

# Internet Technologies Project Report

Marco Seravalli - 6822

24th June 2010

# Contents

<b>1</b>	<b>Overview</b>	<b>4</b>
1.1	Generalities . . . . .	4
<b>2</b>	<b>Functionalities</b>	<b>5</b>
2.1	The roles . . . . .	5
2.1.1	Viewers . . . . .	5
2.1.2	Collaborators . . . . .	5
2.1.3	Managers . . . . .	6
2.1.4	Administrators . . . . .	7
2.2	Manage the participants . . . . .	7
2.3	Manage the universities . . . . .	8
2.4	Manage the users . . . . .	9
2.5	Manage the articles . . . . .	9
2.6	Manage the comments . . . . .	9
2.7	Manage the web site . . . . .	9
2.8	Input check . . . . .	9
<b>3</b>	<b>Technologies &amp; Techniques</b>	<b>12</b>
3.1	XHTML 1.1, CSS2 . . . . .	12
3.2	Javascript . . . . .	12
3.3	Java Server Pages . . . . .	12
3.4	Java EE . . . . .	12
3.5	JDBC . . . . .	13
<b>4</b>	<b>Structure</b>	<b>14</b>
4.1	Database . . . . .	14
4.1.1	Roles . . . . .	14
4.1.2	Users . . . . .	14
4.1.3	Universities . . . . .	14
4.1.4	Participants . . . . .	14
4.1.5	Friday Program . . . . .	15
4.1.6	Articles . . . . .	15
4.1.7	Comments . . . . .	15
4.2	Procedure flow . . . . .	15
4.2.1	Insertion . . . . .	15
4.2.2	Modification . . . . .	16
4.2.3	Deletion . . . . .	16
4.2.4	Login . . . . .	17
4.2.5	Logout . . . . .	17
4.2.6	Retrieve data . . . . .	18
4.3	Security and consistency . . . . .	18

<b>5</b>	<b>Main Classes and methods</b>	<b>20</b>
<b>6</b>	<b>Outcomes</b>	<b>23</b>
6.1	General . . . . .	23
6.2	Difficulties & known issues . . . . .	23
6.3	Conclusion . . . . .	23

# 1 Overview

## 1.1 Generalities

The project has been developed in order to administer the Bolzano Snowdays, the greatest event organised by the students associations at the Free University of Bolzano. Several dozens of participants coming from the whole Europe take part in it.

Therefore, the purpose of this website is to give the possibility to the organisers to manage the enrolment of the universities and their participants. Moreover the administrators should be able to inform all the viewers about the news regarding the event, by posting and managing articles that will appear in the home page.

Furthermore, the organisers are divided between three different types of users, the administrators, that have a complete control on the website, the managers, that have an appropriate control on the universities and the participants, and the collaborators, who can only manage the participants of a single university.

## 2 Functionalities

### 2.1 The roles

In order to be able to deal with a lot of different people from several universities scattered all around Europe, the users of the website have been divided into four classes

- viewers
- collaborators
- managers
- administrators

Each user is not necessarily a participant, s/he can just help organising the event, without taking part in it.

#### 2.1.1 Viewers

As suggested by the name, these kind of users are allowed only to view the website and comment the posted articles, without accessing to the core part. However, the viewers have at their disposal several pages including the ones about the program, the contact, the video and the photos of the event.

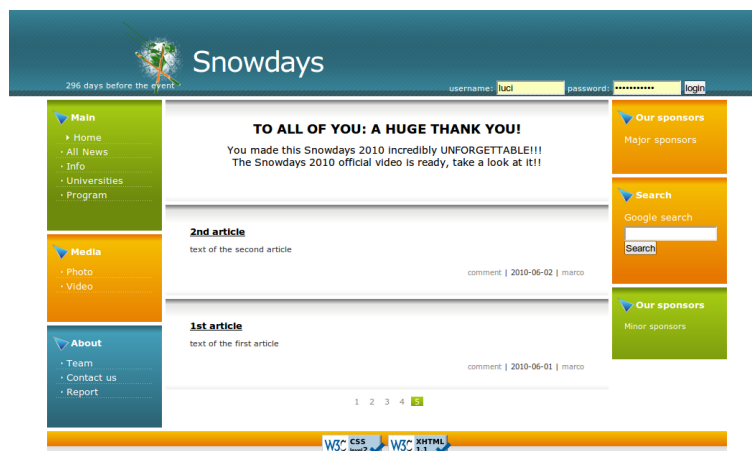


Figure 1: viewer's index

#### 2.1.2 Collaborators

Usually the collaborators are the representatives of their university, their task is to organise the participants from their university. Thus, they are allowed to access through username

and password, both given by the administrators. Each collaborator can add, edit and delete only participants from his/her university, they cannot even view the participants from other universities.

