

Documentație

"Sistem de management a studenților"

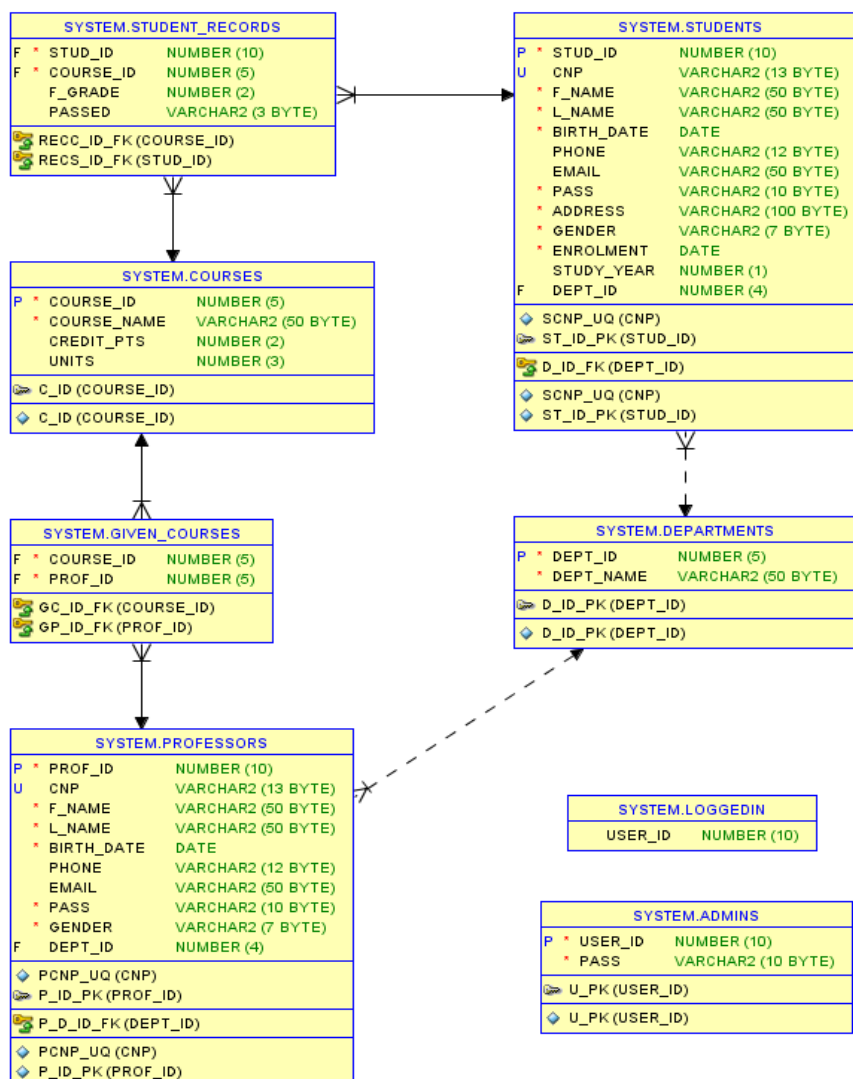
1.Descrierea temei

În circumstanțele mediului în care trăim, educația are un rol esențial în dezvoltarea continuă a oricărei țări. Calitatea pregătirii specialiștilor determină soarta viitoare a universităților. Precum multe întreprinderi industriale, universitățile din întreaga lume au intrat într-o concurență masivă atât pentru piața de furnizori, cât și pentru cea de consum. În acest sens, în universități s-a concluzionat că este necesară crearea condițiilor care să asigure nivelul necesar de pregătire a profesioniștilor, și să învețe a-l utiliza.

În proiectul ales de mine pentru materia Baze de Date am optat pentru crearea unui sistem de gestiune a studenților. Managementul studenților și cadrelor didactice în cadrul unei universități este esențială pentru funcționarea continuă și progresivă a blocurilor didactice. Importanța centralizării datelor și posibilitatea gestionării și manipulării acestora oferă o flexibilitate sporită, permite un management rentabil al timpului atât al angajaților facultății/universității, cât și al studenților.

Această platformă de management a studenților este concepută pentru utilizarea în cadrul unei facultăți. În platformă pot fi vizualizate datele cu caracter personal pentru anumite tipuri de utilizatori, cât și pot fi modificate (adăugate sau șterse) date, de către un tip de utilizatori privilegiați, care au o relevanță majoră pentru funcționarea continuă și organizată a facultății.

