Universitatea Națională de Știință și Tehnologie POLITEHNICA București
Școala Doctorală de Inginerie Industrială și Robotică

TEZĂ DE DOCTORAT

OPTIMIZAREA PROCESELOR DE PLANIFICARE A ACTIVITĂȚILOR DIDACTICE PRIN ALGORITMI SPECIFICI INGINERIEI ȘI DEZVOLTAREA UNEI PLATFORME INFORMATICE INTEGRATE

Student doctorand,

Ing. CÎTU (TUNSOIU) M. Daniela

Conducător științific,

Prof. univ. dr. ing. DOICIN Cristian-Vasile

București
– 2025–

CUPRINS

LISTĂ ABREVIERI . . . . . **VII**
INTRODUCERE . . . . . **XI**
DEFINIȚII ȘI TERMINOLOGIE . . . . . **XIII**
STRUCTURA ȘI CONȚINUTUL TEZEI DE DOCTORAT . . . . . **XXI**
JUSTIFICAREA ALEGERII TEMEI . . . . . **XXIV**

PARTEA I. STADIUL ACTUAL PRIVIND MANAGEMENTUL DOCUMENTELOR DINTR-O UNIVERSITATE

CAPITOLUL 1. SISTEMUL DE ÎNVĂȚĂMÂNT SUPERIOR ȘI FLUXURILE DE DOCUMENTE **1**
1.1. Dinamica Învățământului Superior: Evoluția Numărului de Studenți 2
1.2. Organizarea sistemului de învățământ superior 4
1.3. Structura internă a instituțiilor de învățământ superior 5
1.4. Cicluri universitare 6
1.5. Clasificarea documentelor generate și gestionate într-o universitate/facultate 7
1.5.1. Documente administrative și de organizare academică 8
1.5.2. Documente privind evaluarea și performanța academică 8
1.5.3 Documente de organizare a programului studenților și profesorilor 9
1.6. Documente necesare realizării planificării activității didactice 10
1.6.1 Planul de învățământ 10
1.6.2 Formațiunile de studii 11
1.6.3 Statul de funcții 12
1.6.4 Nota de comandă 13
1.7. Relații de precedență între documentele implicate în realizarea unei planificări a activităților didactice 13

CAPITOLUL 2. STUDIUL APLICAȚIILOR ON-LINE EXISTENTE PENTRU MANAGEMENTUL DOCUMENTELOR DIN UNIVERSITATE **17**
2.1. Managementul documentelor universitare 17
2.2. Tipuri de aplicații utilizate în managementul documentelor universitare 18
2.3. Aplicații pentru managementul activităților administrative 19
2.3.1 Platforma de e-learning Moodle 19
2.3.2 People Soft Campus Solution Oracle 20
2.3.3 University Management System (UMS) 21
2.3.4 Platforma MyGJU Admin 22
2.4. Aplicații pentru planificarea activităților didactice 23
2.4.1 FET (Free Timetabling Software) 24
2.4.2 aSc TimeTables 25
2.4.3 Mimosa 26
2.4.4 Untis Express 26
2.4.5 TimeTabler 28

2.4.6 M-Sched 29
2.4.7 Bullet Time Tabler Education (BTTE) 29

CAPITOLUL 3. ALGORITMI UTILIZAȚI PENTRU MANAGEMENTUL DOCUMENTELOR ÎNTR-O UNIVERSITATE 30
3.1. Algoritmi pentru căutare 32
3.1.1 Algoritmul TF-IDF 32
3.1.2 Algoritmul BM 25 33
3.2. Algoritmi de clusterizare 35
3.2.1 Algoritmi de grupare ierarhică 36
3.2.2 Algoritmi de grupare bazați pe centroizi 38
3.3 Algoritmi de clasificare 41
3.3.1 Algoritmul arborelui de decizie 41
3.3.2 Mașină cu vector suport (SVM) 43
3.3.3 Naïve Bayes 44
3.4 Algoritmi de criptare și securitate 45

CAPITOLUL 4. ALGORITMI UTILIZAȚI ÎN MANAGEMENTUL ACTIVITĂȚILOR DIDACTICE 47
4.1 Algoritm Backtracking 47
4.1.1 Principiul de funcționare 47
4.1.2 Implementare și pseudocod 48
4.1.3 Complexitate, avantaje și dezavantaje 49
4.2 Algoritm Genetic 50
4.2.1 Principiul de implementare 50
4.2.2 Implementare și optimizări 51
4.3 Machine Learning pentru planificarea activităților didactice 55
4.3.1 Definirea proceselor de extragere 57
4.3.2 Extragerea datelor 58
4.3.3 Curățarea și procesarea datelor 59
4.3.4 Modelarea datelor 60
4.3.5 Analiza datelor 61
4.3.6 Ingineria caracteristicilor (Feature engineering) 62
4.3.7 Studiul algoritmilor ML 63
4.3.8 Alegerea algoritmului ML 65

PARTEA II. CONTRIBUȚII PROPRII PRIVIND MODELAREA, OPTIMIZAREA ȘI DEZVOTLAREA UNEI PLATFORME SOFTWARE DEDICATE

CAPITOLUL 5. OBIECTIVE ȘI METODOLOGIA DE CERCETARE 67
5.1. Scopul și obiectivele cercetării 67
5.2. Metodologia cercetării 68

CAPITOLUL 6. ANALIZA SOLUȚIILOR EXISTENTE ȘI A METODELOR DE PLANIFICARE A ACTIVITĂȚILOR DIDACTICE 71
6.1. Analiză comparativă a aplicațiilor existente pentru managementul activităților didactice 71

6.2. Analiza algoritmilor utilizați în cercetările existente 73
6.3. Problemele generării planificării activităților didactice 74
6.3.1 Definiția problemei și restricții 74
6.3.2 Provocările generării unei planificări optime a activităților 76
6.4. Comparație a algoritmilor analizați 77
6.5. Analiză Multicriterială 78
6.5.1 Stabilirea alternativelor 78
6.5.2 Criterii și ponderi 79
6.5.3 Matricea de performanță 79
6.5.4 Normalizarea 79
6.5.5 Ierarhizarea alternativelor 80

CAPITOLUL 7. ANALIZA CERINȚELOR UTILIZATORILOR ȘI STABILIREA CERINȚELOR PENTRU DEZVOLTAREA APLICAȚIEI 81
7.1. Importanța stabilirii cerințelor utilizatorilor 81
7.2. Procesul de stabilire a cerințelor 82
7.3. Metode de colectare a datelor 82
7.4. Colectarea datelor 83
7.4.1 Interviul 83
7.4.2 Observația directă 85
7.4.3 Analiza documentelor existente 87
7.4.4 Sondaje/Formular online 87

CAPITOLUL 8. MODEL MATEMATIC PENTRU MANAGEMENTUL ACTIVITĂȚILOR DIDACTICE 94
8.1. Model matematic ILP 94
8.1.1 Definirea datelor și notațiilor 94
8.1.2 Variabile de decizie și auxiliare 97
8.1.3 Restricții hard și soft 97
8.1.4 Funcția obiectiv 99
8.2. Arbore de decizie 101
8.2.1 Setul de date 101
8.2.2 Criteriile de separare 101
8.2.3 Bagging și vot 102
8.2.4 Aplicare Random Forest Classifier 103

CAPITOLUL 9. PROIECTAREA APLICAȚIEI ȘI A BAZEI DE DATE 105
9.1. Modelul arhitectural ales 105
9.2. Flowchart aplicație - administrator 112
9.3. Flowchart aplicație – cadru didactic 113
9.4. Flowchart aplicație director de departament 114
9.5. Alegerea tipului de bază de date 115
9.5.1 Proiectarea principală a bazei de date 116
9.5.2 Optimizări în urma dezvoltării 118
9.6. Inserare automatizată a informațiilor în baza de date 120
9.6.1 Inserări simple 120
9.6.2 Inserări complexe 121

CAPITOLUL 10. ANALIZA ALGORITMILOR DE GENERARE AUTOMATĂ 127
10.1. Date inițiale și rezultate dorite 127
10.2. Validarea restricțiilor în procesul de planificare a activităților didactice 128
10.3. Algoritm Backtracking 132
10.3.1 Logica algoritmului Backtracking aplicată planificării activităților didactice 132
10.3.2 Implementarea algoritmului 134
10.4. Algoritm Genetic 139
10.4.1 Logica algoritmului Genetic aplicată planificării activităților didactice 139
10.4.2 Implementarea algoritmului 140
10.5. Algoritm Machine Learning (ML) 148
10.5.1 Extragerea datelor – 10 ani 149
10.5.2 Prelucrarea datelor 151
10.5.3 Antrenarea ML 154
10.5.4 Rezolvarea conflictelor 155

CAPITOLUL 11. PROIECTAREA INTERFEȚEI GRAFICE 157
11.1. Autentificare/Login 158
11.2. Home Page 160
11.3. Contul meu 161
11.4. Cadre didactice 163
11.5. Management cadre didactice 164
11.6. Săli 166
11.7. Management săli 168
11.8. Planuri de învățământ 169
11.9. Utilizatori 176
11.10. Preferințe cadre didactice 177
11.11. Alocarea profesorilor pe activități cumulat cu numărul de ore alocat fiecărei discipline 179
11.12. Generarea planificării activități didactice 182

CAPITOLUL 12. ANALIZA ȘI TESTAREA APLICAȚIEI 183
12.1. Alegerea specializărilor pentru observarea rezultatelor 183
12.2. Algoritmul Backtracking 186
12.3. Algoritmul Genetic 189
12.4. Machine Learning 193
12.5. Dificultăți întâmpinate 197
12.6. Comparația celor trei algoritmi 198
CONCLUZII 201
CONTRIBUȚII PROPRII 203
DIRECȚII VIITOARE DE CERCETARE 204
BIBLIOGRAFIE 205
LISTĂ FIGURI 222
LISTĂ TABELE 227
ANEXE 228

INTRODUCERE

În contextul actual al învățământului, gestionarea eficientă a resurselor și a timpului a devenit o provocare esențială, având în vedere volumul mare de documente ce susțin procesul educațional. Sistemele de învățământ moderne, orientate tot mai mult spre digitalizare, necesită soluții avansate pentru planificarea activităților didactice, care să răspundă atât nevoilor instituțiilor, cât și preferințelor cadrelor didactice.

Teza de doctorat își propune să susțină digitalizarea procesului de management a instituțiilor de învățământ superior prin cercetarea unor modele matematice ce susțin algoritmii de planificarea automată a activităților didactice, reducând semnificativ timpul de realizare a acestuia.

În ultimii ani, numărul studenților din învățământul superior românesc a înregistrat fluctuații semnificative. Conform datelor Institutului Național de Statistică (INS), *"în anul universitar 2021-2022, în România erau înmatriculați aproximativ 415.000 studenți la programele de studii universitare de licență, urmând ca în anul universitar 2022-2023 numărul studenților la programul de licență să scadă cu 4%*" (Cononenco, 2024). Fluctuația studenților de la an la an aduce provocări la nivelul planificării și alocării eficiente a resurselor.

STRUCTURA ȘI CONȚINUTUL TEZEI DE DOCTORAT

Teza de doctorat se structurează în 12 capitole, susținută de un număr de 183 de figuri, 19 tabele și 6 anexe.

Partea I: Stadiul actual privind managementul documentelor dintr-o universitate

În primul capitol, **Sistemul de învățământ superior și fluxurile de documente**, se prezintă o imagine de ansamblu asupra organizării sistemului de învățământ superior românesc, inclusiv privind structura internă a unei universități în contextul integrării în Spațiul European al Învățământului Superior (SEHEA) și al implementării Procesului Bologna.

În al doilea capitol, **Studiul aplicațiilor on-line existente pentru managementul documentelor/activităților didactice din universitate**, s-a evidențiat conceptul de digitalizare a procesului de management al documentelor dintr-o universitate, oferindu-se o clasificare a tipurilor de aplicații utilizate în prezent. De asemenea, s-au analizat în detaliu patru aplicații pentru managementul activităților și 7 aplicații pentru generarea automată a planificării activităților didactice.

În al treilea capitol, **Algoritmi utilizați pentru managementul documentelor într-o universitate**, se prezintă în detaliu patru tipuri de algoritmi: pentru căutare, de clusterizare, de clasificare și de criptare, care oferă soluții în implementarea diferitelor probleme din fluxul de documente. Fiecare categorie de algoritm vine cu detalierea a cel puțin doi algoritmi dintre cei mai utilizați în rezolvarea problemelor.

În al patrulea capitol, **Algoritmi utilizați în managementul activităților didactice**, se prezintă trei algoritmi complecși de rezolvare a problemelor de planificare automată a activităților didactice: backtracking, genetic și algoritm AI (Random forest Classifier). Pentru fiecare algoritm se analizează principiul de funcționare, metoda de implementare și un pseudocod pentru a înțelege cât mai ușor etapele algoritmului.

Partea a II-a: Contribuții proprii privind modelarea, optimizarea și dezvoltarea unei platforme software dedicate

În capitolul al cincilea, **Obiective și metodologia de cercetare**, se prezintă obiectivul general al prezentei lucrări, ce constă în optimizarea proceselor de planificare a activităților didactice și dezvoltarea unei platforme informatice integrate.

În capitolul al șaselea, **Analiza soluțiilor existente și a metodelor de planificare a activităților didactice**, s-au realizat două analize comparative detaliate a aplicațiilor existente și a algoritmilor utilizați în cercetările existente. Pe baza observațiilor din analizele extrase s-au definit problemele generare pentru planificarea activităților didactice și s-au conturat provocările generării unei planificări optime.

În capitolul al șaptelea, **Analiza cerințelor utilizatorilor și stabilirea cerințelor pentru dezvoltarea aplicației**, s-au analizat nevoile clienților, pe baza cărora s-au definit restricțiile hard și soft ce trebuie îndeplinite de aplicație. Înțelegerea aprofundată a nevoilor clienților stă la baza unei dezvoltări de succes a platformei informatice integrate pentru planificarea activităților didactice.

În capitolul al optulea, **Model matematic pentru managementul activităților didactice**, se prezintă două modele matematice ce susțin algoritmii aleși pentru dezvoltare.

În capitolul al nouă-lea, **Proiectarea aplicației și a bazei de date**, s-a analizat modelul arhitectural care este avantajos de aplicat pentru aplicația dezvoltată și s-a implementat, folosind containere dezvoltate în framework-ul Docker Compose.

În capitolul al zecelea, **Analiza algoritmilor de generare automată**, s-au prezentat etapele logice și structura codului implementat pentru cei trei algoritmi abordați: Bactraking, Genetic și Random Forest. Fiecare algoritm prezintă avantaje și dezavantaje în dezvoltare.

În capitolul al unsprezece-lea, **Proiectarea interfeței grafice**, se prezintă fiecare pagină dezvoltată în aplicație.

În ultimul capitolul, Capitolul 12, **Analiza și testarea aplicației,** s-au prezentat rezultatele obținute după implementarea algoritmilor.

CAPITOLUL 1
SISTEMUL DE ÎNVĂȚĂMÂNT SUPERIOR ȘI FLUXURILE DE DOCUMENTE

Educația academică ”*are efecte pe termen lung asupra dezvoltării resurselor umane și joacă un rol esențial în răspândirea cunoașterii*” (Socoliuc, Dolghi & Ciubotariu, 2023).

”*De-a lungul timpului, învățământul superior a trecut prin numeroase reforme, fie pentru a satisface nevoile sociale și economice, fie pentru a spori calitatea educației superioare și a facilita adaptarea la modificările din societate. Aceste reforme s-au axat pe diverse arii, cum ar fi conținutul cursurilor, tehnicile de predare, accesibilitatea educației, nivelul calității instituțiilor și adaptabilitatea la cerințele pieței muncii*” (Socoliuc, 2023).

