование базы данных и структуры чат-бота	6
Рабочий проект	1	Программирование базы данных и модулей приложения, подключение и отладка	13
	2	Испытание чат-бота. Тестирование и корректировка по результатам испытаний	3
	3	Разработка программной документации	2
Внедрение	1	Разработка инструкции пользователя	1
Итого			33

Общие затраты труда на разработку и внедрение проекта *Q* определяются следующим образом (3.1):

$$Q = t_1+t_2+t_3+t_4+t_n \quad (3.1)$$

где: t_x – затраты труда на выполнение i-го этапа проекта, дни.

Средняя численность исполнителей при реализации проекта разработки и внедрения ПО определяется соотношением (3.2):

$$N=Q\times T_{см}/F \quad (3.2)$$

где:

Q – затраты труда на выполнение проекта (разработка и внедрение ПО), дни

F – фонд рабочего времени, дни.

Величина фонда рабочего времени определяется соотношением (3.3):

$$F =T\times F_M \quad (3.3)$$

где:

T – время выполнения проекта в месяцах,

F_M – фонд времени в текущем месяце, который рассчитывается из учета общего числа дней в году, числа выходных и праздничных дней:

$$F_M=T_{СМ}\times(D_K-D_B-D_n)/12 \quad (3.4)$$

где:

$T_{см}$ – продолжительность рабочего дня, час,

D_K – общее число дней в году,

D_B – число выходных дней в году,

D_n – число праздничных дней в году.

$$F_m=8ч\times(365-104-14)/12=165 \text{ час.}$$

T=21/21=1 мес.

N=21*8/165=1 чел.

Затраты на выполнение проекта состоят из затрат на заработную плату исполнителям, затрат на закупку или аренду оборудования, затрат на организацию рабочих мест и затрат на накладные расходы (3.5):

$$К = C_{зарп}+C_{об}+C_{орг}+C_{накл} \quad (3.5)$$

где:

$C_{зарп}$ – заработная плата исполнителей, руб.

$С_{об}$ – затраты на обеспечение необходимым оборудованием, руб.

$С_{орг}$ – затраты на организацию рабочих мест, руб.

$С_{накл}$ – накладные расходы, руб.

Затраты на выплату исполнителям заработной платы определяются следующим образом (3.6):

$$С_{зарп} = С_{з.осн} + С_{з.отч} \quad (3.6)$$

где:

$С_{з.осн}$ – основная заработная плата, руб.

$С_{з.отч}$ – отчисление с заработной платы, руб.

$С_{з.доп}$ – дополнительная заработная плата, руб.

$С_{зарп}$=18333+5500=23833 руб.

Таблица 5 – Расчет основной заработной платы исполнителей

№	Должность	Оклад, руб.	Дневной оклад, руб.	Трудозатраты ДНИ	Основная заработная, плата, руб.
1.	Программист	18000	873	33	28809

Дневной оклад исполнителей рассчитывается на основе данных по окладам и времени занятости исполнителей (3.7):

$$О_{дн}=(О_{мес}\times Т_{рд})/F_{м} \quad (3.7)$$

где:

$О_{мес}$ – месячный оклад, руб.,

$Т_{рд.}$ – продолжительность рабочего дня, час,

$F_{м}$ – месячный фонд рабочего времени, час, 2.4.

$$О_{дн} = 18000\times 8/165 = 873 \text{ руб.}$$

Согласно налоговому кодексу РФ отчисления производятся в пенсионный фонд РФ, фонд социального страхования, фонды обязательного медицинского страхования, фонд занятости (3.8):

$$С_{з.отч} = С_{з.осн} \times Н_{отч} \quad (3.8)$$

где:

$Н_{отч}$ – отчисления с заработной платы, %

$$С_{з.отч=} 28809*0.3 = 8642,7 \text{ руб.}$$

Затраты на обеспечение работ оборудованием определяются исходя из состава оборудования и определения необходимости его закупки, либо суммы амортизации (в случае использования уже имеющегося) или аренды. Данные о затратах на закупку оборудования следует внести в таблицу (таблица 6):

Таблица 6 – Затраты на оборудование

№ п/п	Наименование оборудования	Количество единиц оборудования	Затраты на оборудование, руб.
1.	ПК	1	28612
Итого			28612

Сумма амортизации оборудования для проекта составит (3.9):

$$A=\sum(Ki \times H_a \times Ti)/100 \times 12 \quad (3.9)$$

где:

n – количество наименований оборудования,

K_i – стоимость единицы оборудования i-nого наименования руб.,

H_a – норма амортизации, %

T_i – длительность эксплуатации оборудования, мес.

$$A = 28612 \times 0.25 \times 0,7/12 = 948 \text{ руб.}$$

Таблица 7 – Затраты на организацию рабочих мест

Вид	Площадь	Стоимость за 1 кв. м. в год	Описание
Офис	14 кв. м.	22300	Высокоскоростной доступ к сети интернет, удобное расположение, эргономичное рабочее место