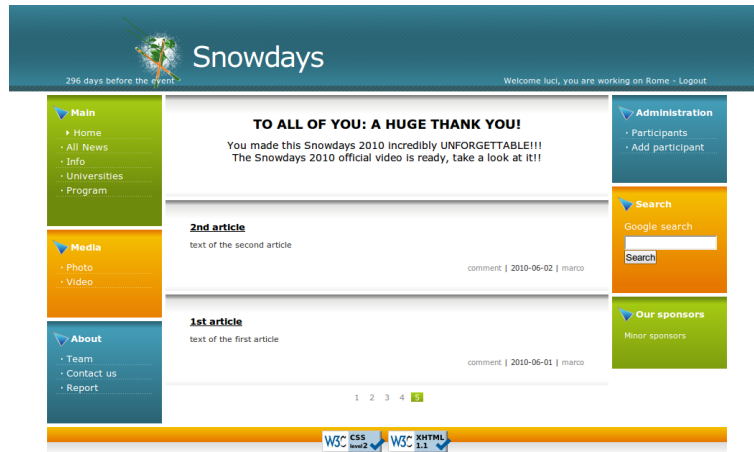


Figure 2: collaborator's index

### 2.1.3 Managers

The managers are basically staff members, they are allowed to view, edit, add and delete the participants from all universities. In addition, they can view, add, edit and delete universities. Moreover they are allowed to view all the users of the system and only the passwords of the collaborators in order to resend them, if it is needed.

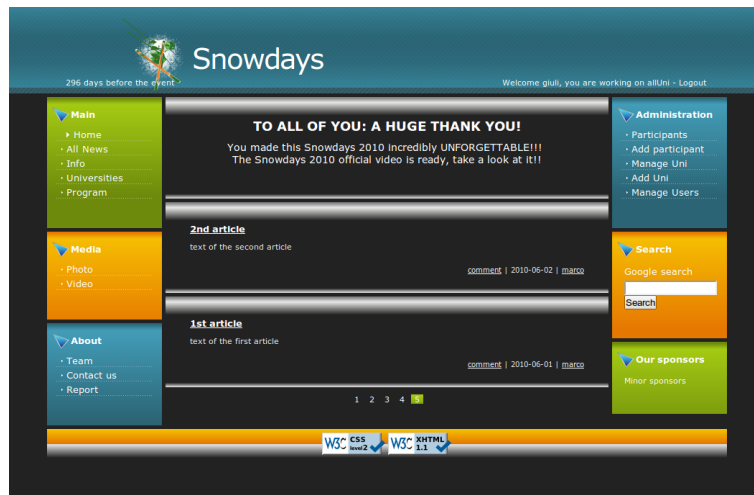


Figure 3: manager's index

### 2.1.4 Administrators

Of course the administrators have a full control of the website, they can manage directly every element. They can manipulate all the participants from all universities, they have full control on universities, users, articles and comments. Furthermore they have the possibility of accessing directly the database, in order to obtain some specific statistics or to change manually some data that is not possible to modify from the pages.

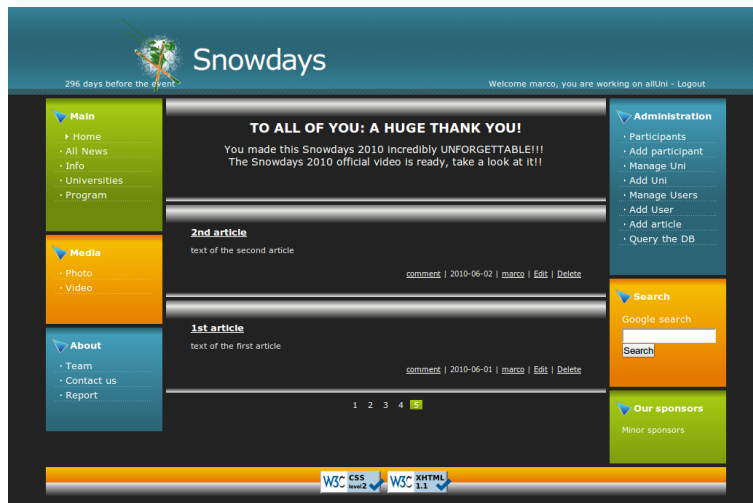


Figure 4: administrator's index

As you can see there are some dynamic changes even in the index page depending on which user and which kind of user is viewing it.

For example there are two sets of graphics, one for the viewers and the collaborators, based on white, and one for the managers and the administrators, based on black. Moreover for the logged users a sponsor box is not displayed in order to leave more space to the administration panel.

## 2.2 Manage the participants

The participants can be viewed and edited by all logged users, but of course with some restrictions for the collaborators. In fact, they can view and manage only the users belonging to their university. On the other hand, the managers and the administrators can view and edit participants from all universities. Moreover, they have the possibility of selecting a single university to work only on it. The currently selected university also appears in the header above the menu and next to the user-name.

There is also a “meta” university called “allUni” that actually cannot not have any participant, cannot be deleted or edited. This university is useful because if it is selected as working university, all participants can be viewed at once. In the further subsections more details about this element will be given.

