University of Pisa

SCUOLA DI INGEGNERIA
Corso di Laurea in Artificial Intelligence and Data Engineering

Task3 Documentation

Candidati
Alice Nannini
Giacomo Mantovani
Marco Parola
Stefano Poleggi

Relatore
Prof. Pietro Ducange

Anno Accademico 2019–2020

Contents

1 Introduction **1**

2 Analysis and workflow **2**

2.1 Requirements 2

2.1.1 Functional requirements 2

2.1.2 Non-functional requirements 2

2.2 Use cases 3

2.2.1 Use Cases Description 3

2.3 Analysis of entities 5

3 Design **6**

3.1 Database Choice 6

3.2 Software Architecture 6

3.3 Structure of the database 6

4 Implementation **8**

4.1 Used Technologies 8

4.2 Java Classes Description 8

4.2.1 Declaration of Student class 8

4.2.2 Declaration of Subject class 9

4.2.3 Declaration of Professor class 9

4.2.4 Declaration of Degree class 9

4.2.5 Declaration of Comment class 10

4.2.6 Database manager 10

4.3 CRUD operations 10

4.3.1 Create 10

4.3.2 Read 11

4.3.3 Update 11

4.3.4 Delete 11

5 User Manual **13**

5.1 Admin Manual 17

1 Introduction

This is an application to browse and evaluate university courses, called **Student-Evaluation**.
The application is developed to allow students (users) to view all the courses of a specific university with related comments.
Looking the table on the right side, a user can browse all subjects and by clicking an element of this table, on the left section, the user can see more information about the chosen element: general information and comments.
In order to filter the list of subjects, there is a choice box, thanks to which the user can select a specific degree course.
In order to leave a comment, it is necessary to log in, otherwise, the application will allow interaction in read-only.
There are two buttons in the bottom left corner to allow students to update or delete their comments. There is another button that allows a user to search students from a suggested list.

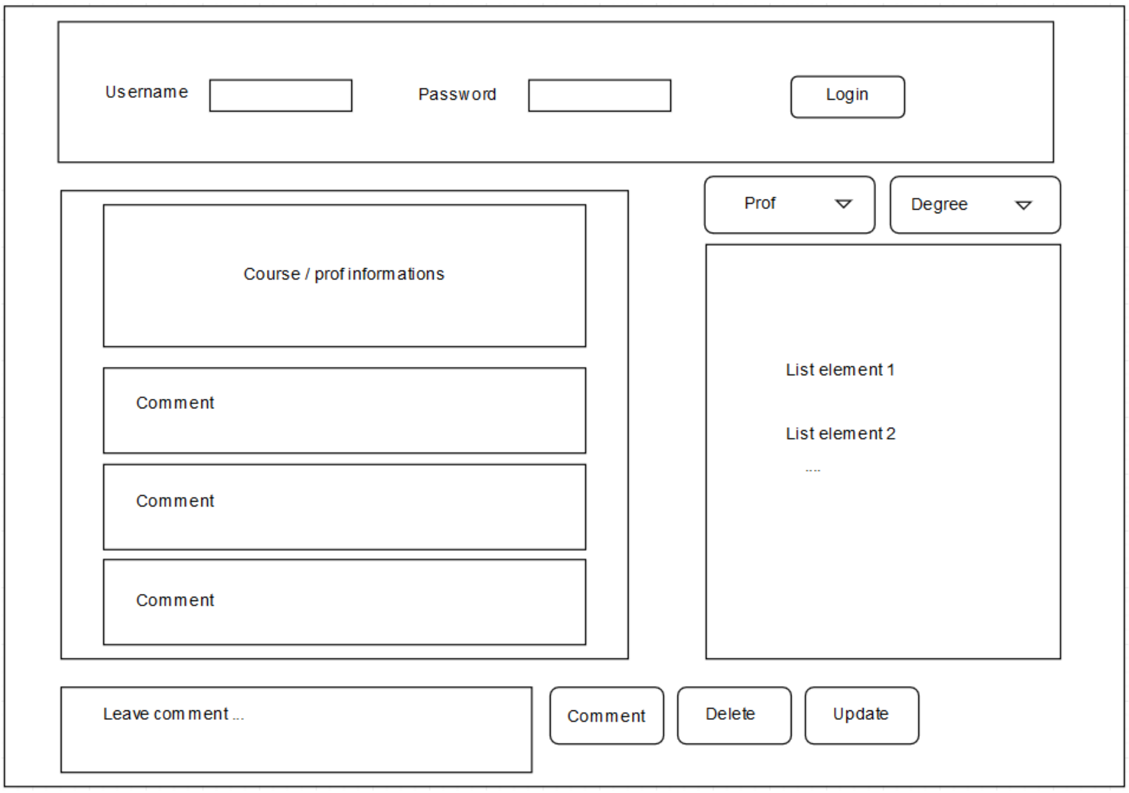


Figure 1: Mockup

2 Analysis and workflow

2.1 Requirements

2.1.1 Functional requirements

The system has to allow the guest to carry out basic functions such as:

- To select a course from the list and view information and comments.
- To select a degree course from the list, filtering subjects.

In addiction to the guest functions, the system has to allow the user to carry out basic functions such as:

- To login to the system.
- To upload comments on a course.
- To update a comment of a course only if the user is the owner.
- To delete a comment of a course only if the user is the owner.
- To add a friend.

The system has to allow the administrator to carry out basic functions such as:

- To login to the system.
- To add a course.
- To update a course.
- To delete a course.
- To add a professor.
- To update a professor.
- To delete a professor.
- To associate a professor to a course.
- To delete any comment.

2.1.2 Non-functional requirements

- Usability, ease of use and intuitiveness of the application by the user.
- The system should provide access to the database with a few seconds of latency.

2.2 Use cases

Actors

- Guest : this actor represents a user who is not logged into the system
- Student : this actor represents a user who is logged into the system
- Admin : this actor represents the administrator of the system

2.2.1 Use Cases Description

Event	**UseCase**	**Actor(s)**	**Description**
Log in, Log out	Login, Logout	Admin, Student	The user logs in/out the application. The system browses the professors' list by the degree course of the logged user and returns it on the interface.
View all the subjects	Browse, Find, Display S	Admin, Student, Guest	The user chooses that he wants to view the list of all subjects. The system browses the data on the db and returns them on the interface.
View the comments of a subject	Browse, Find, View C	Admin, Student, Guest	The user clicks on a record of the subject table. The system browses on the db the comments related to that subject and returns them on the interface.
Add a comment	Add C	Admin, Student	The user submits the text of his comment. The system updates the db and the interface.
Update a comment	Update C	Student	The user selects the comment and commits the new text. The system updates the db and the interface.
Delete a comment	Delete C	Admin, Student	The user selects the comment and submits the delete. The system updates the db and the interface.
View degree courses	Browse, Find, Display D	Admin, Student, Guest	The system browses the degree courses list and returns them on the interface.
Add a subject/professor	Add P/S	Admin	The admin submits the name and other information of the new subject/professor. The system updates the db and the interface.
Update a subject/professor	Update P/S	Admin	The user selects the subject/professor and commits the new information. The system updates the db and the interface.
Delete a subject/professor	Delete P/S	Admin	The user selects the subject/professor and submits the delete. The system updates the db and the interface.
View suggested friends	Browse, Find, Display Student	Student, Admin	The user chooses that he wants to view the suggested friends. The system browses on the db the students and returns them on the interface.
Set find parameter	By Degree, By Subject	Admin, Student, Guest	The user chooses the filter to apply to the query on the database. The system uses it to find the results.

