МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Кафедра 319 «Системы интеллектуального мониторинга»

**КУРСОВАЯ РАБОТА**

по дисциплине «Технология разработки программного обеспечения»

**«Проектирование и разработка веб-приложения классификации новостей с применением методов машинного обучения»**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | | |  | Симдянкин В.Д. |
| Группа | | | М3О – 221М – 19 |  |
| Руководитель | | |  | Полицына Е.В. |
| Оценка |  | Дата защиты «\_\_\_\_» декабря 2020 г. | | |

**Москва 2020**

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)»

Кафедра 319 «Системы интеллектуального мониторинга»

**З А Д А Н И Е**

на курсовую работу по дисциплине

Технология разработки программного обеспечения

Студент М3О – 221М – 19 Симдянкин Владимир Дмитриевич

(№ группы, Ф. И. О.)

Тема Проектирование и разработка веб-приложения классификации новостей с применением методов машинного обучения

Перечень вопросов, подлежащих разработке в курсовой работе:

1. Проектирование архитектуры системы и выбор средств разработки
2. Проектирование и реализация ядра системы – классификатора
3. Проектирование и реализация фронтенда
4. Проектирование и реализация бекенда

Рекомендуемая литература

1. Django [Электронный ресурс]. – Режим доступа: https://docs.djangoproject.com/. – Заглавие с экрана. – (Дата обращения 25.12.20).
2. Scikit-learn [Электронный ресурс]. – Режим доступа: https://scikit-learn.org/. – Заглавие с экрана. – (Дата обращения 25.12.20).

Задание выдано « 12 » сентября 2020 г.

Руководитель Полицына Екатерина Валерьевна, к.т.н., доцент кафедры 319 МАИ

(Ф. И. О., должность, подпись)

Студент Симдянкин В.Д.

(подпись)

**СОДЕРЖАНИЕ**

[1 ОПИСАНИЕ ВОЗМОЖНОСТЕЙ ПРИЛОЖЕНИЯ 4](#_Toc59836447)

[1.1 Общие сведения 4](#_Toc59836448)

[1.2 Назначение и цель 4](#_Toc59836449)

[2 ТРЕБОВАНИЯ К ПРИЛОЖЕНИЮ 5](#_Toc59836450)

[2.1 Функциональные требования 5](#_Toc59836451)

[2.2 Нефункциональные требования 5](#_Toc59836452)

[3 АРХИТЕКТУРА СИСТЕМЫ 6](#_Toc59836453)

[3.1 Схема архитектуры 6](#_Toc59836454)

[3.2 Протоколы взаимодействия 6](#_Toc59836455)

[3.3 Используемые технологии 13](#_Toc59836456)

[3.4 Паттерны проектирования 14](#_Toc59836457)

[4 ОПИСАНИЕ КЛАССИФИКАТОРА 15](#_Toc59836458)

[4.1 Классы 15](#_Toc59836459)

[4.2 Вектор признаков 15](#_Toc59836460)

[4.3 Модель машинного обучения классификатора 15](#_Toc59836461)

[4.4 Обучающая и тестовая коллекция документов 15](#_Toc59836462)

[4.5 Оценка точности классификации 16](#_Toc59836463)

[5 ОПИСАНИЕ ИНФРАСТРУКТУРЫ РАЗРАБОТКИ 17](#_Toc59836464)

[5.1 Система контроля версий 17](#_Toc59836465)

[5.2 Сборка и развертывание приложения 17](#_Toc59836466)

[5.3 Обновление модели классификатора 17](#_Toc59836467)

[6 РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ 18](#_Toc59836468)

[7 ИНТЕРФЕЙС И ВОЗМОЖНОСТИ СИСТЕМЫ 19](#_Toc59836469)

[8 АНАЛИЗ РЕЗУЛЬТАТОВ 23](#_Toc59836470)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 24](#_Toc59836471)

# ОПИСАНИЕ ВОЗМОЖНОСТЕЙ ПРИЛОЖЕНИЯ

## Общие сведения

Веб-приложение классификации новостных статей представляет веб-сервис, который позволяет классифицировать новости и управлять коллекцией собранных новостей с информационного портала «www.techcult.ru».

