МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

Шейх Солтани Мохаммаду Али

Создание системы, повышающей интерес читателей к чтению с помощью методов машинного обучения

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к дипломному проекту

5В070400 – Вычислительная техника и программное обеспечение

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт автоматики и информационных технологий

Кафедра «Программная инженерия»

ДОПУЩЕН К ЗАЩИТЕ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«25» 05 2022 г.

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к дипломному проекту

На тему: «Создание системы, повышающей интерес читателей к чтению с помощью методов машинного обучения»

по специальности 5В070400 – Вычислительная техника и программное обеспечение

Выполнил Шейх Солтани М.А.

Научный руководитель
сениор лектор, доктор PhD
_____ М. Сатымбеков
"20" 05 2022 г.

Рецензент
Доктор PhD
И.О. ассоциированного профессора
_____ Н.О. Мекебаев
_____ 2022 г.

Алматы 2022

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Казахский национальный исследовательский технический университет имени К.И.Сатпаева

Институт информационных и телекоммуникационных технологий

Кафедра "Программная инженерия"

5В070400 – Вычислительная техника и программное обеспечение

УТВЕРЖДАЮ
Заведующая кафедрой ПИ
канд. физ-мат. наук, профессор
_____ А.Н. Молдагулова
«_20_» _05_ 2022 г.

ЗАДАНИЕ
на выполнение дипломного проекта

Обучающемуся *Шейх Солтани Мохаммаду Али*
Тема: *Создание системы, повышающей интерес читателей к чтению с помощью методов машинного обучения*
Утверждена *приказом проректора по академической работе* № 429-17/от
"24" 12 2021 г.
Срок сдачи законченного проекта "25" 05 2022г.
Исходные данные к дипломному проекту: *техническое задание, описание основных функций проекта, use-case диаграмма, ER диаграмма.*
Перечень подлежащих разработке в дипломном проекте вопросов:
а) разработка базы данных;
б) разработка сайта с формой опроса;
в) разработка методов рекомендации книг;
г) реализация функции прохождения опроса.
Перечень графического материала (с точным указанием обязательных чертежей): *представлены 20 слайдов презентации.*
Рекомендуемая основная литература: *из 20 наименований.*

ГРАФИК

подготовки дипломного проекта

Наименование разделов, перечень разрабатываемых вопросов	**Сроки представления научному руководителю и консультантам**	**Примечание**
1. Анализ предметной области, разработка технического задания	18.01.2022	Выполнено
2. Выбор технологий для разработки	21.01.2022	Выполнено
3. Разработка базы данных	27.01.2022	Выполнено
4. Разработки логики взаимодействия	08.02.2022	Выполнено
5. Разработка функционала системы	25.03.2022	Выполнено
6. Тестирование и оптимизация	12.04.2022	Выполнено
7. Написание пояснительной записки к дипломному проекту	29.04.2022	Выполнено

Подписи

консультантов и нормоконтролера на законченный дипломный проект с указанием относящихся к ним разделов проекта

Наименования разделов	**Консультанты, И.О.Ф. (уч. степень, звание)**	**Дата подписания**	**Подпись**
Нормоконтролер	Жекамбаева М.Н. Доктор Ph.D., ассоциированный профессор	20.05.22	
Программное обеспечение	Марғұлан Қ. Магистр техн.наук, лектор	19.05.22	

Научный руководитель _____ Сатымбеков М.

Задание принял к исполнению обучающийся _____ Шейх Солтани М.А.

Дата «17» 11 2021г.

АННОТАЦИЯ

Данный дипломный проект посвящен разработке сайта при помощи фреймворка Python Django.Сайт позволит посетителям получать рекомендации по прочтению подходящих им книг. Рекомендации будут выдаваться на основе личных предпочтений пользователя в литературе.

Для реализации используется язык программирования Python, также фреймворк для веб приложений Django. HTML, CSS, JavaScript используются для создания сайта. Также применяется реляционная база данных для хранения записей, которая встроена в Python.

В качестве рекомендательной системы служит проверка косинусной сходимости между различными книгами.

АҢДАТПА

Бұл дипломдық жоба Python Django шеңберін қолдана отырып сайтты дамытуға арналған.Сайт келушілерге өздеріне сәйкес кітаптарды оқу бойынша ұсыныстар алуға мүмкіндік береді. Ұсыныстар пайдаланушының әдебиеттегі жеке қалауы негізінде беріледі.

Іске асыру үшін Python бағдарламалау тілі, сонымен қатар Django веб-қосымшаларына арналған шеңбер қолданылады. HTML, CSS, JavaScript веб-сайт құру үшін қолданылады. Сондай-ақ, Python-ға енгізілген SQL тіліндегі жазбаларды сақтау үшін реляциялық мәліметтер базасы қолданылады.

Ұсыныс жүйесі ретінде әр түрлі кітаптар арасындағы косинустық конвергенцияны тексеру қолданылады.

ANNOTAION

This graduation project is dedicated to the development of a website using the Python Django framework.The site will allow visitors to receive recommendations for reading books suitable for them. Recommendations will be issued based on the user's personal preferences in the literature.

The Python programming language is used for implementation, as well as a framework for Django web applications. HTML, CSS, JavaScript are used to create a website. A relational database is also used to store records in SQL, which is built into Python.

The recommendation system is a cosine convergence check between different books.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Низкий уровень чтения населения является на данный момент проблемой во всем мире. Чтение не только развивает эрудицию и кругозор человека, но также предостерегает от нейродегенеративных заболеваний. При чтении книг очень хорошо развиваются память и когнитивные способности человека, что приводит к увеличению продолжительности жизни [1]. Но в эпоху социальных сетей и мессенджеров человеку трудно долго быть сконцентрированным на чтении. Одной из важнейших причин этого является переизбыток книжного рынка огромным количеством неинтересной и некачественной литературы.

Данная работа позволит находить подходящие для человека книги, на основании его ответов в опросе, тем самым увеличивая вовлеченность человека в процесс чтения.

На данный момент рекомендательные системы используются для вовлечения людей в любую сферу жизни: от прослушивания музыки, до доставки еды или просмотра фильмов.

1 Исследовательский раздел

1.1 Цель разработки

Цель данного проекта направлена на создание сервиса, увеличивающего заинтересованность человека в чтении при помощи рекомендации книг на основании личных предпочтений пользователя.

1.2 Термины и сокращения

В таблице 1.1 представлены термины и аббревиатуры, использованные в данной дипломной работе.

Таблица-1.1 – Термины и сокращения

Сокращение или термин	Определение
ПО	Программное обеспечение
Фреймворк	Программное обеспечение, позволяющее упростить разработку и объединение нескольких компонентов проекта [2]
HTML	Hypertext Markup Language. Язык гипертекстовой разметки.
CSS	Cascading Style Sheets Формальный язык позволяющий редактировать внешний вид веб-страницы, написанной при помощи языка разметки
SQL	structured query language. Декларативный язык программирования, которые используется для создания и использования баз данных
БД	База данных
JS	JavaScript, язык программирования
ML	Machine Learning, машинное обучение
MVC	Model-View-Controller, концепция программирования описанная Трюгве Реенскаугом в 1978 году

Продолжение таблицы-1.1

MTV	Mode l- Template - View, модель, используемая в Python Django
Датасет	Структурированная информация в виде таблицы, широко используемая при машинном обучении

1.3 Актуальность проблемы на момент написания дипломной работы

В статье Джефри М. Джонса, основанной на опросе Gallup о количестве прочтенных книг, среднее количество прочитанных книг людьми снизилось с 15.6 до 12.6 в 2016 году. В ходе исследования было опрошено 811 человек, в возрасте старше 18 лет [3].

