UNIVERSITATEA POLITEHNICA BUCUREȘTI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEPARTAMENTUL CALCULATOARE

|  |  |
| --- | --- |
| upb | cs |

PROIECT DE DIPLOMĂ

**Aplicație web pentru managementul învățării**

versiunea 2024

Panait Ștefan-Andrei

**Coordonator științific:**

Prof. dr. ing. Andrei Ionescu

BUCUREŞTI

2024

UNIVERSITY POLITEHNICA OF BUCHAREST

FACULTY OF AUTOMATIC CONTROL AND COMPUTERS

COMPUTER SCIENCE DEPARTMENT

|  |  |
| --- | --- |
| upb | cs |

DIPLOMA PROJECT

Aplicație web pentru managementul învățării

2024 version

Ioana Popescu

**Thesis advisor:**

Prof. dr. ing. Andrei Ionescu

BUCHAREST

2024

**CUPRINS**

[Sinopsis 2](#_Toc7074723)

[Abstract 2](#_Toc7074724)

[Mulțumiri 3](#_Toc7074725)

[1 Introducere 4](#_Toc7074726)

[1.1 Context 4](#_Toc7074727)

[1.2 Problema 4](#_Toc7074728)

[1.3 Obiective 4](#_Toc7074729)

[1.4 Structura lucrării 4](#_Toc7074730)

[2 Analiza și specificarea cerințelor 5](#_Toc7074731)

[3 Studiu de piață / Abordări existente 6](#_Toc7074732)

[3.1 Indicații formatare figuri 6](#_Toc7074733)

[4 Soluția propusă 8](#_Toc7074734)

[4.1 Indicații formatare formule 8](#_Toc7074735)

[5 Detalii de implementare 9](#_Toc7074736)

[5.1 Indicații formatare tabele 9](#_Toc7074737)

[6 Studiu de caz / Evaluarea rezultatelor 11](#_Toc7074738)

[7 Concluzii 13](#_Toc7074739)

[8 Bibliografie 14](#_Toc7074740)

[9 Anexe 16](#_Toc7074741)

# Sinopsis

Aplicația web pentru managementul învățării este o platformă ușor de înțeles pentru a facilita educația și studiul online. Această avand o interfață prietenoasă pentru profesori și cursanți să interacționeze, gestioneze cursuri, primimeasca recenzie și să acceseze diferite resurse educaționale.

Aplicația ofera autentificare securizata pentru utilizatori, personalizarea profilului utilizator pentru studenti si instructor, si control de acces pe baza de rol pentru a modifica permisiunile la functionalitati. Profesorii pot modifica modulele cursurilor, sigla si pot da teme elevilor, de asemenea pot da recenzie si note.

# Mulțumiri

(opțional) Aici puteți introduce o secțiunea specială de mulțumiri / acknowledgments.

# Introducere

In domeniul cu o dezvoltare foarte rapida al invatamatului, nevoia de eficienta, flexibilitate, accesibilitate nu a fost niciodata mai mare. Aceasta aplicatie de management al invatarii adreseasa aceasta nevoie, oferind o platforma online complexa pentru educatie si dezvoltare. Aceasta aplicatie este modelata pentru a imbunatatii experienta educationala a studentilor, profesorilor, dar si a unitatilor de invatamant.

La baza aplicatia ofera o interfata centralizata si prietenoasa care faciliteaza interactiunea dintre profesori si cursanti. Cu capabilitati ce varieaza de la creeare corsului si gestionarea acestuia pana la livrarea de continut si evaluare detaliata. Concentrandu-se pe accesibilitate si eficienta, aplicatia web asigura ca utilizatorii pot naviga cu usurinta pe platforma, accesa resurse si pot ramane implicati in calatoria lor de invatare.

## Context

In ultimi ani a fost o schimbare drastica care duce tot mai mult catre educatie online. Aceasta tranzitie a subliniat importanta unei platforme digitale bine dezvolatate pentru a asigura continuitatea educatiei.

## Problema

Invatarea suplimentare este foarte dificila in contextul traditional in care profesorii si elevii sunt constransi din mai multe considerente:

* Limitarile de timp si spatiu
* Resurse limitate
* Lipsa interactivitatii

## Obiective

Imbunatatirea calitatii in invatarea suplimentara atat pentru studenti cat si pentru profesori.

Beneficii pentru profesor:

1. Adaugarea materialelor educationale: Profesorii pot incarca materiale educationale care sunt disponibile elevilor in orice moment
2. Crearea si gestionarea de teme: Profesorii pot adauga probleme in lista de exercitii, de altfel acestia pot alege orice problema din aceasta lista pentru a creea o tema pentru studenti.
3. Evaluarea temelor: Dupa incarcarea temelor de catre studenti profesorii le pot evalua oricand.

Beneficii pentru studenti:

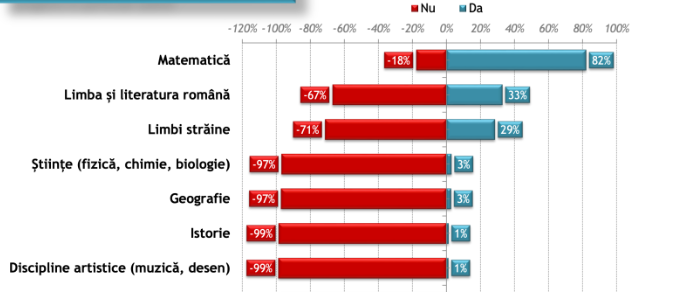
1. Acces la materiale educationale: Studentii pot vizualiza materialele educationale disponibile in orice moment.
2. Incarcarea temelor: Acestia pot incarca temele finalizate asigurandu-se ca lucrarile lor sunt evaluate de profesori.
3. Evaluare: Dupa evaluare acestia pot observa notitele profesorilor in legatura cu tema cat si rezolvarea corecta a acesteia.

## Structura lucrării

Un paragraf în care fiecare dintre secțiunile următoare este prezentată în 1-2 fraze, punând accentul pe elementele cele mai semnificative din fiecare secțiune.

# Analiza și specificarea cerințelor

Putem remarca faptul ca in Romania cultura meditatiilor este una foarte dezvoltata, fiecare parinte isi trimite copilul la meditatii macar in cei mai importanti ani din ciclul de invatamant al copiilor acestora. Un studiu la vinel national realizat de IRES si Societatea Academica din Romania spune ca „Conform studiului realizat de IRES și SAR în rândul părinților, o treime dintre elevi participă la sistemul de meditații, cu o frecvență considerabilă în clasele terminale, atunci când examenele naționale se apropie. În urma chestionării elevilor de gimnaziu și liceu, a rezultat faptul că un sfert dintre elevii din România accesează sistemul de meditații la cel puțin o materie.”[[1]](#footnote-1)



Comparand aceste informatii cu alte tari:

* In Italia 1 din 5 elevi fac meditatii conform unei statistici realizate pe 6000 de elevi de ripetizioni.it, platforma italiana pentru mediatii atat online cat si fizic.[[2]](#footnote-2)
* In germania la fel ca in italia 1 din 5 elevi se adreseaza profesorilor pentru meditatii in invatamantul liceal conform lucrarii stiintifice „Effectiveness of private tutoring during secondary schooling in Germany: Do the duration of private tutoring and tutor qualification affect schoolachievement?” [[3]](#footnote-3)

Din aceste statistici putem concluziona ca studentilor nu le sunt de ajuns orele de pregatire din cadrul scolilor. Astfel aceasta platforma ar actiona ca un pilon educational pentru student.

