МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет імені Тараса Шевченка
Інститут філології
Кафедра української мови та прикладної лінгвістики

Інтегрована електронна лексикографічна система кінематографічних термінів

Кваліфікаційна робота магістра
за спеціальністю 035 «Філологія»,
спеціалізацією 035.10 «Прикладна лінгвістика»,
галузі знань 03 «гуманітарні науки»
ОП "Прикладна лінгвістика
(редакторсько-перекладацька
та експертна діяльність)"
Людмили П'ЯТАЧЕНКО

науковий керівник:
к.філол.н.,доц. Оксана ЗУБАНЬ

рецензент: д.філол.н. Наталія ДАРЧУК

Київ-2021

ЗМІСТ

ВСТУП

На сучасному етапі розвитку інформаційних технологій застосування комп'ютерів стає ледь не основним об'єктом для роботи в усіх сферах діяльності людини. Зростання потреб суспільства в пошуку й опрацюванні інформації зумовило активне впровадження комп'ютерних технологій і у лінгвістичних дослідженнях.

У сучасній лінгвістиці актуальним завданням є створення і використання словників та платформ для вирішення найрізноманітніших завдань: від дослідження лексики певної мови до проведення лінгвістичних досліджень.

Сьогодні лексикографія та термінографія вступили в нову фазу розвитку й активно використовують інноваційні технології в конструюванні електронних лексикографічних систем.

Одним з основних об'єктів сучасних лексикографічних систем є тлумачні словники (здебільшого одномовні), в яких користувачеві надається розгорнуте значення слів засобами обраної мови. Найвідомішими прикладами електронних тлумачних словників, що представлені українською мовою, можна назвати лише «Академічний тлумачний словник сучасної української мови» [32] та словник «Український мовно-інформаційний фонд» [40].

Водночас, активний розвиток переживають, крім тлумачних, також спеціалізовані, зокрема термінологічні словники. У багатьох науково-технічних та культурних сферах відбувається настільки динамічне розширення терміносистеми, що традиційні паперові словники не в змозі його належно відтворювати. Так, наприклад, стрімке поширення кінематографічного мистецтва, його інтеграційний та глобальний характер сприяли появі кінематографічної термінології, яка знайшла своє відображення в національних мовах країн, що користуються досягненнями кінематографії.

Українська кінематографічна термінографія розпочала свій розвиток лише у 1987 році коли вийшов двотомний «Кінословник» [1], укладений С. Д.

Безклубенко, О. Г. Рутковським та мав велику кількість неточностей. А далі, аж у 2007 році, було укладено «Кінословник: терміни, визначення, жарґонізми» В. Н. Мирославського з реєстром у 587 слів, що висвітлив розвиток кіновиробництва, кінопрокату й кінотехніки і став єдиним повноцінним словником кінотермінів.

В епоху інформаційних технологій, електронні лексикографічні системи стали багатофункціональними (інтегрованими) завдяки об'єднанню різних модулів, у тому числі мультимедійних. Підключення мультимедіа, зокрема аудіо модуля серед українських лексикографів поки що не практикується, проте є яскраві приклади створення таких систем американськими та британськими лексикографами. До таких аудіо словників належать «Oxford dictionary with audio» [35] та «Macmillan's online dictionary» [34].

Ступінь розвитку сучасної термінографії у галузі кінематографа та потреба суспільства у багатомодульних лінгвістичних платформах зумовлюють **актуальність та новизну дослідження**, тому що електронні портали такого типу користуються надзвичайно великим попитом у сучасному суспільстві. За допомогою електронних систем користувач має змогу швидко знайти необхідне слово, його переклад, тлумачення, правильне написання, звучання, додаткову інформацію та ілюстративний матеріал. Проте такої сучасної електронної багатомодульної лінгвістичної системи, а особливо з української кінематографічної термінології до сьогодні ще не існує.

Об'єкт дослідження – кінотерміни англійської та української мов.

Предметом дослідження є мультимедійна лексикографічна модель кінотерміносистем англійської та української мов, в аспекті тлумачного, перекладного та аудіо модулів.

Мета роботи - створити інтегровану електронну мультимедійну лексикографічну систему з кінотермінології.

Поставлена мета передбачає розв'язання таких завдань:

1. Проєктування двомовної англо-української бази даних кінематографічних термінів.
2. Створення системи перевірки орфографії кінотермінів.
3. Створення бази даних українських та англійських іменованих сутностей.
4. Створення системи автоматичного поповнення корпусу текстів та укладання конкордансу.
5. Створення аудіо відтворення елементів словникової статті на базі платформи Google Cloud Text-to-speech API.
6. Створення веб-застосунку лексикографічної системи та перевірка ефективності роботи платформи.

Матеріал дослідження складають 100 кінематографічних термінів українською мовою та їхні перекладні еквіваленти англійською мовою; тексти кінематографічної тематики обсягом 23100 текстів.

Методологія дослідження: розроблення програми здійснювалась мовою програмування Python у середовищі розроблення PyCharm Community Edition. Також, крім стандартних бібліотек, були підключені бібліотеки Django [49] (фреймворк для конструювання веб-додатка), nltk [51] (для автоматизованого реферування тексту), lxml [54] та requests [55] (для скачування текстів кінематографічної тематики з Інтернету), stanza [48] (для розпізнавання іменованих сутностей), python-Levenshtein [53], google-cloud-text-to-speech [50].

Теоретичне та практичне значення одержаних результатів визначається можливістю їхнього використання під час викладання лекційних та практичних курсів із таких навчальних дисциплін як «Лексикологія», «Термінознавство», «Лексикографія»; у практиці викладання англійської та української мов, журналістиці. Також проект http://dictengua.pythonanywhere.com/ може бути адаптований під будь-яку термінологію та використовуватись як навчальний портал для різних сфер діяльності.

Прикладний характер дослідження виражається у можливості формування бази даних міжнародних і національних термінів двох мов, а також у використанні результатів роботи для укладання двомовних галузевих словників та їх адаптації у багатофункціональні лінгвістичні платформи..

Структура роботи. Дослідження складається зі вступу, чотирьох розділів — двох теоретичних та двох практичних, висновків, списку використаних джерел та додатків.

РОЗДІЛ 1
СУЧАСНА КІНЕМАТОГРАФІЧНА ТЕРМІНОГРАФІЯ

Сучасна практика укладання словників і новітня теоретична лексикографія із розвитком комп'ютерних технологій набули стрімкого зростання і сприяли створенню великого фонду лексикографічних праць, що описують мову за різними аспектами на основі різних теоретичних засад. Українська лексикографія виконує свою основну функцію: відповідає інтересам суспільства, вирішуючи наукові, навчальні, пізнавальні та культурні завдання, що зумовлює визначення у межах цього напряму прикладної лінгвістики навчальної, довідкової та наукової лексикографії.

Кожен розробник лексикографічних продуктів має бути ознайомлений з базовим розділом лексикографії - термінографією. Цей напрям розвивається дуже стрімко, а її основним предметом є теоретичні та практичні питання щодо укладання й використання спеціалізованих термінологічних словників.

Кожен вияв свого існування людина фіксує певним знаком, кожен крок супроводжується певним лексичним набором. Внаслідок, кожна національна мова наповнюється великою кількістю спеціальних лексичних одиниць – термінів, які потребують обов’язкового упорядкування та уніфікації. Термін є основним складником спеціальної лексики, яку називають термінологією.

Згідно з Дарчук Н.П. «Термін – лексична одиниця (слово чи словосполучення), що вживається у певній підмові та позначає загальне (конкретне чи абстрактне) поняття певної галузі знань» [6, ст. 11]. Кожний крок дослідження у певній сфері, професійному середовищі закріплюється у термінах, бо в них відображаються факти, що спостерігаються і осмислюються дослідниками.

Кожна наука, сфера діяльності чи будь-яка спеціалізація має власну терміносистему. Історія укладання цих терміносистем може бути різною, залежно від давності існування галузі. Так, наприклад, кінематографія є відносно новим напрямом у термінографії, проте термінологію цієї сфери ми

зустрічаємо в повсякденному житті, навіть не сприймаючи цю лексику як терміни.

Попри активне вживання, кінематографічна лексика все ж має термінологічні ознаки. Кінематографічний термін – слово чи словосполучення на позначення кінематографічного поняття, що відображає специфіку явищ кіномистецтва та кіноіндустрії, використовується для номінування поняття кінематографії, і має задовольняти вимогам логічності, однозначності, системності, стислості, лінгвістичної правильності.

У зв'язку з тим, що кінематограф бере свій початок з Європи та Сполучених Штатів Америки, більша частина термінів була створена саме в межах англійської мови. Проте з початком розвитку цієї галузі в Україні, вся термінологія була або адаптована до української мови, або ж були впроваджені питомі відповідники.

1.1. Кінематографічна термінологія: становлення галузі та розвиток традиційної лексикографії

Українська термінографія у напрямку кінематографії лише починає своє існування. Так, на сьогодні існує лише один повноцінний словник кінотермінів - «Кінословник» Мирославського В.Н. [17] створений 2007 року.

Серед особливостей Кінословника можна назвати:

1. Кількість термінологічних статей - 578.

2. У словнику представлені не тільки класичні терміни але і жаргонізми з світового кінематографа.

3. Ця робота є першою в Україні спробою систематизації різнопланової інформації щодо стилів, жанрів і напрямів за всю історію кінематографа.

4. Словник поділений на тематичні параграфи. У деяких з них, як наприклад, в «Кіновиробництво», «Кінопрокат» та «Кінотехніка», подана

розгорнута інформація про розвиток кіновиробництва, кінопрокату і кінотехніки.

Словникова стаття складається з реєстрового слова й дефініції. До деяких термінів подано у дужках англійський переклад, а також до термінів, що позначають жанри фільмів, подані приклади фільмів цих жанрів.

В англійській лексикографії значно більше опрацьовується термінологія кіноіндустрії. Це пов’язано з тим, що саме американський кінематограф є найпопулярнішим та наймасштабнішим у всьому світі.

Серед усіх термінографічних праць цього відносно вузького напряму першим та найвідомішим є Глосарій Термінів Кіно (Film Terms Glossary). У цьому словнику подані найуживаніші терміни як базові для медіаграмотності [33]. Цей словник подає не повний реєстр термінів кіноіндустрії, тому що багато з них занадто незрозумілі або технічні, проте цей перелік покриває більшість важливих термінів і спонукає читача до вивчення термінології та подальших досліджень. При обговоренні фільму і творчості кіномистецтва багато з цих термінів використовуються для опису створення фільму.

До цього глосарію входять терміни таких тематичних груп:

- ключові теорії та аспекти історії кіно та кінокритики;
- жанри кіно;
- жаргони та неологізми;
- спеціальні матеріали (наприклад, види плівкового обладнання);
- основи кінематографії;
- різноманітні процеси, пов’язані з виготовленням фільмів (сценарій, режисура, накладання спецефектів, озвучення, редагування тощо);
- професії кіноіндустрії.

Також можна спостерігати, як багато загальномовних англійських словників почали розмежовувати термінологію різних галузей, у тому числі і виокремили термінологію кінематографії. Серед таких словників “Оксфордський словник (Oxford dictionary)” [35] та “Словник Макмілана (Macmillan’s dictionary)” [34]. Усі названі словники є у відкритому веб-застосунку.

1.2. Електронна термінологічна лексикографія: тенденції розвитку та перспективи

Завдяки сучасним технологіям, комп’ютери надають можливість користувачу зберігати великі обсяги інформації в електронному вигляді. За допомогою таких комп’ютерних систем можна знайти, відредагувати та розширити будь-яку необхідну інформацію, в тому числі і лексикографічну.

Можна виділити декілька особливостей електронної лексикографії:

− можливість зберігати будь-який обсяг інформації завдяки використанню гіперпосилань;

− можливість паралельного пошуку декількох елементів словника;

− наявність функції переходу між різними словниковими статтями;

− можливість використання мультимедійних елементів: звукові й анімаційні ілюстрації;

− постійне доповнення та оновлення словника;

− за правильним виконанням, більшість систем є доступними та простими для користування;

− економія часу та матеріальних ресурсів.

Окрім власне комп’ютерної, можна виокремити ще одну популярну сучасну галузь - корпусну лексикографію.

Корпусна лексикографія представляє суттєві зміни у співвідношенні між текстами мови та описами, що містяться у словниках. Великі мовні корпуси, що зберігаються та аналізуються на комп'ютері, стали найкращим джерелом для створення користувацьких лексикографічних словників.

Основним поняттям у створенні сучасних лексикографічних систем став електронний словник. Це база, яка містить дані у вигляді словникових статей та є доступною у цифровій формі через багато різних носіїв інформації. Електронні словники можна знайти у декількох формах, зокрема:

- як спеціальні портативні пристрої;
- як програми на смартфонах та планшетних комп'ютерах чи комп'ютерному програмному забезпеченні.

Електронні словники поділяються на ті, які виконують тільки одну функцію і ті, які в змозі виконувати декілька функцій.

До електронних словників з однією функцією можна віднести:

- енциклопедичні словники;
- тлумачні словники;
- спеціальні словники (історичний, етимологічний, складних випадків наголошення);
- термінологічні;
- частотні словники;
- словники - конкорданси.

До словників, що виконують більше однієї функції, належать:

- перекладні (двомовні та багатомовні словники);
- інтегровані (ті, що містять взаємозв'язок декількох словників/ модулів);
- аудіословники.

Основним типом електронних словників є одномовні тлумачні словники, головним завданням яких є тлумачення значень слів засобами цієї ж мови. Уже сама назва словника «тлумачний» дає уявлення про його основну функцію - «тлумачити», пояснювати те, що незрозуміло.

Серед найвідоміших електронних тлумачних словників української мови є «Академічний тлумачний словник сучасної української мови» [32], у якому користувач має змогу вести пошук необхідних лексем, а в результаті отримує тлумачення шуканих слів та приклади їх вживання у літературі.

Підвидом тлумачних словників можна назвати термінологічні словники. Зовсім нещодавно тлумачні словники створювались для загальної лексики. А тепер все більшої популярності набирають словники вузького профілю. Вони є надзвичайно корисними для поглибленого вивчення тієї сфери діяльності, яка насамперед цікавить користувача. Відмінністю термінологічного словника від тлумачного є спосіб подання дефініцій. Якщо тлумачний словник подає усі можливі тлумачення слова, то термінологічний - лише те значення, яке належить до певної галузі.

На базі термінологічної лексики нерідко будуються частотні словники та конкорданси.

Частотні словники - це словники, у яких містяться числові характеристики вживаності слів, слова в них розташовуються залежно від частоти вживання слів у текстах певної довжини. Це порівняно новий тип лексикографічних видань, виникнення якого пов’язане з розвитком статистики, обчислювальної техніки.

Сьогодні для статистичних досліджень у мовознавстві широко застосовуються електронні частотні словники. Однак, як правило, вони є дорогим програмним продуктом, а тому не завжди доступні. Тому в навчальному процесі доцільно використовувати частотні словники, створені на основі окремих текстів з використанням загальнодоступних програмних засобів.

Ще одним корисним словником є конкорданс. Це дуже особливий вид словника, адже в ньому кожна лексема розташована не просто в алфавітному порядку і має тлумачення, а ще й додається мінімальний контекст і випадки вживання у різних позиціях. Також, конкорданс інколи називають словником контекстів або ж сполучуваності мовних одиниць.

У XXI столітті, ері великих технологій та можливостей, мовленнєві технології стали одними з найбільш досліджувальних. Так, внаслідок прогресу мовленнєвих технологій лексикографи почали використовувати комп'ютер не лише для укладання частотних та контекстних словників, але й створювати так звані аудіословники, використовуючи автоматичний синтез мовлення. Здебільшого, це словники тлумачні або перекладні, такі як «Oxford dictionary with audio» [35] або «Macmillan's online dictionary» [34]. Ці словники містять аудіовідтворення лексем, що дозволяє почути правильну вимову слова та є корисними для вивчення іноземної мови.

Автоматичний синтез мовлення - це технологія, що дозволяє перетворити вхідну текстову інформацію в озвучене мовлення. При цьому одним із найважливіших аспектів є якість синтезу. Саме від якості залежить придатність використання технології синтезу мовлення на сучасному комерційному рівні. Під системами автоматичного синтезу мовлення (інакше їх ще називають синтезаторами мовлення) в цьому дослідженні розуміються системи, що перетворюють орфографічний текст та іншу інформацію в озвучене мовлення. Загальноприйняте в англійській літературі позначення - TTS (Text To Speech) System - системи перетворення тексту в мову.

Технологія автоматичного синтезу мовлення може бути корисна у найрізноманітніших галузях і напрямах, таких як:

- телекомунікації;
- мобільні пристрої;
- промислові і побутові електронні пристрої;
- автомобільна індустрія;

- освітні системи;
- комп'ютеризовані системи;
- Internet-сервіси.

На сьогодні існує багато систем синтезу мовлення. Наприклад, синтезатор мовлення «Infovox», перша версія якого вийшла у 1982 році й базувалася на синтезі формант, зараз здійснює синтез мовлення американською та британською англійською, голландською, датською, ісландською, іспанською, італійською, німецькою, норвезькою, фінською, французькою та шведською мовами.

Мовний синтезатор від компанії Digital Equipment Corporation доступний трьома мовами: американською англійською, німецькою та іспанською з можливістю вибору із 9 голосів: 4 жіночих, 4 чоловічих та 1 дитячого. Мовний синтезатор компанії AT&T Bell Laboratories з'явився у 1973 році. Зараз система здійснює синтез мовлення для таких мов: англійська, іспанська, італійська, китайська, німецька, російська, румунська, французька, японська.

На базі вище названих синтезаторів було створено найвідоміший на сьогодні синтезатор, який озвучує текст для Windows, Google Cloud Text-to-Speech API, який перетворює текст у мовлення, схоже на людське, у більш ніж 100 варіантах голосів та на 20 мовах. У цій системі застосовуються новаторські дослідження в синтезі мовлення (WaveNet) та потужні нейронні мережі Google для забезпечення високоякісного звуку.

В основу кожної лексикографічної системи покладено термінологію, яка від початку свого становлення стала невід'ємним і важливим складником лексичної системи мови. Крім того, її неможливо відокремити від наукової мови, що дає змогу називати зафіксовані досягнення кожної окремої галузі знань на певному етапі її розвитку. Про рівень прогресу в певній галузі свідчить багатство та досконаліть термінології, що обслуговує цю галузь.

Так, за словами Д. С. Лотте, термінологія – це «…не просто список термінів, а семіологічне вираження певної системи понять, яка, своєю чергою, відбиває певний науковий світогляд» [16, ст. 38].

Термінологію кожної науки чи галузі знань структурують у спеціалізовані термінологічні словники.

Термінологічні словники - це сбази, у яких систематизовано терміни, які вживаються у певній галузі науки, розтлумачено значення їх лише відносно конкретної сфери. Такі словники можуть бути:

- одномовними;

- двомовними (перекладними).

Українська мова має термінологічні словники з багатьох галузей науки:

- мінералогії;

- біології;

- медицини;

- математики;

- літературознавства;

- мовознавства;

- спорту та ін.

Термінологічні словники (якщо виходити зі специфіки побудови словникових статей) не настільки наповнені, як загальномовні. Така особливість виникла через те, що здебільшого вони представляють термінологічну лексику звуженої тематики.

Поява нових словників термінологічного змісту зумовлена розвитком нових технологій, появою та активним розвитком нових сфер людської діяльності та необхідністю структурувати інформацію та лексику, яка буде

відображенням цих сфер. Також, електронні термінологічні системи користуються високим попитом сьогодні через велике бажання людей вивчати нові напрями діяльності, а для програмних розробників такі системи стали об’єктом для розроблення більш масивних за наповненням та багатофункціональних продуктів.

Висновки до першого розділу

Електронна лексикографія використовується як загальний термін для позначення проектування, використання та застосування електронних словників (ЕС), які, в свою чергу, визначаються як насамперед орієнтовані на людину збірки структурованих електронних даних, що дають інформацію про форму, значення та використання слова однією або кількома мовами і зберігаються в різних пристроях (ПК, Інтернет, мобільні пристрої).

Конструювання електронних словників для певної галузі передбачає поглиблене вивчення спеціалізованої термінології і слід чітко відрізняти термін від загальновживаної лексики. Якщо узагальнити всі існуючі тлумачення для поняття «термін», то це слово або вираз, яке має точне значення в деяких випадках або характерно для конкретної галузі науки, мистецтва, професії або предмета.

Сучасний стан електронної лексикографії в Україні доволі спірний, якщо гоаорити про вузькі сфери діяльності. Так, наприклад, розробники електронних систем поки що уникають напрям кінематографії і цьому є пояснення: термінологія індустрії кіно є дуже нестабільною і постійно оновлюється, тому складно зібрати чіткий матеріал для термінологічної системи. Сьогодні існує лише один «Кінословник» [17], який за сучасними параметрами вже є застарілим.

Хоча за вузькою тематикою Українська лексикографія поки що не є лідером, у створенні загальних електронних лексикографічних систем розробники дотримуються найсучасніших методів. Так, досить нещодавно з’явилось поняття інтеграції у словникових системах, завдання якої згідно з

Широковим В. А. «... потребує розроблення методів інтеграції концептуальних моделей, способів подання даних та операційно програмних платформ, а також узгодження зовнішніх представлень відповідних концептуальних схем та їх внутрішніх репрезентацій» [28, ст. 128].

Результатом такої діяльності має бути інтегрована лексикографічна система, яка є об'єднанням кількох програмних засобів, які працюють разом, та як результат інтеграції надання користувачеві можливостей, не притаманних кожному з цих словників поодинці.

Через те, що напрям інтеграції ще не є досить поширеним, головним завданням цієї роботи стало зрозуміти процес та особливості конструювання інтегрованих систем та розробити інтегровану електронну лексикографічну систему кінотермінів

Процес інтеграції термінологічної лексикографічної системи ускладнюється тому, що потрібно поєднувати складні об'єкти, а це неможливо без спеціального програмного забезпечення та комбінування різних за структурою даних.

Для інтеграції в цьому проєкті було обрано найбільш актуальні види лексикографічних систем: термінологічний словник, перекладний, частотний словник, словник-конкорданс та була додана можливість аудіовідтворення кожного елемента словникової статті.

Функціонування програми на головній сторінці відбувається через 3 входи, які є взаємопов'язаними між собою переходами для отримання інформації для кінотермінів:

1 вхід - пошук лексеми або словоформи українською чи англійською мовами у полі «Пошук»;

2 вхід - після натискання кнопки «Пошук» отримуємо реєстр кінотермінів українською мовою, через який є можливість перейти на словникову статтю лексеми, натиснувши на гіперпокликання слова, що цікавить.

3 вхід - після натискання кнопки «Пошук» отримуємо реєстр кінотермінів англійською мовою, через який є можливість перейти на словникову статтю лексеми, натиснувши на гіперпокликання слова, що цікавить.

У межах кожної словникової статті є можливість отримати таку інформацію:

1. Кінотермін з аудіовідтворенням його вимови.
2. Дефініцію терміна та озвучення дефініції.
3. Приклади вживання кінотерміна та його словоформ з озвученням речення, в якому вжито шуканий термін (за наявності декількох контекстів можливість проглянути та прослухати кожен з них).
4. Переклад кінотерміна.

Додатковим елементом цієї інтегрованої системи є частотний словник, в якому можна спостерігати частотність вживання всіх кінотермінів словника у текстах двох створених корпусів (корпусу українських та корпусу англійських текстів).

