UNIVERSITÀ DEGLI STUDI DI PADOVA

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Web Applications, A.Y. 2017/2018
Master Degree in Computer Engineering
Master Degree in ICT for Internet and Multimedia

Homework 1 – Server Side Application Design

Deadline: May 18, 2018

LeoForFriendsWA	Project	
Bonotto	Matteo	1178948
Giovannini	Stefano	1178044
Kalakonda	Srikanth Reddy	1178232
Rossi	Gianmaria	1178526
Zhang	Guangzheng	1178043

Requirements Analysis

1. Objectives and Description of the System

This Web Application concerns the creation of a platform aimed to allow users to exchange information about job interviews. This is performed according to the usual format involving question submission by users and subsequent replies to these. In addition, users can also rate answers and questions; answers can also be commented.

2. Functional Requirements

The Web Application features should follow the subsequent points:

- Users can subscribe to the service by providing an email, a username and a password. Users can also provide additional information about themselves, including: name, surname, academic title and a picture.
- An email can be used to register a user just once.
- Users can just view contents if not logged in.
- Upon log in, users can post any number of different questions, answer to any number of question (at most one answer per question) and comment any number of times every answer.
- Logged users can also provide rating to questions and answer (from one up to five "stars") and also rate the complexity of questions only (Easy, Medium or Hard).
- Users also own an "user rating" computed as the average rating to all their questions and answers. System administrators can promote a user to a "certified" status after a certain number of highly rated answers (see answer rating in subsequent points)
- The number of views reached by every single questions or answer should be registered.

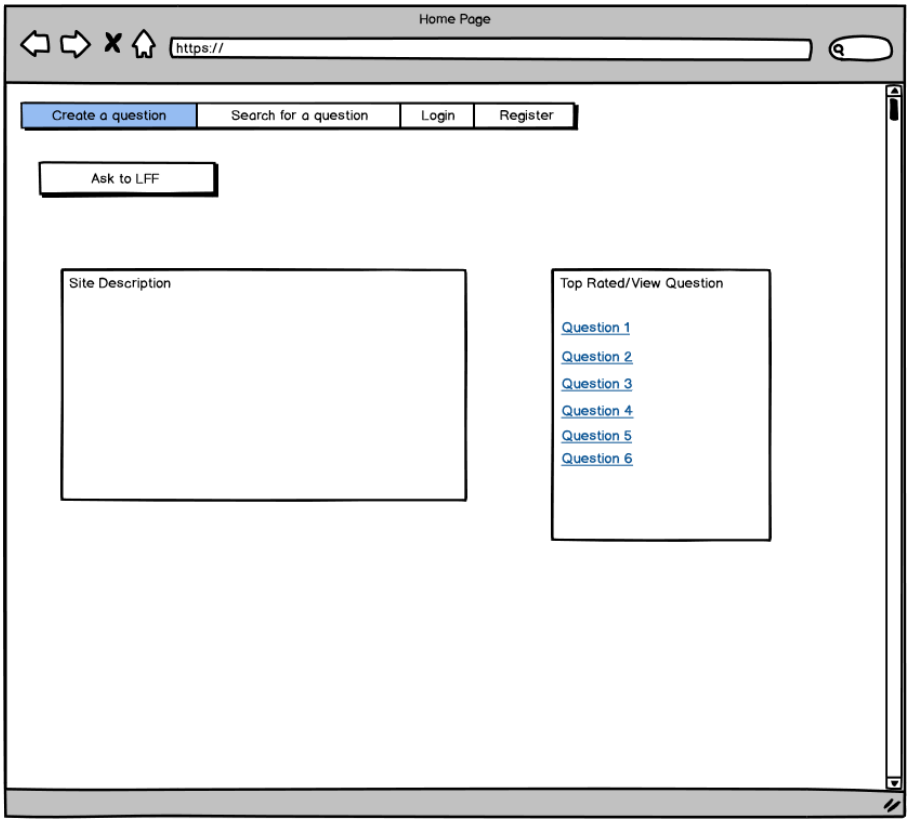
- Each question, each one having its title, has to belong to at least one topic. Topics concern the working ambit from which the question belongs and can be divided into sub-categories. For example, the Topic of ICT (General working framework) can have the subcategories of Web Applications, Databases, Computer Vision, Source Coding, etc.
- A question can also be about a specific company.
- In order to see the list of question created by himself/herself, the user need to go to his/her profile page.
- Users can search a question by title or content.
- Users can search for a topic and see how many questions that topic has, moreover it can select from a list a specific question that belong to a selected topic.
- Users can search for a company and see how many questions that company has, moreover it can select from a list a specific question that belong to a selected company.

Presentation Logic

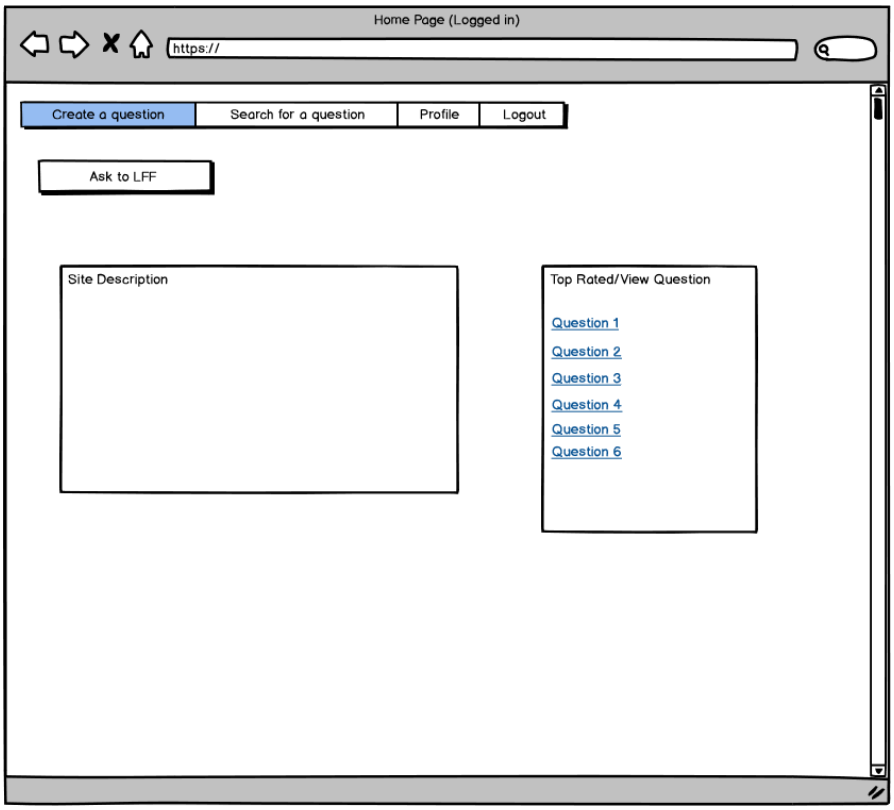
1. Mockup

The following series of images, each one preceded by a brief comment indicating the meaning of the content, represents the mockup design for the web application.

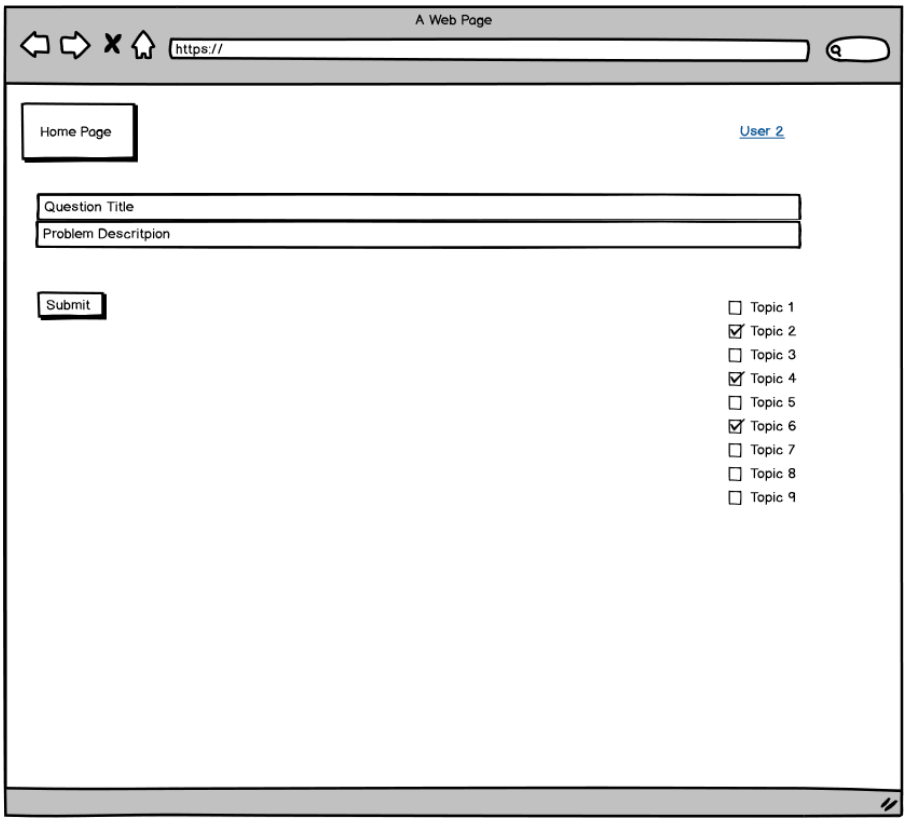
Home page.



