**ROMÂNIA**

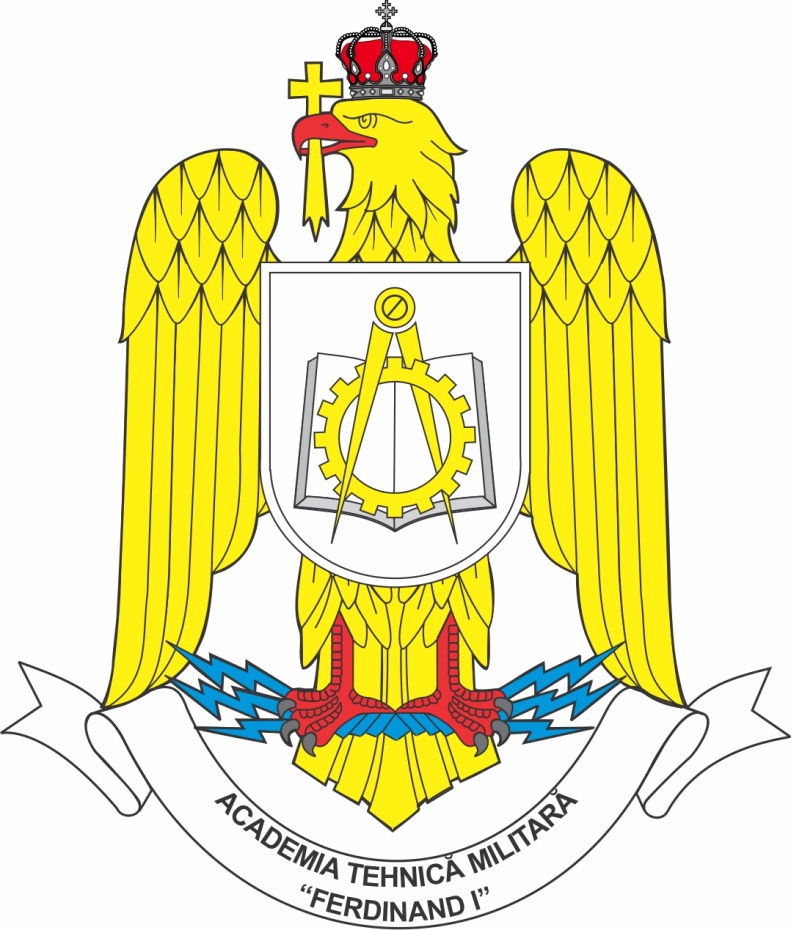
**MINISTERUL APĂRĂRII NAŢIONALE**

**ACADEMIA TEHNICĂ MILITARĂ „FERDINAND I”**

**FACULTATEA DE SISTEME INFORMATICE ŞI SECURITATE CIBERNETICĂ**

***Specializarea: Calculatoare şi sisteme informatice pentru apărare***

***şi securitate naţională***



**TEMA: “ SOLUȚIE WEB PENTRU MANAGEMENTUL FLUXULUI DE DOCUMENTE”**

CONDUCĂTOR ŞTIINŢIFIC:

Col. prof. univ. dr. ing. Mihai TOGAN

ABSOLVENT:

**Stud. sg. maj. STROE Cătălina-Elena**

Conţine \_\_\_\_\_\_\_\_\_\_\_ file

Inventariat sub nr. \_\_\_\_\_\_

Poziţia din indicator: \_\_\_\_

Termen de păstrare: \_\_\_\_\_

**BUCUREŞTI**

**2022**

Abstract

This Graduation Thesis analyses the document management flow of the content stored in a document management system and directs to an implementation of such a system through the use of open-source technologies.

Modeling and monitoring procedures that can be applied to the workflow within some departments will be studied and implemented, through operations of implementing the flow dynamics of the data loaded in the system. The solution will achieve the distribution of the documents, based on groups, roles and access rights to the recipients that have a decision-making role. Quick searches will be performed and the integrity and authenticity of the documents will be guaranteed through a module that integrates digital signatures features needed for certain actions concerning the document. Another functionality consists in the archiving mechanism of the data marked as completed.

The first chapter presents, in broad lines, the theme and its importance, in terms of using a system already existing on the market. The current section also incorporates both the purpose and objectives of the paper, as well as a summary of each chapter, followed by a brief presentation of the results obtained.

The second chapter will analyze similar solutions and existing implementations and will list and illustrate the relevant software components that will be used to create the target application.

The third chapter focuses on the practical part of the application. This consists of the implementation of an electronic document flow management system. In the current division of the paper, the complete requirements will be set and the use cases will be presented. It will also specify the technologies used to implement the components and describe the test cases by issuing a report in this regard. Also, in the current chapter, the problems encountered during the implementation of the project solution will be pointed out.

The final chapter presents a series of conclusions drawn from the paper, which will support further research. This research aims at developing and improving new facilities in the field of electronic document flow management systems.

Rezumat

În lucrarea de față se analizează fluxul de gestionare a conținutului stocat în cadrul unui sistem de management de documente și îndrumă la o variantă de implementare a unui astfel de sistem prin utilizarea tehnologiilor open-source.

Se vor studia și se vor pune astfel în aplicare procedee de modelare și monitorizare ce pot fi aplicate asupra cursului de lucru din cadrul unor departamente, prin operații de implementare a dinamicii de parcurs a datelor încărcate în sistem. Soluția va realiza distribuția documentelor către destinatarii cu rol de decizie, pe bază de grupuri, roluri și drepturi de acces. Se vor realiza căutări rapide și se va garanta integritatea și autenticitatea documentelor prin intermediul unui modul de integrare a unor funcționalități de semnătură digitală necesară la o anumită acțiune asupra documentului. Altă funcționalitate constă în mecanismul de arhivare a datelor marcate ca fiind finalizate.

În primul capitol al acestei lucrări se prezintă, în linii generale, tema și importanța acesteia, prin prisma utilizării unui astfel de sistem deja existent pe piață. În cadrul secțiunii curente mai sunt încorporate atât scopul și obiectivele lucrării, cât și un sumar al fiecărui capitol, urmat de expunerea concisă a rezultatelor obținute.

În capitolul secund se vor analiza soluții similare și implementări deja existente și se vor enumera și exemplifica componentele software relevante ce vor fi folosite pentru realizarea aplicației ţintă.

Capitolul al treilea este concentrat asupra părții practice a aplicației. Aceasta constă în implementarea unui sistem de gestiune al fluxului de documente electronice. În diviziunea curentă a lucrării se vor expune cerințele complete și se vor prezenta cazurile de utilizare. De asemenea, se vor preciza tehnologiile utilizate pentru implementarea componentelor și se vor descrie cazurile de testare prin eliberarea unui raport în acest sens. Tot în cadrul capitolului curent se vor puncta și problemele întâmpinate pe parcursul implementării soluției proiectului.

Capitolul final prezintă o serie de concluzii extrase din cadrul lucrării, ce vor susține cercetările ulterioare. Aceste cercetări au ca scop dezvoltarea și perfecționarea de noi facilități în domeniul sistemelor de management al fluxului de documente electronice.

**CUPRINS**

Cuprins

[1. Introducere 13](#_Toc107793582)

[1.1. Introducere în domeniu 13](#_Toc107793583)

[1.2. Importanța temei alese 13](#_Toc107793584)

[1.3. Scopul și obiectivele lucrării 14](#_Toc107793585)

[2. Sisteme și standarde în domeniul managementului fluxului de documente 15](#_Toc107793586)

[2.1. Generalități 15](#_Toc107793587)

[2.1.1. Soluții similare 15](#_Toc107793588)

[2.1.2. Implementări existente 15](#_Toc107793589)

[2.2. API-uri și componente software relevante 16](#_Toc107793590)

[2.2.2. Generalități 16](#_Toc107793591)

[2.2.3. Mediul integrat de dezvoltare 17](#_Toc107793592)

[2.2.4. Prezentarea componentelor aplicației 17](#_Toc107793593)

[2.2.4.1. Front-End 17](#_Toc107793594)

[2.2.4.2. Back-end 18](#_Toc107793595)

[2.2.4.3. Serverul Web 18](#_Toc107793596)

[2.2.4.4. Baza de Date 18](#_Toc107793597)

[2.2.4.5. Servicii pentru trimiterea mail-urilor 18](#_Toc107793598)

[2.2.4.6. Servicii pentru indexarea documentelor 18](#_Toc107793599)

[3. Implementarea practică 21](#_Toc107793600)

[3.1. Definirea cerințelor complete 21](#_Toc107793601)

[3.2. Prezentarea arhitecturii soluției 22](#_Toc107793602)

[3.3. Cazurile de utilizare 22](#_Toc107793603)

[3.3.1. Potențiali utilizatori. Roluri în cadrul sistemului 22](#_Toc107793604)

[3.3.1.1. Utilizator cu rol de administrator al sistemului 22](#_Toc107793605)

[3.3.1.2. Utilizator cu rol de comandant de subunitate/unitate 24](#_Toc107793606)

[3.3.1.3. Utilizator cu rol de medic specialist 25](#_Toc107793607)

[3.3.1.4. Utilizator cu rol de secretar al universității 26](#_Toc107793608)

[3.3.1.5. Utilizator cu rol de student 27](#_Toc107793609)

[3.3.1.6. Utilizator cu rol de personal civil contractual 28](#_Toc107793610)

[3.3.1.7. Utilizator cu rol de profesor universitar 29](#_Toc107793611)

[3.3.1.8. Utilizator cu rol de vizitator al platformei web 30](#_Toc107793612)

[3.3.2. Diagrame UML 30](#_Toc107793613)

[3.3.2.1. Autentificarea și înregistrarea contului de utilizator 30](#_Toc107793614)

[3.3.2.2. Vizualizarea anunțurilor personale și adăugarea unui nou anunț 32](#_Toc107793615)

[3.3.2.3. Adăugarea și trimiterea unei noi cereri 34](#_Toc107793616)

[3.3.2.4. Vizualizarea propriilor cereri 36](#_Toc107793617)

[3.3.2.5. Arhivare/Indexare cerere 37](#_Toc107793618)

[3.3.2.6. Vizualizarea cererilor arhivate 38](#_Toc107793619)

[3.3.2.7. Căutarea cererilor arhivate în funcție de atribut 40](#_Toc107793620)

[3.3.2.8. Vizualizarea și modificarea structurii grupurilor de utilizatori 40](#_Toc107793621)

[3.3.2.9. Luarea deciziilor 42](#_Toc107793622)

[3.3.2.10. Administrarea modelelor-tip 43](#_Toc107793623)

[3.3.2.11. Fluxuri personalizate 45](#_Toc107793624)

[3.3.2.12. Administrarea conturilor de utilizator 48](#_Toc107793625)

[3.3.2.13. Vizualizarea notificărilor 49](#_Toc107793626)

[3.4. Arhivarea documentelor 50](#_Toc107793627)

[3.4.1. ElasticSearch 50](#_Toc107793628)

[3.4.1.1. Documente 51](#_Toc107793629)

[3.4.1.2. Indici 51](#_Toc107793630)

[3.4.1.3. Index inversat 52](#_Toc107793631)

[3.4.1.4. Tipurile de date 52](#_Toc107793632)

[3.4.2. Implementare 53](#_Toc107793633)

[3.4.3. Formatul obiectelor salvate 55](#_Toc107793634)

[3.5. Mecanisme pentru asigurarea integrității și autenticității documentelor 56](#_Toc107793635)

[*3.5.1.* *Generalități – utilizabilitatea documentelor PDF* 56](#_Toc107793636)

[*3.5.2.* *Semnătura digitală* 56](#_Toc107793637)

[3.5.2.1. Definirea semnăturilor digitale 56](#_Toc107793638)

[3.5.2.2. Importanța semnăturilor digitale 57](#_Toc107793639)

[3.5.2.3. Semnare și verificare semnătură – funcționare 58](#_Toc107793640)

[3.5.2.4. Algoritmul RSA 58](#_Toc107793641)

[3.6. Crearea conturilor de utilizator 62](#_Toc107793642)

[3.6.1. Inregistrarea conturilor de utilizator 62](#_Toc107793643)

[3.6.2. Protocoale utilizate 63](#_Toc107793644)

[3.6.2.1. Simple Mail Transfer Protocol (SMTP) 63](#_Toc107793645)

[3.6.2.2. POP3 (Post Office Protocol 3) 64](#_Toc107793646)

[3.6.2.2.1. Funcționare POP3 64](#_Toc107793647)

[3.6.3. Mercury Mail 65](#_Toc107793648)

[3.6.3.1. Generalități 65](#_Toc107793649)

[3.6.3.2. Configurare Mercury Mail în Windows 65](#_Toc107793650)

[4. Experimete și rezultate ale testării 67](#_Toc107793651)

[4.1. Testarea principalelor funcționalități ale aplicației 67](#_Toc107793652)

[Tabel 1 - Testarea aplicației 72](#_Toc107793653)

[4.2. Probleme întâmpinate 72](#_Toc107793654)

[5. Concluzii 75](#_Toc107793655)

[Anexe 79](#_Toc107793656)

[ANEXA 1 79](#_Toc107793657)

[ANEXA 2 79](#_Toc107793658)

[ANEXA 3 80](#_Toc107793659)

[ANEXA 4 81](#_Toc107793660)

[ANEXA 5 81](#_Toc107793661)

[ANEXA 6 82](#_Toc107793662)

[ANEXA 7 83](#_Toc107793663)

[ANEXA 8 84](#_Toc107793664)

[ANEXA 9 85](#_Toc107793665)

[ANEXA 10 85](#_Toc107793666)

Listă de abrevieri

|  |  |  |
| --- | --- | --- |
| **Nr. crt.** | **Abreviere/Acronim** | **Forma completă** |
|  | API | Application Programming Interface |
|  | EDMS | HyperText Markup Language |
|  | CSS | Cascading Style Sheets |
|  | JS | JavaScript |
|  | HTTPS | Hypertext Transfer Protocol Secure |
|  | SQL | Structured Query Language |
|  | PDF | Portable Document Format |
|  | TXT | Text file |
|  | DOCX | Document Extension |
|  | URL | Uniform Resource Locator |
|  | IT | Information Technology |
|  | UML | Unified Modeling Language |
|  | JSON | JavaScript Object Notation |

Listă de figuri

[Figura 1 - Permisiunile administratorului 23](#_Toc107793301)

[Figura 2 - Permisiunile comandantului 24](#_Toc107793302)

[Figura 3 - Permisiunile medicului specialist 25](#_Toc107793303)

[Figura 4 - Permisiunile secretarului 26](#_Toc107793304)

[Figura 5 - Permisiunile studentului 27](#_Toc107793305)

[Figura 6 - Permisiunile personalului civil contractual 28](#_Toc107793306)

[Figura 7 - Permisiunile profesorului 29](#_Toc107793307)

[Figura 8 - Permisiunile vizitatorului 30](#_Toc107793308)

[Figura 9- Autentificarea și înregistrarea contului de utilizator 32](#_Toc107793309)

[Figura 10 – Vizualizarea anunțurilor personale și adăugarea unui nou anunț 33](#_Toc107793310)

[Figura 11 - Adăugarea și trimiterea unei noi cereri 35](#_Toc107793311)

[Figura 12 - Vizualizarea propriilor cereri 36](#_Toc107793312)

[Figura 13 - Arhivare/Indexare 38](#_Toc107793313)

[Figura 14 - Vizualizare cererilor arhivate 39](#_Toc107793314)

[Figura 15 - Căutarea cererilor arhivate în funcție de atribut 40](#_Toc107793315)

[Figura 16 - Căutarea cererilor arhivate în funcție de atribut 41](#_Toc107793316)

[Figura 17 - Luarea deciziilor 43](#_Toc107793317)

[Figura 18 - Administrarea modelelor-tip 45](#_Toc107793318)

[Figura 19 - Vizualizarea și realizarea fluxurilor personalizate 47](#_Toc107793319)

[Figura 20 - Administrarea conturilor de utilizator 49](#_Toc107793320)

[Figura 21 - Vizualizarea notificărilor 49](#_Toc107793321)

[Figura 22 - Maparea șirurilor în ElasticSearch 52](#_Toc107793322)

[Figura 23 - Interogare SQL 53](#_Toc107793323)

[Figura 24 - Interogare ElasticSearch SQL 53](#_Toc107793324)

[Figura 25 - Formatul obiectelor salvate 55](#_Toc107793325)

[Figura 26- Semnarea și verificarea semnăturii 58](#_Toc107793326)

[Figura 27 -Semnare și verifcare semnătură 59](#_Toc107793327)

[Figura 28 - Algoritmul de semnare RSA 60](#_Toc107793328)

[Figura 29 - Semnarea documentelor 61](#_Toc107793329)

[Figura 30 - Verificarea semnăturii 62](#_Toc107793330)

[Figura 31 - Modul de funcționare al protocolului SMTP pentru transmiterea email-urilor 64](#_Toc107793331)

Listă de tabele

[Tabel 1 - Testarea aplicației 61](#_Toc107327018)

# Introducere

## Introducere în domeniu

Evoluția societății secolului XXI a determinat digitalizarea aproximativ integrală a instituțiilor și companiilor, acestea văzându-se astfel obligate să o folosească în cel mai operant mod cu putință. Acest lucru este fezabil prin investirea în tehnologii atât hardware cât și software, și implicit, prin utilizarea lor în scopuri de eficientizare a serviciilor și gestionare mai eficace a dificultăților întâmpinate.

Domeniul web satisface exemplar cerințele clienților de mai bine de 20 de ani, experimentând o dezvoltare euforică rapidă, un bust economic, o renaștere bazată pe inovație și un progres constant în timp. Reprezentând un mijloc de comunicare și un mediu comercial, web-ul și-a găsit drumul spre dispozitive precum smartphone-uri, tablete sau laptopuri. Termenul anterior menționat a devenit de-a lungul anilor un „catchall” care combină discipline și domenii diferite, fiind utilizat în proiecte personale care vizează experiența creatorului, până la proiecte de nivel amplu pe care instituții mari nu le pot gestiona într-un alt mod mai eficient.

Unul dintre domeniile cu care web-ul se îmbină într-un mod practic este cel criptografic. Criptografia este o modalitate esențială ce asigură secretizarea mesajelor prin criptare și accesul la informație doar celor autorizați. De asemenea, se pune la dispoziție posibilitatea de a semna electronic documente de orice natură pentru a se preveni denaturarea informațiilor conținute de acestea. Un alt aspect important al criptografiei îl reprezintă accesul în mod controlat la resursele rețelei prin parole, pin, mijloace biometrice etc.

Pentru instituțiile care lucrează cu o cantitate substanțială de informație, utilizarea unei soluții de gestionare a acesteia pentru organizare, indexare și control este o necesitate absolută. O variantă practică și modernă pentru managementul fluxului de documente o reprezintă realizarea unui site web, pus la dispoziția angajaților, dar și a managerilor. Această soluție trebuie să asigure, în mod evident, și securizarea informațiilor prin diferite tehnici criptografice.

