**Пояснювальна записка  
до курсової роботи**

на тему: Веб-застосунок для автоматизованого формування \*.pdf файлів

КПІ.ІП-1512.045440.02.81

Київ – 2023

Зміст

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](#_Toc154604008)

[1.1 Загальні положення 6](#_Toc154604009)

[1.2 Змістовний опис і аналіз предметної області 8](#_Toc154604010)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 8](#_Toc154604011)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 8](#_Toc154604012)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 10](#_Toc154604013)

[1.3.3 Аналіз відомих програмних продуктів 11](#_Toc154604014)

[1.4 Аналіз вимог до програмного забезпечення 14](#_Toc154604015)

[1.4.1 Розроблення функціональних вимог 20](#_Toc154604016)

[1.4.2 Розроблення нефункціональних вимог 23](#_Toc154604017)

[1.5 Постановка задачі 24](#_Toc154604018)

[Висновки до розділу 25](#_Toc154604019)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 26](#_Toc154604020)

[2.1 Моделювання та аналіз програмного забезпечення 26](#_Toc154604021)

[2.2 Архітектура програмного забезпечення 32](#_Toc154604022)

[2.3 Конструювання програмного забезпечення 34](#_Toc154604023)

[2.4 Аналіз безпеки даних 37](#_Toc154604024)

[Висновки до розділу 38](#_Toc154604025)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 40](#_Toc154604026)

[3.1 Аналіз якості ПЗ 40](#_Toc154604027)

[3.2 Опис процесів тестування 41](#_Toc154604028)

[3.3 Опис контрольного прикладу 46](#_Toc154604029)

[Висновки до розділу 51](#_Toc154604030)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 52](#_Toc154604031)

[4.1 Розгортання програмного забезпечення 52](#_Toc154604032)

[4.2 Підтримка програмного забезпечення 53](#_Toc154604033)

[Висновки до розділу 54](#_Toc154604034)

[ВИСНОВКИ 55](#_Toc154604035)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 56](#_Toc154604036)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| IDE | – | Integrated Development Environment – інтегроване середовище розробки. |
| API | – | Application programming interface, прикладний програмний Інтерфейс |
| IT | – | Інформаційні технології |
| БД  HTTP  HTML  CSS  BPMN  ORM  SQL  AJAX  PDF  CORS  XSS | –  –  –  –  –  –  –  –  –  –  – | База даних.  HyperText Transfer Protocol – протокол передачі даних, що використовується в комп’ютерних мережах.  HyperText Markup Language – мова гіпертекстової розмітки.  Cascading Style Sheets – каскадна таблиця стилів.  Business Process Model and Notation – модель та нотація бізнес-процесів.  Object-relational mapping – Об’єктно-реляційна проєкція.  Structured query language – мова структурованих запитів.  Asynchronous JavaScript And XML.  Portable Document Format.  Cross-Origin Resource Sharing  Cross-Site Scripting |

Вступ

У в сучасному інформаційному суспільстві велика увага приділяється автоматизації та оптимізації різноманітних процесів, що включають у себе обробку та обмін даними. Зокрема, важливим елементом цього процесу є забезпечення доступу до інформації у формі, яка є зручною для використання та обміну. Серед таких форматів виділяється \*.pdf – один з універсальних стандартів для представлення документів, що забезпечує стабільність та структурованість відображення даних.

Актуальність розробки веб-застосунку для автоматизованого формування \*.pdf файлів обумовлена постійним зростанням потреби в надійних та ефективних інструментах для обробки та відображення інформації у вигляді \*.pdf. Нинішні тенденції в галузі веб-розробки диктують необхідність створення зручних та ергономічних інтерфейсів, які дозволять користувачам автоматизовано та ефективно генерувати \*.pdf файли з різних джерел і форматів даних.

Метою даного дослідження є розробка ефективного веб-застосунку, який дозволить автоматизовано та зручно формувати \*.pdf файли, забезпечуючи при цьому високий рівень функціональності, безпеки та ефективності використання. Перспективи впровадження такого рішення розглядаються в різних сферах, включаючи бізнес, освіту, медицину та інші галузі, де важливо мати доступ до інформації у формі \*.pdf для подальшого використання та обміну.

Дана курсова робота розкриє основні аспекти розробки веб-застосунку для автоматизованого формування \*.pdf файлів, надаючи при цьому детальний аналіз та порівняння існуючих рішень та технічних підходів в цій області.

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Загальні положення

Упродовж останніх років спостерігається виразний тренд у розвитку програмного забезпечення: відбувається зменшення частки традиційних застосунків для комп'ютерів серед усіх створюваних продуктів. Консольні програми, які раніше були популярні, залишаються переважно у ролі утиліт для серверів. В той самий час, відзначається збільшення відсотка нових розробок у сфері веб-застосунків. Такий склад справ обумовлений, передусім, безперервним розвитком та легким доступом до Інтернету, адже ця мережа змінила те, як люди збирають інформацію, виконують свою роботу, купують товари та послуги, спілкуються з друзями та родиною, проводять дозвілля, тощо[1].

Веб-застосунок – це програмне забезпечення, яке працює на веб-сервері і може бути доступне через веб-браузер на будь-якому пристрої з підключенням до Інтернету.

В роботі веб-застосунка використовується наступний механізм: при переході на будь-яку сторінку застосунку клієнт ініціює запит на отримання цієї сторінки через протокол HTTP. Сервер веб-застосунка обробляє отриманий запит, виконує відповідні логічні операції та/або обчислення, і формує веб-сторінку. Після цього сформована сторінка передається через мережу клієнту у відповідь на його запит, також використовуючи протокол HTTP. Важливо відзначити, що сервер може також взаємодіяти з іншими службами, такими як база даних, або інші зовнішні веб-застосунки, що розташовані на інших серверах. З метою забезпечення більшої продуктивності та інтерактивності використовується підхід AJAX, що дозволяє сторінкам веб-застосунку відправляти запити до сервера у фоновому режимі, без перезавантаження сторінки. Це дозволяє лише довантажувати необхідні дані з сервера, спрощуючи та прискорюючи взаємодію з застосунком.

Перші веб-застосунки з'явилися у 1990 році завдяки британському вченому Тіму Бернерс-Лі [2] і були простими програмами, оскільки кожна веб-сторінка була статичним документом. Однак із випуском JavaScript у 1995 році виникла можливість додавати динамічні елементи на стороні клієнта, що покращило користувацький досвід. З того часу веб-програми стали більш інтерактивними та швидкодійними. [3]

Веб-застосунки розповсюджені у різних сферах. Вони включають у себе соціальні мережі, такі як Facebook та Twitter, які полегшують онлайн-спілкування та мереження. Електронні комерційні платформи, зокрема Amazon та eBay, забезпечують можливість онлайн-шопінгу та зручне відслідковування стану замовлень. Також, важливим є використання веб-застосунків у сфері управління контентом, таких як WordPress, Drupal та Joomla, що дозволяють користувачам створювати, редагувати та публікувати контент онлайн без потреби високих технічних навичок.

При розробці веб-застосунків важливо враховувати аспекти доступності, швидкості та безпеки. Розробники повинні оптимізувати інтерфейс для різних пристроїв, враховуючи різні розміри екранів та можливості використання. Також слід акцентувати увагу на захисті даних користувачів і забезпеченні швидкого відгуку веб-застосунку.

Важливою характеристикою веб-програм є їхня можливість взаємодії з віддаленими серверами для отримання та зберігання даних. Це досягається за допомогою сценаріїв на стороні сервера та клієнта, які обробляють логіку віддалено та на веб-браузері користувача відповідно [4]. Ця взаємодія дозволяє веб-додаткам виконувати різноманітні завдання, такі як обробка форм, керування кошиками для покупок та створення інтерактивних елементів для інтерфейсу користувача.

Однією з найбільших переваг веб-застосунків є їх висока доступність та універсальність [4]. Незалежно від того, чи користувач використовує Windows PC, Mac або мобільний пристрій, він можете отримати доступ до веб-застосунку за допомогою сучасного веб-браузера. Це робить їх економічно вигідними для бізнесу, оскільки немає необхідності розробляти окремі застосунки для кожної платформи, та зручними для звичайних користувачів Інтернету, так як веб-додатки завжди увімкнені – отримати доступ до послуг можна 24 години на добу, сім днів на тиждень.

## Змістовний опис і аналіз предметної області

На сьогоднішній день, в умовах активного розвитку ІТ-технологій, процес формування \*.pdf файлів веб-застосунками є важливим етапом оптимізації бізнес-процесів, а також важливою функцією у буденному житті для роботи з текстом, документами та звітами.

Загалом, область, що розглядається, не є новаторською чи передовою в технологіях, вона не стикається зі значними проблемами та не потребує впровадження нових підходів. У цьому контексті вже давно використовуються готові рішення, і їхні можливості можуть варіюватися залежно від конкретного використання.

Проблемою, яка була виявлена при розгляді наявних рішень, полягає у складності та неінтуїтивності інтерфейсу та певного функціоналу, а також певні обмеження функцій у безкоштовних версіях цих аналогів. Для вирішення цього питання метою розробки є розгляд варіантів максимального спрощення інтерфейсів та доступності усіх функцій без підписок в цих веб-застосунках, спрямованих на формування \*.pdf файлів.

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації веб-застосунку з автоматизованим формуванням \*.pdf файлів. Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення.

### Аналіз відомих алгоритмічних та технічних рішень

У даній розробці не має необхідності застосовувати обчислуванні алгоритми.

В контексті розробки веб-застосунка для автоматизованого формування PDF файлів існують різноманітні технічні рішення, спрямовані на оптимізацію процесу та покращення функціональності. У цьому розділі ми розглянемо деякі з них.

* Фреймворки для генерації PDF:

Розробники можуть використовувати різноманітні фреймворки для генерації PDF файлів. Наприклад, бібліотека pdfkit для мови програмування Python дозволяє легко генерувати PDF з HTML або текстових файлів. Інші популярні фреймворки, такі як wkhtmltopdf або Puppeteer, також можуть використовуватися для створення PDF з веб-сторінок.

* Інтеграція з веб-сервісами:

Веб-застосунки часто використовують віддалені веб-сервіси для автоматизації процесу генерації PDF. Наприклад, сервіси, які приймають дані через API та повертають готовий PDF файл, можуть бути інтегровані в веб-застосунок. Це спрощує розробку та робить процес більш масштабованим.

* Шаблонізація PDF:

Застосування шаблонів для формування PDF може значно полегшити розробку та забезпечити єдність оформлення. Використання бібліотек, таких як Jinja2 або Mustache, дозволяє динамічно вставляти дані у вже підготовлені шаблони PDF документів.

* Клієнтські бібліотеки для генерації PDF:

Для створення PDF можна використовувати бібліотеки, що працюють непосредственно на стороні клієнта, використовуючи JavaScript. Наприклад, бібліотека jsPDF дозволяє створювати PDF документи без необхідності відправляти дані на сервер.

Аналіз таких технічних рішень дозволяє визначити оптимальний підхід до розробки веб-застосунка для автоматизованого формування PDF файлів з урахуванням специфіки завдань та потреб користувачів.

### Аналіз допоміжних програмних засобів та засобів розробки

Серед великої кількості доступних інтегрованих середовищ розробки (IDE) для створення веб-програм, можна виділити PyCharm [16], Visual Studio [17] та інші. PyCharm, розроблене компанією JetBrains, є спеціалізованим інструментом для програмістів, які працюють з мовою програмування Python та пов'язаними технологіями. Воно вирізняється своєю розширюваністю, зручним інтерфейсом користувача, вбудованою системою відладки та інтеграцією з системами контролю версій. PyCharm дозволяє автоматизувати рутинні завдання та ефективно вирішувати складні виклики розробки. Однією з ключових переваг IDE є об'єднання всіх необхідних інструментів.

Вибір мови програмування для створення додатків є важливим завданням. Серед різноманіття доступних варіантів можна виділити такі, як Python, Java, PHP, Ruby, C#, Go, NodeJs. Мова Python [18] стала популярним вибором для розробки додатків, і вона використовується з 1991 року. Багато компаній тепер віддають перевагу використанню саме Python, оскільки вона пропонує широкий спектр фреймворків для розробки різноманітного програмного забезпечення. Тому основна мова програмування, що буде використана у роботі, — Python, що дасть змогу швидко та ефективно розробляти серверну частину програми. Крім того, для створення описаного веб-застосунку обрано фреймворк Flask [19], який відомий своєю легкістю та гнучкістю.

Для розробки клієнтської частини існують два популярних підходи: рендеринг на стороні клієнта (Client-Side Rendering, CSR) та рендеринг на стороні сервера (Server-Side Rendering, SSR) [20]. Переваги Server-Side Rendering (SSR) проявляються в покращеному часі завантаження початкової сторінки, дозволяючи миттєво відображати контент без очікування завантаження та виконання JavaScript. SSR також сприяє поліпшеній оптимізації для пошукових систем, оптимізації на пристроях із обмеженими ресурсами та спрощеній реалізації певних функцій порівняно із рендерингом на стороні клієнта.

З іншого боку, CSR має свої переваги, такі як розширена інтерактивність, динамічний зміст сторінок, гнучкість та відокремлення модулів під час розробки. CSR особливо ефективний для створення великої кількості інтерактивних елементів та надання динамічного змісту сторінок.

В даній роботі використовуються обидва підходи, для найкращого забезпечення різноманітних вимог, визначених цілей проекту, покращення продуктивності, та загальної взаємодії з користувачем.

Також використовується система контролю версій GitHub, що дозволить відслідковувати зміни в коді та відновлювати попередні версії кодової частини.

### Аналіз відомих програмних продуктів

У цьому розділі ми проводимо глибокий аналіз успішних ІТ-проектів, із фокусом на важливих аспектах, що можуть визначати ефективні рішення для веб-додатків формування PDF-файлів в сучасному IT-середовищі. Розглядаючи їх переваги та недоліки, ми звертаємо увагу на ключові критерії успіху.

Оскільки веб-додатків у сфері "Формування PDF-файлів" дуже багато, то варто розглянути лише кілька популярних аналогів цього класу. До них відносяться веб-застосунки, такі як "Sodapdf ", "Smallpdf", " Pdf24.org".

**Pdf24.org** [21]— це проект німецької компанії geek software GmbH, що базується в Берліні та була заснована в 2006 році. PDF24 пропонує безкоштовні та прості у використанні PDF-рішення для вирішення багатьох проблем із PDF-файлами в Інтернеті та у вигляді програмного забезпечення для завантаження.

PDF24 пропонує вам набір інструментів, які можна використовувати для продуктивної роботи з PDF-файлами та іншими файлами, включаючи такі функції, як можливості конвертації, об'єднання, розділення файлів, оптичного розпізнавання символів (OCR), створення PDF з тексту та зображень, а також інші інструменти. Інструменти PDF24 були розроблені, щоб бути максимально простими у використанні. Метою було інтуїтивно зрозуміле використання. Також цей веб-застосунок повністю безкоштовний і не має ніяких обмежень.

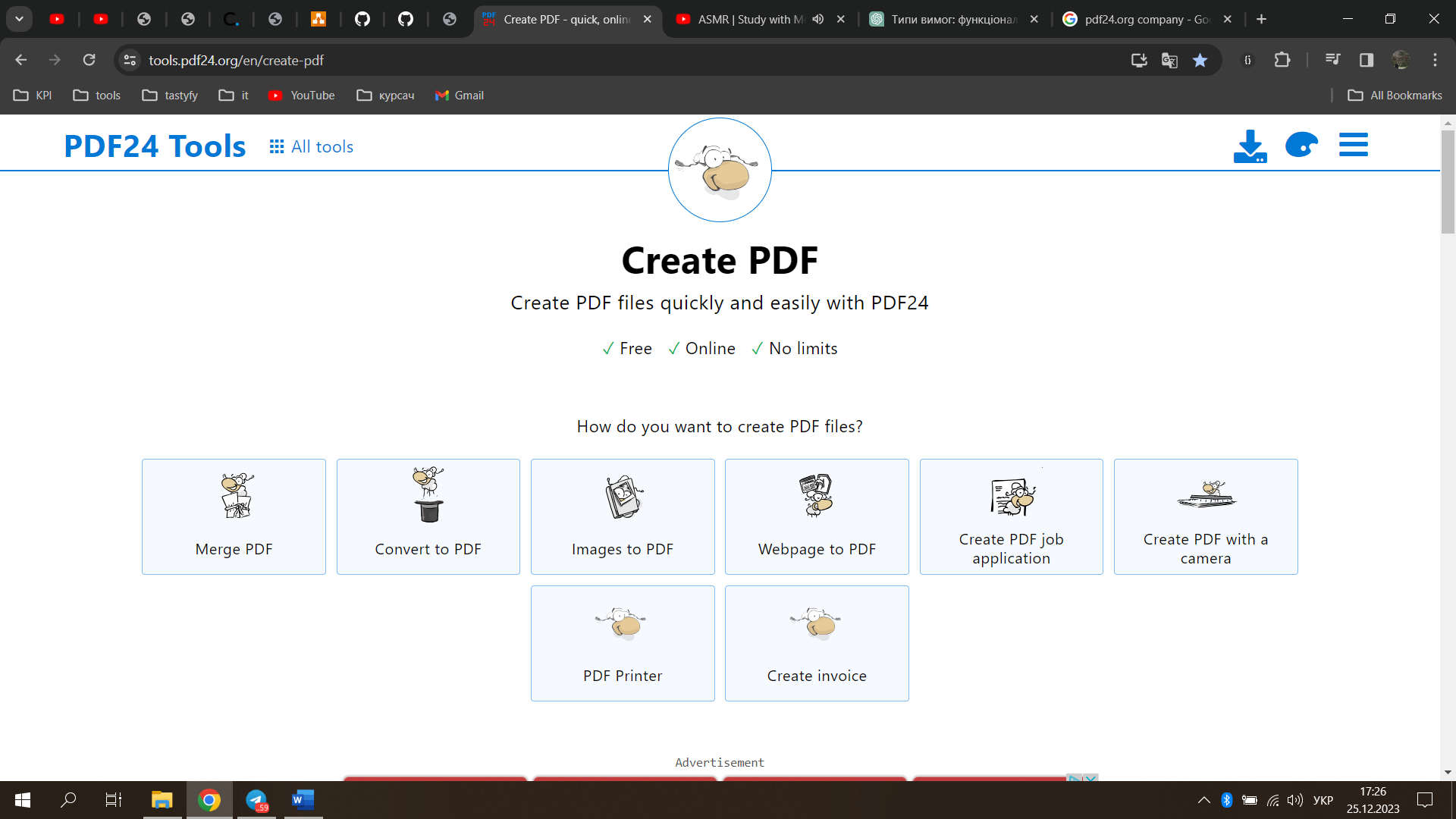


Рисунок 1.1 – Pdf24.org

**Smallpdf** [22]-створений у Швейцарії ще в 2013 році з єдиним інструментом для стиснення PDF-файлів, сьогодні Smallpdf вирішує проблеми безлічі людей у всьому світі, пропонуючи понад 14 простих, безпечних та надійних інструментів для роботи з PDF-файлами, включаючи конвертацію у формат PDF, об'єднання та розділення файлів, редагування тексту, роботу з зображеннями, підписування та інші опції. Smallpdf має дуже простий та зручний, що робить використання сервісу легким для користувачів. Також застосунок підтримує 24 мови для зручності користувачів із різних країн.

