

Министерство образования Республики Беларусь

Учреждение образования
БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет информационных технологий и управления
Кафедра интеллектуальных информационных технологий

К защите допустить:

Заведующий кафедрой ИИТ
_____ Д. В. Шункевич

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе
по дисциплине "Технология проектирования интеллектуальных систем"

Интеллектуальная справочная система "Кинофильмы"

БГУИР КР 1-40 03 01 01 008 ПЗ

Студент гр. 921702
Руководитель

К. С. Боровская
А. В. Жмырко

Минск 2022

СОДЕРЖАНИЕ

Перечень условных обозначений	5
Введение	6
1 Анализ подходов к решению поставленной задачи	7
1.1 Анализ предметной области и потребностей пользователей	7
1.2 Анализ выбора материалов для предметной области	9
1.3 Граф знаний и онтологический подход	10
1.4 Событийно-ориентированная архитектура и многоагентный подход	11
1.5 Анализ требований к пользовательским интерфейсам интеллектуальных систем	13
1.6 Обоснование выбора языка программирования	16
1.6.1 Сравнения и выбор языка программирования	19
1.6.2 Вывод по выбору языка программирования:	20
1.7 Выбор веб-фреймворка	20
1.8 Итоги анализов	21
2 Проектирование системы	22
2.1 Проектирование структуры и выбор реализации предметной области	22
2.2 Проектирование компонента обработки предметной области	25
2.3 Проектирование пользовательского интерфейса	27
3 Разработка системы	30
3.1 Реализации знаний предметной области	30
3.2 Реализация компонента обработки предметной области	31
3.3 Реализация пользовательского интерфейса и тестирование системы	32
Заключение	38

ПЕРЕЧЕНЬ УСЛОВНЫХ ОБОЗНАЧЕНИЙ

В курсовом проекте используются следующие условные обозначения:

ИС — интеллектуальная система;

БЗ — база знаний;

ГЗ — граф знаний (knowledge graph);

ПрО — Предметная область;

ОС — операционная система;

ОО — объектно-ориентированный;

ООП — объектно-ориентированное программирование;

ПИ — пользовательский интерфейс;

ЯП — язык программирования.

ВВЕДЕНИЕ

На сегодняшний день просмотр кино вошел в число если не ежедневных, то точно еженедельных занятий каждого человека. Каждый из нас регулярно задается вопросом: "Что посмотреть?". Обилие спроса рождает большое количество предложений. И на сегодняшний день существует множество платформ, предлагающих сотни и тысячи фильмов для просмотра.

В то же время человечество сейчас переживает явный профицит информации. И все чаще выбор рядовых пользователей останавливается на максимально простых для понимания и лаконичных приложениях. При всем разнообразии существующих платформ, как правило, у каждого человека уже есть свои фавориты, которые максимально подходят под индивидуальные требования и критерии каждого конкретного человека.

В качестве цели курсовой работы в данном семестре была выбрана разработка специализированной (ограниченной в тематике) справочной интеллектуальной системы подбора продуктов кино индустрии в виде web-приложения, с функцией создания контента для социальных сетей.

Отметим, что разработанная система должна быть интуитивно понятна рядовому пользователю.

Для достижения поставленной цели были поставлены следующие задачи:

- а) Анализ подходов к решению поставленной задачи;
- б) Проектирование системы;
- в) Реализация системы.

1 АНАЛИЗ ПОДХОДОВ К РЕШЕНИЮ ПОСТАВЛЕННОЙ ЗАДАЧИ

1.1 Анализ предметной области и потребностей пользователей

Из всех видов искусства кино занимает уникальное место в современном мире, а соответственно и в жизни человека. Кинематограф — это целый социальный институт. Он влияет на жизнь общества, формируя сознания зрителя. В свою очередь, общество требует от кинематографии новых достижений: усовершенствования технических приемов и креативности идей. Таким образом, между обществом и миром кино существует постоянная связь. И эта связь способна передавать наши чувства, привычки, обычаи и традиции, даже погружать человека в мир его иллюзий. Более того, можно сказать, что кино индустрия способна охватывать почти все сферы общественного сознания. Кино дает возможность человеку воспринимать жизнь под другим углом, чем он ее воспринимал до того. Но надо понимать и помнить, что кино не может заменить собой реальную жизнь.

Фильм (кино, кинофильм, телефильм, кинокартина) — отдельное произведение киноискусства. В технологическом плане фильм представляет собой совокупность движущихся изображений (монтажных кадров, связанных единым сюжетом). Каждый монтажный кадр состоит из последовательности фотографических или цифровых неподвижных изображений (кадров), на которых зафиксированы отдельные фазы движения. Фильм, как правило, имеет звуковое сопровождение.[1]

Люди в современном мире привыкли жить в окружении информационных потоков. Сегодня человек за день получает столько информации, сколько 100-150 лет назад он получал за всю жизнь. Одна из проблем заключается в том, что далеко не вся эта информация человеку действительно необходима. С каждым годом поиск необходимой информации в бесконечных потоках становится все более трудным, а необходимость в качественной сортировке, поиске и подаче материалов растет. Фраза: "Кто владеет информацией, тот владеет миром», становится все менее актуальной, а на смену ей приходит фраза: "Миром владеет не тот, кто владеет информацией, а тот, кто сумел ею овладеть".

Мир кино не исключение, количество информации растет, а, на первый взгляд, простой выбор фильма для просмотра на вечер становится все труднее. Нередко можно встретить ситуации, когда человек за поиском подходящего фильма провел времени больше, чем за его просмотром.

Одним из вариантов решения данной проблемы может стать качественно созданная и удобная система, которая будет хранить в себе ограниченную одной тематикой структурированную информацию.

Исходя из этого в рамках курсовой работы было решено реализовать приложение, с ограниченной, но качественной и проверенной базой кинофильмов и мультфильмов и обернуть это все в удобный, простой и понятный рядовому пользователю интерфейс. Таким образом для создания базы были выбраны ПрО Кинофильмы, Мультфильмы, а также ПрО Рейтингов.

Мультфильм, он же анимационный фильм — фильм, выполненный при помощи средств мультипликации, то есть по кадрового запечатления созданных художником объёмных и плоских изображений или объектов предметно-реального мира на кино - и видеоплёнке или на цифровых носителях. [1]

В последнее время мы можем заметить значительно возросший интерес именно к этому элементу кинематографа. Можно предположить, что это обусловлено нестабильной и тревожной ситуацией в мире(прим. пандемия) и желанием уйти от реальности в мир добрых и спокойных иллюзий. Где спрос, там и предложение, поэтому количество создаваемых мультфильмов в последние годы также значительно выросло. В связи с данной информацией, считаем целесообразным выбор данной ПрО.

Рейтинг кинофильмов, в свою очередь, хороший помощник в отборе части информации из огромных её потоков, на основе мнения большого количества голосов(когда составлен на базе отзывов людей без квалификации в данном вопросе), что дает возможность наиболее объективной оценки или на основе мнения кинокритиков и т.п. (когда составлен на базе отзывов людей с соответствующей квалификацией - экспертов).

Так он позволяет отделить большой поток информации от нескольких экземпляров уже оцененных профессионалами и с большей вероятностью попасть не на "мусор а на что-либо стоящее. В связи с данной информацией, считаем целесообразным заполнение базы системы, путем отбора информации на основе одного из кино рейтингов. Это позволит с меньшими затратами сил поддерживать актуальности БЗ.

Для обоснования создания системы проведем анализ запросов пользователей в поисковую систему Google. При помощи веб-приложения Google Trends [2] были проанализированы интересы пользователей Беларуси за последние времена. На рисунке 1.1 изображена исследуемая статистика. Несложно заметить, что количество запросов по поиску фильмов, сериалов или в целом просмотру чего-либо онлайн не опускается ниже 100 за день только на территории Беларуси.

Отсюда следует вывод, что разработка является актуальной и стоит уделить достаточное внимание возможности онлайн просмотра выбранного материала.

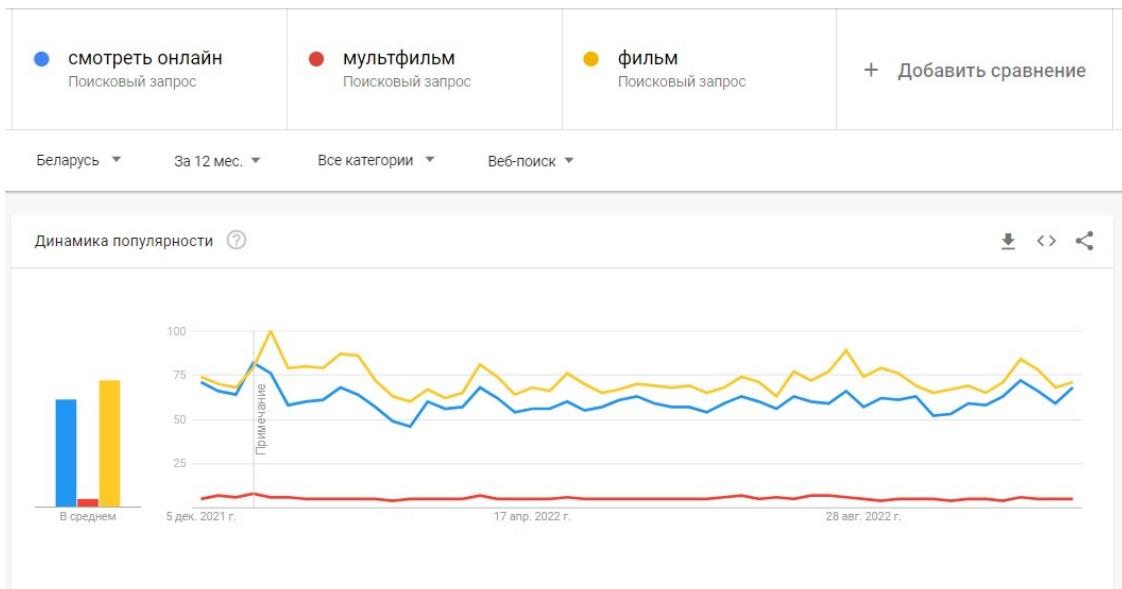


Рисунок 1.1 – Статистика популярности запросов в Google

1.2 Анализ выбора материалов для предметной области

Для удовлетворения нужд пользователей, а также с целью отбора актуальной и качественной (проверенной) информации в предыдущем разделе было решено при заполнении базы основываться в кино рейтингах.

В качестве опорной точки был выбран рейтинг 100 лучших кинокартин мира по версии IMDb. Это список лучших художественных фильмов мира, который формируется на основе оценок, выставляемых картинам зарегистрированными посетителями сайта IMDb.

Помимо основного всеобщего списка, на IMDb составляются отдельные рейтинги по жанрам, по десятилетиям, по возрасту и т.д.. В связи с чем, для актуальности нашей разработки будет целесообразным сделать систему, в которой с легкостью можно заменять и дополнять базу другими элементами, в зависимости от их актуальности, а также от потребностей пользователей.

В выбранном же нами списке расчёт рейтинга фильмов производится на основе подлинной байесовой оценки по следующей формуле:

$$W = \frac{RV + CM}{V + M}$$

W — окончательный рейтинг;

V — число голосов, отданных за фильм;

M — минимальное количество голосов для включения в рейтинг (25 000);

R — средняя оценка фильма (по десятибалльной шкале);

C — средняя оценка среди всех фильмов.

При этом учитываются только голоса зарегистрированных посетителей сайта IMDb, которые голосуют постоянно.[3]

1.3 Граф знаний и онтологический подход

Граф - современная форма представления знаний в сети, состоящая из уникальных сущностей (узлов) и связей между ними (ребер).

