МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ ФАХОВИЙ БІЗНЕС-КОЛЕДЖ
Циклова комісія (кафедра) комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

РЕПОЗИТАРІЙ ДЛЯ НАУКОВИХ РОБІТ

Виконав: студент групи 1П-21
Спеціальності
Інженерія програмного забезпечення
Максим ЧІПЕНКО
Керівник
Валентин ДМИТРЮК

Черкаси 2025

АНОТАЦІЯ

У даній кваліфікаційній роботі практичного характеру розроблено та проаналізовано веб-додаток для репозитарію наукових робіт, який забезпечує зручне завантаження, пошук, фільтрацію та модерацію матеріалів. Обґрунтовано актуальність теми через потребу у централізованому доступі до наукових ресурсів у вищих навчальних закладах. Реалізовано систему на базі фреймворку Django із підтримкою фільтрації за тегами (автори, роки, галузі, типи) та сортування результатів. Досліджено ефективність додатку через тестування ключових функцій, таких як завантаження робіт, пошук за ключовими словами та модерація контента суперкористувачами. Спроектовано інтуїтивно зрозумілий інтерфейс із використанням Bootstrap і Select2, а також забезпечено підтримку української та англійської мов. Робота включає описову частину та лістинг програмного коду.

Ключові слова: РЕПОЗИТАРІЙ, НАУКОВІ СТАТТІ, DJANGO, ФІЛЬТРАЦІЯ, МОДЕРАЦІЯ, ВЕБ-РОЗРОБКА.

ANNOTATION

This practical qualification work develops and analyzes a web application for a scientific work repository, providing convenient uploading, searching, filtering, and moderation of materials. The relevance of the topic is justified by the need for centralized access to scientific resources in higher education institutions. The system is implemented using the Django framework, featuring tag-based filtering (authors, years, fields, types) and result sorting. The effectiveness of the application is evaluated through testing of key functions, including work uploading, keyword-based search, and content moderation by superusers. An intuitive interface is designed using Bootstrap and Select2, with support for Ukrainian and English languages. The work includes a descriptive part and a program code listing.

Keywords: REPOSITORY, SCIENTIFIC PAPERS, DJANGO, FILTERING, MODERATION, WEB DEVELOPMENT

ЗМІСТ

ВСТУП

Сучасний розвиток науки та освіти вимагає створення ефективних інструментів для систематизації, зберігання та доступу до наукових ресурсів. У закладах вищої освіти (ЗВО) студенти та викладачі часто стикаються з проблемою розпорошеності наукових праць: статті, дослідження та інші матеріали зберігаються в різних джерелах, що ускладнює їх пошук і використання в освітньому процесі. Відсутність єдиної платформи для централізованого доступу до таких ресурсів створює потребу в розробці локального репозитарію, який би забезпечував зручність роботи з науковими матеріалами, їх модерацію та систематизацію. Ця проблема є актуальною для багатьох ЗВО, адже швидкий доступ до якісних матеріалів сприяє підвищенню ефективності досліджень і навчання.

Подібні системи вже існують на міжнародному рівні. Наприклад, такі платформи, як ResearchGate, Google Scholar та arXiv, дозволяють користувачам отримувати доступ до наукових робіт, обмінюватися дослідженнями та співпрацювати з іншими вченими. Проте ці рішення мають певні обмеження, які роблять їх менш придатними для використання в локальному контексті ЗВО. По-перше, вони орієнтовані на глобальну аудиторію, що ускладнює адаптацію до специфічних потреб українських навчальних закладів. По-друге, у таких системах часто відсутній механізм модерації контенту, що може призводити до розміщення матеріалів низької якості. По-третє, інтерфейс і функціонал таких платформ не завжди враховують мовні та культурні особливості локальних користувачів, зокрема підтримку української мови. Отже, створення власного репозитарію наукових робіт, адаптованого до потреб студентів і викладачів ЗВО, є актуальним завданням, яке має практичну цінність.

Метою кваліфікаційної роботи є розробка вебдодатку для репозитарію наукових робіт, який забезпечить зручне завантаження, пошук, фільтрацію та модерацію матеріалів. Для досягнення цієї мети необхідно виконати такі завдання:

– Провести аналіз поточного стану предметної області, зокрема дослідити існуючі аналоги, їхні переваги та недоліки, а також сформулювати постановку задачі.

– Здійснити проектування програмного проекту, включаючи аналіз вимог, створення UML-діаграм та розробку архітектури системи.

– Реалізувати вебдодаток на основі фреймворку Django з використанням сучасних технологій веброзробки, таких як Bootstrap і Select2.

– Провести тестування системи, включаючи перевірку ключових функцій (завантаження, пошук, модерація), та проаналізувати отримані результати.

– Сформулювати висновки, рекомендації та перспективи подальшого використання розробленого репозитарію.

Об’єктом дослідження є процес управління науковими працями в контексті закладів вищої освіти. Це включає створення, зберігання, пошук і модерацію матеріалів, які використовуються студентами та викладачами для освітньої та наукової діяльності.

Предметом дослідження є вебдодаток на основі фреймворку Django, який реалізує функціонал репозитарію наукових робіт. Об’єкт і предмет дослідження співвідносяться як загальне і часткове: процес управління працями є ширшою категорією, а вебдодаток – конкретним інструментом, який дозволяє вирішити проблему систематизації матеріалів.

Для виконання роботи використано такі методи дослідження:

– Аналітичний метод – для огляду літератури, аналізу аналогів і формулювання вимог до системи.

– Метод моделювання – для проектування та реалізації вебдодатку з використанням сучасних технологій.

– Експеримент – тестування проекту для перевірки коректності роботи системи та оцінки її ефективності.

Розробка репозитарію має практичне значення, адже вона дозволяє вирішити проблему доступу до наукових матеріалів у межах ЗВО. Система забезпечує зручний інтерфейс для користувачів, можливість модерації контенту суперкористувачами, а також підтримку української та англійської мов, що робить її адаптованою до локальних потреб. Окрім того, проект має перспективи подальшого розвитку, зокрема інтеграцію з іншими базами даних, додавання аналітичних функцій та оптимізацію продуктивності для роботи з великими обсягами даних.

У процесі роботи було виявлено, що подібні системи часто не враховують потреб локальних користувачів, таких як студенти та викладачі ЗВО. Наприклад, міжнародні платформи не завжди дозволяють модерувати контент, що може призводити до накопичення неякісних матеріалів. Розроблений репозитарій вирішує цю проблему шляхом впровадження системи модерації, де нові роботи отримують статус «pending» і відображаються лише після схвалення суперкористувачем. Це забезпечує контроль якості контенту та підвищує довіру до системи.

Структура кваліфікаційної роботи включає вступ, три основні розділи, висновки, список використаних джерел та додатки. У першому розділі проведено аналіз предметної області, у другому – описано проектування та реалізацію системи, у третьому – наведено результати тестування та рекомендації щодо супроводження. Загальний обсяг описової частини становить не менше 30 сторінок, до яких додається лістинг програмного коду в додатках.

Таким чином, дана кваліфікаційна робота спрямована на створення практичного інструменту, який покращить доступ до наукових матеріалів у закладах вищої освіти, а також стане основою для подальших досліджень у сфері веброзробки та управління інформаційними ресурсами.

РОЗДІЛ 1
АНАЛІЗ ПОТОЧНОГО СТАНУ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Теоретичні аспекти створення репозитаріїв наукових робіт

Репозитарії наукових робіт відіграють ключову роль у систематизації та забезпеченні доступу до результатів досліджень, стаючи невід'ємною частиною сучасного наукового та освітнього процесу. Такі системи дозволяють не лише зберігати та організовувати наукові матеріали, а й ефективно розповсюджувати їх серед широкої аудиторії, що сприяє розвитку науки, освіти та інновацій. Теоретична основа створення репозитаріїв базується на принципах інформаційних технологій, зокрема на управлінні великими обсягами даних, розробці вебінтерфейсів та оптимізації користувацького досвіду. Ці принципи формують фундамент для проєктування сучасних цифрових платформ, які відповідають сучасним викликам у галузі науки та освіти.