Також Smallpdf дозволяє користувачам зареєструватись для можливості зберігати персональні налаштування та шаблони. Веб-застосунок є безкоштовним, але має деякі обмеження якщо користувач не придбав підписку, наприклад, сайт не дає можливості конвертації одразу декількох файлів.

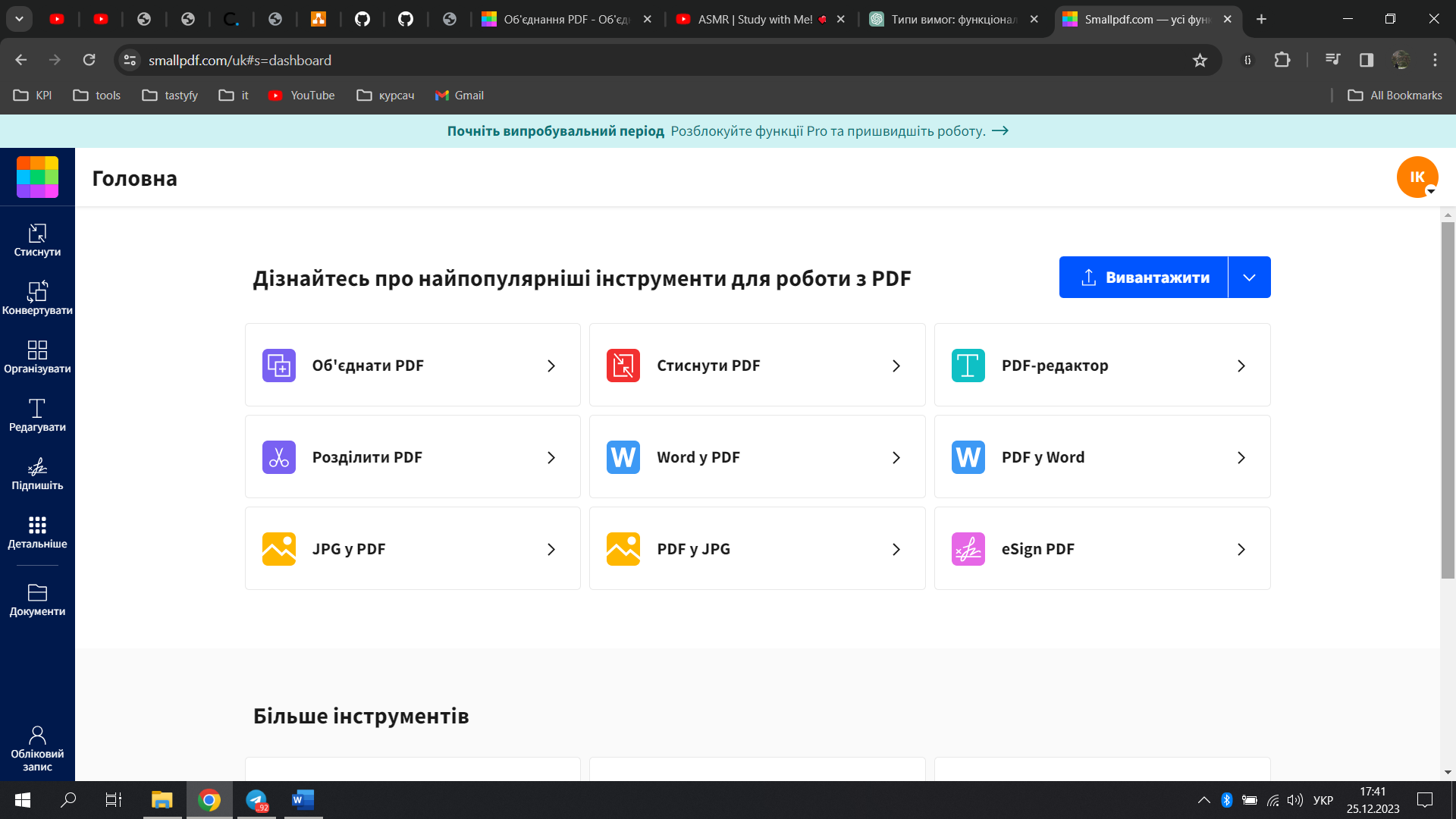


Рисунок 1.2 – Smallpdf

**Sodapdf** [23]- інший сервіс для редагування та конвертації PDF-файлів, що був створений у 2010 році в Канаді. Надає можливості створення, редагування, конвертації, об’єднання та коментування PDF-документів.

Інтерфейс програми забезпечує вигляд документа на основі сторінок, які можна редагувати та обробляти. Сервіс також підтримує декілька мов для міжнародного використання. Sodapdf є безкоштовним веб-застосунком, але у онлайн-версії є деякі розширені функції, як-от оптичне розпізнавання тексту, використати які можна тільки після оплати.

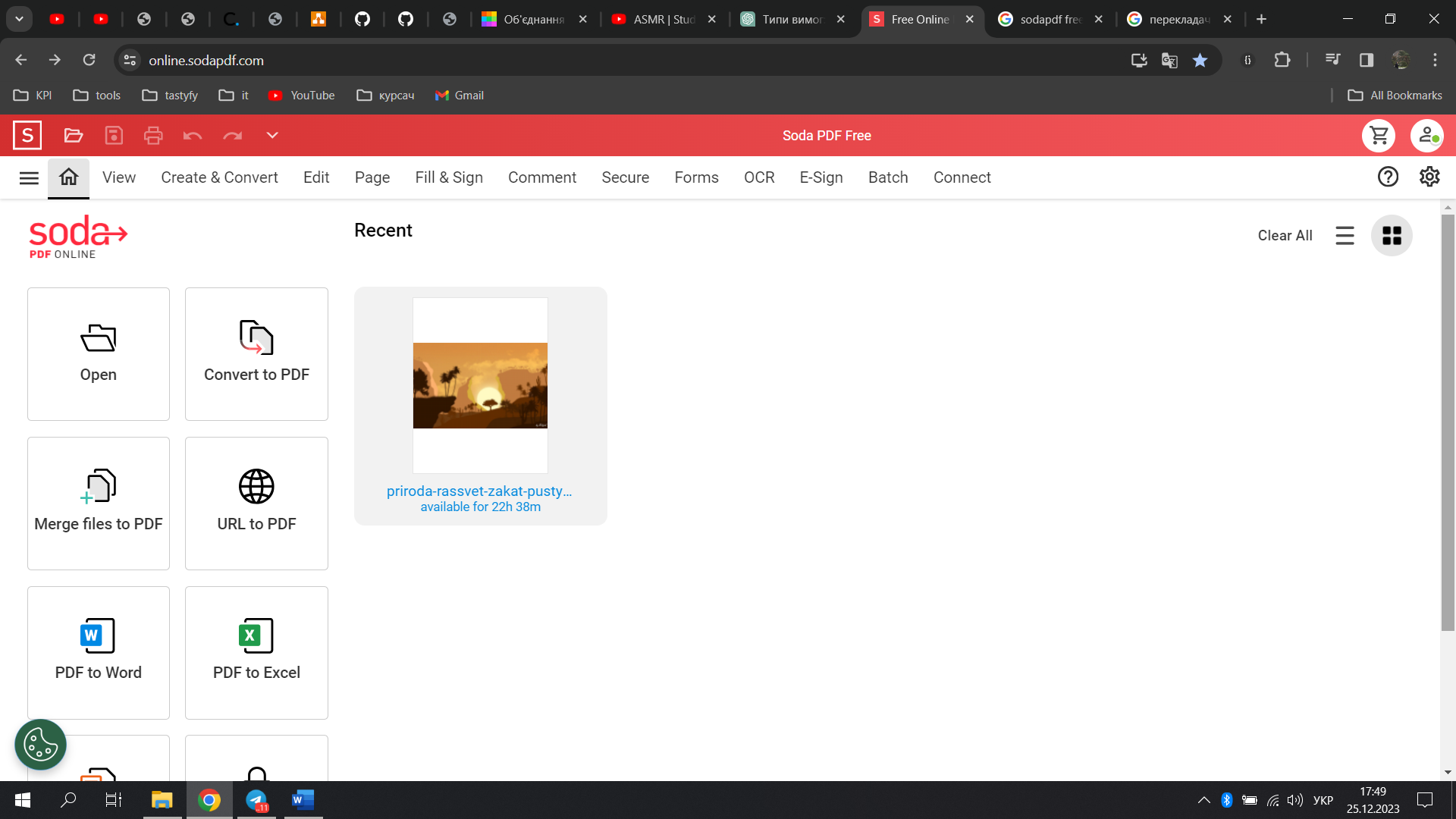


Рисунок 1.3 – Sodapdf

Для порівняння курсової роботи з аналогами можна скористатись таблицею 1.1.

Таблиця 1.1 – Порівняння з аналогами

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Курсова робота (AutoPDFForge) | Smallpdf | Sodapdf | Pdf24.org |
| Конвертація одразу  декількох файлів | + | - | + | + |
| Історія отриманих  файлів користувача | + | + | + | - |
| Редагування PDF | - | + | + | + |
| Простий інтерфейс | + | + | - | + |
| Багатомовний | - | + | + | + |

## Аналіз вимог до програмного забезпечення

Головною функцією розроблюваного програмного забезпечення є автоматизоване формування PDF-файлів з різних джерел та виконання різноманітних операцій з цими файлами. Програма надає зручний та легко зрозумілий інтерфейс для користувача, дозволяючи вибрати одну з чотирьох основних функцій: конвертувати файл у PDF, конвертувати веб-сторінку у PDF, об'єднати декілька PDF або розділити PDF на сторінки. Більш детальний огляд функціоналу можна знайти на рисунку 1.4 де подана діаграма варіантів використання:

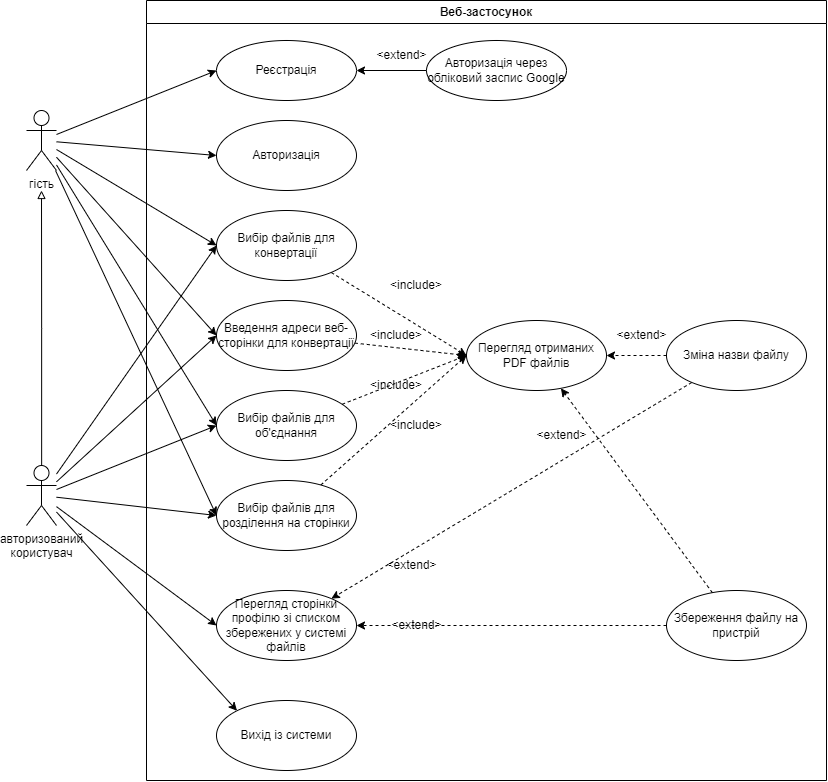


Рисунок 1.4 – Діаграма варіантів використання

В таблицях 1.2 - 1.9 наведені варіанти використання програмного забезпечення.

Таблиця 1.2 - Варіант використання UC-01

|  |  |
| --- | --- |
| Use case name | Реєстрація користувача |
| Use case ID | UC-01 |
| Goals | Реєстрація нового користувача в системі |
| Actors | Гість (незареєстрований користувач) |
| Trigger | Користувач бажає зареєструватися |
| Pre-conditions | - |
| Flow of Events | Користувач переходить на сторінку реєстрації. Далі він обирає спосіб реєстрації з двох варіантів:   1. Користувач вводить у необхідні поля коректні дані: ім’я користувача та пароль. Після заповнення даних, користувач натискає кнопку реєстрації. 2. Або користувач може зареєструватись в системі через обліковий запис Google. Після натискання кнопки "Google", користувач перенаправляється на сторінку авторизації Google, де йому пропонується увійти в свій обліковий запис. Користувач повинен надати згоду на використання облікового запису Google для реєстрації в системі. |
| Extension | У випадку введення не коректних даних, кнопка реєстрації стає неактивною. Якщо якесь конкретне поле введено некоректно, то воно підсвічується червоним надписом. |
| Post-Condition | Створення сторінки користувача, перехід на сторінку авторизації |

Таблиця 1.3 - Варіант використання UC-02

|  |  |
| --- | --- |
| Use case name | Авторизація користувача |
| Use case ID | UC-02 |
| Goals | Користувач успішно авторизується в системі |
| Actors | Гість (неавторизований користувач) |
| Trigger | Користувач бажає авторизуватись |
| Pre-conditions | Користувач уже зареєстрований у системі. Сторінка з формою логіну відкрита |
| Flow of Events | Користувач обирає спосіб авторизації з двох варіантів в залежності від способу реєстрації:   1. Користувач вводить у необхідні поля коректні дані: ім’я користувача та пароль. Після заповнення даних, користувач натискає кнопку логіну. 2. Користувач натискає кнопку "Google", перенаправляється на сторінку авторизації Google, де йому пропонується увійти в свій обліковий запис. Користувач повинен надати згоду на використання облікового запису Google для реєстрації в системі. |
| Extension | У випадку введення не коректних даних, кнопка реєстрації стає неактивною. Якщо якесь конкретне поле введено некоректно, то воно підсвічується червоним надписом. |
| Post-Condition | Користувач успішно увійшов в систему та має доступ до свого облікового запису. |

Таблиця 1.4 - Варіант використання UC-03

|  |  |
| --- | --- |
| Use case name | Вихід з системи |
| Use case ID | UC-03 |
| Goals | Користувач успішно виходить з системи |
| Actors | Авторизований користувач |
| Trigger | Користувач бажає вийти з системи |
| Pre-conditions | Користувач авторизований у системі |
| Flow of Events | 1. Користувач натискає кнопку виходу із системи 2. Система завершує сесію користувача та виводить його на сторінку авторизації. |
| Extension | - |
| Post-Condition | Користувач вийшов із системи та потребує повторної авторизації для отримання доступу. |

Таблиця 1.5 - Варіант використання UC-04

|  |  |
| --- | --- |
| Use case name | Конвертація файлу у PDF |
| Use case ID | UC-04 |
| Goals | Користувач може конвертувати обраний файл у формат PDF. |
| Actors | Авторизований або неавторизований користувач |
| Trigger | Користувач обирає опцію "Конвертувати файл у PDF" на головній сторінці. |
| Pre-conditions | - |
| Flow of Events | 1. Користувач обирає файл для конвертації. 2. Система конвертує файл у формат PDF. 3. Система відображає сторінку з результатами, де користувач може переглянути та завантажити PDF-файл. |
| Extension | У випадку введення не передбачуваних розширень файлів або файл за розміром більше ніж 3.5МБ, кнопка конвертації стає неактивною, система повідомляє користувача про помилку. |
| Post-Condition | Користувач отримав PDF-файл. |

Таблиця 1.6 - Варіант використання UC-05

|  |  |
| --- | --- |
| Use case name | Конвертація веб-сторінки у PDF |
| Use case ID | UC-05 |
| Goals | Користувач може конвертувати обрану веб-сторінку у формат PDF. |
| Actors | Авторизований або неавторизований користувач |
| Trigger | Користувач обирає опцію "Конвертувати веб-сторінку у PDF" на головній сторінці. |
| Pre-conditions | - |
| Flow of Events | 1. Користувач вводить у поле посилання на веб-сторінку для конвертації. 2. Система конвертує сторінку у формат PDF. 3. Система відображає сторінку з результатами, де користувач може переглянути та завантажити PDF-файл. |
| Extension | У випадку введення некоректного посилання, кнопка конвертації стає неактивною, система повідомляє користувача про помилку. |
| Post-Condition | Користувач отримав PDF-файл. |