Також, для того, щоб користувач мав змогу детально ознайомитись з інформацією не лише про окремі терміни але і про відомі імена у світі кіно було створено систему розпізнавання іменованих сутностей. У майбутньому, за допомогою впровадженого модулю з розпізнавання іменованих сутностей стане можливим кластеризація інформації кінематографічної тематики. Наприклад:

Назва фільму -> Імена акторів, які знялися у цьому фільмі -> Імена героїв, яких зіграли ці актори

РОЗДІЛ 2

СУЧАСНІ МЕТОДИ ОБРОБЛЕННЯ ТЕКСТОВОЇ ІНФОРМАЦІЇ

Сучасна лінгвістика дивує швидкістю розвитку та створенням методів для автоматичної оброблення текстової інформації і головним кроком до автоматизації оброблення текстової інформації є машинне навчання.

Машинне навчання - це технологія штучного інтелекту (ШІ), яка надає системам можливість автоматично вчитися на досвіді специфічного програмного забезпечення і може допомогти з високою точністю вирішити складні проблемні моменти. Таку технологію було впроваджено в сучасні лінгвістичні дослідження через те, що за умови якісного програмування подібні системи можуть конкурувати або навіть іноді перевершувати «ручні» методи оброблення даних та значно пришвидшують процес оброблення інформації.

Ключовим напрямом у машинному навчанні є NLP (Natural language processing) або ж оброблення природної мови (ОПМ).

У галузі ОПМ комп’ютери розумно та корисно аналізують та виводять необхідні значення з людської мови. Використовуючи NLP, розробники мають змогу організувати та структурувати знання для виконання таких завдань, як автоматичне реферування, переклад, розпізнавання іменованих сутностей, вилучення зв’язків, емотивний аналіз та кластеризація за темами..

ОПМ допомагає системам швидше аналізувати дані: комп’ютери та системи повинні вміти розуміти великий обсяг мовних даних, щоб відбувалася коректна «співпраця» між подібними алгоритмами. Це пов’язано з тим, що вони можуть вчитися на шаблонах, що містяться в збережених даних.

ОПМ системи спочатку були розроблені, щоб допомогти забезпечити семантичне розуміння мов. Отже, спілкування між людьми та машинами призвело до логічної та позитивної взаємодії. Якщо описати коротко, системи

автоматичного оброблення мови допомагають структурувати великий обсяг різноманітної текстової інформації, аби полегшити людині подальшу роботу з аналізом отриманих даних.

Автоматичне оброблення природної мови призначена для виконання конкретних завдань. Деякі основні завдання АОПМ - це автоматичне реферування, аналіз дискурсу, машинний переклад, розпізнавання мови тощо.

Автоматичне реферування допомагає комп'ютеру подавати користувачу стислий варіант певного тексту, статті, журналів тощо. Аналіз дискурсу допомагає комп'ютеру зрозуміти, як слід підтримувати і поєднувати потік речень. Машинний переклад або МП допомагає комп'ютеру перекладати текст або мовлення з однієї мови на іншу. Розпізнавання мови використовується для визначення текстового подання вимовленного речення. Деякі відомі програми, такі як Siri та Cortana, є чудовими прикладами розпізнавання мови.

2.1 Автоматичне реферування як один із ключових напрямів ОПМ

Автоматичне реферування - це напрям оброблення природної мови, який сьогодні все частіше використовується у повсякденному житті. Метою реферування є створення короткого викладу одного документа або сукупності документів, із повним збереженням змісту та важливих аспектів текстів. В той самий час результат має бути поданий у значно меншому обсязі, ніж початкові файли, в ідеалі, розмір вихідного файлу має бути визначений користувачем.

Автоматичне реферування документів використовується з середини минулого століття, і з тих пір воно розробляється та адаптується до нових технологій. Зокрема, з моменту, як Інтернет став загальнодоступним інформаційним ресурсом, з'явилася потреба у засобах для створення витягів із масиву доступної інформації (це може бути як конкретний запит користувача, так і загальний).

Існують різні види результату у процесі автоматичного реферування. Можна виокремити екстрактивні та абстрактні резюме. Ця відмінність стосується якості представленого результату. Екстракційне резюме, по суті, є зменшеною версією оригіналу, оскільки воно витягує речення з джерела, не змінюючи в них нічого. Абстрактне резюме є більш складним, оскільки воно передає основну інформацію іншими словами без цитування оригіналу.

Крім того, розрізняють реферування окремих та кількох документів. Узагальнення сукупності документів є особливо корисним для вилучення інформації з Інтернету. Можна узагальнити кілька документів, які стосуються однієї заданої теми.

Існує ще одна класифікація автоматичних резюме, яка базується на змісті. Орієнтовне резюме - це те, яке повідомляє читачеві, про що йдеться в одному або кількох текстах. Інформативне резюме відтворює основну інформацію оригіналу і може бути використане як його заміна.

Щодо методів, які використовуються на сьогодні для створення автоматичних рефератів, то, як відомо, більшість систем створюють екстрактивні резюме. Завдання полягає в тому, щоб система визначила, які речення є досить важливими, щоб містити короткий зміст повного тексту та зберегти основну інформацію.

Однак екстрактивні резюме мають багато недоліків, особливо порівняно з резюме, створеним людиною. Оскільки екстрактивні резюме шукають повні речення, включаючи важливу інформацію, вони повністю відтворюють речення і не враховують мовні варіації, читабельність та не видаляють зайві частини з виокремлених речень.

2.2 Розпізнавання іменованих сутностей у лінгвістичному машинному навчанні

Розпізнавання іменованих сутностей або ж суб'єктів (РІС) є важливим завданням виявлення імен у поданому реченні для розуміння природної мови. Іменована організація (ІО) насамперед стосується імені особи, місцезнаходження чи організації, але іноді доводиться враховувати більший набір суб'єктів. Більш формально завдання розпізнавання і класифікації іменованих суб'єктів чи об'єктів можна описати як ідентифікацію іменованих сутностей у машиночитаному тексті за допомогою анотації з тегами категоризації для вилучення інформації. На сьогоднішній день, РІС є новою гілкою у машинному навчанні та стало одним з найскладніших методів оброблення текстів.

Продуктивність системи РІС, заснованої на машинному навчанні, залежить від обсягу даних, що використовуються для навчання системи, та особливостей, що використовуються для побудови моделі. Деякі мови світу мають велику кількість анотованих даних для підготовки досить доброї системи РІС. Однак існує низка мов, які страждають від дефіциту великих даних, котрі анотуються до іменованої організації. Насправді, набір даних навчання для РІС існують лише для обмеженого поєднання доменів та жанрів (наприклад, письмових новин), навіть для тих мов, які досить багаті на інформаційні ресурси.

Іменовані сутності часто є не просто одиничними словами, а фрагментами тексту, напр: Верховна Рада України, Національний банк України, Університет штату Меріленд, округ Балтімор або Центральний банк Австралії. Тому для передбачення належності групи лексем до однієї сутності потрібна деяка модель передбачення або синтаксичного аналізу.

Результатом успішної роботи такої РІС системи є визначення, що виділений сегмент насправді є іменованою сутністю, або, що інколи

важливіше, визначення класифікації іменованої сутності, особливо в ситуаціях, де існує неоднозначність. Наприклад, «Вашингтон» може стосуватися або імені, або місця. «Галактика» може позначати загальний іменних або професійну футбольну команду вищої ліги. Для проведення цього аналізу використовуються моделі максимальної ентропії, приховані моделі Маркова, алгоритми пошуку Вітербі та інші статистичні методи, які зазвичай реалізуються як система машинного навчання.

Історія експериментів з РІС дуже насичена різними напрямами та підходами. Так розпізнавання іменованих сутностей поділяється на два підходи на основі того, яка інформація використовується в текстових даних: синтаксична інформація або семантична інформація.

По-перше, відомо, що система РІС, яка використовує синтаксичну інформацію, добре працює з невеликими і чіткими наборами даних, оскільки синтаксичні вирази у реченні, необхідні для визначення категорії кожного цільового слова, можуть бути легко вилучені з таких текстів. Наприклад, слова, пов’язані з категорією імен, мають бути суб’єктами, про які йде мова в реченні; слова, пов’язані з категорією розташування, з’являлися б відразу після прийменників, а слова, пов’язані з категорією об’єкта, у багатьох випадках мали б бути іменниками. Цей підхід показує задовільну точність, якщо текстові дані мають чіткий та стандартний формат. Отже, для роботи з іменованими сутностями слід вилучити з текстових речень те, що називає об’єкт, наприклад, предмет, тему, кількісне відношення.

Подібна модель має давати багатообіцяючі результати у виділенні інформативних ключових слів, проте всі вони вимагали онтологій (тобто взаємозв'язку між текстовими словами) для побудови правил вилучення. З іншого боку, РІС, що використовує семантичну інформацію, добре відоме своєю надійністю та розширюваністю порівняно із синтаксичним

підходом. Модель автоматично визначає схеми використання кожного слова в тексті та отримує семантичну інформацію із шаблонів на основі алгоритму машинного навчання.

Завдяки спробам створити подібну систему, було опрацьовано безліч комбінацій методів та моделей для гарантування коректної роботи РІС і через те, що сьогодні ще не існує жодної бугатомодулевої системи, яка б мала подібну функцію та була у вільному доступі для користувачів кінцевий продукт цієї роботи є повністю новим та унікальним.

Висновки до другого розділу

Сучасний рівень лінгвістики дивує кожного, адже тепер це не лише теоретичні дослідження або експерименти з «ручною» обробкою великого обсягу інформації, а і активне впровадження сучасних технологій. Настільки активне використання технічних новинок допомагає не лише полегшити роботу лінгвістам, а й забезпечує користувачам швидкий доступ до важливої інформації.

Ще не так давно машинне навчання стало невід’ємною частиною прикладної лінгвістики але вже можна побачити неймовірні корисні результати у різних напрямках. Так, активно почались програмні роботи з автоматичним реферуванням текстів та розпізнаванням іменованих сутностей. І хоча, спроби автоматичного реферування були ще в минулому столітті, зараз ці результати вражають. адже лише за допомогою комп’ютерів можна обробити настільки об'ємну інформацію.

Одним з невід’ємних напрямів лінгвістики також нещодавно стала і робота з розпізнаванням іменованих сутностей (РІС), яка може розцінюватись навіть як один з методів для автоматичного реферування.

Після детального вивчення особливостей автоматичної оброблення природної мови, було прийнято рішення додати до інтегрованої лексикографічної системи можливість автоматичного реферування та розпізнавання іменованих

сутностей.

Кожен з цих модулів є корисним окремо:

1. Автоматичний реферат стане корисним для тих, хто бажає виокремити в тексті основні моменти.
2. оброблення іменованих сутностей допоможе у великих системах кластеризувати інформацію окремої тематики.

Було багато спроб створити модуль, який бі виконував роботу корректно і протестувавши безліч комбінацій було створено продукти, один з яких подає якісний реферат до користувацького тексту, а інший деталізує іменовані сутності.

РОЗДІЛ 3

КОНСТРУЮВАННЯ ЕЛЕКТРОННОЇ ЛЕКСИКОГРАФІЧНОЇ СИСТЕМИ КІНОТЕРМІНІВ

Відповідно до макету словникової статті, описаної у першому розділі на сторінці 18, були сформульовані поетапні завдання конструювання словника термінів кінематографії:

- створення бази даних українських кінотермінів;
- створення бази даних англійських термінів за принципом перекладу реєстру українських термінів;
- автоматичне укладання частотного словника кінематографічних термінів;
- автоматичне укладання корпусу текстів кінематографічної тематики;
- автоматичне укладання конкордансу до реєстру українського та англійського словників;
- автоматичне розпізнавання іменованих сутностей;
- автоматичне реферування тексту.

Реєстр українських кінотермінів укладено за словником Володимира Мирославського “Кінословник. Терміни, визначення, жаргонізми”. Реєстр укладено за принципом підбору 100 найчастотніших та найактуальніших кінотермінів у пошуковій системі Google. Реєстр англомовних термінів укладено способом перекладу реєстру українських термінів.

Корпуси текстів створені методом добуття текстів із сайтів новин кіно: англомовні тексти - 12587 текстів - із сайту Cinemablend [42]; українськомовні – 10535 текстів - із сайту MoviesTape [45], KinoFilms [44] та Кіно-Театр [43].

Укладання конкордансу передбачає використання словоформ для наявних у реєстрі термінів, які були прописані у базі вручну без автоматичної лематизації.

Також до словника було додано реєстр іменованих сутностей. Для цього сформульовані такі завдання:

- Визначення іменованих сутностей із текстів корпусу.
- Автоматичне укладання частотного словника іменованих сутностей.
- Створення бази даних українських іменованих сутностей методом відбору 50 найчастотніших іменованих сутностей, що стосуються кінематографії.
- Створення бази даних англійських іменованих сутностей способом знаходження в реєстрі іменованих сутностей англомовних відповідників відібраних українських іменованих сутностей.
- Автоматичне укладання конкордансу до реєстру відібраних іменованих сутностей.

Крім того, користувачі мають змогу взаємодіяти із системою шляхом введення особистого тексту, до якого застосовується автоматичне реферування.

Задача конструювання інтегрованої електронної лексикографічної системи кінотермінів передбачала створення програмного продукту, за допомогою якого для визначених корпусів текстів кінематографічної тематики можна виконати такі дії:

- пошук тлумачення кінотерміна в термінологічному словнику;
- перегляд алфавітно-частотного словника кінотермінів та іменованих сутностей, пов’язаних з кінотематикою;
- пошук контекстів у корпусі для заданої словоформи кінотерміна, або для словоформи іменованої сутності, пов’язаної з кінотематикою;
- переклад кінотермінів та відображення відповідника для іменованої сутності;
- озвучення (звукове відтворення текстової інформації) термінів, їх тлумачень і контекстів у конкордансі;
- Автоматичне створення реферату на основі введеного користувачем довільного тексту;

- Перегляд короткого опису іменованої сутності;
- Перегляд тексту з корпусу, у якому промарковано іменовані сутності.

3.1. База даних електронного словника кінотермінів

Важливим етапом створення програми є проєктування структури бази даних. База даних програми складається з десяти таблиць, *(див. рис. 3.1)*:

- terms - таблиця термінів;
- wordforms - таблиця словоформ;
- corpora - таблиця корпусів;
- corpora_texts - таблиця з текстами корпусу;
- context - таблиця з контекстами конкордансу;
- wordform_context_relation - таблиця із зв’язками словоформа - контекст;
- terms_ner – таблиця відібраних іменованих сутностей;
- wordforms_ner – таблиця словоформ відібраних іменованих сутностей;
- wordform_context_relation – таблиця із зв’язками словоформа іменованої сутності – контекст;
- named_entities – таблиця, що містить всі видобуті іменовані сутності.

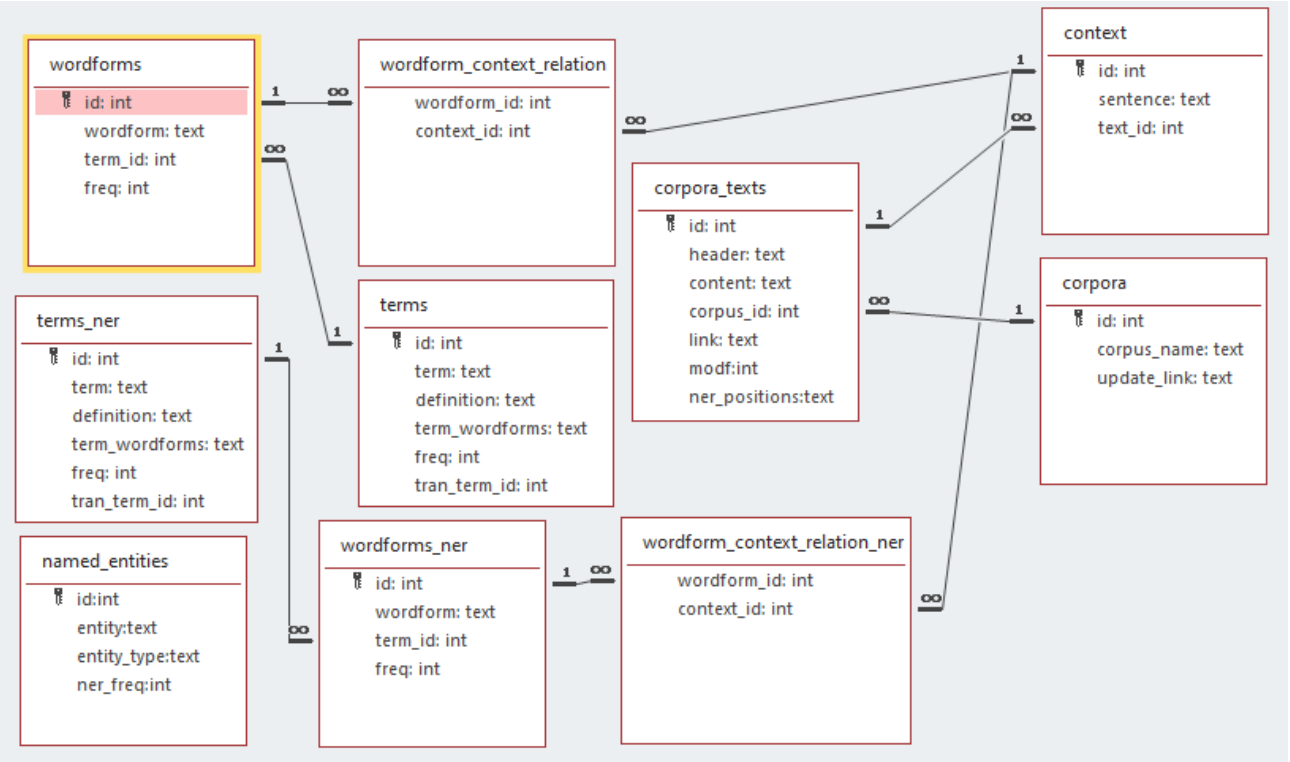


Рис. 3.1. Схема бази даних

Таблиця terms *(див. рис. 3.2)* складається із шести полів: 1) id - номер терміна в таблиці; 2) term - кінотермін, 3) definition - тлумачення терміна; 4) term_wordforms - текст, що відповідає списку словоформ терміна; 5) freq - частота вживання кінотерміна у корпусі; 6) tran_term_id - номер терміна в таблиці для перекладного словника (якщо термін український - поле містить номер відповідного йому англійського терміна та навпаки).

Table: terms

Filter in any column

	id	term	definition	term_wordforms	freq	tran_term_id
	Filter	Filter	Filter	Filter	Filter	Filter
100	100	рецензія	публікація, в якій обговорюється та оцінюється фільм,	рецензія;рецензії;рецензію;рецензією;рецензіє;рецензій;рецензіям;р...	169	200
101	101	2-d	a movie with a simple image	2-d;2d;	32	1
102	102	3d	a movie with images having three dimensional form or appearance	3d;3-d;	444	2
103	103	action movie	a movie genre in which the protagonist or protagonists are thrust into a series of ...	action movie;action movies;	255	3
104	104	blockbuster	an unusually successful hit with widespread popularity and huge sales (especially a ...	blockbuster;blockbusters;	1665	4

Рис. 3.2. Фрагмент таблиці terms

Таблиця wordforms *(див. рис. 3.3)* призначена для зберігання інформації про словоформи та складається з чотирьох полів: 1) id - номер словоформи в таблиці; 2) wordform - словоформа; 3) term_id - поле-зв’язок із таблицею термінів, що позначає номер терміна словоформи; 4) freq - кількість слововживань словоформи.

Table: wordforms

	id	wordform	term_id	freq
	Filter	Filter	Filter	Filter
9	9	бойовику	3	141
10	10	бойовиками	3	3
11	11	бойовиках	3	5
12	12	бойовика	3	315
13	13	бойовикам	3	1
14	14	бойовикові	3	0
15	15	блокбастер	4	101
16	16	блокбастеру	4	21
17	17	блокбастером	4	22
18	18	блокбастери	4	31

Рис. 3.3. Фрагмент таблиці wordforms

Таблиця corpora *(див. рис. 3.4)* містить інформацію про корпуси: номер корпусу в таблиці (поле id), назву корпусу (поле corpus_name) та URL-покликання для оновлення корпусу (поле update_link).

Table: corpora

	id	corpus_name	update_link
	Filter	Filter	Filter
1	1	CinemaBlend	https://www.cinemablend.com/news
2	2	MovieStape	http://moviestape.net/novyny_kino/
3	3	Other texts (English)	NULL
4	4	Other texts (Ukrainian)	NULL
5	5	KinoFilmsUa	https://www.kinofilms.ua/ukr/news/
6	6	KinoTeatrUa	https://kino-teatr.ua/uk/main/news

Рис. 3.4. Таблиця corpora

Таблиця corpora_texts *(див. рис. 3.5)* зберігає тексти корпусів та складається із семи полів: 1) id - номер тексту в таблиці; 2) header - заголовок тексту; 3) content - власне текст, 4) corpus_id - поле зв’язок із таблицею corpora, що позначає номер корпусу тексту; 5) link - текст, що відображає гіперпокликання на сторінку сайту, з якої було взято текст; 6) спеціальний

параметр modif – індикатор, що вказує на якому етапі оброблення перебуває текст: 0 – текст не оброблено, 1 – у тексті розпізнано іменовані сутності, 2 – текст враховано в частотному словнику, 3 – текст враховано при укладанні конкордансу; 7) ner_positions – індекс позицій іменованих сутностей, розпізнаних у тексті.

Table: corpora_texts — Filter in any column

	id	header	content	corpus_id	modif	link	ner_positions
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
5886	5886	Два Уілла Сміта в новому трейлері "Двійника"	"Двійник" – фантастичний екшн режисера Енга Лі ("Життя ...	6	3	https://kino-teatr.ua/uk/news/...	[16063:(39,46);59797:(49,58);...
5887	5887	Генрі Кавілл зіграє Шерлока Холмса	Генрі Кавілл отримав роль у фільмі "Енола Холмс" – адапта...	6	3	https://kino-teatr.ua/uk/news/...	[57672:(6,12);56625:(36,47);...
5888	5888	Галь Гадот в трейлері комедії "Натурал під ...	"Натурал під прикриттям"– ізраїльська комедія 2014 роки ві...	6	3	https://kino-teatr.ua/uk/news/...	[59800:(69,79);10247:(115,124);...
5889	5889	Сальма Хайєк з'явиться в "Вічних" від Marvel?	Схоже, Marvel вирішив отримати всіх зірок сучасного ...	6	3	https://kino-teatr.ua/uk/news/...	[607:(7,13);17187:(89,101);5625...
5890	5890	Stephen Amell Spent His Christmas With Arrow, ...	Stephen Amell's dedication to The CW's Arrow did not take ...	1	3	https://www.cinemablend.com/...	[59808:(0,13)|(437,450)|(573,586...
5891	5891	Netflix's The Ranch Vet Seemingly Debunked ...	Spoilers for the Part 7 finale of Netflix's The Ranch are ...	1	3	https://www.cinemablend.com/...	[59843:(22,23)|(427,428);59844:...
5892	5892	See How Will And Grace's Stars Celebrated ...	Will and Grace has had a fascinating tenure on the small ...	1	3	https://www.cinemablend.com/...	[59881:(9,14)|(484,489)|(651,656...

Рис. 3.5. Фрагмент таблиці corpora_texts

Таблиця context *(див. рис. 3.6)* складається з трьох полів: 1) id - номер контексту в таблиці; 2) поле sentence - зберігає контекст; 3) text_id - поле-зв'язок із таблицею "corpora_texts", позначає номер тексту контексту.

