Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Программное обеспечение ЭВМ и информационные технологии**

ОТЧЕТ ПО ПРОЕКТНО-ТЕХНОЛОГИЧЕСКОЙ ПРАКТИКЕ

Команда **WExL00P**
Группа **ИУ7-26Б**

Студент	_____ *подпись, дата*	Котцов М.Д. *фамилия, и.о.*
Студент	_____ *подпись, дата*	Порошин Д.Ю. *фамилия, и.о.*
Студент	_____ *подпись, дата*	Слепокурова М.Ф. *фамилия, и.о.*
Студент	_____ *подпись, дата*	Кузьмин К. О. *фамилия, и.о.*
Студент	_____ *подпись, дата*	Борисов М.А. *фамилия, и.о.*
Ментор	_____ *подпись, дата*	Садулаева Т.Р. *фамилия, и.о.*
Руководитель практики	_____ *подпись, дата*	Оленев А.А. *фамилия, и.о.*

2020 г.

Оглавление

Введение **3**

Роли в команде 3

Аналитический раздел **4**

Обзор существующих решений 4

Постановка задачи 4

Конструкторский раздел **5**

Декомпозиция 5

Структура программного продукта 8

Требования к модулю регистрации 8

Требования к модулю загрузки 8

Требования к модулю поиска 9

Требования к модулю оценки 9

Алгоритмы и структуры данных 9

Пользовательский интерфейс 12

Тестирование 12

Технологический раздел **16**

Выбор языка программирования 16

Выбор платформы для развёртывания приложения 16

Выбор базы данных 16

Выбор библиотек 17

Реализация тестирования 17

Модель разработки 18

Развертывание 18

Реализация. Характеристики качества. 18

Заключение **19**

Введение

Цель проекта — разработка и запуск чат-бота для мессенджера Telegram, который обеспечит доступ пользователей к учебным материалам, загруженным в базу данных чат-бота.

Основным назначением чат-бота является предоставление студентам МГТУ им. Н.Э. Баумана удобного доступа к учебным материалам с возможностью ранжирования материалов по рейтингу в системе из мессенджера, которым они часто пользуются.

Роли в команде

1. Порошин Даниил — системный администратор, DevOps — занимался настройкой виртуальных окружений, развертыванием приложения, анализом технического аспекта системы взаимодействия участников команды, управлял, настраивал сервер и базы данных. Страницы отчета: 16-20.
2. Котцов Матвей — TeamLead, технический писатель — занимался постановкой задач, контролем за выполнением сроков, предлагал и критиковал те или иные технические решения, принимал участие в рефакторинге кода и написании документации. Страницы отчета: так или иначе 1-20.
3. Слепокурова Мария — дизайнер, тестировщик — занималась разработкой визуальных материалов (логотипа команды, чат-бота), разработкой IDEF0-диаграммы, созданием шаблонов ответов чат-бота на запросы пользователя, написанием модульных тестов и непосредственным тестированием готового продукта. Страницы отчета: 5-7, 12-16.
4. Кузьмин Кирилл — backend разработчик — занимался разработкой чат-бота, в частности — модуля поиска. Страницы отчета: 11.

5. Борисов Максим — backend разработчик — занимался разработкой чат-бота, в частности — модуля загрузки. Страницы отчета: 10.

Аналитический раздел

Обзор существующих решений

Перед постановкой задачи был проведен анализ существующих решений, выявление их недостатков и поиск их возможных решений. Аналогами нашего проекта являются системы управления курсами (moodle-системы). Поскольку проект нацелен на пользователей из числа студентов МГТУ им. Н.Э. Баумана, в качестве основного аналога была выбрана система e-learning.bmstu.ru, в которой преподаватели публикуют учебные материалы по разным дисциплинам.

В результате анализа системы были отмечены следующие недостатки. Система требует повторной авторизации для доступа к курсам спустя некоторое время. Это, в совокупности с необходимостью переходить на отдельный сайт (e-learning.bmstu.ru), увеличивает время доступа студента к учебным материалам. К тому же, в системе отсутствует возможность поиска по материалам как конкретного курса, так и общих по кафедре, факультету, университету. Таким образом, необходимость нового решения была обоснована перечисленными недостатками действующей системы.