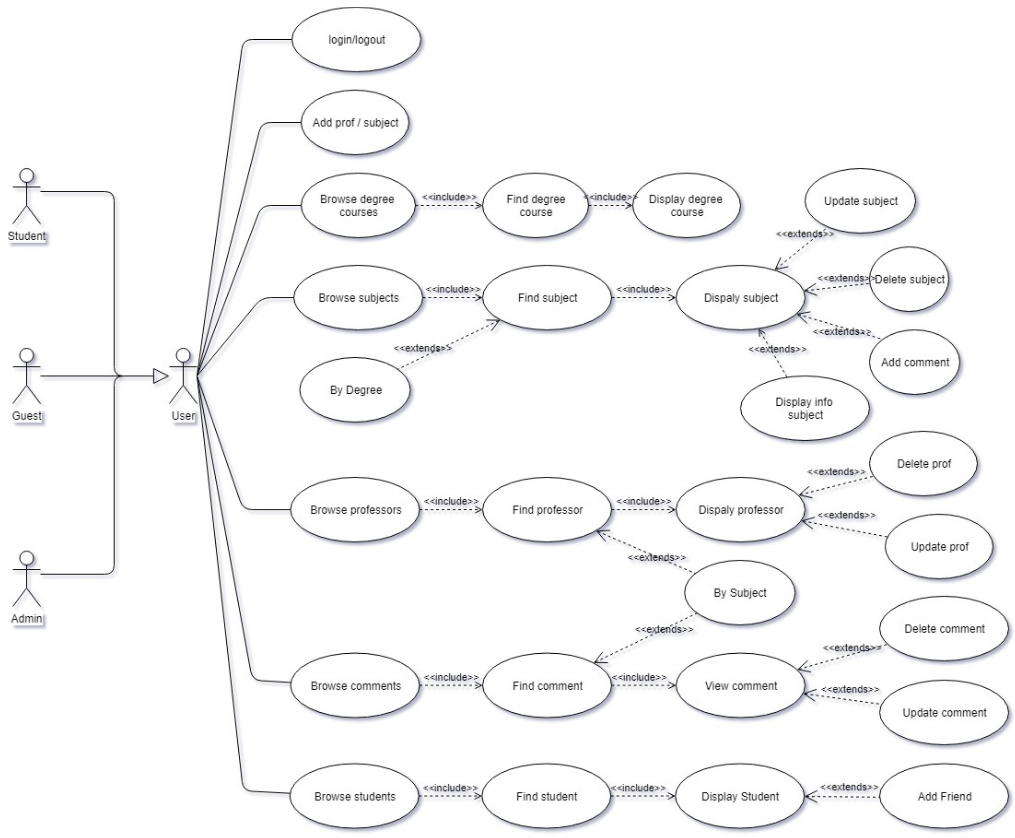


Figure 2: Use cases diagram

2.3 Analysis of entities

This diagram represent the main entities of the application and the relations between them.

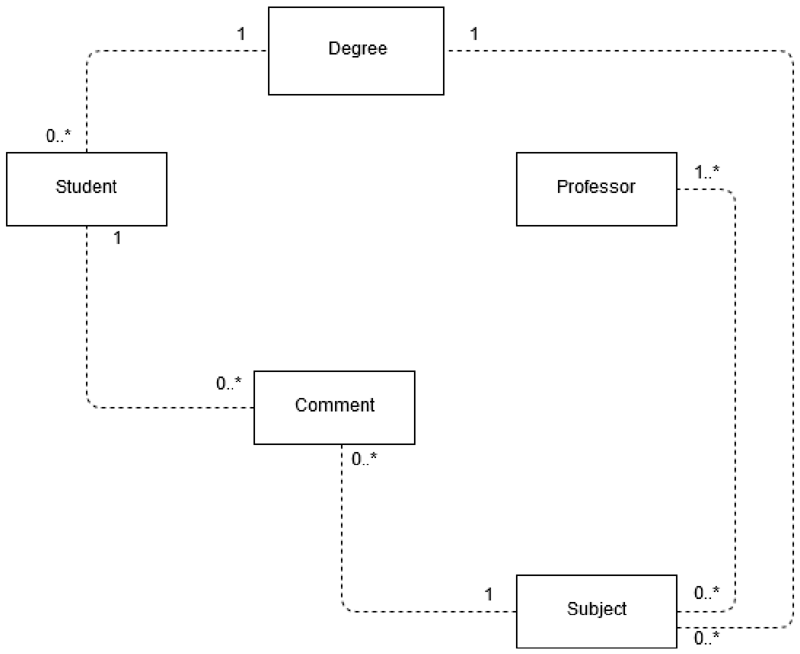


Figure 3: UML analysis diagram

3 Design

3.1 Database Choice

This application is based on many join operations between the entities, in order to obtain each professor associated with a subject, or each comment associated with a subject, or yet display all the subjects associated with a degree course. For this reason, if the amount of data available is very high as expected, a relational database will be computationally expensive. Then the choice fell on a graph database, which manage to work very well and very fast on a model made with a lot of entities and relations. Moreover, a graph database is very useful for some specific queries in the application,for example, showing a suggested friends list.

3.2 Software Architecture

The application is designed over 2 different layers, see figure 4:

- Front-end
- Back-end

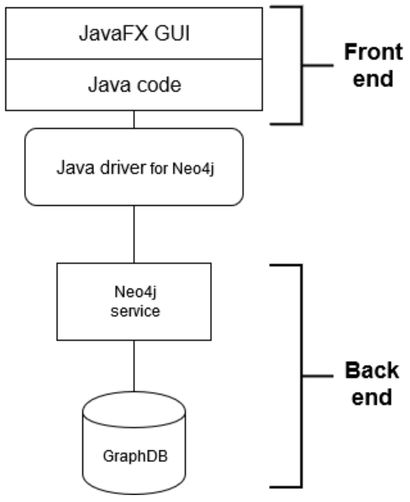


Figure 4: Software architecture diagram

3.3 Structure of the database

In graph databases the entities are modeling using the vertexes; in our model there are the following entities:

- Professor
- Student
- Degree
- Subject
- Comment

Moreover the relations are modeling using the edges; in our model there are the following edges:

- Attends, connects a student to a degree
- Teaches, connects a professor to a subject
- Belongs, connects a subject to a degree
- Wrote, connects a student to a comment
- Knows, connects a student to a student