Pe piață se găsesc produse similare de management al studenților, facultăților sau cursurilor. Majoritatea au variate tipuri de utilizatori și prezintă o gamă largă de funcționalități specifice fiecărui utilizator. Printre acestea se numără studenti.pub, moodle etc.



Figură 1. Diagrama bazei de date

1.1 Descrierea bazei de date

Pentru crearea bazei de date am folosit serverul Oracle : container-registry.oracle.com instalat în docker. Sistemul de gestiune a bazei de date (SGBD) este de tip relațional, suportă limbajul de programare procedurală PL/SQL. Baza de date conține tabele, pachete, funcții, proceduri, triggeri și secvențe. O detaliere mai amplă este descrisă în subpunctele ce urmează.

1.1.1 Diagrama bazei de date

După cum se poate vede în figura 1, diagrama bazei de date este formată din 8 tabele: Students, Professors, Student_Records, Courses, Departments, Given_courses, Logedin, Admins.

1.1.2 Structura tabelelor

Students reprezintă o instanță a utilizatorului student. Tabelul conține următoarele câmpuri: stud_id , reprezintă id-ul studentului, este unic și se generează la inserare, cnp este codul numeric personal al studentului, f_name – prenumele studentului, l_name - numele studentului, birth_date- ziua de naștere a studentului, phone – telefonul de contact al studentului, email – email-ul studentului, pass – parola studentului, se generează automat și este formată din "student" + ultimile 3 cifre din cnp, address – reședința studentului, gender – sex-ul studentului, enrolment – data de înmatriculare a studentului, study_year – anul de studii al studentului, dept_id – departamentul în care studiază studentul. Studentii pot avea id-ul de la 1001 la 9999999999.

Professors este tabelul care păstrează tipul de utilizatori profesor, ca și datele acestuia: prof_id – id-ul unic al profesorului, se generează la inserare, cnp – codul numeric personal, f_name – prenumele profesorului, l_name – numele profesorului, birth_date – data de naștere a profesorului, phone - numărul de telefon, email - adresa de email, pass – parola profesorului "professor", gender – sex-ul profesorului, dept_id – departamentul din care face parte profesorul. Id-ul unic al profesorilor se încadrează în intervalul [1,1000].

Departments stochează departamentele facultății. Dept_id – id-ul departamentului, dept_name – numele departamentului.

Courses salvează toate cursurile disponibile în facultate. Course_id – id-ul cursului, course_name – numele cursului, credit_pts – numărul de puncte credite oferite, units - numărul de capitole din curs.

Given_courses păstrează cursurile oferite de profesori: course_id, id-ul cursului din tabelul courses, prof_id- id-ul profesorului din tabelul professors

Student_records este format din stud_id – id-ul unic al studentului, preluat din tabelul students, course_id – id-ul unic al cursului, preluat din tabelul de cursuri, f_grade - este nota finală, passed – reprezintă statusul studentului în momentul actual, poate fi "yes", "no" sau "none".

Logedin salvează id-ul utilizatorilor logați în sistem, unde user_id este id-ul unic al utilizatorului.

Admins stochează administratorii sistemului, unde user_id este id-ul utilizatorului, iar pass este parola lui. Inițial este un singur administrator care se loghează cu username-ul 0, care este și id-ul, și parola "admin".

1.1.3 Descrierea constrângerilor de integritate

Tabelul students: cheia primară este stud_id, aceasta indică unicitatea utilizatorului student, cnp de asemenea este unic pentru fiecare student, câmpurile f_name, l_name, birth_date, pass, gender și enrolment nu pot fi nule. Cheia străină este dept_id, face conexiunea cu tabelul departments.

Tabelul professors: cheia primară este prof_id, o instanță profesor este unică și se poate identifica prin id, cnp-ul de asemenea este cheie unică nenulă, câmpurile f_name, l_name, birth_date , pass și gender au constrângerea not null. Cheia străină este dept_id, face conexiunea cu tabelul departments.

Tabelul departments are setată cheia primară pe dept_id, este unic în sistem, iar dept_name nu poate fi null.

Tabelul courses: cheia primară este course_id, poate fi identificat un singur curs cu un id, course_name nu poate fi null.

Tabelul given_courses, conține doar chei străine: course_id face conexiunea cu tabelul courses, prof_id, face conexiunea cu tabelul professors.

Tabelul student_records, conține doar chei străine, stud_id , ia id-ul din tabelul students, iar course_id- din tabelul courses.

Tabelul admins: cheia primară este user_id, este unică în sistem, pass nu poate fi null.