Затраты на аренду помещения (3.10):

$$C_{орг} = (C_{кв.м./}12) \times (S \times T_{ар}) \quad (3.10)$$

где:

$C_{кв.м}$ – стоимость аренды одного кв. метра площади за год, руб.,

S – арендуемая площадь рабочего помещения, кв.м.,

$T_{ар}$ – срок аренды, мес.

$$C_{орг} = 22300/12 * 14 \text{ кв.м} * 1{,}1 \text{ м} = 28618 \text{ руб.}$$

Накладные расходы рассчитываются в соответствии с действующей методикой в процентах от основной заработной платы (3.11). Обычно они составляют от 60% до 100%. Так как мы рассчитываем себестоимость одного небольшого проекта, то накладные расходы в нашем случае будут составлять 15%.

$$C_{накл} = \%_{.накл} \times C_{з.осн} \quad (3.11)$$

где:

$\%_{накл}$ – накладные расходы, %

$$C_{накл} = 28618 \times 0{,}15 = 4293 \text{ руб.}$$

Общие затраты составляют (3.12):

$$К_{об} = К_{вн} + К \quad (3.12)$$

$$К_{об} = 1305 + 28809 = 30114 \text{ руб.}$$

Таблица 8 – Структура затрат на создание чат-бота

№	Статьи затрат	Сумма, руб.	Структура, %

Продолжение таблицы 8

1	Заработная плата исполнителям	28809	45
2	Затраты на амортизацию оборудования	948	2
3	Накладные расходы	4393	7
4	Затраты на внедрение	1305	2
5	Затраты на организацию рабочих мест	28612	44
	Итого	64067	100

Таким образом, было рассчитано, что разработка чат-бота обойдется организации в 64067 р. Большую часть суммы составляют оплата труда разработчика и затраты на организацию рабочего места.

Выводы по главе 3

В ходе выполнения третьей главы работы было проведено физическое моделирование чат-бота, при этом были реализованы компоненты и модули системы, разработан программный код, проверена работоспособность созданного программного продукта. Также разработано руководство пользователя и рассчитано экономическое обоснование создания чат-бота.

Заключение

В результате выполнения работы достигнута поставленная цель: разработан чат-бот, поддерживающий анализ и выражение эмоций. При достижении цели решены следующие задачи:

- исследованы особенности анализа и выражения эмоций;
- изучено состояние вопроса анализа и выражения эмоций в современных программных системах;
- определены методы и способы анализа и выражения эмоций;
- изучены способы определения эмоционального направления текстового содержимого;
- проведено логическое проектирование чат-бота, выбраны технологии реализации проекта;
- реализован проект чат-бота.

При выполнении работы был проведен анализ и определены особенности выражения эмоций, психологические аспекты эмоций, рассмотрены техники автоматизации определения эмоций. Также были определены методы и способы анализа выражения эмоций, изучены особенности определения эмоционального направления текстового содержимого.

В процессе проектирования выбраны технологии и сервис для реализации чат-бота, рассмотрены концептуальные аспекты функционирования системы, описана логическая модель работы проектируемого чат-бота. Разработанный чат-бот состоит из нескольких частей, которые при взаимодействии друг с другом обеспечивают корректную работу всего приложения: сама программа, которая выполняет все заявленные требования; база данных, являющаяся хранилищем необходимой информации; платформа, на которой будет работать программа (бот), и сервер, на который будет выложен код программы и откуда будет осуществляться ее запуск. В качестве сервиса для чат-бота выбран Telegram. Помимо стандартного обмена сообщениями в диалогах и группах, в мессенджере можно хранить неограниченное количество файлов, вести каналы (микроблоги),

создавать и использовать боты. При помощи специального API сторонние разработчики могут создавать боты. Telegram-боты – разновидность чат-ботов. Их суть заключается в реакции на определенные сообщения от пользователей. При регистрации бота выдается уникальный ключ, с помощью которого в дальнейшем и будет происходить связь между клиентом и сервером. Функционирование приложения может быть описано следующим образом: когда пользователь взаимодействует с чат-ботом в Telegram, API-интерфейс осуществляет отправку данных о взаимодействии в код по HTTP-запросу, в результате чего код также делает отправку сведений, который обозначают, каким образом нужно реагировать. В результате Bot API можно представить в качестве посредника, осуществляющего взаимодействие бота в Telegram с логикой приложения. Bot API включает следующие компоненты: обновления и методы. К разработчику поступают обновления, которые показывают, как пользователь взаимодействовал с ботом. При этом методы вызова требуются с той целью, чтобы бот мог исполнять ряд действий, среди которых отправление сообщений пользователям.

Процесс реализации чат-бота сопровождался физическим моделированием чат-бота, при этом были реализованы компоненты и модули системы, разработан программный код, проверена работоспособность созданного программного продукта. Также разработано руководство пользователя и рассчитано экономическое обоснование создания чат-бота. В качестве языка программирования чат-бота выбран PHP, средой разработки послужила программа PHPStorm. В качестве API использовано Telegrambot API, создавался бот с помощью Botfather, при этом был получен токен для обращения к API Telegram.

