UNIVERSITATEA POLITEHNICA BUCUREȘTI

FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE

DEPARTAMENTUL CALCULATOARE

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

Online Gradebook

Flavia-Alexandra Popescu

343C3

BUCUREŞTI

2020

Cuprins

[Cuprins 3](#_Toc62318035)

[Cuprins figuri 3](#_Toc62318036)

[1 Introducere 4](#_Toc62318037)

[1.1 Tema propusă 4](#_Toc62318038)

[1.2 Tehnologii folosite 4](#_Toc62318039)

[2 Descrierea bazei de date 4](#_Toc62318040)

[2.1 Diagrama bazei de date 4](#_Toc62318041)

[2.2 Structura tabelelor 5](#_Toc62318042)

[2.3 Descrierea constrângerilor de integritate 6](#_Toc62318043)

[Descrierea procedurilor și funcțiilor 6](#_Toc62318044)

[2.3.1 Funcții 6](#_Toc62318045)

[2.3.2 Proceduri 6](#_Toc62318046)

[3 Descrierea aplicației 6](#_Toc62318047)

[3.1 Structura claselor 6](#_Toc62318048)

[3.2 Diagrama de stări și fluxul de lucru 6](#_Toc62318049)

[3.3 Conexiunea cu baza de date 6](#_Toc62318050)

[4 Interfețe și rapoarte 6](#_Toc62318051)

[5 Concluzii 6](#_Toc62318052)

[6 Bibliografie 6](#_Toc62318053)

Cuprins figuri

[Figura 2‑1ERD-ul bazei de date 3](#_Toc62407174)

[Figura 2‑2 Relațiile între user\_account și celelalte tabele 5](#_Toc62407175)

[Figura 2‑3 Alocarea părinților și a elevilor 6](#_Toc62407176)

[Figura 2‑4 Relațiile între clase și restul tabelelor 6](#_Toc62407177)

[Figura 2‑5 Alocarea profesorilor pe clase 7](#_Toc62407178)

[Figura 2‑6 Alocarea profesorilor pe materii 7](#_Toc62407179)

[Figura 2‑7 Relația între examinare și restul entităților 8](#_Toc62407180)

[Figura 2‑8 Funcțiile folosite 8](#_Toc62407181)

[Figura 2‑9 Procedurile folosite 8](#_Toc62407182)

[Figura 2‑10 Trigger pentru tabela examination 9](#_Toc62407183)

[Figura 2‑11 Trigger pentru tabela teacher 9](#_Toc62407184)

[Figura 4‑1 Pagina de login 11](#_Toc62407185)

[Figura 4‑2 Dashboard-ul profesorului 11](#_Toc62407186)

[Figura 4‑3 Detaliile despre o clasa 11](#_Toc62407187)

[Figura 4‑4 Vizualizare si adaugare note 12](#_Toc62407188)

[Figura 4‑5 Actualizare medie în timp real 12](#_Toc62407189)

[Figura 4‑6 Vizualizare profil și Logout 12](#_Toc62407190)

# Introducere

## Tema propusă

Tema propusă pentru acest proiect constă în bazele unui catalog electornic sub forma unei aplicații web. În implementarea realizată, un profesor poate intra într-un cont creat de administrator, poate vizualiza clasele unde predă (împreună cu detalii despre aceasta precum profilul și diregintele), elevii și notele acestora. De asemenea, poate adăuga note și vizualiza media actualizată după fiecare evaluare și își poate vizualiza profiul de profesor.

## Tehnologii folosite

Pentru baza de date, am ales să folosesc un docker de MySQL împreună cu MySQL Workbench. Motivul pentru care am ales aceasta opțiune este compatibilitatea tematicii proiectului cu modelul relațional vizibil mai ales în relațiile între profesori, clase și elevi.

În implementarea backend-ului am ales sa folosesc python, mai exact, biblioteca flask, ca urmare a constrangerilor de a nu folosi ORM-uri și a facilităților specifice limbajului, potrivite mai ales pentru prototipare.

Nu în ultimul rând, pentru frontend, am folosit HTML, CSS, Bootstrap și jQuery datorită unui oarecare nivel de familiaritate cu aceste limbaje și framework-uri.

# Descrierea bazei de date

## Diagrama bazei de date

În modelarea problemei, am cosniderat cea mai naturală reprezentarea de mai jos.

[Graphical user interface, application, Word

Description automatically generated](https://dbdiagram.io/d/5ffeee5280d742080a362a3f)

Figura 2‑1ERD-ul bazei de date

În cadrul acesteia am folosit 19 entități pentru a reprezenta tipurile de utilizator (administratori, profesori, elevi și parinți) și relațiile dintre aceștia (cu precădere între părinți și elevi).

De asemenea, am modelat materiile predate de profesori, specializarile claselor, examinările, notele, mediile și alocările profesorilor și elevilor pe clase.