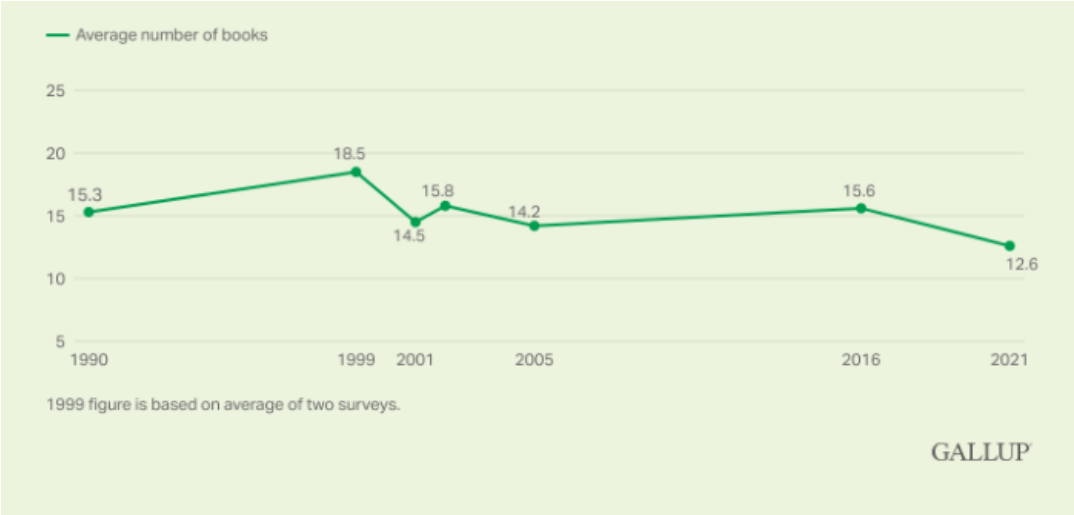

Рисунок-1.1 – График количества прочитанных книг в статье Джефри М. Джонса

Исходя из графика, можно заметить, что в среднем, с 1999 года происходит уменьшение интересов читателей к чтению. Я считаю, что это происходит в том числе из-за избыточности книжного рынка. Было произведено множество исследований в сфере обильности вариантов для выбора и счастьем человека. Чем больше вариантов, между которыми необходимо сделать выбор, тем тяжелее его сделать. Так как в случае предпочтения одного элемента выбора, мы чувствуем потерю остальных вариантов, а также испытываем чувство утерянной возможности наилучшего выбора. Иногда люди предпочитают вовсе отказаться от всех вариантов, чем сделать выбор.

Стоит отметить, что снижение уровня чтения среди населения происходит во всех категориях, но набольшее снижение наблюдается у выпускников колледжей. Если в более пожилом возрасте снижение уровня чтения можно связать с физиологическими проблемами, такими как падение уровня зрения, то в молодом поколении это наиболее связано с использованием большого количества гаджетов, а также увеличения спектра доступных развлечений [4].

	2002-2016	2021	Change
U.S. adults	15.2	12.6	-2.6
Gender			
Men	10.8	9.5	-1.3
Women	19.3	15.7	-3.6
Age			
18-34 years	13.8	13.0	-0.8
35-54 years	14.2	12.5	-1.7
55+ years	16.7	12.0	-4.7
College graduate			
Yes	21.1	14.6	-6.5
No	12.6	11.5	-1.1

GALLUP

Рисунок-1.2 – Изменения количества прочитанных книг, в разбивке по категориям

Второй большой проблемой в чтении является большое количество неинтересной и плохой литературы. Люди начинают читать книгу, но не заканчивают чтение, так как книга не может заинтересовать читателя. Для этого, может быть, множество различных причин, таких как банальность сюжета, утомительный язык повествования или отсутствие новой информации, если речь идет об образовательной литературе.

Решением данных проблем является разработка универсальной системы, которая позволит сократить выбор, а также предложить наиболее интересные книги, для большей вовлеченности читателей. Наиболее интересные книги можно подбирать, основываясь на личных предпочтениях пользователя, а также на основании количества наград, полученных книгой и общем рейтинге книги.

2 Технологический раздел

2.1 Фреймворк Django

Django – это фреймворк, созданный для написания веб-приложений, на основе языка программирования Python [5]. В Django используется шаблон проектирования программ MTV. В данной концепции:

- M соответствует модели, слоя доступа к данным. При помощи этого слоя и происходит взаимодействие со всеми данными, а также можно отследить связь данных между собой.
- T соответствует шаблону, слоя представления данных. При помощи данного слоя происходит отображение данных на экране.
- V соответствует представлению, слоя бизнес-логики. Данный слой является неким связующим звеном между моделью и шаблоном и позволяет получать доступ к данным через модель и отображать ее в соответствии с определенным шаблоном.

2.2 Python

Python – интерпретируемый, объектно-ориентированный высокоуровневый язык программирования, который нашел очень широкое применение во многих областях программирования. Его создателем считается Гвидо ван Россум, разработавший Python в 1991 году [6].

Python хорошо подходит для реализации искусственного интеллекта и машинного обучения благодаря большому количеству имеющихся библиотек, таких как: Pandas, Numpy, Keras, Scikit-learn, SciPy [7].

2.3 HTML

HTML – язык разметки гипертекста, который используется для создания структуры сайта или страницы. Наряду с CSS и JavaScript является одним из трех основных инструментов для создания веб-сайтов [8].

2.4 CSS

CSS – формальный язык, использующийся для создания и редактирования визуальной части веб-страниц, написанных на языке разметки. Преимуществами

являются экономия времени, так как можно прописать один css шаблон для нескольких файлов сразу, а также простота использования [9].

2.5 JavaScript

JavaScript – динамический язык программирования, используемый в веб-разработке, разработке приложений, а также игр [10]. Он позволяет создавать и использовать динамические функции на веб-страницах, которые было бы невозможно сделать только при помощи HTML и CSS.

2.6 Jupyter

Jupyter – интерактивная веб-среда разработки. Его интерфейс позволяет организовать рабочий процесс более удобно, так как можно разбить код по отдельным ячейкам, а также добавлять специальные ячейки с комментариями [11]. Основной областью применения является машинное обучение и визуализация данных.

2.7 Среда разработки

Средой разработки называется ПО, созданное для разработки приложений, и облегчающее их создание. Среды разработки бывают как универсальные, так и созданные для определенного языка программирования. В качестве примера можно привести PyCharm. Это одна из самых популярных сред разработки для языка Python. Одним из главных плюсов является наличие поддержки фреймворка Django. Также он позволяет производить быстрый рефакторинг кода и просмотр документации внутри самого проекта [12]. PyCharm поставляется со множеством модулей, пакетов и инструментов для ускорения разработки на Python

3 Проектная часть

3.1 Коэффициент Отиаи

Бинарная мера сходства, предложенная японским биологом Акирой Отиаи в 1957 году, в дальнейшем обобщенная и нашедшая применение в разнообразных приложениях и за рамками биологии [13]. В анализе данных косинусное сходство является мерой сходства между двумя последовательностями чисел. Для его определения последовательности рассматриваются как векторы во внутреннем пространстве произведений, а сходство по косинусу определяется как косинус угла между ними, то есть точечное произведение векторов, деленное на произведение их длин. Отсюда следует, что косинусное подобие зависит не от величин векторов, а только от их угла. Чем меньше угол, тем меньше расстояние между точками на координатной плоскости. Чем меньше данное расстояние, тем более похожими считаются два сравниваемых элемента. Данный коэффициент таже часто называют косинусным коэффициентом сходимости.

3.2 Архитектура проекта

Проект реализован в виде web-сайта для того, чтобы любой желающий мог зайти на данный сайт и получить рекомендацию по чтению следующей книги на основании своих предпочтений.

Пользователь заходит на сайт и заполняет форму. Затем все данные о выбранных в форме жанрах отправляются в модель, которая на основе коэффициента Отиаи выбирает схожие книги. Также, форма, заполненная пользователем напрямую связана с реляционной базой данных SQLite. После того, как модель выберет подходящую книгу для пользователя, вся информация записывается в базу данных. Таким образом в последствии можно получить информацию о полученных рекомендациях для каждого пользователя. После этого на страницу web-сайта выводится название книги, которую ему стоит прочесть.

3.3 Структура кода проекта

В данном проекте я использую фреймворк для Python – Django. В нем реализуется схема разделения данных MTV, которая во многом похожа на более популярную схему MVC [14].