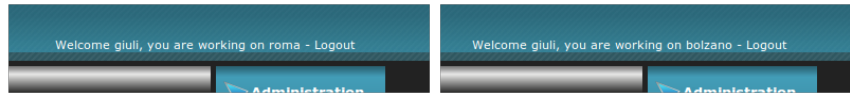


Figure 5: user-name and university

Each university has a maximum number of participants that cannot be exceeded, if a user tries to insert an additional participant an error is displayed to the user.

## 2.3 Manage the universities

The universities can be managed in the same way by the managers and by the administrators, they can view, add, edit and delete universities according to some rules. No university can be deleted if some participants or some users are enrolled at it and the number of participants cannot be less than the current registered participants.

name	total participants	total teams	users		
allUni	15 / 175	7	3	<a href="#">edit</a>	<a href="#">delete</a>
Berlin	4 / 25	2	0	<a href="#">edit</a>	<a href="#">delete</a>
Bozen	6 / 100	3	0	<a href="#">edit</a>	<a href="#">delete</a>
Rome	5 / 50	2	1	<a href="#">edit</a>	<a href="#">delete</a>

Figure 6: Managing the universities

Each university can also have a maximum number of volleyball teams which is decided by the managers or the administrators.

Between all the universities there is a “meta” university called “allUni”. This university cannot have any participant, cannot be edited or deleted. Anyway it is useful because through it can be displayed participants from all universities. Furthermore, this uni contains information about all universities, such as the total number of registered participants, and it is updated whenever another university is changed. Moreover it is used to handle some incoherences during the update of the universities, for example, if a manager is working on “Berlin” and s/he deletes it, the working university is changed to “allUni”.



## **2.4 Manage the users**

Basically, the users can be controlled only by administrators. In fact, they can view every aspect of each single user and edit every one of them. Differently, the managers are only able to see all the users and the passwords only of the collaborators, in order to resend them, if it is necessary.

## **2.5 Manage the articles**

The articles can be viewed by anyone, but can be added, edited and deleted only by the administrators. The author and the date of each article change automatically whenever an article is edited, the date is updated to present and the author is set as the last editor. Finally, the edit page can be accessed by the administrators from every place where the articles are visible (index, archive and article page).

## **2.6 Manage the comments**

Every one should feel free to comment the articles and either criticize or thank the organisers. Thus, the possibility of commenting, it is given to everyone. Anyway, the comments can be edited and deleted only by the administrators. The editing page can be accessed only from the reading page of the article.

## **2.7 Manage the web site**

The administrators have another powerful possibility of managing the website and retrieving some statistics from the database. They are allowed to query directly the database. This feature is provided in order not to create too many pages and leave the possibility to the administrators to perform some changes in the database, like deleting all the universities after the end of the event, or editing the Friday activities.

Finally, if the users have some troubles can contact the administrators directly from the contact page, where there is the email address of each single administrator.

## **2.8 Input check**

Before being accepted from the database all inputs given by the users are checked by the servlets and some sensible characters are substituted in order to have a better rendering and support the standards, for example the stressed vowels and the vowels with dieresis are translated into HTML standards ( 'à' -> '&agrave;'). Moreover in order to guarantee data consistency in the database input fields are checked twice, both on client side, by some scripts, and on server side, by the servlets. Of course if some problems are found they are displayed to the user.

To conclude, further checks and substitutions are made, in order to avoid nasty code and attacks to the system, in fact without these kind of controls some malicious users could

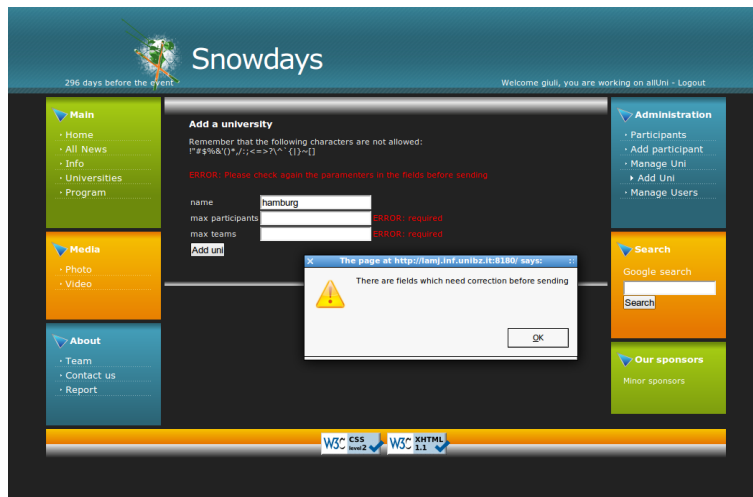


Figure 7: Input check

attack and take full control of the system simply performing a smart comment like the following one:

```
0',8);
INSERT INTO users (username, password, name, surname, email,
                    university, role)
VALUES ('hacker', 'H4Ck3r', 'poor you', 'old admin', 'aa@aa.aa',
        'allUni', 'administrator');
INSERT INTO comments (text, article_id) VALUES('I fooled you!!!
```

The system would have processed the query translating it as

```
INSERT0 INTO comments(text, article_id) VALUES('0',8 );
INSERT INTO users (username, password, name, surname, email,
                    university, role)
VALUES ('hacker', 'H4Ck3r', 'poor you', 'old admin', 'aa@aa.aa',
        'allUni', 'administrator');
INSERT INTO comments (text, article_id) VALUES('I fooled you!!!', 8)
```

producing devastating results.

