UNICHECK

Ім'я користувача:
приховано налаштуваннями конфіденційності

ID перевірки:
1015521899

Дата перевірки:
09.06.2023 10:27:04 EEST

Тип перевірки:
Doc vs Library

Дата звіту:
09.06.2023 11:16:41 EEST

ID користувача:
100011372

Назва документа: **Rudkevych 1-3 Rozdil**

Кількість сторінок: **40** Кількість слів: **6466** Кількість символів: **47085** Розмір файлу: **2.44 MB** ID файлу: **1015176027**

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

5.49%
Схожість

Найбільша схожість: **2.72%** з джерелом з Бібліотеки (ID файлу: **1011466283**)

Пошук збігів з Інтернетом не проводився

5.49% Джерела з Бібліотеки | **61** ..... Сторінка 42

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Замінені символи | **2**

Підозріле форматування | **8 сторінок**

1 ОСНОВНІ ТЕОРЕТИЧНІ ВІДОМОСТІ ПРО СТАТИЧНІ ВЕБ-СТОРІНКИ НА ПЛАТФОРМІ GITHUB

1.1 Поняття статичних веб-сторінок

Створення статичних сторінок є важливою складовою процесу розробки веб-сайтів та веб-додатків. Markdown, Jupyter Notebooks та HTML - це формати, які можуть бути використані для створення статичних сторінок на веб-сайті.

Markdown - це легка мова розмітки, яка використовується для форматування тексту. Вона зазвичай використовується при створенні контенту для Інтернету, наприклад, дописів у блогах, документації та онлайн-форумів. Markdown використовує простий синтаксис для форматування тексту, наприклад, зірочки для виділення (наприклад, курсив), подвійні зірочки для виділення тексту (наприклад, напівжирний) і хеш-символи для створення заголовків (# Заголовок 1, ## Заголовок 2 і т.д.). Інші функції включають створення списків (впорядкованих або невпорядкованих), посилань, зображень і блоків коду. Markdown можна легко конвертувати в HTML, який є мовою, що використовується для створення веб-сторінок. Багато текстових редакторів та онлайн-платформ підтримують Markdown, що робить його популярним вибором для написання контенту для Інтернету.

Розмітка часто використовується в поєднанні з іншими інструментами, такими як статичні генератори сайтів або системи управління контентом, для створення і публікації контенту в Інтернеті. Він також широко використовується в інструментах для спільної роботи, таких як GitHub, де користувачі можуть легко ділитися та співпрацювати над документами у форматі Markdown.

Jupyter Notebook - це інтерактивне середовище для програмування, яке дозволяє користувачам створювати та ділитися документами, що містять живий код, різноманітний текст, форматування, графіки та інші типи мультимедійних вмісту. Jupyter Notebook надає можливість працювати з багатьма мовами програмування, включаючи Python, R, Julia, а також інші.

Одна з найбільш корисних рис Jupyter Notebook полягає у тому, що він дозволяє користувачам виконувати код в окремих комірках, що дозволяє легко тестувати та налагоджувати код. Крім того, Jupyter Notebook надає можливість зберігати результати виконання коду, що полегшує повторне використання та підготовку звітів.

Jupyter Notebook став дуже популярним серед вчених даних та інших спеціалістів, які працюють з даними, оскільки він дозволяє легко візуалізувати та аналізувати дані в одному місці. Крім того, Jupyter Notebook надає можливість створювати інтерактивні документи, що дозволяє користувачам взаємодіяти зі своїми даними, аналізувати їх та робити висновки. Також, Jupyter Notebook підтримує багато різноманітних бібліотек та інструментів, що дозволяють виконувати різні операції з даними, включаючи машинне навчання, наукові обчислення, обробку мови та багато іншого.

Jupyter Notebook надає можливість легко ділитися своїми документами з іншими користувачами, які можуть переглядати та взаємодіяти з ними.

HTML (Hypertext Markup Language) - це код, який використовується для структурування та відображення веб-сторінки та її контенту. Наприклад, контент може бути структурований всередині множини параграфів, маркованих списків або з використанням зображень та таблиць даних. Як видно з назви, ця стаття дасть вам базове розуміння HTML та його функцій.

HTML не є мовою програмування; це мова розмітки і використовується, щоб повідомляти вашому браузеру, як відображати веб-сторінки, які ви відвідуєте. Він може бути складним або простим, залежно від того, як веб-дизайнер хоче. HTML складається з ряду елементів , які ви використовуєте, щоб вкладати або обертати різні частини контенту, щоб змусити контент відображатися або діяти певним чином. Захищаючі теги можуть зробити слово або зображення посиланням на щось ще, можуть зробити слова курсивом, зробити шрифт більшим або меншим і так далі.

Для створення статичних сторінок з використанням Markdown, Jupyter Notebooks та HTML, вам потрібно створити відповідний файл з розширенням ".md", ".ipynb" або ".html" відповідно. Для форматування тексту у Markdown

використовуються спеціальні символи, такі як # для заголовків та * для списків. Для Jupyter Notebooks ви можете використовувати вбудований редактор коду для додавання кодових блоків та створення візуалізацій даних. HTML вимагає використання тегів для визначення різних елементів. Після створення статичних сторінок, їх можна зберегти в репозиторії на GitHub та використовувати GitHub Pages для їх публікації в Інтернеті. GitHub Pages - це безкоштовний сервіс веб-хостингу, який дозволяє розміщувати статичні веб-сторінки на GitHub. Для того, щоб використовувати GitHub Pages, потрібно налаштувати основну гілку вашого репозиторію як основу для вашого сайту та вибрати тему для веб-сайту.

Після налаштування GitHub Pages, ви можете переглянути ваш сайт в Інтернеті, використовуючи URL-адресу, що вказана у вкладці "Settings" вашого репозиторію. Зміни, які ви робите у ваших статичних сторінках, будуть автоматично оновлюватися на вашому веб-сайті після коміту до відповідного файлу.

Щоб спростити процес створення статичних сторінок та автоматизувати їх оновлення, можна використовувати GitHub Actions. GitHub Actions - це інструмент автоматизації розробки, який дозволяє автоматично виконувати дії при виконанні певних подій, таких як коміти до репозиторію або створення Pull Request.