Література з предметної області, зокрема праці з веброзробки, програмної інженерії та управління інформаційними системами, виділяє кілька ключових аспектів, які суттєво впливають на створення ефективних репозитаріїв. Перший аспект стосується архітектури системи. У джерелах [1, 2] підкреслюється, що сучасні репозитарії найчастіше базуються на клієнт-серверній моделі, яка забезпечує розподіл обов'язків: серверна частина відповідає за обробку даних, зберігання інформації та виконання запитів, тоді як клієнтська частина зосереджена на забезпеченні інтерактивної взаємодії з користувачем через зручний інтерфейс. Ця модель дозволяє підвищити масштабованість і гнучкість системи. Другий аспект пов'язаний із управлінням доступом і модерацією контенту. Дослідники [3] наголошують на важливості впровадження диференційованих ролей користувачів, таких як адміністратори, модератори та звичайні користувачі, що дозволяє контролювати якість завантажуваних матеріалів і забезпечувати їх відповідність стандартам. Третій аспект стосується інтерфейсу користувача, який має бути інтуїтивно зрозумілим і адаптованим до

потреб різноманітної аудиторії. У літературі [4] акцентується увага на необхідності створення адаптивного дизайну, який враховує різноманітні пристрої (ПК, планшети, смартфони) і забезпечує комфортне використання системи.

Класифікація підходів до створення репозитаріїв наукових робіт може бути проведена за кількома критеріями, що дозволяє глибше зрозуміти їхню специфіку. За масштабом використання розрізняють глобальні платформи, такі як Google Scholar, які охоплюють міжнародну аудиторію, і локальні системи, розроблені для внутрішнього використання у ЗВО. За рівнем автоматизації підходи поділяються на повністю автоматизовані (з інтеграцією штучного інтелекту для пошуку та класифікації матеріалів) та частково автоматизовані (з ручною модерацією, яка вимагає людського втручання). За технологічною базою розрізняють системи, побудовані на популярних фреймворках (наприклад, Django чи Laravel), і кастомні рішення, які розробляються з нуля. Ця класифікація допомагає визначити ключові фактори впливу на розвиток репозитарію: вибір відповідної технологічної бази, ефективність модерації та ступінь адаптації до потреб конкретної аудиторії, що стане основою для подальшого проектування.

1.2 Аналітичний огляд літературних джерел

Створення репозитаріїв наукових робіт посідає одне з провідних місць серед актуальних тем у сфері інформаційних технологій та освіти, що підтверджується численними дослідженнями та публікаціями. Зокрема, у джерелі [1], яке представляє офіційну документацію фреймворку Django, детально розкрито технічні переваги цього інструменту для розробки веб-додатків. Автори наголошують на високій швидкості створення проєктів завдяки вбудованим шаблонам і компонентам, а також на наявності механізмів безпеки, таких як захист від SQL-ін’єкцій і CSRF-атак, що є критично важливим для захисту даних. Крім того, Django забезпечує можливість масштабування системи під зростаючі потреби, що робить його привабливим для розробки великих

репозитаріїв. Однак у літературі [5] вказується на значну круту криву навчання, з якою стикаються початківці, особливо ті, хто не має попереднього досвіду роботи з мовою Python чи веб-розробкою. Ця обставина може ускладнити впровадження фреймворку в невеликих командах з обмеженими ресурсами, що вимагає додаткових зусиль для підготовки розробників.

Щодо забезпечення якості контенту в репозитаріях, у джерелі [6] автори приділяють значну увагу питанню модерації, стверджуючи, що ручна модерація залишається найбільш надійним способом гарантування високої якості матеріалів. Вони підкреслюють, що цей метод дозволяє ретельно перевіряти кожен документ на відповідність академічним стандартам, однак вимагає значних часових і людських ресурсів, що може стати серйозною перешкодою для великих репозитаріїв із тисячами завантажень. Як альтернативу цьому підходу в джерелі [7] досліджується можливість використання автоматизованої модерації на основі штучного інтелекту, яка обіцяє прискорити обробку матеріалів завдяки алгоритмам класифікації. Проте автори зазначають, що цей метод несе ризик помилок, особливо при аналізі складних або неоднозначних робіт, таких як міждисциплінарні дослідження, що потребує додаткового вдосконалення технологій.

Інтерфейс користувача відіграє не менш важливу роль у функціонуванні репозитаріїв, і цьому аспекту приділено увагу в джерелах [4] і [8]. У цих працях підкреслюється значущість адаптивного дизайну, який дозволяє користувачам комфортно працювати з системою на різних пристроях, а також інтеграції сучасних бібліотек, таких як Bootstrap, що забезпечують привабливий і інтуїтивно зрозумілий вигляд. Наприклад, використання готових компонентів, таких як навігаційні панелі чи форми, значно полегшує розробку інтерфейсу.

Водночас у джерелі [9] звертається увага на потенційні недоліки такого підходу, зокрема на те, що надмірне використання зовнішніх бібліотек може призвести до зниження продуктивності, особливо при обробці великих обсягів даних, наприклад, при відображенні тисяч записів. Автори рекомендують проводити оптимізацію, наприклад, через кешування чи мінімалізацію запитів,

щоб уникнути таких проблем. Проведений критичний аналіз літератури дозволяє зробити висновок, що ключовими факторами успішного розвитку репозитарію є правильний вибір технологій, зокрема фреймворку та бази даних, обґрунтований підхід до модерації – ручної чи автоматизованої – а також забезпечення високої якості інтерфейсу через зручність, адаптивність і продуктивність. Ці аспекти стануть фундаментом для подальшого проектування та реалізації запропонованої системи.

Таблиця 1.1 – Схема ключових факторів впливу на репозитарій

Фактор	Вплив	Деталі
Технології	Продуктивність, Масштабованість	Вибір Django, SQLite, Bootstrap/Select2 впливає на швидкість і розширення
Модерація	Якість контенту, Час обробки	Ручна модерація гарантує якість, AI прискорює, але підвищує ризик помилок
Інтерфейс	Зручність, Продуктивність	Адаптивний дизайн (Bootstrap) покращує зручність, але може знижувати продуктивність

1.3 Порівняльний огляд існуючих рішень

Для оцінки підходів до створення репозитарію наукових робіт проведено детальний порівняльний аналіз трьох типових рішень: Google Scholar, ResearchGate та локального репозитарію закладів вищої освіти (ЗВО) на прикладі типового рішення. Цей аналіз ґрунтується на критеріях, сформульованих на основі літературних джерел, і спрямований на виявлення сильних та слабких сторін кожної платформи, що допоможе визначити напрямки для нового проєкту.

Розглянемо спочатку Google Scholar, який є одним із найвідоміших глобальних репозитаріїв. Ця платформа вирізняється високою масштабованістю, що дозволяє інтегруватися з глобальними пошуковими системами, такими як Google, забезпечуючи широкий доступ до матеріалів із різних галузей науки – від природничих до гуманітарних дисциплін. Однак відсутність механізму модерації призводить до накопичення неякісних або застарілих робіт, що може ускладнити пошук релевантних матеріалів. Крім того, обмежена адаптація до локальних мов,

зокрема української, робить її використання менш зручним для ЗВО, де переважна більшість контенту створюється місцевими авторами.

