Administration GUI for the SonkalWEB

Victoria Lopez Clemente

Supervisor: Ing. Pavel Loupal

Education Program: Computer Technology-structured studies
Option: Computer Science
June 2009


Acknowledgments
I would like to thank everybody who has supported me doing this project. I want to say thanks to my supervisor Ing. Pavel Loupal who has always supported me and helped me in the most difficult moments of the project. Also, I want to thank my friends, especially to Checa and Raulongo who have supported me in my bad moments and gave me strength for continue until I solve them. I want to thank my family too for giving me the chance of studying this year in Praha. Thank you very much.
Administration GUI for SonkalWeb

- Index -

1. Project Definition
   1. Scope of work........................................4

2. Analysis
   1. Short Description about the actors and Use Case....5
   2. Uses Cases Diagram....................................7

3. Design
   1. Context Diagram........................................9
   2. E-R Diagram............................................10
   3. Web page Design.......................................12
   4. Class Diagram..........................................12

4. Implementation of DataBase
   1. DB Schema.............................................13
   2. Implementation of DB..................................18
      1. Example of information in Data Base..............22

5. Software implementation
   1. Coding..................................................25

6. Realisation .................................................30

7. Conclusion ..................................................34

8. Bibliography................................................35
1. Project Definition

The goal of the project is to put into practice the knowledge of SQL and JSP. The project is a Sonkal web extension, where it will have two different kinds of users:

1.1 Scope of work

The project starts as a final degree work, in which we want to put into practice the knowledge of SQL and JSP, managing a database. The project is based primarily on the expansion of the judo Sonkal website for anonymous web users. The aim is to provide efficient management of the site, orientating it to a more dynamic website where users can interact with information stored in the database, creating, editing or deleting data.

In order to do this task, a level of security has been created, where only authorized users can access to it. Because of that, we have created two kinds of users that can access to the system for managing the database. This are the administrator and the editor.

The Editor will be the user with the responsibility of helping the administrator to maintain the website. The Editor will update the data or fill it with information. Some of the tasks of the editor will be: assign a person to a photograph, to a video, write some articles...

The administrator will be responsible to allow / disallow the editor users. On the other hand the administrator is allowed to do the same tasks as the editor, but also it can create, edit or delete data. The administrator will also be responsible of maintaining the relationship between tasks with the system and approve the changes proposed by the editors.
2. Analysis:

2.1 Short Description about the actors and Use Case

Web application, which will develop for manage data about Sonkal web, will be used for two different kinds of users. This application offers some services to be able to do different operations like update photography, edit data, read a list of people...

The Administrator and the Editor interact with the web through a Java application. This application is responsible for managing the data.

The tasks that can be performed by the Administrator are:

- Create a new Rank
- Read a list of ranks
- Edit an existing rank
- Delete an existing rank

- Create a new tool
- Read a list of tools
- Edit an existing tool
- Delete an existing tool

- Create a new event_type
- Read a list of event_type
- Edit an existing event_type
- Delete an existing event_type

- Create a new category
- Read a list of categories
- Edit an existing category
- Delete an existing category

- Create a new technique
- Read a list of techniques
- Edit an existing technique
- Delete an existing technique

- Create a new person
- Read a list of people
- Edit an existing person
- Delete an existing person

- ...
The tasks that will be held by the editor:

- Assign a person to a photograph.
- Assign a person to a video.
- Write an article for an event.
- Set type of a photograph.
- Enter competitors’ results to a competition.
- ...

In each use case, the user will receive a response. Each request will get a valid value in the form of text message. This message will indicate whether the operation being carried out has been implemented without problems, or an error has occurred.

**Create a new rank**

Allows adding a rank to the database of the website. This will be clearly identifiable in the list of ranks by an identifier, the colour, the English colour name and the name of the rank. Related to what I said before, the user may not be able to include an existing rank. This will occur when the colour or colours in English has been used by yet another introduced range.

**Read a list of rank**

Allows listing all ranks created, identifying for each rank: id, colour, English name and colour range.

**Edit an existing rank**

Allows editing an existent rank in the database. Also, the colour, the colour in English and the rank name can be edited. The id can’t be edited though.

**Delete an existing rank**

Allows deleting one or more ranks of the database.

For each table of the explained before, the explanation is the same.
2.2 Use Cases Diagram
**Administration GUI for SonkalWeb**

**Some list-related Use Cases:**

<table>
<thead>
<tr>
<th>ID:</th>
<th>UC01</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CRUD Rank</td>
</tr>
<tr>
<td>User Role:</td>
<td>Administrator</td>
</tr>
<tr>
<td>Description:</td>
<td>User can create, retrieve, update or delete data in the rank table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID:</th>
<th>UC02</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CRUD Pattern</td>
</tr>
<tr>
<td>User Role:</td>
<td>Administrator</td>
</tr>
<tr>
<td>Description:</td>
<td>User can create, retrieve, update or delete data in the pattern table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID:</th>
<th>UC03</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CRUD Tool</td>
</tr>
<tr>
<td>User Role:</td>
<td>Administrator</td>
</tr>
<tr>
<td>Description:</td>
<td>User can create, retrieve, update or delete data in the tool table.</td>
</tr>
</tbody>
</table>

This use case is the same for these tables: event_type, category, technique, person, event, article, link news, photo and video.