За допомогою GitHub Actions, ви можете автоматично створювати нові статичні сторінки з використанням Markdown, Jupyter Notebooks та HTML, та розміщувати їх на вашому веб-сайті GitHub Pages. Наприклад, ви можете використовувати GitHub Actions для автоматичного виконання скрипту, який зберігає ваші Jupyter Notebooks у форматі HTML та додає їх до вашого репозиторію. Це дозволяє вам оновлювати ваш веб-сайт автоматично, не виконуючи рутинні завдання вручну.

1.2 Система контролю версій Git

Git - це система контролю версій (VCS), що використовується для відстеження змін у програмному коді та спільної роботи над проектами з допомогою Інтернету.

Система управління версіями відстежує історію змін у процесі спільної роботи з проектами. У міру того, як розробники вносять зміни до проекту, у будь-який час можна відновити будь-яку більш ранню його версію.

Розробники можуть переглядати журнал проекту, щоб дізнаватися таке: які зміни було внесено;

1. Хто вніс зміни;
2. Які зміни було внесено;
3. Навіщо були потрібні зміни.

Системи управління версіями надають кожному учаснику єдине, узгоджене подання проекту з можливістю відстежувати роботу, що вже ведеться. Можливість перегляду прозорої історії змін із зазначенням їх авторів та вкладу у розробку проекту допомагає учасникам команди працювати узгоджено незалежно один від одного.

У розподіленій системі управління версіями кожен розробник має повну копію проекту та його журналу. На відміну від популярних централізованих систем управління версіями, розподілені системи не вимагають постійного підключення до центрального репозиторію. GIT – це найпопулярніша розподілена система управління версіями. GIT широко застосовується для розробки як програмного забезпечення (ПЗ) з відкритим кодом, так і комерційного ПЗ, надаючи важливі переваги окремим розробникам, командам та компаніям.

GIT дозволяє розробникам переглядати всю тимчасову шкалу змін, рішень та ходу виконання будь-якого проекту в одному місці. З моменту доступу до історії проекту розробник отримує всю необхідну інформацію, щоб розібратися в проекті та почати робити свій внесок.

Розробники працюють у різних часових поясах. Завдяки розподіленій системі керування версіями, як-от GIT, спільну роботу можна вести у будь-який

час, зберігаючи цілісність вихідного коду. За допомогою гілок розробники можуть безпечно пропонувати зміни для робочого коду.

Організації, що використовують GIT, можуть усунути комунікаційні бар'єри між командами, щоб вони могли зосередитися на виконанні своїх завдань з максимальною ефективністю. Крім того, GIT дозволяє узгодити спільну роботу експертів над великими проектами у масштабі всього підприємства.

Основні команди Git

Працюючи з GIT, розробники використовують певні команди для копіювання, створення, зміни та об'єднання коду. Ці команди можна виконувати безпосередньо з командного рядка або за допомогою програми, наприклад GitHub Desktop. Нижче наведено деякі поширені команди для роботи з GIT.

git init — ініціалізує новий репозиторій GIT та починає відстеження існуючого каталогу. До існуючого каталогу додається прихована вкладена папка, в якій розміщується внутрішня структура даних, необхідна для керування версіями.

git clone — створює локальну копію проекту, який існує віддалено. Клон включає всі файли проекту, журнал і гілки.

git add - Готує зміну. GIT відстежує зміни в базі коду розробника, але для включення змін до журналу проекту необхідно готувати їх та створювати моментальні знімки. Ця команда виконує першу частину цього двоетапного процесу, тобто підготовку. Усі підготовлені зміни стануть частиною наступного моментального знімка та журналу проекту. Роздільна підготовка та фіксація дають розробникам повний контроль над історією проекту без необхідності змінювати підхід до написання коду та роботи в цілому.

git commit — зберігає моментальний знімок у журналі проекту та завершує процес відстеження змін. Інакше висловлюючись, фіксація схожа створення фотографії. Все, що було підготовлено за допомогою команди git add, стане частиною моментального знімка під час використання git commit.

git status - виводить стан змін: не відстежуються, змінені чи підготовлені.

git branch - Показує гілки, з якими ведеться локальна робота.

git merge - Виконує злиття ліній розробки. Ця команда зазвичай застосовується для поєднання змін, внесених у двох різних гілках. Наприклад, розробник виконує злиття, коли необхідно поєднати зміни з гілки функції з головною гілкою для розгортання.

git pull — застосовує до локальної лінії розробки оновлення віддалений аналог. Розробники використовують цю команду, якщо колега виконав фіксації у галузі віддаленого репозиторію і ці зміни потрібно відобразити в локальному середовищі.

git push - оновлює віддалений репозиторій з урахуванням фіксацій, виконаних у гілки локально.

1.3 Платформа GitHub

GitHub служить для розміщення репозиторіїв GIT і надає розробникам засоби для поставки якіснішого коду: функції командного рядка, проблеми (ланцюжки обговорень), запити на витягування, перевірка коду та колекція безкоштовних і платних додатків у GitHub Marketplace. Завдяки таким рівням спільної роботи, як потік GitHub, спільнота з 15 мільйонів розробників та екосистема, що включає сотні інтеграцій, GitHub змінює підхід до створення програмного забезпечення.

GitHub дозволяє інтегрувати спільну роботу безпосередньо у процес розробки. Робота організована по репозиторіям, у яких розробники можуть встановлювати вимоги чи давати вказівки учасникам команди. Потім, використовуючи потік GitHub, розробники просто створюють гілку для роботи з оновленнями, фіксують зміни, щоб зберігати їх, відкривають запити на витягування, щоб пропонувати та обговорювати зміни, та виконують злиття запитів на витягування після їх узгодження

Відомості про репозиторії GIT

Репозиторій або проект GIT включає повний набір файлів і папок, пов'язаних з проектом, а також журнал змін кожного файлу. Журнал файлу представлений як моментальні знімки на певні моменти часу. Ці знімки

називаються фіксаціями. Фіксації можна впорядковувати за кількома лініями розробки, які називаються гілками. Оскільки GIT — розподілена система управління версіями, репозиторії є автономними одиницями і будь-який користувач, який має копію репозиторію, може отримувати доступ до всієї бази коду та її історії. За допомогою командного рядка або інших зручних інтерфейсів можливі також наступні дії з репозиторієм GIT: взаємодія з журналом, клонування репозиторію, створення гілок, фіксація, злиття, порівняння змін у різних версіях коду та багато іншого.

За допомогою таких платформ, як GitHub, GIT також надає додаткові можливості для забезпечення прозорості проектів та спільної роботи. Загальнодоступні репозиторії допомагають командам працювати разом створенням максимально якісного кінцевого продукту.