## Importanța temei alese

De-a lungul anilor, managementul fluxului de lucru s-a dezvoltat și a cunoscut creșteri semnificative, pe măsură ce oamenii s-au concentrat asupra modului în care conceptul ar putea fi redefinit într-un context digital. Această trecere treptată de la fizic la virtual are drept beneficii timpul scurt de îndeplinire a sarcinilor, costuri de fabricație și transport mult mai reduse și, implicit, câștig mult mai mare în cadrul instituțiilor. Dezvoltările în ideea de gestionare a fluxului de lucru au avut loc în mai multe etape. Anterior, s-a pus accent pe gestionarea fluxului de lucru în domeniul industrial prin efectuarea unei analize deliberate și raționale a etapelor de lucru.

Transferul de date și schimbul de informații trebuie facute aproape instant, iar marja de eroare trebuie să fie cât mai mică. De altfel, consolidarea unui sistem electronic de acest gen ar permite organizarea eficientă și corectă a nivelului mare de cereri de permisii din partea studenților. Acest lucru se poate realiza prin modalități de filtrare a datelor în funcție de informația conținută în câmpurile completate de către cursanți, prelucrarea datelor devenind în acest fel mai eficientă și implicând totodată o reducere a timpului de lucru.

Astăzi, orice fel de organizație, indiferent de dimensiunile acesteia, prezintă diferite procese de gestionare și organizare a fluxului de lucru, procese bazate pe management de documente oficiale, algoritmi de lucru variați și regulamente interne personalizate în funcție de natura instituției. Acest flux implică mulți manageri și subordonați, sisteme noi, diferite și complexe, dar și numeroase niveluri de lucru.

## Scopul și obiectivele lucrării

Proiectul își propune să gestioneze într-un mod eficient fluxul de documente dintr-o instituție precum este Academia Tehnică Militară „Ferdinand I”. Se caută așadar o soluție eficientă și modernă care să rezolve eventualele probleme legate de transmiterea dinamică a informațiilor din cadrul unei astfel de organizații. Complicațiile întâmpinate pot fi legate de faptul că orice decizie luată trebuie să fie aprobată de către o persoană avizată, hotărârile adoptate pot avea un timp de așteptare îndelungat sau de faptul că informațiile conținute trebuie securizate în funcție de nivelul de acces al angajaților la acestea.

O altă problemă ce poate interveni în procesul de gestionare a documentelor o reprezintă faptul că există o eșalonare a angajaților atât în structura militară, cât și în cea educațională, la rândul lor având diferite sarcini sau cereri: studenți ce pot solicita măriri ale notelor, permisii, părăsiri de garnizoană ori aprobarea părăsirii de țară; cadre care au nevoie de concedii, reînnoiri de contract sau modificarea contractului de muncă; comandanți de companie ori batalion care au nevoie ca anunțurile făcute să aibă loc rapid, astfel încât ordinele să se execute fără întârziere și mai eficient.

Se dorește ca lucrarea de față să permită tuturor membrilor Academiei Tehnice Militare „Ferdinand I”(studenți, cadre militare, profesori universitari și personal civil) accesul la o platformă multifuncțională de gestionare a fluxului de documente. În concordanță cu rolul îndeplinit în cadrul instituției, fiecare persoană va avea posibilități personalizate astfel: comandanții de batalioane/companii vor avea la dispoziție sisteme de gestionare a fluxului de cereri primite de la studenți, cererile fiind procesate într-o ordine ce depinde de natura și nivelul de urgență al acestora; personalul din cadrul birourilor de secretariat va putea gestiona într-un mod automatizat fluxul de rapoarte, iar transmiterea anunțurilor va fi îmbunătățită.

# Sisteme și standarde în domeniul managementului fluxului de documente

## Generalități

Luând în considerare evoluția societății din ultimii ani, se poate afirma că au apărut sau apar și în prezent diverși factori ce accelerează digitalizarea. Acest fenomen ia amploare prin apariția de noi tehnologii, atât hardware cât și software. Cert este faptul că managementul fluxului de documente reprezintă un aspect fără de care majoritatea instituțiilor nu ar putea funcționa corect și practic, indiferent de natura sau de scopul lor.

Birocrația se impune în orice domeniu, fiind, nu numai un mijloc de diferențiere între nivelurile de angajați, cât și o modalitate de gestionare a traficului de informații în interiorul organizației într-un mod eficient și organizat.

### Soluții similare

Dorința și nevoia de disciplină și sistematizare a condus la aglomerarea pieței cu aplicații și softuri ce tratează problematica organizării și a managementului de informație. Câteva exemple de astfel de soluții web sunt:

* Nintex Process Platform
* Legito
* Bynder
* PandaDoc
* XaitPorter
* Docusoft
* MacroView

### Implementări existente

În continuare se vor prezenta câteva dintre funcționalitățile software-ului Legito. Aplicația este una facilă și ușor de utilizat pentru majoritatea consumatorilor, indiferent de domeniul în care aceștia își desfășoară activitatea. Navigarea pe site și întreg fluxul de documente constă în doar câțiva pași simpli pe care utilizatorul trebuie să îi urmeze pentru încărcarea și trimiterea spre utilizarea ulterioară a acestora.

Câteva dintre facilitățile de care se bucură un utilizator al aplicației anterior menționate sunt:

* Fluxuri de lucru automatizate

Fluxurile de lucru Legito asigură că documentele sunt rutate în mod corespunzător prin rutina specifică fiecăruia, cu toate aprobările necesare. Combinat cu notificări și gestionarea inteligentă a documentelor, software-ul oferă o modalitate rapidă și convenabilă pentru persoanele avizate de a acționa asupra documentelor, după cum este necesar.

Fluxurile de lucru sunt aplicate documentelor, ceea ce înseamnă că pot fi utilizate atât pentru documentele Legito (documente create în Legito din șabloane automate), cât și pentru documentele externe (documente fie încărcate în aplicație sau stocate în afara acesteia).

* Semnătura digitală

Software-ul pune la dispoziția utilizatorului posibilitatea de a semna un document PDF folosind atât un eSignature biometric, obținut printr-o aplicație mobilă ce permite înregistrarea unei semnături de acest tip (amprente, iris, scanare facială etc.), cât și un șablon de semnătură digitală realizat folosind funcția DocuSign.

* Gestionarea aprobării documentelor

Se asigură faptul că documentele ajung la persoanele potrivite pentru aprobare. Folosind setări avansate se creează aprobări dinamice, astfel scade riscul de apariție a blocajelor și implicit se reduce durata petrecută de un document în fluxul de lucru.

* Notificări pentru termene limită

Folosirea notificărilor cu termen limită ale aplicației ajută utilizatorul să îndeplinească target-urile asignate. Utilizarea datelor extrase automat din documente Legito sau setarea manuală a termenelor asigură finalizarea la timp a tuturor sarcinilor. Există, de altfel, posibilitatea de a primi notificări în aplicație sau prin e-mail.

## API-uri și componente software relevante

### Generalități

Capitolul curent prezintă și exemplifică mediile de lucru, tehnologiile și limbajele de programare folosite pentru crearea aplicației.

O pagină web este un aspect foarte important la nivel de internet, site-ul web reprezentând, în contemporaneitate, „cartea de vizită” a aproximativ oricărei organizații. Pe această cale, instituțiile își pot valorifica mai ușor obiectul de activitate, principiile, produsele, serviciile, portofoliul sau datele de contact cu ajutorul unui layout grafic atractiv.

Un site web este o colecție de documente, numite pagini. Acestea nu sunt independente unele față de celelalte, ci sunt legate prin sistemul de navigare în cadrul site-ului. Paginile sunt create cu ajutoru limbajului HTML, dar totuși personalizate prin multe alte tehnologii. Cu ajutorul limbajului anterior menționat pot fi incluse într-o pagină web: texte, imagini, legături către alte pagini din site sau către web, liste, tabele și formulare.

### Mediul integrat de dezvoltare

Visual Studio Code este un editor de cod sursă realizat de Microsoft pentru Windows , Linux și macOS. Poate fi utilizat cu o varietate de limbaje de programare, inclusiv Java , JavaScript , Go , Node.js , Python și C ++ . Se bazează pe framework-ul Electron , care este utilizat pentru a dezvolta aplicații web Node.js. Aplicațiile rulează pe motorul Blink.

Out-of-the-box, Visual Studio Code este un suport de bază pentru cele mai comune limbaje de programare. Acest suport de bază include: evidențierea sintaxei, potrivirea parantezelor, plierea codului și fragmente configurabile. Visual Studio Code este livrat și cu IntelliSense pentru JavaScript, TypeScript, JSON , CSS și HTML, precum și suport pentru depanare pentru Node.js. Suportul pentru limbaje suplimentare poate fi oferit de extensiile disponibile gratuit pe piața VS Code.

### Prezentarea componentelor aplicației

Front-End și Back-End sunt cei mai populari doi termeni utilizați în dezvoltarea web. Aceștia sunt esențiali, dar, în principiu, sunt diferiți unul de celălalt. Fiecare parte trebuie să comunice și să funcționeze eficient cu cealaltă ca o singură unitate pentru a îmbunătăți funcționalitatea site-ului web.

#### Front-End

Partea „client-side” a unui site web cu care utilizatorul interacționează direct este denumită “front-end”. Include tot ceea ce utilizatorii experimentează direct: culori și stiluri de text, imagini, grafice și tabele, butoane, culori și meniu de navigare.

Front-End-ul are două părți: design-ul (partea creativă) și dezvoltarea interfeței (partea de cod). HTML, CSS și JavaScript sunt limbajele utilizate pentru dezvoltare. Din punct de vedere al front-end-ului, dezvoltatorul trebuie să se asigure și de faptul că site-ul este „responsive”, adică apare corect pe dispozitive de toate dimensiunile. Site-ului web nu trebuie să se comporte anormal, indiferent de dimensiunea ecranului.

**HTML**

HTML înseamnă „Hypertext Markup Language”. Este folosit pentru a proiecta porțiunea front-end a paginilor web utilizând un limbaj de marcare. HTML este combinația dintre hipertext și limbaj de markup. Hipertextul definește legătura dintre paginile web. Limbajul de marcare este utilizat pentru a defini documentația text din cadrul etichetei care definește structura paginilor web.

**CSS**

Cascading Style Sheets, prescurtat „CSS”, este un limbaj simplu conceput pentru a simplifica procesul de prezentare a paginilor web. CSS permite aplicarea de stiluri paginilor web, independent de HTML-ul care alcătuiește fiecare pagină web. Stilurile se pot atașa elementelor HTML prin intermediul unor fișiere externe sau în cadrul documentului, prin elementul <style> și/sau atributul style.

**JavaScript**

JavaScript (JS) este un limbaj de programare ușor, interpretat sau „just-in-time compiled”. Deși este cel mai bine cunoscut sub numele de limbaj de scriptare pentru paginile web, multe medii non-browser îl folosesc: Node.js, Apache CouchDB și Adobe Acrobat. JavaScript este un limbaj dinamic bazat pe prototip, multi-paradigmă, „single-threaded”, care acceptă stiluri orientate pe obiecte, imperative și declarative (de exemplu, programare funcțională).

#### Back-end

Back-End-ul de obicei constă în trei părți: un server, o aplicație de interfață și o bază de date. Este, de asemenea, acea parte „server-side” prin intermediul căreia se stochează și se organizează datele. Se asigură, de altfel, că tot ce vine din partea de “client-side” funcționează corespunzător.

#### Serverul Web

#### Baza de Date

**MySql**

Cea mai recentă versiune MySQL este una dintre cele mai populare baze de date din lume. Datorită performanței sale dovedite, fiabilității,compatibilității cu toți furnizorii principali de găzduire și ușurinței de utilizare, MySQL a devenit alegerea principală a bazei de date pentru aplicații bazate pe web, utilizate de proprietăți web de profil înalt, inclusiv Facebook, Twitter, YouTube sau Yahoo!.

#### Servicii pentru trimiterea mail-urilor

**XAMPP**

XAMPP este un pachet de stivă de soluții web multiplatformă gratuit și open source dezvoltat de Apache Friends, constând în principal din Apache HTTP Server, baza de date MariaDB și interpreți pentru scripturi scrise în limbajele de programare PHP și Perl.

**MERCURY MAIL**

#### Servicii pentru indexarea documentelor

**Docker**

Docker, un subset al proiectului Moby, este un cadru software pentru construirea, rularea și gestionarea containerelor pe servere și în cloud. Termenul „docker” se poate referi fie la instrumente (comenzile și un demon), fie la formatul de fișier Dockerfile.

**Docker Desktop**

Docker Desktop este o aplicație ușor de instalat pentru mediul Mac sau Windows, care permite construirea și partajarea aplicațiilor și microserviciilor containerizate.

Acesta funcționează cu instrumentele și limbajele de dezvoltare alese și oferă acces la o bibliotecă vastă de imagini și șabloane certificate în Docker Hub.

Unele dintre caracteristicile cheie ale Docker Desktop includ:

* containerizarea și partajarea aplicațiilor pe orice platformă Cloud
* instalare și configurare ușoare
* includerea celei mai recente versiuni de Kubernetes
* performanță rapidă și fiabilă
* montarea volumului pentru cod și date, inclusiv notificări de modificare a fișierelor și acces ușor la containerele care rulează în rețeaua localhost [1]

**ElasticSearch**

Elasticsearch este un motor de căutare bazat pe biblioteca Lucene. Oferă un motor de căutare full-text distribuit, cu o interfață web HTTP și documente JSON schema-free. Clienții oficiali sunt disponibili în Java, .NET (C#), PHP, Python, Apache Groovy, Ruby și multe alte limbaje. Conform clasamentului DB-Engines, Elasticsearch este cel mai popular motor de căutare pentru întreprinderi. [2]

# 

# 

# Implementarea practică

## Definirea cerințelor complete

Tema de proiect se înscrie în tematica dezvoltării software de sisteme informatice și propune realizarea unei soluții web pentru gestiunea și manipularea documentelor uzuale existente la nivelul unei organizații folosind metode exclusiv electronice.

Soluția proiectată va asigura următoarele cerințe tehnice specifice:

* posibilitatea de încărcare în sistem a unor documente electronice ( ex. rapoarte, anunțuri) pentru a fi apoi distribuite electronic către alți utilizatori cu rol de decizie
* la încărcarea documentului, utilizatorul trebuie să poată selecta un anumit flux de lucru pentru document, acțiuni posibile sau necesare la nivelul documentului
* soluția va permite încărcarea de documente în formatele de prezentare actuale cunoscute(PDF, docx, pptx etc.)
* implementarea unui dashboard curent de lucru pentru fiecare utilizator. Acesta trebuie să prezinte documentele de interes având caracter de noutate la nivelul sistemului, prezentarea unui status curent al documentului precum și acțiunile necesare/permise la nivelul documentului(vizualizare, aprobare, aprobare cu semnare electronică, rejectare etc.)
* implementarea mai multor operații posibile ce pot fi aplicate pe flux asupra conținutului documentelor.
* realizarea operațiilor de implementare a dinamicii de parcurs pentru documentele încărcate la nivelul sistemului(parcurgerea documentelor pe flux și în ordine)
* funcționalități de definire și utilizare de șabloane predefinite pentru fluxuri de lucru la nivelul organizației dar și posibilități de definire de fluxuri stabilite ad-hoc la momentul încărcăriii documentelor
* Permiterea de funcționalități de arhivare a documentelor marcate ca fiind finalizate. Pentru arhivare, soluția va integra componente de indexare( ex. tip ElasticSearch)
* implementarea unui modul(ex. web-service extern) pentru integrarea unor funcționalități de semnătură electronică necesară la o anumită acțiune asupra documentului
* implementarea unui modul de management al utilizatorilor și rolurilor definite la nivelul sistemului

## Prezentarea arhitecturii soluției

## Cazurile de utilizare

### Potențiali utilizatori. Roluri în cadrul sistemului

Fiind un sistem complex, realizat pentru organizații de dimensiuni mari, în cadrul cărora lucrează o mare masă de angajați și în cadrul cărora se realizează foarte multe tipuri de acțiuni, capitolul curent este menit să facă o trecere în revistă a tuturor tipurilor de utilizatori. Se vor prezenta denumirile tuturor tipurilor de roluri ale acestora, împreună cu o descriere detaliată a acțiunilor pe care aceștia le pot întreprinde în cadrul aplicației.

Astfel, în cazul în care un vizitator al site-ului web intenționează să își creeze un cont, acesta trebuie să știe cu exactitate care este rolul său în cadrul organizației, ce permisiuni are și implicit ce acțiuni poate realiza, atât pe platformă, cât și în mediul de desfășurare a activității profesionale.

Fiecare rol de utilizator are atribuit un tip de permisiune. Împărțirea în mai multe tipuri de permisiuni oferă o administrare corectă a accesului la conținutul și la înțelegerea sistemului, precizând tipul de acțiuni și operații care ar putea fi realizate de fiecare în parte. În funcție de rolul setat, utilizatorii au următoarele drepturi sau restricții asupra fiecărei funcționalități sau element de conținut specificat:

#### Utilizator cu rol de administrator al sistemului

Administratorul sistemului este cea mai importantă entitate din cadrul sistemului, având cele mai multe roluri de decizie în cadrul acestuia. Autorizările lui sunt comune cu cele ale profesorilor, secretarilor sau comandanților de subunități, iar acestea vor fi explicate și exemplificate ulterior.

El are, în plus față de alți utilizatori, responsabilitatea de a gestiona conturile celorlalți participanți din cadrul organizației, de a modifica structura paginii destinare șabloanelor și modelelor-tip, de a modifica structura grupurilor de utilizatori prin crearea de noi grupuri și mutarea utilizatorilor în interiorul acestora.

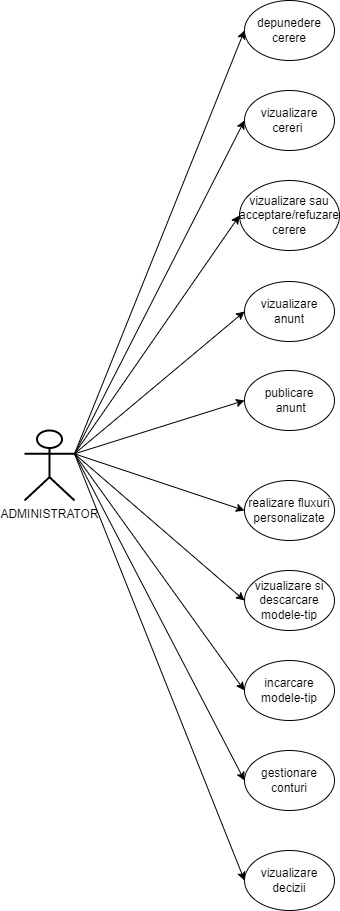


Figura 1 - Permisiunile administratorului

#### Utilizator cu rol de comandant de subunitate/unitate

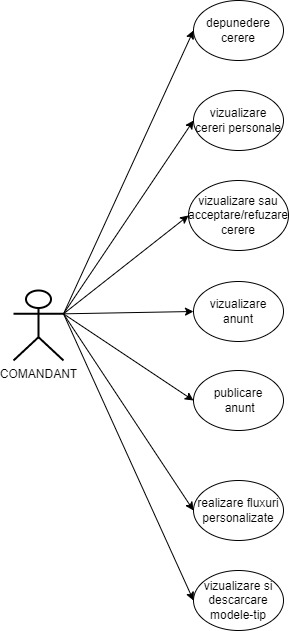


Figura 2 - Permisiunile comandantului

Utilizatorii din această categorie sunt de 2 tipuri

* comandantul întregii unități
* comandanții de subunități, aflați, la rândul lor, în subordinea comandantului unității

Această categorie din rândul potențialilor utilizatori ai sistemului deține permisiuni precum:

* Depunerea cererilor, în special cele legate de activitatea profesională, cum ar fi: raport pentru părăsirea țării (în cazul în care acesta este cadru militar), raport medical, raport pentru concediu etc.
* Vizualizarea anunțurilor legate de activitatea profesională
* Vizualizarea cererilor adresate lui, precum rapoartele de părăsire a garnizoanei a studenților, rapoartele lor de învoire, rapoarte de descazarmare etc.
* Acceptarea sau refuzarea cererilor adresate lui din motive legale
* Publicarea anunțurilor, în special pentru studenți și subalterni (un exemplu de document pe care un comandant de subunitate îl poate încărca în cadrul sistemului este unul legat de schimbarea ordinelor de batalion dintr-o zi stabilită de acesta)
* Realizarea fluxurilor personalizate și vizualizarea sau descărcarea modelelor tip se realizează în aceeași manieră ca în cazul celorlalte tipuri de utilizatori
* Înaintarea rapoartelor subalternilor săi către superiori, în cazul în care situația necesită această acțiune

#### Utilizator cu rol de medic specialist

Autorizările medicului specialist sunt simple, deoarece nici acest tip de utilizator nu poate modifica sau influența într-o foarte mare măsură fluxul de documente în cadrul organizației.