1.1.4 Descrierea procedurilor și funcțiilor

Funcțiile și procedurile sunt salvate în două pachete diferite. Functions_pck și procedures_pck.

- functions_pck.isUserRegistered – verifică dacă utilizatorul este înregistrat
- functions_pck.checkPasswd - verifică parola la logare
- functions_pck.userType - verifică tipul utilizatorului (admin, student sau profesor)
- functions_pck.doCountProfInDept – numara profesorii din departament
- functions_pck.doCountStudentsInDept-numara studenții din departament
- functions_pck.checkDept – verifică dacă departamentul există în baza de date
- functions_pck.doCountFemale – numara câte studenți sunt fete, returnează numărul de fete și genurile studenților distincte
- functions_pck.doCountMen – numara câți băieți studenți sunt
- functions_pck.addCourseStud – adaugă un curs studentului în tabelul student_records
- functions_pck.addGivenCourse – adaugă un curs în tabelul given_courses

- procedures_pck.addStudent – adaugă student în baza de date
- procedures_pck.addProfessor – adaugă profesor în baza de date
- procedures_pck.addDepartment – adaugă un departament în baza de date
- procedures_pck.addCourse – adaugă curs în baza de date
- procedures_pck.loginUser – adaugă user logat
- procedures_pck.logoutUser – deloghează user
- procedures_pck.removeGivenCourse- șterge curs din lista celor predate
- procedures_pck.removeCourse – șterge curs din baza de date
- procedures_pck.removeDepartment – șterge departament
- procedures_pck.removeProfessor – șterge profesor
- procedures_pck.removeStudent – șterge student
- procedures_pck.removeStudRecord – șterge un curs de la student
- procedures_pck.showCourses – arată toate cursurile
- procedures_pck.showDepts – arată toate departamentele
- procedures_pck.getAllInfoStudent – extrage toată informația despre un anumit student
- procedures_pck.getAllInfoProf – extrage toată informația despre un anumit profesor
- procedures_pck.showProfessors- selectează toți profesorii
- procedures_pck.showProfCourses – selectează cursurile unui anumit profesor
- procedures_pck.getCourseData – extrage datele despre un anumit curs
- procedures_pck.showStudents – extrage toți studenții
- procedures_pck.studyYears – extrage anii de studii
- procedures_pck.showStudCourses – extrage cursurile unui anumit student
- procedures_pck.updateCourseStud – actualizează datele cursului unui student

1.2 Descrierea aplicației

Platforma de management al studenților are ca scop esențial organizare structurată a informației despre studenți și pentru studenți. În cadrul aplicației pot exista 2 tipuri de utilizatori: student și/sau profesor, și administratorul bazei de date. Pentru a accesa aplicația fiecare utilizator va trebui să se logheze în sistem cu id-ul unic și parola asignată contului său.

Administratorul bazei de date operează cu toate datele de input și output în cadrul platformei. Astfel acesta poate să creeze și să ștergă departamente / profesori / cursuri / studenți.

Studentul/profesorul poate să își vizualizeze datele personale, cursurile la care este asignat/ pe care le predă.