GitHub Pages - це безкоштовний сервіс, який надається GitHub для створення та розміщення статичних веб-сайтів. Цей сервіс дозволяє розмістити ваші веб-сторінки безкоштовно на хостингу, що належить GitHub.

Основні особливості GitHub Pages:

1. Безкоштовний: Використання GitHub Pages не потребує жодної плати.
2. Статичні веб-сайти: GitHub Pages підтримує лише статичні веб-сайти, тобто такі, що складаються з HTML, CSS та JavaScript файлів. Сервер віддає збережені файли як відповідь на запит клієнта.
3. Простота використання: GitHub Pages легко налаштовувати, і ви можете швидко розпочати роботу з веб-сайтом.
4. Документація: GitHub Pages має добру документацію, що допоможе вам зрозуміти, як цей сервіс працює.
5. Різні варіанти налаштування: Ви можете розміщувати свій сайт на GitHub Pages через веб-інтерфейс або за допомогою Git. Ви можете використовувати створені шаблони, створювати свої власні шаблони, додавати власні файли та інші опції налаштування.

6. Можливості розробки: GitHub Pages може бути використаний як платформа для розробки веб-сайту, в якому ви можете використовувати багато зручних інструментів, які надає GitHub.

7. Підтримка Jekyll: GitHub Pages підтримує Jekyll, що є генератором статичних веб-сайтів на основі Ruby. Ви можете використовувати Jekyll для швидкої розробки веб-сайту та використовувати його на GitHub Pages.

8. Доменні імена: Ви можете налаштувати свій власний домен для веб-сайту на GitHub Pages. Це можна зробити шляхом налаштування DNS вашого домену, щоб вказати, що ваш веб-сайт розміщується на GitHub Pages.

9. Система контролю версій: GitHub Pages заснований на системі контролю версій Git, що дозволяє зберігати історію змін вашого веб-сайту та зберігати копії ваших файлів на GitHub.

10. Безпека: GitHub Pages забезпечує захист від атак та злому за допомогою HTTPS та SSL-шифрування. Крім того, GitHub Pages автоматично застосовує відповідні заголовки безпеки для захисту вашого веб-сайту.

GitHub Pages є відмінним вибором для розміщення статичних веб-сайтів. Цей сервіс надає безкоштовний та простий спосіб створити та розмістити ваш веб-сайт в Інтернеті, не потребуючи від вас додаткових витрат та складних процесів.

GitHub Actions - це функціональність платформи GitHub, яка дозволяє автоматизувати рутинні процеси, пов'язані з розробкою програмного забезпечення, тестуванням, збіркою та розгортанням.

GitHub Actions дозволяє створювати та налаштовувати автоматичні робочі процеси, які можна виконувати на вказаних подіях в репозиторії. Наприклад, ви можете налаштувати автоматичне тестування коду після кожного злиття (merge) з гілкою master або виконувати збірку та розгортання додатку на віддаленому сервері після злиття з гілкою production.

GitHub Actions дозволяє використовувати готові шаблони, які забезпечують стандартні дії, такі як збірка коду, запуск тестів, розгортання на хмарні платформи, надсилання повідомлень на електронну пошту або

месенджери, та інші. Ви можете налаштувати власні дії, використовуючи скрипти на мові програмування, яка підтримується GitHub.

Основні переваги GitHub Actions:

1. Простота використання: GitHub Actions забезпечує простоту налаштування та використання.
2. Інтеграція з GitHub: GitHub Actions є частиною платформи GitHub і повністю інтегровані зі змінами, які відбуваються в репозиторії.
3. Безкоштовність: GitHub Actions безкоштовні для відкритих проектів та дозволяють запускати до 2000 хвилин робочого часу на місяць.
4. Широкий вибір дій: GitHub Actions дозволяє виконувати різні дії на основі подій, які відбуваються в репозиторії.

GitHub Actions допомагає збільшити ефективність процесу розробки програмного забезпечення, скорочує час виконування та дозволяє уникнути помилок, пов'язаних з ручним виконанням рутинних операцій. За допомогою GitHub Actions можна налаштувати автоматичне тестування коду на різних платформах та в різних середовищах, що дозволяє забезпечити високу якість продукту. Крім того, використання GitHub Actions сприяє автоматизації процесу збірки та розгортання додатків, що знижує час, необхідний для публікації нових версій.

Окрім цього, GitHub Actions дозволяє налаштовувати власні дії, що робить платформу більш гнучкою та дозволяє забезпечити повну автоматизацію всіх процесів, пов'язаних з розробкою програмного забезпечення. Для створення власних дій можна використовувати мови програмування, такі як JavaScript, Python, Ruby та інші.

Крім того, GitHub Actions підтримує інтеграцію з різними сервісами, такими як Docker, Amazon Web Services, Microsoft Azure, Google Cloud Platform та інші. Це дозволяє використовувати різноманітні сервіси та інструменти для автоматизації процесу розробки.

Можна налаштувати робочий процес GitHub Actions так, щоб він запускався, коли у сховищі відбувається подія, наприклад, відкривається запит на витяг або створюється проблема. Робочий процес містить одне або кілька

завдань, які можуть виконуватися в послідовному порядку або паралельно. Кожне завдання виконуватиметься всередині власної віртуальної машини або всередині контейнера і має один або кілька кроків, які запускають визначений сценарій або дію, що може спростити ваш робочий процес. GitHub Actions дає змогу створювати власні робочі процеси життєвого циклу розробки програмного забезпечення (SDLC) безпосередньо у сховищі GitHub. Потрібно налаштувати дії GitHub за допомогою синтаксису YAML і зберегти їх як файли робочого процесу у сховищі.

YAML (скорочення від "YAML Ain't Markup Language") - це мова серіалізації даних, придатна для читання людиною. Її часто використовують у конфігураційних файлах, форматах обміну даними та інших програмах, де дані потрібно зберігати або передавати у форматі, який легко читати і записувати.

YAML розроблена таким чином, щоб її було легко читати і писати, а синтаксис використовує відступи і прості символи, такі як двокрапки і тире, для визначення структур даних. Вона часто використовується у фреймворках веб-розробки, таких як Ruby on Rails і Django, а також в інструментах управління конфігурацією, таких як Ansible і SaltStack.

Деякі з ключових особливостей YAML включають

Простий синтаксис: YAML використовує відступи і прості символи, такі як двокрапка і тире, для визначення структур даних, що полегшує читання і запис.