<table>
<thead>
<tr>
<th>ID:</th>
<th>UC14</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CRUD Person Photo</td>
</tr>
<tr>
<td>User Role:</td>
<td>Editor and Administrator</td>
</tr>
<tr>
<td>Description:</td>
<td>User can create, retrieve, update or delete data in the person_photo table.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ID:</th>
<th>UC15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name:</td>
<td>CRUD Person Video</td>
</tr>
<tr>
<td>User Role:</td>
<td>Editor and Administrator</td>
</tr>
<tr>
<td>Description:</td>
<td>User can create, retrieve, update or delete data in the person_video table.</td>
</tr>
</tbody>
</table>

This is the same for article and type photography.
3. Design:

3.1 Context Diagram

This is the context diagram between the users (administrator and editor) and the Administration GUI. In this diagram we can see the different functions its can use by the users.

This context diagram is more general, in it we can see easily how the public user can interact with the web, and the administrator and editor interacts with the database through the administrator GUI Web System.
3.2 E-R Diagram

This diagram shows the data structure of the database and it contains 16 tables.

- Table of rank which stores the information about each existent rank in judo.
- Table of pattern which stores the information about the pattern of each rank.
- Table of tools which stores the information about the tools.
- Table of event_type. This stores the kind of all the types of judo events.
- Table of category which stores the description of each category.
- Table of techniques which explain each technique with the name and a description.
- Table of person. It stores all the information of the users. This table contains the user and the password of each user. This information is important to know which users are allowed to login.
- Table of event which stores the information about judo events.
- Table article which stores the articles created by the users.
- Table of links which stores the information of the most important judo webs and other’s interests.
- Table of news which stores the news created by the users.
- Table of photo. It contains the information about the photos.
- Table of video which stores the information about the videos.
- Table of person_video. In this table, the editors can assign a person to a video.
- Table of person_photo. In this table, the editors can assign a person to a photo.
- Table of photo_type. It contains the types of photos.
3.3 Web page design

When I designed the look of the web page application, I wanted to have two separate pages, one for the login and one for all the content. This second part is divided by 3 areas: One with the list of all the tables (in the left side), other with a menu for retrieving data (in the top), and finally, one more in the middle, which is related to the ones I explained before.

When choosing the colours, I specially selected colours which can be easily read alongside the characters. For this, I have a light blue background and gray characters. In order to make the menu buttons, I've chosen the characters in orange and a blue background.

3.4 Class Diagram

I decided to distribute application logic over several packages and classes. Classes are divided by several packages. Each package contains the information of all the classes used to manage each data of the table, and also to show it to the user.
4. Implementation of Database:

4.1 DB Schema

The application works with this database tables. Each entity has a corresponding relation with the web Sonkal, which are essential for proper maintenance of the information on the web.

Rank Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>colour</td>
<td>Tinytext</td>
</tr>
<tr>
<td>colour_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>rank_name</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>

Pattern

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>id_rank</td>
<td>Integer</td>
</tr>
<tr>
<td>diagram_id_number</td>
<td>TinyInt</td>
</tr>
<tr>
<td>diagram_rows</td>
<td>TinyInt</td>
</tr>
<tr>
<td>diagram_columns</td>
<td>TinyInt</td>
</tr>
</tbody>
</table>

Tool

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description_EN</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>
### Event_type

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>

### Category

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>description</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description_EN</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>

### Technique

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>description</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>note</td>
<td>Tinytext</td>
</tr>
<tr>
<td>note_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>show_in_list</td>
<td>Enum (true, false)</td>
</tr>
</tbody>
</table>

### Person

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>username</td>
<td>Tinytext</td>
</tr>
<tr>
<td>password</td>
<td>Tinytext</td>
</tr>
<tr>
<td>surname</td>
<td>Tinytext</td>
</tr>
<tr>
<td>birth_date</td>
<td>Date</td>
</tr>
<tr>
<td>team</td>
<td>Enum(true, false)</td>
</tr>
</tbody>
</table>
### Administration GUI for SonkalWeb

#### note
- **Type**: Text

#### note_EN
- **Type**: Text

#### active
- **Type**: Enum(true,false)

#### sonkal_member
- **Type**: Enum(true,false)

#### id_rank
- **Type**: Integer

#### exams_date
- **Type**: Date

#### dan_number
- **Type**: Varchar

#### portrait
- **Type**: Tinytext

#### Event

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Type</td>
</tr>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>date</td>
<td>Date</td>
</tr>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>place</td>
<td>Tinytext</td>
</tr>
<tr>
<td>id_event_type</td>
<td>Integer</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>place_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>plan</td>
<td>Enum(true,false)</td>
</tr>
</tbody>
</table>

#### Article

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>id_event</td>
<td>Integer</td>
</tr>
<tr>
<td>publication_date</td>
<td>Date</td>
</tr>
<tr>
<td>header</td>
<td>Tinytext</td>
</tr>
<tr>
<td>body</td>
<td>Text</td>
</tr>
<tr>
<td>header_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>body_EN</td>
<td>Text</td>
</tr>
<tr>
<td>author</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>

#### Link

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
</tbody>
</table>
### Administration GUI for SonkalWeb