## Назначение и цель

С помощью веб-приложения возможно определить, к какой категории относится новостная статья, введенная пользователем. Сервис предоставляет возможность просмотреть собранные новостные статьи, которые представляли собой входные данные для обучения классификатора с возможностью их фильтрации. Также веб-приложение позволяет удалять, изменять собранные новостные статьи. У веб-приложения есть возможность добавить новостную статью в коллекцию.

# ТРЕБОВАНИЯ К ПРИЛОЖЕНИЮ

## Функциональные требования

Функциональные требования к системе:

* добавление новостной статьи;
* удаление новостной статьи;
* изменение новостной статьи;
* вывод новостных статей по страницам;
* фильтрация новостных статей по категориям и диапазону даты публикации;
* классификация текста по категории новостной статьи с помощью методов машинного обучения;
* обновление модели классификатора.
* поиск по содержимому в заголовке и тексте статьи

## Нефункциональные требования

Система должна быть реализована на базе клиент-серверной архитектуры и представлять из себя веб-сервис для классификации новостных статей с графическим пользовательским интерфейсом.

Веб-приложение должно быть разработано с помощью следующих технических средств:

* backend должен быть реализован на языке программирования Python с помощью фреймворка Django;
* СУБД SQlite;
* frontend должен быть реализован с помощью HTML и языка программирования Javascript;
* проект и вся необходимая документация и данные по нему должны храниться в системе контроля версий git на сайте сервиса GitHub.

Система должна быть отказоустойчивой при работе пользователя с системой.

# АРХИТЕКТУРА СИСТЕМЫ

## Схема архитектуры

Архитектура разработанного приложения изображена на рисунке 1.

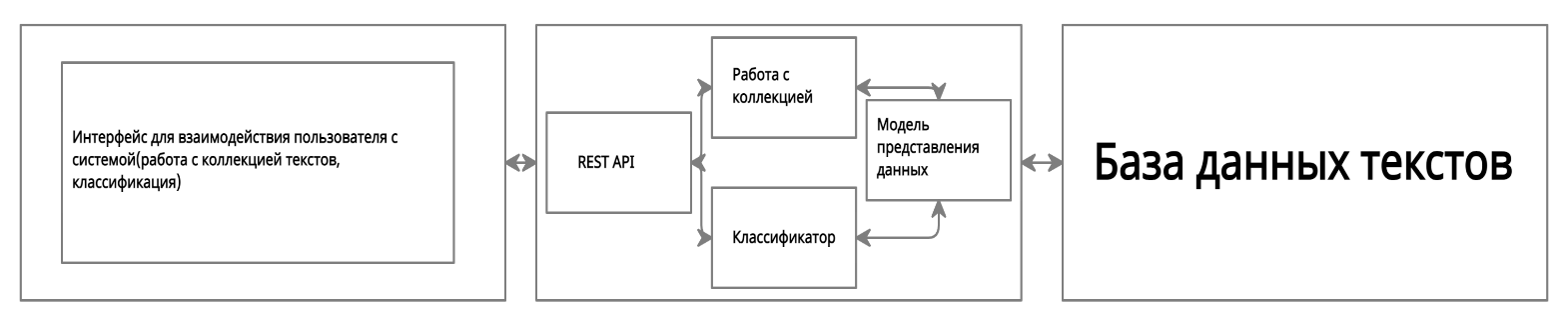


Рисунок 1 – Архитектура приложения

Frontend часть приложения, представляющая собой веб-интерфейс для ввода текста новости и работы с коллекцией текстов взаимодействует с backend частью приложения посредством HTTP запросов и ответов в теле которых присутствует JSON. Backend часть приложения состоит из базы данных, в которой хранится информация о новостных статьях, классификатора и модуля для работы с коллекцией новостей. Классификатор и модуль для работы с коллекцией новостей взаимодействует с базой данных с помощью API для взаимодействия с БД.