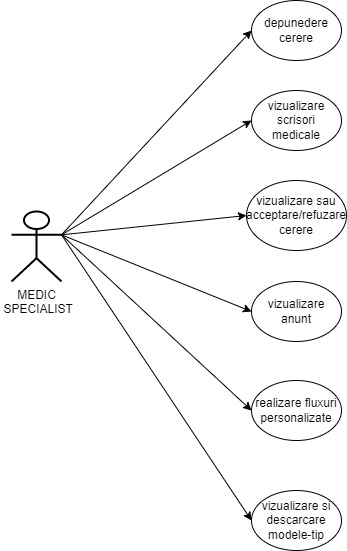


Figura 3 - Permisiunile medicului specialist

Singurele autorizări ale acestuia sunt:

* Depunerea cererilor și rapoartelor legate de activitatea sa profesională, cereri adresate în special superiorilor săi
* Vizualizarea si luarea deciziilor asupra rapoartelor medicale trimise acestuia de către celelalte categorii de utilizatori, în scopul obținerii permisiunii de a primi concedii medicale, învoiri sau orice permisiune de natură medicală
* Vizualizarea anunțurilor din sfera sa de interes
* Realizarea fluxurilor personalizate în vederea fluidizării procesului de transmisie a unui document
* Vizualizarea și descărcarea modelelor-tip, tot în scopul fluidizării anterior menționate

#### Utilizator cu rol de secretar al universității

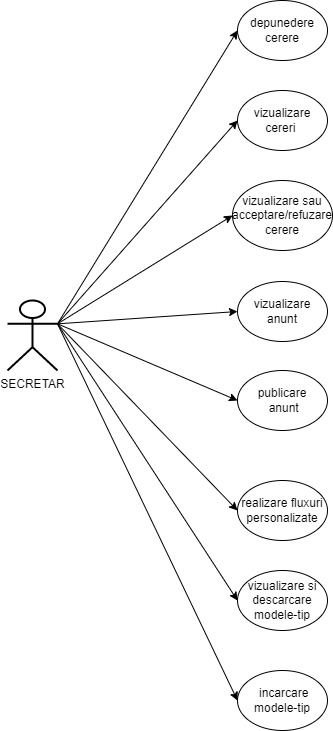


Figura 4 - Permisiunile secretarului

Autorizările pe care le poate avea un secretar sunt următoarele:

* Depunerea cererilor, în special cele legate de activitatea profesională, cum ar fi: raport pentru părăsirea țării (în cazul în care acesta este cadru militar), raport medical, raport pentru concediu etc.
* Vizualizarea anunțurilor legate de activitatea profesională
* Vizualizarea cererilor adresate lui, precum rapoartele pentru adeverințele pentru studenți
* Acceptarea sau refuzarea cererilor adresate lui din motive legale
* Publicarea anunțurilor, în special pentru studenți și profesori (un exemplu de document pe care un secretar îl poate încărca în cadrul sistemului este unul legat de planificarea unui anumit examen, din punct de vedere al datei și sălii în care se va ține examenul respectiv)
* Realizarea fluxurilor personalizate și vizualizarea sau descărcarea modelelor tip se realizează în aceeași manieră ca în cazul celorlalte tipuri de utilizatori
* Asemenea administratorului, și secretarul este autorizat să încarce modele de documente tip în sistem

#### Utilizator cu rol de student

Studenții nu au permisiuni legate de gestionarea resurselor în cadrul organizației deoarece aceștia nu pot influența fluxul de documente/cereri/rapoarte și nu pot lua decizii asupra funcționării activităților unității.

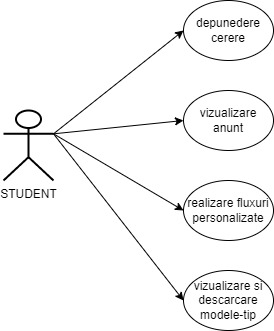


Figura 5 - Permisiunile studentului

Singurele permisiuni pe care le are acest tip de utilizator sunt:

* Depunerea cererilor de tip raport medical, reexaminare, mărire a notei, contestație, învoire, părăsire garnizoană etc.
* Vizualizarea anunțurilor legate de activitatea universitară sau militară
* Realizarea fluxurilor personalizate, asemenea tuturor celorlalte tipuri de utilizatori
* Vizualizarea și descărcarea modelelor tip, în scopul realizării corecte a rapoartelor

#### Utilizator cu rol de personal civil contractual

Personalul civil contractual reprezintă utilizatorul care nu deține comanda niciuneia dintre subunitățile administrației și deci nu deține responsabilitatea față de nicio altă persoană înscrisă în cadrul platformei. Interesele acestuia în cadrul posibilei organizații militare care deține aplicația curentă sunt aceleași cu cele ale unui angajat obișnuit care își desfășoară activitatea într-o organizație civilă.

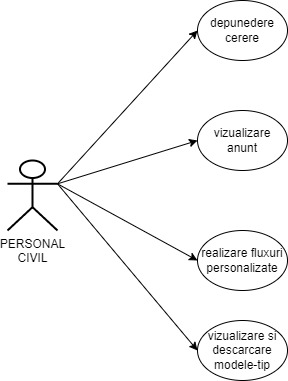


Figura 6 - Permisiunile personalului civil contractual

Așadar, autorizările pe care le deține un astfel de utilizator sunt următoarele:

* Depunerea cererilor către superiorii săi (un exemplu de cerere pe care o poate înscrie în sistem este cererea de concediu de orice natură)
* Vizualizarea anunțurilor adresate lui (un exemplu de anunț care poate viza un angajat civil este o înștiințare legată de zilele legale libere)
* Realizarea fluxurilor personalizate se face în aceeași manieră ca în cazul celorlalți utilizatori

#### Utilizator cu rol de profesor universitar

Profesorii universitari pot fi atât cadre militare, cât și angajați civili. Din acest motiv, pot efectua operații în cadrul aplicației din ambele perspective.

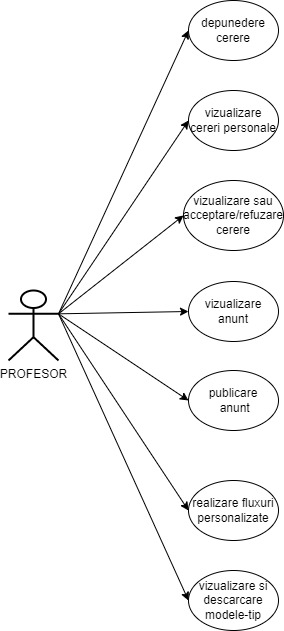


Figura 7 - Permisiunile profesorului

Autorizările pe care le poate avea un profesor sunt următoarele:

* Depunerea cererilor, în special cele legate de activitatea profesională, cum ar fi: raport pentru părăsirea țării (în cazul în care acesta este cadru militar), raport medical, raport pentru concediu etc.
* Vizualizarea anunțurilor legate de activitatea profesională
* Vizualizarea cererilor adresate lui, precum cererile de restanță sau de reexaminare în vederea măririi notei de la studenți
* Acceptarea sau refuzarea cererilor adresate lui din motive legale
* Publicarea anunțurilor, în special pentru studenți (un exemplu de document pe care un profesor îl poate încărca în cadrul sistemului este unul legat de regulile de desfășurare al unui examen sau de data stabilită pentru examen)
* Realizarea fluxurilor personalizate și vizualizarea sau descărcarea modelelor tip se realizează în aceeași manieră ca în cazul celorlalte tipuri de utilizatori

#### Utilizator cu rol de vizitator al platformei web

Vizitatorul aplicației este cel mai simplu tip de utilizator, având doar o singură permisiune în cadrul acesteia. O persoană deține rolul de vizitator în două cazuri:

* nu se află în baza de date a sistemului, nefăcând parte din cadrul organizației căreia îi servește acesta
* face parte din organizație, însă nu deține încă un cont de utilizator din diverse motive

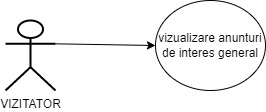


Figura 8 - Permisiunile vizitatorului

Entitatea care deține acest rol este autorizată doar cu vizualizarea anunțurilor care prezintă interes general, având însă posibilitatea de realizare și a altor operațiuni, însă nu înainte de a depune către administrator o cerere de înregistrare a contului.

### Diagrame UML

Diagramele UML au fost create în scopul înțelegerii fluxului aplicației prin prezentarea pe scurt și într-un limbaj non-tehnic stările sistemului, modul de funcționare și unele scenarii.

Aceste diagrame reprezintă o modalitate standard eficientă de a desena modelele software, de a schița proiectările sistemului și de îl documenta într-o manieră clară și concisă, perceptibilă de către potențiali clienți ai unei astfel de aplicații .

#### Autentificarea și înregistrarea contului de utilizator

Autentificarea reprezintă un proces prin intermediul căruia se poate determina dacă entitatea care inițiază procesul acesta este, de fapt, cine sau ce se declară a fi.

Tehnologia de autentificare pune la dispoziție modalități de verificare a acreditărilor unui utilizator. Astfel de verificări constau, spre exemplu, în compararea credențialelor introduse cu conținutul bazelor date sau al serverelor de autentificare a datelor.

Autentificarea este mecanismul cel mai important în cadrul organizațiilor, mai ales în ceea ce privește păstrarea rețelelor securizate sau permiterea doar utilizatorilor autorizați să acceseze resursele protejate ale acesteia.

De cele mai multe ori autentificarea este ușor de confundat cu autorizarea, un alt termen de securitate. Ambii termeni sunt folosiți de obicei ca sinonime, dar sunt diferite ca procese. Autentificarea este un proces de dinaintea mecanismului de autorizare. Autentificarea verifică identitatea utilizatorului, autorizarea verifică dacă utilizatorul curent are permisiunile corecte și drepturile necesare pentru a accesa resursele solicitate. [1]

În cadrul proiectului curent, procesul de autentificare se desfășoară pe parcursul câtorva pași simpli, după cum urmează:

* un utilizator introduce o serie de credențiale, precum numele utilizator sau adresa de email și parola, acestea fiind în măsură să îl identifice ca entitate a sistemului sau serverului pe care încearcă să îl acceseze
* credențiale oferite sistemului sunt verificate în cadrul bazei de date a aplicației sau a site-ului web pentru care utilizatorul se autentifică.
* dacă se trece cu succes de mecanismele de verificare, iar credențialele corespunzând conținutului bazei de date, conectarea utilizatorului la sistem este aprobată, acesta din urmă primind acces la resursele sistemului în funcție de permisiunile sale
* înaintea unei autentificări, utilizatorul de tip vizitator trebuie să se înregistreze în cadrul aplicației prin furnizarea unei cereri către administrator
* sistemul este destinat unor organizații sau unități precum cele militare, fapt pentru care înregistrarea a unui cont se realizează prin trimiterea către administrator a unei cereri ce conține o serie de detalii unice ce vor alcătui ulterior profilul utilizatorului
* aceste detalii unice pe care utilizatorul le introduce în cadrul interfeței urmează, după aprobarea de către administratorul sistemului, să formeze contul utilizatorului pentru accesul la resursele aplicației
* după aprobarea sau respingerea cererii de realizare a contului de utilizator, acestuia din urmă i se vor trimite notificările aferente prin intermediul unui serviciu de email, precum Yahoo Mail sau Gmail
* procesul de modificare a credențialelor unui cont se desfășoară în mod similar cu cel de înregistrare a acestuia, urmându-se aceiași pași de trimitere cerere - primire răspuns

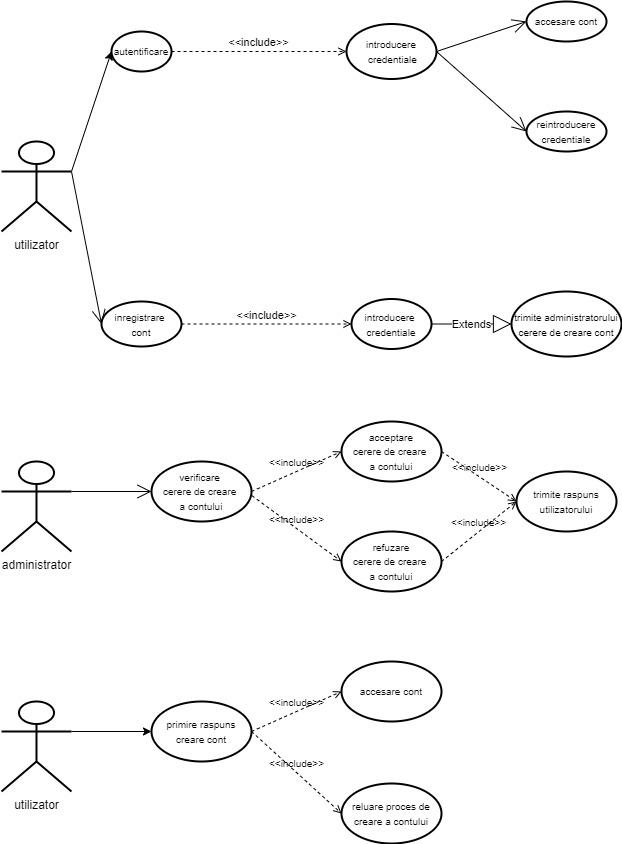


Figura 9- Autentificarea și înregistrarea contului de utilizator

#### Vizualizarea anunțurilor personale și adăugarea unui nou anunț

Proiectul propune o soluție de gestionare a traficului și fluxului de documente, iar unul dintre tipurile de fluxuri propuse către analizare și îmbunătățire este încărcarea, completarea, transmiterea și vizualizarea anunțurilor.

Acestea sunt documente destinate informării unuia sau mai multor utilizatori cu privire la motivele invocate de către autorii anunțurilor.

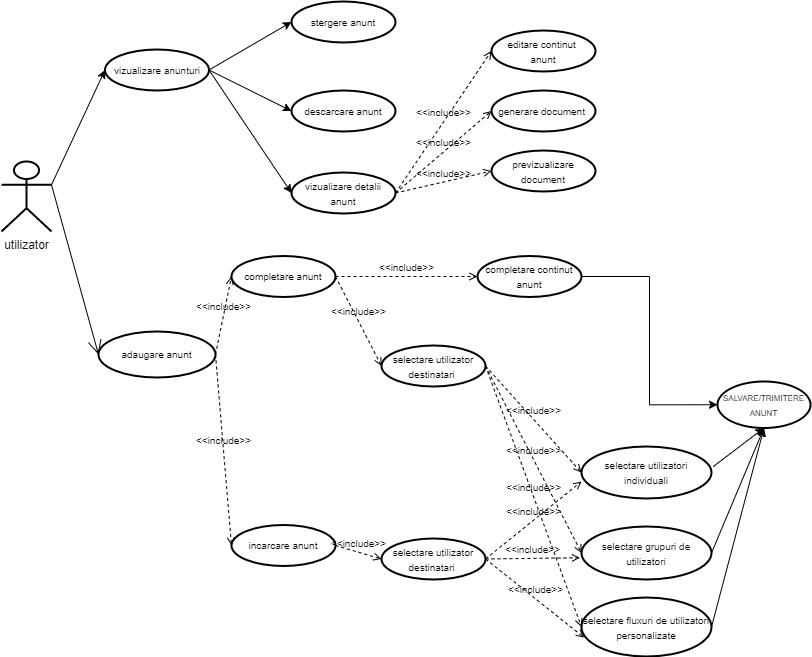


Figura 10 – Vizualizarea anunțurilor personale și adăugarea unui nou anunț

Procesul de vizualizare a unui anunț se desfășoară astfel:

* se pun la dispoziția utilizatorului toate anunțurile destinate acestuia, prezentate sub formă tabelară
* utilizatorul are posibilitatea de a accesa unul dintre aceste documente
* accesarea anterior menționată presupune mai multe tipuri de operații care se pot desfășura asupra anunțului selectat, operații precum:
  + vizualizarea documentului
  + vizualizarea detaliilor documentului(autorul anunțului, data încărcării acestuia în sistem, conținutul documentului etc.)
  + ștergerea anunțurilor care nu prezintă interes
* în cazul în care utilizatorul accesează un anunț propriu, mai are la dispoziție operații precum:
  + editarea conținutului documentului
  + generarea documentului pe baza informațiilor introduse în sistem
  + pre-vizualizarea documentului care urmează să fie generat

Procesul de adăugare a unui anunț se desfășoară astfel:

* utilizatorul are posibilitatea de a completa un anunț prin intermediul interfeței, dar și posibilitatea încărcării acestuia sub diferite formate precum docx, pdf etc.
* completarea anunțului se realizează prin trimiterea către server a detaliilor aferente acestui tip de document, precum titlu, conținut, dată, dar și prin selectarea utilizatorilor cărora le este destinat anunțul
* selectarea utilizatorilor țintă se realizează prin alegerea uneia dintre cele trei opțiuni disponibile:
  + selectarea unui grup de utilizatori, anunțul fiind destinat utilizatorilor care fac parte din grupul selectat (un exemplu de grup este Plutonul 2 din Compania 1 Studenți)
  + selectarea individuală a unui număr de utilizatori
  + selectarea unui flux personalizat, creat anterior în cadrul unei alte secțiuni ce urmează să fie exemplificată și detaliată
* încărcarea anunțului se realizează prin trimiterea către server a detaliilor aferente, mai puține de această dată, întrucât informațiile de interes general vor fi conținute în cadrul documentului
* astfel, se vor completa doar detalii precum titlul anunțului și utilizatorii destinatari ai acestuia
* modalitatea de selectare a utilizatorilor la care trebuie să ajungă documentul este aceeași ca cea menționată anterior

#### Adăugarea și trimiterea unei noi cereri

Fluxul de cereri din cadrul sistemului constă într-un proces de trimitere a documentului în vederea primirii înapoi a unui răspuns. Răspunsul poate fi negativ în cazul în care fluxul conduce la respingerea parțială sau totală a cererii respective sau poate fi pozitiv în cazul în care toți sau o parte dintre superiori, în funcție de drepturile de decizie ale acestora, au acceptat-o.