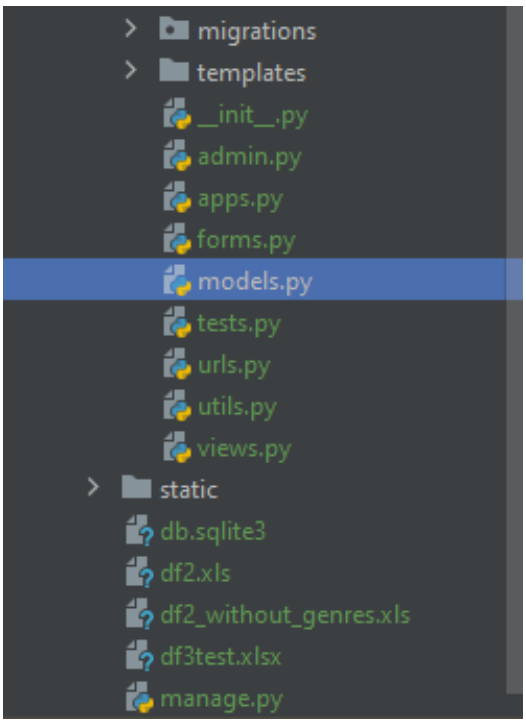

Рисунок-3.1 – Основные файлы в структуре проекта

В файле models.py хранится класс с моделью данных. Там заполняется информация о модели и всех ее полях, которая в последующем будет использоваться в БД.

Файл urls.py хранит в себе саму рекомендательную систему. Именно этот файл отвечает за проверку коэффициента Отиаи и ранжирования книг по количеству наград и рейтингу.

Urls.py хранит в себе маршрутизацию для любого проекта. Здесь файлы связываются между собой и указываются дополнительные ссылки для маршрутизации.

Файл forms.py хранит в себе информацию о форме, которая выводится на главной странице.

```
widgets = {
    'firstname': forms.TextInput(attrs = {'class':'form-control nm'}),
    'lastname': forms.TextInput(attrs = {'class':'form-control nm'}),
    'Fiction': forms.CheckboxInput(attrs= {'class' : 'form-check-input'}),
    'Romance': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
    'Fantasy': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
    'Young_Adult': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
    'Contemporary': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
```

Рисунок-3.2 – Создание формы в forms.py

Для каждого элемента формы создается соответствующее поле, которое указывает на тип элемента.

В моем проекте используются только 2 типа данных в форме:

- TextInput – позволяет ввести небольшие текстовые значения, такие как имя и фамилия пользователя;
- CheckboxInput – позволяет передать значение типа bool, чтобы проверить интерес пользователей к отдельным жанрам.

3.4 UML диаграммы

UML – это способ визуализации работы системы при помощи различных диаграмм [15]. UML расшифровывается, как унифицированный язык моделирования. В наше время перед созданием любой системы или приложения производится ее моделирование, и для данных целей применяют различные виды UML диаграмм. Главным плюсом является наглядность, позволяющая легко разобраться в структуре всего проекта.

3.4.1 Use-case диаграмма

Диаграмма Use-case демонстрирует функциональность системы для различных типов пользователей, таких как посетитель и администратор [16]. Данная диаграмма показывает все действия, которые может сделать пользователь, а также все действия, которые производит сама система.

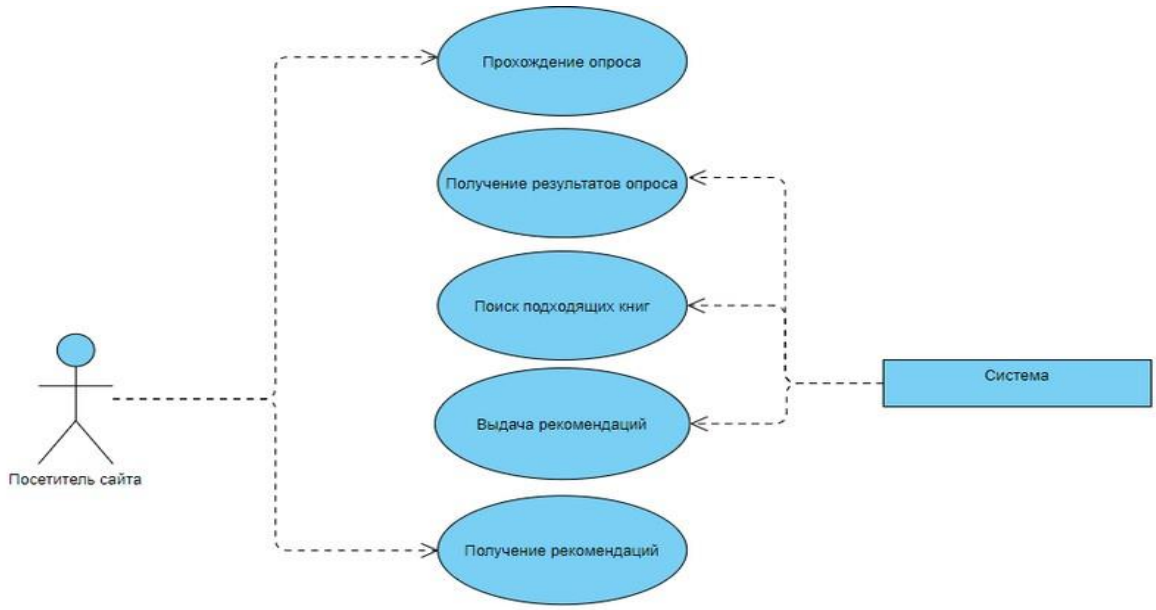

Рисунок-3.3 – Диаграмма использования

Прохождение опроса является основной функцией пользователя, так как именно на основании этого опроса система будет работать. Прохождение опроса включает в себя ввод данных о своих любимых жанрах в литературе, а также введение фамилии и имени.

Получение результатов опроса является первым действием системы, и оно реализовано при помощи встроенных возможностей Python Django.

Поиск подходящих книг является главной функцией, которую выполняет система. Поиск производится на основании схожести книг по жанрам, а также учитывает рейтинг книги, процент людей, кому понравилась книга, и количество плаченных ей наград.

Выдача рекомендаций производится при помощи записи информации о пользователе в базу данных, а затем вывода информации на экран пользователя.

Получение рекомендаций является ключевой функцией, ради которой пользователь заходил на сайт и проходил опрос. В заключении он может получить название книги, которая наиболее подходит к его предпочтениям.

3.4.2 ER диаграмма

В данной работе, помимо основной таблицы с пользователями также используются книжный датасет, из которого берется вся информация о книгах.

Таблица с пользователями хранит информацию, которую люди добавили при прохождении опроса. Primary key является поле ID. Для каждой записи автоматически генерируется ID для того, чтобы в последствии можно было обращаться при помощи него к записи. Также последним полем является поле, с рекомендуемой книгой.

Датасет “Best Books Ever Dataset” состоит из 25 полей [17]. Не все поля заполнены на 100%. К примеру, поле SETTING заполнено лишь у 22% пользователей. Для рекомендательной системы использовались поля, со следующим содержанием:

- Title – содержит название книги;
- Series – показывает, является ли книга частью книжной серии, или выступает как отдельное произведение;
- Author – поле, в котором записан автор произведения;
- Rating – рейтинг книги “Global goodreads rating”;
- Description – краткое описание книги;
- Language – язык, на котором написано произведение;
- Genres – содержит все жанры, которым соответствует книга;
- Awards – содержит все награды, которых удостоилось произведение;
- LikedPercent – поле, демонстрирующее интерес читателей к данной книге на основании платформы GoodReads.