Список используемой литературы

Научная и методическая литература

1. Большакова, Е.И. Автоматическая обработка текстов на естественном языке и анализ данных: учеб. пособие / Е.И. Большакова, К.В. Воронцов, Н.Э. Ефремова, Э.С. Клышинский, Н.В. Лукашевич, А.С. Сапин. – М.: Изд-во НИУ ВШЭ, 2017. – 269 с. – Текст : непосредственный.

2. Боярский, К.К. Введение в компьютерную лингвистику: учеб. пособие. – СПб: НИУ ИТМО, 2013. – 72 с. – Текст : непосредственный.

3. Васильев, В.Г. Классификация отзывов пользователей с использованием фрагментных правил / В.Г. Васильев, М.В. Худякова, С. Давыдов // Компьютерная лингвистика и интеллектуальные технологии: по материалам ежегодной международной конференции «Диалог». Вып. 11 (18), – М.: Изд-во РГГУ, 2012. – С. 66-76. – Текст : непосредственный.

4. Вежбицкая, А. Толкование эмоциональных концептов / А. Вежбицкая // Язык. Культура. Познание: сб. ст. – М.: Русские словари, 1996. – С. 201. – Текст : непосредственный.

5. Вересников, Ю.К. О надежности систем обработки информации / Ю.К. Вересников // Актуальные проблемы современной науки. – 2011. – № 2 (58). – С. 193-195. – Текст : непосредственный.

6. Егорова, А.А. Базы данных: учебно-методическое пособие по проведению практических занятий / А.А. Егорова. – М.: МГТУ ГА, 2017. – 36 с. – Текст : непосредственный.

7. Елиферов, В.Г. Бизнес-процессы: регламентация и управление: / В.Г. Елиферов, В.В. Репин.: ИНФРА–М, 2011. – 319 с. – Текст : непосредственный.

8. Жуков, Р.А. СУБД с открытым исходным кодом: возможность применения алгоритмов распараллеливания / Р.А. Жуков // Технические науки. – 2015. – № 1–2 (35–36). – С. 20-21. – Текст : непосредственный.

9. Захаров, В.Н. Инструменты моделирования бизнес-процессов / В.Н Захаров // Вестник МГОУ. Серия: Экономика. – 2016. – №3. – С. 48-53. – Текст : непосредственный.

10. Кальян, В.П. Исследование применимости артикуляционных моделей в задачах распознавания эмоций по речи / В.П. Кальян. Докл. 9-й Междунар. конф. «Интеллектуализация обработки информации». – М.: ТОРУС ПРЕСС, 2011. – С. 334-349. – Текст: непосредственный.

11. Кальян, В.П. Морфология ситуации в системе распознавания эмоционального состояния человека по речи / В.П. Кальян // Модели и методы распознавания речи. – М.: ВЦ РАН им. А. А. Дородницына, 2012. – С. 92-102. – Текст : непосредственный.

12. Кальян, В.П. Построение алгоритмов распознавания эмоционального состояния человека по пара- и экстралингвистическим особенностям речи / В.П. Кальян // Модели и методы распознавания речи. – М.: ВЦ РАН им. А.А. Дородницына, 2010. – С. 24-46. – Текст : непосредственный.

13. Кальян, В.П. Разработка алгоритмов распознавания эмоционального состояния человека по паралингвистическим особенностям речи / В.П. Кальян // Докл. 15-й Всеросс. конф. «Математические методы распознавания образов». – М.: МАКС-Пресс, 2011. – С. 334-349. – Текст : непосредственный.

14. Котельников, Е.В. Автоматический анализ тональности текстов на основе методов машинного обучения / Е.В. Котельникова, М.В. Клековкина // Компьютерная лингвистика и интеллектуальные технологии: по материалам ежегодной международной конференции «Диалог». Вып. 11 (18), – М.: Изд-во РГГУ, 2012. – С. 27-36. – Текст : непосредственный.

15. Коцюба, И.Ю. Основы проектирования информационных систем. Учебное пособие / И.Ю. Коцюба, А.В. Чунаев, А.Н. Шиков. – СПб: Университет ИТМО, 2015. – 206 с. – Текст : непосредственный.

16. Орлова, Н.Н. Языковые средства выражения эмоций: синтаксический аспект: автореф. Дис. На соискание ученой степени канд. фил. Наук / Н.Н. Орлова; ФГОУ ВПО «Южный федеральный университет». – Ростов-на-Дону, 2009. – 24 с. – Текст : непосредственный.

17. Пазельская, А. Метод определения эмоций в текстах на русском языке / А. Пазельская, А. Соловьёв. Труды международной конференции «Диалог, 2011». Р.510-522. – Текст : непосредственный.

18. Поляков, П.Ю. Исследование применимости методов тематической классификации в задаче классификации отзывов о книгах / П.Ю. Поляков, М.В. Калинина, В.В. Плешко // Компьютерная лингвистика и интеллектуальные технологии: по материалам ежегодной международной конференции «Диалог». Вып. 11 (18), – М.: Изд-во РГГУ, 2012. – С. 51-59. – Текст : непосредственный.