Drepturile de decizie a utilizatorilor se pot stabili în ordine ierarhică, în ordinea stabilită de către administratorul platformei în momentul creării unui cont de utilizator sau în momentul creării unul flux personalizat.

Există, de altfel, posibilitatea modificării priorității de către autorul raportului sau a cererii, prin inversarea ordinii în cadrul fluxurilor create de acesta anterior.

Prioritatea unui superior în luarea deciziilor asupra unui document constă, pe de-o parte, în poziția pe care o ocupă în cadrul fluxului de transmitere a documentului sau, pe de altă parte, în posibilitatea întreruperii acestui flux.

Astfel, daca unul dintr-un număr limitat de destinatari ai cererii ocupă o poziție superioară în ierarhia stabilită, acesta poate lua decizia finală asupra cererii, modificând statusul final cu propria sa decizie. Însă, dacă un destinatar care ocupă o poziție inferioară altora are posibilitatea întreruperii fluxului, cererea nu va mai ajunge la restul superiorilor.

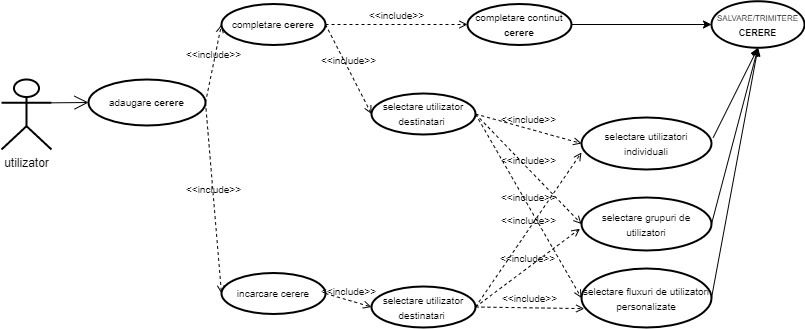


Figura 11 - Adăugarea și trimiterea unei noi cereri

Cererile disponibile în cadrul implementării curente sunt de mai multe tipuri, astfel:

* învoire;
* permisie;
* raport medical;
* concediu;
* restanță;
* mărire a notelor;
* introducere dispozitiv în unitate;
* părăsire țară;
* adeverință student;
* emitere documente;

Procesul de depunere a unei cereri se desfășoară astfel:

* utilizatorul are posibilitatea de a completa o cerere prin intermediul interfeței, dar și posibilitatea încărcării acesteia sub diferite formate precum docx, pdf etc.
* completarea cererii se realizează prin trimiterea către server a detaliilor aferente acestui tip de document, precum:
  + selectarea tipului de cerere care urmează apoi a fi generată și transmisă către destinatar
  + data de început și de sfârșit a acțiunii care urmează să fie realizată de către autorul raportului
  + localitatea, județul și intervalul orar necesare desfășurării activităților pentru care se face această cerere
  + motivul legal al raportului și argumentarea acestuia
  + modalitatea de deplasare a utilizatorului în locația sau locațiile specificate
* selectarea utilizatorilor țintă se realizează prin alegerea uneia dintre cele trei opțiuni disponibile:
  + selectarea unui grup de utilizatori, anunțul fiind destinat utilizatorilor care fac parte din grupul selectat (un exemplu de grup este Plutonul 2 din Compania 1 Studenți)
  + selectarea individuală a unui număr de utilizatori
  + selectarea unui flux personalizat, creat anterior în cadrul unei alte secțiuni ce urmează să fie exemplificată și detaliată
* **în plus** față de secțiunea de anunțuri este posibilitatea selectării unui flux IMPLICIT care constă în trimiterea documentului către grupul inițial din care face parte utilizatorul
* grupul inițial se stabilește de către utilizator în momentul creării contului sau se poate stabili ulterior de către administratorul platformei
* încărcarea cererii se realizează prin trimiterea către server a detaliilor aferente, mai puține de această dată, întrucât informațiile de interes general vor fi conținute în cadrul documentului
* astfel, se vor completa doar detalii precum tipul de cerere, în vederea filtrării ulterioare într-un mod mai eficient
* modalitatea de selectare a utilizatorilor la care trebuie să ajungă documentul este aceeași ca cea menționată anterior

#### Vizualizarea propriilor cereri

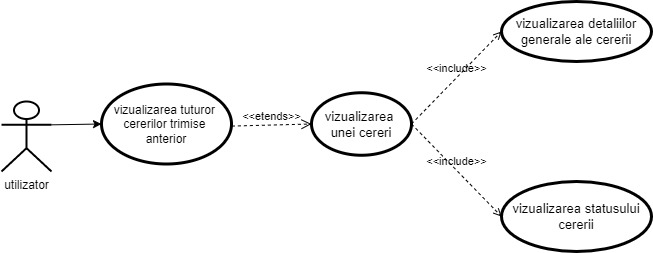


Figura 12 - Vizualizarea propriilor cereri

Orice utilizator deține sau, dacă nu, își creează pe parcurs un istoric al transmiterii documentului. Având posibilitatea trimiterii sau recepționării unui număr considerabil de cereri și rapoarte, este util ca acestea să poată fi păstrate și centralizate pentru manevrarea lor mai ușoară.

Așadar, oricărei categorii de utilizator îi este pusă la dispoziție vizualizarea documentelor încărcate anterior, cu diferite scopuri, sub formă tabelară. În contextul lizibilității, cererile pot fi filtrate în funcție de categoria din care fac parte. Deci, este realizat scenariul în care utilizatorul poate vizualiza doar cererile de tip „învoire”.

Filtrarea documentelor, pe lângă cea prezentată anterior, se mai poate realiza și în funcție de statusul „ARHIVAT” sau „CURENT”. Din categoria documentelor arhivate pot face parte:

* documentele marcate ca fiind finalizate, adică documentele ale căror statusuri finale sunt „ACCEPTAT” sau „REFUZAT”;
* documentele care au fost încărcate la o dată, iar aceasta a depășit data limită stabilită de către utilizator;

După alegerea unei cereri din cadrul tabelului, interfața afișează, tot sub formă tabelară, detalii generale ale selecției anterioare, dar cu posibilitatea vizualizării statusului curent al cererii.

#### Arhivare/Indexare cerere

Interfața oferă utilizatorilor, indiferent de categoria din care fac parte aceștia, opțiuni de selectare a unuia dintre documentele personale care se află încă în proces de aprobare/respingere sau a documentelor al căror flux de transmisie s-a încheiat.

Vizualizarea statusului unei cereri, imediat după selectarea individuală a acesteia, presupune atât posibilitatea vizualizării câtorva detalii specifice ale fluxului documentului, cât și posibilitatea de arhivare a documentului marcat ca „FINALIZAT”.

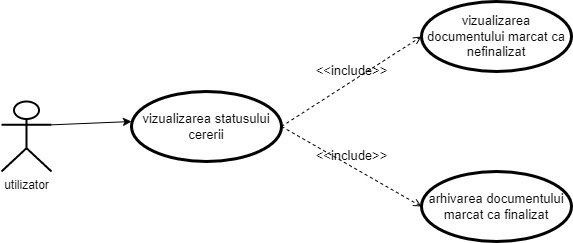


Figura 13 - Arhivare/Indexare

Procesul de arhivare/indexare a unei cereri se desfășoară astfel:

* utilizatorul accesează secțiunea de cereri, afișate sub formă tabelară
* urmează apoi selectarea uneia dintre cererile puse la dispoziție de către sistem, cereri care fac parte din istoricul acțiunilor utilizatorului în cadrul platformei
* prin accesarea butonului VIZUALIZARE intervin posibilități de vizualizare a detaliilor documentului marcat ca finalizat, dar și posibilități de arhivare a acestuia
* documentele astfel arhivate pot fi vizualizate tot din secțiunea de cereri, prin intermediul butonului ARHIVATE
* butonul anterior menționat are doar rol de filtrare a tuturor cererilor, făcând diferența între cele curente, cu un flux de transmisie încă activ, și între documentele finalizate și arhivate
* de altfel, vizualizarea și filtrarea documentelor finalizate se poate realiza și în cadrul unei alte secțiuni, Cereri arhivate, iar cazurile de utilizare pentru acest scenariu vor fi explicate într-un subcapitol viitor

#### Vizualizarea cererilor arhivate

Arhivarea cererilor are ca scop filtrarea și eliberarea tabelelor de vizualizare a documentelor în proces de transmitere, în scopul lizibilității și a unei căutări ulterioare eficiente. Sistemul pune la dispoziție secțiunea intitulată Cereri arhivate.

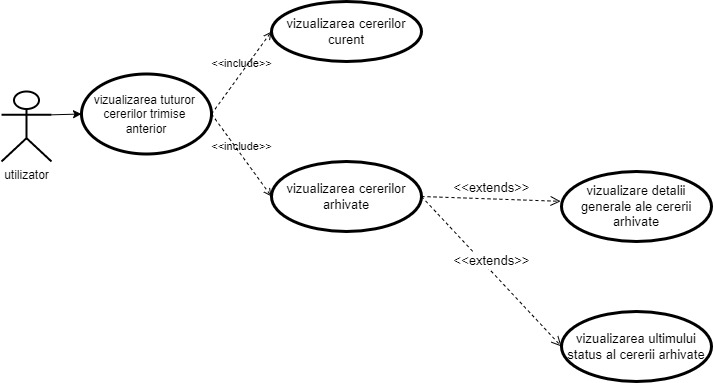


Figura 14 - Vizualizare cererilor arhivate

Filtrarea și căutarea documentelor se realizează în felul următor:

* în momentul accesării paginii web menționate se vor încărca și afișa toate cererile care au fost arhivate anterior (din motivul statusului final ACCEPTAT/REFUZAT sau din motivul depășirii datei stabilite de către administratorul platformei)
* cererile încărcate automat în momentul accesării paginii pot fi vizualizate, la fel ca în cazul documentelor încă în proces de transmitere, astfel
  + selectarea uneia dintre cererile puse la dispoziție de către sistem, cereri care fac parte din istoricul acțiunilor utilizatorului în cadrul platformei și care au fost arhivate din unul dintre motivele explicate în cadrul subcapitolului pentru arhivarea documentelor
  + prin accesarea butonului VIZUALIZARE intervin posibilități de vizualizare a detaliilor documentului marcat ca finalizat, dar și posibilități de vizualizare a ultimului status al acestuia, înainte de a fi arhivat
  + diferența de această dată, față de vizualizarea documentelor încă nefinalizate este, din punct de vedere funcțional, dar și din punct de vedere al design-ului, că butonul pentru arhivare nu mai apare vizibil pentru documentele deja arhivate

#### Căutarea cererilor arhivate în funcție de atribut

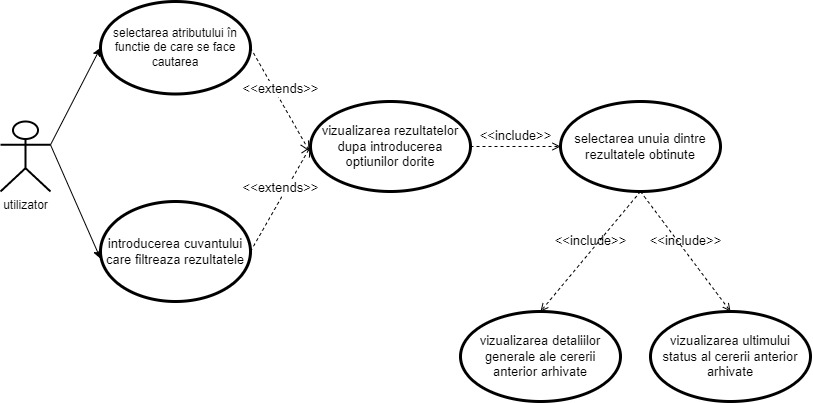


Figura 15 - Căutarea cererilor arhivate în funcție de atribut

* sunt disponibile mecanisme de filtrare a rezultatelor în funcție de parametrii precum
  + localitatea sau județul de deplasare (părăsire sau nu a garnizoanei)
  + adresa de email a utilizatorului cu rol de autor a rapoartelor, în cazul în care utilizatorul autentificat este altul decât administratorul serviciului
  + adresa de email a oricărui utilizator cu rol de autor al cererilor disponibile în cadrul acestei secțiuni, în cazul în care cel autentificat este administratorul platformei, el având posibilitatea vizualizării tuturor cererilor arhivate
  + motivul sau conținutul cererii arhivate
* în momentul introducerii oricărui cuvânt într-un spațiu destinat, sistemul va face filtrarea rezultatelor, ținându-se cont atât de parametrul selectat, cât și de cuvântul introdus
* astfel, dacă un utilizator va selecta opțiunea EMAIL și va introduce cuvântul ana\_..., rezultatele obținute și afișate vor fi toate cererile ai căror utilizatori au adrese de email cu prefixul introdus (spre exemplu ana\_maria@gmail.com, ana\_1234@yahoo.com etc.)

#### Vizualizarea și modificarea structurii grupurilor de utilizatori

Utilizatorii potențiali ai unui astfel de sistem fac parte sau se încadrează în diverse structuri, cu posibilitatea deținerii de roluri multiple și efectuării numeroaselor operații, mai ales asupra fluxurilor de documente.

Proiectul curent este destinat unităților sau organizațiilor care au nevoie de un context sigur și securizat de desfășurare a activității profesionale. În astfel de instituții, singura entitate autorizată să realizeze modificări asupra structurii grupurilor de utilizatori este administratorul platformei.

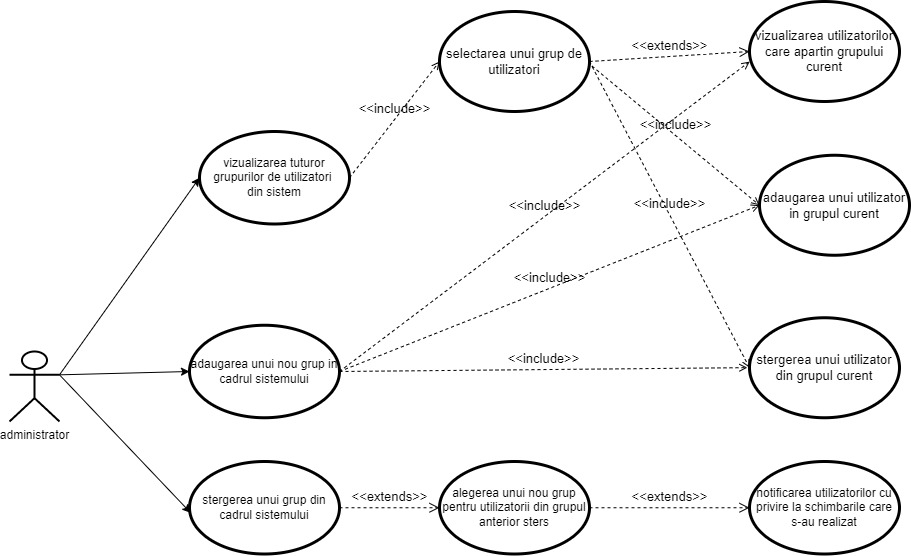


Figura 16 - Căutarea cererilor arhivate în funcție de atribut

Înainte de a realiza orice acțiune sau operație asupra grupurilor de utilizatori, administratorul are posibilitatea vizualizării, sub formă tabelară, a acestora.

Selectarea unuia dintre grupurile disponibile include:

* vizualizarea sub formă tabelară a utilizatorilor care fac parte din grupul selectat
* posibilitatea adăugării sau eliminării unui student din cadrul grupului selectat

Alte operații care pot fi realizate în cadrul paginii web curente sunt:

* adăugarea unui nou grup în cadrul sistemului
* după adăugarea grupului în sistem, utilizatorul poate efectua asupra acestuia toate operațiile enumerate anterior, operații referitoare de utilizatorii care aparțin grupului în cauză
* eliminarea unuia sau a mai multor grupuri din sistem
* eliminarea unui grup presupune și eliminarea utilizatorilor din structura acestuia
* orice modificare realizată presupune și include automat notificarea tuturor utilizatorului cu informațiile actualizate

#### Luarea deciziilor

Una dintre cele mai importante atribuții ale unui utilizator cu rol de comandant, administrator, secretar sau profesor este luarea deciziilor asupra fluxului de documente primite.

Primul contact pe care îl are cu acestea în momentul autentificării sunt notificările aferente. Prin accesarea paginii destinate luării deciziilor, un astfel de utilizator poate parcurge mai mulți pași către luarea deciziilor finale, astfel:

* Vizualizarea tuturor cererilor destinate lui se realizează sub formă tabelară, ca mai apoi să poată fi accesate pe rând
* După selectarea uneia dintre cereri, utilizatorul are posibilități precum:
  + Posibilitatea de a vizualiza deciziile anterioare luate asupra cererii de către restul entităților din cadrul fluxului, autorizate cu efectuarea operațiilor asupra documentului primit
  + Posibilitatea de a descărca documentele ce conțin cererile selectate, în vederea vizualizării acestora de pe computerul personal
  + Posibilitatea filtrării documentelor cu scop de cerere în funcție de categoria din care acestea fac parte sau în funcție de statusul actual al acestora (statusurile pot fi ACCEPTAT, REFUZAT sau ÎN AȘTEPTARE)
  + Acceptarea cererilor prin accesarea butonului ACCEPTĂ și modificarea statusului acestora
  + Refuzarea cererilor prin accesarea butonului REFUZĂ, completarea motivului pentru care a luat decizia respectivă și modificarea statusului acestora
  + Completarea motivului pentru care se realizează refuzul asupra cererii constituie un factor important în ceea ce privește luarea deciziilor de către restul utilizatorilor participanți la fluxul de decizie stabilit de către emițătorul raportului

În ceea ce privește completarea motivului în momentul refuzării unui raport, se poate afirma faptul că această operațiune poate sau nu poate influența deciziile celorlalți superiori care încă nu au trimis un răspuns emițătorului. Un scenariu concludent situației este acela în care comandantul de pluton va refuza o cerere pe motiv disciplinar. Întrucât relația profesională a emițătorului cu acest utilizator este mai strânsă decât cea cu restul comandanților cu rol de decizie mai important ierarhic, în momentul luării deciziei asupra raportului, un comandant de batalion se poate ghida sau nu după decizia precedentă. În acest caz, acțiunea sa asupra cererii în cauză poate fi realizată pe baza încrederii în comandantul de pluton.