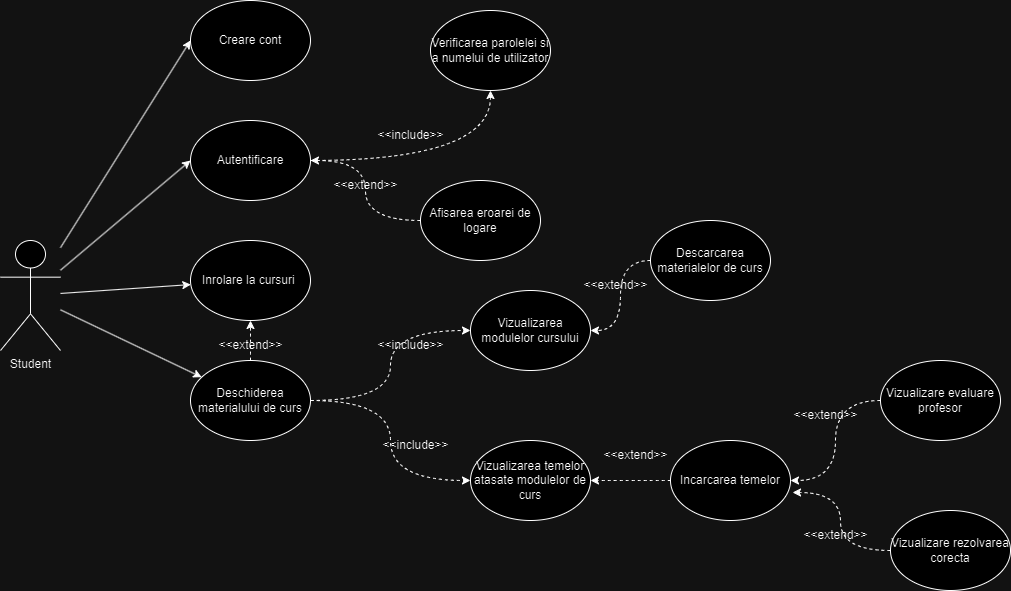
## Cerinte nonfunctionale

Cerintele nonfunctionale se refera la specificatiile care descriu cum ar trebui sa se comporte un sistem, acestea se concentreaza pe calitatea si performanta aplicatiei si sunt esentiale pentru a asigura o experienta optima pentru utilizatori.

1. Securitate: datele utilizatorilor sunt protejate la orice moment de timp
2. Performanta: timpii de raspuns si viteza aplicatiei sa nu fie un inpediment in folosirea acesteia
3. Viteza: Aplicatia trebuie sa raspunda cerintelor intrun timp optim.
4. Fiabilitate: Aplicatia v-a functiona fara erori care sa afecteze disponabilitatea aplicatiei.

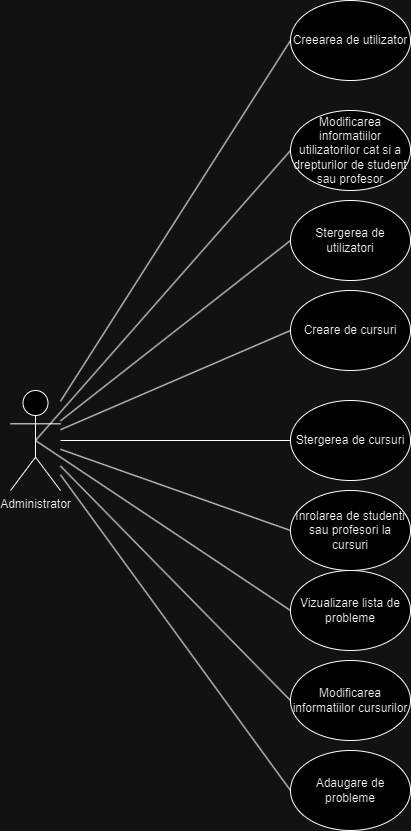
## Cerinte functionale

Cerintele functionale se axeaza pe cum aplicatia functioneaza daca atinge sau nu anumite caracteristici dorite. In aplicatia propusa sunt 3 roluri: student, profesor si administrator. In continuare voi prezenta procesul de folosire a aplicatiei atasand diagrame ale cazurilor de utilizare.



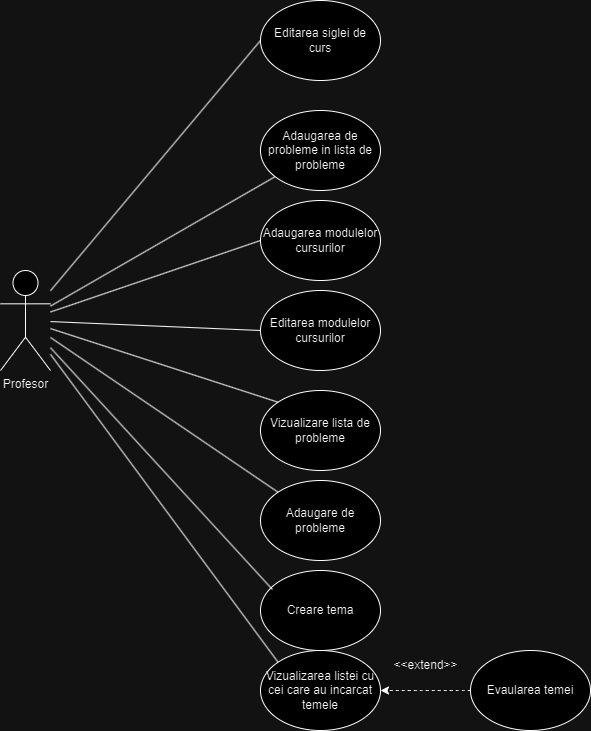
In rolul de student procesul este urmatorul:

1. Creearea contului: Oricarei creeare de cont nou ii este atribuit automat rolul de student.
2. Autentificarea: La autentificare studentul isi adauga credentialele urmand ca acestea sa fie verificate. Daca acestea sunt corecte utilizatorul este redirectat pe pagina unde sunt vizibile cursurile. Daca credentialele sunt incorecte acestuia ii va fi afisat un mesaj de eroare.
3. Inrolare la cursuri: Utilizatorul se poate inrola singur la cursurile la care crede ca are nevoie de studiu suplimentar.
4. Deschiderea materialului de curs: Dupa inrolarea la curs este accesibila deschiderea si privirea materialelor disponibile.
5. Vizualizarea modulelor cursului: Dupa deschiderea materialului de curs sunt vizibile modulele cursului. Un exemplu de modul: Titlul: „Polinoame”, Descriere: „Acest modul te va ajuta sa intelegi polinoamele” si un fisier atasat Suport curs: polinoame.pdf.
6. Descarcarea suportului de curs: Studentul poate descarca suportul de curs.
7. Vizualizarea temelor atasate modulelor de curs: Pentru fiecare modul cursantul poate avea atasat o tema.
8. Incarcarea temelor: Dupa ce studentul considera ca tema este rezolvata acesta poate incarca tema pentru evaluarea unui profesor.
9. Vizualizare evaluare profesor: Elevul poate observa comentariile profesorului asupra temei incarcate.
10. Vizualizare rezolvare corecta: Dupa evaluarea temei de catre profesor rezolvarea corecta a exercitiului este disponibila elevului.



In rolul de administrator procesul este urmatorul:

1. Creearea de utilizator: administratorul poate creea utilizatori
2. Modificarea informatiilor utilizatorilor cat si a drepturilor de student sau profesor: administratorul poate schimba informatiile utilizatorului de ex numele, rolul sau adresa de email.
3. Stergerea de utilizatori
4. Creearea de cursuri: administratorul poate creea cursuri
5. Modificarea informatiilor cursurilor: Administratorul poate schimba titlul, descrierea cat si categoria din care face parte cursul
6. Stergerea de cursuri
7. Inrolarea de studenti sau profesori la cursuri: Elevul se poate inrola si singur la cursuri dar profesorul se poate inrola doar daca este inrolat de administrator. Ratiunea din spatele acestei functionalitati este ca doar un administrator poate stii daca un profesor este bun la o materie. Astfel nu se v-a intampla de exemplu ca un profesor de Informatica sa corecteze si sa puna materiale la Geografie.
8. Vizualizare lista de probleme: Administratorul poate vizualiza lista de probleme, acesta poate edita problemele sau de asemenea sterge probleme.
9. Adaugare de probleme: Administratorul de asemenea poate adauga probleme.



In rolul de profesor procesul este urmatorul:

1. Editarea siglei de curs: profesorul poate schimba sigla la cursurile la care acesta este inrolat.
2. Adaugarea de probleme in lista de probleme: profesorul poate adauga probleme
3. Adaugarea modulelor cursurilor: acesta poate adauga module noi pentru curs
4. Editarea modulelor de curs: Acesta poate edita titlul modulului, descrierea, cat si poate sterge sau incarca suportul de curs aferent modulului.
5. Vizualizare lista de probleme: Profesorul poate vizualiza lista de probleme pentru a alege o problema din aceea lista pe care sa o dea ca tema.
6. Adaugare de probleme: Acesta poate sa adauge probleme.
7. Creare de tema: Profesorul poate creea teme alegand o problema din lista de probleme. Acesta poate de asemenea sa seteze un o zi limita pentru incarcarea acesteia.
8. Vizualizarea listei cu elevii care au incarcat temele: Profesorul poate vizualiza lista cu cei care au incarcat temele, apoi cu ajutorul unui modul de cautare acesta poate cauta dupa mai multe criterii cum ar fii titlul temei sau numele elevului.
9. Evaluarea temei: Acesta poate da o recenzie temelor incarcate cat si o nota.

# Studiu de piață / Abordări existente

[Dezvoltare de produs] Ce soluții similare există pe piață? Care sunt limitările lor / pentru ce cazuri de utilizare sau pentru ce tip de clienți produsele existente pe piață nu răspund cerințelor? Care sunt indicatorii pe baza cărora sunt evaluate aceste produse, de către potențiali clienți, și unde sunt lipsurile/ care este oportunitatea generată de lipsurile acestea?

[Cercetare] Metode existente (sau „State of the Art“) se referă, de regulă, la nivelul curent de dezvoltare: care este starea curentă a domeniului, unde ne găsim, care este contextul. Care sunt soluțiile actuale prezente în literatura de specialitate și care sunt limitările lor? Ce direcții de explorare sunt recomandate în literatura de specialitate? Literatura de sp

* Dezvoltare de produs] Sunt analizate superficial câteva produse de pe piață;
* [Cercetare] analiza literaturii limitata la grupuri de cercetare din România;
* [Ambele] Sunt descrise tehnologiile folosite în lucrare.

Criterii pentru calificativul *Satisfăcător*:

* [Dezvoltare de produs] Există un interviu, un client, analiza cerințelor este elaborată pe baza interviului.
* [Cercetare] analiza literaturii de specialitate din lume, fără poziționarea precisă a lucrării în peisajului domeniului studiat;
* [Ambele] Sunt descrise câteva tehnologii alternative pentru fiecare din tehnologiile folosite în lucrare. Există o argumentare referitoare la alegere.

Criterii pentru calificativul *Bine*:

* [Dezvoltare de produs] Proces iterativ pe baza unor interviuri cu mai mulți clienți, dezvoltare MVP, reevaluare cerințe;
* [Cercetare] analiza literaturii de specialitate din lume, cu poziționarea precisă a lucrării în peisajul actual al domeniului studiat;
* [Ambele] Sunt descrise tehnologii alternative. Sunt analizate cantitativ și calitativ, folosite benchmarkuri și teste efectuate de student. Analiza este rezumată prin tabele și grafice.

## Indicații formatare figuri

Figurile utilizate în document vor fi centrate și numerotate (de exemplu Figura 1).

Orice figură ce nu este realizată de către autorul lucrării va fi în mod obligatoriu citată fie la final (de exemplu Figura 2 este preluată din documentul [1]), fie cel puțin într-o notă de subsol (a se vedea Figura 2). Orice figură ce depășește ca dimensiune 50% dintr-o pagină, va fi mutată la anexe. Toate figurile din cadrul tezei vor fi referite în text. Exemplu: Figura 1 prezintă o schemă de principiu pentru un amplificator inversor cu AO.