Table: context — Filter in any column

	id	sentence	text_id
	Filter	Filter	Filter
248248	248248	However, one part of the resort is back at work as construction is continuing on Avengers Campus, the new land being added to Disney California Adventure.	12562
248249	248249	While we're no more sure when the new land will be ready, it did just add a massive new prop to the land in the form of an Avengers QuinJet and now you can get what will likely be the closest ...	12562
248250	248250	Disney Parks, Experiences, and Consumer Products Chairman Josh D'Amaro shared a couple of images to Instagram of him inside the new Avengers Campus.	12562
248251	248251	The first shows him standing by the new QuinJet display, showing off the size of the thing.	12562
248252	248252	We also see him with Disneyland Resort President Kevin Potrock, and Marvel Studios chief Kevin Feige, with the jet in the background which shows some work that needs to still be done.	12562

Рис. 3.6. Фрагмент таблиці context

Таблиця wordform_context_relation (*див. рис. 3.7*) зі зв'язками - словоформа – контекст містить id словоформи та id контексту. На відміну від інших таблиць, ця таблиця не потребує поля первинного ключа для ідентифікації записів, оскільки кожен запис (пара словоформа – контекст) є унікальним у межах таблиці.

Table: wordform_context_relation

	wordform_id	context_id
	Filter	Filter
313776	1640	523164
313777	1640	523677
313778	1641	88926
313779	1641	89006
313780	1641	89698

Рис. 3.7. Фрагмент таблиці wordform_context_relation

Зв’язок словоформа-контекст нам необхідний і для побудови частотного словника.

Схеми таблиць terms_ner, wordforms_ner, wordform_context_relation_ner збігаються зі схемами таблиць terms, wordforms та wordform_context_relation відповідно.

Таблиця named_entities складається з чотирьох полів: 1) id іменованої сутності, 2) entity – власне іменована сутність, 3) – entity_type – тип іменованої сутності; 4) ner_freq – частота іменованої сутності в усіх текстах корпусу.

Наповнення бази даних складається з таких етапів:

1. Імпорт словоформ із текстового файлу;
2. Завантаження (скачування) текстів тематики кінематографу із сайтів;
3. Розпізнавання іменованих сутностей та їх збереження в базі даних;
4. Укладання частотного словника;
5. Укладання конкордансу.

Для словоформ, які складаються з більше, ніж одного слова, крім словоформ, розділених звичайним пробілом (Unicode-символ 32), також генеруються ці ж словоформи, але розділені нероздільним пробілом (Unicode-символ 160).

Приклади словоформ: “Ніколай Костер-Вальдау (Джеймі Ланністер з "Ігри престолів") приєднається до Тому Крузу і Моргану Фрімену на зйомках ***фантастичного фільму*** "Oblivion", де мова йтиме про те, як група землян пручається мерзенним інопланетянам, які захопили Землю.”(https://www.kinofilms.ua/ukr/news/4184/)

“Режисер Террі Гілліам сумнівається, що ще буде ***знімати фільми*** вмайбутньому” (https://www.kinofilms.ua/ukr/news/3859/)

“"Я взагалі-то досяг моменту, коли вже починаю сумніватися, що буду ще ***знімати фільми***. Я став легко відволікатися і зробився надмірно жадібним, щоб продовжувати робити фільми ... А коли я все ж беруся за них, все йде шкереберть. Раніше я думав, що можу досягати всього силою волі, але тепер я старше і розумію, що так не буває ".” (https://www.kinofilms.ua/ukr/news/3859/)

“Ходять чутки, що Енді і Лана Вачовскі обговорюють з Наталі Портман можливість її участь у прийдешньому ***науково-фантастичному фільмі*** Вачовскі під назвою "Схід Юпітера" (Jupiter Ascending).” (https://www.kinofilms.ua/ukr/news/3888/)

“Купуйте квитки на сеанси «***Чорної пантери***» на нашому сайті.” (https://www.kinofilms.ua/ukr/news/14953/)

“В листопаді Момоа вирішив відвідати зйомки мегапопулярного фентезі « ***Гра престолів*** », де возз'єднався зі своєю екранною дружиною Кларк.” (https://www.kinofilms.ua/ukr/news/14491/)

“Персонажі « Першого месника: Протистояння » – окрема тема, і саме на ній тримається вся ця історія і ***Кіновсесвіт Marvel*** як такий.” (https://www.kinofilms.ua/ukr/news/8308/)

Додавання словоформ до таблиць wordforms та wordforms_ner після прочитання txt-файлів словоформ до відповідних полів таблиць terms та terms_ner продемонстровано таким кодом:

```
        def process_wordforms(self):
wordforms_to_insert = []
# залежно від вибраного режиму, вказується назва таблиці, з якої буде
# прочитано список словоформ термінів
if self.ner_mode:
    terms_table_name = "terms_ner"
else:
    terms_table_name = "terms"
# Видобування списків словоформ термінів з відповідної таблиці
# (id терміна, список його словоформ, розділений крапкою з комою)
wordforms_list_of_specific_term = \
    self.dbf.get_wordforms_from_terms_table(terms_table_name)
# Назва таблиці словоформ, де буде збережено словоформи
if self.ner_mode:
    wordforms_table_name = "wordforms_ner"
else:
    wordforms_table_name = "wordforms"
# словоформи, які наявні в таблиці словоформ
wordforms_already_in_db = \
    self.dbf.get_wordforms_from_wordforms_table(wordforms_table_name)
for i in range(len(wordforms_list_of_specific_term)):
    # розглядається список словоформ певного терміна із таблиці термінів
    id_ = wordforms_list_of_specific_term[i][0]  # id терміна
    # перевіряється, чи в полі списку словоформ значення не NULL
    if wordforms_list_of_specific_term[i][1] is not None:
        # список словоформ подається як список окремих словоформ
        lex_and_wordforms = \
            wordforms_list_of_specific_term[i][1].split(";")
        # розглядається кожна словоформа із списку словоформ
        for j in range(len(lex_and_wordforms)):
            wordform = lex_and_wordforms[j]
            # якщо словоформа не перебуває в таблиці словоформ,
            # дана словоформа додається до таблиці з відповідним id
            # терміна
            if wordform != "" and wordform not in \
                    wordforms_already_in_db:
                wordforms_to_insert.append((wordform, id_))
            # словоформа поділяється на окремі слова
            split_wform = wordform.split()
            if len(split_wform) > 1:  # якщо слів в словоформі декілька
                # відбувається заміна символа пробіла на символ
                # нероздільного пробіла
                new_wf = wordform.replace(chr(32), chr(160))
                # якщо таку словоформу ще не додано
```

```
            if new_wf != "" and new_wf not in \
                wordforms_already_in_db:
                # додається словоформа, слова якої розділено
                # нероздільним пробілом
                wordforms_to_insert.append((new_wf, id_))
    # збереження словоформ у відповідній таблиці словоформ бази даних
    self.dbf.add_new_wordforms_to_db(wordforms_to_insert,
                                     wordforms_table_name)
```

Скачування текстів з сайтів здійснено за допомогою веб-краулінгу – це дослідження гіпертекстової структури веб-сторінки з метою створення записів індексації пошуку.

Для скачування текстів необхідно встановити такі пайтон-бібліотеки: **requests, lxml**.

Алгоритм видобування текстів з веб-сайту:

1. Дослідити першу сторінку з текстами з метою з'ясування структури, так званої, навігаційної панелі, де наявні адреси покликання на наступні сторінки.
2. Для кожної сторінки, на яких розміщено preview-версії текстів (зазвичай, це 10-15 preview-версій на одній сторінці, залежить від конкретного сайту) зібрати покликання на сторінки з повними версіями текстів.
3. На кожній сторінці з текстом зібрати такі дані, як заголовок та текстовий контент, додати ці дані до бази даних включно з покликанням на цю веб-сторінку.

Крок 1 полягає у визначенні вигляду покликання на наступні сторінки з текстами:

```
xpath_pages = tree.xpath("//div[@class='navigation']/a")
# отримуємо покликання з панелі із номерами сторінок
pages_urls = [x.attrib['href'] for x in xpath_pages]
```

Крок 2 реалізується за допомогою таких функцій:

```
def collect_article_links(self, tree):
  # оброблення сторінок з анонсами
```

```
            web1 = tree.xpath("//div[contains(@class, 'fl-r info')]/h2/a")
            # отримуємо покликання анонсів на цій сторінці
            articles_urls = [x.attrib['href'] for x in web1]
            self.articles_links.extend(articles_urls)

        def pages_content(self, num):
            # функція для отримання даних з певної сторінки:
            # http://moviestape.net/novyny_kino/page/2/
            # http://moviestape.net/novyny_kino/page/10/
            page_url = "http://moviestape.net/novyny_kino/page/{0}/".format(num)
            page = requests.get(page_url)
            tree = html.fromstring(page.content)
            self.collect_article_links(tree)

        def download_articles(self, start_page, end_page):
            for i in range(start_page, end_page + 1):
                self.pages_content(i)  # оброблення контенту сторінок
```

Крок номер три зазначеного вище алгоритму реалізується за допомогою такої функції:

```
        def get_article(self, page_url):
    # видобування статті з анонсом новини
    ts = 5
    try:
        # звернення до веб-сайту за сторінкою
        page = requests.get(page_url)
    except requests.exceptions.ConnectionError:
        # якщо відмовлено у з'єднанні, призупиняємо роботу програми на
        # 5 секунд
        print("Connection error. Retry in {0} seconds...".format(ts))
        sleep(ts)
        page = requests.get(page_url)
        ts += 1

    tree = html.fromstring(page.content)
    # отримуємо заголовок новини
    header = tree.xpath("//h1/text()")[0]
    # отримуємо текст анонсу новини (частинами)
```

```
    article_parts = tree.xpath("//section//text()")
    # об'єднуємо частини у єдиний текст
    article = " ".join(article_parts)
    article = self.normalize_article_text(article)
    # зберігаємо до списку у форматі:
    # 1. заголовок
    # 2. вміст
    # 3. покликання на анонс
    self.articles.append([header, article, page_url])
```

Автоматичне видобування іменованих сутностей здійснено за допомогою python-бібліотеки stanza [46].

У цій бібліотеці за замовчуванням немає підтримки розпізнавання іменованих сутностей українською мовою. Але файл моделі для розпізнавання іменованих сутностей українською мовою (*Файл uk_languk_nertagger.pt*) можна завантажити за посиланням під номером 48 у списку використаних джерел.

Процес видобування іменованих сутностей за допомогою бібліотеки stanza можна здійснити як за допомогою центрального процесора комп’ютера, так і за допомогою графічного процесора.

Встановлення бібліотеки stanza рекомендується здійснювати всередині віртуального середовища.

Етапи встановлення бібліотеки stanza для встановлення на локальному комп’ютері без підтримки графічного процесора:

Створити віртуальне середовище розроблення. У командному рядку виконати команду: python -m venv my_project_dictionary_venv

Активувати віртуальне середовище за допомогою команди: my_project_dictionary_venv\Scripts\activate.bat

Встановити бібліотеку stanza за допомогою команди: pip install stanza==1.1.1

У даній дипломній роботі не розглядається використання бібліотеки stanza на локальному комп’ютері з підтримкою графічного процесора.

Натомість, код, призначений для видобування іменованих сутностей, було запущено на хмарному сервісі Google Colaboratory з підтримкою графічного процесора.

Було проведено два експерименти видобування іменованих сутностей з ідентичним корпусом текстів – один на центральному процесорі локального комп’ютера, інший – на графічному процесорі в Google Colaboratory. У першому експерименті видобування тривало близько 39 годин, тоді як у другому експерименті видобування було здійснено за 97 хвилин, що у 24 рази швидше порівняно з першим експериментом.

Збереження розпізнаних іменованих сутностей здійснено в таблиці ‘named_entities’ бази даних. Ця таблиця зберігає таку інформацію: id іменованої сутності, власне сутність, тип сутності, частота сутності в корпусі текстів. Також у таблиці ‘corpora_texts’ введено поле ner_positions, що зберігає індекс іменованих сутностей для кожного тексту у такому форматі [id_сутності_1:(позиція_1)|(позиція_2)|…|(позиція_n);id_сутності_2:…;id_сутності_n:…;]

Позиція – індекси початкового та кінцевого символів у тексті, що позначають іменовану сутність.

Відповідна функція для побудови індексу іменованих сутностей має вигляд (на вхід подається словник у форматі {id_сутності: список позицій сутності}):

```
        def parse_positions_dict(dict_of_ner_positions):
    # зміна result - результат оброблення індексу у вигляді текстового рядка
    result = "["
    # для кожного елемента у словнику
    for index_i, (entity_id, list_of_positions) in \
            enumerate(dict_of_ner_positions.items()):
        # додавання id сутності
        result += str(entity_id) + ":"
        # розглядається список позицій даної іменованої сутності
```

```
    for position_index, position in enumerate(list_of_positions):
        # додається позиція (індекс_поч.поз., індекс_кінц.поз.)
        result += "({0},{1})".format(*position)
        # коли позиція в списку не є останньою,
        # потрібно вставити роздільник |
        if position_index < len(list_of_positions) - 1:
            result += "|"
        else:
            result += ";"  # коли позиція остання в списку - роздільник ;
    result += "]"
    return result
```

Етапи розпізнавання іменованих сутностей:

1. Визначити мову тексту – якщо текст містить хоча б одну літеру кирилиці – українська, інакше англійська.
2. Розпізнавання іменованих сутностей визначеною мовою бібліотекою stanza. Отримали список розпізнаних іменованих сутностей.
3. Знайдені іменовані сутності та їх типи додаємо до словника всіх знайдених іменованих сутностей.
4. Укладаємо словник позицій кожної іменованої сутності.
5. Формуємо індекс позицій усіх іменованих сутностей даного тексту.
6. Іменовані сутності та їх типи зберігаємо до таблиці named_entities.
7. Індекс позицій іменованих сутностей зберігаємо в полі ner_positions таблиці corpora_texts.

Повний код процесу розпізнавання та оброблення іменованих сутностей має вигляд:

```
        def main_ner():
    start_time = time()
    # виведення на екран версії PyTorch, що використовується
    print("PyTorch version", torch.__version__)
    # змінна, у якій зберігається частотний словник іменованих сутностей
    ner_freq = defaultdict(int)
    # word_int_dict - словник словників: {ім.сутн.:{тип сутності:id сутності}}
    word_int_dict = defaultdict(dict)
```

```
# оголошення екземпляру класу для бази даних
dbf = DatabaseFunctions()
# виклик функції створення таблиць
# (таблиця створюється, коли її ще не створено)
dbf.create_tables()
# вміст таблиці named_entities у базі даних
entities_in_db = dbf.select_named_entities()
for record_from_db in entities_in_db:
    # наповнення даними з таблиці named_entities
    word_int_dict[record_from_db[1]][record_from_db[2]] = record_from_db[0]
    # наповнення частотного словника іменованих сутностей
    ner_freq[record_from_db[0]] = record_from_db[3]
# завантаження текстів з корпусу
all_texts = dbf.get_all_texts_from_corpus(modif=0)
# завантаження англійської моделі для розпізнавання сутностей
stanza.download('en')
# завантаження української моделі
stanza.download('uk')
# спеціальна змінна, яка ініціалізує модель для української мови
nlp_ua = stanza.Pipeline(lang='uk',  # мова
                         processors='tokenize,ner',  # перелік обробників
                         ner_model_path='uk_languk_nertagger.pt',  # модель
                         ner_forward_charlm_path="",
                         ner_backward_charlm_path="")
# ініціалізація моделі англійської мови
nlp_en = stanza.Pipeline(lang='en', processors='tokenize,ner')
# лічильник потрібний для того, щоб визначити id поточної іменованої сутн.
counter = len(entities_in_db)
le = len(all_texts)
updating_rows = []
for text_index, id_and_text in enumerate(all_texts):
    # заміна символа кінця рядка на символ пробіл
    text = id_and_text[1].replace("\n", " ")
    # виведення на екран індексу текста, що наразі обробляється
    print("{0}/{1}".format(text_index + 1, le))
    # словник зі списком індексів позицій іменованих сутностей
    positions_dict = defaultdict(list)
    # визначення мови
    language = determine_language(id_and_text[1])
    # застосування необхідної stanza-моделі відповідно до мови тексту
    if language == 'en':
        doc = nlp_en(text)
    else:
        doc = nlp_ua(text)
```

```
        for entity in doc.entities:
            try:
                # перевірка, чи дана іменована сутність з таким типом вже
                # була додана до словника
                word_int_dict[entity.text][entity.type]
            except KeyError:
                counter += 1  # збільшення лічильника на одиницю
                # додавання нової іменованої сутності до словника
                word_int_dict[entity.text][entity.type] = counter
            # видобування збереженого id збереженої іменованої сутності
            id_of_entity = word_int_dict[entity.text][entity.type]
            # додавання позицій іменованої сутності до словника індексу позицій
            positions_dict[id_of_entity].append([entity.start_char,
                                                 entity.end_char])
            ner_freq[id_of_entity] += 1  # підрахунок частоти даної ім. сутності
        # оброблення індексу позицій іменованої сутності до вигляду текстового
        # рядка
        positions_as_string = parse_positions_dict(positions_dict)
        # збереження до списку: id тексту в таблиці corpora_texts та рядок
        # індексу позицій буде збережено до поля ner_positions
        updating_rows.append((id_and_text[0], positions_as_string))
    recs = []
    # збір даних із словника word_int_dict, зведення до вигляду структури
    # таблиці named_entities
    for entity, v in word_int_dict.items():
        for entity_type, entity_id in v.items():
            recs.append((entity_id, entity, entity_type, ner_freq[entity_id]))
    dbf.drop_table("named_entities")  # видалення старої версії таблиці
    dbf.create_tables()  # таблиця named_entities створюється заново
    # збереження нових даних до таблиці named_entities
    dbf.add_new_named_entities(recs)
    # збереження індексу позиції до поля ner_positions таблиці corpora_texts
    dbf.update_corpora_with_ner_data(updating_rows)
    # оновлення ідентифікатора modif - розпізнавання іменованих сутностей
    # для даних текстів
    dbf.update_modif(old_value=0, new_value=1)
    end_time = time()  # завершення відліку часу
    # Виведення на екран інформації про час виконання
    print("Elapsed time {0} seconds".format(str(end_time - start_time)))
    print("Done")
```

Укладання частотного словника потребує коректного алгоритму для підрахунку частот словоформ:

1. Прочитати тексти корпусу з бази даних; об’єднати тексти в один великий текст та конвертувати до нижнього регістру.

2. Прочитати словоформи з бази даних.

3. Для кожної словоформи дослідити, зі скількох слів складається словоформа; розділити список словоформ на два словники: ті, що складаються з одного слова, та ті, що складаються з двох і більше слів.

4. Для кожної словоформи встановити відповідність {id словоформи: id лексеми}.

5. Для кожної словоформи, що складається з двох і більше слів:

- за допомогою регулярного виразу знайти в корпусі всі входження цієї словоформи з урахуванням нескінченної кількості пробілів між частинами словоформи, наприклад «художніх фільмів» (5 пробілів між словами).

- у корпусі замінити пробіли у знайдених входженнях на один символ нижнього підкреслення “_”.(художніх_фільмів).

6. Розбити корпус на слововживання.

7. Створити словники у форматі {id слова: кількість вживань} для словоформ та лексем.

8. Порахувати, скільки разів зустрічається кожна словоформа в корпусі, значення записати до словника частот словоформ.

9. Для кожної словоформи встановити відповідні лексеми.

10. Записати дані до бази даних.

3.2. Особливості створення програмного забезпечення для електронної лексикографічної системи

Розроблення програмного забезпечення здійснювалось мовою програмування Python у середовищі розроблення PyCharm Community Edition. Для зручності роботи та забезпечення доступності для більш широкого кола пристроїв було вирішено розробити програму у вигляді веб-додатка, розмістивши її на веб-хостингу pythonanywhere.com. Розроблення веб-додатка здійснювалось за допомогою веб-фреймворку Django.

Крім того, у програмі використано хмарні технології (Google Cloud Text-to-speech API) для підтримки озвучення термінів, їхніх тлумачень і контекстів у конкордансі.

Системою керування базами даних (СКБД) було обрано реляційну СКБД SQLite. Ця СКБД була обрана з таких причин:

- база даних зберігається в одному файлі на диску;
- простота реалізації, яка досягається за рахунок того, що перед початком виконання транзакції весь файл, що зберігає базу даних, блокується;
- простий, легкий у використанні API (Application Programming Interface — Інтерфейс прикладного програмування);
- крос-платформованість: підтримуються операційні системи: Unix (Linux і Mac OS X), OS/2, Windows (Win32 і WinCE). Легко переноситься на інші системи.

Для розроблення програми використано декілька пайтон-бібліотек, що не входять до переліку стандартних, зокрема:

- бібліотека Django – веб-фреймворк, за допомогою якого здійснювалось розроблення веб-додатка [49].
- бібліотека python-Levenshtein - використовується для визначення відстані Левенштейна [53].

- бібліотека google-cloud-text-to-speech - використовується для взаємодії пайтон-програм із сервером Google Cloud для здійснення озвучення [50].
- бібліотеки nltk [51] та stop-words [52] - використовуються для токенізації речень у процесі автоматичного реферування тексту.

Пошук потрібного кінотерміна в реєстрі здійснюється введенням до спеціально відведеного поля на веб-сторінці. Відповідно, можливі такі сценарії:

- Користувач увів текст, що є терміном, присутнім у базі даних.
- Користувач увів текст, що не є терміном, який присутній у базі.
- Користувач залишив поле пошуку порожнім.

За другим сценарієм випадку користувачу буде запропоновано вибрати з-поміж п’яти термінів із бази, що графічно найбільш близькі до тексту, введеного користувачем. Для цього використовується відстань Левенштейна - для кожного терміна бази даних обчислюється відстань до введеного користувачем тексту. Користувачеві пропонується обрати серед п’яти термінів з найбільшою відстанню Левенштейна (*див. додаток 1*).

Для озвучення кінотермінів, тлумачень та контекстів використано платформу хмарних технологій Google Cloud. Озвучення здійснено за допомогою Google Cloud Text-to-speech API. Для цього потрібно здійснити таку послідовність дій:

1. Створити проєкт на платформі GoogleCloud.

2. Увімкнути Text-to-speech API.

3. Створити сервісний аккаунт (це зроблено для того, щоб використовувати Google Cloud на локальному комп’ютері під час розроблення програми та на веб-сервері після публікації сайту на веб-хостингу). У результаті отримали json-файл з конфігурацією облікового запису GoogleCloud. Цей файл необхідно зберегти в теку з проєктом програми.

4. Налаштувати роботу GoogleCloud на локальному комп’ютері.

Для налаштування на локальному комп’ютері потрібно:

1. Відкрити файл з налаштуваннями сайту: mysite/settings.py

2. У змінну CREDENTIALS_FILENAME записати ім’я збереженого json-файлу.

3. У змінну ENABLE_TTS_AUDIO записати значення True.

Функція, яка здійснює синтез мовлення має такий вигляд (*див. додаток 2*).

Озвучення здійснюється за таким алгоритмом:

- Спочатку ініціалізується клієнт, що передасть на сервер необхідну для озвучення інформацію.
- Далі вказується текст, який необхідно озвучити.
- Здійснюється налаштування параметрів синтезатора – вказується мова для озвучення, а також голос, яким необхідно озвучити текст.
- Вказується формат аудіофайлу, до якого буде збережено озвучений текст.
- Здійснюється передача даних на сервер.
- Відбувається збереження озвученого тексту до файлу.

Автоматичне реферування тексту здійснюється за допомогою показника TF-IDF – Term Frequency – Inverse Document Frequency та складається з таких кроків:

1. Токенізація речень.
2. Побудова частотного словника слів для кожного речення.
3. Побудова TF-матриці. Для кожного слова t у реченні обчислюється TF = частота t в реченні / загальна кількість слів в реченні.

4. Побудова таблиці, у якій визначається, у скількох реченнях зустрічається слово t.

5. Побудова IDF-матриці. Для кожного слова t у реченні обчислюється IDF = ln(Загальна кількість речень у тексті / Кількість речень, у яких міститься слово t).