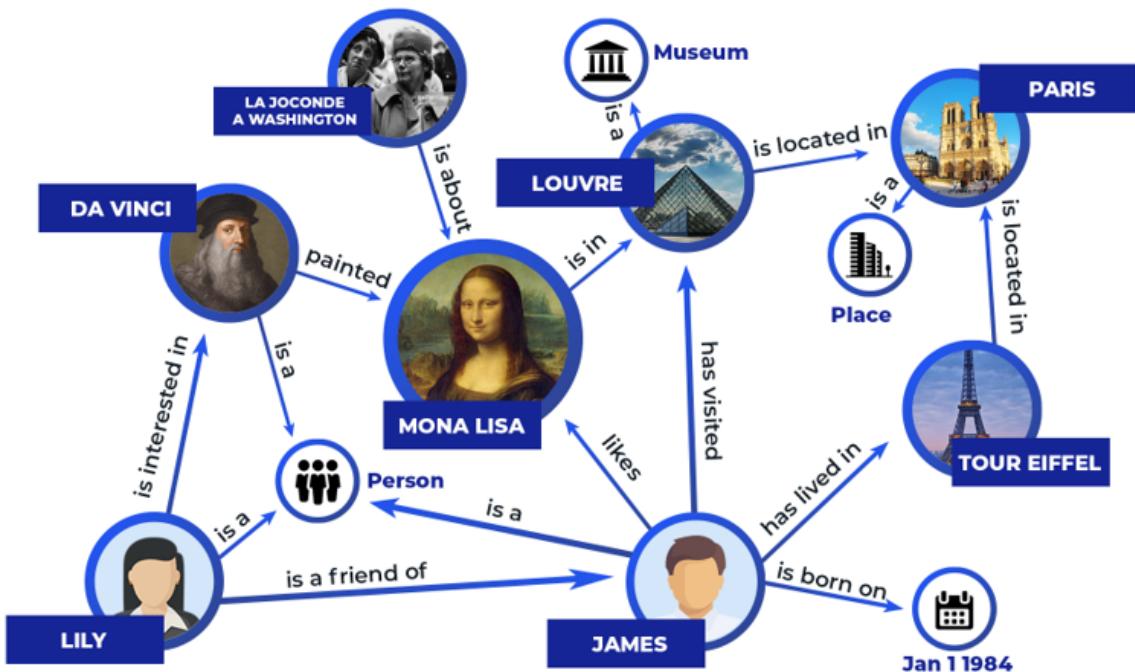


Рисунок 1.2 – Граф знаний

Положительные свойства графа знаний:

- Окружающий мир довольно хорошо структурирован - есть сущности и связи между ними;
- Связи часто имеют еще большее значение, чем сущности;
- Графы - естественная абстракция над сетевыми структурами (карты, деревья решений, визуализации);
- Графы хорошо изучены математически.

Под онтологией понимается явное описание множества объектов и связей между ними (структурированный словарь). Формально онтология состоит из понятий (терминов), организованных в таксономию, их описаний и правил вывода. По определению Тома Грубера, применившего это понятие в области ИТ, онтология - это спецификация концептуализации. Визуализация понятия онтологии представлена на рис. 1.3.

Онтологический инжиниринг - это методология и технология проектирования, разработки и использования онтологий для структурирования и тиражирования знаний в различных предметных областях и приложениях.

Изучив должным образом и вникнув в суть описанных выше понятий, становится очевидной необходимость использования знаний о ГЗ и онтологическом подходе при проектировании и реализации системы. [4]

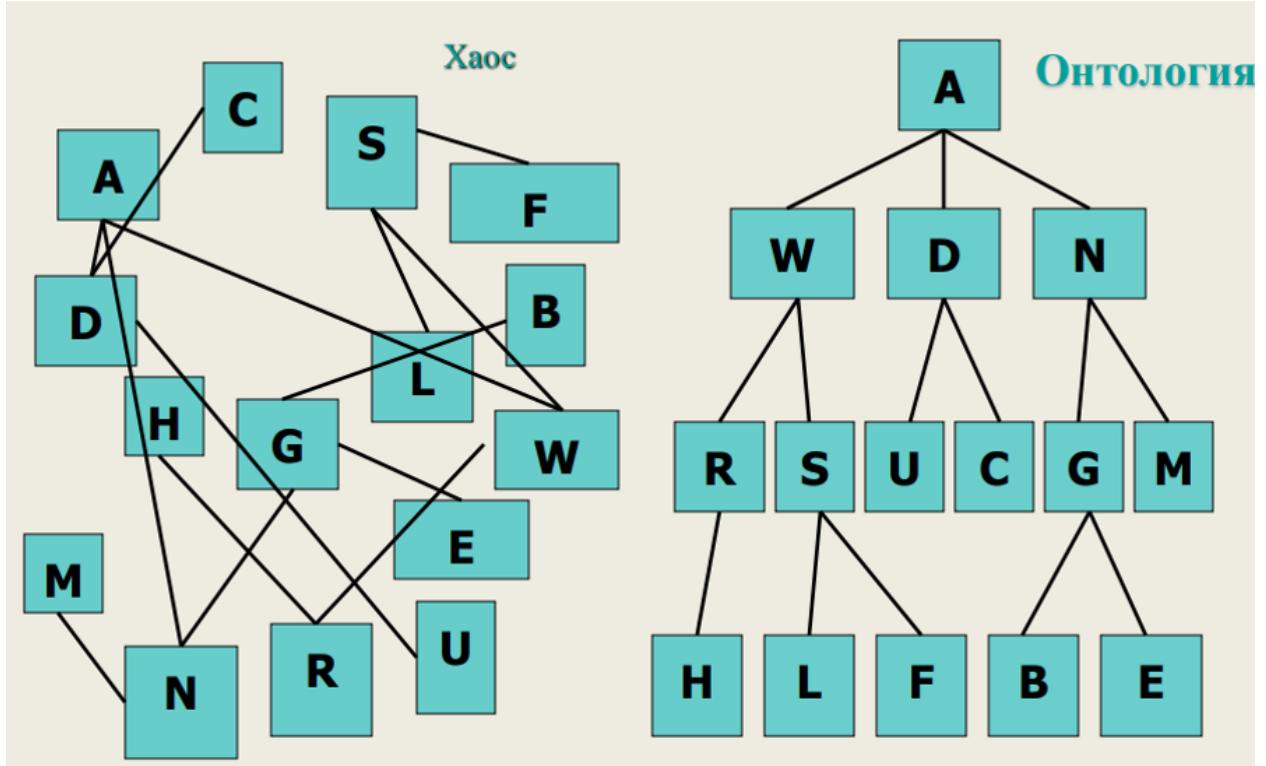


Рисунок 1.3 – Визуализация понятия онтологии

Алгоритм создания онтологии [4]:

- Формирование глоссария предметной области;
- Установление связей между понятиями глоссария;
- Категоризация понятий и формирование метапонятий (снизу);
- Детализация (сверху-вниз);
- Ре-инжиниринг (уточнение, разрешение противоречий, синонимии, избыточности, перестройка, дополнение).

1.4 Событийно-ориентированная архитектура и многоагентный подход

Событийно-ориентированная архитектура (EDA) – это парадигма программной архитектуры, способствующая порождению, обнаружению, потреблению событий и реакции на них. [5] Визуализация EDA представлена на картинке 1.4.

- Не звоните нам, мы сами вам позвоним;
- Среда (framework) вызывает ваши компоненты, а не вы вызываете библиотеку;
- Вы регистрируете обработчики, framework вызывает их;
- Каждый обработчик – конечный автомат (обработка события приводит к изменению переменных состояния).[4]

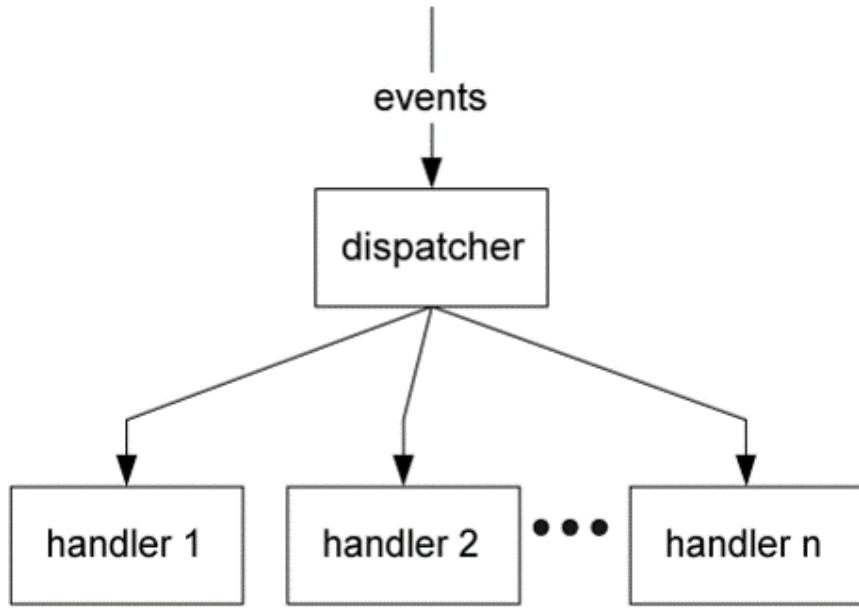


Рисунок 1.4 – Визуализация понятия EDA

Многоагентный подход, агент - исполнитель логически атомарных процессов; семантическая память как «доска объявлений». [4].

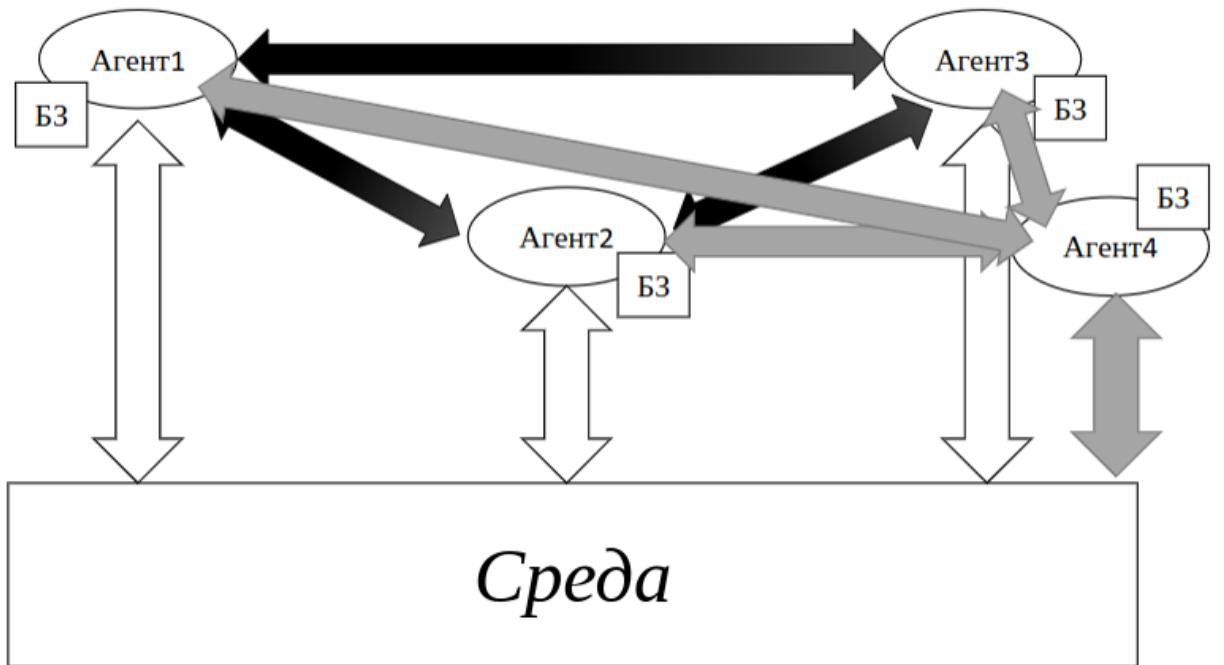


Рисунок 1.5 – Взаимодействие агентов: современный подход(JADE, Go!, Gama, т.д.)

Таким образом, изучив данные понятия, можно прийти к выводу о логичности и правильности использования данных подходов в дальнейшем проектировании и разработке системы.

1.5 Анализ требований к пользовательским интерфейсам интеллектуальных систем

Пользовательский интерфейс — интерфейс, обеспечивающий передачу информации между пользователем-человеком и программно-аппаратными компонентами компьютерной системы.

Главным достоинством хорошего пользовательского интерфейса является то, что пользователь всегда чувствует, что он управляет пользовательским интерфейсом, а не наоборот. Для достижения такого уровня взаимодействия к пользовательскому интерфейсу предъявляются следующие требования[6]:

– Естественность — интерфейс не должен вынуждать пользователя существенно изменять привычные для него способы решения задачи. Это, в частности, означает, что сообщения и результаты, выдаваемые приложением, не должны требовать дополнительных пояснений. Целесообразно также сохранить систему обозначений и терминологию, используемые в конкретной ПрО. Использование знакомых пользователю понятий и образов обеспечивает интуитивно понятный интерфейс при выполнении его заданий.

– Согласованность - интерфейс должен позволять пользователям переносить имеющиеся знания на новые задачи, осваивать новые аспекты быстрее, и благодаря этому фокусировать внимание на решаемой задаче, а не тратить время на понимание различий в использовании тех или иных элементов управления, команд и прочих компонентов ПИ. Согласованность важна для всех аспектов интерфейса, включая имена команд, визуальное представление информации и поведение интерактивных элементов. Для реализации свойства согласованности в создаваемом программном обеспечении, необходимо учитывать его различные аспекты.

а) Согласованность в пределах системы - одна и та же команда должна выполнять одни и те же функции, где бы она ни встретилась, причем одним и тем же образом.