<table>
<thead>
<tr>
<th>Field</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Tinytext</td>
</tr>
<tr>
<td>www</td>
<td>Tinytext</td>
</tr>
<tr>
<td>contact</td>
<td>Tinytext</td>
</tr>
<tr>
<td>street</td>
<td>Tinytext</td>
</tr>
<tr>
<td>house_number</td>
<td>Varchar</td>
</tr>
<tr>
<td>city</td>
<td>Tinytext</td>
</tr>
<tr>
<td>name_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>country</td>
<td>Tinytext</td>
</tr>
<tr>
<td>country_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>continent</td>
<td>Tinytext</td>
</tr>
<tr>
<td>continent_EN</td>
<td>Tinytext</td>
</tr>
<tr>
<td>postcode</td>
<td>Varchar</td>
</tr>
<tr>
<td>telf</td>
<td>Tinytext</td>
</tr>
<tr>
<td>fax</td>
<td>Tinytext</td>
</tr>
<tr>
<td>email</td>
<td>Tinytext</td>
</tr>
<tr>
<td>note</td>
<td>Text</td>
</tr>
<tr>
<td>note_EN</td>
<td>Text</td>
</tr>
<tr>
<td>federation_member</td>
<td>Enum(false,true)</td>
</tr>
<tr>
<td>valid</td>
<td>Enum(false,true)</td>
</tr>
</tbody>
</table>

### News

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>date</td>
<td>DateTime</td>
</tr>
<tr>
<td>expiry_date</td>
<td>Date</td>
</tr>
<tr>
<td>text</td>
<td>Text</td>
</tr>
<tr>
<td>text_EN</td>
<td>Text</td>
</tr>
</tbody>
</table>

### Photo

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>id_event</td>
<td>Integer</td>
</tr>
<tr>
<td>id_photo_type</td>
<td>Tinyint</td>
</tr>
<tr>
<td>file</td>
<td>Varchar</td>
</tr>
</tbody>
</table>
### Administration GUI for SonkalWeb

**description** | Text  
---|---  
**description_EN** | Text  
**viewcount** | Integer  

#### Video

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>id_event</td>
<td>Integer</td>
</tr>
<tr>
<td>file</td>
<td>Tinytext</td>
</tr>
<tr>
<td>description</td>
<td>Text</td>
</tr>
<tr>
<td>description_EN</td>
<td>Text</td>
</tr>
<tr>
<td>view_count</td>
<td>Integer</td>
</tr>
</tbody>
</table>

#### Person_photo

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_person</td>
<td>Integer</td>
</tr>
<tr>
<td>id_photo</td>
<td>Integer</td>
</tr>
</tbody>
</table>

#### Person_video

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id_person</td>
<td>Integer</td>
</tr>
<tr>
<td>id_video</td>
<td>Integer</td>
</tr>
</tbody>
</table>

#### Photo_type

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Integer</td>
</tr>
<tr>
<td>type</td>
<td>Tinytext</td>
</tr>
<tr>
<td>type_EN</td>
<td>Tinytext</td>
</tr>
</tbody>
</table>
4.2 Implementation of BD

A script of this table is follows:

**Table rank:**

```sql
CREATE TABLE IF NOT EXISTS `rank` (
  `id` INT(10) NOT NULL,
  `colour` TINYTEXT NOT NULL,
  `colour_EN` TINYTEXT NOT NULL,
  `rank_name` TINYTEXT NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE = InnoDB;
```

**Table person:**

```sql
CREATE TABLE IF NOT EXISTS `person` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `name` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `username` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `password` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `surname` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `birth_date` DATE NULL,
  `team` ENUM('false','true') CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL,
  `note` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL,
  `note_EN` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL,
  `active` ENUM('false','true') NULL DEFAULT 'true',
  `sonkal_member` ENUM('false', 'true') NULL,
  `id_rank` INT(10) NULL,
  `exams_date` DATE NULL,
  `dan_number` VARCHAR(9) NULL,
  `portrait` TINYTEXT NULL,
  PRIMARY KEY (`id`),
  INDEX `Rank_Person_FK` (`id_rank` ASC),
  FOREIGN KEY (`id_rank`) REFERENCES `rank` (`id` ON DELETE RESTRICT ON UPDATE CASCADE)
) ENGINE = InnoDB
AUTO_INCREMENT = 304;
```

**Table pattern:**

```sql
CREATE TABLE IF NOT EXISTS `pattern` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `name` TINYTEXT NOT NULL,
  `name_EN` TINYTEXT NULL,
  `description` TINYTEXT NOT NULL,
  `description_EN` TINYTEXT NULL,
  `id_rank` INT(10) NOT NULL,
  `diagram_id_number` INT(10) NOT NULL,
  `diagram_rows` TINYINT(3) NOT NULL,
  `diagram_columns` TINYINT(3) NOT NULL,
  PRIMARY KEY (`id`),
  INDEX `Rank_Pattern_FK` (`id_rank` ASC),
  FOREIGN KEY (`id_rank`) REFERENCES `rank` (`id` ON DELETE RESTRICT ON UPDATE CASCADE)
) ENGINE = InnoDB
```

ENGINE = InnoDB
AUTO_INCREMENT = 304;

Administration GUI for SonkalWeb

Table tool:
CREATE TABLE IF NOT EXISTS `tool` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `name` TINYTEXT NOT NULL,
  `name_EN` TINYTEXT NULL,
  `description` TINYTEXT NOT NULL,
  `description_EN` TINYTEXT NULL,
  PRIMARY KEY (`id`)
) ENGINE = InnoDB