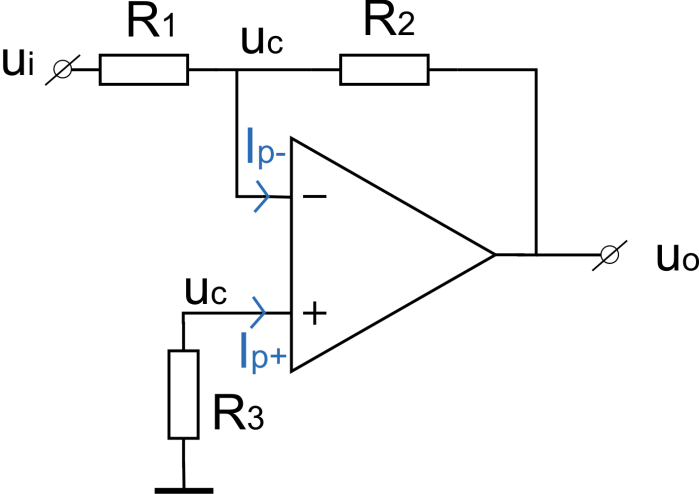


Figura 1 Amplificator inversor

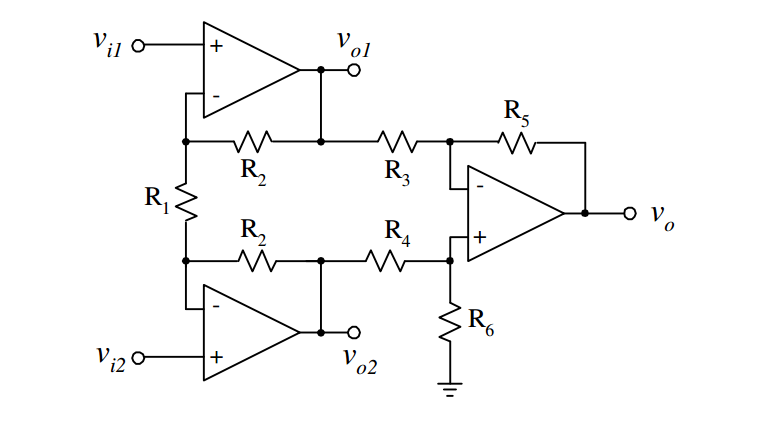


Figura 2 Amplificator de instrumentație cu 3 AO-uri [[4]](#footnote-4)

# Soluția propusă

Abordarea pe care am ales sa merg provine din documentarea pietei la momentul actual. Astfel dupa o analiza amanuntita am observat ca o buna parte din marile companii precum JPMorgan Chase Bank, Intellibus, Deutsche Bank si multe altele folosesc Java Spring Boot pentru backend. Pentru frontend am ales sa merg pe Angular folosind typescript cu toate ca React este mai popular in randul aplicatiilor mai mici dar pentru proiectele mari este mai folosit Angular. Pentru baza de date am ales sa merg pe o baza de date relationala, respectiv MySql, aceasta fiind open-source, de asemenea fiind cunoscuta pentru performanta si scalabilitate.

## Spring Boot

Spring este cel mai popular framework de java. Titulatura de cel mai popular framework este obtinuta din viteza, simplitatea si productivitatea acestuia.

Spring este un framework pentru a construi aplicatii java, open-source, care ofera o infrastructura complexa si usor de mentinut pentru dezvoltarea aplicatiilor. Principalele avantaje in folosirea Spring:

1. Inversion of control – acesta este un principiu de proiectare care ofera modularitate programului, oferind frameworkului control asupra unor portiuni de cod sau a unor obiecte. Acest lucru este realizabil prin Dependency Injection. Dependency Injection este mecanismul prin care controlul este inversat de la programator la framework.
2. Accesul la date – simplificarea accesului la date folosind tehnologii precum Hibernate, JPA (Java persistence API), MyBatis, EclipseLink. Acestea reduc complexitatea interactiunii cu baza de date.
3. Ideal pentru construirea de RESTful API datorita adnotarilor simpliste: @RestController, @GetMapping, @PostMapping simplifică definiția și gestionarea. Aceste adnotari fac codul usor de citit si intretinut permitand o dezvoltare usor facila a enpointurilor de tip CRUD (create, read, update, delete).
4. Securitatea – securitatea este usor de facilitat datorita frameworkului Spring Security JWT (json-web token) care ajuta in autentificare si autorizare. De asemenea ofera protectie impotriva atacurilor de tip CSRF

----------------------------------ACCESS LA DATE SPECIFIC PE APLICATIA Mea------------------------------

In solutia mea am ales sa folosesc JPA impreuna cu hibernate. JPA fiind o solutie de a controla accesul la baza de date iar hibernate implementarea acestuia. Aceasta abstractizeaza toata complexitatea implementarii in spatele unor interfete si reguli.

JPA si hibernate sunt un mod de programare object-relational mapping, acesta fiind un mod de a alinia codul cu baza de date. Se structureaza foarte bine datorita modului de programare orientat pe obiecte, fiecare clasa din codul java este mapata la un tabel.

Am ales sa folosesc acest model la acces de date fata de un model ca MyBatis pentru optimizarea si abstractizarea pe care o aduce jpa si hibernate acestea avand incorporat caching.

Cu toate ca MyBatis ofera mai mult control asupra interogarilor sql, am decis ca automatizarea ORM este de ajutor pentru mine datorita faptului ca mentinerea interogarilor este mult mai usor de facilitat.

----------------------------------------------------SECURITATE------------------------------------------------------

Pentru partea de securitate folosesc Spring Security si json-web token pentru autentificare si autorizarea diferitelor roluri din aplicatie. Odata ce userul se autentifica acestuia ii este generat un jwt token care este trimis din frontend inapoi, in fiecare cerere pe care acesta o face, pentru a confirma ca acesta este insusi userul pentru care tokenul a fost generat.

Explicație Detaliată

Autentificare: Utilizatorul introduce credențialele (de exemplu, nume de utilizator și parolă) și trimite o cerere de autentificare către server.

Serverul verifică credențialele utilizatorului. Dacă autentificarea este reușită, se generează un token JWT care poate contine informatii despre utilizator si rolurile acestuia. Aceste informatii se numesc claims.

Tokenul JWT este semnat folosind o cheie secretă pentru a asigura integritatea și autenticitatea acestuia.

Tokenul JWT generat este trimis către frontend și stocat de obicei in memoria locala a browserului sau un cookies.

Autorizare:

La fiecare cerere ulterioara din frontend este trimis alaturi de informatiile cererii in antet tokenul. Serverul verifica tokenul asupra validitatii, daca tokenul este valid acesta permite cererea altfel nu trimitand o eroare.

## Angular

Angular este un framework open-source care se bazeaza pe typescript si ruleaza pe Node.js pentru creearea aplicatiilor web.

De ce Typescript si nu Javascript?

Unul dintre cele mai mari avantaje pe care consider ca le are typescript asupra javascript este faptul ca acesta are erori de compilare datorita static typing-ului in loc de dynamic typing pentru javascript.

Angular se bazeaza pe componente, utilizatorul avand abilitatea de a-si definii singur componente. In loc de <div></div> pot avea <my-component></my-component>. De asemenea acesta este un foarte mare avantaj acestea fiind similare claselor din programarea orientata pe obiecte. Proprietatile care le fac sa se asemene sunt organizarea si claritatea codului mult mai optima, reutilizarea codului si intretinerea mult mai facila.