б) Согласованность в пределах различных систем - поддерживая согласованность с интерфейсом, предоставляемым конкретной системой, её интерфейс может опираться на те знания и навыки пользователя, которые он получил ранее при работе с другими системами.

– Эффективность — должен позволять предотвращать ситуации, которые, вероятно, закончатся ошибками. Он также должен уметь адаптироваться к потенциальным ошибкам пользователя и облегчать ему процесс устранения последствий таких ошибок.

Даже при наличии хорошо спроектированного пользовательского интерфейса, соответствующего указанным выше требованиям, пользователи могут делать те или иные ошибки. Можно выявить два основных типа ошибок:

а) ошибки физического типа - случайный выбор неправильной команды или данных;

б) ошибки логического типа - принятие неправильного решения на выбор команды или данных

Эффективный интерфейс должен принимать во внимание эти вещи и на каждом этапе работы разрешать только соответствующий набор действий и предупреждать пользователей о тех ситуациях, где они могут повредить системе или фрагментам базы знаний; еще лучше, если у пользователя существует возможность отменить или исправить выполненные действия.

Таким образом, можно определить несколько фундаментальных принципов при проектировании эффективных интерфейсов:

– Эффективные интерфейсы должны быть очевидными и внушать своему пользователю чувство контроля. Необходимо, чтобы пользователь мог одним взглядом окинуть весь спектр своих возможностей, понять, как достичь своих целей и выполнить работу.

– Эффективные интерфейсы не должны беспокоить пользователя внутренним взаимодействием с системой. Необходимо бережное и непрерывное сохранение работы, с предоставлением пользователю возможности отменить любое действие в любое время.

– Эффективные приложения должны выполнять максимум работы, требуя при этом минимум информации от пользователя.

Единственный путь установить предпочтения пользователя - это тестирование. Анализ или рассуждения «с позиций пользователя» могут быть применены на предварительных этапах, но не могут заменить тестирование.

Также, важно будет отметить о необходимости обратной связи — интерфейс должен обеспечивать обратную связь для действий пользователя. Каждое действие пользователя должно получать визуальное, а иногда и звуковое подтверждение того, что система восприняла введенную команду; при этом вид реакции, по возможности, должен учитывать природу выполненного действия. Когда пользовательский интерфейс обрабатывает поступившую задачу, полезно предоставить пользователю информацию относительно состояния процесса, а также возможность прервать этот процесс в случае необходимости. В среднем пользователь способен выдержать несколько секунд ожидания ответной реакции.

Не менее важным при проектировании ПИ будет помнить о простоте — интерфейс должен обеспечивать легкость в изучении и использовании. Кроме того, он должен предоставлять доступ ко всему перечню функциональных возможностей, предусмотренных конкретной системой. Один из возможных путей поддержания простоты — представление на экране информации, минимально необходимой для выполнения пользователем очередного шага. Многословные командные имена или сообщения, непродуманные или избыточные фразы затрудняют извлечение существенной информации.

Другой путь к созданию простого интерфейса — размещение и представление компонентов ПИ на экране с учетом их смыслового значения и логической взаимосвязи. Это позволяет использовать в процессе работы ассоциативное мышление пользователя, задействовать ментальный модели и не спорить с ними [7]. Можно помочь пользователям управлять сложностью отображаемой информации, используя последовательное раскрытие (диалоговые окна, меню). Последовательное раскрытие предполагает такую организацию информации, при которой в каждый момент времени на экране находится только та ее часть, которая необходима для выполнения очередного шага. Сокращая объем информации, представленной пользователю, уменьшается объем информации, подлежащей обработке.

Еще одной важной деталью ПИ является визуальная привлекательность — корректное визуальное представление компонентов пользовательского интерфейса обеспечивает передачу важной дополнительной информации об их поведении и взаимодействии. В то же время следует помнить, что каждый визуализированный компонент ПИ, который появляется на экране, потенциально требует внимания пользователя, которое не безгранично. Следует организовывать пользовательский интерфейс так, чтобы он не только содействовал пониманию пользователем представленной информации, но и позволял бы сосредоточиться на ее существенных аспектах.

Качество интерфейса сложно оценить количественными характеристиками, однако более или менее объективную его оценку можно получить, опираясь на следующие частные показатели:

а) время, необходимое определенному пользователю для достижения заданного уровня знаний и навыков по работе с пользовательским интерфейсом;

б) сохранение полученных навыков по истечении некоторого времени (например, после недельного перерыва пользователь должен выполнить определенную последовательность действий за заданное время).

в) скорость решения задачи, используя возможности пользовательского интерфейса; при этом должно оцениваться не производительность системы и не скорость ввода данных с клавиатуры, а время, необходимое для достижения цели решаемой задачи.

г) субъективная удовлетворенность пользователя при работе с пользовательским интерфейсом конкретной системы (которая количественно может быть выражена в процентах или оценкой по п-балльной шкале).

Таким образом, можно сделать вывод о необходимости выделения достаточного количества времени и внимания аспекту проектирования пользовательского интерфейса, так как от него по большей части зависит успех приложения.

1.6 Обоснование выбора языка программирования

Выбор производится между тремя ранее изученными в рамках учебной программы языками: C++, Java, Python.

C++ — компилируемый, статически типизированный язык программирования общего назначения. Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространенные контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. Широко используется для разработки программного обеспечения. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Язык оказал огромное влияние на другие языки программирования, в первую очередь на Java и C. [8]

Плюсы:

- а) Быстрое выполнение кода по сравнению с более высокоуровневыми языками (Python, C, Java и другими);
- б) Эффективен в работе с памятью;
- в) Отсутствие определённых стандартов на ввод-вывод, графику, геометрию, диалог, доступ к базам данных и прочим типовым приложениям. Стандарты на графику, доступ к базам данных и т. д. являются недостатком, если программист хочет определить свой собственный стандарт.

Минусы:

- а) Наличие множества возможностей, нарушающих принципы безопасности приводит к тому, что в C+-программы может легко закрасться трудноуловимая ошибка. Вместо контроля со стороны компилятора разработчики вынуждены придерживаться весьма нетривиальных правил кодирования. По сути эти правила ограничивают C++ рамками некоего более безопасного подъязыка. Большинство проблем безопасности C++ унаследовано от C, но важную роль в этом вопросе играет и отказ автора языка от идеи использовать автоматическое управление памятью (сборка мусора).
- б) Сложно поддерживать программу с большим количеством строк кода. Исходные коды C++ будут иметь больше строк по сравнению с Python;
- в) Менее масштабируем по сравнению с Java;
- г) Исходный код C++ будет скомпилирован в native код (машинный код). Таким образом, исполняемые файлы C++ зависят от платформы. В случае Java и Python обе платформы являются кроссплатформенными и доступны для всех основных операционных систем;
- д) Отсутствие встроенной многопоточности.

Java — строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems (в последующем приобретенной компанией Oracle). Работа ведется сообществом, организованным через Java Community Process; язык и основные реализующие его технологии распространяются по лицензии GPL. Права на торговую марку принадлежат корпорации Oracle. Приложения Java обычно транслируются в специальный байт-код, поэтому они могут работать на любой компьютерной архитектуре, для которой существует реализация виртуальной Java-машины. Стабильно занимает высокие места в рейтингах популярности языков программирования (2-е место в рейтингах IEEE Spectrum (2020) и TIOBE (2021)). [9]

Плюсы:

- а) Безопасность: отсутствие поддержки указателей и адресной арифметики, автоматическое управление памятью со сборкой мусора, встроенные средства, защищающие от распространенных ошибок программ C++ таких, как переполнение буфера или выход за границы массива;
- б) Разработанная система модулей и раздельной компиляции, быстрее и менее подвержена ошибкам, чем препроцессор и ручная сборка C++;
- в) Полная стандартизация и исполнение в виртуальной машине, развитое окружение, включающее библиотеки для графики, интерфейса пользователя, доступа к БД прочих типовых задач, в итоге - многоплатформенность;
- г) Встроенная многопоточность;
- д) Объектная подсистема Java в значительно более высокой степени, чем C++ и Python, отвечает фундаментальному принципу ООП «всё — объект». Интерфейсы позволяют обеспечить большинство преимуществ множественного наследования, не вызывая его негативных эффектов;
- е) Рефлексия значительно более развита, чем в C++ и позволяет определять и изменять структуру объектов во время работы программы.

Минусы:

- а) В Java есть четко определенные стандарты на ввод-вывод, графику, геометрию, диалог, доступ к базам данных и прочим типовым приложениям;
- б) Использование сборщика мусора и виртуальной машины создают труднопреодолимые ограничения. Программы на Java, как правило, медленнее чем в C++, но быстрее чем Python, требуют значительно больше памяти, к тому же виртуальная машина изолирует программу от ОС;
- в) В отличие от C++ и Python, Java является чисто ОО языком, без возможности процедурного программирования. Для объявления свободных функций или глобальных переменных необходимо создавать фиктивные классы, содержащие только static члены. Для задания главной функции даже самой простой программы на Java необходимо поместить её в класс;
- г) Фреймворки тяжеловесные и сложные в конфигурации. Для Spring даже написали Spring Boot – по сути, фреймворк над фреймворком.

Python - высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ. Язык является полностью объектно-ориентированным — все являются объектами. Необычной особенностью языка является выделение блоков кода пробельными отступами. Синтаксис ядра языка минималистичен, за счёт чего на практике редко возникает необходимость обращаться к документации, сам же язык известен как интерпретируемый и используется, в том числе для написания скриптов.

Стандартная библиотека включает большой набор полезных переносимых функций, начиная от функционала для работы с текстом и заканчивая средствами для написания сетевых приложений. Дополнительные возможности, такие как математическое моделирование, работа с оборудованием, написание веб-приложений или разработка игр, могут реализовываться посредством обширного количества сторонних библиотек, а также интеграцией библиотек, написанных на Си или C++, при этом и сам интерпретатор Python может интегрироваться в проекты, написанные на этих языках. Существует и специализированный репозиторий программного обеспечения, написанного на Python, — PyPI. Данный репозиторий предоставляет средства для простой установки пакетов в операционную систему и стал стандартом де-факто для Python. [7]

Плюсы:

- а) Автоматическое управление памятью;
- б) Хорошая читабельность кода;
- в) Доступен на всех платформах операционных систем UNIX, MS-DOS, Mac OS, Windows и Linux и других Unix-подобных ОС;
- г) Сильная совместимость с Unix, аппаратным обеспечением, сторонним программным обеспечением с огромной библиотекой;
- д) Подход к ООП в Python упрощает программирование, делает код более понятным и одновременно добавляет гибкости языку.

Минусы:

- а) Низкая скорость работы программ;
- б) Высокое потребление памяти;
- в) Касательно ООП в Python в отличие от C++ и Java отсутствуют модификаторы доступа к полям и методам класса;
- г) Высокий уровень зависимости от системных библиотек. В результате затрудняется перенос на другие системы. Проблема решается посредством Virtualenv, однако у этого инструмента свои недостатки: костыли, избыточность полных методов изоляции, дублирование системных библиотек;

1.6.1 Сравнения и выбор языка программирования

Python сравнивается с C++/Java с точки зрения лаконичности, простоты и гибкости Python. Можно сравнить «Hello, world» - программы, записанные на каждом из языков.

C++	<pre>#include <iostream> int main() { std::cout << "Hello, world!" << std::endl; return 0; }</pre>
Java	<pre>public class HelloClass { public static void main(String[] args) { System.out.println("Hello, world!"); } }</pre>
Python	<pre>print("Hello, world!")</pre>

Рисунок 1.6 – Сравнение кода языков программирования

Касательно ООП в Python в отличие от C++ и Java отсутствуют модификаторы доступа к полям и методам класса, атрибуты и поля у объектов могут создаваться на лету в ходе исполнения программы, а все методы являются виртуальными. По сравнению с Java Python позволяет также перегружать операторы, что дает возможность использовать выражения близкие к естественным. В совокупности подход к ООП в Python упрощает программирование, делает код более понятным и одновременно добавляет гибкости языку. С другой стороны, скорость выполнения кода на Python (как и других интерпретируемых языков) значительно ниже, чем скорость выполнения аналогичного кода на C++ и обычно ожидается ниже, чем в Java. Код на C++ получается производительнее Python, но занимает больше строк. Согласно исследованиям алгоритмов, применяемых в биоинформатике, Python показал себя более гибким чем C++, а Java оказалась компромиссным решением между производительностью C++ и гибкостью Python.

В Java и Python все объекты создаются в куче, в то время как C++ позволяет создавать объекты как в куче, так и на стеке, в зависимости от используемого синтаксиса. На производительность также влияет способ доступа к данным в памяти. В C++ и Java доступ к данным происходит по постоянным смещениям в памяти, в то время как в Python — через хеш-таблицы. Использование указателей в C++ может быть довольно сложным для понимания среди новичков, и овладение навыками правильного использования указателей может занять некоторое время.