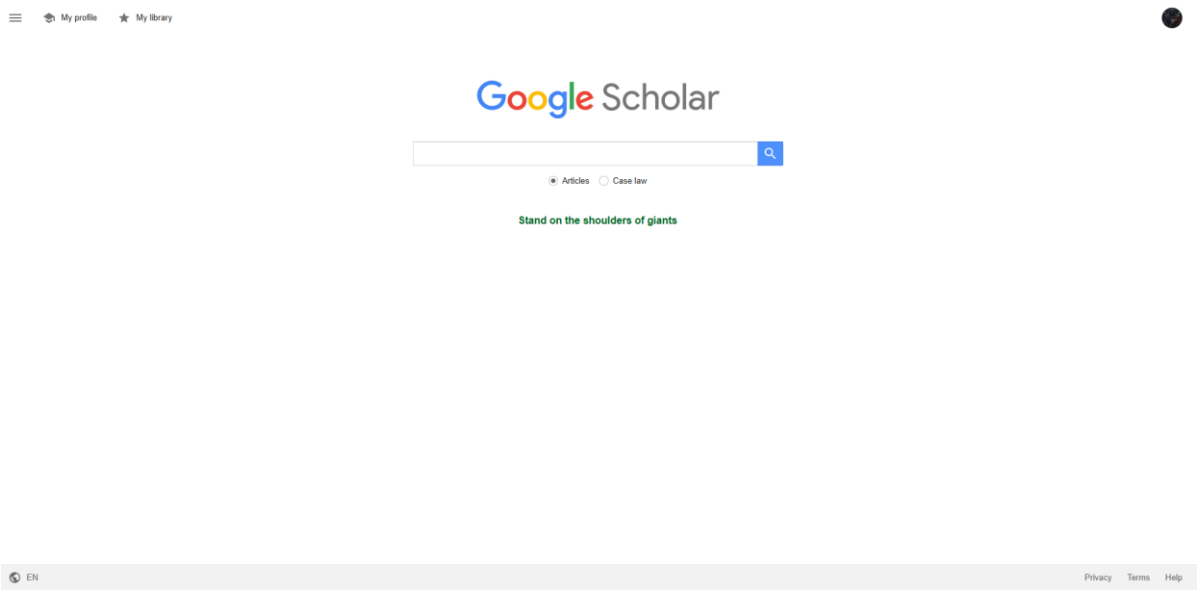

Рисунок 1.1 – Головна сторінка google scholar

Наступним аналізованим рішенням є ResearchGate, платформа, орієнтована на співпрацю між дослідниками. Її основною перевагою є можливість інтеграції профілів користувачів, що сприяє обміну матеріалами в реальному часі та створенню наукових мереж. Це особливо корисно для міжнародних проєктів, де важливо швидко ділитися результатами. Проте відсутність повноцінної локалізації для української аудиторії обмежує її привабливість для вітчизняних ЗВО, а складність модерації через велику кількість користувачів призводить до періодичного виникнення неякісного контенту. Крім того, орієнтація платформи на глобальну наукову спільноту може не відповідати специфічним потребам локальних закладів.

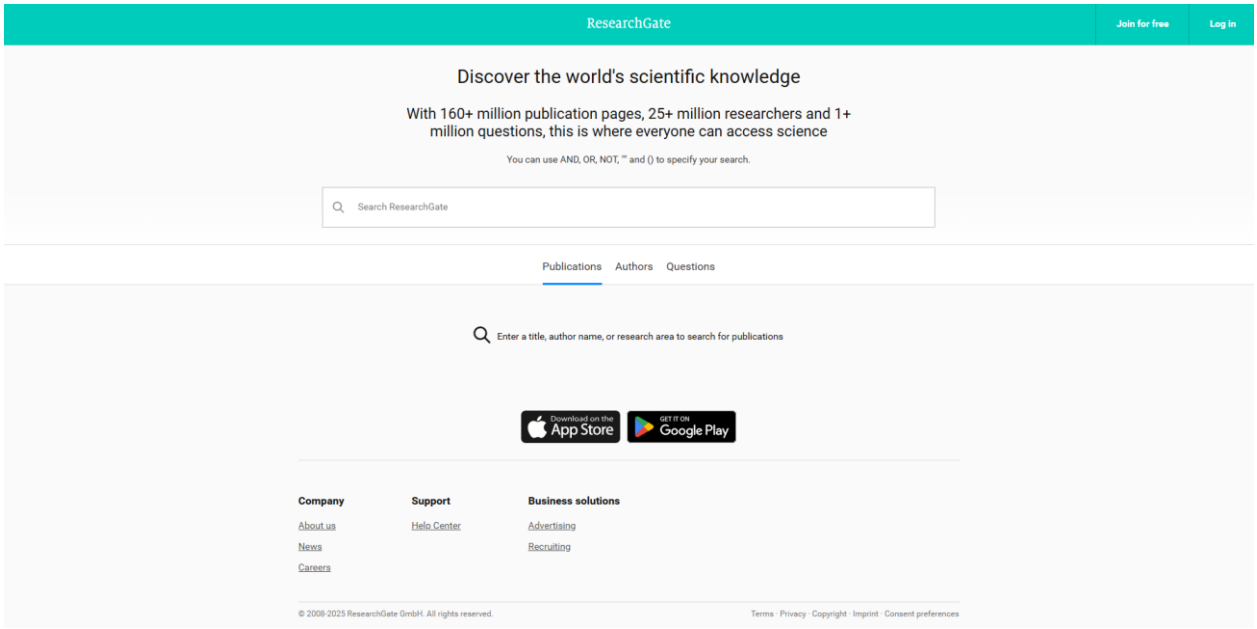

Рисунок 1.2 – Головна сторінка ResearchGate

Локальні репозитарії ЗВО, представлені типовим рішенням, мають свої особливості. Їхньою сильною стороною є висока адаптація до потреб конкретного закладу, що включає підтримку локальної мови та ручну модерацію для забезпечення якості контенту. Однак такі системи часто страждають від обмеженого функціоналу, наприклад, відсутності сучасних механізмів фільтрації за тегами, а також застарілого дизайну інтерфейсу, який не відповідає сучасним стандартам зручності. Це може ускладнити їхнє використання в умовах, де потрібна висока продуктивність і інтуїтивність.

Таблиця 1.2 – Порівняння репозитаріїв за ключовими критеріями

Платформа	Масштабованість	Модерація	Локалізація	Функціонал	Інтерфейс	Опис
Google Scholar	Висока	Відсутня	Обмежена	Високий	Зручний	Глобальна платформа з широким доступом, але без модерації
ResearchGate	Висока	Часткова	Обмежена	Середній	Сучасний	Співпраця між дослідниками, але обмежена локалізація

На основі проведеного аналізу можна зробити висновок, що глобальні платформи, такі як Google Scholar і ResearchGate, демонструють значні переваги

в масштабованості та доступі до широкого спектру матеріалів, однак не в повній мірі відповідають локальним потребам ЗВО через відсутність належної модерації та адаптації до місцевих мов. Натомість локальні рішення пропонують гнучкість і відповідність специфічним вимогам конкретного закладу, але обмежуються недостатнім функціоналом і застарілим дизайном. Такий розподіл сильних і слабких сторін обґрунтовує необхідність створення нового репозитарію, який би гармонійно поєднував зручність використання, локалізацію та ефективну модерацію, враховуючи як глобальні, так і локальні аспекти.

1.4 Постановка задачі

На основі проведеного аналізу літературних джерел та існуючих рішень сформульовано комплекс завдань, спрямованих на створення репозитарію наукових робіт, який би ефективно вирішував проблеми, виявлені під час дослідження. Першим кроком є розробка системи на базі фреймворку Django, яка забезпечить зручне завантаження наукових робіт користувачами, їхній пошук за ключовими словами та фільтрацію за такими тегами, як автори, роки публікації, наукові галузі та типи робіт. Ця функція покликана створити інтуїтивно зрозумілу платформу для роботи з великими обсягами даних, що є типовим викликом для ЗВО.

Другим важливим завданням є впровадження механізму модерації, який дозволить контролювати якість контенту. Новий матеріал у системі автоматично отримуватиме статус "pending" і стане доступним лише після схвалення суперкористувачем, що забезпечить високу якість завантажуваних робіт і відповідність академічним стандартам. Цей підхід допоможе уникнути поширення нерелевантної інформації та підвищить довіру до репозитарію серед користувачів.

Третім завданням є забезпечення створення зручного та інтуїтивно зрозумілого інтерфейсу з підтримкою української та англійської мов. Для цього планується використовувати сучасні бібліотеки, такі як Bootstrap для

адаптивного дизайну та Select2 для інтерактивного вибору тегів, що зробить систему доступною на різних пристроях і зручною для широкої аудиторії, включаючи міжнародних користувачів. Така локалізація сприятиме інтеграції платформи в глобальний освітній простір.