Users		
PK	ID	INTEGER
	FIRSTNAME	VARCHAR(150)
	LASTNAME	VARCHAR(150)
	CREATED_AT	DATETIME
	UPDATED_AT	DATETIME
	FICTION	BOOL
	ROMANCE	BOOL
	FANTASY	BOOL
	YOUNG_ADULT	BOOL
	CONTEMPORARY	BOOL
	NONFICTION	BOOL
	ADULT	BOOL
	NOVELS	BOOL
	MYSTERY	BOOL
	HISTORICAL_FICTION	BOOL
	CLASSICS	BOOL
	ADVENTURE	BOOL
	HISTORICAL	BOOL
	PARANORMAL	BOOL
	LITERATURE	BOOL
	SCIENCE_FICTION	BOOL
	CHILDRENS	BOOL
	THRILLER	BOOL
	MAGIC	BOOL
	HUMOR	BOOL
	HISTORY	BOOL
	CRIME	BOOL
	CONTEMPORARY_ROMANCE	BOOL
	SUSPENSE	BOOL
	URBAN_FANTASY	BOOL
	MIDDLE_GRADE	BOOL
	CHICK_LIT	BOOL
	SCIENCE_FICTION_FANTASY	BOOL
	SUPERNATURAL	BOOL
	BIOGRAPHY	BOOL
	MYSTERY_THRILLER	BOOL
	PARANORMAL_ROMANCE	BOOL
	HORROR	BOOL
	TEEN	BOOL
	PHILOSOPHY	BOOL
	ADULT_FICTION	BOOL
	SHORT_STORIES	BOOL
	LITERARY_FICTION	BOOL
	BRITISH_LITERATURE	BOOL
	REALISTIC_FICTION	BOOL
	DRAMA	BOOL
	RELIGION	BOOL
	NEW_ADULT	BOOL
	MEMOIR	BOOL
	WAR	BOOL
	TWENTIETH_CENTURY	BOOL
	VAMPIRES	BOOL
	CHRISTIAN	BOOL
	GRAPHIC_NOVELS	BOOL
	AMERICAN	BOOL
	RESULTBOOK	VARCHAR(150)

Books.csv	
PK	BOOKID
	TITLE
	SERIES
	AUTHOR
	RATING
	DESCRIPTION
	LANGUAGE
	ISBN
	GENRES
	CHARACTERS
	BOOKFORMAT
	EDITION
	PAGES
	PUBLISHER
	PUBLISHDATE
	FIRSTPUBLISHDATE
	AWARDS
	NUMRATINGS
	RATINGSBYSTARS
	LIKEDPERCENT
	SETTING
	COVERIMG
	BBESCORE
	BBEVOTES
	PRICE

Рисунок-3.4 – ER - диаграмма

3.5 Сбор данных

Для первоначального сбора данных от пользователей я создал форму в Html файле при помощи встроенных инструментов Django для работы с формами. Также, для создания внешнего вида сайта, я использовал файл CSS и

прописал там соответствующие атрибуты дизайна для классов и отдельных типов элементов. Чтобы пользователям было удобнее заполнять форму, перед каждым элементом типа Input есть надпись, указывающая на данные, которые следует ввести. Данные поля используются на первой странице формы, для заполнения имени и фамилии пользователя.

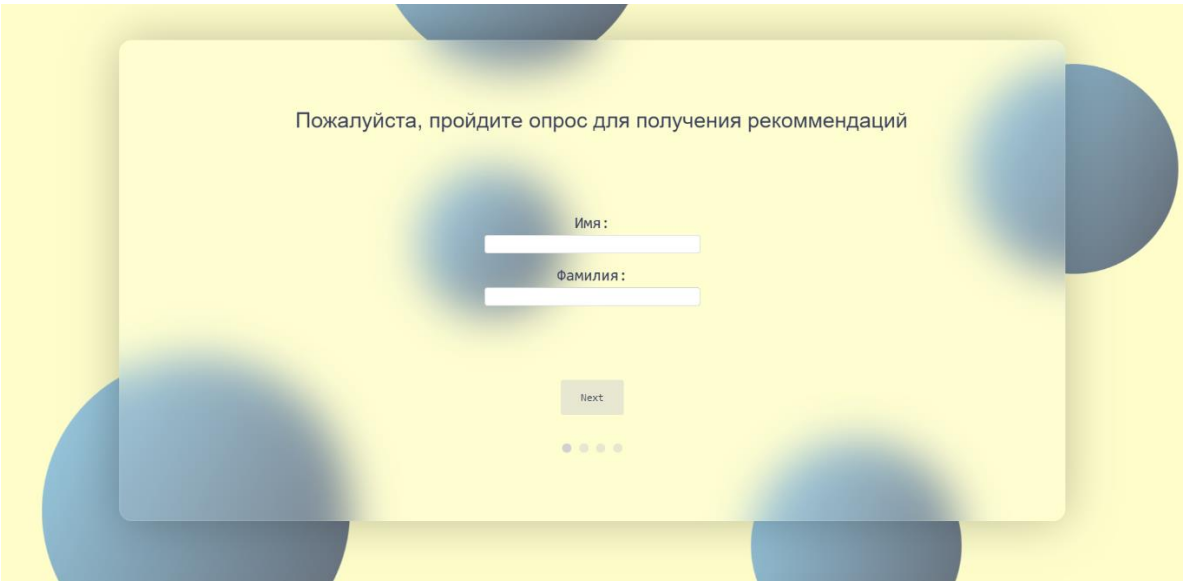

Рисунок-3.5 – Первая страница формы опроса

Также я решил разделить форму на несколько страниц, чтобы не перегружать интерфейс сайта и не выводить огромную форму на первую же страницу. Для этого необходимо было создать две кнопки, для перемещения между страницами. Я реализовал появление кнопки перехода на предыдущую страницу на второй и последующих станицах.

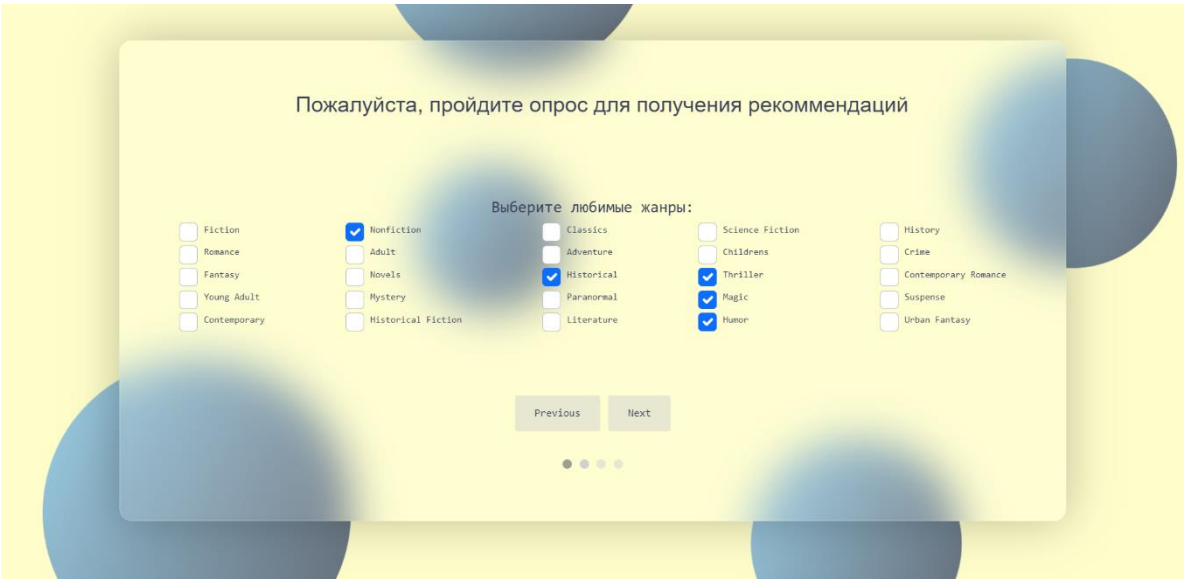

Рисунок-3.6 – Вторая страница формы опроса

На второй и третьей странице формы расположены Input с типом checkbox для выбора любимых жанров пользователя. Каждый элемент передает значения типа Boolean с информацией о необходимых читателю жанрах.

