

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE  
DEPARTAMENTUL CALCULATOARE

## PROIECT

*la disciplina Introducerea în baza de date*

Aplicație de gestionare a unei platforme de  
studiu în cadrul unei facultăți

Profesor îndrumător: Ivan Cosmina  
Studenti: Bustan-Jeflea Ștefania,  
Chiorean Bogdan  
Grupa 30223  
An academic: 2022-2023



## Cuprins

I. Introducere.....	3
II. Analiza cerintelor utilizatorilor (specificatiile de proiect) .....	3
2.1. Ipoteze specifice domeniului ales pentru proiect (cerințe, constrângeri) .....	3
2.2. Organizare structurata (tabelar) a cerințelor utilizator .....	3
2.3. Determinarea si caracterizarea de profiluri de utilizatori .....	3
III. Modelul de date si descrierea acestuia.....	4
3.1. Entități si attributele lor.....	4
3.2. Normalizarea datelor.....	5
3.2.1. Prima formă de normalizare(1NF).....	6
3.2.2. A doua formă de normalizare(2NF).....	6
3.2.3. A treia formă de normalizare(3NF) .....	6
3.2.4. Forma normala Boyce-Codd(BCNF).....	6
3.3. Relații.....	6
IV. Detalii de implementare .....	7
4.1. Descrierea funcțională a modulelor.....	7
4.1.1.1. MySQL Workbench .....	7
4.1.1.2. Eclipse .....	7
4.1.1.3. IntelliJ.....	8
4.1.2.1. SQL.....	8
4.1.2.2. Java .....	9
V. Bibliografie .....	10



## **I.Introducere**

Proiectul este o aplicatie de gestionare a unei platforme de studiu in cadrul unei facultati. Aceasta aplicatie foloseste un sistem de gestiune pentru baze de date MySQL, iar interactiunea cu ea se realizeaza doar prin interfata grafica. Functionalitatile pe care le va oferi programul vizeaza operatii ce tin de gestiunea studentilor, profesorilor si administrarea operatiilor curente din cadrul unor programe de studiu.

Accesul la aplicatie se face pe baza unor conturi de autoidentificare care contin o adresa de mail si o parola. Exista 4 tipuri de utilizatori: *student*, *profesor*, *administrator*, *super-administrator*. Fiecare are drepturi diferite asupra manipularii bazei de date.

## **II. Analiza cerintelor utilizatorilor (specificatiile de proiect)**

### **2.1. Ipoteze specifice domeniului ales pentru proiect (cerințe, constrângeri)**

Aplicatia va permite gestiunea cu usurinta a activitatilor didactice si astfel a interațiunilor dintre studenti si profesori. Cursurile vor putea fi predate de mai multi profesori și au una sau mai multe tipuri de activitati (ex: curs, seminar, laborator).

Studentii se vor putea inscrie la cursuri si sunt asignati profesorului cu cei mai putini studenti la data inscrierii, iar acestia vor primi note pentru fiecare tip de activitate de la cursul respective. Profesorul stabileste din interfata grafica impartirea procentuala pe tipurile de activitati (ex. 20% seminar, 35% laborator, 45% curs/examenul de la curs). Ulterior, profesorul poate programa activitatile (curs, seminar, laborator, colocviu, examen) intr-un calendar, pe zile si ore, specificand si numarul maxim de participanti. Activitatile pot fi programate doar in viitor.



Totodata, studentii au posibilitatea sa se inscrie in grupuri de studiu pentru o anumita materie, daca sunt inscrisi la materia respectiva. Acolo ei pot sa-si adauge colegi de la cursul cursul pentru care a fost creat grupul respective. La fel ca si la cursuri, studentii pot sa-si creeze activitati de aprofundare a cursului. Ei pot participa la o activitatea de grup doar daca nu se suprapune cu o activitatea de curs la care e planificat sa mearga.

