МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК

ВИПУСКНА РОБОТА

на тему:

«Сайт домашньої фільмотеки»

Завідувач
випускаючої кафедри **Довбиш А.С.**

Керівник роботи **Проценко О. Б.**

Студента групи ІН – 72 **Покутній О. Ю.**

СУМИ 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Кафедра комп'ютерних наук

Затверджую_____

Зав. кафедри Довбиш А.С.

"_____" _____2021 р.

ЗАВДАННЯ

до випускної роботи

Студента четвертого курсу, групи ІН-72 спеціальності "Комп'ютерні науки" денної форми навчання Покутнього Олексія Юрійовича.

Тема: " Сайт домашньої фільмотеки "

Затверджена наказом по СумДУ

№_____ от _____2021 г.

Зміст пояснювальної записки: 1) аналітичний огляд методів відслідковування користувачів; 2) постановка завдання й формування завдань дослідження; 3) опис основних положень, технологій, бібліотек і критеріїв, що використовуються додатком для відслідковування та обробки користувачів деякого веб-ресурсу; 5) розробка сайту домашньої фільмотеки; 6) аналіз результатів.

Дата видачі завдання "_____"_____2021 р.

Керівник випускної роботи _____ Проценко О.Б.

Завдання прийняв до виконання ____ Покутній О.Ю.

РЕФЕРАТ

Записка: 51 стор., 31 рис., 14 джерел.

Об’єкт дослідження — сайт домашньої фільмотеки.

Мета роботи — є створення інтерфейсу за допомогою мови гіпер-тесктової розмітки HTML та каскадної таблиці стилів CSS, відображення даної бази даних за допомогою мови програмування JavaScript, створення бази даних фільмів за допомогою SQL Server Management Studio.

Результати — було розроблено веб-сайт для перегляду фільмів. Створено нормалізовану базу даних. За допомогою сформованих стилів **HTML** та **CSS** було розроблено адаптивний веб-сайт та зроблено сайт динамічним за допомогою **JavaScript**.

ВЕБ-ДОДАТОК ДЛЯ ПЕРЕГЛЯДУ ФІЛЬМІВ ОНЛАЙН,
МЕТОД ПЕРЕГЛЯДУ, JAVASCRIPT, SQL,
АНАЛІТИЧНА СИСТЕМА

ЗМІСТ

ВСТУП

Актуальність:

На даний момент існує багато додатків із фільмами. У кожного з них красиве оформлення, достатньо великий набір різноманітних фільмів: від мелодрам до фільмів жахів, від Квентіна Торрентіно до Стівена Спілберга. Але в них є також достатньо мінусів:

1. Постійна реклама (при вході на сторінку, на задньому фоні сторінки, при натисканні на будь-яку невідому кнопку, та навіть під час перегляду фільма.
2. Не всі з цих додадків вчасно викладають новинку фільму в HD якості, а якщо викладають, то це найчастіше піратська версія.
3. Недостатня якість зображення.
4. Відсутність вибору мови країни

Так як автор є щірим шанувальником фільмів різних жанрів, дуже часто п омітно багато проблем на подібних проектах. Проблема в них лише одна: недостатньо розвинена та рідкооновлювана база даних. Саме тому а втор узяв тему фільмів для дослідження.

Предметом дослідження роботи є комплекс вправ, націлений на відображення фільмів у веб-додатку, що складається з двох частин : бази даних та веб-сторінки.

1 ІНФОРМАЦІЙНИЙ ОГЛЯД

1.1 Огляд проблемної області

1.1.1 Поняття “фільмотека”

Фільмотека, синематека - установа, що займається збором, зберіганням, технічним опрацюванням кінофільмів, а також їх вивченням і популяризації. У 1938р. організувалася Міжнародна федерація кіноархівів (FIAF), в завдання якої входить популяризація кращих творів світового кіно, міжнародний обмін фільмів. Серед найбільших фільмотек: Держфільмофонд Росії, Французька синематека, Національна кінобібліотеку Британського кіноінституту, Центральний фільмархів Польщі, кінобібліотеку музею сучасного мистецтва в США. [1]

1.1.2 Класифікація фільмів

Жанри фільмів важливі для сценаристів та кіноаудиторії, оскільки вони визначають тональне сподівання. Деякі основні жанри фільмів включають:

Бойовик: Фільми у жанрі бойовиків швидко розвиваються і включають багато дій, таких як сцени боїв, сцени погоні. У них можуть бути супер герої, бойові мистецтва або захоплюючі трюки. Ці високооктанові фільми більше стосуються виконання сюжету, а не самого сюжету. Бойовики призначені для захоплюючого перегляду та залишення членів аудиторії на краю своїх місць. Фільми про поліцейських, катастрофи та деякі шпигунські фільми підпадають під категорію бойовиків.

Пригоди: жанр пригод настільки схожий на жанр бойовиків, що пригодницькі фільми часто класифікуються як бойовики / пригодницькі фільми. Фільми у пригодницькому жанрі, як правило, містять ті самі основні жанрові елементи бойовика, де обстановка є головною різницею. Пригодницькі фільми, як правило, знімаються в екзотичному, далекому або незнайомому регіоні.

Комедія: Комедійні фільми веселі та розважальні. Фільми цього жанру зосереджені навколо комедійної передумови - зазвичай ставлять когось у

складну, забавну або жартівливу ситуацію, з якою вони не готові впоратися. Хороші комедійні фільми менше стосуються постійних жартів, а більше - представлення загальноприйнятої, реальної історії зі складними персонажами, які засвоюють важливий урок. Темна комедія (чорна комедія), романтична комедія, пародія / підробка та комедія - це приклади комедійних піджанрів.

Драма: У жанрі драми представлені історії з високими ставками та безліччю конфліктів. Вони керуються сюжетом і вимагають, щоб кожен персонаж і сцена рухали історію вперед. Драми дотримуються чітко визначеної структури розповідного сюжету, зображуючи реальні сценарії або екстремальні ситуації з емоційно керованими персонажами. Фільми, що підпадають під драматургічні піджанри, включають історичну драму, романтичну драму, підліткову драму, медичну драму, докудраму та фільм-нуар.

Фентезі: Фільми у жанрі фентезі містять магічні та надприродні елементи, яких немає в реальному світі. Хоча деякі фільми поєднують реальну обстановку з фантастичними елементами, багато хто створює цілком уявні всесвіти зі своїми законами, логікою та популяціями уявних рас та істот. Як і науково-фантастичні фільми, фантастичні фільми є умоглядними, але не прив'язані до реальності чи наукового факту. Висока фантазія, казки та магічний реалізм - все це піджанри фентезі.

Фільми жахів: ці фільми містять елементи, які залишають у людей надзвичайне почуття страху. Фільми жахів часто включають серійних вбивць або монстрів як наполегливих, злих антагоністів, щоб зіграти на страх або кошмари глядачів. Аудиторія, яка любить жанр жахів, шукає ці фільми спеціально для прискорення адреналіну, виробленого привидами, кров’ю, монстрами та страхами. Фільми, які потрапляють у піджанри жахів, включають історії про привидів, готичні фільми жахів, науково-фантастичні фільми жахів, надприродні фільми, темні фантастичні фільми, психологічні

фільми жахів та фільми про слєшер. Дізнайтеся, як написати сценарій жахів із нашим вичерпним путівником тут.

Мюзикли: музичні фільми вплітають пісні або музичні номери в розповідь, щоб розвивати історію або подальший розвиток персонажів. Мюзикли часто прив'язують до любовних фільмів, але не обмежуються цим жанром. Музичні фільми включають великі сценичні постановки, інтегруючи важливі передумови або елементи характеру в послідовності.

Таємничі фільми: це фільми про головоломку і в них часто присутній детектив або аматорський злодій, який намагається її розгадати. Таємничі фільми сповнені напруги, і головний герой шукає підказки чи докази протягом усього фільму, поєднуючи події та опитуючи підозрюваних, щоб вирішити центральне питання. Нуари, що проварюються, та поліцейські процедури - це дві підкатегорії, які часто підпадають під жанр таємниць.

Романтичні: Романтичні фільми - це любовні історії. Вони зосереджуються навколо двох головних героїв, які досліджують деякі елементи любові, такі як стосунки, жертви, шлюб, одержимість чи руйнування. У романтичних фільмах іноді трапляються такі труднощі, як хвороба, зрада, трагедія чи інші перешкоди для подолання любовних інтересів. Романтичні комедії, готична романтика та романтичний екшен - деякі популярні романтичні піджанри.

Наукова фантастика: науково-фантастичний жанр будує світи та альтернативні реалії, наповнені уявними елементами, яких немає в реальному світі. Наукова фантастика охоплює широкий спектр тем, які часто досліджують подорожі в часі, космічні подорожі, визначаються в майбутньому та розглядають наслідки технологічного та наукового прогресу. Науково-фантастичні фільми, як правило, передбачають скрупульозне будівництво світу з великою увагою до деталей, щоб аудиторія повірила в історію та всесвіт.

Спортивні: Фільми у спортивному жанрі будуть зосереджені навколо команди, окремого гравця чи вболівальника, а сам вид спорту буде використаний для мотивації сюжету та продовження історії. Ці фільми не повністю зосереджені на самому виді спорту, однак, головним чином, використовуючи його як фон, щоб надати контекст емоційним дугам головних героїв. Спортивні фільми можуть бути драматичними або комічними, і часто вони мають на меті алегоричні.

Трилер: Трилери вміло поєднують загадку, напругу та очікування в одну захоплюючу історію. Успішні трилери мають гарний темп, часто представляють червоні оселедці, розкривають повороти сюжету та розкривають інформацію в потрібні моменти, щоб заінтригувати аудиторію. Трилери часто включають аспект "тикаючих годин", коли ставка встановлюється на обмежену кількість часу. Кримінальні фільми, політичні трилери та техно-трилери представлені в жанрі трилер.

Вестерн: вестерни розповідають казку про ковбоя або стрільця, який переслідує поза законом на Дикому Заході. Головний герой часто прагне помститися і наприкінці зіткнеться зі злочинцем на двобої або перестрілці. Вестерни - це яскраві постановки на американському Заході, такі як пустеля, гори чи рівнини, які можуть надихати та інформувати персонажів та дії. Космічні вестерни та науково-фантастичні вестерни - це всі піджанри західної категорії.

1.2 Огляд подібних рішень

Ми живемо в XXI столітті, в час інформаційних технологій. Зараз ні одна людина не може уявити своє життя без гаджетів. Вони роблять наше життя легшим та більш комфортним. Різноманітні цифрові пристрої допомагають у різних сферах нашого життя від роботи до відпочинку. На мою думку, гарний відпочинок – один з головних чинників успіху, тому що лише людина, яка добре відпочила, зможе плідно працювати. Я думаю не велика кількість

людей відмовиться подивитися гарний фільм зі своєю сім’єю після важкого трудового дня тому я і обрав саме тему кінематографа.

Зараз є велика кількість різних сайтів для перегляду фільмів і кожен може вибрати той, який підходить саме йому. Одні з найпопулярніших: **hdrezka, stream, megogo, divan.tv, ivi.** Кожен з цих сайтів намагається привабити якомога більше користувачів на свій сайт за допомогою різних методів, наприклад: унікальність, зручність у використанні, приємний інтерфейс, гарна якість зображення.

Netflix

Американський відеосервіс, що належить однойменній компанії зі Сполучених Штатів, яка була заснованої в 1997 році. Спочатку вона займалася кінопрокатом, відправляючі диски з фільмами поштою. У 1999 році був запущений інтернет-сервіс для доступу до відео за запитом. З 2013 року Netflix запустив власне виробництво телесеріалів, які доступні тільки користувачам платформи. Особливість цього сервісу - серіали тут з’являються відразу цілим сезоном і доступні для перегляду в будь-який час після релізу. Доступ до сервісу платний і обмежений територією США і кількох країн світу. Тестовий період для безкоштовного користування послугою становить 30 днів.

IVI

Російська медіакомпанія, фільми у цій платформі доступні для безкоштовного звантаження на території Європи. За платну підписку надається доступ до преміум-контенту в пакеті «ivi +». Туди входять деякі платні фільми і категорія «Блокбастери». Сервіс заробляє на рекламі в форматі відеороликів до, під час і після перегляду відео. Підписка коштує 100грн(на 01.06.2021) на місяць, ціна кожного блокбастера визначається окремо.

Stream

Цей відеосервіс працює за дещо іншою схемою, ніж ivi. Тут також є безкоштовний каталог (фільми, серіали, мультфільми), є підписка, можна взяти фільми напрокат, але на відміну від ivi можна ще й скачати фільм для офлайнового перегляду. Фільм, узятий напрокат, можна подивитися протягом 7-30 днів, на сам перегляд відводиться дві доби. Вартість прокату – 10-20грн(на 01.06.2021). Фільмів тут доступно менше, ніж на ivi, але ви мажте змогу переглянути їх без реклами.