Table event_type:
CREATE TABLE IF NOT EXISTS `event_type` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `name` TINYTEXT NOT NULL,
  `name_EN` TINYTEXT NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE = InnoDB;

Table category:
CREATE TABLE IF NOT EXISTS `category` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `description` TINYTEXT NOT NULL,
  `description_EN` TINYTEXT NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE = InnoDB

Table event:
CREATE TABLE IF NOT EXISTS `event` (
  `id` INT(10) NOT NULL AUTO_INCREMENT,
  `date` DATE NOT NULL,
  `name` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `place` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL,
  `id_event_type` INT(10) NOT NULL,
  `name_EN` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL,
  `place_EN` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL,
  `plan` ENUM('false','true') NOT NULL DEFAULT 'false',
  PRIMARY KEY (`id`),
  INDEX `id_Event_type_FK` (`id_event_type` ASC),
  CONSTRAINT `id_Event_type_FK` FOREIGN KEY (`id_event_type`) REFERENCES `event_type` (`id`) ON DELETE RESTRICT ON UPDATE CASCADE)
ENGINE = InnoDB
AUTO_INCREMENT = 425
Table person
CREATE TABLE IF NOT EXISTS `person` (  
`id` INT(10) NOT NULL AUTO_INCREMENT,  
`name` TINYTEXT CHARACTER SET 'cp1250' COLLATE `cp1250_czech_cs` NOT NULL,  
`surname` TINYTEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NOT NULL,  
`birth_date` DATE NULL,  
`team` ENUM('false','true') CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs',  
`note` TEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NULL DEFAULT NULL,  
`note_EN` TEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NULL DEFAULT NULL,  
`active` ENUM('false','true') NULL DEFAULT 'true',  
`sonkal_member` ENUM('false','true') NULL,  
`id_rank` INT(10) NULL,  
`exams_date` DATE NULL,  
`dan_number` VARCHAR(9) NULL,  
`portrait` TINYTEXT NULL,  
PRIMARY KEY (`id`) ,  
INDEX `Rank_Person_FK` (`id_rank` ASC),  
CONSTRAINT `Rank_Person_FK`  
FOREIGN KEY (`id_rank`)  
REFERENCES `rank` (`id`)  
ON DELETE RESTRICT  
ON UPDATE CASCADE)  
ENGINE = InnoDB  
AUTO_INCREMENT = 304;

Table Technique:
CREATE TABLE IF NOT EXISTS `technique` (  
`id` INT(10) NOT NULL AUTO_INCREMENT,  
`description` TINYTEXT NULL,  
`description_EN` TINYTEXT NULL,  
`name` TINYTEXT NOT NULL,  
`name_EN` TINYTEXT NULL,  
`note` TINYTEXT NULL,  
`note_EN` TINYTEXT NULL,  
`show_in_list` ENUM('true', 'false') NOT NULL,  
PRIMARY KEY (`id`) )  
ENGINE = InnoDB;

Table article:
CREATE TABLE IF NOT EXISTS `article` (  
`id` INT(10) NOT NULL AUTO_INCREMENT,  
`id_event` INT(10) NULL,  
`publication_date` DATE NOT NULL,  
`header` TINYTEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NOT NULL,  
`body` TEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NOT NULL,  
`header_EN` TINYTEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NOT NULL,  
`body_EN` TEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NOT NULL,  
`author` TINYTEXT CHARACTER SET `cp1250` COLLATE `cp1250_czech_cs` NULL,  
PRIMARY KEY (`id`) ,  
INDEX `id_Event_FK` (`id_event` ASC),  
CONSTRAINT `id_Event_FK`  
FOREIGN KEY (`id_event`)  
REFERENCES `event` (`id`)  
ON DELETE SET NULL  
ON UPDATE CASCADE)  
ENGINE = InnoDB  
AUTO_INCREMENT = 266;
### Table link:

```sql
CREATE TABLE IF NOT EXISTS `link` (
  `id` INT(10) NOT NULL AUTO_INCREMENT ,
  `name` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `www` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `contact` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `street` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `house_number` VARCHAR(10) CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL ,
  `city` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `name_EN` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `country` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `continent` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `postcode` VARCHAR(5) CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `tel` TINYTEXT NULL DEFAULT NULL ,
  `fax` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `email` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `note` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `note_EN` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL DEFAULT NULL ,
  `federation_member` ENUM('false','true') NOT NULL DEFAULT 'false' ,
  `valid` ENUM('false','true') CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  PRIMARY KEY (`id`) )
ENGINE = MyISAM
AUTO_INCREMENT = 206
```

### Table news:

```sql
CREATE TABLE IF NOT EXISTS `news` (
  `id` INT(10) NOT NULL AUTO_INCREMENT ,
  `date` DATETIME NOT NULL ,
  `expiry_date` DATE NOT NULL ,
  `text` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `text_EN` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL ,
  PRIMARY KEY (`id`) )
ENGINE = MyISAM
AUTO_INCREMENT = 206
```