## Протоколы взаимодействия

Разработанная система является RESTful веб-сервисом. Используемые в системе API:

* Страница классификатора.

GET /

Возвращает html-документ страницы классификатора.

* Страница для работы с коллекцией текстов.

GET /collection

Возвращает html-документ страницы для работы с коллекцией текстов.

* Добавление статьи в коллекцию.

GET /add-article

Возвращает html-документ страницы для добавления текста в коллекцию.

* Страница новостной статьи.

GET /collection/{id}

Возвращает html документ страницы новостной статьи

* Страница изменения новостной статьи.

GET /collection/update/{id}

Возвращает html документ страницы для изменения новостной статьи.

* Обучение и сохранение модели классификатора.

POST /api/fit-and-save-model

Возвращает HTML ответ со статусом 200 при успешном обучении и сохранении модели классификатора.

* Классификация новостной статьи.

POST /api/classification

На вход принимает http запрос с json в котором хранится текст статьи и возвращает http ответ в теле которого json в котором хранится название категории.

* Добавление новостной статьи.

POST /api/collection/add-article

Принимает hhtp запрос в теле которого json который содержит информацию о добавляемой статьи. Возвращает http ответ со статусом 200 в случае успешного добавления статьи.

* Вывод новостной статьи.

GET /api/collection/{id}

Возвращает http ответ в теле которого json содержащий информацию о статьях.

* Изменение новостной статьи.

PUT /api/collection/{id}

Принимает http запрос в теле которого json с новой информацией о статье. Возвращает http ответ со статусом 200 при успешном изменении новостной статьи.

* Удаление новостной статьи.

DELETE /api/collection/{id}

Возвращает http ответ со статусом 200 при успешном удалении статьи.

* Вывод коллекции текстов по странично.

GET /api/collection/page/{page}

Возвращает html ответ в теле которого json который содержит следующую информацию: длину всей выборки , количество статей на странице, список новостных статей с их кратким описанием. На вход поступает непосредственно номер страницы.

* Фильтрация коллекции текстов по категории.

GET /api/collection/category/{category}/page/{page}

Возвращает html ответ в теле которого json который содержит следующую информацию: длину всей выборки , количество статей на странице, список новостных статей с их кратким описанием. На вход поступает непосредственно категория и номер страницы.

* Фильтрация коллекции текстов по дате публикации.

GET /api/collection/dates/{leftDate}/{rightDate}/page/{page}

Возвращает html ответ в теле которого json который содержит следующую информацию: длину всей выборки , количество статей на странице, список новостных статей с их кратким описанием. На вход поступает непосредственно левая граница диапазона , правая граница диапазона и номер страницы.

* Поиск по содержимому в коллекции текстов.

GET /api/collection/search/{searchLine}/page/{page}

Возвращает html ответ в теле которого json который содержит следующую информацию: длину всей выборки , количество статей на странице, список новостных статей с их кратким описанием. На вход поступает поисковая строка запроса и непосредственно номер страницы.

## Используемые технологии

Для реализации серверной части использовался язык программирования Python, а также фреймворк Django.

В качестве СУБД используется SQlite.

Для создания модели классификатора используется модуль языка Python scikit-learn, для получения стоп слов использовался модуль языка Python NLTK.

В качестве средства реализации пользовательского интерфейса используется HTML и Javascript.

В качестве системы контроля версий использовался сервис GitHub, представляющий из себя графический интерфейс для технологии git – распределенной системы управления версиями.

## Паттерны проектирования

При разработке использовался паттерн MVC, который позволяет разрабатывать бизнес-логику и визуальное представление отдельно.

# ОПИСАНИЕ КЛАССИФИКАТОРА

## Классы

Классификатор определяет следующие классы:

* наука;
* гаджеты;
* роботы;
* техника;
* космос;

## Вектор признаков

В качестве модели представления текста новостных статей использовался метод bag-of-words. При этом проводилась начальная подготовка текста: удалялись лишние неинформативные символы, а также различные стоп-слова. Вектором признаков является масштабированное отношение количества вхождений слова среди всех новостей на количество слов в тексте по методу tf-idf.