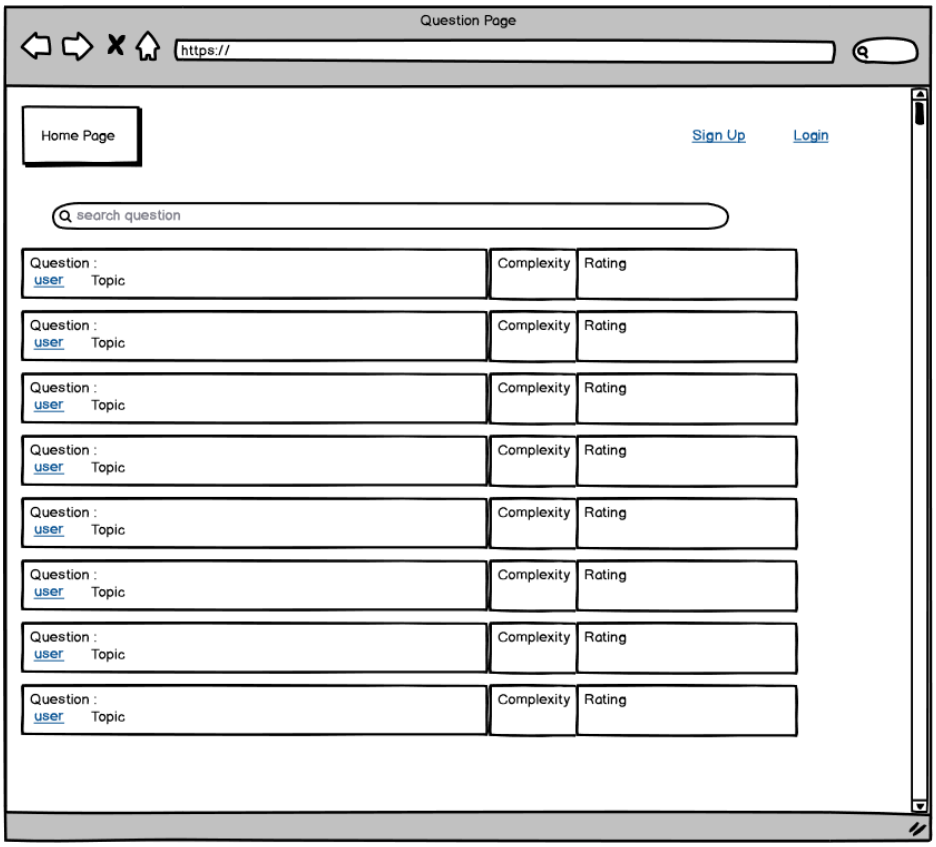
Home page after login.



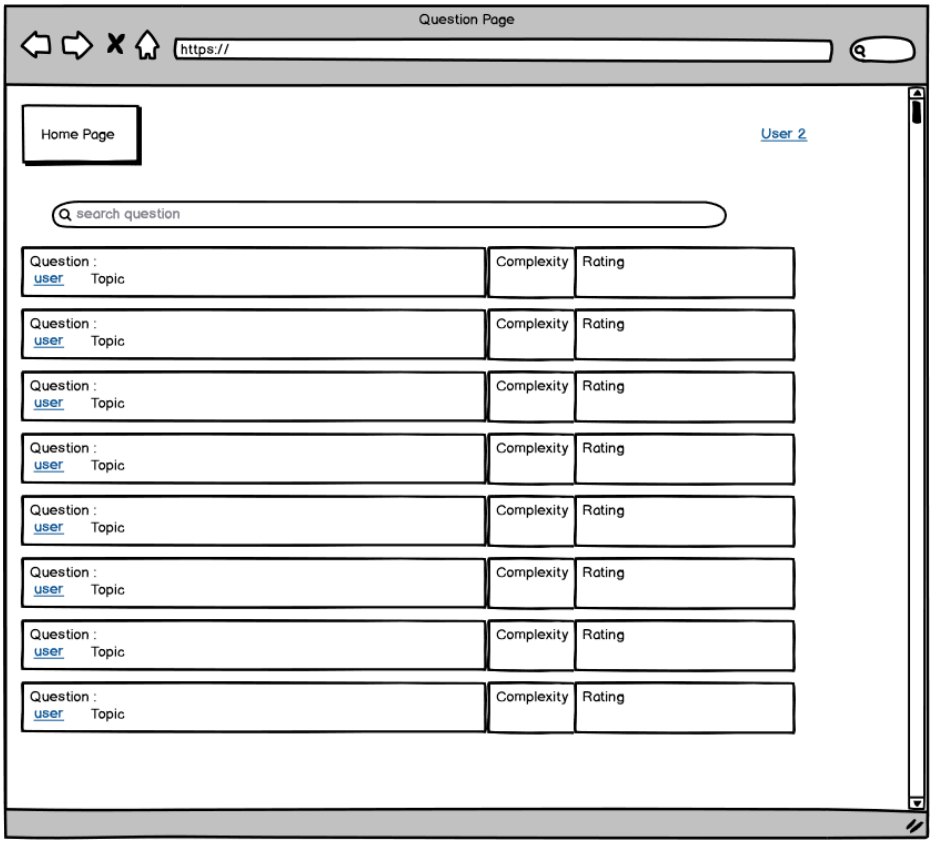
Question creation page.



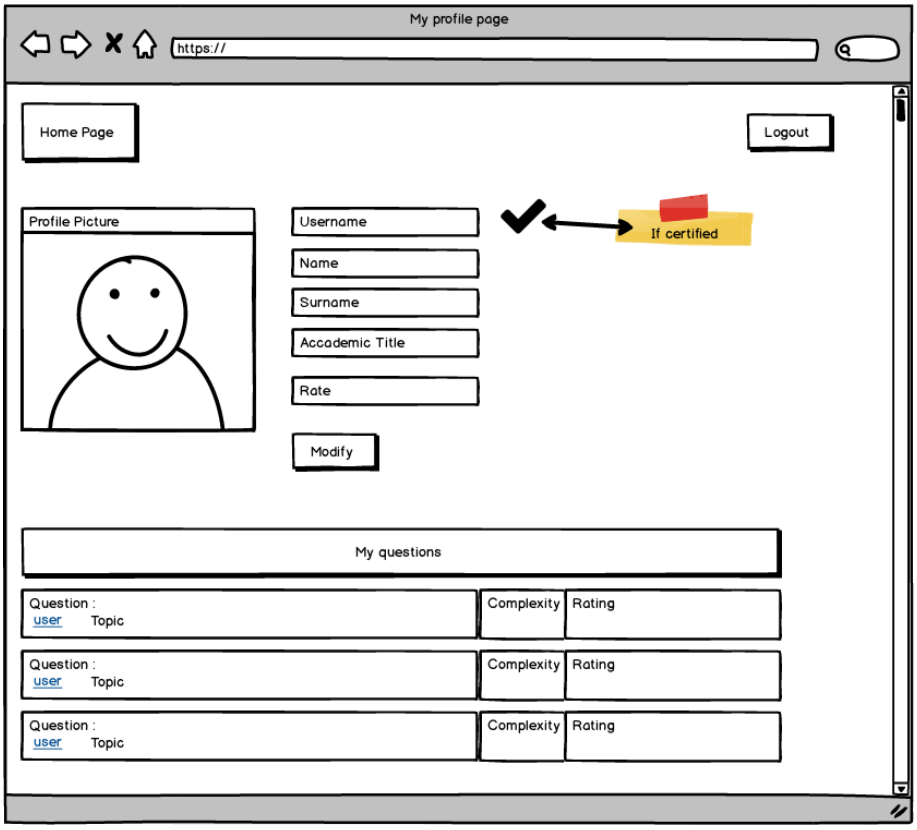
List of all the questions as seen by a non-authenticated user.