Megogo.net

Онлайн-кінотеатр доступний у 15 країнах СНД. Споживання контенту тут безкоштовне, платний тариф надається для користувачів, які проживають за межами цих 15 країн. Послуга почала працювати в листопаді 2011 року. Щомісячна аудиторія послуги - 27 мільйонів чоловік.

Цей інтернет-кінотеатр заробляє на відеорекламі. Реклама відображаються глядачеві перед переглядом фільму, після нього та коли ви робите паузу. Ряд нових релізів доступні лише після оплати вартості вибраного фільму чи серіалу.

Divan.tv

Українська служба доступу до Інтернет-телебачення та легального відеовмісту. Проект розпочався в червні 2011 року. Спочатку позиціонувався як телевізійна приставка, користувач може переглядати телевізійні канали, фільми та слухати аудіоконтент. Автором проекту та його першим інвестором став український бізнесмен Андрій Колодюк, засновник Aventures Capital. На даний момент існує Інтернет-сервіс, який можна використовувати як з придбанням приставки, так і на «розумних» телевізорах, робочих столах, смартфонах і планшетах.

Головні переваги перегляду фільмів в Інтернеті - це, звичайно, нескінченно широкий вибір на будь-який, навіть найвибагливіший смак. Крім того, немає залежності від часу показу фільму, ви можете дивитись у будь-

який зручний час, натискати паузу, коли захочете. Основним недоліком є те, що якість фільмів часто залишає бажати кращого. Це стосується переважно нових фільмів.

1.3 Постановка задачі

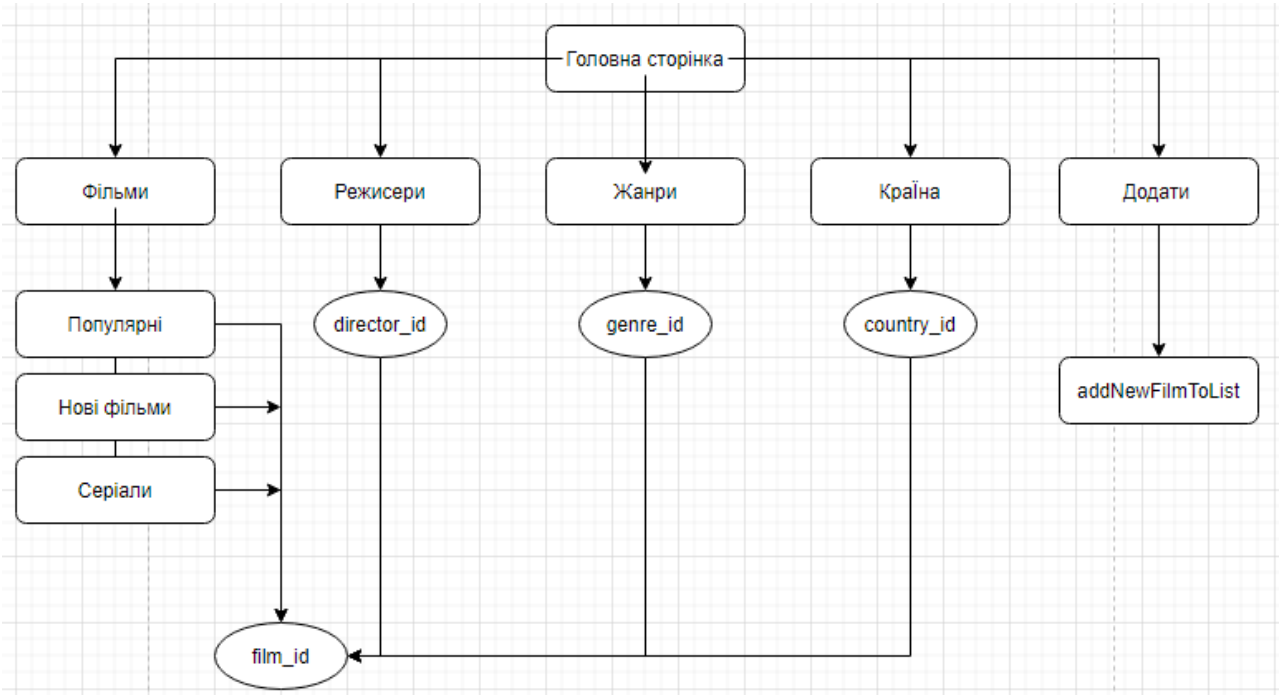
1.3.1 Предмет розробки

Предметом розробки є сайт домашньої фільмотеки.

Призначення сайту - зробити пошук фільму для перегляду більш зручним.

1.3.2 Структура сайту

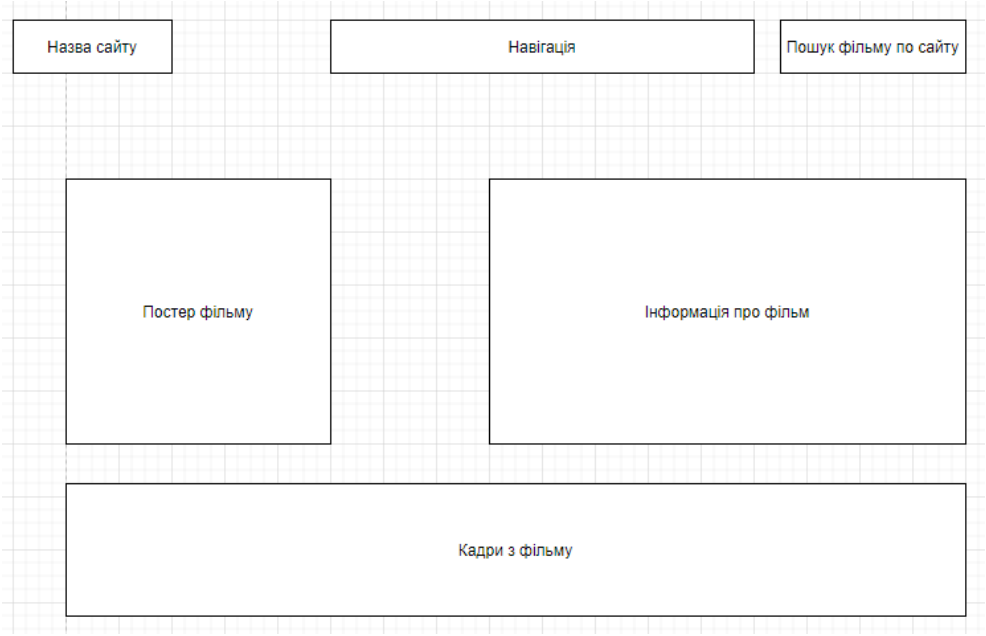
Вибір відповідного фільму не дуже просте завдання, у кожного з нас свій унікальний смак, також велику роль грає твій настрій саме під час пошуку. Тому структура сайту повинна бути легкою для зрозуміння, зручною у використанні та не містити зайвої інформації. Отже, це питання потрібно продумати дуже добре.



мал 1.1 - Структура сайту

UX дизайн сайту

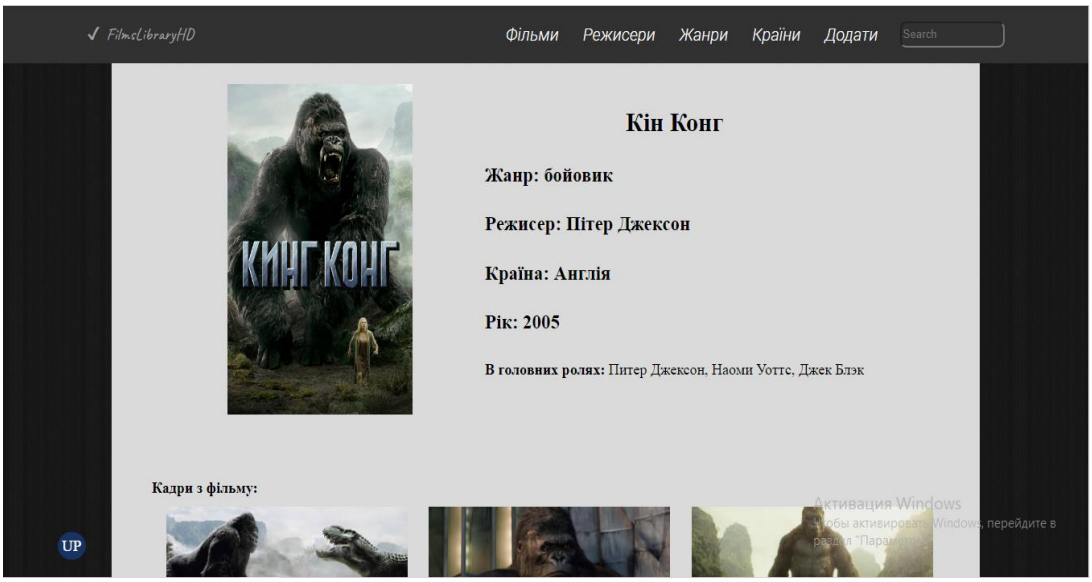
UX дизайн (User Experience) - це досвід або враження, які користувач отримує від роботи з вашим інтерфейсом. Простіше кажучи, це структура самої веб-сторінки без графічно красиво відображених елементів. На основі дизайну UX створюється UI дизайн (мал. 1.2).



мал 1.2 UX дизайн сайту

UI дизайн сайту

UI дизайн (User Interface) - це те, як виглядає інтерфейс і яких фізичних характеристик він набуває. Тобто за допомогою графічних редакторів (PhotoShop / Figma) створити візуальний шаблон сайту (вигляд в браузері) (мал. 1.3).



мал 1.3 UI дизайн сайту

1.3.3 Вимоги до сайту

Перелік вимог до даного сайту:

- **Адаптивність.** Сайт має виглядати добре як на ПК, так і в мобільній версії. Блоки не повинні виходити за межі сторінки та не може бути зайвих відступів між ними. Також слід попрацювати над текстом, щоб він не був занадто великим або навпаки малим для обох версій.
- **Система пошуку по сайту.** Щоб користувач за бажанням міг ввести назву фільму у поле для вводу та швидко знайти саме той фільм у переліку який йому потрібен.
- **Додавання фільму.** Кожен бажаючий може додати фільм у вже існуючий перелік, якщо даного фільму там немає. Він може це зробити після того, як натисне кнопку «Додати» та введе всі потрібні дані про цей фільм.
- **Плавні переходи, анімації, приємний інтерфейс.** Для того щоб клієнту було цікаво відвідувати сайт, треба додати анімації, блоки повинні плавно з’являтися на сайті, гарне поєднання кольорів, яке не повинно відразу кидатися в очі, це все повинно задовольнити користувача щоб він захотів і пізніше відвідати цей сайт.

1.3.6 Зміст сайту

Дані сайту будуть динамічно витягуватися з бази даних, яку розроблятиме розробник. Дані в базі даних будуть взяті з офіційних характеристик фільму.

1.3.7 Етапи створення сайту

- Створення концепції сайту для перегляду фільмів, розробка та узгодження технічного завдання;
- Розробка макета сайту, що включає всі графічні і інтерактивні елементи;
- Програмування та підключення модулів управління;
- Підготовка контенту - створення, оптимізація.
- Тестування сайту, внесення корективів;
- Повноцінна робота сайту

Таким чином, онлайн кінотеатр - найзручніший спосіб перегляду фільмів, тому що ти можеш дивитися улюблені фільми у гарній якості зображення будь-де, у будь-який час. Ти можеш вибрати саме той фільм, який хочеш подивитися саме у цей момент, зважаючи на свій настрій та компанію, навіть не потрібно виходити з дому, потрібен лише ноутбук, комп’ютер або смартфон з інтернетом, зараз це є не проблема.

У даному розділі було визначена назва магазину – **Films Library Hd,** було постановлене технічне завдання, за яким буде проходити розробка, розроблена структура веб-сайту: легка для зрозуміння, зручна у використанні та не містить зайвої інформації. Були визначені та розроблені UX та UI дизайни, перераховані деякі вимоги до функціонування сайту: *адаптивність, система пошуку по сайту, додавання фільму, плавні переходи, анімації, приємний інтерфейс.*

Також були описані етапи розвитку сайту домашньої фільмотеки **Films Library Hd.**

2 ВИБІР МЕТОДІВ РІШЕННЯ ЗАДАЧІ

2.1 Концепція і зміст веб-сайту

2.1.1 Основна концепція

Цільова аудиторія сайту люди, які полюбляють дивитися фільми та надають перевагу дивитися їх в мережі інтернет, а не в кінотеатрі. Головне завдання сайту - надавати необхідну інформацію відвідувачам, а так само можливість залишити коментарі про обраної організації.