Постановка задачи

В рамках проекта были поставлены следующие задачи:

1. Разработка интерфейса чат-бота
2. Создание механизма информационного взаимодействия между чат-ботом и базой данных
3. Написание технической документации;
4. Демонстрация прототипа чат-бота, обеспечивающего:

a. загрузку пользовательских учебных материалов;
b. поиск по загруженным учебным материалам и их ранжирование;
c. оценку загруженных учебных материалов;
d. регистрацию пользователей чат-бота.

К чат-боту предъявлены следующие требования:

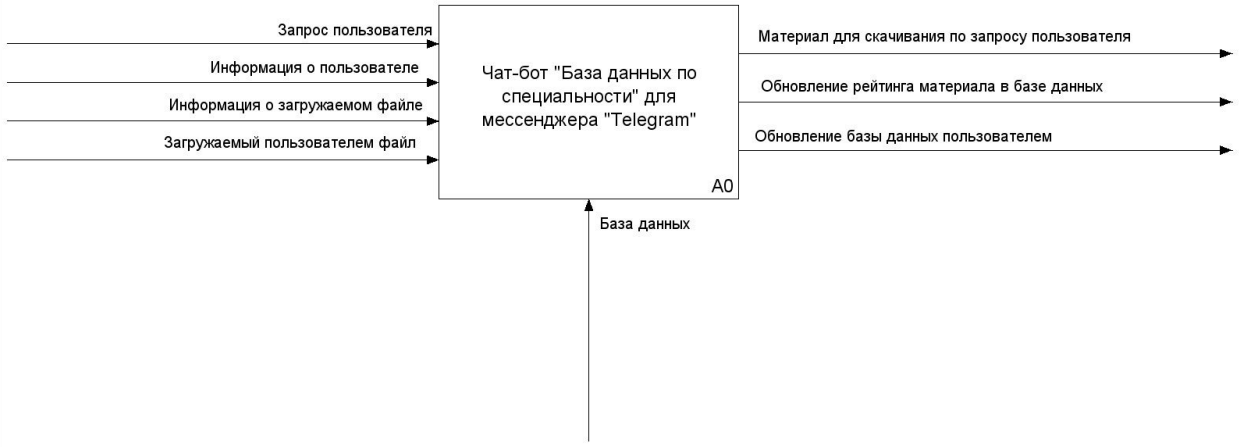
1. Разрабатываемый чат-бот должен соответствовать принципу расширяемости, т.е. иметь возможность наращивания своей функциональности, добавления новых источников данных
2. Разрабатываемый чат-бот должен работать с мобильными устройствами
3. Разрабатываемый чат-бот должен иметь комплекс средств и мер обеспечения информационной безопасности, позволяя жестко разграничивать права доступа пользователей к данным
4. Чат-бот работает как в адресной книге
5. Чат-бот не может первым начать общение с пользователем (первичное общение, рассылка спама должна быть исключена)

Конструкторский раздел

Декомпозиция

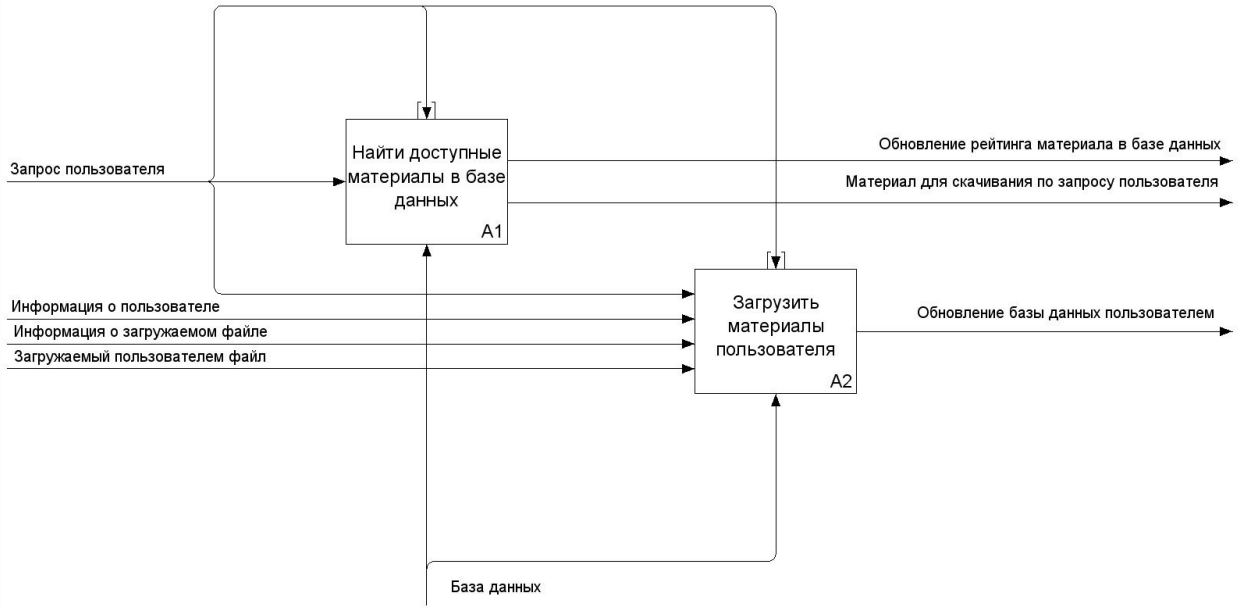
Перед началом реализации были разработаны IDEF0-диаграммы для отражения декомпозиции поставленной задачи. Это позволило избежать архитектурных ошибок на ранних стадиях разработки программного продукта.