Таблиця 1.7 - Варіант використання UC-06

|  |  |
| --- | --- |
| Use case name | Об'єднання декількох PDF |
| Use case ID | UC-06 |
| Goals | Користувач може об'єднати обрані PDF-файли у єдиний PDF. |
| Actors | Авторизований або неавторизований користувач |
| Trigger | Користувач обирає опцію "Об'єднати декілька PDF" на головній сторінці. |
| Pre-conditions | - |
| Flow of Events | 1. Користувач обирає кілька PDF-файлів для об'єднання. 2. Система об'єднує обрані файли в один PDF. 3. Система відображає сторінку з результатами, де користувач може переглянути та завантажити новий PDF-файл. |
| Extension | У випадку введення не передбачуваних розширень файлів або файл за розміром більше ніж 3.5МБ, кнопка об’єднання стає неактивною, система повідомляє користувача про помилку. |
| Post-Condition | Користувач отримав єдиний PDF-файл, який об'єднав обрані файли. |

Таблиця 1.8 - Варіант використання UC-07

|  |  |
| --- | --- |
| Use case name | Розділення PDF на сторінки |
| Use case ID | UC-07 |
| Goals | Користувач може розділити обраний PDF-файл на окремі сторінки. |
| Actors | Авторизований або неавторизований користувач |
| Trigger | Користувач обирає опцію "Розділити PDF на сторінки" на головній сторінці. |
| Pre-conditions | - |
| Flow of Events | 1. Користувач обирає PDF-файл для розділення. 2. Система розділяє обраний файл на окремі сторінки. 3. Система відображає сторінку з результатами, де користувач може переглянути та завантажити окремі сторінки PDF або один zip-файл. |
| Extension | У випадку введення не передбачуваних розширень файлів або файл за розміром більше ніж 3.5МБ, кнопка розділення стає неактивною, система повідомляє користувача про помилку. |
| Post-Condition | Користувач отримав окремі сторінки PDF-файлу. |

Таблиця 1.9 - Варіант використання UC-08

|  |  |
| --- | --- |
| Use case name | Перегляд профілю користувача |
| Use case ID | UC-08 |
| Goals | Користувач може переглядати свій профіль, включаючи всі коли-небуть отримані PDF-файли. |
| Actors | Авторизований користувач |
| Trigger | Користувач обирає опцію "Профіль" після авторизації. |
| Pre-conditions | Користувач повинен бути авторизований в системі. |
| Flow of Events | 1. Користувач переходить на сторінку свого профілю. 2. Система відображає список всіх отриманих PDF-файлів, які зберігаються в базі даних для цього користувача. 3. Користувач може переглянути та завантажити PDF файли. |
| Extension | Файли зберігаються у базі даних, зі умови що користувач був авторизований у систему під час їх створення. |
| Post-Condition | Користувач має можливість переглядати та взаємодіяти із своїми PDF-файлами на сторінці профілю. |

### Розроблення функціональних вимог

Відповідно до діаграми варіантів використання було також сформульовано низку функціональних вимог до розроблювальної системи. В таблицях 1.10 – 1.20 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити на рисунку 1.5.

Таблиця 1.10 – Функціональна вимога FR-1

|  |  |
| --- | --- |
| Назва | Реєстрація користувача |
| Опис | Система повинна надавати можливість реєстрації користувачів шляхом введення унікального логіну та паролю. Система також повинна забезпечувати можливість реєстрації користувачів через обліковий запис Google. |

Таблиця 1.11 – Функціональна вимога FR-2

|  |  |
| --- | --- |
| Назва | Авторизація користувача |
| Опис | Система повинна забезпечувати можливість авторизації користувачів за допомогою введення логіну та паролю. Система також повинна забезпечувати можливість авторизації користувачів через обліковий запис Google. |

Таблиця 1.13– Функціональна вимога FR-3

|  |  |
| --- | --- |
| Назва | Вихід із системи |
| Опис | Система повинна дозволяти користувачеві виходити з облікового запису та завершувати сеанс роботи в системі. |

Таблиця 1.14 – Функціональна вимога FR-4

|  |  |
| --- | --- |
| Назва | Конвертація файлів у формат PDF |
| Опис | Система повинна забезпечувати можливість конвертації різних типів файлів у формат PDF за допомогою відповідного інтерфейсу користувача. |

Таблиця 1.15 – Функціональна вимога FR-5

|  |  |
| --- | --- |
| Назва | Конвертація файлів у формат PDF |
| Опис | Система повинна забезпечувати можливість конвертації різних типів файлів у формат PDF за допомогою відповідного інтерфейсу користувача. |

Таблиця 1.16 – Функціональна вимога FR-6

|  |  |
| --- | --- |
| Назва | Об'єднання PDF-файлів |
| Опис | Система повинна надавати можливість об'єднувати кілька PDF-файлів в один файл для зручного управління документами. |

Таблиця 1.17 – Функціональна вимога FR-7

|  |  |
| --- | --- |
| Назва | Розділення PDF-файлів |
| Опис | Система повинна забезпечувати можливість розділення PDF-файлів на окремі сторінки. |

Таблиця 1.18 – Функціональна вимога FR-8

|  |  |
| --- | --- |
| Назва | Перегляд та завантаження PDF-файлів |
| Опис | Система повинна дозволяти користувачам переглядати та завантажувати створені PDF-файли з можливістю вказати нове ім'я для цих файлів. |

Таблиця 1.19 – Функціональна вимога FR-9

|  |  |
| --- | --- |
| Назва | Перегляд усіх створених PDF-файлів на сторінці профілю |
| Опис | Система повинна надавати авторизованим користувачам можливість переглядати список створених за увесь час PDF-файлів на їхній сторінці профілю. |

Таблиця 1.20 – Функціональна вимога FR-10

|  |  |
| --- | --- |
| Назва | Конфігурація порядку та обрання файлів |
| Опис | Система повинна надавати користувачам можливість налаштування порядку обробки та вибору конкретних файлів, які будуть використані у процесах конвертації, об'єднання та розділення. Це здійснюється через можливість додавання нових файлів, їх видалення та зміни порядку завантажених файлів у відповідній області, на відповідній сторінці функції. |

Таблиця трасування вимог [15] є важливим інструментом системного аналізу та управління проектами, спрямованим на відстеження взаємозв'язків між різними елементами проекту. Основна мета цього інструмента полягає в установленні зв'язку між вихідними вимогами до системи і конкретними елементами, які втілюють ці вимоги.

У сфері розробки програмного забезпечення, таблиця трасування вимог зазвичай включає перелік вимог або функціональних можливостей, а також інші аспекти, такі як сценарії використання (use-case), тестові сценарії, компоненти системи тощо. Кожен запис в таблиці асоціюється з конкретними вимогами, які він втілює чи тестує.

Ця таблиця надає структурований погляд на відношення між різними елементами проекту і вихідними вимогами, що допомагає забезпечити повноту та відповідність функціональності системи вимогам

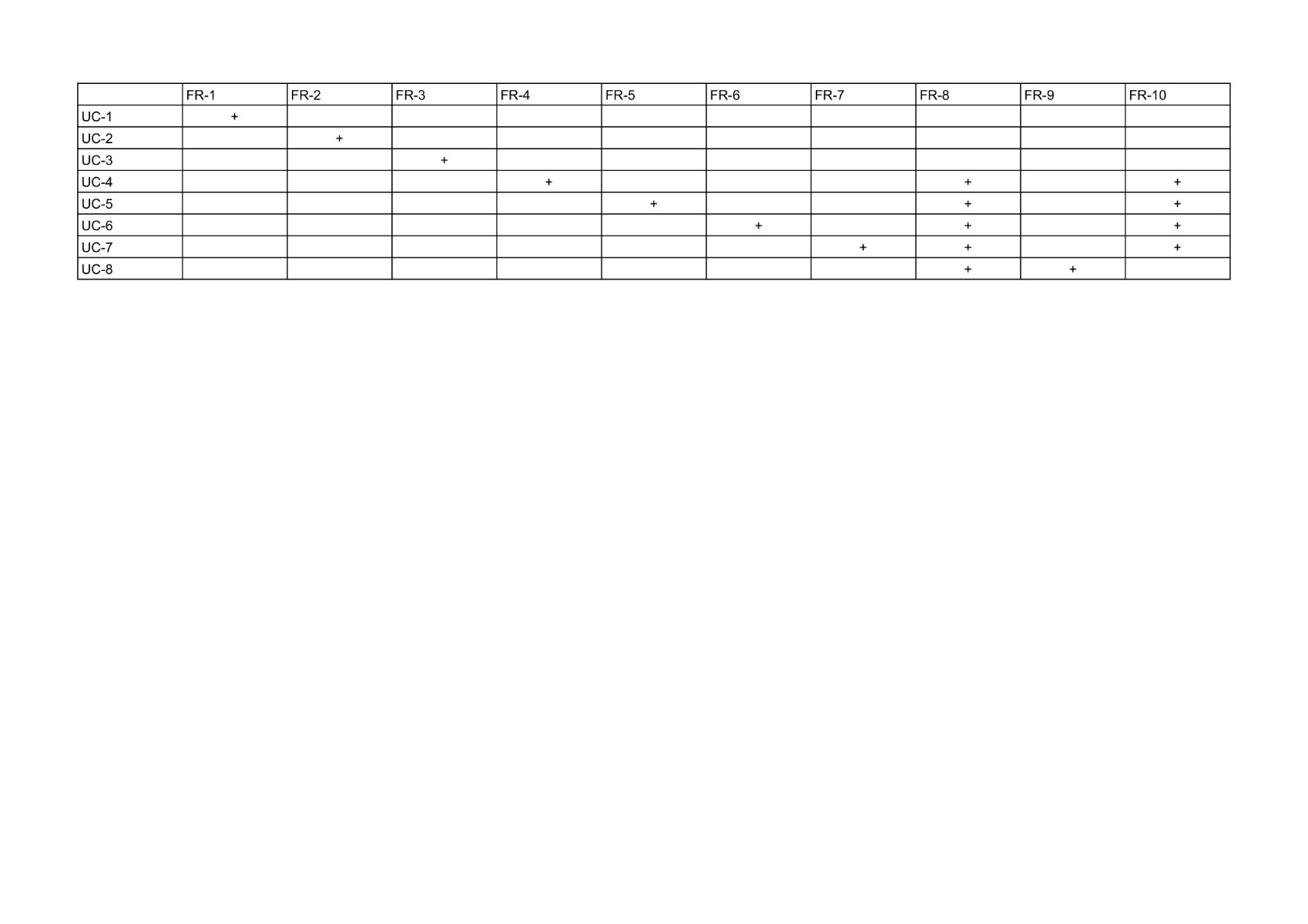


Рисунок 1.5 – Матриця трасування вимог

### Розроблення нефункціональних вимог

Нефункціональні вимоги - це аспекти системи або програмного продукту, які не визначають його основну функціональність, але визначають його якість, характеристики та обмеження.

Майбутня система є веб-застосунком, що призначена для автоматизації процесу формування PDF-файлів, надаючи користувачам можливість зручно та ефективно конвертувати файли у PDF, об'єднувати та розділяти PDF, а також виконувати інші операції з цими документами.

Даний застосунок побудований за принципом клієнт-серверної архітектури й повинен відповідати низці нефункціональних вимог, що наведені нижче:

Таблиця 1.21 Нефункціональні вимоги

|  |  |  |
| --- | --- | --- |
| Номер | Назва | Опис |
| NFR-1 | Сумісність | Забезпечення сумісності із браузерами Google Chrome і Opera та пристроями для зручного використання користувачами. |
| NFR-2 | Надійність | Система повинна бути стійкою до помилок та забезпечувати контроль введення інформації та захист від некоректних дій користувача. При виникненні помилок, система повинна надавати зрозумілі та інформативні повідомлення користувачеві. |
| NFR-3 | Безпека | Система повинна гарантувати конфіденційність інформації та захищати від атак і несанкціонованого доступу під час операцій. |
| NFR-4 | Зручність інтерфейсу | Забезпечення зручного, інтуїтивно зрозумілого та привабливого інтерфейсу для користувачів різних категорій. |
| NFR-5 | Вхідні дані | Система повинна контролювати типи вхідних файлів і дозволяти тільки завантаження файлів із розширеннями: DOC, DOCX, TXT, PNG, JPEG, CSV, XLSX, JPG, PDF, розмір яких не повинен перевищувати 3.5 МБ. |
| NFR-6 | Вихідні дані | Вихідними файлами системи повинні бути PDF або ZIP-архіви. |

## Постановка задачі

Веб-додаток розроблено з метою автоматизації управління процесом формування PDF-файлів, забезпечуючи користувачам інструменти для зручного та ефективного виконання операцій з PDF-документами. Головна ідея полягає в створенні інтуїтивно зрозумілого інтерфейсу для кінцевих користувачів, які можуть не мати великого досвіду в галузі обчислювальної техніки.

Програма має виконувати такі завдання та функції:

* Авторизація користувачів у системі; Забезпечення простої реєстрації та авторизації користувача у системі двома шляхами, через поля логіну та паролю, а також через обліковий запис Google.
* Конвертація в PDF: Забезпечення можливості користувачам конвертувати різноманітні типи файлів (текстові, зображення, тощо) а також веб-сторінки в формат PDF.
* Об'єднання та розділення PDF: Реалізація функцій для об'єднання декількох PDF-файлів в один або, навпаки, розділення одного PDF-файлу на окремі сторінки чи документи.
* Попередній перегляд та завантаження: Надання можливості переглядати створені PDF-файли та змінювати імена файлів перед їх завантаженням, а також зручний механізм завантаження на пристрій користувача.
* Інтуїтивний інтерфейс: Розробка інтерфейсу, що легко засвоюється користувачами, з орієнтацією на тих, хто не є фахівцями в області обчислювальної техніки.

## Висновки до розділу

У даному розділі було проаналізовано вимоги до програмного забезпечення, яке буде розроблено для веб-застосунку AutoPDFForge, який буде використовуватись для автоматизованого формування \*.pdf файлів. Описано загальні концепції та поняття, пов'язані з розроблюваним веб-застосунком, розкрито принцип його роботи у контексті даної предметної області. Проведено аналіз програмних аналогів та визначено функціональні та нефункціональні вимоги.

У розробці клієнтської частини веб-застосунку буде використано HTML, CSS та JavaScript. Цей вибір обумовлений їхньою простотою в освоєнні, гнучкістю, швидкістю та продуктивністю, а також можливістю масштабування. Додатковим фактором у виборі цих технологій є наявний досвід у їхньому використанні [5].

Серверну частину застосунку буде розроблено з використанням фреймворку Flask на основі мови програмування Python. Flask обрано завдяки своїй легкості вивчення та використання. Він пропонує мінімалістичну структуру, що не обмежує розробників і дає можливість використовувати різноманітні бібліотеки. Flask також славиться розширюваністю, забезпечуючи велику кількість розширень для додавання нового функціоналу. Використання Jinja2 для шаблонізації та вбудована підтримка безпеки роблять його ефективним інструментом [6]. Також враховано наявний досвід у розробці програмного забезпечення з використанням мови програмування Python,

# МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У динамічному сучасному інформаційному середовищі розробка програмного забезпечення відіграє критичну роль у вирішенні складних завдань та досягненні стратегічних цілей в різних сферах індустрії та бізнесу. Процес розробки програмного забезпечення стає викликом, який вимагає глибокого розуміння вимог користувачів та використання ефективних методів моделювання та конструювання.

Моделювання та конструювання програмного забезпечення визначають майбутній продукт на ранніх етапах його життєвого циклу. Ці етапи включають в себе визначення концепцій, архітектури та алгоритмів, які лежать в основі подальшого розвитку. Ефективне моделювання сприяє визначенню оптимальних рішень та забезпечує гнучкість системи, тоді як конструювання перетворює ці концепції у функціональний та надійний програмний продукт.

Ця курсова робота ретельно розглядає процеси моделювання та конструювання програмного забезпечення, досліджуючи основні підходи, методи та інструменти, які використовуються на цих етапах розробки. Акцент роботи зосереджено на ключових концепціях та рішеннях, що визначають успішну розробку програмного забезпечення в сучасному інформаційному середовищі.

## Моделювання та аналіз програмного забезпечення

Для моделювання бізнес-процесів було обрано найбільш відповідну нотацію - BPMN. BPMN визнано як стандарт для моделювання бізнес-процесів, що забезпечує уніфікований мовний інструмент для спільного розуміння, аналізу та оптимізації бізнес-процесів у корпоративному середовищі. Цей стандарт використовує символи та правила, спрощуючи створення графічних представлень процесів і полегшуючи ефективну взаємодію між бізнес-аналітиками та розробниками програмного забезпечення.[7]

Використання Бізнес-процесної мережі (BPMN) сприяє стандартизації моделювання бізнес-процесів, забезпечуючи єдиний набір термінів та концепцій на міжнародному рівні. Це робить можливим узгоджену роботу різних команд і організацій під час визначення та вдосконалення бізнес-процесів.