1.6.2 Вывод по выбору языка программирования:

Исходя из заданных целей и сравнения языков программирования был выбран Python. Поскольку в мире Python много качественных библиотек, хороший инструментарий для работы с большим количеством данных, следовательно, не нужно изобретать велосипед, если надо срочно решить какую-то задачу. Также в веб-разработке стали очень популярны Python-фреймворки такие, как Django и Flask. Они облегчают процесс написания на языке Python кода серверной части приложений.

1.7 Выбор веб-фреймворка

Веб-фреймворк или каркас веб-приложений — фреймворк, предназначенный для создания динамических веб-сайтов, сетевых приложений, сервисов или ресурсов. Он упрощает разработку и избавляет от необходимости написания рутинного кода. Многие фреймворки упрощают доступ к базам данных, разработку интерфейса, и также уменьшают дублирование кода [10].

Streamlit — это веб-фреймворк, предназначенный для исследователей данных для простого развертывания моделей и визуализаций с использованием Python. Это быстро и минималистично, а также красиво и удобно. Есть встроенные виджеты для пользовательского ввода, такие как загрузка изображений, ползунки, ввод текста и другие знакомые элементы HTML, такие как флажки и переключатели. Всякий раз, когда пользователь взаимодействует с потоковым приложением, сценарий python перезапускается сверху вниз, что важно учитывать при рассмотрении различных состояний вашего приложения.[11]

Преимущества Streamlit [12]:

- Легко учиться - практически не требуется обучения;
- Удобный для пользователя (удобный для разработчиков);
- Небольшое количество времени на сборку;
- Поддерживает большое количество интеграций и создание собственных модулей, в том числе поддержка онтологического подхода;
- Позволяет быстро создавать пользовательский интерфейс соответствующий большому количеству требований эффективного пользовательского интерфейса(см. п. 1.5.).

Ввиду простоты, легкости в использовании, эффективности и поддержки всех перечисленных в пунктах выше подходов в качестве фреймворка в данной работе был выбран молодой, развивающийся и гибкий Streamlit.

1.8 Итоги анализов

Исходя из проанализированной информации, можно сказать, что умение быстро находить нужную информацию в современном мире является одним из ключевых навыков(см. п. 1.1), в связи с чем возникает необходимость в системах, способных структурировано, качественно, интеллектуально и удобно помогать справляться с задачей поиска и хранения информации(см. п. 1.3 и 1.4). Также можно сказать, что ограничение информации (выбор только определенной её части в соответствие с заданными критериями) зачастую играет не менее важную роль. (см. п. 1.2) В качестве решения этих проблем может выступать проектируемая нами система.

Отметим, что выбор предметных областей связан, в том числе, с уже проделанной работой над данными ПрО в предыдущих курсовых проектах, а значит достаточно изучен автором, для дальнейшей эффективной работы.

Система должна предоставлять пользователю возможность удобного поиска актуальных продуктов кино индустрии и реализовывать потребность в дальнейшем высказывании своих мыслей в социальные сети(как вариант качественной проработки полученной информации в связи с просмотром кинокартины). Выбранные нами технологии(см. п. 1.6 и п 1.7) могут позволить нам качественно выполнить поставленные выше задачи.

Также не будем забывать, что исходя из п. 1.5 немаловажной частью является создание качественного, удобного, простого и понятного пользовательского интерфейса. Это необходимо для предоставления возможности пользоваться системой любому человеку вне зависимости от уровня его навыков и уровня знакомства с подобными системами.

Технические возможности выбранного фреймворка и ЯП, а также рассмотренных выше подходов, на базе которых разрабатывается система, позволяют реализовать функции, недоступные либо сложно реализуемые в других системах. Однако технология также накладывает определённые ограничения на некоторые аспекты разработки функционирования системы.

Для достижения поставленной задачи обозначим следующие действия:

- а) Выделить ПрО, отобрать и проверить материал для ее описания;
- б) Определить ключевые элементы выбранной предметной области и их взаимодействие с остальной системой; Структурировать и задать способ хранения знаний ПрО;
- в) Спроектировать и реализовать компонент поиска и генерации в нужном виде фрагмента базы знаний;
- г) Спроектировать и реализовать компонент генерации качественно нового материала для дальнейшего употребления его пользователем (фото контент для соц. сетей с отзывом пользователя);
- д) Спроектировать и реализовать эффективный ПИ.
- е) Протестировать корректность работы системы.

2 ПРОЕКТИРОВАНИЕ СИСТЕМЫ

2.1 Проектирование структуры и выбор реализации предметной области

В качестве реализуемых предметных областей были выбраны ПрО Мультфильмов и ПрО фильмов (обоснования см. в п. 1.1 и 1.8). Также в п. 1.2 было определено, что для эффективности системы стоит ввести меры некой ее ограниченности. В качестве рамок для получения знаний о ПрО было выбрано понятие рейтингов (рейтинг 100 лучших фильмов по версии IMDb). В п. 1.3 была также определена польза от представления знаний ПрО используя онтологический подход, а именно - граф знаний.

Также отметим необходимость определения исключительно минимальной информации о каждом объекте для последующей простоты заменяемости выбранного рейтинга на другие под запрашиваемую тематику и требования. (прим. список новогодних кинофильмов, список фильмов на тему искусственного интеллекта и др.), а также для избежание профицита информации, что может отпугнуть пользователя.

Определим область знаний, минимально необходимую для достаточной осведомленности пользователя о каждом отдельном объекте кино индустрии(фильм/мультфильм).

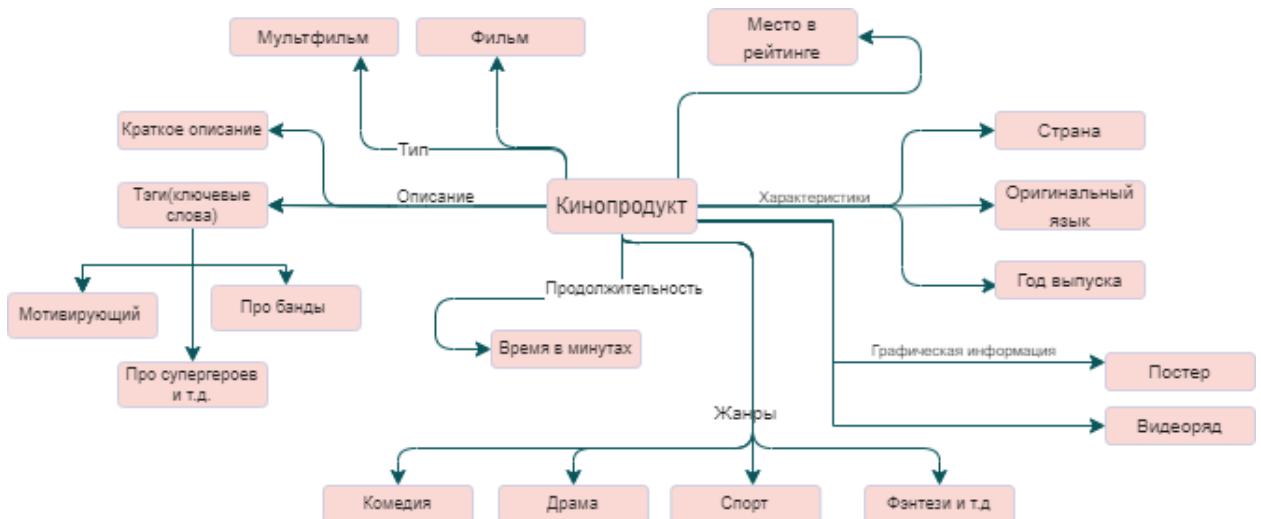


Рисунок 2.1 – Знания об элементах ПрО представленные в виде mind-карты

Получим следующий список полей знаний, необходим к заполнению для каждого элемента рейтинга:

- название кинокартины;
- постер к кинокартине;
- краткое описание кинокартины;
- ссылка на просмотр кинокартины;

- д) год выпуска кинокартины;
- е) продолжительность кинокартины;
- ж) страна производства кинокартины;
- з) оригинальный язык кинокартины;
- и) жанры кинокартины;
- к) тип кинокартины (фильм/мультфильм);
- л) ключевые слова/теги.

В качестве основных справочных порталов для уточнения информации были обозначены следующие сервисы:

а) Википедия (англ. Wikipedia, произносится [wikipidi] или [wkipidi]) — общедоступная многоязычная универсальная интернет-энциклопедия со свободным контентом, реализованная на принципах вики;

б) Кинопоиск — крупнейший русскоязычный интернет-сервис о кино. С 2018 года также доступен онлайн-кинотеатр (до 2 ноября 2021 имевший отдельное название — КиноПоиск HD) с несколькими тысячами фильмов, сериалов, мультифильмов, в том числе премьерных и эксклюзивных;

в) Kinogo — популярный онлайн-кинотеатр (наряду с HD-режимом) в России и странах СНГ, в основном в котором собраны различные зарубежные фильмы и телесериалы, также представлен и российский кинематограф.

В качестве структуры для хранения и обработки знаний ПрО был выбран граф знаний, реализация которого производится на базе компонента Streamlit - "Agraph component Мощная и легкая библиотека для визуализации и создания графов (в т.ч графов знаний). [13].



Рисунок 2.2 – Пример представления элементов ПрО в виде графа знаний используя agraph component

Представление Node и Edge в модуле agraph.

```

1  class Node:
2      def __init__(self,
3          id,
4          title=None, # displayed if hovered
5          label=None, # displayed inside the node
6          color="#AC0E99",
7          shape="dot",
8          size=25,
9          **kwargs
10     ):
11         self.id=id
12         if not title:
13             self.title=id
14         else:
15             self.title=title
16             self.label = label
17             self.shape=shape # image, circularImage, diamond, dot, star, triangle, triangleDown, hexagon, square and icon
18             self.size=size
19             self.color=color #F0D2B5 #F4B894 #F7A7A6 #D8EB2
20             self.__dict__.update(kwargs)
21         self.__dict__.update(kwargs)
22
23     def to_dict(self):
24         return self.__dict__

```

```

3  class Edge:
4      """
5          https://visjs.github.io/vis-network/docs/network/edges.html
6      """
7      def __init__(self,
8          source,
9          target,
10         color="#F7A7A6",
11         **kwargs
12     ):
13         self.source=source
14         self.__dict__['from']=source
15         self.to=target
16         self.color=color
17         self.__dict__.update(**kwargs)
18
19     def to_dict(self):
20         return self.__dict__

```

Рисунок 2.3 – Реализация компонента agraph

Иерархически знания о предметной области можно представить следующим образом: рис. 2.5 - иерархия узлов графа. А также список отношений между ними:

- а) year of issue - год выпуска;
- б) duration - продолжительность;
- в) country - страна производства;
- г) original language - оригинальный язык;
- д) genre - жанр;
- е) place - место в рейтинге;
- ж) type - тип;
- з) tags - ключевые слова.

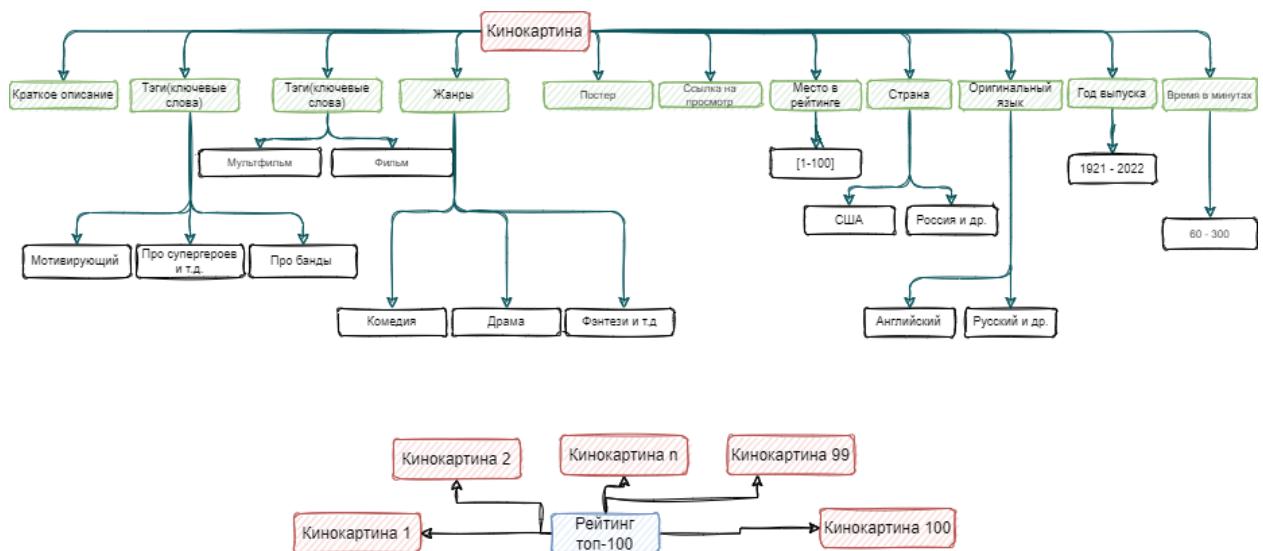


Рисунок 2.4 – Иерархическое представление знаний ПрО

2.2 Проектирование компонента обработки предметной области

Для корректной работы системы и реализации всего задуманного функционала необходимо разработать следующие модули обработки ГЗ:

- а) компонент подбора кинокартин по заданным параметрам и без них;
- б) компонент генерации фото-контента для социальных сетей;
- в) компонент визуализации графа знаний по найденной информации.