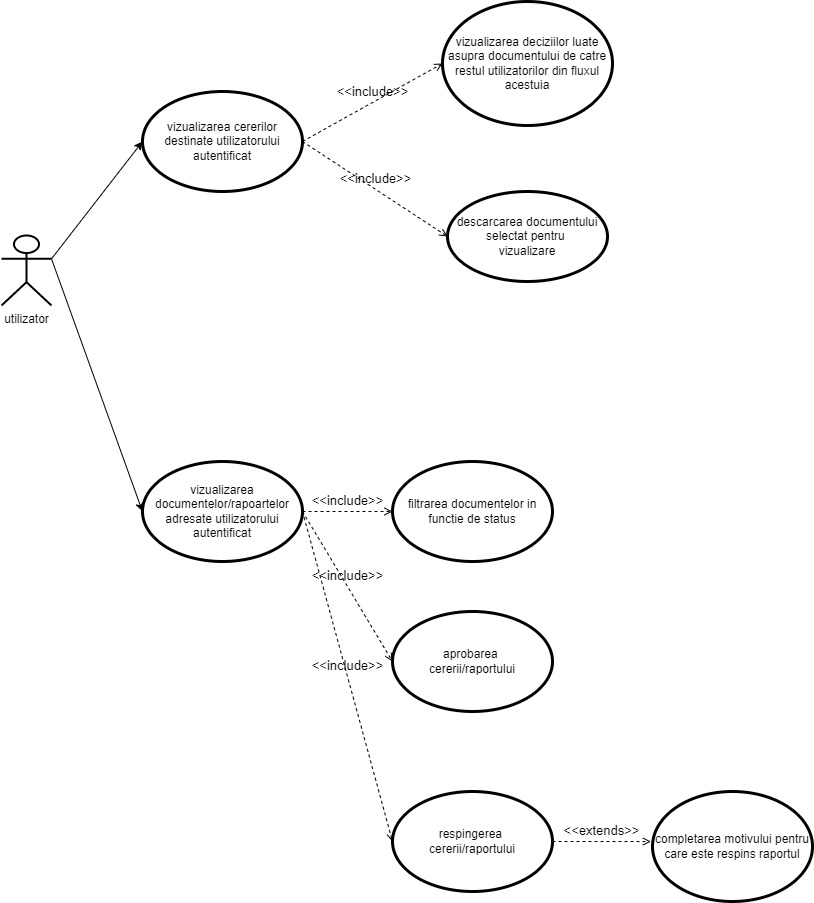


Figura 17 - Luarea deciziilor

#### Administrarea modelelor-tip

O problemă des întâlnită în cadrul marilor organizațiilor este lipsa de documentare sau de informare în ceea ce privește formatul documentelor tip, clasificate sau neclasificate.

Completarea greșită a unei cereri și transmiterea mai departe către destinatari a acesteia poate îngreuna managementul fluxului documentului respectiv.

Într-un context fizic, trimiterea unui raport completat în mod greșit de către autor poate duce la consumarea unor resurse, precum timpul și materialele utilizate.

Având in vedere că soluția prezentată propune trecerea din spațiul fizic către cel virtual, se elimină, în cazul unui document greșit realizat, problema resurselor materiale consumate. Rămâne însă valabilă problema timpului de așteptare a unui răspuns în ceea ce privește acceptarea sau refuzarea unui document de tip cerere.

În sensul rezolvării unor astfel de probleme, implementarea curentă cuprinde o secțiune de păstrare și centralizare a unor șabloane. Adăugarea acestora de către utilizatorii autorizați are ca scop simplificarea procesului de completare fizică a unui raport, în cazul în care utilizatorului îi este mai comodă această variantă și nu varianta completării raportului prin intermediul interfeței.

Astfel, în cazul în care un utilizator de tipul „STUDENT” intenționează să depună un raport de mărire a notei, dar nu știe formatul oficial al acestuia, el va accesa pagina web responsabilă cu centralizarea modelelor documentelor și o va descărca pentru utilizarea ulterioară.

În sensul administrării acestei secțiuni, entității sistemului cu rol autorizat i se pun la dispoziție funcționalități specifice și trebuie să parcurgă următorii pași:

* Pentru adăugarea unei noi categorii de documente va accesa bara destinată acestei operații
* În cadrul secțiunii „Adăugare categorie nouă”, administratorul va putea completa atât denumirea noii categorii, cât și categoria căreia aceasta îi este anexată
* În cazul necompletării numelui noii categorii sau în cazul neselectării uneia dintre categoriile din care aceasta va face parte va apărea un mesaj de atenționare în acest sens
* Pentru adăugarea unui nou șablon în cadrul unei categorii va accesa bara destinată acestei operații
* În cadrul secțiunii „Adăugare șablon nou”, administratorul va putea completa atât denumirea noului șablon, va selecta categoria din care va face parte acesta, dar va trebui să și încarce din computerul personal documentul aferent
* În cazul necompletării denumirii noului șablon pe care intenționează să în încarce pe platformă, neselectării categoriei sau în cazul în care nu încarcă niciun document, va apărea un mesaj de atenționare în acest sens

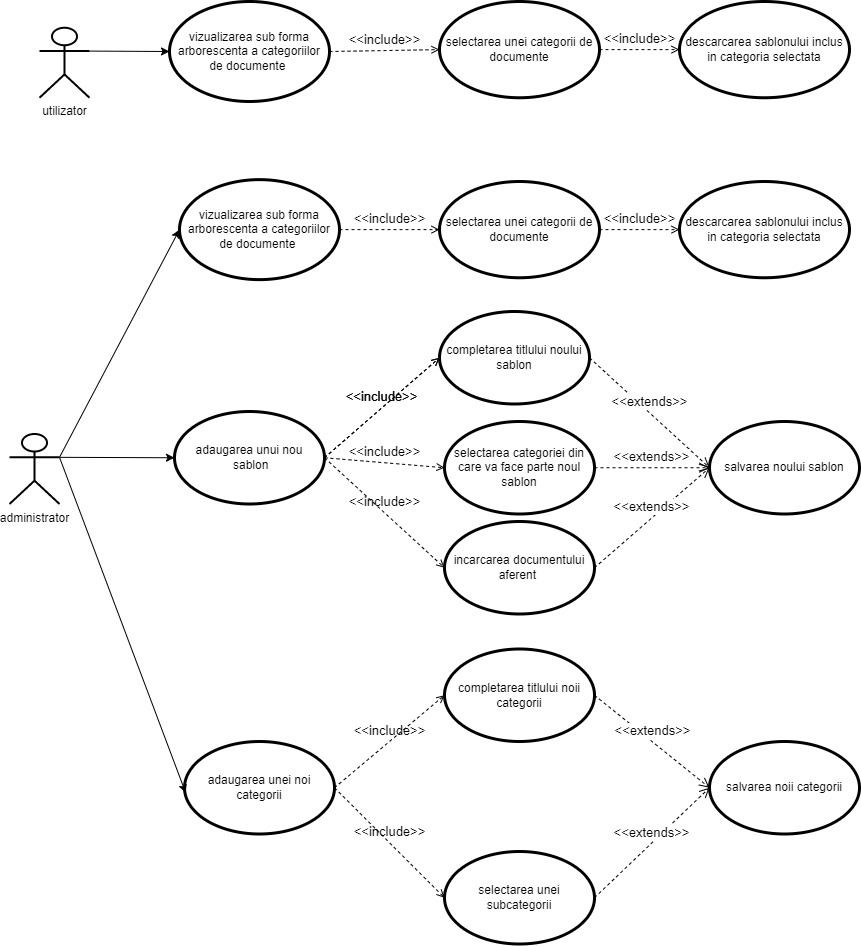


Figura 18 - Administrarea modelelor-tip

#### Fluxuri personalizate

Proiectul curent pune la dispoziția potențialilor utilizatori ai sistemului posibilități de definire a propriilor fluxuri de transmitere a documentelor.

Având în vedere faptul că procesul de management al transmiterii documentelor se poate îngreuna din cauza diversității tipurilor de documente, a fost implementată posibilitatea definirii propriilor fluxuri. Această funcționalitate survine în urma schimbărilor care pot avea loc în procesul de transmitere a unei cereri sau în procesul notificare a grupurilor sau utilizatorilor individuali cu noile anunțuri disponibile.

Astfel, în contextul trimiterii de către un student a unui raport medical către persoanele avizate să ia decizii asupra acestuia, pot interveni diverse probleme. Exemple de astfel de probleme sunt:

* Medicul personal de familie sau medicul specialist este în concediu sau nu poate ajunge la program în perioada în care va trebui aprobat raportul în cauză
* Comandantul de companie, următoarea persoană din ierarhie care trebuie să ia o decizie asupra documentului, lipsește din motive similare cu cele anterior enumerate
* Înaintarea raportului trebuie grăbită în funcție de gravitatea problemei medicale a autorului raportului respectiv

În aceste cazuri, o soluție rezonabilă și rapidă o reprezintă transmiterea documentului ce conține raportul către potențialii înlocuitori ai utilizatorilor care compun fluxul inițial. În acest sens a fost pusă la dispoziția utilizatorului posibilitatea de definire a fluxurilor, altele decât fluxul inițial stabilit la crearea contului.

Realizarea fluxurilor personalizate se compune din următorii pași:

* Accesarea secțiunii pusă la dispoziția utilizatorilor care pot trimite o cerere, în special a utilizatorilor de tip STUDENT
* Accesarea butonului de adăugarea e unui nou flux
* Selectarea utilizatorilor cu rol de decizie în cadrul viitorului flux realizat
* Se va alege un titlu sau o denumire pentru acest flux în scopul selectării mai ușoare a acestuia în cazul în care, la transmiterea unui document, va alege opțiunea de SELECTARE FLUX PERSONALIZAT (spre exemplu, la completarea unui raport medical, utilizatorul va selecta din lista de fluxuri personalizate a unui eventual flux creat anterior și căruia i-a pus denumirea „Flux pentru învoiri medicale”)
* Salvarea noului flux prin accesarea butonului SALVEAZĂ FLUX

Vizualizarea propriilor fluxuri realizate anterior se realizează prin expunerea acestora sub formă tabelară. Vizualizarea utilizatorilor care compun un flux selectat se realizează prin accesarea butonului VEZI FLUX.

Se pot face modificări asupra structurii fluxului selectat, iar operațiile aferente acestei acțiuni sunt următoarele:

* După selectarea unuia dintre aceste fluxuri și după accesarea butonul VEZI FLUX vor fi afișați, tot sub formă tabelară, utilizatorii componenți ai acestuia
* Printr-o componentă de tip „drag-and-drop” se va putea schimba ordinea și prioritatea participanților la fluxul curent de decizie, această schimbare având ca efecte modificarea ordinii de primire viitoarelor rapoarte
* În cazul în care cel care a realizat fluxul selectat va dori să realizeze și alte modificări în ceea ce privește structura acestuia, se pun la dispoziția sa funcționalități de adăugare sau ștergere a utilizatorilor din lista cu priorități, pe lângă cele de modificare a ordinii din listă

Un exemplu în acest sens este cazul în care unui utilizator de tip student i se va asigna, pe timpul anului universitar, un nou comandant de companie, fără ca cel prezent să fie eliminat din sistem. Cei doi comandanți de companie vor avea același grad și aceleași autorizări în ceea ce privește managementul fluxul de documente în cadrul structurii pe care o au în subordine. În acest caz, va rămâne la libera decizie a studentului pe care dintre aceștia îi va selecta în momentul în care va încărca un document asupra căruia trebuie să ia decizia un comandant de companie. Astfel, va avea posibilitatea selectării individuale a entităților cu rol de decizie sau va putea, mai simplu, să modifice fluxul personalizat pe care îl va putea reutiliza de câte ori va fi nevoie.

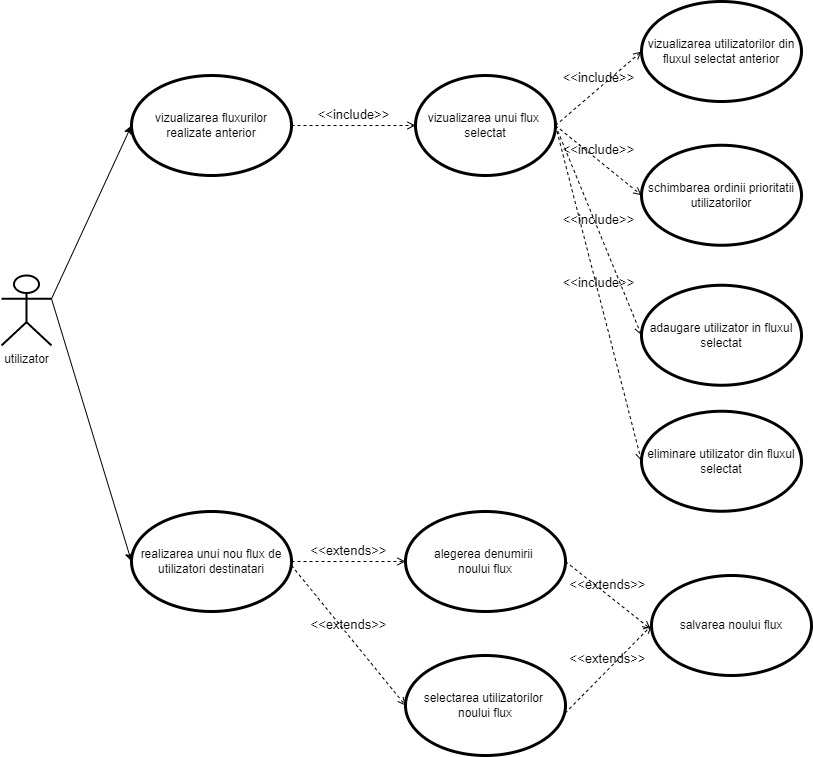


Figura 19 - Vizualizarea și realizarea fluxurilor personalizate

#### Administrarea conturilor de utilizator

Administratorul, ca în cazul tuturor platformelor web, este entitatea cu cele mai multe autorizări în ceea ce privește realizarea operațiunilor asupra elementelor disponibile pe platformă. În ceea ce privește administrarea conturilor tuturor celorlalți utilizatori care beneficiază de funcționalitățile puse la dispoziție de către platforma web, administratorul are, în plus față de alte site-uri web, rolul și responsabilitatea de a accepta sau refuza cererile de creare sau modificare a conturilor. Acest lucru se datorează naturii organizației, aceasta neavând posibilitatea de a permite înregistrarea oricărui vizitator pe platformă.

Deținerea și manevrarea unui volum mare de documente clasificate sunt principalele motive pentru care se pune accent asupra entităților și restricționării, restrângerii sau filtrării drepturilor acestora de a efectua anumite operații în cadrul aplicației.

În vederea vizualizării detaliilor despre utilizatori, administratorul are de parcurs următorii pași:

* Accesarea secțiunii pentru vizualizarea tuturor utilizatorilor din cadrul platformei
* Selectarea unuia dintre utilizatori și vizualizarea detaliilor ce alcătuiesc contul și profilul acestuia

În vederea modificării structurii bazei de date în care se stochează profilurile utilizatorilor, administratorul are următoarele autorizări:

* Poate realiza eliminarea unui utilizator din baza de date, împreună cu toate acțiunile sale care se află în desfășurare în momentul ștergerii
* crearea unui cont de utilizator, prin adăugarea acestuia în baza de date
* oferirea de drepturi de acces la facilitățile platformei și notificarea, prin intermediul unui serviciu de mesagerie precum Email, cu credențialele aferente
* luarea deciziilor asupra cererilor de creare de cont sau de modificare a credențialelor și notificarea ulterioară a vizitatorilor cu privire la rezultatele deciziilor

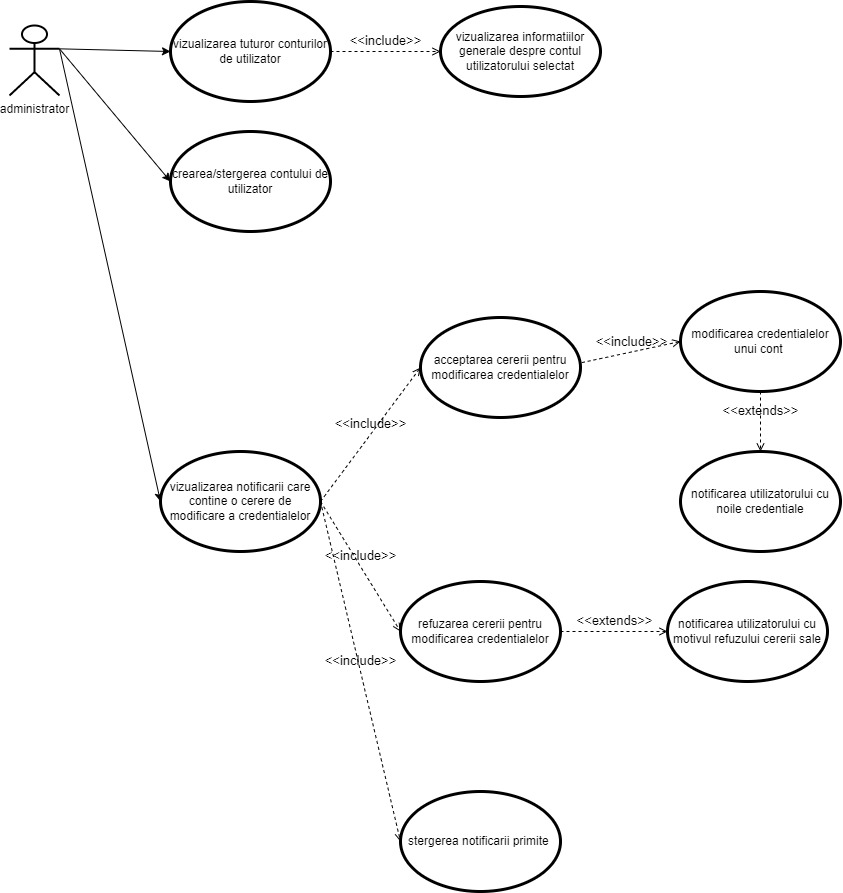


Figura 20 - Administrarea conturilor de utilizator

#### Vizualizarea notificărilor

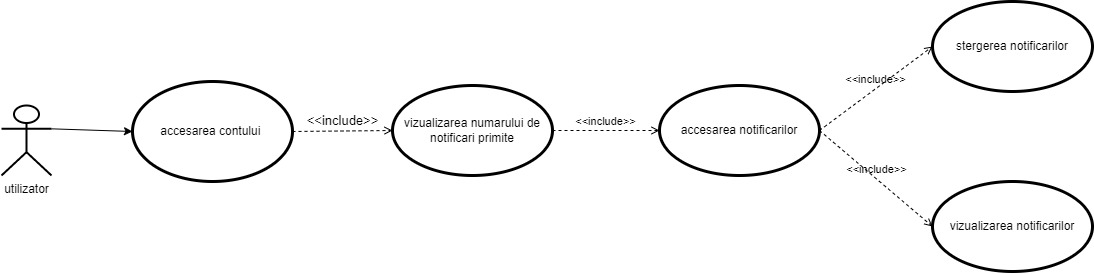


Figura 21 - Vizualizarea notificărilor

Centrul de notificări este o funcție care oferă o prezentare generală a alertelor din diverse aplicații. Prin intermediul acestuia se afișează notificări în momentul în care utilizatorul beneficiază de o acțiune asociată contului său, în loc să necesite căutarea acesteia în cadrul fiecărei pagini web din sistem. Utilizatorii pot alege ce acțiuni apar în Centrul de notificări și cum sunt gestionate.

Notificările sunt, de obicei, formate din următoarele elemente: un titlu, un mesaj, o imagine si o adresa URL. Un titlu atractiv și clar este necesar, precum este și un text nu prea lung. Imaginile nu sunt strict necesare, dar pot fi foarte convingătoare, daca sunt alese corect.