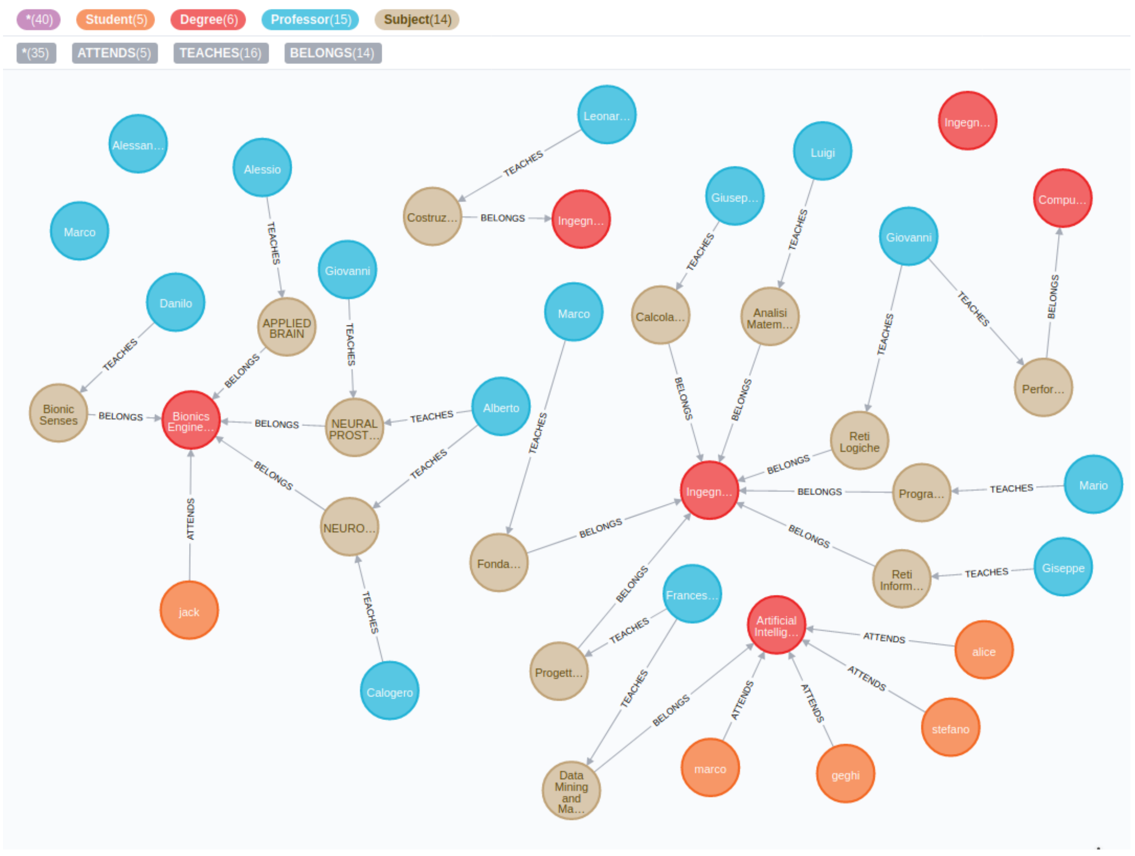


Figure 5: Graph model

4 Implementation

4.1 Used Technologies

The application is developed in java programming language, version 11.0.4, and in JavaFX system to create the GUI, version 11, so it should run on each platform in which JVM is installed, but the application is tested and guardantee on Ubuntu 16 and Window OS. Moreover Maven is used to build and mantain the project, version 3.8.0.
The java driver for Neo4j manages the comunication between the client application layer and the database backend layer, version 3.2.1.
For the backend layer it is used a graph database: Neo4j, version 3.2.1.
So this application is tested using these technologies, considering these particular versions: for other versions the correct execution isn't guaranteed .

4.2 Java Classes Description

In this section are listed the classes who model the entities.

- Student
- Professor
- Subject
- Comment
- Degree

Each of them have some attributes and the related Get and Set methods.

4.2.1 Declaration of Student class

```
  public class Student{
private final SimpleIntegerProperty id;
    private final SimpleBooleanProperty admin;
    private final SimpleStringProperty username;
    private final SimpleStringProperty password;
    private final SimpleObjectProperty<Degree> degree;
    private final SimpleListProperty<Student> friends;

    // CONSTRUCTOR
    public Student(int i, String u, String p, Degree d, boolean a) {
        id = new SimpleIntegerProperty(i);
        admin = new SimpleBooleanProperty(a);
        username = new SimpleStringProperty(u);
        password = new SimpleStringProperty(p);
        degree = new SimpleObjectProperty<Degree>(d);
        friends = new SimpleListProperty<Student>(null);
    }
    //GETTERS AND SETTERS
    ...
}
```

4.2.2 Declaration of Subject class

```
public class Subject {

    private final SimpleIntegerProperty id;
    private final SimpleStringProperty name;
    private final SimpleIntegerProperty credits;
    private final SimpleStringProperty info;
    private final SimpleIntegerProperty degree;
    private final SimpleListProperty<Professor> professors;

    // CONSTRUCTOR
    public Subject(int i, String n, int c, String inf, int d) {
        id = new SimpleIntegerProperty(i);
        name = new SimpleStringProperty(n);
        credits = new SimpleIntegerProperty(c);
        info = new SimpleStringProperty(inf);
        degree = new SimpleIntegerProperty(d);
        professors = new SimpleListProperty<Professor>(null);
}
...
}
```

4.2.3 Declaration of Professor class

```
public class Professor{
    private final SimpleIntegerProperty id;
    private final SimpleStringProperty name;
    private final SimpleStringProperty surname;

    // CONSTRUCTOR
    public Professor(int i, String n, String s) {
        name = new SimpleStringProperty(n);
        surname = new SimpleStringProperty(s);
        id = new SimpleIntegerProperty(i);
    }
...
}
```

4.2.4 Declaration of Degree class

```
public class Degree {
    private final SimpleIntegerProperty id;
    private final SimpleStringProperty name;

    public Degree(int i, String n) {
        id = new SimpleIntegerProperty(i);
        name = new SimpleStringProperty(n);
    }
    ...
}
```

4.2.5 Declaration of Comment class

```
public class Comment{

    private final SimpleIntegerProperty id;
    private final SimpleStringProperty text;
    private final SimpleIntegerProperty student;
    private final SimpleStringProperty date;
    private final SimpleIntegerProperty subject;

    public Comment(int i, String t, int s, int sub, String d) {
        id = new SimpleIntegerProperty(i);
        text = new SimpleStringProperty(t);
        student = new SimpleIntegerProperty(s);
        date = new SimpleStringProperty(d);
        subject = new SimpleIntegerProperty(sub);
    }
...
}
```

4.2.6 Database manager

The other important class in the project is the DbManager. It handles the connection with Neo4j and provides all methods to interact with it.

```
  public class DbManager implements AutoCloseable {
       private final Driver driver;
       private final String uri = "bolt://localhost:7687";
       private final String user = "user";
       private final String password = "pwd";

       public DbManager() {
          driver = GraphDatabase.driver(uri, AuthTokens.basic(user, password));
       }

    // ... crud operations ...

       @Override
       public void close() throws Exception {
          driver.close();
       }
  }
```

4.3 CRUD operations

All crud operations are implemented using the **Cypher query language**; it is a declarative graph query language that allows for expressive and efficient querying and updating of a property graph.

