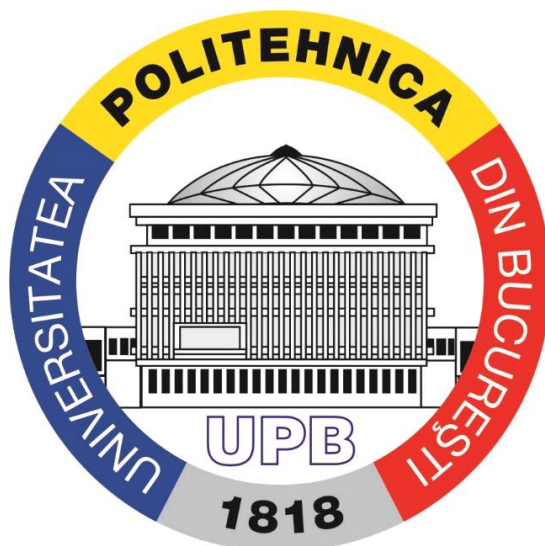


UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE

ABD 2017 - 2019



PROIECT DE DIZERTAȚIE
Detectarea similaritatilor intre documente

Coordonator:

Prof.dr.ing.Florin Radulescu

Student:

Rotarescu Ciprian Marian

CUPRINS

1. INTRODUCERE.....	
2. CONTINUTUL DOCUMENTATIEI.....	
3. CARACTERISTICILE TEHNOLOGIILOR PROPUSE PENTRU DEZVOLTARE.....	
3.1 JAVASCRIPT.....	
3.2 ANGULAR.JS.....	
3.3 NODE.JS.....	
3.4 HTML.....	
3.5 CSS	
4. IMPLEMENTARI & CONCLUZII.....	
4.1 ALGORITMUL CREATE DE JOHN RESIG.....	
4.2 DRAFTABLE COMPARE API.....	
4.3 LEVENSHTAIN ALGORITHM.....	
4.4 TRIGRAM COMPARISON.....	
4.5 COSINE SILIMARITY.....	
4.6 JARO-WINKLER ALGORITHM	
5. ALEGEREA METODEI DE IMPLEMENTARE.....	
6. CONFIGURARE BACKEND.....	
7. APLICATIE UI/UX.....	
8. TESTE.....	
9. CONCLUZII.....	
10. BIBLIOGRAFIE.....	

LISTA FIGURILOR, TABELELOR ȘI A PLANȘELOR

Figura 1 - NodeJs create server.....	
Figura 2- HTML sample.....	
Figura 3 - Algoritmul John Resig (1).....	
Figura 4 - Algoritmul John Resig (2).....	
Figura 5 - Algoritmul John Resig - read/upload.....	
Figura 6 - Algoritmul John Resig - interfata.....	
Figura 7 - Algoritmul John Resig - lista pdf.....	
Figura 8 - Algoritmul John Resig - rezultat 1.....	
Figura 9 - Algoritmul John Resig - rezultat 2.....	
Figura 10 Draftable - Modificare titlu.....	
Figura 11 Draftable - Modificare titlu si header.....	
Figura 12 Algoritmul Levenshtein - formula.....	
Figura 13 Cosine Similarity - formula.....	
Figura 14 Cosine Similarity - exemplu.....	
Figura 15 Cosine Similarity - cuvinte gasite.....	
Figura 16 Cosine Similarity - numararea cuvintelor.....	
Figura 17 Cosine Similarity - vectori.....	
Figura 18 Algoritmul Jaro Winker- formula.....	
Figura 19 Algoritmul Jaro Winker- exemplu.....	
Figura 20 Algoritmul Jaro Winker- distanta Jaro.....	
Figura 21 Algoritmul Jaro Winker- formula distantei Jaro-Winkler.....	
Figura 22 Algoritmul Jaro Winker- distanta Jaro-Winkler.....	
Figura 23 Draftable - prezentare.....	
Figura 24 Draftable - credentiale api.....	
Figura 25 Draftable - cont.....	
Figura 26 Backend files.....	
Figura 27 backend implementation (1)	
Figura 28 backend implementation (2)	
Figura 28 backend implementation (3)	
Figura 29 backend implementation (4)	
Figura 30 backend implementation (5)	
Figura 31 backend implementation (6)	
Figura 32 backend implementation (7)	
Figura 33 backend implementation (8)	
Figura 34 backend implementation (9)	

Figura 35 Mongodb DB	
Figura 36 Mongodb collections.....	
Figura 37 Mongodb UI - Login.....	
Figura 38 Mongodb UI - Dashboard.....	
Figura 39 Mongodb UI - Dashboard pdf files.....	
Figura 40 Mongodb UI - Library.....	
Figura 41 Mongodb UI - Create.....	
Figura 42 Teste - "Measuring the health of open source software ecosystems Beyond".....	
Figura 43 Teste - "Measuring the health of open source software ecosystems Beyond".....	
Figura 44 Teste - " Prezentare ISBD SQL vs NoSQL ".....	
Figura 45 Teste - " Prezentare ISBD SQL vs NoSQL ".....	
Figura 46 Teste - " Prezentare ISBD SQL vs NoSQL ".....	
Figura 47 Teste - "Diacritice".....	

Detectarea plagiatului pentru texte în limba română

Tema isi propune sa se finalizeze cu o dizertatie in cadrul careia sa existe si o implementare a unui program de testare a documentelor pentru a descoperi eventuale diferente intre acestea.

1. Introducere

Tema de cercetare aleasă implică implementarea unei aplicații de testare a documentelor, pentru a ajuta un user in a descoperi diferentele intre diverse doua documente. Acest lucru poate ajuta o persoana sa vada ce modificari au intervenit intr-un document intr-o perioada mare de timp (lucrare stiintifica, carti, etc).

Pentru implementarea acestei teme propunem implementarea unei aplicații web ce permite utilizatorului să compare fișiere de tip pdf prin care să se deducă modificările / similaritățile dintre acestea. Este aleasa varianta web a aplicatiei datorita accesului usor al acestuia de catre orice utilizator si datorita faptului ca nu necesita nici o instalare pe un anumit device.

Pentru aceasta aplicatie am ales sa folosesc AngularJS 1.6 ca si tehnologie principala si NodeJS ca si backend. Aceste tehnologii mentionate pot fi structurate intr-o arhitectura de model MVC (Model – View – Controller), ceea ce permite organizarea codului, separarea logicii, o optimizarea imbunatatita si de asemenea permite modificarea codului mult mai usor (atat pentru dezvoltatorul initial cat si pentru alti participanti).

2. Continutul documentatiei

In cadrul acestui document se va evidentiata date tehnice legate de cod ul folosit, ce solutii au fost gasite pentru implementarea temei alese, rezultate si teste, solutia acceptata pentru implementare, implementarea solutiei acceptate in aplicatia de tip web si rezultatele obtinute.

3. Caracteristicile tehnologiilor propuse pentru dezvoltare

In acest capitol sunt prezentate caracteristicile tehnologiilor propuse pentru dezvoltarea temei propuse. Acestea sunt principalele tehnologii esențiale pentru implemetarea temei de detectare a textelor plagiate.

3.1 Javascript

JavaScript este folosit cel mai frecvent limbaj de programare folosit pe partea de frontend a unei aplicatii web. Cu ajutorul acestuia se pot implementa diverse interactiuni intre browser si user (butoane, evenimente la click, animatii, etc), conectarea unei aplicatii cu o baza de date, transmiterea de date catre o baza de date, extragerea de informatatii dintr-o baza de date, calcule complexe si procesari avansate ce pot crea un user experience foarte bun.

Datorita simplitatii acestui limbaj de programare s-au produs un numar mare de librarii utile, foarte multe fiind open source iar comunitatea din spatele acestora fiind una semnificativa ajuta dezvoltarea aplicatiilor web.

3.2 AngularJS

Datorita popularitatii pe care o are Javascript-ul si a dezvoltarii acestuia de catre intregi comunitati s-au dezvoltat diverse framework-uri care are la baza acestuia Javascript.

AngularJS este un astfel de framework ce poate fi folosit pentru aplicațiile web dinamice. Vă permite să utilizați codul HTML pentru a insera logica de javascript în acesta folosind anumite simboluri ({{ }}) sau comenzi speciale pentru a performa loops sau alte taskuri generale într-o aplicație (ng-repeat, ng-show, ng-hide, etc).

AngularJS “data binding” și “dependency injection” ajută dezvoltatorul să economisească timp în a scrie cod pentru taskuri generale cum ar fi actualizarea datelor atât în view cât și în controller.

Așa cum am menționat anterior AngularJS simplifică dezvoltarea aplicațiilor prin stabilirea unor seturi de reguli și de comenzi pentru a face viața unui dezvoltator mai ușoară dar trebuie știut de la bun început că acest framework este doar o soluție pentru un anumit tip de problemă iar de asemenea se poate stabili că nu orice tip de aplicație este potrivită pentru AngularJS. AngularJS a fost construit având ca scop construirea de aplicații web. Din fericire, o mare parte din aplicațiile utilizate din viața de zi cu zi sunt aplicații web.

Câteva din elementele cheie ale lui AngularJS sunt:

Data bindings, acestea sunt simbolizate prin {{}} iar acestea ajută în transmiterea de variabile/functii din controller în HTML. De exemplu dacă dorești să populezi textul unui div cu datele dintr-o variabilă din angular transmiterea acestora se poate face prin :

```
$scope.name = "John Doe";
```

```
<div>{{ name }}</div>;
```

Structuri de control DOM cum ar fi repetarea de date dintr-un array, afișarea, ascunderea segmentelor dintr-un DOM:

```
$scope.hideTemplate = false;
```

```
<div ng-if="hideTemplate"></div>
```

În cazul de mai sus comanda ng-if poate afișa sau ascunde un conținut dacă conținutul din "" este translatat într-un boolean de true.

În cazul în care un ng-if este false conținutul cât și elementul HTML este scos complet din DOM și acest lucru poate fi arătat cu ajutorul unui inspect dat din browser.

```
<div ng-show="hideTemplate"></div>
```

```
<div ng-hide="hideTemplate"></div>
```

Comenzile de mai sus sunt folosite pentru a afișa sau ascunde anumite porțiuni din DOM (document object model).

```
$scope.listArray = [1,2,3,4,5];
```

```
<ul>
```

```
<li ng-repeat="list in listArray"> {{ list }} </li>
```

```
</ul>
```

Ng-repeat este folosit pentru a itera prin liste și de asemenea pentru afișarea datelor în DOM. Pe lângă afișarea acestora se pot adăuga comenzi adiționale pentru a extra indexul acestora sau diverse informații în funcție de conținutul listei (list of objects).

AngularJS permite dezvoltatorului să își compună singuri propriile instrucțiuni de DOM cu ajutorul directivelor și a componentelor.

3.3 NodeJS

Node.js este o platforma construita cu ajutorul limbajului de programare C++ si cu ajutorul acestuia se pot accesa fisiere de pe device-ul instalat (laptop/calculator), se pot crea fisiere/edita fisiere, se poate crea un server de backend pentru diverse aplicatii si multe altele.

Node este proiectat pentru a construi aplicații scalabile si se pot implementa aplicatii de tip real time chat cum ar fi "whatsapp". În exemplul de mai jos figureaza crearea unui server de transmite "hello world". Prin ajutorul acestui server se pot face mai multe conexiuni ce pot fi gestionate simultan.

```
const http = require('http');

const hostname = '127.0.0.1';
const port = 3000;

const server = http.createServer((req, res) => {
  res.statusCode = 200;
  res.setHeader('Content-Type', 'text/plain');
  res.end('Hello World\n');
});

server.listen(port, hostname, () => {
  console.log(`Server running at http://${hostname}:${port}/`);
});
```

Figura 1 - NodeJS create server

Merita mentionat faptul ca te poti conecta la diverse baze de date (MySQL, SQL, NoSQL) cu ajutorul pachetelor NPM si de asemenea sa transmiți date catre orice aplicatie se dorește. Aceste pachete NPM sunt dezvoltate de catre dezvoltatori sau echipe de dezvoltatori ce rezolva diverse probleme din viata de zi cu zi a unui developer (editare de fisiere pdf, algoritmi complexi, etc).