6. Обчислення показника TF-IDF – поелементне множення елементів матриці TF та елементи матриці IDF.

7. Ранжування речень: для кожного речення обчислюється сума показників TF-IDF кожного слова в реченні.

8. Обчислюється поріг для реферування тексту: середнє значення показників TF-IDF – для тексту обчислюється сума показників TF-IDF кожного речення і сума ділиться на кількість речень.

9. До реферованого тексту потрапляють речення, для яких значення показника TF-IDF більші або рівні за значення порогу реферування.

Для здійснення токенізації речень в алгоритмі автоматичного реферування тексту потрібно встановити такі Python-бібліотеки: nltk, stop-words.

Висновки до третього розділу

Важливим етапом при конструюванні електронної лінгвістичної системи кінотермінів є проєктування структури бази даних та її наповнення відповідними даними. Сформульовано послідовні кроки наповнення бази даних відповідно до поставлених завдань при створенні словника термінів кінематографічної тематики та наведено опис програмно-технічного забезпечення для розроблення електронної лінгвістичної системи.

Зокрема, наведено детальні алгоритми завантаження текстів з Інтернет-сайтів, процесу розпізнавання іменованих сутностей, а також укладання частотного словника для текстів корпусу.

Наведена у даному розділі інформація є підґрунтям для створення інтерфейсу лінгвістичної системи у вигляді веб-застосунку (див. розділ 4).

РОЗДІЛ 4
СТВОРЕННЯ ВЕБ-ДОДАТКА ЕЛЕКТРОННОЇ ЛЕКСИКОГРАФІЧНОЇ СИСТЕМИ

Програму реалізовано у вигляді веб-застосунку з використанням веб-фреймворку Django.

Веб-додатки Django, що орієнтовані на роботу з базами даних дотримуються архітектурного шаблону Модель-Представлення-Контролер Model-View-Controller (MVC) [56]. Термін «Модель» відповідає за логіку доступу до даних, «Представлення» відповідає за ту частину програми, яка визначає, що і як показувати на веб-сторінках, «Контроллер» - визначає, яке представлення варто використати, залежно від дій користувача. У Django архітектура MVC дещо інша: за відображення даних на сторінках відповідають шаблони (Template), а за бізнес-логіку, як отримати доступ до моделей та відобразити певний шаблон, залежно від дій користувача, відповідає відображення (View). Отже, орієнтується на архітектурний шаблон Model-Template-View (MTV).

Архітектурний шаблон MTV розглянемо на прикладі автоматичного реферування введеного користувачем тексту.

Для створення Django-моделі потрібно з’ясувати, якими є властивості об’єктів у процесі автоматичного реферування. Наприклад, розглядаючи кожен термін як об’єкт, він має такі властивості: власне термін, дефініція терміна, список словоформ терміна та ін.

У Django-моделі кожен об’єкт матиме дві властивості – текст, який увів користувач (вхідний текст) та текст, отриманий в процесі автоматичного реферування (вихідний текст).

Під час створення Django-веб-додатка є файл, у якому записуються всі моделі. Для кожної моделі створюється свій клас, у якому прописані

властивості об’єкта моделі та методи – дії, які необхідно виконати із цим об’єктом – у нашому випадку отримати для вхідного тексту реферат.

```
from django.db import models  # імпорт для підтримки django-моделей
# імпорт функції, що виконує автоматичне реферування
from dict.python_prog.text_summarization import run_summarization

class Record(models.Model):
    # модель (таблиця в базі даних Django) для авт. реферування
    text = models.TextField()  # властивість оригінальний текст
    summary_text = models.TextField(null=True)  # властивість текст-резюме

    def save_record(self):
        # функція для запуску процесу реферування
        self.summary_text = run_summarization(self.text)
        # збереження отриманого тексту до бази у вигляді текстового рядка
        # списку речень
        self.save()
```

Для нашої моделі в базі даних Django буде створено таблицю dict_record (*див. рис. 4.1*), що містить три поля – поле id – позначає номер введеного тексту, поле text – текст, який був введений користувачем, summary_text – реферат тексту.

Table: dict_record — Filter in any column

	id	text	summary_text
	Filter	Filter	Filter
1	1	What is jkr? Any start-up and promising company need to have a strong investor, such as a team of jkr.co, ...	['What is jkr?', 'If all this is present in one project, then this is exactly what JKR enterprise is looking for.', 'J...
2	2	The original Skegness was situated farther east at the mouth of The Wash. Its Norse name refers to a ...	['By the 14th century, it was a locally important port for coastal trade.', 'The natural sea defences which ...
3	3	У мережі з'явився перший кадр біографічної драми « Мінамата », в якій номінант на премію « Оскар » ...	['На першому кадрі Депп з бородою і в окулярах позує в образі Вільяма Юджина Сміта з фотоапаратом...

Рис. 4.1. Фрагмент таблиці dict_record

Для того, щоб користувач мав змогу вводити текст на веб-сторінці **створюється форма** для нашої моделі:

```
from django import forms  # імпорт Django-форм

from .models import Record  # імпорт необхідного класу моделі

class RecordForm(forms.ModelForm):
    # клас форми, що прив'язана до моделі, необхідної для реферування
    class Meta:
```

```
    model = Record  # вказується, який саме клас моделі використати
    # вказується перелік полів моделі, доступних користувачу для введення
    # у веб-формі
    fields = ('text',)
```

Із файлу моделей імпортували модель Record, створили клас RecordForm, вказали, що в якості моделі використовується модель "Record", fields – вказуємо список полів (властивостей об'єкта моделі Record) доступних для введення користувачем.

Наступний крок – це створення відображення (View). Відображення відповідає за «логіку» програми – зчитується інформація з моделі та передається шаблону. У даному випадку потрібно створити два шаблони – веб-сторінка для введення користувачем тексту та веб-сторінка, яка відображатиме введений користувачем текст та реферат тексту. Відповідно, для кожного з двох шаблонів створено відображення.

Відображення прописуються у спеціальному файлі views.py

```
        # імпорт спеціальних функцій для відображень
from django.shortcuts import render, get_object_or_404, redirect

from .forms import RecordForm  # імпорт необхідної форми
from .models import Record  # імпорт необхідної моделі


def record_detail(request, pk):
    # метод-відображення сторінки перегляду результатів реферування
    # record - екземпляр класу моделі, дані отримані з рядка таблиці
    # в базі даних відповідно до вказаного параметра pk - id необхідного рядка
    record = get_object_or_404(Record, pk=pk)
    # перенаправлення на веб-сторінку перегляду результату
    return render(request, 'dict/record_detail.html', {'record': record})


def record_new(request):
    # метод-відображення сторінки для введення тексту для реферування
    if request.method == "POST":
        # користувач надіслав введений текст через форм - сторінку отримано
        # методом POST
        # екземпляр класу форми, що містить введений користувачем текст
```

```
        form = RecordForm(request.POST)
        if form.is_valid():  # якщо форма є валідною
            # екземпляр класу моделі - введений текст передано до класу моделі
            model_object = form.save(commit=False)
            # виклик методу моделі для запуску функції реферування
            # та збереження в базі
            model_object.save_record()
            # перенаправлення до сторінки перегляду результату реферування
            return redirect('record_detail', pk=model_object.pk)
    else:
        # завантажується веб-сторінка з формою для введення тексту - метод GET
        form = RecordForm()
    return render(request, 'dict/record_edit.html', {'form': form})
```

record_new – відображення для шаблону веб-сторінки, на якій міститься форма для введення тексту користувачем.

Форма на цій веб-сторінці надсилається на веб-сервер методом “POST”. Параметри форми зберігаються в тілі HTTP-запиту, тому їх не видно в адресному рядку браузера.

Отже, припустимо, користувач ввів текст та надіслав форму на веб-сервер. Після цього сторінка завантажується методом “POST”.

У змінній form – екземпляр класу RecordForm. Уведений користувачем текст отримується через request.POST.

if form.is_valid() – перевіряється, чи форма є валідною (тобто чи було введено користувачем текст; користувач не ввів текст до форми – форма не є валідною)

model_object = form.save(commit=False) – готуємо до збереження об’єкт моделі RecordForm, що наразі містить уведений користувачем текст, але поки що не зберігаємо його, оскільки потрібно ще здійснити реферування цього тексту.

model_object.save_record() – викликаємо метод save_record() – цей метод виконає реферування тексту та здійснить збереження даних у базі даних Django.

return redirect('record_detail', pk=model_object.pk) – після збереження даних відбувається перенаправлення на сторінку, яка відобразить введений користувачем текст та текст реферату. Це здійснюється за допомогою функції redirect, у якій вказується назва відображення, до якого потрібно перейти, а також pk – це id щойно збереженого запису таблиці Record бази даних Django.

Наступний крок – створення шаблонів веб-сторінок.

У файлі-шаблоні record_edit.html розміщується необхідна форма:

```
        <div class="content"><!--розділ контенту-->
 <h3>Новий текст / New text</h3>

<form method="post">{% csrf_token %}
      {{ form.as_p }}
  <input type="submit" id="btn_full_width" value="SAVE">
</form>

</div>
```

У файлі шаблоні record_detail.html розміщується таблиця, що складається з двох колонок – в одній оригінальний текст, у другій колонці – реферат. За допомогою відображення record_detail певний запис таблиці Record із бази даних Django передається шаблону на веб-сторінку. Доступ до полів переданого запису Record здійснюється за допомогою Django-тегів: {{record.text}} та {{record.summary.text}}

```
        <div class="content"><!--розділ контенту-->
  <table>
    <tbody>
    <tr>
      <td>{{record.text}}</td>
      <td>{{record.summary_text}}</td>
    </tr>
    </tbody>
```

```
    </table>
</div>
```

Насправді, все залежить від того, у якому вигляді зберігаються об’єкти-властивості моделі у відповідній базі даних Django. У вищезазначеному прикладі summary_text містив реферат у вигляді однієї текстової змінної. Для різноманітності відображення даних на веб-сторінці можна зберігати summary_text у базі даних Django у вигляді списку текстових рядків, де кожен текстовий рядок - це окреме речення реферату а на веб-сторінці подати результати у вигляді таблиці:

Оригінальний текст	Речення1 реферату
	Речення2 реферату
	Речення N реферату

Для забезпечення такої поведінки потрібно внести зміни до відповідного класу моделі Record у файлі Django-моделей:

```
        def record_to_html(self):
    # функція для відображення тексту реферату у вигляді html-таблиці
    # конвертація змінної із текстового рядка до python-списку
    sentences_list = eval(self.summary_text)
    len_sentences = len(sentences_list)  # кількість речень
    if len_sentences > 0:  # якщо кількість речень > 0
        html_code = """<tr><td
rowspan="{0}">{1}</td><td>{2}</td></tr>""".format(len_sentences, self.text,
sentences_list[0])
    else:
        # якщо не вдалося побудувати резюме, вивести лише оригінальний текст
        html_code = """<tr><td
rowspan="{0}">{1}</td><td>{2}</td></tr>""".format(len_sentences, self.text, "")
    for i in range(1, len(sentences_list)):
        html_code += "<tr><td>{0}</td></tr>".format(sentences_list[i])
    self.summary_text = html_code
```

Також, зміни стосуватимуться відповідного методу-відображення record_detail у файлі з відображеннями views.py

```
        def record_detail(request, pk):
    # метод-відображення сторінки перегляду результатів реферування
```

```
    # record - екземпляр класу моделі, дані отримані з рядка таблиці
    # в базі даних відповідно до вказаного параметра pk - id необхідного рядка
    record = get_object_or_404(Record, pk=pk)
    # перетворення збереженого у базі даних списку речень реферату тексту
    # до вигляду html_таблиці
    record.record_to_html()
    # перенаправлення на веб-сторінку перегляду результату
    return render(request, 'dict/record_detail.html', {'record': record})
```

Відповідно, тепер у тілі таблиці у шаблонному файлі сторінки record_detail.html достатньо розмістити лише шаблонний тег {{record.summary_text}}

```
            <table>
    <tbody>
    {{record.summary_text}}
    </tbody>
</table>
```

У файлі urls.py прописується відповідність, яке відображення слід використати при переході користувача на певну сторінку.

```
            from django.conf import settings  # імпорт модуля з налаштуваннями сайту
# імпорт для підтримки оформлення посилань на сторінки
from django.conf.urls import url
# імпорт для підтримки покликання на теку зі статичними файлами
from django.conf.urls.static import static
from . import views  # імпорт модуля з відображеннями

urlpatterns = [
    # на головній сторінці - відображення сторінки термінів
    url(r'^$', views.dictionary, name='dictionary'),
    # відображення сторінки з термінами
    url(r'^dictionary/$', views.dictionary, name='dictionary'),
    # відображення сторінки частотного словника
    url(r'^freq_dict/$', views.freq_dict, name='freq_dict'),
    # відображення сторінки частотних характеристик словоформ термінів
    url(r'^freq_wordform/$', views.freq_wordform, name='freq_wordform'),
    # відображення на сторінку перегляду джерела контексту
    url(r'^show_text/$', views.show_text, name='show_text'),
    # відображення сторінки конкорданса
    url(r'^concordance/$', views.concordance, name='concordance'),
    # відображення для сторінки введення тексту реферування
```

```
    url(r'^record/new/$', views.record_new, name='record_new'),
    # відображення сторінки перегляду результатів реферування
    url(r'^record/(?P<pk>[0-9]+)/$', views.record_detail, name='record_detail'),

    # відображення для термінів іменованих сутностей
    url(r'^dictionary_ner/$', views.dictionary_ner, name='dictionary_ner'),
    # сторінка з частотним словником ім. сутн.
    url(r'^freq_dict_ner/$', views.freq_dict_ner, name='freq_dict_ner'),
    # частотні характеристики словоформ (терміни іменованих сутн.)
    url(r'^freq_wordform_ner/$', views.freq_wordform_ner,
        name='freq_wordform_ner'),
    # перегляд джерела контексту з промаркованими іменованими сутностями
    url(r'^show_text_ner/$', views.show_text_ner, name='show_text_ner'),
    # сторінка конкордансу іменованих сутностей
    url(r'^concordance_ner/$', views.concordance_ner, name='concordance_ner'),
    # потрібно додати покликання на теку статичних файлів
] + static(settings.STATIC_URL, document_root=settings.STATIC_ROOT)
```

Отже, при переході користувача на веб-сторінку вибирається відображення, яке потрібно використати. Викликається метод відповідного відображення із файлу views.py, відображення виконується, і оброблені дані передаються до html-шаблону і веб-сторінка відображається.

Веб-сторінки для введення тексту, для якого потрібно здійснити реферування, та виведення реферату можна переглянути у додатках (*див. додатки 22-23*).

Готовий сайт інтегрованої електронної лексикографічної системи кінотермінів розміщено за покликанням http://dictengua.pythonanywhere.com/.

Варто зазначити, що у файлі веб-сторінки містяться шаблонні теги Django: {{termin}}, {{termin_def}}, {{context_}} та ін. (*див. додаток 3*)

Після оброблення python-програмою, дані будуть розміщені на веб-сторінці на місця, де вказані дані шаблонні теги.

4.1. Реалізація модулю тлумачно-перекладацького словника

У файлі 'terms_page.py' міститься клас TermsWebPage, що визначає взаємодію між Python-програмою та веб-сторінкою. (*див. додаток 4*)

При переході на сторінку термінів, виконується функція proceed(). (*див. додаток 5*)

Існує три варіанти розвитку подій:

- Перший варіант – сторінка завантажується методом get (коли користувач перейшов на сторінку тлумачного словника з іншої сторінки – поле пошуку порожнє);

- Другий варіант – після надсилання форми сторінка завантажується методом post (коли користувач натиснув кнопку «SEARCH/ПОШУК»);

- Третій варіант – після надсилання форми сторінка завантажується методом get (коли користувач натиснув на покликання «Translation/Переклад»);

Коли дані на сторінці було оброблено (знайдено відповідне тлумачення терміна, контекст), їх необхідно розмістити на веб-сторінці. Для цього використовується функція render. Розглянемо роботу функції на прикладі розміщення тлумачення на сторінці:

1. На веб-сторінці розміщено шаблонний вираз {{termin_def}}.
2. У функцію render передаємо тіло запиту (self.request), шлях до html-шаблону та словник шаблонних тегів вказавши відповідний тег як ключ словника та знайдене в базі тлумачення definition як значення:

```
render(self.request, "dict/dictionary.html",
{"termin_def": definition})
```

Таким чином, на веб-сторінці на місці, де було розміщено шаблонний тег, з’явиться необхідне тлумачення.

Розглянемо оброблення даних на веб-сторінці пайтон-програмою. На сторінці термінів розміщено кнопку SEARCH / ПОШУК, після натискання на яку здійснюється пошук уведеного користувачем терміна.

```
<form id="my_form"><!--Оголошення форми-->
{% csrf_token %}
<input formaction="/{{html_form_action}}/" formmethod="post" type="submit"
    value="SEARCH / ПОШУК">
```

Також варто звернути увагу на поле для збереження параметрів та введення терміна:

- `<input autofocus id="term" name="term"`
  `placeholder="Term... / Термін..." type="text"`
  `value="{{termin}}">`

Приховане поле, що зберігає id терміна для перекладного словника (translation_id) (наприклад, для англомовного терміна «actor» зберігається id україномовного терміна «актор»):

- `<input id="tr_id" name="tr_id" type="hidden" value="{{tran_id}}">`
- Гіперпокликання “Translation/Переклад”, що відкриє статтю відповідного терміна перекладного словника:

```
<a href="javascript:{}"
onclick="document.getElementById('my_form').submit();
return false;">Translation/Переклад</a>
```

Кнопка SEARCH/ПОШУК використовує метод post для надсилання сторінки. Це означає, що всі параметри html-форми надіслані на сервер за допомогою HTTP-заголовків. Дані, що були надіслані на сервер, користувач не може бачити в рядку адреси в браузері.

Покликання Translation/Переклад використовує метод get. Параметри надсилаються на сервер за допомогою URL-адреси, тому користувач бачить дані, що були надіслані на сервер, у рядку адреси в браузері.

Оброблення даних здійснюється за допомогою функції process_form(). *(див. додаток б).*

У випадку другого варіанту, відповідний клас Terms ініціалізується текстом, який ввів користувач до пошукового поля. Спочатку перевіряється, чи є даний текст одним із термінів, збереженим у таблиці термінів бази даних.

Якщо введений текст є терміном з бази:

· Із бази даних видобувається наступна інформація: translation_id – id відповідника даного терміну у двомовному словнику; тлумачення терміну;

· Здійснюється пошук у конкордансі (див. розділ 5.3) для видобування випадковим чином обраного контексту вживання даного терміну: визначається id даного контексту в таблиці context бази даних. На веб-сторінці розміщується підзаголовок «Context / Контекст», поруч із контекстом розміщується зображення мікрофона, до якого прив’язується покликання для аудіовідтворення даного контексту; Також розміщуються покликання Show source / Показати джерело та Show more context / Показати більше контекстів;

· На веб-сторінці поруч з терміном розміщується зображення мікрофона для відтворення аудіо терміну;

· На веб-сторінці розміщується тлумачення терміну та зображення мікрофона для відтворення аудіо тлумачення;

Якщо не терміном:

· У випадку введеного порожнього рядка на веб-сторінці відображається напис Terms Registry / Реєстр термінів та виводиться список усіх термінів, наявних в таблиці термінів бази даних включно з покликаннями на них;

· Якщо поле пошуку непорожнє – здійснюється пошук п’яти графічно найбільш близьких термінів до введеного користувачем тексту за

допомогою відстані Левенштейна; виводиться список із даними запропонованими термінами включно з покликаннями на них;

- На веб-сторінку виводиться текст, введений користувачем, причому зображення мікрофона поруч з даним текстом не розміщується.

У випадку третього варіанту, відповідний клас Terms ініціалізується id даного терміну, Потім, за даним id із таблиці термінів видобувається власне термін, і далі все відбувається за вищенаведеним сценарієм, коли термін наявний в базі даних.

4.2. Реалізація модулів частотного словника та конкордансу

Дизайн сторінки частотного словника складається з таких елементів:

- поле для фільтрації лексем за першими літерами;
- випадний список вибору поля, за яким здійснюється сортування (за частотою, за алфавітом);
- сортування термінів за алфавітом або в інверсійному порядку; впорядкування статистичних даних за ранговим списком спаду або зросту абсолютних частот;
- поля для введення обмеження лексем за частотою (вибір лексем, у яких кількість вживань перебуває в певних межах, наприклад, >5; <10; від 20 до 50);
- кнопка пошуку терміна;
- частотний словник лексем у табличному представленні.

Крім того, кожна лексема в таблиці містить покликання, натиснувши на нього, користувач переходить на сторінку, де відображено частоту вживання словоформ цієї лексеми.

Такий дизайн сторінки потребує різного відображення відповідно до ширини вікна браузера. На пристроях з маленькою роздільною здатністю екрану (на телефонах) дизайн, розроблений для великих екранів, виглядає нерепрезентативно. Тому, було вирішено внести спеціальний CSS-код, що регулює стиль відображення сторінки частотного словника відповідно до ширини вікна браузера *(див. додаток 7)*.

Алгоритм пошуку контекстів схожий на описаний вище алгоритм пошуку в тлумачному словнику. Якщо лексему знайдено в базі, буде виведено контексти її словоформ, інакше користувачеві пропонується вибрати лексему зі списку найбільш схожих слів.

Одразу після оброблення частотного словника відбувається укладання конкордансу. Цей процес складається з таких кроків:

1. Поділ текстів корпусу на речення.

2. Додавання речень та id текстів, у яких знаходиться аналізоване речення, до таблиці контекстів бази даних. *(див. додаток 8)*.

3. Формування зв'язків словоформа – контекст. До таблиці wordform_context_relation додаються пари [id словоформи – id контексту]. Для кожної словоформи перевіряємо, чи зустрічається вона в деякому контексті. Якщо так – додаємо відповідну пару *(див. додаток 9)*.

Відбір текстів конкордансу для заданої лексеми здійснюється за допомогою SQL-запиту *(див. додаток 10)*.

Через великий обсяг текстів у корпусах кожен термін може мати різну кількість контекстів. Розміщення великої кількості речень на одній веб-сторінці може призвести до перевантаження браузера та пристрою в цілому. Тому, для відображення контекстів на веб-сторінці використано прийом "Lazy Loading" – ледаче завантаження. Завантажується не більше 10 контекстів на одну сторінку, інші контексти користувач має змогу переглянути, скориставшись навігаційним меню переходу до іншої сторінки *(див. додаток 11)*.

Після кожного з контекстів вставляємо знак мікрофона, а також покликання «Show source / Показати джерело», натиснувши на нього користувач побачить повний текст контексту, що його цікавить.

4.3. Реалізація звукового модулю: синтез Text-to-speech

Після того, як тлумачення терміна та контекст були видобуті з бази, викликаємо функцію для озвучення та збереження аудіоданих до MP3-файлів *(див. додаток 12)*.

Після оброблення даних на сервері сторінка завантажується, щоб відобразити знайдені дані. При завантаженні сторінки також завантажуються аудіофайли.

Після завантаження сторінки, де показано знайдені тлумачення та контекст, користувач може натиснути на зображення з мікрофоном, щоб відтворити вже завантажені аудіо-файли.

Озвучення контекстів конкордансу не відбувається при завантаженні сторінки. Замість цього до зображень мікрофона біля кожного з контекстів прив'язуємо гіперпокликання на цю ж саму сторінку конкордансу, але з додатковим параметром play, що відповідає за номер контексту в конкордансі, що необхідно озвучити.

Натиснувши на зображення мікрофона, відбувається перехід за цим гіперпокликанням. Параметр play містить номер контексту, який необхідно озвучити. *(див. додаток 13)*.