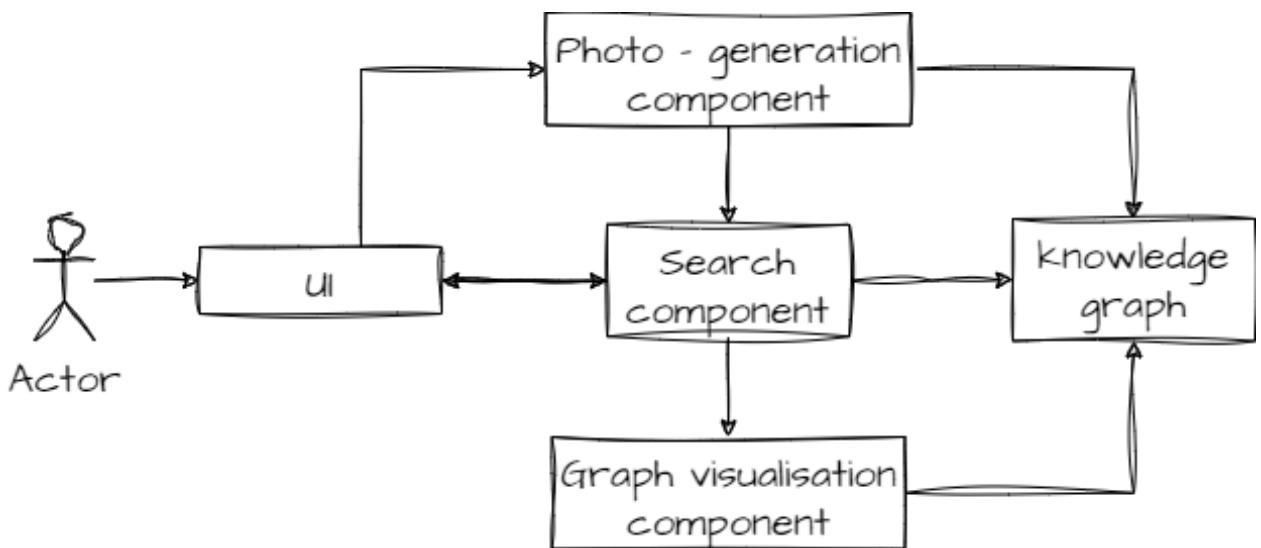


Рисунок 2.5 – Компоненты системы

Планируемый способ подбора продуктов кино индустрии основывается на выборе пользователя из предложенных критериев. То есть подразумевается, что система предлагает пользователю выбрать то, что ему больше подходит, используя поля в виде (checkbox, radiobutton и др.) для выбора описанных выше характеристик кинокартин и формирует новый список с суженными областями поиска, пока не будет выделен список из нескольких продуктов, удовлетворяющих всем запросам пользователя.

В таком случае общий алгоритм обработки данных будет выглядеть следующим образом. После начала работы системы получает стартовую страницу сайта. Для получения информации, которая там содержится, система получает сигналы от соответствующих кнопок пользовательского интерфейса. В зависимости от типа содержимого данные сортируются и упорядочиваются. В зависимости от последующих действий пользователя система использует построенные связи для перемещения между объектами и получения необходимых данных путем повторения аналогичных шагов.

Общая схема взаимодействия системы и пользователя представлена на рисунке 2.6.

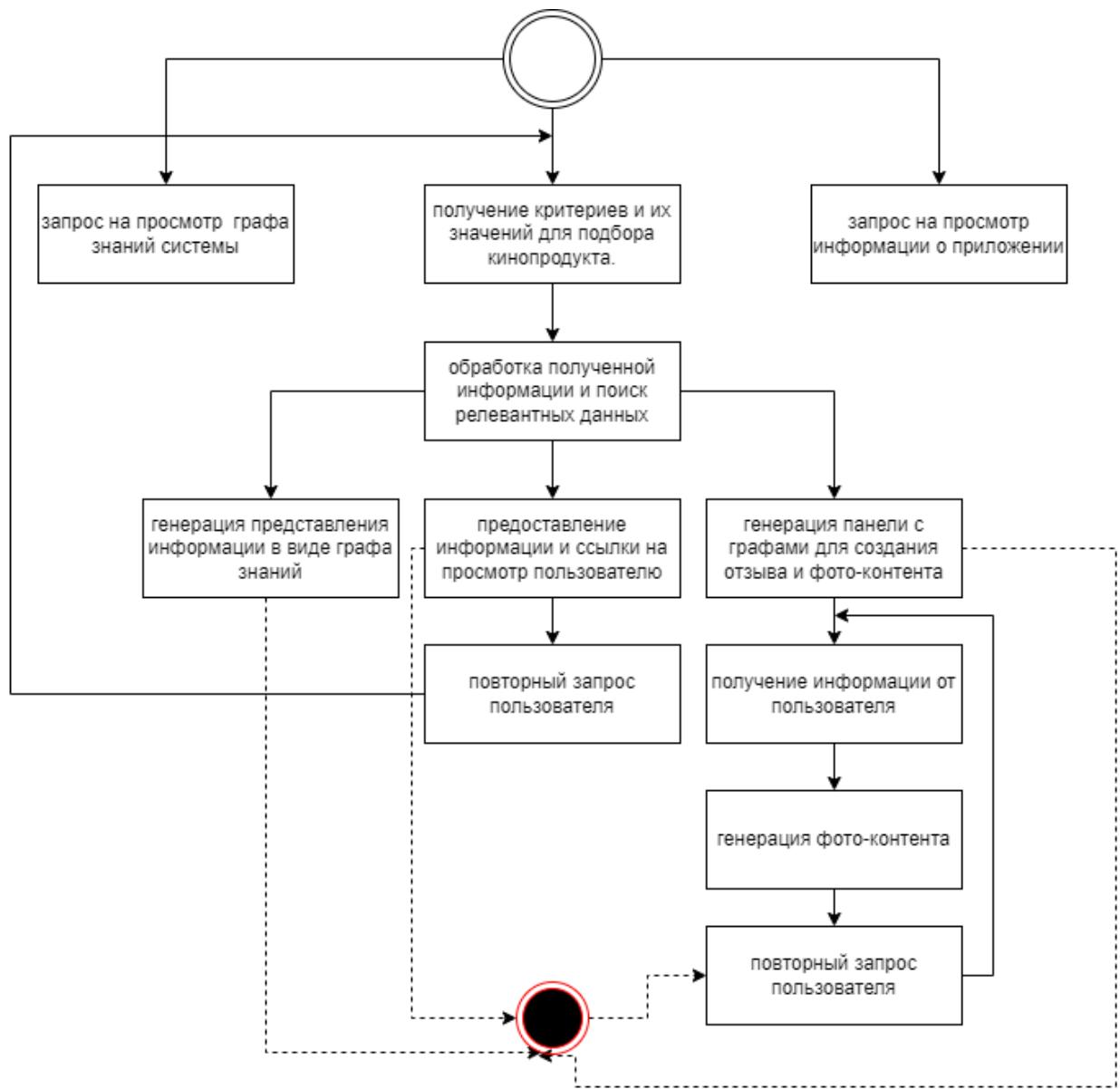


Рисунок 2.6 – Основные этапы взаимодействия системы с пользователем

Разработка модуля генерации фото-контента основывается на возможностях библиотеки pillow. Pillow и его предшественник PIL – это оригинальные библиотеки Python для работы с изображениями. Несмотря на то, что существуют другие библиотеки Python для обработки изображений, Pillow остается важным инструментом для понимания и работы в целом. Для оперирования и обработки изображений Pillow предоставляет инструменты, аналогичные тем, которые можно найти в программном обеспечении, таком как Photoshop.

Шаблон предполагаемого получаемого изображения представлен на рисунке 2.7. Размеры изображения соответствуют общим стандартам и размерам insta-stories и равны 1080x1920.

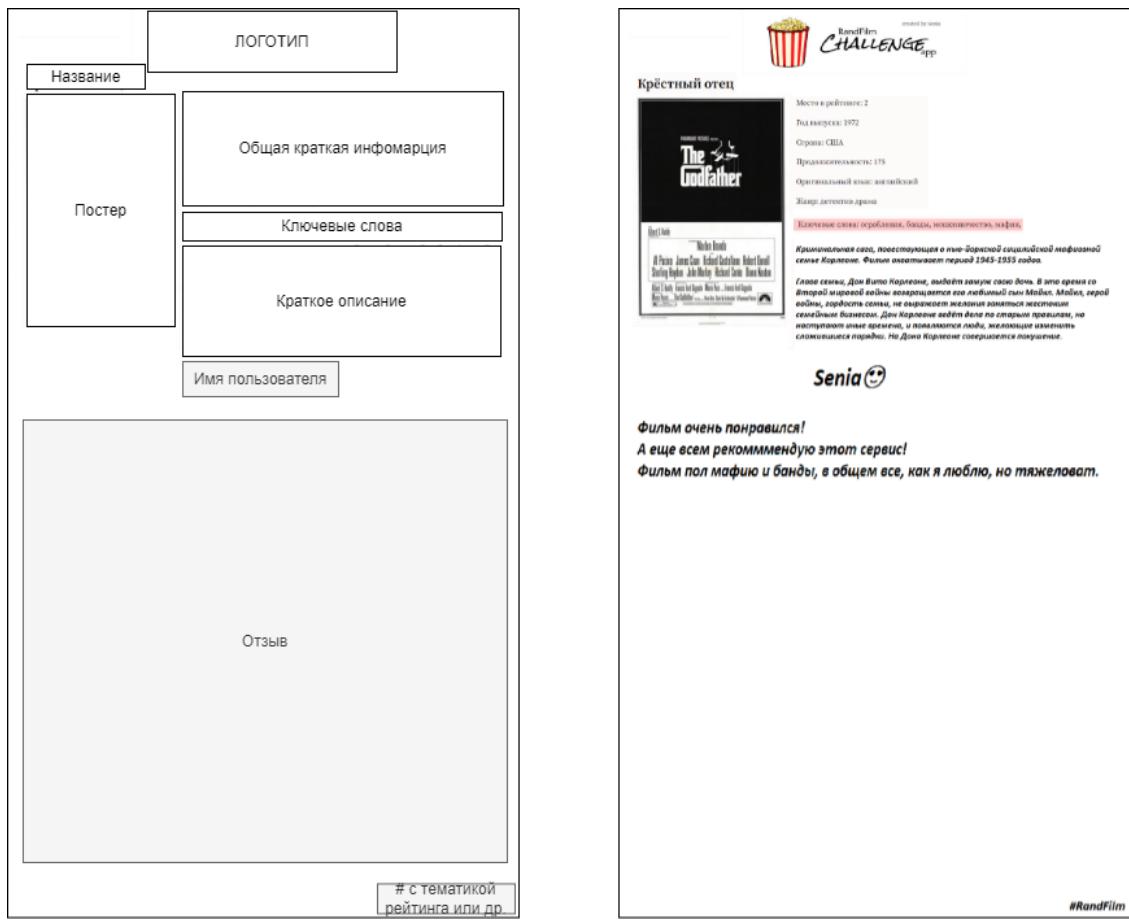


Рисунок 2.7 – Шаблон фото с отзывом о пользователе

Компонент представления общего и локального (составленного по подобранной кинокартине) графа знаний основывает на аналогичной реализованной функции компонента agraph streamlit.

2.3 Проектирование пользовательского интерфейса

Эффективный интерфейс должен принимать во внимание требования описанные в п. 1.5 и на каждом этапе работы разрешать только соответствующий набор действий и предупреждать пользователей о тех ситуациях, где они могут повредить системе или фрагментам базы знаний.

Интерфейс должен быть очевидными, простым и внушать своему пользователю чувство контроля. т.е пользователь мог одним взглядом окинуть весь спектр своих возможностей и понять, как достичь своих целей.