De asemenea trebuie mentionat ca este foarte usor sa instalezi NodeJS si exista mult support in cazul in care exista probleme de instalare sau altele.

3.4 HTML

Acronimul HTML reprezintă Hypertext Markup Language si este folosit pentru a scrie conținut pe web, orice aplicatie web are cel puțin o pagina de tip HTML. HTML a fost creat în 1991 de Tim Berners-Lee , creatorul oficial și fondator al ceea ce acum știm ca World Wide Web.

Limba HTML consta dintr-o serie de coduri scurte introduce într-un fisier de tip html pentru ca un browser sa il poata citi si interpreta codul inserat in acesta.

Acest browser citește fișierul și traduce textul într-o formă vizibilă. Pentru editarea codului pentru această aplicație am folosit Sublime Text 3 iar un cod HTML arată ca și în figura următoare:

```
<!DOCTYPE html>
<html>
<head>
  <title></title>
</head>
<body>

</body>
</html>
```

Figura 2 - HTML sample

HTML poate încorpora programe scrise într-o limbă de scripting, cum ar fi JavaScript, care afectează comportamentul și conținutul paginilor web. Includerea CSS definește aspectul și aspectul conținutului.

HTML-ul poate fi afectat de diverse plugin-uri sau framework-uri ce poate rezulta în manipularea acestuia și afișarea unui conținut nou/editat.

3.5 CSS

CSS-ul poate fi folosit pentru a stiliza conținutul web. CSS-ul poate îmbunătăți accesibilitatea conținutului, poate oferi mai multă flexibilitate și control în specificarea caracteristicilor de prezentare, permite mai multor pagini HTML să împărtășească formatarea specificând CSS relevant într-un fișier .css separat și reducând complexitatea și repetarea conținutului structural.

4. Implementari & concluzii

Pentru implementarea acestei teme s-a făcut o cercetare mai amănunțită asupra posibilităților de implementare a temei și în urma acestor există o serie de pro/cons. În următoarele pagini sunt redate implementările respective și concluziile de pe urma acestora.

4.1 Algoritm creat de John Resig

Pentru detectarea textelor dintr-un document am ales să folosesc un algoritm pentru textelor creat de John Resig, care folosește o licență MIT. Acest algoritm permite diferențierea textelor din două locații și produce un text final cu modificările evidențiate sub diverse culori / subliniate.

În următoarea figură se poate observa rezultatul diferențelor detectate dintre două texte aproape similare.

First Text original text

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Second Text Textele de culoare diferita sunt introduse random

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, "**inserted text**" when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was "**inserted text**" popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing "**inserted text**" software like Aldus PageMaker including versions of Lorem Ipsum.

Result

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, "inserted text" when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was "inserted text" popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing "inserted text" software like Aldus PageMaker including versions of Lorem Ipsum.

Figura 3 - Algoritmul John Resig (1)

In figura de mai sus sunt afisate trei texte "First Text", "Second Text" si "Result". "First Text", "Second Text" sunt interpretate ca si doua documente diferite iar al treilea text reprezinta rezultatul dintre cele anterioare.

In acest test am dorit sa compar continutul din "Second text" cu "First Text". Al doilea document contine cuvinte evidentiata in culoarea mov si au textul "inserted text" ce au fost introduse special pentru a genera o comparare cu "First text".

Rezultatul produs a fost de a evidentia cuvintele introduse "inserted text" din al doilea document cu o culoare albastra si de asemenea subliniate.

De asemenea s-a facut si un test in cazul incare situatia este inversata ("First Text" contine cuvinte in plus) iar rezultatul se poate evidentia imaginea de mai jos.

First Text original text

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, "**inserted text**" when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was "**inserted text**" popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing "**inserted text**" software like Aldus PageMaker including versions of Lorem Ipsum.

Second Text

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing software like Aldus PageMaker including versions of Lorem Ipsum.

Result

Lorem Ipsum is simply dummy text of the printing and typesetting industry. Lorem Ipsum has been the industry's standard dummy text ever since the 1500s, "inserted text" when an unknown printer took a galley of type and scrambled it to make a type specimen book. It has survived not only five centuries, but also the leap into electronic typesetting, remaining essentially unchanged. It was "inserted text" popularised in the 1960s with the release of Letraset sheets containing Lorem Ipsum passages, and more recently with desktop publishing "inserted text" software like Aldus PageMaker including versions of Lorem Ipsum.

Figura 4 - Algoritmul John Resig (2)

In acest caz rezultatul compara al doilea document cu primul iar cuvintele ce nu se regasesc in documentul al doilea sunt taiate si evidentiata cu culoarea rosie.

Pentru continut de dimensiuni reduse acest algoritm face exact ce a fost propus dar pentru continut de dimensiuni medii/mari rezultatul nu este chiar acela dorit. In urma unor teste mai

Pentru vizualizarea documentelor din baza de date am dorit sa le evidentializez sub forma unei liste in care fiecare element reprezinta un document.

Documents stored in DB

Test 1.pdf

Test 3.pdf

Prezentare ISBD SQL vs NoSQL.pdf

Figura 7 - Algoritmul John Resig - lista pdf

De asemenea trebuie evidentiat ca s-a introdus un eveniment la click pe aceste elemente. Atunci cand se face un click pe acest element el, frontend-ul compara textul din sectiunea "Uploaded PDF sample" cu continutul ascuns din elementul din lista selectat.

Text detector v0.1

Read PDF

Browse...

Test 1.pdf

Read File

Upload PDF

Browse...

Upload File

Documents stored in DB

Test 1.pdf

Test 3.pdf

Prezentare ISBD SQL vs NoSQL.pdf

Text extract

Uploaded PDF sample

Title test1 Lorem ipsum dolor sit amet, at sententiae intellegbat ius. In vim noster detrahit probatus, eos at falli auliam. Mei inermis dissentiunt et, eam eu vocent labitur. Cum ut ceteros facilis voluptatibus, eos in causae accusam efficiantur. Te eum congue aliquip omitantur, at vix dico scripsit. Te mei atqui alienum constituam, ex pri aperian intellegat, pri no consul constituam. Eos ut eripuit intellegbat, ei sint aliquando usu. Quo rebum porro meliore at, vix voluptua assentior ut, in sanctus quaestio consetur usu. Est aequae lobortis deseruisse ei, per ad stet quas. An sed omnis tincidunt, vide aperiri id sed. Pri in velit eusmodi delicata, ex pro tantas feugiat. Est et solum pertinacia dissentiit, brute simul his ne. An aperiam con saluta honestatis cum, an sale officiis scribentur cum, mea et molestie suavitae. Sed debet invidunt assueverit an, ea sed ipsum fastidii. Eum vocent habemus praesent ne, pri inani epicurei ex, nam ex error aequae. In brute necessitatibus est, qui debet phaedrum id. Oratio philosophia ea pri, sea no tantas necessitatibus. Summo essent equidem per ad, in vel brute volumus. Quo velit nominavi at, an has quod efficiendi reformidans. Ut justo temporibus eum, usu erat rebum no. Ad choro utamur admodum mea, perf ecto quaerendum eos ut. Mei et eros omnes invidunt. Utinam vulpate pri te, et odio scriptorem cum, mutat tempor debitis id has. Nibh animal liberavisse ne per, stet labores nam et, case sanctus eu duo. Cum tempor disputando ea, maiorum imperdiet eam et, mei viris ignota salutandi cu. Falli invenire est ne. El vis habeo oportere vulpate, nam nobis viderer facilis no. Lorem ipsum dolor sit amet, at sententiae intellegbat ius. In vim noster detrahit probatus, eos at falli auliam. Mei inermis dissentiunt t et, eam eu vocent labitur. Cum ut ceteros facilis voluptatibus, eos in causae accusam efficiantur. Te eum congue aliquip omitantur, at vix dico scripsit. Te mei atqui alienum constituam, ex pri aperian intellegat, pri no consul constituam. Eos ut eripuit intellegbat, ei sint aliquando usu. Quo rebum porro meliore at, vix voluptua assentior ut, in sanctus quaestio consetur usu. Est aequae lobortis deseruisse ei, per ad stet quas. An sed omnis tincidunt, vide aperiri id sed. Pri in velit eusmodi de licata, ex pro tantas feugiat. Est et solum pertinacia dissentiit, brute simul his ne. An aperiam consulu honestatis cum, an sale officiis scribentur c um, mea et molestie suavitae. Sed debet invidunt assueverit an, ea sed ipsum fastidii. Eum vocent habemus praesent ne, pri inani epicurei ex, nam ex error aequae. In brute necessitatibus est, qui debet phaedrum id. Oratio philosophia ea pri, sea no tantas necessitatibus. Summo essent equidem per ad, in vel brute volumus. Quo velit nominavi at, an has quod efficiendi reformidans. Ut justo temporibus eum, usu erat rebum no. Ad choro utamur admodum mea, perfect o quaerendum eos ut. Mei et eros omnes invidunt. Utinam vulpate pri te, et odio scriptorem cum, mutat tempor debitis id has. Nibh animal liberavisse ne per, stet labores nam et, case sanctus eu duo. Cum tempor disputando ea, maiorum imperdiet eam et, mei viris ignota salutandi cu. Falli invenire est n e. El vis habeo oportere vulpate, nam nobis viderer facilis no. Lorem ipsum dolor sit amet, at sententiae intellegbat ius. In vim noster detrahit probatus, eos at falli auliam. Mei inermis dissentiunt et, eam eu vocent labitur. Cum ut ceteros facilis voluptatibus, eos in causae accusam efficiantur. Te eum congue aliquip omitantur, at vix dico scripsit. Te mei atqui alienum constituam, ex pri aperian intellegat, pri no consul constituam. Eos ut eripuit intellegbat, ei sint aliquando usu. Quo rebum porro meliore at, vix voluptua assentior ut, in sanctus quaestio consetur usu. Est aequae lobortis deseruisse ei, per ad stet quas. An sed omnis tincidunt, vide aperiri id sed. Pri in velit eusmodi delicata, ex pro tantas feugiat. Est et solum pertinacia dissentiit, brute simul his ne. An aperiam consulu honestatis cum, an sale officiis scribentur c um, mea et molestie suavitae. Sed debet invidunt assueverit an, ea sed ipsum fastidii. Eum vocent habemus praesent ne, pri inani epicurei ex, nam ex error aequae. In brute necessitatibus est, qui debet phaedrum id. Oratio philosophia ea pri, sea no tantas necessitatibus. Summo essent equidem per ad, in vel brute volumus. Quo velit nominavi at, an has quod efficiendi reformidans. Ut justo temporibus eum, usu erat rebum no. Ad choro utamur admodum mea, perfect o quaerendum eos ut. Mei et eros omnes invidunt. Utinam vulpate pri te, et odio scriptorem cum, mutat tempor debitis id has. Nibh animal liberavisse ne per, stet labores nam et, case sanctus eu duo. Cum tempor disputando ea, maiorum imperdiet eam et, mei viris ignota salutandi cu. Falli invenire est ne. El vis habeo oportere vulpate, nam nobis viderer e facilis no.