## MySQL

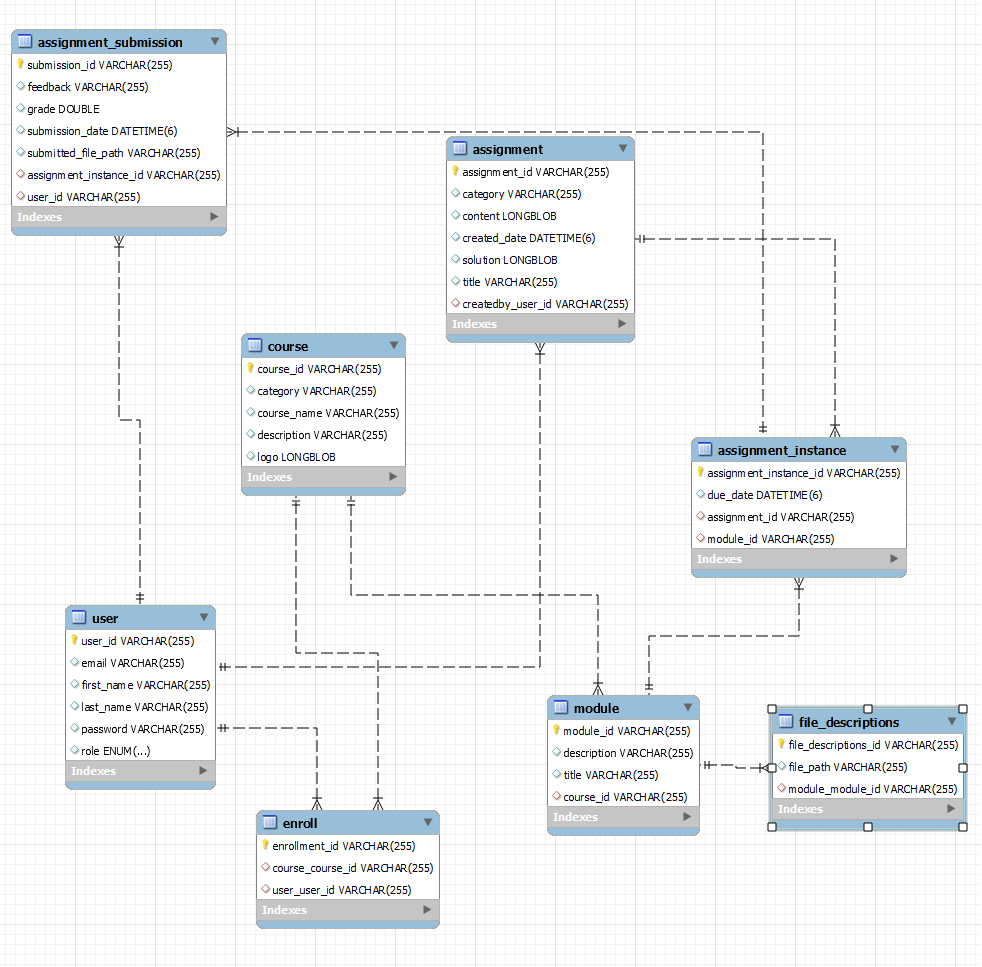
MySQL este un sistem de gestionare a bazelor de date relaționale (RDBMS) open-source. Este unul dintre cele mai populare și utilizate motoare de baze de date din lume, cunoscut pentru performanța sa, fiabilitatea și ușurința în utilizare. MySQL este dezvoltat, distribuit și susținut de Oracle.

## Open-Source

Am mentionat faptul ca alegerile mele in construirea aplicatiei sunt open-source. Astfel voi argumenta care sunt avantajele:

1. Transparenta: dezvoltatorii pot inspecta codul sursa pentru a observa daca acesta este functional, de asemenea acesta este un factor bun si din punct de vedere al securitatii.
2. Securitatea: datorita transparentei dezvoltatorii pot analiza codul de vulnerabilitati ascunse. Legea lui Linus Torvalds „Given enough eyeballs, all bugs are shallow" – aceasta inseamna cu cat mai multa lume lucreaza si testeaza aplicatia problemele aplicatiei dispar.
3. Valoare educationala
4. Suport si comunitate: datorita faptului ca este open-source suportul si comunitatea este pe masura fiind foarte usor sa gasesti informatii. De asemenea toti dezvoltatorii pot aduce contributii.
5. Inovare si dezvoltare rapida: datorita faptului ca toata lumea poate contribuii duce la dezvoltarea rapida si aducerea a noi inovatii in tehnologie.

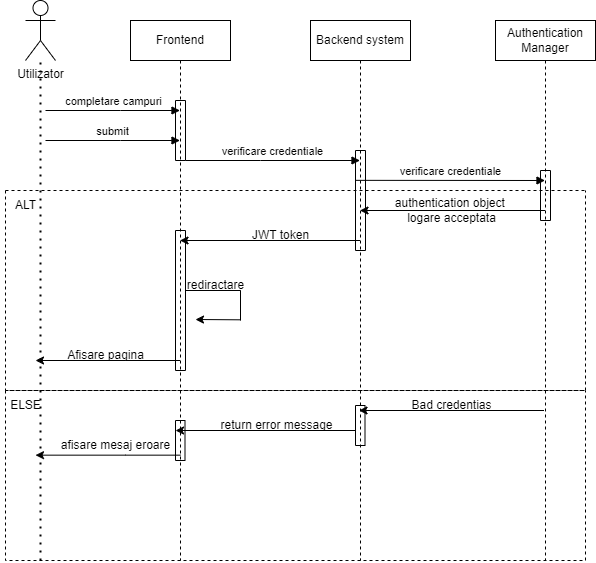
## Arhitectura bazei de date



1. Tabela user: Aceasta contine informatiile despre utilizatori. Coloanele sunt: user\_id cheie primara, email first\_name last\_name, password si role.
2. Tabela course: Aceasta contine informatiile despre cursuri. Coloanele sunt course\_id cheie primara, category, course\_name, description si logo.
3. Tabela module: Aceasta contine informatii despre modulele din curs, fiecare modul fiind legat de un curs. Coloanele sunt module\_id cheie primara, description, title si course\_id cheie straina.
4. Tabela file\_description: Aceasta contine caile pentru fisierele din module. Coloanele sunt: file\_descriptions\_id cheie primara, file\_path, module\_module\_id cheie straina.
5. Tabela assignment: Aceasta contine informatii despre exercitiile care pot fi folosite ca teme. Coloane: assignment\_id cheie primara, category, content, created\_date, solution, title si creadedby\_user\_id cheie straina
6. Tabela assignment\_instance: Aceasta tabela contine informatii despre teme, fiind legata de module. Coloane: assignment\_instance\_id cheie primara, due\_date, assignment\_id cheie straina, module\_id cheie straina
7. Tabela assignment\_submission: Aceasta tabela contine informatii despre temele incarcate ale studentilor. Coloane: submission\_id cheie primara, feedback, grade, submission\_date, submitted\_file\_path, assignment\_instance\_id cheie straina si user\_id cheie straina.
8. Tabela enroll: In aceasta tabela este retinut ce utilizator este inregistrat la un anumit curs. Coloane enrollment\_id cheie primara, course\_course\_id cheie straina si user\_user\_id cheie straina.