## Structura tabelelor

După cum se vede în diagrama prezentată anterior, am folosit următoarele tabele:

1. **user\_account**
   * pentru un cont de utilizator
   * are un id și descrie numele de utilizator, parola, tipul utilizatorului
2. **user\_types**
   * tabel de tip dicționar <id\_tip, tip\_utilizator>
   * descrie tipurile de utilizator din sistem, anume: administrator, profesor, elev și părinte
3. **admin**
   * tabel care descrie utilizatorul cu rol de administrator
   * acesta are un user\_id, cnp, nume, prenume și un id indicând către tipul de adminsitrator (exemplu: director, secretar, inspector, etc.)
4. **admin\_type**
   * tabel de tip dicționar <id\_tip\_admin, tip\_admin>
5. **student**
   * tabel reprezentând elevul
   * acesta este identificat printr-un user\_id, un număr matricol, id-ul clasei din care face parte, nume, prenume, adresa de email și adresa domiciliului
6. **parent**
   * tabela cu parintii elevilor, caracterizată prin: user\_id, cnp, prenume, nume, email, telefon
7. **parent\_allocation**
   * tabela care face alocarea părinților și a elevilor, contine intrari cu id-ul părinților si id-ul (sau id-urile) copiilor aceștiora
8. **class**
   * identifică o clasă prin id, an, nume id-ul dirigintelui și id-ul specializării
9. **specialization**
   * tabel de tip dicționar <id\_tip\_specializare, tip\_specializare>
10. **teacher**
    * profesorul, reprezentat prin id, cnp, nume, prenume, email, telefon și o valorea booleana reprezentând cazul în care acesta este diriginte la o clasă
11. **subject**
    * materiile, reprezentate ca un dicționar <id\_materie, tip\_materie>
12. **teacher\_allocation**
    * alocarea profesorilor pe materii
    * conține intrari cu id-urile profesorilor și materiilor predate de aceștia
13. **class\_allocation**
    * alocarea profesorilor pe clase
    * intrări cu id-urile profesorilor, materiei și clasa alocată
14. **examination**
    * tabelă reprezentând activitatea de evaluare
    * are un id și conține id-ul profesorului, elevului și materiei la care se face evaluarea
    * conține și data examinării și id-ul corespunzător tipului de examinare
15. **examination\_type**
    * tabel de tip dicționar <id\_tip\_examinare, tip\_examinare>
16. **grade**
    * tabel asociat celui de examinare, conține id-ul examinării, nota și feedback pentru elev, respectiv părintele acestuia
17. **mean\_grade**
    * tabel cu mediile eleviilor
    * identificate printr-un id și conținând id-ul profesorului, elevului, materiei, nota și semestrul
18. **log**
    * tabel unde se pun log-uri cu evenimente și actualizări ale bazei de date
19. **log\_type**
    * tabel de tip dicționar <id\_tip\_log, tip\_log>

## Descrierea constrângerilor de integritate

Majoritatea tabelelor sunt identificate printr-o cheie primară. În această categorie intră: user\_account, examination, mean\_grade, student, class, specialization, log.

Identificate cu ajutorul unei chei primare sunt și tabelele de tip dicționar, precum: user\_types, admin\_type, examination\_type, log\_type și specialization.

Toate tabelele care au câmpuri drept chei străine pentru cele de mai sus, precum student, teacher, parent (pentru user\_account) sau examination (pentru grade și examination\_type) au constrangerea de unicitate. Există totodată și aceste tabele de alocări, care au rolul de a marca relațiile între entități precum: alocarea profesorilor pe materii și clase (teacher\_allocation și class\_allocation) sau examinarea (care leagă notele de profesori, materii și elevi).

În ceea ce privește relațiile între tabele, acestea sunt evidențiate mai pe larg în cele ce urmează.

Tabelele admin, parent, student și teacher reprezintă toate categorii de utilizatori, de aceea au un câmp numit user\_id, care reprezintă o constrângere de tip foreign key catre tabela user\_account, unde user\_id este cheie primară. În ceea ce privește tabela log, aceasta este corelată cu tabela user\_account pentru a indica utilizatorul care a făcut acțiunea ce este logată.



Figura 2‑2 Relațiile între user\_account și celelalte tabele

Tabela parent\_allocation, are rolul de a rezolva relația many-to-many între entitățile părinte-elev (un elev poate avea unul sau mai mulți părinți, iar un părinte poate avea unul sau mai mulți copii înscriși la aceeași școală).



Figura 2‑3 Alocarea părinților și a elevilor

Tabela class, este referită de multiple tabele, ea fiind una din componentele centrale ale modelării problemei. Astfel, refera tabela teacher indicând dirigintele, este referită de elev și specializare și este prezentă și în alocarea profesorilor pe materii și clase.

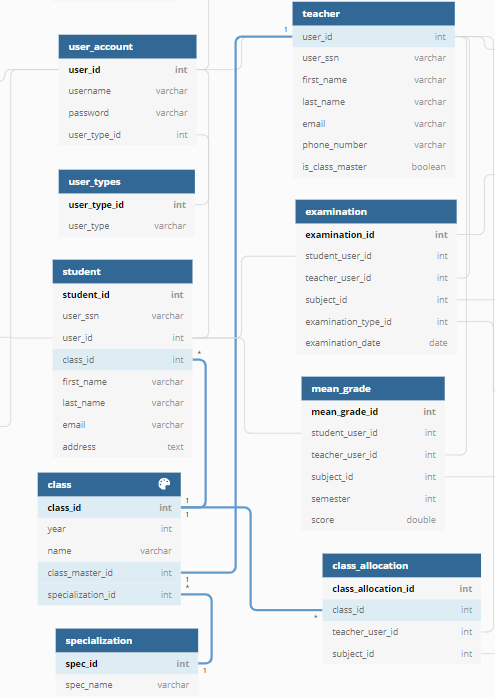


Figura 2‑4 Relațiile între clase și restul tabelelor

Alocarea profesorilor pe clase se face prin intermediul tabelei class\_allocation. Motivul pentru care aceasta este necesară pe lângă alocarea profesorilor pe materii, este acela că un profesor poate preda mai multe materii la o școală. Aceasta rezolvă și relația many-to-many între profesor și clase.