Підтримка складних структур даних: YAML підтримує списки, словники та інші складні структури даних, що робить її придатною для зберігання та передачі даних будь-якого типу.

Зручний для читання людиною та машинного аналізу: Файли YAML легко читаються і розуміються людиною, а також можуть аналізуватися машинами.

Розширюваність: YAML можна розширювати за допомогою спеціальних тегів і директив, що дозволяє адаптувати його до різних випадків використання.

YAML часто використовується разом з іншими мовами програмування та інструментами, такими як Python, Ruby та Ansible, для визначення параметрів конфігурації, структур даних та іншої важливої інформації.

Включення коментарів: YAML дозволяє розробникам додавати коментарі до свого коду, щоб пояснити, що робить код, або надати контекст для інших розробників, які можуть працювати над тією ж самою кодовою базою.

Перевірка даних: YAML підтримує використання схем для перевірки даних і гарантує, що вони відповідають певному формату або структурі.

Серіалізація даних: YAML можна використовувати для серіалізації даних, що означає, що вони можуть бути перетворені у формат, який можна легко передавати або зберігати. Це робить її популярним вибором для веб-додатків і форматів обміну даними.

Мовна діагностика: YAML не прив'язаний до будь-якої конкретної мови програмування або платформи, що робить його гнучким вибором для розробників, які працюють з декількома мовами або потребують обміну даними між різними системами.

Конфігураційні файли: YAML зазвичай використовується в конфігураційних файлах додатків, таких як налаштування підключення до бази даних, налаштування сервера та специфічні параметри конфігурації додатків.

Тестові дані: YAML можна використовувати для визначення тестових даних для фреймворків автоматизованого тестування, що робить його корисним інструментом для розробників, яким потрібно швидко і легко створювати великі обсяги тестових даних.

Загалом, YAML - це універсальна і проста у використанні мова, яка добре підходить для широкого спектру випадків використання, від конфігураційних файлів до форматів обміну даними і генерації тестових даних. Простий синтаксис і зручний для читання формат роблять її популярною серед розробників, яким потрібен швидкий і ефективний спосіб зберігання та передачі даних.

Робочі процеси — це спеціальні автоматизовані процеси, які можна налаштувати у сховищі для створення, тестування, упаковки, випуску або розгортання будь-якого проекту на GitHub. Є можливість писати окремі завдання, які називаються діями, і об’єднувати їх, щоб створити індивідуальний робочий процес. Після того, як успішно створено файл робочого процесу YAML

і запущено робочий процес, з'являються журнали збірки, результати тестів, артефакти та статуси для кожного кроку робочого процесу. Його можна налаштувати для створення проекту на різних платформах розробки.

Використання GitHub Actions є особливо корисним для команд розробників, які працюють з великими проектами та потребують постійної підтримки та автоматизації процесу розробки. Для роботи з GitHub Actions не потрібно мати додаткових знань або навичок, оскільки це простий та інтуїтивно зрозумілий інструмент, який дозволяє автоматизувати будь-які процеси розробки програмного забезпечення.

2. НАЛАШТУВАННЯ СЕРЕДОВИЩА РОЗРОБКИ СТАТИЧНИХ WEB-СТОРІНОК НА ПЛАТФОРМІ GITHUB

2.1 Intelige Idea

IntelliJ IDEA (часто називають просто "IntelliJ") - це інтегроване середовище розробки (IDE), розроблене компанією JetBrains, що базується в Празі, Чеська Республіка. IntelliJ IDEA покликана допомогти розробникам програмного забезпечення писати високоякісний код швидше та ефективніше. Вона підтримує широкий спектр мов програмування, включаючи Java, Kotlin, Python, Ruby та JavaScript.

Ось деякі з основних можливостей IntelliJ IDEA:

Аналіз коду та навігація: IntelliJ IDEA включає потужні інструменти аналізу коду та навігації, які можуть допомогти розробникам швидко знаходити та виправляти помилки у своєму коді. Він може виділити потенційні проблеми та запропонувати рефакторинг коду для покращення якості коду.

Завершення коду: IntelliJ IDEA забезпечує інтелектуальне завершення коду, яке пропонує завершення коду на основі поточного контексту, що робить написання коду швидшим і простішим для розробників.

Інтегрований контроль версій: IntelliJ IDEA включає вбудовану підтримку популярних систем контролю версій, таких як Git, Subversion та Mercurial. Це дозволяє розробникам легко керувати змінами коду та співпрацювати з іншими розробниками над однією і тією ж кодовою базою.

Інструменти для тестування та налагодження: IntelliJ IDEA включає в себе різноманітні інструменти тестування та налагодження, які полегшують написання та тестування коду. Наприклад, вбудований відладчик, який дозволяє розробникам переглядати свій код і швидше виявляти проблеми.

Інструменти рефакторингу: IntelliJ IDEA включає в себе різноманітні інструменти рефакторингу коду, які можуть допомогти розробникам покращити якість та ремонтопридатність їх коду. Сюди входять такі функції, як перейменування змінних, вилучення методів та оптимізація імпорту.

Екосистема плагінів: IntelliJ IDEA має велику екосистему плагінів, яка дозволяє розробникам розширювати функціональність системи та налаштовувати її відповідно до своїх потреб. Існують плагіни для широкого спектру випадків використання, таких як розробка баз даних, веб-розробка та мобільна розробка.

Загалом, IntelliJ IDEA - це потужне та гнучке середовище розробки, яке широко використовується розробниками по всьому світу. Широкий набір функцій та підтримка багатьох мов програмування роблять її популярним вибором для розробників, які працюють над різноманітними проектами.

Для завантаження Intelig Idea потрібно зайти на офіційний сайт JetBrains і завантажити Community Edition так як вона є безкоштовною, зображено на рисунку 2.1.

Рисунок 2.1 − Сайт Intelige Idea

Далі нам необхідно обрати шлях, де буде встановлено програму Intelige. Найчастіше встановлюваний шлях за замовчуванням знаходиться в папці "Program Files" на диску C: вашого комп'ютера. Однак, у залежності від наших потреб і налаштувань, ми можемо вибрати будь-який інший шлях для встановлення програми. Зверніть увагу, що шлях повинен бути зазначений з

урахуванням прав доступу, тому, можливо, знадобиться запустити програму в режимі адміністратора, щоб мати необхідні дозволи на встановлення.