Интерфейс должен выполнять максимум работы, требуя при этом минимум информации от пользователя.

Интерфейс должен быть визуально приятным для пользователя и не должен нарушать сохранившиеся у него ментальные модели пользования подобными системами.

Таким образом для каждого критерия были подобраны максимально удобные для пользователя виджеты взаимодействия, а именно:

- а) checkbox - для выбора необходимых пользователю критериев;
- б) slider - для указания границ диапазона критериев год, продолжительность и место в рейтинге;
- в) selectbox - для критериев страна, оригинальный язык, тип;
- г) multiselect - для критериев жанр и теги.

Основную информацию решено расположить справа, а вспомогательную (прим. информация о приложении, граф, панель с отзывом) слева, в связи с аналогичным расположением модулей на различных кино порталах. Также принято решение реализовать область с плеером, чтобы обеспечить пользователя всем необходимым и не вынуждать искать его дополнительный контент на посторонних порталах.

Шаблоны пользовательского интерфейса представлены на рис. 2.8 и 2.9.

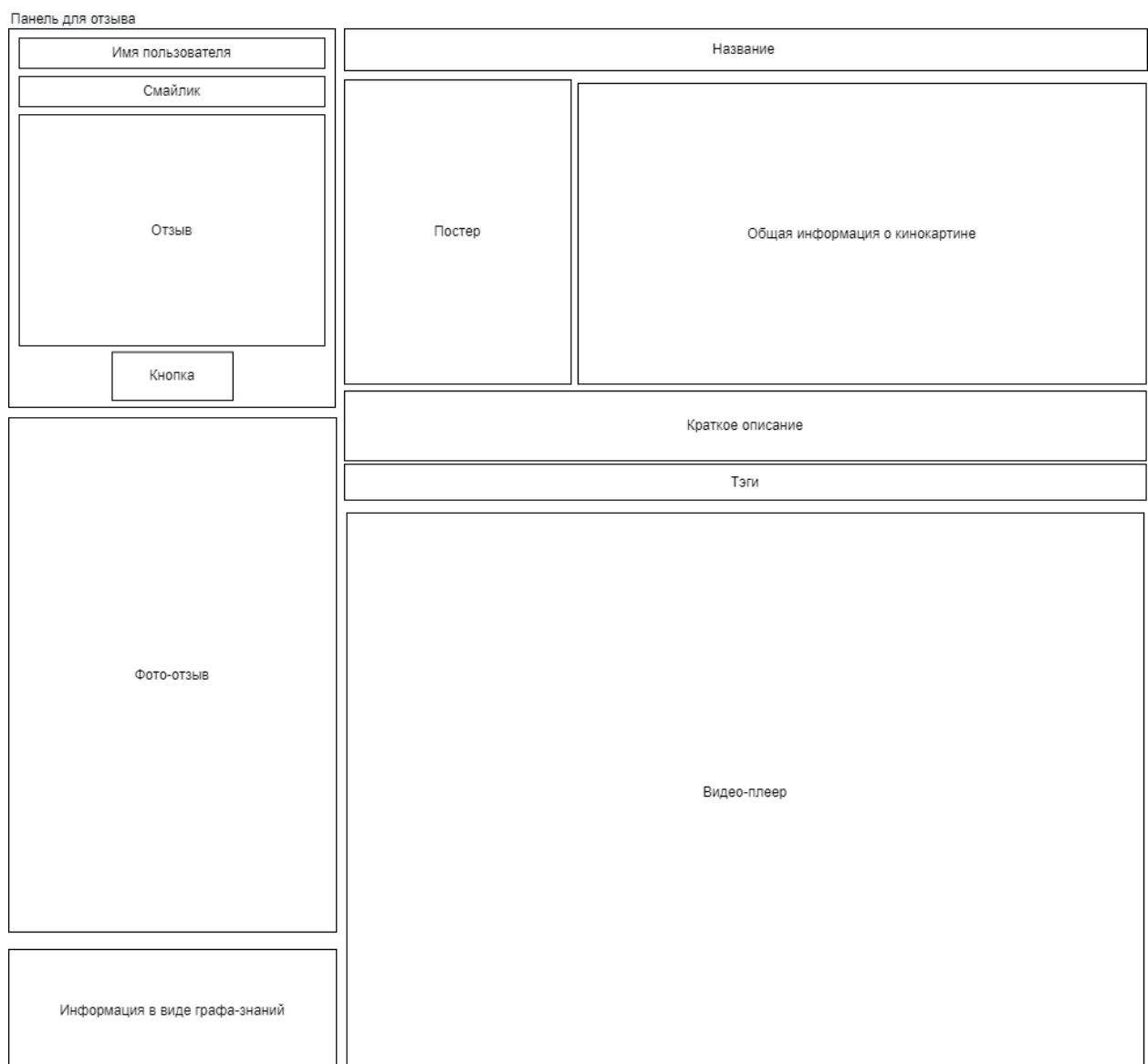


Рисунок 2.8 – Шаблон пользовательского интерфейса вывода информации

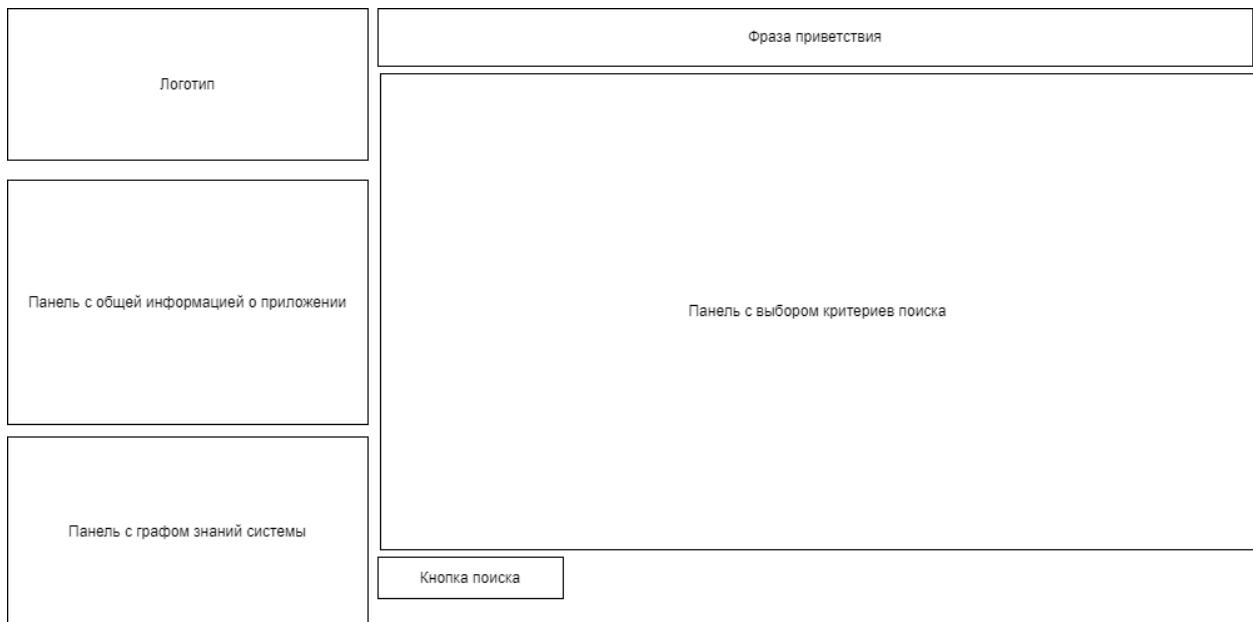


Рисунок 2.9 – Шаблон пользовательского интерфейса ввода информации

В качестве названия выберем легко произносимое и говорящее название RandFilm и спроектируем запоминающийся и не менее простой логотип.



Рисунок 2.10 – Логотип системы

3 РАЗРАБОТКА СИСТЕМЫ

3.1 Реализации знаний предметной области

По результатам поставленных задач на базе возможностей agraph был реализован граф знаний содержащий описанные в п. 2.1 знания. Итоговый граф состоит из 9 различных отношений, а также отношения включения одного подмножества в другое. 100 узлов с кинокартинами и более чем из 1000 узлов с их характеристиками.

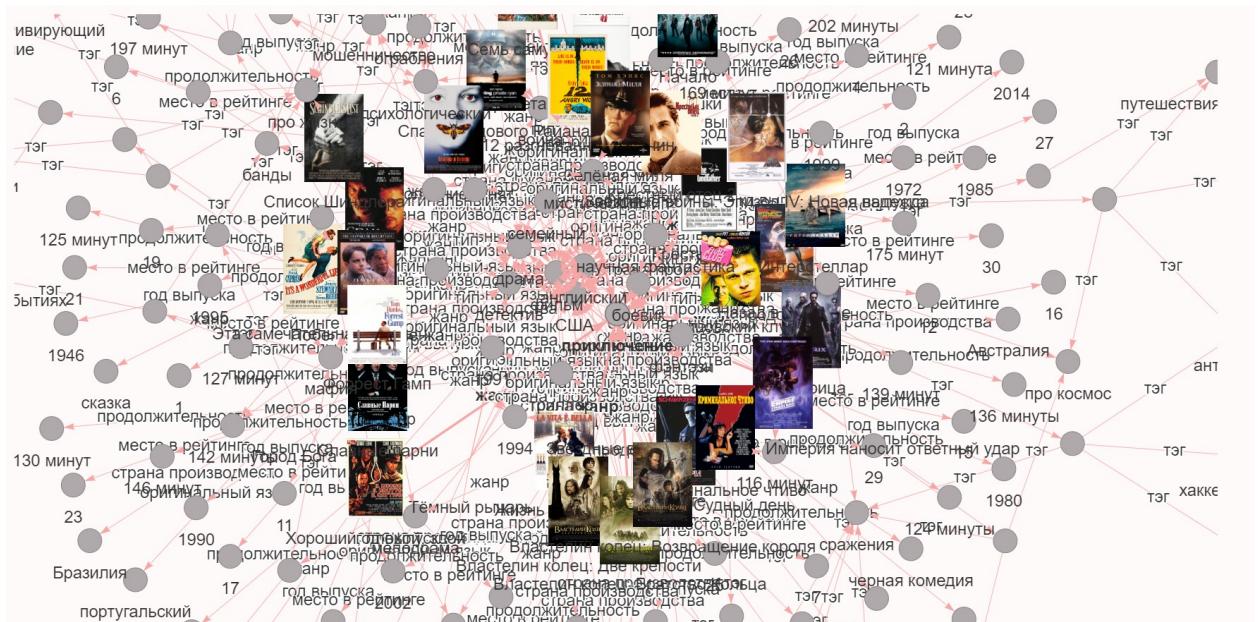


Рисунок 3.1 – Часть графа знаний системы

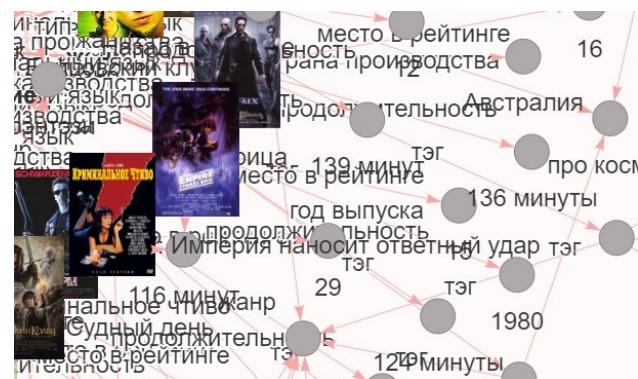


Рисунок 3.2 – Часть графа знаний системы

Пример добавления кинокартины в коде представлен на рис. 3.2. Благодаря компоненту agraph добавление является очень простым и интуитивно понятным, что позволяет при необходимости с легкостью создавать подобные графы-знаний и интегрировать их в систему.

```

147 # 3 - Тёмный рыцарь
148 text = """Бэтмен поднимает ставки в войне с криминалом. С помощью лейтенанта Джима Гордона и прокурора Харви Дента он на
149 kg_nodes_film.append(Node(id="Тёмный рыцарь",
150                         label="Тёмный рыцарь",
151                         title=text,
152                         size=25,
153                         link="https://youtu.be/sVJRsibWr70",
154                         shape="image",
155                         #image="https://upload.wikimedia.org/wikipedia/ru/f/f4/Тёмный_рыцарь_%282008%29_постер.jpg"
156                         image="https://clck.ru/32o6f2"))
157
158 kg_nodes_year_of_issue.append(Node(id="2008", label="2008", color="#b0a9ab", size=10, shape="dot"))
159 kg_edges_year_of_issue.append(Edge(source="Тёмный рыцарь", label="год выпуска", target="2008"))
160
161 kg_nodes_duration.append(Node(id="152", label="152 минут", color="#b0a9ab", size=10, shape="dot"))
162 kg_edges_duration.append(Edge(source="Тёмный рыцарь", label="продолжительность", target="152"))
163
164 kg_edges_country.append(Edge(source="Тёмный рыцарь", label="страна производства", target="США"))
165 kg_edges_original_language.append(Edge(source="Тёмный рыцарь", label="оригинальный язык", target="английский"))
166
167 kg_nodes_genre.append(Node(id="боевик", label="боевик", color="#b0a9ab", size=10, shape="dot"))
168 kg_edges_genre.append(Edge(source="Тёмный рыцарь", label="жанр", target="боевик"))
169 kg_edges_genre.append(Edge(source="Тёмный рыцарь", label="жанр", target="детектив"))
170 kg_edges_genre.append(Edge(source="Тёмный рыцарь", label="жанр", target="драма"))
171
172 kg_nodes_place.append(Node(id="3", label="3", color="#b0a9ab", size=10, shape="dot"))
173 kg_edges_place.append(Edge(source="Тёмный рыцарь", label="место в рейтинге", target="3"))
174 kg_edges_type.append(Edge(source="Тёмный рыцарь", label="тип", target="фильм"))
175
176 kg_nodes_tag.append(Node(id="тэги Тёмный рыцарь", label="тэги Тёмный рыцарь", color="#b0a9ab", size=10, shape="dot"))
177 kg_edges_tag.append(Edge(source="Тёмный рыцарь", label="тэг", target="тэги Тёмный рыцарь"))
178
179 kg_nodes_tags.append(Node(id="супергерой", label="супергерой", color="#b0a9ab", size=10, shape="dot"))
180 kg_edges_tags.append(Edge(source="тэги Тёмный рыцарь", label="тэг", target="супергерой"))
181 kg_edges_tags.append(Edge(source="тэги Тёмный рыцарь", label="тэг", target="мафия"))