4.3.1 Create

This method of the DbManager class guarantees the creation of a comment. A new vertex is inserted on the graph and is connected with a student node using an incoming edge called WROTE. Moreover it is connected with subject node, using an outgoing edge called ABOUT.

```
public void createComment(String text, Student student, int subjectId) {
   try(Session session = driver.session()){
       session.writeTransaction( tx -> {
           tx.run( "MATCH (e:Student) WHERE id(e) = $idStudent " +
                   "MATCH (s:Subject) WHERE id(s) = $idSubject " +
                   "CREATE " +
                   "(a:Comment {text: $text, date: $date})," +
                   "(e)-[:WROTE]->(a)," +
                   "(a)-[:ABOUT]->(s);",
             Values.parameters("idStudent",student.getId(),"idSubject",subjectId,"text",text,
                   "date",new SimpleDateFormat("yyyy/MM/dd HH:mm:ss").format(new Date())) );
           return null;
       });
   }
}
```

4.3.2 Read

This method reads from the database all the degrees and inserts them in a list.

```
public List<Degree> getDegreeCourses() {
   try(Session session = driver.session()){
      return session.readTransaction( tx -> {
         List<Degree> list = new ArrayList<>();
         StatementResult sr = tx.run( "MATCH (dd:Degree) RETURN ID(dd), dd.name;");

         while(sr.hasNext()) {
            Record r = sr.next();
            list.add(new Degree(r.get("ID(dd)").asInt(),r.get("dd.name").asString()));
         }
         return list;
      });
   }
}
```

4.3.3 Update

The following snippet of code shows how to execute an updating operation of an entity professor.

```
  public void updateProfessor(int profId, String name, String surname) {
     try(Session session = driver.session()){
        session.writeTransaction( tx -> {
           tx.run("MATCH (p:Professor) WHERE ID(p) = $profId\n" +
                 "SET p.name = $name, p.surname = $surname;",
             Values.parameters("profId",profId,"name",name,"surname",surname) );
           return null;
        });
     }
  }
```

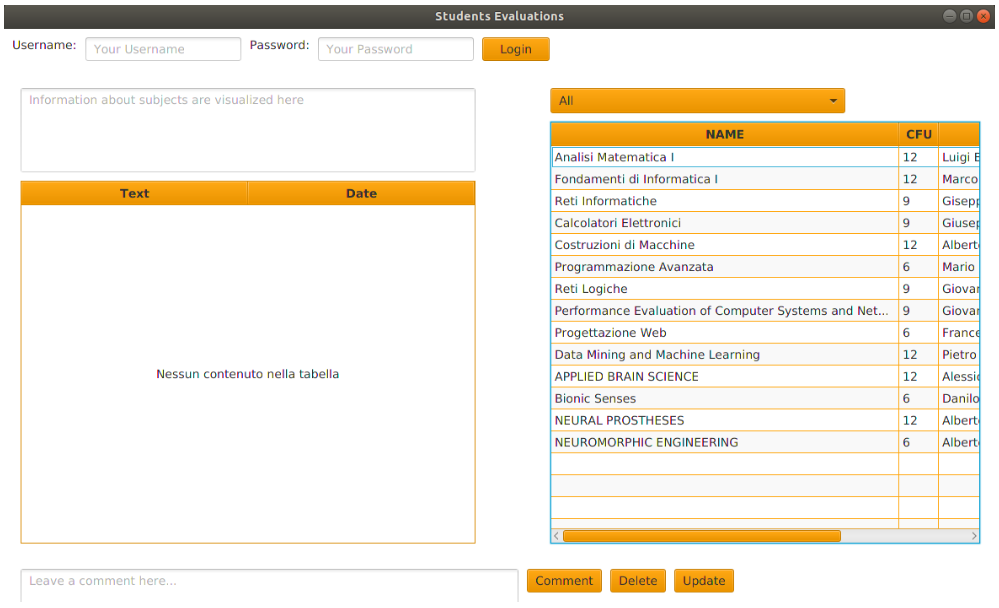
4.3.4 Delete

This example of deleting operation shows how to remove a subject, given its id, and the related relations.

```
public void deleteSubject(int subjectId) {
   try(Session session = driver.session()){
      session.writeTransaction( tx -> {
         tx.run( "MATCH (n:Subject) WHERE id(n) = $idSubject DETACH DELETE n;",
           Values.parameters("idSubject",subjectId));
         return null;
      });
   }
}
```

5 User Manual

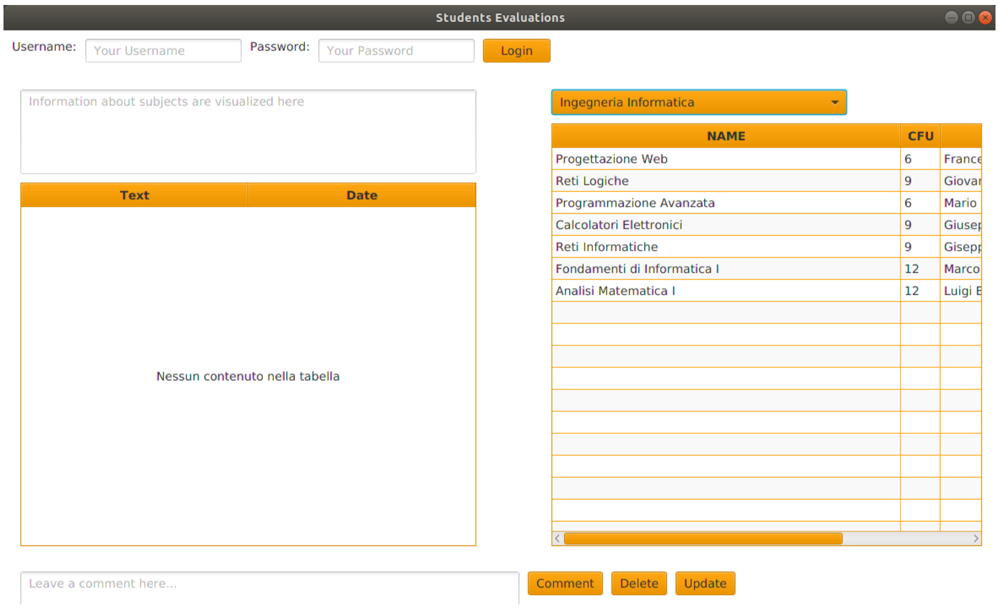
When you first run the application, the interface you get is the one in figure 6.



