June 2019

UNIVERSITÀ DEGLI STUDI DI TORINO

TECNOLOGIE WEB

Beatriz Martínez Andelo

1 Introduction

Nowadays, the Language Exchange Social is a well-known concept used to describe those pubs in which people gather to improve their language in a natural and free way, however due to the scarcity of speakers of other languages, the existence of these pubs is reduced. For that the purpose of this web page is to amplify the means of interaction between these people.

The web page is a social network in which you have registered speakers with whom you can interact in order to improve a language. These people can be selected by your preferences of amusements and/or the language that you want to talk. This main goal of the web page is to let people get in touch and lets them interact by Skype or other means if the user lets the information on the description of his profile.

The site is composed of a main menu in which the user can view his favorite speakers. He can also select each of them to see more information about it, such as his score according to other users, the number of votes, his available time, the languages which he speaks, the topics he likes to talk about and a brief description of himself. The user have the option to give his Skype to let people contact you and start to practise, but this option only appears if you mark a contact as favourite.

The user can change his information whenever they want, as well as the languages he masters or the topics he want to talk about.

To finish he will also have a navigation bar where he can search for new users to mark as a favorite.

2 Project structuring

The project is structured in the web pages, common files, controller, css, images and the model.

The common directory contains two directories. The first, ”components” contains all the necessary functions to create the pages and common sections as timepicker, dragging or rating with their respectives css ans js. The second one, ”defines” contains all the constraints for the images. Besides, it contains the navbar and the footer which appears in every page.

The controller directory contains all the AJAX calls, and the scripts to manipulate the DOM.

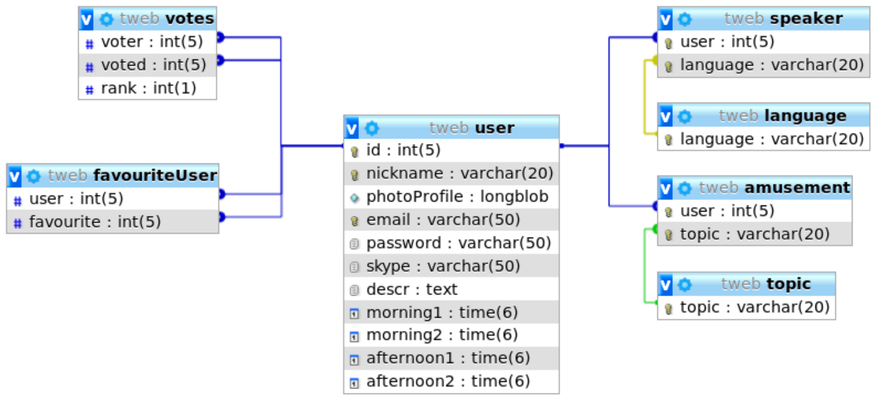
The css directory contains some styles for every page, but on the project we can find inline styles.

The images directory contains all the images of languages, topics or buttons.

At last, the model directory contains all the calls to the database and here is where we call the functions defined on common directory, to send the code to the client.

3 Database

The database has the following tables:



4 Responsive-Web

For resolving the cross-validation problem the page was made responsive using flex styles. The page have a grid structure which adapts to the width of the screen reducing or increasing his number of columns until it reaches three. Besides, the measures of the pages are in rem, a measure that adapts too to the screen dimension.

5 Functionality

In this section, the operation of each project page and component will be explained.

5.1 Login/logout

This is the initial screen of the project. In it you can find a modal trigger button, which when pressed displays the Sing In section and a button which directs you to the Sing Up section. In the Sign In section are two input fields for the email and the password of a user.
When you submit, the form calls the function checkLogin on controller directory. It makes an AJAX post call to checkingLogin.php on the model directory, where the database is consulted if that email and that password belong to a single user. If it is true, init a session with the id, nickname and photoProfile of the user and response with a Success. Else it responses with Invalid.
When the response arrives, displays a message letting the user know if it was success operation or not. In the case of a success operation you are redirected to home page.

5.2 Sign Up

A form can be seem in which if the validation requirements are not met, a message appears down the input field asking for its correction. When the form is submited, the AJAX function signUp is called in the controller. Here it is verified again that the form is valid and if there is no one with the same nickname and/or email, a new user is registered in the database and the data stored in the login is stored in session. When the call is finished, the user will be redirected to the home page.

5.3 Home

On this page, the user's favorite speakers are retrieved through an embedded fetch function to the recoverFavourites model. Each of the user panels has a button that redirects you to the page seeProfile, passing the nickname of the user pressed by the GET method of the URL. Here the navbar and the footer appears for the first time.

5.4 User profiles

The content of this page varies depending on who is viewing the page and the variable nickname of the URL. This serves to: display the user's own profile and have the option to edit your content, view the details of a user, add or remove it from favorites and finally add, change or delete your vote. In case of being a favorite user who sees the page and it has a valid Skype, they could converse through the application from the page. All this is managed by the model.