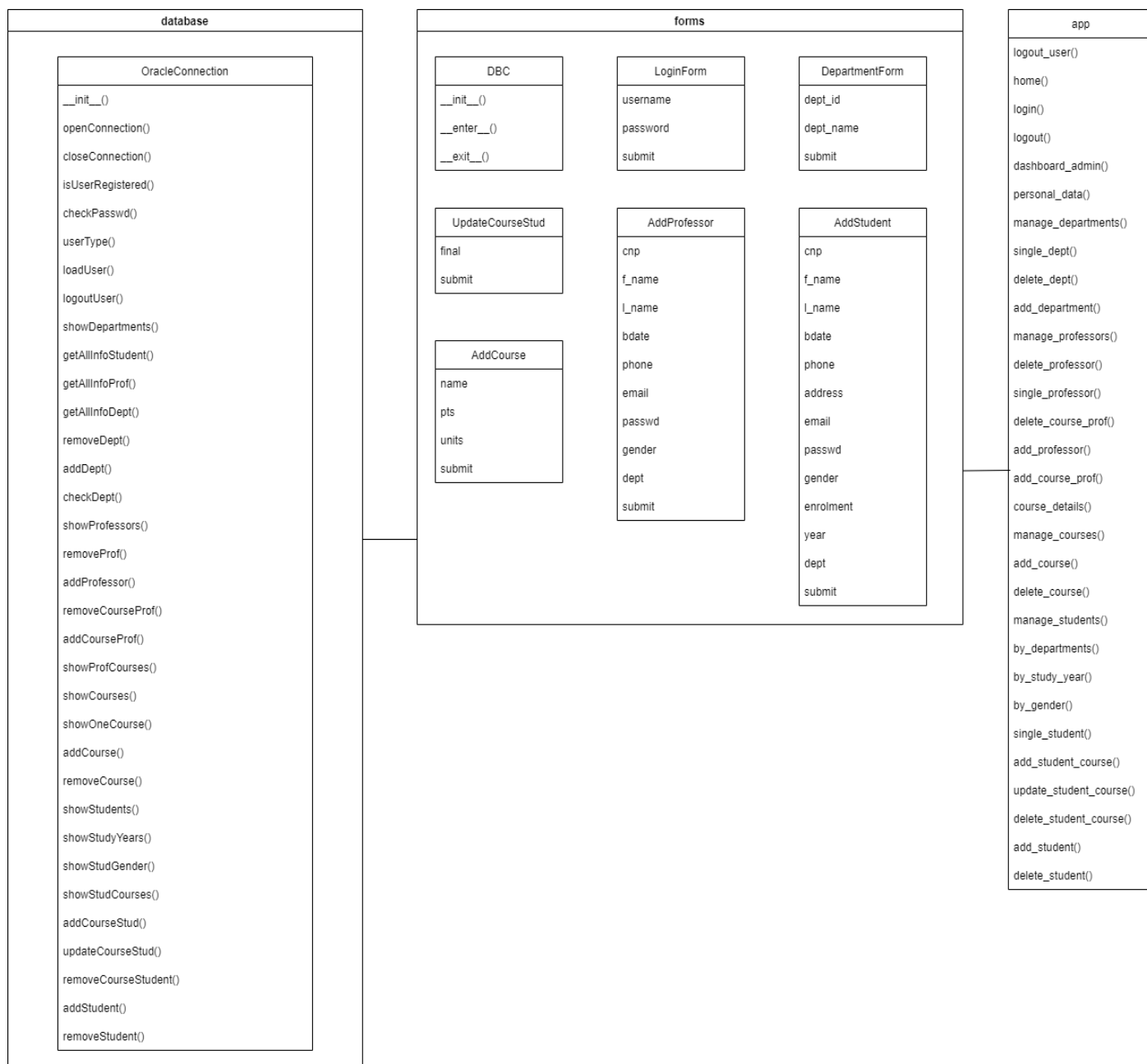
1.2.1 Diagrama de clase

A se vedea figura 2.

1.2.2 Structura claselor

Pentru implementare am folosit limbajul de programare python. Cele 3 containere primare sunt 3 module de python în care am implementat aplicația.

Database conține o singură clasă: OracleConnection care realizează conexiunea dintre baza de date și front-end. Metodele acestei clase invocă baza de date și extrag informațiile necesare. Ele extrag datele din funcțiile și procedurile descrise în oracle.