### Table photo:

```sql
CREATE TABLE IF NOT EXISTS `photo` (
  `id` INT(10) NOT NULL AUTO_INCREMENT ,
  `id_event` INT(10) NULL ,
  `id_photo_type` TINYINT(3) NOT NULL ,
  `file` VARCHAR(12) CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
  `description` TEXT NULL ,
  `description_EN` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL ,
  `view_count` INT(10) NOT NULL DEFAULT '0' ,
  PRIMARY KEY (`id`),
  INDEX `id_Event_Photo_FK` (`id_event` ASC),
  INDEX `id_Photo_theme_FK` (`id_photo_type` ASC),
CONSTRAINT `id_Event_Photo_FK`
  FOREIGN KEY (`id_event`) REFERENCES `event` (`id`) ON DELETE RESTRICT ON UPDATE CASCADE,
CONSTRAINT `id_Photo_theme_FK`
  FOREIGN KEY (`id_photo_type`) REFERENCES `photo_type` (`id`) ON DELETE RESTRICT ON UPDATE CASCADE)
ENGINE = InnoDB
AUTO_INCREMENT = 29722;
```
Table video:

CREATE TABLE IF NOT EXISTS `video` ( 
`id` INT(10) NOT NULL AUTO_INCREMENT ,
`id_event` INT(10) NULL ,
`file` TINYTEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NOT NULL ,
`description` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL ,
`description_EN` TEXT CHARACTER SET 'cp1250' COLLATE 'cp1250_czech_cs' NULL ,
`view_count` INT(10) NOT NULL DEFAULT '0' ,
PRIMARY KEY (`id`) ,
INDEX `id_Event_Video_FK` (`id_event` ASC) ,
CONSTRAINT `id_Event_Video_FK`
 FOREIGN KEY (`id_event`) 
 REFERENCES `event` (`id`) 
 ON DELETE RESTRICT 
 ON UPDATE CASCADE) 
ENGINE = InnoDB 
AUTO_INCREMENT = 331;

4.2.1 Example of Information in Database

The database we have is big, so in order to understand them better, I’m about to show a simple example of some information stored in the database:

Rank

<table>
<thead>
<tr>
<th>id</th>
<th>colour</th>
<th>colour_EN</th>
<th>rank_name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>bílý</td>
<td>white</td>
<td>first</td>
</tr>
<tr>
<td>2</td>
<td>žlutý</td>
<td>yellow</td>
<td>second</td>
</tr>
</tbody>
</table>

Pattern

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>name_EN</th>
<th>description</th>
<th>description_EN</th>
<th>id_rank</th>
<th>diagram_id_number</th>
<th>diagram_rows</th>
<th>diagram_columns</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>Saúl</td>
<td>Sajurji</td>
<td>nbyrjorisesies</td>
<td>White belt</td>
<td>1</td>
<td>12</td>
<td>127</td>
<td>123</td>
</tr>
<tr>
<td>5</td>
<td>Dan-Gun</td>
<td>Dan-Gun</td>
<td>erga se volutatem</td>
<td>Yellow Belt</td>
<td>2</td>
<td>332</td>
<td>12</td>
<td>23</td>
</tr>
</tbody>
</table>

Tool

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>name_EN</th>
<th>description</th>
<th>description_EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tool</td>
<td>Tool_EN</td>
<td>La primera herramienta</td>
<td>The first Tool</td>
</tr>
<tr>
<td>2</td>
<td>nteres quac fides</td>
<td>first</td>
<td>Auximumque proficiscitur</td>
<td>the first tool</td>
</tr>
<tr>
<td>3</td>
<td>essel data</td>
<td>second</td>
<td>Curio summa omnium voluntate Iguvium recipit</td>
<td>the second tool</td>
</tr>
</tbody>
</table>

Event Type

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>name_EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>inicio tumultus</td>
<td>tournamet</td>
</tr>
<tr>
<td>5</td>
<td>inimicis defendant</td>
<td>competition</td>
</tr>
</tbody>
</table>
### Administration GUI for SonkalWeb

#### Category

<table>
<thead>
<tr>
<th>id</th>
<th>description</th>
<th>description_EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>T?etlí</td>
<td>Third</td>
</tr>
<tr>
<td>4</td>
<td>?tvrtá</td>
<td>Fourth</td>
</tr>
</tbody>
</table>

#### Technique

<table>
<thead>
<tr>
<th>id</th>
<th>description</th>
<th>name</th>
<th>description_EN</th>
<th>name_EN</th>
<th>note</th>
<th>note_EN</th>
<th>show_in_list</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>technic of Saúl Jungi</td>
<td>Gurnun so ap juomuk kaunde baro jugi</td>
<td>Gurnun so ap juomuk kaunde baro jugi</td>
<td>NULL</td>
<td>NULL</td>
<td>true</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>technic of Saúl Jungi</td>
<td>Gurnun so ap juomuk kaunde baro jugi</td>
<td>Gurnun so ap juomuk kaunde baro jugi</td>
<td>Right walking stance, middle punch with front fist...</td>
<td>Right walking stance, middle punch with front fist...</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### Person

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>username</th>
<th>password</th>
<th>surname</th>
<th>birth_date</th>
<th>team</th>
<th>note</th>
<th>note_EN</th>
<th>active</th>
<th>sonkal_member</th>
<th>id_rank</th>
<th>exams_date</th>
<th>dan_number</th>
<th>portrait</th>
</tr>
</thead>
</table>

#### Event

<table>
<thead>
<tr>
<th>date</th>
<th>name</th>
<th>place</th>
<th>id_event_type</th>
<th>name_EN</th>
<th>place_EN</th>
<th>plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>2009-07-04</td>
<td>Ariminum cum ea</td>
<td>Praha, Czech Republic</td>
<td>4</td>
<td>World Tournament</td>
<td>Praha, Czech Republic</td>
<td>true</td>
</tr>
<tr>
<td>2009-06-24</td>
<td>Accoptis mandulis Roscius</td>
<td>Brno, Czech Republic</td>
<td>5</td>
<td>World Competition</td>
<td>Brno, Czech Republic</td>
<td>false</td>
</tr>
</tbody>
</table>

