Politechnika
Śląska

PRACA MAGISTERSKA

Moduł do obsługi importów danych finansowych z plików PDF

Mateusz MARCZEWSKI
Nr albumu: 282700

Kierunek: Informatyka
Specjalność: Internet i technologie sieciowe

PROWADZĄCY PRACĘ
Dr hab. inż. Arkadiusz Biernacki
KATEDRA Sieci i Systemów Komputerowych
Wydział Automatyki, Elektroniki i Informatyki

Gliwice 2023

Tytuł pracy

Moduł do obsługi importów danych finansowych z plików PDF

Streszczenie

Głównym celem tego projektu jest opracowanie praktycznego oprogramowania, które automatyzuje proces wydobycia istotnych danych z tabel zawartych w dokumentach o formacie *PDF* (Portable Document Format). Wybór formatu *PDF* jako głównego celu wynika z jego szerokiego zastosowania i dostępności licznych zbiorów danych prezentowanych w danym typie plików.

Słowa kluczowe

PDF, ekstrakcja danych, Adobe, wykrywanie tabel, ekstrakcja tabel

Thesis title

Module for importing inventory data from PDF files

Abstract

The main goal of this project is to develop practical software that automates the process of extracting relevant data from tables contained within documents in *PDF* (Portable Document Format) format. The choice of the *PDF* format as the primary target stems from its wide usage and the availability of numerous datasets presented in this file type.

Key words

PDF, data extraction, Adobe, table detection, table extraction

Spis treści

Rozdział 1

Wstęp

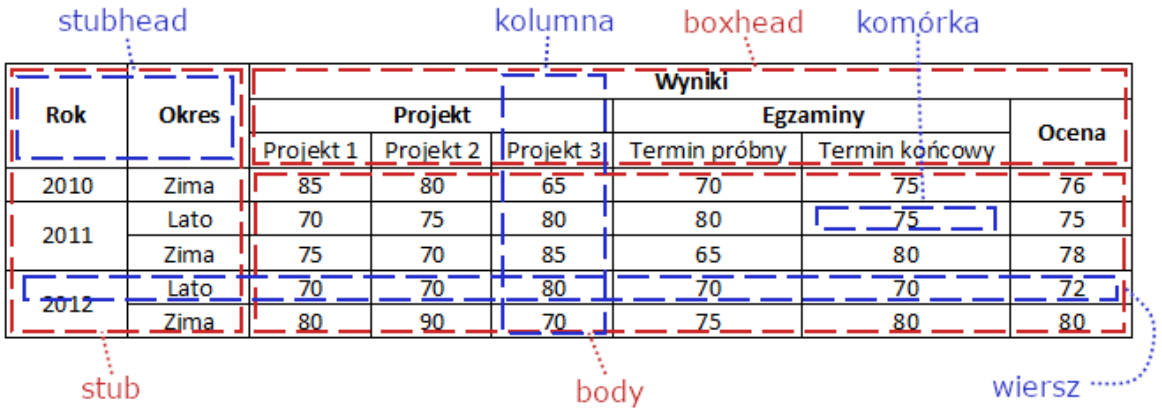
1.1 Wprowadzenie w zagadnienie

Importowanie danych z różnych źródeł, takich jak faktury zamówień, potwierdzenia sprzedaży, dane o załadunkach i tym podobne, a następnie importowanie ich do firmowych baz danych, jest problemem, nad którym chce się pochylić niniejsza praca. Jako sposób prezentacji takich danych często używane są tabele, jest to intuicyjny i bardzo efektywny sposób zapisu dużych zbiorów danych. Jest to metoda stosowana niezależnie od tego jaki typ danych chcemy przedstawić, z tego powodu nie ma żadnego standardu określającego jak takie tabele powinny być przedstawiane. Często różnią się one w zależności od celu i preferencji twórcy, jednak w obecnych czasach cyfryzacji, zazwyczaj będą one również dostępne w formie plików *PDF* (Portable Document Format), lub będzie możliwe ich do tej postaci zeskanowanie. Znacznie ułatwia to ich przechowywanie i podgląd, jednak utrudnia ekstrakcje danych z takich plików.