Меню тлумачного словника програми працює у такий спосіб: користувач вводить термін, що його цікавить та натискає на кнопку «SEARCH/ПОШУК».

Залежно від того, що було введено до поля терміна можливі такі сценарії виконання програми: (*див. додаток 14*)

- Поле терміна залишено порожнім: у поле результату виводяться гіперпокликання усіх термінів. Коли користувач натискає на певне покликання, відбувається перенаправлення до обраного терміна.
- До поля терміна введено кінотермін, що є в базі: до поля результату виводиться тлумачення цього терміна і один із контекстів цього терміна (обирається випадковим чином).

Також на сторінці подаються зображення мікрофонів біля тлумачення, контексту та власне кінотерміна. Також відображається гіперпокликання «Translation / Переклад», що дозволяє для україномовного терміна показати відповідний йому англомовний і навпаки. Користувач може ввести термін без урахування регістру символів (наприклад, «КіНОСцЕНарІЙ»).

Якщо до поля терміна введено текст, що не є терміном із бази: користувачеві пропонується обрати з-поміж п’яти термінів, що найбільше схожі на введений ним текст. Після вибору терміна із запропонованих, відбувається відображення словникової статті (див. додаток 14).

Якщо поле фільтрації лексем на сторінці частотного словника залишити порожнім і натиснути кнопку «SEARCH/ПОШУК», у таблиці частот будуть відображені всі лексеми.

Також можливим є сортування або за частотою лексем, в алфавітному порядку, або за інвертованим алфавітом (обирається із випадного списку «Сортування за»). Якщо встановити прапорець «За зростанням» дані будуть впорядковані в алфавітному порядку (якщо за алфавітом – від А до Я, якщо за частотою – то за ранговим списком росту частот). Якщо прапорець «За зростанням» не встановлено – дані впорядковуються в оберненому алфавітному порядку.

Якщо ввести певне слово в поле пошуку, то будуть знайдені терміни, які починаються першими літерами введеного користувачем слова.

Також є можливість обмежити виведення лексем за кількістю частот. Поле «Частота з» означає, що будуть виведені лише терміни, у яких частота більша рівна за значення, уведене в поле «Частота з».

Якщо використовувати і поле «Частота з» і поле «Частота по», будуть виведені терміни, частоти яких знаходяться у вказаних межах (*див. додаток 15)*.

Якщо натиснути на певну лексему в таблиці частот, відбувається перенаправлення на сторінку з частотами вживання словоформ цієї лексеми. (*див. додаток 16*).

Сторінка з конкордансом містить поле для введення лексеми, кнопку «SEARCH/ПОШУК».

Якщо залишити поле для введення лексеми порожнім і натиснути кнопку «ПОШУК» сторінка залишиться порожньою. Якщо ввести лексему (термін), що є в базі, будуть виведені контексти словоформ цієї лексеми. А сама словоформа виділяється в контексті жовтим кольором (*див. додаток 17).*

Крім того, у кінці кожного контексту є покликання «Показати джерело», що відкриває сторінку з відповідним текстом корпусу, і виділяє контекст жовтим кольором (*див. додаток 18)*.

4.4 Реалізація інтерфейсу для реєстру іменованих сутностей

Дизайн сторінки реєстру іменованих сутностей візуально не відрізняється від дизайну сторінки частотного словника. (див. додаток 19) Однак у таблиці лексем та частот містяться покликання на модифіковану сторінку перегляду частот вживання словоформ для лексеми.

Різниця між сторінкою, на яку вказує покликання у частотному словнику та сторінкою, на яку покликається реєстр іменованих сутностей, полягає в тому, що в останній, крім таблиці частот вживання словоформ, також містяться два додаткові покликання: перше – на сторінку з коротким описом іменованої сутності та обраним випадковим чином контекстом конкордансу іменованої сутності; ця сторінка подібна до сторінки реєстру кінотермінів, що дозволяє здійснювати пошук за іменованими сутностями; друге – на сторінку з контекстами конкордансу цієїіменованої сутності, відповідно, ця сторінка дозволяє здійснювати пошук контекстів у конкордансі іменованих сутностей (див. додаток 20).

На сторінці перегляду контекстів конкордансу іменованих сутностей за покликанням «Show source / Показати джерело” відкривається сторінка перегляду тексту із корпусу (див. додаток 21). При цьому, на відміну від аналогічної сторінки перегляду тексту, де здійснювалось маркування вибраного контексту, маркуються усі іменовані сутності тексту на основі індексу позицій іменованих сутностей.

Висновки до четвертого розділу

На початку роботи було поставлене завдання сконструювати інтегровану електронну лексикографічну систему кінотермінів, за допомогою якої для визначених кінотермінів та корпусів текстів можна отримати лексикографічну інформацію непритаманну для кожної підсистеми окремо.

У результаті проведеної роботи було створено систему, результатом інтеграції якої є подання користувачеві можливості, не притаманні кожній з них поодинці: дефініцію терміна, переклад, частотні характеристики термінів та їх словоформ у межах корпусу текстів, пошук контекстів лексеми, перегляд текстів корпусів та озвучення усіх елементів словникової статті (терміна, тлумачення, контексту).

Розроблена програма, яку можна знайти перейшовши за посиланням http://dictengua.pythonanywhere.com/, може бути використана як для задоволення суспільних потреб, так і з метою покращення програмних продуктів, пов’язаних з базовим аналізом текстів та укладанням контекстних словників на базі корпусу текстів.

ВИСНОВКИ

На сучасному етапі розвитку інформаційних технологій застосування комп'ютерів стає ледь не основним об'єктом для роботи в усіх сферах діяльності людини. Зростання потреб суспільства в пошуку й опрацюванні інформації зумовило активне впровадження комп'ютерних технологій і у лінгвістичних дослідженнях.

У сучасній лінгвістиці популярним є створення і використання словників та платформ для вирішення найрізноманітніших завдань: від дослідження лексики певної мови до проведення лінгвістичних експериментів.

Сьогодні лексикографія та термінографія вступили в нову фазу розвитку та активно використовують інноваційні технології в конструюванні електронних лексикографічних систем.

Стрімке поширення кінематографічного мистецтва, його інтеграційний та глобальний характер сприяли появі кінематографічної термінології. Проте українська кінематографічна термінографія лише починає своє існування. Так само як і у створенні інтегрованих лексикографічних систем. Саме ступінь розвитку сучасної термінографії у галузі кінематографа і зумовив мету цієї роботи: створити інтегровану електронну лексикографічну систему кінотермінів.

Під час процесу створення інтегрованої системи було виконано такі завдання:

1. Спроектована база даних кінематографічних термінів, яка складається з десяти таблиць, кожна з яких реляційно пов'язана між собою.

2. Створена система перевірки орфографії кінотермінів.

3. Було створене аудіовідтворення елементів словникової статті на базі платформи Google Cloud Text-to-speech API.

4. Спроектовано модуль для автоматичного реферування текстів.

5. Створено підсистему для розпізнавання іменованих сутностей.

6. Створений інтерфейс інтегрованої електронної лінгвістичної системи (на основі кінотермінів).

Для інтеграції було обрано найбільш актуальні види лексикографічних систем: термінологічний словник, перекладний, частотний словник, словник-конкорданс та була додана можливість аудіовідтворення кожного елементу словникової статті.

Розроблення програми здійснювалась мовою програмування Python у середовищі розроблення PyCharm Community Edition. Також, крім стандартних бібліотек, були підключені бібліотеки Django (фреймворк для конструювання веб-додатка), nltk (для автоматизованого реферування тексту), lxml та requests (для скачування текстів кінематографічної тематики з Інтернету), stanza (для розпізнавання іменованих сутностей), python-Levenshtein, google-cloud-text-to-speech.

У результаті проведеної роботи було створено інтегровану електронну лінгвістичну систему кінотермінів, результатом інтеграції якої є подання користувачеві можливості, не притаманні кожній з них поодинці: дефініцію кінотерміна, переклад, частотні характеристики термінів та їх словоформ у межах корпусу текстів, пошук контекстів певної лексеми, перегляд текстів корпусів та озвучення всіх елементів словникової статті (терміна, тлумачення, контексту). Також забезпечена можливість оброблення іменованих сутностей та автоматичного реферування текстів, які користувач може додавати самостійно. Отже, мети, сформульованої на початку роботи, було досягнуто.

Звісно, таку систему можна вдосконалити, додавши інші мови перекладу або інші словникові модулі. Тестування програми свідчить, що в межах поставленої задачі вдалося досягти вагомих результатів. Також така платформа буде корисною для будь якої сфери діяльності з більшим обсягом термінології.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Безклубенко С. Д. Український Енциклопедичний кінословник. Том 1.Основні терміни та поняття / С. Д. Безклубенко, О. Г. Рутковський. – К. :КНУКІМ, 2006. 280с.

2. Божко Е. С. Дослідження термінології англійської мови сфери «кіно» :постановка проблеми / Е. С. Божко // Культура народов Причерноморья. – 2009. 90 с.

3. Булик-Верхола С. З., Наконечна Г. В., Теглівець Ю. В. Основи термінознавства : навч. посіб. /. – 3-є вид., допов. – Львів : Львівська політехніка, 2016. 308 с.

4. Дарчук Н.П. Лангенбах М. О. Електронний словник мови Тараса Шевченка: методика і технології укладання / Н. Дарчук, М. Лангенбах // Українське мовознавство. - Київ, 2014. 115 с.

5. Дарчук Н. П. Комп'ютерна лінгвістика (автоматичне опрацювання тексту): підручник.— К.: Видавничо-поліграфічний центр “Київський університет”, 2008. 351 с.

6. Дарчук Н.П., Сорокін В.М., Термін в інформатиці, Посібник - Київ, 2013. 142 с.

7. Єрмоленко С. Я. Українська мова. Короткий тлумачний словник лінгвістичних термінів / С.Я. Єрмоленко, С. П. Бибик, О. Г. Тодор. – Київ : Либідь, 2001. 221 с.

8. Карпенко Ю. О. Вступ до мовознавства : підручник. — Киів-Одеса : Либідь. — 1991. 224 с.

9. Карпіловська Є. А. Вступ до комп’ютерної лінгвістики.— Донецьк: Юго-Восток, 2003. 188 с.

10. Касарес Х. Введение в современную лексикографию / Перев. с испан. – М., 1958. 355 с.

11. Комова М. В. Українська термінологічна лексикографія / М. В. Комова. – Львів, 2002. 312 с.

12. Кровицька О.В. Українська лексикографія: теорія і практика. – Львів, 2005. 175 с.

13. Лейчик В. М. Терминоведение : предмет, методы, структура / В. М. Лейчик. – 3-е изд. – М. : Изд-во ЛКИ, 2007. 256 с.

14. Лендау С.І. Словники: Мистецтво та ремесло лексикографії / З англійської переклала О. Кочерга. – К., 2012. 480 с.

15. Лепеха Т. В. Лексико-семантичні та словотвірно-структурні особливості судово-медичної термінології: дис… канд. філол. наук: 10.02.01 / Лепеха Таїсія Василівна. – Д., 2000. 201 с.

16. Лотте Д. С. Как работать над терминологией. Основы и методы: [пособие сост. по трудам Д. С. Лотте и Ком. науч.-техн. терминол. АН СССР] / Д. С. Лотте. – М.: Наука, 1968. 76 с.

17. Мирославський В. Н. Кінословник. Терміни, визначення, жаргонізми В. Н. Мирославський. – Харків : Фоліо, 2007. 328 с.

18. Панько Т. І., Кочан І. М., Мацюк Г. П. Українське термінознавство: підручник . – Львів : Світ, 1994. 216 с.

19. Перебийніс В.І., Сорокін В.М. Традиційна та комп'ютерна лексикографія: Навчальний посібник. Вид: Київ 2009 - Видавничий центр КНЛУ. 218 с.

20. Полюга Л. М. Складові елементи словників: із досвіду лексикографа / Л. М. Полюга // Записки з українського мовознавства. Зб. наук. праць — Одеса, 2006. 307с.

21. Роміцин А. А. Українське радянське кіномистецтво 1941–1954 : Нариси/ А. А. Роміцин. – К. : Вид-во АН УРСР, 1959. 229 с.

22. Рыбин С.В. Синтез речи Учебное пособие. Вид: Санкт-Петербург 2014. 92 с.

23. Ступак І. В. Кінематографічна термінологія французької та української мов: особливості розвитку/ І. В. Ступак, Н. М. Лоскутова // Актуальні проблеми романо-германської філології та прикладної лінгвістики : наук. журнал / редкол. В. І. Кушнерик та ін. – Чернівці : РОДОВІД, 2016. 2016 с.

24. Ступин Л. П. Лексикография английского языка: Учеб. пособие для студентов ин- тов и фак. иностр. яз. / Леонид Павлович Ступин. – М.: Высшая школа, 1985. 167 с.

25. Суперанская А.В., Подольская Н.В., Васильева Н.В. Общая терминология: Вопросы теории. Вид: Москва, 2014. 243 с.

26. Томіленко Л. М. 'Термінологічна лексика в сучасній тлумачній лексикографії української літературної мови', Монографія Івано-Франківськ Фоліант 2015. 160 с.

27. Черницький В.Б. Комп'ютерна лексикографія: [Навч. посібник] – Нац. ун-т кораблебудування ім. адм. Макарова. – Миколаїв: НУК, 2004/ 180 с/

28. Широков В. А. Комп'ютерна лексикографія / В. А. Широков. – К.: Наук. думка, 2011. 351 с.

29. Широков В. А., Бугаков О. В., Грязнухіна Т. О. та ін. Корпусна лінгвістика / – К.: Довіра, 2005. 472 с.

30. Широков, В. А. Технологічні основи сучасної тлумачної лексикографії / В. А. Широков, О. Г. Рабулець [та ін.] . Мовознавство. - 2002.- № 6. 48 с.

31. Magee C. Computational Lexicography B.A. (Mod.) CSLL Final Year Project, May 2000, Supervisor: Dr. Carl Vogel. 116 p.

СПИСОК ЕЛЕКТРОННИХ РЕСУРСІВ

32. Академічний тлумачний словник української мови. URL: http://sum.in.ua/

33. Глосарій Термінів Кіно. URL: https://www.filmsite.org/

34. Електронний словник Макмілана. URL: https://www.macmillandictionary.com

35. Електронний Оксфордський словник. URL: https://en.oxforddictionaries.com

36. Єрмолова Я.А. Частотні словники та їх використання.

URL: http://ekhnuir.univer.kharkov.ua/bitstream/123456789/5991/2/Ermolova.pdf

37. Синтезатор мовлення “Infovox”. URL: http://www.infovox.se

38. Лекція з лексикографії.

URL: http://elib.lutsk-ntu.com.ua/book/fof/ippy/2010/10-166/page8.html

39. Словники України: словозміна, транскрипція, фразеологія, синонімія, антонімія / В. А. Широков, та ін. — К.: Довіра, Український мовно-інформаційний фонд, 2001-2010. URL: http://lcorp.ulif.org.ua/dictua

40. Український мовно-інформаційний фонд національної академії наук України. URL: https://services.ulif.org.ua/expl/Entry/index?wordid=1&page=0#

41. Bell Labs Text-to-Speech Synthesis. URL: http://www.bell-labs.com/project/tts

42. Cinemablend. URL: https://www.cinemablend.com

43. Kino/Teatr/Кіно-Театр. URL: https://kino-teatr.ua/

44. KinoFilms. URL: https://www.kinofilms.ua/

45. MoviesTape. URL: http://moviestape.net

46. Named Entity Recognition - Stanza.

URL: https://stanfordnlp.github.io/stanza/ner.html

47. Quickstart: Using the client libraries | Cloud Text-to-Speech API | Google Cloud. URL: https://cloud.google.com/text-to-speech/docs/quickstart-client-libraries#client-libraries-usage-python

48. Release Stanza NER. URL: https://github.com/gawy

49. Release Django. URL: https://www.djangoproject.com/

50. Release google-cloud-texttospeech. URL: https://pypi.org/project/google-cloud-texttospeech/

51. Release Natural Language Toolkit — NLTK 3.6.2 documentation. URL: https://www.nltk.org/

52. Release Stop Words. URL: https://pypi.org/project/stop-words/

53. Release python-Levenshtein. URL: https://pypi.org/project/python-Levenshtein/

54. Release Processing XML and HTML with Python. URL: https://lxml.de/

55. Release requests. URL: https://pypi.org/project/requests/

56. DjangoBook по-русски. URL: https://djbook.ru/ch05s02.html

ДОДАТКИ

Додаток 1.

Визначення пропозицій термінів за допомогою відстані Левенштейна

```
    def if_no_term_in_base(self):
        dbf = DatabaseFunctions()
        sc = SpellChecker()  # клас, що відповідає за підбір найбільш
        # схожих термінів, якщо термін введено з помилкою
        term_to_search = self.word.lower()
        if self.ner_mode:
            terms_table = "terms_ner"
        else:
            terms_table = "terms"
        all_terms_and_ids = dbf.get_all_terms_and_their_ids(terms_table)
        if not self.term_exist:
            # якщо поле пошуку порожнє, буде виведено список усіх термінів
            if term_to_search == "":
                candidates = all_terms_and_ids
            else:
                # відсортуємо всі терміни бази за коефіцієнтом відстані
                # Левенштейна для введеного користувачем терміна
                all_terms_and_ids.sort(key=lambda x: sc.score(x[1],
                                                              term_to_search),
                                       reverse=True)
                # обмеження п'ятьма найбільш схожими термінами
                candidates = all_terms_and_ids[:5]
            return self.show_suggestions(candidates)
        return ""
```

```
import Levenshtein


class SpellChecker:
    @staticmethod
    def score(original, tested):
        d = Levenshtein.distance(original, tested)
        return max(0.0, 1.0 - 1.0 * d / len(original))
```

Додаток 2.

Озвучення тексту за допомогою Google Cloud

```
def convert_text_to_speech(text, output_file):
    # Instantiates a client
    client = texttospeech.TextToSpeechClient()

    # Set the text input to be synthesized
    synthesis_input = texttospeech.types.SynthesisInput(text=text)

    # Build the voice request, select the language code and the ssml
    # voice gender ("neutral")
    language = determine_language(text)
    voice = texttospeech.types.VoiceSelectionParams(
        language_code=language,
        ssml_gender=texttospeech.enums.SsmlVoiceGender.NEUTRAL)

    # Select the type of audio file you want returned
    audio_config = texttospeech.types.AudioConfig(
        audio_encoding=texttospeech.enums.AudioEncoding.MP3)

    # Perform the text-to-speech request on the text input with the selected
    # voice parameters and audio file type
    response = client.synthesize_speech(synthesis_input, voice, audio_config)

    # The response's audio_content is binary.
    with open(output_file, 'wb') as out:
        # Write the response to the output file.
        out.write(response.audio_content)
        print('Audio content written to file "{}"'.format(output_file))
```

Додаток 3.

Код для веб сторінки термінів

```
<link rel="stylesheet" href="/static/css/style.css">
</head>
<body>
<!--Панель меню-->
<ul class="sidenav">
    <li><a class="active" href="/dictionary">Dictionary / Словник</a></li>
    <li><a href="/freq_dict">Freq dict / Частотний словник</a></li>
    <li><a href="/concordance">Concordance / Конкорданс</a></li>
    <li><a href="/freq_dict_ner">NER / Реєстр ім. сутностей</a></li>
    <li><a href="/record/new">New text / Новий текст</a>
</ul>

<div class="content"><!--розділ контенту-->
    <h3>{{termin_reg}}</h3>
    <form id="my_form"><!--Оголошення форми-->
        {% csrf_token %}
        <input formaction="/{{html_form_action}}/" formmethod="post" type="submit"
               value="SEARCH / ПОШУК">
        <div style="overflow: hidden; padding-right: .5em;">
            <input autofocus id="term" name="term"
                   placeholder="Term... / Термін..." type="text"
                   value="{{termin|escape}}">
            <input id="tr_id" name="tr_id" type="hidden" value="{{tran_id}}">
        </div>
        {{termin_and_sound}}
        <p>{{termin_def}}<img id="def_icon" onclick="audio_def.play ( )"
                              src="{{definition_src}}"></p>


        <p>{{context_}}</p>
    </form>
</div>
</body>
{% endautoescape %}
</html>
```

Додаток 4.

Взаємодія між Python-програмою та веб-сторінкою

```
# Клас оброблення сторінки термінів
class TermsWebPage:
    def __init__(self, request, ner_mode=False):
        self.request = request
        self.ner_mode = ner_mode

    def display_form(self, term="", definition="", term_and_sound="",
                     def_source="", tran_term_id="", context="", term_reg="",
                     current_term_id="", context_id=""):
        ...
        if self.ner_mode:
            html_form_action = "dictionary_ner"
        else:
            html_form_action = "dictionary"
        return render(self.request, "dict/dictionary.html",
                      {"termin": term, "termin_def": definition,
                       "termin_and_sound": term_and_sound, "context_": context,
                       "definition_src": def_source, "tran_id": tran_term_id,
                       "termin_reg": term_reg,
                       "current_term_id": current_term_id,
                       "context_id": context_id,
                       "html_form_action": html_form_action})

    def process_form(self, form, call_by_id):
        ...
```

Додаток 5.

Перехід на сторінку термінів

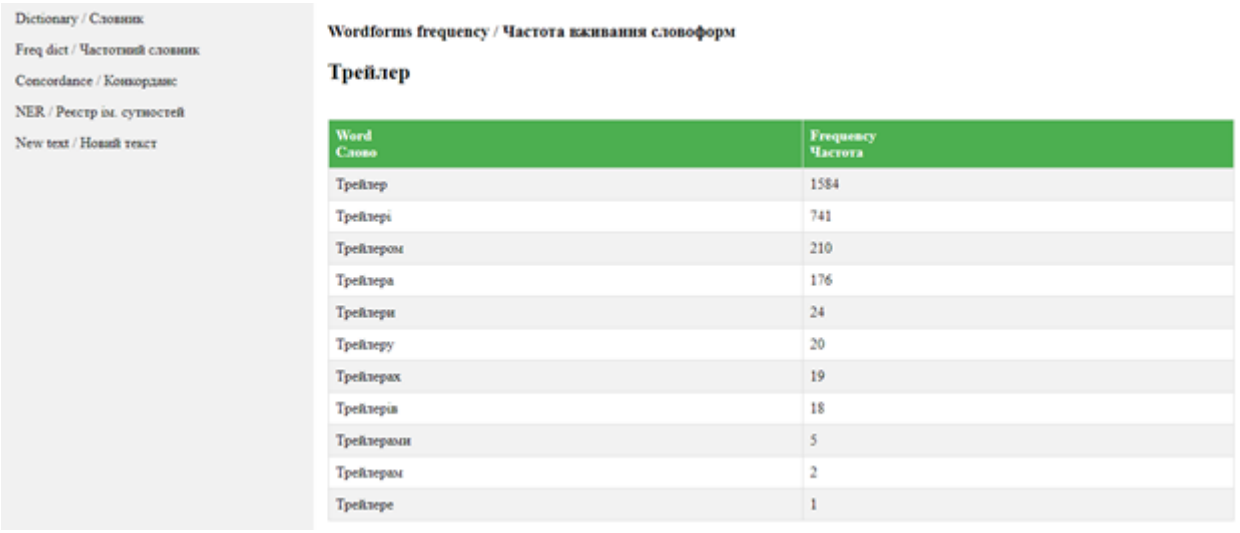
```
    def proceed(self, form):
        ...
        if self.request.method == "POST":
            return self.process_form(form, call_by_id=False)
        else:
            if not self.request.GET:
                return self.display_form()
            else:
                return self.process_form(form, call_by_id=True)
```

Додаток 6.