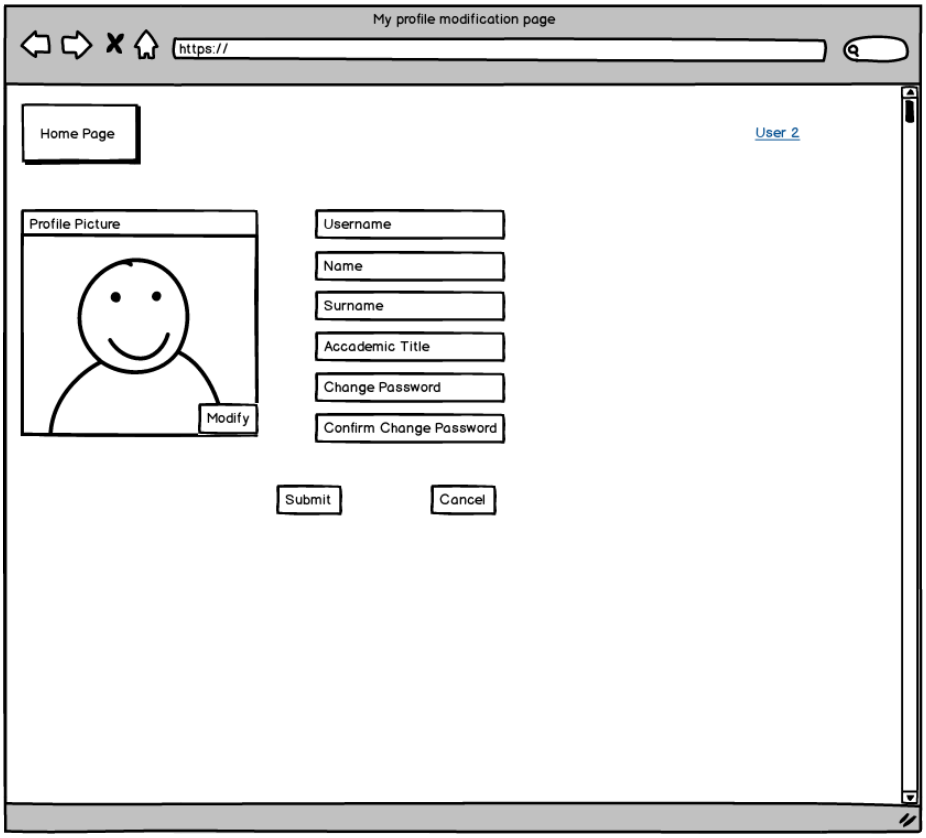
List of all the questions as seen by an authenticated user.



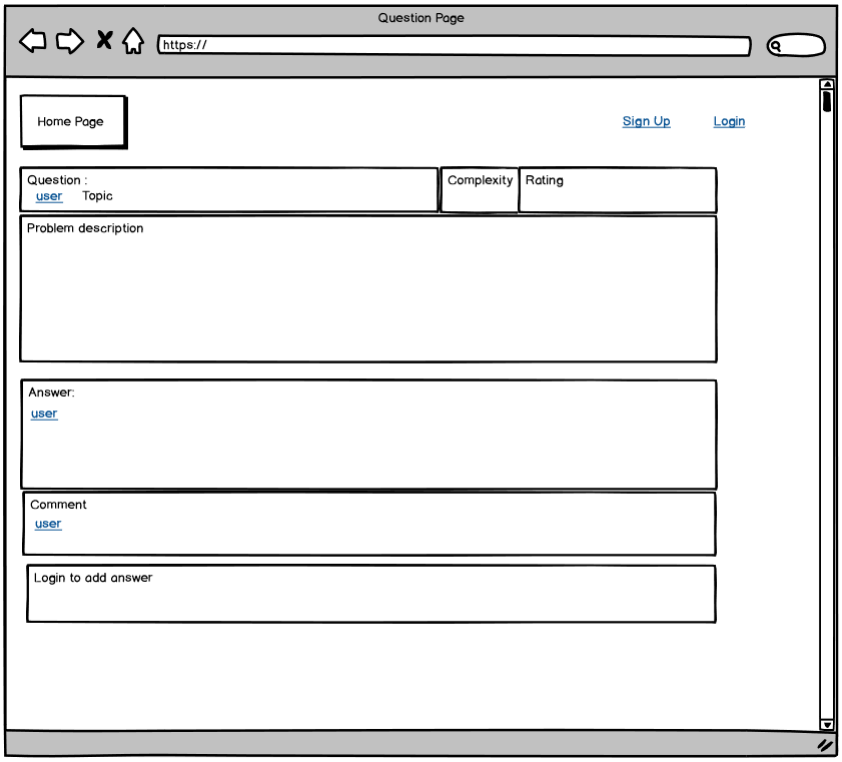
Profile of an user as seen upon authentication.



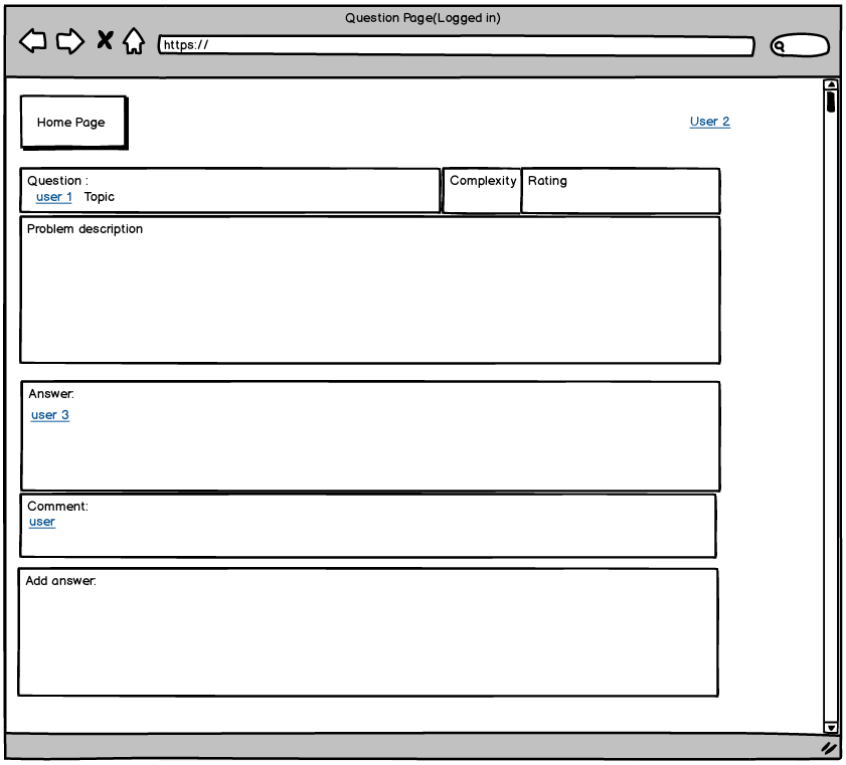
Modification page for an user profile.



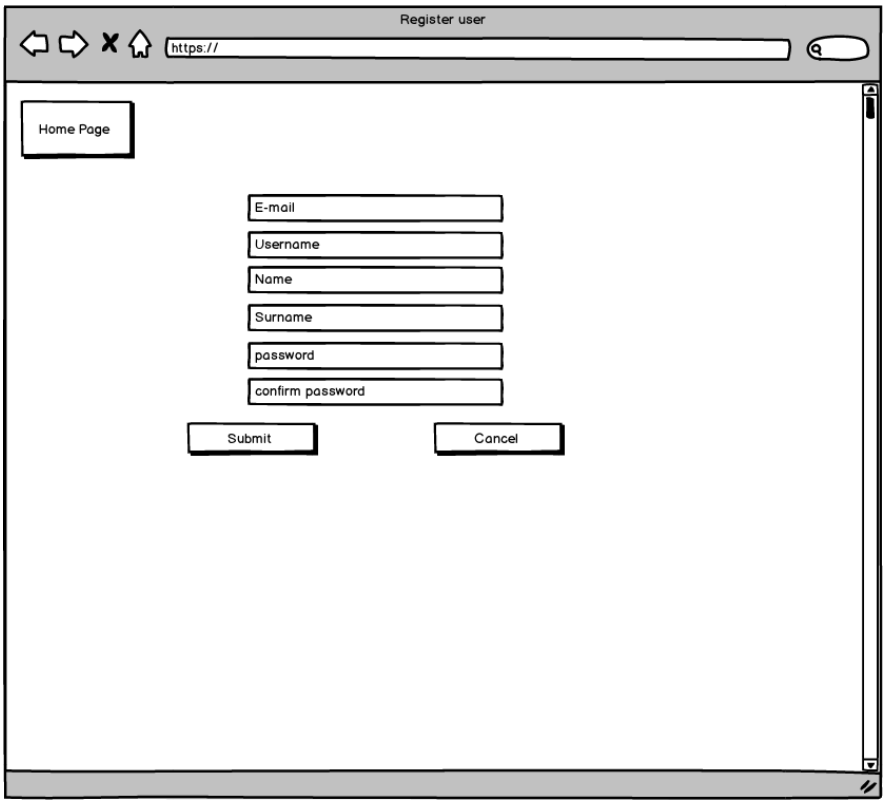
Question page as seen by a non-authenticated visitor.