Result

Identic text is 100 %

Title test1 Lorem ipsum dolor sit amet, at sententiae intellegbat ius. In vim noster detrahit probatus, eos at falli auliam. Mei inermis dissentiunt et, eam eu vocent labitur. Cum ut ceteros facilis voluptatibus, eos in causae accusam efficiantur. Te eum congue aliquip omitantur, at vix dico scripsit. Te mei atqui alienum constituam, ex pri aperian intellegat, pri no consul constituam. Eos ut eripuit intellegbat, ei sint aliquando usu. Quo rebum porro meliore at, vix voluptua assentior ut, in sanctus quaestio consetur usu. Est aequae lobortis deseruisse ei, per ad stet quas. An sed omnis tincidunt, vide aperiri id sed. Pri in velit eusmodi de licata, ex pro tantas feugiat. Est et solum pertinacia dissentiit, brute simul his ne. An aperiam consulu honestatis cum, an sale officiis scribentur c um, mea et molestie suavitae. Sed debet invidunt assueverit an, ea sed ipsum fastidii. Eum vocent habemus praesent ne, pri inani epicurei ex, nam ex error aequae. In brute necessitatibus est, qui debet phaedrum id. Oratio philosophia ea pri, sea no tantas necessitatibus. Summo essent equidem per ad, in vel brute volumus. Quo velit nominavi at, an has quod efficiendi reformidans. Ut justo temporibus eum, usu erat rebum no. Ad choro utamur admodum mea, perfect o quaerendum eos ut. Mei et eros omnes invidunt. Utinam vulpate pri te, et odio scriptorem cum, mutat tempor debitis id has. Nibh animal liberavisse ne per, stet labores nam et, case sanctus eu duo. Cum tempor disputando ea, maiorum imperdiet eam et, mei viris ignota salutandi cu. Falli invenire est n e. El vis habeo oportere vulpate, nam nobis viderer facilis no. Lorem ipsum dolor sit amet, at sententiae intellegbat ius. In vim noster detrahit probatus, eos at falli auliam. Mei inermis dissentiunt et, eam eu vocent labitur. Cum ut ceteros facilis voluptatibus, eos in causae accusam efficiantur. Te eum congue aliquip omitantur, at vix dico scripsit. Te mei atqui alienum constituam, ex pri aperian intellegat, pri no consul constituam. Eos ut eripuit intellegbat, ei sint aliquando usu. Quo rebum porro meliore at, vix voluptua assentior ut, in sanctus quaestio consetur usu. Est aequae lobortis deseruisse ei, per ad stet quas. An sed omnis tincidunt, vide aperiri id sed. Pri in velit eusmodi delicata, ex pro tantas feugiat. Est et solum pertinacia dissentiit, brute simul his ne. An aperiam consulu honestatis cum, an sale officiis scribentur c um, mea et molestie suavitae. Sed debet invidunt assueverit an, ea sed ipsum fastidii. Eum vocent habemus praesent ne, pri inani epicurei ex, nam ex error aequae. In brute necessitatibus est, qui debet phaedrum id. Oratio philosophia ea pri, sea no tantas necessitatibus. Summo essent equidem per ad, in vel brute volumus. Quo velit nominavi at, an has quod efficiendi reformidans. Ut justo temporibus eum, usu erat rebum no. Ad choro utamur admodum mea, perfect o quaerendum eos ut. Mei et eros omnes invidunt. Utinam vulpate pri te, et odio scriptorem cum, mutat tempor debitis id has. Nibh animal liberavisse ne per, stet labores nam et, case sanctus eu duo. Cum tempor disputando ea, maiorum imperdiet eam et, mei viris ignota salutandi cu. Falli invenire est ne. El vis habeo oportere vulpate, nam nobis viderer e facilis no.

Figura 8 - Algoritmul John Resig - rezultat 1

In experimentul urmatoar folosind aceeasi pasi dar un alt obiect din baza de date avem urmatoarul rezultat.

In urma analizei se poate vedea ca doar 61% din text este identic cu fisierul propus pentru testare.

Figura 9 - Algoritmul John Resig - rezultat 2

Dupa o analiza mai detaliata al continutului am constatat ca exista o marja de eroare intre procentajul dat de algoritm 61% si realitate.

Acest lucru a fost posibil datorita analizei zonei evidentiata in partea dreapta. Acest lucru indica faptul ca acest algoritm functioneaza mai bine pe documentele cu text scurte decat cele indicate pentru tema aleasa.

4.2 Draftable Compare API

Pentru detectarea textelor dintr-un document am ales sa folosesc un API popular numit "Draftable Compare API" (<https://github.com/draftable/compare-api-node-client>).

Acest API vine si cu un demo pentru a vizualiza rezultatul dat de catre acesta librarie.

Rezultatul dat de catre dezvoltatori sta la baza compararii a doua fisiere(unul pdf si altul rtf). numit "left.rtf" si "right.pdf". Intre aceste doua fisiere sunt evidentiata diferentele de cuvintelor dintre cele documente prin nuanzare de background al cuvintelor lipsa sau adaugate (rosu/verde).

Url-urile pentru aceste doua fisiere sunt urmatoarele: 1.<https://api.draftable.com/static/test-documents/code-of-conduct/left.rtf>;

2.<https://api.draftable.com/static/test-documents/code-of-conduct/right.pdf>

In Imaginile de mai jos sunt evidentiata modificarile pe care acestea le au:

1.Modificari la baza titlului ("STANDARDS/STATEMENT, ETHICS/STANDARDS");

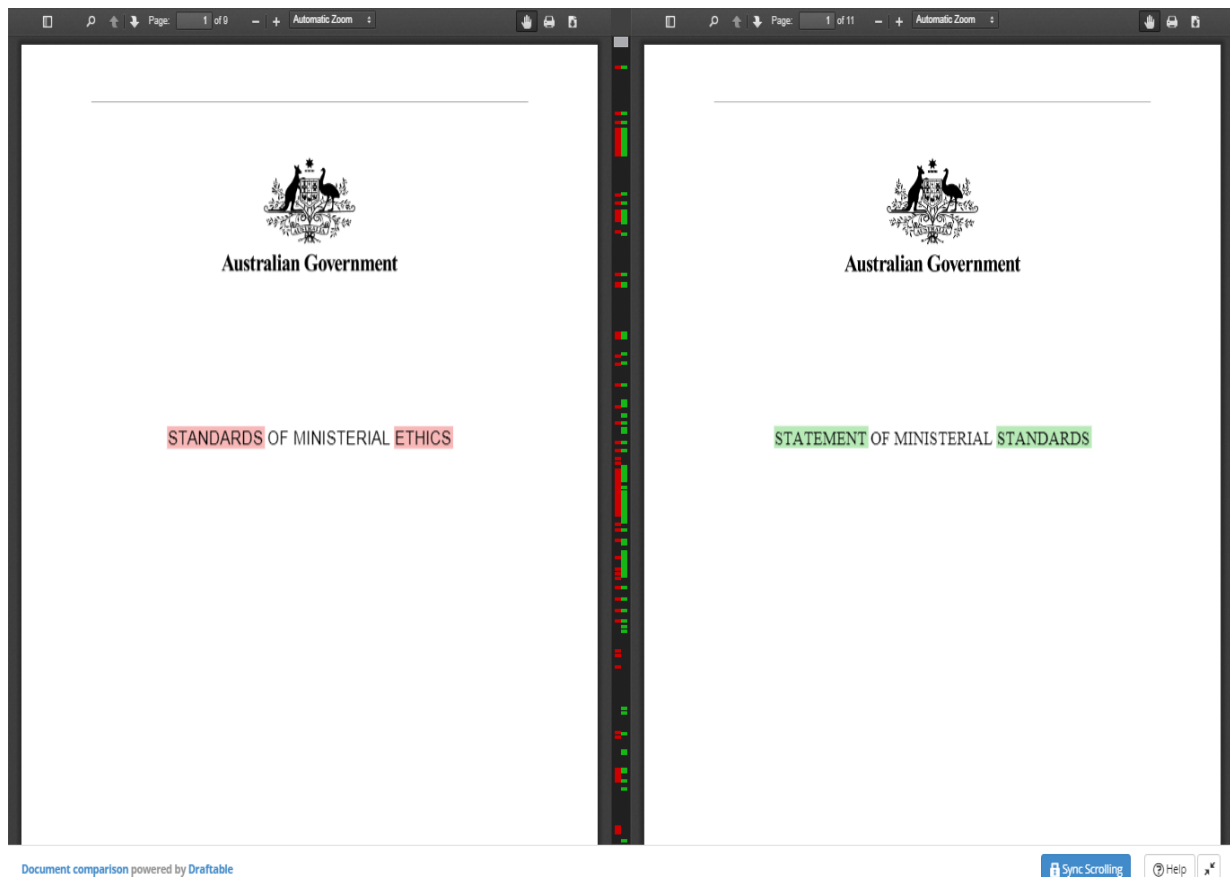


Figura 10 Draftable - Modificare titlu

2. Modificari in baza cuprinsului;

Contents	Page No
FOREWORD.....	1
STANDARDS OF MINISTERIAL ETHICS.....	2
1. Principles.....	2
2. Integrity.....	3
Directorships etc.....	3
Shareholdings.....	4
Family members.....	4
Other forms of employment.....	4
Gifts.....	5
Employment of family members.....	5
Post-ministerial employment.....	5
3. Fairness.....	5
4. Accountability.....	6
5. Responsibility.....	6
6. The Public Interest.....	6
7. Implementation.....	6
8. Contact with Lobbyists.....	7

Document comparison powered by Draftable

Sync scrolling Share Help

Figura 10 Draftable - Modificare cuprins

3.Modificari in baza continutului.

Contents	Page No
FOREWORD.....	1
STATEMENT OF MINISTERIAL STANDARDS.....	2
1. Principles.....	2
2. Integrity.....	3
Directorships etc.....	3
Shareholdings.....	4
Family members.....	5
Other forms of employment.....	5
Gifts.....	6
Employment of family members.....	6
Post-ministerial employment.....	6
3. Fairness.....	7
4. Accountability.....	7
5. Responsibility.....	7
6. The Public Interest.....	8
7. Implementation.....	8
8. Contact with Lobbyists.....	8

Document comparison powered by Draftable

Sync scrolling Share Help

Figura 10 Draftable - Modificare continut

4. Modificar in baza titlurilor si header-ului

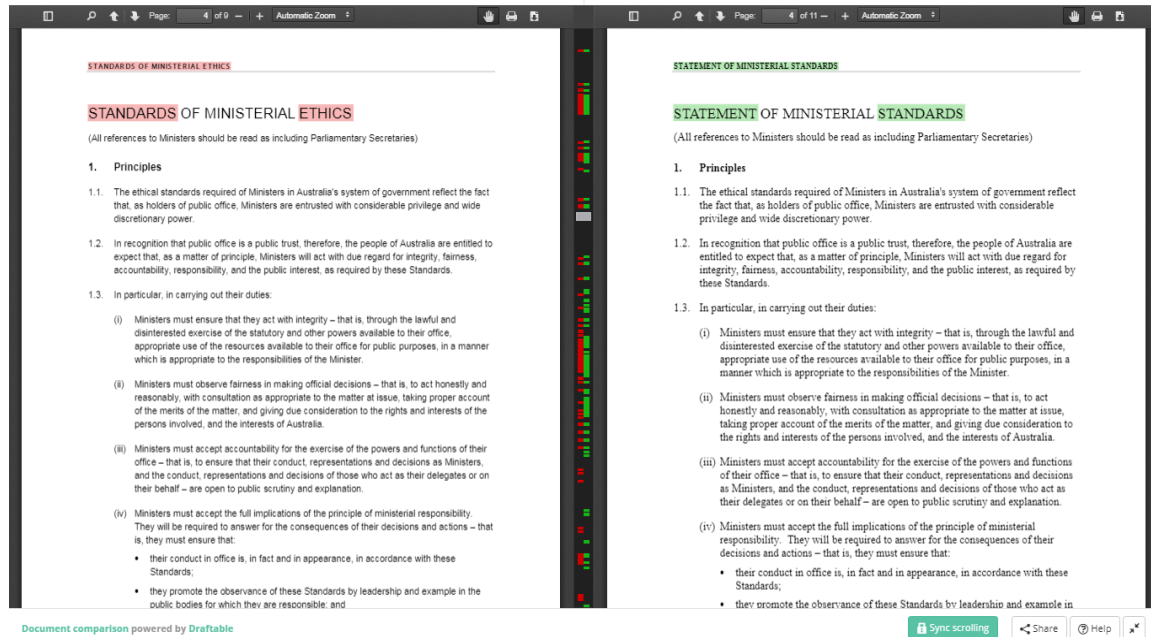


Figura 11 Draftable - Modificare titlu si header

Acest exemplu dat de catre dezvoltatori constituie exact ce ce trebuie pentru tema aleasa. De asemenea ne si arata vizual toate diferentele intre cele doua documente prin compararea pdf-urilor in paralel si sunt evidentiata toate modificarile in ambele parti prin backgrounduri diferite.