Fortunately, this kind of attack is not more possible and the output produced is a simple comment.

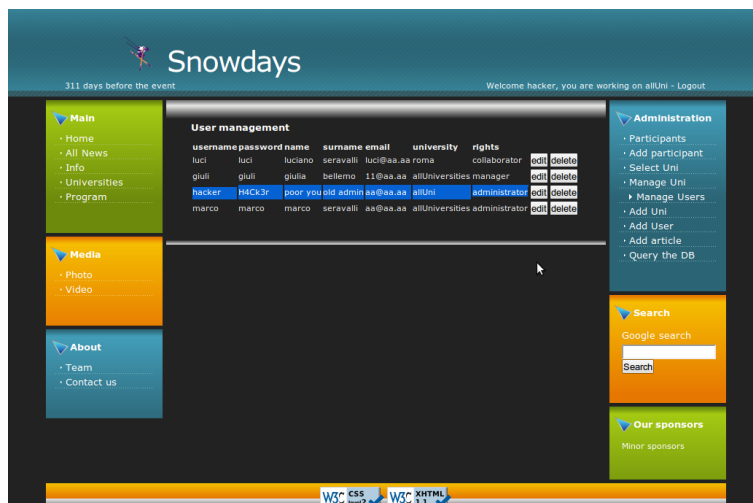


Figure 8: Successful attack

## 3 Technologies & Techniques

The website was developed using the technologies and the techniques studied during the course, which basically are Java EE, JDBC, Java Server Pages, Javascript, CSS, HTML. The whole system has been developed to run on Apache Tomcat 6.0 or successive version, the database management system is PostgreSQL 8.3. The project has been tested on Internet Explorer 8, Google Chrome 5 and Firefox 3.5.

### 3.1 XHTML 1.1, CSS2

To develop the static content, XHTML 1.1 and CSS2 have been used in order to comply to recent standards and to give the possibility to the website to be used and visualised correctly not only in the imminent future. Two different style sheets have been developed. One for the viewers and the collaborators based on white, and one based on black for the managers and the administrators. These choice is motivated by the intention of giving even more importance to the advanced users and to make these aware of their role, in order to induce them to treat carefully the data they are managing.

The chosen layout is a common layout with the menu on the left and the banners (in this case the sponsors) on the right. Anyway, this layout changes when a user logs in, in fact one of the sponsor pane is removed in order to be replaced by the administration menu. The menu is not simply added above the sponsor pane in order not to have too many panes on the right side and sometimes have a rather long page with few contents.

### 3.2 Javascript

Javascript is used to check the correctness of the input in each form, if some errors or incoherences are found, an alert box is displayed to the user. However, this is an additional check made on the client, because the checks are also performed on server side, in order to always have consistent data in the database.

### 3.3 Java Server Pages

The java server pages are used mainly to handle and display the beans and the contents passed by the servlets and to avoid the possibility of accessing reserved areas to non registered or non allowed users. If a user tries to access a restricted area a simple error message is displayed.

### 3.4 Java EE

The servlets have been used to handle all the requests of the user to edit, add and delete some elements, different servlets have been used for each kind of element. Moreover also the login is made using a servlet. Additionally, every servlet sends a redirect or forwards to another page when it has finished its task.

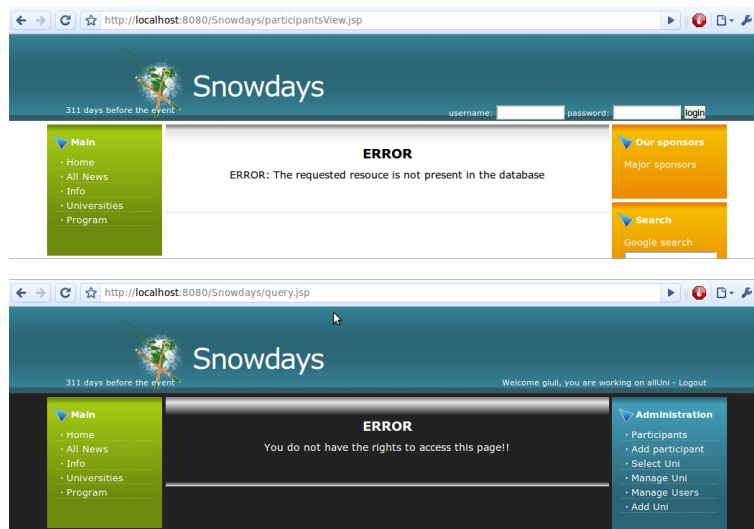


Figure 9: Not enough rights

On the other hand the beans have been used only to exchange information between the different parts of the system. Almost every table in the database has a corresponding bean.

### 3.5 JDBC

The JDBC is used to connect to the database and for every connection a common pattern has been used.