Оброблення та демонстрація даних на веб-сервері

```
def process_form(self, form, call_by_id):
    ...
    term = form.data.get("term", "")
    # Визначаємо translation_id (зберігається на формі у прихованому полі)
    term_id = form.data.get("tr_id", "")
    # Є два "режими" визначення потрібного терміна:
    # 1) Той термін, який ввів користувач
    # 2) Для двомовного словника потрібний термін задається за допомогою
    # id (відобразити термін з таким id)
    if call_by_id:
        # Коли визначаємо за id
        # Клас Terms - клас з пайтон-кодом для роботи з базою
        # даних (дістаємо з бази, напр. тлумачення)
        terms_class = Terms(term_id=term_id, by_id=call_by_id,
                            ner_mode=self.ner_mode)
        # Визнаємо власне термін за вказаним id цого терміна
        terms_class.get_term_by_its_id()
        # Перевірка, чи існує введений термін в базі даних
        terms_class.is_term_exist_in_base()
    else:
        # Якщо ж користувач вказав термін, ініціалізуємо клас терміном
        terms_class = Terms(term=term, by_id=call_by_id,
                            ner_mode=self.ner_mode)
        # Перевірка, чи існує введений термін в базі даних
        terms_class.is_term_exist_in_base()
```

Інтерфейс на екранах пристроїв з різною роздільною здатністю



Dictionary / Словник

Freq dict / Частотний словник

Concordance / Конкорданс

NER / Реєстр ім. сутностей

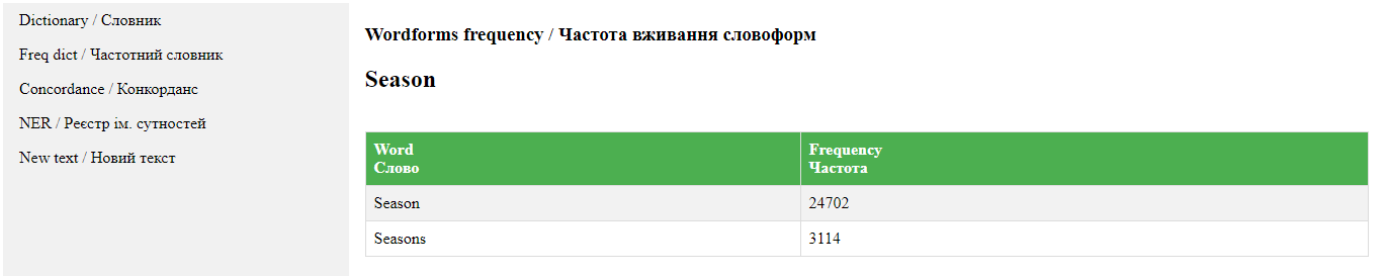
New text / Новий текст

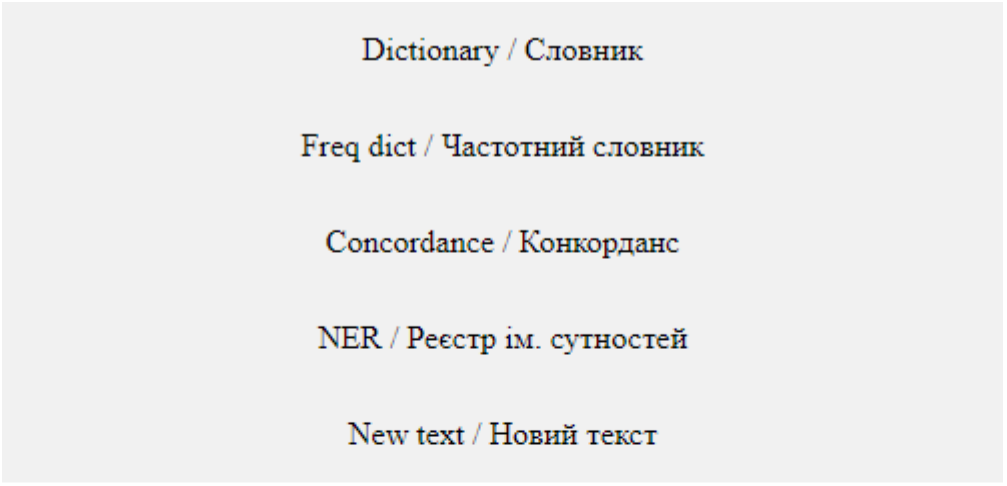
Wordforms frequency / Частота вживання словоформ

Трейлер

Word Слово	Frequency Частота
Трейлер	1584
Трейлері	741
Трейлером	210
Трейлера	176
Трейлери	24
Трейлеру	20
Трейлерах	19
Трейлерів	18
Трейлерами	5
Трейлерам	2
Трейлере	1

Інтерфейс на екранах пристроїв з різною роздільною здатністю





Wordforms frequency / Частота вживання словоформ

Season

Word Слово	Frequency Частота
Season	24702
Seasons	3114

Додаток 8.

Поділ текстів корпусу на речення та додавання речень та id текстів, у якому знаходиться речення

```
# розбиття корпусу на речення та додавання контекстів до бази
def add_concordance_contexts_to_db(self, data):
    # знайдемо контексти, які вже є в базі
    contexts_and_ids = self.dbf.get_contexts()
    if len(contexts_and_ids) > 0:
        contexts_already_in_db = list(zip(*contexts_and_ids))[1]
    else:
        contexts_already_in_db = []
    # будемо зберігати пари {речення - id тексту},
    # а потім додамо ці пари до таблиці контекстів
    contexts = []
    ts = TextSplitter()
    data_len = len(data)
    print("Splitting into sentences")
    for index, record in enumerate(data):
        print(index + 1, data_len)
        text_id = record[0]
        content = record[1]
        # розбиття на речення
        sentences = ts.split_text(content, 0)
        for sentence in sentences:
            if sentence != "":
                # контекст додається у випадку, коли його ще немає у базі
                if sentence not in contexts_already_in_db:
                    contexts.append((sentence, text_id))
    # додавання контекстів до бази
    self.dbf.add_new_contexts_to_db(contexts)
```

Додаток 9.

Формування зв'язків словоформа – контекст

```
# визначення зв'язків словоформа - контекст
def wordform_context_relation(self):
    if self.ner_mode:
        table_name = "wordform_context_relation_ner"
    else:
        table_name = "wordform_context_relation"
    # знайдемо зв'язки, які вже додано до бази
    relations_already_in_db =\
        self.dbf.get_wordform_context_relations(table_name)
    relations = []  # список для збереження зв'язків словоформа - контекст
    # з бази даних дістаємо контексти та їх id
    contexts_from_db = self.dbf.get_contexts_to_process(modif=2)
    # формуємо словник словоформ бази у форматі {словоформа: id словоформи)
    wordforms_from_db = self.generate_wordforms_dict()
    # w - словоформа, w_id - її id
    print("Wordform-context relation")
    counter = 1
    data_len = len(wordforms_from_db.items())
    for w, w_id in wordforms_from_db.items():
        print(counter, data_len)
        # регулярний вираз для виявлення словоформи у реченні
        p = "{0}{1}{0}".format(r"\b", w)
        for record in contexts_from_db:
            context_id = record[0]
            sentence = record[1]
            find = re.findall(p, sentence.lower())

            # якщо словоформу знайдено у реченні
            if len(find) > 0:
                relation_to_insert = (w_id, context_id)
                if relation_to_insert not in relations_already_in_db:
                    # додаємо зв'язок словоформи та контексту до списку
                    # relations, якщо його ще не додано до бази
                    relations.append(relation_to_insert)
        counter += 1
    # додавання зв'язків до бази
    self.dbf.add_new_wordform_context_relations_to_db(relations, table_name)
```

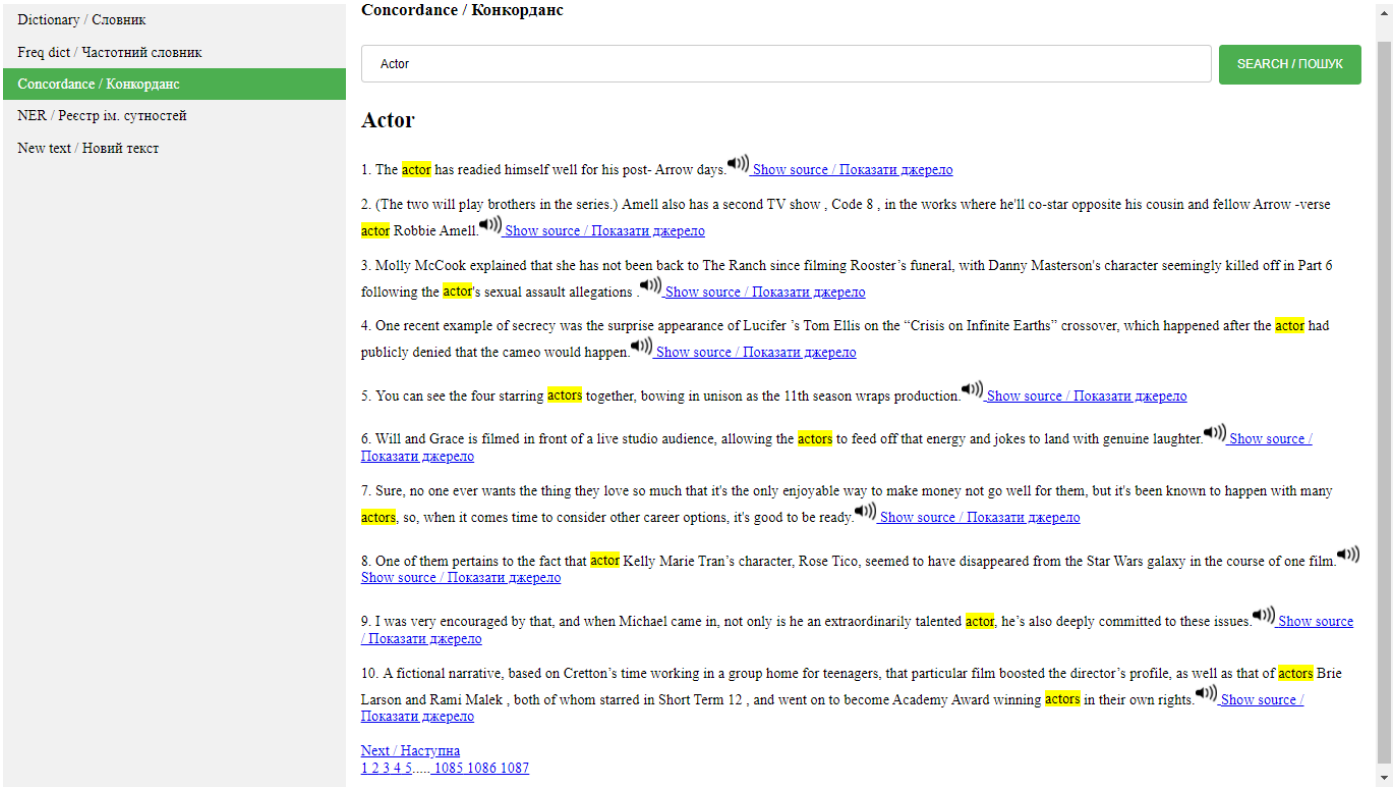
Додаток 10.

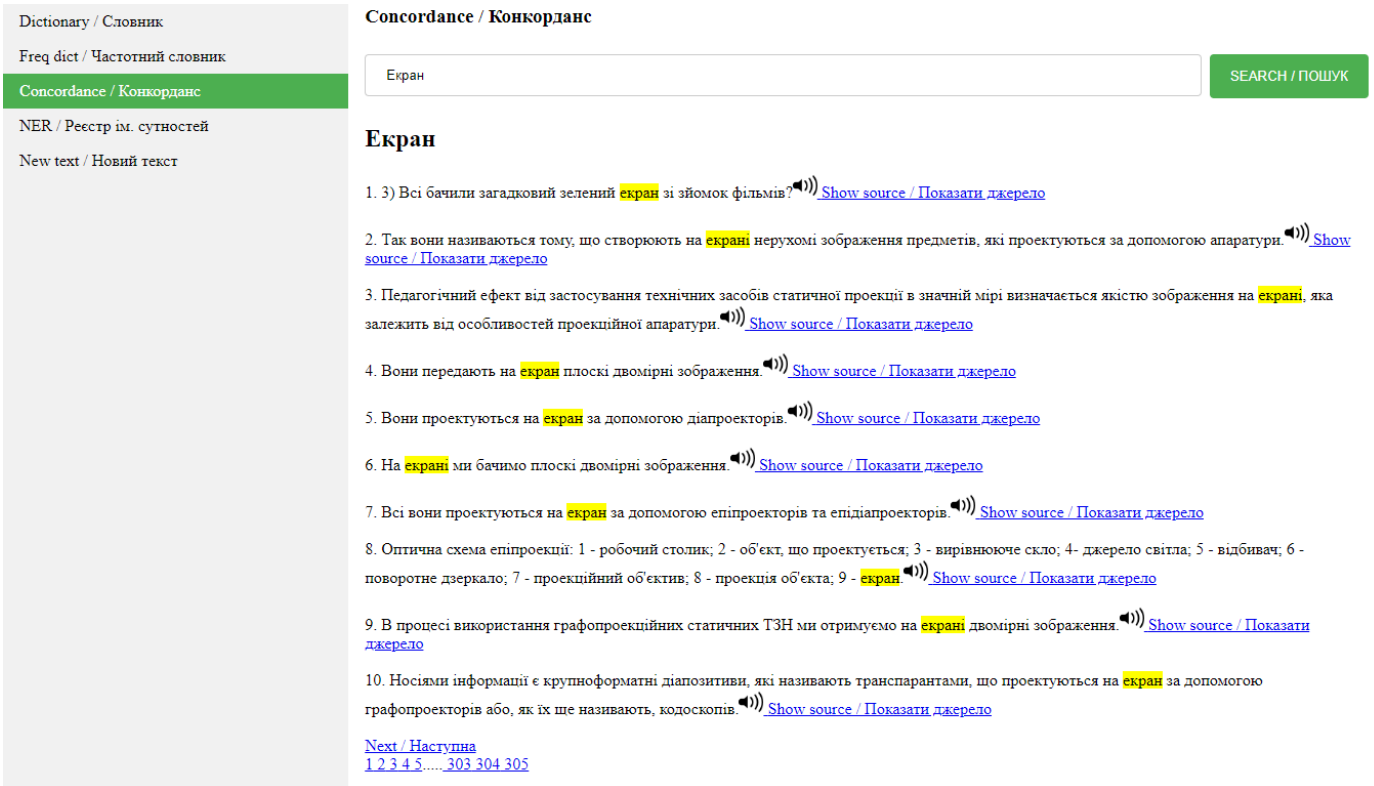
Відбір текстів конкордансу для заданої лексеми

```
# знаходження контекстів конкордансу вказаного терміну
def get_contexts_of_given_term(self, term, terms_table, wordforms_table,
                               wordform_context_relation_table,
                               last_ukr_term_num):
    sql_command = """
    SELECT f1, f2, f3 FROM
    (
        SELECT context.text_id as f1, context.sentence as f2,
        context.id as f3, corpora_texts.corpus_id,
            CASE WHEN
            ({1}.id < {5} AND corpora_texts.corpus_id IN (2,4,5,6)) OR
            ({1}.id > {4} AND corpora_texts.corpus_id IN (1,3))
            THEN 1 ELSE 0
        END AS my_field
        FROM context
        JOIN {3}
        ON context.id = {3}.context_id
        JOIN {2}
        ON {2}.id = {3}.wordform_id
        JOIN {1} ON {1}.id = {2}.term_id
        JOIN corpora_texts ON context.text_id = corpora_texts.id
        WHERE {1}.term = "{0}"
        GROUP BY {3}.context_id
        HAVING my_field = 1
    )""".format(term, terms_table, wordforms_table,
                wordform_context_relation_table, last_ukr_term_num,
                last_ukr_term_num + 1)
```

Додаток 11.

Інтерфейс контекстів





Додаток 12.

Функція для озвучення та збереження аудіоданих до MP3-файлів

```
# озвучення терміна
if term_to_display_on_page != "":
    convert_text_to_speech(term_to_display_on_page,
                           "dict/static/mp3/term_{0}.mp3".
                           format(current_term_id))
# озвучення тлумачення
if definition != "":
    convert_text_to_speech(definition,
                           "dict/static/mp3/definition_{0}.mp3".
                           format(current_term_id))
# озвучення контексту
if context_div != "":
    convert_text_to_speech(random_context,
                           "dict/static/mp3/context_{0}.mp3".
                           format(context_id))
```

Додаток 13.

Параметр play, що містить номер контексту,

який необхідно озвучити

```
if play != "":
    # дістаємо цей контекст з конкордансу
    text_to_play = concordance_class.concordance[int(play)][1]
    # озвучення контексту
    convert_text_to_speech(text_to_play,
                           "dict/static/mp3/context_{0}.mp3".
                           format(play))
    # назва javascript-функції на сторінці для відтворення аудіо при
    # завантаженні сторінки
    play_now = "play_audio()"
else:
    play_now = ""
```

Додаток 14.

Загальний інтерфейс словника термінів

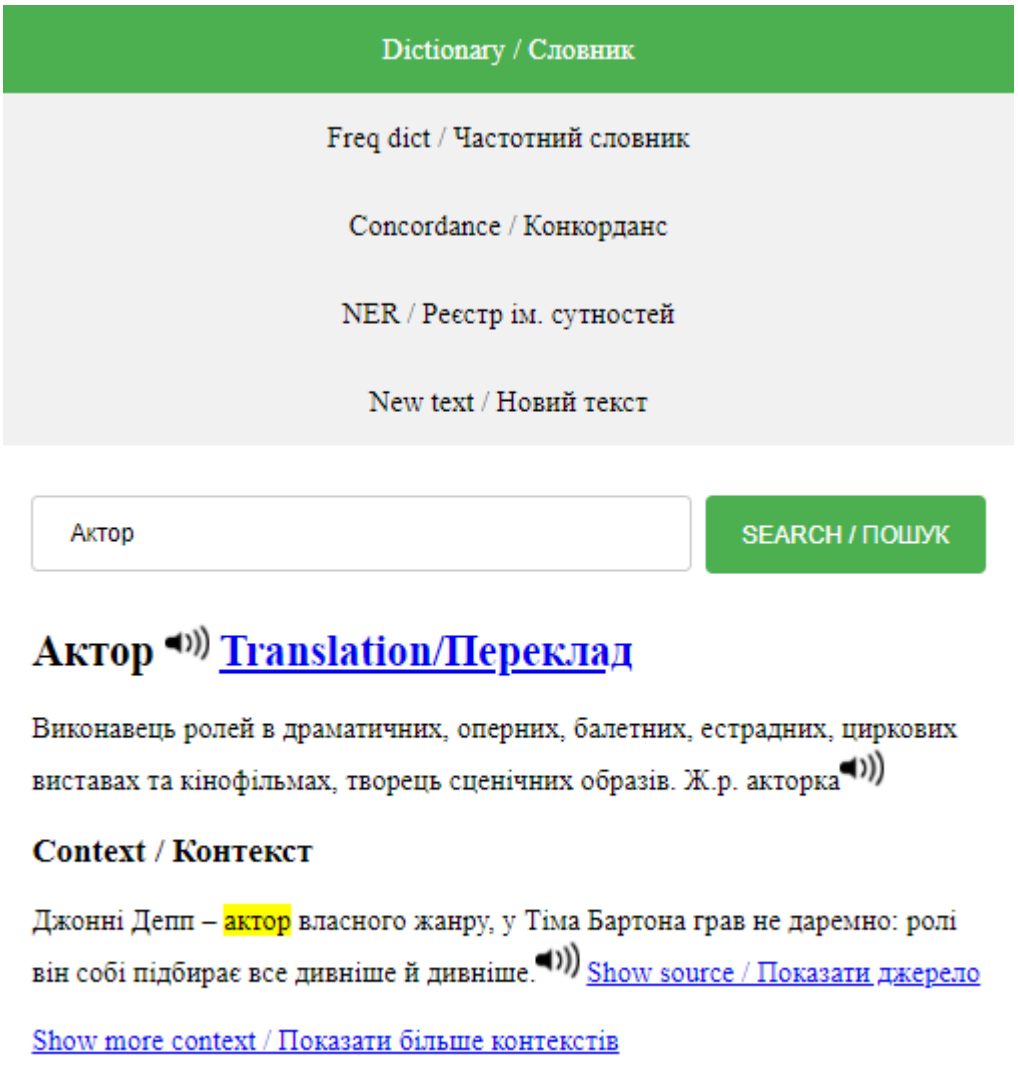
Terms Registry / Реєстр термінів

Term... / Термін...