Fiind spuse partile pozitive despre acesta librerie exista de asemenea si cons. Una dintre ele fiind faptul ca exista o limita de utilizari free/luna iar depasirea acestei limite necesita un abonament lunar platit pentru dezvoltatori.

4.3 Algoritmul Levenshtein

Distanța Levenshtein este folosită în acest algoritm pentru a măsura diferența dintre două secvențe de caractere. Distanța dintre două cuvinte/texte reprezintă numărul minim de editări (inserări, ștergeri sau schimbări de caracter) necesare pentru a transforma un cuvânt în altul. Denumirea algoritmului vine de la matematicianul Vladimir Levenshtein.

Din punct de vedere matematic distanța între două șiruri de caractere folosind acest algoritm (a,b) este:

$$\text{lev}_{a,b}(|a|, |b|) \text{ Unde } \begin{cases} \max(i, j) & \text{if } \min(i, j) = 0, \\ \min \begin{cases} \text{lev}_{a,b}(i-1, j) + 1 \\ \text{lev}_{a,b}(i, j-1) + 1 \\ \text{lev}_{a,b}(i-1, j-1) + 1_{(a_i \neq b_j)} \end{cases} & \text{otherwise.} \end{cases}$$

Figura 12 Algoritmul Levenshtein - formula

De exemplu, distanța dintre "pisoii" și "așezat" (în engleză) distanța Levenshtein este de 3, deoarece următoarele trei modificări se schimbă una în cealaltă și nu există nicio modalitate de a face acest lucru cu mai puțin de trei modificări:

k itten → s itten (substituirea lui "s" pentru "k")
sitt e n → sitt i n (înlocuirea lui "i" cu "e")
sittin → sittin g (introducerea "g" la sfârșit).

Acest algoritm este ideal pentru siruri de texte de mici dimensiuni si poate fi folosit pentru aplicatii de verificare a ortografiei, sisteme de corectie sau alte aplicatii ce au ca scop traducerea textelor.

4.4 Compararea Trigram

Aceasta metoda de comparare poate fi folosita pentru a compara doua stringuri si reprezinta secvente de n-gram. Pentru a demonstra acesta metoda avem cuvantul 'martha' si al doilea cuvant 'marhta' (acest cuvant este similar cu cel anterior doar ca sunt inversate caracterele 'h' si 't') iar rezultatul este urmatorul :

Secventele n-gram pentru cuvantul "martha" sunt : { mar art rth tha }

Secventele n-gram pentru cuvantul "marhta" sunt: { mar arh rht hta }

Pentru a detecta similaritatea intre cele doua texte impartim numarul de n-grams indentice intre cele doua cuvinte iar in cazul de sus este 1 { mar } cu numarul de n-grams unice dintre cele doua cuvinte 7 { mar art rth tha arh rht hta }.

Astfel pentru cuvintele "martha" si "marhta" similaritatea intre cele doua stringuri este de 14% (1/7).

4.5 Cosine Similarity

Cosine Similarity intre doua stringuri este reprezentata ca punctul de reprezentare vectorial al acestora.

```
similitudine  
  
= cos (a, b)  
  
= produsul punctual (a, b) / ( norma (a) * norma (b))  
  
= ab / || a || * || b ||
```

Figura 13 Cosine Similarity - formula

Un exemplu pentru acest algoritm poate fi redat in forma urmatoare:

```
Julie loves me more than Linda loves me  
  
Jane likes me more than Julie loves me
```

Figura 14 Cosine Similarity - exemplu

Scopul final este de a observa cat de asemanatoare sunt cele doua texte (ordinea acestora nu conteaza) iar din aceste texte se formeaza o lista cu cuvinte.


```
me Julie loves Linda than more likes Jane
```

Figura 15 Cosine Similarity - cuvinte gasite

Dupa crearea acestui array se numara aparitia fiecarui cuvant in ambele texte.

me	2	2
Jane	0	1
Julie	1	1
Linda	1	0
likes	0	1
loves	2	1
more	1	1
than	1	1

Figura 16 Cosine Similarity - numararea cuvintelor

Cu toate acestea, nu ne interesează cuvintele. Suntem interesați doar de acei doi vectori verticali de numărare. De exemplu, există două exemple de "eu" în fiecare text. Vom decide cât de apropiate sunt aceste două texte între ele prin calcularea unei funcții a celor doi vectori, și anume cosinusul unghiului dintre ele.

Cei doi vectori formați în urma listei de mai sus sunt:

```
a: [2, 1, 0, 2, 0, 1, 1, 1]
b: [2, 1, 1, 1, 1, 0, 1, 1]
```

Figura 17 Cosine Similarity - vectori

Aproximativitatea dintre cele două texte este de 0.822 (1 reprezintă 100%).

4.6 Jaro-Winkler Algorithm

Acest algoritm se bazează pe măsurarea distanței de editare dintre două frecvențe. Distanța Jaro-Winkler folosește o scală de prefix care oferă evaluări mai favorabile pentru șiruri de caractere care se potrivesc de la început pentru o lungime prestabilită.

Formula pentru aceasta este de forma următoare:

$$d_j = \frac{1}{3} \left(\frac{m}{|s_1|} + \frac{m}{|s_2|} + \frac{m - t}{m} \right)$$

Formula de distanță Jaro

Figura 18 Algoritmul Jaro Winkler - formula

d_j - reprezintă distanța Jaro;

m - reprezintă numărul de caractere care se potrivesc (între s_1 și s_2)

t - reprezinta jumatatea din numarul de transpozitii

s1 - reprezinta lungimea primului string

s2 - reprezinta lungimea de-al doilea string

Pentru a da un exemplu o sa folosim tot cuvintele "martha" si "marhta" iar rezultatele sunt urmatoarele:

```
m = 6
t = 2/2 = 1 (2 cupluri de caractere necorespunzătoare, a 4-a și a
5-a) {t / h; h / t}
| s1 | = 6
| s2 | = 6
```

Figura 19 Algoritmul Jaro Winker - exemplu

```
dj = (%) (6/6 + 6/6 + (6-1) / 6) = % 17/6 = 0,944

Distanța de distanță = 94,4%
```

Figura 20 Algoritmul Jaro Winker- distanța Jaro

Formula de mai sus ne da distanța Jaro iar cu aceasta putem calcula distanța Jaro-Winkler. Similaritatea Jaro-Winker folosește o scară prefixă p care dă un rating mai favorabil la șirurile care se potrivesc de la început pentru o lungime prestabilită l.

p este un factor de scalare constant pentru cât de mult scorul este ajustat în sus pentru a avea prefixe comune. Valoarea standard a acestei constante în lucrarea lui Winkler este $p = 0,1$.

l este lungimea prefixului comun la începutul șirului (până la maxim 4 caractere).

$$d_w = d_j + (lp(1 - d_j))$$

Formula de distanță Jaro-Winkler

Figura 21 Algoritmul Jaro Winker- formula distanței Jaro-Winkler

Deci, înapoi la exemplul "martha" / "marhta", să luăm o lungime de prefix de $l = 3$ (care se referă la "mar"). Ajungem la:

```
dw = 0,944 + ((0,1 * 3) (1-0,944)) = 0,944 + 0,3 * 0,056 = 0,961

Distanța Jaro-Winkler = 96,1%
```

Figura 22 Algoritmul Jaro Winker- distanța Jaro-Winkler

În urma celor menționate putem stabili că similaritatea dintre cele două cuvinte "martha" și "marhta" este de aproximativ 96.1%.

În urma experimentelor anterioare am optat să folosesc biblioteca Draftable Compare API datorită faptului că poate crea o experiență mai bună datorită view-ului creat de API și datorită faptului că se poate integra cu tehnologiile optate pentru crearea aplicației.

5 Alegerea metodei de implementare

În urma experimentelor anterioare am optat să folosesc biblioteca Draftable Compare API datorită faptului că poate crea o experiență mai bună datorită view-ului creat de API și datorită faptului că se poate integra cu tehnologiile optate pentru crearea aplicației.

Algoritmul creat de John Resign nu este recomandat pentru documente de dimensiuni medii/mari și lasă de asemenea problema aspectului vizual deoarece este foarte greu de modificat un fișier de tip pdf pentru a face un highlight la text sau de a modifica conținutul acestuia.

De asemenea există suport online pentru Draftable Compare API pentru cazul în care există vreo problemă cu această bibliotecă. Oricine dorește să implementeze algoritmul lui John Resign nu beneficiază nici de un suport sau vreo garanție.

Draftable Compare API - Node.js Client Library

This is a thin Javascript client for Draftable's [document comparison API](#). It wraps the available endpoints, and handles authentication and signing for you. The library is [available on npm](#) as `@draftable/compare-api`.

See the [full API documentation](#) for an introduction to the API, usage notes, and other references.

Getting started

- Sign up for free at api.draftable.com to get your credentials.
- `npm install @draftable/compare-api`
- Instantiate the client:

```
const client = require('@draftable/compare-api').client(<yourAccountId>, <yourAuthToken>);
const comparisons = client.comparisons;
```

- Start creating comparisons:

```
comparisons.create({
  left: {
    source: 'https://api.draftable.com/static/test-documents/code-of-conduct/left.rtf',
    fileType: 'rtf',
  },
  right: {
    source: 'https://api.draftable.com/static/test-documents/code-of-conduct/right.pdf',
    fileType: 'pdf',
  },
}).then(function(comparison) {
  console.log("Comparison created:", comparison);
  # This generates a signed viewer URL that can be used to access the private comparison.
  # By default, the URL will expire in 30 minutes. See the documentation for `signedViewerURL(...)`.
  console.log("Viewer URL (expires in 30 min):", comparisons.signedViewerURL(comparison.identifier));
});
```

Figura 23 Draftable - prezentare

Pentru a folosi acest API este necesar un cont de pe site-ul "<https://api.draftable.com/>". După ce este creat un cont pentru development se poate accesa secțiunea "account" în care sunt stocate tokenurile pentru a folosi acest API (aceste token-uri sunt folosite în NodeJS pentru autentificare).