În domeniul educației terțiare au avut loc ”*schimbări semnificative generate de internaționalizarea pieței muncii, mobilitătea studenților şi a cadrelor didactice, creșterea competitivității pentru obținerea unui loc de muncă adecvat pregătirii și garantarea dezvoltării libere a personalității tinerilor*” (Bleikli, 2000; Solbrekke, Karseth, 2006).

1.1 Dinamica Învățământului Superior: Evoluția Numărului de Studenți

Evoluția tehnologiei a provocat o creștere a numărului de studenți care doreau să urmeze studii superioare la nivel european. Conform unui studiu realizat de Eurostat în anul 2022, în Europa studiază aproximativ 18,8 milioane de studenți, înscriși la cele trei cicluri de învățământ superior: licență (59%), masterat (36%) și doctorat (5%) (Tertiary education statistics, 2024).

Conform statisticilor realizate de Eurostat în perioada 2015-2022, ”se înregistrează o creștere substanțială a numărului de studenți (Fig. 1.2) înscriși la studii superioare în ultimii ani” (cu excepția anului 2022 când s-a observat o mică scădere) ceea ce a dus la o nevoie tot mai mare de eficientizare a proceselor administrative ale universităților.

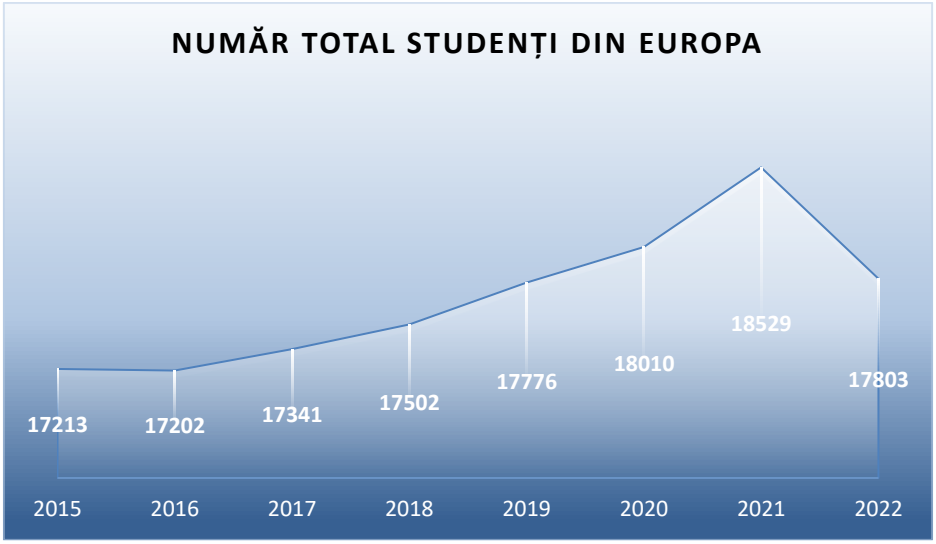


Fig. 1. 1 Numărul de studenți în perioada 2015-2022 în UE în milioane (Tertiary education statistics, 2022)

1.2 Organizarea sistemului de învățământ superior

"Învățământul superior *este organizat exclusiv de furnizori de educație și instituții de învățământ superior care dețin autorizarea de funcționare provizorie sau acreditarea, potrivit prevederilor legale*", conform art. 7, capitolul 2 al Legii Învățământului Superior 199/2023.

În cadrul sistemului educațional românesc există 89 de instituții de învățământ superior, publice și private, amplasate în 23 de localități urbane:

- ✓ *"52 de instituții acreditate de învățământ superior de stat (46 civile și 7 militare)*";
- ✓ *"37 instituții de învățământ superior private (34 acreditate și 3 autorizate provizoriu) "* (Raport anual privind Metarankingul Național, 2023).

1.6 Documente necesare realizării planificării activităților didactice

Dezvoltarea unei planificări a activităților didactice în cadrul unei instituții din sistemul de învățământ superior din România necesita o serie de documente care să furnizeze informații necesare. Principalele documente din care se extrag informații necesare pentru implementarea unei planificări a activităților didactice sunt:

1. Planul de învățământ
2. Formațiile de studii
3. Statul de funcții
4. Situația sălilor de clasă
5. Fișe de disciplină



Fig.1. 2 Documente necesare unui planificări academice

1.7 Relații de precedență între documentele implicate în realizarea planificării activităților didactice

În orice instituție publică există proceduri operaționale pentru circuitul logic al documentelor necesare desfășurării procesului didactic. Fiecare facultate are diferite programe de studii pe care trebuie să le gestioneze intern. Astfel, pentru planificarea activităților didactice unei facultății (pentru toți anii și toate programele de studii), relațiile de precedență dintre documentele necesare sunt foarte importante.

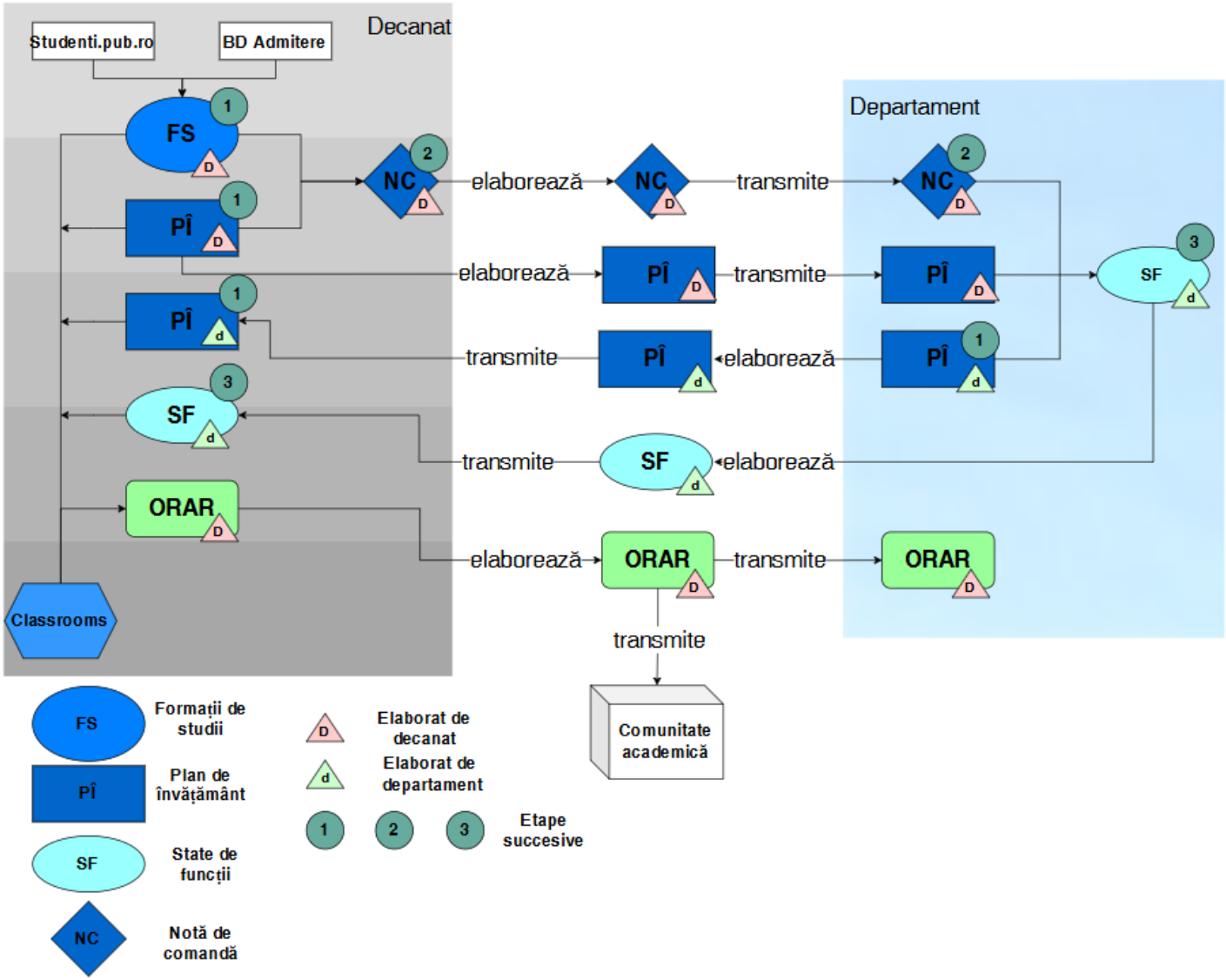


Fig.1. 3 Flux de documente pentru elaborarea planificării activităților didactice (Tunsoiu et al, 2025)

CAPITOLUL 2
STUDIUL APLICAȚIILOR ON-LINE EXISTENTE PENTRU MANAGEMENTUL DOCUMENTELOR DIN UNIVERSITATE

2.1 Introducere în managementul documentelor universitare

Conceptul de document a suferit modificări continue determinate, în principal, de evoluția tehnologiilor informaționale. *"Metodele tradiționale de gestionare a documentelor, caracterizate prin gestionarea fișierelor pe hârtie, nu sunt potrivite pentru mediul dinamic și digitalizat în care se dezvoltă educația modernă*" (Imeri, Memeti, Sadiki & Idrizi, 2024).

Piața mondială a sistemelor de gestionare a documentelor a fost evaluată la peste 7,52 miliarde USD în 2020 iar estimările indică atingerea valorii de 14,82 miliarde până în 2029 (Mondor, 2022). Asociația pentru inteligența managementului informațional (AIIM) afirmă că managementul documentelor, denumit și sistem de management al documentelor (DMS), este *"procesul de stocare, organizare și monitorizare a documentelor electronice într-un mod controlat și structurat, bazat pe fluxuri predefinite*" (Imeri, 2024).

2.2 Tipuri de aplicații utilizate în managementul documentelor universitare

Valorificarea informațiilor de la diferitele sisteme existente pentru managementul documentelor, conduce la dezvoltarea unor aplicații software care să înlocuiască vechile proceduri de lucru. Variantele disponibile, Fig. 2.1:

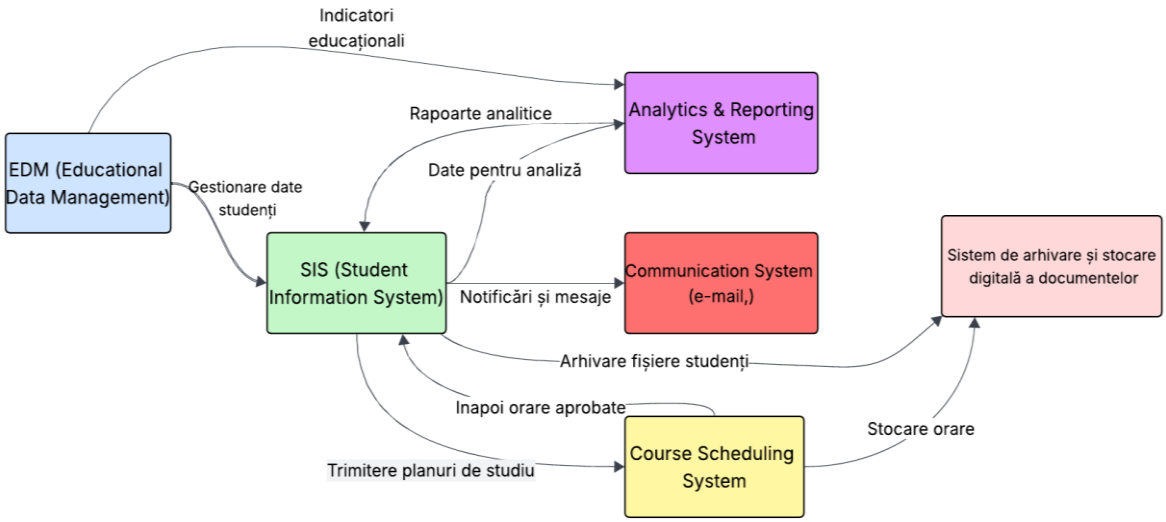


Fig. 2. 1 Tipuri de aplicații utilizate

CAPITOLUL 3
ALGORITMI UTILIZAȚI PENTRU MANAGEMENTUL DOCUMENTELOR ÎNTR-O UNIVERSITATE

În domeniul educației, obiectivele importante includ trecerea la gestionarea electronică a documentelor, îmbunătățirea calității proceselor educaționale și a personalului academic, dezvoltarea de noi abordări ale stocării sigure și procesării rapide a datelor. Pentru rezolvarea problemelor este necesară ”*crearea și implementarea sistemului electronic de gestionare a documentelor, ținând seama de particularitățile domeniului*” (Krasnyansk, Ostroukh, Karpushkin & Obukhov, 2016).

Un management eficient al documentelor trebuie să aibă în vedere aspecte precum digitalizarea documentelor, stocarea și organizarea, căutarea, securitatea datelor, arhivarea, dar și automatizarea proceselor (Struyf, Hubert & Rousseeuw, 1997).

Pe baza acestora, s-au dezvoltat diverși algoritmi care ajută la implementarea mult mai ușoară a aplicațiilor software:

1. Algoritmi pentru căutare, cum sunt TF-IDF (Term Frequency-Inverse Document Frequency) și BM25 (Best Matching 25), care atribuie ponderi cuvintelor cheie din documente în funcție de frecvența și importanța lor;
2. Algoritmi de clusterizare, cum este K-Means sau K-Medoids, ce pot fi utilizați pentru a grupa documentele similare în clustere, facilitând organizarea și accesul la documente asemănătoare;
3. Algoritmi de clasificare, cum sunt arborii de decizie, mașinile cu vectori suport (SVM) sau Naïve Bayes, ce pot fi utilizați pentru a clasifica documentele în categorii sau tematici bazate pe conținut;
4. Algoritmi de criptare și securitate, cum este AES (Advanced Encryption Standard), pot fi utilizați pentru a asigura securitatea datelor sensibile din documente, permițând accesul doar persoanelor autorizate.

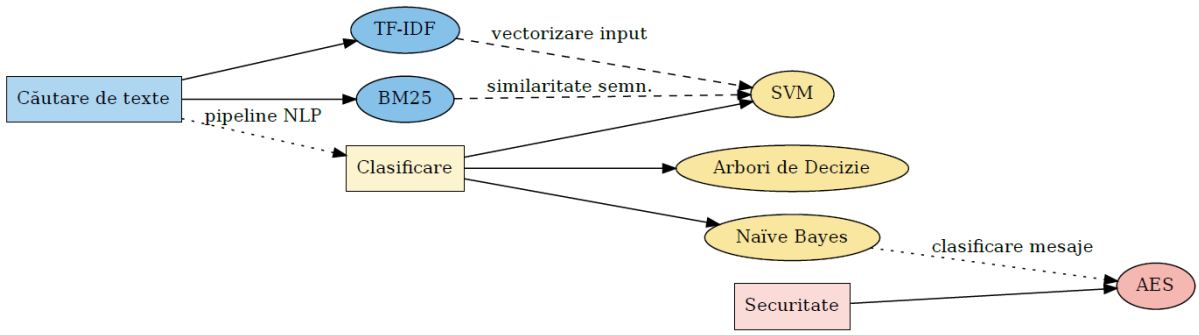


Fig. 3. 1 Ordinea folosirii algoritmilor implementați

CAPITOLUL 4
ALGORITMI UTILIZAȚI ÎN MANAGEMENTUL ACTIVITĂȚILOR DIDACTICE

4.1 Algoritmul Backtracking

Backtracking implică găsirea unei soluții în mod incremental, încercând diferite opțiuni și anulându-le dacă duc la o soluție nevalidă . Este folosit în mod obișnuit în situațiile în care trebuie să se exploreze mai multe posibilități pentru a rezolva o problemă, cum ar fi căutarea unei căi într-un labirint, rezolvarea de puzzle-uri precum Sudoku sau rezolvarea unei planificări a activităților. Când se ajunge la o soluție nevalidă, algoritmul se întoarce la punctul de decizie anterior și explorează o cale diferită până când se găsește o soluție validă sau se epuizează toate posibilitățile (WsCube Tech, 2024).