Accesarea sistemului de notificări în cadrul aplicației curente reprezintă primul contact al acestuia cu interfața și se realizează astfel:

* După autentificarea cu succes la aplicație, utilizatorul are la dispoziție în bara de navigare o iconiță specifică notificărilor
* În momentul în care trece cu mouse-ul pe iconiță, vor apărea tot în zona de sus a ecranului texte scurte prin intermediul cărora se pot identifica natura mesajelor primite
* În momentul accesării centrului de notificări, se va deschide pagina aferentă acestuia, iar notificările vor apărea sub formă tabelară, împreună cu câteva detalii specifice
* Utilizatorul are posibilitatea, după vizualizarea notificărilor proprii, să le șteargă din centrul destinat acestora, în scopul lizibilității și filtrării mesajelor deja vizualizare

## Arhivarea documentelor

### ElasticSearch

Elasticsearch este un motor de căutare și de analiză distribuit, bazat pe biblioteca Lucene, gratuit și open-source pentru toate tipurile de date, inclusiv textuale, numerice, geospațiale, structurate și nestructurate.

Elasticsearch este dezvoltat în Java, iar clienții oficiali sunt disponibili în Java, .NET (C#), PHP, Python, Apache Groovy, Ruby și multe alte limbi.

Cunoscut pentru API-urile REST simple, natura distribuită, viteza și scalabilitatea, Elasticsearch este componenta centrală a Elastic Stack, un set de instrumente gratuite și open-source pentru asimilarea, îmbogățirea, stocarea, analizarea și vizualizarea datelor.

Există așadar trei mari concepte logice care alcătuiesc structura acestui motor de căutare:

* documente
* tipuri
* indici

#### Documente

Documentele și fișierele reprezintă unitatea de bază de informații care pot fi indexate prin intermediul Elasticsearch. Acestea sunt manevrate în format JSON, adică în formatul global de schimb de date pe internet.

Un document poate fi asociat în cadrul ElasticSearch cu un rând dintr-o bază de date relațională, reprezentând o anumită entitate – ceea ce se caută. În acest mediu de lucru, un document poate însemna atât text, cât și oricare alte date structurate codificate în JSON, date precum numere sau șiruri.

Fiecare document dispune de, idem unui element dintr-o bază de date, un ID unic și un anumit tip de date, care descriu ce fel de entitate este documentul. De exemplu, un document poate reprezenta un articol de enciclopedie sau intrări de jurnal de pe un server web.

În cazul de față, documentele introduse în ElasticSearch, sunt cererile încărcate sau completate de un utilizator pe platforma web de gestionare a fluxului de documente. După finalizarea tuturor operațiilor posibile asupra documentului respectiv (operații precum încărcare/completare/acceptare/respingere/etc.), acesta va fi indexat prin intermediul ElasticSearch.

Motorul de căutare salvează astfel date precum:

* adresa de email a autorului cererii
* județul și orașul de deplasare
* tipul documentului în funcție de forma în care a fost salvat(document sau completat pe platformă)
* motivul pentru care a fost refuzat, în cazul în care a fost refuzat

Astfel, soluția de căutare propune căutarea documentelor indexate atât în funcție de șirul de caractere introdus într-un câmp specific, cât și în funcție de atributul selectat ulterior.

#### Indici

Un index este o colecție de documente care au caracteristici similare, reprezentând, de asemenea, entitatea de cel mai înalt nivel la care se poate interoga în Elasticsearch.

Un index poate fi asociat cu o bază de date dintr-o schemă de bază de date relațională. Oricare documente dintr-un index sunt de obicei legate logic. În contextul unui site de comerț electronic, de exemplu, poate exista un index pentru Clienți, unul pentru Produse, unul pentru Comenzi și așa mai departe.

Acest element specific ElasticSearch se identifică printr-un nume folosit pentru a face referire la index în timpul efectuării operațiunilor de indexare, căutare, actualizare și ștergere împotriva documentelor din acesta.

#### Index inversat

Un index în Elasticsearch este de fapt ceea ce se numește index inversat, adică mecanismul prin care funcționează toate motoarele de căutare. Este o structură de date care stochează o mapare de la conținut, cum ar fi cuvinte sau numere, la locațiile sale într-un document sau într-un set de documente. Practic, este o structură de date asemănătoare hashmap-ului care direcționează de la un cuvânt la un document.

Un index inversat nu stochează direct șiruri de caractere și, în schimb, împarte fiecare document în termeni de căutare individuali (adică fiecare cuvânt), apoi mapează fiecare termen de căutare la documentele în care apar termenii de căutare.

#### Tipurile de date

Nu toate tipurile din Elasticsearch au un echivalent în SQL și invers, de aceea, Elasticsearch SQL utilizează particularitățile tipului de date ale primului față de cel din urmă, deoarece în cele din urmă Elasticsearch este depozitul de suport.

Pe lângă tipurile specifice SQL, Elasticsearch SQL acceptă și tipuri specifice SQL care nu au un echivalent în Elasticsearch. Astfel de tipuri nu pot fi încărcate din Elasticsearch (deoarece nu știe despre ele), totuși pot fi utilizate în interiorul Elasticsearch SQL în interogări sau în rezultatele acestora.

Ca exemplu, putem lua în considerare următoarea mapare a șirurilor:



Figura 22 - Maparea șirurilor în ElasticSearch

Un exemplu de interogare SQL este următorul:

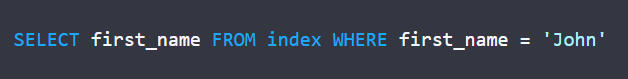


Figura 23 - Interogare SQL

Un exemplu de interogare ElasticSearch SQL este următorul:

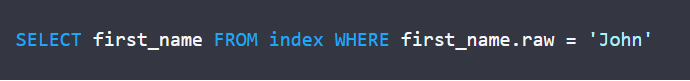


Figura 24 - Interogare ElasticSearch SQL

Acestea două sunt echivalente deoarece Elasticsearch SQL preia automat câmpul multiplu „raw” din „raw” pentru o potrivire exactă.

### Implementare

Am ales opțiunea integrării unei imagini de ElasticSearch în cadrul unui docker, în loc de instalarea acestuia local. Imaginea conține așa-zisa ”rețetă” pentru mașina virtuală.

Pașii pentru integrarea imaginii care conține ElasticSearch sunt următorii:

* se creează containerul pe baza imaginii de docker folosind următoarea comandă

***docker network create elastic***

* se pornește Elasticsearch în Docker.
* se generează o parolă pentru utilizatorul elastic și se iese către terminal, plus un token de înregistrare

***docker run --name es01 --net elastic -p 9200:9200 -p 9300:9300 -it docker.elastic.co/elasticsearch/elasticsearch:8.3.1***

Rolul ElasticSearch este de a stoca obiectele create în cadrul acestuia. În proiectul curent s-a realizat crearea unui obiect de tip cerere, aceasta fiind asociată cu o bază de date în cadrul acestui search-engine.

Configurarea s-a realizat pentru vizualizarea elementelor bazei de date în format JSON pe portul 9200. Adăugarea în baza de date se poate realiza printr-o cerere de tip POST, iar extragerea elementelor prin cereri de tip GET. Se dorește însă integrarea componentelor de adăugare și extragere a datelor din ElasticSearch în interfața pe care aplicația o pune la dispoziție. Pentru a realiza acest lucru se urmează pașii:

* în fișierul pom.xml se adaugă dependințele pentru search-engine pentru a putea utiliza clase care lucrează cu ElasticSearch
* se realizează clase de configurare care conține funcții pentru stabilirea portului de conectare
* se realizează o clasă pentru maparea modelului de tip cerere
* se creează un repository care extinde repository-ul implicit al ElasticSearch-ului; prin intermediul acestuia se vor salva cererile în baza de date
* se creează o clasă de tip Cerere pentru generarea unui obiect care va fi inserat ulterior în baza de date
* în repository sunt realizate interogări specifice de găsire a elementelor în funcție de parametrii stabiliți
* cererile de căutare apelează controllerul care apelează la rândul lui repository-ul pentru a căuta în baza de date

### Formatul obiectelor salvate



Figura - Formatul obiectelor salvate

## Mecanisme pentru asigurarea integrității și autenticității documentelor

### *Generalități – utilizabilitatea documentelor PDF*

Documentele PDF au devenit renumite întrucât acest format poate fi de încredere în ceea ce privește asigurarea unei ieșiri consistente, fie că este pe ecran sau în tipărire.

În anii care au urmat, a apărut o abundență de instrumente noi de la Adobe, precum și de la furnizori de software terți. În timp, formatului PDF i-au fost adăugate o mulțime de funcționalități, iar acesta este unul dintre motivele pentru care a devenit formatul de document preferat de alegere în multe sectoare și industrii profesionale.

Capitolul curent se concentrează asupra unui aspect specific al fișierelor PDF: semnăturile digitale.

### *Semnătura digitală*

#### Definirea semnăturilor digitale

Semnătura digitală reprezintă o modalitate de “amprentare electronică” sub forma unui mesaj codificat. Aceasta asociază în siguranță un semnatar cu un document într-o tranzacție înregistrată. Semnăturile utilizează formatul acceptat, adică infrastructura cheilor publice (PKI), oferind atât un nivel ridicat de securitate, cât și acceptare universală în utilizabilitate.

O infrastructură de cheie publică (PKI) este un set de roluri, politici, hardware, software și proceduri necesare pentru a:

* crea certificate digitale
* gestiona certificate digitale
* distribui certificate digitale
* utiliza certificate digitale
* stoca certificate digitale
* revoca certificate digitale
* gestiona criptarea cheii publice.

PKI este, de altfel, un set de cerințe care permit crearea de semnături digitale, printre altele. Prin PKI, fiecare tranzacție de semnătură digitală include o pereche de chei:

* cheia privată
* cheia publică

O cheie criptografică reprezintă algoritmul matematic utilizat pentru a cripta sau decripta datele și are forma unui șir de biți combinați cu datele pentru a rezulta textul criptat. Cheia criptografică poate fi folosită și pentru a decripta datele înapoi în textul original.

Astfel, cheia privată nu se împărtășește, după cum sugerează și numele acesteia, și se utilizează doar de semnatar pentru a semna electronic documentele.

Cheia publică este disponibilă tuturor și utilizată de către cei care au nevoie să valideze semnătura electronică a semnatarului.

Pentru a obține o semnătură digitală validă din punct de vedere juridic, sistemele utilizate pentru crearea și validarea semnăturii trebuie să fie considerate de încredere. Astfel, formatul semnăturii electronice trebuie să permită verificarea validității acesteia indiferent dacă semnatarul sau persoana care verifică semnătura neagă/repudiază ulterior semnarea. [2]

Cerințele necesare ale aplicațiile cu rol în crearea și validarea unei semnături digitale pentru ca aceasta să fie considerată legală din punct de vedere juridic sunt următoarele:

* să asigure protecția cheii private de semnătură împotriva folosirii neautorizate sau dezvăluirii acesteia unor atacatori cu un potențial ridicat; [2]
* să afișeze toate datele pe care utilizatorul urmează să le semneze într-o manieră lizibilă pentru acesta; [2]
* să pună la dispoziție utilizatorului mijloacele necesare prin care acesta să poată decide dacă va semna sau nu datele afișate; [2]
* să nu modifice datele de semnat înainte sau pe parcursul procesului de semnare; [2]
* să afișeze validitatea sau invaliditatea unei semnături într-o manieră care să elimine orice ambiguitate; [2]
* să afișeze informații despre semnatarul documentului; [2]
* să nu declare o semnătură invalidă ca fiind validă și invers. [2]

#### Importanța semnăturilor digitale

În cazul în care un document care are valoare legală conține informații importante despre drepturi și obligații, trebuie asigurată autenticitatea. Aceasta reprezintă o modalitate prin care se poate asigura faptul că autorii documentelor nu își pot nega angajamentele notate.

În plus, există posibilitatea ca un astfel de document să trebuiască trimis prin poștă electronică sau mail, vizualizat și/sau stocat de către diferite părți. În anumite locuri din fluxul de lucru, în multiple momente, documentul poate fi modificat, fie:

* voluntar (de exemplu pentru adăugarea unei semnături suplimentare);
* involuntar (de exemplu din cauza unei erori de transmisie);
* în mod deliberat (dacă se dorește crearea unui fals din documentul original);

Timp de secole, s-a încercat rezolvarea acestei probleme prin intermediul semnăturii fizice pe hârtie. În prezent, putem folosi semnăturile digitale pentru a asigura:

• integritatea documentului — acesta nu a fost modificat undeva în fluxul de lucru(reprezintă exact ceea ce a fost creat de autorul său);

• autenticitatea documentului —asigurarea că autorul documentului este cine credem noi că este (și nu altcineva);

• non-repudierea —asigurarea că autorul nu își poate nega calitatea de autor(de asemenea, nu poate nega autenticitatea semnăturii sale pe un document);

#### Semnare și verificare semnătură – funcționare

O semnătură digitală se bazează pe un algoritm de hash-ing și criptografia cu cheie publică. Când o entitate intenționează să semneze datele pe care vrea să le protejeze, acesta aplică un algoritm de hash-ing asupra datelor respective. Apoi urmează procesul de criptare a rezultatului obținut utilizând cu cheia privată. Valoarea obținută în urma celor doua operații consecutive se numește semnătură digitală. Dacă se modifică până și un singur bit din datele originale, automat se va genera o semnătură digitală complet diferită.

Verificarea unei semnături digitale este procesul opus explicat anterior. Verificarea unei semnături va spune dacă datele semnate s-au schimbat sau nu. În plus, oferă dovada originii mesajului și reprezintă și o metodă de verificare a integrității mesajului. Procesul de verificare a semnăturii electronice constă în următorii pași de bază:

* semnătura este decriptată folosind cheia publică pentru a produce valoarea hash inițială;
* datele care au fost semnate au fost trecute, în procesul de semnare, prin intermediul algoritmului de hash;
* dacă cele două valori hash, prin comparație, se potrivesc, atunci semnătura a fost verificată și se poate spune că este validă;

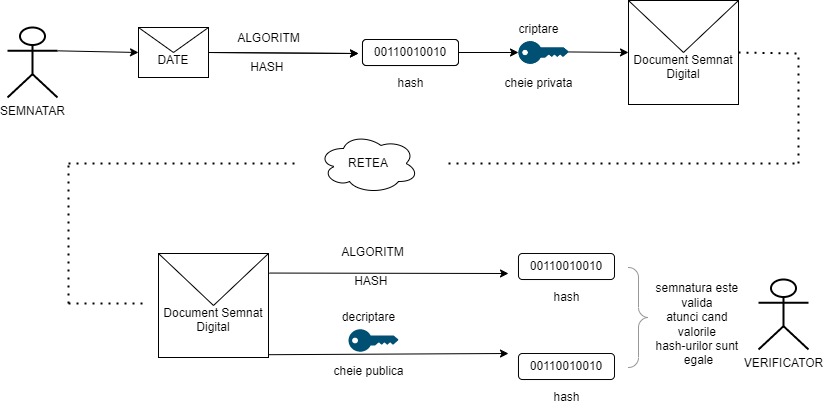


Figura 26- Semnarea și verificarea semnăturii

#### Algoritmul RSA

În criptografie, RSA este un algoritm criptografic cu chei publice, primul algoritm utilizat atât pentru criptare, cât si pentru semnătură electronică. Puterea sa criptografică se bazează pe dificultatea problemei factorizării numerelor întregi, problema la care se reduce criptanaliza RSA si pentru care toți algoritmii de rezolvare cunoscuți au complexitate exponențială. Există însă câteva metode de criptanaliză care ocolesc factorizarea efectivă, exploatând maniere eronate de implementare efectiva a schemei de criptare. [3]

**Funcționare**

RSA este un algoritm de criptare pe blocuri. Aceasta înseamnă că atât textul clar cât și cel cifrat sunt numere intre 0 si n-1, cu un n ales. Un mesaj de dimensiune mai mare decât log n este împărțit în segmente de lungime corespunzătoare, numite blocuri, care sunt cifrate rând pe rând. De asemenea, ca algoritm criptografic cu chei publice, funcționează pe baza unei perechi de chei legate matematic între ele: o cheie publică, cunoscută de toata lumea, și una secretă, cunoscută doar de deținătorul acesteia. [3]

**Generarea cheilor**

* Se generează două numere prime, de preferat mari, p si q;
* Se calculează n = pq si Φ = (p-1)(q-1)
* Se alege un întreg aleator e, 1 < e < Φ astfel încât cmmdc(e, Φ) = 1.
* Perechea (n, e) este cheia publică.
* Folosind algoritmul lui Euclid extins, se calculează întregul d, unicul cu proprietatea ca de ≡ 1mod Φ. (n, d) constituie cheia secretă. [3]

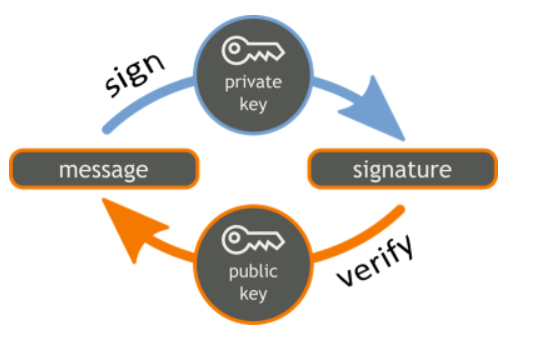


Figura 27 -Semnare și verifcare semnătură

Semnăturile digitale depind în mod unic de conținutul mesajului semnat și de cheia secretă a unei entități. Pentru a obține o semnătură se pleacă de la datele inițiale care se trec printr-un algoritm de hash, în cazul curent SHA1. Acest hash generat va trece apoi printr-o funcție de semnare pusă la dispoziție de API-ul Java. Pentru folosirea funcției de semnare, serverul are nevoie de cheia privată a utilizatorului care dorește să semneze documentul.

Hash-ul reprezintă o funcție care returnează o valoarea aproape unică pentru fiecare mesaj sau fișier în parte, iar dimensiunea acestuia are o valoare finită. Generarea de chei publice și private ale fiecărui utilizator se realizează în momentul creării contului.

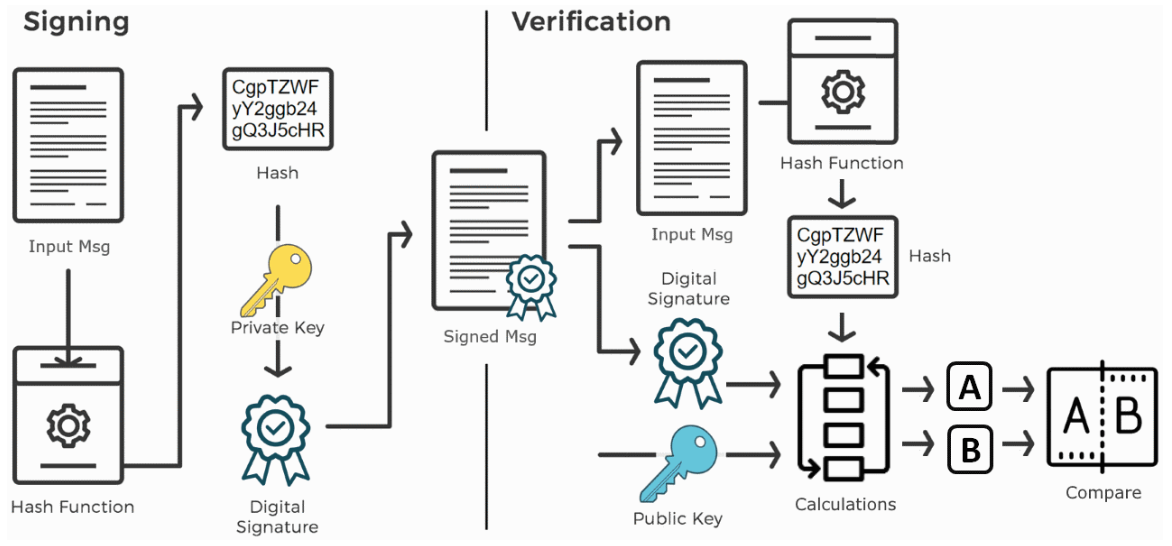


Figura 28 - Algoritmul de semnare RSA

Verificarea semnăturii constă în regenerarea hash-ului, decriptarea și compararea celor două digest-uri. La verificare se folosește cheia publică a destinatarului, iar la generarea semnăturii se folosește cheia privată.