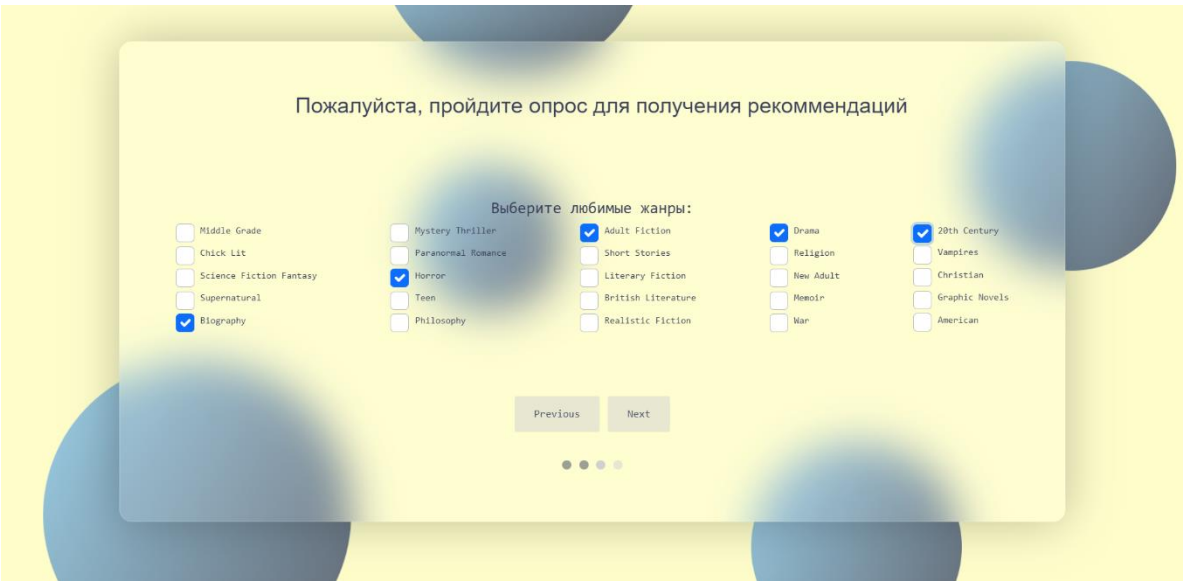

Рисунок-3.7 – Третья страница формы опроса

На четвертой странице появляется input с type="submit" и value="Post", который необходим для отправки данных из формы в модель, а затем и в базу данных. После нажатия на кнопку на этой странице действия пользователя заканчиваются и необходимо немного подождать, для получения результата.

Рисунок-3.8 – Завершающая страница формы опроса

4 Экспериментальный раздел

4.1 Запись информации в базу данных

В данном проекте использована реляционная база данных SQLite, так как данная СУБД обеспечивает высокую скорость работы и является автономной и кроссплатформенной [18]. Для записи данных в нее я реализовал метод Post после прохождения опроса. Функция Recommendation принимает все значения, переданные при нажатии кнопки завершения опроса. Также в данной функции считываются заранее подготовленные excel файлы с датасетом книг и жанрами.

```
def Recommendation(post):
    df2 = pd.read_excel("df2.xls")
    dfgenres = pd.read_excel("df3test.xlsx")
    post = post.dict()
    for k in post:
        if post[k] == 'on':
            #post[k] = 1
            dfgenres[k]=1
```

Рисунок-4.1 – Начало функции Recommendation

Далее происходит замена всех значений, полученных из полей checkbox, так как они передаются в виде “on” вместо 1. После выдачи рекомендаций, происходит запись в базу данных.

	id	firstname	lastna	created at	updated at	Fiction	Romance	Fantasy	Young Ac	Contemp	Nonfictio
1	1	Ali	Shei...	08.04.2022	08.04.2022	1	0	1	0	0	0
2	2	Asdan	Ma...	2022-04-24 09:44:39.449259	2022-04-24 09:44:39.449259	0	0	0	0	0	0
3	3	Konstantin	Bor...	2022-04-24 10:44:24.118744	2022-04-24 10:44:24.118744	0	0	0	0	0	1
4	4	Grisha	Kor...	2022-04-24 12:11:38.023006	2022-04-24 12:11:38.023006	1	0	0	0	0	0
5	5	Grisha	Bro...	2022-04-24 13:43:50.517768	2022-04-24 13:43:50.517768	1	0	0	0	0	1
6	6	Kirill	Bro...	2022-04-24 13:49:37.811998	2022-04-24 13:49:37.811998	1	0	0	0	0	1
7	7	alex	Bro...	2022-04-24 13:51:20.507761	2022-04-24 13:51:20.507761	1	0	0	0	0	1
8	8	Lil	Bro...	2022-04-25 05:17:10.702867	2022-04-25 05:17:10.702867	1	0	0	0	0	0
9	9	Grisha	Ma...	2022-04-25 05:21:50.788946	2022-04-25 05:21:50.788946	0	1	1	0	0	0
10	10	Asdan	Bro...	2022-04-25 05:23:23.794769	2022-04-25 05:23:23.794769	1	0	0	0	0	0
11	11	Asdan	Bro...	2022-04-25 06:26:30.618495	2022-04-25 06:26:30.618495	1	0	0	0	0	0
12	12	Grisha	Kor...	2022-04-25 09:13:15.045848	2022-04-25 09:13:15.045848	0	1	0	0	0	0
13	13	Grisha	Kor...	2022-04-25 09:20:28.932358	2022-04-25 09:20:28.932358	0	1	1	0	0	0
14	14	Ali	Shei...	2022-04-25 19:02:47.827102	2022-04-25 19:02:47.827102	1	0	0	0	0	0
15	15	Kirill	Bor...	2022-04-25 19:11:08.597202	2022-04-25 19:11:08.597202	0	0	0	0	0	0

Рисунок-4.2 – Пример данных из таблицы с пользователями

У каждой записи заполняются поля имени, фамилии, выбранных жанров и выданной рекомендации. Помимо этого, реализованы два поля, отвечающие за время создания записи, и время, когда запись в последний раз редактировали.

4.2 Работа с датасетом

Для данного проекта я использовал датасет “Best Books Ever Dataset”, состоящий из 52477 книг.

Для последующей работы я изменил структуру датасета. Сначала я удалил колонки 'isbn', 'characters', 'bookFormat', 'edition', 'publisher', 'numRatings', 'ratingsByStars', 'setting', 'coverImg', 'bbeScore', 'bbeVotes', 'price', 'publishDate', 'firstPublishDate', так как они не будут использованы при выдаче рекомендаций.

Далее я преобразовал колонку с наградами “awards”. В данном проекте более целесообразно не выводить все награды, а посчитать их количество и тем самым получить дополнительную метрику для рекомендаций. Ведь чем больше наград покажет count, тем больше вероятность, что книга хорошая.

```
df.loc[(df.awards =='[]' ), 'awards'] = 0
df=df.fillna(0)
df['awards_count'] = df['awards']
def normalize(row):
    count = len(re.findall("[(]", str(row)))
    count2 = str(count)
    return count2
df['awards_count'] = df['awards_count'].apply(normalize)
df['awards_count']
```

```
0        44
1        10
2         4
3         0
4        28
         ..
52473     0
52474     0
52475     1
52476     0
52477     0
Name: awards_count, Length: 52478, dtype: object
```

Рисунок-4.3 – Функция для подсчета наград каждой книги

Основной метрикой в рекомендации книг являются жанры, но в данном датасете все данные о жанрах записаны в один столбец. У каждой книги может быть максимум 10 жанров. При помощи функции str.split я сделал матрицу жанров 52477*10 и удалил все скобки и лишние знаки.

```
genresdf = df['genres'].str.split(',',expand=True)
genresdf=genresdf.fillna(0)
genresdf
```