4.2. Algoritmul Genetic

Algoritmul genetic (GA) este un algoritm de optimizare care este inspirat din selecția naturală. Este un algoritm de căutare bazat pe populație, care utilizează conceptul de supraviețuire a celui mai potrivit (Katoch, Chauhan & Kumar, 2021).

4.2.2 Implementare și optimizări

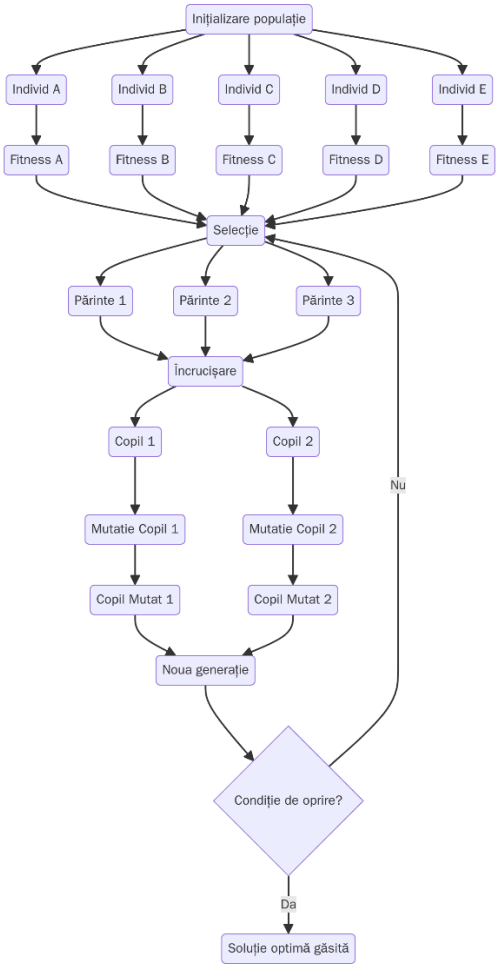


Fig. 4. 1 Etapele implementării GA

4.3 Machine Learning pentru planificarea activităților didactice

Învățarea automată reprezintă o paradigmă fundamentală a inteligenței artificiale, care presupune capacitatea unui sistem de a extrage cunoștințe și modele din date istorice, în scopul îmbunătățirii performanței în sarcini viitoare. Esența acestui domeniu constă în dezvoltarea și aplicarea unor metode capabile să generalizeze pe baza experienței anterioare, fără a necesita intervenție umană explicită. (Burkart & Huber, 2021). **Machine Learning** este un domeniu al "*inteligenței artificiale ce permite algoritmilor să învețe din date inițiale și să realizeze predicții fără a fi explicit programate*" (Mahesh, 2020). Algoritmii ML sunt împărțiți în mai multe categorii, în funcție de tipul de învățare și de natura problemei pe care o rezolvă (Mahesh, 2020).

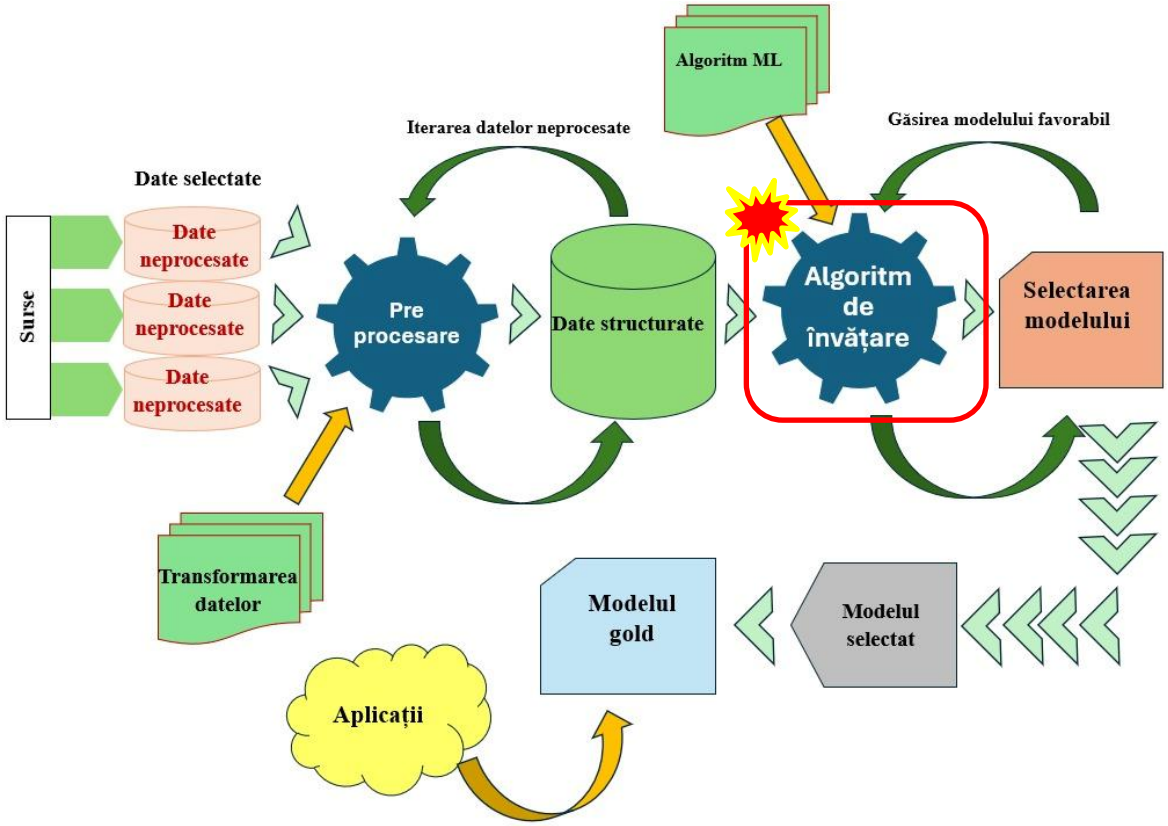


Fig. 4. 2 Procesul învățării automate (adaptat după Ojha, 2019)

CAPITOLUL 5
OBIECTIVE ȘI METODOLOGIA DE CERCETARE

Studiul privind modelarea și optimizarea procesului de management al activităților didactice, dar și dezvoltarea unei aplicații pentru managementul documentelor și a planificării activităților academice dintr-o instituție de învățământ superior are la bază, în principal, cercetarea, dezvoltarea și testarea unor algoritmi de generarea a unei soluții optime ce satisface cerințele și restricțiilor hard și soft, venind în sprijinul personalului implicat în realizarea manuală, semestrială a planificării activităților didactice.

În partea a II-a a tezei de doctorat, Contribuții proprii privind modelarea, optimizarea și dezvoltarea unei platforme software dedicate, se vor valida obiectivele propuse în acest capitol.

CAPITOLUL 6
ANALIZA SOLUȚIILOR EXISTENTE ȘI A METODELOR DE PLANIFICARE A ACTIVITĂȚILOR DIDACTICE

Generarea automată a planificării activităților didactice reprezintă un element important în modernizarea managementului educațional, având în vedere creșterea constantă a volumului de date și a complexității restricțiilor. Acest capitol oferă o sinteză aplicativă și teoretică asupra soluțiilor existente, a algoritmilor utilizați și a dificultăților asociate problemei, cu scopul de a fundamenta direcțiile ulterioare de cercetare.

De asemenea, pe baza celor trei algoritmi analizați se va realiza o analiză multicriterială cu ajutorul căruia se poate observa ordinea, în funcție de rezultatele obținute prin ponderizarea criteriilor stabilite.

6. 1. Analiză comparativă a aplicațiilor existente pentru managementul activităților didactice

În cadrul acestui capitol, se realizează o analiză comparativă a aplicațiilor existente pentru a putea identifica modul în care produsul satisface nevoile clienților. Se analizează pentru cele 6 aplicații prezentate în capitolul 2 următoarele caracteristici: interfața grafică, personalizare, generare automată, gestionarea resurselor, rapoarte, export, suport și documentație și buget, prezentate în Tabelul 6.1.

În urma analizei celor 17 articole științifice în care s-au realizat diferite încercări de a identifica cele mai bune soluții pentru planificarea automată a activităților didactice se poate concluziona că nu există un algoritm universal care să răspundă în totalitate cerințelor unei facultăți cu o anumită dimensiune, diferite tipuri de restricții și planuri de învățământ diferite de la an la an. S-a putut observa că diferite combinații de algoritmi (HGA, ILS+HC) pot obține performanțe mai bune față de metodele individuale.

Pentru probleme mari se recomandă utilizarea metaeuristicilor probabilistice (SA, TS, MOSA), doar că necesită diferite reglaje la parametrii de generare, iar pentru probleme mici se recomandă metodele deterministrice (HC, MIP) ce oferă rezultate rapide și eficiente.

Toți algoritmii utilizați în lucrările analizate sunt algoritmi pentru probleme de optimizare, clasificați în 6 ramuri: căutări locale, metaeuristici, multi-obiectiv, evolutivi, metode exacte și hibrizi (Fig. 2.22).

6.3 Problemele generării planificării activităților didactice

6.3.1 Definiția problemei și restricții

Problema planificării activităților didactice constă în organizarea activităților (de exemplu, cursuri, seminarii) astfel încât să se optimizeze utilizarea resurselor disponibile și să se respecte diverse restricții. Aceste constrângeri se împart în două categorii: constrângeri hard și constrângeri soft (Burke & Petrovic, 2002).

O planificare a activităților didactice validă trebuie să respecte următoarele **restricții tari (hard constraints):**

- ❖ Activitatea unei discipline trebuie programată și atribuită unor slot-uri de timp distincte
- ❖ Toate cursurile din același curriculum predate de același profesor trebuie programate în perioade diferite (Abdullah, Turabieh, McCollum & McMullan, 2012)
- ❖ Un profesor nu poate preda două cursuri simultan (Chen, 2021)

Necesitățile care pot fi omise în cadrul unei planificări a activităților didactice valide se numesc restricții soft și pot fi următoarele:

- ✓ Distribuirea uniformă a cursurilor pentru profesori și studenți pe parcursul săptămânii.
- ✓ Evitarea programării cursurilor la ore nepotrivite sau nepreferate, cum ar fi începutul zilei sau sfârșitul zilei (Chen, 2021).
- ✓ Evitarea alocării orelor în intervalul pauzei de masă (Chen, 2021)

6.5 Analiză Multicriterială

„Analiza Multicriterială (AMC) descrie orice abordare structurată utilizată pentru a determina preferințele generale dintre mai multe opțiuni alternative, care opțiuni conduc la îndeplinirea unui anumit număr de obiective diferite” (Roman, 2012). Această metoda este folosită pentru a ordona opțiunile disponibile pe baza unui set prestabilit de criterii și ponderi care reflectă importanța relativă a fiecăruia. ” *AMC permite luarea în considerare simultană a mai multor criterii în situații complexe*” (Roman, 2012). Pentru AMC a algoritmilor selectați se vor utiliza cinci criterii pentru elaborarea analizei.

6.5.3 Matricea de performanță

În cadrul matricei de performanță, Tabelul 6.5 se regăsesc toate valorile criteriilor pentru fiecare alternativă studiată. Astfel, pe fiecare rând din cadrul matricei de performanță se regăsește alternativa, iar pe coloane performanța fiecărei etape în funcție de criteriu. În tabelul x este prezentată matricea de performanță pentru analiza multicriterială a celor trei algoritmi analizați.

Tabel 6. 1 Matricea de performanță

Alternativă	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**
Backtracking	9.0	95	1	2	4	**10.56**
Algoritm Genetic	3.5	85	4	4	3	**24.29**
Random Forest	5.0	75	5	3	2	**15.00**

6.5.4 Normalizarea

Normalizarea este metoda prin care sunt aduse la o scară comună toate valorile criteriilor oferite în tabelul 6.7, care sunt exprimate în diferite unități de măsură (Triantaphyllou, 2000). În analiza multicriterială, normalizarea este un pas esențial care permite compararea alternativelor exprimate în unități diferite. Aceasta transformă valorile brute ale criteriilor într-o scală comună, de obicei [0, 1], menținând relațiile de proporționalitate (Triantaphyllou, 2000). Unitățile de măsură *"sunt transformate folosind o funcție valorică sau o procedură de standardizare, urmând ca punctajele obținute să își piardă dimensiunea și unitatea de măsură*" (Roman, 2012).

Pentru normalizarea valorilor se folosește metoda Min-Max. Această metodă prezintă pentru fiecare tip de normalizare (maximizare sau minimizare) câte o formulă. Astfel, pentru criteriul C1 și C5 aplicăm minimizarea, iar pentru criteriile C2, C3, C4, C6 aplicăm maximizare după următoarele formule:

Pentru maximizare: $r_{ij}=\frac{x_{ij}-x_j^{min}}{x_j^{max}-x_j^{min}}$ (6.1) Pentru minimizare: $r_{ij}=\frac{x_j^{max}-x_{ij}}{x_j^{max}-x_j^{min}}$ (6.2)

Unde:

- x_{ij} = valoarea alternativului *i* pentru criteriul *j*
- r_{ij} = valoarea normalizată
- x_j^{max}, x_j^{min}= valorile maxime și minime pentru criteriul respectiv

În cazul criteriului C1, timp de execuție, valoarea minima este de 3,5 iar valoarea maximă este de 9.0. Dacă se aplică formula 6.1 rezultă:

$r_{11} = \frac{9-9}{9-3,5} = 0$ $r_{12} = \frac{9-3,5}{9-3,5} = 1$ $r_{13} = \frac{9-5}{9-3,5} = 0,727$ (6.3)

Tabel 6. 2 Normalizarea ponderilor

Alternativă	**C1**	**C2**	**C3**	**C4**	**C5**	**C6**
Backtracking	0	1	0	0	0	0
Algoritm Genetic	0,15	0,5	0,109	1	0,5	1
Random Forest	0,727	0	1	0,5	1	0,312

6.5.5 Ierarhizarea alternativelor

În prima etapă pentru ierarhizarea alternativelor se înmulțesc ponderile și valoarea fiecărui criteriu după normalizare. Astfel, ierarhizarea etapelor se face prin însumarea ponderilor aferente rezultate după prima etapă. Pentru criteriul C2 ponderea este de 0,25, conform tabelului x, iar valorile pentru ierarhizare sunt:

I_{21} = 1 * 0,25 = 0,25 I_{22} =0,5 * 0,25 = 0,125 I_{23} = 0 * 0,25 = 0 (6.4)

Astfel, în tabelul 6.8 se pot observa valorile rezultate în urma calculelor efectuate:

Tabel 6. 3 Ierarhizarea alternativelor

Alternativă	C1	C2	C3	C4	C5	C6	Total
Backtracking	0	0,25	0	0	0	0	0,25
Algoritm Genetic	0,15	0,125	0,1125	0,15	0,05	0,2	**0,7875**
Random Forest	0,109	0	0,15	0,075	0,1	0,0624	0,4964

În urma realizării analizei multicriteriale cu ajutorul metodei Min-Max, algoritmul genetic obține cel mai mare scor total (0.7875), ceea ce confirmă echilibrul său între eficiență, calitate și adaptabilitate. Deși Backtracking oferă soluții excelente în ceea ce privește calitatea, limitele sale de execuție și scalabilitate îl dezavantajează în aplicații practice. Random Forest, în schimb, obține un scor mediu, remarcându-se mai ales la capitolele de scalabilitate și ușurință în implementare.