**Semnarea documentelor**



Figura 29 - Semnarea documentelor

Funcționalități Java utilizate:

* KeyFactory.getInstance(„RSA”) - crearea unui obiect de tip KeyFactory
* KeyFactory.generatePrivate(cheiePrivata) - folosită pentru a genera un obiect cheie privată din specificația cheii furnizate
* Signature.getInstance(„SHA1WithRSA”)
* Sig.initSign(userPrivKey) – inițializează semnătura pentru semnare
* Sig.update(data) – actualizează octeții care vor fi semnați sau verificați
* Sign() -semnarea propriu-zisă

**Verificarea semnăturii**

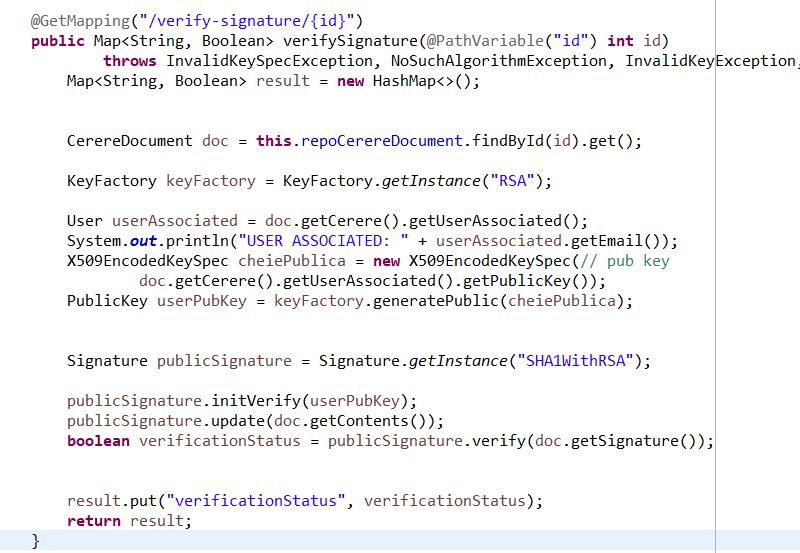


Figura 30 - Verificarea semnăturii

## Crearea conturilor de utilizator

### Inregistrarea conturilor de utilizator

Înregistrarea contului de utilizator se realizează într-un mod specific, astfel:

* Vizitatorul aplicației accesează secțiunea de creare a conturilor și completează câmpurile aferente
* Trimite cererea de autentificare, iar serverul o redirecționează administratorului
* Acesta va accesa pagina destinată gestionării utilizatorilor și va realiza operații de acceptare sau refuzare a cererilor primite
* În momentul realizării acestor operații, aplicația va:
  + trimite un email configurat în cadrul API-ului Java la o adresă de email a utilizatorului creată prin intermediul serviciului Mercury Mail
  + email-ul de confirmare conține un link de configurare a contului
  + odată accesat link-ul primit, contul va fi confirmat
  + pe lângă confirmarea pe care trebuie să o realizeze utilizatorul care dorește să se înregistreze pe platformă, acesta va mai primi un mail în care este notificat dacă utilizatorul cu rol de decizie i-a aprobat sau i-a refuzat înregistrarea contului
  + utilizatorul se va putea autentifica abia după ce aceste două operațiuni sunt realizate cu succes
  + email-ul de confirmare conține în structura acestuia informații precum token-ul de autentificare generat la creare contului, codul de identificare al utilizatorului și calea către pagina de autentificare pentru asigurarea că nu se va autentifica cu datele unui alt utilizator

### Protocoale utilizate

#### Simple Mail Transfer Protocol (SMTP)

E-mailul a devenit una dintre cele mai utilizate metode de transmisie a mesajelor, iar acest lucru este posibil datorită protocolului SMTP. Acesta este un protocol folosit pentru transmitere, iar protocoalele POP (protocolul oficiului poștal) sau IMAP (protocolul de acces la mesaje internet) sunt folosite pentru preluarea mesajelor de la destinatar.

SMTP este un protocol de nivel de aplicație. Astfel, clientul care dorește să trimită e-mailul deschide o conexiune TCP la serverul SMTP și apoi trimite e-mailul prin conexiune. Serverul SMTP este un mod de ascultare permanent. De îndată ce ascultă o conexiune TCP de la orice client, procesul SMTP inițiază o conexiune prin portul 25. După stabilirea cu succes a unei conexiuni TCP, procesul client trimite e-mailul instantaneu.

Un client SMTP care dorește să trimită e-mailul va contacta direct gazda SMTP a destinației, pentru a trimite e-mailul la destinație. Serverul SMTP va păstra e-mailul pentru el însuși până când este copiat cu succes pe SMTP-ul receptorului.

Clientul SMTP este cel care inițiază sesiunea, iar serverul SMTP este cel care răspunde la cererea de sesiune. Primul menționat începe sesiunea, iar cel din urmă răspunde la cerere.

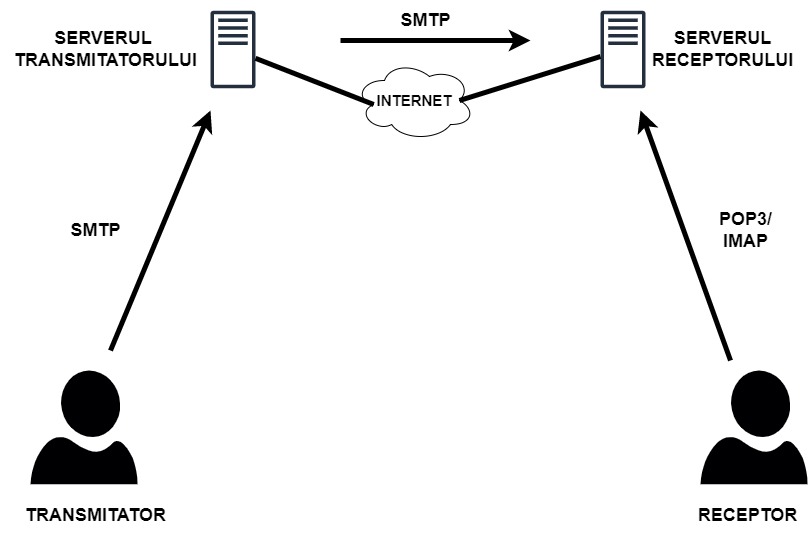


Figura 31 - Modul de funcționare al protocolului SMTP pentru transmiterea email-urilor

#### POP3 (Post Office Protocol 3)

Post Office Protocol 3, sau POP3, este cel mai frecvent utilizat protocol pentru primirea de e-mailuri pe internet. Acesta este utilizat pentru a primi e-mailuri de la un server la distanță și a le trimite către un client local.

POP3 este un protocol client-server unidirecțional în care e-mailul este primit și păstrat pe serverul de e-mail.

Un destinatar sau clientul său de e-mail poate descărca periodic e-mailuri de pe server folosind POP3. Astfel, POP3 oferă un mijloc de descărcare a e-mailului de pe un server către client, astfel încât destinatarul să poată vizualiza e-mailul offline. POP3 poate fi gândit ca un serviciu de „stocare și redirecționare”.

Odată ce e-mailul este pe client, POP3 îl șterge apoi de pe server. Protocolul va funcționa cu condiția ca programul de e-mail să fie configurat pentru a găzdui POP3. Fiecare server de e-mail POP3 are o adresă diferită care trebuie introdusă în programul de e-mail pentru ca acesta să se conecteze la protocol. De asemenea, utilizatorii trebuie să introducă numele de utilizator și parola pentru a primi e-mailuri cu succes.

#### Funcționare POP3

* Serverul pornește serviciul POP3 ascultând pe portul TCP 110.
* Când un client dorește să folosească POP3 pentru preluarea e-mailului, acesta stabilește o conexiune TCP cu gazda serverului.
* Odată stabilită această conexiune, serverul POP3 trimite un salut. În acest moment, sesiunea intră în starea de autorizare .
* În starea de tranzacție care urmează, clientul și serverul schimbă comenzi și răspunsuri până când conexiunea este fie închisă, fie întreruptă.
* Comenzile de la client constau în cuvinte cheie care nu țin cont de majuscule, eventual urmate de argumente.
* Răspunsurile de la server constau dintr-un indicator de stare și un cuvânt cheie, care pot fi urmate de informații suplimentare.
* Când clientul lansează comanda quit , sesiunea intră în starea de actualizare
* Serverul POP3 eliberează orice resurse dobândite în timpul stării tranzacției și spune „la revedere”, adică atunci când conexiunea TCP este închisă.
* După ce sesiunea POP3 intră în starea de actualizare , serverul POP3 șterge mesajul.

### Mercury Mail

#### Generalități

Pachetul Mercury Mail din XAMPP pentru Windows este o modalitate excelentă de a începe să trimiteți e-mailuri din openemr sau din orice program CMS open source plasat în xampp. Configurarea este ușoară, trebuie doar să urmați pașii:

Beneficiile Mercury Mail în openEMR

* Ușor de configurat și integrat cu pachetul XAMPP
* Mementourile de întâlnire vor fi trimise automat folosind Batchcom/Automatic\_notification prin cronjob
* E-mailurile de memento pentru pacient (mementouri de alertă) vor fi trimise automat
* Sigur

#### Configurare Mercury Mail în Windows

Etape de configurare:

* în primul rând se dezactivează serverul HTTP al lui Mercury, astfel încât să nu intre în conflict cu serverul APACHE;
* în panoul Mercury deschis, se accesează „Configurare” -> „Module de protocol”;
* pentru a putea trimite e-mailuri externe, trebuie dezactivată opțiunea „MercuryE SMTP end-to-end delivery client” și activată opțiunea „MercuryC SMTP relaying client”;
* se dorește trimiterea de la localhost, prin urmare se verifică dacă „localhost” este valoarea „nume de internet pentru acest sistem” ;
* se accesează „Configurare” -> „Server SMTP MercuryS”;
* se alege fila „General” și se adaugă un nume pentru serverul SMTP sub „Announce myself as”;
* sub „Ascultați pe portul TCP/IP” se completează „25”, acesta este portul SMTP;
* se adaugă „127.0.0.1” la „Interfață IP de utilizat”, acesta este IP-ul local al computerului personal;
* se limitează accesul la server, astfel încât numai mașina locală să îl poată accesa:
* sub „Controlul conexiunii” se accesează „Adăugați restricție” și se adaugă intervalul IP de la „127.0.0.1” până la „127.0.0.1”;
* se selectează „Permite conexiuni”;
* se lasă toate casetele de selectare deselectate;
* urmează configurarea pentru MercuryP POP3;
* Se accesează „Configurare”-> „Server POP3 MercuryP”, se selectează fila „General”;
* „Ascultă pe portul TCP” -> „110” și „Interfață IP de utilizat” -> „127.0.0.1”;
* urmează configurarea pentru „MercuryC SMTP Client”;
* se accesează „Configurare” -> „Client SMTP MercuryC”;
* pentru a trimite e-mail la adrese externe trebuie să existe un server SMTP extern;
* se introduce adresa SMTP-ului sub „Nume gazdă inteligent”, de exemplu „smtp.gmail.com”;
* în funcție de modul în care se accesează serverul, se completează valorile din „Port/Tip de conexiune”:
* pentru un SMTP „normal” care ar fi probabil portul 25 și „Normal (fără criptare SSL)”;
* majoritatea accesează SMTP prin SSL (dacă se folosește gmail), acesta ar fi portul 465 și „criptare SSL folosind conexiune directă”;
* în cele din urmă se completează „Nume de utilizator de conectare” și „Parola” care sunt în mod normal furnizate de gazda web personală sau se introduce ID-ul și parola Gmail ;
* verificarea utilizatorilor Mercury constă în:
  + accesare „Configurare”-> „Gestionați utilizatorii locali”;
  + ar trebui să existe cel puțin utilizatorii „Admin” și „postmaster”, ambii cu drepturi administrative. Dacă nu, trebuie adăugați;
  + urmează configurarea Java pentru realizarea funcționalităților de transmitere a mail-urilor în mod automat; [4]

# Experimete și rezultate ale testării

## Testarea principalelor funcționalități ale aplicației

Testarea unor funcționalități ale aplicației se va realiza conform cu următorul tabel:

|  |  |  |  |
| --- | --- | --- | --- |
| **Denumire test** | **Descriere test** | **Input** | **Rezultat așteptat** |
| **Logare sau înregistrare** | Se vor verifica mecanismele de creare și utilizare a contului. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Intrarea în scena de meniu principal în caz de succes. În caz contrar, vor fi afișate mesaje de eroare corespunzătoare. |
| **Verificare cerere de modificare a credențialelor** | Se vor verifica mecanismele de verificare și modificare a credențialelor. | Seturi de date pentru completarea field-urilor. Acestea pot include și date greșite. | Schimbarea cu succes a credențialelor de către administrator. |
| **Verificare vizualizare anunțuri** | Se vor testa mecanismele de  vizualizare a tuturor anunțurilor. | Trimiterea unei cereri pentru interogarea anunțurilor disponibile. | Vizualizarea cu succes a anunțurilor și posibilitatea vizualizării detaliilor acestora. |
| **Verificare respingere/acceptare raport.** | Se vor testa mecanismele de  vizualizare și acceptare/respingere a rapoartelor. | Trimiterea unei cereri pentru  verificarea rapoartelor pentru a putea fi acceptate sau respinse de către utilizator. | Respingerea sau acceptarea unui raport și trimiterea unui mesaj de înștiințare către autorul raportului. |
| **Testare vizualizare utilizatori** | Se vor testa mecanismele de  vizualizare a detaliilor utilizatorilor din baza de date. | Trimiterea unei cereri pentru interogarea utilizatorilor disponibili în baza de date. | Vizualizarea cu succes a utilizatorilor și posibilitatea vizualizării detaliilor despre aceștia. |
| **Verificare adăugare raport** | Se vor verifica mecanismele de creare și adăugare a unui raport. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Intrarea în scena de adăugare raport finalizată în caz de succes. În caz contrar, vor fi afișate mesaje de eroare corespunzătoare și posibilitatea re-completării field-urilor aferente. |
| **Verificare completare raport** | Se vor verifica mecanismele de completare a unui raport. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Completarea unui raport cu succes după completarea datelor specifice fiecărui tip de raport. Notificarea utilizatorului în cazul necompletării unor câmpuri. |
| **Verificare completare anunț** | Se vor verifica mecanismele de completare a unui anunț. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Completarea unui raport cu succes după completarea datelor specifice fiecărui tip de anunț. Notificarea utilizatorului în cazul necompletării unor câmpuri. |
| **Verificare încărcare anunț** | Se vor verifica mecanismele de încărcare a unui raport. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Încărcarea unui raport cu succes. Notificarea utilizatorului în cazul neselectării unui document din computerul personal. |
| **Verificare vizualizare status cerere** | Se vor verifica mecanismele de vizualizare a statusului unui raport. | Trimiterea unei cereri pentru interogarea statusurilor rapoartelor în baza de date. | Vizualizarea cu succes a statusurilor. |
| **Verificare vizualizare flux cerere** | Se vor verifica mecanismele de vizualizare a unui flux de transmisie. | Trimiterea unei cereri pentru interogarea fluxurilor din baza de date. | Vizualizarea cu succes a fluxurilor personalizate. |
| **Verificare arhivare cerere** | Se vor verifica mecanismele de arhivare a cererilor finalizate. | Trimiterea unei cereri pentru  verificarea rapoartelor pentru a putea fi arhivate de către utilizator. | Realizarea cu succes a operațiilor de arhivare. |
| **Verificare vizualizare cereri arhivate** | Se vor verifica mecanismele de vizualizare a cererilor arhivate. | Trimiterea unei cereri pentru interogarea cererilor arhivate baza de date. | Vizualizarea cu succes a cererilor arhivate anterior. |
| **Verificare căutare cerere arhivată** | Se vor verifica mecanismele de căutare a cererilor arhivate. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Realizarea cu succes a căutării cererii în funcție de atributele selectate. |
| **Verificare vizualizare grupuri** | Se vor verifica mecanismele de vizualizare a grupurilor. | Trimiterea unei cereri pentru interogarea grupurilor disponibile în baza de date. | Vizualizarea cu succes a grupurilor disponibile în cadrul platformei. |
| **Verificare modificare structură grupuri** | Se vor verifica mecanismele de modificare a structurii grupurilor. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Realizarea cu succes a modificării structurii unui grup selectat, urmată de notificarea utilizatorului cu mesaje specifice. |
| **Verificare vizualizare șabloane** | Se vor verifica mecanismele de vizualizare a modelelor-tip. | Trimiterea unei cereri pentru interogarea șabloanelor disponibile în baza de date. | Vizualizarea cu succes a șabloanelor din aplicație. |
| **Verificare încărcare șabloane** | Se vor verifica mecanismele de încărcare a șabloanelor. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Încărcarea cu succes a modelelor tip. |
| **Verificare adăugare categorie pentru șabloane** | Se vor verifica mecanismele de a categoriilor în cadrul șabloanelor. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Adăugarea cu succes a noilor categorii introduse de la tastatură. |
| **Verificare descărcare șabloane** | Se vor verifica mecanismele de descărcare a șabloanelor. | Trimiterea unei cereri pentru  verificarea șabloanelor pentru a putea fi descărcate de către utilizator. | Descărcarea cu succes a șabloanelor disponibile sau selectate. |
| **Verificare creare cont** | Se vor verifica mecanismele de creare a conturilor de utilizator. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Vizualizarea cu succes a mail-urilor de confirmare primite prin poșta electronică. |
| **Verificare vizualizare decizii** | Se vor verifica mecanismele de vizualizare a deciziilor anterioare. | Trimiterea unei cereri pentru interogarea cererilor disponibile în baza de date. | Vizualizarea cu succes a deciziilor anterioare luate asupra cererilor selectate. |
| **Verificare semnătura** | Se vor verifica mecanismele de verificare a semnăturii. | Trimiterea unei cereri pentru interogarea cererilor disponibile în baza de date. | Vizualizarea cu succes a semnăturii digitale aplicate de sistem asupra documentelor încărcate. |
| **Verificare selectare flux personalizat** | Se vor verifica mecanismele de selectare a unui flux personalizat. | Trimiterea unei cereri pentru interogarea fluxurilor personalizate disponibile în baza de date. | Selectarea cu succes a unuia dintre fluxurile personalizate create anterior și posibilitatea folosirii acestora la momentul încărcării unui document. |
| **Verificare generare flux personalizat** | Se vor verifica mecanismele de realizare a unui flux personalizat. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Generarea cu succes a unui nou flux de transmisie a documentelor sau cererilor. |
| **Verificare modificare structură flux personalizat** | Se vor verifica mecanismele de modificare a structurii unui flux personalizat. | Seturi de date pentru completarea field-urilor din cadrul paginilor. Acestea pot include și date greșite. | Modificarea cu succes a structurii unui flux personalizat prin adăugarea sau eliminarea unui utilizator din cadrul acestuia. |
| **Verificare selectare utilizatori destinatari individuali** | Se vor verifica mecanismele de selectare individuală a utilizatorilor destinatari. | Seturi de date pentru selectarea opțiunilor din cadrul paginilor. | Selectarea cu succes a utilizatorilor în mod individual pentru transmisia unui document. |