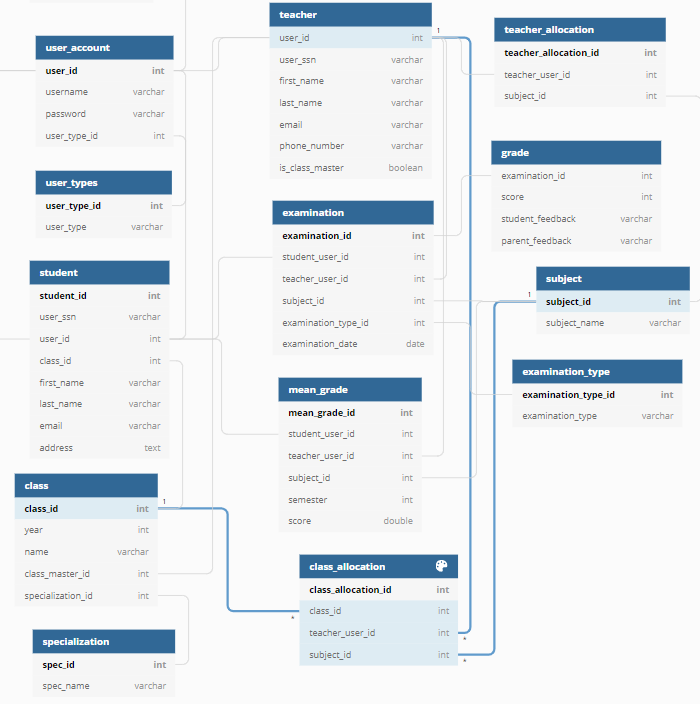


Figura 2‑5 Alocarea profesorilor pe clase

Tabela subject\_allocation are rolul de a rezolva relația many-to-many între profesori și materii (o materie poate fi predată de mai mulți profesori și un profesor poate preda mai multe materii).

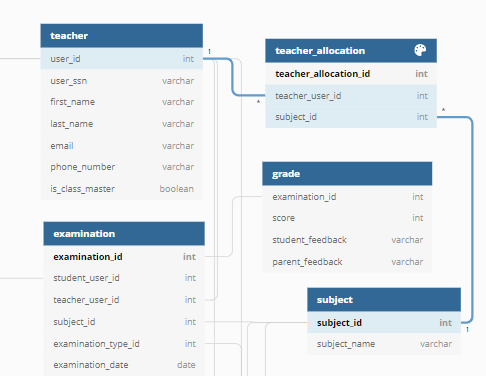


Figura 2‑6 Alocarea profesorilor pe materii

Referitor la tabela examination, este făcută pentru a rezolva relația many-to-many între elev și note.

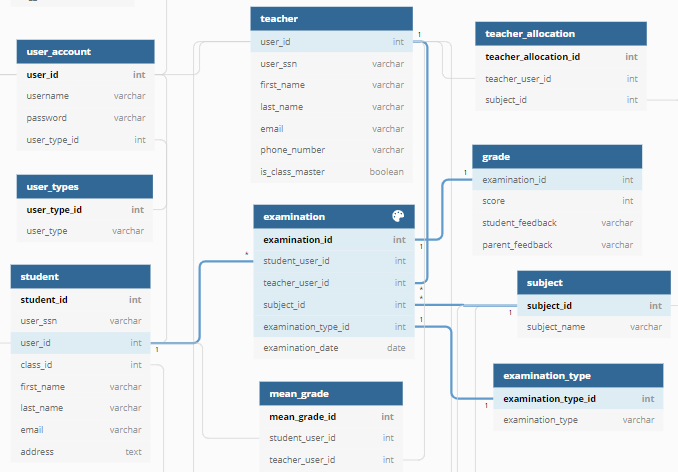


Figura 2‑7 Relația între examinare și restul entităților

## Descrierea procedurilor și funcțiilor

În implementarea proiectului am folosit **4 funcții** și **9 proceduri**.

### Funcții

Am creat funcții pentru autentificarea utilizatorilor, obținerea id-ului dirigintelui unei clase, aflarea specializării unei clase și aflarea materiei predate de un profesor.

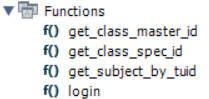


Figura 2‑8 Funcțiile folosite

### Proceduri

Procedurile sunt folostie fie pentru obținerea de informații ce curpind mai multe randuri din unul sau mai multe tabele, în cazu care sunt prelucrate următor pentru întocmirea rapoartelor. În această categorie intră toate procedurile gettere.

În plus, există câteva proceduri cu rolul de a adăuga sau actualiza informații din baza de date, cum este cazul procedurilor: add\_examination, compute\_average (care folosește **1 cursor**) și update\_teacher\_profile.

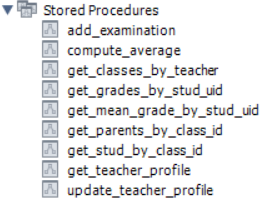


Figura 2‑9 Procedurile folosite

### Triggere

Am ales să folosec **2 triggere**: unul pe tabela examination și altul pe tabela log.

În cazul tabelei examination, trigger-ul are rolul de a adăuga nota aferentă examinării în tabela grades.

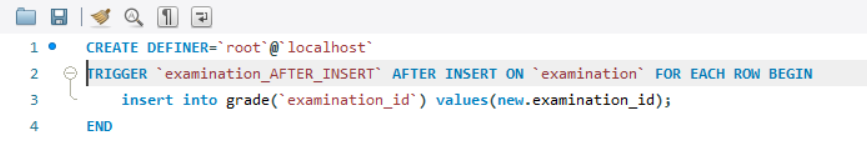


Figura 2‑10 Trigger pentru tabela examination

Pentru tabela teacher am creat un trigger astfel încât actualizarea profilului să declanșeze un log.