Проаналізувавши заздалегідь вимоги та задачі до розроблювальної системи, було висунуто такі головні бізнес-процеси:

* реєстрація;
* авторизація;
* конвертація у PDF;
* об’єднання PDF;
* розділення PDF на сторінки;

Побачити діаграми бізнес-процесів розроблюваного застосунку можна на рисунках 2.1 – 2.5

Опис послідовності створення облікового запису користувача:

* користувач переходить на сторінку реєстрації;
* користувач заповнює поля реєстрації;
* якщо введені поля, не відповідають шаблону заповнення на клієнтській стороні, відповідні поля підсвічуються помилкою;

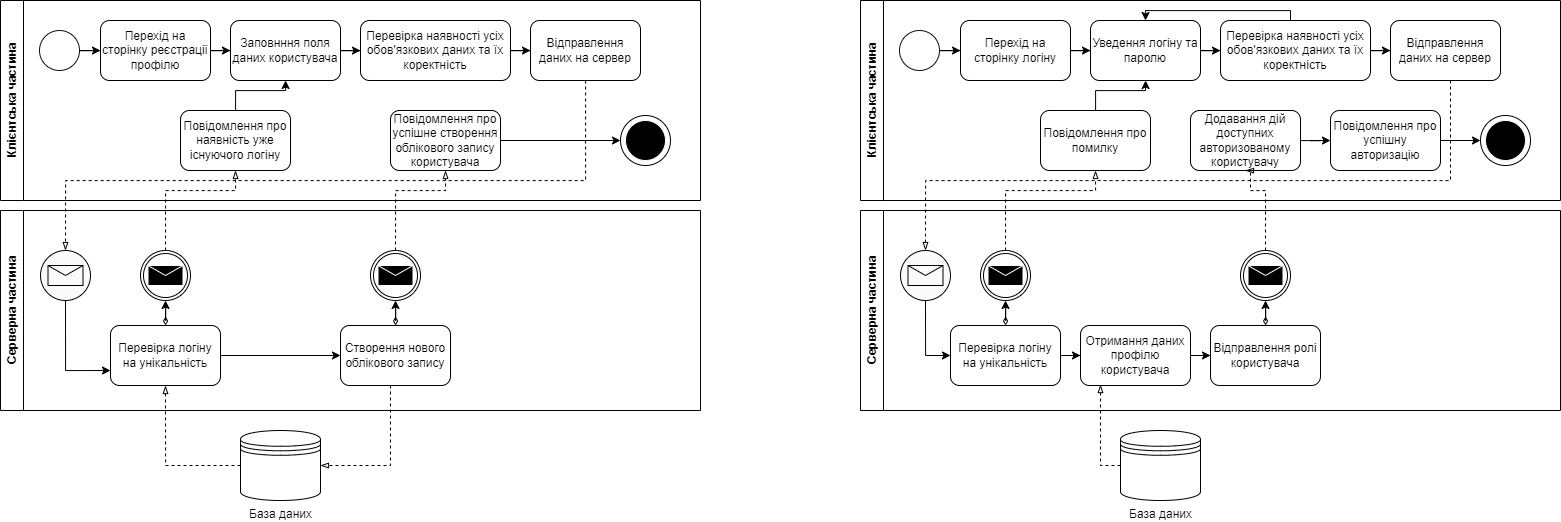


Рисунок 2.1 - Діаграма бізнес-процесу реєстрації

Опис послідовності авторизації користувача:

* користувач переходить на сторінку авторизації;
* користувач заповнює поля реєстрації;
* якщо введені поля, не відповідають шаблону заповнення на клієнтській стороні, відповідні поля підсвічуються помилкою;
* якщо введені дані не збігаються з даними у БД, виникаєповідомлення з попередженням;

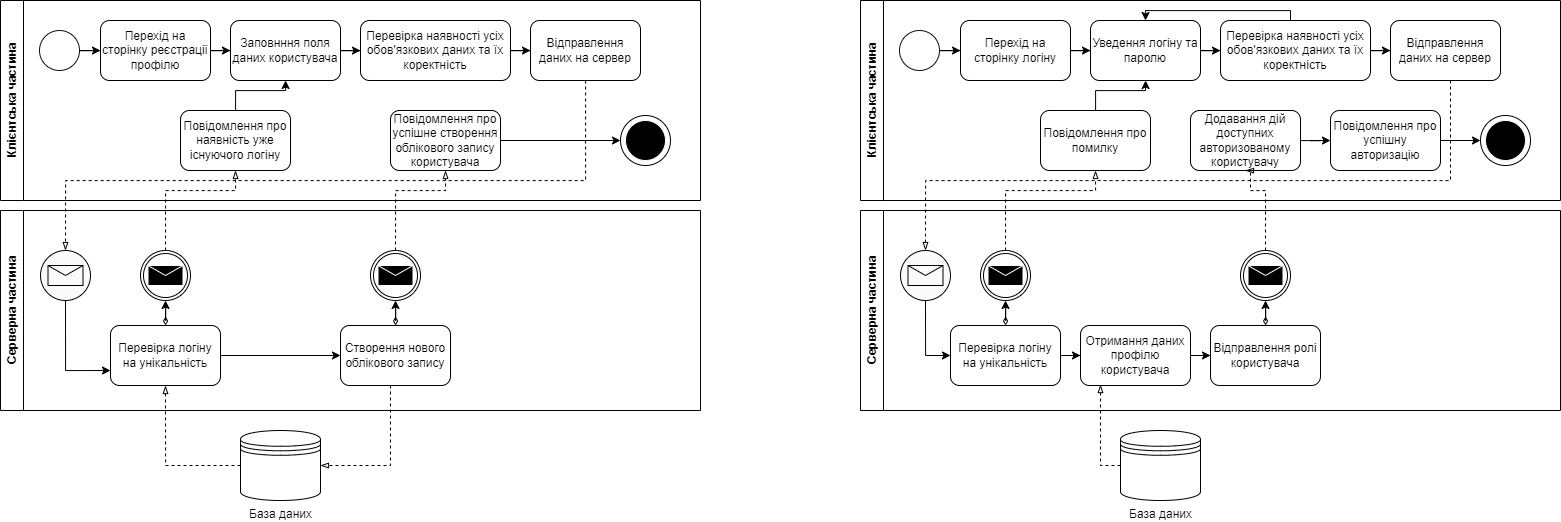


Рисунок 2.2 - Діаграма бізнес-процесу авторизації

Опис процесу конвертації у PDF:

* користувач обирає файли (або вводить адресу веб-сторінки) для конвертації;
* користувач запускає процес конвертації за допомогою відповідної кнопки;
* якщо файли не відповідають дозволеним форматам або розмірам, виникає повідомлення з попередженням;
* користувач отримує сформовані PDF-файли;

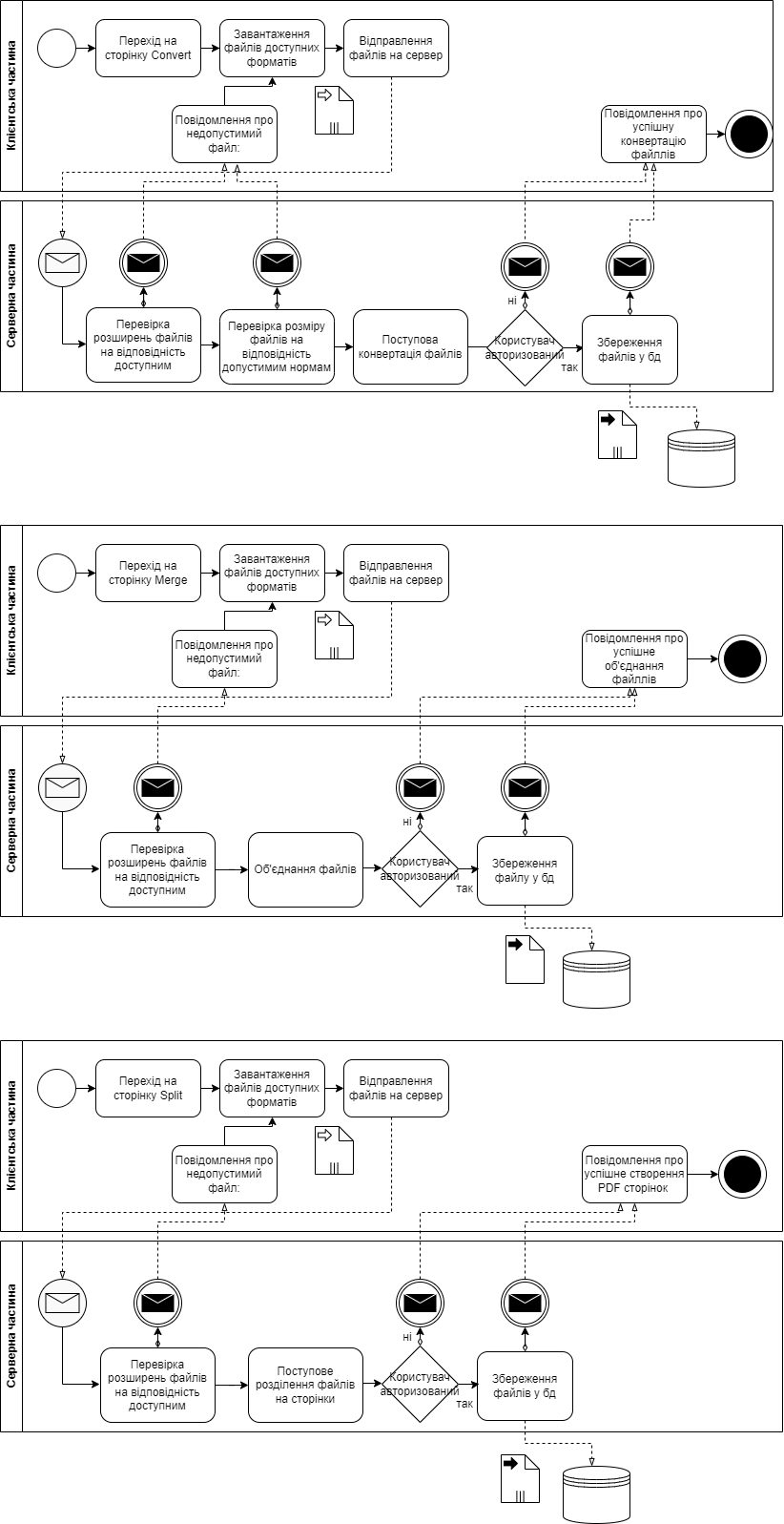


Рисунок 2.3 - Діаграма бізнес-процесу конвертації

Опис процесу об'єднання PDF-файлів:

* користувач обирає PDF-файли для об'єднання;
* користувач запускає процес об’єднання за допомогою відповідної кнопки;
* якщо файли не відповідають дозволену формату, виникає повідомлення з попередженням;
* користувач отримує сформований PDF-файл;

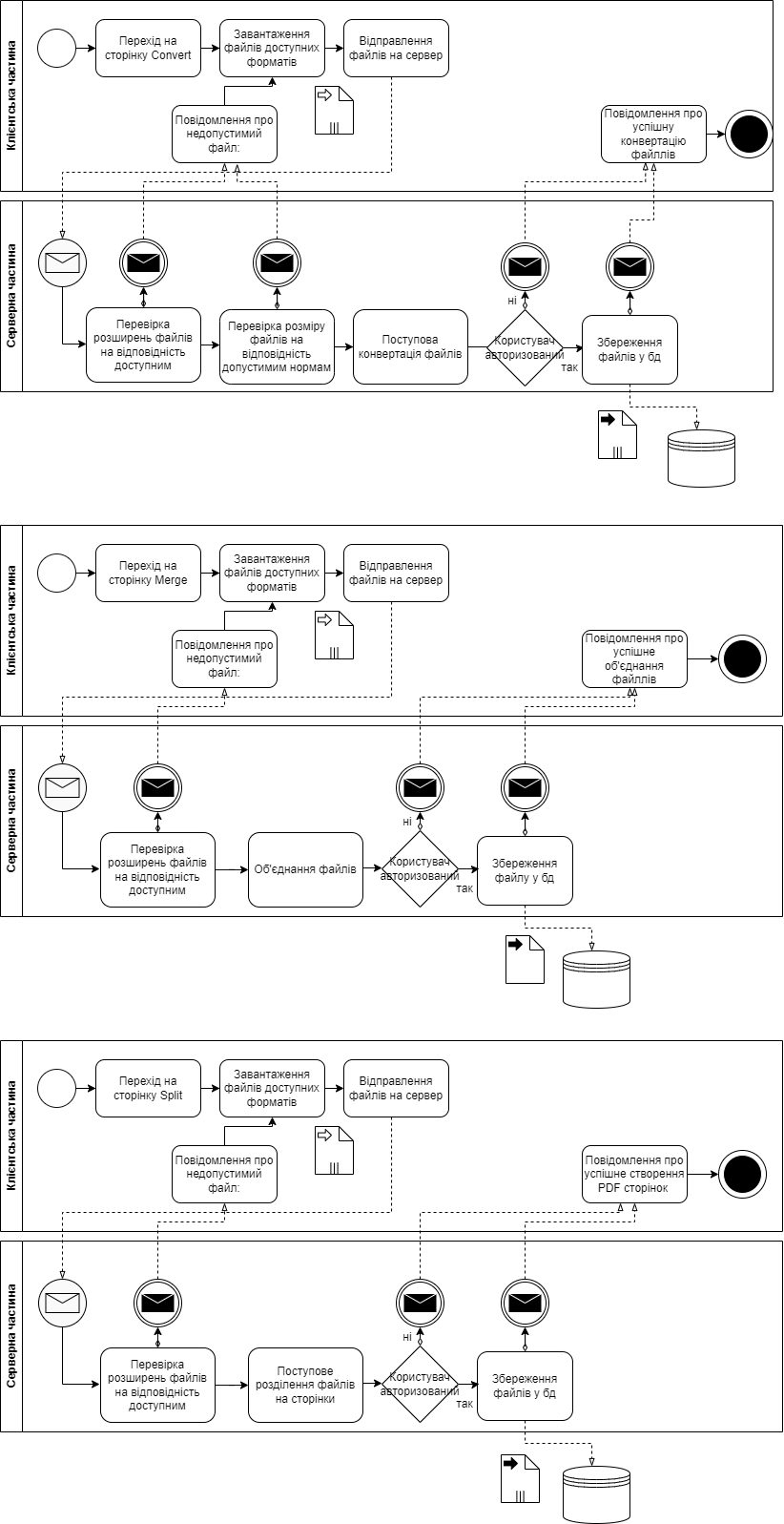


Рисунок 2.4 - Діаграма бізнес-процесу об’єднання

Розділення PDF-файлів:

* користувач обирає PDF-файл для розділення;
* користувач запускає процес розділення за допомогою відповідної кнопки;
* якщо файли не відповідають дозволену формату, виникає повідомлення з попередженням;
* користувач отримує сформовані PDF-файл;

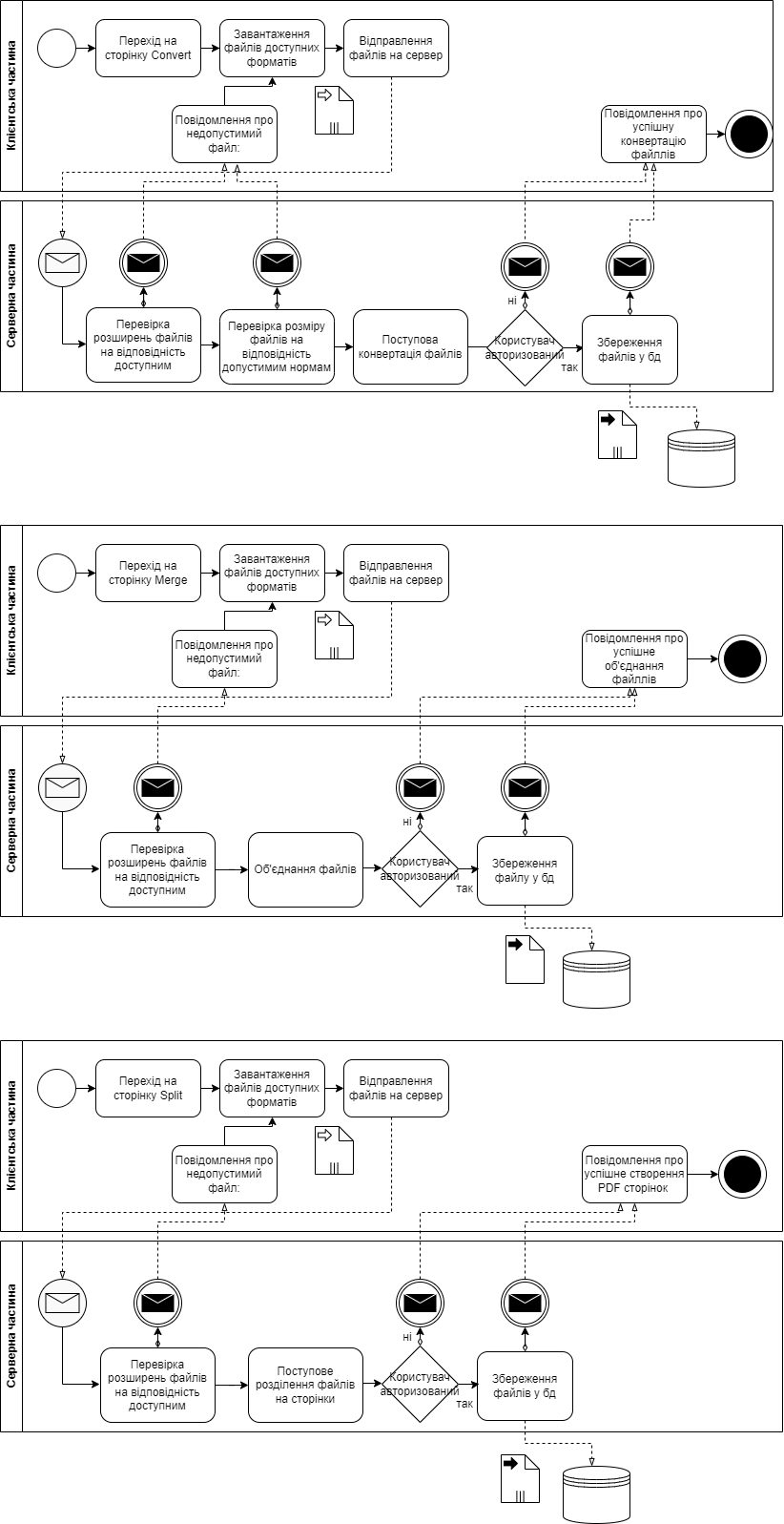


Рисунок 2.5 - Діаграма бізнес-процесу розділення

## Архітектура програмного забезпечення

Створюваний веб-застосунок, що є об'єктом даної курсової роботи, використовує клієнт-серверну архітектуру. Клієнтська частина реалізована з використанням HTML, CSS та JavaScript, в той час як серверна частина використовує Flask фреймворк. Для взаємодії з базою даних використовується SQLAlchemy, яке є популярною бібліотекою ORM для Python. SQLAlchemy надає потужні засоби для взаємодії з базами даних у Flask-додатках, дозволяючи легко виконувати операції створення, читання, оновлення та видалення даних в базі.[8]

Концепція клієнт-серверної архітектури передбачає розділення програмного забезпечення на дві логічні частини: клієнт та сервер. Сервер відповідає за зберігання та цілісність даних, обробку запитів клієнта і містить основну логіку роботи програми. З іншого боку, клієнт взаємодіє з користувачем та отримує дані від сервера при необхідності. Зв'язок клієнта і сервера відбувається через протоколи обміну даними, такі як HTTP або HTTPS. Клієнт формує запити, а сервер відповідає на ці запити, забезпечуючи таким чином взаємодію веб-додатку. Основні переваги такої архітектури включають централізацію, масштабованість, легкість управління, безпеку даних та доступність. [9]

Загальна архітектура системи зображена на рисунку 2.6.