	0	1	2	3	4	5	6	7	8	9
0	['Young Adult'	'Fiction'	'Dystopia'	'Fantasy'	'Science Fiction'	'Romance'	'Adventure'	'Teen'	'Post Apocalyptic'	'Action']
1	['Fantasy'	'Young Adult'	'Fiction'	'Magic'	'Childrens'	'Adventure'	'Audiobook'	'Middle Grade'	'Classics'	'Science Fiction Fantasy']
2	['Classics'	'Fiction'	'Historical Fiction'	'School'	'Literature'	'Young Adult'	'Historical'	'Novels'	'Read For School'	'High School']
3	['Classics'	'Fiction'	'Romance'	'Historical Fiction'	'Literature'	'Historical'	'Novels'	'Historical Romance'	'Classic Literature'	'Adult']
4	['Young Adult'	'Fantasy'	'Romance'	'Vampires'	'Fiction'	'Paranormal'	'Paranormal Romance'	'Supernatural'	'Teen'	'Urban Fantasy']
...	...	...	...	...	...	...	...	...	...	...
52473	['Vampires'	'Paranormal'	'Young Adult'	'Romance'	'Fantasy'	'Paranormal Romance'	'Magic'	'Urban Fantasy'	'New Adult'	'Werewolves']
52474	['Mystery'	'Young Adult']	0	0	0	0	0	0	0	0
52475	['Fantasy'	'Young Adult'	'Paranormal'	'Angels'	'Romance'	'Demons'	'Supernatural'	'Paranormal Romance'	'Urban Fantasy'	'Fiction']
52476	['Fiction'	'Mystery'	'Historical Fiction'	'Adventure'	'Christian Fiction'	'Historical'	'Religion'	'Suspense'	'Christian'	'Archaeology']
52477	['Lds Fiction'	'Historical Fiction'	'Young Adult'	'Fiction'	'Fantasy'	'Lds'	'Historical'	'Romance'	'Adventure'	'Teen']

Рисунок-4.4 – Преобразованный датасет с жанрами

Затем, при помощи функции append я объединил все столбцы в один, длиной 524780. Данные в столбцах были записаны некорректно, так как перед некоторыми жанрами стоял знак пробела, но я исправил это при помощи функции lstrip(). Далее я посчитал количество вхождений каждого жанра в данном столбце и отсортировал самые часто встречающиеся 50 жанров, так как в датасете есть большое количество жанров, которые относятся только к одной книге. Затем я записал их в excel и транспонировал данную таблицу.

К первым 50 жанрам относятся: Fiction, Romance, Fantasy, Young Adult, Contemporary, Nonfiction, Adult, Novels, Mystery, Historical Fiction, Classics, Adventure, Historical, Paranormal, Literature, Science Fiction, Childrens, Thriller, Magic, Humor, History, Crime, Contemporary Romance, Suspense, Urban Fantasy, Middle Grade, Chick Lit, Science Fiction Fantasy, Supernatural, Biography, Mystery Thriller, Paranormal Romance, Horror, Teen, Philosophy, Adult Fiction, Short Stories, Literary Fiction, British Literature, Realistic Fiction, Drama, Religion, New Adult, Memoir, War, 20th Century, Vampires, Christian, Graphic Novels и American.

```
c
```

```
0         Young Adult
1             Fantasy
2            Classics
3            Classics
4         Young Adult
             ...
52473      Werewolves
52474               0
52475         Fiction
52476     Archaeology
52477            Teen
Length: 524780, dtype: object
```

```
def delSpace(row):
    row = row.lstrip()
    return row
```

```
c=c.apply(delSpace)
```

```
x =c.value_counts()
x
```

```
0                     112439
Fiction                31638
Romance                15495
Fantasy                15046
Young Adult            11869
                        ...
Polyamorous                1
Harlequin Presents         1
Did Not Finish             1
Cultural Heritage          1
Jewellery                  1
Length: 984, dtype: int64
```

Рисунок-4.5 – Алгоритм получения 50 самых популярных жанров

Для данных жанров я создал отдельную колонку и заполнил ее значениями 0 и 1 для каждой книги в зависимости от того, к каким жанрам относится данное произведение.

```
for index, row in df2.iterrows():
    print(index)
    for i in top_genres_names:
        if(("'"+i+"'") in row['genres']):
            row[i]=1
            df2[i][index]=1
        else:
            row[i]=0
            df2[i][index]=0
```

```
0
1
```

Рисунок-4.6 – Заполнение колонок с жанрами для каждой книги

После этого я удалил колонки с перечислением жанров и наград, так как вместо них использовал уже преобразованные колонки.

Все преобразования, проведенные с датасетом я сделал один раз, а затем записал данные в excel. Из данного файла будут считываться данные при запуске рекомендательной системы.

Проводить каждый раз при запуске рекомендательной системы все данные преобразования значительно замедлили бы систему, и тем самым ухудшили бы опыт использования системы.

4.3 Разработка рекомендательной системы

Для своей рекомендательной системы я использовал косинусную сходимость, также называемую коэффициентом Отиаи. Данный коэффициент очень частно используется в рекомендательных системах [19]. Он позволяет вычислить сходство между двумя векторами. Именно поэтому данные о жанрах книги передаются в виде 1 и 0. Формулой является:

$$K = \frac{|A \cap B|}{\sqrt{|A| * |B|}}, \quad (4.1)$$

Где A и B – произвольные множества.

Для начала я сравниваю данные о жанрах, выбранных пользователем с каждой книгой из датасета на основании косинусной сходимости, и добавляю столбец с данным значением в датафрейм.

```
B [15]: dfcos = pd.DataFrame(cos_sim)
        dfcos
```

Out[15]:

	0
0	0.142857
1	0.125988
2	0.142857
3	0.133631
4	0.119523
...	...
52473	0.000000
52474	0.267261
52475	0.133631
52476	0.400892
52477	0.133631

Рисунок-4.7 – Колонка с коэффициентом Отиаи

Затем я нахожу все книги, с максимальным значением в данном столбце при помощи функции max. Косинусная сходимость может принимать любое значение от 0 до 1. Чем меньше угол между объектами на координатной плоскости, тем меньше расстояние между объектами, и тем больше объекты похожи друг на друга. Также, стоит отметить, что максимальное значение может быть сразу у нескольких книг в датасете.

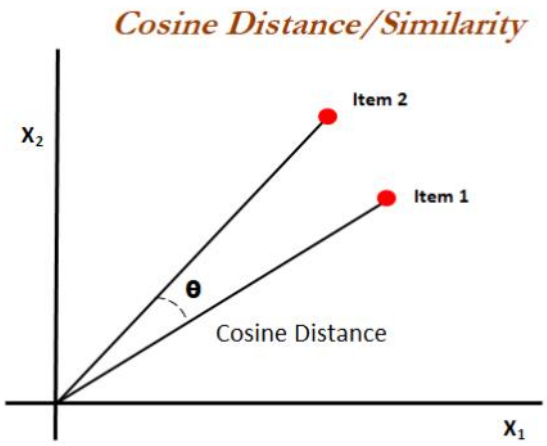

Рисунок-4.8 – Пример коэффициента Отиаи на координатной плоскости [20]

Я выбираю все книги с максимальным значением и суммирую их значения в колонках awards_count, rating и likedPercent. Затем, для каждой из этих книг я делю указанное у них число awards_count, rating и likedPercent на полученную сумму. Таким образом я получил еще 3 дополнительных колонки с процентом наград и рейтинга. Я их преобразую в одну общую метрику, как показано на рисунке 4.5. Данная метрика выражает усредненный рейтинг книги, в основном ориентирующийся на количество полученных произведением наград.

```
df2test['prob'] = df2test['awards_prob'] * 0.95 + df2test['rating_prob'] * 0.025 + df2test['liked_prob'] * 0.025
```

Рисунок-4.9 – Формула расчета поля “prob”

После этого я выбираю одну из книг, на основе вероятностей, которая записана в поле 'prob'.

```
df2test = df2_without_genres.loc[df2_without_genres['cos'] == df2_without_genres['cos'].max()]

if len(df2test)==1:
    df2test['choice'] = 1
else:
    if df2test['awards_count'].sum()>0:
        df2test['awards_prob'] = df2test['awards_count'] / df2test['awards_count'].sum()
    else:
        df2test['awards_prob']=0
    if df2test['rating'].sum()>0:
        df2test['rating_prob'] = df2test['rating'] / df2test['rating'].sum()
    else:
        df2test['rating_prob']=0
    if df2test['likedPercent'].sum()>0:
        df2test['liked_prob'] = df2test['likedPercent'] / df2test['likedPercent'].sum()
    else:
        df2test['liked_prob']=0
    df2test['prob'] = df2test['awards_prob'] * 0.95 + df2test['rating_prob'] * 0.025 + df2test['liked_prob'] * 0.025

    sample = df2test.groupby('cos').apply(lambda x: x.sample(weights=x['prob']))
    choices = sample.reset_index(drop=True, level=0).index
    df2test['choice'] = df2test.index.isin(choices).astype(int)
recommend = df2test.loc[df2test['choice'] == 1]
```

Рисунок-4.10 – Условия расчета дополнительного показателя “prob”

При расчете поля “prob” я использовал условия if и else для реализации различного поведения алгоритма, в зависимости от количества книг с максимальным значением коэффициента Отиаи.