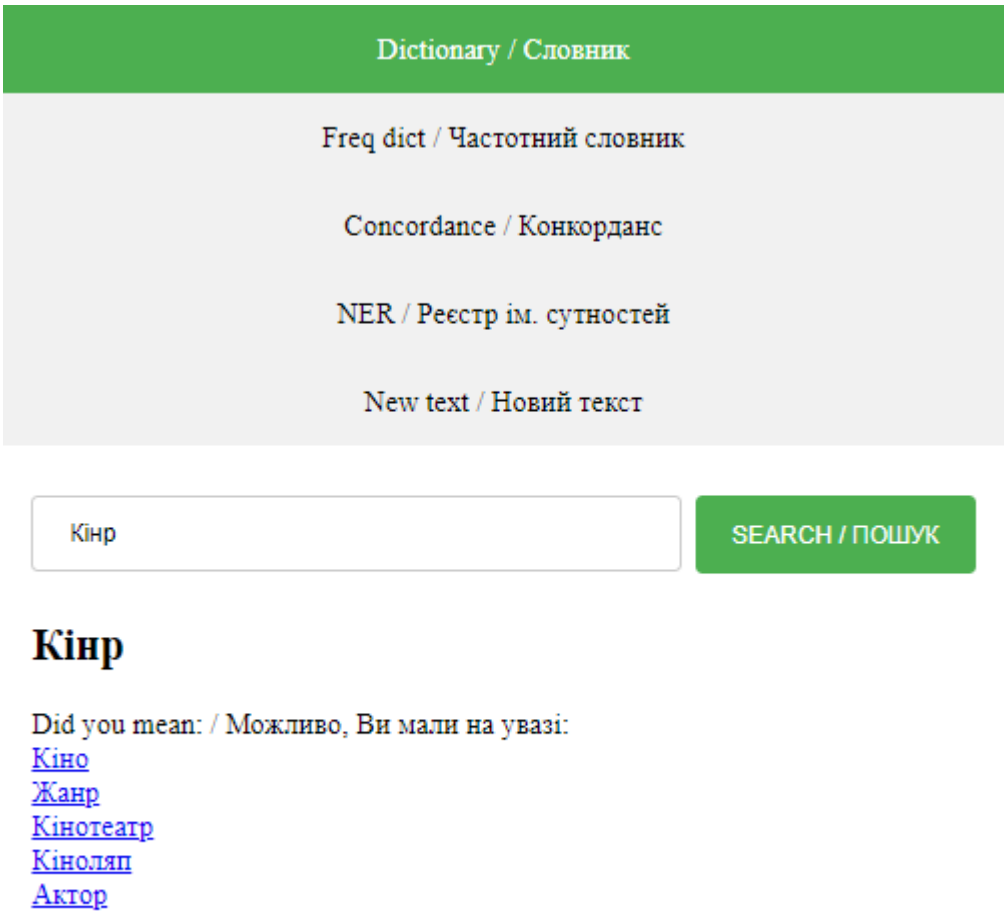
SEARCH / ПОШУК

Українські кінотерміни	Англійські кінотерміни
2д	2-d
3д	3d
Аймакс	Acting
Актор	Action Movie
Анонс	Actor
Анімація	Animation
Аніме	Anime
Арт-хаус	Art-house
Бадді-муві	B-movie
Байопік	Biopic
Блокбастер	Blockbuster
Бойовик	Body Double

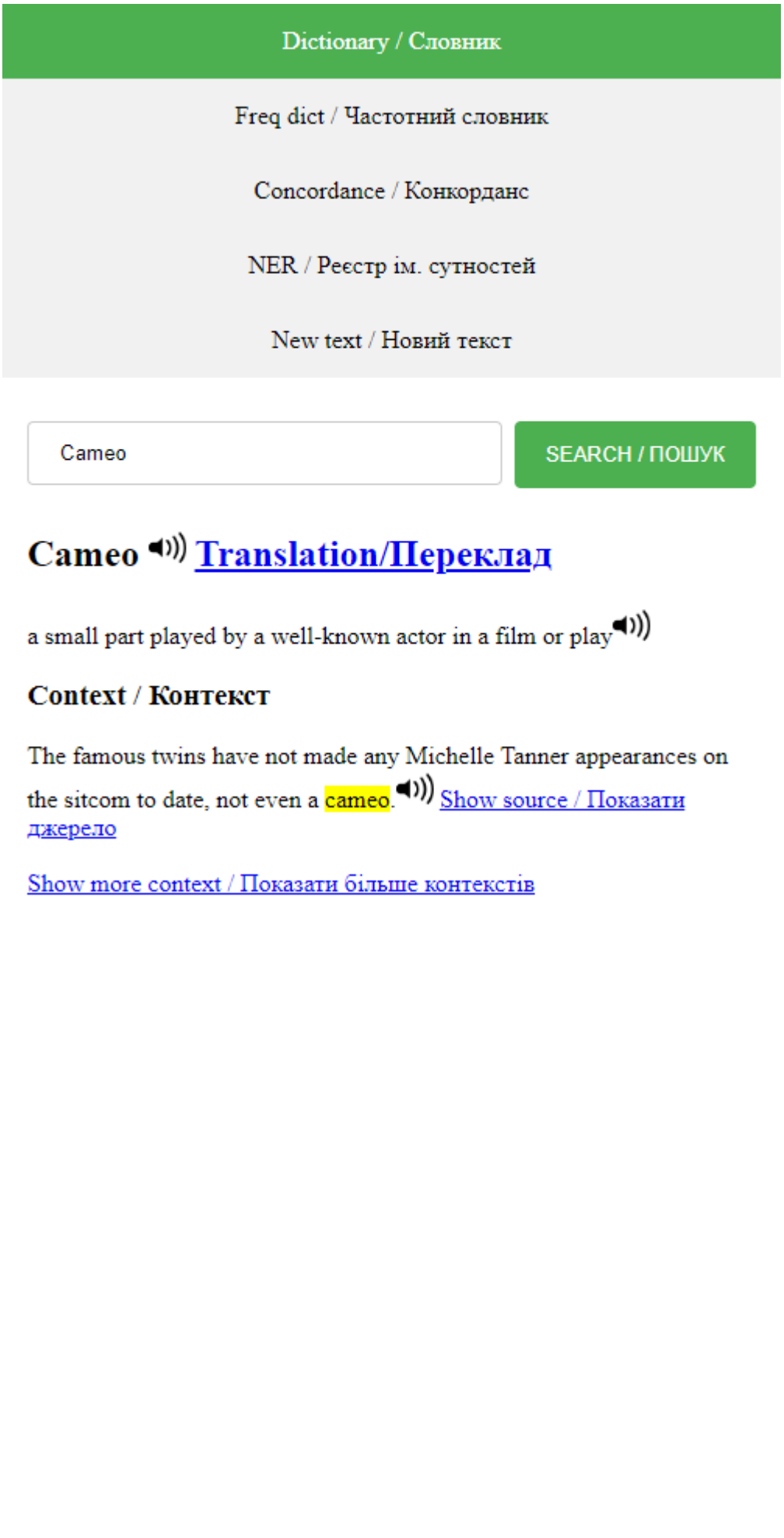
Загальний інтерфейс словника термінів



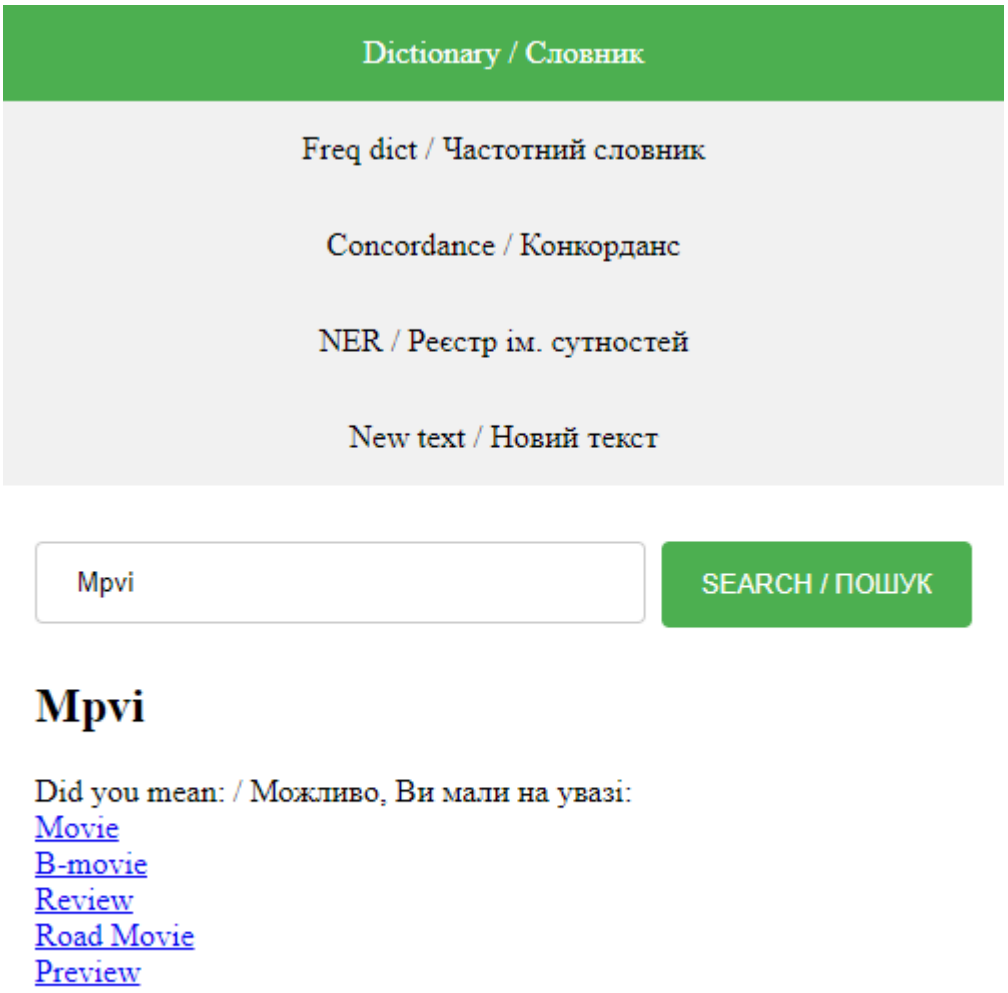
Загальний інтерфейс словника термінів



Загальний інтерфейс словника термінів



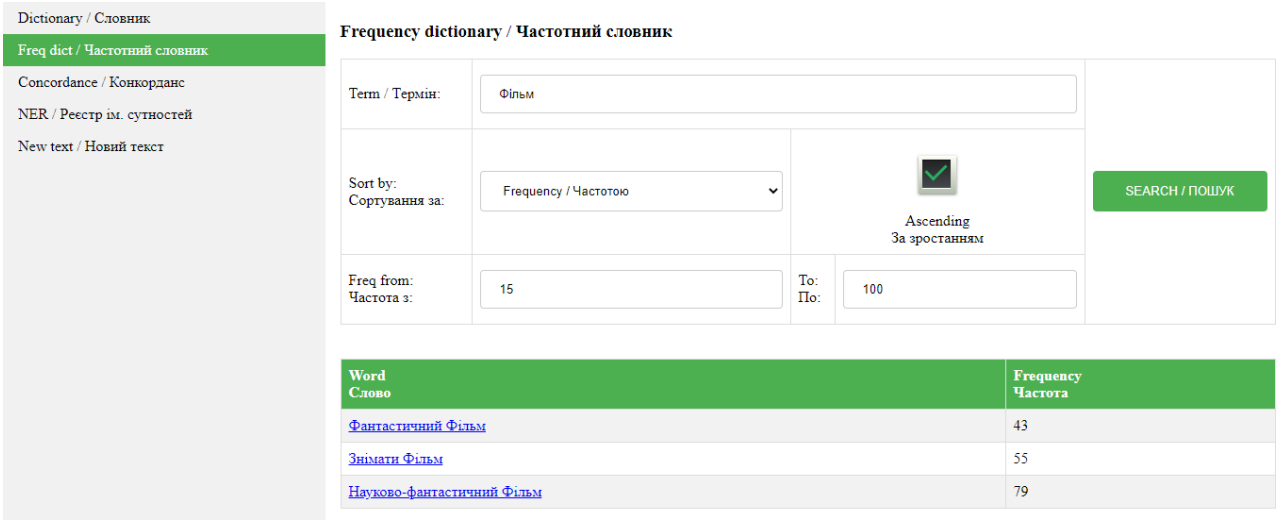
Загальний інтерфейс словника термінів



Додаток 15.

Задання параметрів пошуку кінотерміна

“фільм” у частотному словнику



Додаток 16.

Сторінка з частотами словоформ для лексеми “серія”

Dictionary / Словник

Freq dict / Частотний словник

Concordance / Конкорданс

NER / Реєстр ім. сутностей

New text / Новий текст

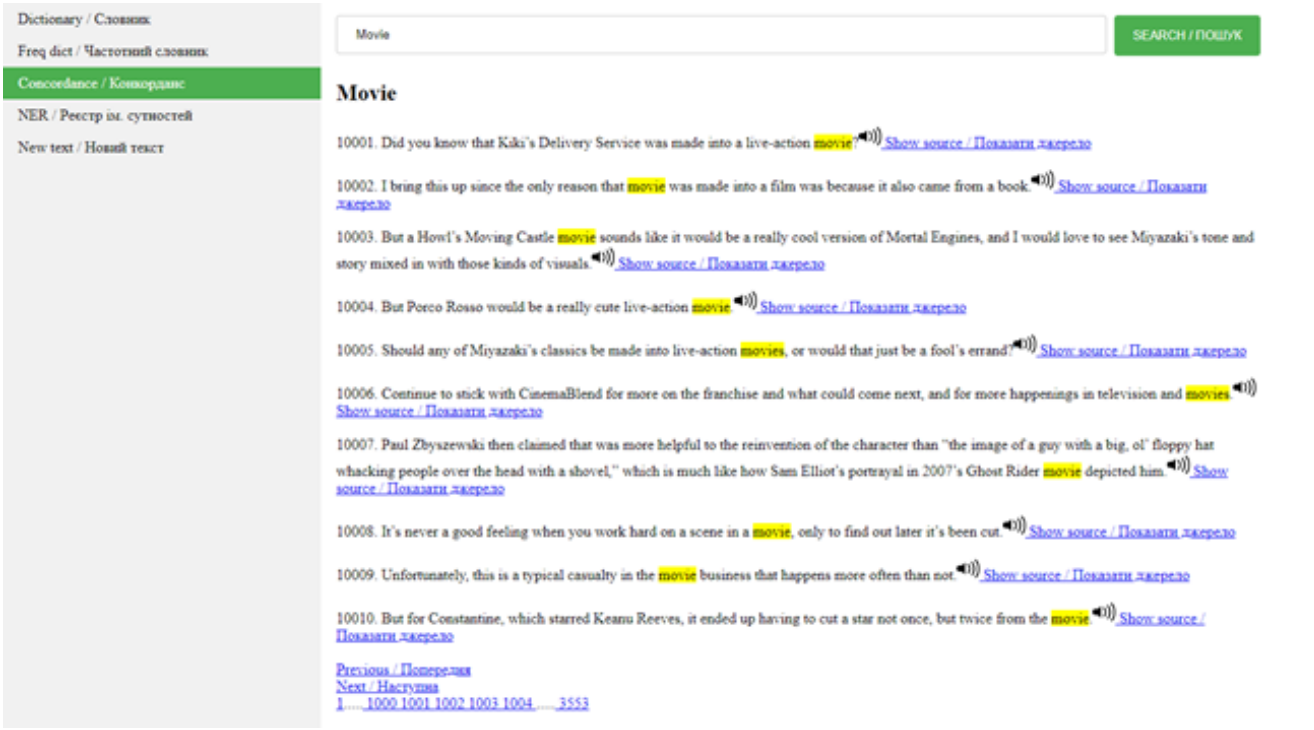
Wordforms frequency / Частота вживання словоформ

Серія

Word Слово	Frequency Частота
Серії	1505
Серія	567
Серій	344
Серіях	154
Серію	151
Серіями	52
Серією	44
Серіям	4

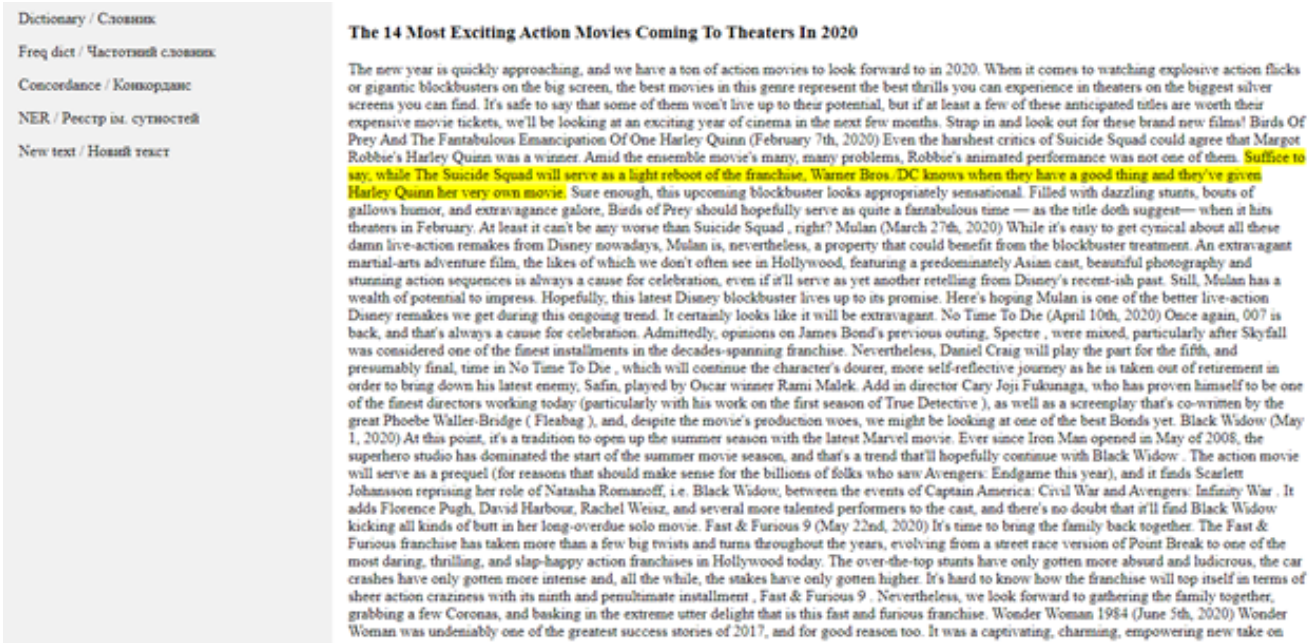
Додаток 17.

Виведення контекстів словоформ для лексеми «Movie»



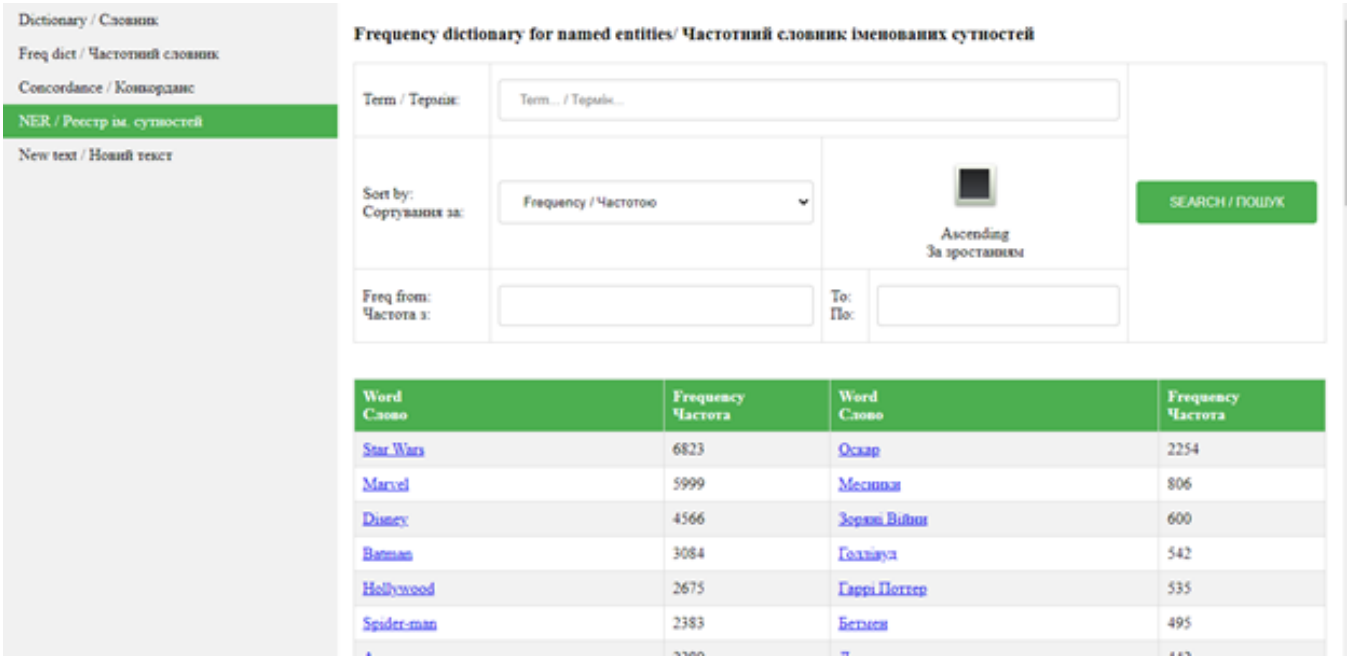
Додаток 18.

Перегляд джерела контексту



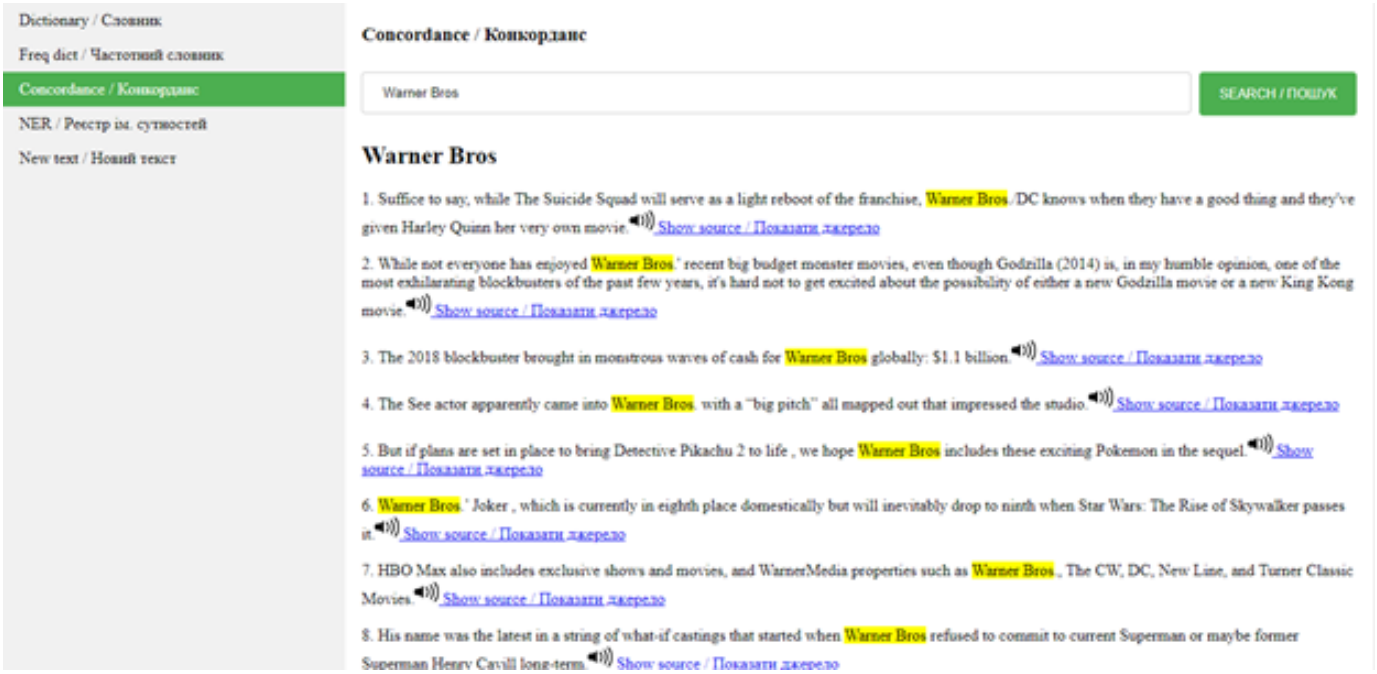
Додаток 19.

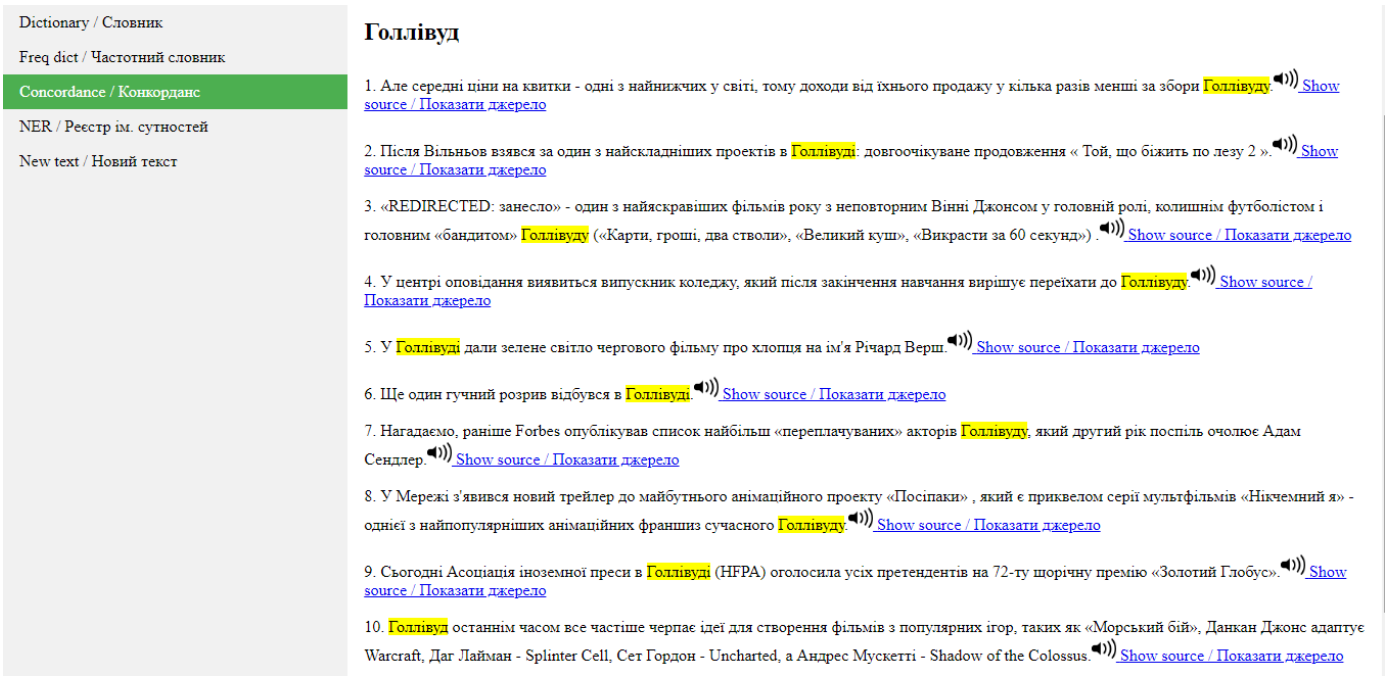
Частотний словник реєстру іменованих сутностей



Додаток 20.