NAME	CFU	
Analisi Matematica I	12	Luigi E
Fondamenti di Informatica I	12	Marco
Reti Informatiche	9	Gisepp
Calcolatori Elettronici	9	Giusep
Costruzioni di Macchine	12	Albert
Programmazione Avanzata	6	Mario
Reti Logiche	9	Giovar
Performance Evaluation of Computer Systems and Net...	9	Giovar
Progettazione Web	6	France
Data Mining and Machine Learning	12	Pietro
APPLIED BRAIN SCIENCE	12	Alessi
Bionic Senses	6	Danilo
NEURAL PROSTHESES	12	Albert
NEUROMORPHIC ENGINEERING	6	Albert

Figure 6: First view of the application

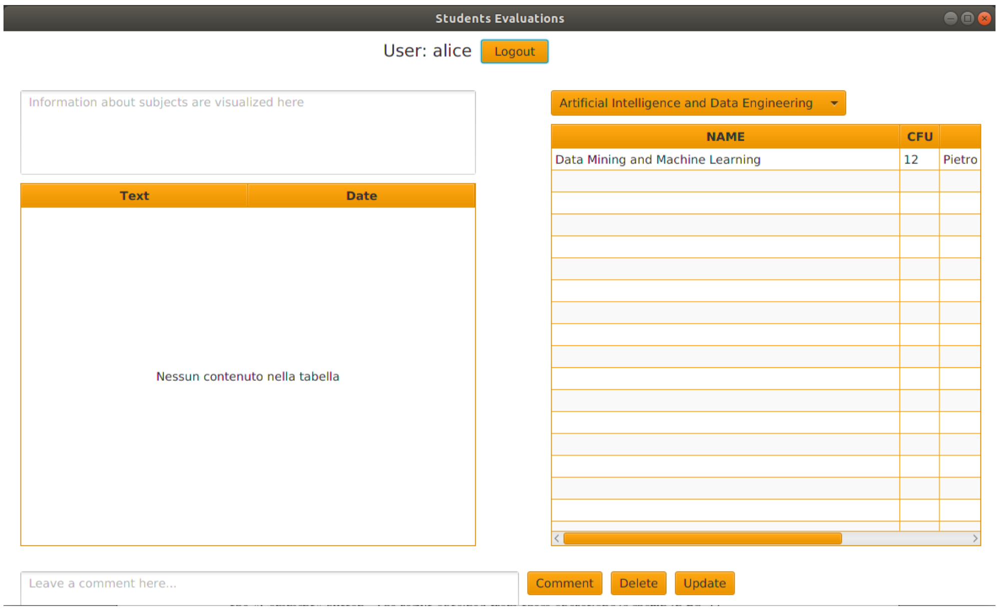
The default display includes the list of all registered subjects in the table on the right. You can also choose to display the subjects of a single degree course, using the drop-down menu on the right (fig. 7).



NAME	CFU	
Progettazione Web	6	France
Reti Logiche	9	Giovar
Programmazione Avanzata	6	Mario
Calcolatori Elettronici	9	Giusep
Reti Informatiche	9	Gisepp
Fondamenti di Informatica I	12	Marco
Analisi Matematica I	12	Luigi E

Figure 7: Selection of subjects filtered by the "Ingegneria Informatica" degree course

If you have a registered account, you can log in to the application, so that the comments' operations aren't blocked. Enter your username and your password in the suited fields at the top and click on "Login" (fig. 8).


Students Evaluations
User: alice
Logout
Information about subjects are visualized here
Artificial Intelligence and Data Engineering
Comment
Delete
Update
Leave a comment here...

Text	Date
Nessun contenuto nella tabella	

NAME	CFU	
Data Mining and Machine Learning	12	Pietro

Figure 8: Application interface after the user "Alice" has logged in

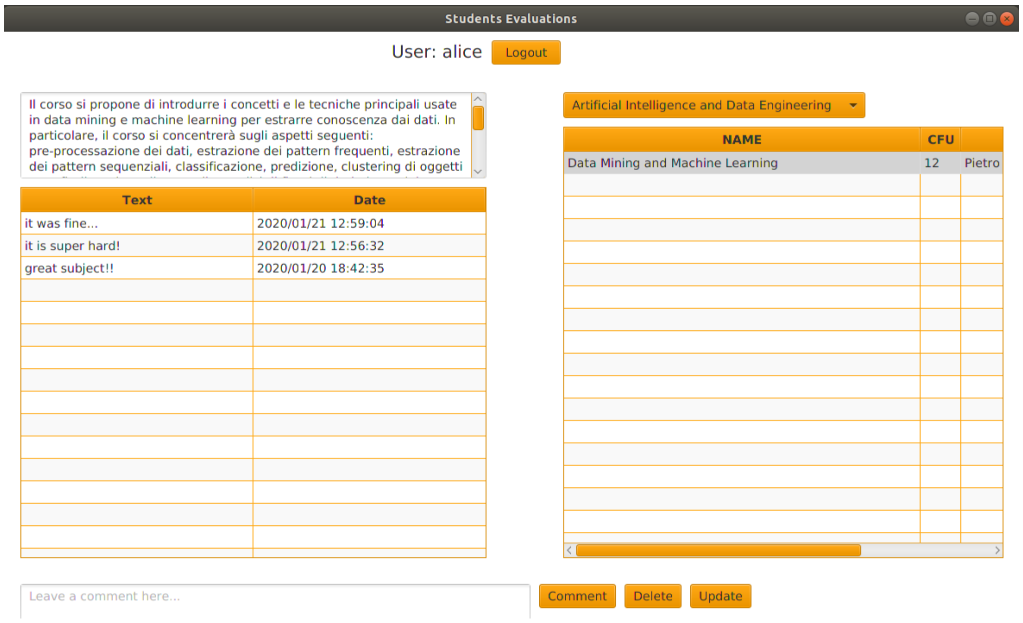
If you now want to be able to see the comments associated with a particular subject, you have to click on the name of the subject: in the table on the left the list of comments already received will appear (fig. 9). With this operation, you'll be able to visualize also the information related to that particular subject.

To leave a comment, you need to enter the text in the field below the table and then click on the "Comment" button. The result obtained from these operations is shown in fig. 10.

You can also decide to modify the comment you just uploaded or another comment you made on a previous session. To do so, you need to click on the comment you want to update, change the text in the field below the table and then click on the "Update" button (fig. 11). Finally you have the chance to delete your comment, by clicking on "Delete" after selecting it. Notice that you can modify or delete just the comments that you made.

To log out, just click on the appropriate button at the top, next to the user label.

Moreover, if you don't have a registered username, you can still browse through the application, search for subjects' information and read all comments. You are just unable to leave or change any comments.