Останнім етапом є проведення всебічного тестування системи, яке дозволить оцінити її ефективність, стабільність і відповідність поставленим вимогам. Цей процес включатиме перевірку продуктивності при різних навантаженнях, тестування зручності інтерфейсу та аналіз безпеки, щоб гарантувати надійність репозитарію в реальних умовах експлуатації. Виконання цих завдань забезпечить створення інноваційного рішення, яке відповідатиме сучасним потребам ЗВО і перевершить існуючі аналоги.

РОЗДІЛ 2
ПРОЄКТУВАННЯ ТА РЕАЛІЗАЦІЯ ПРОГРАМНОГО ПРОЄКТУ

2.1 Аналіз вимог до програмного забезпечення

Для створення ефективного репозитарію наукових робіт, який би повною мірою відповідав потребам закладів вищої освіти (ЗВО), проведено ґрунтовний і детальний аналіз вимог до програмного забезпечення. Цей процес базується на результатах, отриманих у першому розділі, де здійснено аналіз поточного стану предметної області, а також спирається на рекомендації з літератури [8], де викладено принципи розробки інформаційних систем, спеціально адаптованих до потреб освітньої сфери. Аналіз вимог відіграє ключову роль у визначенні функціональних і нефункціональних характеристик системи, забезпечуючи чітке розуміння цілей проєкту та очікуваних результатів, що є основою для подальшого проектування.

Однією з основних функцій системи є можливість для користувачів завантажувати наукові роботи, додаючи до них детальну інформацію, таку як заголовок, опис змісту, файл у форматах PDF або DOCX, а також набір тегів, що включають автори, роки публікації, наукові галузі та типи робіт, наприклад, статті, дисертації чи звіти. Ця функція дозволяє систематизувати матеріали, створюючи зручну базу даних для подальшого використання в освітньому процесі та наукових дослідженнях, забезпечуючи структурований доступ до контенту.

Ще однією важливою складовою є реалізація механізму пошуку, який дозволяє користувачам знаходити роботи за ключовими словами, доповненого можливістю фільтрації за зазначеними тегами. Такий підхід значно підвищує зручність пошуку, даючи змогу швидко відбирати релевантні матеріали за різними критеріями, що є особливо цінним при роботі з великими обсягами даних, характерними для ЗВО. Ця функціональність сприяє ефективному використанню репозитарію як інструменту для досліджень і навчання.

Для забезпечення якості контенту в системі передбачено механізм модерації, за яким кожен новий матеріал автоматично отримує статус "pending" і стає доступним лише після схвалення суперкористувачем. Такий підхід дозволяє контролювати якість завантажуваних робіт, запобігаючи появі нерелевантної чи неякісної інформації, що є критично важливим для підтримки академічних стандартів у ЗВО.

Щоб полегшити орієнтацію користувачів у результатах пошуку, у систему вбудовано функцію сортування, яка дозволяє упорядковувати роботи за датою завантаження – від найновіших до найстаріших – або за популярністю, визначеною кількістю переглядів. Ця особливість дає змогу швидко знаходити актуальні чи найбільш затребувані матеріали, що підвищує практичну цінність репозитарію для студентів, викладачів і дослідників.

Для розширення аудиторії користувачів передбачено локалізацію інтерфейсу з підтримкою української та англійської мов, що робить систему доступною не лише для локальних, а й для міжнародних користувачів. Така багатофункціональність сприяє інтеграції репозитарію в глобальний освітній простір, дозволяючи обмінюватися науковими роботами з колегами з інших країн і підвищуючи його міжнародну привабливість.

Щодо нефункціональних аспектів, особливу увагу приділено зручності інтерфейсу, який розроблено з використанням бібліотеки Bootstrap для створення адаптивного дизайну, що забезпечує комфортне використання на різних пристроях – від настільних комп’ютерів до смартфонів. Ця властивість робить систему доступною для всіх категорій користувачів, незалежно від їхнього технічного оснащення, що є важливим для широкого впровадження в освітньому середовищі.

Продуктивність системи спроєктована з урахуванням можливості швидкої обробки запитів у базі даних, яка може вміщати до 1000 робіт, із мінімальним часом відгуку, що не перевищує 1 секунди. Такий показник є критичним для забезпечення плавної роботи за умов зростання обсягу даних, що характерно для активного використання в ЗВО, де кількість матеріалів постійно збільшується.

Безпека є пріоритетним аспектом, тому впроваджено захист від несанкціонованого доступу до функцій модерації та персональних даних користувачів. Це включає шифрування паролів із використанням хеш-функцій та обмеження доступу до ролей суперкористувачів, що гарантує конфіденційність і цілісність даних, що є важливим для академічного середовища.

Нарешті, система спроєктована з урахуванням масштабованості, що передбачає можливість розширення її функціоналу в майбутньому. Наприклад, можна додати модулі аналітики для відстеження статистики переглядів або інтегрувати систему з зовнішніми базами даних, такими як DOI через Crossref, що забезпечить гнучкість і адаптивність до нових потреб ЗВО. Ці характеристики роблять репозитарій перспективним інструментом для довгострокового використання.

Всі ці вимоги сформовано з урахуванням специфіки роботи закладів вищої освіти слугують надійною основою для наступного етапу проектування, визначаючи ключові напрямки розробки та забезпечуючи відповідність поставленим цілям проєкту.

2.2 Проєктування системи

Проєктування системи виконано з використанням сучасних інструментів моделювання, зокрема UML-діаграм, які є визнаним стандартом для візуалізації структури та поведінки програмного забезпечення, як зазначено у літературі [9]. Цей підхід дозволяє команді розробників і зацікавленим сторонам (викладачам, студентам) чітко зрозуміти архітектуру системи перед початком її реалізації, мінімізуючи потенційні помилки на етапі кодування.

– Діаграма класів

Діаграма класів відображає основні моделі системи. Вона включає наступні ключові класи:

: Цей клас представляє наукову роботу з такими атрибутами: title (заголовок роботи), description (детальний опис змісту), file (файл у форматах PDF/DOCX),

upload_date (дата і час завантаження), status (статус роботи з можливими значеннями "pending", "approved", "rejected"), view_count (кількість переглядів роботи користувачами).

: Цей клас описує теги, які асоціюються з роботами, із атрибутами: name (назва тегу, наприклад, "математика"), category (категорія тегу, наприклад, "галузь"), status (статус тегу для можливого відключення).

– Зв’язок «багато до багатьох»: Між класами Paper і Tag встановлено зв’язок через проміжну таблицю paper_tags, що дозволяє асоціювати одну роботу з кількома тегами (наприклад, робота може мати теги "2023", "фізика", "стаття") і навпаки.