## Descrierea fluxului functionalitatilor

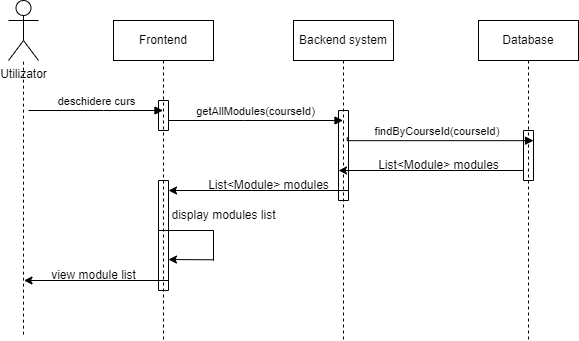
Am ales să folosesc diagrame de secvență pentru a ilustra fluxul funcționalităților. Aceste diagrame sunt vitale pentru înțelegerea modului în care diferitele părți ale unui sistem lucrează împreună pentru a îndeplini anumite funcționalități. Diagramele de secvență oferă o imagine vizuală clară a ordinii mesajelor și a interacțiunilor dintre obiectele sistemului, făcând procesul mai ușor de urmărit și de găsit probleme sau îmbunătățiri.



Interacțiunile dintre utilizator, front-end, back-end și managerul de autentificare sunt ilustrate în diagrama de secvență a fluxului de autentificare din figura X. Utilizatorul completează câmpurile (email și parolă) și apasă pe submit. Din front-end se trimite o cerere de tip POST către back-end. Back-end-ul folosește Authentication Manager pentru a verifica credențialele în baza de date.

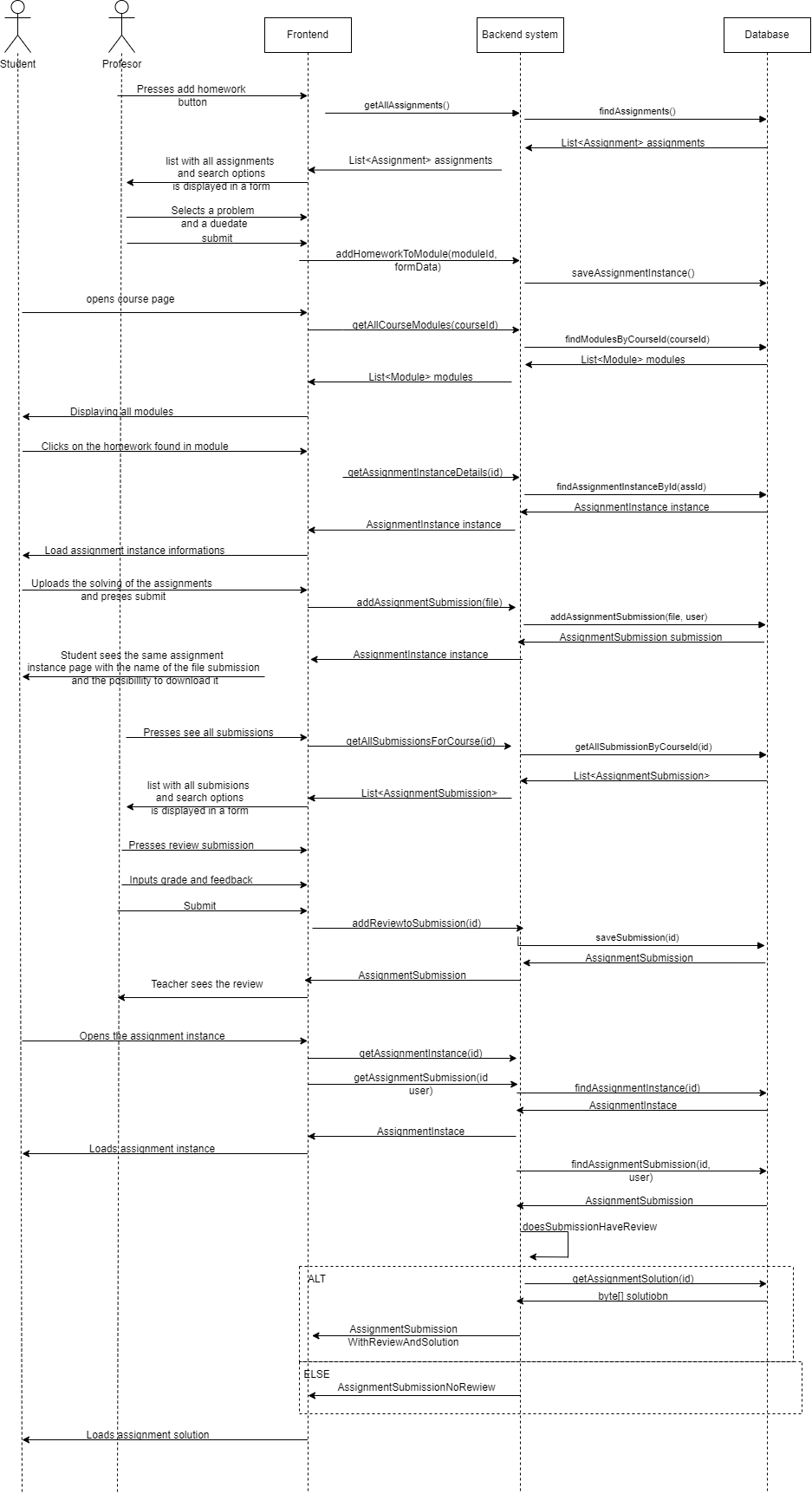
Dacă credențialele sunt corecte, se întoarce un obiect de autentificare. În back-end se generează un token JWT care conține proprietățile utilizatorului, cum ar fi rolul și emailul. Front-end-ul redirecționează utilizatorul pe pagina specifică rolului acestuia: pentru student și profesor pe pagina principală unde sunt afișate cursurile, iar pentru admin pe pagina specială pentru administrator si tokenul este salvat in local storage-ul browserului.

Dacă credențialele sunt incorecte, se întoarce un mesaj HTTP "Bad credentials", iar utilizatorului i se afișează un mesaj de eroare.



Interacțiunile dintre utilizator, front-end, back-end și baza de date sunt ilustrate în diagrama de secvență a fluxului de accesare a unui curs din figura X. Utilizatorul deschide cursul pentru a vizualiza lista de module disponibile. Din front-end se trimite o cerere de tip GET către back-end. Back-end-ul folosește baza de date pentru a găsi modulele asociate cursului specificat.