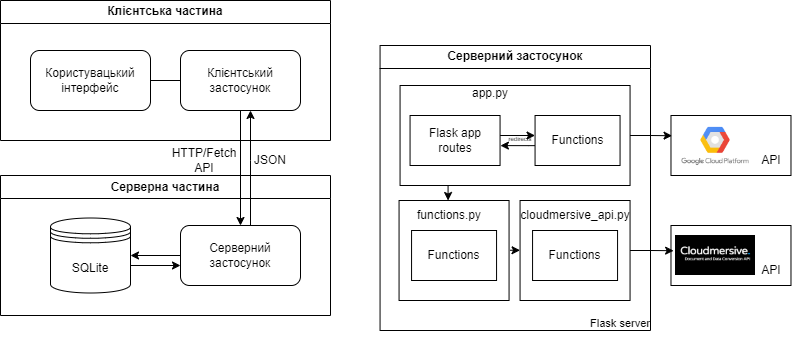


Рисунок 2.6 – Загальна архітектура розроблюваного веб-застосунку

Бізнес логіка веб-застосунку реалізована в серверному застосунку. Розглянемо його логічні компоненти на рисунку 2.7.

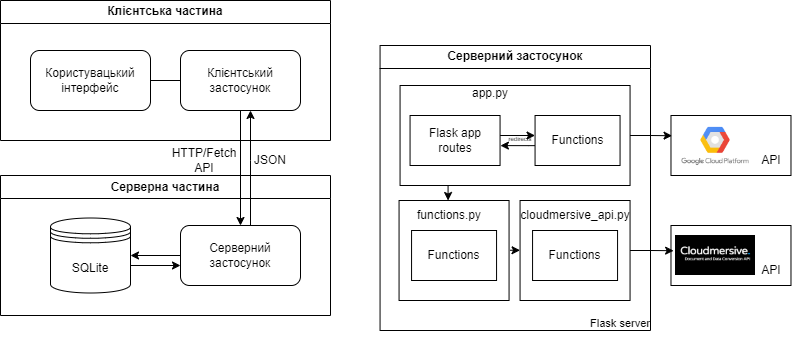


Рисунок 2.7 – Логічні компоненти серверного застосунку

Сервер веб-застосунку розроблюється на фреймворку Flask і має основний файл app.py, що виконує роль контролеру для сервера, він в свою чергу викликає функціонал інших компонентів програми. Він відповідає за створення зв’язку між сервером і БД, та за створення самих таблиць БД. А також за зв’язок з Google Cloud Platform API, що використовується для авторизації через обліковий засіб Google.

Файл functions.py містить в собі бібліотеки та функції, що використовуються для виконання основного функціоналу застосунку, тобто те, що пов’язано з роботою над PDF файлами. Також цей компонент взаємодіє з cloudmersive\_api.py який створений для зв’язку з Cloudmersive Api, одним з засобів для конвертації word документів.

Під час обробки запиту від клієнта сервером, файл app.py викликає відповідні функції чи методи, свої чи з інших компонентів. Ці компоненти, у свою чергу, повертають необхідні дані чи відповіді для передачі їх клієнту.

Розглянемо складові частини клієнтської частини застосунку та вибрані інструменти їх реалізації:

* Основна структура та вигляд веб-сторінок

Для створення та оформлення веб-сторінок, використовується HTML для розмітки та CSS для стилізації.

* Клієнтська логіка

Для написання коду, який виконується на боці клієнта та забезпечує взаємодію та динаміку на сторінці використовується JavaScript.

* API-виклики

Було обрано використовувати Fetch API [24], а в деяких випадках, наприклад відправка інформації з форм реєстрації та авторизації, використовувались HTTP запити);

* Маршрутизація (навігація між веб-сторінками – HTML, JavaScript);

## Конструювання програмного забезпечення

Веб-застосунок, який реалізується у межах даної курсової роботи, не вимагає впровадження конкретних алгоритмів чи специфічних технічних рішень.

В якості системи управління базами даних використовується SQLite, яка інтегрована відразу в Flask за допомогою розширення Flask-SQLAlchemy, що спрощує використання SQLAlchemy з Flask, налаштовуючи загальні об’єкти та шаблони для використання цих об’єктів, наприклад сеанс, прив’язаний до кожного веб-запиту, моделі та механізмию. SQLite — це бібліотека C, яка надає легку дискову базу даних, яка не потребує окремого серверного процесу та дозволяє отримати доступ до бази даних за допомогою нестандартного варіанту мови запитів SQL. Деякі програми можуть використовувати SQLite для внутрішнього зберігання даних. [11]

База даних серверу призначена для зберігання користувачів, а також даних про їх отримані pdf файли. Опис таблиць бази даних наведено у таблицях 2.11 - 2.12. Модель бази даних наведена на рисунку 2.12.

Таблиця 2.11 – Опис таблиці user

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| user | id | Integer | Ідентифікаційний номер користувача |
| username | varchar(100) | Логін користувача |
| password | varchar(20) | Пароль користувача (зашифрований) |

Таблиця 2.12 – Опис таблиці file

|  |  |  |  |
| --- | --- | --- | --- |
| Таблиця | Назва поля | Тип даних | Опис |
| file | id | Integer | Ідентифікаційний номер файлу |
| filename | varchar (100) | Назва файлу |
| data | LargeBinary | Дані файлу (бінарні дані) |
| created\_at | DateTime | Дата та час створення файлу |
| user\_id | Integer | Зовнішній ключ, посилання на id користувача з таблиці user |
| file\_size | Float | Розмір файлу в мегабайтах |

На рисунку 2.2 можна побачити модель бази даних:

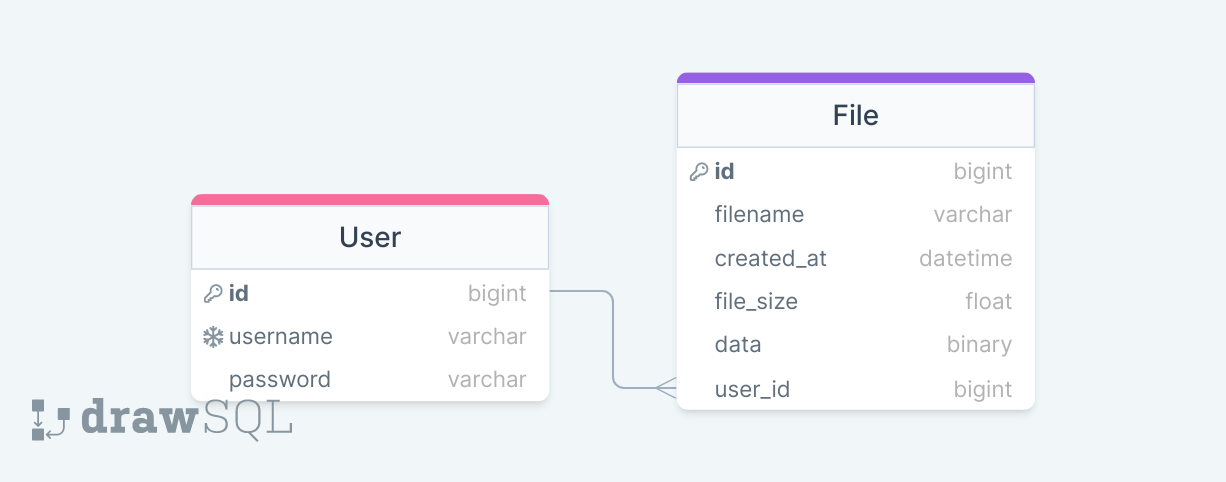


Рисунок 2.2 – Модель бази даних

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.13.

Таблиця 2.13 – Опис утиліт

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 1 | PyCharm | Головне середовище розробки програмного забезпечення курсової роботи. |
| 2 | Git | Система керування версіями. |

Таблиця 2.14 – Опис стороннього програмного забезпечення

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва ПЗ | Опис застосування |
| 1 | Google Cloud Console | Веб-інтерфейс для управління послугами та ресурсами Google Cloud Platform (GCP). У даній розробці використовується для забезпечення реєстрації та авторизації користувачів через обліковий запис Google. |
| 2 | SQLite | Вбудована база даних, яка забезпечує ефективне зберігання та обробку даних без потреби в екстернальному сервері баз даних. |
| 3 | Сloudmersive | Веб-служба, що пропонує API рішення для обробки даних, виявлення шкідливих програм, конвертації файлів, розпізнавання тексту та багатьох інших функцій. В даній розробці використовується API для конвертації деяких типів файлів у формат PDF. |

## Аналіз безпеки даних

Для захисту від загроз безпеки даних у веб-додатку, розробленому з використанням фреймворку Flask та шаблонів Jinja2, були застосовані передові практики написання програмного коду та рекомендації від групи розробників Flask та Python.

При розробці застосунку ми використаємо вбудовані засоби захисту Flask-фреймворку: налаштуємо Cross-Origin Resource Sharing (CORS) [12].

Однією з ключових аспектів безпеки є захист від атак типу XSS (Cross-Site Scripting), що є вразливістю інтерактивних веб-систем. XSS виникає, коли на веб-сторінки потрапляють користувацькі скрипти, що може призвести до атаки на користувача.[10] В даному контексті, застосовуючи шаблони Jinja2, важливо систематично блокувати потенційно небезпечні значення, які можуть включати в себе користувацькі введення або інші динамічні дані. Flask, разом з Jinja2, дозволяє уникати XSS-атак, попереджаючи небезпечні вставки даних у веб-сторінках.

Також налаштовано систему так, щоб обмежити можливість завантаження файлів лише до деяких дозволених типів, зменшуючи тим самим ризик використання шкідливих або виконавчих файлів і завантаження їх до бази даних. Це досягається шляхом використання перевірок на сервері перед збереженням файлів, які враховують їхні типи та розширення.

## Висновки до розділу

У третьому розділі курсової роботи було виконано моделювання та конструювання програмного забезпечення, були розглянуті різні аспекти архітектури додатку та його компоненти.

У розділі була розглянута архітектура програмного забезпечення, де були визначені основні компоненти системи та залежності між ними. Процес розробки включав детальний аналіз та розуміння вимог веб-застосунку формування \*.pdf файлів, що дало змогу визначити ключові функціональні можливості, необхідні для ефективної роботи системи. На основі цього аналізу були розроблені алгоритми, які забезпечують правильну реалізацію функціональності системи. Були розроблені алгоритми для роботи з авторизацією користувачів, конвертацією та роботою з PDF файлами, переглядом та завантаженням файлів.

В розділі моделювання та конструювання програмного забезпечення було проведено детальний опис структури та організації бекенду (Back-end). Зокрема, була розроблена база даних, яка забезпечує збереження даних про користувачів та їх файлів. Це створює основу для зручної та ефективної роботи з даними в системі.

Також були розроблені бізнес-процеси веб-застосунку, що відображають послідовність дій та взаємодій у системі. Це дозволяє краще розуміти логіку роботи системи та взаємодію з користувачами.

Крім того, проведено опис структури та організації клієнтської частини (Front-end) веб-застосунку. Були визначені механізми взаємодії з користувачем, включаючи відображення файлів. Це дозволяє забезпечити зручний та інтуїтивно зрозумілий інтерфейс для користувачів системи.

В останньому підрозділі було проведено детальний аналіз потенційних загроз для веб-застосунків та перечислено всі заходи, яких було вжито задля безпеки розроблюваної системи.

В цілому, розділ моделювання та конструювання програмного забезпечення виявився надзвичайно важливим. Він допоміг створити чітку архітектуру додатку, яка відповідає функціональним вимогам та забезпечує зручну та ефективну роботу для користувачів.

# АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

У даному розділі ми звернемо увагу на характеристики якості та стабільності розробленого програмного продукту, а також виконаємо процес тестування з метою підтвердження його надійності та ефективності.

## Аналіз якості ПЗ

Із збільшенням кількості сучасних технічних засобів та можливостей програмного забезпечення розширюється сфера його відповідальності та рівень довіри до нього. Вже сьогодні люди довіряють програмам вирішувати важливі завдання, що часто пов'язані зі здоров'ям, безпекою та життям. При вирішенні завдань такого характеру, якість цих продуктів має бути належного рівня на всіх етапах їхнього життєвого циклу.

Для успішного контролю якості необхідні формалізовані, кількісні оцінки показників якості. Такими оцінками є метрики ПЗ, які розташовані на найнижчому, фундаментальному рівні моделі якості програмного забезпечення згідно зі стандартами і виступають основоположними.

Аналіз якості ПЗ було проведено з допомогою веб-застосунку «Count LOC online – CodeTabs» [13], який автоматично генерує звіт якості коду за різними метриками.

NOM (Number of Modules) представляє собою кількість складних модулів, які були ідентифіковані аналізатором. Сюди включаються всі класи та будь-які інші модулі, що містять визначені функції-члени. Загальна кількість окремих модулів, ідентифікованих у коді, становить 39. Ця метрика важлива для приблизної оцінки складності проекту, оскільки зі зростанням розміру проекту збільшується кількість файлів, що ускладнює його обслуговування.

LOC (Lines of Code) [14] визначається як кількість непорожніх рядків вихідного коду без коментарів, обчислених аналізатором. Загальна кількість рядків коду становить 3703.

COM (Lines of Comments) визначається як кількість рядків коментарів, які були виявлені аналізатором. Загальна кількість рядків коментарів оцінюється в 38 рядків. Ця метрика може бути значущою для великої кодової бази.

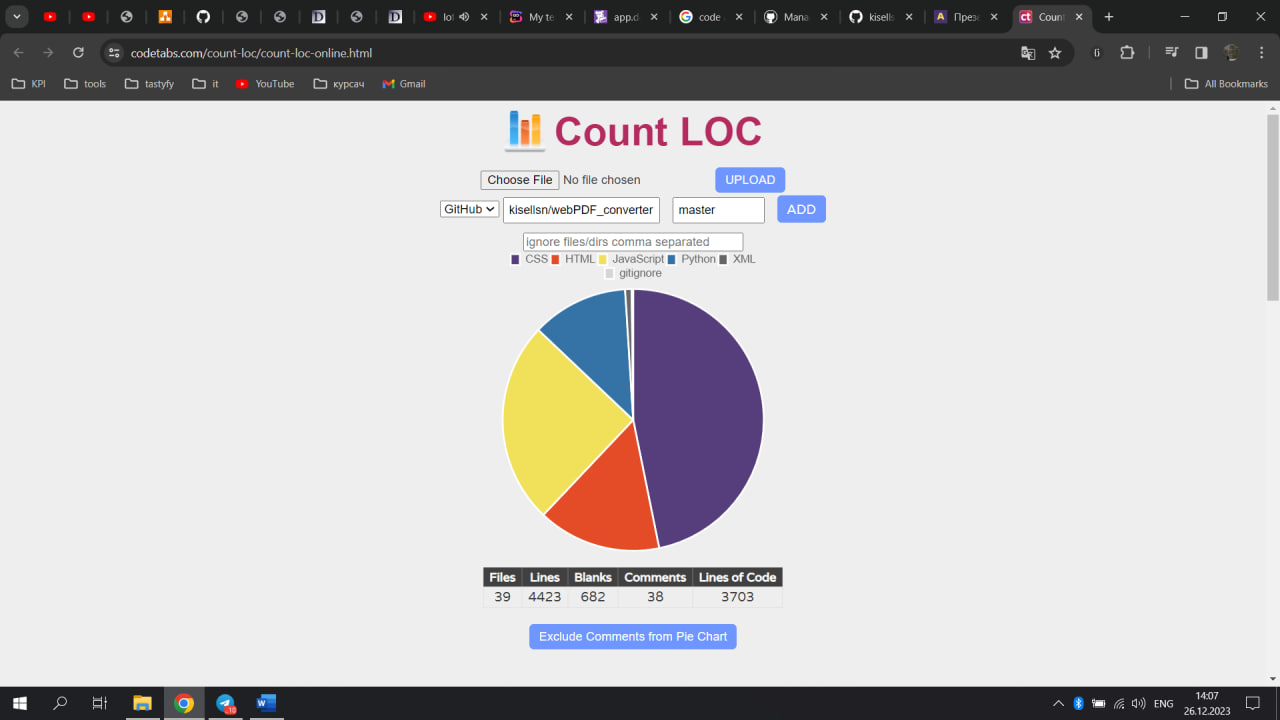


Рисунок 3.1 – Звіт якості коду

## Опис процесів тестування

Тестування програмного забезпечення є ключовим етапом у життєвому циклі розробки, оскільки воно не лише дозволяє оцінити якість готового продукту, але й перевіряє виконання поставлених вимог та функцій, що повинні бути реалізовані. Цей процес дає можливість охопити різні аспекти застосунку, ефективно використовуючи наявні часові та людські ресурси, і виявити критичні проблеми перед впровадженням.

Для тестування створеної системи був обраний метод функціонального тестування, який дозволяє перевірити правильність виконання поставлених задач. Основні функції системи, які було вирішено перевірити, включають:

* реєстрацію у системі (Таблиця 3.1),
* авторизацію у системі (Таблиця 3.2),
* конвертацію файлів у формат PDF (Таблиця 3.3),
* конвертацію веб-сторінок у формат PDF (Таблиця 3.4),
* об'єднання декількох PDF-файлів (Таблиця 3.5),
* розділення PDF-файлів (Таблиця 3.6),
* завантаження файлів на пристрій (Таблиця 3.7),
* коректне відображення профілю користувача

Успішне тестування буде відзначатися правильним збереженням даних в базу даних, видачею повідомлень про помилки при введенні некоректних даних, коректними вихідними даними функцій, що тестуються, та правильним відображенням інтерфейсу системи.