Students Evaluations

User: alice Logout

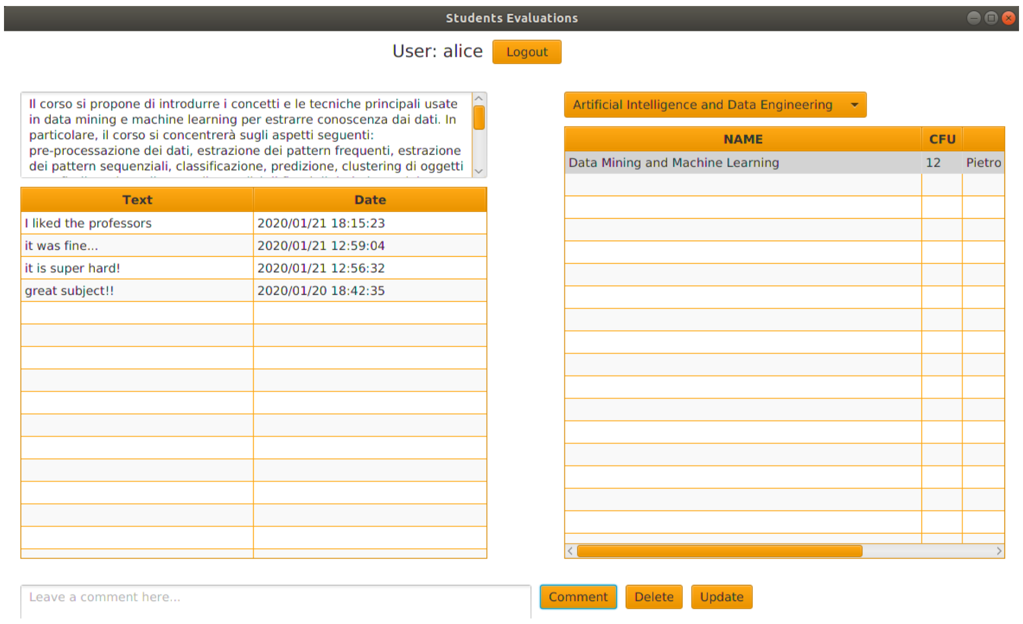
Il corso si propone di introdurre i concetti e le tecniche principali usate in data mining e machine learning per estrarre conoscenza dai dati. In particolare, il corso si concentrerà sugli aspetti seguenti: pre-processazione dei dati, estrazione dei pattern frequenti, estrazione dei pattern sequenziali, classificazione, predizione, clustering di oggetti

Text	Date
it was fine...	2020/01/21 12:59:04
it is super hard!	2020/01/21 12:56:32
great subject!!	2020/01/20 18:42:35

Artificial Intelligence and Data Engineering

NAME	CFU	
Data Mining and Machine Learning	12	Pietro

Figure 9: Displaying the comments related to a subject



Students Evaluations

User: alice Logout

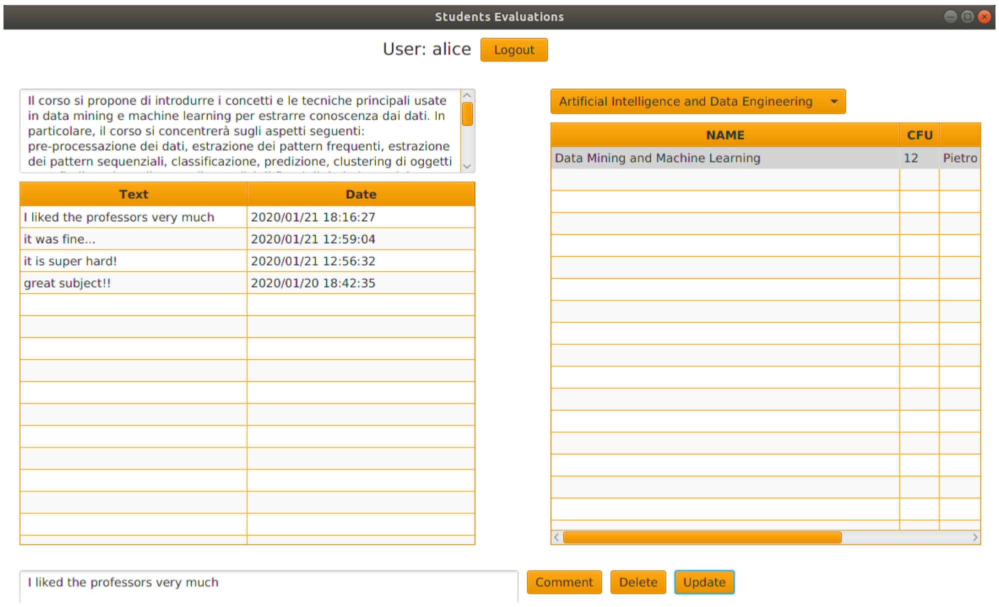
Il corso si propone di introdurre i concetti e le tecniche principali usate in data mining e machine learning per estrarre conoscenza dai dati. In particolare, il corso si concentrerà sugli aspetti seguenti: pre-processazione dei dati, estrazione dei pattern frequenti, estrazione dei pattern sequenziali, classificazione, predizione, clustering di oggetti

Text	Date
I liked the professors	2020/01/21 18:15:23
it was fine...	2020/01/21 12:59:04
it is super hard!	2020/01/21 12:56:32
great subject!!	2020/01/20 18:42:35

Artificial Intelligence and Data Engineering

NAME	CFU	
Data Mining and Machine Learning	12	Pietro

Figure 10: Interface after adding a comment



Il corso si propone di introdurre i concetti e le tecniche principali usate in data mining e machine learning per estrarre conoscenza dai dati. In particolare, il corso si concentrerà sugli aspetti seguenti: pre-processazione dei dati, estrazione dei pattern frequenti, estrazione dei pattern sequenziali, classificazione, predizione, clustering di oggetti

Text	Date
I liked the professors very much	2020/01/21 18:16:27
it was fine...	2020/01/21 12:59:04
it is super hard!	2020/01/21 12:56:32
great subject!!	2020/01/20 18:42:35

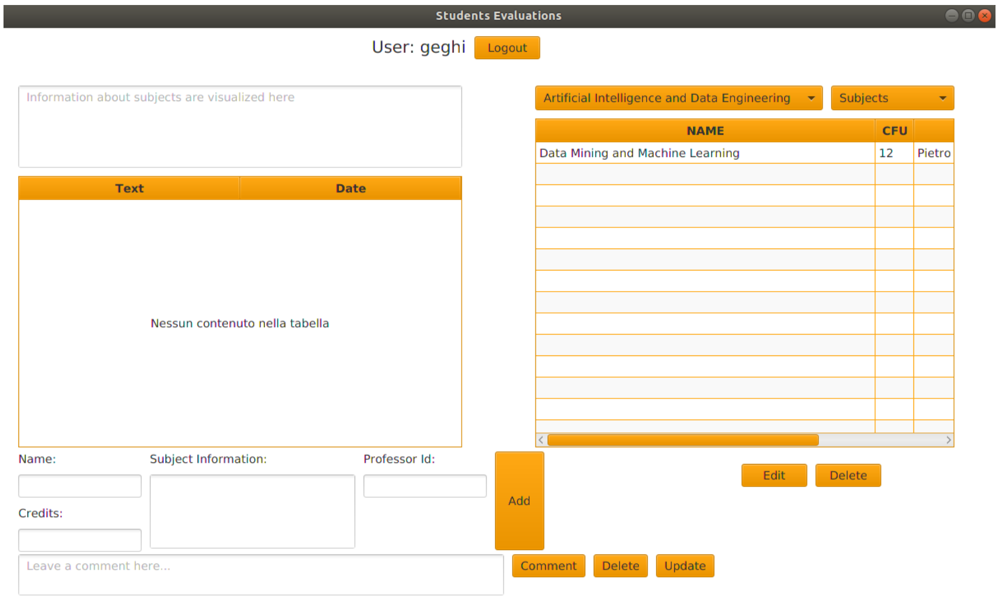
NAME	CFU	
Data Mining and Machine Learning	12	Pietro

I liked the professors very much

Figure 11: Interface after updating a comment

5.1 Admin Manual

If you have an admin user, you are entitled to make changes both on the professors' and the subjects' lists. You need to log in inserting your username and password, and the application will recognize you as the administrator and show up the buttons for modifying the data (fig. 12).