Виходячи з поставлених завдань, web-сайт повинен надавати такі можливості:

- при першому відвідуванні сайту користувач повинен зрозуміти, для чого служить даний сайт, скласти загальне враження про нього, з'ясувати свої потреби по сайту. Для цього йому повинен бути доступний перелік усіх послуг;

2.1.2 Детальна концепція веб-сайту

Дизайн веб - сайту включає в себе: кольорове оформлення, елементи навігації по сайту, текстову інформацію, опис послуг пропонованих організацією. Дизайн сайту грає важливу роль в створенні самого сайту, а саме він повинен відповідати наступним вимогам:

- зовнішній вигляд сайту повинен відповідати обраній тематиці;
- навігація по сайту повинна бути зручною для користувача;
- головні матеріали, пропозиції та категорії послуг повинні бути розташовані на першому плані, щоб зацікавити відвідувача;

У верхній частині сторінки поміщається назва сайту, що відбиває його тему. Зображення у вікні браузера складається з трьох основних елементів:

- назва сайту;
- головне інформаційне меню зліва сторінки;

Назва сайту знаходиться на кожній сторінці і не змінюється в залежності від місцезнаходження на сайті. Меню призначається для навігації по сайту, в ньому відображаються посилання на всі сторінки сайту. Основний зміст сторінки змінюється в залежності від місцезнаходження користувача.

2.2 Архітектура веб-додатку

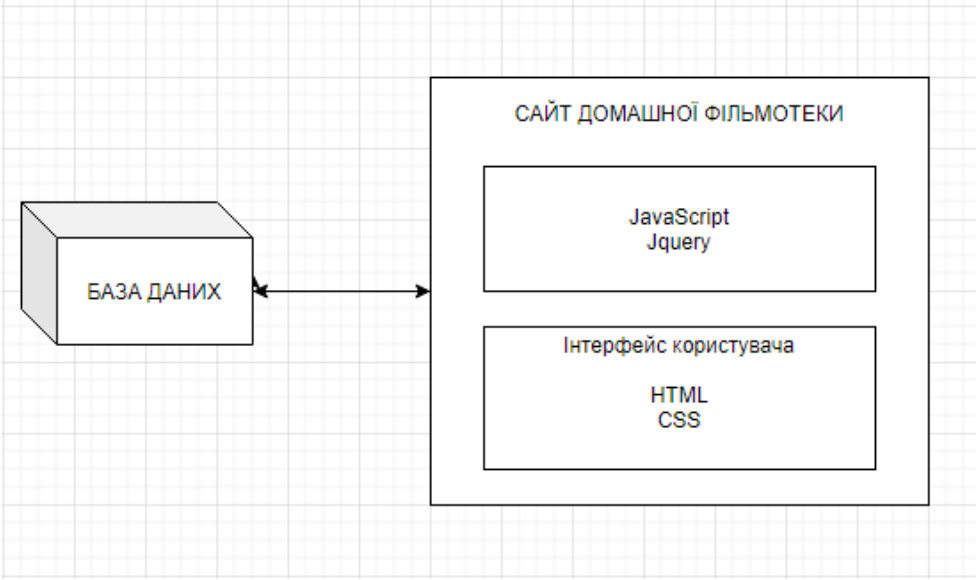
Для того, щоб сайт працював швидко та ефективно, а також працював належним чином, необхідно розробити архітектуру програмного забезпечення сайту.

Люди у світі програмного забезпечення давно сперечаються щодо визначення архітектури. Для деяких це щось на зразок фундаментальної організації системи або способу з'єднання компонентів найвищого рівня. Моє мислення щодо цього питання повністю співпадає з думкою Ральфом Джонсоном, який поставив під сумнів цю фразу, аргументуючи тим, що не існує об'єктивного способу визначити, що є фундаментальним або високим рівнем, і що кращий погляд на архітектуру - це спільне розуміння того, що експерти-розробники мають конструкцію системи.[2]

5 найкращих моделей архітектури програмного забезпечення:

- Багаторівнева архітектура
- Подієво-орієнтована архітектура
- Мікроядерна архітектура.
- Мікросервісная архітектура
- Просторова архітектура
- Космічна архітектура.

При розробці архітектури сайту домашньої кінотеки були враховані декілька основних аспектів: використання баз даних в проекті для роботи з інформацією, використання **JavaScript** та його фреймворку **jQuery**, також мова розмітки гіпертексту **HTML** і таблиця каскадних стилів **CSS**.



мал 2.1 Архітектура сайту домашньої фільмотеки

При будь-якій розробці систем, незалежно від їх складності, важливим критерієм гарної якості є простота у розумінні. Я маю на увазі, що структура розробленої системи повинна бути зрозумілою для інших розробників, оскільки більшість проектів потребують деяких оновлення та коригування, і дуже часто проекти створюються та редагуються не різними розробниками.

2.3 Програмні засоби реалізації проекту

Для розробки та впровадження сайту були обрані наступні програмні засоби:

- мова розмітки гіпертексту **html** і **css**;
- мова програмування **JavaScript** та бібліотека **jQuery**;
- бази даних в SQL Server Management Studio;
- Sublime text.

*Мова розмітки гіпертексту **HTML** та **CSS***

HTML перекладається як мова розмітки гіпертексту. Це дозволяє користувачеві створювати та структурувати розділи, абзаци, заголовки, посилання та цитати для веб-сторінок та додатків. HTML не є мовою програмування, тобто не має можливості створювати динамічну функціональність. Натомість це дозволяє упорядковувати та форматувати

документи, подібно до Microsoft Word. Під час роботи з HTML ми використовуємо прості структури коду (теги та атрибути) для розмітки сторінки веб-сайту. Наприклад, ми можемо створити абзац, помістивши вкладений текст у початковий тег і закриття .

*Мова програмування **JavaScript** та бібліотека **jQuery***

JavaScript - це мова сценаріїв, яка використовується для створення та управління динамічним вмістом веб-сайту, тобто будь-що, що переміщується, оновлюється чи іншим чином оновлюється на екрані, не потребуючи перезавантаження веб-сторінки. Такі функції, як:

- анімована графіка
- слайд-шоу фото
- пропозиції щодо автозаповнення тексту
- інтерактивні форми

Ще кращий спосіб зрозуміти, що робить JavaScript, - це думати про певні веб-функції, якими ви користуєтеся щодня і, ймовірно, сприймаєте їх як належне - наприклад, коли ваша хронологія Facebook автоматично оновлюється на вашому екрані або Google пропонує пошукові терміни на основі кількох запущених вами букв. В обох випадках - це JavaScript.

jQuery - це бібліотека **JavaScript**, яка дозволяє веб-розробникам додавати додаткові функції до своїх веб-сайтів. Після завантаження бібліотеки **jQuery** веб-сторінка може викликати будь-яку функцію **jQuery**, що підтримується бібліотекою. Поширені приклади включають модифікацію тексту, обробку даних форми, переміщення елементів на сторінці та виконання анімації

*Бази даних в **SQL Server Management Studio***

База даних - це організований набір структурованої інформації або даних, які зазвичай зберігаються в електронному вигляді в комп'ютерній системі. База даних зазвичай контролюється системою управління базами даних (СУБД). Дані та бази даних, поряд з відповідними програмами,

називаються системами баз даних Дані в найпоширеніших типах баз даних, що функціонують сьогодні, зазвичай моделюються в рядки та стовпці в серії таблиць, щоб зробити обробку та запити даних ефективними. Тоді ви зможете легко отримати до них доступ, керувати ними, змінювати, оновлювати, контролювати та впорядковувати їх. Більшість баз даних використовують структуровану мову запитів (SQL) для запису та запиту даних.

SQL Server Management Studio (SSMS) - це інтегроване середовище для управління будь-якою інфраструктурою **SQL**. Ви можете використовувати SSMS для доступу, налаштування, управління, адміністрування та розробки всіх компонентів **SQL Server**, бази даних **SQL Azure** та **Analytics Azure Synapse**.

Sublime text

Sublime Text Editor - це складний текстовий редактор, який широко використовується розробниками. Він включає великі функції, такі як підсвічування синтаксису, автоматичне відступ, розпізнавання типу файлів, бічна панель, макроси, плагіни та пакети, що спрощують роботу з базою коду. Цільова аудиторія цього підручника - розробники **JavaScript** та **Python**. Веб-розробники, які шукають відповідний текстовий редактор, такий як IDE, також використовуватимуть цей редактор.

Отже, у другому розділі було обрано програмні засоби реалізації цього проекту, була розроблена архітектура програмного забезпечення сайту, описано та створено концептуальну, фізичну та логічну модель бази даних фільмотеки. Тож, вибравши програмні засоби реалізації сайту та маючи схему базу даних можна перейти до третього розділу, де автор розпочне безпосередньо розробку самого сайту та заповнить базу даними та буде застосувати її у додатку.

3 ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Структура файлів проекту

Зазвичай веб-сайт складається з таких основних компонентів: .html, .css, .js документи та інших файлів, наприклад, зображення (.png, .jpg) та шрифти (.ttf, .woff).

У своєму проекті я вирішив додати до основних компонентів додаткові, а саме бібліотеку **jQuery.** Це – легка та доступна бібліотека **JavaScript** гасло якої - "писати менше, робити більше". Мета **jQuery** - набагато спростити використання **JavaScript** на вашому веб-сайті. **jQuery** приймає багато загальних завдань, для виконання яких потрібно багато рядків коду **JavaScript** і обертає їх у методи, які можна викликати одним рядком коду.

Ось наприклад, на моєму веб-сайті я зробив випадаючий список меню навігацію за допомогою цієї бібліотеки. Якщо користувач захоче зайти на цей сайт зі смартфона, то він матиме трішки інакший вигляд, тому що розміри екрана компютера та смартфона досить різні і дуже не зручно коли сайт не налаштований під користувачів мобільних телефонів та планшетів. Ось так виглядає меню навігації на маленькому екрані :

Рисунок 3.1 - Початковий вигляд

При натисканні на слово «Menu», відкриється випадаючий список та користувач зможе обрати те, що йому потрібно.

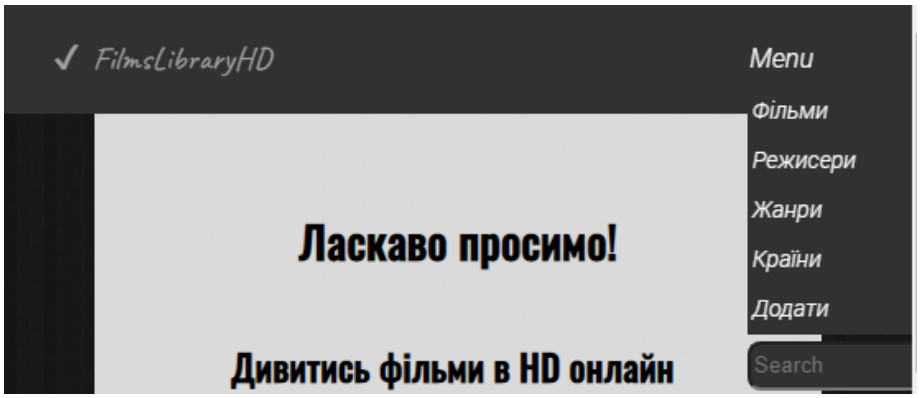


Рисунок 3.2 - Кінцевий вигляд

Ось таку просту у розробці, але дуже зручну функцію я зробив за допомогою **jQuery** і це зайняло лише декілька 9 строчок коду, але якщо б реалізація відбувалась за допомогою лише **JavaScript**, то коду було б на багато більше. Ось приклад коду **jQuery:**

```
$(document).ready(function (){
    $(".menu-toggle").on("click", function(){
        $(".menu").slideToggle(300, function(){
            if($(this).css("display") === "none"){
                $(this).removeAttr("style");
            }
        });
    });
});
```

Рисунок 3.3 - Реалізація за допомогою **jQuery**

3.2 Розробка бази даних фільмотеки

Для того, щоб розпочати розробку бази даних, потрібно краще зрозуміти всю схему і всі взаємозв'язки в ній та розробити три моделі баз даних:

• **Концептуальна модель** є основою для побудови бази даних.

• **Логічна модель** будується після концептуальної, в якій будуються таблиці та взаємозв'язки між таблицями.