Шлях, який ми вибрали для встановлення Intelige, зображений на рисунку 2.2.

Рисунок 2.2 − Вибір папки встановлення

Після вибору шляху для встановлення програми Intelige, ми побачимо наступне вікно, що дозволяє налаштувати інсталяцію. Це дуже важлива частина процесу встановлення, оскільки від налаштувань залежить подальше використання програми. На цьому етапі ми можемо вибрати, які компоненти програми ми хочемо встановити, які додаткові опції встановлення ми хочемо включити. На рисунку 2.3 зображено налаштування інсталяції Intelige. Перед продовженням встановлення програми, переконайтеся, що ви налаштували установку згідно своїх потреб і вимог.

Рисунок 2.3 − Налаштування інсталяції

Вибір папки для інсталяції програми Intelige є однією з найважливіших опцій, що користувачі повинні зробити під час процесу встановлення. Залежно від наших потреб та умов роботи з програмою, може бути вигіднішим вибрати одну чи декілька папок для збереження програмних файлів.

Якщо ми вирішимо вибрати відповідну папку в меню "пуск", то ми зможемо швидко знайти і запустити програму, не витрачаючи часу на пошук папки в провіднику, зображено на рисунку 2.4.

Рисунок 2.4 − Вибір папки в меню пуск

2.2 Реєстрація на GitHub

Основні кроки реєстрації на Github наступні:

Щоб мати можливість користуватися сервісом Github, потрібно створити свій обліковий запис на сайті. Для цього перейдіть на сайт github.com і знайдіть кнопку "Sign up", яка розміщена в правому верхньому куті сторінки.

Натисніть на цю кнопку, щоб перейти на сторінку реєстрації. Тут ви повинні буде ввести свій електронний адрес та створити пароль, який буде захищати ваш обліковий запис від несанкціонованого доступу, зображено на рисунку 2.5

Рисунок 2.5 − Ввід пошти і паролю на GitHub

Після цього потрібно вказати своє ім'я користувача (username), яке буде використовуватися для створення посилання на ваш профіль на GitHub, ім'я користувача на Github може складатися з будь-якої комбінації літер, цифр та деяких спецсимволів, але не може містити пробіли. Воно повинно бути унікальним і не повторюватися з іншими іменами користувачів на Github, зображено на рисунку 2.6

Рисунок 2.6 − Ввід ім’я користувача

Після введення ім'я користувача на Github, що буде використовуватися для створення посилання на ваш профіль, вам потрібно буде пройти тест на робота, зображений на рисунку 2.7. Цей тест допомагає підтвердити, що ви людина, а не комп'ютерна програма або бот, що намагається створити фальшивий профіль на платформі. У тесті на робота вам будуть показані різні зображення і вам потрібно буде відзначити ті, на яких зображені конкретні об'єкти або сцени, наприклад, автомобілі, магазини або пейзажі. Це захистить ваш обліковий запис від несанкціонованого доступу та допоможе зберегти інформацію про ваші проекти в безпеці на GitHub.

Рисунок 2.7 − Тест на робота

Після успішного завершення тесту на робота та введення електронної пошти, вам буде надіслано лист з підтвердженням створення облікового запису

на GitHub. У ньому буде посилання, яке потрібно буде перейти, щоб підтвердити створення облікового запису. Після підтвердження облікового запису вам буде запропоновано обрати тип акаунту - безкоштовний (Free) або платний (Pro). Якщо ви обрати платний акаунт, то вам буде запропоновано вказати спосіб оплати. Це може бути здійснено шляхом введення реквізитів банківської карти або через платіжну систему, яка підтримується на GitHub. Зображення процесу вибору типу акаунту та способу оплати можна побачити на рисунку 2.8. Пам'ятайте, що безкоштовний акаунт дозволяє використовувати базовий функціонал GitHub, але платний акаунт може давати більше можливостей та функціоналу для вашого профілю.

Рисунок 2.8 − Вибір типу акаунту

2.3 З’єднання Github і Intelige Idea

Основні кроки, як з'єднати GitHub і Intelige Idea за допомогою токену, наступні:

Для створення особистого токена доступу на GitHub вам потрібно буде дотриматися кількох кроків. По-перше, вам потрібно буде увійти в свій акаунт на GitHub за допомогою свого імені користувача та пароля. Після успішного входу в систему, ви побачите свій особистий кабінет на GitHub. Далі потрібно

перейти до налаштувань вашого профілю, для чого виберіть опцію "Settings". Це меню знаходиться у верхньому правому куті екрану, як зображено на рисунку 2.9.

Рисунок 2.9 – Налаштування профілю

У розділі "Settings" ви знайдете багато різних опцій для налаштування свого профілю та контролю вашої активності на GitHub. Серед цих налаштувань ви знайдете опцію "Developer settings", де ви зможете створити токен доступу до вашого акаунту, зображено на рисунку 2.10.

Рисунок 2.10 − Створення токену

Після вибору цієї опції, ви побачите кнопку "Generate new token", яку потрібно натиснути, щоб створити новий токен доступу. Далі, вам потрібно буде вказати деякі налаштування для токену, такі як обмеження доступу, строк дії токену та інші параметри. Після цього ви зможете зберегти створений токен та використовувати його для доступу до різних ресурсів на GitHub, зображено на рисунку 2.11.

Рисунок 2.11 − Налаштування токену

Після того, як ви успішно створили особистий токен доступу на GitHub, його можна побачити в розділі "Personal access tokens" (Особисті токени доступу) на сторінці "Settings". Токен зображено в таблиці разом з описом його прав доступу. Крім того, на цій сторінці ви можете редагувати свій токен, видаляти його та створювати нові. Згенерований токен дозволяє здійснювати різноманітні дії на GitHub, такі як створення нових репозиторіїв, додавання коментарів, відправлення запитів на злиття (pull requests) та інше. Використання токенів доступу є безпечним способом забезпечення вашого аккаунту на GitHub

і зменшення ризиків несанкціонованого доступу до ваших проектів, зображено на рисунку 2.12.

Рисунок 2.12 − Згенерований токен

Щоб налаштувати IntelliJ IDEA для використання з GitHub, вам потрібно відкрити програму та вибрати "Configure" у початковому вікні. Потім вам потрібно вибрати "Settings", що відкриє вікно налаштувань, зображено на рисунку 2.13.