Таблиця 3.1 – Тест 1.1

|  |  |
| --- | --- |
| Тест | Реєстрація користувача |
| Номер тесту | 1.1 |
| Початковий стан системи | Користувач знаходиться на сторінці реєстрації |
| Вхідні данні | Ім’я користувача, пароль |
| Опис проведення тесту | У відповідні поля вводяться: логін користувача, який до цього не був зареєстрований в системі, пароль від 8 до 20 символів. Після цього користувач натискає на кнопку SingUp. |
| Очікуваний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |
| Фактичний результат | Реєстрація проходить успішно, користувач додається у систему і перенаправляється на сторінку авторизації. |

Таблиця 3.2 – Тест 1.2

|  |  |
| --- | --- |
| Тест | Авторизація користувача |
| Номер тесту | 1.2 |
| Початковий стан системи | Користувач знаходиться на сторінці авторизації |
| Вхідні данні | Ім’я користувача, пароль |
| Опис проведення тесту | У відповідні поля вводяться: логін користувача, який до цього вже був зареєстрований в системі та пароль від 8 до 20 символів, що співпадає з паролем у БД до цього логіну. Після цього користувач натискає на кнопку LogIn. |
| Очікуваний результат | Авторизація проходить успішно, користувач перенаправляється на сторінку профілю. |
| Фактичний результат | Авторизація проходить успішно, користувач перенаправляється на сторінку профілю. |

Таблиця 3.3 – Тест 1.3

|  |  |
| --- | --- |
| Тест | Конвертація файлу у формат PDF |
| Номер тесту | 1.3 |
| Початковий стан системи | Користувач знаходиться на сторінці завантаження файлів для їх конвертації |
| Вхідні данні | Файл/файли певних форматів |
| Опис проведення тесту | Користувач завантажує файли обраного формату у відповідну область та натискає кнопку "Конвертувати". Система перевіряє розширення файлів, і якщо всі файли відповідають допустимим форматам, вони автоматично конвертуються. |
| Очікуваний результат | Система має конвертувати обраний файл у формат PDF та надавати можливість перегляду та завантаження отриманого PDF. |
| Фактичний результат | Конвертація пройшла успішно, користувач може переглядати та завантажувати отриманий PDF. |

Таблиця 3.4 – Тест 1.4

|  |  |
| --- | --- |
| Тест | Конвертація веб-сторінки у формат PDF |
| Номер тесту | 1.4 |
| Початковий стан системи | Користувач знаходиться на сторінці конвертації веб-сторінки |
| Вхідні данні | Валідна адреса веб-сторінки |
| Опис проведення тесту | Користувач вводить адресу веб-сторінки у відповідне поле та натискає кнопку "Конвертувати". Система перевіряє посилання на коректність, і якщо сторінкаа існує, вона автоматично конвертується. |
| Очікуваний результат | Система має конвертувати вказану веб-сторінку у формат PDF та надавати можливість перегляду та завантаження отриманого PDF. |
| Фактичний результат | Конвертація пройшла успішно, користувач може переглядати та завантажувати отриманий PDF. |

Таблиця 3.5 – Тест 1.5

|  |  |
| --- | --- |
| Тест | Об'єднання декількох PDF-файлів |
| Номер тесту | 1.5 |
| Початковий стан системи | Користувач знаходиться на сторінці завантаження файлів для об’єднання. |
| Вхідні данні | Файли формату PDF |
| Опис проведення тесту | Користувач завантажує PDF файли у відповідну область та натискає кнопку "Об’єднати". Система перевіряє розширення файлів, і якщо всі файли відповідають допустимим форматам, вони автоматично об’єднуються у вказаному порядку. |
| Очікуваний результат | Система має об’єднувати обрані файли в обин документ та надавати можливість перегляду та завантаження отриманого PDF. |
| Фактичний результат | Об’єднання пройшло успішно, користувач може переглядати та завантажувати отриманий PDF. |

Таблиця 3.6 – Тест 1.6

|  |  |
| --- | --- |
| Тест | Розділення PDF-файлів |
| Номер тесту | 1.6 |
| Початковий стан системи | Користувач знаходиться на сторінці завантаження файлів для їх розділення. |
| Вхідні данні | Файли формату PDF |
| Опис проведення тесту | Користувач завантажує PDF файли у відповідну область та натискає кнопку "Розділити". Система перевіряє розширення файлів, і якщо всі файли відповідають допустимим форматам, вони розділяються на окремі сторінки. |
| Очікуваний результат | Система має розділяти обрані файли на окремі сторінки та надавати можливість перегляду та завантаження отриманих PDF сторінок. |
| Фактичний результат | Розділення пройшло успішно, користувач може переглядати та завантажувати отримані PDF сторінки. |

Таблиця 3.7 – Тест 1.7

|  |  |
| --- | --- |
| Тест | Завантаження файлів на пристрій |
| Номер тесту | 1.7 |
| Початковий стан системи | Користувач знаходиться на відображення PDF файлів. |
| Вхідні данні | PDF файли |
| Опис проведення тесту | Користувач вибирає файл для завантаження, натискає кнопку "Завантажити". |
| Очікуваний результат | Система має завантажувати вказаний файл на пристрій користувача. |
| Фактичний результат | Завантаження пройшло успішно, користувач може знайти завантажений файл на своєму пристрої. |

Таблиця 3.8 – Тест 1.8

|  |  |
| --- | --- |
| Тест | Відображення профілю користувача |
| Номер тесту | 1.8 |
| Початковий стан системи | Користувач авторизований та перебуває на своїй сторінці профілю |
| Вхідні данні | - |
| Опис проведення тесту | Користувач авторизується в системі, переходить на сторінку профілю |
| Очікуваний результат | Система має відображати дані профілю та усі PDF файли користувача, збережені у БД. |
| Фактичний результат | Система успішно відображає дані профілю та усі PDF файли користувача, збережені у БД. |

## Опис контрольного прикладу

У попередньому розділі було протестовано усі реалізовані функції створеного веб-застосунку. Тому в описі контрольного прикладу будемо намагатися відтворити взаємодію користувача з веб-застосунком.

Тестування проводилось на пристрої, що відповідає рекомендованій конфігурації технічних засобів (Додаток А), з операційною системою Windows 10 та в браузері Google Chrome 119 (64-біт).

Відтворимо сценарій, де користувач хоче конвертувати декілька файлів різних типів у PDF і завантажити результат на пристрій. Для цього відкриємо у браузері адресу початкової сторінки сайту. Як бачимо на рисунку 3.2 запуск пройшов успішно, користувач бачить початкову сторінку.

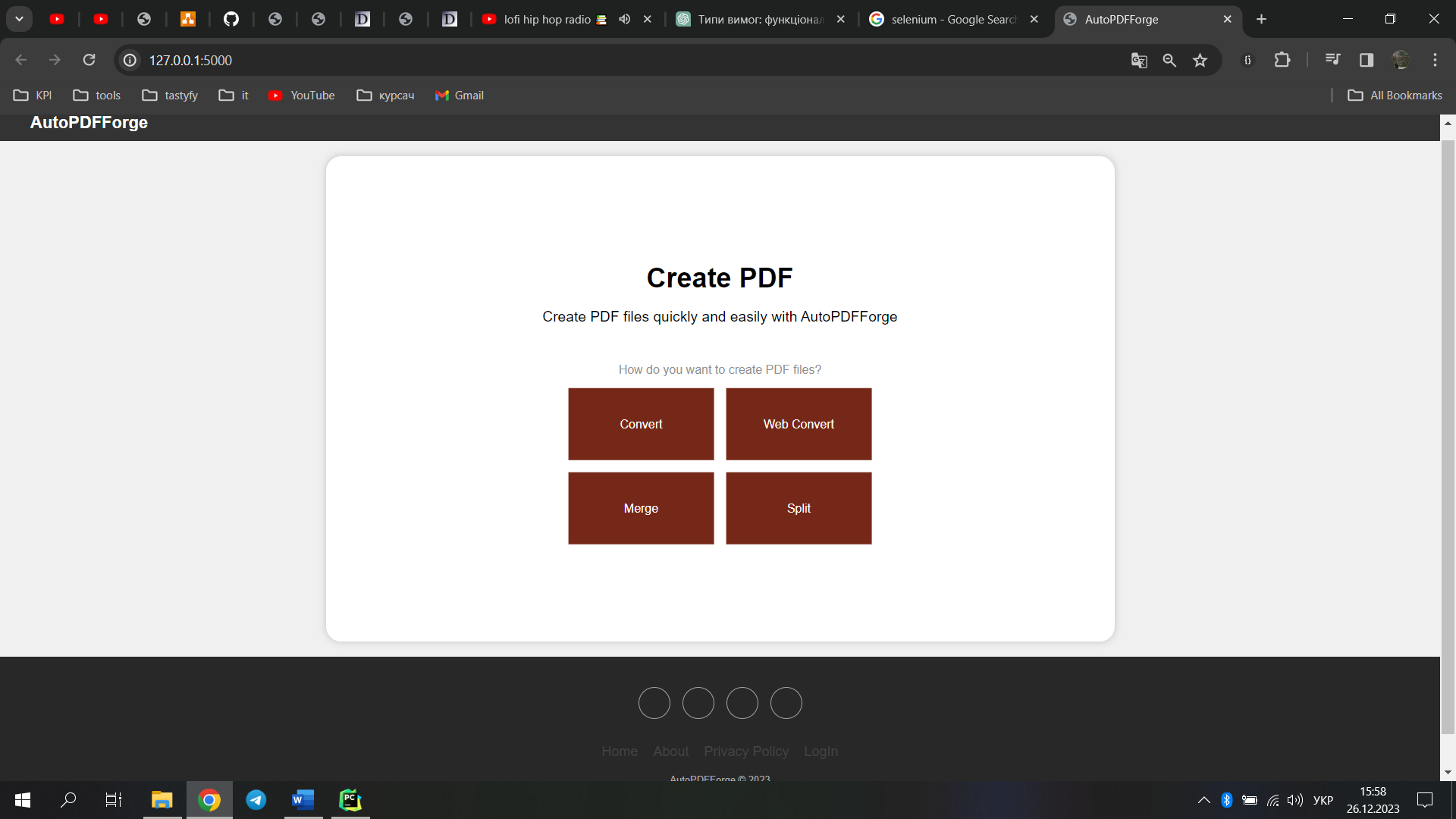


Рисунок 3.2 – Початкова сторінка

Далі перевіримо реєстрацію та авторизацію у системі з тестовими даними логіну kisellsn та паролю 12345678 (рисунок 3.3). Також продемонструємо реакцію сторінок на некоректне заповнення полів на рисунку 3.9.

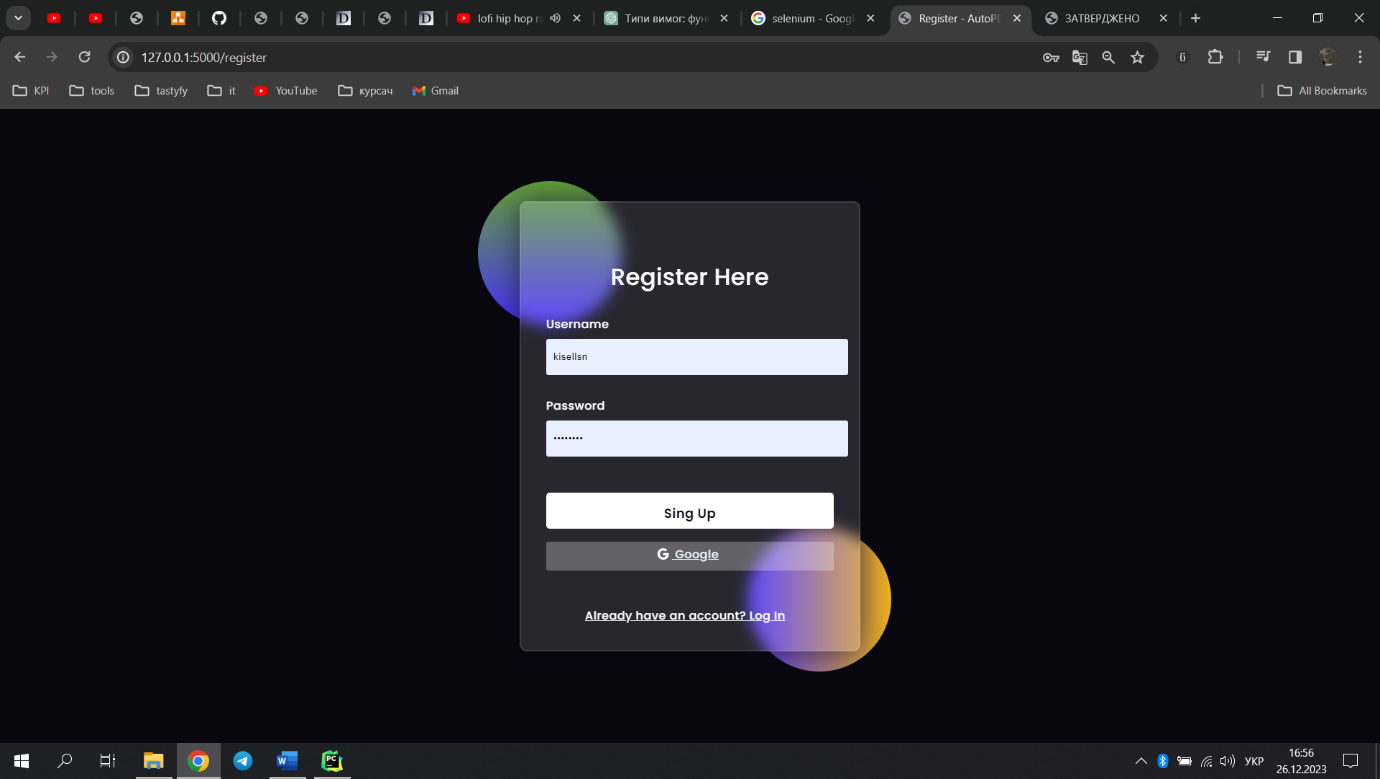
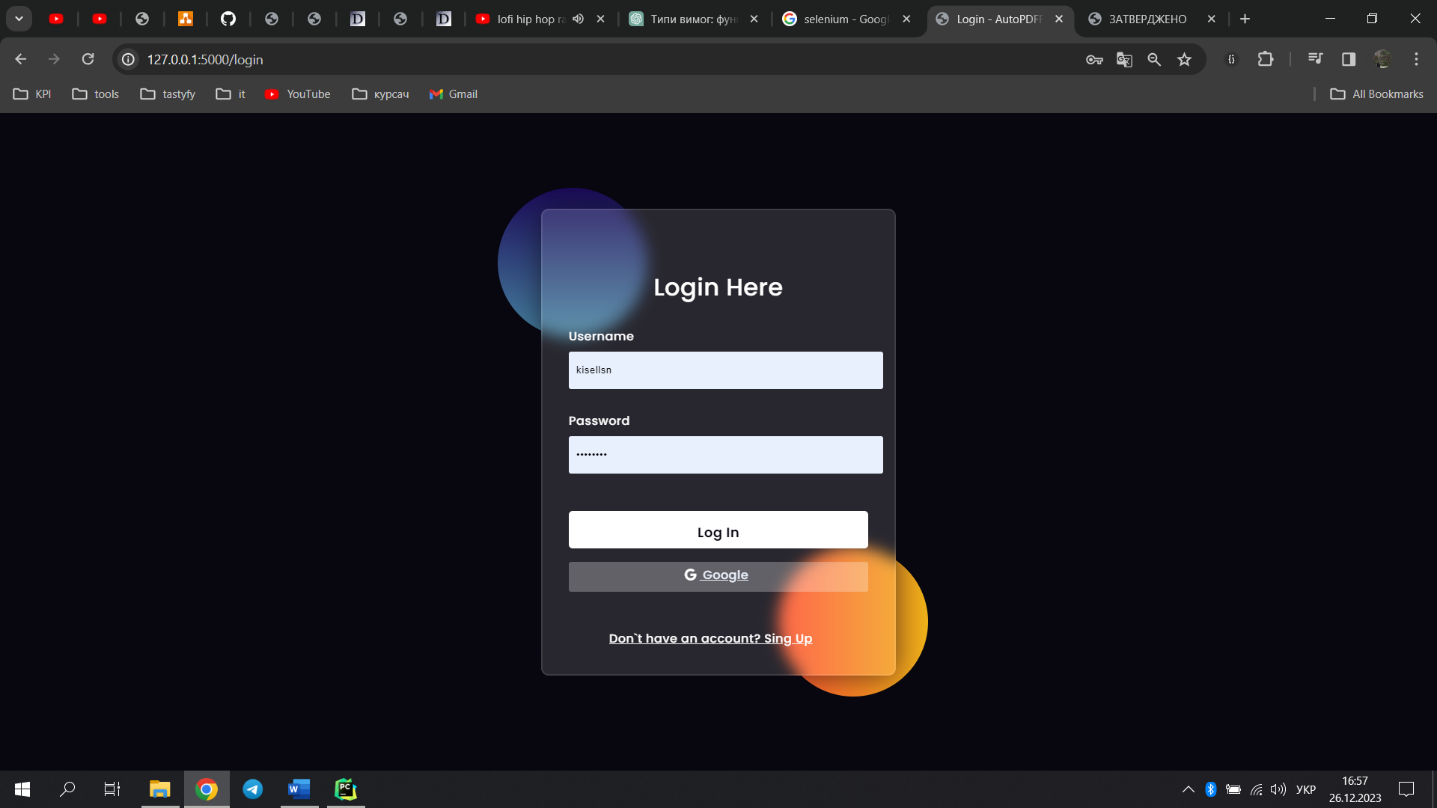
 

Рисунок 3.3 – Сторінки реєстрації та авторизації

У результаті нас перенаправило на сторінку профілю де відображаються наші дані користувача (рисунок 3.4).