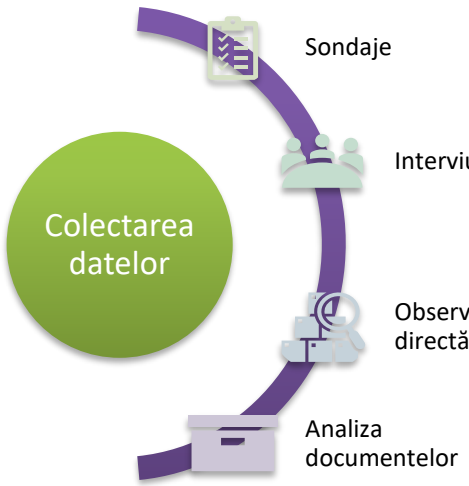
CAPITOLUL 7
ANALIZA CERINȚELOR UTILIZATORILOR ȘI STABILIREA CERINȚELOR PENTRU DEZVOLTAREA APLICAȚIEI

Dezvoltarea unei aplicații personalizate presupune configurarea operațiunilor în funcție de nevoile clienților sau ale utilizatorilor. Încadrarea preferințelor și nevoilor utilizatorilor în dezvoltarea arhitecturală a aplicației este importantă deoarece produsul final este evaluat pe baza ușurinței de utilizare și a eficienței rezultatelor obținute (Verma, Thompson & Louviere, 1999)

7. 3. Metode de colectare a datelor

Modalitățile de culegere a datelor sunt importante, deoarece modul în care informațiile colectate sunt utilizate poate genera abordarea analitică folosită de cercetător în găsirea celor mai bune soluții pentru nevoile identificate. Așadar, sunt folosite următoarele metode, Fig. 7.2:

1. Sondaje sau chestionare cu diferite tipuri de întrebări folosite pentru documentarea percepțiilor, atitudinilor, credințelor sau cunoștințelor într-un eșantion de indivizi clar definit;
2. Interviuri - folosite pentru a culege informații de la indivizi față în față. Interviurile sunt ideale atunci când sunt folosite pentru a documenta relatările participanților, percepțiile sau poveștile despre atitudinile față de și răspunsurile la anumite situații sau fenomene. Datele interviurilor sunt folosite pentru a genera structura clara a aplicației și elementele absolut necesare pentru buna funcționare (Paradis et al. 2016).



3. Observația directă - folosită pentru a observa procesul de realizare actual al unui orar și tehnicile utilizate de persoanele implicate în generarea sa pentru a găsi cele

mai fezabile soluții care să fie accesibile și pentru studenți dar și pentru cadrele didactice.

4. Analiza documentelor existente pentru studierea regulamentelor și procedurilor care pot influența cerințele aplicației

Fig. 7. 1 Metodele de colectare a datelor

5. Grupuri de lucru sunt *"folosite pentru a culege informații în cadrul grupului, fie prin întrebări predeterminate de interviu pe care moderatorii le adresează participanților la rândul său,, pentru a stimula conversațiile de grup*" (Paradis et al. 2016).

7. 4. Colectarea datelor

În cadrul tezei de doctorat au fost folosite primele patru metode de colectare a datelor pentru a culege toate informațiile necesare de dezvoltare a aplicației. Metoda principală de determinare a cerințelor critice a fost interviul realizat cu decanul facultății, persoana direct responsabilă de organizarea cadrelor didactice și a studenților.

7.4.1 Interviul

Scopul interviului a fost de stabilire a cerințelor esențiale pentru dezvoltarea aplicației, identificarea problemelor actuale în procesul de realizare manuală a planificării activităților didactice, înțelegerea regulilor și constrângerilor ce trebuie respectate și analiza posibilităților de optimizare și automatizare a procesului.

Interviul a fost structurat în trei secțiuni, fiecare abordând un aspect important al aplicației de planificare a activităților didactice.

7.4.2 Observația directă

În realizarea manuală a planificării activităților didactice se implică un număr de 4 cadre didactice și un prodecan responsabil. Cadrele didactice realizează o schiță pentru fiecare departament din cadrul Facultății de Inginerie Industrială și Robotică. Schițele sunt verificate și alocate în săli specifice tipului de activitate de prodecan. Validarea finală a planificării se realizează după ce toate cadrele didactice implicate în programul de specializare își dau acordul asupra orelor alocate, adică verifică că nu există suprapuneri pe orarul personal.

7.4.3 Analiza documentelor existente

În procesul de dezvoltare a aplicației de management al documentelor și al planificării activităților didactice, **analiza documentelor existente** joacă un rol esențial în înțelegerea reglementărilor academice, a normelor administrative ce trebuie respectate. Această etapă permite alinierea funcționalităților aplicației la cadrul legal și operațional al universității, al Ministerului Educației și Cercetării și al ARACIS.

S-au analizat o serie de documente relevante incluzând regulamente interne ale universității legate de organizarea anului universitar și norme ale cadrelor didactice, dar și planurile de învățământ ce oferă informații relevante despre denumirea disciplinelor, activitățile și numărul de ore asociate fiecărei discipline. Următoarele documente analizate au fost Legea învățământului superior și standardele ARACIS, de unde au rezultat restricții privind numărul de studenți/grupă sau semigrupă.

7.4.4 Sondaje/Formular online

O altă etapă din cadrul analizei cerințelor de dezvoltare a aplicației a fost realizarea unui formular on-line implementat în Google Form cu scopul de a analiza opiniile cadrelor didactice, directorilor de departament și studenților cu privire la planificarea activităților didactice și de a formula restricțiile hard și soft. Formularul a fost completat de un număr de 112 persoane: 69 de studenți, 37 de cadre didactice și 6 directori de departament.

Informațiile obținute prin cele patru metode de colectare a datelor vor sta la baza definirii cerințelor funcționale și a restricțiilor hard și soft. Pe baza acestora se va structura arhitectura software a aplicației, astfel încât să răspundă nevoilor reale ale facultății și să reducă timpul și efortul necesar pentru planificarea activităților didactice.

CAPITOLUL 8
MODEL MATEMATIC PENTRU MANAGEMENTUL ACTIVITĂȚILOR DIDACTICE

Vechea problemă de timetabling în care se alocă activitățile disciplinelor din planul de învățământ profesorilor titulari și se găsesc intervale de timp și săli corespunzătoare, la care trebuie respectate diferite restricții impuse de facultatea are în spate diferite modele matematice ce au fost studiate. Problema are o formulare matematică standard: este o problemă de alocare cu resurse multiple, combinând colorarea de grafuri, programarea liniară întreagă (ILP - Programare Liniară cu Variabile Întregi/MIP - Programare Mixtă cu Variabile Întregi), programarea cu restricții (CP) și SAT (Even, Itai & Shamir, 1976).

8. 1. Model matematic ILP

8.1.1 Definirea datelor și notațiilor

În continuarea se vor prezenta seturile și datele de intrare pentru a putea definii funcția obiectiv.

Seturi de date

- D: setul zilelor (luni – vineri);
- Pentru fiecare zi d ∈ D_d, H_d = {1,…,md}: pozițiile din ziua d (pot fi inegale de la o zi la alta).
- T: setul sloturilor elementare de timp (5 zile x 5 sloturi/zi).

Notăm $T_d \subseteq T$ sloturile din ziua d ∈ D;

- R: setul sălilor;

Se definesc datele de intrare pe baza seturilor de date definite anterior:

Pentru a putea definii constrângerile impuse e necesar să se definească ecuații de mulțimi ce sunt folosite. Astfel, cu ajutorul datelor de intrare D, H_d, d_e și A_e se calculează:

- **Starturi valide**: mulțimea ce dă toate pozițiile de start posibile pentru evenimentul e, care pot fi alocate în totalitate în ziua d (Daskalaki, Birbas & Housos, 2004).
- **Sloturi acoperite de un start**: Acest set enumeră sloturile elementare (d,h') pe care le ocupă evenimentul e dacă începe la (d,h), adică este fix intervalul continuu de timp pentru activitatea respectivă (Daskalaki, Birbas & Housos, 2004).

8.1.3 Restricții hard și soft

Se definesc matematic **restricțiile hard** ce trebuie îndeplinite de model.

1. **Alocare unică** - fiecare eveniment e este planificat o singură dată:

$$\sum_{(d,h)\in s_e^{valid}} \sum_{r\in R} x_{e,d,h,r} = 1, \forall\, e \in \mathrm{E} \qquad (8.10)$$

2. Verificare **disponibilitate profesor și sală** și capacitatea sălii este respectată:

$$x_{e,d,h,r} \leq avail^P_{p(e),d,h} avail^R_{r,d,h}\, 1[enr_e \leq xcap_r] qual_{e,r}\,, \forall e,d,h,r \qquad (8.11)$$

Dacă profesorul este disponibil atunci valoarea este 1, dacă nu este 0.

$$avail^R_{r,d,h} - \text{disponibilitatea sălii r la ziua d, ora h.}$$

Dacă sala este liberă atunci valoarea este 1, dacă nu este 0.

$$1[enr_e \leq xcap_r] - \text{verifică dacă capacitatea sălii este adecvată numărului de studenți}$$

Dacă studenții încap în sală este 1, altfel este 0

$$qual_{e,r} - \text{verifică dacă dotările sălii sunt conforme activității}$$

8.1.5 Funcția obiectiv

Modelul ILP respectă în totalitate restricțiile hard care îți garantează că soluția este fezabilă, dar pentru alegerea celei mai bune soluții se adaugă funcția obiectiv sau criteriul de optimizare. Funcția obiectiv este o ecuație matematică ce calculează calitatea soluției în funcție de restricțiile puse.

8.2 Arbore de decizie

Al treilea algoritm selectat pentru planificarea activităților didactice este Random Forest Classiffier ce este susținut de o colecție de arbori de decizie ce sunt combinați prin tehnici de tip **bagging** și **vot majoritar** pentru a dau un model robust și precis.

Pe baza informațiilor din capitolul 3 (arbori de decizie: ID3 și C4.5) și capitolul 4 (Alegerea algoritmului ML), în continuare se va prezenta modelul matematic ce susține Random Forest Classiffier: arbore de decizie, criteriile de separare pentru un arbore, bootstrap aggregating (bagging) și votul.

8.2.1 Setul de date

Pentru construcția unui arbore de decizie se va porni de la setul de antrenament (Wang, & Chen, 2016) notat

$$D= \{(x_i,y_i) \mid i=1,\dots,n \},\ x_i \in R^p,\ y_i \in Y, \tag{25}$$

unde:

$x_i \in R^p$ este vectorul de caracteristici (features) asociat lui i, adică $x_i = (x_{i1}, x_{i2},\dots, x_{ip})$

$y_i \in Y$ este eticheta (label) pentru observația i, iar mulțimea Y cuprinde toate valorile posibile ale clasei (Wang, & Chen, 2016).

8.2.2 Criteriile de separare

Pentru fiecare nod există un subset de date $S \subseteq D$. Se alege o caracteristică x_i și un prag θ care împart instanțele în două subseturi (Wang, & Chen, 2016). Separarea subseturilor se face pentru a obține informații cât mai pure intr-o frunză a arborelui – adică să conțină o singură clasă. Măsura matematică a separării claselor este impuritatea ce se calculează cu indicele Gini și entropia (Breiman, 2001).

Indicele Gini

$$\text{Gini(S)} = 1 - \sum_{c \in Y} p_c^2 \ ,\ p_c = \frac{|\{(x,y) \in S\colon y = c\mid}{|S|}, \tag{27}$$

unde p_c este proporția instanțelor din setul S care aparțin clasei c.

$p_c = 1 \rightarrow Gini(S) = 0$, atunci când toate instanțele sunt din aceeași clasă (puritate perfectă)

Entropia

$$\text{Entropy(S)} = -\sum_{c \in Y} p_c \log p_c \ , \tag{28}$$

unde p_c este proporția instanțelor din setul S care aparțin clasei c.

Entropy(S) = 0 atunci când există o singură clasă

Impuritatea ”medie” după separare (Breiman, 2001):

$$I_{split}(S,j,\theta) = \frac{|S_L|}{|S|} I(S_L) + \frac{|S_R|}{|S|} I(S_R) \ , \tag{29}$$

unde S_R și S_L sunt cele două subseturi

$I(\cdot)$ poate fi indicele Gini sau Entropia

Se alege caracteristica și pragul care minimizează impuritatea după separare pentru a obține frunze cât mai omogene (Breiman, 2001):

$$\text{Gain}(S,j,\theta) = I(S) - I_{split}(S,j,\theta) \tag{30}$$

8.2.3 Bagging și vot

Ideea de bază din spatele bagging-ului este *"de a antrena mai multe modele folosind subseturi aleatorii ale datelor de antrenament și apoi de a combina predicțiile lor pentru a reduce varianța și a îmbunătăți acuratețea generală a modelului*" (Brown, 2023).

CAPITOLUL 9
PROIECTAREA APLICAȚIEI ȘI A BAZEI DE DATE

Planificarea activităților didactice pentru studenți, dar și pentru cadrele didactice, a ajuns o problemă mai mult decât complicată, chiar complexă. Fiecare planificare trebuie să includă o serie de informații interconectate: sala unde se susțin orele, disciplina predată, tipul orelor susținute (curs, laborator, seminar, proiect), cadrul didactic titular, intervalul orar necesar susținerii disciplinei, dar și ziua din săptămână.

Dezvoltarea microserviciilor necesită o echipă lărgită care să împartă sarcinile pe layerele de dezvoltare a aplicației, de exemplu, în cazul planificării activităților didactice, microserviciile pot fi: gestionarea utilizatorilor (login), gestionarea preferințelor pentru cadrele didactice, gestionarea disciplinelor și atribuirii cadrelor didactice la fiecare disciplină, planificatorul de activități, gestionare sălilor, notificări și interfața grafică. Pe un principiu asemănător se va implementa aplicația, fiecare element va avea componente separate pe partea de interfață grafică, iar backend-ul va fi scris în același limbaj de programare în care se vor aborda toate cerințele.

Am hotărât ca arhitectura software utilizată pentru dezvoltarea aplicației să fie una hibridă, combină arhitectura monolitică cu tehnologii moderne precum containerizarea bazei de date, a serviciului PHP și Python (Fig. 9.3). Argumentarea deciziei se bazează pe următoarele considerente:

- ✓ Separarea serviciilor în containere oferă procesare paralelă și calcul intensiv, dar și update-uri independente;
- ✓ Bazate pe necesitățile funcționale: python pentru dezvoltarea algoritmilor, PHP procesare date în backend și baza de date pentru structura complexă a datelor;
- ✓ De performanță și scalabilitate: procesele algoritmilor nu afectează accesarea interfeței și scalabilitate diferențiată pentru fiecare serviciu;
- ✓ Toleranța la erori – atunci când serviciul Python este picat nu afectează întreaga aplicație.

Pentru planificarea automată se va dezvolta o aplicație web, accesată în principal printr-un browser. Realizarea aplicației implică multiple niveluri (Fig. 9. 4) și componente care lucrează împreună pentru a oferi funcționalități și pentru a interacționa cu utilizatorii.

9.2. Flowchart aplicație - administrator

Administratorul aplicației este persoana responsabilă de datele introduse, necesare pentru automatizarea planificării activităților didactice. Activitățile acestuia, în cadrul aplicației, sunt strict

legate de introducerea datelor ce evidențiază **managementul cadrelor didactice și al activităților didactice ale acestora, managementul sălilor de predare, managementul formațiilor de studiu și a calendarului academic pe structura anului universitar**. Toate aceste informații se înregistrează în baza de date și sunt folosite în totalitate de algoritmul de generare automată a activităților didactice.

9.3 Flowchart aplicație – cadru didactic

Un cadru didactic are drepturi limitate de operare în aplicație, el putând interveni asupra planificării activităților didactice doar prin exprimarea unor intervale orare în care nu poate să susțină activități didactice cu studenții.