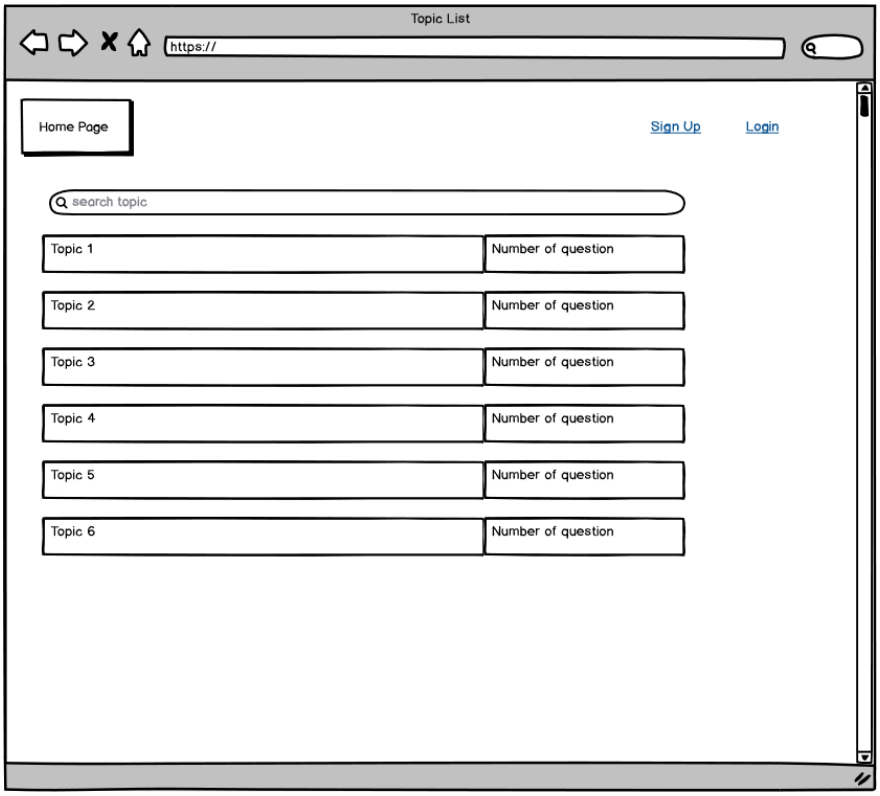
Question page as seen upon authentication.



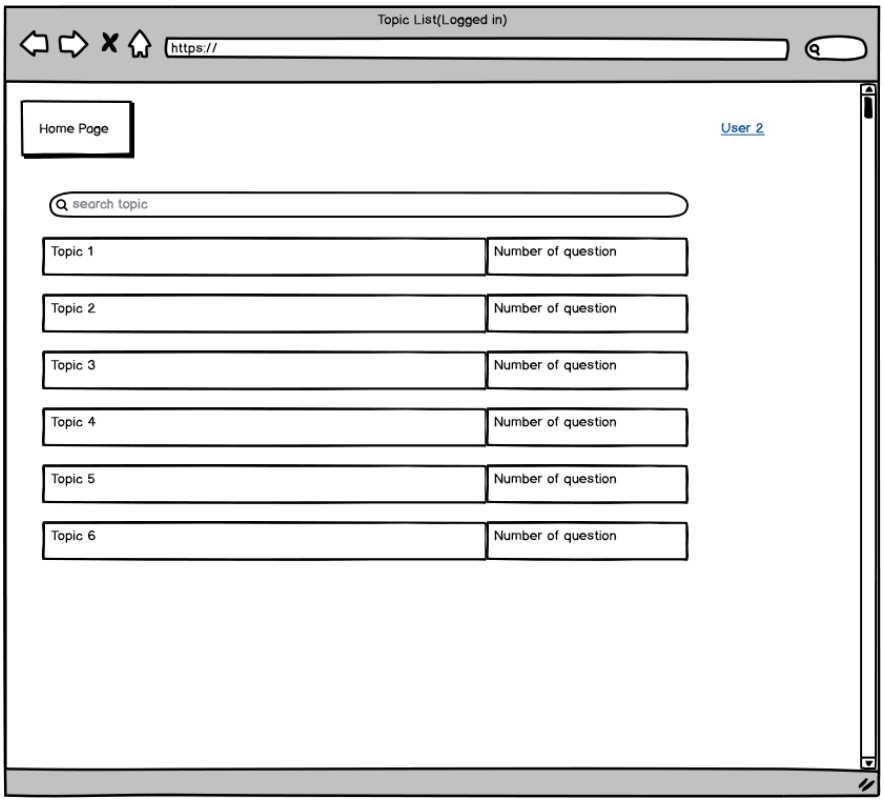
Registration page.



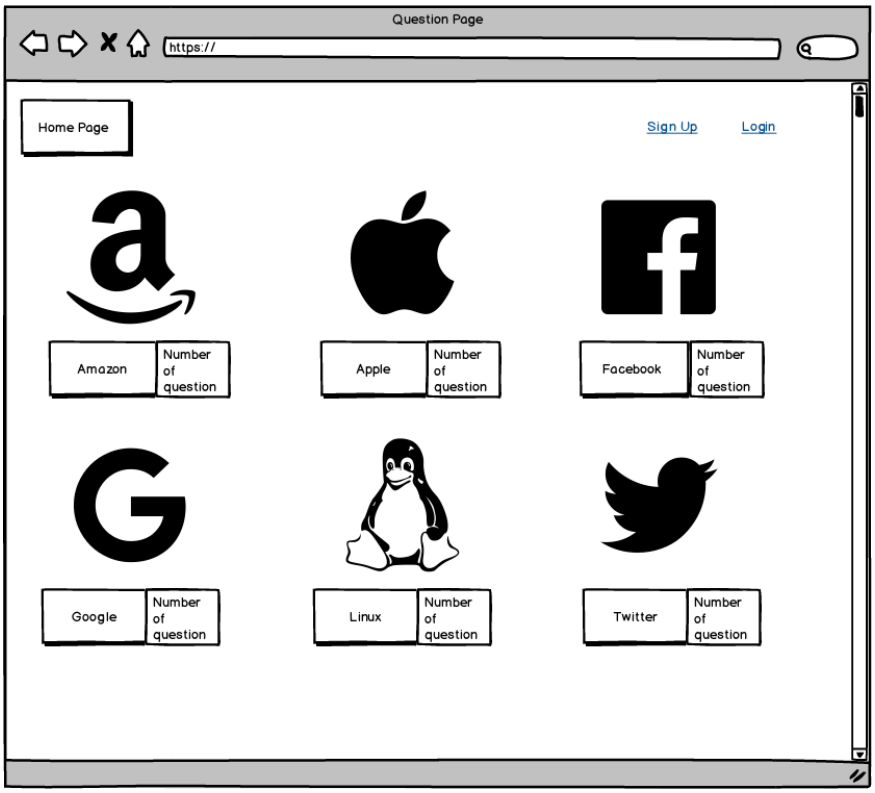
List of topics as seen by a non-authenticated visitor.



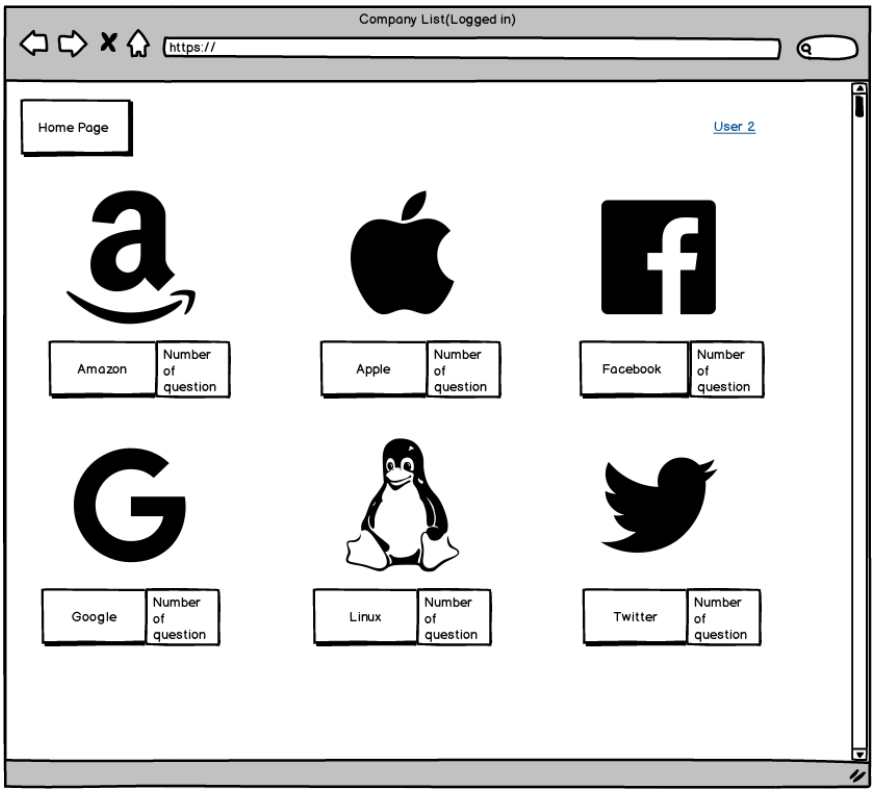
List of questions as seen upon authentication.



Page listing all the companies with alongside with their logos.