#### Article

<table>
<thead>
<tr>
<th>id</th>
<th>id_event</th>
<th>publication_date</th>
<th>header</th>
<th>body</th>
<th>header_EN</th>
<th>body_EN</th>
<th>author</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>2009-06-06</td>
<td>Cum Caesaris Cepuran porvent biquc consulse. Pompeianisque invent; positalta Caesaris renuntial ... hasco erat a &amp; Caesar</td>
<td>The competition start soon</td>
<td>La nueva competicion empezara el dia 4 de julio...</td>
<td>Victoria Lopez</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2009-06-16</td>
<td>Ill re de terbnt respondent scriptaque ad eum ma...</td>
<td>The new Pavilion brno ready.</td>
<td>The pavilion was inaugurated on 12-4-2009. Tomorro...</td>
<td>Victoria Lopez</td>
<td></td>
</tr>
</tbody>
</table>

#### Link

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>www</th>
<th>contact</th>
<th>street</th>
<th>house_number</th>
<th>city</th>
<th>name_EN</th>
<th>country</th>
<th>country_EN</th>
<th>continent</th>
<th>continent_EN</th>
<th>postcode</th>
<th>tel</th>
</tr>
</thead>
<tbody>
<tr>
<td>206</td>
<td>Sonkal Web</td>
<td><a href="http://www.sonkal.cz/">http://www.sonkal.cz/</a></td>
<td>c/ Porta Artois</td>
<td>13</td>
<td>Praga</td>
<td>Sonkal Web</td>
<td>Czech Republic</td>
<td>Europe</td>
<td>Europe</td>
<td>48032</td>
<td>983343012</td>
<td></td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>Budovskys</td>
<td><a href="http://www.budovskys.com/">http://www.budovskys.com/</a></td>
<td><a href="mailto:staff@budovskys.com">staff@budovskys.com</a></td>
<td>Bencarlo, 36</td>
<td>Praga</td>
<td>Budovskys</td>
<td>Czech Republic</td>
<td>Czech Republic</td>
<td>Europe</td>
<td>30283</td>
<td>77327384</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>fax</th>
<th>email</th>
<th>note</th>
<th>note_EN</th>
<th>federation_member</th>
<th>valid</th>
</tr>
</thead>
<tbody>
<tr>
<td>983728201</td>
<td><a href="mailto:sonkal@teakwondo.cz">sonkal@teakwondo.cz</a></td>
<td>NULL</td>
<td>NULL</td>
<td>true</td>
<td>true</td>
</tr>
<tr>
<td>672732222</td>
<td><a href="mailto:staff@budovskys.com">staff@budovskys.com</a></td>
<td>Web of Judo Girls</td>
<td>true</td>
<td>false</td>
<td></td>
</tr>
</tbody>
</table>

#### News

<table>
<thead>
<tr>
<th>id</th>
<th>date</th>
<th>expiry_date</th>
<th>text</th>
<th>text_EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2009-06-20 18:25:20</td>
<td>2009-06-24</td>
<td>Prohledněte si 170 fotografii z naší návštěvy ve V...</td>
<td>Look at 170 photos from visit of Sonkal's girls in...</td>
</tr>
<tr>
<td>2</td>
<td>2009-06-11 18:25:46</td>
<td>2009-06-17</td>
<td>Sobotní dopoledne 5. června patřilo focení nalich ...</td>
<td>Saturday 5th June morning belonged to shooting Son...</td>
</tr>
</tbody>
</table>
### Administration GUI for SonkalWeb

#### Photo

<table>
<thead>
<tr>
<th>id</th>
<th>id_event</th>
<th>id_photo_type</th>
<th>file</th>
<th>description</th>
<th>description_EN</th>
<th>view_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td><a href="http://www.r">http://www.r</a></td>
<td>mitterat; quae si facesset, Pompeium in.</td>
<td>world tournament team</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>3</td>
<td><a href="http://www.g">http://www.g</a></td>
<td>at. Illi re deliberata respondent scr</td>
<td>Photo off victoria</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Video

<table>
<thead>
<tr>
<th>id</th>
<th>id_event</th>
<th>file</th>
<th>description</th>
<th>description_EN</th>
<th>view_count</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>Final_tournament.avi</td>
<td>doloreatat, id sum logati veniant, quaeque impera...</td>
<td>Video about the final of the tournament</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>SONKAL_AND_NATIONAL_TEAM.avi</td>
<td>se cupidissime facturos pollicentur. Milit</td>
<td>presentation of the sonkal team</td>
<td>0</td>
</tr>
</tbody>
</table>

#### Person_photo

<table>
<thead>
<tr>
<th>id_photo</th>
<th>id_person</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Person_video

<table>
<thead>
<tr>
<th>id_video</th>
<th>id_person</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

#### Photo_type

<table>
<thead>
<tr>
<th>id</th>
<th>type</th>
<th>type_EN</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>avi</td>
<td>avi</td>
</tr>
<tr>
<td>4</td>
<td>mpeg</td>
<td>mpeg</td>
</tr>
</tbody>
</table>
5. Software Implementation

5.1 Coding

In this chapter I will describe some interesting implementation details of web application.