## Модель машинного обучения классификатора

В качестве модели машинного обучения был выбран метод опорных векторов.

## Обучающая и тестовая коллекция документов

В качестве обучающей коллекции были взяты 3448 новостных статей.

В качестве тестовой коллекции были взяты 1150 новостных статей из общей коллекции текстов.

## Оценка точности классификации

Оценка точности разработанного классификатора оценивалась по нескольким критериям: точности (precision), полноте (recall), F-мере и точности (accuracy).

Точность (accuracy) классификации равна 89%. Оценки точности по категориям представлены в таблице 1.

Таблица 1 – Оценка точности обученного классификатора по категориям

|  |  |  |  |
| --- | --- | --- | --- |
| Категория | Точность (precision) | Полнота (recall) | F-мера |
| Наука | 0.88 | 0.95 | 0.91 |
| Гаджеты | 0.91 | 0.88 | 0.89 |
| Роботы | 0.93 | 0.81 | 0.86 |
| Техника | 0.87 | 0.85 | 0.86 |
| Космос | 0.88 | 0.91 | 0.89 |
| Взвешенное среднее | 0.89 | 0.89 | 0.89 |

# ОПИСАНИЕ ИНФРАСТРУКТУРЫ РАЗРАБОТКИ

## Система контроля версий

Проектирование и реализация приложения велась локально на компьютере разработчика при помощи Microsoft Visual Studio 2017. При разработке системы использовалась система контроля версий git. Исходный код хранится локально на компьютере разработчика, и удаленно, на сайте сервиса GitHub: https://github.com/Genrovsky/ClassificatorArticle

## Сборка и развертывание приложения

Для успешного запуска веб-приложения необходимо наличие интерпретатора Python 3.0 (64 бит) или выше, а также операционной системы Windows 7 и выше. Необходимо скачать архив с проектом, разархивировать папку ArticleClassificationWebService. Далее через командную строку активировать виртуальное окружение python c помощь команд:

* cd {pathTo}\ ArticleClassificationWebService;
* pythonEnv\Scripts\activate.bat.

После активации окружения, достаточно запустить web-приложение по данной команде: python manage.py runserver 80.

## Обновление модели классификатора

Для обновления модели классификатора, достаточно послать POST запрос по ссылке /api/fit-and-save-model.

# РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Для тестирования системы применялось ручное тестирование. Тестирование проводилось методом черного ящика.

В процессе тестирования использовался веб-браузер Google Chrome. Краткое описание тестов и результаты по ним представлены в таблице 2.

Таблица 2 – Краткое описание тестов и результаты тестирования.

|  |  |
| --- | --- |
| Краткое описание тестов | Результат тестирования |
| Ввод текста новостной статьи для классификации и последующая его классификация | Был выведен корректный результат классификации текста |
| Ввод пустого текста новостной статьи для классификации и последующая его классификация | Было выведено сообщение о том, что был введен пустой текст |
| Добавление новой новостной статьи | Было выведено сообщение о том, что статья успешно добавлена. |
| Добавление новой новостной статьи с некорректными параметрами | Было выведено сообщение о том, что параметры некорректны |
| Изменение новостной статьи | Было выведено сообщение о том, что статья успешно изменена |
| Изменение новостной статьи с некорректными параметрами | Было выведено сообщение о том, что параметры некорректны |
| Удаление новостной статьи | Было выведено сообщение о том, что статья удалена |
| Вывод страницы с новостными статьями | При загрузке страницы была выведена первая страница с коллекцией новостей |
| Вывод новостной статьи | При загрузке страницы с новостной статье, информация по ней была успешно отображена |
| Вывод несуществующей новостной статьи | Было выведено сообщение о том, что такой статьи не существует |

# ИНТЕРФЕЙС И ВОЗМОЖНОСТИ СИСТЕМЫ

Система предоставляет пользователю классифицировать текст и просматривать коллекцию новостных статей, а также производить операции с отдельными новостями.

Классификация текста происходит при нажатии кнопки «Классифицировать». Интерфейс страницы классификации текстов представлен на рисунке 2.