## 4 Structure

### 4.1 Database

The database is organised in several tables as you can see in the ER scheme

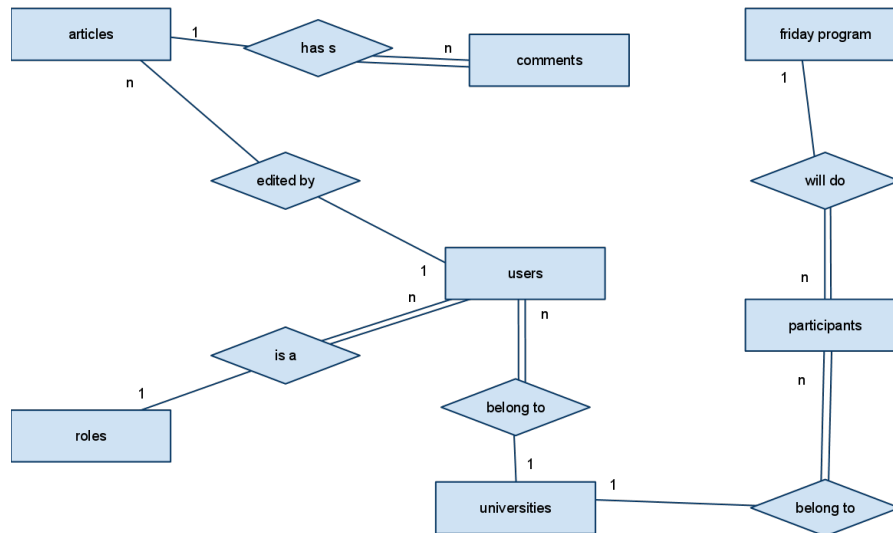


Figure 10: ER diagram

#### 4.1.1 Roles

This table defines the roles and an id for each role.

#### 4.1.2 Users

This table contains, the username, which is the primary key of the relation, it also contains the password, the name, the surname, the email, the university and the role of each user. All the fields but the name and surname are mandatory. University and role are foreign keys that refer to universities and roles respectively.

#### 4.1.3 Universities

Universities is one of the most important tables, it contains the name of the university, the maximum number of participants and the maximum number of teams for that university, both these numbers cannot be less than 0.

#### 4.1.4 Participants

The participants relation contains all the information useful for the organisers like the name, the surname, the sex, the birth date, the email, the phone number, the university, the

image for the pass, the Friday program a participant will take part in, a possible alimentary intolerance, if s/he rents something the shoe number and the size of the t-shirt. Almost all the fields are required, the rental is not required, but if it is filled the shoe number should be filled too, in order to have consistent data in the database

#### 4.1.5 Friday Program

This is just a list of activities, it is needed because it keeps the database consistent.

#### 4.1.6 Articles

The articles table contains the title, the text, the author, the date and the id of the article.

#### 4.1.7 Comments

This relation simply contains the comment id, the id of the article and the text of the comment itself.

### 4.2 Procedure flow

For every similar action an almost equal procedure has been adopted. For example the article insertion is similar to the university insertion which in turn is similar to the participant one. Thus, for space reasons, the insertion, edit, delete and retrieve procedure of a generic element will be explained, but the reader should be aware that the procedures for all elements work almost in the same way.

#### 4.2.1 Insertion

The insertion works as follows:

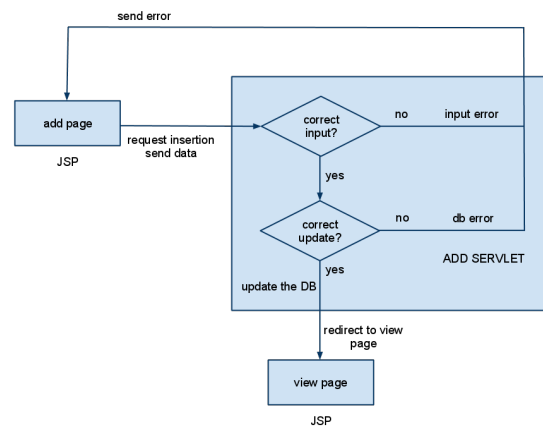


Figure 11: Insertion

The user requires an insertion, if the parameters are correct the servlet tries to insert the passed data in the database, if somewhere an error occurs it is notified to the user.

#### 4.2.2 Modification

The following scheme explains the modification:

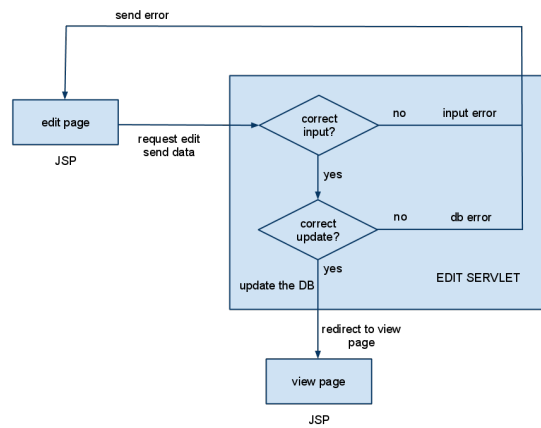


Figure 12: Modification

The modification works very similarly to the insertion.

