For our class diagram, we have Users (professors and students) who plays a central role. They can vote for a resource (that is not theirs). They add stars (Call: resource.add_star()). When selecting the number of stars they give to a resource, a questionnaire will appear (Call: resource.show_question () showing all questions related to this resource). Then a user has to answer the questions simply by clicking on a answer proposal. Finally he submits his questionnaire and he will be shown average voting count of this answered questions (Call: resource.show_question_votes()).

For storing this in the database, we need some tables: a

- Question table containing question_id (for question identification), Question_statement (String representing the actual question), Restricted (Value giving the user permission to see the question).
- Answer table containing answer_id (for answer identification), Answer (String representing the answer).
- Questionnaire table containing questionnaire_id (for identification), question_id (for identifying question), answer_id (answer identification). This table is used to link questions and there possible answers.
- Rating table containing resource_id, question_id, answer_id and user_id to link an answer by a user to a question in a resource.

Here is who the tables should look like:

Rating Table:

resource_id (int)	question_id (int)	answer_id (int)	user_id (int)
0	2	1	3246
0	1	1	3246
0	3	2	3246
24	1	2	3246
24	2	1	9999

User 3246 has answered the question id 1,2,3 for resource id 0 with answer id 1,1,2
User 3246 has answered the question 1 for resource 24 with answer id 2
User 9999 has answered the question 2 of resource 24 with answer id 1
! id_resource,id_question,id_user is UNIQUE ! => a user can only give 1 answer to a question for a resource.

Question Table

id_question(int)	Question_statement (String)	Restricted (0 for all users, 1 for teacher, 2 for students)
0	Etoiles	0
1	Le contenu est-il approprié	1
2	blablabal	1
3	Avez vous compris la resource ?	1

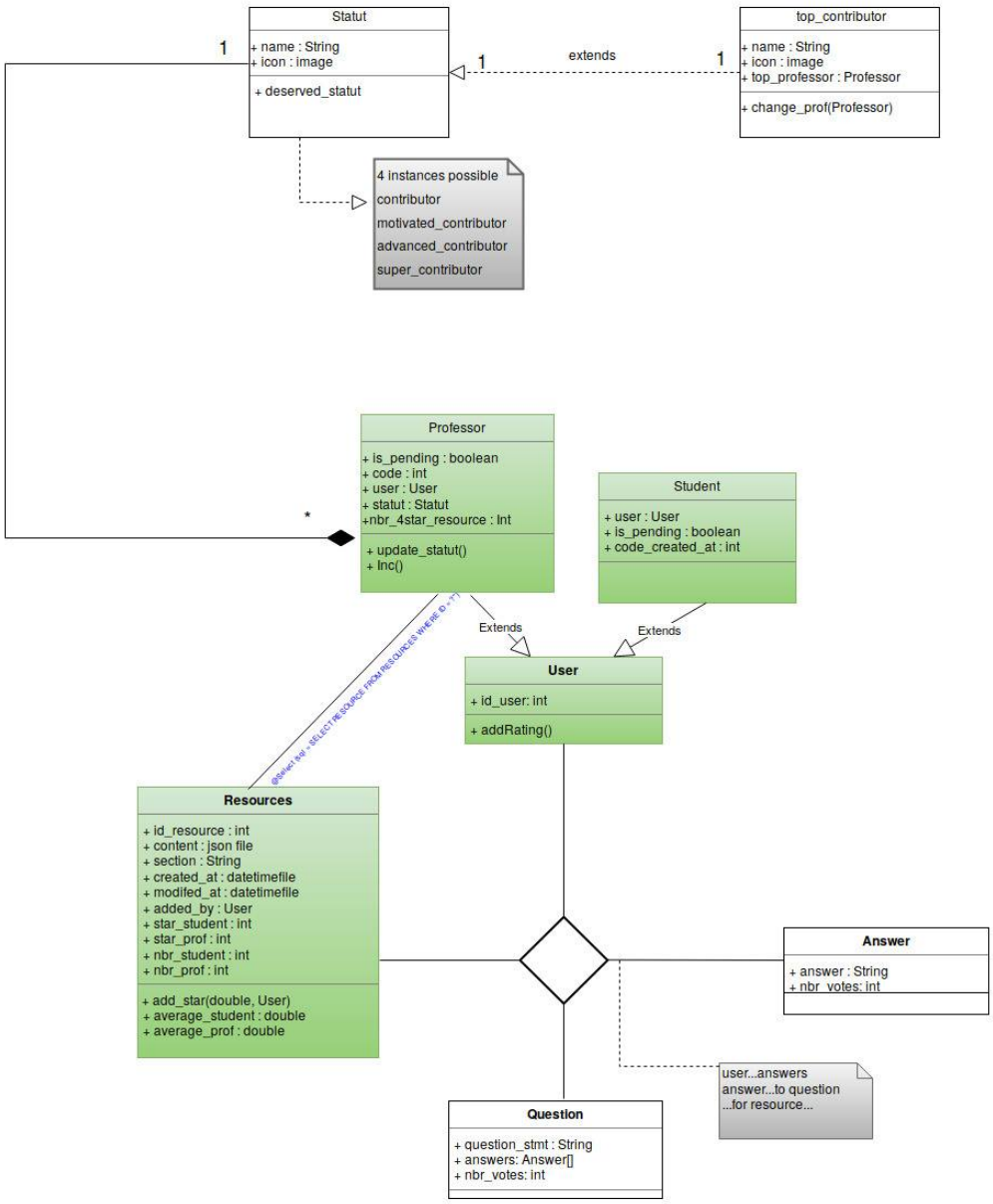
Answer Table

answer_id	Answer (String)
0	Un peu
1	Pas du tout
2	Beaucoup
3	Suffisant

Questionnaire Table

question_id	answer_id
0	1
0	0
0	2
1	2
1	1

The other part of the class diagram is the status part with links a status to a User more specific a teacher. The top contributor is an extension of a simple status that can only have 1 teacher (It is unique for the whole system)



Class diagram

Integration

To integrate our module in the existing application, we will need to add some attributes and methods to the existing python classes Resource, Student, User and Professor (the complete list of things to add can be found in the class diagram). We will also need to define new python classes: Answer, Question, Status and top_contributor and their corresponding attributes and methods as defined in the class diagram. For the database, we will need to add two of tables: one linking resources and the user who added it and one linking the user rating the resource, the resource, the question and it's answer, one linking each question/answer with it's statement and number of votes. We will also need to add some attributes in the existing tables, which correspond to the fields added

in the existing classes. For the first prototype, we are going to concentrate ourselves on the implementation of the status and vote systems; we plan on having a prototype where a user can vote for a resource and when the necessary number of resources with 4 stars is reached, the status of the corresponding professor should change. For the second prototype, we plan to have the questionnaire running; when a user votes for a resource, a questionnaire pops up and asks up to 10 questions about the resource. When the user has finished with the questionnaire, the application shows the statistics of the answers by all users (e.g. 60% of the users voted "Très bien" to "Avez-vous bien compris la resource ?", 30% voted "Pas du tout" and 10% voted "Bien"). The planning isn't changed.