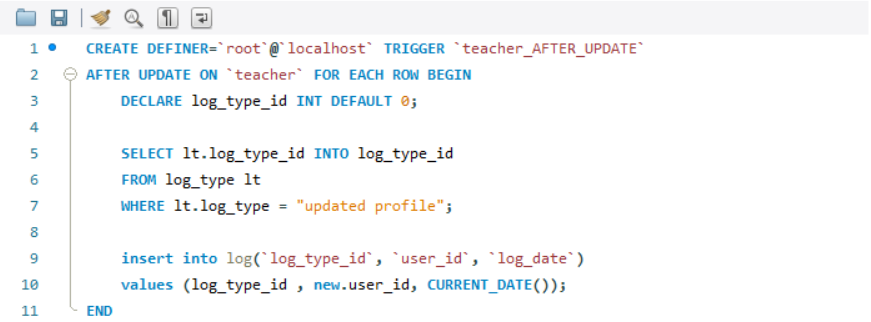


Figura 2‑11 Trigger pentru tabela teacher

# Descrierea aplicației

## Structura claselor

Dat fiind faptul ca nu am folosit ORM-uri și am ales să folosesc flask și limbajul python, proiectul nu este organizat pe clase, ci există un singur fișier în backend, anume app.py, iar pentru frontend, există aprofimativ un fișier HTML pentru fiecare interfață și un script în javascript pentru fiecare (mai multe detalii în readme).

## Conexiunea cu baza de date

Conexiunea backend-ului cu baza de date se face la începutul programului, folosind flask și specificând adresa IP și portul pe care rulează containerul de baze de date.

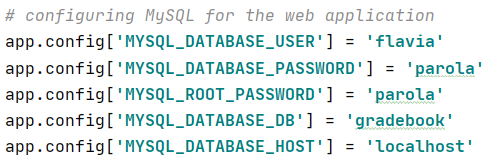


Figure 3‑2 Conexiunea cu baza de date

## Diagrama de stări și fluxul de lucru

Mai jos se află diagrama flucului de lucru:

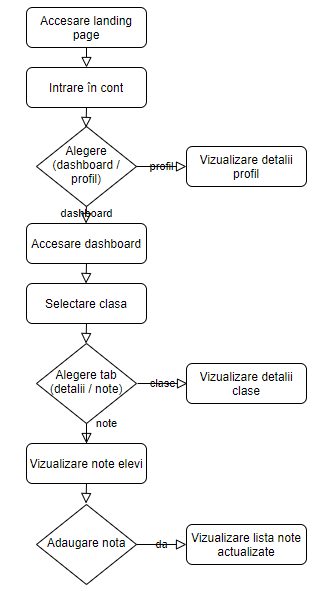


Figure 3‑1 Diagrama fluxului de lucru

# Interfețe și rapoarte

În cadrul aplicației există următoarele **interfețe**:

1. pagina de login
2. dashboard-ul profesorului, unde poate vizualiza clasele la care predă
3. detaliile despre o clasa: diriginte, specializare
4. notele elevilor de la o clasa (tot de aici se și adaugă notele)
5. pagina de profil