• **Фізична модель** будується за логічною на її основі. Типи даних додаються до логічної моделі після цього, коли поля даних типів поля записуються, модель фізичної бази даних готова.[0034]

3.2.1 Концептуальна модель

Концептуальна модель - це візуальна схема, намальована у певних, раніше прийнятих позначеннях, яка детально демонструє взаємозв'язок між об'єктами та їх характеристиками. Створюється вона для подальшого проектування бази даних та її перекладу, наприклад, у реляційну базу даних. На концептуальній моделі у візуально зручній формі реєструються зв'язки між об'єктами даних та їх характеристиками.[6]

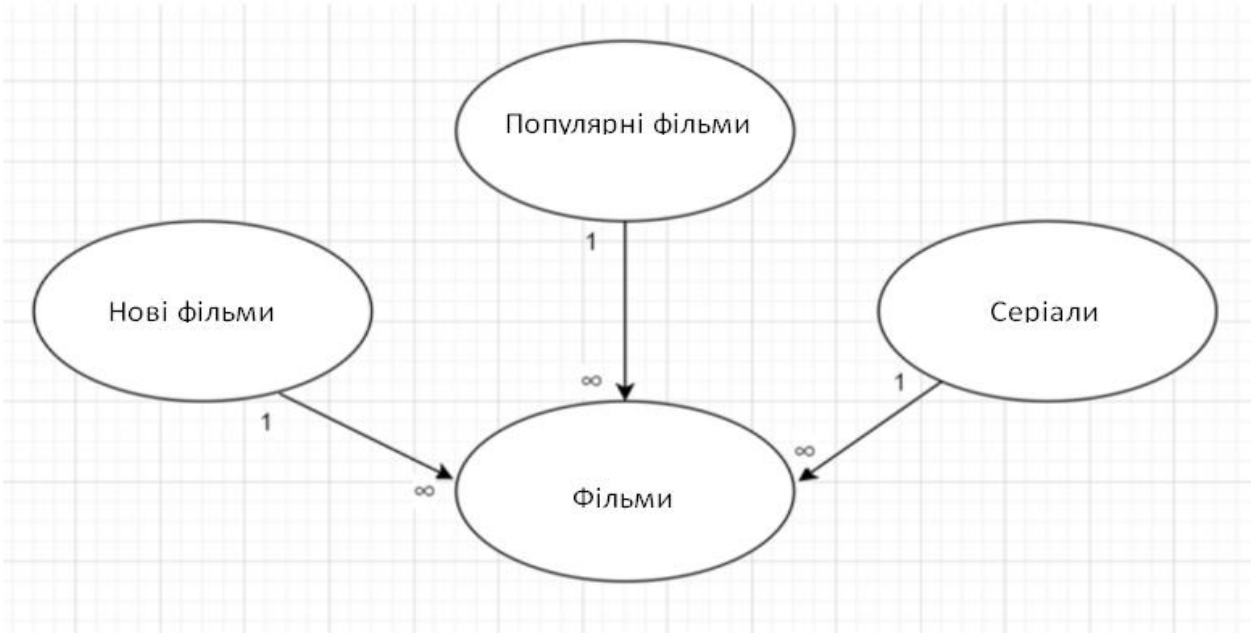


Рисунок 3.4 - Концептуальна модель бази даних

Концептуальна модель становить основу побудови бази даних. Вона містить ключові назви таблиць. Це:

- Види Фільмів(Movies_options);
- Популярні фільми(Popular_films);
- Нові фільми(New_films);
- Серіали(Soap_opera);
- Країна(Country);
- Жанр(Genre);
- Режисер(Director);

Серед цих семи таблиць є найголовніша «Movies_options» всі ж інші таблиці мають зв’язок один до багатьох.

3.2.2 Логічна модель

Якість розробленої бази даних повністю залежить від хорошого розвитку окремих етапів її проектування. Немає сумнівів, що якісна розробка логічної моделі, забезпечує доступність бази даних предметної області та визначає структуру фізичної бази даних та її експлуатаційні характеристики. Логічні моделі даних додають додаткову інформацію до елементів концептуальної моделі. Визначає структуру елементів даних та встановлює взаємозв'язки між ними. Перевага логічної моделі даних створюється у створенні, формуванні основи для фізичної моделі.[8] Однак структура моделювання залишається загальною. Щодо нормалізацію відносин - скористаємося набором даних, забезпечимо їх безперервність та зменшимо витрати на обслуговування бази даних.

Розглянемо спочатку в письмовому вигляді логічну модель бази даних домашньої фільмотеки:

- **Movies_options** – найголовніша таблиця з даними, яка має поле ID - первинний ключ та ідентифікатор та “Oprion_name” яка містить назву виду фільму.
- **Фільми** (Popular_films, New_films, Soap_opera) – ці таблиці містять дані про фільми або серіали та має такі поля: “ID - первинний ключ та ідентифікатор”, “ Title ”, “Genre_id ”, ”Country_id” “Year_of_issue”. А таблиця Popular_films містить додаткове поле “Director ” , так як так як найвідоміші фільми знімають найвідоміші режисери
- **Genres** (жанри) – таблиця містить дані про жанри фільмів та має такі поля: “ID - первинний ключ та ідентифікатор”, “Name”.
- **Directors** (режисери) – таблиця містить дані про режисерів та має такі поля: “ID - первинний ключ та ідентифікатор”, “Name”, “Surname”, “Country”.

- **Countries** (країни) - таблиця містить дані про країни де був знятий фільм та має такі поля: “ID - первинний ключ та ідентифікатор ”, “Name”.

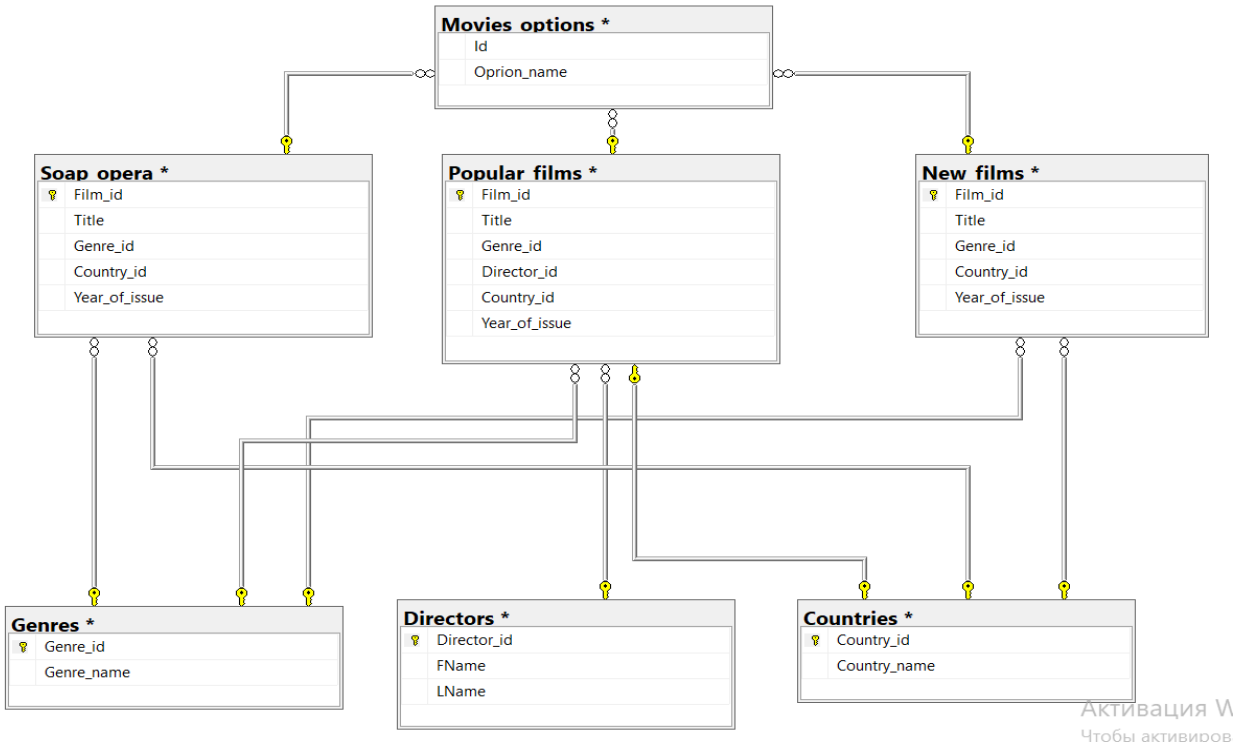


Рисунок 3.5 - Логічна модель бази даних

3.2.3 Фізична модель

Фізична модель бази даних відображає, як модель буде побудована в базі даних. Модель фізичної бази даних показує всі структури таблиць, включаючи ім'я стовпця, тип даних стовпця, обмеження стовпців, первинний ключ, зовнішній ключ та зв'язки між таблицями. Особливості фізичної моделі даних включають:

- Специфікація всіх таблиць і стовпців.
- Денормалізація відбувається за вимогою0 користувача.
- Фізичні міркування можуть спричинити відмінність фізичної моделі даних від логічної моделі даних.

Етапи розробки фізичної моделі даних:

1. Перетворення об’єктів у таблиці.

2. Перетворити відносини на **ForeignKey**.

3. Перетворення атрибутів у стовпеці.

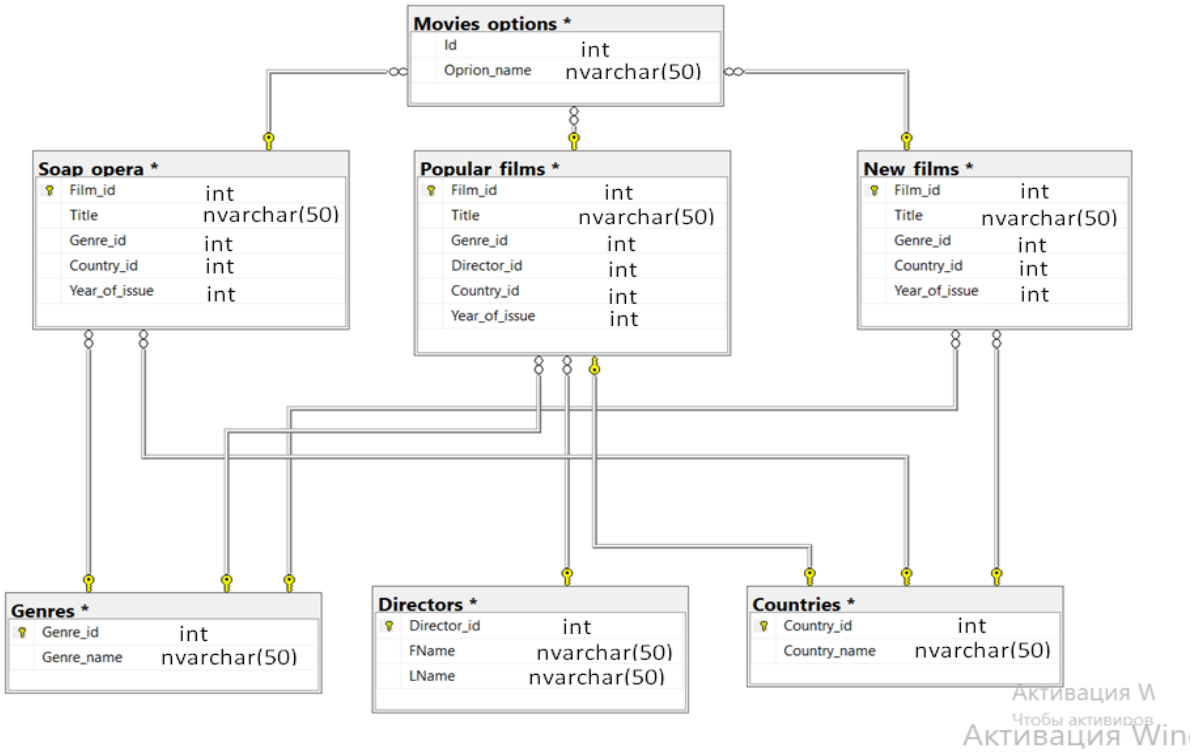


Рисунок 3.6 - Фізична модель бази даних

3.2.4 Створення бази даних

Для кращої оптимізації проекту та швидшого його розвитку нам потрібно створити базу даних. Це ми зробимо завдяки **Microsoft SQL Server** та мови SQL.

Створення бази даних

CREATE DATABASE – два ключові слова для створення нової бази даних, після цих слів пишеться назва бази даних в один рядок. Так ми створили базу даних з назвою FilmsDB.