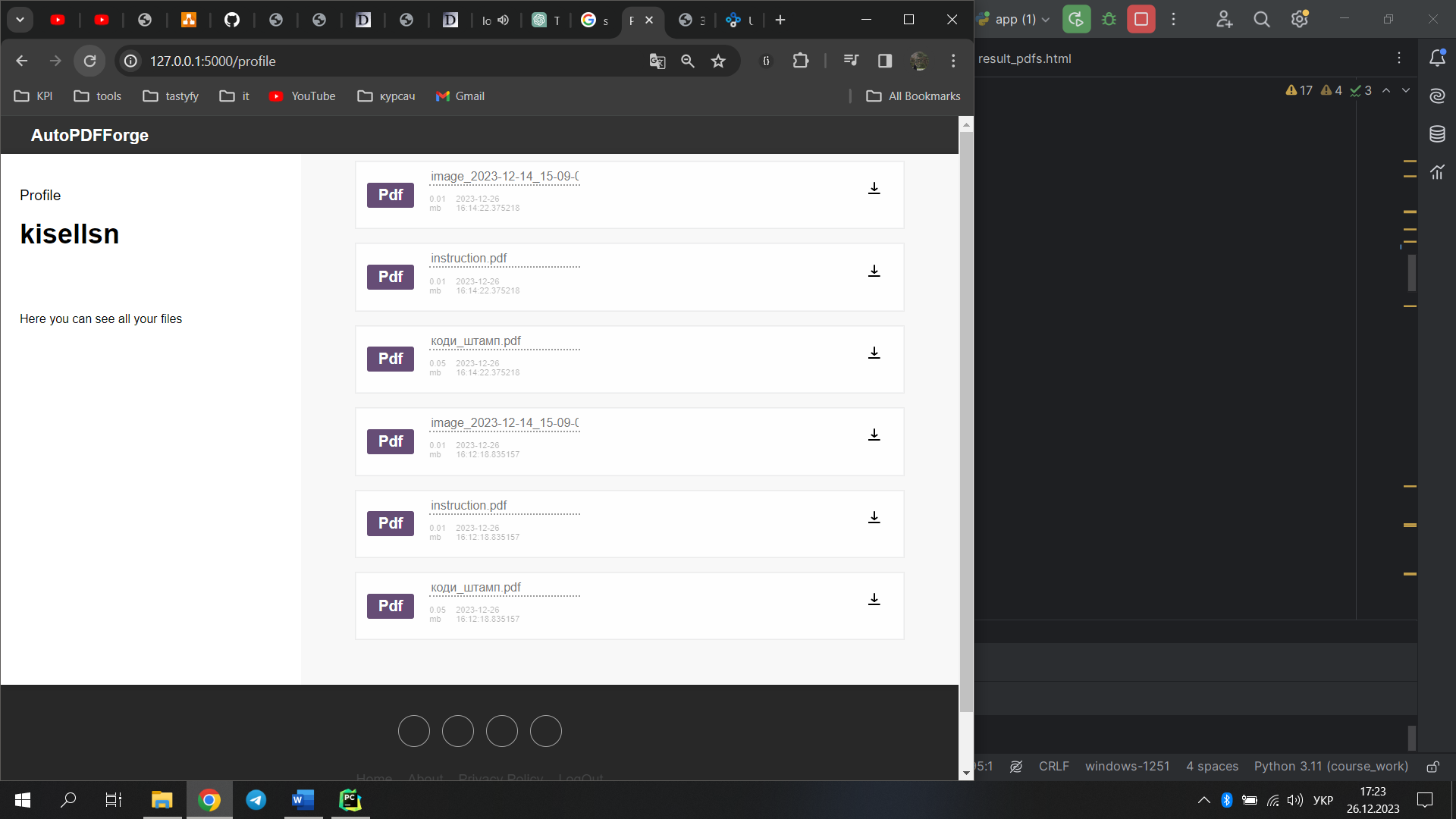


Рисунок 3.4 – Сторінка профілю

Для перевірки функціональності веб-застосунку перевіримо конвертацію файлів у пдф, для цього перейдемо на відповідну сторінку і завантажимо файли різних розширень, такі як png, txt, docx (рисунок 3.5). При недопустимих форматах (у даному випадку була спроба завантажити пдф файл) система повідомить користувача про помилку (рисунок 3.6). При розмірі файлу більше ніж допустима норма система повідомить користувача про помилку (рисунок 3.10). Результат виконання конвертації можна побачити на рисунку 3.7.

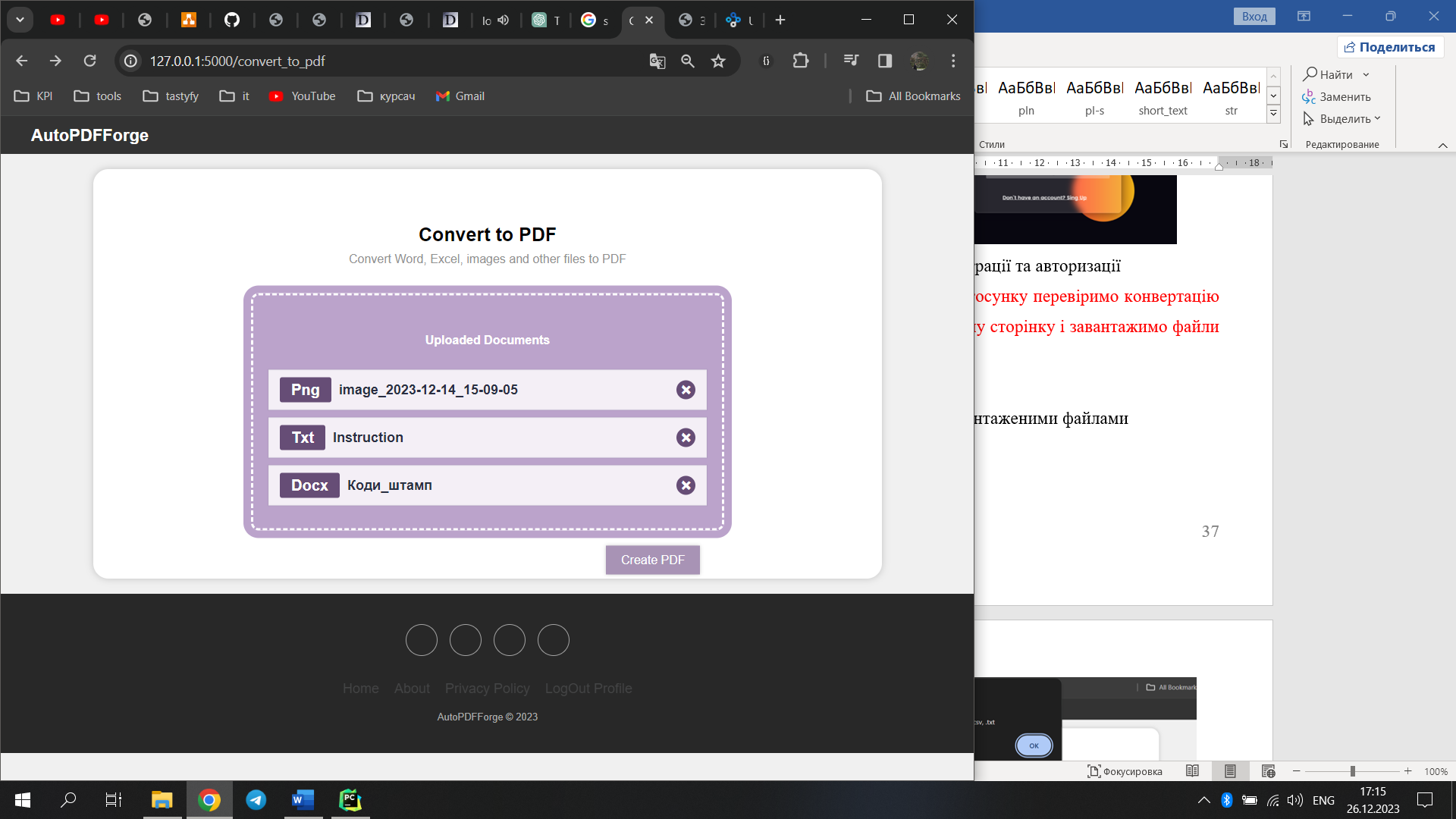


Рисунок 3.5 – Сторінка із завантаженими файлами

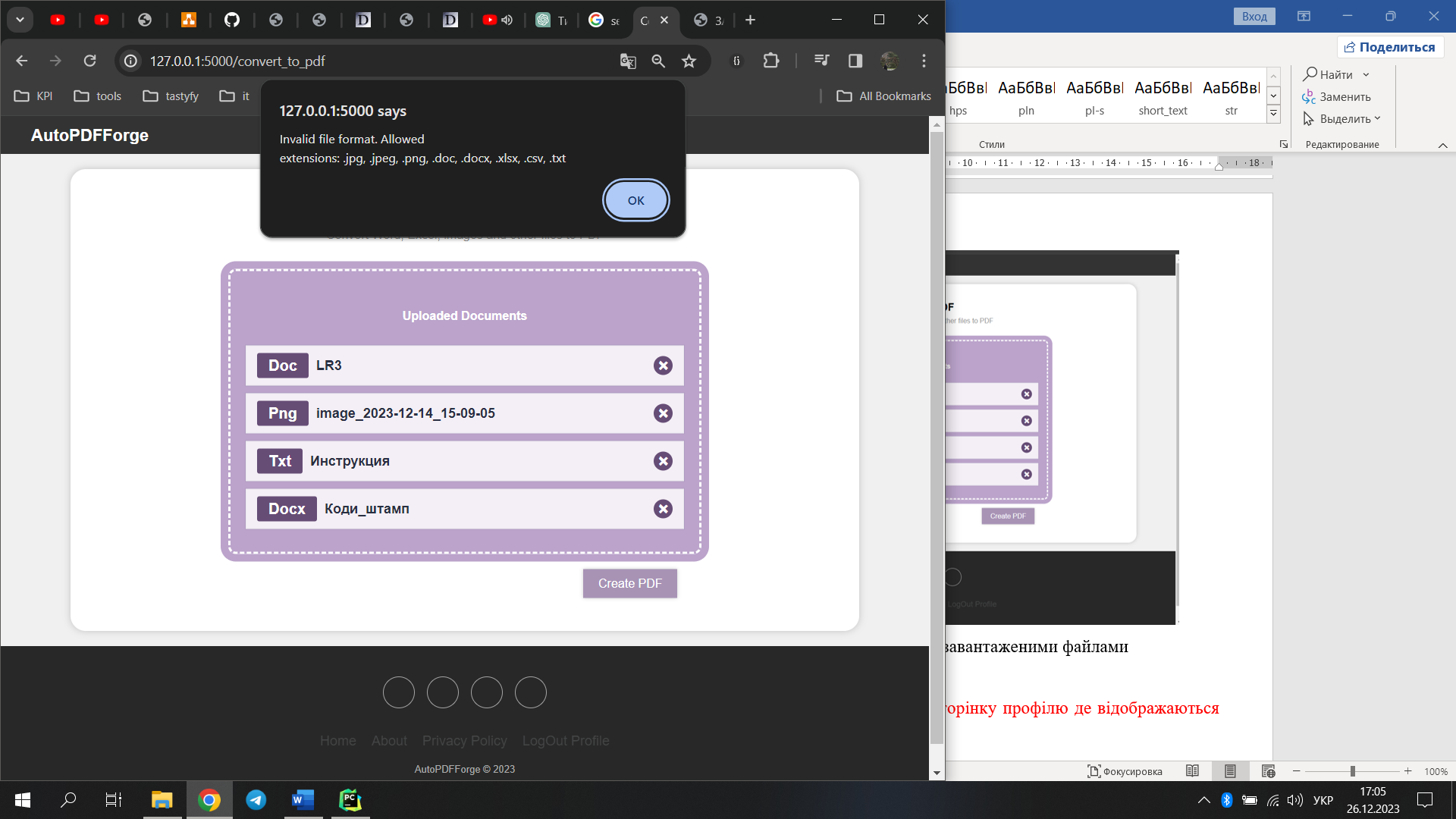


Рисунок 3.6 – Повідомлення при недопустимому форматі файлу

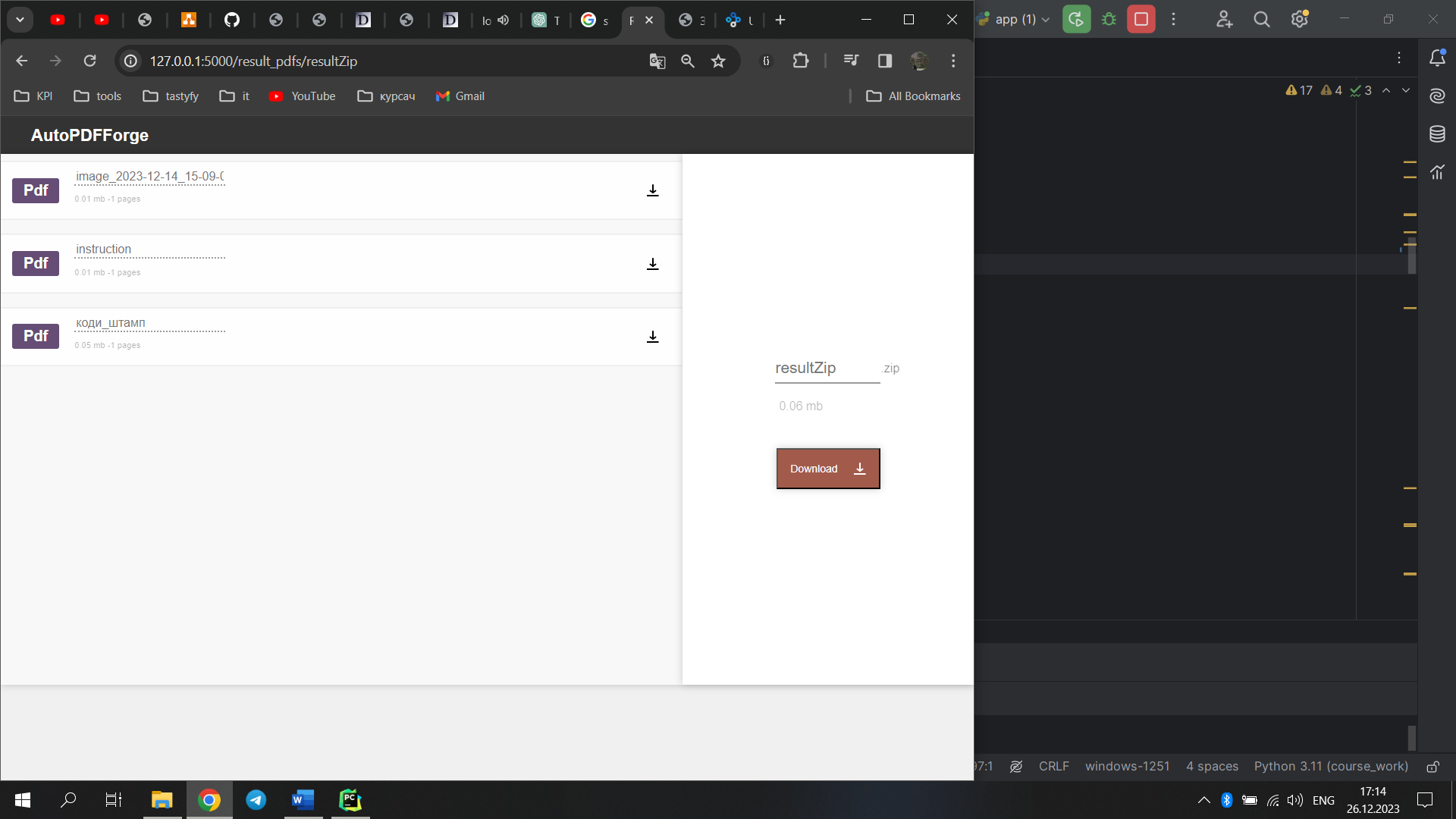


Рисунок 3.7– Конвертовані файли

Для завантаження конвертованих даних достатньо натиснути кнопу біля відповідного файлу (рисунок 3.8).