W celu rozwiązania tego problemu, wiele firm wykorzystuje specjalnie przygotowane aplikacje, które pozwalają na automatyzację procesu importowania danych z plików *PDF*. Takie aplikacje mogą importować dane w postaci plików *PDF*, a następnie wykrywać w nich najważniejsze dane i eksportować je do żądanego formatu, obsługiwanego przez bazę danych lub inne aplikacje służące do zarządzania danymi. Takie aplikacje mogą przechowywać dane w własnym formacie, i pozwalać na łatwy dostęp do zasobów firmy, nawet bez znajomości działania baz danych.

1.2 Cel pracy

Celem tej pracy jest utworzenie własnej aplikacji, która umożliwi łatwe wyeksportowanie danych z tabel znajdujących się w plikach o rozszerzeniu *PDF*, do łatwiej obsługiwanych plików wyjściowych w formacie *CSV* (comma-separated values). Aplikacja zostanie stworzona w taki sposób, aby umożliwiała samodzielne użytkowanie programu



Rysunek 1.1: Budowa tebeli z wyszczególnieniem omówionych elementów

bezpośrednio z dokumentami PDF, a także wykorzystywanie jej jako część innych narzędzi i projektów. Wyjście stworzonej aplikacji oprogramowania ma umożliwić łatwą, dalszą obsługę wyeksportowanych danych, dlatego zdecydowano się wykorzystać wyżej wspomniany format CSV. Cały kod źródłowy tego projektu pracy dyplomowej zostanie udostępniony online na platformie www.github.com

1.3 Budowa Tabeli

Jednym z częściej spotykanych modeli koncepcyjnych tabeli jest ten zaproponowany przez Wanga [10], a później rozbudowany przez Hursta [**Hurst**]. Wang definiuje tabelę jako podzieloną na cztery główne obszary:

- the stub czyli pień, jest to kolumna, która zawiera nagłówki wierszy i podnagłówki;
- boxhead tworzący pojemnik na nagłówki kolumn (z wyłączeniem nagłówka kolumny)
- stubhead, nagłówek pierwszej kolumny, który zawiera nagłówek dla kolumny,
- body czyli główną treść tabeli, część ta zawiera faktyczne dane tabeli.

W niniejszej pracy magisterskiej definicje Wanga zostały nieco dostosowane, tak aby nagłówek kolumny był uważany za część kolumny. Warto wspomnieć, że oczywiście nie każda tabela jest tak samo zbudowana. Na przykład, dla dużej liczby tabel, nagłówki kolumn nie są zebrane w jednej ramce. Oprócz tych definicji, praca magisterska korzysta z definicji takich elementów tabeli jak: nagłówek, kolumna, wiersz, komórka. Rysunek 1.1 ilustruje te definicje.

1.4 Charakterystyka rozdziałów

W niniejszej pracy naukowej przedstawiamy szereg rozdziałów, które składają się na kompleksową analizę modułu do importowania danych inwentaryzacyjnych z plików *PDF*. Każdy z tych rozdziałów ma swoje unikalne znaczenie i koncentruje się na konkretnych aspektach badania. Poniżej przedstawiamy krótką charakterystykę poszczególnych rozdziałów:

Rozdział 2: Definiowanie wymagań

W tym rozdziale skupiamy się na ustaleniu konkretnych wymagań dla modułu lub biblioteki służącej do importowania danych z plików *PDF*. Analizujemy różne typy danych, które mają zostać zaimportowane, oraz identyfikujemy pola w plikach *PDF*, które będą przechwytywane. Przeanalizujemy również inne formaty plików, z którymi moduł będzie pracował, aby zapewnić wszechstronność i elastyczność rozwiązania.

Rozdział 3: Badanie istniejących rozwiązań

W tym rozdziale skoncentrujemy się na dokładnym przejrzeniu istniejących modułów i bibliotek dostępnych na rynku do importowania danych z plików *PDF*. Przeanalizujemy funkcje, jakie oferują te rozwiązania oraz przetestujemy ich możliwości. Celem tego rozdziału jest zgłębienie istniejących rozwiązań i dostrzeżenie ich mocnych i słabych stron w celu stworzenia solidnej podstawy do dalszych badań.