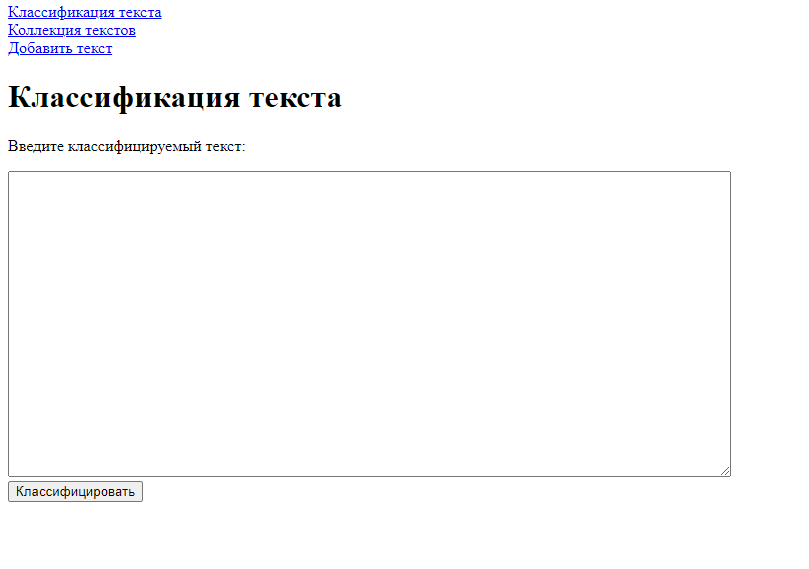


Рисунок 2 – Интерфейс классификации новостной статьи

Для фильтрации коллекции текстов необходимо задать желаемые параметры и нажать кнопку «Найти». Интерфейс для работы с коллекцией новостных статей представлен на рисунке 3.

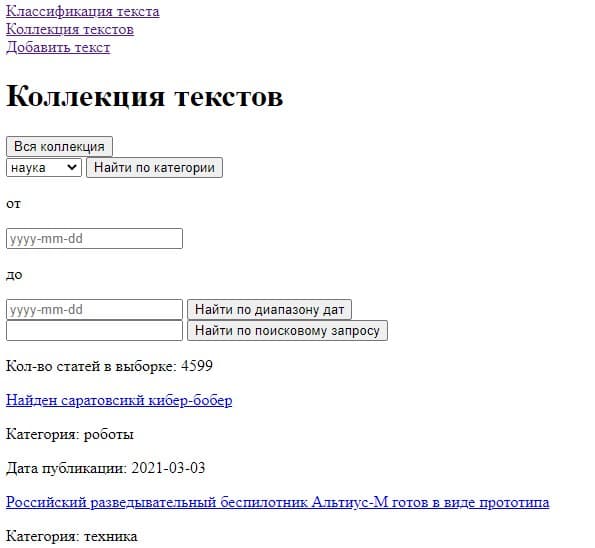


Рисунок 3 – Интерфейс для работы с коллекцией новостей

Найденные новостные статьи можно просматривать постранично, переключаясь между страницами. Для изменения используется кнопка «Редактировать», а для удаления «Удалить». Интерфейс для отображения коллекции новостей представлен на рисунке 4. Интерфейс для работы с отдельной новостью представлен на рисунке 5.

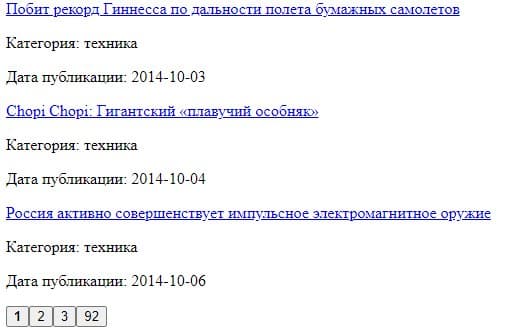


Рисунок 4 – Интерфейс для отображения коллекции новостей



Рисунок 5 – Интерфейс для работы с отдельной новостью

Для добавления новой статьи необходимо ввести данные о ней в форму, затем нажать на кнопку «Добавить статью». Интерфейс для добавления новой статьи представлен на рисунке 6.