ИСПОЛЬЗУЕТСЯ В:	АВТОР: WExL00P	ДАТА: 04.05.2020	РАЗРАБАТЫВАЕТСЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
МГТУ им. Н.Э.Баумана	ПРОЕКТ: Чат-бот "База данных по специальности"	РЕВИЗИЯ: 05.05.2020	ЧЕРНОВИК			**ВЕРХ**
	ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10		РЕКОМЕНДОВАНО			
			ПУБЛИКАЦИЯ			



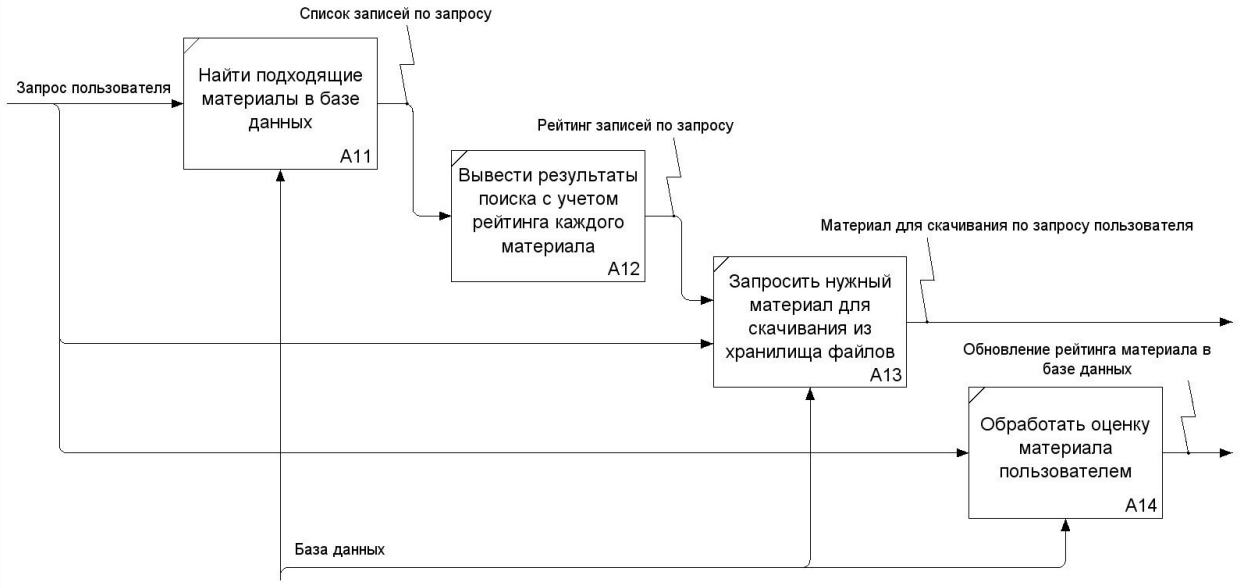
Ветка: A-0	Название: Чат-бот "База данных по специальности" для мессенджера "Telegram"	Номер: 1