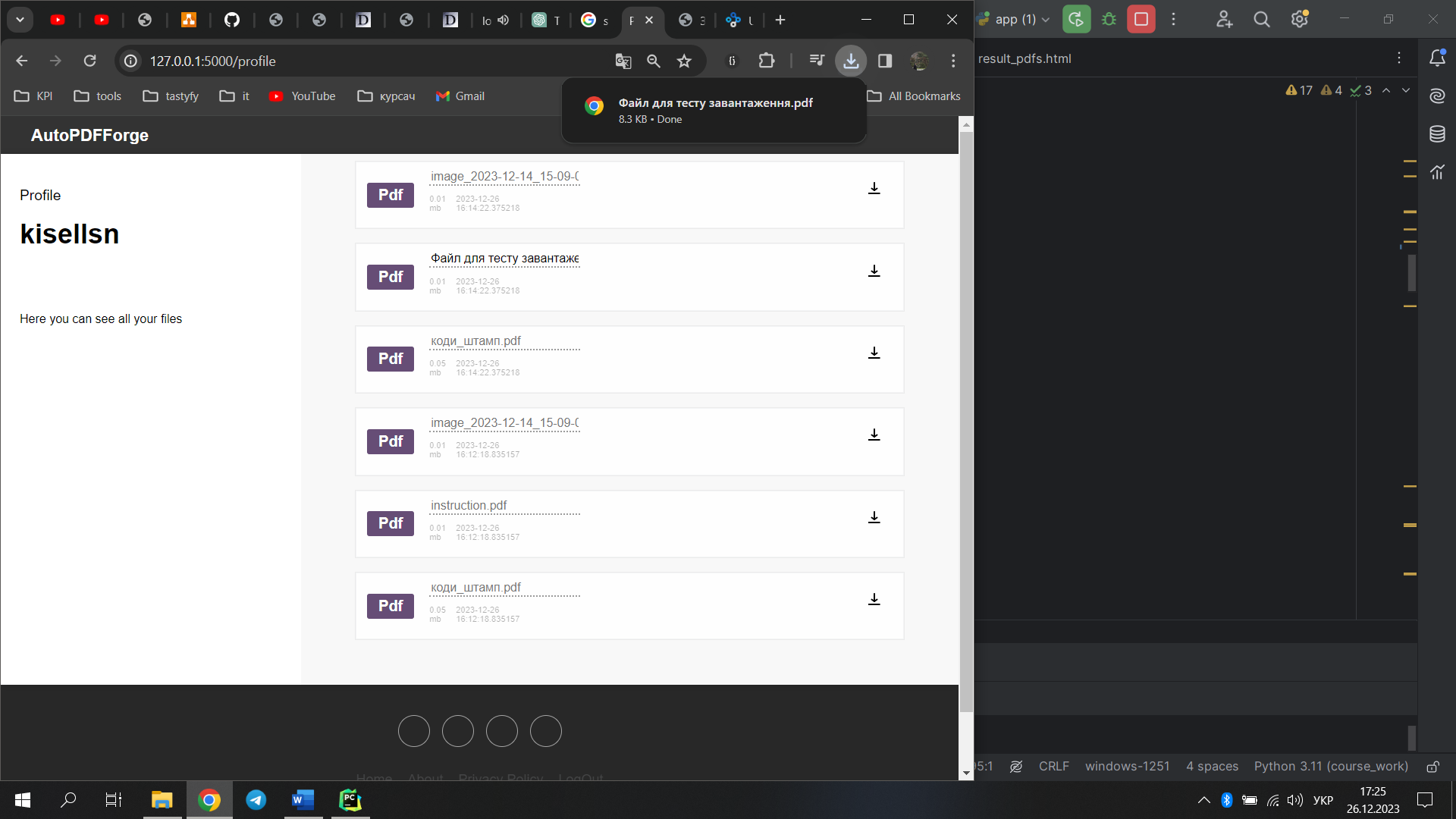
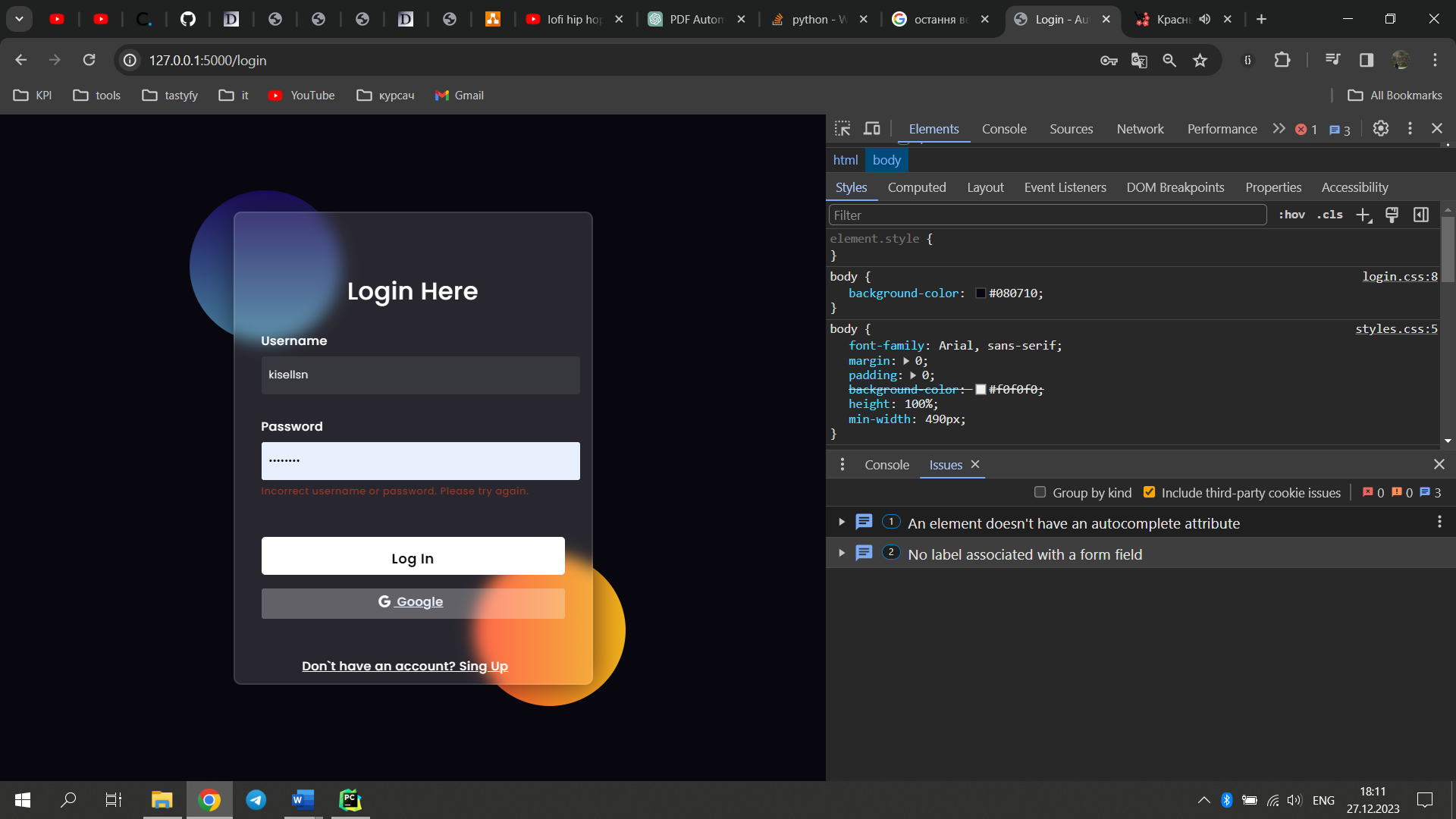


Рисунок 3.8– Завантаження файлу



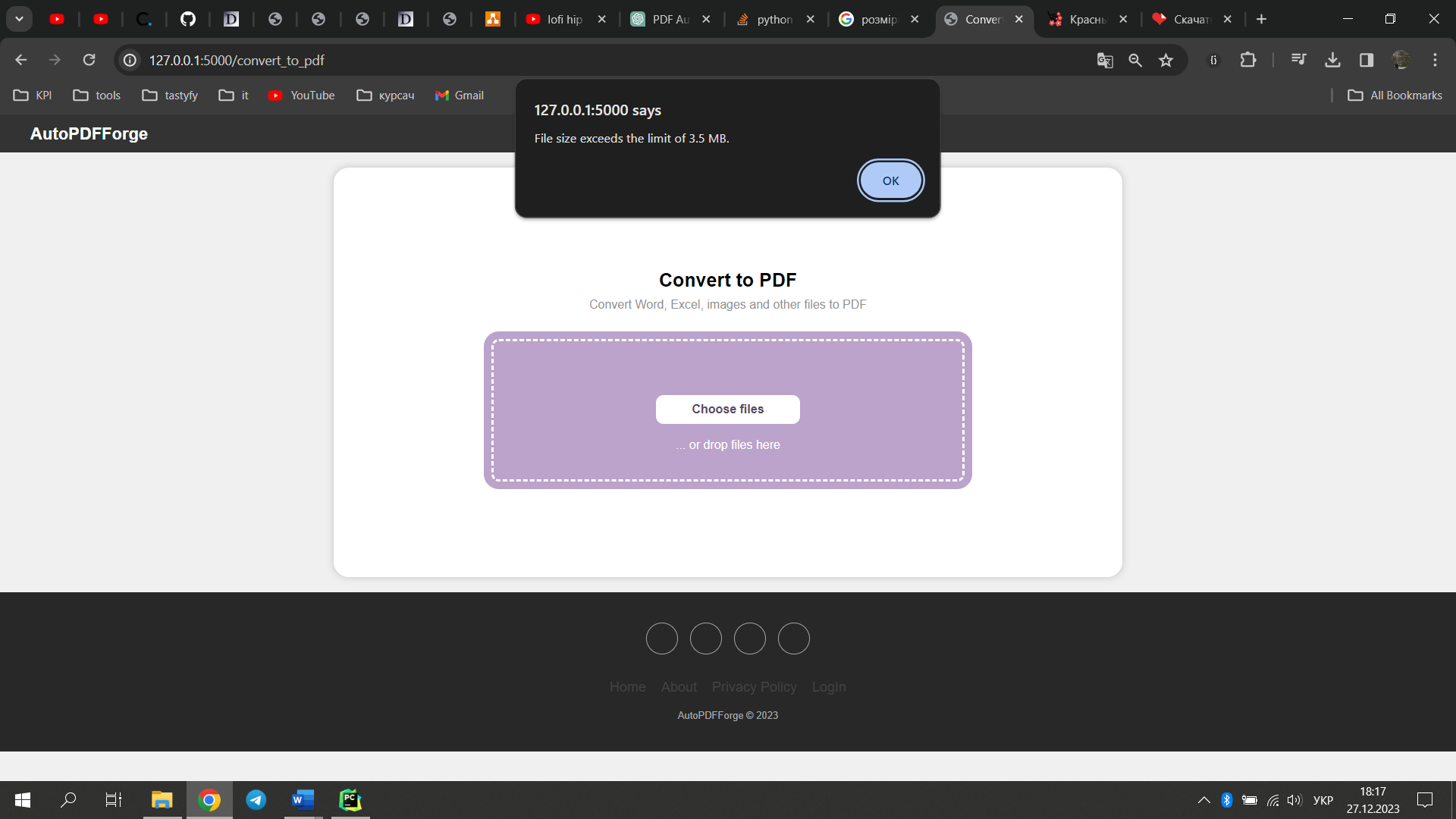


Рисунок 3.10 – Повідомлення при недопустимому розмірі файлу

## Висновки до розділу

У даному розділі проведено докладний аналіз та перевірку якості програмного забезпечення. Функціональне тестування застосунку було виконано, і жодних помилок у роботі програми не виявлено. Крім того, здійснено тестування коректної роботи клієнтської частини веб-застосунку, використовуючи браузер Chrome, і отримано позитивні результати.

У результаті роботи над цим розділом було підтверджено належну функціональність програми. Виявлені можливі проблеми були ідентифіковані та виправлені, що сприяє забезпеченню стабільної та ефективної роботи веб-додатка в цілому.

# ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Розгортання програмного забезпечення

Хоч веб-застосунок, що розглядається, має можливість запуску на різних операційних системах, його функціональність передбачає основну сумісність з операційною системою Windows 10. Таким чином, надалі буде надано кроки для розгортання на цій конкретній операційній системі. Варто відзначити, що хоча існують аналогічні технології для інших операційних систем, таких як MAC OS X чи Linux, використання їх вимагатиме додаткових знань і часу для налагодження.

Для розгортання AutoPDFForge локально потрібно виконати такі кроки:

* завантажити IDE PyCharm 2022(або новішу версію) або будь-яке інше середовище розробки для роботи на Python.
* завантажити систему баз даних SQLite.
* cклонувати webPDF\_converter репозиторій:

git clone https://github.com/kisellsn/webPDF\_converter.git

* створити .env файл та заповнити його наступними даними:

SECRET\_KEY – генерується випадковим чином та використовується для безпечного підпису сесій та інших даних.

GOOGLE\_CLIENT\_ID і GOOGLE\_CLIENT\_SECRET – ідентифікатор та секретний ключ клієнта , які використовуються для аутентифікації та авторизації додатка в системі Google. Потрібно мати аккаунт в Google Cloud Console.

GOOGLE\_DISCOVERY\_URL – “https://accounts.google.com/.well-known/openid-configuration”.

APIKEY – ключ API, який служить для ідентифікації вашого додатка при використанні API від Cloudmersive.

* створити базу даних та таблиці в ній:

python from app import db

db.create\_all()

* створити папки, необхідні для роботи:

cd backend\_func

mkdir result\_zip

mkdir resulted\_files

mkdir uploaded\_files

* запустити програму:

python app.py

Відкрити веб-застосунок можна перейшовши у браузері за посиланням http://127.0.0.1:5000/

## Підтримка програмного забезпечення

Для отримання оновлень програмного забезпечення веб-застосунку, ви можна скористатися тим самим GitHub репозиторієм, який містить актуальні дані та оновлені версії програми. Процес оновлення може бути виконаний шляхом клонування цього репозиторію та виконання відповідних дій, що описані у пункті 4.1. Перед оновленням проекту бажано окремо зберегти створені файли БД в директорії instance, щоб не втратити важливі дані.

При будь-яких змінах коду у структурі БД потрібно оновити саму БД. Для цього знадобиться:

* встановити розширення Flask-Migrate:

pip install flask-migrate

* створити migration репозиторій:

flask db init

* створити початкову міграцію:

flask db migrate

* запустити команду оновлення:

flask db upgrade

## Висновки до розділу

У даному розділі було описано кроки розгортання програмного забезпечення веб-застосунку та вказано, які компоненти повинні бути встановленні заздалегідь на ПК, щоб успішно запустити програму.

# ВИСНОВКИ

У ході виконання курсової роботи було успішно спроєктовано та реалізовано веб-застосунок для автоматизованого формування \*.pdf файлів, відповідно до поставлених завдань та визначених у технічному завданні. Програма була створена в середовищі розробки PyCharm, а у якості БД використано SQLite. Головною метою даної системи є полегшити роботи з PDF файлами, що включає використання таких функцій як конвертація, об'єднання та розділення.

У першому розділі було наведено приклади готових існуючих рішень, описано їх головні переваги та недоліки, визначено основні функціональні вимоги, визначено основні нефункціональні вимоги, визначено основні функції модуля, створену діаграму вимог та побудовано таблицю варіантів використання даного додатку.

В другому розділі описано архітектуру програмного продукту, основні принципи, які було використано при розробці, технології, за допомогою яких розробили продукт, побудовано систему зберігання даних користувача та його файлів, описано використання захисту даних, який використовується в системі.

У третьому розділі було проведено тестування етапів застосунку, з детальним описом кожного тесту та результатів його виконання. А також було проведено ретельний аналіз якості програмного застосунку. У результаті вивчення цього розділу було підтверджено належну функціональність програми. Виявлені можливі проблеми були ідентифіковані та виправлені.

В четвертому розділі було описано розгортання програмного продукту, що треба для роботи з ним локально.

Під час реалізації веб-застосунку, відзначено його успішне тестування на пристроях з різними версіями та розмірами екранів. Це підтверджує його адаптованість та ефективність на різних пристроях, що є важливим аспектом для користувачів з різних категорій.

Підсумовуючи результати, можна зазначити, що основна мета була досягнута, а система успішно реалізує ряд важливих функцій, спрямованих на полегшення та прискорення роботи з PDF-документами. Досягнуті результати підтверджують доцільність та практичну значущість роботи. Зокрема, використання розробленого веб-застосунку можливе в різних галузях, де потрібна автоматизована генерація \*.pdf документів, таких як бізнес-звіти, освітні матеріали, медичні звіти тощо.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. The Future of Web Apps [Електронний ресурс] - https://www.computer.org/csdl/magazine/it/2011/05/mit2011050012/13rRUx0xPBi
2. Історія виникнення веб-сайта [Електронний ресурс] - <https://prog.bobrodobro.ru/39686>
3. Web application [Електронний ресурс] - <https://www.britannica.com/topic/Web-application>
4. What is a web application - what you need to know in 2023 [Електронний ресурс] - <https://www.lightflows.co.uk/blog/what-is-a-web-application-what-you-need-to-know-in-2023/#:~:text=Web%20applications%20can%20be%20used,access%20to%20information%20and%20services>
5. Що таке JavaScript та навіщо його вчити? [Електронний ресурс] - https://wezom.academy/ua/chto-takoe-javascript-i-zachem-ego-uchit/
6. Full Stack Python [Електронний ресурс] - https://www.fullstackpython.com/flask.html
7. The (not so) secret power of BPMN [Електронний ресурс] - https://camunda.com/bpmn/
8. Flask-sqlalchemy [Електронний ресурс] - https://flask-sqlalchemy.palletsprojects.com/en/3.1.x/
9. Client Server Architecture – Detailed Explanation [Електронний ресурс] - <https://www.interviewbit.com/blog/client-server-architecture/>
10. Cross Site Scripting (XSS) [Електронний ресурс] - <https://owasp.org/www-community/attacks/xss/#:~:text=Overview,to%20a%20different%20end%20user>.
11. What is SQLite? [Електронний ресурс] – <https://www.sqlite.org/index.html>.
12. Flask-Cors Documentation [Електронний ресурс] – <https://readthedocs.org/projects/flask-cors/downloads/pdf/latest/>
13. Count LOC [Електронний ресурс] – <https://codetabs.com/count-loc/count-loc-online.html>
14. LOC (Lines of Code) [Електронний ресурс] — https://buildpulse.io/blog/differentways-of-measuring-code-complexity.
15. Трасування вимог [Електронний ресурс] — <https://studfile.net/preview/10015384/page:4/>
16. PyCharm [Електронний ресурс] — https://www.jetbrains.com/pycharm/
17. Для чого потрібна Microsoft Visual Studio? [Електронний ресурс] — https://soft360.pl/blog/post/microsoft-visual-studio-do-czego-sluzy.html
18. What Is Python? [Електронний ресурс] — https://aloa.co/blog/why-python#:~:text=Ease%20Of%20Learning%20And%20Use,easy%20learning%20curve%20for%20everyone.
19. 6 Reasons Why Flask is Better Framework for Web Application Development [Електронний ресурс] — https://able.bio/hardikshah/6-reasons-why-flask-is-better-framework-for-web-application-development--cd398f73
20. Client-Side Vs. Server-Side Rendering [Електронний ресурс] — https://www.searchenginejournal.com/client-side-vs-server-side/482574/
21. PDF24 [Електронний ресурс] — https://www.pdf24.org/en/
22. Smallpdf [Електронний ресурс] — https://smallpdf.com/uk
23. Sodapdf [Електронний ресурс] —https://online.sodapdf.com/
24. Fetch API [Електронний ресурс] —https://developer.mozilla.org/en-US/docs/Web/API/Fetch\_API