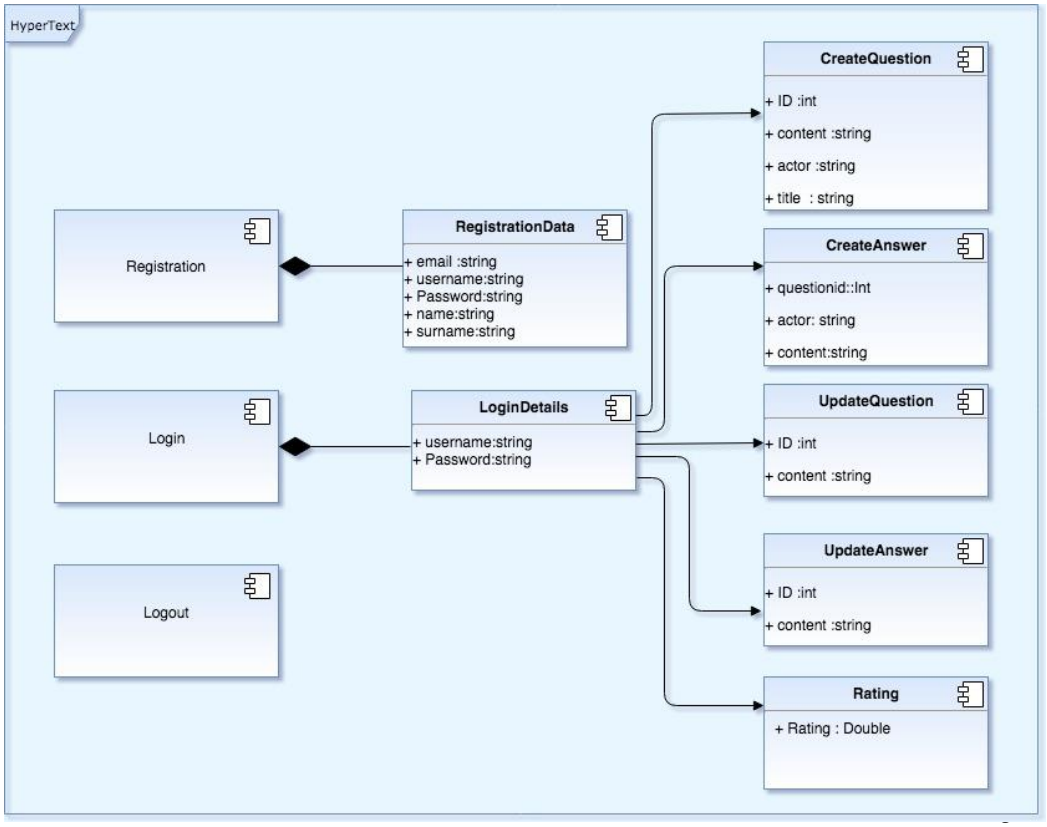


Page listing all the companies with alongside with their logos as seen upon login.



2. Hypertext Model

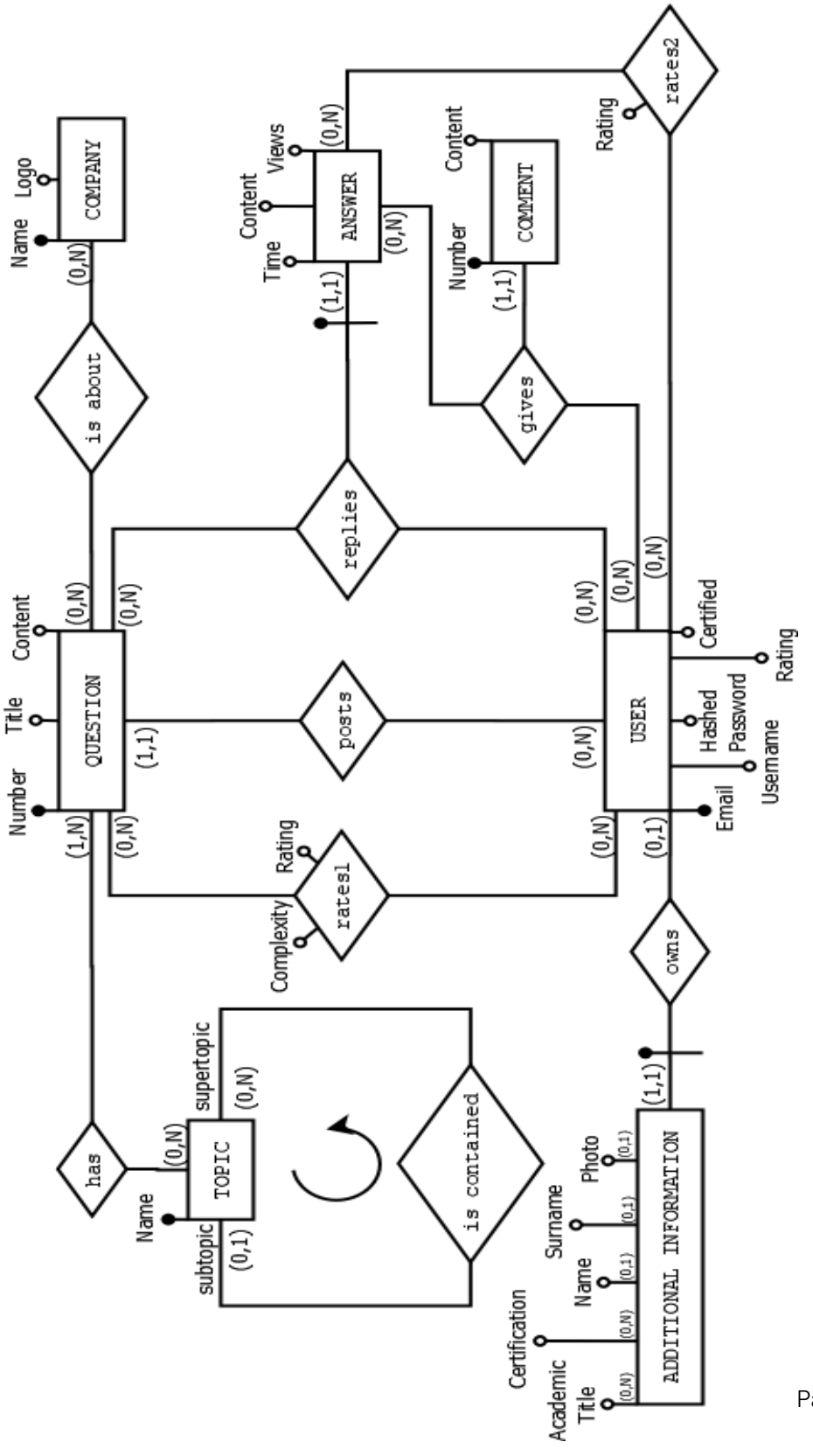
The following image provides a sample representation of the hypertext model for the web application. It illustrates the data to be inserted at the moment of registration and that, upon login, a user can create and update his questions and answers and provide ratings.