Рисунок 2.13 − Налаштування

У цьому вікні вам необхідно вибрати пункт меню "Version Control" і потім обрати GitHub у списку доступних систем контролю версій. Далі вибрати опцію

"GitHub", яка дозволяє налаштувати доступ до репозиторіїв на GitHub, які будуть використовуватися для зберігання коду. Це зображено на рисунку 2.14.

Рисунок 2.14 – Контроль версій

Після того, як ми обрали "GitHub" у пункті "Version Control" в налаштуваннях IntelliJ IDEA, ми потрапляємо до наступного вікна. Тут ми бачимо список наших GitHub акаунтів, які ми можемо підключити до нашого проекту. Для того, щоб додати нового користувача, необхідно натиснути на кнопку «плюс», яка знаходиться зліва від списку. Це дозволить додати новий GitHub акаунт, з яким ми можемо працювати у нашому проекті. Даний процес зображено на рисунку 2.15.

Рисунок 2.15 – Додавання нового користувача

Після того як ви вибрали "GitHub" у налаштуваннях IntelliJ IDEA, вам буде запропоновано вибрати спосіб авторизації на GitHub. Оберіть опцію "Log in with Token", яка зображена на рисунку 2.16. Ця опція дозволяє авторизуватися в GitHub за допомогою токена доступу, який ви створили раніше.

Рисунок 2.16 – Вибір підключення

Після вибору "Log in with Token" відкриється вікно для введення токену доступу. Необхідно вставити токен, який ви створили раніше, в поле "Token". Потім натисніть кнопку "Log In", щоб зберегти налаштування.

Це дозволить IntelliJ IDEA використовувати ваш особистий токен доступу GitHub для підключення до вашого облікового запису, зображено на рисунку 2.17.

Рисунок 2.17 – Вставка токен

Після виконання усіх попередніх кроків, ми можемо почати комфортно працювати з GitHub. Зокрема, ми можемо створювати нові репозиторії, редагувати вже існуючі, додавати та видаляти файли, створювати нові гілки та об'єднувати їх, коментувати та переглядати код, а також багато іншого.

2.4 Перевірка роботи середовища

Для перевірки роботи нашого середовища розробки та забезпечення коректності додавання файлів до репозиторію, ми створимо файл index.html та додамо його до нашого репозиторію на GitHub.

Щоб створити файл index.html в IntelliJ, слід виконати наступні кроки:

Клацнути правою кнопкою миші на папку, в якій буде створюватись файл.

В контекстному меню вибрати пункт "New" > "HTML File", зображено на рисунку 2.18.

Рисунок 2.18 − Створення html файлу

Для того щоб дати назву у вікні "New HTML File" ввести назву файлу index.html, зображено на рисунку 2.19.

Рисунок 2.19 − Названия файлу

Після створення index.html файлу і його збереження на комп'ютері, для додавання його до репозиторію на GitHub потрібно виконати наступні кроки:

1. В Intelige зайти в меню "Git" та вибрати "Commit File", зображено на рисунку 2.20.

Рисунок 2.20 − Commit

2. У вікні "Commit" ввести опис змін, які ви внесли до файлу index.html, зображено на рисунку 2.21

Рисунок 2.21 – Опис коміту

3. Вибрати опцію "Commit and Push" для того, щоб запушити коміт на GitHub.

Після успішного завантаження змін на сервер, можна перевірити, чи дійсно вони з'явилися на сторінці репозиторію на GitHub, зображено на рисунку 2.22.

Рисунок 2.22 – Доданий файл на GitHub

Таким чином, за допомогою Git та Intelige ми можемо з легкістю додавати нові файли до нашого репозиторію на GitHub та контролювати версію нашого коду.

3 СТВОРЕННЯ СТАТИЧНОЇ ВЕБ-СТОРІНКИ

3.1 Встановлення та налаштування mkdocs-material

Для виконання даної дипломної роботи я використовуватиму MKDocs Material, одну з найпопулярніших тем інтерфейсу для MKDocs. MKDocs Material є розширенням для MKDocs, яке надає красивий і сучасний дизайн вашої документації. Завдяки його функціональності і стилізації, MKDocs Material дозволяє зосередитися на створенні високоякісної документації, що легко читається та навігується.

За допомогою MKDocs Material ви зможете створювати красиві сторінки документації з використанням зручного розміщення контенту, зручного меню навігації та можливостей налаштування теми. Це дозволяє зосередитися на важливому - створенні змісту вашої дипломної роботи, не витрачаючи час на складання складних структур або дизайну.

Крім того, MKDocs Material підтримує різні функціональні можливості, такі як пошук, темну та світлу теми, анімацію та інші елементи, які можна використовувати для поліпшення документації вашої дипломної роботи.

Загалом, MKDocs Material є потужним інструментом для створення стильної та професійної документації, який допоможе вам зробити вашу дипломну роботу більш доступною та зрозумілою для вашої аудиторії.

Для встановлення mkdocs-material на комп'ютері, потрібно спочатку встановити Python, pip і модуль venv (віртуальне середовище). Ось кроки, які ви можете виконати для кожного з них:

Встановлення Python:

1. Завантажте встановник Python з офіційного веб-сайту Python (https://www.python.org).
2. Виберіть потрібну версію Python для своєї операційної системи (наприклад, версія 3.11.3).

3. Запустіть встановник і включіть опцію "Add Python to PATH" (Додати Python до змінних середовища), зображено на рисунку 3.1 (на рисунку зображена не актуальна версія python).

Рисунок 3.1 − Інсталяція python і pip

Встановлення модуля venv:

Відкрийте командний рядок (на Windows це може бути командний рядок або PowerShell) або термінал (на Linux або macOS). Виконайте наступну команду для встановлення модуля venv за допомогою pip, приклад команди зображено на рисунку 3.2.

```
PS C:\Users\yurar\Desktop> python -m venv env
```

Рисунок 3.2 – Приклад команди python –m venv env

Для створення проекту на mkdocs-material і рекомендується використовувати віртуальне середовище (venv). Це дозволяє ізолювати залежності вашого проекту від інших проектів та забезпечити чисте середовище для розробки.

Ось кроки, які потрібно виконати для активації віртуального середовища та створення проекту на mkdocs-material:

Потрібно відкрити командний рядок (на Windows це може бути командний рядок або PowerShell) або термінал (на Linux або macOS).

Потрібно перейти до папки, де потрібно створити свій проект.

Потрібно активувати віртуальне середовище, приклад команди зображено на рисунку 3.3.