### **2.2. Organizare structurata (tabelar) a cerințelor utilizator**



Initial am creat tabele pentru fiecare tip de utilizator prezentat în ipoteza: Admin, Profesor, Student. Acestea mostenesc o entitate de tip utilizator unde sunt stocate atributele comune. Acestea au fost urmate de entitatile create pentru curs, grupuri și activități. Următorul pas a fost de stabilire a relațiilor dintre aceste tabele și despicarea relațiilor de tip many-to-many prin crearea unor tabele de legătură. La final am creat o tabela pentru mesaje cu scopul de a realiza chat-ul grupurilor de studenți.

### **2.3.Determinarea si caracterizarea de profiluri de utilizatori**

#### **1. Super-Administratori:**

-  pot gestiona conturile tuturor utilizatorilor (inclusiv administratorilor)
-  pot gestiona cursurile si pot asigna profesori la acestea

#### **2. Administrator**

-  pot gestiona conturile profesorilor si al studentilor
-  pot gestiona cursurile si pot asigna profesori la acestea



3. Profesor

- + pot sa isi vizualizeze datele personale
- + pot sa modifice procentele notei finale de la cursurile pe care le sustine
- + pot nota studentii
- + pot sa creeze activitati pentru cursurile la care predau
- + pot sa vizualizeze activitatile saptamanle

4. Student

- + pot sa isi vizualizeze datele personale
- + pot sa se inscrie la cursuri
- + pot sa isi vizualizeze notele si sa isi descarce catalogul
- + pot sa isi vizualizeze activitatile saptamanale
- + pot sa creeze activitati de grup
- + pot sa se inscrie la activita de curs/grup
- + pot sa se inscrie in grupuri, si sa adauge alti membri
- + pot sa vorbeasca in grupuri

### III. Modelul de date si descrierea acestuia

#### 3.1 Entități si atributele lor

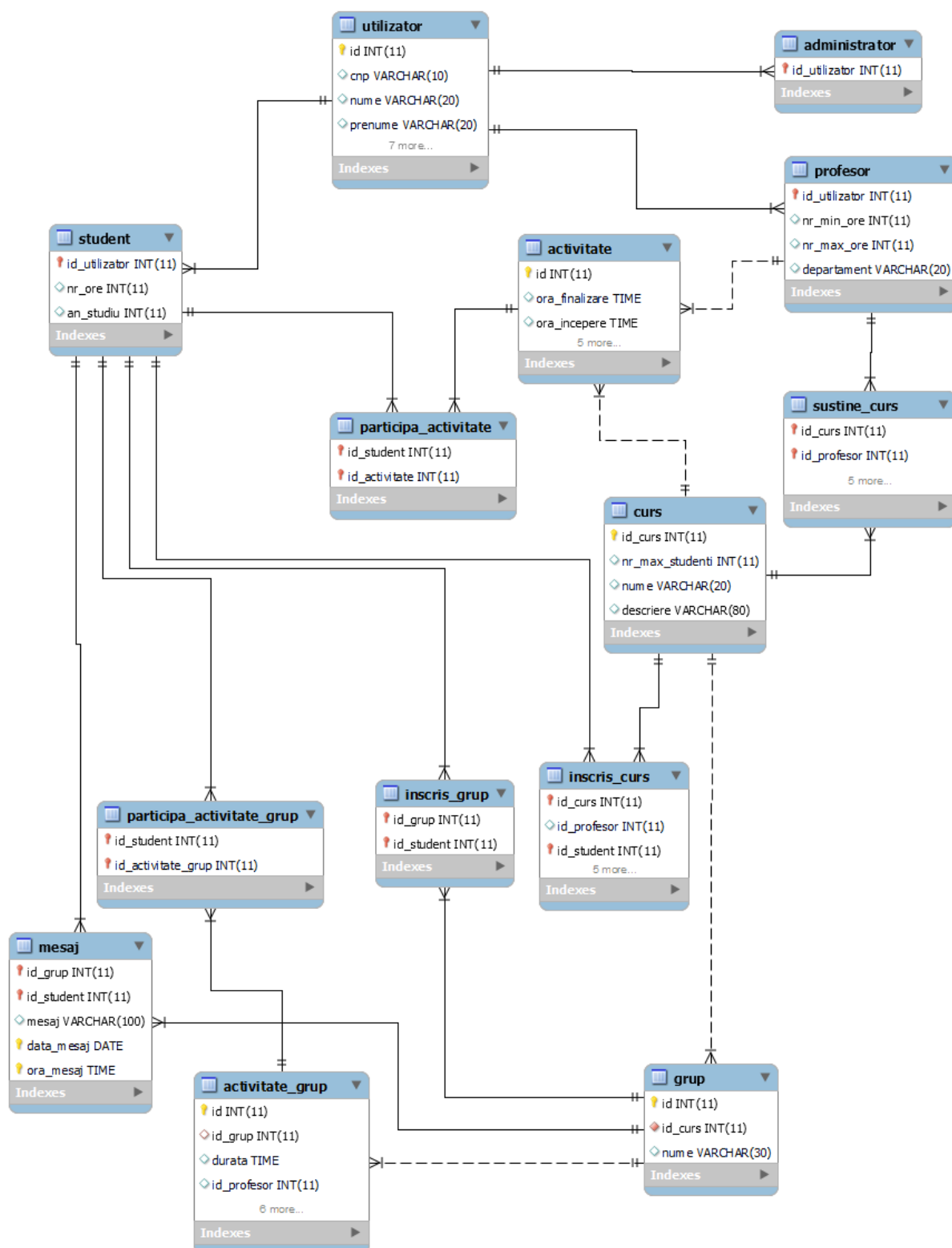
Baza de date creată conține 14 tabele cu atribute specifice, după cum urmează:

- *utilizator*: **id**, **cnp**, nume, prenume, adresa, telefon, **mail**, parola, cont iban, nr. contract, functie (0-admin / 1-profesor / 2-student)
- *student*: **id utilizator**, nr. de ore, anul de studiu
- *profesor*: **id utilizator**, nr. minim de ore, nr. maxim de ore, departament
- *administrator*: **id utilizator**
- *curs*: **id curs**, nr. maxim studenti, **nume**, descriere
- *sustine\_curs*: **id curs**, **id profesor**, procentele notei finale la curs
- *inscris\_curs*: **id curs**, **id student**, id\_profesor notele de la cursul respectiv
- *activitate*: **id**, orele intre care se tine, **id curs**, **id profesor**, nr. maxim de studenti, data, tip (0-curs/ 1-seminar/ 2-laborator/ 3-colocviu/ 4-examen)
- *participa\_activitate*: **id student**, **id activitate**,
- *grup*: **id**, **id curs**, nume
- *inscris\_grup*: **id grup**, **id student**
- *mesaj*: **id grup**, **id student**, mesajul, **data mesaj**, **ora mesaj**
- *activitate\_grup*: **id**, **id grup**, durata, id profesor, data si ora creari, data si ora inceperii activitatii, durata de expirare, nr. minim de participanti
- *participa\_activitate\_grup*: **id student**, **id activitate grup**



### 3.2. Normalizarea datelor

Se va prezenta pe scurt evoluția bazei de date, precum și felul în care aceasta a evoluat pentru a îndeplini cerințele impuse de formele normale.





### 3.2.1. Prima formă de normalizare(1NF)

Prima formă normală este o formă normală utilizată în normalizarea bazelor de date. Prima formă normală exclude posibilitatea existenței grupurilor repetitive cerând ca fiecare câmp într-o bază de date să cuprindă numai o valoare atomică. De asemenea, prima formă normală cere și ca fiecare înregistrare să fie definită astfel încât să fie identificată în mod unic prin intermediul unei chei primare. În modelul bazei de date nu exista valori de tip tablou care să creeze redundanța.

### 3.2.2 A doua formă de normalizare(2NF)

O relație este în 2NF dacă este în 1NF. A doua formă normală cere ca toate elementele unei tabele să fie dependente funcțional de totalitatea cheii primare. Dacă unul sau mai multe elemente sunt dependente funcțional numai de o parte a cheii primare, atunci ele trebuie să fie separate în tabele diferite. În tabelele care au cheie compusă, fiecare atribut să depindă de toate coloanele care compun cheia primară. Dacă un tabel are cheie unică el intră automat în a doua formă de normalizare. În relațiile dintre table sunt utilizate attribute de tip id, astfel încât să nu se salveze inefficient date în mai multe locuri.