```

Рисунок 3.3 – Листинг программы по созданию элементов графа знаний

3.2 Реализация компонента обработки предметной области

Компонент создания фото- отзыва реализован на базе библиотеки pillow и состоит из последовательного добавления полей выделенных на шаблоне рис. 2.7 на новую картинку. Данные для отзыва поступают от пользователя(имя, отзыв) и вызывают от подметодов компонента поиска.

```

13 # ФОТО-отзыв
14 def get_post(name, smile, review):
15     st.balloons()
16
17     #logo
18     new_img = Image.new('RGB', (1080, 1920), '#fffafa')
19     logo = Image.open("logo.png")
20     logo = logo.resize((logo.width // 2, logo.height // 2))
21     new_img.paste(logo, (330, 5), logo)
22     #title
23     font = ImageFont.truetype("arial.ttf", size=36)
24     pencil = ImageDraw.Draw(new_img)
25     pencil.text((50, 150), str(get_title()), font=font, fill='black')

```

Рисунок 3.4 – Часть листинга программы по созданию фото- отзыва

По результатам поставленных задач был реализован компонент поиска подходящего кино продукта. Поиск основывается на постепенном отсеивании неподходящих фильмов исходя из введенных пользователем критерий(если таковые имеются), далее программа выбирает либо самый подходящий из найденных вариантов, либо случайный из подходящих(если таковых несколько).

```

106     if country_checkbox:
107         for i in ans:
108             for j in kg_top100.kg_edges_country:
109                 if (i.label == j.source) and (str(country) == str(j.to)):
110                     a.append(i)
111             ans = a
112             a = []
113
114     if language_checkbox:
115         for i in ans:
116             for j in kg_top100.kg_edges_original_language:
117                 if (i.label == j.source) and (str(language) == str(j.to)):
118                     a.append(i)
119             ans = a
120             a = []

```

Рисунок 3.5 – Часть листинга программы по поиску ответа на запрос пользователя

3.3 Реализация пользовательского интерфейса и тестирование системы

Пользовательский интерфейс был реализован на базе возможностей фреймворка streamlit и на основе шаблонов ПИ рис. 2.8 и 2.9.

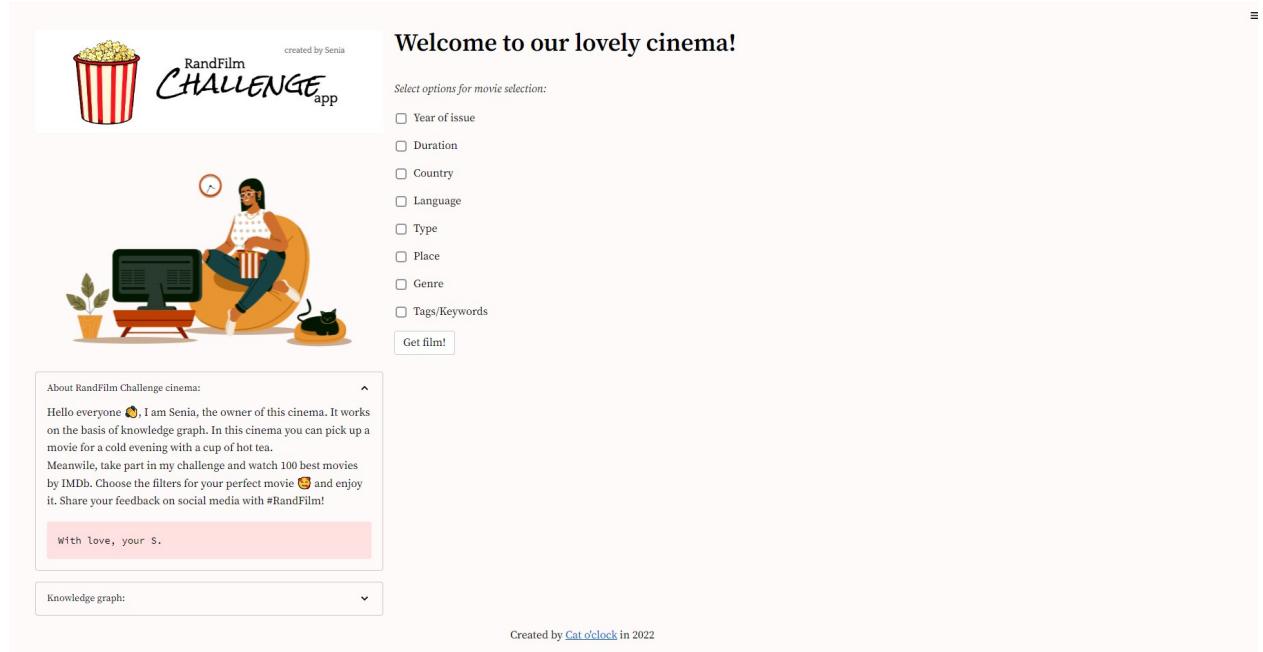


Рисунок 3.6 – Пользовательский интерфейс

На странице приложения можно увидеть боковую панель с общей информацией о приложении.



Рисунок 3.7 – Пользовательский интерфейс

Общий граф знаний, на котором работает система.

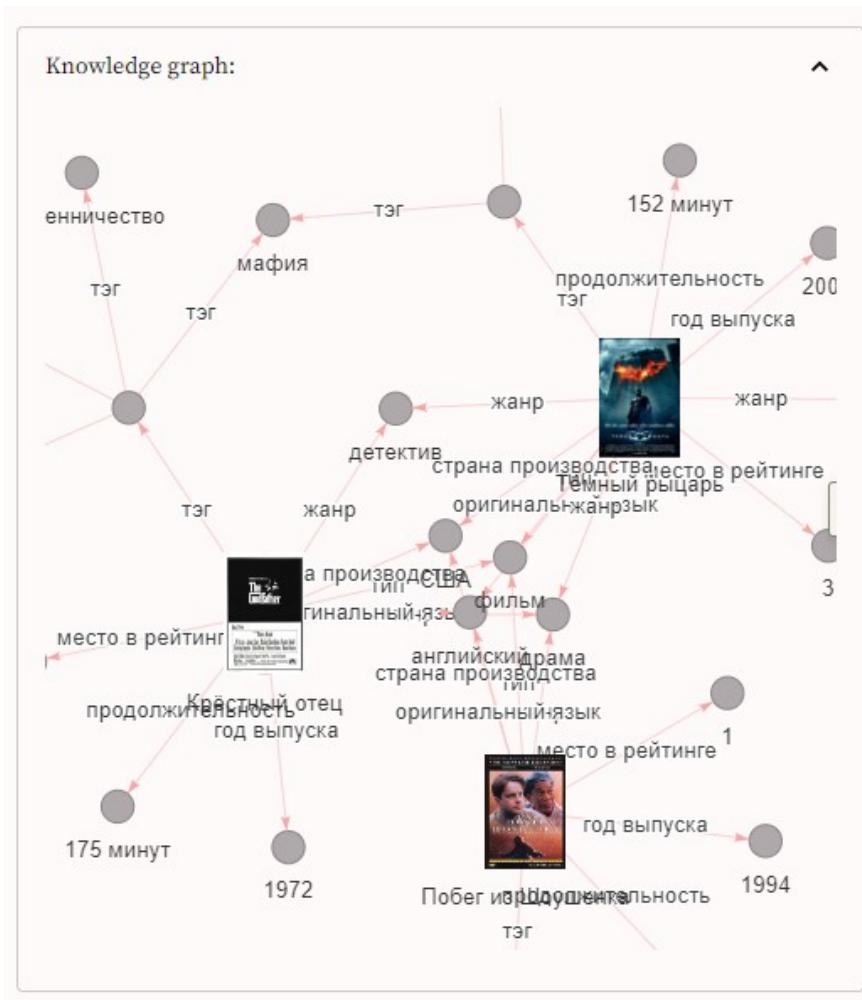


Рисунок 3.8 – Пользовательский интерфейс

А также основную панель с приветствием и призывом выбрать критерии для поиска.

Welcome to our lovely cinema!

Select options for movie selection:

Year of issue

Duration

Choose duration (in minutes)

60 78 184 300

Country

Language

Type

Choose type

Place

Genre

Choose genre

детектив

Tags/Keywords

Рисунок 3.9 – Пользовательский интерфейс

После выбора критериев необходимо нажать на кнопку "Get Film" сначала появится поле загрузки информации, а затем приложение выдаст вам всю необходимую информацию для дальнейшего взаимодействия.



Рисунок 3.10 – Пользовательский интерфейс

Слева от информации о фильме и от видеоплейера также появится боковая панель, с дублированием данных в виде графа знаний и полем для отзыва, которое пользователь может заполнить для получения фотоконтента(для своих социальных сетей, либо личных архивов и др.). Отзыв состоит из трех пунктов - ввод имени, выбор смайлика с реакцией и, непосредственно, самого отзыва. Любое из полей может остаться незаполненным по желанию пользователя.

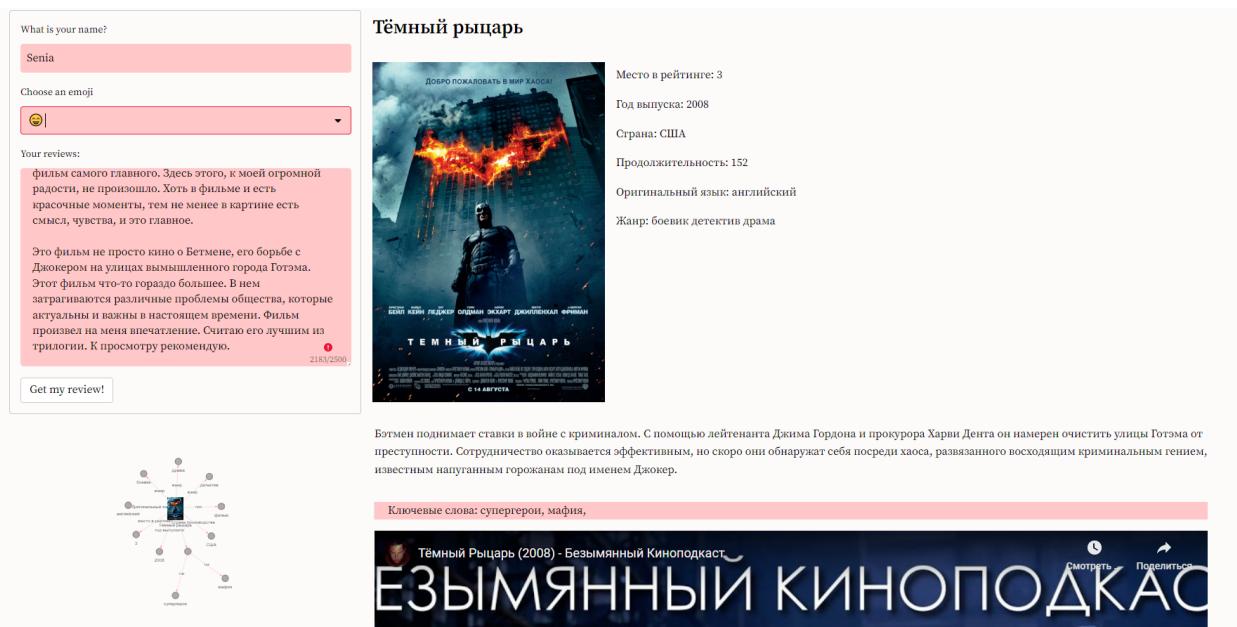


Рисунок 3.11 – Пользовательский интерфейс

После ввода необходимой информации пользователю необходимо нажать кнопку "Get review" программа запустит анимацию шаров.