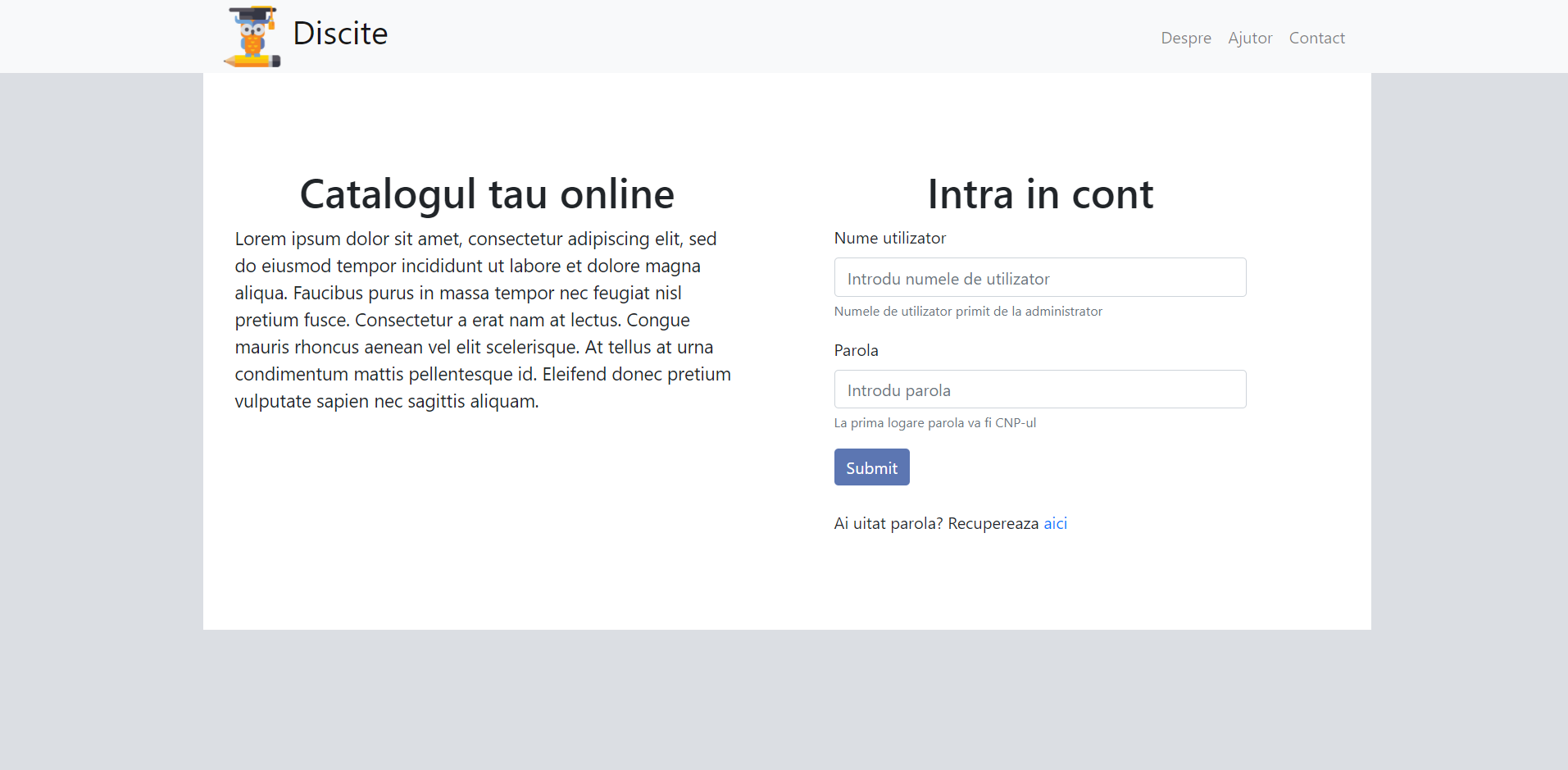


Figura 4‑1 Pagina de login

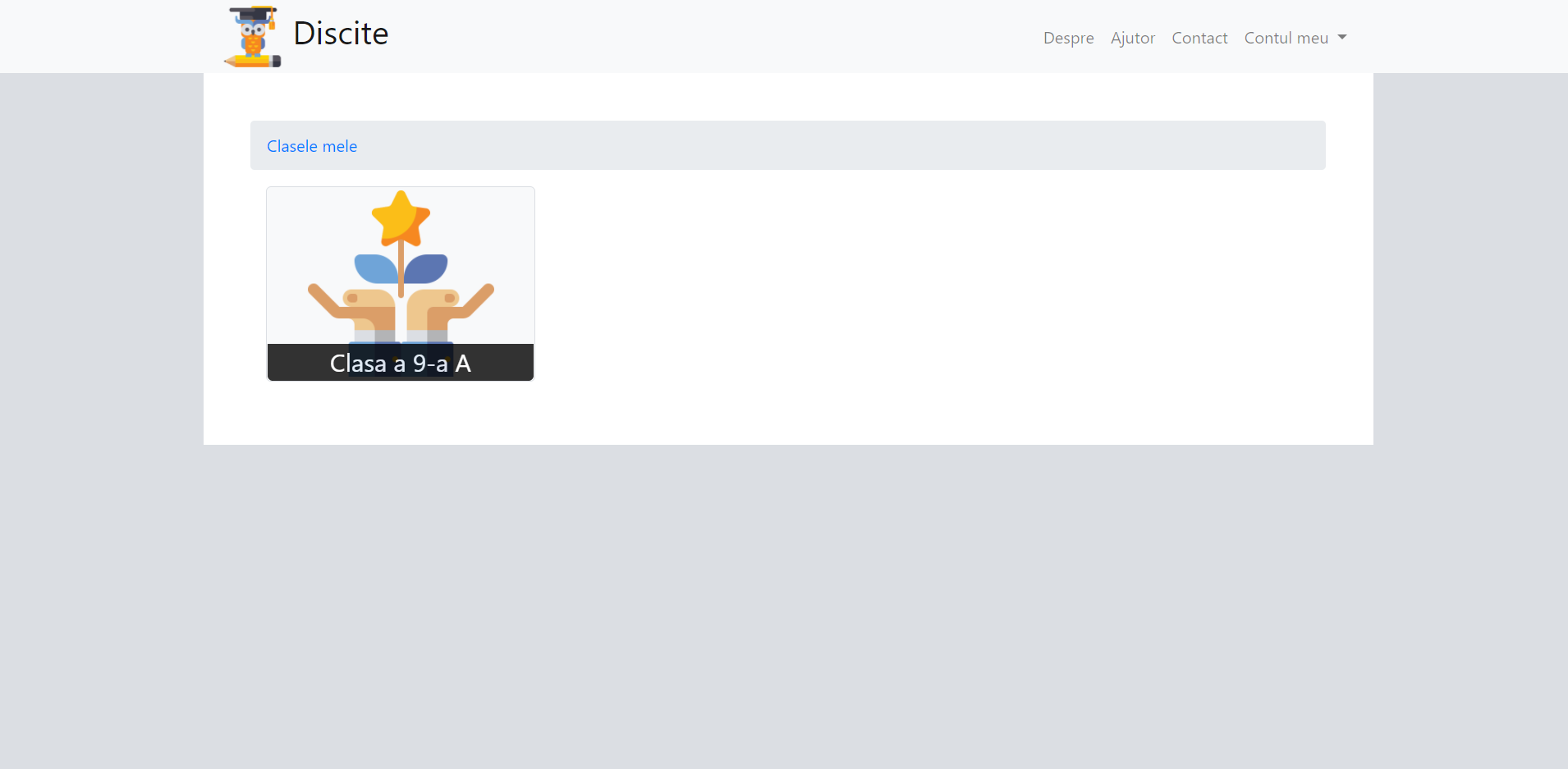


Figura 4‑2 Dashboard-ul profesorului

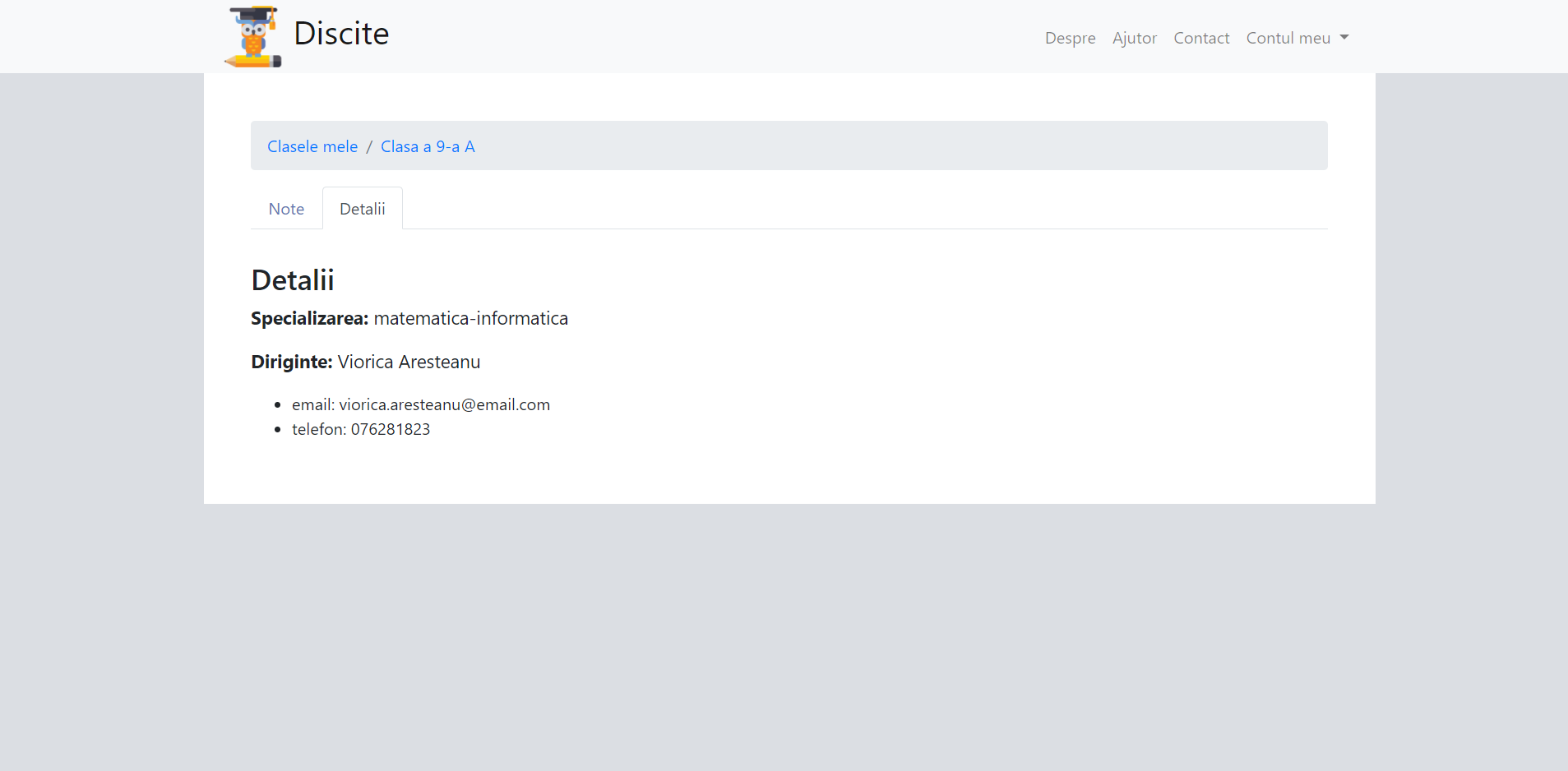


Figura 4‑3 Detaliile despre o clasa

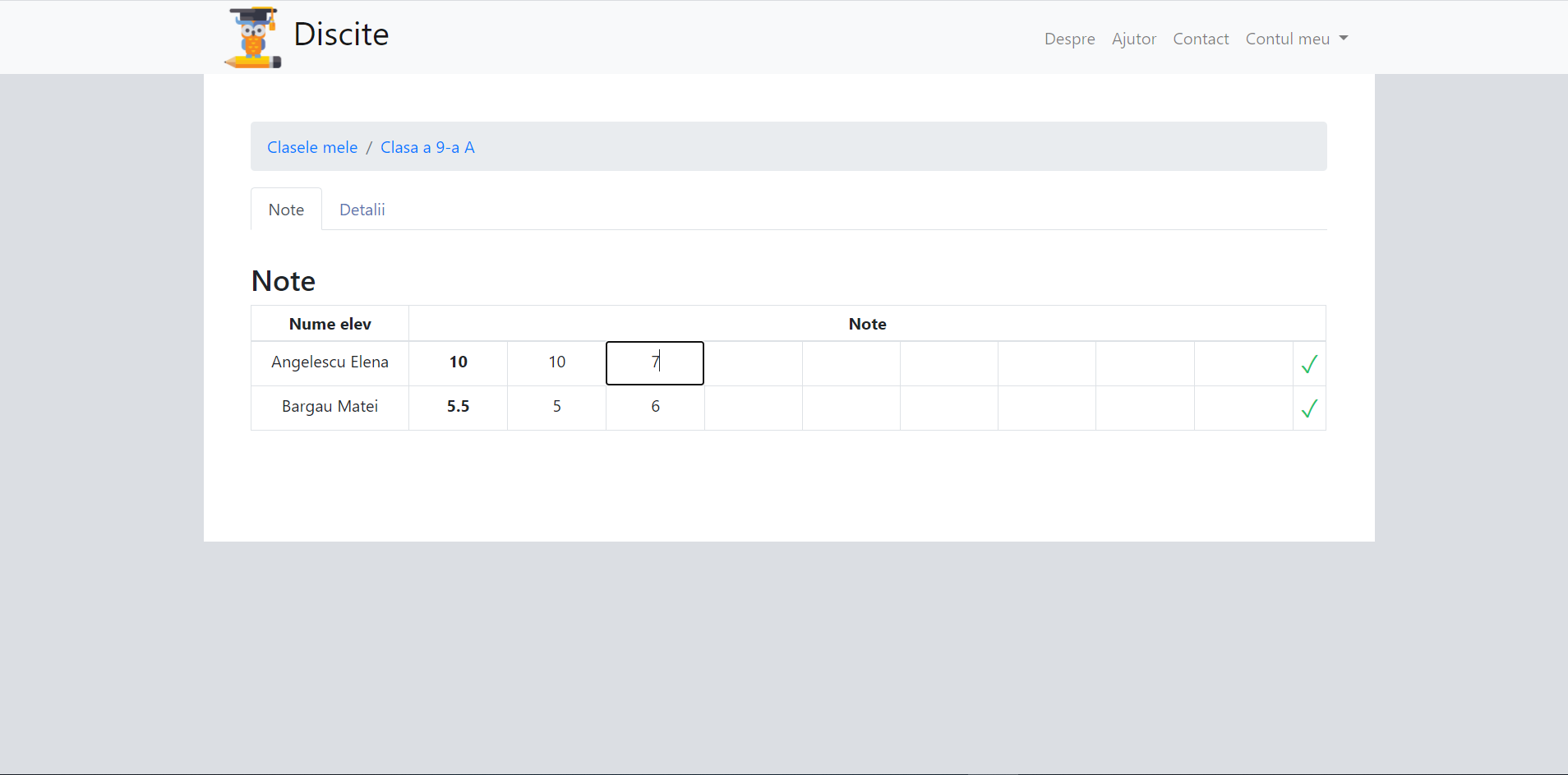


Figura 4‑4 Vizualizare si adaugare note