Rozdział 4: Ocena dostępnych opcji

W tym rozdziale przeprowadzimy szczegółową ocenę dostępnych opcji na podstawie wcześniej zdefiniowanych wymagań. Porównamy różne rozwiązania pod kątem ich zgodności z określonymi wymaganiami, aby wybrać najlepszą opcję dla dalszych badań i implementacji. Porównane zostaną również nasza opinia oraz opinie i oceny dostępne w artykułach naukowych, w celu poprawnej analizy rozwiązania.

Rozdział 5: Rozwój istniejących rozwiązań

W tym rozdziale skupimy się na rozwoju istniejących rozwiązań, takich jak dostosowanie ich do specyficznych potrzeb projektu. Przeanalizujemy również wymagane zasoby i umiejętności potrzebne do kontynuacji rozwoju i utrzymania modułu opartego na danej bibliotece.

Rozdział 6: Rozwój własnego rozwiązania

Ten rozdział koncentruje się na własnym rozwiązaniu, uwzględniając wyniki poprzednich analiz i badań. Przeanalizujemy wyniki, które udało się osiągnąć przy wykorzystaniu własnego rozwiązania i porównamy je z istniejącymi na rynku.

Rozdział 7: Dokumentacja i udostępnianie wybranego rozwiązania

W tym rozdziale omówimy proces tworzenia dokumentacji dla wybranego rozwiązania, aby umożliwić innym użytkownikom skorzystanie z modułu. Przeanalizujemy również możliwość udostępnienia rozwiązania jako projektu open-source dla społeczności programistycznej.

Rozdział 8: Podsumowanie

W ostatnim rozdziale dokonamy podsumowania całego badania, podkreślając najważniejsze wnioski i rezultaty. Przedstawimy również implikacje wynikające z naszej pracy badawczej oraz sugestie dotyczące przyszłych kierunków rozwoju i badań w tej dziedzinie.

Każdy z tych rozdziałów przyczynia się do pełnego zrozumienia problemu i tworzy spójną strukturę pracy badawczej dotyczącej modułu do importowania danych inwentaryzacyjnych z plików PDF.

Rozdział 2

Definiowanie wymagań

2.1 Formaty plików występujące w pracy

W tym miejscu opisane zostaną formaty plików, z których korzystano podczas tworzenia te pracy, to jest, *PDF* oraz *CSV*.

2.1.1 Pliki PDF

Format *PDF* (Portable Document Format) jest zbudowany zgodnie z określonymi specyfikacjami opracowanymi przez firmę Adobe Systems na początku lat 90. Format ten ma na celu zapewnienie niezależności od platformy i zachowanie spójności formatowania, niezależnie od urządzenia, na którym jest wyświetlany[3]. Specyfikacja *PDF* została udostępniona bezpłatnie w 1993 roku, ale pozostała jako format własnościowy, aż w 2008 roku została oficjalnie wydana jako otwarty standard (ISO 32000-1:2008)[1] Ta licencja przyznaje bezpłatne prawa licencyjne do wszystkich patentów należących do Adobe, które są niezbędne do tworzenia, używania, sprzedaży i dystrybucji implementacji zgodnych z formatem *PDF* [4]. Oprócz tych cech, pliki *PDF* oferują dobrą kompresję, zmniejszając rozmiar pliku i czyniąc format idealnym do dystrybucji online. Logo Adobe PDF jest pokazane na Rys. 3.2.

Rysunek 2.1: Logo formatu PDF firmy Adobe jest rozpoznawalne dla wielu osób ze względu na popularność [2]

W pierwszej kolejności warto zwrócić uwagę na różnice między starszymi a nowszymi wersjami formatu *PDF*. Pliki *PDF* są stale rozwijane i udoskonalane, dlatego ważne jest, aby zrozumieć, jakie funkcje i możliwości są dostępne w różnych wersjach. Niektóre starsze wersje mogą nie obsługiwać niektórych zaawansowanych funkcji, co należy wziąć pod uwagę przy tworzeniu modułu importującego.