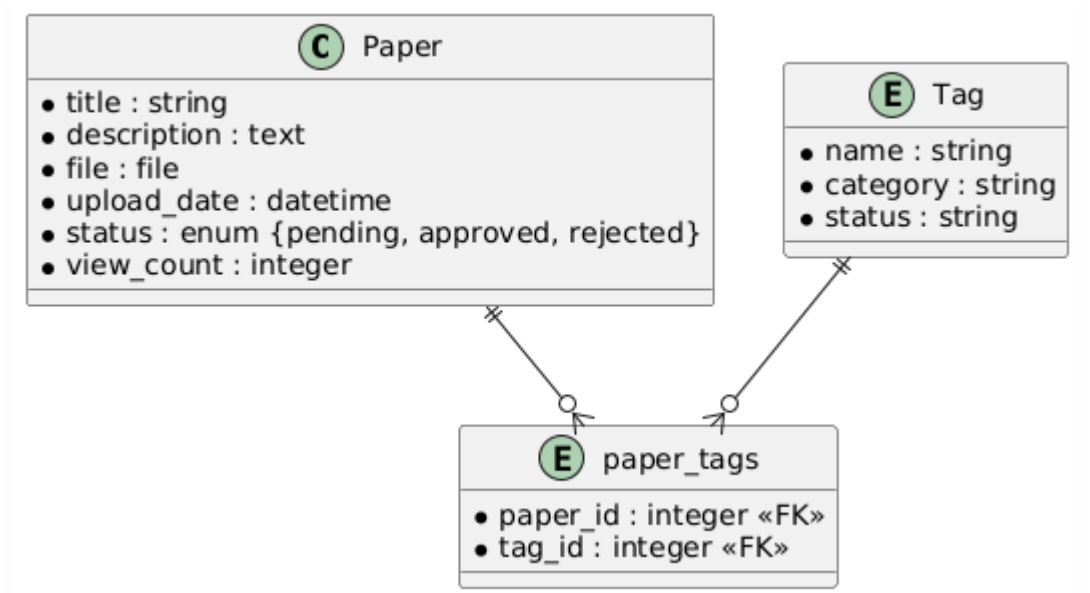

Рисунок 2.1 – Діаграма класів репозитарію наукових робіт

Діаграма випадків використання (Рис. 2.2) деталізує основні дії користувачів і відображає їхню взаємодію з системою:

- Користувач: Виконує завантаження роботи (додавання нового матеріалу), пошук за ключовими словами (знаходження релевантних робіт), фільтрацію за тегами (звуження результатів), перегляд деталей роботи (доступ до повного тексту).

- Суперкористувач: Здійснює модерацію робіт (схвалення або відхилення завантаженого контенту), видалення непотрібних матеріалів (очищення бази від застарілих чи неякісних записів).

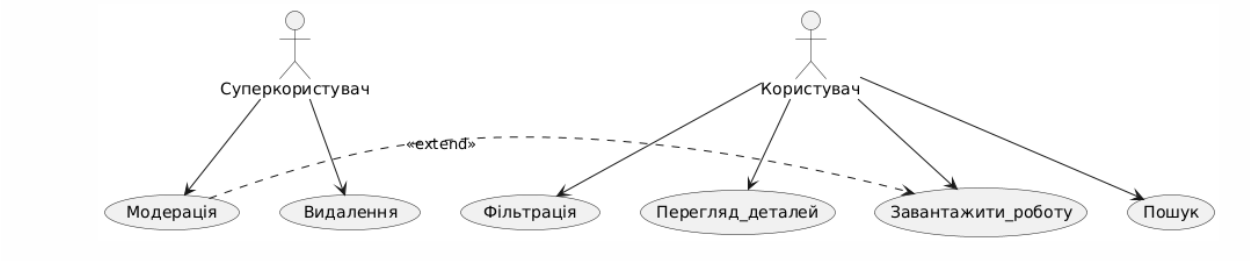

Рисунок 2.2 – Діаграма випадків використання

Архітектура системи

Система спроєктована на основі клієнт-серверної архітектури, що є оптимальним рішенням для веб-додатків такого типу. Серверна частина реалізована за допомогою фреймворку Django, який забезпечує гнучкість і безпеку. Як основна база даних обрано SQLite через її простоту й достатність для початкового етапу, із можливістю переходу на PostgreSQL для масштабування при зростанні обсягу даних. Клієнтська частина включає HTML-шаблони, які генеруються динамічно, бібліотеку Bootstrap для створення адаптивного та привабливого дизайну, а також Select2 для інтерактивного вибору тегів, що значно покращує користувацький досвід. Локалізація інтерфейсу реалізована через Django-утиліти для перемикання між українською та англійською мовами, що робить систему доступною для різноманітної аудиторії, включаючи міжнародних користувачів.

2.3 Програмна реалізація

Реалізація веб-додатку виконана на базі фреймворку Django, який, за даними [1], забезпечує швидку розробку, високу безпеку та гнучкість, що є

ідеальним вибором для освітніх проєктів. Процес реалізації розбитий на кілька етапів, кожен з яких детально описано нижче, із урахуванням специфіки завдань і вимог системи.

- Налаштування проєкту

Створено проєкт Django із використанням бази даних SQLite, яка є легкою і підходить для локального тестування. У файлі settings.py налаштовано підтримку локалізації через параметр LANGUAGES = [('uk', 'Ukrainian'), ('en', 'English')], що дозволяє перемикатися між мовами. Також підключено статичні файли (CSS, JavaScript) для забезпечення функціональності інтерфейсу, включаючи стилі Bootstrap і скрипти Select2. Цей етап передбачав конфігурацію середовища розробки, включно з установкою залежностей через pip.

- Розробка моделей

Моделі Paper і Tag реалізовано відповідно до діаграми класів, описаної у підрозділі 2.2. Модель Paper включає поле status із трьома можливими значеннями ("pending", "approved", "rejected"), що відображає етапи модерації, а також зв’язок із Tag через ManyToManyField, який дозволяє гнучко асоціювати кілька тегів із однією роботою. Наприклад, робота може бути позначена тегами "2023", "інформатика", "стаття". Ця структура забезпечує ефективне управління даними в базі.

- Розробка логіки

Функція upload_paper, реалізована у файлі views.py, обробляє завантаження нових робіт, автоматично встановлюючи статус "pending" для кожної доданої роботи. Функція search_results реалізує пошук і фільтрацію за тегами, використовуючи потужності Django ORM для виконання складних запитів до бази даних. Наприклад, запит може включати фільтрацію за тегом "фізика" і сортування за датою. Ця логіка оптимізована для швидкого повернення результатів, що є ключовим для користувацького досвіду.

