MANAGEMENTUL ACTIVITĂȚILOR DE SECRETARIAT DINTR-O UNIVERSITATE

# Descrierea aplicației

Scopul nostru este de a administra într-un mod cât mai practic activitățile unei universități, prin intermediul unei aplicații web. Cu ajutorul acesteia, se încearcă centralizarea tuturor datelor, atât ale studenților, cât și ale angajaților, într-o singură bază de date, la care fiecare va avea acces în funcție de gradul său de permisiune. Prin implementarea mai multor tipuri de conturi, și anume student, profesor, secretar și admin, se vor pune la dispoziție meniuri personalizate, care vor ușura munca atât a personalului, cât și a studenților. De exemplu, unui profesor i se oferă posibilitatea vizualizării orarului personal, notării studenților, vizualizare de statistici și depunere de documente standardizate. La fel și in cazul studenților, li se oferă posibilitatea accesării catalogului, întocmirii de rapoarte, vizualizarea orarului, ș.a.m.d. .

# Scenariile de utilizare

## Administratorul

Are gradul de permisiune cel mai ridicat, în sensul în care acesta este cel care poate adauga sau șterge orice tip de cont, atât de student, cât și de angajat (profesor, admin , sau secretar). Admin-ul are și permisiunea de a reseta parolele sau numele utilizatorilor, are posibilitatea de arhivare a datelor despre studenții care au absolvit, poate actualiza date despre sălile de clasă existente și se poate ocupa și de reprogramarea unui modul în orar.

## Secretarul

Secretarul este cel care se ocupă de gestionarea conturilor de tip student și profesor. Acesta are posibilitatea de a vizualiza studenții, sălile de clasă, de a crea conturi de student sau profesor (respectiv ștergere), de a adăuga materii noi în programă, de a crea documente necesare profesorului/studentului și de a vedea datele personale ale studenților și profesorilor.

## Studentul

Are posibilitatea de generare a unui raport tipizat, în funcție de tipul acestuia, poate vizualiza orarul personalizat în funcție de dată și poate vizualiza notele obținute în funcție de materie.

## Profesorul

Are posibilitatea de a vizualiza orarul personal, în funcție de dată, poate de asemenea vizualiza care este media studenților la o anumită materie, sau care sunt stdenții cu restanță, poate nota studenții direct din aplicația web și poate cere o situație cu notele fiecărui student, în funcție de nume.

# Funcționarea aplicației

La lansarea aplicației web, pagina inițială este cea de Log In. În funcție de ce tip de cont folosim, vom fi redirecționați către o interfață diferită. Exemple de username și parolă în funcție de rol:

* Admin username: admin.admin@mta.ro & parola: 123456789
* Secretar username: maria.ion@mta.ro & parola: 123456
* Student username: irimin.dan@mta.ro & parola: 123456789
* Profesor username: avram.dan@mta & parola: 123456789

În cadrul interfeței **Admin**, sunt implementate următoarele funcționalități:

* Adaugare Angajat, unde trebuie introduse numele și prenumele acestuia, este setată parola, se selecteaza funcția și rolul, apoi salariul
* Adaugare Student, unde se introduc numele și prenumele acestuia, se setează parola, se alege arma și apoi grupa, iar apoi se introduce valoarea soldei
* Ștergere Angajat, unde se introduc numele și prenumele angajatului pe care dorim să îl eliminăn din baza de date
* Ștergere Student, unde introducem de asemena numele și prenumele studentului
* Resetare Parolă, introducem username-ul pentru care dorim să facem modificarea parolei, apoi noua parolă
* Resetare Username, unde se introduce numele de utilizator actual, apoi cel cu care dorim să fie înlocuit
* Arhivare Studenți, trebuie doar apăsat butonul de arhivare și se va genera o arhiva .zip cu toate datele despre fiecare student din an terinal
* Actualizare Clasă, unde trebuie să alegem clasa pe care dorim să o actualizăm, selectăm tipul acesteia, apoi introducem capacitatea
* Actualizare Orar, unde alegem o materie și data + ora la care dorim să adaugăm modulul în orar

În cadrul interfeței **Secretar**, sunt implementate următoarele funcționalități:

* Adaugare Student, unde se introduc numele și prenumele acestuia, se setează parola, se alege arma și apoi grupa, iar apoi se introduce valoarea soldei
* Adaugare Profesor, unde trebuie introduse numele și prenumele acestuia, este setată parola, se selecteaza rolul, apoi salariul
* Ștergere Student, unde introducem numele și prenumele studentului
* Ștergere Profesor, unde se introduc numele și prenumele acestuia
* Creare Document, unde introducem numele și prenumele persoanei care dorește să genereze documentul, apoi titlul și conținutul
* Adăugare Materie, unde inttroducem denumirea acesteia, numele și prenumele profesorului și numărul de credite
* Vizualizare Student, unde introducem numele și prenumele studentului despre care dorim să aflăm informații (exemplu Popescu Liliana)
* Vizualizare Profesor, la fel ca și in cazul studentului, sunt introduse numele și prenumele (exemplu Avram Dan)
* Vizualizare Clasă, se introduce numele clasei (exemplu A0-01)
* Vizualizare Cursuri afișează un panou interactiv cu toate materiile și informații despe acestea, care sunt existente în baza de date
* Vizualizare Studenți, afișează toate informațiile despre toți studenții din baza de date

În cadrul interfeței **Student**, sunt implementate următoarele funcționalități:

* Afișare Orar, unde se introduce data pentru care dorim să aflăm ce module sunt stabilite (exemplu pentru Irimin Dan: 04 Octombrie 2021)
* Afișare Note, unde alegem materia pentru care dorim sa aflăm nota obținută (exemplu pentru Irimin Dan: POO )
* Creare Raport, unde introducem numele și prenumele persoanei către care înaintăm raportul, apoi tipul acestuia

În cadrul interfeței **Profesor**, sunt implementate următoarele funcționalități:

* Vizualizare Orar, unde introducem data pentru care dorim să aflăm orarul (exemplu pentru Avram Dan: 04 Octombrie 2021)
* Vizualizarea Mediei, unde selectăm numele materiei pentru care dorim să aflăm media notelor
* Vizualizare Studenți Restanți, unde selectăm materia pentru care dorim să vedem dacă există astfel de cazuri (exemplu POO)
* Vizualizare Statistici, unde alegem materia dorită
* Adăugare Notă, unde introducem numele și prenumele studentului pe care dorim să îl notăm, selectăm materia, nota și apoi data
* Vizualizare Note Student, unde introducem numele și prenumele studentului pentru care vrem să aflăm toate notele ( exemplu Popescu Liliana)

Funcția de Sign Out este implementată pentru fiecare tip de cont.

# Funcționalități neimplementate

* Posibilitatea de a schimba grupa sau arma din care face parte un anume student
* Posibilitatea vizualizării orarului unei săli
* Posibilitatea afișării atât a materiilor anterioare(împreună cu notele obținute) dar și a celor viitoare și curente pentru un anumit student, în funcție de facultatea și arma acestuia
* Arhiva cu datele despre studenții absolvenți este generată local, nu introdusă într-o bază de date

# Arhitectura aplicației

Pentru implementarea aplicației administrative am utilizat 3 mari componente:

* Aplicație server
* Aplicație client
* Baza de date

Pentru implementarea aplicației server am utilizat framework-ul utilizat împreuna cu limbajul Java numit Spring. Această aplicație pune la dispoziție aplicației client posibilitatea de a lucra cu datele angajaților și studenților ce sunt stocate în baza de date. Aceasta expune mai multe rute ce sunt accesate din aplicația client pentru a prelucra datele necesare implementării proiectului. Cel mai important modul din aplicația este cel ce lucrează cu baza de date, reprezentat de clasa Database care conține toate metodele prin care baza de date este interogată pentru a prelucra diferite informații.

Aplicația client reprezintă interfața dintre aplicația server și utilizator. Aceasta bune la dispoziție diferite pagini web care permit utilizatorului să acceseze diferite informații sau să facă anumite modificări sau adăugiri in funcție de necesitatea sa. Pentru implementarea acestei aplicații a fost utilizat framework-ul Vaadin. Pentru a conecta aplicația client cu back-end-ul am utilizat modulul reprezentat de clasa ApiRequest ce pune la dispozitie metoda necesară pentru a face un request către back-end.

Pentru lucrul cu o bază de date am utilizat serviciile celor de la Microsoft Azure, de la care am folosit un server de tip Microsoft SQL Server pentru utilizarea unei baze de date.