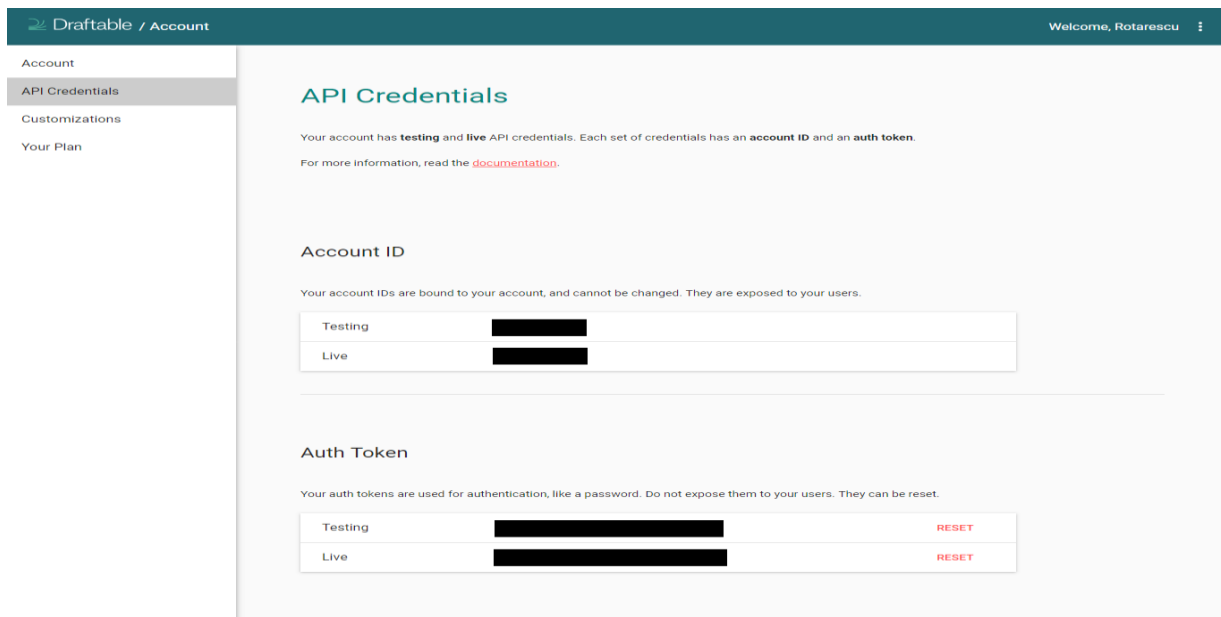


Figura 24 Draftable - credentiale api

Tot de pe acest site se poate accesa documentatia la acest API si anume metodele de autentificare, tehnologiile disponibile si de asemenea resursele disponibile. Fiecare cont are un anumit numar de utilizari disponibile pe luna iar aceasta difera in functie de tipul contului (contul free are in jur de 200 de comparari de text disponibile pe luna)

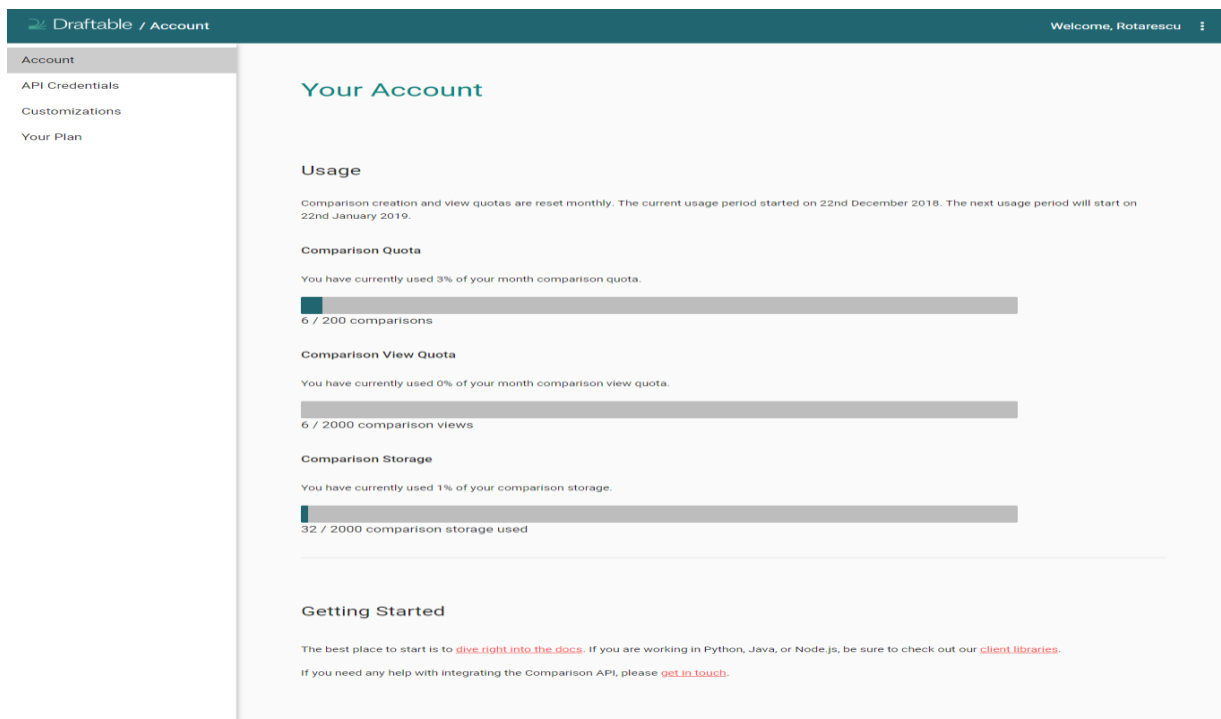


Figura 25 Draftable - cont

6. Configurare backend

Pentru ca acest proiect sa foloseasca resursele mentionate anterior partea de backend este configurata in NodeJS. Cu ajutorul acestuia se poate configura un server de backend ce poate fi folosit pentru a manipula fisiere, autentificare si multe altele.






	files	04-Jan-19 1:17 PM	File folder	
	node_modules	09-Dec-18 11:08 AM	File folder	
	main	09-Dec-18 11:52 AM	JS File	7 KB
	package.json	09-Dec-18 11:08 AM	JSON File	1 KB
	package-lock.json	09-Dec-18 11:08 AM	JSON File	26 KB

Figura 26 Backend files

Fisierul ce este responsabil pentru backend este main.js din folder-ul backend iar in acesta avem create un server cu cateva configurari de baza pentru server, autentificarea pentru Draftable API (e nevoie de user si parola // line 9)

```
1  const express = require('express');
2  const app = express();
3  const mongoose = require('mongoose');
4  const fs = require('fs');
5  const removeAccents = require('remove-accents');
6  // const PDFParser = require("pdf2json");
7  const pdf = require('pdf-parse');
8
9  const client = require('@draftable/compare-api').client("xxxxxx-test", "xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx");
10 const comparisons = client.comparisons;
11
12 var busboy = require('connect-busboy');
13 // let pdfParser = new PDFParser(this,1);
14 app.use(busboy());
15
16 // use it before all route definitions
17 app.use(function (req, res, next) {
18
19   // Website you wish to allow to connect
20   res.setHeader('Access-Control-Allow-Origin', '*');
21
22   // Request methods you wish to allow
23   res.setHeader('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH, DELETE');
24
25   // Request headers you wish to allow
26   res.setHeader('Access-Control-Allow-Headers', 'Cache-Control, Access-Control-Allow-Headers, Origin,Accept, X-Requested-With, Content-Type, Acc
27
28   // Set to true if you need the website to include cookies in the requests sent
29   // to the API (e.g. in case you use sessions)
30   res.setHeader('Access-Control-Allow-Credentials', true);
31
32   // Pass to next layer of middleware
33   next();
34 });
35
```

Figura 27 backend implementation (1)

Serverul este creat cu cateva routes/rute de backend de care se poate lega frontend-ul pentru a compara fisiere, inserare useri si login:

1. Prima ruta este "/compareApi". Aceasta foloseste doua fisiere stocate pe un server online si sunt special modificate pentru a se observa modificarile pe ace acestea le au. Acest exemplu este creat de catre dezvoltatorii API-ului "Draftable" si este dat pentru a oferi un exemplu rapid al API-ului.

```

app.get('/compareApi', function (req, res) {
  comparisons.create({
    left: {
      source: 'https://api.draftable.com/static/test-documents/code-of-conduct/left.rtf',
      fileType: 'rtf',
    },
    right: {
      source: 'https://api.draftable.com/static/test-documents/code-of-conduct/right.pdf',
      fileType: 'pdf',
    },
  }).then(function(comparison) {
    console.log("Comparison created:", comparison);
    // # This generates a signed viewer URL that can be used to access the private comparison.
    // # By default, the URL will expire in 30 minutes. See the documentation for 'signedViewerURL(...)'.
    console.log("Viewer URL (expires in 30 min):", comparisons.signedViewerURL(comparison.identifier));
    res.send( {
      URL: comparisons.signedViewerURL(comparison.identifier),
      comparison: comparison
    })
  });
});

```

Figura 28 backend implementation (2)

2. A doua ruta creata este `"/compareTwoFilesApi"` iar cu ajutorul acesteia se pot trimite doua fisiere tip PDF de pe calculator in backend si sa se extraga diferentele dintre cele doua documente folosind metoda `comparison.create()` (file inputs).

```

app.post('/compareTwoFilesApi', function (req, res) {
  var fstream;
  var paths = [];
  req.pipe(req.busboy);
  req.busboy.on('file', function (fieldname, file, filename) {
    fstream = fs.createWriteStream(__dirname + '/files/' + removeAccents(filename));
    paths.push(__dirname + '/files/' + removeAccents(filename));
    file.pipe(fstream);
  });
  req.busboy.on('finish', function() {
    console.log( fs.existsSync(paths[0]), fs.existsSync(paths[1]) )
    if(fs.existsSync(paths[0]) && fs.existsSync(paths[1])){
      comparisons.create({
        left: {
          source: fs.readFileSync(paths[0]),
          fileType: 'pdf',
        },
        right: {
          source: fs.readFileSync(paths[1]),
          fileType: 'pdf',
        },
      }).then(function(comparison) {
        console.log("Comparison created:", comparison);
        // # This generates a signed viewer URL that can be used to access the private comparison.
        // # By default, the URL will expire in 30 minutes. See the documentation for 'signedViewerURL(...)'.
        console.log("Viewer URL (expires in 30 min):", comparisons.signedViewerURL(comparison.identifier));
        res.send( {
          URL: comparisons.signedViewerURL(comparison.identifier),
          comparison: comparison
        })
      });
    }
  });
});

```

Figura 28 backend implementation (3)

3. A treia ruta creata este `"/login"` iar cu ajutorul careia un utilizator se poate autentifica in aplicatie.

```

app.post('/login', function (req, res) {
  let user = new users({ username: req.body.username, password: req.body.password});
  users.findOne({ username: req.body.username, password: req.body.password}, (err, resp) => {
    console.log(err, resp)
    if (err) return res.status(500).send({user, msg: "Error", status:500});
    if (!resp) return res.status(200).send({user, msg: "No user found!", status:200});
    return res.status(200).send({data: resp, msg: "User found!", status:200});
  });
});

```

Figura 29 backend implementation (4)

4. O alta ruta este cea de "/addUser" in care un utilizator admin poate crea un alt utilizator.

```
app.post('/addUser', function (req, res) {
  let user = new users({ username: req.body.username, password: req.body.password, type: req.body.type});
  user.save(err => {
    if (err) return res.status(500).send({user, msg: "Error", status:500});
    return res.status(200).send({user, msg: "User created", status:200});
  });
});
```

Figura 30 backend implementation (5)

5. O alta ruta de backend este '/compareLocalFiles' cu ajutorul careia un utilizator poate alege doua fisiere salvate intr-o baza de date.

```
app.post('/compareLocalFiles', function (req, res) {
  if(req.body.one && req.body.two){
    comparisons.create({
      left: {
        source: fs.readFileSync(__dirname + '/files/' + removeAccents(req.body.one)) ,
        fileType: 'pdf',
      },
      right: {
        source: fs.readFileSync(__dirname + '/files/' + removeAccents(req.body.two)) ,
        fileType: 'pdf',
      },
    }).then(function(comparison) {
      console.log("Comparison created:", comparison);
      // # This generates a signed viewer URL that can be used to access the private comparison.
      // # By default, the URL will expire in 30 minutes. See the documentation for `signedViewerURL(...)`.
      console.log("Viewer URL (expires in 30 min):", comparisons.signedViewerURL(comparison.identifier));
      res.send( {
        URL: comparisons.signedViewerURL(comparison.identifier),
        comparison: comparison
      })
    });
  }
});
```