Conform Fig. 9. 10, fluxul de activitate pentru un cadru didactic în aplicație este structurat în mai mulți pași logici:

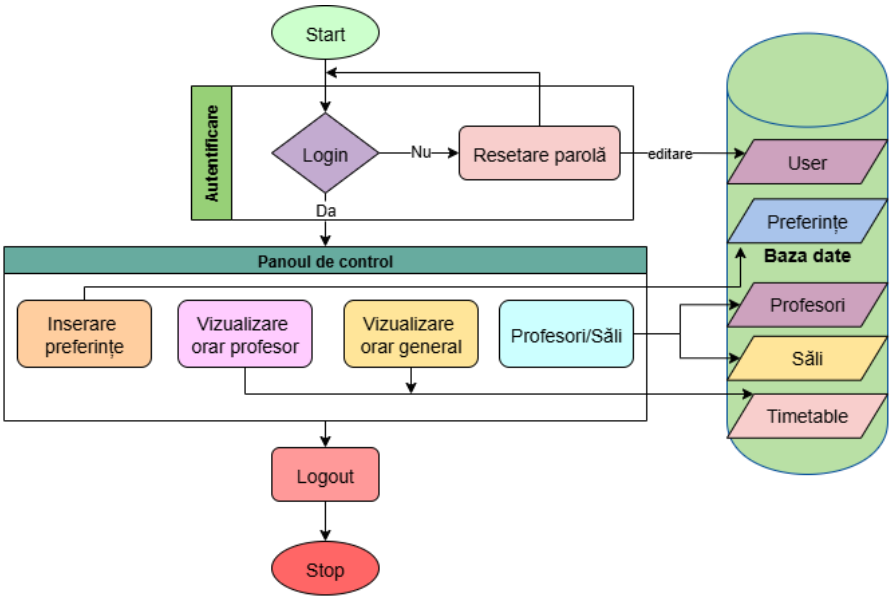


Fig. 9. 1 Flowchart aplicație profesor

9.4 Flowchart aplicație director de departament

Un alt rol pe care îl poate deține un utilizator este director de departament. Directorul de departament este și el cadrul didactic având aceleași drepturi în platformă, la care se adaugă, în panoul de control, două componente principale: gestiune plan de învățământ și alocare cadre didactice pe discipline. În cadrul modulului Gestiune plan de învățământ (Gestiune PÎ), utilizatorul poate să insereze sau să editeze planurile de învățământ existente, iar în cadrul celui de-al doilea modul, conform normelor din statele de funcții, utilizatorul alocă fiecărei discipline cadrele didactice titulare (Alocare profesori), acestea fiind, singurele modificări aduse structurii aplicației dezvoltate față de cele ale unui cadru didactic, Gestionare PÎ și Alocare profesori. Aceste funcționalități sunt parte a automatizării planificării activităților, astfel putându-se realiza modificări asupra planurilor de învățământ fără a fi necesară o nouă versiune. De asemenea, alocarea profesorilor reprezintă un beneficiu adus utilizatorului în atribuirea semi-automată a cadrelor didactice la disciplinele aferente.

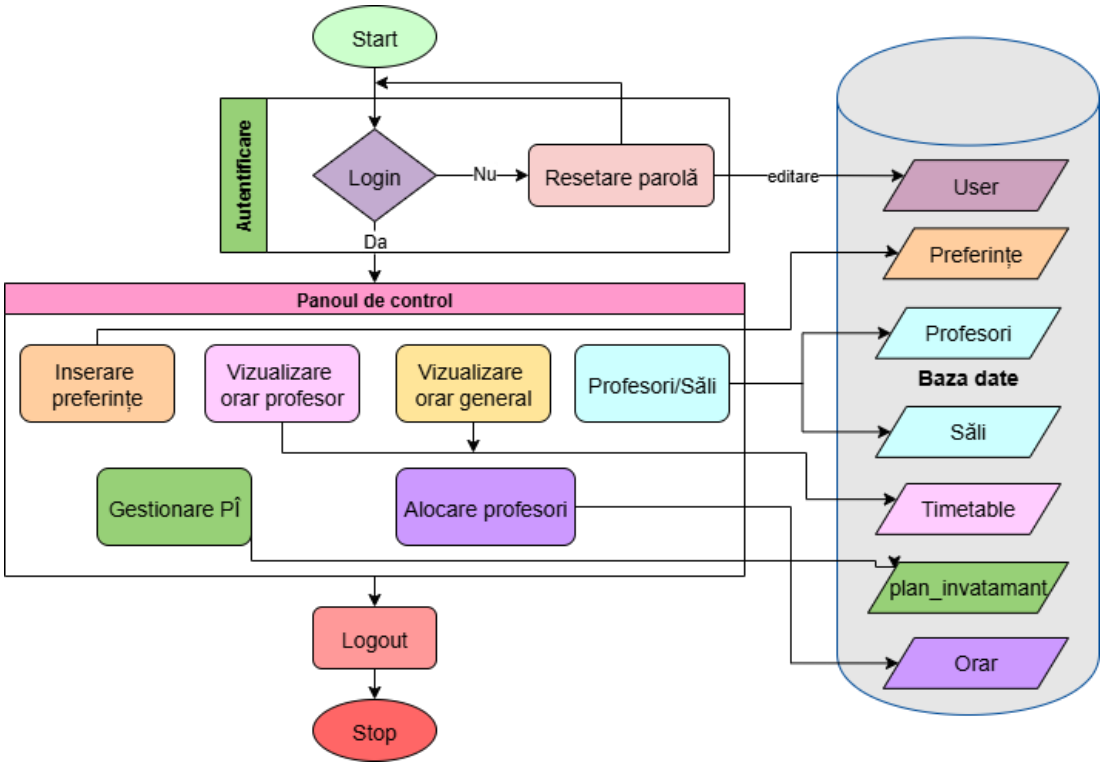


Fig. 9. 2 Flowchart aplicație director departament

9.5 Alegerea tipului de bază de date

Dezvoltarea unui algoritm pentru planificarea activităților diactice pornește de la proiectarea și popularea unor baze de date cu informațiile esențiale și relaționarea tabelelor pentru o gestionare eficientă a informațiilor. Prima etapă în procesul de proiectare o reprezintă identificarea entităților și a relațiilor dintre acestea (Scripcariu, 2011). Se va alege ca sistem de management al bazelor de date MariaDB.

9.5.1 Proiectarea principală a bazei de date

În urma analizei cerințelor aplicației, alegerea entităților reprezintă un element cheie pe baza căruia se identifică elementele ce au un rol important în dezvoltarea aplicației. Astfel, entitățile de bază pentru dezvoltarea generatorului de planificare a activităților sunt: utilizatori, sălile, cadrele didactice, planuri_învățământ și asocierea profesor-curs. Fiecare entitate a tabelelor din baza de date are atribute specifice descrise în Anexa 1.

9.5.2 Optimizări în urma dezvoltării

Dezvoltarea aplicației aduce noi modificări a structurii bazei de date. Aceste modificări provin de la automatizarea întregului proces de introducere a informațiilor necesare pentru planificarea activităților didactice. Astfel, prima modificare a fost la tabela rooms, devenită Sali_materii, în care s-a introdus Disciplina_id, sala_obligatorie și sala_pc, câmpuri necesare pentru

a evidenția dacă o sală este specifică unei anumite discipline. Necesitatea introducerii acestor câmpuri a apărut de la diferite discipline (de exemplu: Chimie, Fizica, Tehnologia Materialelor etc) ce se puteau susține doar în anumite săli, săli în care nu se puteau atribuii alte discipline. De asemenea, s-au evidențiat sălile cu calculatoare, unde se atribuiau laboratoarele sau proiectele ce nu se țineau în săli predefinite.

O altă modificare majoră a fost la tabela course_activity, care s-a transformat în plan_invatamant_specializare_abreviere_an_Inceput_anSfarsit_semestrul, care se șterge după finalizarea ciclului de învățare pentru programului de studii, pentru a nu încărca baza de date cu informații nefolositoare. Aceasta schimbare s-a produs datorită modificărilor anuale a planurilor de învățământ. De asemenea, se eficientizează procesul de introducere și citire a planurilor pentru o anumită specializare dintr-un anumit an.

CAPITOLUL 10
IMPLEMENTAREA ALGORITMILOR DE GENERARE AUTOMATĂ

Planificarea automată a activităților didactice reprezintă o provocare majoră pentru programatorii implicați în proiectarea algoritmilor, datorită numeroaselor cerințe personalizate pentru profesori și activitățile susținute. În continuare se vor analiza trei algoritmi care au caracteristici unice si avantaje în obținerea unor soluții optime și rezolvarea problemelor complexe de alocare conform restricțiilor impuse. Cei trei algoritmi analizați sunt: backtracking – una dintre metodele clasice pentru obținerea unor soluții valide, algoritm genetic – metodă folosită în inteligență artificială bazată pe procesul natural de selecție și machine learning – metodă actuală în plină dezvoltare ce implică un proces lung de prelucrare a datelor ce urmează a fi învățate automat și oferirea unor soluții noi pe baza datele învățate.

Implementarea algoritmilor se realizează pe baza cerințelor inițiale și a restricțiilor impuse.

10.1 Date inițiale și rezultate dorite

Majoritatea informațiilor necesare pentru dezvoltarea unui algoritm de planificare automată a activităților didactice se regăsesc în baza de date a aplicației. O parte sunt introduse manual din interfața grafică, iar o parte se regăsesc deja înregistrate.

Informațiile principale de care are nevoie un algoritm de planificare (Fig. 10. 1) a activităților didactice se regăsesc în:

1. planul de învățământ – discipline, activitățile fiecărei disciplină și numărul de ore alocat pentru predare;
2. statele de funcții – cadrele didactice asociate fiecărei activități ale unei discipline;
3. disponibilitatea cadrelor didactice;
4. numărul de grupe și semigrupe pentru fiecare specializare și an în parte;
5. sălile cu tipul de sală și capacitate.

Rezultatul dorit, obținut în urma rulării unei aplicații care implementează un algoritm trebuie să fie o planificare a activităților didactice pentru fiecare program de studii (PS) în parte, ce acoperă toate activitățile disciplinelor din planul de învățământ (12 PS X 8 discipline/PS X 2 activități/ disciplină = 192 activități). Planificarea activităților rezultată trebuie să conțină date despre disciplină, tipul de activitate, paritate, titular și sală. Astfel, se dorește ca planificarea activităților să fie afișată într-o formă tabelară a unei matrice cu următoarea structură (Tabel 10. 1), în care culorile reprezintă tipul de activitate, pe baza soluțiilor generate:

Tabel 10. 1 Structură orar conform cerințe

Zi/Interval orar	Grupa 1 semigrupa 1		Grupa 1 semigrupa 2	
Luni				
8-9 9-10	Disciplina Sală Titular *(aici e impar)*	Disciplina Sală Titular *(aici e par)*	Disciplina Sală Titular *(aici e impar)*	Disciplina Sală Titular *(aici e par)*
10-11 11-12	Disciplina Sală Titular		Disciplina Sală Titular	
12-13 13-14 14-15	Disciplina Sală Titular			
15-16				

10.2. Validarea restricțiilor în procesul de planificare a activităților didactice

Restricțiile pot fi împărțite în două categorii: restricții **hard** (care trebuie respectate obligatoriu) și restricții **soft** (opționale, care influențează calitatea soluției). Din analiza cerințelor din cadrul Facultății de Inginerie Industrială și Robotică, au fost elaborate restricțiile pe baza cărora se va încerca implementarea celor doi algoritmi aleși.

10.3 Algoritmul Backtracking

Algoritmul backtracking este o metodă utilizată pentru a găsi soluții la probleme complexe. Spre deosebire de alți algoritmi, backtracking-ul este mai rapid, deoarece elimină o mare parte dintre soluțiile imposibile printr-o singură parcurgere a datelor. Acest algoritm funcționează prin împărțirea problemei principale în subprobleme mai mici, pe care le rezolvă pe rând. După ce găsește o soluție pentru prima subproblemă, algoritmul trece la următoarea, bazându-se pe soluția anterioară. Dacă nu reușește să rezolve o subproblemă, algoritmul revine la pașii anteriori (backtrack) și caută o altă soluție care să rezolve atât subproblema curentă, cât și pe cele precedente, continuând apoi cu celelalte subprobleme.

10.3.1 Logica algoritmului Backtracking aplicată planificării activităților

Se cunoaște că algoritmul Backtracking realizează căutare sistematică prin toate combinațiile posibile de alocări, cu revenire atunci când selecția duce la un conflict. Conform datelor tehnice despre algoritm prezentate în Capitolul 4, se va explica logica algoritmului Backtracking aplicat planificării activităților didactice:

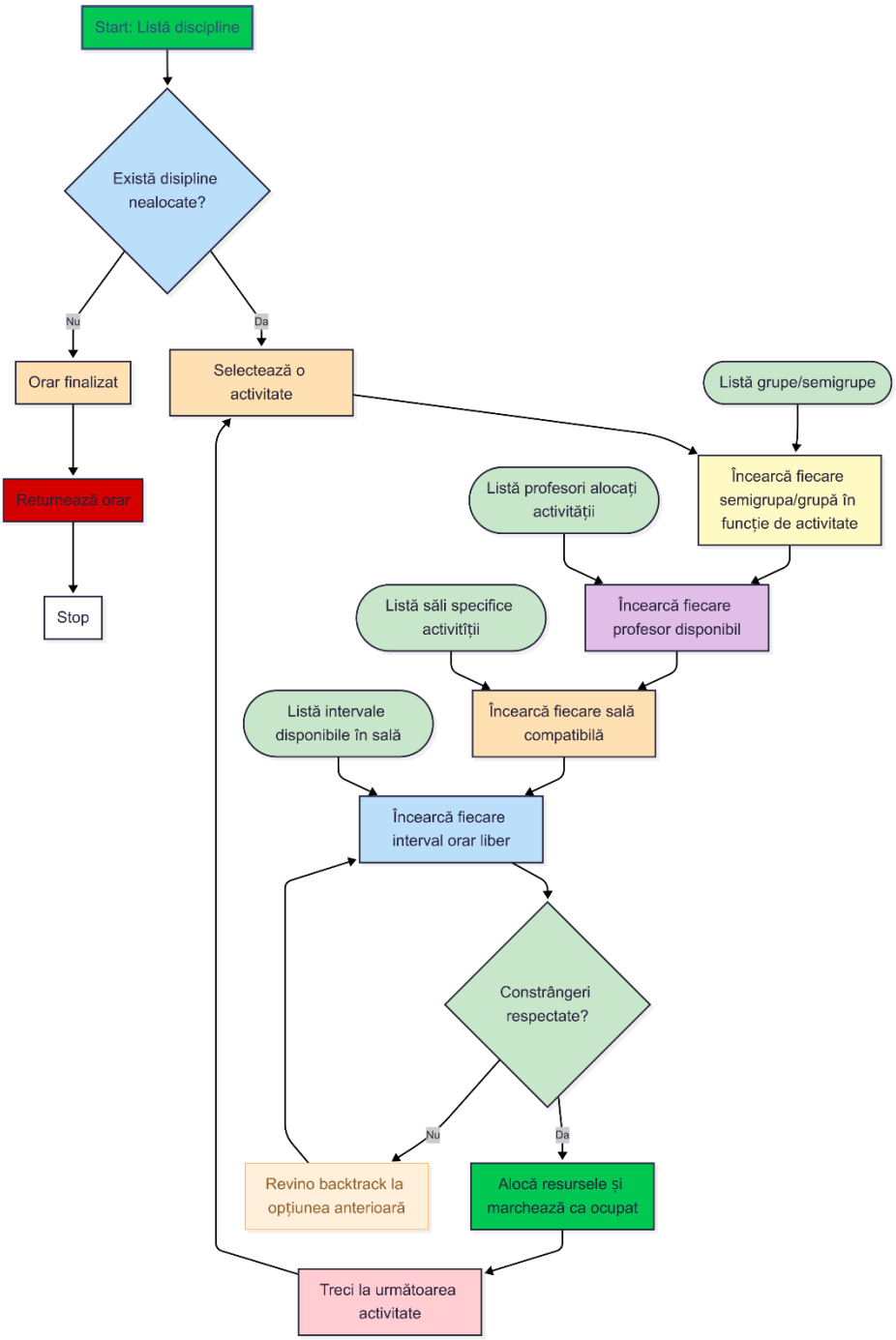


Fig. 10. 1 Schema logică algoritm Bactracking

10.4 Algoritmul Genetic

În urma cercetărilor multiplelor metode de dezvoltare a unei planificări automate a activităților, s-a constat că algoritmii genetici sunt cei mai utilizați pentru implementarea unei aplicații de generare a planificării activităților. Acesta este capabil să genereze un număr suficient de mare de soluții care depășesc restricțiile hard, evitând suprapunerile sălilor, profesorilor și grupelor de studenți.