19. Провотар, А.И. Особенности и проблемы виртуального общения с помощью чат-ботов / А.И. Провотар, К.А. Клочко // Научные труды Винницкого национального технического университета. 2013. – № 3. С. 2. – Текст : непосредственный.

20. Рудаков, А.В. Технология разработки программных продуктов: учеб. пособие для студ. / А.В. Рудаков. – 5-е изд., стер. – М.: Издательский центр «Академия», 2010. – 208 с. – Текст : непосредственный.

21. Смирнова, Г.Н. Проектирование экономических информационных систем (часть 1) / Г.Н. Смирнова, Ю.Ф. Тельнов – М.: МЭСИ, 2004. – 223 с. – Текст : непосредственный.

22. Смирнова, О.А. Компоненты эмотивного текста (на материале английского языка): дипломная работа / О.А. Смирнова. Волжский университет им. В.Н. Татищева. – Тольятти, 2010. – 65 с. – Текст : непосредственный.

23. Соболь, Е.Ю. Языковые средства репрезентации эмоции удивления в английском художественном тексте / Е.Ю. Соболь // Вестник МГОУ. Серия «лингвистика». – М.: 2011. Вып. №3. – 235 с. – Текст : непосредственный.

24. Соснина, Е.П. Введение в прикладную лингвистику: учебное пособие / Е.П. Соснина. – Ульяновск: УлГТУ, 2012. – 110 с. – Текст : непосредственный.

25. Станиславчик, Е.Н. «Информационные технологии в школьном образовании» / Е.Н. Станиславчик. – М.: Ось-89, 2017. – 128 с. – Текст : непосредственный.

26. Чанина, М.А. Разработка БТС распознавания изменённого психоэмоционального состояния диктора по речевому сигналу // Электронный журнал «Молодёжный научно-технический вестник». 2012, июль. – № 07. ФГБОУ ВПО «МГТУ им. Н.Э. Баумана». Эл. № ФС77-51038/467655. – Текст : непосредственный.

27. Чернявский, Д.И. Моделирование и реинжиниринг бизнес-процессов: учебное пособие / Д.И. Чернявский, Д.В. Рудаков. – Омск: ОмГТУ, 2010. – 84 с. – Текст : непосредственный.

28. Шаховский, В.И. Лингвистика эмоций / В.И. Шаховский// Филологическая наука. – 2007. – №5. – с.3-13. – Текст : непосредственный.

29. Шураков, В.В. Автоматизированное рабочее место для статистической обработки данных / В.В. Шураков, Д.М. Дайитбегов, С.В. Мизрохи, С.В. Ясеновский. – М.: Финансы и статистика, 1990. – 190 с. – Текст : непосредственный.

Электронные ресурсы

30. Апресян, В.Ю. Эмоции: современные американские исследования [Электронный ресурс] / В.Ю. Апресян. – Текст : электронный. – URL: http://lpcs.math.msu.su/~uspensky/journals/siio/34/34_04_APRESYAN.pdf (дата обращения: 08.02.2020) – Режим доступа: для авторизир. пользователей.

31. Брестер, К.Ю. Коллективный эволюционный метод многокритериальной оптимизации в задачах анализа речевых сигналов. Дисс. канд. техн. наук. – Красноярск: 2013. 143 с. Текст : электронный. – URL: http://research.sfu-kras.ru/sites/research.sfu-kras.ru/files/Dissertaciya_ Brester_K.Yu_.pdf. (дата обращения: 18.04.2020). – Режим доступа: для авторизир. пользователей.

32. Вартанов, А.В. Антропоморфный метод распознавания эмоций в звучащей речи [Электронный ресурс] / А.В. Вартанов // Национальный психологический ж., 2013. №2[10]. С. 69-79. Текст : электронный. – URL: http://www.psy.msu.ru/science/npj/ journals/npj-no10-2013.pdf (дата обращения: 18.04.2020). – Режим доступа: для авторизир. пользователей.

33. Людоговский, А. Моделирование бизнес-процессов [Электронный ресурс] / А. Людоговский // Инф. портал «Script coding». – Текст : электронный. – URL: http://www.script-coding.com/bp.htm l (дата обращения: 17.02.2020). – Режим доступа: для авторизир. пользователей.

34. Сайт муниципального бюджетного общеобразовательного учреждения «Средняя школа № 17» [Электронный ресурс] / Текст : электронный. – URL: http://17.edunur.ru/index.php/mbou17/ (дата обращения: 17.02.2020). – Режим доступа: для авторизир. пользователей.

35. Сидоров, К.В. Анализ признаков эмоционально окрашенной речи / К.В. Сидоров, Н.Н. Филатов // Известия ЮФУ. Технические науки, 2012. – Т. 134. №9. С. 39-45. Текст : электронный. – URL: http://eprints.tstu.tver.ru/ 69/1/3.pdf (дата обращения: 18.04.2020). – Режим доступа: для авторизир. пользователей.