#### 4.2.3 Deletion

The deletion follows the next scheme:

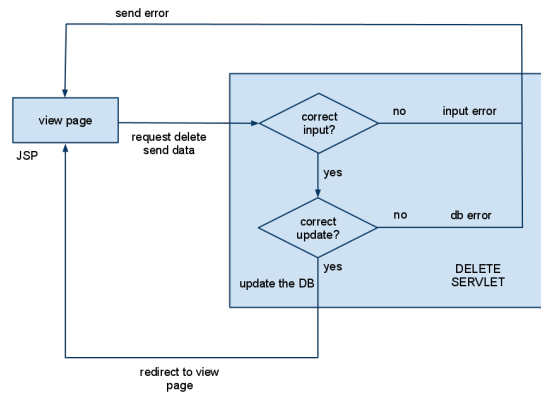


Figure 13: Deletion

Also the deletion works similarly to the insertion and the modification.



#### 4.2.4 Login

The login is implemented in this way:

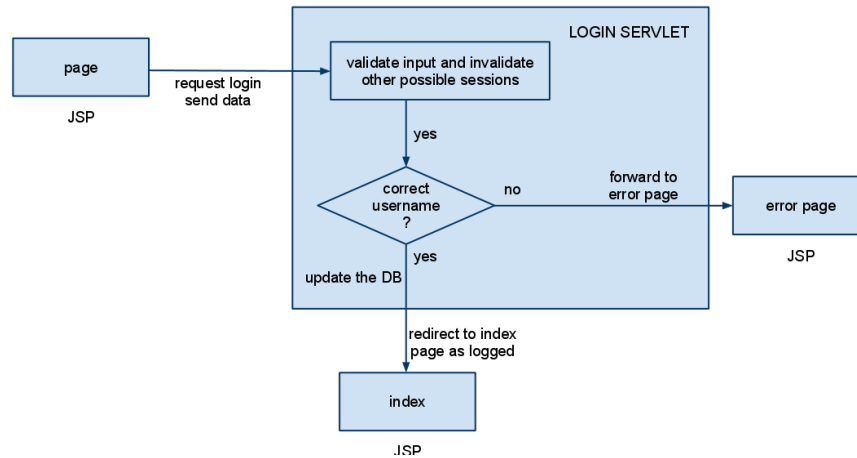


Figure 14: Login

First of all, the servlet invalidates other possible sessions stored for the current client. After that, before controlling the passed username and password, validates the parameters by deleting some invalid characters, in order to avoid attacks. Then the list of all possible users is retrieved and if there is a match with the passed values, a user bean is populated and the client is redirect to the main page, otherwise the servlet forwards to an error page.

#### 4.2.5 Logout

The logout works as follows:

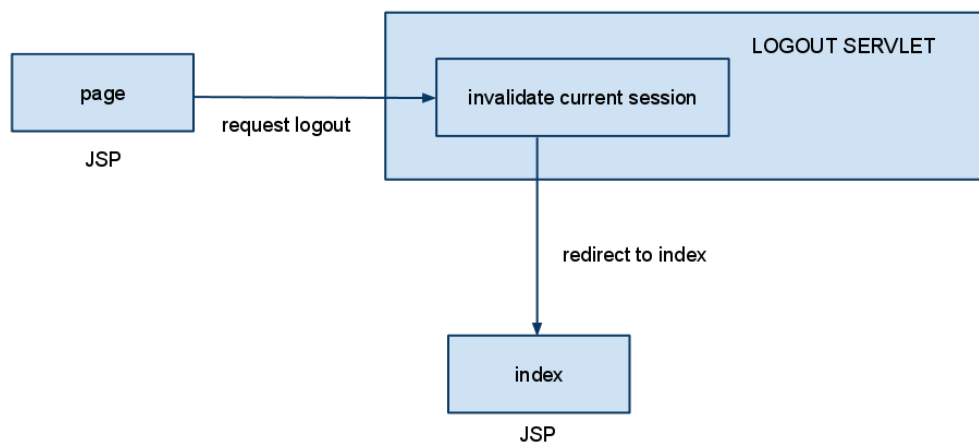


Figure 15: Logout

When the user requests the logout the current session is invalidated and the client is redirected to the index page.