10.4.1 Logica algoritmului Genetic aplicată planificării activităților didactice

Așa cum au fost prezentați și în Capitolul 4, algoritmii genetici sunt o metodă euristică de optimizare bazată pe procesul de selecție naturală și evoluție biologică. Acesta operează prin modificarea iterativă a unei populații de soluții individuale, selectând aleatoriu indivizi pe baza valorii lor de fitness pentru a deveni părinți și utilizându-i pentru a genera descendenți în populațiile următoare.

Există cinci faze importante ce sunt analizate în algoritmii genetici: populația inițială, funcția de fitness, selecția, încrucișarea și mutația, Fig. 10. 9.

Proiectarea funcției de fitness este esențială, deoarece rezultatele algoritmului depind de modul în care aceasta este gândită. Valoarea de fitness a fiecărui individ este determinată prin aplicarea funcției de fitness, care reprezintă o funcție specifică problemei și evaluează performanța soluțiilor.

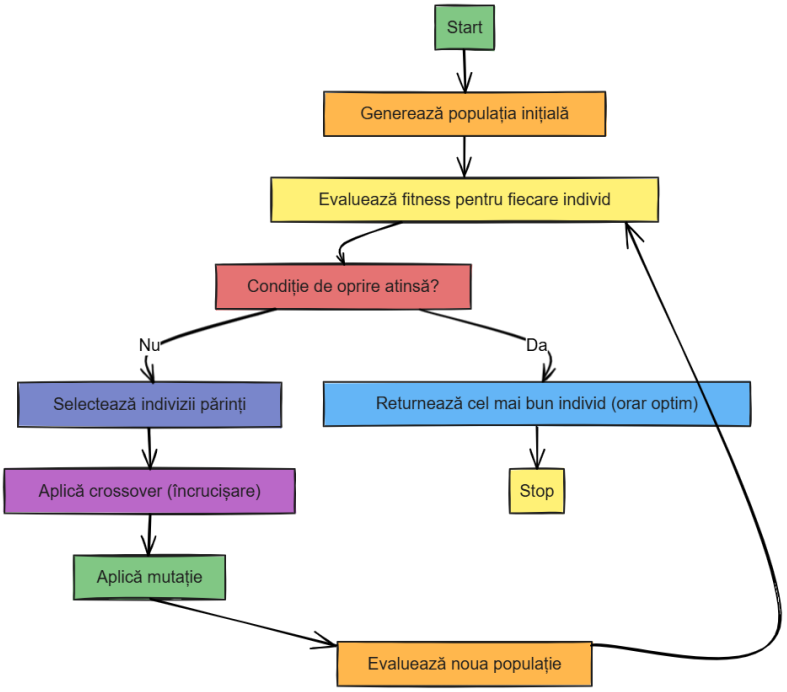


Fig. 10. 2 Schema logică algoritm Genetic

10.5 Algoritm Machine Learning (ML)

Pe baza documentației existente, din capitolul 4, în care sunt descrișii pașii necesari dezvoltării unui algoritm de învățare automată, s-a realizat o metodologie de implementare cu ajutorul căreia se vor extrage, prelucra și pregăti datele pentru predicția finală. În urma finalizării procesului de extragere a datelor se vor urma câteva etape din algoritmul de învățare automată (cu ajutorul limbajului de programare Python), fiecare etapă fiind transpusă în subprograme folosind:

- OrariiRun.py – orchestratorul tuturor subprogramelor create,
- ConfigReader – subprogramul ce apelează configurațiile JSON definite în configurator,
- DataReader – subprogramul care citește toate datele din folderul raw_data,
- DataProcessor – prelucrează datele pentru uniformizare,
- DataWriter – generează fișiere cu conținutul pe care îl doresc,
- DataPredictor – prezice date noi cu ajutorul unui algoritm ales,
- DataConflictSolver – verifică conflictele existente și le rezolvă.

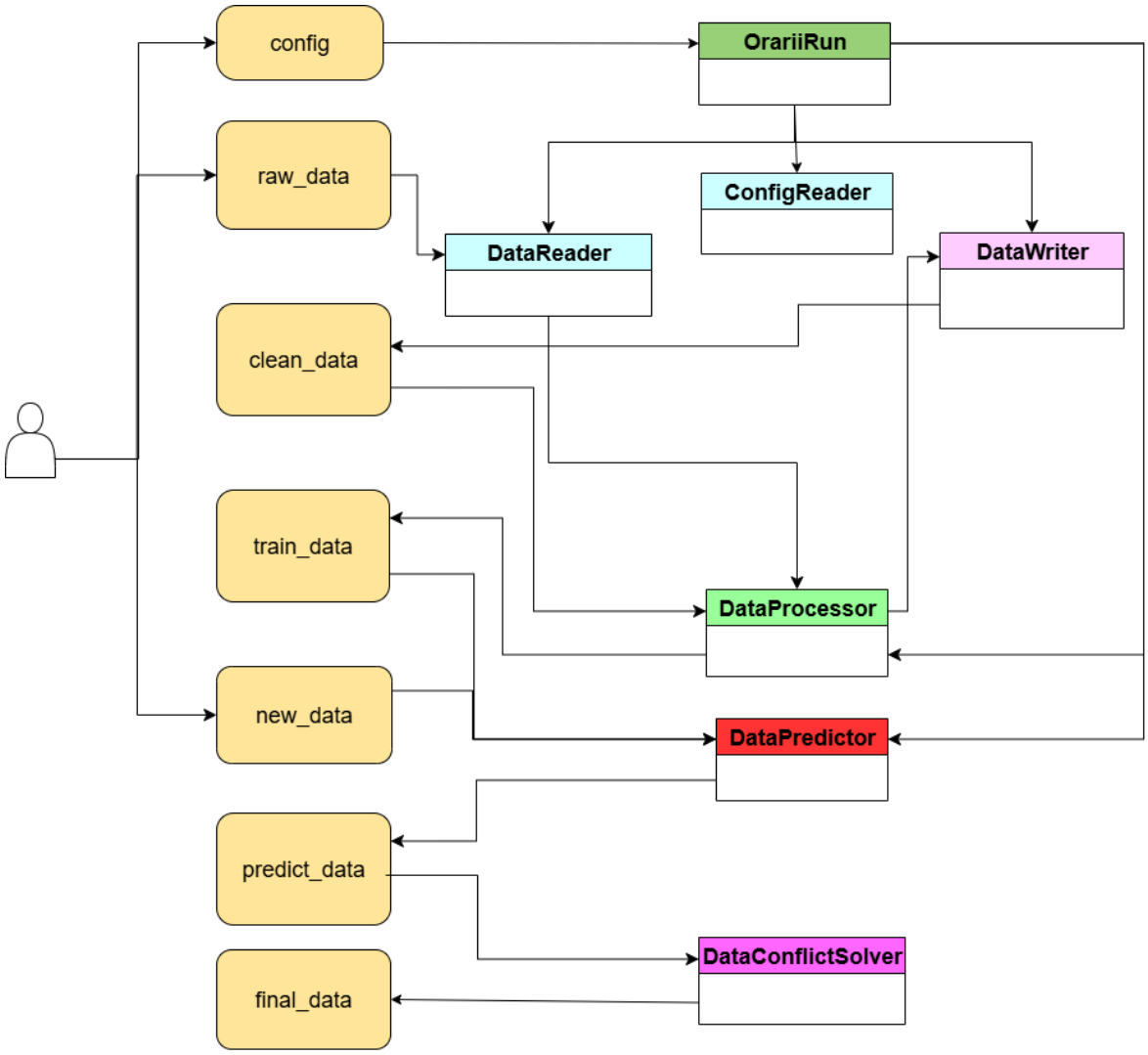


Fig. 10. 3 Metodologie de implementare ML

10.5.4. Rezolvarea conflictelor

Ca în orice generare de soluții folosind diverși algoritm ML, acestea trebuie ”șlefuite” pentru a elimina diferite neconcordanțe introduse pe baza valorilor din setul de date de training. Pentru acestea se folosește clasa ConflictSolver în care sunt analizate 4 cazuri critice ce trebuie să fie rezolvate înainte de a genera soluția finală: capacitatea sălii să fie mai mică decât numărul de studenți din semigrupă/grupă sau specializare și să nu existe suprapuneri în sală sau în orarul profesorilor sau al grupelor.

Pentru aceasta s-a implementat metoda privată detect_conflicts() în interiorul căreia se compară fiecare sală atribuită unei activități cu numărul de studenți existenți în tabelul groups, și în cazul în care condiția nu este respectată se stochează linia într-o variabilă de conflicte. Dacă condiția este îndeplinită se trece la următoarele verificări și anume identificarea dublurilor în urma grupării după sală, profesor sau grupă/semigrupă. Dacă există dubluri, se stochează în continuare la variabila conflicte ce urmează a se rezolva (Fig. 10. 26).

Rezolvarea conflictelor reprezintă o nouă metodă ce este abordată în ConflictSolver, în cadrul căreia se analizează din nou cele două criterii importante: dimensiunea sălii și existența dublajelor. Se parcurge lista de conflicte generată anterior și se analizează pentru fiecare criteriu în parte. Dacă găsește un conflict generat din verificare dimensiunii sălii, se caută o sală disponibilă care să îndeplinească condiția de capacitate și o înlocuiește în fișierul cu datele prezise. Dacă găsește un conflict generat de suprapuneri trebuie să verifice mai multe condiții.

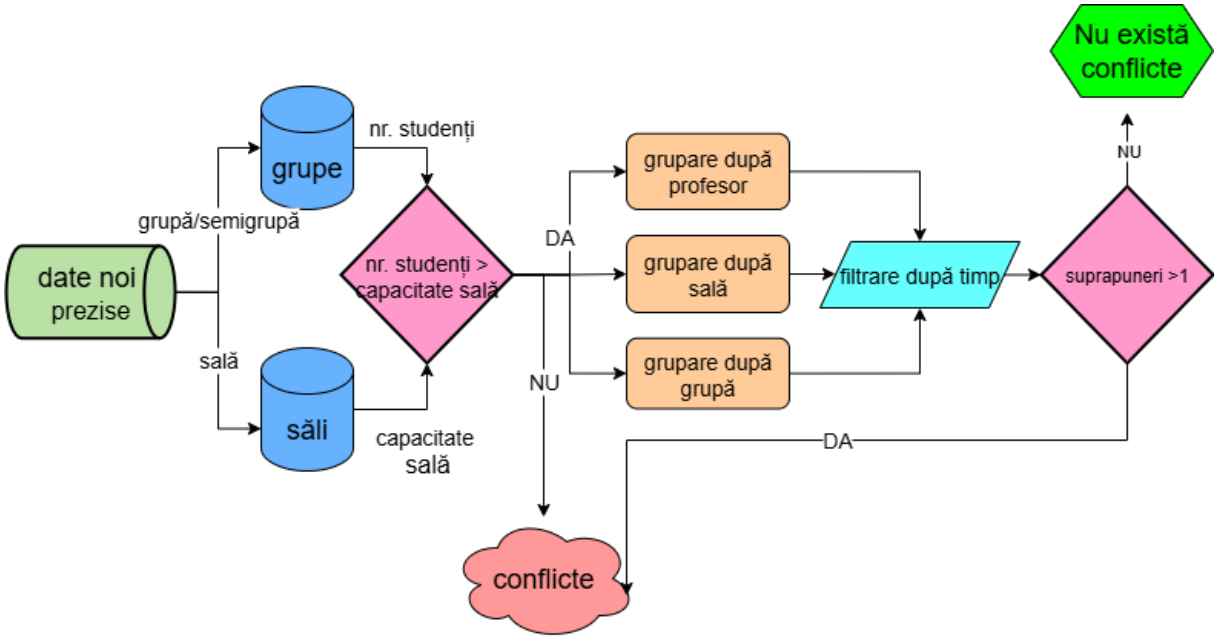


Fig. 10. 4 Identificarea conflictelor

În cazul în care suprapunerea se identifică pe sală, se caută o sală liberă cu capacitate corespunzătoare și se înlocuiește în fișierul cu date. Dacă găsește suprapunere pe grupă, identifică disciplina și caută un alt interval orar disponibil în care o poate aloca. Ultima condiție de verificare a suprapunerilor pe profesori se rezolvă prin verificarea unei mapări dintre disciplină și profesor și identificarea disponibilității acestuia și apoi alocarea acestuia în fișierul cu date.

CAPITOLUL 11
PROIECTAREA INTERFEȚEI GRAFICE

Una dintre cele 3 componente principale ale dezvoltării unei aplicații o reprezintă interfața grafică (GUI - Graphical User Interface), care facilitează contactul dintre utilizator și aplicație.

Interfața grafică a fost structurată astfel încât să prezinte informații despre planificarea activităților didactice și să permită utilizatorilor să efectueze acțiuni rapide și structurate, cum ar fi completarea de formulare, navigarea în pagini, selectarea de opțiuni și multe altele. Principalele componente ale aplicației dezvoltate sunt: autentificare/login, pagina principală, profesori, săli, preferințele profesorilor, planurile de învățământ, pagina pentru alocare profesori la discipline și generatorul de planificare a activităților didactice.

Organizarea codului pentru interfața grafică s-a realizat cu ajutorul framework-ului Angular. Paginile au fost structurate pe componente, iar fiecare componentă este compusă din trei fișiere principale care lucrează împreună pentru a defini aspectul, stilul și funcționalitatea acelei componente, de tip: HTML, CSS și TypeScript.

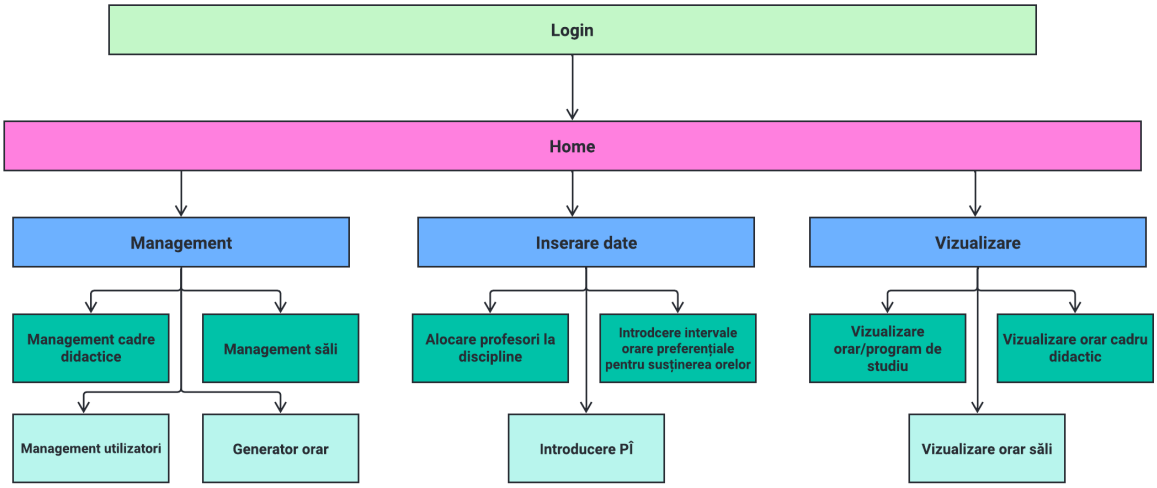


Fig. 11. 1 Structura paginilor din interfață grafică

11.1. Autentificare/Login

Aplicația de management al planificării activităților didactice are ca utilizatori cadrele didactice dintr-o facultate, administratorii aplicației și directorii de departament. Gestionarea informațiilor private și securizarea accesului la resursele și funcționalitățile aplicației au necesitat dezvoltarea unei interfețe de autentificare/login sau înregistrare. În cadrul aplicației, necesitatea dezvoltării unei componente de login vine de la faptul că informațiile ce fac posibilă planificarea automată a activităților didactice au un caracter privat.