ИСПОЛЬЗУЕТСЯ В:	АВТОР: WExL00P	ДАТА: 04.05.2020	РАЗРАБАТЫВАЕТСЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
МГТУ им. Н.Э.Баумана	ПРОЕКТ: Чат-бот "База данных по специальности"	РЕВИЗИЯ: 05.05.2020	ЧЕРНОВИК			
	ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10		РЕКОМЕНДОВАНО			
			ПУБЛИКАЦИЯ			



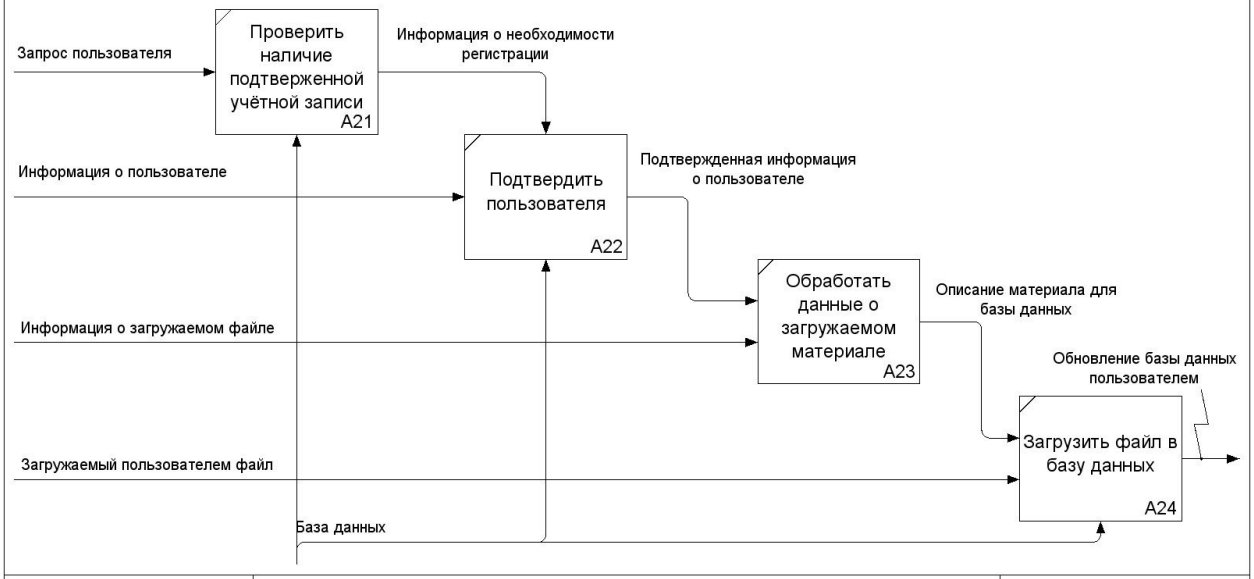
Ветка: A0	Название: Чат-бот "База данных по специальности" для мессенджера "Telegram"	Номер: 2

ИСПОЛЬЗУЕТСЯ В:	АВТОР: WExL00P	ДАТА: 04.05.2020	РАЗРАБАТЫВАЕТСЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
МГТУ им. Н.Э.Баумана	ПРОЕКТ: Чат-бот "База данных по специальности"	РЕВИЗИЯ: 05.05.2020	ЧЕРНОВИК			
	ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10		РЕКОМЕНДОВАНО			
			ПУБЛИКАЦИЯ			



Ветка: A1	Название: Найти доступные материалы в базе данных	Номер: 3

ИСПОЛЬЗУЕТСЯ В:	АВТОР: WExL00P	ДАТА: 04.05.2020	РАЗРАБАТЫВАЕТСЯ	ЧИТАТЕЛЬ	ДАТА	КОНТЕКСТ:
МГТУ им. Н.Э.Баумана	ПРОЕКТ: Чат-бот "База данных по специальности"	РЕВИЗИЯ: 05.05.2020	ЧЕРНОВИК			
	ЗАМЕЧАНИЯ: 1 2 3 4 5 6 7 8 9 10		РЕКОМЕНДОВАНО			
			ПУБЛИКАЦИЯ			



Ветка: A2	Название: Загрузить материалы пользователя	Номер: 4

Структура программного продукта

В структуре программного продукта есть три основных составляющих: пользователь, который взаимодействует с чат-ботом посредством сообщений в мессенджере Telegram, база данных, в которой хранится информация о пользователях, загружаемых файлах и их рейтинг, а также облачное хранилище Telegram, в котором находятся все загружаемые файлы.