Створення таблиць

Далі потрібно створити таблиці для наповнення бази даними. Для цього треба застосувати команду CREATE TABLE та вписати назву таблиці, далі в дужках перераховати стовбці для кожної з таблиць та розділити кожен з них комою:

```
CREATE TABLE Films
(
    Film_id INT NOT NULL IDENTITY,
    Title NVARCHAR(20) NOT NULL,
    Genre_id INT NOT NULL,
    Director_id INT NOT NULL,
    Country_id INT NOT NULL,
    Year_of_issue INT NOT NULL
)
GO

CREATE TABLE Directors_
(
    Director_id INT NOT NULL IDENTITY,
    FName NVARCHAR(20) NOT NULL,
    LName NVARCHAR(20) NOT NULL,
)
GO

CREATE TABLE Genre_
(
    Genre_id INT NOT NULL IDENTITY,
    Genre_name NVARCHAR(20) NOT NULL,
)
```

Рисунок 3.7 - Створення таблиць Films, Directors, Genres

CREATE TABLE – за допомогою цих слів у базі даних створюється нова таблиця, в перекладі (створити таблицю).

При створенні таблиці застосовані такі типи даних та ключові слова як:

NVARCHAR(n) - строкові дані фіксованого розміру. n визначає розмір рядка в байтах і повинні мати значення від 1 до 8000.

Int– цілочисловий тип, діапазон чисел якого від -2 147 483 648 до 2 147 483 647.

INT – числовий тип даних SQL , що зберігає цілі числа зі знаком чи без знаку в діапазоні від -2 147 483 648 до 2 147 483 647. Займає 4 байта. Всі цілочисельні типи даних, а також типи, що зберігають десяткові дроби, підтримують властивість IDENTITY.

Слово «**IDENTITY**» означає, що при додаванні записів у таблицю значення атрибута, позначеного цим ключовим словом, вказувати не треба, оскільки значення будуть додаватися автоматично.

«**NOTNULL**» означає, що значення цього атрибута не може бути пропущеним при додаванні записів у таблицю.

Створення первинних ключів

У таблиці може бути тільки один первинний ключ, який може складатися з одного або декількох полів. Коли в якості первинного ключа використовуються кілька полів, він називається складеним. Якщо для таблиці первинний ключ задано в певному полі, то в цьому полі не може міститися двох записів з однаковими значеннями. Нижче наведено код, в якому стовпець ID визначається в якості первинного ключа таблиць Directors, Genres, Countries.

```
------
ALTER TABLE Directors
ADD
PRIMARY KEY(Director_id)
GO

ALTER TABLE Genres
ADD
PRIMARY KEY(Genre_id)
GO

ALTER TABLE Countries
ADD
PRIMARY KEY(Country_id)
GO
```

Рисунок 3.8 - Створення первинних ключів для таблиць

Створення відношень

Для того щоб встановити зв'язки між таблицями треба задати зовнішній ключ (FOREIGN KEY) потрібним нам таблицям. Для цього треба використати команду ALTER TABLE , вписати назву таблиці та за допомогою команди ADD додати зовнішній ключ до вибраної в дужках колонки яка ссилається (REFERENCES) на іншу таблицю.

На наступному рисунку представлено код для встановлення відношення:

```
ALTER TABLE Popular_films
ADD
FOREIGN KEY(Director_id) REFERENCES Directors(Director_id)
GO
---
ALTER TABLE Popular_films
ADD
FOREIGN KEY(Genre_id) REFERENCES Genres(Genre_id)
GO

ALTER TABLE Popular_films
ADD
FOREIGN KEY(Country_id) REFERENCES Countries(Country_id)
GO
```

Рисунок 3.9 - Створення відношень

Після проходження даної операції можна помітити залежності між таблицями бази даних. Дана процедура проходиться задля уникнення неточності в базі даних, тобто якщо були внесені інші дані в одну з дочірніх таблиць, то й дані зміняться і в батьківській.

Внесення даних в таблицю

Всі таблиці створені. Можна наповнити таблиці даними. Для цього треба: використати «INSERT » та назву таблиці, в дужках після якої вписати стовпці, в які треба додавати дані. Після цього треба вписати «VALUES», що означає «значення» та записати кожен запис в окремих дужках, розділених комою. В ті атрибути, що були позначені ключовим словом «IDENTITY», значення заносити не потрібно, тому що це відбувається автоматично.

```
INSERT Popular_films
( Title, Genre_id,  Director_id, Country_id, Year_of_issue )
VALUES
( 'Парк Юрского периода' , 9 , 1 , 3, 1993 ),
( 'Спасти рядового Райана' , 1 , 1, 3,  1998 ),
( 'Инопланетянин' , 4 , 1, 3, 1982 ),
( 'Властелин Колец' , 1, 7, 4, 2003),
( 'Кин Конг' , 1, 7, 4, 2005),
( 'Хоббит' , 1, 6 , 4 , 2012),
( 'Остров проклятых' , 8, 3 , 3 , 2010),
( 'Волк с Уолл-стрит' , 2, 3 ,3  , 2013),
( 'Молчание' , 3, 3 , 3 , 2016),
( 'Тёмный рыцарь' , 1, 4, 4, 2008),
( 'Начало' , 4, 4, 4, 2010),
( 'Удача Логана' , 1, 5,3 ,2016 ),
( 'Нокаут' , 4, 5, 2 ,2012 ),
( 'Чужой' , 3, 6, 7, 1979),
( 'Гладиатор' , 4, 6, 5, 2000 ),
( 'Джанго освобождённый' , 1 , 7, 6, 2012 ),
( 'Убить Билла' , 1 , 7 , 3 , 2003 ),
( 'Хэнкок' , 1 , 8 , 3 , 2008 ),
( 'Кибер' , 1 , 8 , 2 , 2015 ),
( 'Аватар' , 1 , 9, 3 , 2009 ),
( 'Титаник' , 5 , 9 , 3 , 1997 ),
( 'Успешные люди' , 2 , 10, 8 , 2019 ),
( 'Фарго' , 2 , 10 , 8 , 1996 ),
( 'Лабиринт фавна' , 8 , 11 , 7 , 2006),
( 'Хэл-бой' , 1 ,11 , 3 , 2004 ),
( 'Бойцовский клуб' , 4 ,12 , 3 , 1999 ),
( 'Семь' , 4 ,12 , 3 , 1995 ),
( 'Чарли и шоколадная фабрика' , 9 ,13 , 8 , 2005 ),
( 'Дамбо' , 9 , 13 , 8 , 2019 ),
( 'Приколисты' , 2 ,14 , 3 , 2009 ),
( 'Немножко беременна' , 2 ,14 , 3 , 2007 ),
( 'Проклятие' , 3 ,15 , 10 , 2006 ),
( 'Человек-паук' , 6 ,15 , 3 , 2002),
( 'Чудо-женщина' , 9 ,16 , 6 , 2017 ),
( 'Человек из стали' , 9 ,16 , 7 , 2013 ),
( 'Черный лебедь' , 5 ,17 , 8 , 2010 ),
( 'Мама' , 5 , 17 , 5 , 2017 ),
( '28 дней спустя' , 7 ,18 , 7 , 2002 ),
( 'Пекло' , 7 ,18 , 7 , 2007 ),
( 'Наркокурьер' , 8 ,19 , 5 , 2018 ),
( 'Таинственная река' , 8 , 19 , 8 , 2003 ),
( 'Взлом' , 1 ,20 , 1 , 2017),
( 'Взрывная волна' , 5 ,20 , 1 , 2017 ),
( 'Сталлинград' , 4 , 21 , 1 , 2013 ),
( '9 рота' , 1 , 21 , 1 , 2005 ),
( 'Последний богатырь' , 9 ,22 , 1 , 2017 ),
( 'О чем говорят мужчины' , 2 ,22 , 1 , 2010 );
```

Рисунок 3.10 - Заповнення таблиці Popular_films

Тепер маємо повністю створену базу даних фільмів, яка може бути використана для створення додатку основою для якого буде ця база даних.

3.2.5 Експорт бази даних

Для того щоб здійснити підключення даної бази даних до додатку треба спочатку експортувати базу у потрібному форматі. **JavaScript** підтримує лише один формат даних – це **JSON.**


```
{
  "blogs" : {
    "blog" : [
      {
        "needspassword" : true,
        "id" : 73,
        "name" : "Bloxus test",
        "url" : "http:\/\/remote.bloxus.com\/"
      },
      {
        "needspassword" : false,
        "id" : 74,
        "name" : "Manila Test",
        "url" : "http:\/\/flickrtest1.userland.com\/"
      }
    ]
  },
  "stat" : "ok"
}
```


Рисунок 3.11 - Приклад зовнішнього вигляду бази даних у JSON форматі

Для того щоб здійснити таке перетворення ми здійснюємо такі команди у Microsoft Server Management Studio:

- **SELECT** (обрати) – для того щоб обрати стовбець у таблиці та додати до команди його назву.
- **AS** (як) – для того щоб зберегти цей стовбець у відповідну назву яку треба вказати в квадратних дужках ([ ]).
- **FROM** (звідки) – для того щоб програма зрозуміла **з** якої саме таблиці будуть братися дані, та додати саму назву таблиці.

- **FOR JSON** (для json) – найголовніше ключове слово для того щоб здійснити перетворення, тут ми вказуємо до якого саме формату воно буде зроблене.
- **PATH** (шлях) - для того щоб зберегти повний контроль над форматом вихідних даних JSON, також використовується AUTO, щоб відформатувати вихідні дані JSON автоматично на основі структури інструкції SELECT.
- **ROOT** (корінь) – для того зоб вказати назву в дужках іменованого кореневого елемента.

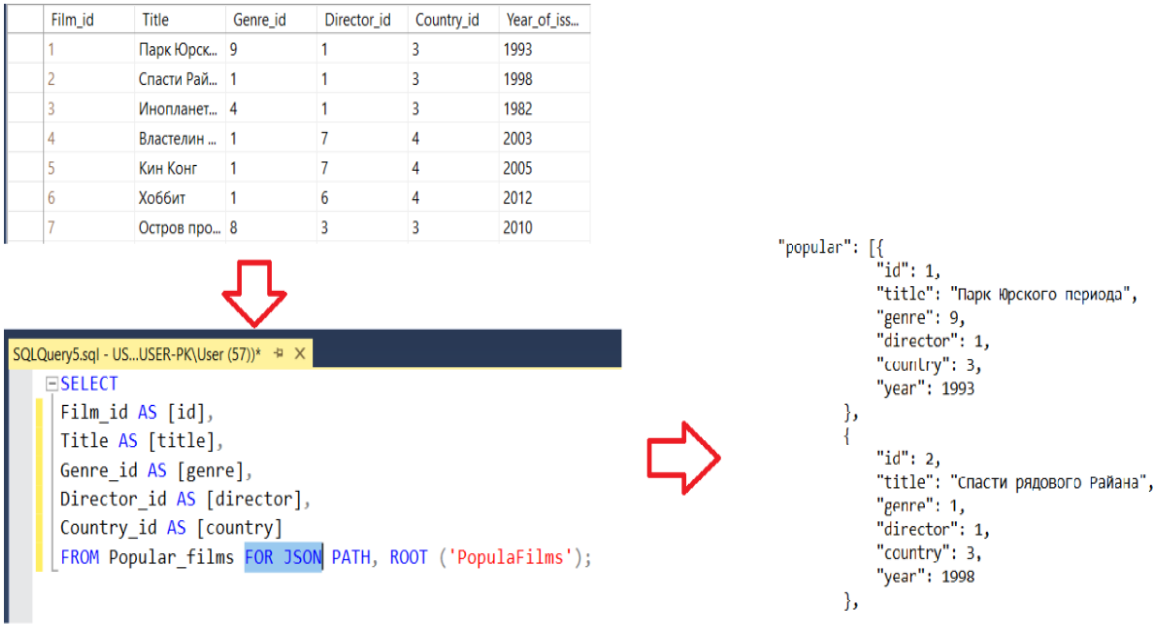


Рисунок 3.12 - Перетворення бази даних з формату SQL у формат JSON

Проробивши дану операцію можна побачити результат:

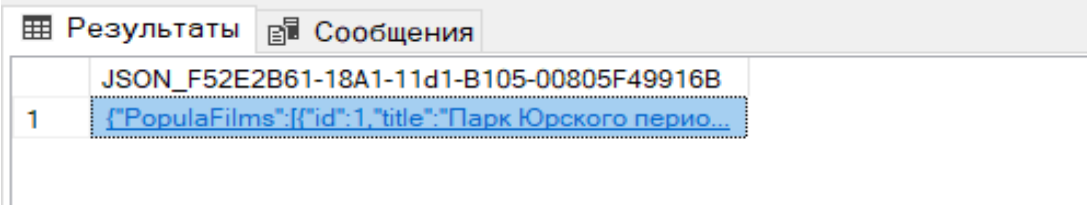


Рисунок 3.13 Посилання на файл JSON

Результат - це посилання на текстовий файл який потрібно зберегти у папці проекту, у якому і буде здійснене підключення даної бази даних.