#### 4.2.6 Retrieve data

The data retrieving follows the next scheme:

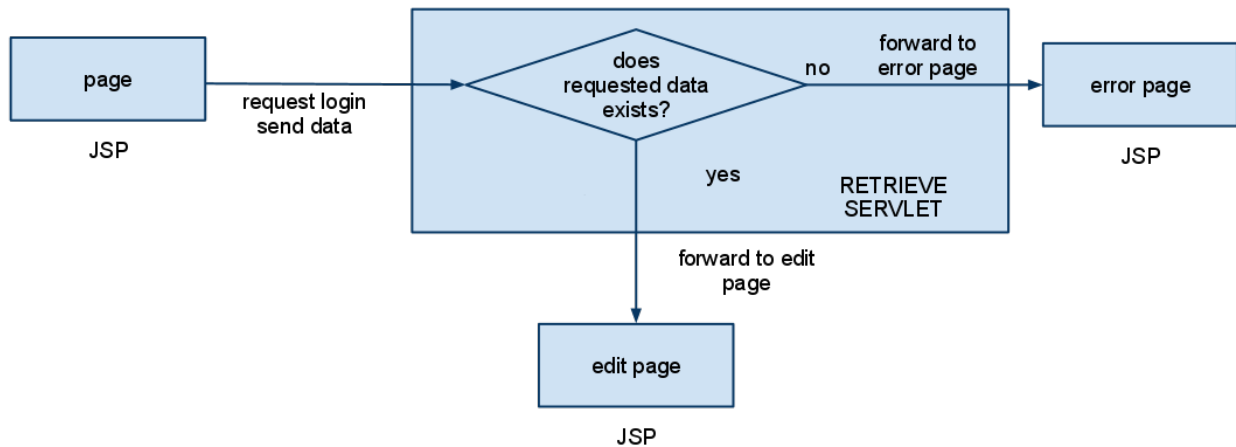


Figure 16: Access

The data is requested from a servlet, if it is present in the database, it is displayed to the user, otherwise the request is forwarded to an error page.

### 4.3 Security and consistency

In the system the given input is directly passed to the database as few times as possible, in order to decrease the possibility of external attacks. For example in the ordering of the participants the order parameter is checked and a predefined query is run. Moreover, all passed parameters are checked by the servlets before being passed to the database.

Furthermore, even if some wrong parameter is passed the system do not crash and an error message is displayed to the user, this is done in order not to display inconsistent or wrong data, and this check is performed twice, on the servlet and on the JSP by Javascript.

Concluding, the data present in the database is kept as consistent as possible through two checks of input, both on the client and on the server side.

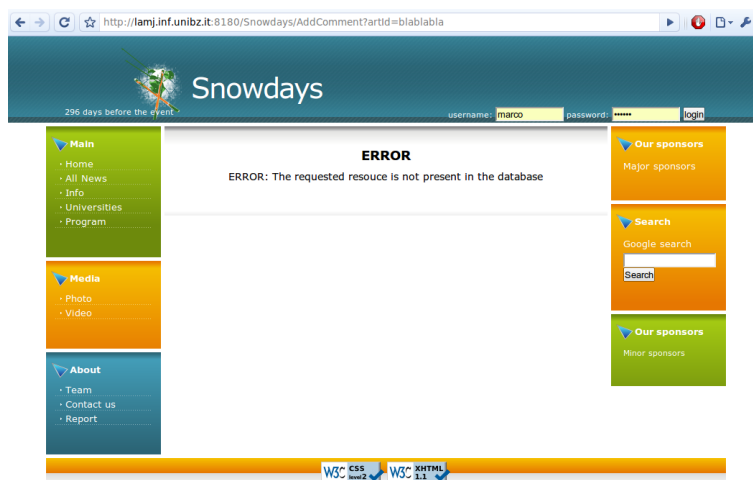
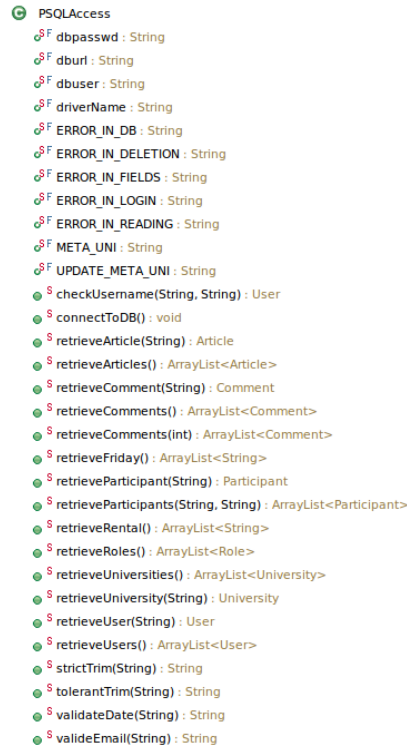


Figure 17: Error returned while trying to access non existent data

## 5 Main Classes and methods

One of the most important classes is probably PSQLAccess, in fact this class contains all the methods to retrieve the different kind of data from the database. All the methods are stored here in order to have them all in only one place.



```
PSQLAccess
  dbpasswd : String
  dburl : String
  dbuser : String
  driverName : String
  ERROR_IN_DB : String
  ERROR_IN_DELETION : String
  ERROR_IN_FIELDS : String
  ERROR_IN_LOGIN : String
  ERROR_IN_READING : String
  META_UNI : String
  UPDATE_META_UNI : String
  checkUsername(String, String) : User
  connectToDB() : void
  retrieveArticle(String) : Article
  retrieveArticles() : ArrayList<Article>
  retrieveComment(String) : Comment
  retrieveComments() : ArrayList<Comment>
  retrieveComments(int) : ArrayList<Comment>
  retrieveFriday() : ArrayList<String>
  retrieveParticipant(String) : Participant
  retrieveParticipants(String, String) : ArrayList<Participant>
  retrieveRental() : ArrayList<String>
  retrieveRoles() : ArrayList<Role>
  retrieveUniversities() : ArrayList<University>
  retrieveUniversity(String) : University
  retrieveUser(String) : User
  retrieveUsers() : ArrayList<User>
  strictTrim(String) : String
  tolerantTrim(String) : String
  validateDate(String) : String
  valideEmail(String) : String
```