Kolejnym aspektem jest analiza możliwości skryptowania i interaktywności w plikach *PDF*. Niektóre pliki *PDF* mogą zawierać skrypty JavaScript lub elementy interaktywne, takie jak formularze, przyciski, linki itp. Istotne jest zrozumienie tych elementów i określenie, czy i w jaki sposób można je obsłużyć podczas importowania danych.

Dodatkowo, badamy w tym typie dokumentów wykorzystuje się różne technologie i standardy związane z potrzebami użytkownika. Są nimi standardy takie jak PDF/A (standard archiwizacji), PDF/X (standard dla publikacji drukowanych) czy PDF/UA (standard dostępności). W zależności od specyfiki aplikacji i wymagań postawionych przed rozwiązaniem, konieczne może być uwzględnienie tych standardów i zapewnienie zgodności modułu importującego.

Plik *PDF* składa się z różnych elementów i struktur, które razem tworzą dokument. Jego strukturę możemy podzielić na 4 części:

- Header - informacje o strukturze i metadanych pliku *PDF*, takich jak wersja specyfikacji *PDF*, typ pliku, używane czcionki, rozmiar strony itp.

- Body - treść dokumentu, taką jak tekst, obrazy, grafiki, tabele itp. Ciało składa się z sekwencji obiektów PDF, które są odpowiedzialne za przechowywanie danych i struktury dokumentu.

- Xref table - spis wszystkich obiektów, używanych w dokumencie. Każdy obiekt ma unikalny numer identyfikacyjny i jest przechowywany w formacie klucz-wartość.

- Trailer - informacje o lokalizacji i numerze generacji wszystkich obiektów w pliku *PDF*. Jest to wykorzystywane do odnalezienia i odzyskania obiektów podczas odczytu lub modyfikacji pliku.

Trudność przy obsłudze plików typu *PDF* jest zauważalna przy przeanalizowaniu tej struktury. Zazwyczaj pliki te odczytuje się od końca, wynika to z tego, że na końcu zawarte są informacje o wersji zapisu, oraz położeniu najważniejszych struktur.

Analiza formatów plików *PDF* pozwala nam lepiej zrozumieć złożoność i różnorodność plików, które będą importowane do tworzonej aplikacji. Pozwala to na dokładne określenie wymagań tworzonego modułu i odpowiednie dostosowanie go do różnych formatów plików *PDF*, które będzie obsługiwać.

2.1.2 Pliki CSV

W pliku *CSV* dane są przechowywane w formie tabeli, gdzie każdy wiersz reprezentuje rekord, a poszczególne pola w rekordzie są oddzielane separatorem, najczęściej przecinkiem, jak sugeruje sama nazwa tego formatu, jednak inne separatory, takie jak średnik, również są stosowane w poszczególnych przypadkach. Pliki *CSV* wyróżnia to, że mimo możliwości przechowywania złożonych tabel z danymi, jest on formatem tekstowym. Oznacza to, że dane są zapisywane jako tekst, bez złożonej struktury danych, a dopiero następnie można je wyświetlać w postaci tabel lub arkuszy. Można w nich przechowywać dzięki temu różne rodzaje danych, takie jak tekst, liczby, daty itp. bez znacznego zwiększenia ich objętości pamięciowej. Wyżej wspomniane wariacje w postaci wykorzystanych separatorów, pozwalają na zawieranie dowolnych znaków specjalnych, takich jak przecinki w danych liczbowych.

2.2 Przegląd typów danych do importowania

Plik *PDF* może zawierać różne formy zapisu informacji, takie jak tekst, tabele, obrazy, formularze itp. W celu efektywnego importowania danych z tych plików do innego formatu, konieczne jest zidentyfikowanie i zrozumienie charakterystyki tych typów danych.

Przede wszystkim, analizujemy sposoby ekstrakcji tekstu z plików *PDF*. Tekst może występować zarówno w postaci prostej, takiej jak akapity, nagłówki lub stopki. Jednak często występuje również w bardziej złożonej postaci np. listy, tabele.