Dacă cererea este validă, back-end-ul interoghează baza de date pentru a găsi toate modulele asociate cursului cu ID-ul specificat. Baza de date returnează o listă de module către back-end. Back-end-ul trimite lista de module către front-end, care o afișează utilizatorului.



In diagrama din figura x este prezentata interactiunea a diferitelor roluri, student si profesor, cu sistemul. Profesorul este in pagina de curs, acesta apasa pe butonul add homework aflat in unul dintre module. Ii este afisat un formular in care are un modul de cautare si toate problemele existente in lista de exercitii. Acesta alege unul dintre exercitii dupa care selecteaza un termen limita pentru incarcarea temei si apasa pe butonul add homework, pagina de curs este updatata, fiind vizibila noua tema propusa.

Studentul deschide pagina de curs, vede tema propusa de profesor intr-un modul acesta apasa click pe tema. O noua pagina se deschide. Aceasta contine informatiile temei. Elevul termina tema si se decide sa o incarce. Apasa pe butonul upload homework alege fisierul si da submit.

Profesorul apasa pe butonul „See all submisions”, astfel acesta este redirectat pe o pagina care contine toate incarcarile studentilor. Apasa pe butonul de review, se deschide o pagina cu formular de unde acesta poate descarca ceea ce a incarcat studentul. Acesta completeaza recenzie si apasa pe butonul de submit.

Studentul deschide tema, acum este disponibila recenzia profesorului, nota cat si rezolvarea corecta a temei. Daca nu ar fi fost inca adaugata o recenzie acesta avea vizibil decat ultima lui incarcare pe care acesta putea sa o descarce si sa vada ce se afla.

# Detalii de implementare

În plus fata de capitolul precedent acesta conține elemente specifice ale rezolvării problemei care au presupus dificultăți deosebite din punct de vedere tehnic. Pot fi incluse configurații, secvențe de cod, pseudo-cod, implementări ale unor algoritmi, analize ale unor date, scripturi de testare. De asemenea, poate fi detaliat modul în care au fost utilizate tehnologiile introduse in capitolul 3.

Criterii pentru calificativul Nesatisfăcător:

* Sunt prezentate pe scurt scheme și pseudo-cod.

Criterii pentru calificativul Satisfăcător:

* Descriere sumara a implementării, prezentarea unor secvențe nerelevante de cod, scheme, etc.

Criterii pentru calificativul Bine:

* Descrierea detaliată a algoritmilor/structurilor utilizați; Prezentarea etapizată a dezvoltării, inclusiv cu dificultăți de implementare întâmpinate, soluții descoperite; (dacă este cazul) demonstrarea corectitudinii algoritmilor utilizați.

## Indicații formatare tabele

Se recomandă utilizarea tabelelor de forma celui de mai jos. Font: Calibri, 9.

Orice tabel prezent în teză va fi referit în text; exemplu: a se vedea Tabel 1.

Tabel 1 – Sumarizare criterii

|  |  |  |
| --- | --- | --- |
| Calificativ | Criteriu | Observații |
| Nesatisfăcător | Sunt prezentate pe scurt scheme și pseudo-cod |  |
| Satisfăcător | Descriere sumara a implementării, prezentarea unor secvențe nerelevante de cod, scheme, etc. |  |
| Bine | Descrierea detaliată a algoritmilor/structurilor utilizați; Prezentarea etapizată a dezvoltării, inclusiv cu dificultăți de implementare întâmpinate, soluții descoperite; (dacă este cazul) demonstrarea corectitudinii algoritmilor utilizați. | Pot fi incluse configurații, secvente de cod, pseudo-cod, implementări ale unor algoritmi, analize ale unor date, scripturi de testare. |

# Studiu de caz / Evaluarea rezultatelor

Acest capitol trebuie să răspundă, în principiu, la **2 întrebări** și să se încheie cu **o discuție** a rezultatelor obținute. Cele doua întrebări la care trebuie sa se răspundă sunt:

1) **Merge corect**? (Conform specificațiilor extrase în capitolul 2);

Evaluarea dacă merge corect se face pe baza cerințelor identificate în capitolele anterioare.

2) Cât de bine merge / cum se compară cu soluțiile existente? (pe baza unor metrici clare).

Evaluarea cât de bine merge trebuie să fie bazată pe procente, timpi, cantitate, numere, **comparativ cu soluțiile prezentate în capitolul 3**. Poate fi vorba de performanță, overhead, resurse consumate, scalabilitate etc.

În realizarea discuției, se vor utiliza tabele cu procente, rezultate numerice și grafice. În mod obișnuit, aici se fac comparații și teste comparative cu alte proiecte similare (dacă există) și se extrag puncte tari și puncte slabe. Se ține cont de avantajele menționate și se demonstrează viabilitatea abordării / aplicației, de dorit prin comparație cu alte abordări (dacă acest lucru este posibil). Cuvântul cheie la evaluare este „metrică“: trebuie să aveți noțiuni măsurabile și cuantificabile. În cadrul procesului de evaluare, explicați datele, tabelele și graficele pe care le prezentați și insistați pe relevanța lor, în următorul stil: „este de preferat ... deoarece …“; explicați cititorului nu doar datele ci și semnificația lor și cum sunt acestea interpretate. Din această interpretare trebuie să rezulte poziționarea proiectului vostru printre alternativele existente, precum și cum poate fi acesta îmbunătățit în continuare.

Criterii pentru calificativul *Nesatisfăcător*:

* Aplicația este testată dar rulează pe calculatorul studentului, nu există posibilități de testare, nu a fost validată cu clienți / utilizatori;
* Nu au fost realizate comparații cu alte sisteme similare.

Criterii pentru calificativul *Satisfăcător*:

* [Dezvoltare de produs] Există teste unitare și de integrare, există o strategie de punere în funcțiune (*deployment*), există validare minimală cu clienții / utilizatorii.
* [Cercetare] Principalele componente și soluția în ansamblu au fost evaluate din punct de vedere al performanței, însă nu sunt folosite seturi de date standard, există unele erori de interpretare a datelor.
* [Ambele] Discuție minimală asupra relevanței rezultatelor prezentate, comparație minimală cu alte sisteme similare.

Criterii pentru calificativul *Bine*:

* [Dezvoltare de produs] Teste unitare și de integrare, instrumente de punere in funcțiune (*deployment*) utilizate și care arată lucru constant de-a lungul semestrului, lucrare validată cu clienții / utilizatorii, produs în producție.
* [Cercetare] Componentele și soluția în ansamblu au fost evaluate din punct de vedere al performanței, folosind seturi de date standard și cu o interpretare corectă a rezultatelor.
* [Ambele] Discuție cu prezentarea calitativă și cantitativă a rezultatelor, precum și a relevanței acestor rezultate printr-o comparație complexă cu alte sisteme similare.