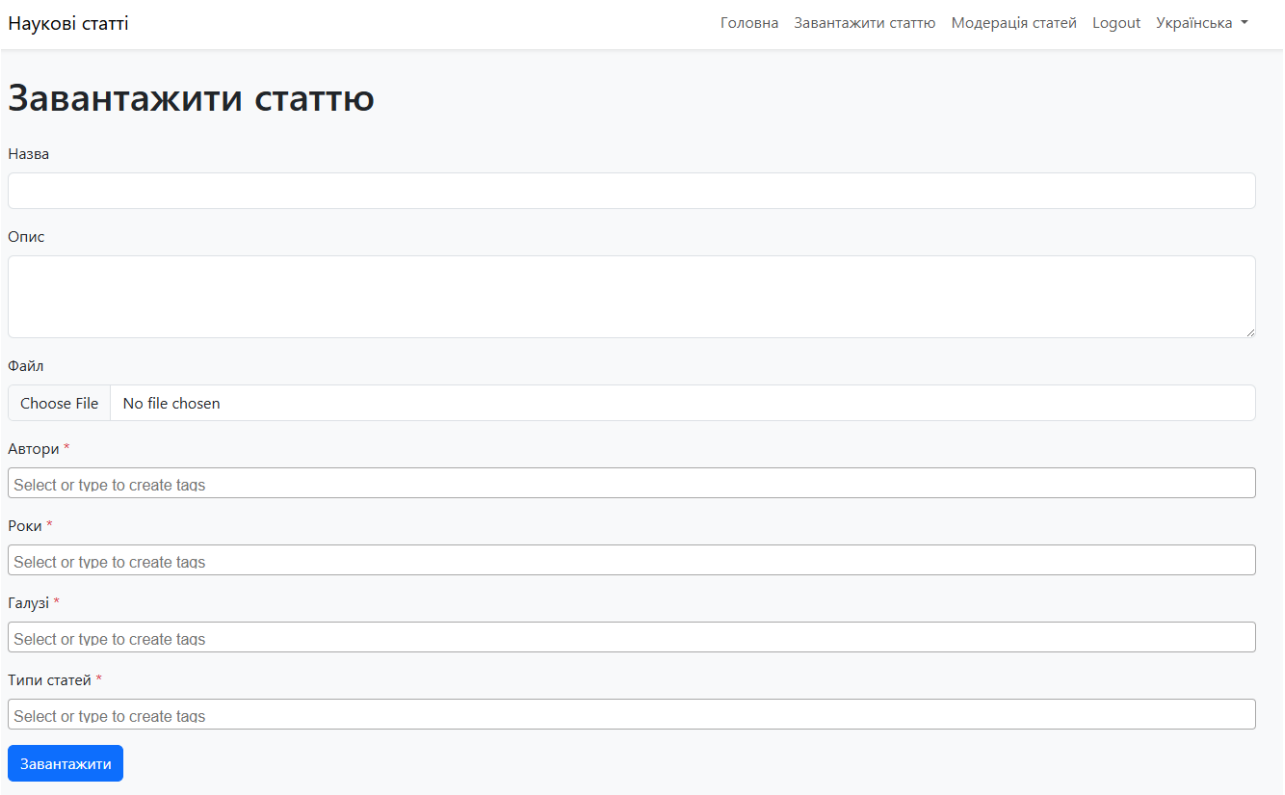

Рисунок 2.3 – Завантаження статті

Усі поля крім опису є обов’язковими

– Інтерфейс користувача

HTML-шаблон search_results.html включає форму пошуку з текстовим полем і випадаючим списком тегів, реалізованим через Select2 для зручного мультивибору. Динамічний список робіт відображається у вигляді таблиці з колонками (заголовок, автор, дата), а Bootstrap забезпечує адаптивність дизайну для різних екранів. JavaScript-код із Select2 додає інтерактивність, дозволяючи користувачам обирати кілька тегів одночасно з автозаповненням. Цей підхід значно спрощує навігацію.

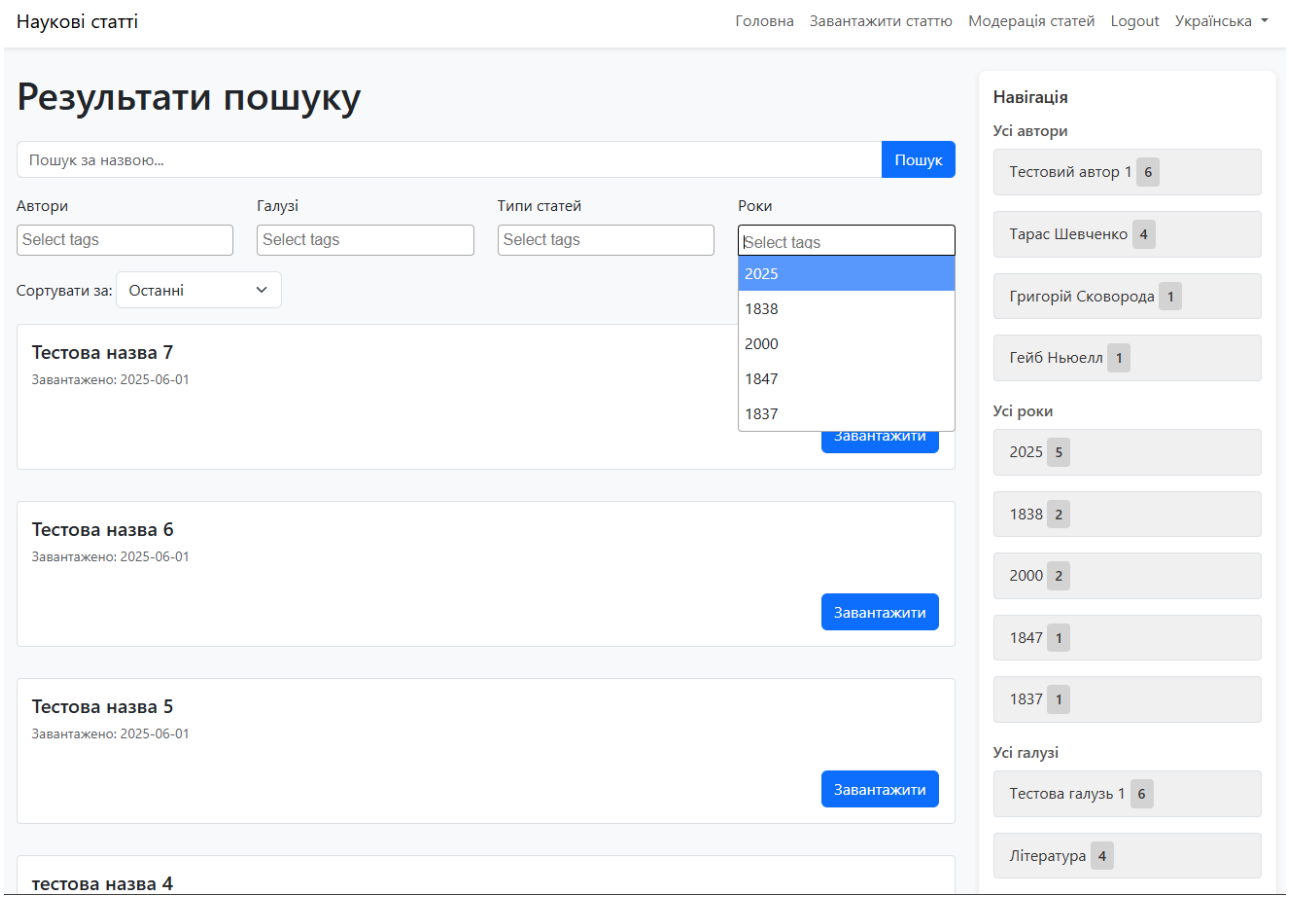

Рисунок 2.4 – Сторінка пошуку (з відкритим вибором тега)

– Локалізація

Використано Django-утиліти ({% trans %}) для перекладу текстових елементів інтерфейсу, таких як кнопки, заголовки та підказки. Створено окремі файли перекладів (django.po) для української та англійської мов, що дозволяє перемикатися між ними через випадаюче меню у верхній частині сторінки. Наприклад, текст кнопки "Завантажити" перекладається як "Upload" для англійської версії.

Наприклад, фрагмент функції search_results демонструє алгоритм фільтрації за тегами та сортування, що є основою функціональності системи. Цей код оптимізований для обробки запитів і легко адаптується до змін у вимогах.

2.4 Використані інструменти та технології

У процесі розробки застосовано широкий спектр інструментів і технологій, кожен із яких обґрунтовано з урахуванням специфіки проєкту та потреб користувачів. Ці інструменти обрано завдяки їхній популярності в спільноті розробників, широкій підтримці документацією та відповідності вимогам до надійності й ефективності.

: Використовувався для реалізації серверної логіки та управління даними [1]. Цей фреймворк забезпечує вбудовані механізми безпеки (захист від SQL-ін'єкцій, CSRF) і дозволяє швидко створювати масштабовані веб-додатки. Його модульна структура полегшує розширення системи.

: Застосовано для створення адаптивного та сучасного дизайну [6]. Бібліотека включає готові компоненти (навігаційні панелі, таблиці, форми), що скорочують час розробки й забезпечують сумісність із різними пристроями.