Figura 31 backend implementation (6)

6. O alta ruta de backend este '/getDocuments' cu ajutorul careia se incarca documentele salvate intr-un DB.

```
app.get('/getAllDocuments', function (req, res) {
  filesaves.find({}, function(err, text) {
    res.send(text)
  })
});
```

🔍 All files (4)

Prezentare ISBD SQL vs NoSQL.pdf
Prezentare ISBD SQL vs NoSQL - Copy.pdf
Measuring the health of open source software ecosystems Beyond.pdf
document.pdf

Figura 32 33 backend implementation (7)

6. O alta ruta de backend este '/uploadPDF' cu ajutorul careia se insereaza datele unui document intr-un DB.

```

app.post('/uploadPDF', function (req, res) {
  var fstream;
  req.pipe(req.busboy);
  req.busboy.on('file', function (fieldname, file, filename) {
    fstream = fs.createWriteStream(__dirname + '/files/' + removeAccents(filename));
    file.pipe(fstream);
    fstream.on('close', function () {
      let dataBuffer = fs.readFileSync(fstream.path);
      pdf(dataBuffer).then(function(data) {
        let doc = new filesaves({ title: filename });
        doc.save(err => {
          if (err) return res.status(500).send(err);
          return res.status(200).send(doc);
        });
      });
    });
  });
});
});
});

```

Figura 34 backend implementation (8)

Pentru acest proiect am ales sa folosesc baze de date NoSQL datorita faptului ca documentele pe care le dorim analizate nu pot avea un pattern concret sau o structura bine definita. Datorita acestui lucru consider ca baze de date non relationale sunt mai eficiente pentru stocarea datelor.

Am ales pentru acest proiect sa folosesc baze de date de pe site-ul <https://mlab.com>. Cu ajutorul acestui site pot sa creez baze de date tip NoSQL pentru a stoca date si a rula cateva teste.

Pentru acest proiecta am facut o baza de date numita "cheat". In aceast DB am creat o colectie pentru stocarea datelor numita "filesaves". Bazele de date NoSQL folosesc colectii in loc de tabele asa cum este traditional in SQL/MySQL.

Home

MongoDB Deployments

Create from backup Create new

Development and Utility Single-node deployments intended for environments that do not require high availability.

DEPLOYMENT	PLAN TYPE	RAM	SIZE	SIZE ON DISK
ds235328/cheat	Sandbox	shared	56.92 KB	272.00 MB

Environments

Create new

None exist at this time. Click "Create new" to create an [mLab Environment](#) (VPO).

Figura 35 Mongoddb DB

In aceasta baza de date avem doua colectii de date si anume filesaves si users. Filesaves contine datele despre fisierele comparate si users contine date despre conturile utilizatorilor.

Sandbox databases do not have redundancy and therefore are not suitable for production. Read our documentation on [how to upgrade](#).

Collections Users Stats Backups Tools

Collections

Delete all collections Add collection

NAME	DOCUMENTS	CAPPED?	SIZE
filesaves	4	false	8.42 KB
users	2	false	8.20 KB

Figura 36 Mongoddb collections

7. Aplicatie UI/UX

In acest capitol este prezentata aplicatia din punct de vedere al user interface (UI). Pentru aceasta aplicatie am folosit libraria de css Bootstrap pentru reda un aspect vizual mai frumos si sa fie de asemenea user friendly. Paginile principale pe care aceasta aplicatie o are sunt:

7.1 Login

Orice aplicatie are o pagina de login iar in consecinta si acest web app are o pagina dedicata autentificarii utilizatorului.

Este o autentificare simpla prin username/password iar aceste credentiale pot fi generate de utilizatori de tip admin.

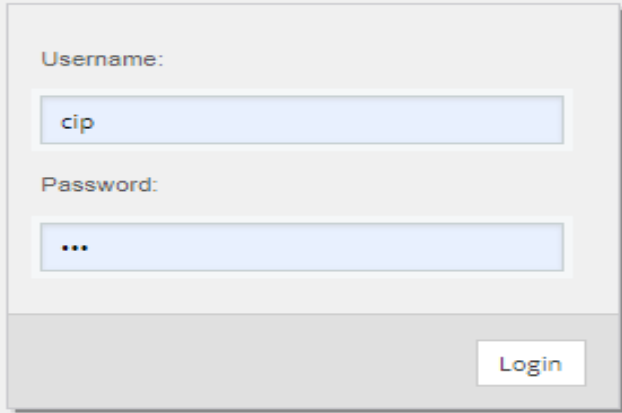
The image shows a login form with a light gray background. It contains two input fields: the first is labeled 'Username:' and contains the text 'cip'; the second is labeled 'Password:' and contains three dots '...'. Below these fields is a 'Login' button. The form is enclosed in a thin gray border.

Figura 37 MongoDB UI - Login

7.2 Dashboard

Pagina principala a aplicatiei o reprezinta pagina "Dashboard" in care un utilizator poate sa compare documentele de tip pdf fie prin file input sau prin selectarea documentelor salvate in baza de date prin lista din stanga paginii. Ficare actiune are o sectiune separata in pagina iar din punct de vedere UX este foarte usor de interpretat chiar si fara un manual de utilizare.

In partea din stanga a paginii se afla rezultatul obtinut de catre Comparison API iar acesta dureaza cateva momente de initializare din momentul in care faci o comparare de fisiere. Pentru a adauga documente in sectiunea "All files" se acceseaza pagina "Library" din headerul aplicatiei.

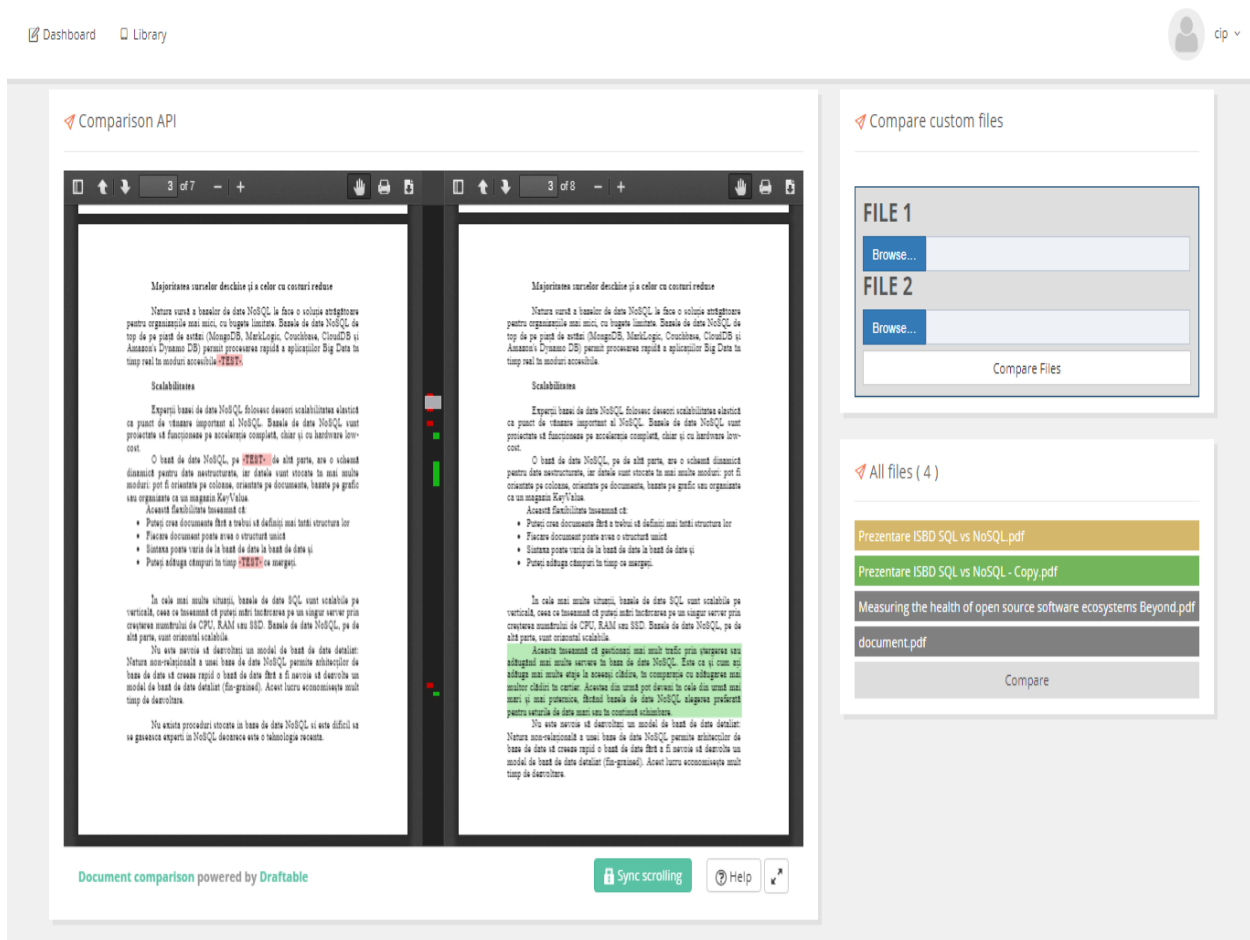


Figura 38 Mongodb UI - Dashboard

In partea de UI utilizatorul are in partea dreapta din pagina de dashboard o lista in care poate selecta ce documente sa compare

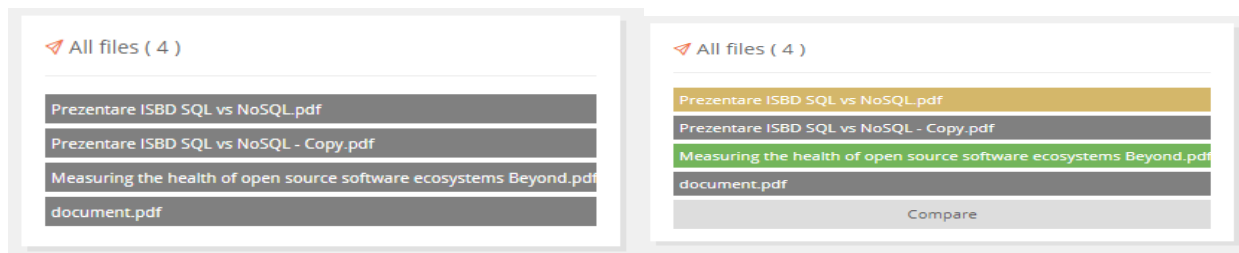


Figura 39 Mongodb UI - Dashboard pdf files

7.3 Library

Aceasta pagina este responsabila de a adauga datele unui fisier pdf (numele, tipul fisierului de ex) in baza de date NoSQL si de asemenea de a afisa ce fisiere sunt disponibile in DB. Pentru a adauga un fisier pdf se foloseste de formularul din aceasta pagina iar dupa ce utilizatorul a incarcat fisierul si a

apasat butonul de upload, aplicatia o sa se actualizeze si deasemenea numele fisierului trimis in baza de date este redat in lista de sub formular.