### 3.2.3. A treia formă de normalizare(3NF)

O relație este în 3NF dacă este în 2NF și nu există dependențe funcționale tranzitive față de cheia primară(nu există dependențe funcționale între attributele non-chei). În baza de date attributele sunt dependente de cheile primare din tabelele în care se afla.

### 3.2.4 Forma normală Boyce-Codd(BCNF)

Relațiile din baza de date trebuie proiectate astfel încât să nu aibă nici dependențe parțiale, nici dependențe tranzitive, deoarece acestea duc la apariția anomaliilor de reactualizare. Formele 2NF și 3NF elimină dependențele parțiale și tranzitive de cheia primară, dar nu tratează situațiile în care rămân astfel de dependențe față de cheile candidat ale unei relații. Forma normală Boyce- se bazează pe dependențele funcționale care iau în considerație toate cheile candidat dintr-o relație. Pentru o relație cu o singură cheie candidat, formele 3NF și BCNF sunt echivalente. . Acest lucru se întâmplă și în cadrul bazei de date realizate. Aici avem doar chei candidate (supercheie ireductibilă, minimală) unice. La relațiile many to many dintre table folosim doar id-uri pentru identificarea tuplelor astfel încât nu se rețin informații extra care ar putea forma redundanța cu alte date.

## 3.3. Relații

Relațiile într-o bază de date relațională pot fi:

- 1:1 (*one to one*): fiecărei linii în primul tabel îi corespunde cel mult o singură linie în al doilea (o singură persoană are o singură adresă sau un singur card)

În cadrul bazei de date realizate avem relație 1:1 între tabelele *student* și *utilizator*. În cadrul tablei *utilizator* vom reține datele personale pentru fiecare tip de utilizator. Iar în tabela *student* se vor reține date în plus pe care le are un utilizator de tip student față de



utilizatorii de celalte tipuri. Astfel, fiecărui student îi corespunde o singură linie din tablea *utilizator*. La fel si *student* si *utilizator*, *administrator* si *utilizator*.

- 1:m (*one to many*): fiecărei linii în primul tabel îi pot corespunde mai multe în al doilea și fiecărei linii al doilea îi corespunde exact una în primul

În cadrul bazei de date realizate avem relație „one to many” între tabelele *student* și *mesaje*. Prin urmare, un student poate să scrie mai multe mesaje într-un grup, dar unui mesaj îi corespunde un singur student.

- m:m (*many to many*): unei linii în primul tabel îi pot corespunde mai multe în al doilea și unei linii în al doilea tabel îi pot corespunde mai multe în primul

În cadrul bazei de date realizate avem relație *many to many* între tabelele *profesor* și *curs*, intrucat un profesor poate să sustina mai multe cursuri și un curs poate să fie sustinut de mai mulți profesori. Aceasta relație a fost redusă la două relații „one to many” prin introducerea tablei *sustine curs*.

## IV. Detalii de implementare

### 4.1. Descierea funcțională a modulelor

Un site dinamic presupune o îmbinare eficienta a diferitelor limbaje de programare. Pentru realizarea aplicatiei s-a folosit MySQL și Java (JavaFX). Pentru început am scris cod SQL în MySQL Workbench, pentru crearea tabelor, popularea acestora, crearea procedurilor, după care am început sa scriem codul pentru interfata si apelarea procedurilor in Eclipse/IntelliJ.