Литература на иностранном языке

36. Abdul-Kader S.A. Survey on Chatbot Design Techniques in Speech Conversation Systems / S.A. Abdul-Kader, J. Woods // International Journal of Advanced Computer Science and Applications. – 2015. – Vol. 6, No. 7. – P. 72-80.

37. Bellman R.E. Adaptive Control Processes : A Guided Tour / R.E. Bellman // Princeton University Press Princeton, New Jersey, 1961. – 255 p.

38. Bot Code Examples –Telegram APIs [Электронный ресурс]. – Режим доступа: https://core.Telegram.org/bots/samples(дата обращения: 13.03.2020).

39. Carenini G. Extracting knowledge from evaluative text / G. Carenini, Ng R.E. Zwart // Proceedings of the 3rd international conference on Knowledge capture. – 2005. – P.11-18.

40. Dahiya M. A tool of conversation: Chatbot / M. Dahiya // International Journal of Computer Sciences and Engineering. – 2017. – Vol. 5, No. 5. – P. 158-161.

41. Gold B. Speech and Audio Signal Processing/ B. Gold, N. Morgan // Wiley, 2000. – 250 p.

42. Goldberg D.E. Genetic Algorithms in Search, Optimization and Machine Learning / D.E. Goldberg // Addison-Wesley Longman Publishing Co. Boston, 1st edition, 1989. – 372 p.

43. Hu M. Mining, and summarizing customer reviews / M. Hu, B. Liu // Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, 2004. – P. 168-177.

44. Kluwer T. From chatbots to dialog systems / T. Kluwer // Conversational agents and natural language interaction: Techniques and effective practices. – IGI Global, 2011. – P. 1-22.

45. Lerman K. Sentiment summarization: evaluating and learning user preferences / K. Lerman, S. Blair-Goldensohn, R. Mcdonald // Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics, 2009. P. – 514-522.

46. Lu Y. Rated aspect summarization of short comments / Y. Lu, C. Zhai, N. Sundaresan // Proceedings of the 18th international conference on World wide web, 2009. – P. 131-140.

47. Mikic Fonte F.A. NLAST: A natural language assistant for students / F.A. Mikic Fonte, M.L. Nistal, J.C. Burguillo Rial, M.C. Rodríguez // IEEE Global Engineering Education Conference (EDUCON), 2016. – P. 709-713.

48. Ng R., Pauls A. Multi-document summarization of evaluative text / R. Ng, A. Pauls // Proceedings of the 11st Conference of the European Chapter of the Association for Computational Linguistics, 2006. – P. 305-312.

49. Oatley K. Best Laid Schemes: the Psychology of Emotions / K. Oatley. – Cambridge: University Press, 1992. – 525 p.

50. Pang B. Opinion Mining and Sentiment Analysis / B. Pang, L. Lee // Foundations and Trends in Information Retrieval. – 2008. – Vol. 2, № 1-2. – P.1-135.

51. Prabowo R. Sentiment analysis: A combined approach / R. Prabowo, M. Thelwall // Journal of Informetrics. – 2009. – Vol. 3, № 2. – P. 143-157.

52. Sayed S. Android based Chat-Bot / S. Sayed, R. Jain, B. Lokhandwala, F. Barodawala, M. Rajkotwala // International Journal of Computer Applications. – 2016. – Vol. 137, № 10. – P. 29-32.

53. Schmid H. Improvements in part-of-speech tagging with an application to german / H. Schmid // In: Proceedings of the ACL SIGDAT – Workshop, Citeseer, 1995. – P. 47-50.

54. Schmid H. Probabilistic part-of-speech tagging using decision trees / H. Schmid // In: Proceedings of the international conference on new methods in language processing, Citeseer, 1994. – Vol. 12. – P. 44-49.

55. Schuller B. Evolutionary feature generation in speech emotion recognition / B. Schuller, S. Reiter, G. Rigoll // Multimedia and Expo, 2006 IEEE International Conference. – 2006. – P. 5-8.

56. Schuller B. The INTERSPEECH 2009 emotion challenge / B. Schuller, S. Steidl, A. Batliner // Interspeech, 2009. – P. 312-315.

57. Sharoff S. Designing and evaluating a Russian Tagset / S. Sharoff, M. Kopotev, T. Erjavec, A. Feldman, D. Divjak // In: LREC, 2008. – P. 279-285.

58. Snyder B. Multiple Aspect Ranking using the Good Grief Algorithm / B. Snyder, R. Barzilay // Proceedings of the Joint Human Language Technology: North American Chapter of the ACL Conference HLT-NAACL, 2007. – P. 300-307.

59. Soroka A. New Method of Speech Signals Adaptive Features Construction Based on the Wavelet-like Transform and Support Vector Machines / A. Soroka, P. Kovalets, I. Kheidorov // Springer, Speech and Computer Lecture Notes in Computer Science. – 2014. – Vol. 8773. – P. 308-314.