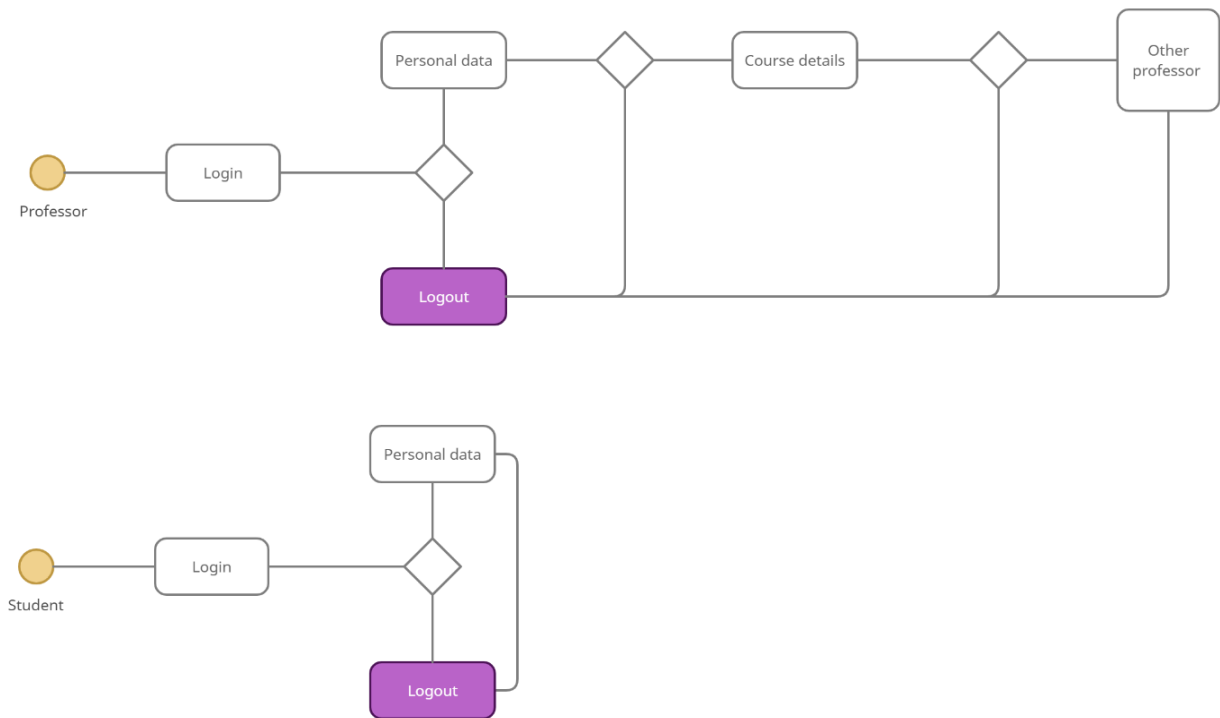


Figură 2. Diagrama de clase

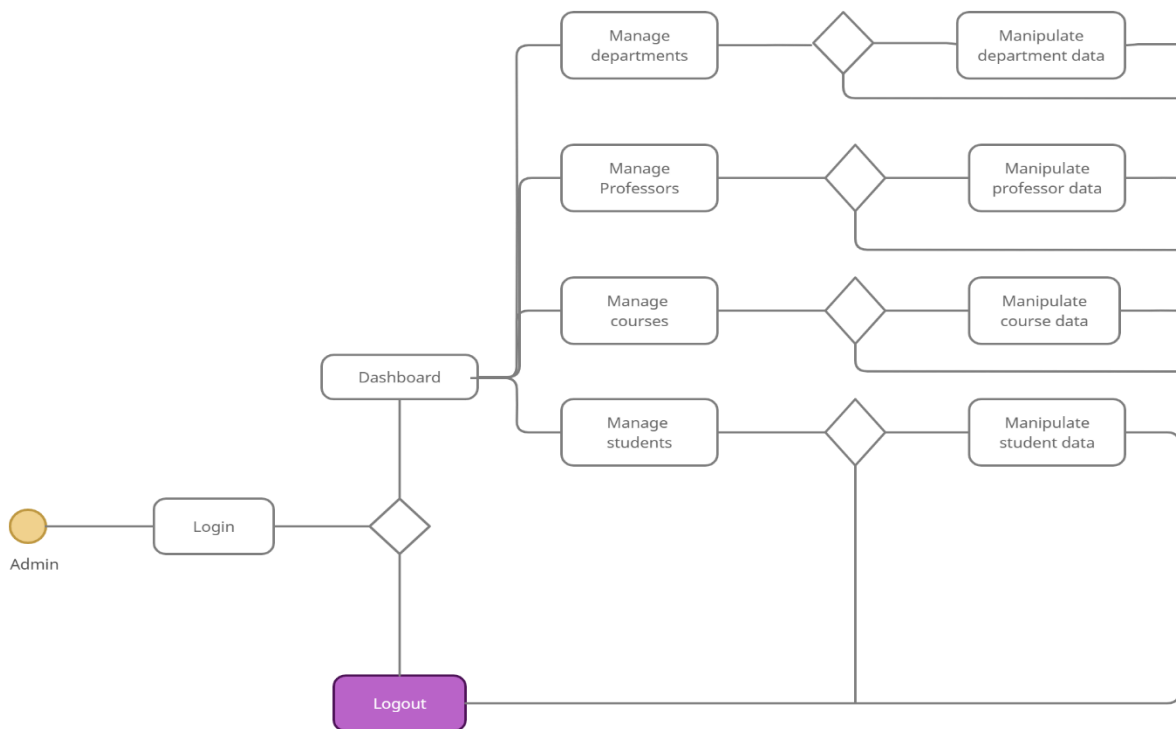
Forms este modulul de python care include 7 clase diferite. DBC deschide și închide o conexiune a bazei de date. Din app de fiecare dată când este efectuată o cerere către baza de date se creează o instanță DBC care închide și deschide conexiunea cu baza de date. Restul claselor reprezintă formulare de logare și înregistrare a noilor date în baza de date pentru adăugarea unui student, profesor, curs, departament și actualizare.