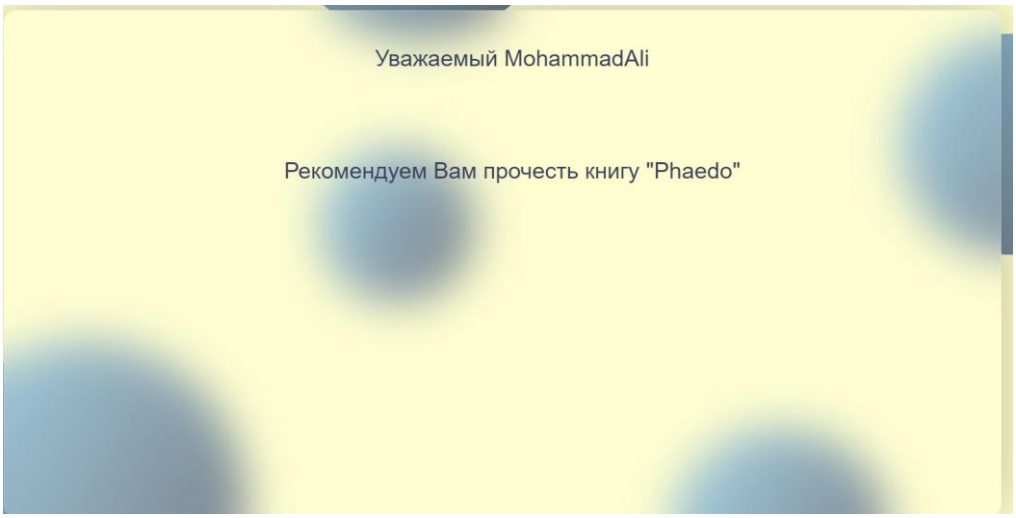

Рисунок-4.11 – Пример финальной рекомендации книги пользователю

На выходе данной функции я получаю одну книгу, которая выводится пользователю на экран и рекомендуется к прочтению, так как является наиболее подходящей к его интересам. В систему также можно внедрить вывод изображения с обложкой книги, либо описание произведения.

ЗАКЛЮЧЕНИЕ

Данная дипломная работа соответствует всем поставленным целям и задачам. Получилось реализовать все функции, позволяющие увеличить интерес читателей к чтению. Также в проекте использовались методы машинного обучения для построения рекомендательной системы.

Была реализована не только рекомендация книг на основании предпочитаемых пользователям жанрах, но и на основании рейтинга книги и количества полученных ей наград.

В ходе постановки задачи было выявлено, что рекомендательные системы сейчас используются для помощи при заказе товаров из интернет-магазинов, просмотре фильмов и видео, прослушивании музыки, но очень мало представлены на уровне рекомендации книг. Проект позволяет уменьшить проблему с падением уровня чтения населения при помощи облегчения выбора читателя.

Так как данный проект разработан при помощи Python Django он является легко встраиваемым в уже готовые сайты или web приложения.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1 Avni Bavishi, Martin D. Slade, Becca R. Levy. A chapter a day: Association of book reading with longevity // Электронная версия на сайте https://www.sciencedirect.com/science/article/abs/pii/S0277953616303689.

2 Фреймворки в веб-разработке // Электронная версия на сайте https://web-creator.ru/articles/about_frameworks.

3 Jefferey M. Jones. Americans Reading Fewer Books Than in Past // Электронная версия на сайте https://news.gallup.com/poll/388541/americans-reading-fewer-books-past.aspx.

4 Jean M. Twenge. Trends in U.S. Adolescents' Media Use, 1976 –2016: The Rise of Digital Media, the Decline of TV, and the (Near) Demise of Print // Электронная версия на сайте https://www.apa.org/pubs/journals/releases/ppm-ppm0000203.pdf

5 Введение в Django // Электронная версия на сайте https://metanit.com/python/django/1.1.php.

6 Python // Электронная версия на сайте https://ru.wikipedia.org/wiki/Python.

7 Топ 8 библиотек Python для машинного обучения и искусственного интеллекта // Электронная версия на сайте https://pythonist.ru/top-8-bibliotek-python-dlya-mashinnogo-obucheniya-i-iskusstvennogo-intellekta/

8 Основы HTML // Электронная версия на сайте https://developer.mozilla.org/ru/docs/Learn/Getting_started_with_the_web/HTML_basics

9 Введение в CSS // Электронная версия на сайте http://htmlbook.ru/samcss/vvedenie-v-css

10 JavaScript // Электронная версия на сайте https://www.tadviser.ru/index.php/Продукт:JavaScript

11 JupyterLab - среда разработки данных Python следующего поколения // Электронная версия на сайте https://www.machinelearningmastery.ru/jupyterlab-a-next-gen-python-data-science-ide-562d216b023d/

12 PyCharm: IDE для Python // Электронная версия на сайте https://timeweb.com/ru/community/articles/pycharm-ide-dlya-python-1#:~:text=Преимущества PyCharm&text=PyCharm позволяет быстро производить рефакторинг,PyGTK и многие другие инструменты.

13 Коэффициент Отиаи // Электронная версия на сайте https://ru.wikipedia.org/wiki/Коэффициент_Отиаи#:~:text=Коэффициент Отиаи (мера Отиаи, коэффициент,приложениях и за рамками биологии..

14 Руководство по Django часть 2: создание скелета Электронная версия на сайте https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/skeleton_website.

15 Юлия Шепталина. UML-диаграммы для моделирования процессов и архитектуры проекта // Электронная версия на сайте https://highload.today/uml-diagrammy/

16 Основы UML — диаграммы использования (use-case) // Электронная версия на сайте https://pro-prof.com/archives/2594.

17 Best Books Ever Dataset // Электронная версия на сайте https://zenodo.org/record/4265096#.YmwfydpByUk.

18 SQLite // Электронная версия на сайте https://blog.skillfactory.ru/glossary/sqlite/

19 Машинное обучение: рекомендательные системы // Электронная версия на сайте https://vc.ru/ml/132779-mashinnoe-obuchenie-rekomendatelnye-sistemy

20 Cosine similarity // Электронная версия на сайте https://www.oreilly.com/library/view/statistics-for-machine/9781788295758/eb9cd609-e44a-40a2-9c3a-f16fc4f5289a.xhtml

Приложение А
(обязательное)

Техническое задание

А.1 Техническое задание на разработку системы, повышающей интерес читателей к чтению с помощью методов машинного обучения

Настоящее техническое задание распространяется на разработку системы, повышающей интерес читателей к чтению с помощью методов машинного обучения. Данная система позволит увеличить уровень вовлеченности людей в чтение при помощи облегчения выбора книг, а также подбора наиболее подходящих книг для каждого человека.

А.1.1 Основание для разработки

Система разрабатывается на основании распоряжения научного руководителя

А.1.2 Назначение

Система предназначена для сбора данных о предпочтениях людей в чтении и рекомендации книг на основе этих предпочтений, и тем самым увеличения интересов читателей к чтению

А.1.3 Требования к функциональным характеристикам

При помощи фреймворка Python Django и HTML, CSS, JavaScript необходимо создать сайт с рекомендательной системой, построенной на языке Python. Также в проекте необходимо использовать реляционную базу данных

На сайте должна быть реализована форма для прохождения опроса о предпочтениях пользователя в чтении. После прохождения опроса, пользователь сможет получить рекомендацию о наиболее подходящей к прочтению книге.