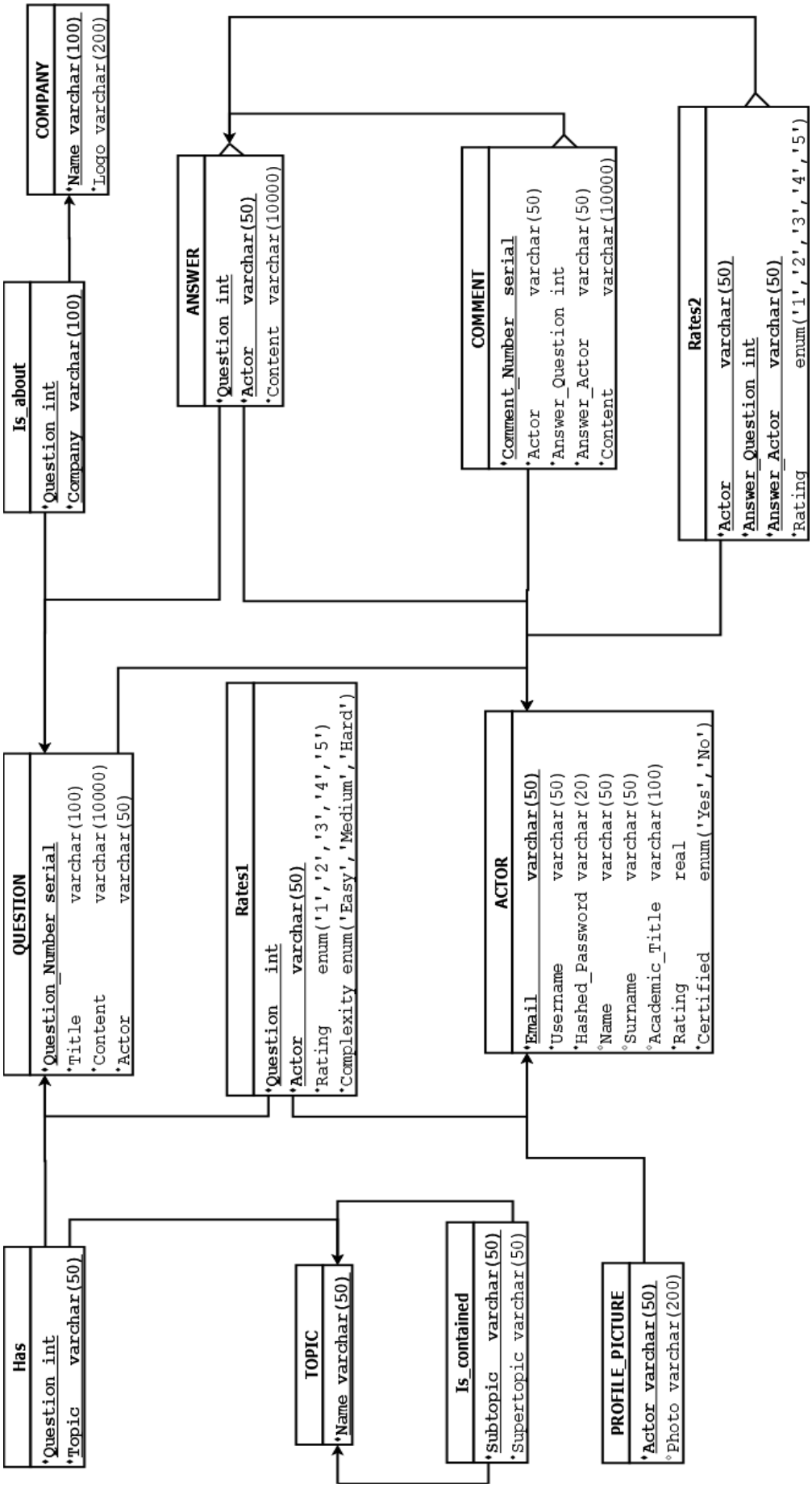
Data Logic

The two following sections contain the full ER schema for the Web Applications database and the corresponding mapping to Relational Schema.