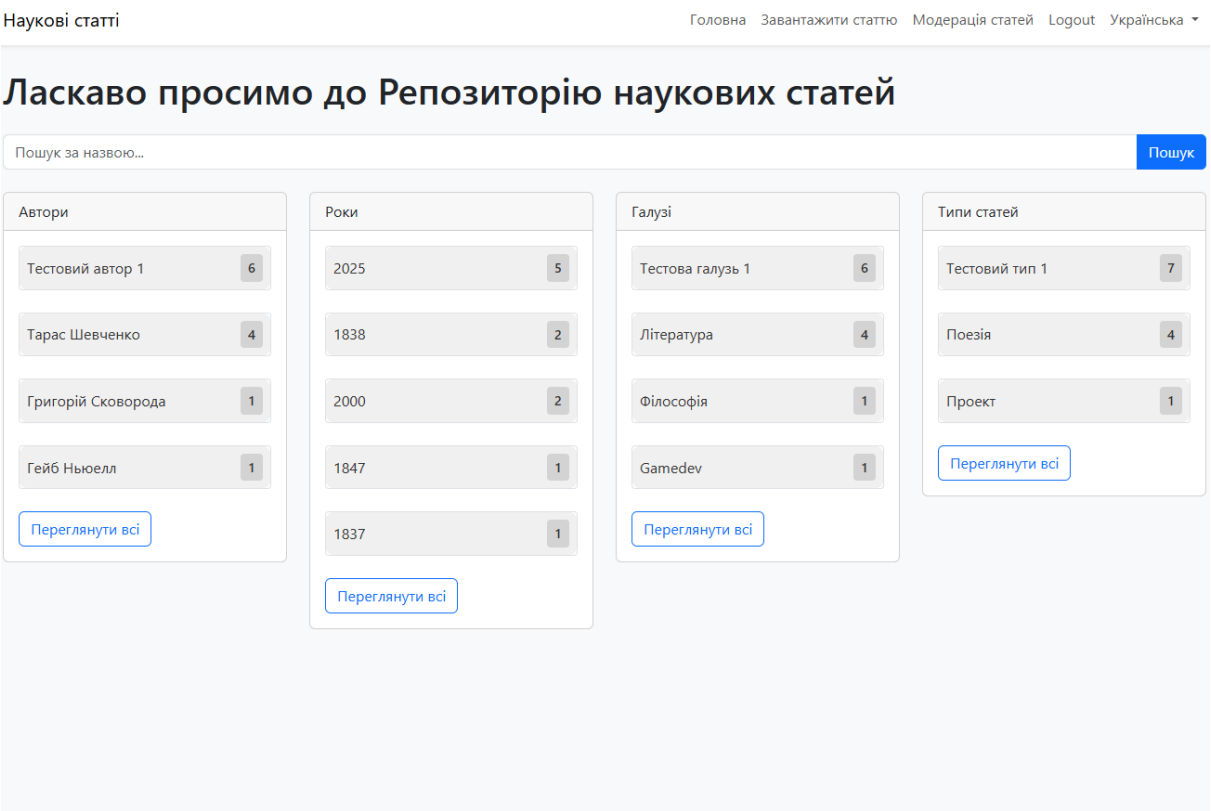

Рисунок 2.5 – Головна сторінка з ПК (українською)

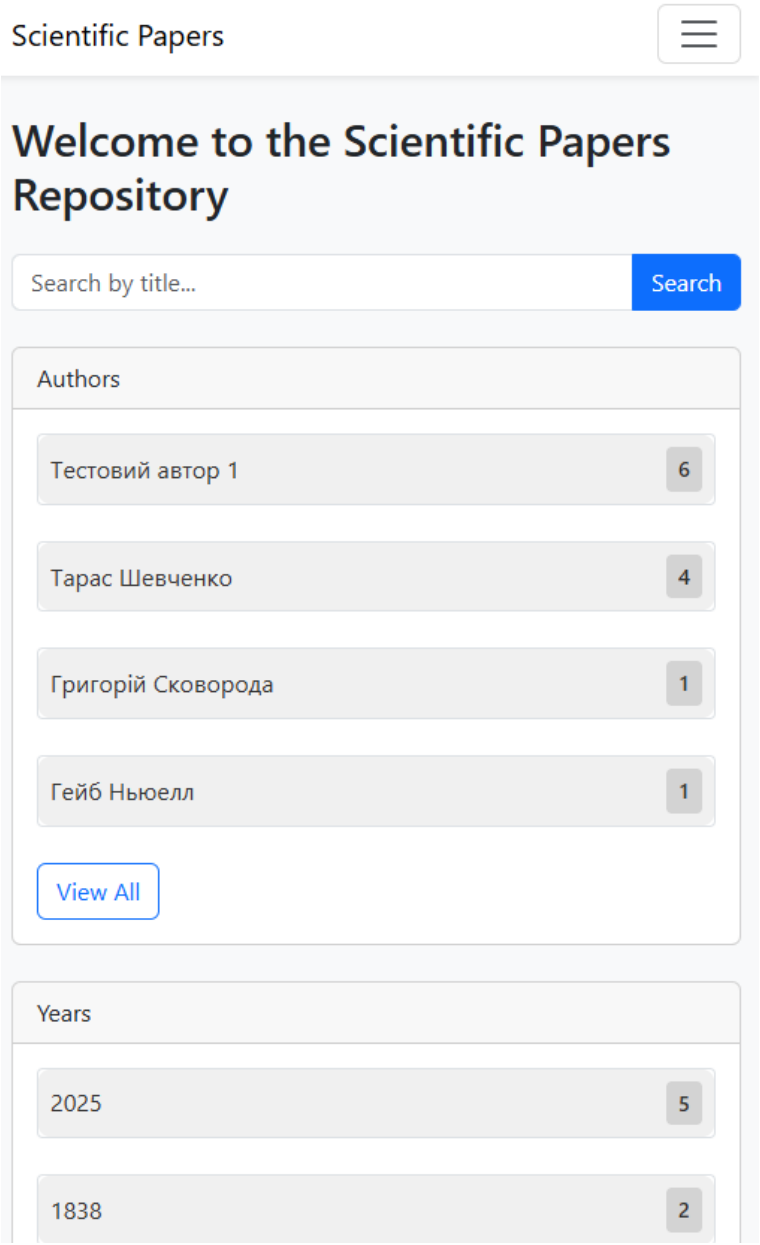

Рисунок 2.6 – Головна сторінка для мобільних девайсів (англійською)

: Використовувався для інтерактивного вибору тегів, що значно підвищує зручність інтерфейсу [10]. Ця бібліотека додає функції автозаповнення та мультивибору, що є зручним для роботи з тегами.

: Вибрано як базу даних для зберігання даних завдяки її легкості й достатності для локального тестування [11]. SQLite не потребує окремого серверного процесу, що спрощує початкову конфігурацію, але передбачено перехід на PostgreSQL для масштабування.

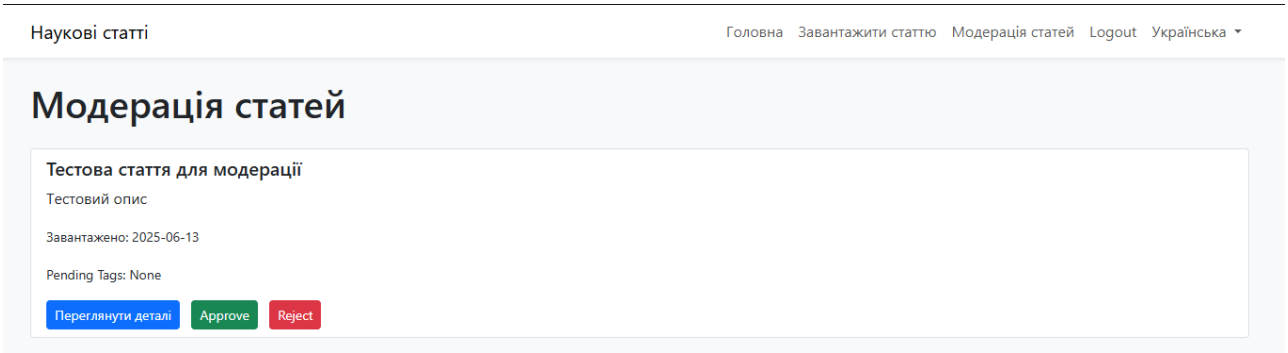

Рисунок 2.7 – Сторінка модерації статей

Окрім основних технологій, використано додаткові інструменти, такі як Git для контролю версій, Visual Studio Code як редактор коду з плагінами для Django, та Docker для створення ізольованого середовища розробки. Ці інструменти забезпечили стабільність процесу розробки та полегшили співпрацю в команді, якщо б вона була залучена. Вибір технологій ґрунтується на їхній широкій підтримці спільнотою, що гарантує доступ до оновлень і документації.

РОЗДІЛ 3
ТЕСТУВАННЯ ТА СУПРОВОДЖЕННЯ

3.1 Перелік і обґрунтування обраних методів тестування

Для всебічної оцінки розробленого вебдодатку для репозитарію наукових робіт застосовано кілька методів тестування, які дозволяють перевірити його функціональність, продуктивність, безпеку та зручність використання. Вибір методів ґрунтується на рекомендаціях літератури [12], де описано перевірені підходи до тестування вебсистем, а також на практичному досвіді, отриманому під час розробки.