60. Temko A. Classification of acoustic events using SVM-based clustering schemes / A. Temko, C. Nadeu // Pattern Recognition. – 2006. – Vol. 39, № 4. – P. 682-694.

61. Titov I. A Joint Model of Text and Aspect Ratings for Sentiment Summarization / I. Titov, R. Mcdonald // Proceedings of ACL-08: HLT, 2008. – P. 308-316.

62. Xue-ying Zhang Adaptive bands filter bank optimized by genetic algorithm for robust speech recognition system / Xue-ying Zhang, Li-xia Huang, G. Evangelista // J. of Central South University of Technology. – 2011. – Vol. 18. – P. 1595-1601.

Приложение А

Исходный код чат-бота

```
<?php

header('Content-Type: text/html; charset=utf-8');

// подключаем API
require_once("vendor/autoload.php");

// создаем переменную бота
$token = "1092442739:AAGTQJxfxzbHfDTxPJQDihun8zAZcP26lTc";
$bot = new \TelegramBot\Api\Client($token);

// если бот еще не зарегистрирован - регистрируем
if (!file_exists("registered.trigger")) {
    /**
     * файл registered.trigger будет создаваться после регистрации бота.
     * если этого файла нет значит бот не зарегистрирован
     */

    // URl текущей страницы
    $page_url = "https://" . $_SERVER["SERVER_NAME"] . $_SERVER["REQUEST_URI"];
    $result = $bot->setWebhook($page_url);
    if ($result) {
        file_put_contents("registered.trigger", time()); // создаем файл дабы прекратить повторные регистрации
```

```
}
}

// обязательное. Запуск бота
$bot->command('start', function ($message) use ($bot) {
    $answer = 'Добро пожаловать!';
    $name = "Привет" . " " . $message->getChat()->getFirstName() . " Тебя приветствует EmoBot, который тебя выслушает и поможет!";
    $bot->sendMessage($message->getChat()->getId(), $name);
});

// помощь
$bot->command('help', function ($message) use ($bot) {
    $answer = 'Команды:
    /help - помощь';
    $bot->sendMessage($message->getChat()->getId(), $answer);
});

// общение
$bot->on(function ($Update) use ($bot) {
    $message = $Update->getMessage();
    $mtext = $message->getText();
    $cid = $message->getChat()->getId();

    // connect db
    $servername = "localhost";
    $username = "u936519911_adm";
```

```
        $password = "ZfnFATK";
        $dbname = "u936519911_adm";

        // Create connection
        $db = mysqli_connect($servername, $username, $password, $dbname);
        mysqli_set_charset($db, 'utf8mb4');
        $sqlSelect = "SELECT * FROM questions_answers WHERE question = '{$mtext}'";
        $result = $db->query($sqlSelect);
        if (!empty($result)) {
            foreach ($result
                    as $row) {
                $answer = $row['answer'];
                $bot->sendMessage($message->getChat()->getId(), $answer);
            }
        }
        mysqli_close($db);

        if (mb_stripos($mtext, "добрый день") !== false) {
            $bot->sendMessage($message->getChat()->getId(), "Что Вас интересует?");
        }

    }, function ($message) use ($name) {
        return true; // когда тут true - команда проходит
    });
```

```
// запускаем обработку
$bot->run();

<?php
/**
 * Created by PhpStorm.
 * File: admin-a.php
 * Date: 30/04/2020
 * Time: 18:37
 */

if (isset($_POST)) {
    // Create connection
    // connect db
    $servername = "localhost";
    $username = "u936519911_adm";
    $password = "ZfnFATK";
    $dbname = "u936519911_adm";

    $question = $_POST['question'];
    $answer = $_POST['answer'];
    $db = mysqli_connect($servername, $username, $password, $dbname);
    $sqlSelect = "INSERT INTO questions_answers (question, answer) VALUES
('{$question}', '{$answer}')";
    $result = $db->query($sqlSelect);
}
```

```
require_once('includes/header.php');
?>
   <div class="container mt-5">

      <form method="post">
         <div class="form-group">
            <label for="question">Вопрос </label>
            <input type="text" class="form-control" id="question"
placeholder="Вопрос"
                   name="question">
         </div>
         <div class="form-group">
            <label for="answer">Ответ</label>
            <input type="text" class="form-control" id="answer"
placeholder="Ответ"
                   name="answer">
         </div>
         <button type="submit" class="btn btn-primary">Сохранить</button>
         <a href="" class="btn btn-primary">Отмена</a>
      </form>
   </div>
<?php
require_once('includes/footer.php');

<?php
/**
 * Copyright (C) 2019 Phppot
```

```
 *
 * Distributed under MIT license with an exception that,
 * you don’t have to include the full MIT License in your code.