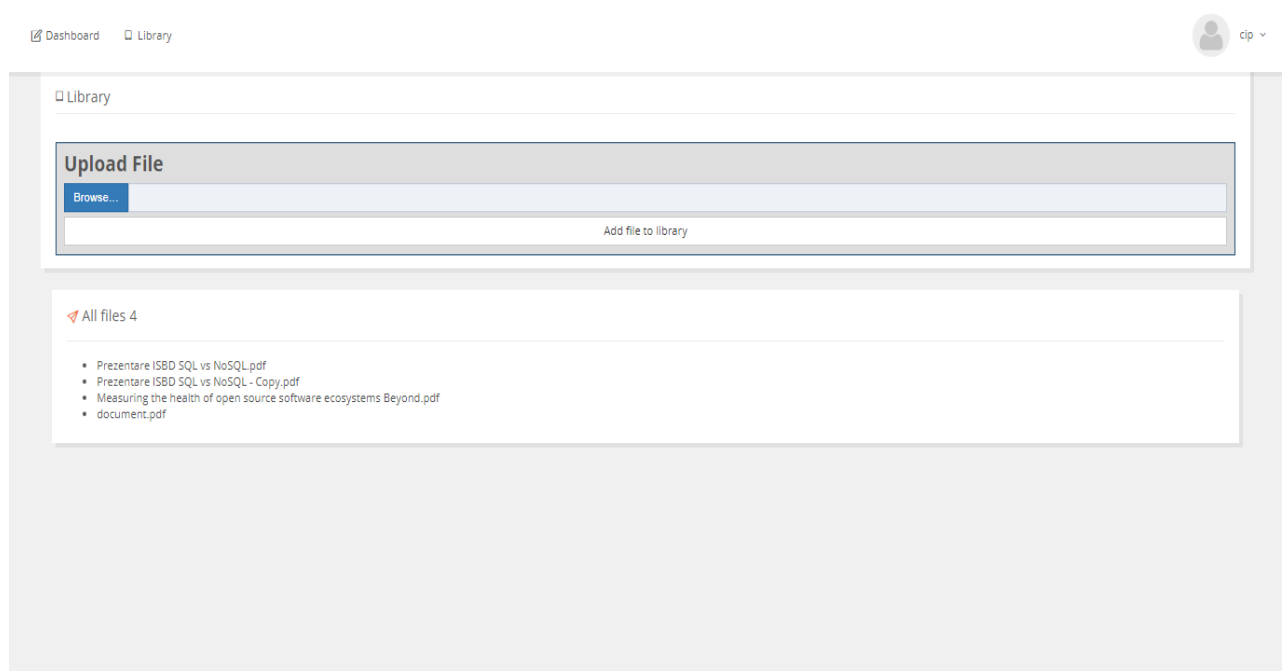


Figura 40 MongoDB UI - Library

7.3 Create

Ultima setiune din aceasta aplicatie este cea de create utilizator. Aceasta sectiune este prezenta doar utilizatorilor de tip "admin". In acest formular sunt necesare numele userului nou creat, parola acestuia si de asemenea ce tip de utilizator o sa aiba contul nou creat ("normal" / "admin").

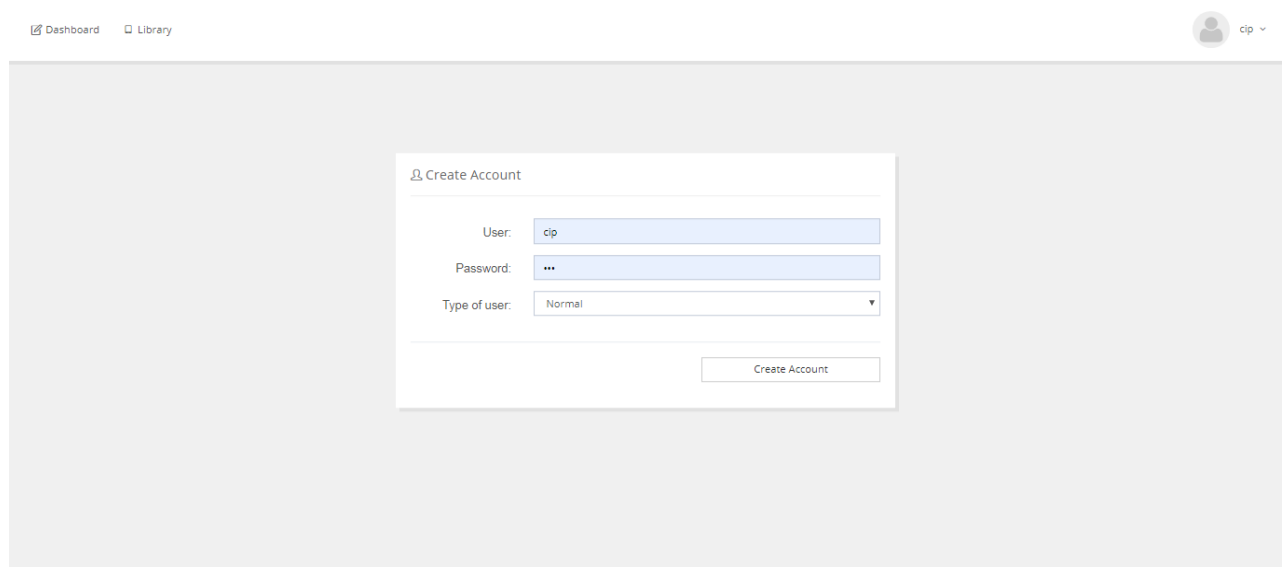


Figura 41 MongoDB UI - Create

8. Teste

În acest capitol sunt redată rezultatele aplicației implementând biblioteca Draftable Compare API menționată anterior. De asemenea este bine de menționat că sunt folosite câteva documente aproape identice, acestea având câteva modificări cum ar fi cuvinte, numere sau blocuri de text modificate sau extrase.

Pentru primul test am folosit un document numit "Measuring the health of open source software ecosystems Beyond" și am creat o clonă a acestuia adăugând prefixul -copy la finalul documentului. Clona documentului are câteva cuvinte adăugate ("inserted text"), modificat anul de pe prima pagină (2019-2018) și sunt extrase câteva blocuri de text din document.

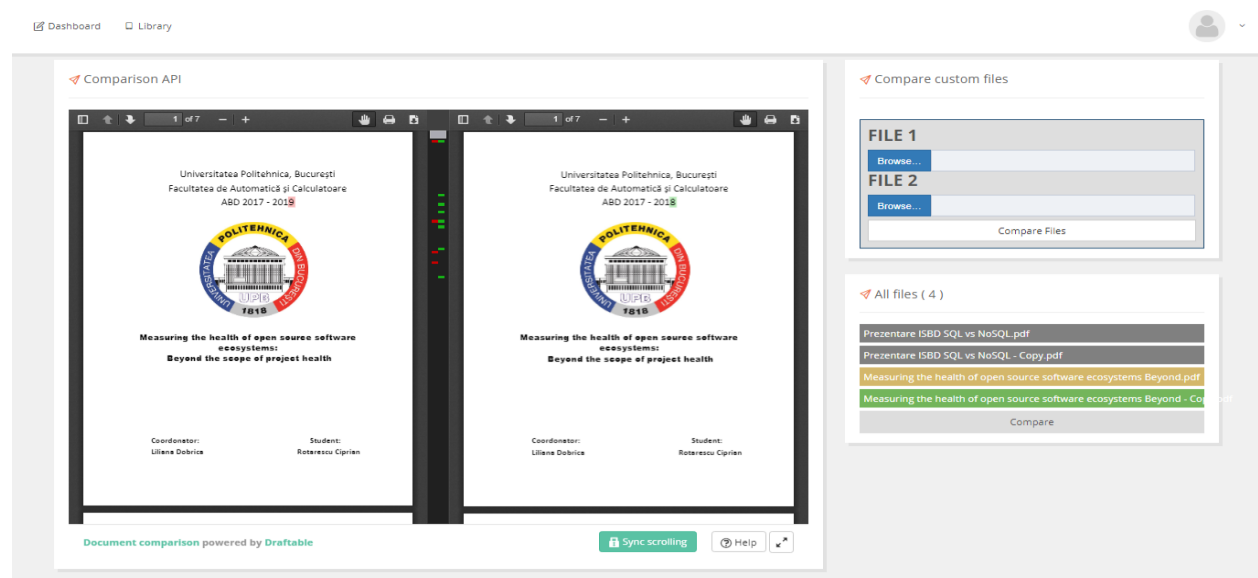


Figura 42 Teste - "Measuring the health of open source software ecosystems Beyond"

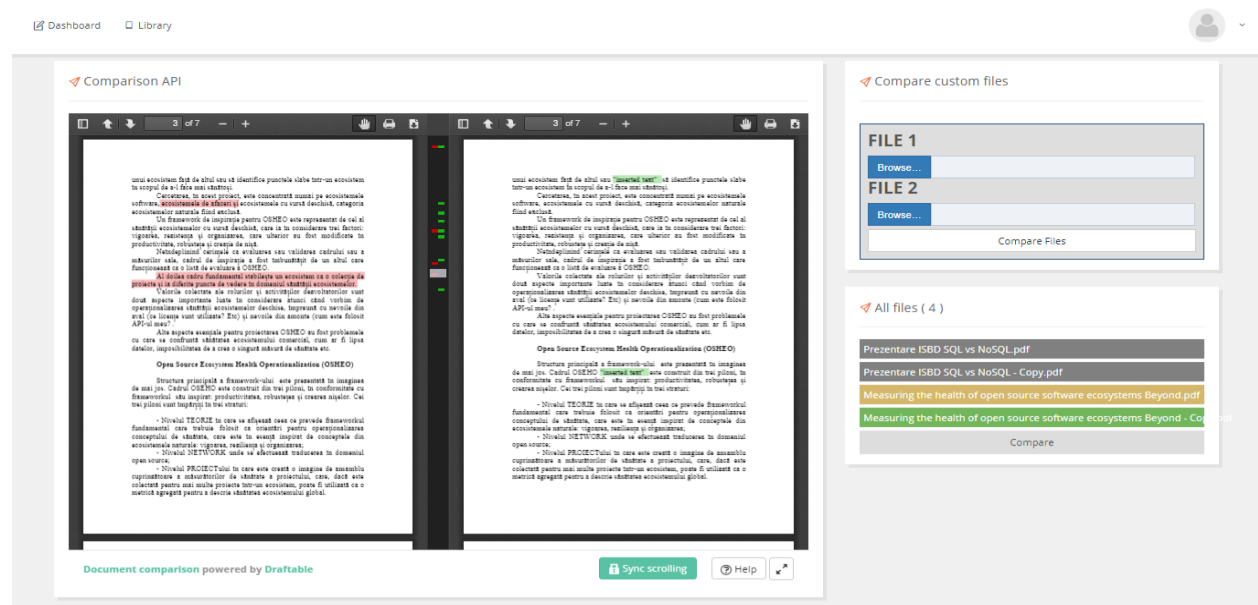


Figura 43 Teste - "Measuring the health of open source software ecosystems Beyond"

Se pot evidenta cuvintele adaugate/modificate in partea dreapta ca au un fundal de culoarea verde iar in partea stanga aceleasi cuvinte au un fundal rosu. E de mentionat ca si blocurile de text extrase in partea dreapta au un fundal rosu in documentul din stanga.

Un alt exemplu pe care l-am testat cu ajutorul aplicatiei ce face subiectul acestuei dizertatii, a fost de a compara alte doua documente tip pdf numite "Prezentare ISBD SQL vs NoSQL" si "Prezentare ISBD SQL vs NoSQL - Copy".