# Concluzii

În acest capitol este sumarizat întreg proiectul, de la obiective, la implementare, si la relevanta rezultatelor obținute. În finalul capitolului poate exista o subsecțiune de „Dezvoltări ulterioare“.  
Criterii pentru calificativul *Nesatisfăcător*:

* Concluziile nu sunt corelate cu conținutul lucrării;

Criterii pentru calificativul *Satisfăcător*:

* Concluziile sunt corelate cu conținutul lucrării, însă nu se oferă o imagine asupra calității și relevantei rezultatelor obținute;

Criterii pentru calificativul *Bine*:

* Concluziile sunt corelate cu conținutul lucrării, și se oferă o imagine precisa asupra relevantei și calității rezultatelor obținute în cadrul proiectului.

# Bibliografie

* Trebuie respectat **un singur standard** de trimiteri bibliografice (citare), **dintre** următoarele alternative:
  + APA (<http://pitt.libguides.com/c.php?g=12108&p=64730>)
  + IEEE (<https://ieee-dataport.org/sites/default/files/analysis/27/IEEE%20Citation%20Guidelines.pdf>)
  + Harvard (<https://libweb.anglia.ac.uk/referencing/harvard.htm>)
  + Cu numerotarea referințelor în ordine alfabetică sau în ordinea apariției în text (de exemplu, stilul cu numere folosit de unele publicații ACM - <https://www.acm.org/publications/authors/reference-formatting>)
* Toate referințele din acest capitol trebuie să fie referite în text. Exemple:
  + [Articol jurnal]: [2];
  + [Articol conferință]: [3];
  + [Carte]: [4]
  + [Weblink]: [5]
  + [Application report] [6]

|  |  |
| --- | --- |
| [1] | J. Silva-Martinez, "ELEN-325. Introduction to Electronic Circuits: A Design Approach," 2008. [Online]. Available: http://www.ece.tamu.edu/~spalermo/ecen325/Section%20III.pdf. |
| [2] | H. Baali, H. Djelouat, A. Amira and F. Bensaali, "Empowering Technology Enabled Care Using IoT and Smart Devices: A Review," *IEEE Sensors Journal,* vol. 18, no. 5, pp. 1790-1809, 2018. |
| [3] | A. Haroon, S. Akram, M. A. Shah and A. Wahid, "E-Lithe: A Lightweight Secure DTLS for IoT," in *IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Toronto, 2017. |
| [4] | A. K. Jain and R. C. Dubes., Algorithms for Clustering Data, Upper Saddle River: Prentice-Hall, Inc., 1988. |
| [5] | "Kernel panic! What are Meltdown and Spectre, the bugs affecting nearly every computer and device?," techcrunch.com, 2018. [Online]. Available: https://techcrunch.com/2018/01/03/kernel-panic-what-are-meltdown-and-spectre-the-bugs-affecting-nearly-every-computer-and-device. [Accessed 14 02 2018]. |
| [6] | E. Rogers, "Understanding Buck-Boost Power Stages in Switch Mode Power Supplies," Texas Instruments, 2007. |

* NU utilizați referințe la Wikipedia sau alte surse fără autor asumat.
* Pentru referințe la articole relevante accesibile în web (descrise prin URL) se va nota la bibliografie și data accesării.
* Mai multe detalii despre citarea referințelor din internet se pot regăsi la:
  + <http://www.writinghelp-central.com/apa-citation-internet.html>
  + <http://www.webliminal.com/search/search-web13.html>
* Note de subsol se utilizează dacă referiți un link mai puțin semnificativ o singură dată; Dacă nota este citată de mai multe ori, atunci utilizați o referință bibliografică.
* Dacă o imagine este introdusă în text și nu este realizată de către autorul lucrării, trebuie citată sursa ei (ca notă de subsol sau referință - este de preferat utilizarea unei note de subsol).
* Referințele se pun direct legate de text (de exemplu „KVM [1] uses“, „as stated by Popescu and Ionescu [12]”, etc.). Nu este recomandat să folosiți formulări de tipul „[1] uses”, „as stated in [12]“, „as described in [11]“ etc.
* Afirmațiile de forma „are numerous“, „have grown exponentially“, „are among the most used“, „are an important topic“ trebuie să fie acoperite cu citări, date concrete si analize comparative.
  + Mai ales în capitolele de introducere, „state of the art“, „related work“ sau „background“ trebuie să vă argumentați afirmațiile prin citări. Fiți autocritici și gândiți-vă dacă afirmațiile au nevoie de citări, chiar și cele pe care le considerați evidente.
  + Cea mai mare parte dintre citări vor fi în capitolele de introducere „state of the art“, „related work“ sau „background“.
* Toate intrările bibliografice trebuie citate în text. Nu le adăugați pur și simplu la final.
* Nu copiați sau traduceți niciodată din surse de informație de orice tip (online, offline, cărți, etc.). Dacă totuși doriți să oferiți, prin excepție, un citat celebru - de maxim 1 frază- utilizați ghilimele și evident menționați sursa.
* Dacă reformulați idei sau creați un paragraf rezumat al unor idei folosind cuvintele voastre, precizați cu citare (referință bibliografică) sau cu notă de subsol sursa sau sursele de unde ați preluat ideile.

# Anexe

Anexele sunt opționale.

Ce poate intra în anexe:

* Exemplu de fișier de configurare sau compilare;
* Un tabel mai mare de ½ pagină;
* O figura mai mare mai mare de ½ pagină;
* O secvență de cod sursa mai mare de ½ pagină;
* Un set de capturi de ecran („screenshot”-uri);
* Un exemplu de rulare a unor comenzi plus rezultatul („output”-ul) acestora;
* În anexe intră lucruri care ocupă mai mult de o pagină ce ar întrerupe firul natural de parcurgere al textului.

1. https://www.edupedu.ro/piata-meditatiilor-estimata-la-peste-120-de-milioane-de-lei-pe-luna-pe-baza-datelor-publice-si-a-cercetarilor-recente-analiza/ [↑](#footnote-ref-1)
2. https://www.rainews.it/articoli/2024/04/scuola-1-studente-su-5-va-a-ripetizioni-cosi-le-famiglie-spendono-circa-450-euro-7a461864-a445-4a71-849c-ff05fbf738b6.html#:~:text=Il%20trend%20%C3%A8%20confortante%2C%20sia,si%20%C3%A8%20passati%20al%2024%25 [↑](#footnote-ref-2)
3. https://www.sciencedirect.com/science/article/pii/S0959475219303093 [↑](#footnote-ref-3)
4. © http://www.ece.tamu.edu/~spalermo/ecen325/Section%20III.pdf [↑](#footnote-ref-4)