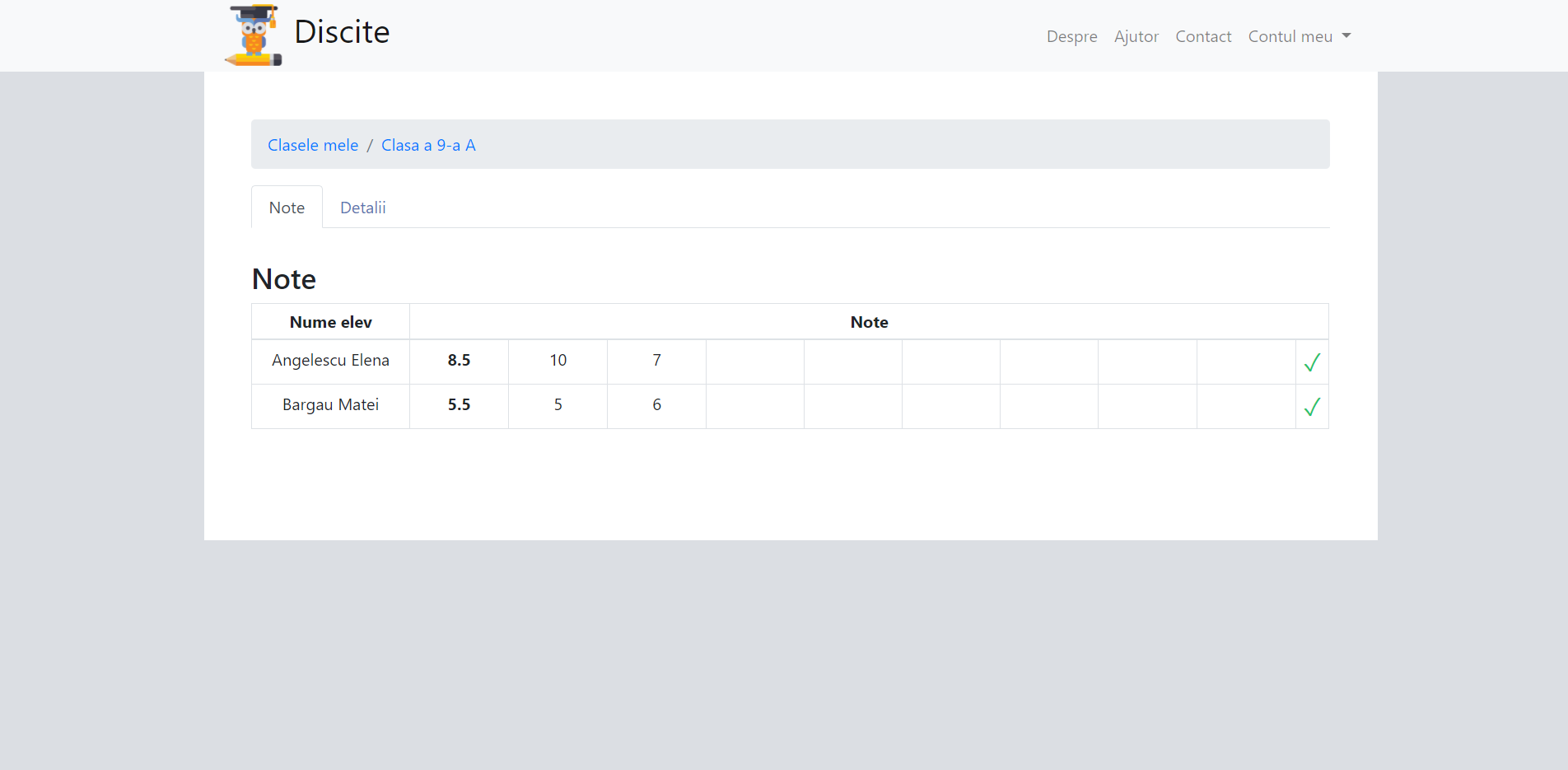


Figura 4‑5 Actualizare medie în timp real

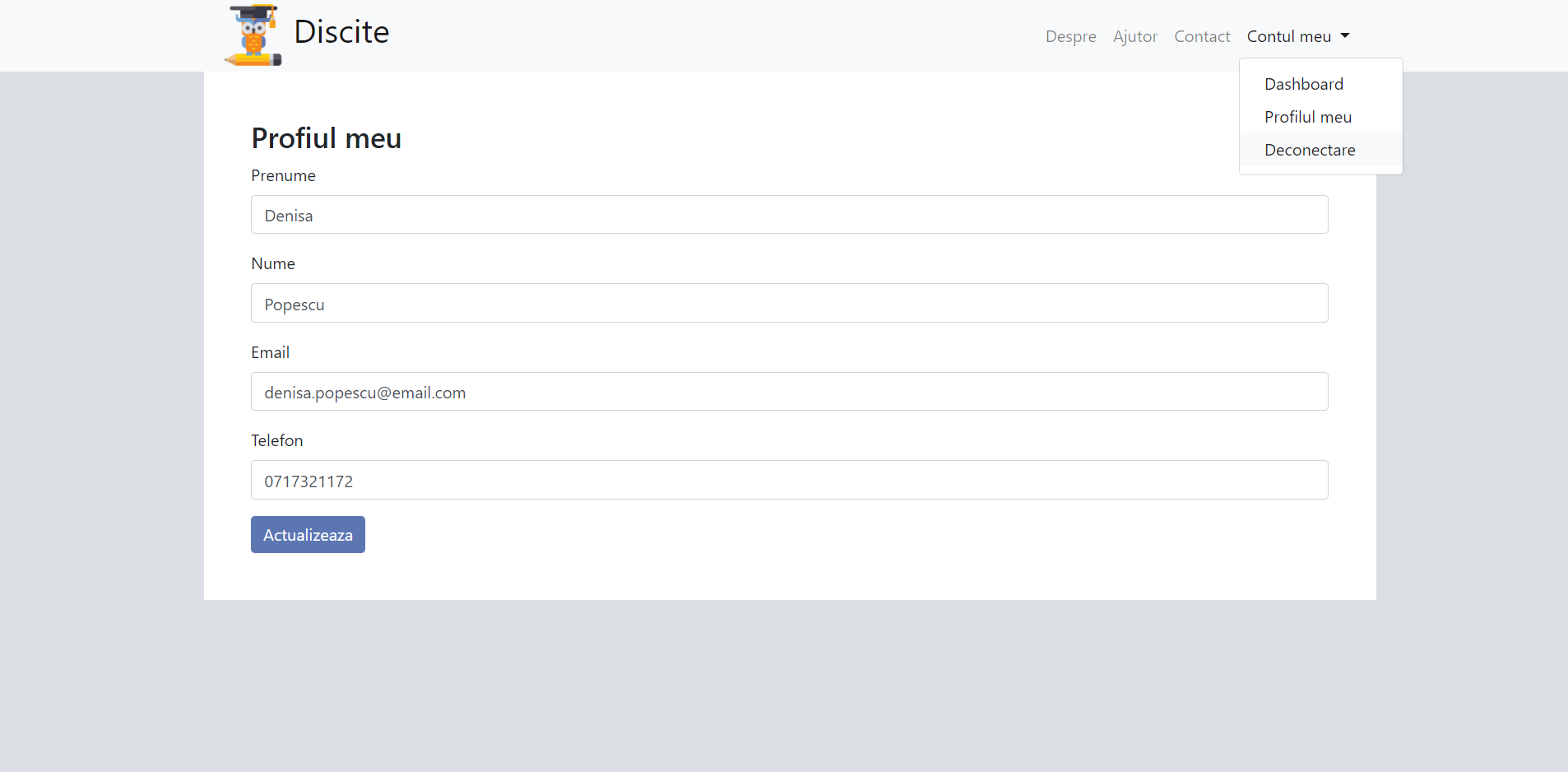


Figura 4‑6 Vizualizare profil și Logout

**Rapoartele** folosite sunt:

1. vizualizarea claselor
2. vizualizarea detaliilor despre o clasa
3. vizualizarea notelor
4. vizualizarea detaliilor despre profilul de utilizator

# Concluzii

Proiectul ca un întreg se dorește a fi mult mai amplu decât implementarea realizată, dar, dat fiind timpul limitat, am dezvoltat o parte semnificativă din interfața dedicată profesorului.

Dificultatea proiectului a venit din faptul că s-a cerut crearea unei aplicații full-stack, dar limitandu-ne la tehnologii mai puțin folosite (aproape toate framework-urile folosesc ORM-uri).

# Bibliografie

Utilitar pentru crearea bazelor de date: <https://dbdiagram.io/>

Documentația flask: <https://flask.palletsprojects.com/en/1.1.x/>

Documentația Bootstrap: <https://getbootstrap.com/docs/4.0/getting-started/introduction/>

Dockerizare: <https://hub.docker.com/_/mysql>