# Ingineria programarii
# Proiect 2017-2018
# Grupa B3

In zilele noastre exista o colectie impresionanta de limbaje de programare si framework-uri din care am putea alege. Acest lucru poate fi atat un beneficiu dar si un dezavantaj. Deoarece avem destul de multe optiuni, uneori tindem sa alegem un anumit limbaj sau un framework dupa posibilitatile oferite de acestea, dar acest fapt poate, in unele cazuri, sa ne ingreuneze munca ne fiind familiarizati cu limbajul sau framework-ul.

Pentru a putea verifica corectitudinea lucrarilor avem nevoie de un compilator (ex. GNU). Un avantaj al compilatorului GNU este portabilitatea si usoara instalare a acestuia. Un dezavantaj destul de serios este faptul ca nu se pot folosi in programele incarcate functii ca strrev, itoa etc.

Baza de date va fi una te tip SQL, deoarece acestea folosesc un model de date relational ce este alcatuit dintr-un ansamblu de tabele (relatii) impreuna cu relatiile dintre ele. Un avantaj al folosirii acestui tip de baze de date ar fi acela ca este foarte bine documentata pe internet si putem gasi un raspuns la o anumita problema intr-un timp scurt, lucru ce favorizeaza eficienta in lucru. Dezavantajele ar fi acelea ca optimizarile se fac prin definitiile index. Acest lucru putand fi dificil atunci cand un utilizator populeaza baza de date, fiind mereu nevoie de a se restructura pentru eficienta.

Pentru a da posibilitatea de a compara doua sau mai multe lucrari, vom creea, cu ajutorul atat al unor scripturi javascript cat si a codului scris in html si css, o interfata cat mai atractiva si usor de folosit pentru utilizatori.

In cadrul exportarii si importarii formularelor sau al rapoartelor din baza de date cat si a generarii de date bazate pe ele, vom apela la limbajul python pentru a face acest lucru posibil. Avantajul acestei metode il contitue faptul ca este bine documentata pe internet, este eficient si usor de inteles. Dezavantajul ar putea interveni atunci cand construim baza de date, in elaborarea corecta a tabelelelor si a relatiilor dintre ele.

Pentru partea de back-end alegem sa folosim ca limbaj de programare Java, deoarece este un limbaj matur, existand solutii pentru majoritatea problemelor intalnite. Avand avatajul ca este un limbaj foarte bine dezvoltat, solutionarea problemelor intalnite nu este o problema, gasind usor raspund la orice intrebare. Dezavantajul ar fi acela ca exista posibilitatea ca anumite functionalitati care vor fi sau nu adaugate ulterior sa necesite anumite functionalitati pe care Java nu le poate rezolva. Ca si framework pentru Java, vom folosi Spring, deoarece este cel mai bine cotat framework de backend pentru Java, impreuna cu un server Apache si Maven.

Pentru partea de front-end alegem, pentru inceput, sa nu folosim niciun framework, deoarece unele dintre ele nu sunt usor de inteles pentru toata lumea. Daca anumite functionalitati vor necesita un anumit framework vom folosi unul. Avantajele nefolosirii unui framework sunt acelea ca putem stapani mult mai bine tot ceea ce avem de facut mult mai usor decat daca o anumita functionalitatea a unui framework ar face-o pentru noi. Dezavantajul ar fi eficienta si viteza de codare, acest lucru fiind datorat lipsei unui framework ce ar putea face anumite lucruri automat pentru noi.

Ca si design pattern, vom folosi MVC, deoarece este unul dintre cele mai populare, usor de inteles si usor de pus in practica. Acest lucru ne va asigura o structura clara a codului.

Plagiatul in texte este o problema de ingrijorare crescanda a comunitatii academice. Acum, cea mai frecventa plagiatura a textului apare facand o varietate de modificari minore care includ inserarea, stergerea sau substituirea cuvintelor. Astfel de schimbari simple necesita comparatii excesive de sir.

### *Algoritm folosind distanta Hamming*

Una dintre cele mai populare masuri este distanta Hamming, care este egala cu numarul de simboluri de neconcordanta intre doua siruri de lungime egala, sau infinit daca sirurile au lungimi diferite. Distanta Hamming este complet incompatibila, deoarece cele doua texte probabil ca nu vor avea aceeasi lungime.