În urma autentificării reușite, pe baza numelui și a funcției din tabela de user_register, pagina principală conține câmpuri diferite, (Fig. 11. 5), astfel:

- Admin: Profesori/Săli, ILSA (Intelligent Logic Scheduling Algorithm), Utilizatori, Planuri de Învățământ, Management Profesori, Management Săli

- Director departament: Preferințele mele, Profesori/Săli, Alocare profesori, Planuri de Învățământ, Orar personal și Vizualizare orar
- Profesori: Preferințele mele, Orar personal, Vizualizare orar și Profesori/Săli.

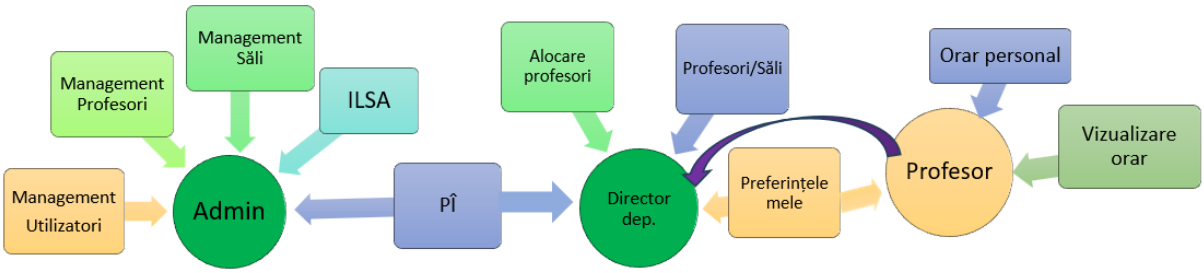


Fig. 11. 2 Structura paginilor în funcție de rolul utilizatorilor

O pagină dedicată managementului cadrelor didactice ce susțin disciplinele din Facultate de Inginerie Industrială și Robotică este necesară pentru a facilita administrarea eficientă și rapidă, Fig. 11. 13.

Toate cererile trimise din frontend (GET, POST sau PUT) sunt procesate de server care execută o operațiune SQL de tip INSERT/UPDATE/DELETE (4), înregistrând informațiile în baza de date. După finalizarea cu succes a acestei operațiuni, baza de date revine cu un mesaj de succes (5) către server, iar acesta răspunde cu un mesaj HTTP 200 (acțiune finalizată cu success) către frontend. Astfel, utilizatorul este informat cu un mesaj de tip pop-up statusul operației realizate: ”Datele au fost salvate cu succes” (7), ”Utilizatorul a fost șters/modificat” (8), Fig. 11. 14.

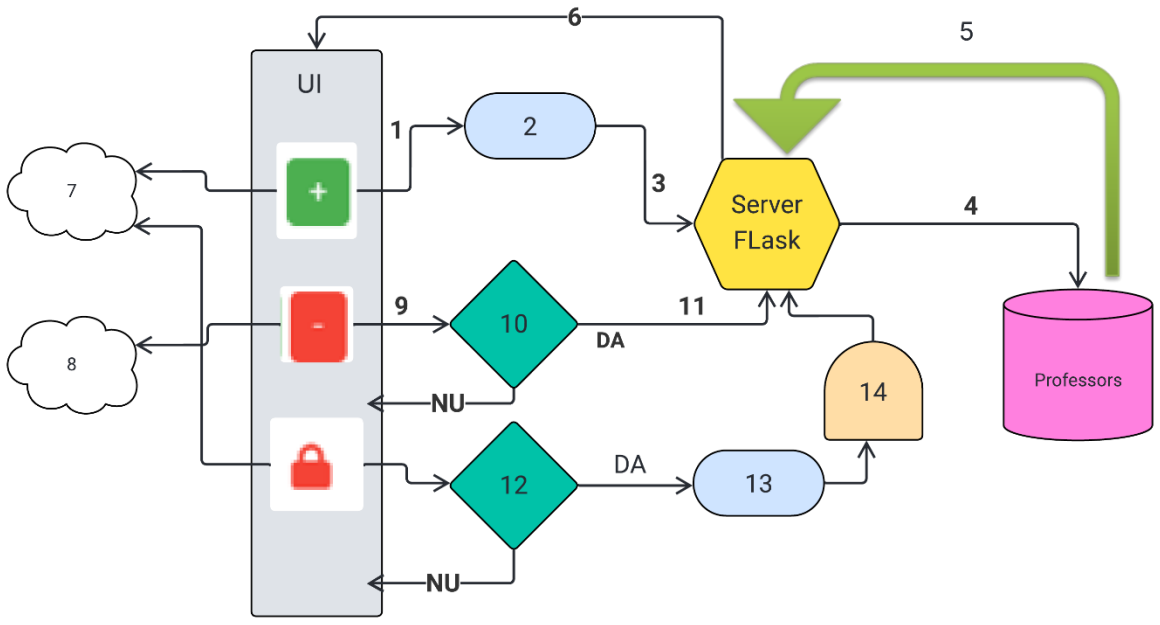


Fig. 11. 3 Flow funcționalități adăugare, editare și ștergere cadru didactic

11.6. Săli

Pagina de săli este dedicată cadrelor didactice și directorului de departament, având rol de vizualizare a informațiilor despre săli, și anume numele sălii, capacitatea și tipul de activitate ce se poate desfășura (curs, seminar, proiect, laborator tehnic sau laborator cu calculatoare), Fig. 11. 16. Pentru vizualizarea informațiilor din fișierul de TypeScript al componentei, se trimite un request de tip GET către serverul Flask, acesta conectându-se la tabela rooms selectează toate datele existente, introducându-le într-o lista ce este retrimisă (Fig. 11. 15).

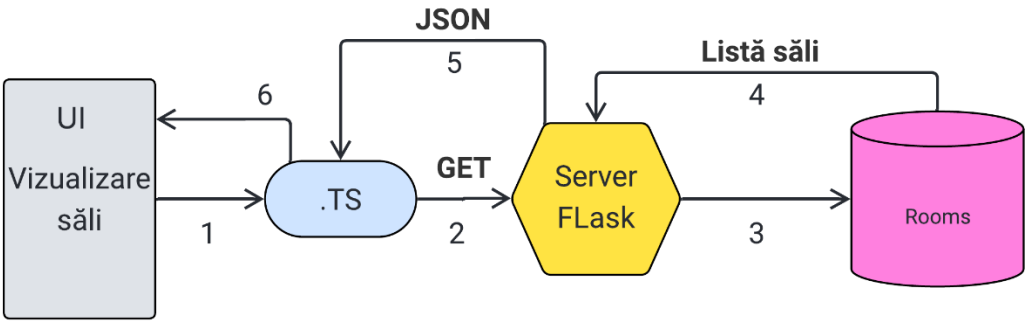


Fig. 11. 4 Procedură GET listă cadre didactice

Dezvoltarea unei interfețe pentru gestionarea informațiilor relevante din planurile de învățământ eficientizează foarte mult timpul de introducere în baza de date a informațiilor. Persoanele care pot accesa această interfață sunt doar administratorii și directorii de departamente. În cadrul interfeței a fost construită o zonă de vizualizare a planurilor deja introduse și a secțiuni de încărcare, validare și salvare pentru fiecare specializare în parte, Fig. 11. 21.

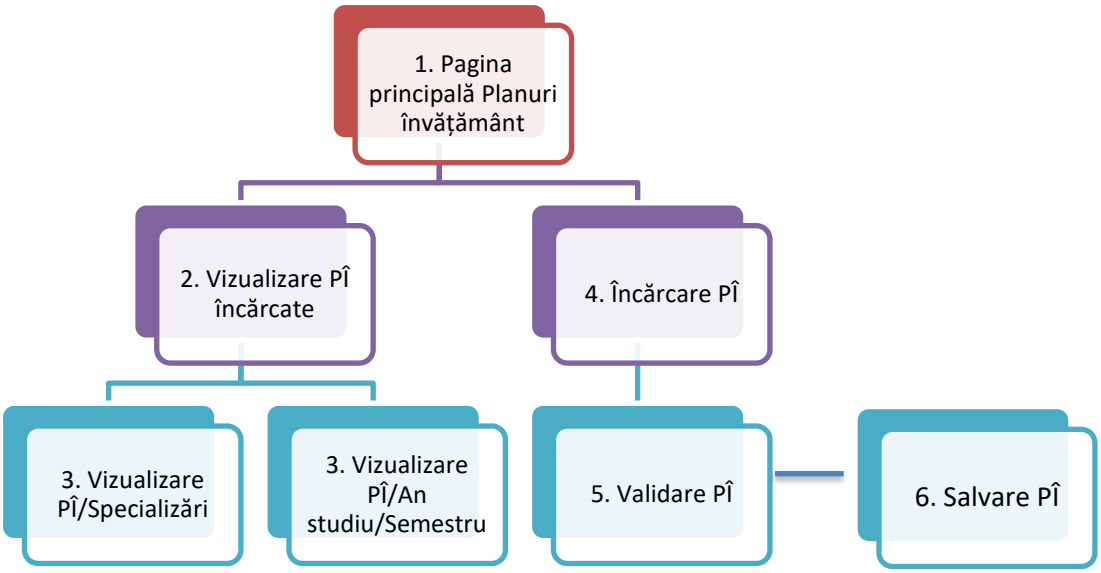


Fig. 11. 5 Structura subpaginilor Plan Învățământ

CAPITOLUL 12
ANALIZA ȘI TESTAREA APLICAȚIEI

În cadrul acestui capitol se vor prezenta rezultatele și problemele întâmpinate în timpul dezvoltării celor 3 tipuri de algoritmi. Pentru algoritmii genetici și învățarea automată (ML), s-au putut genera rapoarte și grafice detaliate pentru diverse componente, datorită unor librării Python avansate precum matplotlib, seaborn, pandas și scikit-learn. Aceste librării au permis analiza eficientă a rezultatelor, oferind o viziune de ansamblu asupra acestora. Pentru algoritmul backtracking, generarea de rapoarte automate este limitată, deoarece acesta se bazează pe explorarea exhaustivă a soluțiilor posibile, astfel încât analiza se face manual, prin interpretarea datelor rezultate și a timpilor de execuție, fără utilizarea unor tool-uri specializate de vizualizare.

12.1 Alegerea specializărilor pentru observarea rezultatelor

În prezent în cadrul Facultății de Inginerie Industrială și Robotică există 10 programe de studii (PS) care au 4 ani de studiu, 1 PS ce are 3 ani de studiu si 2 PS ce au 2 ani de studiu, adică un total de 47 de generații de studenți pentru toate PS.

Dorința de a genera o planificare a activităților didactice validă pentru toate PS ale facultății fiind foarte mare, s-a încercat generarea automată a planificării cu toți cei 3 algoritmi și s-au observat nenumărate probleme întâmpinate care au dus la restructurarea numărului de specializări pentru a putea obține mai puține rezultate ce se pot compara.

În timpul implementării algoritmelor de generare (backtracking și genetic) prima problemă apărută a fost lipsa intervalelor orare în săli pentru alocarea tuturor seminarilor/proiectelor pentru toate specializările. Pentru a verifica acestor informații, cu datele concrete oferite de Facultatea de Inginerie Industrială și Robotică, s-au calculat pe baza planurilor de învățământ și a numărului de grupe/semigrupe existente numărul necesar de intervale orare pentru a putea susține toate activitățile disciplinelor.

12.2 Algoritmul Backtracking

În cadrul algoritmului backtracking, rezultatele obținute au fost sub formă de fișier .csv. Pentru a putea observa respectarea constrângerilor și corectitudinea datelor s-au prelucrat manual o parte din informații. Astfel, s-au ordonat după specializare, an, zi și interval orar datele, iar apoi s-a folosit o aplicație web generatoare de conținut AI, ChatGPT, pentru a aranja sub forma planificării activităților didactice dorită. Planificarea obținută este returnată sub formă de fișier Excel și are următoarea formă (Fig. 12. 1):

Zi	Interval	611		612	dual
		Sg.1	Sg.2	Sg. 1	Sg.2
Monday	08:00–10:00	Programarea calculatoarelor È™i limbaj de programare 1 curs, Tom Savu, CD 016	Programarea calculatoarelor È™i limbaj de programare 1 curs, Tom Savu, CD 016	Programarea calculatoarelor È™i limbaj de programare 1 curs, Tom Savu, CD 016	Programarea calculatoarelor È™i limbaj de programare 1 curs, Tom Savu, CD 016
	10:00–12:00	Chimie curs, Mariana Prodana, CD 016	Chimie curs, Mariana Prodana, CD 016	Chimie curs, Mariana Prodana, CD 016	Chimie curs, Mariana Prodana, CD 016
	12:00–14:00		Chimie laborator, George Burc, EG 005 - par	Limba engleza seminar, Alexandra Salcianu, CB 201	Limba engleza seminar, Alexandra Salcianu, CB 201
	14:00–16:00			Chimie laborator, George Burc, EG 005 - par	
	16:00–18:00				
	18:00–20:00				
Tuesday	08:00–10:00	Programarea calculatoarelor È™i limbaj de programare 1 laborator, Vlad GheorghiÈ›Äƒ, CB 202	Programarea calculatoarelor È™i limbaj de programare 1 laborator, Bogdan Jugravu, CB 203		D esen tehnic È™i infograficÄƒ 1 laborator, Iulius Stroe, BN 323
	10:00–12:00	Bazele ingineriei industriale 1 proiect, Corina Botez, BN 323	Bazele ingineriei industriale 1 proiect, Corina Botez, BN 323	Desen tehnic È™i infograficÄƒ 1 laborator, Iulius Stroe, BN 323	limbaj de programare 1 laborator, Bogdan Jugravu, CB 203
	12:00–14:00	Desen tehnic È™i infograficÄƒ 1 laborator, Iulius Stroe, BN 323		Mecanica 1 seminar, Costel Popa, CB 021	Mecanica 1 seminar, Costel Popa, CB 021
	14:00–16:00	Limba engleza seminar, Alexandra Salcianu, BN 027	D esen tehnic È™i infograficÄƒ 1 laborator, Iulius Stroe, BN 323	Programarea calculatoarelor È™i limbaj de programare 1 laborator, Bogdan Jugravu, CB 203	
	16:00–18:00	Mecanica 1 curs, Prunau Maria, CD 016	Mecanica 1 curs, Prunau Maria, CD 016	Mecanica 1 curs, Prunau Maria, CD 016	Mecanica 1 curs, Prunau Maria, CD 016

Fig. 12. 1 Planificarea activităților didactice obținută cu backtracking

În aceste condiții s-au generat toate planificările activităților pentru anii I-IV pentru toate specializările selectate și a fost analizată respectarea constrângerilor propuse, obținându-se următoarele rezultate (Tabel 12. 2).

12.3 Algoritmul Genetic

În urma rulării algoritmului genetic, au fost generate soluții viabile salvate într-o bază de date. Atingerea obiectivelor propuse s-a concretizat prin implementarea unei funcții de generare a unei matrice în format HTML pentru o vizualizarea mai rapidă a datelor generate de algoritm (Fig. x). Pe baza analizei rezultatelor obținute pentru cele 4 specializări s-au observat următoarele:

- Alocarea unor discipline în intervale orare, dar fără alocare unei săli;
- Alocarea unor discipline în intervale orare, dar fără alocare unui profesor;
- Inserarea unor ferestre de pauză între ore;
- Distribuție neuniformă a activităților pe zile.

Pentru fiecare specializare se descarcă planificarea activităților generată în format .html cu următoarea denumire: orar_specializare_serie_an.html a cărui interfață grafică arată că în Fig. 12. 2.