Kolejnym istotnym typem danych są tabele. Pliki *PDF*, które będziemy analizować zawierają tabele z danymi, które muszą być zaimportowane do innego formatu w taki sposób, aby zachować strukturę tabelaryczną. Celem tej pracy jest przedstawienie metody identyfikacji tabel oraz ekstrakcji danych z nich, co zostanie uczynione w kolejnych rozdziałach. Tabele jakie mogą występować w plikach *PDF* to zarówno prostsze zbiory danych bez łączeń komórek, jak i bardziej zaawansowane struktury tabelaryczne, różnią się one przede wszystkim położeniem komórek. Nie wszystkie komórki muszą zawierać opis jaki typ danych jest w nich zapisany, dodatkowo zapis ten może być niespójny jeżeli tabela byłaby uzupełniana przez innych pracowników.

W plikach *PDF* mogą również występować obrazy, formularze, lub inne struktury, które są zapisane w części Xref pliku *PDF*, przez co możemy je łatwo ominąć podczas ekstrakcji tabeli.

Przegląd typów danych do importowania umożliwia nam lepsze zrozumienie różnorodności informacji zawartych w plikach *PDF* oraz wyznaczenie kierunku dalszych badań i rozwoju modułu importującego.

2.3 Problemy związane z importowaniem danych

Cały proces ekstrakcji danych opiera się na wcześniejszym zdefiniowaniu informacji, na których będą operować algorytmy. Biblioteka o nazwie pdfminer[9] została wykorzystana w celu ekstrakcji tekstu z plików *PDF*, a algorytmy wykorzystane w projekcie, przetwarzają i interpretują informacje o składnikach pliku i w wypadku potrzeby wyeksportowania tekstu, przekazują je do podanej biblioteki. Dostępne dane nie są tak szczegółowe, jak by się mogło wydawać. Na przykład, dany blok zawierający słowo nie ma dostępnych informacji o użytej czcionce ani stylu czcionki. Podczas wyodrębniania tekstu z pliku *PDF*, proces wyodrębniania zazwyczaj traktuje cały tekst jako znajdujący się na tym samym poziomie linii bazowej.

Tekst w dokumencie *PDF* jest otrzymywany dla każdej pojedynczej strony w postaci prostokątnych obszarów zawierających pojedyncze słowo tekstu. Każdy prostokątny obszar elementu jest zdefiniowany za pomocą współrzędnych, co nadaje mu dokładne i jednoznaczne położenie oraz rozmiar (szerokość i wysokość) na stronie. Boki pudełek tekstowych zawsze są wyrównane do krawędzi stron; innymi słowy, nie ma przechylonych ani skośnych prostokątów. Formatowanie tekstu takie jak tekst w postaci indeksu górnego lub dolnego, wykorzystane czcionki, pogrubienia, pochylenia itp. zazwyczaj osiąga się poprzez metryki czcionek i dostosowania pozycji, a nie poprzez wyraźne znaczniki.

Ponadto, nie ma żadnych informacji na temat tego, czy dane słowa są częściami zdania, tabeli czy innej struktury. Asocjacja słów i tworzenie zdań (oraz komórek tabeli) musi być wykonane przez algorytmy ekstrakcji danych poprzez analizowanie i przetwarzanie elementów na stronie. Jeśli słowo jest rozdzielone myślnikiem na końcu linii i kontynuuje na kolejnej, jest ono podzielone na dwie zupełnie niepowiązane ramki tekstu.

Współrzędne ramek tekstu są zapisywane za pomocą punktów. W formacie *PDF* punkt to typograficzna jednostka miary, która jest definiowana jako 1/72 cala. Ta spójna jednostka jest wykorzystywana w plikach *PDF* do pozycjonowania i określania nie tylko pozycji i rozmiaru tekstu ale również innych elementów, w tym obrazów i tabel. Użycie punktów jako jednostki pomiarowej w plikach *PDF* zapewnia spójność układu i wizualnego wyglądu treści bez względu na urządzenie wyświetlające lub drukujące. Poniżej znajduje się przeliczenie rozmiaru punktu z cali na milimetry

$$punkt = \frac{1}{72} cala = \frac{25.4}{72} mm = 0.3527(7) mm \tag{2.1}$$

Z tego powodu cały proces ekstrakcji danych z tabel znajdujących się w plikach *PDF*, można podzielić na mniejsze indywidualne zadania

1. Odczytywanie zawartości pliku *PDF*.
2. Rozpoznawanie linii separatorów i siatek.

3. Lokalizowanie obszarów z tabelami w dokumencie.
4. Określanie struktury wierszy i kolumn tabeli.
5. Określanie wierszy nagłówków tabeli.
6. Formatowanie i importowanie danych z tabeli do aplikacji w języku Python.
7. Eksportowanie tabeli do pliku o formacie *CSV*.