App implementează toate funcțiile pentru front-end. Aici sunt specificate rutele și metodele (POST și GET) necesare efectuării cererii.

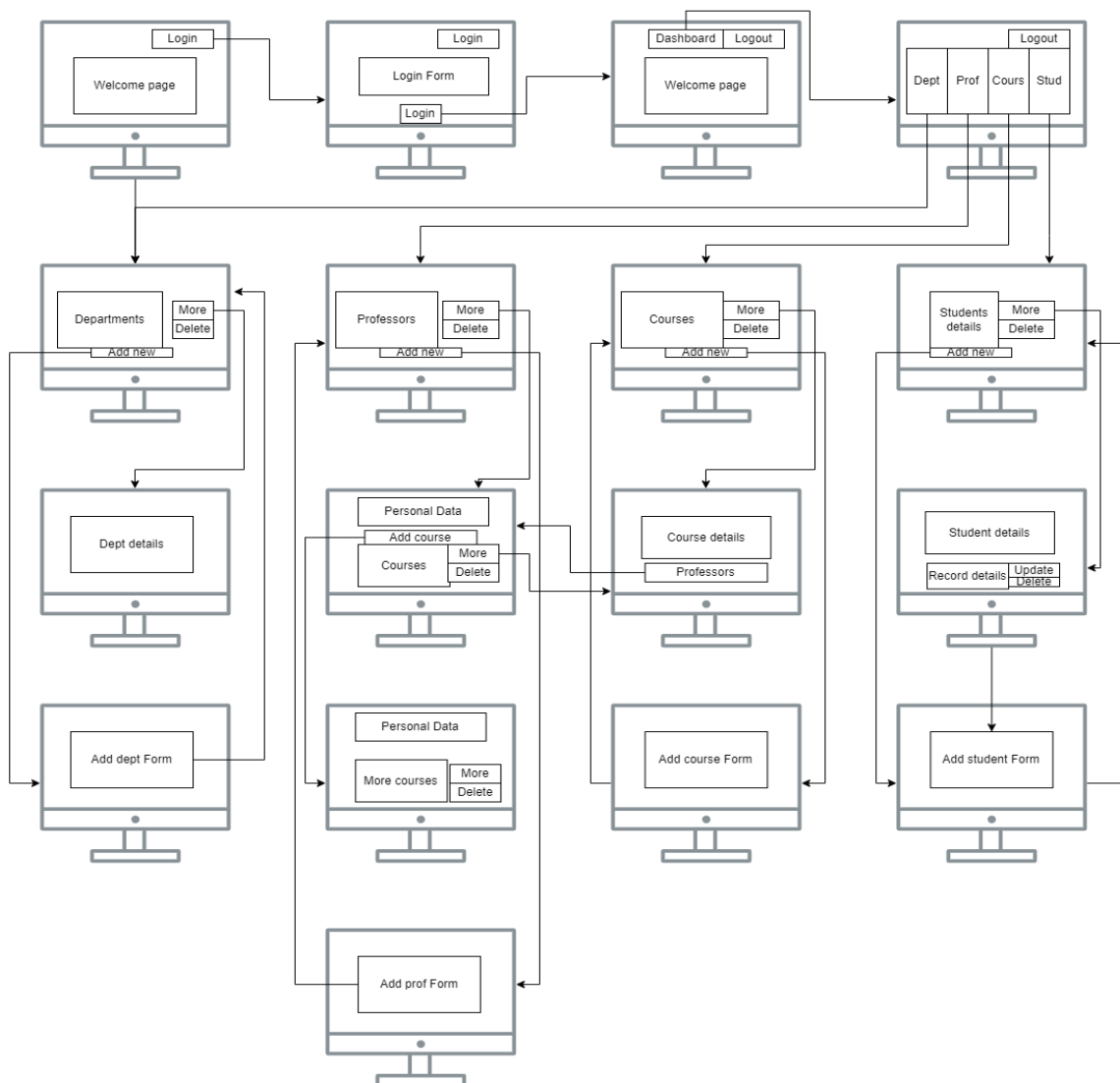
1.2.2 Diagrama de stări și fluxul de lucru pentru aplicație