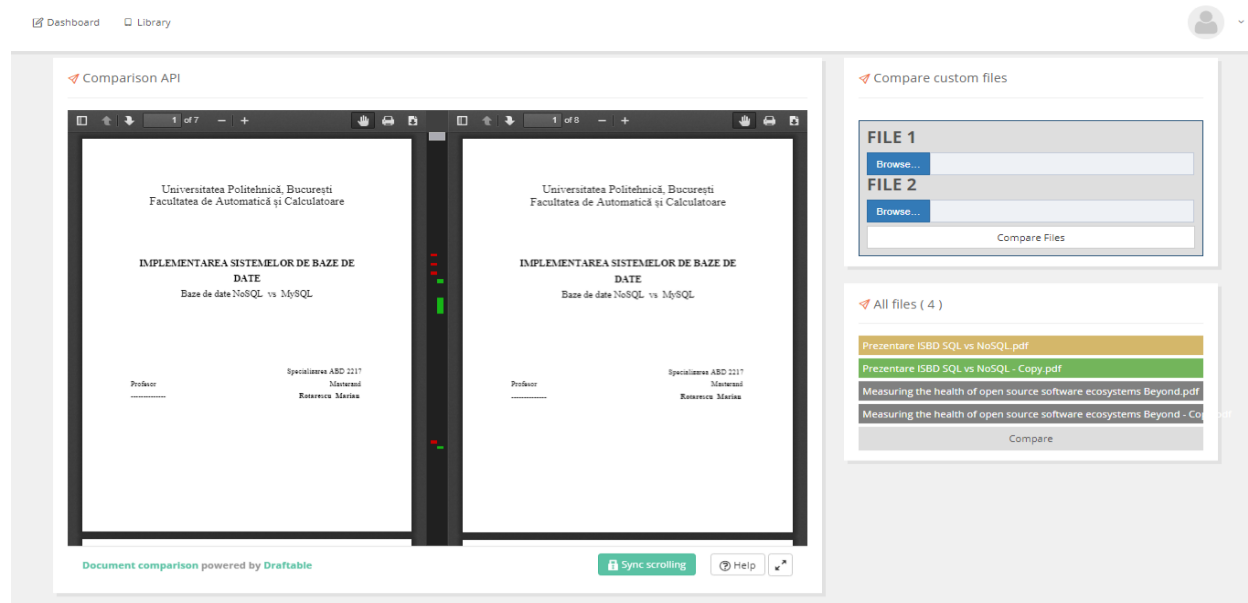


Figura 44 Teste - " Prezentare ISBD SQL vs NoSQL "

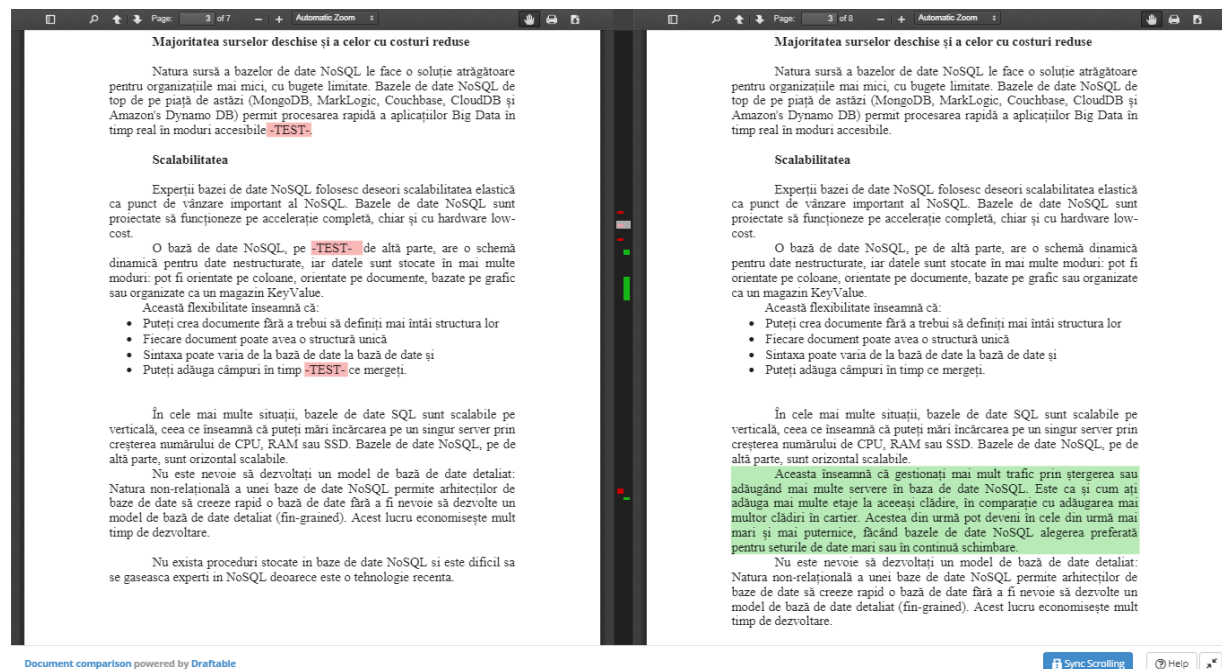


Figura 45 Teste - " Prezentare ISBD SQL vs NoSQL "

Experimentul este asemanator cu cel anterior avand doar o simpla observatie si anume ca descrierea cuvintelor sunt scrise sub forma "-TEST-"

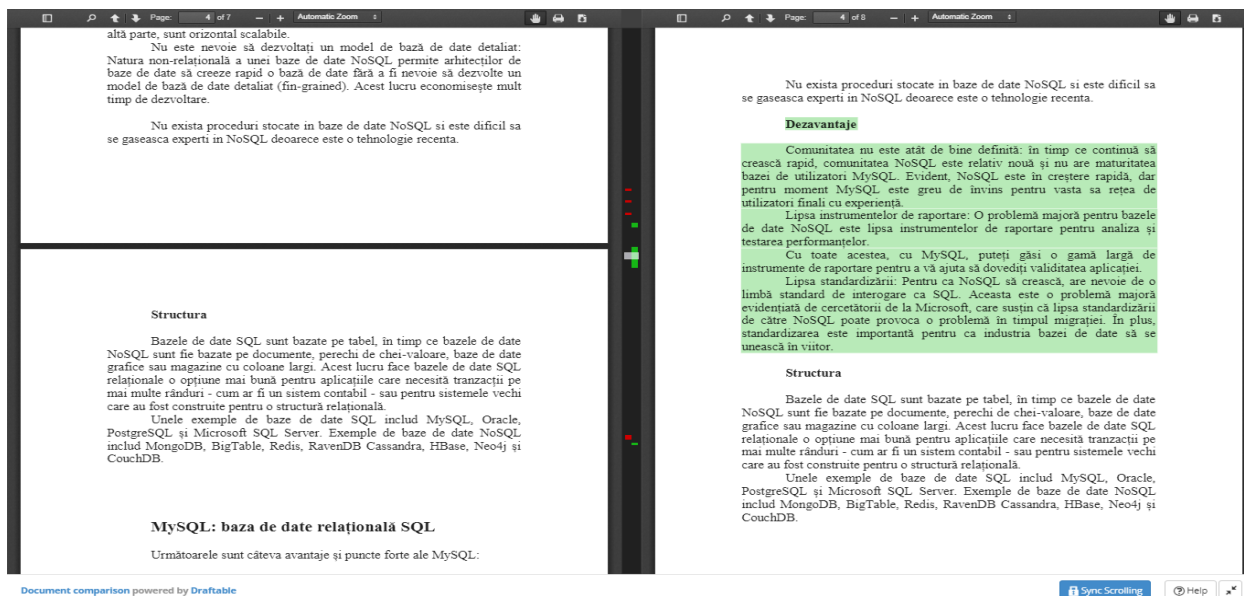


Figura 46 Teste - "Prezentare ISBD SQL vs NoSQL "

Sectiunea "Dezavantaje" a fost decupat dintr-un document si se poate observa fara probleme in imaginea de mai sus.

De asemenea este important de mentionat ca acest API poate sa identifice textele cu diacritice pentru documentele in limba Romana.

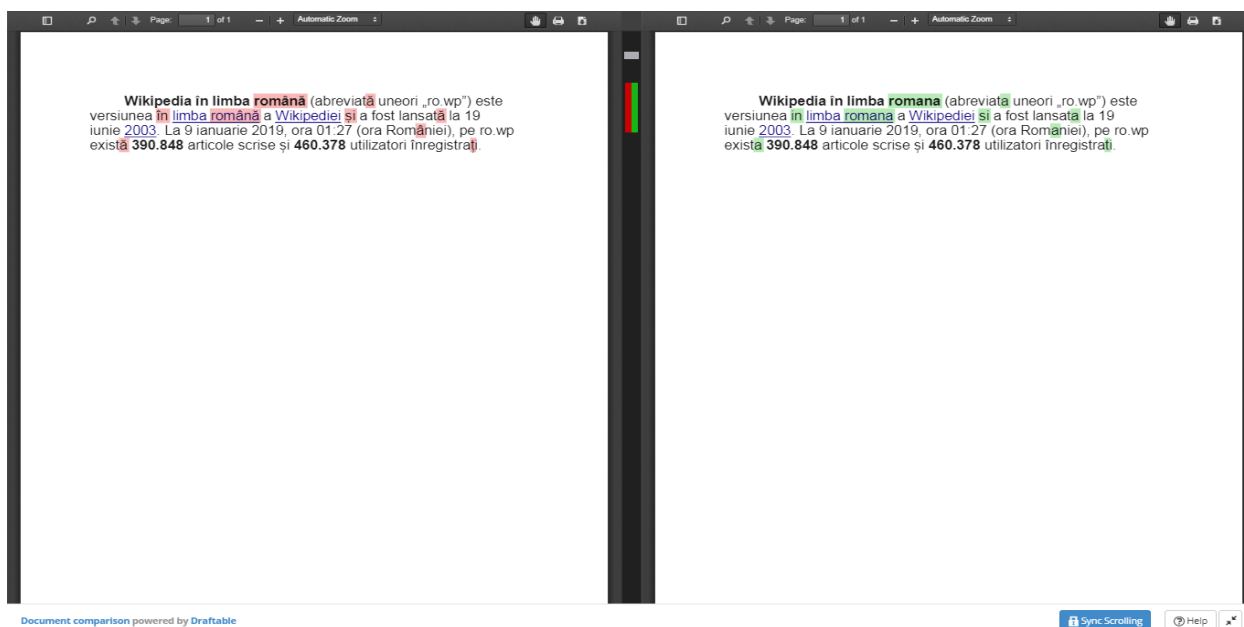


Figura 47 Teste - "Diacritice"

9. Concluzii

Pentru detectarea plagiatului si in urma testelor mentionate s-a dovedit faptul ca acest API - DRAFTABLE poate fi folosit pentru indeplinirea scopului propus . Exista insa si un drawback legat de acest API si anume ca trebuie platit pentru a avea acces la un numar ridicat de comparari si alte beneficii.

Algoritmii prezentati in capitolul 4 sunt folositori dar exista problema modului de afisare a rezultatelor. Datorita acestei probleme Draftable API este solutia cea mai rapida pentru rezolvarea atat acestui task de afisare cat si a corectitudinii datelor.

Aceasta aplicatie poate fi folositoare pentru persoanele care doresc sa vada evolutia documentelor printr-o interfata simpla si usor de folosit (de exemplu evolutia unei documentatii).

Aplicatia poate fi imbunatatita prin implementarea modului de autentificare prin email / facebook / gmail si de asemenea de inserare a altor functionalitati.

10. Bibliografie

<https://api.draftable.com/>

<https://nodejs.org/en/about/>

<https://github.com/draftable/compare-api-node-client>

<https://www.quirksmode.org/js/intro.html>

https://www.tutorialspoint.com/nodejs/nodejs_introduction.htm

<https://en.wikipedia.org/wiki/JavaScript>

<https://www.lifewire.com/what-is-html-3482374>

https://en.wikipedia.org/wiki/Cascading_Style_Sheets

<https://docs.angularjs.org/guide/introduction>

<https://www.upwork.com/hiring/development/angularjs-basics/>

<https://medium.com/@sumn2u/string-similarity-comparision-in-js-with-examples-4bae35f13968>

https://en.wikipedia.org/wiki/Levenshtein_distance