This application is implemented and coded in JSP, PHPMyAdmin and localhost server in order to visit the web page and test it, so we send a request through http protocol requesting the information in the web server (MAMP Server) which is used for manage the website locally.

For each package I created the correspondent web pages, which each of them has its own correspondent function into the package in order to manage the data.

I created a standard folder structure in my application. The structure is like this:

- \src /java /"table_name"----> contains all java packages, different for each table
- \web ----> contains cascading style and the web pages commons for all the tables
- \web /"table_name" ----> contains the pages different for each table

Let’s imagine that an user wants to insert a register. The application will work like this: The server responds by sending a form that the user has to fill. Once completed, the user will have to click on the button and the application will send the form through the browser to the server where it will be stored.
I will explain with further detail each menu option to list, create, edit or delete a registry entry (this example uses the table rank) (in this example uses the table rank)

**List Rank**

“List rank” lists all the registry entries stored in the database. Now, the user have to click “List” button and then the server, as soon as receiving this request, it will send through the browser the list of all the stored rank.

The function is this:

```java
public static String getRanks() { //save all the registers from the table in a LinkedList
    LinkedList<Rank> RanksList = new LinkedList<Rank>();
    try {
        Class.forName(className).newInstance();
        Connection conexion = DriverManager.getConnection(host + dataBase, user, password);
        Statement st = conexion.createStatement();
        ResultSet rs = st.executeQuery("select * from rank"); //select all registers from DB
        while (rs.next()) // while exist more register I save in ranklist
        {
            Rank rank = new Rank();
            rank.setId(rs.getInt("id"));
            rank.setColour(rs.getString("colour")); //the name between the "" is the name of the column
            rank.setColour_en(rs.getString("colour_en"));
            rank.setRank_name(rs.getString("rank_name"));
            RanksList.add(rank); //add rank to the list
        }
        rs.close(); //close query sql
        st.close(); //close stament
        conexion.close(); //close conexion with DB
    } catch (Exception e) {
        e.printStackTrace();
    }
    return rankResult(RanksList);
}
```

**New rank**

New rank consists in a function where a new rank is created. Also, a new form is created too, in which the user will have to enter the data that will be checked with java later on. Finally, once the user has fill the form and the data has been checked, the new data is stored into the database.

```java
public static void rankCreate(String colour, String colour_EN, String rank_name) {
    //si apanyamos el rank_create.jsp esta funcion no nos hace falta para nada.
    try {
        if (colour != null || colour_EN != null || rank_name != null || !"".equals(colour) || !"".equals(colour_EN)
        || !"".equals(rank_name)) {
            Class.forName(className).newInstance();
            Connection conexion = DriverManager.getConnection(host + dataBase, user, password);
            Statement st = conexion.createStatement();
            st.executeUpdate("INSERT INTO rank (id , colour , colour_EN , rank_name) +
```
Administration GUI for SonkalWeb

" VALUES ( NULL, '" + colour + "," + colour_EN + "," + rank_name + ")");
}
// close conexion with DB
}
} catch (Exception e) {
  e.printStackTrace();
}
}

Edit Rank

Edit rank is a function where the user edits an existing rank. The user introduces the parameters in a form to search for it and it is sent to the server through the browser. Then, the server will respond with all the records similar to the search entered before. The user chooses the registry entry to edit, and he will enter the new values through a form. Once the values are checked, these are sent to the server and are stored in the database.

The function responsible for this work would be this:

```java
public static void rankUpdate(int id, String newColour, String newColour_EN, String newRank_name) {
  try {
    Class.forName(className).newInstance();
    Connection conn = DriverManager.getConnection(host + dataBase, user, password);
    Statement st = conn.createStatement();
    ResultSet rst = st.executeQuery("select * from rank where id = "+id+";  
    rst.next();
    Rank rank = new Rank();
    rank.setColour(rst.getString("colour"));
    rank.setColour_en(rst.getString("colour_en"));
    rank.setRank_name(rst.getString("rank_name"));

    if ("".equals(newColour) || newColour == null) { //if newColour is empty, it save the old value
      newColour = rank.colour;
    }
    if (newColour_EN.equals("") || newColour_EN == null) {//if newColour_EN is empty, it save the old value
      newColour_EN = rank.colour;
    }
    if (newRank_name.equals("") || newRank_name == null) { //if newRank_Name is empty, it save the old value
      newRank_name = rank.rank_name;
    }
    rst.close();
    st.close();
    conn.close();
  }catch (Exception e) {
    e.printStackTrace();
  }
}
```
Administration GUI for SonkalWeb

// create Statement
Statement stmt = con.createStatement();
// create the SQL query

stmt.executeUpdate( // modifying the parameters to new values.
    "UPDATE rank " +
    "SET colour = " + newColour + "," +
    "colour_EN = " + newColour_EN + "," +
    "rank_name = " + newRank_name + "
    "WHERE id = " + id + ");

stmt.close();
// close conexion
con.close();
}

} catch (Exception e) {
    e.printStackTrace();
}

Delete Rank

"Delete Rank" is responsible for deleting a register or multiple registers in the database. It works in a similar way to edit. Through a form, the user enters the parameters to find the record or records to delete. Through the button "Accept", the server receives these queries and returns the corresponding list of rank. The user through some checkboxes, selects the records to delete. The server receives this list and then deletes the selected registers and updates the database.