Рис 3.3 Приклад команди .\env\Scripts\activate

Після активації віртуального середовища потрібно встановити mkdocs та mkdocs-material, приклад зображено на рисунку 3.4.

Рисунок 3.4 – Приклад команди pip install mkdocs-material

Після встановлення можна створити новий проект mkdocs, приклад команди зображено на рисунку 3.5.

Рисунок 3.6 – Приклад команди mkdocs new my-project

3.2 Створення основної структури статичної веб-сторінки.

Файл mkdocs.yml є конфігураційним файлом для інструмента MkDocs, який дозволяє генерувати статичні сайти з використанням файлів Markdown. Цей файл містить налаштування проекту, такі як назва сайту, навігаційне меню, тема оформлення та інші параметри. Ось опис різних полів, які можна використовувати в файлі mkdocs.yml:

site_name: Це поле визначає назву вашого веб-сайту.

pages: Це поле визначає структуру сторінок вашого сайту та їх розташування у вигляді дерева. Ви можете вказувати шляхи до ваших Markdown файлів та їх назви.

nav: Це поле визначає навігаційне меню вашого сайту. Ви можете вказати різні сторінки та підсторінки, які будуть включені в меню.

theme: Це поле визначає тему оформлення вашого сайту. Ви можете вибрати вбудовану тему або встановити сторонню тему, зображено на рисунку 3.7.

Рисунок 3.7 – Mkdocs.yaml

index.md: Це головна сторінка вашого сайту, яка вітає відвідувачів і надає загальний огляд вашого проекту або теми. Вона може містити опис проекту, його цілі та основні функції. Також може бути посилання на розділи документації або інші важливі сторінки вашого сайту.

about.md: Ця сторінка призначена для розміщення детального опису вашого проекту або команди, яка працює над проектом. Вона може включати інформацію про цілі проекту, його історію, основні функціональні можливості або будь-яку іншу важливу інформацію, яку ви хочете поділитися з відвідувачами.

getting-started.md: Цей файл містить посібник або кроки, які потрібно виконати, щоб почати використовувати ваш проект. Він може містити інструкції

щодо установки та налаштування проекту, налаштування середовища розробки, підготовки базового коду або будь-які інші кроки, які потрібно виконати, щоб почати роботу з проектом.

usage.md: Ця сторінка надає детальну інформацію про використання вашого проекту або його функціональність. Вона може містити приклади коду, описи API, інструкції щодо використання різних функцій або детальніше роз'яснення концепцій, що використовуються у вашому проекті.

api.md: Цей файл описує API або інтерфейси програмування вашого проекту. Він містить інформацію про доступні ендпоінти, параметри, формати запитів та відповідей, а також опис різних функцій або класів, які можна використовувати при взаємодії з вашим проектом.

3.3 Наповнення документації для статичного сайту перегляду фільмів

У рамках розробки веб-сайту для дипломної роботи, одним з важливих аспектів є створення статичних сторінок. Для досягнення цієї мети використовується мова розмітки Markdown. Markdown є простим у використанні і зрозумілим форматом для написання контенту, який легко перетворюється в HTML.

Синтаксис Markdown

Markdown надає зручний і простий у використанні синтаксис для форматування тексту. Основні елементи синтаксису включають:

Заголовки: Заголовки в Markdown позначаються символами #. Заголовок першого рівня використовує один символ #, заголовок другого рівня - два символи ##, і так далі.

Списки: Markdown підтримує марковані списки, які позначаються символом - або *. Також можна використовувати номеровані списки, використовуючи цифри.

Форматування тексту: Для надання тексту жирного шрифту використовуються символи **текст**, для курсиву - символи *текст*, а для поєднання жирного шрифту з курсивом - символи ***текст***.

Посилання: Посилання в Markdown створюються за допомогою синтаксису [текст](URL). Текст в квадратних дужках відображається як посилання, а URL в круглих дужках вказує на адресу, на яку слід перейти при натисканні на посилання.

Переваги використання Markdown

Використання Markdown для наповнення статичних сторінок має декілька переваг:

Простота: Markdown має простий і легкий у використанні синтаксис, який швидко освоюється. Він не вимагає великої кількості коду для форматування тексту.

Читабельність: Markdown дозволяє легко створювати структурований та зрозумілий контент. Він надає можливість швидко виділити заголовки, списки, посилання та інші елементи, що полегшують сприйняття інформації.

Переносимість: Файли Markdown можна легко перетворити в HTML або інші формати для подальшого використання на веб-сайтах або в інших додатках. Markdown є стандартизованим форматом, що робить його переносимим між різними платформами та редакторами.

Ефективність: Використання Markdown дозволяє швидко створювати та редагувати статичні сторінки. Його легкість використання дозволяє зосередитись на наповненні контентом, замість витрачання часу на складний форматування.

На сторінці "Home" ви знайдете повний опис нашого проекту, включаючи використані технології, основні функції та іншу корисну інформацію. Ця сторінка призначена для того, щоб ви могли швидко і легко ознайомитись з основними аспектами нашого веб-сайту результат наповнення зображено на рисунку 3.8.

Рисунок 3.8 – Вкладка «Home»

На сторінці "About" ви знайдете інформацію про нашу команду та все те, що ми пропонуємо, зображено на рисунку 3.9.

Рисунок 3.9 – Вкладка «About»

На сторінці "Getting Started" ви знайдете кроки, необхідні для початку роботи з веб-сайтом, зображено на рисунку 3.10.

Рисунок 3.10 − Вкладка «Getting Started»

На сторінці "Usage" ми надаємо інструкції та приклади використання нашого продукту, зображено на рисунку 3.11

Рисунок 3.11 − Вкладка «Usage»

На сторінці "API" знаходиться інформація про використання MovieDB API для отримання даних про фільми.

Рисунок 3.12 − Вкладка «API»

3.4 Розгортання проекту на Github Pages

Для успішного розгортання проекту на GitHub Pages потрібно виконати декілька додаткових кроків:

1. Відкрити термінал або командний рядок та перейдіть до кореневої папки проекту mkdocs.
2. Виконати команду mkdocs build, щоб створити збірку проекту. Ця команда збере всі сторінки, ресурси та шаблони та згенерує вихідні файли, які будуть готові для розгортання. Приклад команди наведено на рисунку 3.13.

Рисунок 3.13 – Приклад команди mkdocs build

3. Після успішного завершення команди mkdocs build, у вас з'явиться нова папка з назвою site (або іншою, якщо ви налаштували іншу властивість site_dir у файлі mkdocs.yml), що містить згенеровану збірку, зображено на рисунку 3.14.