Figură 3. Diagrama de stare pentru profesor și student



Figură 4. Diagrama de stare pentru administrator



Figură 5. Fluxul de lucru pentru utilizatorul admin

1.2.3 Prezentarea modului în care se face conexiunea cu baza de date

Proiectul a fost implementat folosind limbajul de programare Python. IDE folosit: PyCharm Community Edition 2021.3.3. Conexiunea cu baza de date am realizat-o prin intermediul pachetului cx_Oracle și Oracle Instant client. Am creat o clasă, OracleConnection care realizează conexiunea cu baza de date.

```
class OracleConnection:

    def __init__(self, host, port, schema, username, password):
        self.host = host
        self.port = port
        self.schema = schema
        self.username = username
        self.password = password
        self.cursor = None

    def openConnection(self):
        try:
            dsn_tns = cx_Oracle.makedsn(self.host, self.port, self.schema)
            self.db = cx_Oracle.connect(self.username, self.password, dsn_tns)
            self.cursor = self.db.cursor()
            print("Connection open!")
        except Exception as e:
```

```

        print("Connection not open!")
        print(e)

    def closeConnection(self):
        try:
            self.cursor.close()
            self.db.close()
            print("Connection close!")
        except Exception as e:
            print("Connection not closed!")
            print(e)

```

Pe partea de front-end am creat o nouă clasă care inițializează conexiunea, o deschide de fiecare dată când este nevoie să opereze cu baza de date, și o închide când procesul este terminat, pentru a omite posibilitatea creării mai multor sesiuni în paralel.

```

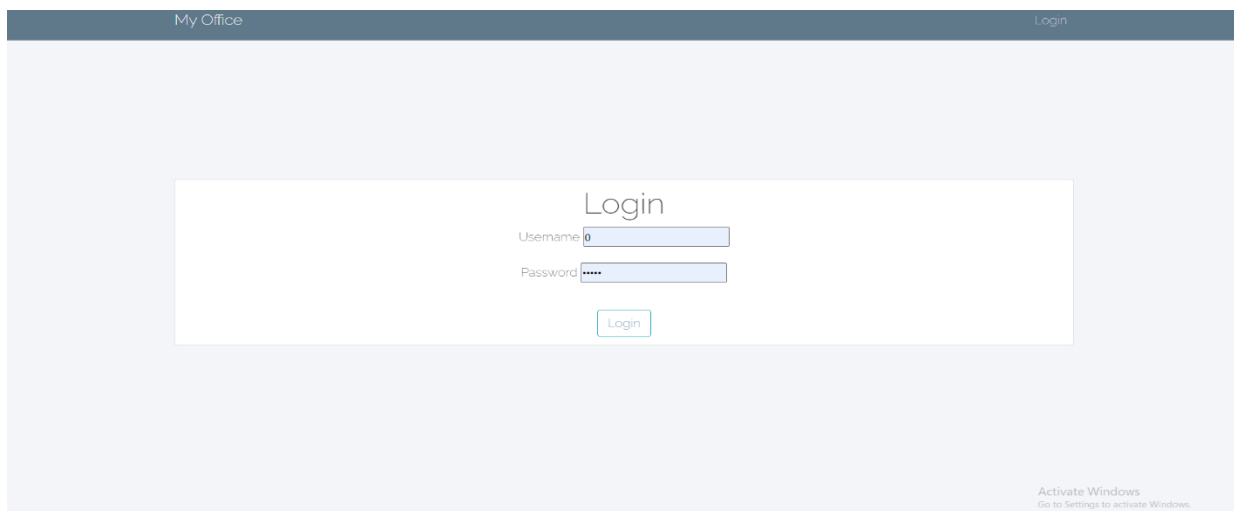
class DBC:
    def __init__(self):
        self.db = OracleConnection('host', port, 'SID', 'user', 'pass')

    def __enter__(self):
        self.db.openConnection()
        return self.db

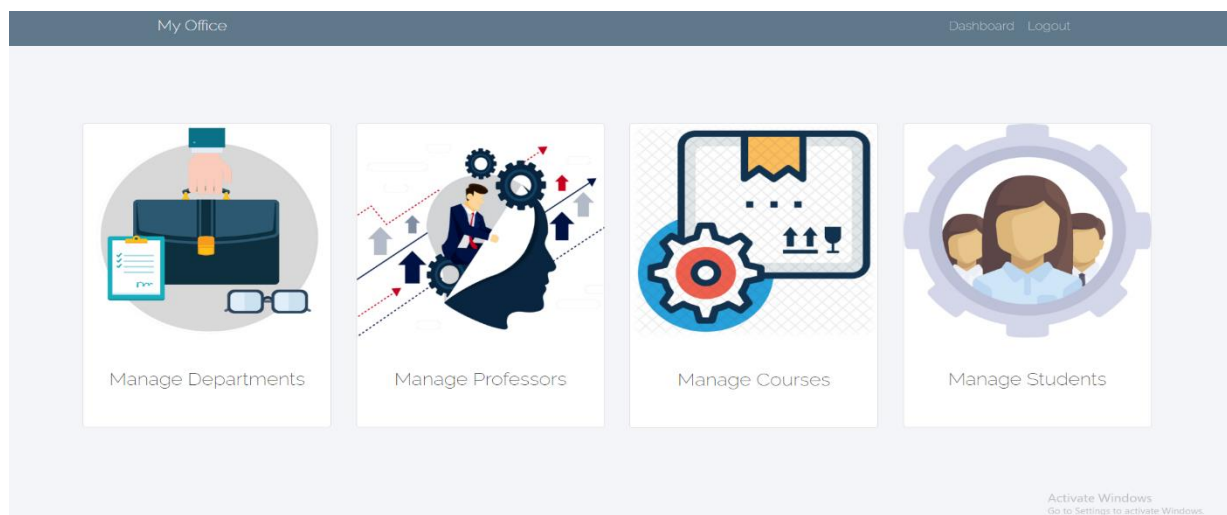
    def __exit__(self, exc_type, exc_val, exc_tb):
        self.db.closeConnection()