###### Tabel 1 - Testarea aplicației

## Probleme întâmpinate

Realizarea oricărui tip de aplicație poate întâmpina probleme de orice natură. În cazul proiectului curent, acestea au constat în următoarele aspecte:

* Dificultatea găsirii unei soluții optime de trimitere a mail-urilor de confirmare pentru utilizatorii care efectuează operații de înregistrare a contului
* Dificultatea integrării componentelor serviciului ElasticSearch
  + Inițial au fost implementate funcționalități de arhivare clasică pentru documentele finalizate, fiind mutate dintr-un tabel al bazei de date MySQL în altul specific
  + Pe aceste documente nu se putea realiza o căutare optimă și dinamică, motiv pentru care au fost configurate serviciile ElasticSearch și implementate funcționalitățile aferente
* Dificultatea realizării unui management optim pentru toate tipurile de documente
  + Dificultatea a venit în urma necesității realizării unui management al multiplelor tipuri de documente
  + Aceste documente au caracteristici diferite și sunt adresate mai multor tipuri de utilizatori în funcție de rolurile lor în cadrul aplicației
* Dificultatea stabilirii luării deciziilor asupra documentelor
  + La fel ca managementul tipurilor de documente, și cel al luării deciziilor asupra acestora a constituit un impediment în ceea ce privește realizarea rapidă a acestei funcționalități
  + Dificultatea a venit în urma transmiterii cererilor către utilizatorii cu roluri și autorizări diferite
  + Au fost realizate astfel secțiuni de decizie specifice fiecărui utilizator
* Dificultatea de diferențiere a cererilor sau anunțurilor încărcate sub forma unui document de cele completate în cadrul aplicației
  + problema a apărut în momentul stocării cererilor sau anunțurilor în baza de date deoarece se dorea diferențierea clară a celor completate de cele încărcate
  + S-au realizat tabele diferite și legături specifice prin intermediul cheilor private sau publice, păstrându-se elementele comune ale acestora
* Dificultatea gestionării optime a fluxului de transmisie către utilizatorii destinatari ai documentelor
  + Problema a apărut în urma încercării de acoperire a aproximativ tuturor modalităților de stabilire a utilizatorilor destinatari ai cererilor
  + Inițial exista un singur flux implicit, stabilit în momentul creării bazei de date
  + S-a încercat rezolvarea acestei probleme prin atribuirea utilizatorului de tip administrator a unor funcționalități de creare a fluxurilor pentru fiecare utilizator în parte, dar și acest lucru a constituit o problemă
  + Administratorul nu poate stabili fluxurile pentru fiecare utilizator, de aceea a fost adăugată o componentă de generare a propriilor fluxuri, în funcție de dorințele și posibilitățile fiecărui utilizator
  + O altă funcționalitate menită să rezolve această problema a fost posibilitatea selectării utilizatorilor în mod individual, fără a se alege un flux
  + În ceea ce privește luarea deciziilor, utilizatorul poate stabili în momentul selectării unui flux și dacă destinatarii pot sau nu întrerupe fluxul respectiv

# Concluzii

Luând în considerare evoluția societății din ultimii ani, se poate afirma că au apărut sau apar și în prezent diverși factori ce accelerează digitalizarea. Acest fenomen ia amploare prin apariția de noi tehnologii, atât hardware cât și software. Cert este faptul că managementul fluxului de documente reprezintă un aspect fără de care majoritatea instituțiilor nu ar putea funcționa corect și practic, indiferent de natura sau de scopul lor.

O astfel de abordare de introducere în cadrul conceptelor militare reprezintă o soluție cu potențial uriaș din perspectiva rezolvării problemelor la nivel birocratic, probleme des întâlnite în unitățile militare de pe suprafața țării. Birocrația este nu numai un mijloc de diferențiere între nivelurile de angajați, cât și o modalitate de gestionare a traficului de informații în interiorul organizației, iar digitalizarea ei ar eficientiza transferul de date atât din punct de vedere funcțional, structural, dar și din punctul de vedere al managementul timpului.

Prin intermediul proiectului curent s-a reușit:

* gestionarea numeroaselor tipuri de cereri în cadrul organizațiilor
* diferențierea și gestionarea eficientă a utilizatorilor
* transmiterea rapidă a informațiilor prin notificare
* evidențierea documentelor marcate ca fiind finalizate prin indexare/arhivare
* asigurarea autenticității documentelor prin intermediul semnăturii digitale
* îmbunătățirea transmisiei cererilor prin intermediul fluxurilor personalizate
* administrarea modelelor-tip într-un mediu controlat
* utilizarea serviciilor de poștă electronică pentru confirmarea conturilor
* gestionarea numeroaselor tipuri de cereri în cadrul organizațiilor

O astfel de implementare este utilă în zilele noastre deoarece constituie un mijloc eficient de diferențiere între nivelurile de angajați din cadrul unei organizații, o modalitate de gestionare a traficului de informații prin realizarea funcționalităților de fluidizare a transferului de documente. De asemenea constituie un mediu sigur pentru păstrarea și verificarea datelor, punând la dispoziția utilizatorilor o gamă largă de opțiuni într-un mediu compact.

Bibliografie

|  |  |
| --- | --- |
| [1] | M. E. Shacklett, „Authentication,” Available: https://www.techtarget.com/searchsecurity/definition/authentication. |
| [2] | Internet, „What Is a Digital Signature?,” . Available: https://www.instantssl.com/digital-signature. |
| [3] | A. Sims, „„THE DIFFERENCE BETWEEN DIGITAL SIGNATURES AND ELECTRONIC SIGNATURES,” Available: https://www.signix.com/blog/difference-between-electronic-and-digital-signature. |
| [4] | M. Steward, „Send mail on local host via Mercury with Xampp,” . Available: https://www.c-sharpcorner.com/UploadFile/c8aa13/send-mail-on-local-host-via-mercury-with-xampp/. |
| [5] | S. DAVIDSON, „TYPES OF ELECTRONIC SIGNATURES & WHEN TO USE THEM,” . Available: https://www.digicert.com/blog/types-of-electronic-signatures-and-when-to-use-them. |

# 

# Anexe

## ANEXA 1

@GetMapping("/verify-signature/{id}")

**public** Map<String, Boolean> verifySignature(@PathVariable("id") **int** id)

**throws** InvalidKeySpecException, NoSuchAlgorithmException, InvalidKeyException, SignatureException {

Map<String, Boolean> result = **new** HashMap<>();

CerereDocument doc = **this**.repoCerereDocument.findById(id).get();

KeyFactory keyFactory = KeyFactory.*getInstance*("RSA");

User userAssociated = doc.getCerere().getUserAssociated();

System.***out***.println("USER ASSOCIATED: " + userAssociated.getEmail());

X509EncodedKeySpec cheiePublica = **new** X509EncodedKeySpec(// pub key

doc.getCerere().getUserAssociated().getPublicKey());

PublicKey userPubKey = keyFactory.generatePublic(cheiePublica);

Signature publicSignature = Signature.*getInstance*("SHA1WithRSA");

publicSignature.initVerify(userPubKey);

publicSignature.update(doc.getContents());

**boolean** verificationStatus = publicSignature.verify(doc.getSignature());

result.put("verificationStatus", verificationStatus);

**return** result;

}

## ANEXA 2

**private** **void** saveCerereFlowForCerereAndUserStatus2Pending(Cerere cerereSalvata,

CerereDetailed cerereDetailedSalvata, User userInCharge, **int** priority) {

FlowCerere flow = **new** FlowCerere();

flow.setCerere(cerereSalvata);

flow.setCerereDetailed(cerereDetailedSalvata);

flow.setMotiv(**null**);

flow.setStatus(2);

flow.setCanInterrupt(1);

flow.setPriority(priority);

flow.setSuperior(userInCharge);

Message message = **new** Message();

message.setContents("New cerere added");

message.setDatePosted(**new** Date());

message.setMessageType("system");

message.setReceiver(userInCharge);

**this**.repoMessage.save(message);

**this**.repoFlow.save(flow);

}

## ANEXA 3

**public** String storeFile(MultipartFile file, String documentType, Integer idCerere) {

String fileName = StringUtils.*cleanPath*(file.getOriginalFilename());

**try** {

**if** (fileName.contains("..")) {

**throw** **new** RuntimeException("Sorry! Filename contains invalid path sequence " + fileName);

}

Path targetLocation = **this**.fileStorageLocation.resolve(fileName);

Files.*copy*(file.getInputStream(), targetLocation, StandardCopyOption.***REPLACE\_EXISTING***);

**if** (documentType != **null**) {

**if** (documentType.equals("cerere")) {

CerereDocument doc = **new** CerereDocument();

doc.setContents(file.getInputStream().readAllBytes());

doc.setDocumentType(file.getContentType());

Cerere cerere = **this**.repoCerere.findById(idCerere).get();

doc.setCerere(cerere);

doc.setFilename(fileName);

**try** {

KeyFactory keyFactory = KeyFactory.*getInstance*("RSA");

PKCS8EncodedKeySpec cheiePrivata = **new** PKCS8EncodedKeySpec(

cerere.getUserAssociated().getPrivateKey());

PrivateKey userPrivKey = keyFactory.generatePrivate(cheiePrivata);

Signature sig = Signature.*getInstance*("SHA1WithRSA");

sig.initSign(userPrivKey);

sig.update(doc.getContents());

**byte**[] signatureBytes = sig.sign();

doc.setSignature(signatureBytes);

} **catch** (Exception e) {

System.***out***.println("ERR GENERATING KEYS");

e.printStackTrace();

}

**this**.repoCerereDocument.save(doc);

Message message = **new** Message();

message.setContents("New cerere added");

message.setDatePosted(**new** Date());

message.setMessageType("system");

message.setReceiver(cerere.getUserAssociated());

**this**.repoMessage.save(message);

}

}

**return** fileName;

} **catch** (IOException ex) {

**throw** **new** RuntimeException("Could not store file " + fileName + ". Please try again!", ex);

}

}

## ANEXA 4

**public** Anunt storeFileAnunt(MultipartFile file, String title) {

String fileName = StringUtils.*cleanPath*(file.getOriginalFilename());

**try** {

**if** (fileName.contains("..")) {

**throw** **new** RuntimeException("Sorry! Filename contains invalid path sequence " + fileName);

}

Path targetLocation = **this**.fileStorageLocation.resolve(fileName);

Files.*copy*(file.getInputStream(), targetLocation, StandardCopyOption.***REPLACE\_EXISTING***);

Anunt anuntDocument = **new** Anunt();

anuntDocument.setTitlu(title);

anuntDocument.setContents(file.getInputStream().readAllBytes());

anuntDocument.setDocumentType(file.getContentType());

anuntDocument.setFilename(fileName);

Anunt saved = **this**.repoAnunt.save(anuntDocument);

**return** saved;

} **catch** (IOException ex) {

**throw** **new** RuntimeException("Could not store file " + fileName + ". Please try again!", ex);

}

}

## ANEXA 5

**public** CategorieSablon storeFileSablon(MultipartFile file, String sablonLeafCategoryNodeName,

**int** categorieParinteId) {

String fileName = StringUtils.*cleanPath*(file.getOriginalFilename());

**try** {

**if** (fileName.contains("..")) {

**throw** **new** RuntimeException("Sorry! Filename contains invalid path sequence " + fileName);

}

Path targetLocation = **this**.fileStorageLocation.resolve(fileName);

Files.*copy*(file.getInputStream(), targetLocation, StandardCopyOption.***REPLACE\_EXISTING***);

CategorieSablon categorieSablon = **new** CategorieSablon();

categorieSablon.setCategoryName(sablonLeafCategoryNodeName);

categorieSablon.setFile(file.getInputStream().readAllBytes());

categorieSablon.setDocumentType(file.getContentType());

CategorieSablon parinte = **this**.repoCategorieSablon.findById(categorieParinteId).get();

categorieSablon.setCategorieParinte(parinte);

categorieSablon.setFilename(fileName);

CategorieSablon saved = **this**.repoCategorieSablon.save(categorieSablon);

**return** saved;

} **catch** (IOException ex) {

**throw** **new** RuntimeException("Could not store file " + fileName + ". Please try again!", ex);

}

}

## ANEXA 6

@GetMapping("/cereri/all")

**public** Iterable<CerereESModel> findAllCereriES() {

**return** cerereESRepository.findAll();

}

@GetMapping("/cereri-pagination/{page}/{size}/{sort}/{order}")

**public** Page<CerereESModel> getCereriArchivedPaginated(@PathVariable("page") **int** page,

@PathVariable("size") **int** size, @PathVariable("sort") String sort, @PathVariable("order") String order) {

Pageable pageCurrent = PageRequest.*of*(page, size,

order.equals("asc") ? Sort.*by*(sort).ascending() : Sort.*by*(sort).descending());

**return** **this**.cerereESRepository.findAll(pageCurrent);

}

@GetMapping("/cereri/search-by/{categorySearch}/{cuvantCautat}/{page}/{size}/{sort}/{order}")

**public** Page<CerereESModel> searchCriteriuAndCuvantCautat(@PathVariable("cuvantCautat") String cuvantCautat,

@PathVariable("categorySearch") String categorySearch, @PathVariable("page") **int** page,

@PathVariable("size") **int** size, @PathVariable("sort") String sort, @PathVariable("order") String order) {

Pageable pageCurrent = PageRequest.*of*(page, size,

order.equals("asc") ? Sort.*by*(sort).ascending() : Sort.*by*(sort).descending());

Page<CerereESModel> result = **null**;

**if** (categorySearch.equals("motiv")) {

result = cerereESRepository.searchCustomQueryMotiv(cuvantCautat, pageCurrent);

} **else** **if** (categorySearch.equals("email")) {

result = cerereESRepository.searchCustomQueryEmail(cuvantCautat, pageCurrent);

} **else** **if** (categorySearch.equals("judet")) {

result = cerereESRepository.searchCustomQueryJudet(cuvantCautat, pageCurrent);

}

**return** result;

}

## ANEXA 7

@GetMapping("/all") //

**public** Iterable<Anunt> getAllAnunturi(){

**return** repoAnunt.findAll();

}

@GetMapping("/all-paginated/{page}/{size}/{sort}/{order}")

**public** Page<Anunt> allUsersPaginated(@PathVariable("page") **int** page, @PathVariable("size") **int** size, @PathVariable("sort") String sort, @PathVariable("order") String order){

Pageable pageCurrent = PageRequest.*of*(page, size , order.equals("asc") ? Sort.*by*(sort).ascending() : Sort.*by*(sort).descending());

Page<Anunt> messages = repoAnuntPagination.findAll(pageCurrent);

**return** messages;

}

**private** **byte**[] generatePdfInFilesystem(Anunt anunt) **throws** FileNotFoundException, IOException {

Document document = **new** Document();

ByteArrayOutputStream byteArrayOutputStream = **new** ByteArrayOutputStream();

**try** {

PdfWriter.*getInstance*(document, byteArrayOutputStream);

document.open();

Paragraph p = **new** Paragraph();

p.add(anunt.getTitlu());

p.setAlignment(Element.***ALIGN\_CENTER***);

document.add(p);

Paragraph p2 = **new** Paragraph();

p2.add(anunt.getContinut()); // no alignment

document.add(p2);

Font f = **new** Font();

f.setStyle(Font.***BOLD***);

f.setSize(8);

document.add(**new** Paragraph("This is my paragraph 3", f));

// close

document.close();

System.***out***.println("Done");

} **catch** (DocumentException e) {

e.printStackTrace();

}

**byte**[] pdfBytes = byteArrayOutputStream.toByteArray();

**return** pdfBytes;

}

@GetMapping("/generate-anunt-pdf/{id}")

**public** ResponseEntity<**byte**[]> generatePdf(@PathVariable("id") **int** id) **throws** InterruptedException, FileNotFoundException, IOException {

Optional<Anunt> anuntCautatOptional = repoAnunt.findById(id);

**if** (anuntCautatOptional.isPresent()) {

//

**byte**[] fisierBytes = **this**.generatePdfInFilesystem(anuntCautatOptional.get());

HttpHeaders headers = **new** HttpHeaders();

headers.setContentType(MediaType.***APPLICATION\_PDF***);

// Here you have to set the actual filename of your pdf

String filename = anuntCautatOptional.get().getTitlu()+".pdf";

headers.setContentDispositionFormData(filename, filename);

headers.setCacheControl("must-revalidate, post-check=0, pre-check=0");

ResponseEntity<**byte**[]> response = **new** ResponseEntity<>(fisierBytes, headers, HttpStatus.***OK***);

**return** response;

}

**return** **null**;

}

## ANEXA 8

@GetMapping("/select-by-id/{id}")

**public** Anunt selectById(@PathVariable("id") **int** id) **throws** InterruptedException {

Optional<Anunt> anuntCautatOptional = repoAnunt.findById(id);

**if** (anuntCautatOptional.isPresent()) {

**return** anuntCautatOptional.get();

}

**return** **null**;

}

## ANEXA 9

@PostMapping("/associate-groups-with-anunt/{idAnunt}")

**public** Anunt associateWithGroups(Principal principal, @PathVariable("idAnunt") **int** idAnunt, @RequestBody List<Integer> groupIds) **throws** InexistingUserException {

Anunt anunt = **this**.repoAnunt.findById(idAnunt).get();

Iterable<Group> allGroupsForAnunt = **this**.repoGroup.findAllById(groupIds);

Set<User> userDestinari = **new** HashSet<>();

**for**(Group g : allGroupsForAnunt) {

userDestinari.addAll(g.getStudents());

}

**for**(User user: userDestinari) {

user.getAnunturi().add(anunt);

**this**.messageService.sendMessage(principal, user, MessageService.***NEW\_MESSAGE\_ANUNT***, MessageType.***CHAT***);

}

**this**.repoUser.saveAll(userDestinari);

**return** anunt;

}

## ANEXA 10

@PutMapping("/update")

**public** Anunt updateAnunt(@RequestBody Anunt anuntModificat, Principal principal) {

Optional<User> user = **this**.userService.getUser(principal);

**if**(!user.isPresent()) {

**throw** **new** RuntimeException("Nobody logged in");

}

**boolean** hasAccess = **false**;

Anunt anuntDb = **this**.repoAnunt.findById(anuntModificat.getId()).get();

hasAccess = **this**.userService.isAdmin(user.get())

||

(anuntDb .getPoster() != **null** ? anuntDb .getPoster().equals(user.get()) : **false**) ;

**if**(hasAccess){

**return** repoAnunt.save(anuntModificat);

}

**throw** **new** RuntimeException("No access");

}

@DeleteMapping("/delete/{id}")

**public** Anunt deleteAnunt(@PathVariable("id") **int** id) {

Anunt anuntulSters = repoAnunt.findById(id).get();

repoAnunt.delete(anuntulSters);

**return** anuntulSters;

}