The function responsible for deleting the records would be this:

public static String rankDelete(String[] id, boolean confirm) {
    String result = "";
    try {
        Class.forName(className).newInstance();
        Connection conn = DriverManager.getConnection(host + database, user, password);

        if (!confirm) { //list the registers
            result = "<h2>Are this the ranks that you want to delete?</h2>" +
                "<table >" +
                "<tr >" +
                "<td id </td>" +
                "<td color </td>" +
                "<td color_en </td>" +
                "<td rank_name</td>" +
                "</tr>";

            int temp;

            for (int i = 0; i < id.length; i++) { //read the array
                Statement st = conn.createStatement();
                temp = Integer.parseInt(id[i]);

                result = result + "<tr >" +
                    "<td id </td>" +
                    "<td color </td>" +
                    "<td color_en </td>" +
                    "<td rank_name</td>" +
                    "</tr>");
            }
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
    return result;
}
ResultSet rs = st.executeQuery("SELECT * FROM rank WHERE id = " + temp + ""); //search the register which id is the same that id[i]
rs.next();

Rank rank = new Rank(); //save the register in the rankList
rank.setId(rs.getInt("id"));
rank.setColour(rs.getString("colour"));
rank.setColour_en(rs.getString("colour_en"));
rank.setRank_name(rs.getString("rank_name"));

result = result + "<tr>"; //save the register in the string result
result = result + "<td>" + rank.getId() + "</td>";
result = result + "<td>" + rank.getColour() + "</td>";
result = result + "<td>" + rank.getColour_en() + "</td>";
result = result + "<td>" + rank.getRank_name() + "</td>";
result = result + "</tr>";

rs.close(); //close query sql
st.close(); //close statement
}

result = result + /*form for accept or edit the list of registers to delete*/
"</table>" + 
"<form method="post" action="deleteRank.jsp" name="ConfirmDeleteRank">" + 
"<input type="submit" name="accept" value="Accept">" + 
"<input type="submit" name="edit" value="Edit">" + 
"</form>";

conn.close(); //close connection with DB

Class.forName(className).newInstance();
Connection connect = DriverManager.getConnection(host + dataBase, user, password);

Statement stt = connect.createStatement();

if (confirm) { //if the list of registers is correct, confirm is true
for (int i = 0; i < id.length; i++) { //read the array
stt.executeUpdate("DELETE FROM rank " + 
"WHERE id = " + id[i] + "");
}
result = "Save Successful";
}

stt.close();
connect.close();
} catch (Exception e) {
e.printStackTrace();
}
return result;
}
6. Realisation

In this part I’m going to explain how the GUI works from the point of view of an authorised user.

This index shows the different tables, and the menus, where the buttons are to list the tables, create new data, edit, or delete data.

Login

This page is the security page, where only the administrator and editor can enter. Here, we must write the username and the password to authorize us to enter this section.
List Ranks

With this page the user can know what the data in the database is.

New Rank page

This page shows a form where the user can enter the data of the new rank. When all the parameters are written, the user may click in the “create button” and the new rank is saved in the database.
Administration GUI for SonkalWeb

**Edit Rank page**

When a user wants to change some parameters in the database, it can be changed with a form to search the registry, choosing between different options...

and finally entering the news values to the registry.
User may click in the button “edit” to perform the operation and save the new data in the database.

**Delete Rank page**
Administration GUI for SonkalWeb

This page shows the list of all the ranks with a checkbox, one per each registry. The user can choose all the registries that want to delete. This page redirects to another.

... to accept the election. In order to finish the operation, the user may click in the Accept button or the user may click in the Edit button in order to choose different registers.
7 Conclusion

When I started planning the project, its schedules, etc... It seemed to have a lot of time to do my work development and implementation, but when I really started, I realize that I was wrong as I didn’t had much time.

To have a complete control over the whole structure of the website I was about to develop, a part from seeing several similar pages, the challenging part was learning to use software I never used and understanding HTM and JSP languages. I’ve never done anything like this before with JSP and my knowledge and design in HTML were very limited.

This allowed me to understand how a web page works. The creation of the structure of the website began testing it a few times, and finally getting into the definitive.

For each CRUD menu item of the web page, I must find lots of information, testing it every time I changed it, slowly, until the project worked in each of the steps I’ve made.

I hope my project can teach people who read it, or at least give ideas that are clearly enough to understand.

I’m planning to expand this project and improve it as far as my possibilities and capabilities allow me to do it, learning and evolving into this branch of computer science.
8. Bibliography

- Tutorials of SQL and JSP
  - http://www.htmlpoint.com/sql/sql_02.htm
  - http://www.jsptut.com/

- Introduction to servers:
  - http://geneura.ugr.es/~jmerelo/JSP/

- GlassFish page
  - https://glassfish.dev.java.net/

- Some pages which help me with the code:
  - http://www.nabble.com/Tomcat-derby-JSP-%22java.sql.SQLException:-No-suitable-driver%22-td20776666.html

- Some pages which help me with the diagrams:
  - http://www.mitecnologico.com/Main/DiagramasEntidadRelacionER
  - http://www.visualcase.com/tutorials/class-diagram.htm

- Software:
  - http://tomcat.apache.org/
  - http://www.eclipse.org/
  - http://www.netbeans.org/