 * In essense, you can use it on commercial software, modify and distribute free.
 * Though not mandatory, you are requested to attribute this URL in your code or website.
 */

namespace Phppot;

use mysqli;

/**
 * Generic datasource class for handling DB operations.
 * Uses MySqli and PreparedStatements.
 *
 * @version 2.5 - recordCount function added
 */
class DataSource
{

    // PHP 7.1.0 visibility modifiers are allowed for class constants.
    // when using above 7.1.0, declare the below constants as private
    private const HOST = '31.220.20.81';

    private const USERNAME = 'u936519911_adm';
```

```
private const PASSWORD = 'ZfnFATK';

private const DATABASENAME = 'u936519911_adm';

private $conn;

/**
 * PHP implicitly takes care of cleanup for default connection types.
 * So no need to worry about closing the connection.
 *
 * Singletons not required in PHP as there is no
 * concept of shared memory.
 * Every object lives only for a request.
 *
 * Keeping things simple and that works!
 */
function __construct()
{
   $this->conn = $this->getConnection();
}

/**
 * If connection object is needed use this method and get access to it.
 * Otherwise, use the below methods for insert / update / etc.
 *
 * @return mysqli
 */
```

```
    public function getConnection()
        {
            $conn = new mysqli(self::HOST, self::USERNAME, self::PASSWORD, self::DATABASENAME);

            if (mysqli_connect_errno()) {
                trigger_error("Problem with connecting to database.");
            }

            $conn->set_charset("utf8mb4");
            return $conn;
        }

        /**
         * To get database results
         *
         * @param string $query
         * @param string $paramType
         * @param array $paramArray
         * @return array
         */
        public function select($query, $paramType = "", $paramArray = array())
        {
            $stmt = $this->conn->prepare($query);

            if (!empty($paramType) && !empty($paramArray)) {
```

```
    $this->bindQueryParams($stmt, $paramType, $paramArray);
    }
    $stmt->execute();
    $result = $stmt->get_result();

    if ($result->num_rows > 0) {
        while ($row = $result->fetch_assoc()) {
            $resultset[] = $row;
        }
    }

    if (!empty($resultset)) {
        return $resultset;
    }
}

/**
 * To insert
 *
 * @param string $query
 * @param string $paramType
 * @param array $paramArray
 * @return int
 */
public function insert($query, $paramType, $paramArray)
{
    $stmt = $this->conn->prepare($query);
```

```
    $this->bindQueryParams($stmt, $paramType, $paramArray);

    $stmt->execute();
    $insertId = $stmt->insert_id;
    return $insertId;
}

/**
 * To execute query
 *
 * @param string $query
 * @param string $paramType
 * @param array $paramArray
 */
public function execute($query, $paramType = "", $paramArray = array())
{
    $stmt = $this->conn->prepare($query);

    if (!empty($paramType) && !empty($paramArray)) {
        $this->bindQueryParams($stmt, $paramType, $paramArray);
    }
    $stmt->execute();
}

/**
 * 1.
 * Prepares parameter binding
```

```
 * 2. Bind prameters to the sql statement
 *
 * @param string $stmt
 * @param string $paramType
 * @param array $paramArray
 */
public function bindQueryParams($stmt, $paramType, $paramArray = array())
{
    $paramValueReference[] = &$paramType;
    for ($i = 0; $i < count($paramArray); $i++) {
        $paramValueReference[] = &$paramArray[$i];
    }
    call_user_func_array(array(
        $stmt,
        'bind_param'
    ), $paramValueReference);
}

/**
 * To get database results
 *
 * @param string $query
 * @param string $paramType
 * @param array $paramArray
 * @return array
 */
```

```
    public function getRecordCount($query, $paramType = "", $paramArray = array())
        {
            $stmt = $this->conn->prepare($query);
            if (!empty($paramType) && !empty($paramArray)) {

                $this->bindQueryParams($stmt, $paramType, $paramArray);
            }
            $stmt->execute();
            $stmt->store_result();
            $recordCount = $stmt->num_rows;

            return $recordCount;
        }
    }

    <?php
    require_once("includes/header.php");
    ?>
        <div class="container mt-5">
            <hr>
            <div class="row">
                <div class="col-md-12">
                    <a href="admin-a.php" class="btn btn-info">Add question and answer</a>
                    <hr>
```