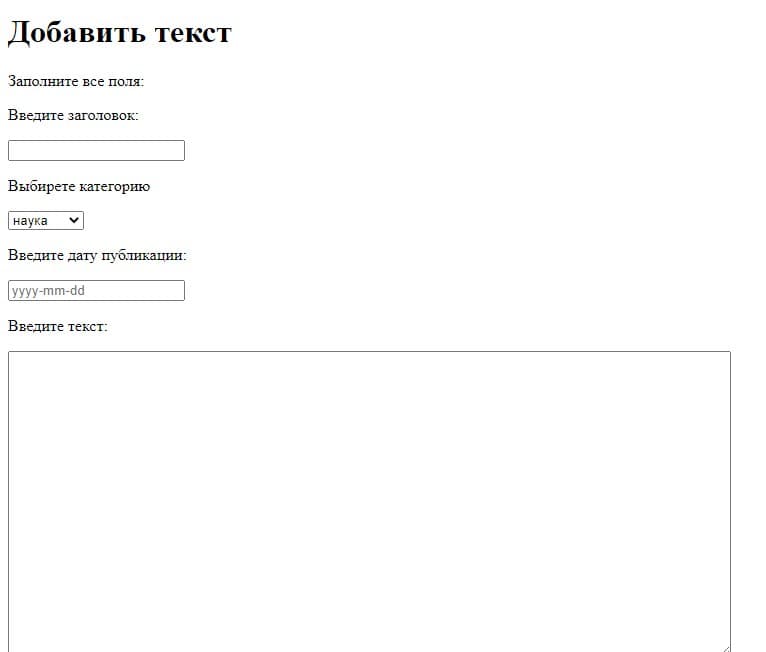


Рисунок 6 – Интерфейс для добавления новой статьи

# АНАЛИЗ РЕЗУЛЬТАТОВ

Таким образом, был спроектировано и реализовано web-приложение для классификации новостных статей и работы с их коллекцией.

Разработанное web-приложение не требует больших затрат на поддержку, сопровождение и обновление.

Так же имеется возможность добавить новые web-страницы и функционал, но для изменения уже разработанного функционала потребуется приложить достаточно усилий. При изменении структуры новостной статьи потребуется изменить большое количество программных единиц.

Что касается отказоустойчивости, web-приложение защищено от ввода некорректных данных введенных пользователем. Но при изменении структуры приложения, а также различных метафайлов возможно нарушение работы приложения.

Классификатор имеют хорошую точность распознавания категорий, а именно 89% процентов. Но в дальнейшем возможно повысить точность работы путем лучшей подготовки текста.

Frontend часть преимущественно разрабатывалась для работы с браузером Chrome , поэтому возможно могут возникнуть трудности при работе с другими браузерами.

Дизайн приложения оставляет желать лучшего , было бы вполне правильным решением обновить его в дальнейшем в духе с последними тенденциями разработки интерфейсов.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Techcult [Электронный ресурс]. – Режим доступа: https://www.techcult.ru/. – Заглавие с экрана. – (Дата обращения: 25.12.20).
2. Python [Электронный ресурс]. – Режим доступа: https://www.python.org/. – Заглавие с экрана. – (Дата обращения: 25.12.20).
3. Django [Электронный ресурс]. – Режим доступа: https://docs.djangoproject.com/. – Заглавие с экрана. – (Дата обращения 25.12.20).
4. Scikit-learn [Электронный ресурс]. – Режим доступа: https://scikit-learn.org/. – Заглавие с экрана. – (Дата обращения 25.12.20).
5. Htmlbook [Электронный ресурс]. – Режим доступа: http://htmlbook.ru/. – Заглавие с экрана. – (Дата обращения 25.12.20).
6. Современный учебник javascript [Электронный ресурс]. – Режим доступа: https://learn.javascript.ru/. – Заглавие с экрана. – (Дата обращения 25.12.20).