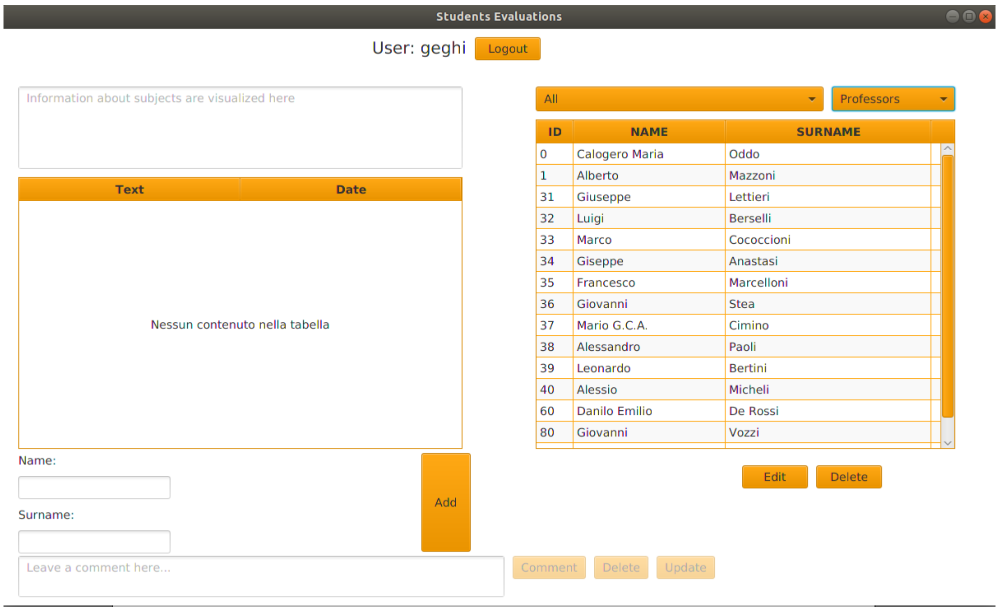


Text	Date
Nessun contenuto nella tabella	

NAME	CFU	
Data Mining and Machine Learning	12	Pietro

Figure 12: Interface after the administrator has logged in

As an administrator, there's one extra table available, that you can view by choosing on the choice box appeared on the right (fig. 13). So you can see all the professors on the database and also adding new ones or updating the ones already in there.

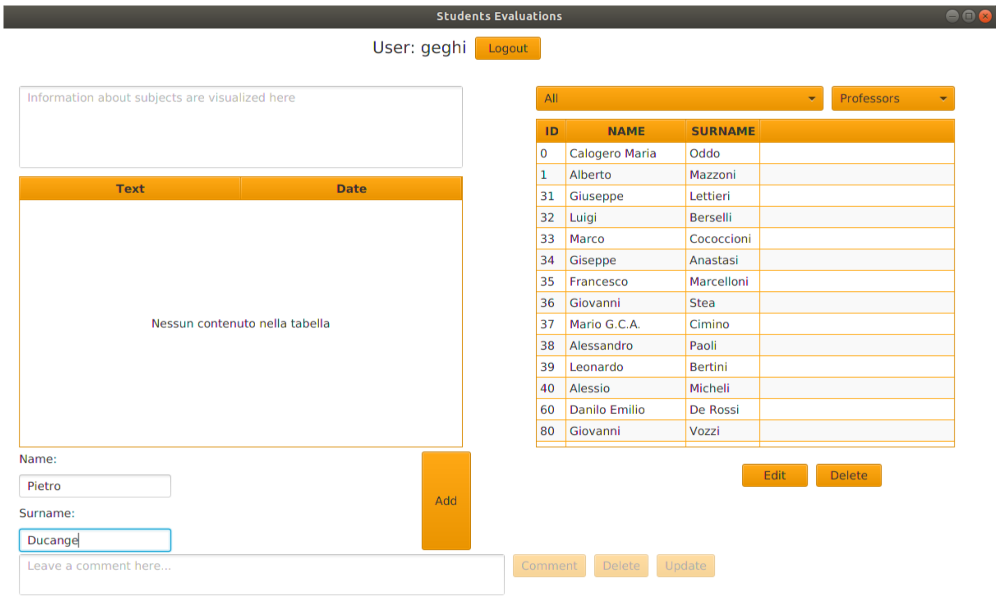


Text	Date
Nessun contenuto nella tabella	

ID	NAME	SURNAME
0	Calogero Maria	Oddo
1	Alberto	Mazzoni
31	Giuseppe	Lettieri
32	Luigi	Berselli
33	Marco	Cococcioni
34	Giseppe	Anastasi
35	Francesco	Marcelloni
36	Giovanni	Stea
37	Mario G.C.A.	Cimino
38	Alessandro	Paoli
39	Leonardo	Bertini
40	Alessio	Micheli
60	Danilo Emilio	De Rossi
80	Giovanni	Vozzi

Figure 13: Displaying the professors' table

You can choose to add a new professor, using the input fields at the bottom left. You have to specify the name and the surname, then press the "Add" button (fig. 14).



Text	Date
Nessun contenuto nella tabella	

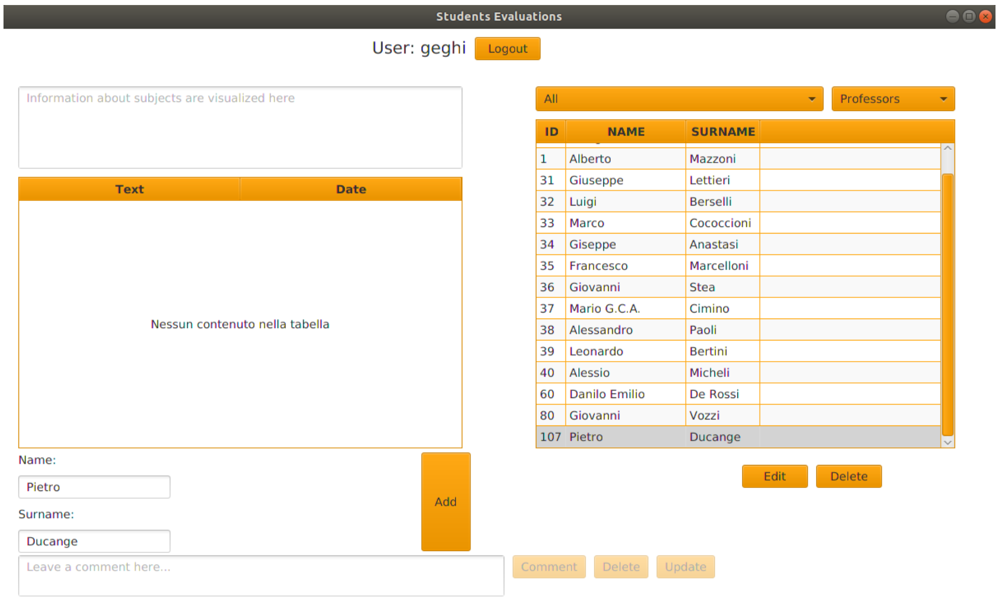
Name: Pietro

Surname: Ducange

ID	NAME	SURNAME
0	Calogero Maria	Oddo
1	Alberto	Mazzoni
31	Giuseppe	Lettieri
32	Luigi	Berselli
33	Marco	Cococcioni
34	Giseppe	Anastasi
35	Francesco	Marcelloni
36	Giovanni	Stea
37	Mario G.C.A.	Cimino
38	Alessandro	Paoli
39	Leonardo	Bertini
40	Alessio	Micheli
60	Danilo Emilio	De Rossi
80	Giovanni	Vozzi