Orar An 1 IAII

Zi/Interval	611_1	611_2	612_1	612_2	613_1	613_2
Monday 08:00-10:00		Educație fizică 1 (CB-005) Seminar (para)				
Monday 10:00-12:00		Algebră liniară, geometrie analitică și diferențială (CE-301bis) Seminar (para)			Limbi moderne 1 - Franceză (CB-301) Curs (impara)	Limbi moderne 1 - Franceză (CB-301) Curs (impara)
Monday 12:00-14:00	Chimie (AN-017) Curs	Chimie (AN-017) Curs	Chimie (AN-017) Curs	Chimie (AN-017) Curs	Chimie (AN-017) Curs	Chimie (AN-017) Curs
Monday 14:00-16:00	Limbi moderne 1 - Engleză (CE-301bis) Seminar	Limbi moderne 1 - Engleză (CE-301bis) Seminar				Desen tehnic și infografică 1 (CK-006) Laborator (3 ore)
Monday 16:00-18:00		Desen tehnic și infografică 1 (CK-007) Laborator (3 ore)	Chimie (CA-013) Laborator (para)			
Monday 18:00-20:00						
Tuesday 08:00-10:00			Programarea calculatoarelor și limbaje de programare 1 (CB-109) Laborator		Fizică (CK-009d) Laborator (para)	Algebră liniară, geometrie analitică și diferențială (CB-005) Seminar (para)
Tuesday 10:00-12:00	Limbi moderne 1 - Engleză (CD-016) Curs (impara)	Limbi moderne 1 - Engleză (CD-016) Curs (impara)	Limbi moderne 1 - Engleză (CD-016) Curs (impara)	Limbi moderne 1 - Engleză (CD-016) Curs (impara)		Educație fizică 1 (CB-008) Seminar (para)
Tuesday 12:00-14:00	Algebră liniară, geometrie analitică și diferențială (CB-006) Seminar (para)			Desen tehnic și infografică 1 (CB-009) Laborator (3 ore)		
Tuesday 14:00-16:00			Educație fizică 1 (CB-005) Seminar (para)			
Tuesday 16:00-18:00	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)	Desen tehnic și infografică 1 (CD-008) Curs (3 ore)
Tuesday 18:00-20:00						
Wednesday 08:00-10:00	Educație fizică 1 (CB-005) Seminar (para)		Algebră liniară, geometrie analitică și diferențială (CB-008) Seminar (para)			
Wednesday 10:00-12:00	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)	Programarea calculatoarelor și limbaje de programare 1 (AN-028) Curs (3 ore)

Fig. 12. 2 Planificarea activităților didactice obținută cu algoritmul genetic

Conform principiului fundamental pentru algoritmul genetic, Eiben și Smith afirmă că: ”Calitatea celei mai bune soluții găsite de un algoritm evolutiv depinde de mărimea populației și de numărul de generații. Populațiile mai mari tind să producă o diversitate mai bună și necesită mai puține generații pentru a converge, în timp ce populațiile mici pot avea nevoie de mai multe generații pentru a explora eficient” (Eiben & Smith, 2003).

Se folosește limbajul de programare Python pentru a calcula valorile obținute și pentru a afișa graficul 3D generat. Librăria folosită pentru calcularea scorului este numpy (Numerical Python) – cu ajutorul căreia se creează funcția de model Mahfound, iar pentru afișarea graficului s-au utilizat librăriile matplotlib – folosite pentru vizualizarea datelor sub formă de grafice și Axes3D – un modul al matplotlib în care se pot crea vizualizări 3D ale unor valori. Pe baza datelor din matrice se plotează graficul 3D pentru numărul de generații, populație și scorul fitness estimat. În fig. 12. 5, se poate observa că scorul de fitness crește rapid în primele faze, însă atinge o suprafață plană odată cu atingerea unui prag de 200–250 de indivizi și peste 80 de generații. Acest comportament reflectă o saturație naturală a algoritmului, în care creșterea resurselor nu mai aduce îmbunătățiri semnificative ale performanței.

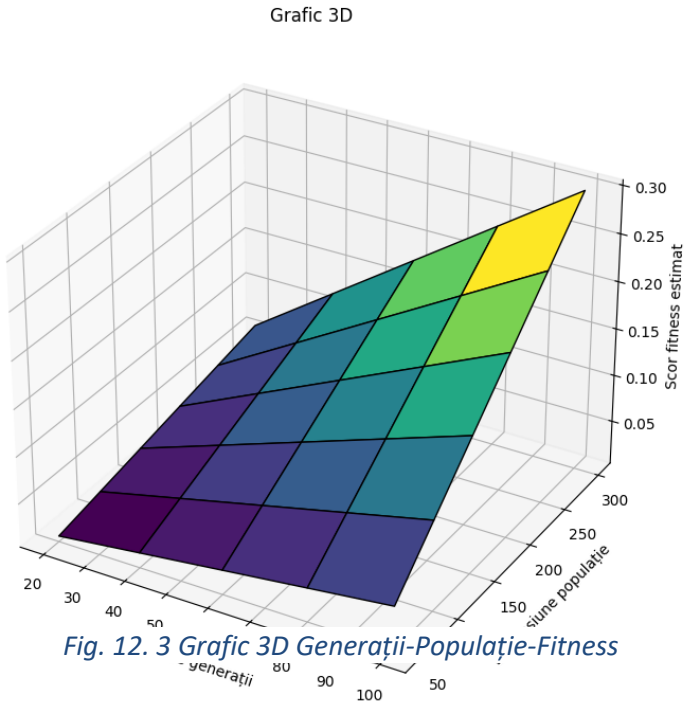


Fig. 12. 3 Grafic 3D Generații-Populație-Fitness

12.4 Machine Learning

Pe baza prelucrărilor efectuate pentru datele extrase din planificările activităților didactice existente din ultimii 10 ani, s-au obținut date de antrenare pentru prezicerea noilor date. Astfel, s-a aplicat algoritmul Random Forest Classifier, configurat cu parametri standard: 100 de arbori de decizie (n_estimators=100), criteriul de separare Gini (criterion='gini') și fără limită de adâncime (max_depth=None) și s-au prezis 3 parametrii necesari realizării unei planificări și anume, ziua, intervalul orar și sala. Pentru asigurarea reproductibilității, s-a setat o ”sămânță” aleatoare (random_state=42).

În această etapă de antrenare, datele au fost împărțite utilizând metoda *train_test_split* din biblioteca *scikit-learn*, alocând 80% din eșantion pentru antrenare și 20% pentru testare. Pentru fiecare parametru specific al planificării activităților didactice — ziua, intervalul orar și sala — s-a antrenat câte un model distinct, analizând fiecare predicție ca o problemă de clasificare multi-clasă. Modelele antrenate au fost stocate într-o structură internă (self.models), și au fost extrase din fișierul train_data.csv, permițând accesul și evaluarea individuală a fiecărui parametru.

Toate aceste informații au fost prelucrate parțial manual și cu ajutorul unui instrument AI au fost transpuse sub forma de matrice cu zilele și intervale orare. Astfel, a rezultat o planificare a activităților cu o afișare aproximativă cu varianta existentă, care poate fi validat doar după ce profesorii îl pot consulta și verifica. Pe baza analizelor efectuate, se poate afirma faptul că este o variantă stabilă a unui orar generat automat ce satisface restricțiile soft impuse.

Ziua /	611-1	611-2	612	dual
Luni - 08			Educație fizică / Pricop Adrian / sala sport (par)	Educație fizică / Pricop Adrian / sala sport (par)
Luni - 10	Educație fizică / Pricop Adrian / sala sport (impar)	Educație fizică / Pricop Adrian / sala sport (impar)	Chimie / George Burcă / eg005 (par)	Chimie / Mariana Prodana / eg005 (par)
Luni - 12	Limba engleză / Alexandra Salcianu / bn027 (impar)	Limba engleză / Alexandra Salcianu / bn027 (impar)	Bazele ing. / Corina Botez / bn323 (sapt. 1-7)	Bazele ing. / Corina Botez / bn323 (sapt. 1-7)
Luni - 14	Bazele ing. / Corina Botez / bn323 (sapt. 1-7)	Bazele ing. / Corina Botez / bn323 (sapt. 1-7)		
Marți - 08	desen tehnic / Sorin Botez / bn308 (saptamanal)	desen tehnic / Iulius Stroe / bn323 (saptamanal)		
Marți - 12	Algebra lin. / Cristina Calineata / cb006 (par)	Algebra lin. / Cristina Calineata / cb006 (par)		
			Mecanica / Costel Popa / bn023 (impar)	Mecanica / Costel Popa / bn023 (impar)
Marți - 14			Algebra lin. / Cristina Calineata / cb007 (par)	Algebra lin. / Cristina Calineata / cb007 (par)
Marți - 16			Limba engleză / Alexandra Salcianu / cb201 (par)	Limba engleză / Alexandra Salcianu / cb201 (par)
Miercuri - 10	Mecanica / Costel Popa / cb021 (impar)	Mecanica / Costel Popa / cb021 (impar)	Știința materialelor / Marius Vasilescu / jk205 (par)	
Miercuri - 12	desen tehnic / Sorin Botez / cd016 (impar)	desen tehnic / Sorin Botez / cd016 (impar)	desen tehnic / Sorin Botez / cd016 (impar)	desen tehnic / Sorin Botez / cd016 (impar)
	Engleză / Alexandra Salcianu / cd016 (par)	Engleză / Alexandra Salcianu / cd016 (par)	Engleză / Alexandra Salcianu / cd016 (par)	Engleză / Alexandra Salcianu / cd016 (par)

Fig. 12. 4 Planificarea activităților didactice generată cu ML

12.6 Comparația celor trei algoritmi

Cei trei algoritmi analizați în cadrul prezentei teze — backtracking, algoritmul genetic și Random Forest Classifier— oferă abordări diferite asupra problemei generării unei planificări a activităților universitare. Algoritmul Backtracking fiind un algoritm exact, este capabil să găsească o soluție validă, dar cu un cost computațional ridicat, ceea ce îl face greu de utilizat și aplicat pentru mai multe constrângeri. Algoritmul genetic (metodă evolutivă) oferă un bun compromis între calitate și timpul de execuție, reușind să genereze o planificare a activităților optimă în funcție de constrângerile impuse, dar fără a garanta o soluție complet validă. Random Forest Classifier, ca algoritm de învățare automată, funcționează eficient din punct de vedere al unui număr mediu de resurse folosite, însă este dependent de date istorice etichetate și nu are capacitate de optimizare sau validare a logici unei planificări a activităților. Astfel, fiecare metodă prezintă puncte forte distincte și poate fi aleasă în funcție de contextul aplicării sau se pot realiza două algoritme pentru a obține soluții optime.

Pentru o analiza grafic cei trei algoritmi s-a realizat un script prin care s-au acordat un scor pe o scară de la 0-10 criteriilor principale și s-au reprezentat grafic rezultatele obținute.

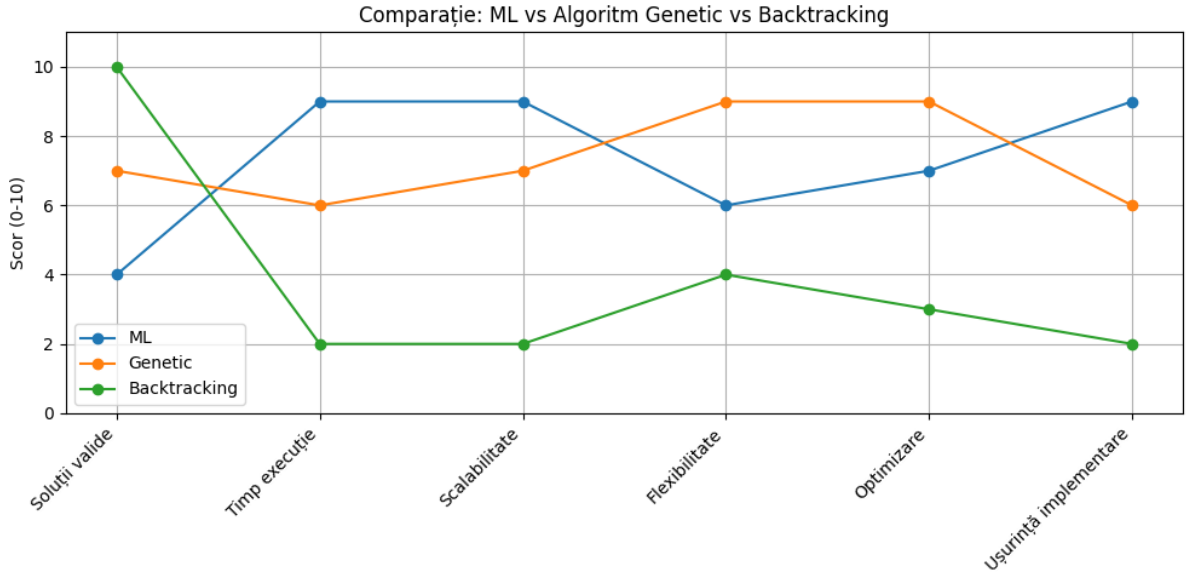


Fig. 12. 5 Grafic comparație algoritmi

CONCLUZII

Una dintre cele mai grele decizii de luat în generarea unei planificări a activităților didactice automat este alegerea tipului de algoritm ce se dorește a se implementa. Analiza detaliată a numeroaselor aplicații existente în universități și online a bonificat alegerea celor trei algoritmi implementați în cadrul acestei teze. S-au analizat trei algoritmi relevanți pentru rezolvarea acestei probleme, Backtracking, Genetic și Random Forest Classifier, urmărind atât acuratețea și eficiența lor, cât și capacitatea de adaptare la cazuri concrete. Implementarea acestor algoritmi s-a dovedit a fi o abordare viabilă, fiecare având avantaje, dezavantaje și limitări specifice.

CONTRIBUȚII PROPRII

Lucrarea de față are atât un caracter teoretic, cât și unul aplicativ și aduce o serie de contribuții originale privind modelare, implementare algoritmilor și dezvoltarea unei aplicații pentru planificarea activităților didactice din Facultatea de Inginerie Industrială și Robotică.

4. *Contribuții practice privind dezvoltarea aplicației*
    a. Proiectarea arhitecturii software pentru dezvoltarea aplicației;
    b. Proiectarea, dezvoltarea și optimizarea bazei de date ce este necesară pentru implementarea aplicației software;
    c. Implementarea algoritmilor Backtracking, Genetic și Random Forest Classifier cu ajutorul limbajului de programare Python;
    d. Proiectarea, dezvoltarea și optimizarea interfeței grafice (frontend) a aplicației software în Angular;
    e. Testarea algoritmilor și a interfeței grafice și elaborarea comparațiilor între rezultatele obținute pentru fiecare tip de algoritm implementat.

DIRECȚII VIITOARE DE CERCETARE

În urma dezvoltărilor realizate și a analizei celor trei algoritmi, se poate confirma posibilitatea de optimizare sau implementare hibridă. Complexitatea problemei planificării automate a activităților didactice și existența unui număr mare de diferiți algoritmi de optimizare, crește considerabil aria de dezvoltare a mai multor direcții de cercetare. Dezvoltarea unei soluții scalabile, eficiente și rapide, în care inteligența artificială, teoria grafurilor și diverși algoritmi se ”întrunesc” pentru a răspunde nevoilor reale ale instituțiilor de învățământ superior.

O primă direcție o reprezintă optimizarea parametrilor algoritmului genetic (mutație, încrucișare și dimensiunea populației). Alegerea unor valori potrivite pentru dimensiunea populației, rata de mutație sau criteriile de selecție poate influența decisiv calitatea soluțiilor existente. Viitoare cercetări pot viza algoritmi de tip self-tuning, care își ajustează automat parametrii în funcție de evoluția soluției. Astfel, se poate reduce intervenția umană asupra parametrilor și alegerea acestora pe baza unor criterii.