Требования к модулю регистрации

Регистрация пользователя происходит в момент первой попытки загрузить, оценить или скачать учебный материал. Каждый пользователь, претендующий на право загружать, скачивать и оценивать материалы, предоставляет следующую информацию:

1. имя и фамилия
2. адрес электронной почты в домене bmstu.ru

Подтверждение пользователя происходит с помощью ввода специального кода, отправляемого ему в письме на адрес электронной почты, указанной при регистрации. До момента подтверждения почты пользователь считается неподтвержденным и имеет соответствующую запись об этом в базе данных.

Требования к модулю загрузки

Перед загрузкой учебного материала необходимо убедиться, что пользователь зарегистрирован в системе.

Поля при загрузке учебного материала:

1. файл в формате docx, doc, pdf или pptx;
2. заголовок (краткая информация о содержимом) файла;
3. курс;
4. предмет;

5. рейтинг материала (по умолчанию равен 0).

Предусмотреть базовую проверку на корректность файла: расширение файла соответствует обозначенным выше, текстовые поля имеют корректную длину, поле курс хранит корректное целое число.

Требования к модулю поиска

Необходимо реализовать возможность поиска по одному или нескольким словам, причем поиск осуществляется по всем полям базы данных (название и предмет, как минимум).

Вывод результатов поиска происходит в порядке неубывания рейтинга учебного материала, чтобы пользователю не приходилось листать вверх в поиске лучшего материала. Результаты поиска включают в себя отображение рейтинга, полей файла, ссылки/кнопки на скачивание и кнопки оценки материала.

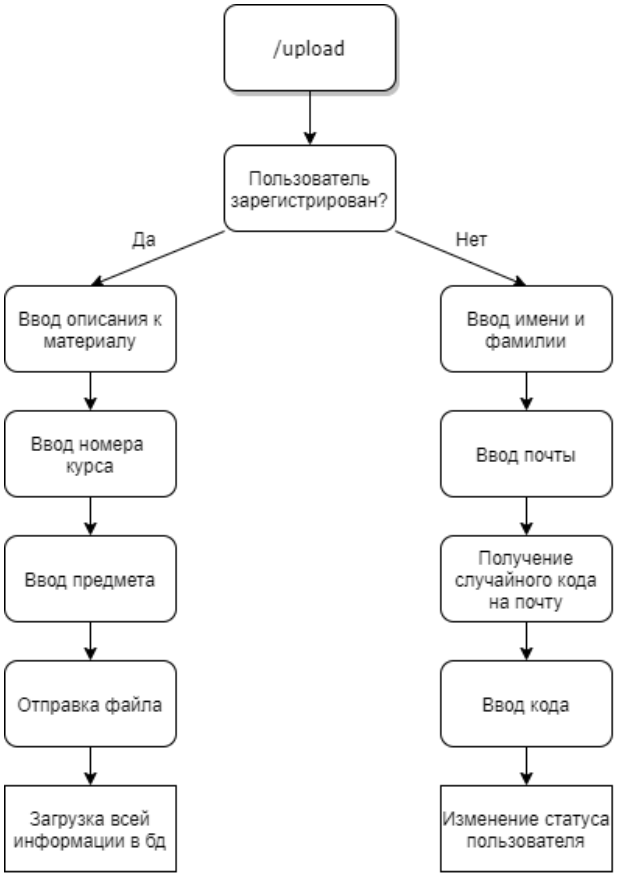
Требования к модулю оценки

Предоставить пользователю возможность поставить учебному материалу оценку -1 или +1. Пользователь может оценить учебный материал только один раз. Предоставить пользователю возможность отозвать свою оценку.