Figure 14: Adding a new professor

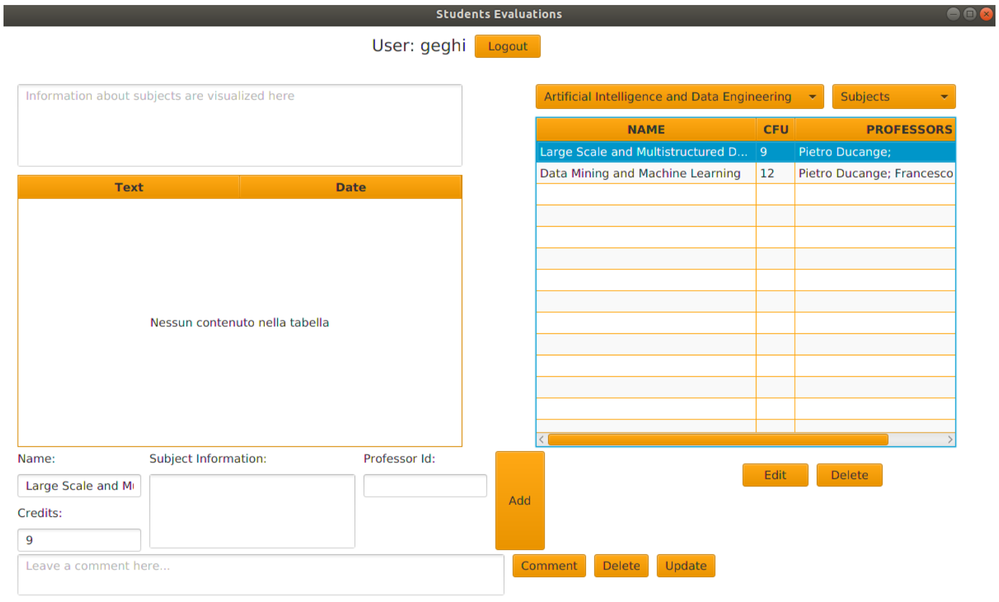
You can also modify the data related to a professor: click on the professor you are interested in and change the information shown in the apposite input fields. Finally, you have the chance to delete a professor by clicking on the "Delete" button after selecting the wanted professor (fig. 15).



ID	NAME	SURNAME
1	Alberto	Mazzoni
31	Giuseppe	Lettieri
32	Luigi	Berselli
33	Marco	Cococcioni
34	Giseppe	Anastasi
35	Francesco	Marcelloni
36	Giovanni	Stea
37	Mario G.C.A.	Cimino
38	Alessandro	Paoli
39	Leonardo	Bertini
40	Alessio	Micheli
60	Danilo Emilio	De Rossi
80	Giovanni	Vozzi
107	Pietro	Ducange

Figure 15: Screen of the application's interface from which you can either update or delete a professor

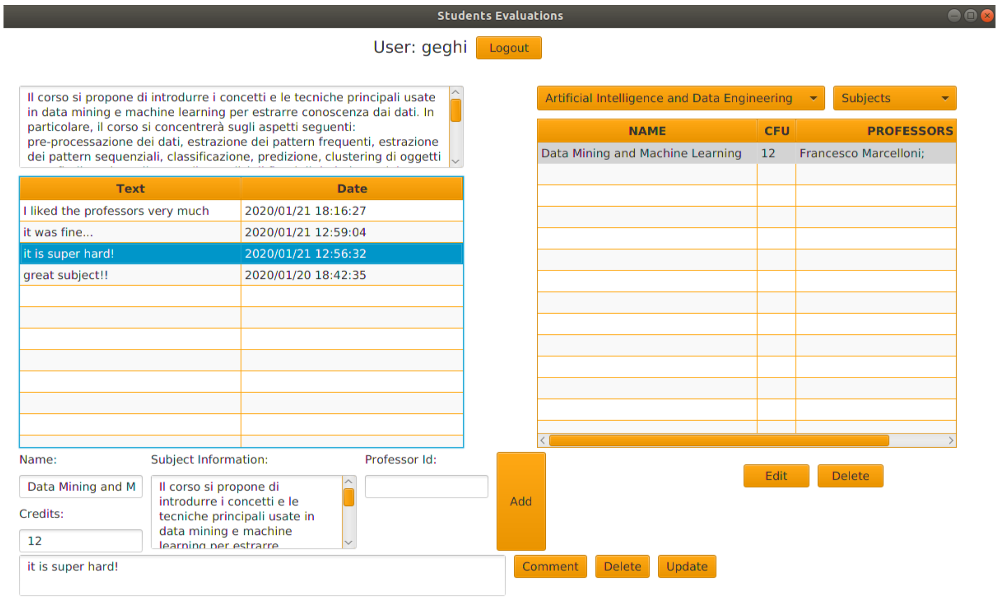
All these operations are available for the subjects as well. The only difference is that when you want to add a new subject you also need to specify the id of the professor teaching it (or a list of ids, separeted by commas, if there are more professors teaching it). Moreover, you must have precisely displayed in the table the subjects of the same degree course of the new one (fig. 16).



NAME	CFU	PROFESSORS
Large Scale and Multistructured D...	9	Pietro Ducange;
Data Mining and Machine Learning	12	Pietro Ducange; Francesco

Figure 16: Interface after adding a new subject, ready to modify or delete it

The administrator can delete comments posted by all the users, too. Just click on the comment and then on the "Delete" button (fig. 17).



Il corso si propone di introdurre i concetti e le tecniche principali usate in data mining e machine learning per estrarre conoscenza dai dati. In particolare, il corso si concentrerà sugli aspetti seguenti: pre-processazione dei dati, estrazione dei pattern frequenti, estrazione dei pattern sequenziali, classificazione, predizione, clustering di oggetti

Text	Date
I liked the professors very much	2020/01/21 18:16:27
it was fine...	2020/01/21 12:59:04
it is super hard!	2020/01/21 12:56:32
great subject!!	2020/01/20 18:42:35

NAME	CFU	PROFESSORS
Data Mining and Machine Learning	12	Francesco Marcelloni;

Name: Data Mining and M

Credits: 12

Subject Information: Il corso si propone di introdurre i concetti e le tecniche principali usate in data mining e machine learning per estrarre

Professor Id:

it is super hard!

Figure 17: Screen of the application's interface from which the admin can delete a comment