### *Algoritm folosind distanta Levenshtein*

Valoarea distantei descrie numarul minim de stergeri, insertii sau substitutii care sunt necesare pentru a transforma un sir (sursa) in alt (tinta). Spre deosebire de distanta Hamming, distanta Levenshtein functioneaza pe siruri cu o lungime inegala. Cu cat distanta dintre Levenshtein este mai mare, cu atat mai mare este diferenta dintre siruri de caractere. De exemplu, de la "test" la "test", distanta Levenshtein este 0, deoarece atat sirurile sursa cat si tinta sunt identice. Nu sunt necesare transformari. In schimb, de la "test" la "echipa", distanta Levenshtein este de 2 - trebuie facute doua substitutii pentru a transforma "testul" in "echipa". Algoritmul pentru a calcula distant Levenshtein este numit algoritmul Wagner-Fischer. Algoritmul atribuie fiecare simbol al unui sir unei coloane intr-o grila gigantica si fiecare simbol al celuilalt sir la un rand. Apoi, pornind in coltul din stanga sus si inundand diagonala peste grila, se umple fiecare patrat cu numarul de modificari necesare pentru a intoarce sirul care se termina cu coloana corespunzatoare in sirul care se termina cu randul corespunzator. Experimentele efectuate arata ca Levenshtein distanta este o masura buna pentru detectarea plagiatului, iar o limita de 70% este considerata suficienta pentru a servi codul plagiat si codul non-plagiat.
Cu toate acestea, algoritmul nu este sigur. Cateva strategii de a forta o detectie falsa sau o falsa trecere exista.

### *Algoritmul LCS*

Avand doua secvente A si B, cu lungimi m si n, respectiv, unde $m \leq n$, The longest common subsequence (LCS) reprezinta gasirea subsecventei comune a lui A si B cu lungime maxima. O subsecventa a secventei. este o secventa care poate fi obtinuta prin stergerea
unui numar arbitrar de elemente in pozitii arbitrare. Gasirea numarului minim de operatii de editare
(inserare, stergere si substituire) la transformarea secventei A in B este definita ca editarea problema de distanta In cazul in care costul de o
inserarea sau o stergere este de 1 si costul unei singure
substitutii este 2 (vazuta ca o insertie plus o stergere), atunci aceasta distanta de editare speciala a doua secvente A si B pot fi obtinute prin LCS.

### *Algoritmul Rabin-Karp*

Algoritmul Rabin-Karp este un algoritm de potrivire a sirurilor care utilizeaza o functie hash

pentru a compara sirul cautat cu un substring intr-un text . Daca ambele valori hash sunt aceleasi, atunci comparatia se va face din nou pe caractere. Daca rezultatele ambelor nu sunt aceleasi, atunci substringul va fi mutat spre dreapta. Fiecare caracter, subsecventa textului, va fi comparat. Daca valorile hash nu sunt la fel, algoritmul va calcula valoarea hash pentru caracterele urmatoare subsecvente, iar daca valorile hash sunt aceleasi, atunci algoritmul va efectua o comparatie intre modelul si caracterul subsecvent. In acest fel, ar exista o singura comparatie per subsecventa textului doar daca valorile hash sunt egale. Este mult mai indicat pentru documente de dimensiune mai mare decat alti algoritmi.

### *Algoritmul Jaro-Winkler*

Algoritmul Jaro-Winkler este un algoritm pentru masurarea asemanarii dintre doua siruri si cea mai mare parte a acestui algoritm sunt folosite in domeniul detectiei duplicarii . Cu cat valoarea distantei Jaro-Winkler este mai mare pentru doua siruri de caractere indica o similaritate mai mare a celor doua siruri. Valoarea normala este 0, ceea ce indica absenta similitudinii si 1 care indica existenta unor asemanari exacte .
Baza acestui algoritm are trei parti, si anume:
1. Calculati lungimea sirului,
2. Gasiti acelasi numar de caractere in cele doua siruri,
3. Gasiti cantitatea de transpunere.
Algoritmul Jaro-Winkler Distance nu poate decat sa examineze asemanarea dintre documente identice sau similare.

***Materialul a fost realizat de Proca Teodor si Elena Vizitiu.***