Перегляд контекстів іменованої сутності





Додаток 21.

Перегляд джерела контексту з розміченими іменованими сутностями

Dictionary / Словник

Freq dict / Частотний словник

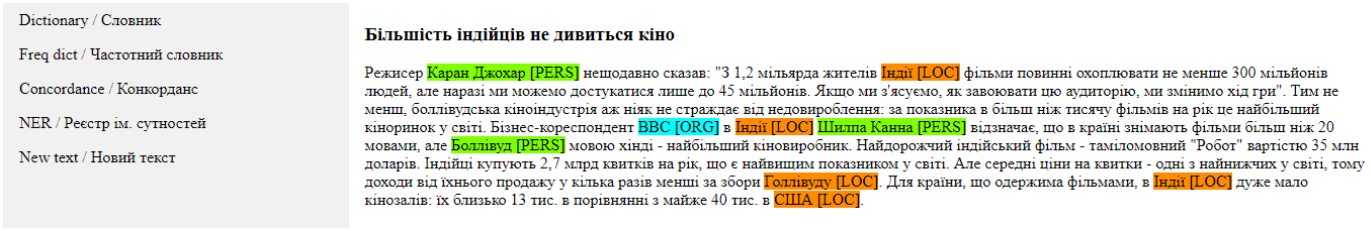
Concordance / Конкорданс

NER / Реєстр ім. сутностей

New text / Новий текст

The 14 Most Exciting Action Movies Coming To Theaters In 2020

The new year [DATE] is quickly approaching, and we have a ton of action movies to look forward to in 2020 [DATE]. When it comes to watching explosive action flicks or gigantic blockbusters on the big screen, the best movies in this genre represent the best thrills you can experience in theaters on the biggest silver screens you can find. It's safe to say that some of them won't live up to their potential, but if at least a few of these anticipated titles are worth their expensive movie tickets, we'll be looking at an exciting year [DATE] of cinema in the next few months [DATE]. Strap in and look out for these brand new films! Birds Of Prey And The Fantabulous Emancipation Of One Harley Quinn [WORK_OF_ART] (February 7th, 2020 [DATE]) Even the harshest critics of Suicide Squad [ORG] could agree that Margot Robbie's [PERSON] Harley Quinn was a winner. Amid the ensemble movie's many, many problems, Robbie's animated performance was not one [CARDINAL] of them. Suffice to say, while The Suicide Squad [ORG] will serve as a light reboot of the franchise, Warner Bros. DC [ORG] knows when they have a good thing and they've given Harley Quinn her very own movie. Sure enough, this upcoming blockbuster looks appropriately sensational. Filled with dazzling stunts, bouts of gallows humor, and extravagance galore, Birds of Prey [WORK_OF_ART] should hopefully serve as quite a fantabulous time — as the title doth suggest— when it hits theaters in February. At least it can't be any worse than Suicide Squad [ORG] , right? Mulan [PERSON] (March 27th, 2020 [DATE]) While it's easy to get cynical about all these damn live-action remakes from Disney [ORG] nowadays, Mulan [PERSON] is, nevertheless, a property that could benefit from the blockbuster treatment. An extravagant martial-arts adventure film, the likes of which we don't often see in Hollywood [GPE], featuring a predominately Asian [NORP] cast, beautiful photography and stunning action sequences is always a cause for celebration, even if it'll serve as yet another retelling from Disney [ORG]'s recent-ish past. Still, Mulan [PERSON] has a wealth of potential to impress. Hopefully, this latest Disney [ORG] blockbuster lives up to its promise. Here's hoping Mulan [PERSON] is one [CARDINAL] of the better live-action Disney [ORG] remakes we get during this ongoing trend. It certainly looks like it will be extravagant. No Time To Die [EVENT] (April 10th, 2020 [DATE]) Once again, 007 [CARDINAL] is back, and that's always a cause for celebration. Admittedly, opinions on James Bond's [PERSON] previous outing, Spectre [ORG] , were mixed, particularly after Skyfall [ORG] was considered one [CARDINAL] of the finest installments in the decades [DATE]-spanning franchise. Nevertheless, Daniel Craig [PERSON] will play the part for the fifth [ORDINAL], and presumably final, time in No Time To Die [EVENT] , which will continue the character's dourer, more self-reflective journey as he is taken out of retirement in order to bring down his latest enemy, Safin [PERSON], played by Oscar [PERSON] winner Rami Malek [PERSON]. Add in director Cary Joji Fukunaga [PERSON], who has proven himself to be one [CARDINAL] of the finest directors working today [DATE] (particularly with his work on the first [ORDINAL] season of True Detective [WORK_OF_ART]), as well as a screenplay that's co-written by the great Phoebe Waller-Bridge [WORK_OF_ART] (Fleabag [PERSON]), and, despite the movie's production woes, we might be looking at one [CARDINAL] of the best Bonds yet. Black Widow [ORG] (May 1, 2020 [DATE]) At this point, it's a tradition to open up the summer [DATE] season with the latest Marvel [ORG] movie. Ever since Iron Man [WORK_OF_ART] opened in May of 2008 [DATE], the superhero studio has dominated the start of the summer [DATE] movie season, and that's a trend that'll hopefully continue with Black Widow [ORG] . The action movie will serve as a prequel (for reasons that should make sense for the billions [CARDINAL] of folks who saw Avengers: Endgame [WORK_OF_ART] this year [DATE]), and it finds Scarlett Johansson [PERSON] reprising her role of Natasha Romanoff [PERSON], a.e. Black Widow [ORG], between the events of Captain America [WORK_OF_ART]: Civil War and Avengers: Infinity War [EVENT] . It adds Florence Pugh [PERSON], David Harbour [PERSON], Rachel Weisz [PERSON], and several more talented performers to the cast, and there's no doubt that it'll find Black Widow [ORG] kicking all kinds of butt in her long-overdue solo movie. Fast & Furious 9 [CARDINAL] (May 22nd, 2020 [DATE]) It's time to bring the family back together. The Fast & Furious [ORG] franchise has taken more than a few big twists and turns throughout the years [DATE], evolving from a street race version of Point Break



Dictionary / Словник

Freq dict / Частотний словник

Concordance / Конкорданс

NER / Реєстр ім. сутностей

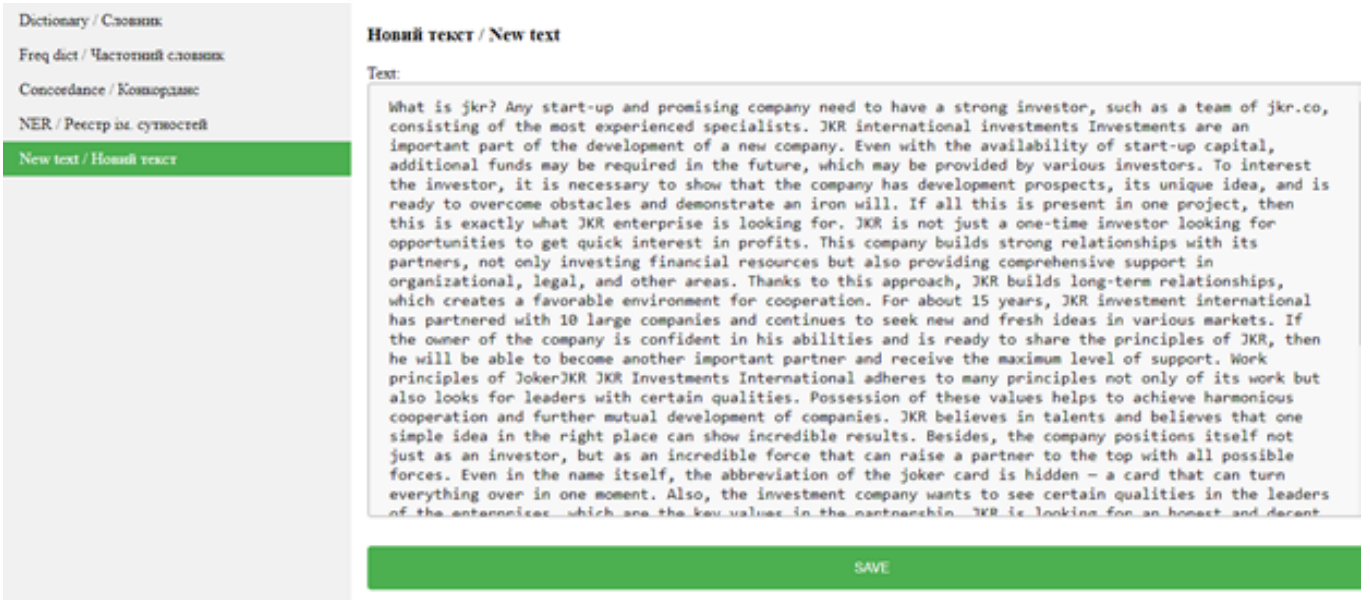
New text / Новий текст

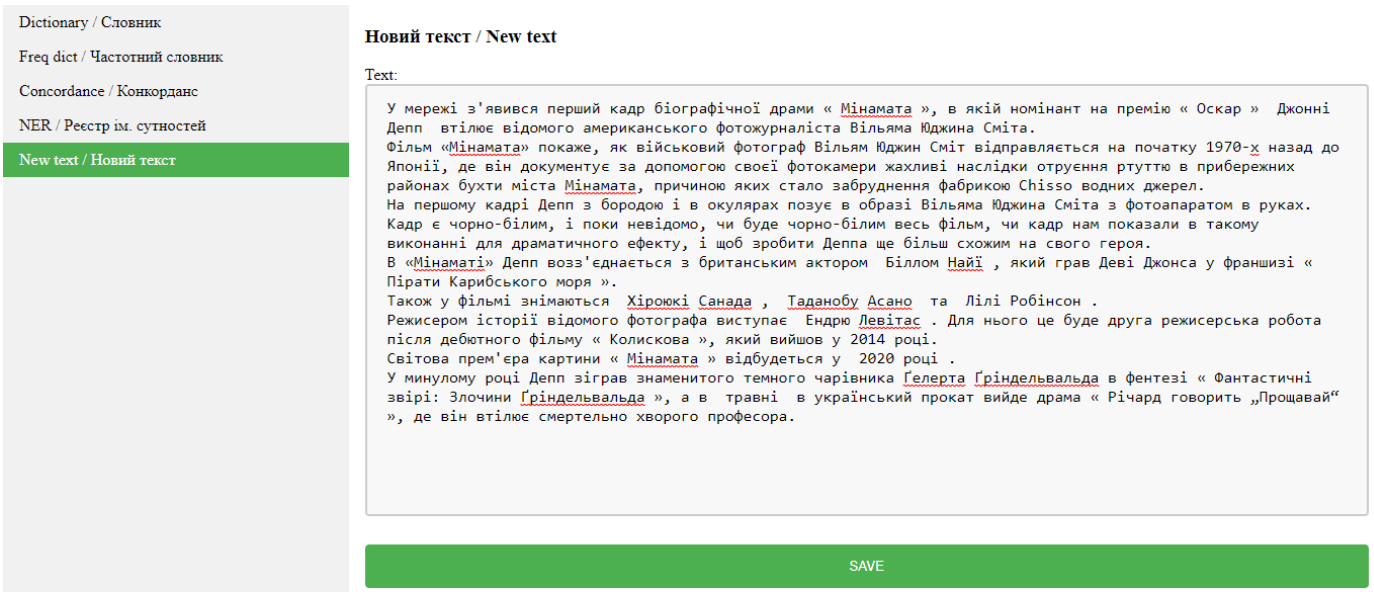
Більшість індійців не дивиться кіно

Режисер Каран Джохар [PERS] нещодавно сказав: "З 1,2 мільярда жителів Індії [LOC] фільми повинні охоплювати не менше 300 мільйонів людей, але наразі ми можемо достукатися лише до 45 мільйонів. Якщо ми з'ясуємо, як завоювати цю аудиторію, ми змінимо хід гри". Тим не менш, боллівудська кіноіндустрія аж ніяк не страждає від недовироблення: за показника в більш ніж тисячу фільмів на рік це найбільший кіноринок у світі. Бізнес-кореспондент BBC [ORG] в Індії [LOC] Шилпа Канна [PERS] відзначає, що в країні знімають фільми більш ніж 20 мовами, але Боллівуд [PERS] мовою хінді - найбільший кіновиробник. Найдорожчий індійський фільм - таміломовний "Робот" вартістю 35 млн доларів. Індійці купують 2,7 млрд квитків на рік, що є найвищим показником у світі. Але середні ціни на квитки - одні з найнижчих у світі, тому доходи від їхнього продажу у кілька разів менші за збори Голлівуду [LOC]. Для країни, що одержима фільмами, в Індії [LOC] дуже мало кінозалів: їх близько 13 тис. в порівнянні з майже 40 тис. в США [LOC].

Додаток 22.

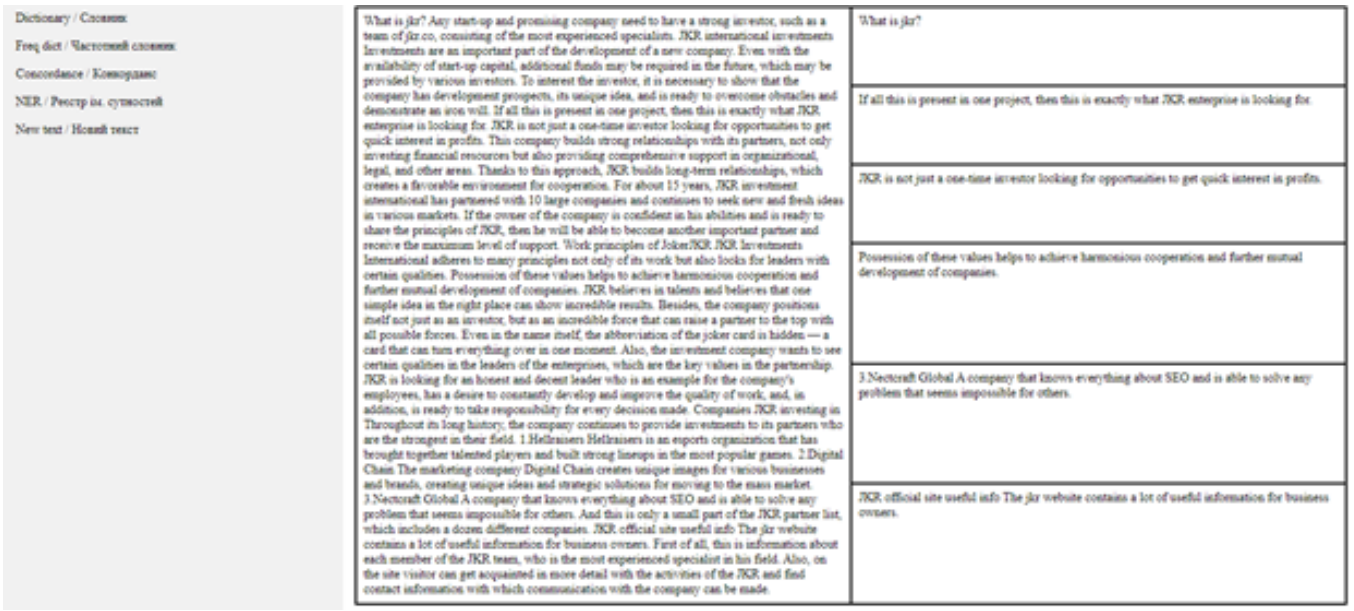
Введення тексту для автоматичного реферування





Додаток 23.

Результати процесу автоматичного реферування



Dictionary / Словник

Freq dict / Частотний словник

Concordance / Конкорданс

NER / Реєстр ім. сутностей

New text / Новий текст

What is jkr? Any start-up and promising company need to have a strong investor, such as a team of jkr.co, consisting of the most experienced specialists. JKR international investments Investments are an important part of the development of a new company. Even with the availability of start-up capital, additional funds may be required in the future, which may be provided by various investors. To interest the investor, it is necessary to show that the company has development prospects, its unique idea, and is ready to overcome obstacles and demonstrate an iron will. If all this is present in one project, then this is exactly what JKR enterprise is looking for. JKR is not just a one-time investor looking for opportunities to get quick interest in profits. Thin company builds strong relationships with its partners, not only investing financial resources but also providing comprehensive support in organizational, legal, and other areas. Thanks to this approach, JKR builds long-term relationships, which creates a favorable environment for cooperation. For about 15 years, JKR investment international has partnered with 10 large companies and continues to seek new and fresh ideas in various markets. If the owner of the company is confident in his abilities and is ready to share the principles of JKR, then he will be able to become another important partner and receive the maximum level of support. Work principles of JokerJKR JKR Investments International adheres to many principles not only of its work but also looks for leaders with certain qualities. Possession of these values helps to achieve harmonious cooperation and further mutual development of companies. JKR believes in talents and believes that one simple idea in the right place can show incredible results. Besides, the company positions itself not just as an investor, but as an incredible force that can raise a partner to the top with all possible forces. Even in the name itself, the abbreviation of the joker card is hidden — a card that can turn everything over in one moment. Also, the investment company wants to see certain qualities in the leaders of the enterprises, which are the key values in the partnership. JKR is looking for an honest and decent leader who is an example for the company's employees, has a desire to constantly develop and improve the quality of work, and, in addition, is ready to take responsibility for every decision made. Companies JKR investing in Throughout its long history, the company continues to provide investments to its partners who are the strongest in their field. 1.Hellrainers Hellrainers is an esports organization that has brought together talented players and built strong lineups in the most popular games. 2.Digital Chain The marketing company Digital Chain creates unique images for various businesses and brands, creating unique ideas and strategic solutions for moving to the mass market. 3.Nectcraft Global A company that knows everything about SEO and is able to solve any problem that seems impossible for others. And this is only a small part of the JKR partner list, which includes a dozen different companies. JKR official site useful info The jkr website contains a lot of useful information for business owners. First of all, this is information about each member of the JKR team, who is the most experienced specialist in his field. Also, on the site visitor can get acquainted in more detail with the activities of the JKR and find contact information with which communication with the company can be made.

What is jkr?

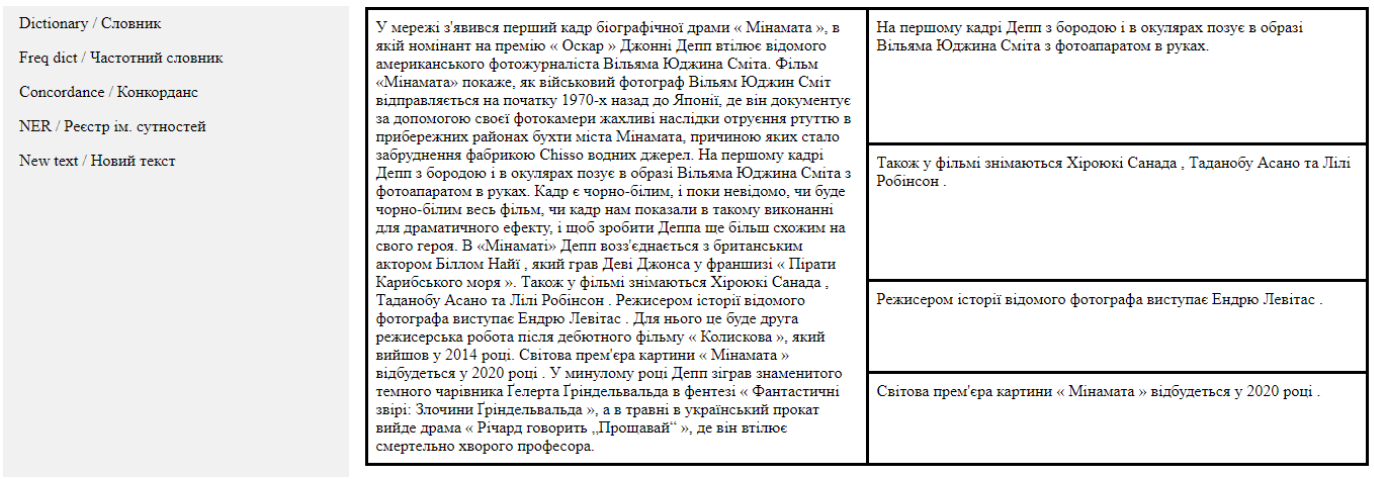
If all this is present in one project, then this is exactly what JKR enterprise is looking for.

JKR is not just a one-time investor looking for opportunities to get quick interest in profits.

Possession of these values helps to achieve harmonious cooperation and further mutual development of companies.

3.Nectcraft Global A company that knows everything about SEO and is able to solve any problem that seems impossible for others.

JKR official site useful info The jkr website contains a lot of useful information for business owners.



Dictionary / Словник

Freq dict / Частотний словник

Concordance / Конкорданс

NER / Реєстр ім. сутностей

New text / Новий текст

У мережі з'явився перший кадр біографічної драми « Мінамата », в якій номінант на премію « Оскар » Джонні Депп втілює відомого американського фотожурналіста Вільяма Юджина Сміта. Фільм «Мінамата» покаже, як військовий фотограф Вільям Юджин Сміт відправляється на початку 1970-х назад до Японії, де він документує за допомогою своєї фотокамери жахливі наслідки отруєння ртуттю в прибережних районах бухти міста Мінамата, причиною яких стало забруднення фабрикою Chisso водних джерел. На першому кадрі Депп з бородою і в окулярах позує в образі Вільяма Юджина Сміта з фотоапаратом в руках. Кадр є чорно-білим, і поки невідомо, чи буде чорно-білим весь фільм, чи кадр нам показали в такому виконанні для драматичного ефекту, і шоб зробити Деппа ще більш схожим на свого героя. В «Мінаматі» Депп возз'єднається з британським актором Біллом Найї , який грав Деві Джонса у франшизі « Пірати Карибського моря ». Також у фільмі знімаються Хіроюкі Санада , Таданобу Асано та Лілі Робінсон . Режисером історії відомого фотографа виступає Ендрю Левітас . Для нього це буде друга режисерська робота після дебютного фільму « Колискова », який вийшов у 2014 році. Світова прем'єра картини « Мінамата » відбудеться у 2020 році . У минулому році Депп зіграв знаменитого темного чарівника Ґелерта Ґріндельвальда в фентезі « Фантастичні звірі: Злочини Ґріндельвальда », а в травні в український прокат вийде драма « Річард говорить „Прощавай“ », де він втілює смертельно хворого професора.

На першому кадрі Депп з бородою і в окулярах позує в образі Вільяма Юджина Сміта з фотоапаратом в руках.

Також у фільмі знімаються Хіроюкі Санада , Таданобу Асано та Лілі Робінсон .

Режисером історії відомого фотографа виступає Ендрю Левітас .

Світова прем'єра картини « Мінамата » відбудеться у 2020 році .