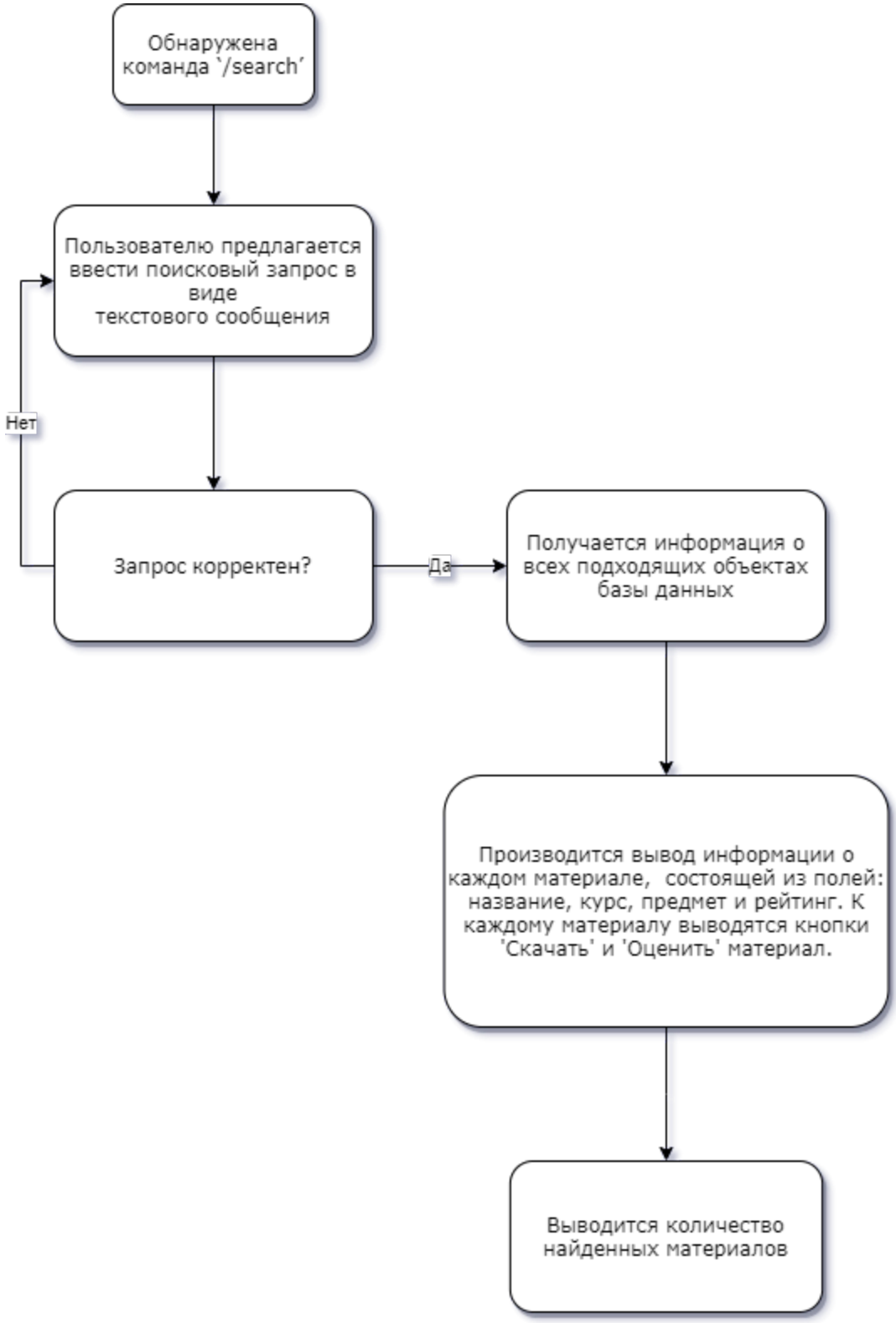
Алгоритмы и структуры данных

В качестве структур данных использовались реляционные базы данных и key-value хранилище.

Ниже приведены визуализации алгоритмов основных команд чат-бота.



Примечание: команда /cancel позволяет прервать режим загрузки на любом шаге


Обнаружена команда `/search´
Пользователю предлагается ввести поисковый запрос в виде текстового сообщения
Нет
Запрос корректен?
Да
Получается информация о всех подходящих объектах базы данных
Производится вывод информации о каждом материале, состоящей из полей: название, курс, предмет и рейтинг. К каждому материалу выводятся кнопки 'Скачать' и 'Оценить' материал.
Выводится количество найденных материалов

Пользовательский интерфейс

Пользовательский интерфейс реализуется с использованием готовых элементов управления, предоставляемых Telegram (кнопки, сообщения).