#### 4.1.1. Tooluri

##### 4.1.1.1. MySQL Workbench

MySQL Workbench este un instrument grafic pentru a lucra cu serverele și bazele de date MySQL. MySQL Workbench este prevăzut să lucreze cu versiunile de MySQL Server 5.1 și mai sus. MySQL Workbench tinde să fie un instrument ce acoperă cele mai importante activități de gestionare a bazelor de date:

- SQL Development: permite să gestionezi conexiunile la serverele MySQL, la fel oferă posibilitatea de a executa interogări SQL.
- Data Modeling: permite să creezi modele de baze de date în mod grafic, cât și editarea a bazelor de date deja existente. Table Editor este menit pentru editare de tabele, coloane, indici, triggere, partiționare, opțiuni, inserturi și privilegii, rutine și view-uri.
- Server Administration: permite să crezi și administrezi instanțele de server.
- Data Migration: permite să migrezi pe MySQL datele din Microsoft SQL Server, Sybase ASE, SQLite, SQL Anywhere, PostgreSQL.

##### 4.1.1.2 Eclipse

Eclipse este un mediu de dezvoltare open-source scris preponderent în Java. Acesta poate fi folosit pentru a dezvolta aplicații Java și, prin intermediul unor plug-in-uri, în



alte limbaje, cum ar fi C, C++, COBOL, Python, Perl și PHP. Interfața grafică în Eclipse este scris folosind setul de instrumente SWT. Acesta din urmă, spre deosebire de Swing (care emite independent controale grafice), utilizează componentele grafice ale sistemului de operare dat. Interfața de utilizator Eclipse depinde, de asemenea, de un strat intermediar GUI numit JFace, care simplifică construirea unei interfețe utilizator bazate pe SWT.

#### 4.1.1.3 IntelliJ

IDEA IntelliJ este un mediu de dezvoltare integrat (IDE) scris în Java pentru dezvoltarea de software de calculator. Este dezvoltat de JetBrains (cunoscut anterior sub numele de IntelliJ) și este disponibil sub formă de Apache 2 licențiat ediție comunitară, și într-o proprietate ediție comercială. Ambele pot fi utilizate pentru dezvoltarea comercială. Prima versiune a IntelliJ IDEA a fost lansată în ianuarie 2001 și a fost una dintre primele IDE Java disponibile cu navigare avansată în cod și refactorizarea codului capabilități integrate. În decembrie 2014, Google a anunțat versiunea 1.0 a Android Studio, un sursă deschisă IDE pentru Android aplicații, bazate pe ediția comunității open source a IntelliJ IDEA. Alte medii de dezvoltare bazate pe cadrul IntelliJ includ AppCode, CLion, DataGrip, GoLand, PhpStorm, PyCharm, Călăreț, RubyMine, WebStorm, și MPS.

### 4.1.2. Limbaje de programare

#### 4.1.2.1. SQL

SQL (Structured Query Language) este un limbaj de programare specific pentru manipularea datelor în sistemele de manipulare a bazelor de date relaționale (RDBMS), iar la origine este un limbaj bazat pe algebra relațională. Acesta are ca scop inserarea datelor, interogații, actualizare și ștergere, modificarea și crearea schemelor, precum și controlul accesului la date. A devenit un standard în domeniu (standardizat ANSI-ISO), fiind cel mai popular limbaj utilizat pentru crearea, modificarea, regăsirea și manipularea datelor de către SGBDurile relaționale. SQL permite atât accesul la conținutul bazelor de date, cât și la structura acestora.

➤ Exemplu de creare tabel în SQL pentru memorarea utilizatorilor:

```
-- 0 admin / 1 profesor / 2 student
Drop table if exists utilizator;
create table if not exists utilizator(
id int primary key auto_increment not null,
cnp varchar(10) unique,
nume varchar(20),
prenume varchar(20),
adresa varchar(70),
telefon varchar(12),
mail varchar(30) not null unique,
parola varchar(30) not null,
cont_iban varchar(50),
nr_contract int not null,
functie int
);
```





➤ Exemplu de procedura pentru adaugarea unui student la o activitate

```
DROP procedure IF EXISTS participare_activitate;  
DELIMITER //  
CREATE procedure participare_activitate(id_stud int, id_act int)  
begin  
    declare nr_stud int;  
    declare nr_max int;  
    select count(*) into nr_stud from participa_activitate where id_activitate = id_act;  
    select nr_max_studenti into nr_max from activitate where id = id_act;  
    if(nr_stud < nr_max)  
    then  
        insert into participa_activitate(`id_student`,`id_activitate`)  
        values(id_stud,id_act);  
    else  
        select 'activitate plina';  
    end if;  
end;  
//
```

Pentru inceput se numara toti studenti deja inscrisi la aceasta activitate pentru a verifica daca mai sunt locuri disponibile, dupa care se insereaza in tabela respectiva o noua tupla cu id-ul studentului si id-ul activitatii respective.