Dzięki przeprowadzeniu z powodzeniem wyżej wymienionych kroków, uzyskamy pożądany efekt, którym jest tabela w edytowalnym formacie zapisana w osobnym pliku.

Rozdział 3

Algorytmy

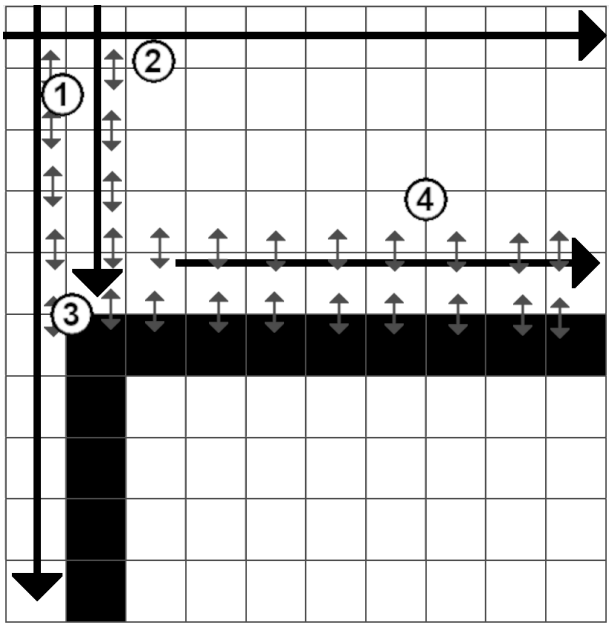
3.1 Odczytywanie zawartości pliku *PDF*

Ponieważ tekst w danej pracy jest eksportowany przez zewnętrzną bibliotekę, nie jest wymagane dalsze jej omówienie. Biblioteka pdfminer jest jednym z najczęściej wykorzystywanych narzędzi w celu ekstrakcji danych z plików *PDF*.

3.2 Rozpoznawanie linii separatorów i siatek

Bez informacji o liniach i obszarach prostokątnych na stronie *PDF*, poprawne zdefiniowanie komórek byłoby w większości przypadków zadaniem niemal niemożliwym. Dlatego kolejnym algorytmem jest wykrywanie jedynych dwóch rodzajów linii separujących, które mogą wystąpić w tym formacie. Są to proste linie pionowe i proste linie poziome, z powodu na rzadkie wykorzystanie innych linii jako separatory, linie krzywe i skośne nie są brane pod uwagę. Pierwszym krokiem jest analiza pikseli obrazu w skali szarości, następnie algorytm porównuje sąsiednie piksele od lewego górnego rogu strony, kierując się w dół, szukając różnic intensywności powyżej określonego progu. Po znalezieniu pary pikseli o wystarczającej różnicy, algorytm analizuje sąsiednie piksele po prawej stronie od znalezionej pary. Pozwala to na wyznaczenie lewego i prawego brzegu obiektu, a następnie zarejestrowanie go jako linię poziomą. Graficzne przedstawienie przebiegu algorytmu znajduje się na rysunku 3.1 poniżej.