Тестирование

При написании модульных тестов для функций, располагающихся в `app/check_correct.py`, были выделены следующие классы эквивалентности:

1. **def is_title_correct(message)**

   Проверка названия материала

Класс эквивалентности	Ожидаемый результат
пустое сообщение	False
сообщение, состоящее только из пробельных символов	False
сообщение, состоящее только из специальных символов	False
не сообщение (не текстовая информация)	False
сообщение, состоящее только из цифр	False
сообщение, состоящее только из букв и пробельных символов	True
сообщение, состоящее из букв, цифр и пробельных символов	True
сообщение, состоящее только из букв	True
сообщение, состоящее из букв и специальных символов	True
сообщение, превосходящее по длине максимальное значение (>256 символов)	False

2. **def test_is_course_correct()**

   Проверка курса предмета

Класс эквивалентности	Ожидаемый результат
пустое сообщение	False
сообщение, состоящее только из пробельных символов	False
сообщение, состоящее только из специальных символов	False
не сообщение (не текстовая информация)	False
сообщение, содержащее буквы	False
сообщение, содержащее цифру 0	False
сообщение, содержащее малое отрицательное значение	False
сообщение, содержащее дробное число	False
сообщение, содержащее специальные символы	False
сообщение, содержащее текстовое представление числа	False
сообщение, содержащее большое положительное число (превышающее максимальное возможное значение курса)	False
сообщение, содержащее корректное число и пробельные символы	False
сообщение, содержащее корректное максимальное значение курса	True
сообщение, содержащее корректное минимальное значение курса	True

3. **def test_is_subject_correct()**

   Проверка предмета материала

Класс эквивалентности	Ожидаемый результат
пустое сообщение	False
сообщение, состоящее только из пробельных символов	False

сообщение, состоящее только из специальных символов	False
не сообщение (не текстовая информация)	False
сообщение, состоящее только из цифр	False
сообщение, состоящее только из специальных символов	False
сообщение, состоящее из букв (корректного предмета) и специальных символов	False
сообщение, содержащее некорректно записанный предмет	False
сообщение, содержащее корректно записанный предмет строчными буквами	True
сообщение, содержащее корректно записанный предмет заглавными буквами	True

4. **test_is_name_surname_correct()**

   Проверка имени и фамилии пользователя

Класс эквивалентности	Ожидаемый результат
пустое сообщение	False
сообщение, состоящее только из пробельных символов	False
сообщение, состоящее только из специальных символов	False
не сообщение (не текстовая информация)	False
сообщение, состоящее только из цифр	False
сообщение, состоящее только из специальных символов	False
сообщение, содержащее только одну букву	False
сообщение, состоящее только из одного слова	False
сообщение, состоящее из двух слов, разделённых не пробельным символами (то же, что и несколько слов)	False

сообщение, состоящее из двух слов, одно из которых содержит цифры	True
сообщение, состоящее из двух слов, содержащих специальные символы (дефисное написание)	True
сообщение, состоящее из двух слов, длина одного из которых минимальна (две буквы)	True

5. **test_is_email_correct()**

   Проверка адреса электронной почты пользователя

Класс эквивалентности	Ожидаемый результат
пустое сообщение	False
сообщение, состоящее только из пробельных символов	False
сообщение, содержащее только символ @	False
сообщение, содержащее только символ @ и специальные символы	False
не сообщение (не текстовая информация)	False
сообщение, содержащее только символ @ и буквы после него	False
сообщение, содержащее символ @ и буквы до и после него	False
сообщение, содержащее почтовый адрес, не являющимся корректным в общем смысле	False
сообщение, содержащее запрещенный почтовый адрес в допустимом домене (all)	False
сообщение, состоящее только из букв	False
сообщение, содержащее два символа @	False
сообщение, не содержащее символа @	False
сообщение, начинающееся с символа @	False

сообщение, содержащее первый вариант корректной и допустимой записи почтового адреса (...@bmstu.ru)	True
сообщение, содержащее второй вариант корректной и допустимой записи почтового адреса (...@student.bmstu.ru)	True

Технологический раздел

Выбор языка программирования

В качестве языка программирования был выбран **Python 3.7** по следующим причинам:

1. все члены команды обладали минимальным опытом работы с данным языком программирования;
2. его популярность позволила избавиться от долгого решения нетривиальных проблем в коде, от проблем в вопросе развертывания приложения;
3. наличие большого количества проверенных библиотек (стандартных и сторонних), позволило, используя лучшие практики разработки, получить работающий продукт в кратчайшие сроки.