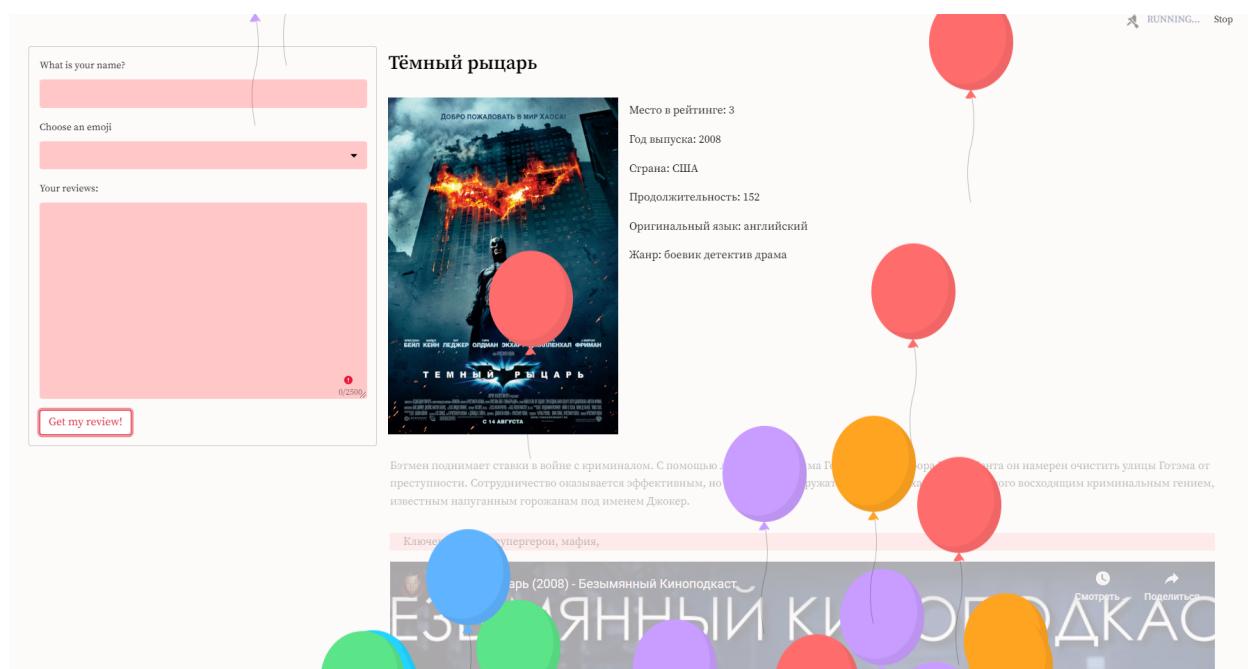


Рисунок 3.12 – Пользовательский интерфейс

После чего программа сгенерирует и загрузит в левую часть экрана фото-отзыв.

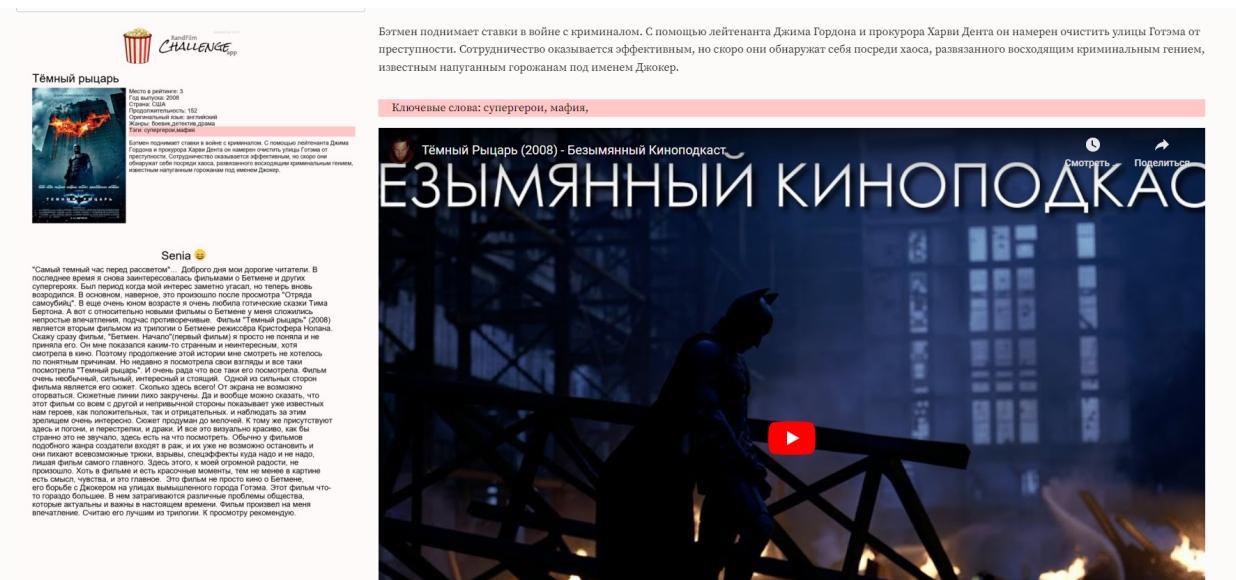


Рисунок 3.13 – Пользовательский интерфейс

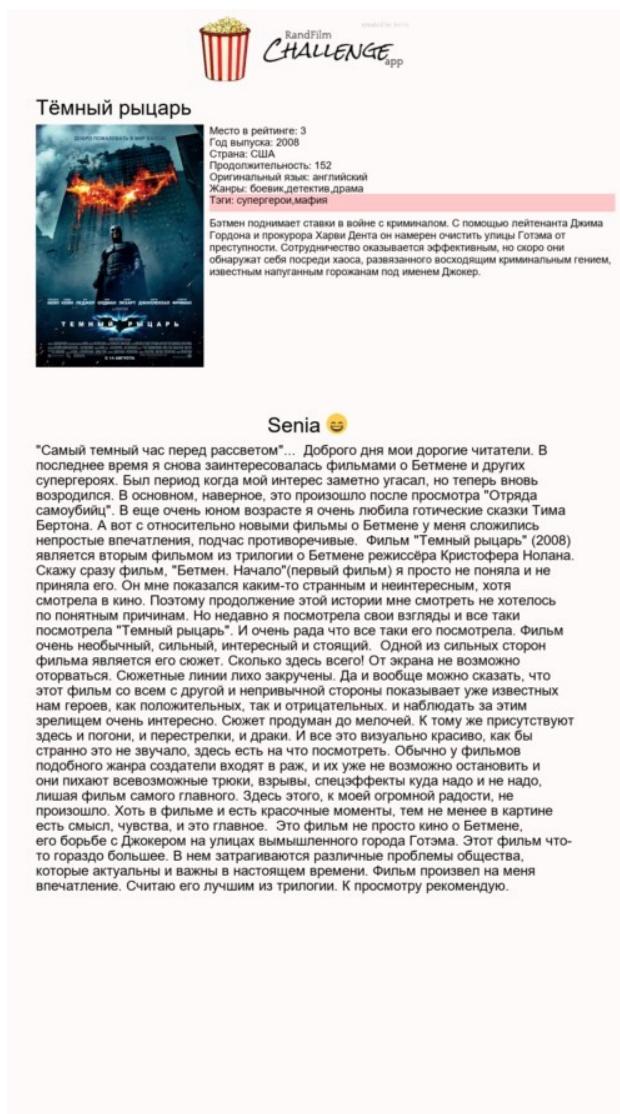


Рисунок 3.14 – Отзыв пользователя в виде фото-контента

Таким образом мною была проведена разработка необходимого функционала, тестирование работоспособности и отказоустойчивости разрабатываемой ИСС по поставленным задачам.

Данная программа создана с целью помочи в рациональной организации свободного времени пользователей: подбора фильмов, который поможет правильно, с пользой и качественно отдохнуть(за счет освобождения пользователя от бремени выбора и проверки качества кинокартины).

А также с целью предоставления пользователям возможности дальнейшей обработки и применения полученной информации (через написание отзыва), для разрушения стереотипной и разрушающей системы исключительного потребления информации, вместо его последующего применения в жизни. Данным отзывом пользователь может поделиться на личных страницах социальных сетей, либо оставить для личного пользования (прим. ведение дневника).

Данный приложение является узконаправленной, специализированной программой не имеющей аналогов. Имеет простой и понятый пользовательский интерфейс, не отвлекающий, а сопутствующий реализации главной задачи.

Код приложения написан по принципу "Открыт для расширения, закрыт для модификации". Данная версия приложения открыта для дополнения и расширения, а ее реализация способствует своему развитию.

ЗАКЛЮЧЕНИЕ

В результате были выполнены следующие задачи:

- Проведен Анализ;
- Изучены новые способы реализации поставленных целей;
- Проведен подбор информации, ее обработка и представление в виде графа знаний - реализовано 100 фильмов (10 типов отношений и понятий, а также более 1000 структурированных в иерархическом виде экземпляров);
- Разработаны компоненты обработки, предназначенные для генерации новых знаний в виде фото-контента и поиска подходящих под запросы пользователя элементов обозначенных ПрО;
- Спроектирован и реализован эффективный пользовательский интерфейс, удовлетворяющий все выявленные в Анализе требования;
- Разработанная система протестирована на корректность работы.

Также в данном семестре совместно с проектом "Кинофильмы" была проведена работа над корректировкой и актуализацией БЗ ИСС по Кинофильмам на базе технологии OSTIS, выявлены и устраниены синтаксические и семантические ошибки - исправлено 34 понятия, актуальная база добавлена на официальный репозиторий github, обозначены цели и задачи(создание документации проекта, создание полноценных приложений (прим. чат-бот) на базе системы и др.) для дальнейшего развития проекта(в том числе и на данный семестр), проведены собеседования в указный выше проект. На протяжении всего семестра происходило курирование проекта, содействие в проектировании и реализации целей проекта. Что является логическим завершением работы над указанным проектом, а также завершением темы всех курсовых работ(проектов) предыдущих лет.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

- [1] Wikipedia [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/>. — Дата доступа: 30.11.2022.
- [2] Google trends[Электронный ресурс]. — Режим доступа: <https://trends.google.ru/trends/?geo=BY>. — Дата доступа: 30.11.2022.
- [3] Кинорейтинг[Электронный ресурс]. — Режим доступа: <https://clck.ru/9HXwE>. — Дата доступа: 30.11.2022.
- [4] Курсовое проектирование[Компьютерная презентация]. — кафедра ИИТ, 5-й семестр, Ковалёв М.В. и Садовский М. Е.
- [5] habr[Электронный ресурс]. — Режим доступа: <https://habr.com/ru/company/piter/blog/530514/>. — Дата доступа: 30.11.2022.
- [6] Интеллектуальный интерфейс [Электронный ресурс]. — Режим доступа: https://studopedia.ru/18_40648_intellektualniy-interfeys.html. — Дата доступа: 30.11.2022.
- [7] Ментальные модели[Реферат]. — кафедра ИИТ, студентка 921702 Боровская К.С.
- [8] C++ [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/C%2B%2B>. — Дата доступа: 30.11.2022.
- [9] Java [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/Java>. — Дата доступа: 30.11.2022.
- [10] Streamlit [Электронный ресурс]. — Режим доступа: <https://ru.wikipedia.org/wiki/Python>. — Дата доступа: 30.11.2022.
- [11] Streamlit [Электронный ресурс]. — Режим доступа: [://clck.ru/32pDEF](https://clck.ru/32pDEF).
- [12] Streamlit [Электронный ресурс]. — Режим доступа: <https://clck.ru/32pDE7>. — Дата доступа: 30.11.2022.
- [13] Streamlit Agraph [Электронный ресурс]. — Режим доступа: <https://blog.streamlit.io/the-streamlit-agraph-component/#you-can-run-this-without-tripystore>. — Дата доступа: 30.11.2022.
- [14] О.В. Каравеев, В.Г. Конюший. Многоагентные системы и средства их разработки / В.Г. Конюший О.В. Каравеев. — 4-е изд. — Издательство «Советская литература», 2009. — 254 с.