As in home, this page retrieves the data through an embedded function fetch in which the seeProfile model is used.

5.5 Edit Profile

The user profile can be changed or only the password at the same time. Below are explained the two cases:

5.5.1 Change Profile

On this page as in the previous ones, the data is collected through a embedded function fetch.
This page is accessed by clicking the edit button within the seeProfile page when the nickname of the URL is equal to your nickname. Then the fetch function uses the model editProfileData. It can be edited everything except your email and your password in this section.
When you submit the form, the function editProfile of the controller is called. It retrieves all the inputs, the draggable containers and the timepickers. A control bit is also sent to control if the user has loaded a profile photo. This controller used the model editProfilePost.
On the model you update the data of the user, but if the bit of control of the photoProfile is off, it does not update the photoProfile. All elements belonging to the identifier of the current user are dropped from the amusement and speaker tables. Then all the edited topics and languages are inserted.

Votes There are two functions on the voteUser controller:

- voteUser: This function calls to voteUser model, where a vote is maked or updated, depending if it existed previously.
- removeUser: This other function calls to removeVoteUser model, where if a vote existed, it deletes it.

When any of these finishes you should update the value of opinions by jquery.

5.5.2 Change Password

This section can be reached from the navbar, displaying the dropdown and selecting "Change Password". When entering you can see a container with an input in which you must enter the user's current password. If you submit the form a request is made by the confirmPassword function of the changePassword controller. On the model is checked if it is its real password. If this is not correct, it does not do anything, but if it is correct the input and the button are blocked and a lower container is filled which allows you to change the password.

If this second container is submitted, the changePassword function of the same controller is called, which calls the changePassword model where the password is updated.

5.6 Navbar

Here we have a metasearch engine where every time you submit, you call the userProfileSearch controller where the data will be sent to the search model. When you retrieve it, the data container will be hidden and a container below with the content of the query will be displayed. The query changes in the model according to the values selected by means of "if" statements.

6 Features

6.1 Usability

It has been used on the editProfile page and on the signUp page, containers to give feedback to the users. Besides, the content is organised with a grid design and follows the guides of style.

6.2 Interaction

In order to complete this section, it has been decided to implement a drag and drop components for choosing languages or topics of a user. It can be found on singUp or editProfile pages. The files that composed this component are: "components/dragging.php", "components/css/draging.css" and "components/js/dragging.js".

dragging.php This file defines the functions that displays the draggable containers. The draggable section is composed by two draggable containers, the first belongs to the user and the second one is calculated by disjoint operations of the user elements with the complete set. Besides, there is a function for converting all the images to draggable images and recover the constraint name as identifier of the image to insert later on the database. The type is set as well for behavior reasons, that are explained on "components/js/draggin.js".

dragging.js These functions control the behaviour of the dragging elements. First, to understand what is going to be explained below, the domain of the type can be "language" or "topic", being this the type of draggable containers and images. The function drag is used to make an image draggable, and the type defines what kind of image is. The function drop is used to make a container droppable, and as before, the type defines what kind of container is. The type is used to not allow an image with a different kind of container to be dropped on it. It can be done checking on the event of drop if the image has the same type that the container, else the event do not do anything. The dropImage function allows to exchange positions of images either on the same container or in his affiliate container.

6.3 Database queries

An example of this is the navbar. It has been explained before, but now it will look more in depth. It is controlled by if statements. If nothing was selected it retrieves all the users. If language was selected and topic not, the query results as:

```
"SELECT u.id, u.nickname, u.photoProfile
FROM user u, speaker l
WHERE u.id=l.user AND l.language=[language]"
```

If topic was selected and language not, the query results as:

```
"SELECT u.id, u.nickname, u.photoProfile
FROM user u, amusement t
WHERE u.id=t.user AND t.topic=[topic]"
```

If everything was selected , the query results as:

```
"SELECT u.id, u.nickname, u.photoProfile
FROM user u, speaker l, amusement t
WHERE u.id=t.user AND l.language=[language] AND t.topic=[topic]"
```

When any of these queries finished, the search of the languages and topics belonging to the recovered users must be carried out.

Finally, the user panels are shown as on the main page.

The search can be done from all the pages that have navbar. Every page with navbar have a own controller, but all of them calls the same model, search.

6.4 Data validation

To complete this section, a jquery library called jquery-validation was used to simplify the data validation process in the frontend. Also in the backend its used this library to check the form is well formed in respect to the rules and if and only if it is valid make the AJAX call.

6.5 Security

For avoiding "SQL Injection" I used simple quotes around the paramethers. Besides, the passwords are encrypted by md5 hash function on the database. The security would be complete when the project will be deploy with an https protocol.