Upload question and answer

Продолжение Приложения А

```
            </div>
        </div>
    </div>
<?php require_once("includes/footer.php"); ?>

<?php
require_once("includes/header.php");
?>
<?php

use Phppot\DataSource;

require_once 'DataSource.php';
$db = new DataSource();
$conn = $db->getConnection();

if (isset($_POST["import"])) {

    $fileName = $_FILES["file"]["tmp_name"];

    if ($_FILES["file"]["size"] > 0) {

        $file = fopen($fileName, "r");

        while (($column = fgetcsv($file, 10000, ";")) !== FALSE) {
```

```
    $question = "";
```

Продолжение Приложения А

```
    if (isset($column[0])) {
        $question = mysqli_real_escape_string($conn, $column[0]);
    }
    $answer = "";
    if (isset($column[1])) {
        $answer = mysqli_real_escape_string($conn, $column[1]);
    }

    $sqlInsert = "INSERT into questions_answers (question,answer)
          values (?,?)";
    $paramType = "ss";
    $paramArray = array(
        $question,
        $answer
    );
    $insertId = $db->insert($sqlInsert, $paramType, $paramArray);

    if (!empty($insertId)) {
        $type = "success";
        $message = "CSV Data Imported into the Database";
    } else {
        $type = "error";
        $message = "Problem in Importing CSV Data";
    }
}
```

```
    }
}
```

Продолжение Приложения А

```
?>
    <div class="container mt-5">
        <h2>Импорт данных для чат-бота (формат CSV)</h2>
        <div id="response"
            class="<?php if (!empty($type)) {
                echo $type . " display-block";
            } ?>">
            <?php if (!empty($message)) {
                echo $message;
            } ?>
        </div>
        <div class="container">
            <div class="row">
                <form class="form form-horizontal" action="" method="post"
                    name="frmCSVImport" id="frmCSVImport"
                    enctype="multipart/form-data">
                    <div class="input-row">
                        <label class="col-md-4 control-label">Выбрать CSV
                            файл</label> <input type="file" name="file"
                                id="file" accept=".csv">
                        <button type="submit" id="submit" name="import"
                            class="btn btn-primary">Import
                        </button>
                    </div>
                </form>
```

```
</div>
<hr/>
```

Продолжение Приложения А

```
<?php
$sqlSelect = "SELECT id, question, answer FROM questions_answers";
$result = $db->select($sqlSelect);
if (!empty($result)) {
    ?>
    <table class="table table-bordered table-hover" id='userTable'>
        <thead>
        <tr>
            <th>ID</th>
            <th>Question</th>
            <th>Answer</th>
        </tr>
        </thead>
        <?php
        foreach ($result

        as $row) {
        ?>
        <tbody>
        <tr>
            <td><?php echo $row['id']; ?></td>
            <td><?php echo $row['question']; ?></td>
            <td><?php echo $row['answer']; ?></td>
        </tr>
        <?php
```

```
                }
```

Продолжение Приложения А

```
            while($row = $result->fetch_object()){
                echo '<strong>question: </strong>'.$row->question.'<br/>'.
                    '<strong>answer: </strong>'.$row->answer.'<br/>';

                }

                ?>
                </tbody>
            </table>
        <?php } ?>
    </div>
</div>
<?php require_once("includes/footer.php"); ?>

<!DOCTYPE html>
<html lang="ru">
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>
        <?php
        if (isset($title)) {
            echo $title;
        } else {
```

```
        echo "Админ-панель";
    }
```

Продолжение Приложения А

```
  ?>
</title>
<!-- Bootstrap -->
<link href="css/bootstrap-4.4.1.css" rel="stylesheet">
<meta name="theme-color" content="#ffffff">

<script src="jquery-3.2.1.min.js"></script>
<style>
    .success {
        background: #c7efd9;
        border: #bbe2cd 1px solid;
    }
</style>
<script type="text/javascript">
    $(document).ready(function () {
        $("#frmCSVImport").on("submit", function () {

            $("#response").attr("class", "");
            $("#response").html("");
            var fileType = ".csv";
            var regex = new RegExp("([a-zA-Z0-9\s_\\.\-:])+(" + fileType + ")$");
            if (!regex.test($("#file").val().toLowerCase())) {
                $("#response").addClass("error");
                $("#response").addClass("display-block");
```

```
                    $("#response").html("Invalid File. Upload : <b>" + fileType + "</b>
Files.");
                    return false;
                                Продолжение Приложения А
                }
                    return true;
                });
            });
        </script>

    </head>
    <body>
    <nav class="navbar fixed-top navbar-expand-lg navbar-dark bg-primary">
        <div class="container">
            <a class="navbar-brand" href="#">Admin</a>
            <div class="collapse navbar-collapse" id="navbarSupportedContent">
                <ul class="navbar-nav mr-auto">
                    <li class="nav-item active">
                        <a class="nav-link" href="/">Main <span class="sr-
only">(current)</span></a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/admin-a.php">Add Question and
Answer</a>
                    </li>
                    <li class="nav-item">
                        <a class="nav-link" href="/question-answer-list.php">List and
Upload</a>
```

```
                </li>
              </ul>
            </div>
```

Продолжение Приложения А

```
          </div>
        </nav>
        <hr>
        <footer>
          <div class="container pt-5">
            <div class="row">
              <div class="col-12">
                <!--                <footer class="fixed-bottom navbar navbar-dark bg-
primary">-->
                <footer class="fixed-bottom navbar navbar-dark bg-primary mt-4">
                  <div class="container mt-2">
                    <h6 style="color: white">Copyright © EmoBot Admin. All rights
reserved <a href="../admin/"
                                                                    style="color: black"
                                                                    class="align-
right">admin</a>
                      <a style="color: black" href="/emo-bot/server/admin">Program-
O</a>
                    </h6>
                  </div>
                </footer>
              </div>
            </div>
          </div>
```

```
</footer>
</body>
</html>
```