W przypadku linii pionowych, algorytm wykonuje swoją pracę analogicznie, jednak początkowym jego zadaniem jest określenie szerokości oraz wysokości tabeli, a dopiero następnie analizuje on konkretne rozłożenie jej komórek. Algorytm posiada określony kontrast, który jest wymagany do wykrycia linii, wymaganą grubość oraz długość linii, dlatego nie dla każdego przypadku przy wykorzystaniu ustawień ogólnych zostanie uzyskany oczekiwany rezultat. Określenie takich limitów dla detekcji linii jest również niezbędne dla sytuacji kiedy czcionka ma małe odstępy i algorytm mógłby rozpoznać sąsiadujące



Rysunek 3.1: Graficzna reprezentacja algorytmu detekcji linii poziomych. Algorytm analizuje kolejne piksele od lewego górnego rogu w dół (1), porównując sąsiadujące pionowo piksele. Gdy osiągnie dolną krawędź strony, algorytm przechodzi do kolejnej kolumny pikseli(2). Kiedy znaleziona zostanie kontrastująca para pikseli(3), algorytm przechodzi wzdłuż krawędzi w prawo(4), zatrzymując się przy zakończeniu krawędzi(nie widnieje na grafice), następnie algorytm kontynuuje od pierwszego punktu w dół, poszukując kolejnej krawędzi pionowej.

litery jako linię poziomą(rys. 3.2), lub kiedy użyta zostanie duża czcionka wraz z literami posiadającymi długie linie, takimi jak znaki "q", "l", "p", itp.

Rysunek 3.2: W przypadku niektórych czcionek istnieje możliwość wykrycia błędnej linii poziomej lub pionowej, zwłaszcza przy plikach w niższej rozdzielczości

3.3 Lokalizowanie obszarów z tabelami w dokumencie

3.4 Określanie struktury wierszy i kolumn tabeli

3.5 Określanie wierszy nagłówków tabeli

3.6 Formatowanie i importowanie danych z tabeli do aplikacji w języku Python

3.7 Eksportowanie tabeli do pliku o formacie *CSV*

Rozdział 4

Badanie istniejących rozwiązań

tekst

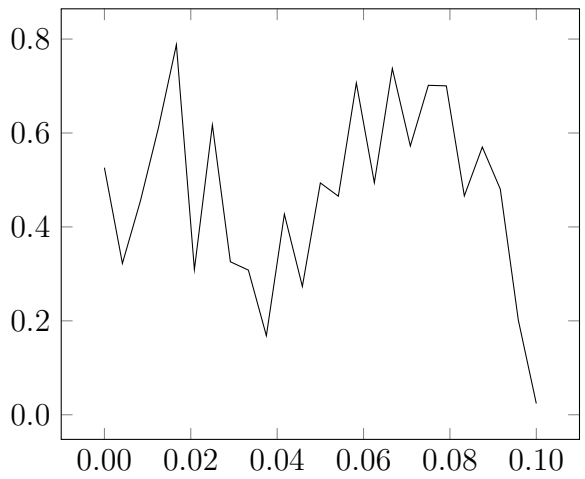
4.1 Przegląd istniejących modułów i bibliotek do importowania danych z PDF

4.2 Funkcje istniejących rozwiązań

4.3 Oceny użytkowników i opinie na temat istniejących rozwiązań

W całym dokumencie powinny znajdować się odniesienia do zawartych w nim ilustracji (rys. 4.1).

Tekst dokumentu powinien również zawierać odniesienia do tabel (tab. 4.1).



Rysunek 4.1: Wykres przebiegu funkcji.

Tabela 4.1: Opis tabeli nad nią.

	metoda						
			alg. 3			alg. 4, $\gamma = 2$	
ζ	alg. 1	alg. 2	$\alpha = 1.5$	$\alpha = 2$	$\alpha = 3$	$\beta = 0.1$	$\beta = -0.1$
0	8.3250	1.45305	7.5791	14.8517	20.0028	1.16396	1.1365
5	0.6111	2.27126	6.9952	13.8560	18.6064	1.18659	1.1630
10	11.6126	2.69218	6.2520	12.5202	16.8278	1.23180	1.2045
15	0.5665	2.95046	5.7753	11.4588	15.4837	1.25131	1.2614
20	15.8728	3.07225	5.3071	10.3935	13.8738	1.25307	1.2217
25	0.9791	3.19034	5.4575	9.9533	13.0721	1.27104	1.2640
30	2.0228	3.27474	5.7461	9.7164	12.2637	1.33404	1.3209
35	13.4210	3.36086	6.6735	10.0442	12.0270	1.35385	1.3059
40	13.2226	3.36420	7.7248	10.4495	12.0379	1.34919	1.2768
45	12.8445	3.47436	8.5539	10.8552	12.2773	1.42303	1.4362
50	12.9245	3.58228	9.2702	11.2183	12.3990	1.40922	1.3724

Rozdział 5

Ocena dostępnych opcji

Odwołania do literatury: książek [8], artykułów w czasopismach [7], materiałów konferencyjnych [6] i stron www [5].