Выбор платформы для развёртывания приложения

Для развертывания приложения была выбрана PaaS-платформа **Heroku**. Этот выбор был обусловлен поддержкой «из коробки» выбранного языка программирования, простотой подключения базы данных, простотой сборки приложения, низкой ценой.

Выбор базы данных

Для хранения данных приложения использовалась композиция следующих решений:

1. данные о зарегистрированных пользователях, материалах в базе, их оценках хранятся в реляционной базе данных **PostgreSQL** (предоставляемой Heroku);
2. последовательность сообщений бота, введенные пользователем в процессе регистрации или загрузки нового материала данные сохраняются в key-value хранилище **Redis** (также предоставляемым Heroku);
3. файлы, загружаемые пользователем, сохраняются в хранилище **Telegram**, доступ к которым осуществляется через персистентный уникальный идентификатор.

Выбор библиотек

Для разработки приложения были выбраны библиотеки:

1. **pyTelegramBotAPI** предоставляет абстракцию над Telegram API в виде идиоматических для Python приемов к написанию кода;
2. **sqlalchemy** предоставляет работу с реляционными СУБД с применением технологии ORM — связыванию БД с концепциями ООП;
3. **psycopg2-binary** в качестве драйвера для работы с БД PostgreSQL;
4. **redis** в качестве драйвера для работы с БД Redis;
5. **pytest** для организации модульного тестирования;
6. **email_validator** для работы с электронными письмами при регистрации пользователей.

Реализация тестирования

Для тестирования некоторых частей программы (верификация и форматирование данных) использовались модульные тесты, запуск которых осуществляется при помощи библиотеки **pytest**. Функциональное тестирование и поиск проблем, ведущих к некорректной работе программы осуществлялся тестировщиками.

Модель разработки

В качестве модели разработки в условиях сжатых сроков была выбрана инкрементальная модель разработки.

Развертывание

Для развертывания используются готовые средства, предоставляемые Heroku. Для разного принципа работы при локальной разработке и работе приложения на удаленном сервере была реализована прослойка. Принципы ее работы:

1. при запуске приложения на удаленном сервере используется механизм webhook для взаимодействия с Telegram (сервер ожидает POST-запросы с обновлениями);
2. при локальном запуске используется механизм long polling (локальный сервер сам запрашивает новые данные). При этом веб-хук удаляется при локальном запуске и вновь устанавливается при остановке локального сервера

Такая модель позволила совместить быстрое тестирование написанного кода с возможностью поиска проблем в программе в любое время любым членом команды.

Реализация. Характеристики качества.

Практичность. Взаимодействие с пользователем происходит по заранее установленным предсказуемым шаблонам, включающим подсказки на каждом этапе выполнения команд пользователя.

Функциональность. Требуемая в ТЗ функциональность (верификация пользователя, просмотр материалов, его оценка, загрузка собственных материалов) была реализована в полном объеме.

Сопровождаемость. В реализации программы были использованы стандартные практики и методы разработки приложений подобного рода (в том числе благодаря использованию широко популярных открытых библиотек), что делает код предсказуемым, а сопровождаемость программы простой.

Надежность. Программа использует PaaS, что обеспечивает высокую надежность работы серверов. По состоянию на 8 июня 2020 г. платформа Heroku обеспечивает время бесперебойной работы в 99.999972%. Программа была протестирована на наличие уязвимостей, найденные проблемы были устранены. На текущий момент тривиальные способы вызвать некорректную работу программы исключены.

Эффективность. Использование проверенных библиотек и сервисов позволило добиться быстрого выполнения запросов пользователей.

Мобильность. Программа использует в качестве платформы взаимодействия с пользователем мессенджер Telegram, что обеспечивает доступность на любых платформах, на которых доступен мессенджер. На данный момент это Android, iOS, Windows Phone, Windows, macOS и Linux.

Заключение

В ходе практики была в полном объеме реализована поставленная задача. Выполнение поставленной задачи состояло из шагов:

1. Формулировка задачи, её декомпозиция;
2. Оформление формального технического задания;
3. Распределение ролей в команде;
4. Выбор технологий для реализации;
5. Реализация программы;
6. Отладка программы;

Все шаги были успешно выполнены и завершены в срок. Полученный программный продукт соответствует заявленным к нему требованиям.

В ходе выполнения практики был получен опыт командной разработки, усовершенствованы знания в работе с системой контроля версий Git и языке программирования Python.