Основные функции, выполнение которых должно быть обеспечено системой:

- сбор данных о предпочтениях пользователя

- получение информации из формы и сохранение ее в базе данных
- выдача рекомендаций, основанных на предпочтениях пользователя

А.1.4 Требования к надежности

Обеспечить конфиденциальность данных и не передавать в открытый доступ базу данных с информацией о пользователях. Также необходимо обеспечить достаточную скорость работы для получения своевременных рекомендаций.

Приложение Б
(обязательное)

Текст программы

Ниже приведен код основных разделов системы.
1. Utils.py

```
#Подключение необходимых библиотек
import pandas as pd
import numpy as np
from numpy import dot
from numpy.linalg import norm
from sklearn.metrics.pairwise import cosine_similarity
#Функция, в которой реализована рекомендация книг
def Recommendation(post):
    df2 = pd.read_excel("df2.xls")
    dfgenres = pd.read_excel("df3test.xlsx")
    post = post.dict()
    for k in post:
        if post[k] == 'on':
            dfgenres[k]=1
    df2_without_genres = pd.read_excel('df2_without_genres.xls')

    df2genres = df2
    df3genres = dfgenres
    cos_sim = cosine_similarity(df2genres, df3genres)
    df2_without_genres['cos'] = pd.DataFrame(cos_sim)
    df2test = df2_without_genres.loc[df2_without_genres['cos'] ==
df2_without_genres['cos'].max()]
    if len(df2test)==1:
        df2test['choice'] = 1
    else:
        if df2test['awards_count'].sum()>0:
            df2test['awards_prob'] = df2test['awards_count'] /
df2test['awards_count'].sum()
        else:
            df2test['awards_prob']=0
        if df2test['rating'].sum()>0:
            df2test['rating_prob'] = df2test['rating'] / df2test['rating'].sum()
        else:
            df2test['rating_prob']=0
        if df2test['likedPercent'].sum()>0:
```

Продолжение приложения Б

```
                df2test['liked_prob'] = df2test['likedPercent'] /
df2test['likedPercent'].sum()
            else:
                df2test['liked_prob']=0
            df2test['prob'] = df2test['awards_prob'] * 0.95 + df2test['rating_prob'] *
0.025 + df2test['liked_prob'] * 0.025

            sample = df2test.groupby('cos').apply(lambda x:
x.sample(weights=x['prob']))
            choices = sample.reset_index(drop=True, level=0).index
            df2test['choice'] = df2test.index.isin(choices).astype(int)
        recommend = df2test.loc[df2test['choice'] == 1]

        return recommend['title'].to_string(index=False)
```

2. Forms.py

#Реализация формы, при помощи которой будет производится запись данных в базу

```
from .models import Users
from django.forms import ModelForm
from django import forms


class UsersForm(ModelForm):
    class Meta:
        model = Users
        fields = [
            'firstname', 'lastname', 'Fiction', 'Romance', 'Fantasy', 'Young_Adult',
'Contemporary', 'Nonfiction', 'Adult', 'Novels',
            'Mystery', 'Historical_Fiction', 'Classics', 'Adventure', 'Historical',
'Paranormal', 'Literature', 'Science_Fiction',
            'Childrens', 'Thriller', 'Magic', 'Humor', 'History', 'Crime',
'Contemporary_Romance', 'Suspense', 'Urban_Fantasy',
            'Middle_Grade', 'Chick_Lit', 'Science_Fiction_Fantasy', 'Supernatural',
'Biography', 'Mystery_Thriller', 'Paranormal_Romance',
            'Horror', 'Teen', 'Philosophy', 'Adult_Fiction', 'Short_Stories',
'Literary_Fiction', 'British_Literature', 'Realistic_Fiction',
            'Drama', 'Religion', 'New_Adult', 'Memoir', 'War', 'twentieth_Century',
'Vampires', 'Christian', 'Graphic_Novels', 'American',
        ]
        widgets = {
```

```
            'firstname': forms.TextInput(attrs = {'class':'form-control nm'}),
            'lastname': forms.TextInput(attrs = {'class':'form-control nm'}),
            'Fiction': forms.CheckboxInput(attrs= {'class' : 'form-check-input'}),
            'Romance': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Fantasy': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Young_Adult': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Contemporary': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Nonfiction': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Adult': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Novels': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Mystery': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Historical_Fiction': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Classics': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Adventure': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Historical': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Paranormal': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Literature': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Science_Fiction': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Childrens': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Thriller': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Magic': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Humor': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'History': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Crime': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Contemporary_Romance': forms.CheckboxInput(attrs={'class': 'form-
check-input'}),
            'Suspense': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Urban_Fantasy': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Middle_Grade': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Chick_Lit': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
            'Science_Fiction_Fantasy': forms.CheckboxInput(attrs={'class': 'form-
check-input'}),
            'Supernatural': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
            'Biography': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
```

```
                'Mystery_Thriller': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Paranormal_Romance': forms.CheckboxInput(attrs={'class': 'form-
check-input'}),
                'Horror': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Teen': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Philosophy': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Adult_Fiction': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Short_Stories': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Literary_Fiction': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'British_Literature': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Realistic_Fiction': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Drama': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Religion': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'New_Adult': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Memoir': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'War': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'twentieth_Century': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'Vampires': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Christian': forms.CheckboxInput(attrs={'class': 'form-check-input'}),
                'Graphic_Novels': forms.CheckboxInput(attrs={'class': 'form-check-
input'}),
                'American': forms.CheckboxInput(attrs={'class': 'form-check-input'}),


            }
```

3. Books.ipynb

```
import pandas as pd
import numpy as np
import re
import numpy as np
df = pd.read_csv('Books.csv')
df = df.drop(['isbn',
'characters','bookFormat','edition','publisher','numRatings','ratingsByStars','setting','co
verImg','bbeScore','bbeVotes','price','publishDate','firstPublishDate'], axis=1)
#функция подсчета наград
```

Продолжение приложения Б

```
df.loc[(df.awards =='[]' ), 'awards'] = 0
df=df.fillna(0)
df['awards_count'] = df['awards']
def normalize(row):
    count = len(re.findall("[(]", str(row)))
    count2 = str(count)
    return count2
df['awards_count'] = df['awards_count'].apply(normalize)
df['awards_count']

#Создание матрицы жанров
genresdf = df['genres'].str.split(',',expand=True)
genresdf=genresdf.fillna(0)
genresdf
def repl(row):
    row = str(row).replace('[', '')
    row = str(row).replace("'", '')
    row = str(row).replace(']', '')
    return row

genresdf[0] = genresdf[0].apply(repl)
genresdf[1] = genresdf[1].apply(repl)
genresdf[2] = genresdf[2].apply(repl)
genresdf[3] = genresdf[3].apply(repl)
genresdf[4] = genresdf[4].apply(repl)
genresdf[5] = genresdf[5].apply(repl)
genresdf[6] = genresdf[6].apply(repl)
genresdf[7] = genresdf[7].apply(repl)
genresdf[8] = genresdf[8].apply(repl)
genresdf[9] = genresdf[9].apply(repl)

c = genresdf[0].append(genresdf[1])
c = c.append(genresdf[2])
c = c.append(genresdf[3])
c = c.append(genresdf[4])
c = c.append(genresdf[5])
c = c.append(genresdf[6])
c = c.append(genresdf[7])
c = c.append(genresdf[8])
c = c.append(genresdf[9])

def delSpace(row):
```

```
    row = row.lstrip()
    return row
c=c.apply(delSpace)

genres_cnt = pd.read_excel('genres.xls')
TopGenres = genres_cnt[0:0]
TopGenres
df2 = df
df2 = pd.concat([df2,TopGenres], axis = 1)

def author(row):
    row = str(row).split(',', 1)[0]
    row = str(row).split('(', 1)[0]
    return row
df2['author'] = df2['author'].apply(author)

top_genres_names = df2.columns[12:]

for index, row in df2.iterrows():
    print(index)
    for i in top_genres_names:
        if(("'"+i+"'") in row['genres']):
            row[i]=1
            df2[i][index]=1
        else:
            row[i]=0
            df2[i][index]=0
```