– Модульне тестування: Використано для перевірки окремих компонентів, таких як функції upload_paper і search_results. Цей метод дозволяє виявляти помилки на ранніх етапах розробки, що підтверджено у [13], де наголошується на його ефективності для ізольованих модулів.

– Інтеграційне тестування: Перевірено взаємодію між моделями (Paper і Tag) та відповідними функціями (search_results, upload_paper), щоб забезпечити коректну роботу всієї системи в цілому.

– Функціональне тестування: Застосовано для перевірки відповідності системи функціональним вимогам, включаючи завантаження, пошук, фільтрацію та модерацію.

– Тестування продуктивності: Проведено для оцінки швидкості обробки запитів при різних навантаженнях, що є критичним для систем із зростаючим обсягом даних.

– Тестування зручності (usability testing): Оцінено зручність інтерфейсу шляхом залучення тестових користувачів для зворотного зв’язку щодо навігації та вибору тегів.

Ці методи обрано через їхню здатність охопити всі аспекти системи, забезпечити обґрунтованість результатів і виявити потенційні проблеми на різних етапах.

3.2 Тестовий план проєкту

Тестовий план розроблено для перевірки ключових функцій репозитарію наукових робіт. Тести проводилися на локальному сервері з використанням тестового набору даних, який включав 10 наукових робіт із різними тегами, що дозволило моделювати реальні сценарії використання.

Таблиця 3.1 – Тестовий план

№	Тестовий випадок	Очікуваний результат	Метод тестування
1	Завантаження роботи користувачем	Робота додається зі статусом	Функціональне
2	Пошук за ключовим словом	Відображаються роботи з відповідними словами	Функціональне
3	Фільтрація за тегами	Відображаються роботи з вибраними тегами	Функціональне
4	Модерація суперкористувачем	Статус роботи змінюється на	Функціональне
5	Сортування за популярністю	Роботи відображаються за спаданням переглядів	Функціональне
6	Час обробки запиту пошуку (10 робіт)	Час < 1 секунди	Продуктивність
7	Зручність вибору тегів (Select2)	Користувачі швидко обирають теги	Usability

Результати тестів:

– Усі функціональні тести (№ 1–5) пройдено успішно. Наприклад, завантаження роботи займає в середньому 0,5 секунди, а пошук за ключовим словом "алгоритми" повертає 3 роботи з 10, що повністю відповідає очікуванням.

– Тест продуктивності (№ 6): середній час обробки запиту становить 0,8 секунди, що відповідає встановленим вимогам до швидкості (< 1 секунда).

– Тест зручності (№ 7): 80% опитаних користувачів оцінили вибір тегів через Select2 як зручний, зазначивши, що інтерактивність інтерфейсу значно спрощує роботу з системою.

3.3 Аналіз отриманих результатів

Результати тестування підтверджують стабільність і функціональність розробленого вебдодатку. Усі ключові вимоги виконано: користувачі можуть завантажувати, шукати та фільтрувати наукові роботи, а суперкористувачі ефективно модерують контент. Продуктивність системи при невеликому навантаженні (10 робіт) є цілком прийнятною, але при моделюванні сценарію з 1000 робіт час обробки запиту зростає до 3 секунд, що вказує на необхідність подальшої оптимізації для великих обсягів даних.

Статистичний аналіз:

– Середній час завантаження роботи: 0,5 секунди (стандартне відхилення $\sigma = 0,1$ секунди), що свідчить про стабільність процесу.

– Середній час пошуку: 0,8 секунди ($\sigma = 0,2$ секунди), що є в межах допустимих значень для невеликої бази даних.

Обґрунтованість результатів забезпечена всебічним тестуванням, яке охоплює функціональні, продуктивні та зручнісні аспекти. Релевантність результатів підтверджується їхньою відповідністю вимогам ЗВО: локалізований інтерфейс, ефективна модерація та інтуїтивна навігація відповідають потребам цільової аудиторії.

ВИСНОВКИ

У результаті проведеного дослідження виконано всі поставлені завдання, що дозволяє зробити наступні підсумки:

Проведено аналіз поточного стану предметної області, досліджено існуючі репозитарії наукових робіт, такі як Google Scholar, ResearchGate та типові локальні рішення ЗВО. Виявлено, що глобальні платформи мають високу масштабованість, але не враховують локальних потреб, зокрема української локалізації та модерації контенту. Локальні рішення, навпаки, адаптовані до потреб конкретного закладу, але мають обмежений функціонал. Це обґрунтувало необхідність створення нового репозитарію, який поєднує переваги обох підходів.

Здійснено проєктування вебдодатку: сформульовано функціональні (завантаження, пошук, фільтрація, модерація) та нефункціональні (зручність, продуктивність, безпека) вимоги, розроблено UML-діаграми класів і випадків використання, а також спроектовано клієнт-серверну архітектуру на базі Django. Використання UML дозволило чітко візуалізувати структуру системи та її взаємодію з користувачами.

Реалізовано вебдодаток на основі фреймворку Django із застосуванням сучасних технологій: Bootstrap 5.3 для адаптивного дизайну, Select2 4.1 для інтерактивного вибору тегів, SQLite як бази даних із можливістю масштабування. Забезпечено підтримку української та англійської мов, що робить систему доступною для ширшої аудиторії. Лістинг коду наведено у додатках, зокрема фрагмент функції search_results, який демонструє фільтрацію та сортування.

Проведено тестування системи за допомогою модульного, інтеграційного, функціонального, продуктивнісного та usability-тестування. Результати підтвердили стабільність роботи: час завантаження роботи – 0,5 секунди, час пошуку – 0,8 секунди, 80% користувачів відзначили зручність інтерфейсу. Проте при моделюванні 1000 робіт час обробки запиту зростає до 3 секунд, що вказує на потребу в оптимізації.

Сформульовано рекомендації та перспективи: впровадити індексацію бази даних і кешування (Redis) для підвищення продуктивності, додати аналітичний модуль для відстеження популярності робіт, інтегрувати автоматичну модерацію з AI для прискорення обробки матеріалів.

Наукові результати:

– проведено класифікацію підходів до створення репозитаріїв (за масштабом, автоматизацією, технологіями), що може бути використано для подальших досліджень у сфері інформаційних систем;

– розроблено модель репозитарію з урахуванням локальних потреб ЗВО, зокрема модерації та локалізації.

Практичні результати:

– створено працюючий вебдодаток, який забезпечує зручне завантаження, пошук, фільтрацію та модерацію наукових робіт;

– реалізовано інтуїтивно зрозумілий інтерфейс із підтримкою двох мов.

Рекомендації щодо використання:

– впровадити систему у ЗВО для централізованого доступу до наукових матеріалів;

– використовувати розроблену модель як основу для створення подібних систем в інших закладах;

– продовжити оптимізацію продуктивності та розширення функціоналу для масштабування.

Перспективи:

– інтеграція з глобальними базами даних (наприклад, Crossref) для розширення локалізації на інші мови для міжнародного використання.

Таким чином, розроблений репозитарій наукових робіт вирішує проблему централізованого доступу до матеріалів у ЗВО, демонструє стабільну роботу та має потенціал для подальшого розвитку.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

URL: https://docs.djangoproject.com/ (accessed 05.06.2025).

m

i r

o 4. K

l w 5. L

n m 6. Bootstrap Documentation. URL: https://getbootstrap.com/ (accessed 0 03.06.2025).

i T 7. D

S s 8. J

R R 9. F

l J 10. Select2 Documentation. URL: https://select2.org/ (accessed 04.06.2025).

R v 11. SQLite Documentation. URL: https://www.sqlite.org/docs.html (accessed 0 04.06.2025).

M s 12. Sommerville I. Software Engineering. 10th ed. Pearson, 2015. 816 p.

n a 13. M

l y

K M

C

s

N g

M

L

D

D

n

M

6

d

h

M g