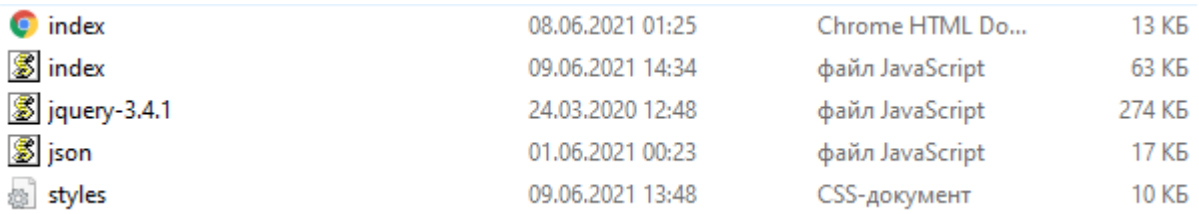


Рисунок 3.14 - Файли проекту

3.3 Розробка інтернет магазину

Спочатку нам потрібно видалити зайве з тегу src, що було створено програмою автоматично, потім ми створюємо глобальні папки для таблиць, зображень, компонентів, баз даних тощо.

3.3.1 Навігація

Навігація у цьому проекті відбувається за допомогою **JS**(мал 3.10). Функція navHandler відповідає за це. При натисканні на відповідну кнопку в навігації(вигляд навігації **HTML** (мал 3.9))буде відкриватися відповідна сторінка. Наприклад, якщо ти хочеш подивитися жанри фільмів які присутні на даному сайті, тобі потрібно натиснути на «Жанри» в навігаційному меню і в тебе з’явиться їх перелік та вже звідти, вибравши відповідний жанр та натиснувши на нього, з’явиться перелік відповідних фільмів, але якщо ти відразу натиснеш на фільми, то в тебе перед очима з’явиться ще декілька варіантів вибору, а саме: «Популярні фільми», «Нові фільми» та «Серіали». Змінна «button» відповідає за відповідну натиснуту кнопку, а «array» - за те, які саме дані потрібно показати.

```
<ul class="header-navigation-list menu">
    <li class="header-navigation-list-item-collapse">
        <a href="#" class="films-options">Фільми</a>
        <div class="collapse-div">
            <ul class="collapse-div-list">
                <li class="popular-films"><a href="#">Популярні фільми</a></li>
                <li class="new-films"><a href="#">Нові фільми</a></li>
                <li class="soap-opera"><a href="#">Серіали</a></li>
            </ul>
        </div>
    </li>

    <li class="header-navigation-list-item">
        <a href="#" class="directors-options">Режисери</a>
    </li>

    <li class="header-navigation-list-item">
        <a href="#" class="genres-options">Жанри</a>
    </li>

    <li class="header-navigation-list-item">
        <a href="#" class="countries-options">Країни</a>
    </li>

    <li class="header-navigation-list-item">
        <a href="#" class="add-new-film">Додати</a>
    </li>
    <div class="search">
        <input type="text" id="input" name="" placeholder="Search">
    </div>
    </li>
</ul>
```

Рисунок 3.15 - вигляд навігації **HTML**

```
let navHandler = (button, array) => {
    button.onclick = () => {
        let listArray = array.map((item, key) => `<li class="${key}">
            <a href="#">
                ${item.name ? (item.surname ? item.name + ' ' + item.surname : item.name) : item.title}
            </a>
        </li>`);
        queryUl.innerHTML = listArray.join('');
        hideAllSections(queryUl);
        queryFilm.innerHTML = '';
        queryFilmDetails.innerHTML = '';
        queryFilmDescript.innerHTML = '';
        queryFilmPhotoes.innerHTML = '';
    }
}
```

Рисунок 3.16 - вигляд навігації **HTML**

У базі даних ми створили 6 масивів **JavaScript,** в яких вводимо віжповідні данні: popular, newFilms, soapOpera, directors, countries, genres. Відповідно до них є сторінки зі списком цих даних: популярні фільми, нові фільми, серіали, режисери фільмів, країна походження фільму та їх жанри. Далі ми повинні відобразити ці данні на відповідних сторінках. Для оптимізації коду я використав цикли.

Після виходу **EcmaScript 7** до стандартних циклів **forEach**, **for** and **while**, додали нові, які можуть змінювати сам масив або повертати новостворений відповідно до заданих умов.[10] Ось приклад даних масивів: **sort**, **filter**, **map**. **Sort** та **filter** перебирає відповідний масив і повертає потрібний елемент на кожній ітерації, а **map(**саме його ми і використовуємо**)** – повертає вже змінений масив даних.

Останній пункт в навігації – це «Додати фільм» при натисканні на кнопку інтерфейс буде виглядати так:

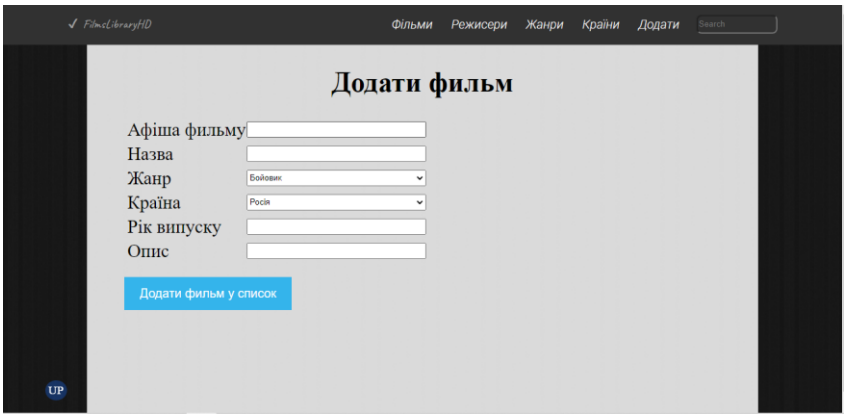


Рисунок 3.17 – додання нового фільму

Коли потрібно додати новий фільм треба відкрити цю вкладку і тоді коли всі поля вводу будуть заповнені, фільм за відповідною інформацією буде доданий у список «Нових фільмів».

3.3.2 Відображення даних про фільм на сторінці

Після того як користувач вибере фільм який йому сподобався, він зможе подивитися інформацію про нього, а саме: афішу фільму, деякі дані про фільм та головні кадри з нього. Далі я наведу приклад вигляду сторінки на основі одного фільму(Рисунок 3.18).

Рисунок 3.18 - Сторінка відображення даних про обраний фільм

Ця сторінка була реалізована також за допомогою **JS**, частина коду наведена нижче(Рисунок 3.19). Спочатку перевіряємо який саме вид фільму вибрав користувач(1), а вже потім вводимо відповідні дані цього фільму за допомогою функції «defineArrayResults»(2)(Рисунок 3.20) та відповідних змінних(3).

```
if (infoArrays[i] === popularFilms || infoArrays[i] === newFilms || infoArrays[i] === soapOpera) {
    queryFilm.innerHTML = `
        <img src="${defineArrayResults[1].image}"/>
    `
    queryFilmDetails.innerHTML = `
        <h1>${defineArrayResults[1].title}</h1>

        <h4>Жанр: ${genresNames[--defineArrayResults[1].genre] }</h4>
        ${defineArrayResults[1].director ? '<h4>Режисер: ' + directorsNames[--defineArrayResults[1].director] + '</h4>' : ''}
        <h4>Країна: ${countriesNames[--defineArrayResults[1].country]}</h4>
        <h4>Рік: ${defineArrayResults[1].year}</h4>
        <p>В головних ролях: <span>${defineArrayResults[1].actors}</span></p>

    `
    queryFilmDescript.innerHTML = `
        <p>Кадри з фільму:</p>
    `
    queryFilmPhotoes.innerHTML = `
        <img src="${defineArrayResults[1].img1}"/>
        <img src="${defineArrayResults[1].img2}"/>
        <img src="${defineArrayResults[1].img3}"/>
    `

    queryFilmDetails.style.display = 'block';
    queryFilm.style.display = 'block';
    queryFilmDescript.style.display = 'block';
```

Рисунок 3.19 - Відображення даних про фільм **JS**

```
defineArrayResults = defineArray(infoArrays[i], e.target.innerHTML.trim());
```

Рисунок 3.20 - Реалізація «defineArrayResults»

3.3.3 Реалізація функції пошуку по сайту

Щоб зробити пошук фільму, режисера, жанру, країни більш легшим та зручним для користувачів сайту, була зроблена функція пошуку. На сторінці вона знаходиться біля навігаційного меню(Рисунок 3.21)

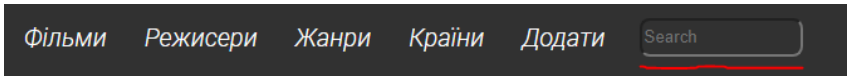


Рисунок 3.21 – Пошук

Дана функція була написана на мові **JS**(Рисунок 3.22)(повний код програми знаходиться в додатку index.js). Спочатку в змінній «input» вказуємо полу куди ми будемо вводити текст пошуку. Атрибут подій **oninput**(1) робить так, що наведений нижче код **JS** буде працювати лише тоді, коли наша змінна отримує ввід будь-який змінних від користувача. У змінну «list» записуємо область пошуку даних, а вже потім у циклі **forEach** перевіряємо введені дані. За допомогою властивості «innerText» вибирає текст з наведених даних та копіює текст, який відображається цим елементом в браузері. Ця властивість враховує всі стилі, застосовані до елементу. Метод «toLowerCase()» повертає значення рядка, в нижній регістр, за допомогою його, ми можемо вводити букви не зважаючи на регістр. Метод «search()» шукає в рядку вказане значення та повертає відповідну позицію[12]. Значення пошуку може бути рядком або регулярним виразом Отже, якщо введений текст не співпаде з якимось даними с пошукового списку, ці дані зникнуть, за допомогою додавання класу «hide»(Рисунок 3.23) до цих елементів.

```
let input = document.querySelector('#input');
input.oninput = function() {
    let value = this.value.trim();
    let list = document.querySelectorAll('.query-info-list li');

    if(value != ''){
        list.forEach(elem => {
            if(elem.innerText.toLowerCase().search(value) == -1){
                elem.classList.add('hide');
            }
        });
    } else {
        list.forEach(elem => {
            elem.classList.remove('hide');
        });
    }
};
```

Рисунок 3.22 - Реалізація пошуку **JS**

```
.hide{
    display: none;
}
```

Рисунок 3.23 - Клас «hide» **CSS**

Далі наведено декілька скріншотів прикладу роботи програми:

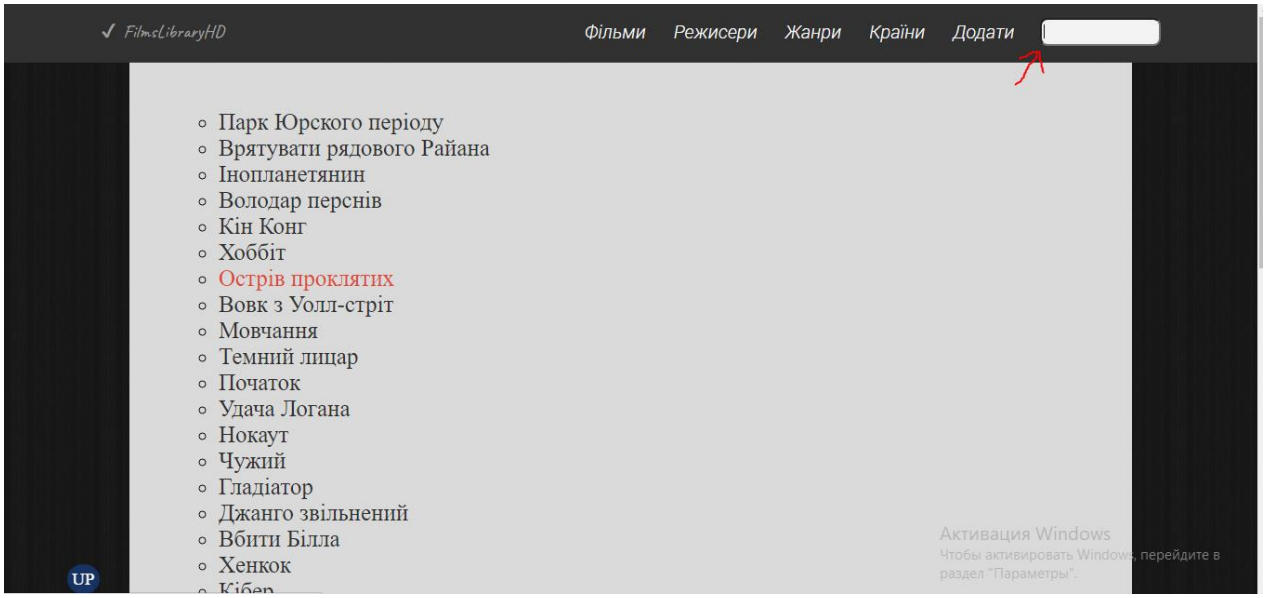


Рисунок 3.25 - Загальний вигляд сторінки без введених даних

Але якщо ми введемо певну комбінацію букв і знайдеться співпадіння у відповідному переліку(Рисунок 3.26), то на екрані залишаться лише елементи які співпадають.