➤ Procedura pentru aduagarea unui nou utilizator

```
begin  
    Insert into utilizator(`cnp`,`nume`,`prenume`,`adresa`,`telefon`,`mail`,`parola`,`cont_iban`,`nr_contract`,`functie`)  
    values (cnp,nume,prenume,adresa,telefon,mail,parola,cont_iban,nr_contract,functie);  
    select @id:=id from utilizator where cnp = cnp;  
    if(functie = 0)  
    then insert into administrator(`id_utilizator`) values (@id);  
    else  
        if(functie = 1)  
        then  
            insert into profesor(`id_utilizator`,`nr_min_ore`,`nr_max_ore`,`departament`) values (@id,nr_min,nr_max,depart);  
        else  
            insert into student(`id_utilizator`,`nr_ore`,`an_studiu`) values (@id,nr_min,nr_max);  
        end if;  
    end if;  
end ;
```

Parametrii pe care ii va primi aceasta procedura sunt aceeasi, insa in aceasta procedura vor avea loc insemnari si in una dintre tabelele *administrator*, *profesor*, *student* in functie de valoarea parametrului functie. (0- administrator, 1- profesor, 2- student).

#### 4.1.2.2. Java

Java este un limbaj de programare pe obiecte ce este destinat sa ruleze pe orice platforma. Este un limbaj complex bazat pe clase care ofera pachete pentru o multitudine de aplicatii. În cadrul proiectului am utilizat pachetele: JDBC și JavaFX.

Pachetul JDBC este destinat manipulării unei baze de date în cadrul limbajului Java. Cu ajutorul acestuia am accesat baza de date și am gestionat informațiile acesteia. Pachetul cuprinde mai multe clase destinate pentru fiecare tip de acțiune precum: Connection, Statement, ResultSet etc..



- Un exemplu de cod care apelează din Java funcția de logare din MySQL

```
CallableStatement call = connection.prepareCall("{? =call logare(?,?)}");  
call.registerOutParameter(1, Types.INTEGER);  
call.setString("mail", m);  
call.setString("parolaa", pas);  
call.execute();  
id = call.getInt(1);
```

Pachetul JavaFx este una dintre modalitățile în care se pot crea interfețe de GUI în Java. Acesta oferă o schemă prin care se pot organiza scene care conțin elementele de interacțiune cu utilizatorul precum: butoane, imagini, liste, texte.

- Exemplu de funcții și obiecte din JavaFX

```
primaryStage.setTitle("UTCN");  
scene = Login.getStartScene(primaryStage, connection);  
// scene = Admin.getAdminScene(false);  
// scene = Profesor.getProfScene();  
primaryStage.setScene(scene);  
primaryStage.show();
```

## V. Bibliografie

- ✓ [https://wikicro.icu/wiki/IntelliJ\\_IDEA](https://wikicro.icu/wiki/IntelliJ_IDEA)
- ✓ [https://ro.wikipedia.org/wiki/Java\\_\(limbaj\\_de\\_programare\)](https://ro.wikipedia.org/wiki/Java_(limbaj_de_programare))
- ✓ <https://ro.wikipedia.org/wiki/MySQL>
- ✓ [https://ro.wikipedia.org/wiki/Eclipse\\_\(software\)](https://ro.wikipedia.org/wiki/Eclipse_(software))
- ✓ [https://en.wikipedia.org/wiki/MySQL\\_Workbench](https://en.wikipedia.org/wiki/MySQL_Workbench)