Równania powinny być numerowane

$$punkt = \frac{1}{72} cala = \frac{25.4}{72} mm = 0.3527(7) mm \tag{5.1}$$

5.1 Porównanie istniejących rozwiązań na podstawie zdefiniowanych wymagań

Rozdział 6

Rozwój istniejących rozwiązań

6.1 Rozważania dotyczące budowy niestandardowego modułu

6.2 Zasoby i umiejętności wymagane do rozwoju i utrzymania modułu

Rozdział 7

Rozwój wybranego rozwiązania

7.1 Testowanie wybranego rozwiązania

7.2 Udoskonalenie wybranego rozwiązania na podstawie napotkanych problemów

Rozdział 8

Dokumentacja i udostępnianie wybranego rozwiązania

8.1 Dokumentacja rozwiązania

8.2 Udostępnianie rozwiązania jako projektu open-source

Rozdział 9

Podsumowanie

9.1 Podsumowanie badania

9.2 Implikacje dla przyszłej pracy

- syntetyczny opis wykonanych prac
- wnioski
- możliwość rozwoju, kontynuacji prac, potencjalne nowe kierunki
- Czy cel pracy zrealizowany?

Bibliografia

[1] ISO 32000-1:2008. *Document management — Portable document format — Part 1: PDF 1.7*. 2008. URL: https://www.iso.org/standard/51502.html (term. wiz. 12. 08. 2023).

[2] Adobe Systems Incorporated. *Adobe*. 2023. URL: https://www.adobe.com/legal/permissions/icons-web-logos.html (term. wiz. 18. 08. 2023).

[3] Adobe Systems Incorporated. *PDF Reference, Sixth Edition, 1.7*. 2006. URL: https://opensource.adobe.com/dc-acrobat-sdk-docs/pdfstandards/pdfreference1.7old.pdf (term. wiz. 12. 08. 2023).

[4] Adobe Systems Incorporated. *Public Patent License, ISO 32000-1*. 2008. URL: http://www.adobe.com/pdf/pdfs/ISO32000-1PublicPatentLicense.pdf (term. wiz. 13. 08. 2023).

[5] Imię Nazwisko i Imię Nazwisko. *Tytuł strony internetowej*. 2021. URL: http://gdzies/w/internecie/internet.html (term. wiz. 30. 09. 2021).

[6] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. „Tytuł artykułu konferencyjnego". W: *Nazwa konferecji*. 2006, s. 5346–5349.

[7] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. „Tytuł artykułu w czasopiśmie". W: *Tytuł czasopisma* 157.8 (2016), s. 1092–1113.

[8] Imię Nazwisko, Imię Nazwisko i Imię Nazwisko. *Tytuł książki*. Warszawa: Wydawnictwo, 2017. ISBN: 83-204-3229-9-434.

[9] Yasuke Shinyama. *pdfminer 20191125*. 2019. URL: https://pypi.org/project/pdfminer/ (term. wiz. 20. 08. 2023).

[10] X. Wang. „Praca doktorancka pt. Tabular Abstraction, Editing and Formatting." W: *University of Waterloo* (1996).

Dodatki

Spis skrótów i symboli

PDF kwas deoksyrybonukleinowy (ang. *deoxyribonucleic acid*)

CSV model – widok – kontroler (ang. *model–view–controller*)

Lista dodatkowych plików, uzupełniających tekst pracy (jeżeli dotyczy)

W systemie do pracy dołączono dodatkowe pliki zawierające:

- źródła programu,
- zbiory danych użyte w eksperymentach,
- film pokazujący działanie opracowanego oprogramowania lub zaprojektowanego i wykonanego urządzenia,
- itp.

Spis rysunków

Spis tabel