Рисунок 3.14 − Папка «site»

4. Перейдіть до репозиторію GitHub, на який ви хочете розгорнути свій сайт.

5. Завантажте всі файли з папки site у гілку main вашого репозиторію на GitHub. Ви можете це зробити через інтерфейс GitHub або використовуючи Git-команди.

6. Після завантаження файлів на GitHub, потрібно перейти в налаштування репозиторію, зображено на рисунку 3.15.

Рисунок 3.15 – Налаштування репозиторію

7. У верхній частині сторінки налаштувань вашого репозиторію на GitHub знаходиться ряд вкладок, серед яких ви знайдете вкладку "Pages". Потрібно натиснути на цю вкладку, щоб перейти до налаштувань GitHub Pages для вашого репозиторію, зображено на рисунку 3.16.

Рисунок 3.16 – Pages

8. Вкладка "Pages" на GitHub надає можливість налаштувати розгортання вашого репозиторію як веб-сайту на GitHub Pages. Це дозволяє вам легко опублікувати ваш проект, документацію або будь-який інший веб-сайт прямо з вашого репозиторію GitHub.

У розділі "Pages" ви можете знайти і налаштувати наступні параметри:

Source (Джерело): Виберіть гілку, з якої буде розгортатися ваш веб-сайт. Зазвичай це гілка "main", але ви можете вибрати будь-яку іншу гілку або гілку з певного каталогу.

Theme (Тема): Застосування теми Jekyll до вашого сайту. Ви можете вибрати одну з доступних тем, які надаються GitHub, або вказати власну тему, знаходячись у вашому репозиторії.

Custom domain (Спеціальний домен): Якщо у вас є власний домен, ви можете налаштувати його для використання з вашим веб-сайтом GitHub Pages. GitHub надає детальні інструкції щодо налаштування DNS записів для вашого домену.

Enforce HTTPS (Обов'язкове використання HTTPS): Ця опція встановлює HTTPS з'єднання для вашого веб-сайту. При включенні цієї опції ваш сайт буде доступний лише за допомогою захищеного протоколу HTTPS.

Visibility (Видимість): Якщо ви користуєтесь GitHub Enterprise, ви можете обмежити доступ до вашого веб-сайту GitHub Pages, публікуючи його приватно.

Приватно опублікований сайт можуть переглядати лише користувачі з правами на читання для репозиторію, з якого він публікується. Це дозволяє обмінюватися внутрішньою документацією або базою знань зі співробітниками вашої організації.

На вкладці "Pages" ви також можете побачити інформацію про останнє розгортання вашого сайту та посилання на додаткові ресурси, де можна дізнатись більше про налаштування та розгортання GitHub Pages. За допомогою цієї вкладки ви можете зручно налаштувати та керувати вашим веб-сайтом на GitHub Pages безпосередньо з вашого репозиторію, зображено на рисунку 3.17.

Рисунок 3.17 − Вкладка Pages

Після вибору вкладки "Pages" на GitHub вам потрібно вибрати гілку, з якої ви хочете розгорнути ваш веб-сайт. Зазвичай це гілка "main" або "master", але ви також можете вибрати будь-яку іншу гілку в залежності від вашої конфігурації.

На сторінці "Pages" ви побачите розділ під назвою "Source" або "Джерело". У цьому розділі ви можете вибрати відповідну гілку з випадаючого списку або вказати шлях до певного каталогу, якщо ваш веб-сайт знаходиться в окремому підкаталозі вашого репозиторію.

Після вибору гілки та налаштування інших параметрів, ви можете зберегти налаштування та розпочати процес розгортання вашого веб-сайту на GitHub Pages, зображено на рисунку 3.18.

Рисунок 3.18 − Вибір гілки

Після вибору гілки на сторінці "Pages" на GitHub ви зможете перейти на свій сайт, який розгорнутий за допомогою GitHub Pages.

На сторінці "Pages" після збереження налаштувань і вибору гілки ви побачите повідомлення, що ваш сайт розгорнутий. Крім того, буде надано URL-адресу вашого сайту, який буде виглядати як "https://[ваше ім'я користувача].github.io/[назва репозиторію]". Наприклад, якщо ваше ім'я користувача - "Datek1", а назва репозиторію - "site", то URL-адреса буде виглядати як "https://datek1.github.io/site".

Клацніть на цю URL-адресу, щоб перейти на ваш веб-сайт, який розгорнутий на GitHub Pages. Там ви зможете переглянути ваші сторінки, навігацію, вміст та інші елементи вашого сайту, зображено на рисунку 3.19.

Рисунок 3.19 − URL розгорнутої веб-сторінки

Схожість

Джерела з Бібліотеки 61

1	Студентська робота	ID файлу: 1011466283	Навчальний заклад: National Technical University of Ukra	2 Джерело	2.72%
2	Студентська робота	ID файлу: 1011347959	Навчальний заклад: National Aviation University	38 Джерело	1.45%
3	Студентська робота	ID файлу: 1014737345	Навчальний заклад: Lviv Polytechnic National University		1.27%
4	Студентська робота	ID файлу: 1013497000	Навчальний заклад: Izmail State University of Humanities	6 Джерело	1.24%
5	Студентська робота	ID файлу: 1008317743	Навчальний заклад: Taras Shevchenko National University of Kyiv		1.07%
6	Студентська робота	ID файлу: 1008256809	Навчальний заклад: National University of Life and Envir	2 Джерело	0.77%
7	Студентська робота	ID файлу: 1014813482	Навчальний заклад: Lviv Polytechnic National University		0.28%
8	Студентська робота	ID файлу: 1015146119	Навчальний заклад: National Aviation University		0.2%
9	Студентська робота	ID файлу: 1009495386	Навчальний заклад: National Technical University of Ukraine “Kyi...		0.2%
10	Студентська робота	ID файлу: 1014641477	Навчальний заклад: Taras Shevchenko National Universit	2 Джерело	0.15%
11	Студентська робота	ID файлу: 1014705673	Навчальний заклад: National Technical University of Ukra	2 Джерело	0.14%
12	Студентська робота	ID файлу: 1008257046	Навчальний заклад: National University of Life and Envir	3 Джерело	0.12%
13	Студентська робота	ID файлу: 1015089825	Навчальний заклад: National Technical University of Ukraine “Kyi...		0.12%