```

1.3 Capturi de ecran pentru interfețe și rapoarte



Figură 6. Pagina de login cont de administrator



Figură 7. Dashboard administrator

My Office						Dashboard Logout	
All students						Select view	Add new student
Study year 1							
ID	CNP	Full Name	Phone	Email	Study Year	Details	Delete
1016	1991207111114	Gheorghe Condorache	+4078298323	georgech@yahoo.com	1	Details	Delete
Study year 2							
ID	CNP	Full Name	Phone	Email	Study Year	Details	Delete
1004	1000413000001	Cristian Donici	+4071123111	cristi.x.don@gmail.com	2	Details	Delete
1020	1010801111112	Marian Cristescu	+40756565656	cristescu.marian@gmail.com	2	Details	Delete
Study year 4						Activate Windows Go to Settings to activate Windows.	

Figură 8. Dashboard administrator. Pagina de management a studenților ordonată după anul de studii

My Office

DashboardLogout

Department details

Department ID	2
Department Name	Calculatoare
Students Enroled	19
Professors	8

Figură 9. Dashboard administrator. Managementul Departamentelor. Detalii ale departamentului de Calculatoare. Include 2 funcții distincte de agregare (students enroled, professors)

1.4 Concluzii

Pentru implementarea cerinței de proiect am ales crearea unei platforme de management a studenților. Am creat proiectul folosind pentru back-end: serverul Oracle, limbajul procedural PL/SQL și mediul de dezvoltare SQL Developer, precum și limbajul de programare python. IDE folosit: PyCharm Community Edition 2021.3.3. Pentru front-end am folosit python, și framework-ul de dezvoltare web Flask, Bootstrap, HTML și CSS.

Proiectul mi-a oferit șansa de a face cunoștință cu dezvoltarea web full-stack, un domeniu care este foarte popular în circumstanțele în care ne aflăm. Mi-a plăcut să lucrez și să cunosc cum funcționează o bază de date la nivel de proiect. Am învățat cum fac conexiunea cu o bază de date. Chiar dacă pe alocuri au apărut erori pentru care au fost necesare zile până a le elucida, totuși a fost un prim proiect de genul dat pe care îl realizez complet singură, pot să afirm că mi-a plăcut, a fost interesant și foarte util. Un punct forte al proiectului a fost faptul că temele au fost alese de noi individual și unic, ceea ce înseamnă că conceptul și dezvoltarea a fost complet la latitudinea noastră (a studenților).

1.5 Bibliografie

- [1]. Conectarea la baza de date. <https://ocw.cs.pub.ro/courses/bd2/laboratoare/09>
- [2]. Proceduri și funcții. <https://ocw.cs.pub.ro/courses/bd2/laboratoare/05>
- [3]. Pachete și trigger. <https://ocw.cs.pub.ro/courses/bd2/laboratoare/06>
- [4]. Flask. <https://flask.palletsprojects.com/en/2.0.x/>
- [5]. Bootstrap 4. <https://getbootstrap.com/docs/4.0/getting-started/download/>