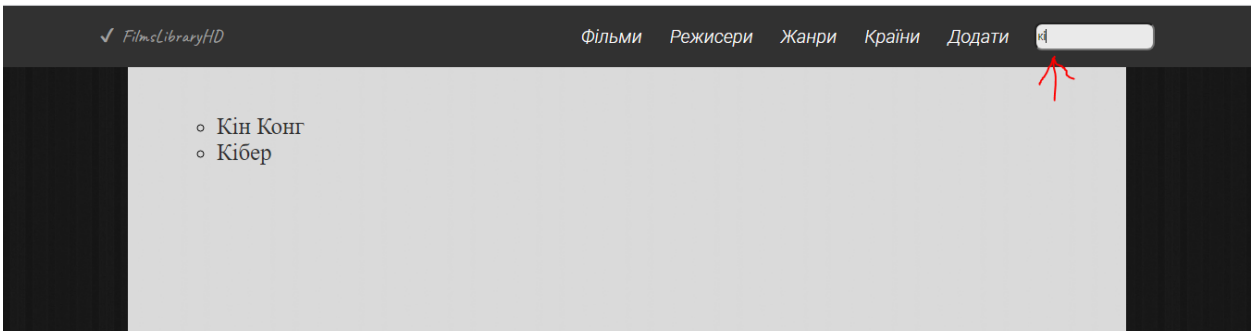


Рисунок 3.26 - Вигляд сторінки з введеними даними в поле пошуку

3.3.4 Адаптивність сайту

Адаптивність будь-якого сайту, один з головних критеріїв його оцінювання. Якщо сайт динамічний, добре працює на всіх розмірах екрану то

сайт можна вважати гарним. Я думаю кожен з нас намагався відвідати певний сайт зі смартфону та стикався з незрозумілим видом сайту. Блоки та текст бути занадто великого розміру, виглядають за рамки сторінки, блоки можуть навіть заходять один на одного, погодьтесь, це виглядає дуже моторошно, а у використанні навіть гірше. Тому автор зробив цей сайт адаптивним. Вона досягається за допомогою технології **flexbox**(Рисунок 3.27). Це одновимірний метод макетування для розміщення елементів у рядки або стовпці. Предмети гнуться, щоб заповнити додатковий простір, і стискаються, щоб поміститися в менші простори. Технологія **flexbox** дуже проста у використанні та не потребує написання багатьох строчок коду. Потрібно лише вказати у відповідному блоці властивість «display» та надати їй значення «flex», після цього вказати деякі додаткові параметри і готово.

```
.info_news{
    display: flex;
    justify-content: space-between;
    flex-wrap: wrap;
}
```

Рисунок 3.27 Приклад технології **flexbox**

Але на мою думку, повної адаптивності сайту лише за допомогою технології **flexbox** досягти неможливо. Для цього у **CSS** є функція медіа запитів(Рисунок 3.28). Її реалізація займає трішки більше часу, але інколи вона потрібна. Використання медіа запитів – дуже популярна техніка доставки таблиці стилів до смартфонів, не великих ноутбуків та планшетів. У цій функції є як переваги так і недоліки. Вона чудово підходять для адаптації макетів до різних розмірів екрану, але зовсім не підходить для створення модульних конструкцій. Технології розвиваються, тож модульний **CSS** зараз досить складний і медіа-запити надають дуже мало інформація або взагалі не допомагають. На мою думку, медіа запити потрібно використовувати частково, лише за для надання певних стилів елементам на відповідному екрані.

```css
@media screen and (max-width: 580px){
    .menu{
        display: none;
        position: absolute;
        right: 0;
        top: 50px;
        background-color: #313131;
        width: 100px;
        height: 150px;
        line-height: 2.2;
    }
    .menu-toggle{
        margin-right: 30px;
        display: block;
        color: white;
        cursor: pointer;
    }
    .logo{
        margin: auto 0;
    }
    .collapse-div-list{
        max-width: 110px;
        max-height: 130px;
        background-color: #313131;
        right: -10px;
        top: -20px;
        border: none;
        position: absolute;
        line-height: 1.6;
    }
    .collapse-div-list li{
        font-size: 12px;
        border: none;
    }
}
```

Рисунок 3.28 - Приклад медіа запитів у цьому проекті

Підсумовуючи, у даному розділі були повністю розглянуті засоби реалізації розробленого інтерфейсу, логіки додатку, показані всі технології які використовувалися під час розроблення даного сайту, завдяки правильно продуманій схемі бази даних, яку розробили у розділі 2, автор розробив сайт фільмотеки з достатньо широким об’ємом фільмів, можливістю переходу на обраний фільм з інформацією про нього. Ознайомившись з інтерфейсом та основними командами для розробки бази даних **SQL Server Management Studio** було наглядно показано всі етапи створення даної бази даних. При створенні створені бази даних були розглянуті такі команди як CREATE DATABASE, CREATE TABLE, INSERT (COLUMNS) VALUES, та типи даних, що застосовувались при створенні таблиць. Також навели приклад роботи сайту, як відбувається перехід між сторінками, описали функцію пошуку по сайту, яка дозволяє шукати не лише назву фільму, але режисерів, країни цих фільмів та їх жанр та зробили його адаптивним.

ВИСНОВКИ

Отже, у даній роботі було розроблено «Сайт домашньої фільмотеки». Розглянуто усі моделі бази даних: концептуальну, фізичну та логічну, та побудовано відповідні діаграми. В ході розробки сайту була розглянута архітектура даного програмного забезпечення. Створено повністю нормалізовану базу даних фільмів та проведено її перетворення у потрібний формат. З використанням мови розмітки **HTML**, стилізації **CSS** було розроблено адаптивний під будь-який пристрій сайт з приємним інтерфейсом та за допомогою мови програмування **JavaScript** було підключено базу даних фільмів та динамічне відображення кожного з них.

Для розробки цього сайту були використані такі технології:

- PhotoShop – графічний редактор для створення UX / UI веб-дизайну;
- Draw.io – для розробки діаграм, завдяки яким було створено візуальну структуру проекту;
- jQuery – бібліотека JavaScript, яка була розроблена для оптимізації коду програми;
- Технологія flexbox та медіа запити – для адаптивності сайту;
- Microsoft SQL Server – для розробки та заповнення бази даних.
- JSON – для того, щоб здійснити підключення даної бази даних до додатку

Отже, на виході ми маємо додаток «Домашня фільмотека» який порадує користувачів своєю доступністю, якістю та інтерфейсом.

СПИСОК ЛІТЕРАТУРИ

1. Фільмотека – вікіпедія https://uk.wikipedia.org/wiki/%D0%A4%D1%96%D0%BB%D1%8C%D0%BC%D0%BE%D1%82%D0%B5%D0%BA%D0%B0
2. Мартин Фаулер, Кому потрібна архитектура? 2016р. - стр.1 - https://martinfowler.com/ieeeSoftware/whoNeedsArchitect.pdf
3. Шьоніг Х'ю – PostgreSQL 1-е видання 2018р. – 14стр.
4. Джеймс Р. ГроффПол Н. ВайнбергЕндрю Дж. Опель, SQL: повне керівництво. 3-е видання 2016р. – 145стр.
5. Кляйн К., Хант Б SQL, довідник. 2-е видання, 2016р. – 121стр.
6. Фаро С., Лерми П, рефакторинг SQL-додатків, 2-е видання 2017р. – 25стр.
7. Регіна О. Обе, PostgreSQL: Вгору та запуск: Практичний посібник із розширеною базою даних з відкритим вихідним кодом, 3-е видання, 2016р. – 264стр.
8. Алан Больe, Основні принципи SQL, 3-е видання, 2017р.
9. Томас Нілд, початок роботи з SQL: практичний підхід для початківців 1-е видання 2019р. – 251стр.
10. Стівен Фюерштейн, Oracle PL / SQL Language Pocket Reference: Посібник з PL / SQL Language Fundamentals 5-го видання 2020р.
11. Робін Ніксон, навчання PHP, MySQL, JavaScript, CSS та HTML5. Покрокове керівництво зі створення динамічних веб-сайтів, 3-е видання, 2018р. – 59стр.
12. Девід Фленаган, JavaScript: кишеньковий довідник 1-е видання, 2019р.
13. Е. Браун, вивчаємо JavaScript: керівництво по створенню сучасних веб-сайтів, 1-е видання, 2020р. – 219стр.
14. Дуглас Крокфорд, JavaScript: хороші частини, 1-е видання, 2016р – 158стр.
15. Алан Больe, Основні принципи SQL, 3-е видання, 2017р.

ДОДАТОК

Index.html

```
<body>
    <header>
        <div class="header">
            <nav class="header-navigation">
                <div class="logo">
                   <a           href="index.html"           class="name"
id="name">FilmsLibraryHD</a>
                </div>
                <div class="menu-toggle">
                    <p>Menu</p>
                </div>

                <ul class="header-navigation-list menu">
                    <li class="header-navigation-list-item-collapse">
                        <a href="#" class="films-options">Фільми</a>
                        <div class="collapse-div">
                            <ul class="collapse-div-list">
                                <li              class="popular-films"><a
href="#">Популярні фільми</a></li>
                                <li  class="new-films"><a  href="#">Нові
фільми</a></li>
                                <li                class="soap-opera"><a
href="#">Серіали</a></li>
                            </ul>
                        </div>
                    </li>

                    <li class="header-navigation-list-item">
                        <a           href="#"           class="directors-
options">Режисери</a>
                    </li>

                    <li class="header-navigation-list-item">
                        <a href="#" class="genres-options">Жанри</a>
                    </li>

                    <li class="header-navigation-list-item">
                        <a href="#" class="countries-options">Країни</a>
                    </li>
```

```
                    <li class="header-navigation-list-item">
                        <a href="#" class="add-new-film">Додати</a>
                    </li>
                    <div class="search">
                        <input     type="text"     id="input"     name=""
placeholder="Search">
                    </div>
                    </li>
                </ul>
            </nav>
        </div>
    </header>
    <section class="main">
        <div id="up">

        </div>
        <div class="up-arrow">
            <a href="#up">UP</a>
        </div>
        <div class="query-info">
            <ul class="query-info-list">

            </ul>
            <div class="query-info-greeting" id="query-info-greeting">
                <h1> Ласкаво просимо! </h1>
                <h2>Дивитись фільми в HD онлайн</h2>
                <p>Чим себе зайняти після важких трудових буднів?
Повсякденне життя пропонує масу варіантів, але практично кожен людина на
нашій планеті любить переглядати улюблені кінокартини. Ми створили
зручний і унікальний у своєму роді кінотеатр для перегляду відео в
комфортних для тебе умовах. Тобі більше ніколи не доведеться шукати
якусь вільну хвилинку, щоб знайти підходящі кінотеатри, встигнути купити
в касі або забронювати через інтернет квитки на улюблені місця. Усе це
залишилося позаду великих перспектив дивитися фільми онлайн в хорошому
HD якості на нашому сайті. дорогий гість ресурсу, пропонуємо тобі прямо
зараз зануритися в дивно захоплюючий світ - новинки кінопрокату доступні
всімкористувачам цілодобово!</p>

                <h2>Серіали онлайн</h2>
                <p>Що ж стосується запропонованого списку фільмів і
серіалів, які ти можеш тут дивитися в HD якості, то він постійно
```