1. Entity-Relationship Schema



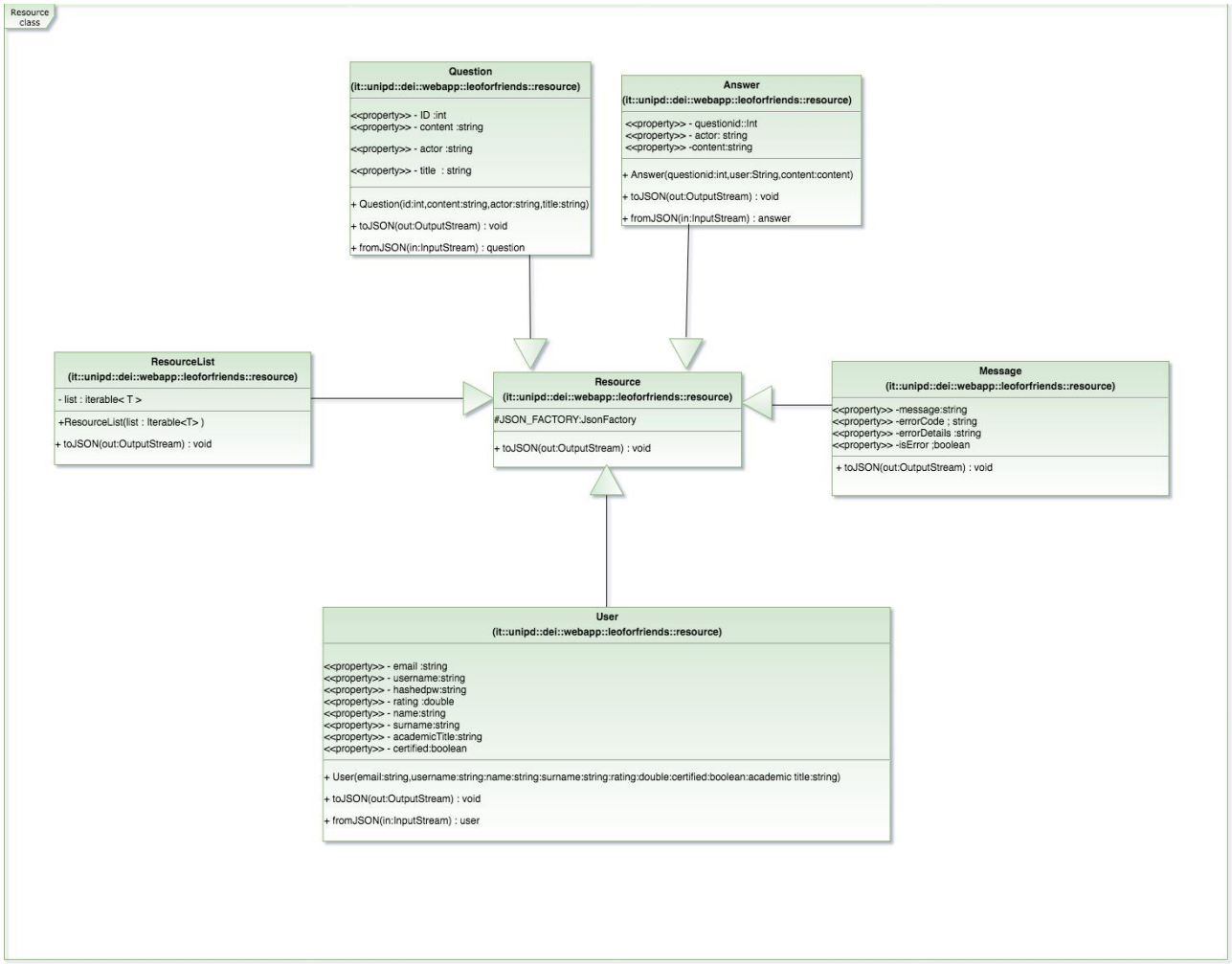
2. Relational Schema

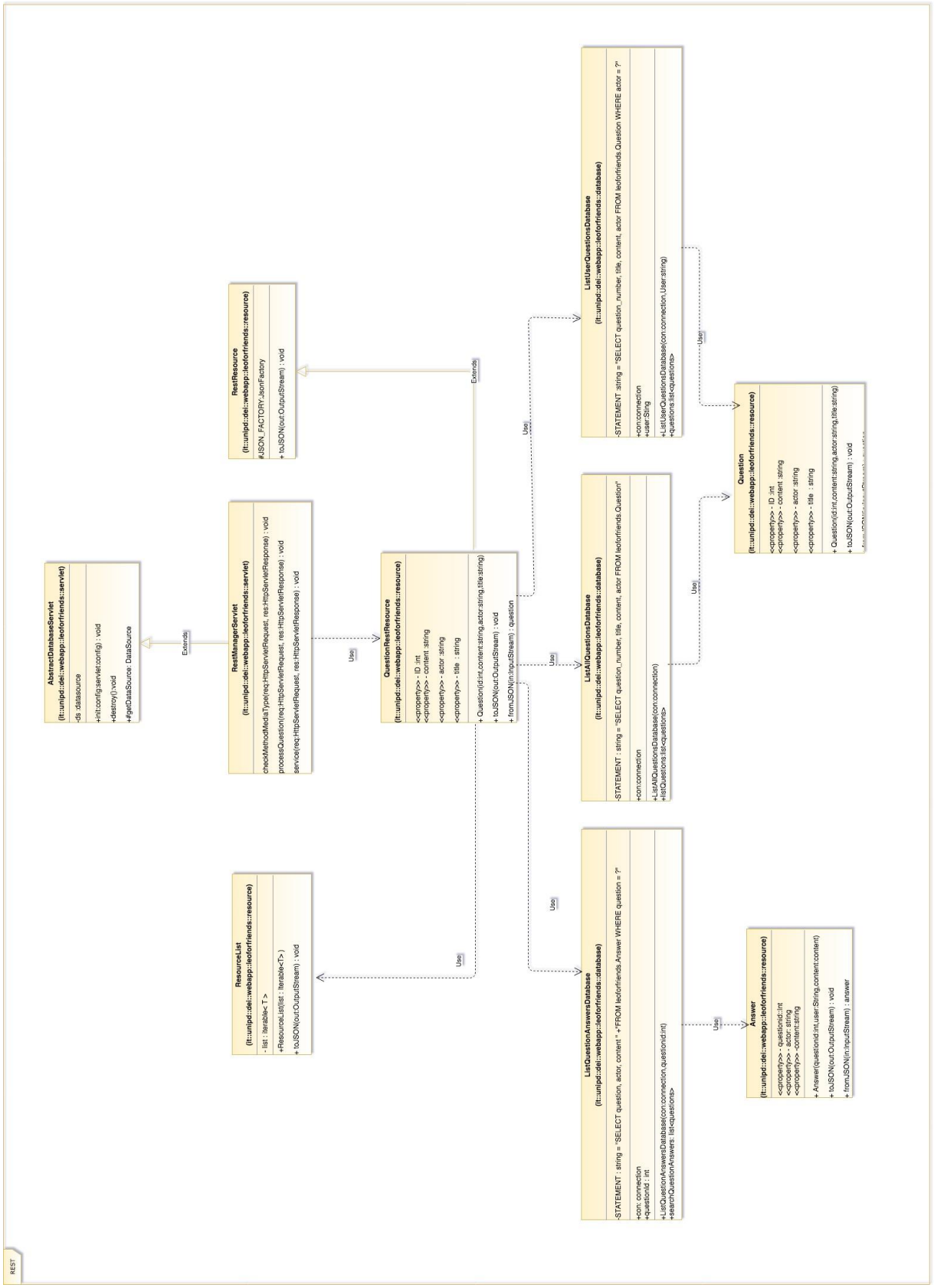


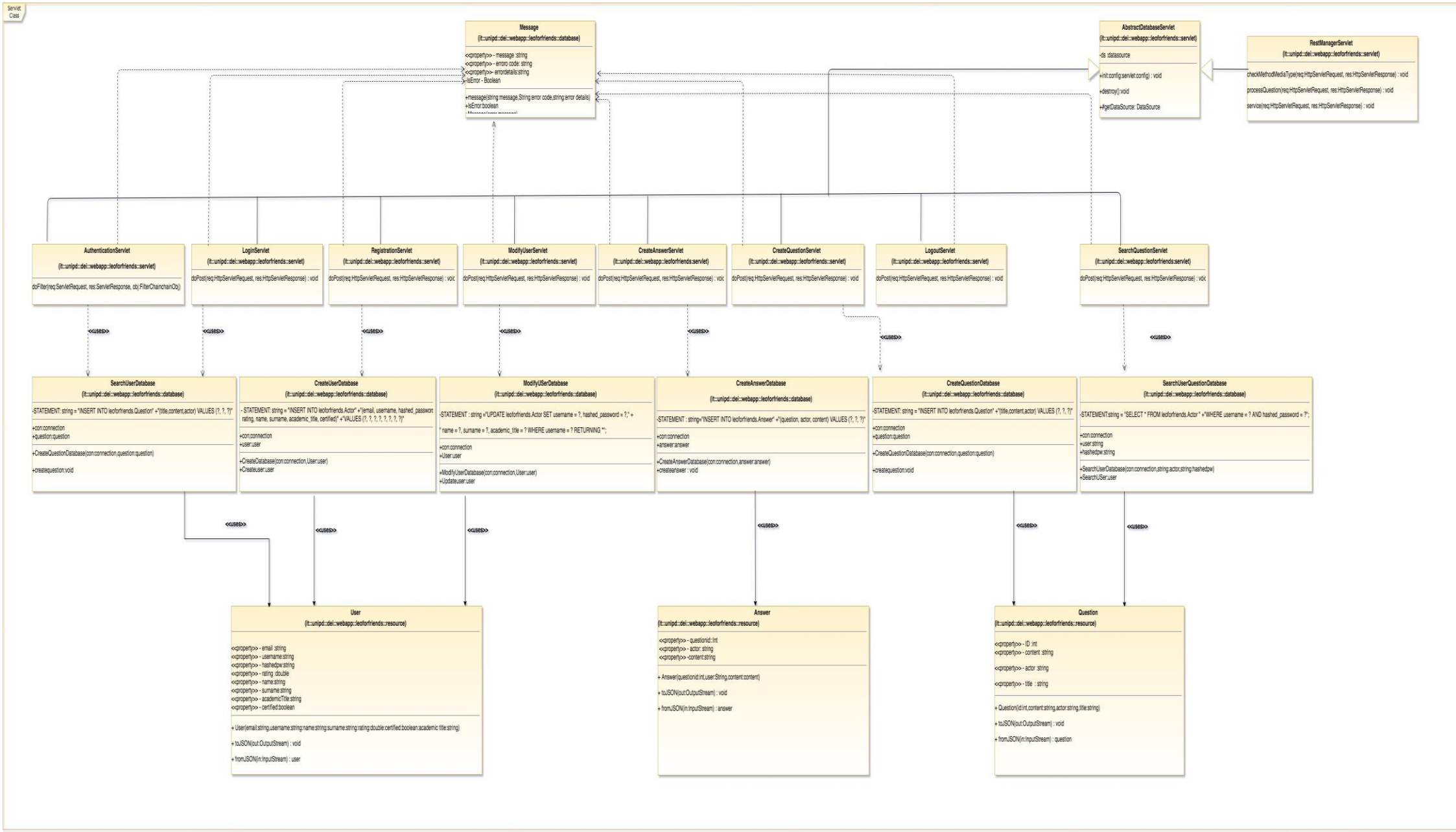
Business Logic

1. UML Class Schema

This section contains a list of images explaining the dependencies between the classes composing the web application.
The first one illustrates the dependencies of the resource (JavaBeans) part of the application: all the classes extend the Resource one in order to inherit JSON-related methods.
The second image represents the class dependencies of the classes used to expose REST services (better explained in the dedicated section at the end of this document).
Eventually, the last image illustrates how the class composing the servlet part (interfaced via JSPs): all the Servlet extend AbstractDatabaseServlet to manage the pool of connections to the DB and use a similarly-named DAO class to retrieve information from the Data Layer. These last classes make use of the Resources illustrated by the first image.