Figure 18: PSQLAccess

Two important methods are the ones that replace some sensitive characters that could lead to some nasty code.

```
public static String tolerantTrim(String s){
    if(s != null){
        s = s.trim();
        // at first place because otherwise it substitutes the & inserted b
        // other replacements
        s = s.replaceAll("&", "&");
        s = s.replaceAll("<", "&lt;");
        s = s.replaceAll(">", "&gt;");
        s = s.replaceAll("è", "&egrave;");
        s = s.replaceAll("Ë", "&egrave;");
        s = s.replaceAll("ì", "&igrave;");
        s = s.replaceAll("Ï", "&igrave;");
```

```

        s = s.replaceAll("ò", "&ograve;");
        s = s.replaceAll("Ã", "&ograve;");
        s = s.replaceAll("ö", "&ouml;");
        s = s.replaceAll("ù", "&ugrave;");
        s = s.replaceAll("Û", "&ugrave;");
        s = s.replaceAll("ü", "&uuml;");
        s = s.replaceAll("à", "&agrave;");
        s = s.replaceAll("Ã", "&agrave;");
        s = s.replaceAll("ä", "&auml;");
        //the result of replacement is not perfect
        //anyway it is good enough and
        //it is an acceptable tradeoff
        s = s.replaceAll("'", "&prime;");
        s = s.replaceAll("\", "&rdquo;");
    }
    else {
        s = "";
    }
    return s;
}

public static String strictTrim(String s){
    if(s != null){
        s = s.trim();
        //all the following characters are deleted
        s = s.replaceAll("[!\"#$%&'()*+,-./:;<=>?\\^_`{|}~\\[\\backslash]", "");
    }
    else {
        s = "";
    }
    s = tolerantTrim(s);
    return s;
}

```

Another interesting class is AddParticipant, which uses javazoom libraries to upload and store the passed files on the disk.

Here is the method that actually handles the uploaded files:

```

public static String canUploadImg(MultipartFormDataRequest mrequest,
    String path, String id, String oldImgPath) throws Exception{
    // Uses MultipartFormDataRequest to parse the HTTP request.
    String todo = null;
    if (mrequest != null) todo = mrequest.getParameter("todo");
    if ( (todo != null) && (todo.equalsIgnoreCase("upload")) ){

```

```

Hashtable files = mrequest.GetFiles();
if ( (files != null) && (!files.isEmpty()) ){
    UploadFile file = (UploadFile) files.get("uploadfile");
    if (file != null &&
        file.getFileSize() <= (200*1024) &&
        (file.getContentType().equals("image/gif") ||
         file.getContentType().equals("image/png") ||
         file.getContentType().equals("image/jpeg"))){
        int lastPoint = file.getFileName().lastIndexOf('.');
        String fileName = id + file.getFileName().substring(lastPoint,
            file.getFileName().length());
        file.setFileName(fileName);
        UploadBean upBean = new UploadBean();
        //delete the old image if required
        if(oldImgPath != null){
            File oldImg = new File(path + oldImgPath);
            oldImg.delete();
        }
        upBean.setFolderstore(path);
        upBean.store(mrequest, "uploadfile");
        return fileName;
    }
}
else {
    //no file or wrong file uploaded
}
}
return null;
}

```

## **6 Outcomes**

### **6.1 General**

For this project no status error (40X or 50X) have been voluntarily used. It was preferred to try to handle each possible situation and by either forwarding or displaying an error message to remain always within the website. This, in order to provide a better user experience and to minimize the inconvenience for the users while surfing.

The names of elements were given according to their logical purpose, for example all the JSPs start with the name of the element that should be displayed, on the other hand all the servlets names begin with the action they perform.

### **6.2 Difficulties & known issues**

One of the main difficulties was the fact that we learned how to develop the website little by little, for this reason I preferred to start developing almost at the end of the course in order to know better all patterns and capabilities of the environment. Thus, it has been pretty difficult to develop the whole system in a very short time. Some other difficulties raised while trying to write the scripts, because of the incompatibility of some browsers and while trying to upload the participants' images, in particular for some unknown I did not manage to upload images different from .gif from MS-IE user agents, if the user surfs with it simply a different list of extensions is displayed. There are some pages which are not strict XHTML1.1, the photo and the video pages, due to the embedded parts.

### **6.3 Conclusion**

This has been a very interesting and challenging project and I think it was very useful to understand a part of all the possible problems that can rise while developing such kind of application. Moreover, it helped me to learn how difficult is building a solid system and thinking about all possible misuses and tentatives of breaking security.