розширюється і доповнюється картинами найпопулярніших хітів Голлівуду. Словом, кожен шанувальник високоякісного світового кінематографа обов'язково знайде на нашому сайті те, що йому принесе море задоволення від перегляду онлайн в домашніх умовах! Клич друзів, і ти удово проведеш час разом з близькими і рідними людьми - наш ресурс стане прекрасним акомпанементом для</p>

```
                <h2>Незабаром у кіно</h2>
                <div class="wrap">
                    <div class="info_news">

                        <div class="news">
                            <img
src="https://www.film.ru/sites/default/files/movies/posters/42057921-
1503209.jpg">
                            <div class="text">
                                <p   class="films-name">Вибір   Фредеріка
Фітцелла</p>
                                <p class="date">3 червня</p>
                            </div>
                        </div>

                        <div class="news">
                            <img                        src="https://kino-
teatr.ua/public/main/films/2021-05/x4_poster_60a4dd3ba4e8c.jpg">
                            <div class="text">
                                <p class="films-name">Тихе місце 2</p>
                                <p class="date">3 червня</p>
                            </div>
                        </div>

                        <div class="news">
                            <img
src="https://kinoafisha.ua/upload/2019/03/films/8774/22d81t4ptelohranite
l-jen-killera.jpeg">
                            <div class="text">
                                <p  class="films-name">Охоронець  дружини
кілера</p>
                                <p class="date">16 червня</p>
                            </div>
                        </div>
```

```
                        <div class="news">
                            <img    src="https://planetakino.ua/res/get-
poster/00000000000000000000000000003010/vend.jpg">
                            <div class="text">
                                <p class="films-name">Гості на віллі</p>
                                <p class="date">24 червня</p>
                            </div>
                        </div>
                        <div class="news">
                            <img
src="https://st.kinobase.org/storage/360x534/posters/2021/01/2eaea80ac59
8f65a2526.jpg">
                            <div class="text">
                                <p   class="films-name">Сама   божевільна
весілля</p>
                                <p class="date">1 липня</p>
                            </div>
                        </div>
                        <div class="news">
                            <img
src="https://avatars.mds.yandex.net/get-kinopoisk-
image/1946459/bd17e982-3798-481f-8e69-955e516e9988/600x900">
                            <div class="text">
                                <p class="films-name">Чорна вдова</p>
                                <p class="date">9 липня</p>
                            </div>
                        </div>
                    </div>
                </div>

            </div>

            <div class="query-info-film">
                <div class="query-info-film-common">
                    <div class="query-info-film-title">

                    </div>
                    <div class="query-info-film-datails">

                    </div>
                </div>
                <div class="query-info-film-img">
```

```
                    <div class="query-info-film-descript">

                    </div>
                    <div class="query-info-film-photoes">

                    </div>
                </div>
            </div>
            <div class="main-form">
                <h1>Додати фильм</h1>
                <form>
                    <div class="main-form-inputs">
                        <div class="main-form-inputs-item">
                            <label            for="new-film-poster">Афіша
фильму</label>
                            <input type="text" id="new-film-poster">
                        </div>

                        <div class="main-form-inputs-item">
                            <label for="new-film-title">Назва</label>
                            <input type="text" id="new-film-title">
                        </div>
                        <div class="main-form-inputs-item">
                            <label for="new-film-genre">Жанр</label>
                            <select id="new-film-genre">
                                <option value="1">Бойовик</option>
                                <option value="2">Комедія</option>
                                <option value="3">Фільми жахів</option>
                                <option value="4">Драма</option>
                                <option value="5">Мелодрама</option>
                                <option value="6">Фантастика</option>
                                <option value="7">Фентезі</option>
                                <option value="8">Трилер</option>
                                <option value="9">Пригоди</option>
                            </select>
                        </div>

                        <div class="main-form-inputs-item">
                            <label for="new-film-country">Країна</label>
                            <select id="new-film-country">
                                <option value="1">Росія</option>
                                <option value="2">Німеччина</option>
```

```
                                <option value="3">США</option>
                                <option value="4">Англія</option>
                                <option value="5">Франція</option>
                                <option value="6">Голландія</option>
                                <option value="7">Австралія</option>
                                <option value="8">Австрія</option>
                                <option value="9">Китай</option>
                            </select>
                        </div>

                        <div class="main-form-inputs-item">
                            <label                for="new-film-year">Рік
випуску</label>
                            <input type="number" id="new-film-year">
                        </div>

                        <div class="main-form-inputs-item opo">
                            <label for="new-film-about">Опис</label>
                            <input type="text" id="new-film-about">
                        </div>

                        <button class="add-new-film-button">Додати фильм
у список</button>
                    </div>
                </form>
            </div>
        </div>
    </section>
</body>
</html>
```

Index.js

```
$(document).ready(function (){
    $(".menu-toggle").on("click", function(){
        $(".menu").slideToggle(300, function(){
            if($(this).css("display") === "none"){
                $(this).removeAttr("style");
            }
        });
    });
});
```

```
window.onload = () => {
    let filmsButton = document.querySelector('.header-navigation-list-item-collapse');
    let popularFilmsButton = document.querySelector('.popular-films');
    let newFilmsButton = document.querySelector('.new-films');
    let soapOperaButton = document.querySelector('.soap-opera');
    let genresButton = document.querySelector('.genres-options');
    let counriesButton = document.querySelector('.countries-options');
    let directorsButton = document.querySelector('.directors-options');
    let addNewFilmButton = document.querySelector('.add-new-film');
    let addNewFilmToListButton = document.querySelector('.add-new-film-button')
    let mainGreeting = document.querySelector('.query-info-greeting');
    let queryFilm = document.querySelector('.query-info-film-title');
    let queryFilmDetails = document.querySelector('.query-info-film-datails');
    let queryFilmPhotoes = document.querySelector('.query-info-film-photoes');
    let queryFilmDescript = document.querySelector('.query-info-film-descript');
    let collapseDiv = document.querySelector('.collapse-div');
    let queryUl = document.querySelector('.query-info-list');
    let mainForm = document.querySelector('.main-form');
    let genresNames = genres.map((item) => item.name);
    let directorsNames = directors.map((item) => item.name);
    let countriesNames = countries.map((item) => item.name);

    let popularFilms = films.popular;
    let newFilms = films.newFilms;
    let soapOpera = films.soapOpera;

    let sectionArray = [ mainGreeting, queryFilm, queryFilmDetails, queryUl, mainForm ]
    let hideAllSections = (obj) => {
        for(let i = 0; i < sectionArray.length; i++ ){
            sectionArray[i].style.display = 'none';
        }
        obj.style.display = 'block';
    }

    filmsButton.onmouseenter = () => {
        collapseDiv.style.display = 'block';
```

```
    }
    filmsButton.onmouseleave = () => {
        collapseDiv.style.display = 'none';
    }
    addNewFilmButton.onclick = () => {
        hideAllSections(mainForm);
    }
    addNewFilmToListButton.onclick = () => {
        let newFilmPoster = document.querySelector('#new-film-poster');
        let newFilmTitle = document.querySelector('#new-film-title');
        let newFilmAbout = document.querySelector('#new-film-about');
        let newFilmGenre = document.querySelector('#new-film-genre');
        let    newFilmCountry    =    document.querySelector('#new-film-
country');
        let newFilmYear = document.querySelector('#new-film-year');
        let inputArray = [newFilmPoster, newFilmTitle, newFilmYear]
        console.log(newFilmGenre.value)
        if     (newFilmPoster.value     &&     newFilmTitle.value     &&
newFilmAbout.value   &&newFilmGenre.value   &&   newFilmCountry.value   &&
newFilmYear.value){
            let newFilm = {
                "image": newFilmPoster.value,
                "title": newFilmTitle.value,
                "about": newFilmAbout.value,
                "genre": newFilmGenre.value,
                "country": newFilmCountry.value,
                "year": newFilmYear.value
            }
            films.newFilms.push(newFilm);
            for (let i = 0; i < inputArray.length; i++) {
                inputArray[i].value = ''
            }
            alert('Дані успішно внесені!')
        } else{
            alert('Заповніть всі указані поля!')
        }
    }

    let navHandler = (button, array) => {
        button.onclick = () => {
            let    listArray    =    array.map((item,    key)    =>    `<li
class="${key}">
```

```
                <a href="#">
                    ${item.name ? (item.surname ? item.name + ' ' +
item.surname : item.name) : item.title}
                </a>
            </li>`);
            queryUl.innerHTML = listArray.join('');
            hideAllSections(queryUl);
            queryFilm.innerHTML = '';
            queryFilmDetails.innerHTML = '';
            queryFilmDescript.innerHTML = '';
            queryFilmPhotoes.innerHTML = '';
        }
    }

    navHandler(genresButton, genres);
    navHandler(popularFilmsButton, popularFilms);
    navHandler(newFilmsButton, newFilms);
    navHandler(soapOperaButton, soapOpera);
    navHandler(counriesButton, countries);
    navHandler(directorsButton, directors);

    let defineArray = (array, word) => {
        let result;
        let machingObj;
        for (let i = 0; i <= array.length; i++) {

            for (let bla in array[i]) {

                if (array[i][bla] === word) {

                    machingObj = array[i];
                    result = true;
                    break;
                } else {
                    result = false;
                }
            }
            if (result === true) {
                break;
            }
        }
        return [result, machingObj]
```

```
    }

    queryUl.onclick = (e) => {
        if (e.target.tagName === 'A') {

            queryUl.style.display = 'none';

            let infoArrays = [popularFilms, newFilms, soapOpera,
directors, countries, genres];
            let defineArrayResults;

            for (let i = 0; i < infoArrays.length; i++) {

                defineArrayResults = defineArray(infoArrays[i],
e.target.innerHTML.trim());

                if (defineArrayResults[0]) {

                    if (infoArrays[i] === popularFilms || infoArrays[i]
=== newFilms || infoArrays[i] === soapOpera) {
                        queryFilm.innerHTML = `
                            <img src="${defineArrayResults[1].image}"/>
                        `
                        queryFilmDetails.innerHTML = `
                            <h1>${defineArrayResults[1].title}</h1>

                            <h4>Жанр: ${genresNames[--
defineArrayResults[1].genre] }</h4>
                            ${defineArrayResults[1].director ?
'<h4>Режисер: ' + directorsNames[--defineArrayResults[1].director] +
'</h4>' : ''}
                            <h4>Країна: ${countriesNames[--
defineArrayResults[1].country]}</h4>
                            <h4>Рік: ${defineArrayResults[1].year}</h4>
                            <p>В головних ролях:
<span>${defineArrayResults[1].actors}</span></p>

                        `
                        queryFilmDescript.innerHTML = `
                            <p>Кадри з фільму:</p>
                        `
                        queryFilmPhotoes.innerHTML = `
```

```
                    <img src="${defineArrayResults[1].img1}"/>
                    <img src="${defineArrayResults[1].img2}"/>
                    <img src="${defineArrayResults[1].img3}"/>
                `

                queryFilmDetails.style.display = 'block';
                queryFilm.style.display = 'block';
                queryFilmDescript.style.display = 'block';

            } else {
                let clearedArrayInfo = [];
                let unclearedArrayInfo = [];

                let creatingUnclearedArrayInfo = (array, data)
=> {
                    unclearedArrayInfo.push(array.map((item) =>
item[data] === defineArrayResults[1].id ? item.title : null));
                }

                if (infoArrays[i] === directors) {
                    creatingUnclearedArrayInfo(popularFilms,
"director");
                } else if (infoArrays[i] === genres) {
                    creatingUnclearedArrayInfo(popularFilms,
"genre");
                    creatingUnclearedArrayInfo(newFilms,
"genre");
                    creatingUnclearedArrayInfo(soapOpera,
"genre");
                } else {
                    creatingUnclearedArrayInfo(popularFilms,
"country");
                    creatingUnclearedArrayInfo(newFilms,
"country");
                    creatingUnclearedArrayInfo(soapOpera,
"country");
                }
                if (unclearedArrayInfo.length === 3) {
                    unclearedArrayInfo.filter((item) => {
                        let bla = item.filter((i) => i !== null)
                        bla.map((item) => {
                            clearedArrayInfo.push(item)
```

```
                                })
                            })
                        } else {
                            unclearedArrayInfo.filter((item)          =>
clearedArrayInfo = item.filter( i => i !== null));
                        }
                        let queryArray = clearedArrayInfo.map((item) =>
`<li><a href="#">${item}</a></li>`);
                        console.log(queryArray)
                        queryUl.innerHTML = queryArray.join('');
                        queryUl.style.display = 'block';
                        navHandler(genresButton, genres);
                        navHandler(counriesButton, countries);
                        navHandler(directorsButton, directors);
                    }
                }
            }
        }

        let input = document.querySelector('#input');
        input.oninput = function() {
            let value = this.value.trim();
            let list = document.querySelectorAll('.query-info-list li');

            if(value != ''){
                list.forEach(elem => {
                    if(elem.innerText.toLowerCase().search(value) == -1){
                        elem.classList.add('hide');
                    }
                });
            } else {
                list.forEach(elem => {
                    elem.classList.remove('hide');
                });
            }
        };
    }
```