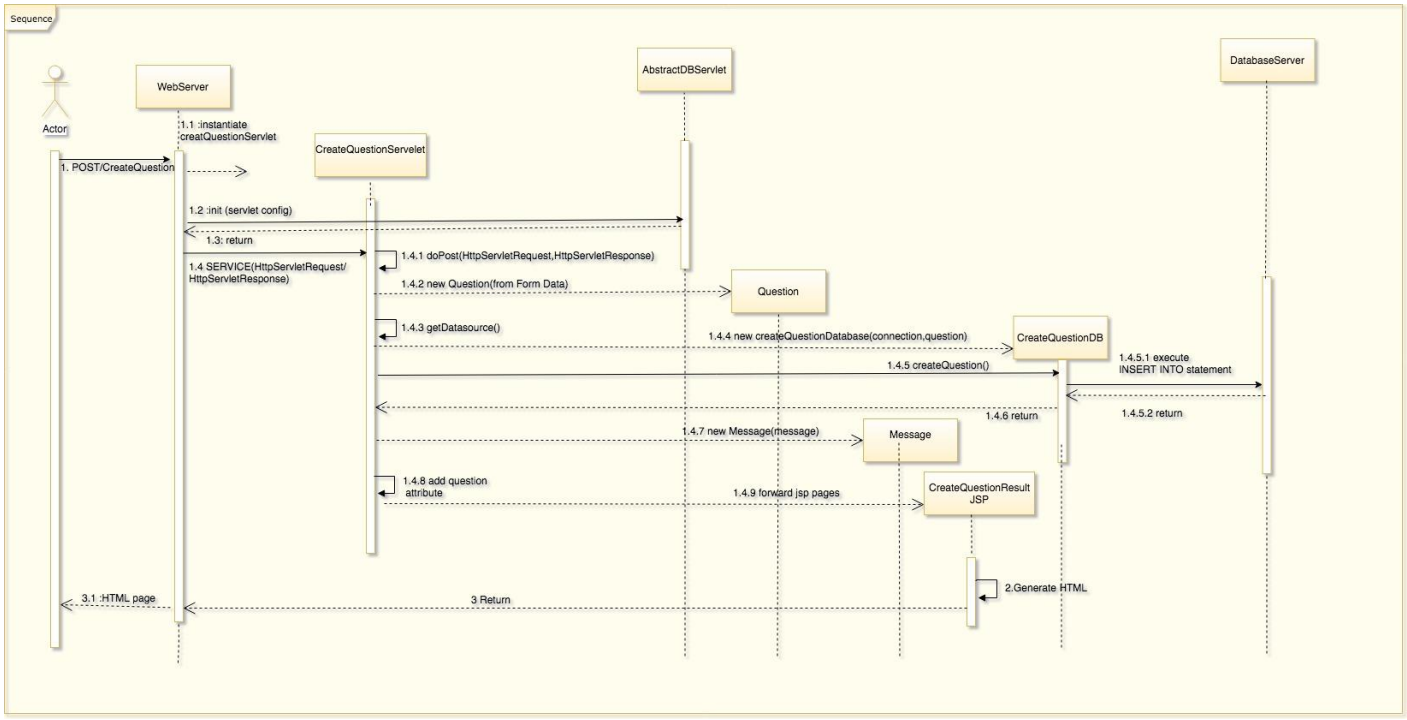



REST
AbstractDatabaseServlet
(It::unipd::dei::webapp::leofortriends::servlet)
-ds :datasource
+init:config:servlet:conf(g) : void
+destroy():void
+#getDataSource: DataSource
Extends
RestManagerServlet
(It::unipd::dei::webapp::leofortriends::servlet)
checkMethodMediaType(req:HttpServletRequest, res:HttpServletResponse) : void
processQuestion(req:HttpServletRequest, res:HttpServletResponse) : void
service(req:HttpServletRequest, res:HttpServletResponse) : void
RestResource
(It::unipd::dei::webapp::leofortriends::resource)
#JSON_FACTORY:JsonFactory
+ toJSON(out:OutputStream) : void
Extends
Use
QuestionRestResource
(It::unipd::dei::webapp::leofortriends::resource)
<<property>> - ID :int
<<property>> - content :string
<<property>> - actor :string
<<property>> - title : string
+ Question(id:int,content:string,actor:string,title:string)
+ toJSON(out:OutputStream) : void
+ fromJSON(in:InputStream) : question
Use
ResourceList
(It::unipd::dei::webapp::leofortriends::resource)
- list : Iterable< T >
+ResourceList(list : Iterable<T>)
+ toJSON(out:OutputStream) : void
Use
Use
ListUserQuestionsDatabase
(It::unipd::dei::webapp::leofortriends::database)
-STATEMENT :string = "SELECT question_number, title, content, actor FROM leofortriends.Question WHERE actor = ?"
+con:connection
+user:String
+ListUserQuestionsDatabase(con:connection,User:string)
+questions:list<questions>
Use
ListAllQuestionsDatabase
(It::unipd::dei::webapp::leofortriends::database)
-STATEMENT : string = "SELECT question, number, title, content, actor FROM leofortriends.Question"
+con:connection
+ListAllQuestionsDatabase(con:connection)
+listQuestions:list<questions>
Use
Question
(It::unipd::dei::webapp::leofortriends::resource)
<<property>> - ID :int
<<property>> - content :string
<<property>> - actor :string
<<property>> - title : string
+ Question(id:int,content:string,actor:string,title:string)
+ toJSON(out:OutputStream) : void
Use
ListQuestionAnswersDatabase
(It::unipd::dei::webapp::leofortriends::database)
-STATEMENT : string = "SELECT question, actor, content " +"FROM leofortriends.Answer WHERE question = ?"
+con: connection
+questionId : int
+ListQuestionAnswersDatabase(con:connection,questionId:int)
+searchQuestionAnswers: list<questions>
Use
Answer
(It::unipd::dei::webapp::leofortriends::resource)
<<property>> - questionId::int
<<property>> - actor: string
<<property>> -content:string
+ Answer(questionId:int,user:String,content:content)
+ toJSON(out:OutputStream) : void
+ fromJSON(in:InputStream) : answer


Servlet Class
Message
(it::unipd::dei::webapp::leoforfriends::database)
<<property>> - message :string
<<property>> - errorn code: string
<<property>> - errordetails:string
-isError - Boolean
+message(string:message,String:error code,string:error details)
+isError:boolean
AbstractDatabaseServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
-ds :datasource
+init:config.servlet.config) : void
+destroy():void
+fgetDataSource: DataSource
RestManagerServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
checkMethodMediaType(req:HttpServletRequest, res:HttpServletResponse) : void
processQuestion(req:HttpServletRequest, res:HttpServletResponse) : void
service(req:HttpServletRequest, res:HttpServletResponse) : void
AuthenticationServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doFilter(req:ServletRequest, res:ServletResponse, obj:FilterChainchainObj)
LoginServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : void
RegistrationServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : voic
ModifyUserServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : void
CreateAnswerServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : voic
CreateQuestionServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : void
LogoutServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : void
SearchQuestionServlet
(it::unipd::dei::webapp::leoforfriends::servlet)
doPost(req:HttpServletRequest, res:HttpServletResponse) : voic
<<uses>>
SearchUserDatabase
(it::unipd::dei::webapp::leoforfriends::database)
-STATEMENT: string = "INSERT INTO leoforfriends.Question" +"(title,content,actor) VALUES (?, ?, ?)"
+con:connection
+question:question
+CreateQuestionDatabase(con:connection,question:question)
+createquestion:void
CreateUserDatabase
(it::unipd::dei::webapp::leoforfriends::database)
- STATEMENT: string = "INSERT INTO leoforfriends.Actor" +"(email, username, hashed_password rating, name, surname, academic_title, certified)" +"VALUES (?, ?, ?, ?, ?, ?, ?, ?)"
+con:connection
+user:user
+CreateDatabase(con:connection,User:user)
+Createuser:user
ModifyUSerDatabase
(it::unipd::dei::webapp::leoforfriends::database)
-STATEMENT : string ="UPDATE leoforfriends.Actor SET username = ?, hashed_password = ?," +
" name = ?, surname = ?, academic_title = ? WHERE username = ? RETURNING *";
+con:connection
+User:user
+ModifyUserDatabase(con:connection,User:user)
+Updateuser:user
CreateAnswerDatabase
(it::unipd::dei::webapp::leoforfriends::database)
-STATEMENT : string="INSERT INTO leoforfriends.Answer" +"(question, actor, content) VALUES (?, ?, ?)"
+con:connection
+answer:answer
+CreateAnswerDatabase(con:connection,answer:answer)
+createanswer : void
CreateQuestionDatabase
(it::unipd::dei::webapp::leoforfriends::database)
-STATEMENT: string = "INSERT INTO leoforfriends.Question" +"(title,content,actor) VALUES (?, ?, ?)"
+con:connection
+question:question
+CreateQuestionDatabase(con:connection,question:question)
+createquestion:void
SearchUserQuestionDatabase
(it::unipd::dei::webapp::leoforfriends::database)
-STATEMENT:string = "SELECT * FROM leoforfriends.Actor " +"WHERE username = ? AND hashed_password = ?";
=con:connection
=user:string
=hashedpw:string
+SearchUserDatabase(con:connection,string:actor,string:hashedpw)
+SearchUSer:user
<<uses>>
User
(it::unipd::dei::webapp::leoforfriends::resource)
<<property>> - email :string
<<property>> - username:string
<<property>> - hashedpw:string
<<property>> - rating :double
<<property>> - name:string
<<property>> - surname:string
<<property>> - academicTitle:string
<<property>> - certified:boolean
+ User(email:string,username:string,name:string,surname:string,rating:double,certified:boolean,academic title:string)
+ toJSON(out:OutputStream) : void
+ fromJSON(in:InputStream) : user
Answer
(it::unipd::dei::webapp::leoforfriends::resource)
<<property>> - questionid:Int
<<property>> - actor: string
<<property>> -content:string
+ Answer(questionid:int,user:String,content:content)
+ toJSON(out:OutputStream) : void
+ fromJSON(in:InputStream) : answer
Question
(it::unipd::dei::webapp::leoforfriends::resource)
<<property>> - ID :int
<<property>> - content :string
<<property>> - actor :string
<<property>> - title : string
+ Question(id:int,content:string,actor:string,title:string)
+ toJSON(out:OutputStream) : void
+ fromJSON(in:InputStream) : question

2. UML Sequence Diagram

The below figure illustrates a representative instance of the UML sequence diagrams for the web applications. More specifically, it explains the processes instantiated in the server upon the posting of a new question: when an instance of CreateQuestionServlet is called via POST method, the Web Server configurates it and invokes the doPost method (requiring the connection to the DB provided by AbstractDatabaseServlet methods) which passes the posted content do the CreateQuestionDatabase class for insertion into the DB Server. The result is returned to the servlet, which forwards the result to a JSP page called create-question-result.



3. REST API Documentation

This web application exposes, via REST API, the functionalities listed in the following table.

URI	METHOD	DESCRIPTION
REST/qestions	GET	List all the questions.
REST/questions/users/{user id	GET	List all the question of the user.
REST/questions/question/{id}	GET	List all the answer of a question.

Here follows a list of three examples of JSON resources to be passed to each URI for the proper functioning of the web application.

GET REST/questions HTTP/1.1

Accept = application/JSON


```
{
   “resource-list”:[
                {
                   “question”:{
                              “id”:1,
                              “user”:”zhangguang”
                              “title”:”job interview in Wtd company”
                              “content”:”In my CV do I need to put all the previous work experience?”
                               “link”:”http://leoforfriends-1.00/REST/question/1”
                          }
                },
                {
                   “question”:{
                              “id”:2
                              “user”:”gian”
                              “title”:”job interview in Google company”
                              “content”:”Tomorrow I have a job interview in the Google company. What I need to bring with me?”
                              “link”:”http://leoforfriends-1.00/REST/question/2”
                          }
                },
                {
                   “question”:{
                              “id”:3,
                              “user”:”zhangguang”
                              “title”:”job interview in Amazon company”
                              “content”:Yesterday I had a job interview in the Amazon company and they said that they will call me when they have the result. I want to ask more or less how many days do I need to wait”
                              “link”:”http://leoforfriends-1.00/REST/question/3”
                          }
                }
        ]
}
```

GET REST/questions/users/zhangguang HTTP/1.1

Accept = application/JSON


```
{
   “resource-list”:[
                  {
                     “question”:{
                              “id”:1,
                              “title”:”job interview in Wtd company”
                              “content”:”In my CV do I need to put all the previous work experience?”
                               “link”:”http://leoforfriends-1.00/REST/question/1”
                           }
                  },
                  {
                     “question”:{
                              “id”:3,
                              “title”:”job interview in Amazon company”
                              “content”:Yesterday I had a job interview in the Amazon company and
                               they said that they will call me when they have the result. I want to ask
                               more or less how many days do I need to wait”
                               “link”:”http://leoforfriends-1.00/REST/question/3”
                           }
                  }
         ]

}
```


GET REST/questions/question/3 HTTP/1.1

Accept = application/JSON


```
{
   “resource-list”:[
            {
               “answer”{
                        “id”:3,
                        “username”=”giova”
                        “content”=”At least 5-6 days”
                           }
                  },
                  {
                     “answer”{
                        “id”:3,
                        “username”=”gian”
                        “content”=”A friend of mine received it after 5 days”
                           }
                  }
         ]
}
```