PROJET WEB

Erasmus – A life-changing experience

Ana-Maria OPRESCU

SUMMARY

1.Introduction
2. Functional framework
   2.1 Use case
      2.1.1. Create an account
      2.1.2. Log in
      2.1.3. Management of the account
      2.1.4. Log out
   2.2. Functional constraints
   2.3. Database
3. Logic architecture
   3.1.Structure
   3.2. Database
      3.2.1 SGBD
      3.2.2 MCD
      3.2.3 Triggers
4. Deployment architecture
5. At the end of the day

1. Introduction

Aim of the project:

ERASMUS is an association that tries to reunite all the Erasmus students across the Europe. It also tries to help students collect all the information they need in order to obtain a real image about what Erasmus means.

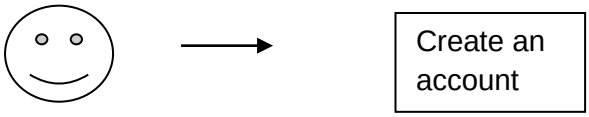
2. Functional framework

2.1. Use case

2.1.1. Create an account

This use case describes the part of becoming a member (user) of the site. A person can become a member only after they have filled the form available on the site.

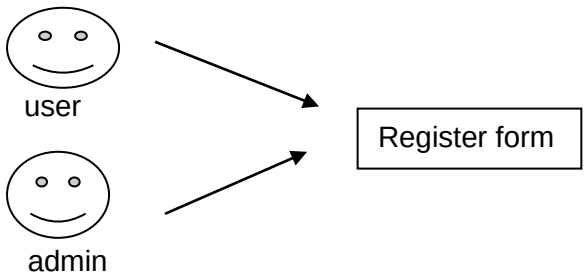
To enjoy at maximum the functionalities of this site, it is mandatory to become a member. On the other hand, you do not have to be already an Erasmus student to become a member of the site.



Note: It is only one admin, there is no “create account” for him, this part is only for the members.

2.1.2. Log in

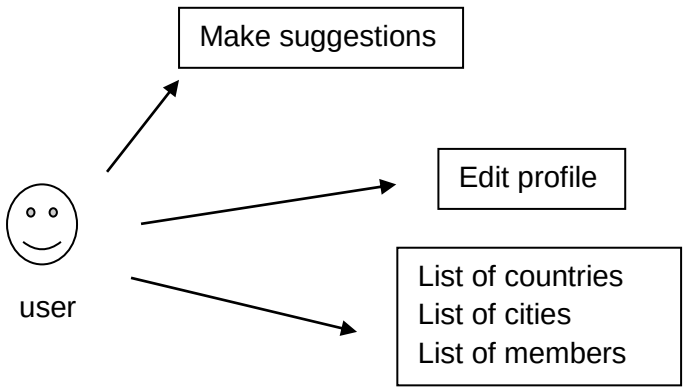
This use case describes the connection of member or admin , both of them use the same form.



2.1.3. Mangement of the account

This use case describes all the activities that a member/admin can do on the site. I will present it in 2 parts, admin view and member view.

- Member view:



What I wanted to do:

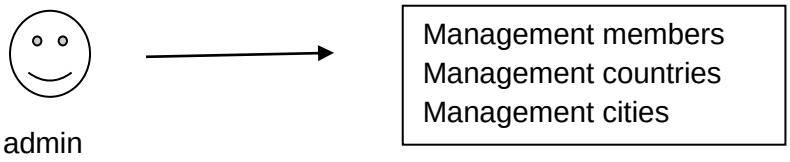
- a page with each part country, city,member;
- having the possibility to edit their account (account including a picture + being specified if he/she is in the moment an Erasmus student and the place);
- having the possibility to search countries/cities in different ways: alphabetic, by popularity (it should be calculated taking into account the numbers of Erasmus a country have per year) etc.
- having the possibility to search members by name, country/city of origin and country/city of destination (where they are having their Erasmus mobility);
- having the possibility to find from a given country, all the cities of this country where Erasmus students are;
- having the possibility to make a suggestion about a new city/country where are Erasmus students;
- having the possibility see all the information about a country (name, language spoken,capital, currency, some important things/tips to know about that country and some phrases you need to know in that language) and about a city (name and geographical location.

What I succeeded to do:

- a page with each part country, city,member;
- can search a country, city, member but only in a alphabetic ordre;
- can make a suggestion to add a new country/city;

- can see the information about city/country/member, but the design it is not the one expected;
- there is a list for such category country/city/member;

Admin view:



What I wanted to do:

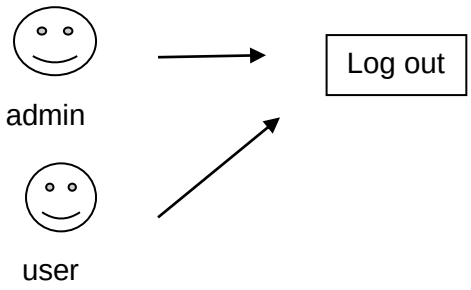
- possibility to edit, add, delete, search a country;
- possibility to edit, add, search a city;
- possibility to delete a member.

What I succeeded to do:

- possibility to edit,add, delete, search a country;
- possibility to edit, delete, search city;
- possibility to delete a member.

2.1.4. Log out

This part describes the action “log out”, which is the same for both user and admin.Cette étape concerne simplement la déconnexion du système du membre.



2.2. Functional constraints

Security

This part is reserved to explain the measures I did to guarantee the security of the site

What i did :

- avoid the insert risks by using functions that convert the all the information received from the user in characters ;
- the important information of the members such as the password, are not visible, they are encrypted;

What should I have done in addition:

- use token and secure cookies for having a better level of security of the information.

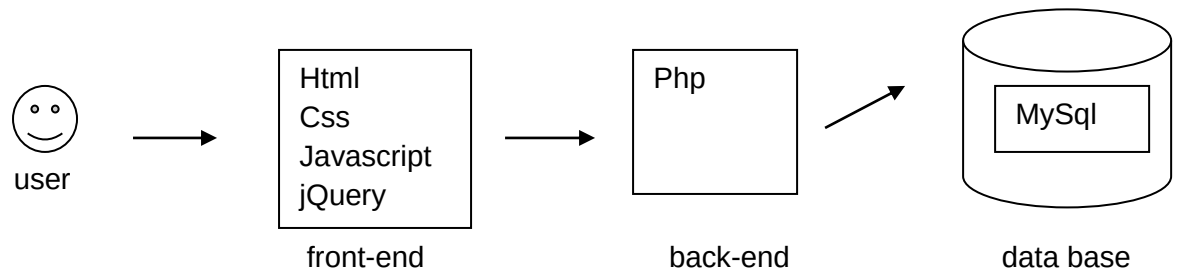
2.3. Database

All the data is saved in a relational database (SQL). As it comes for back ups, I did:

- I had a local data base, used for the tests of the site;
- The code is also saved in a git folder;
  gitHub link : https://github.com/AnaOprescu/projetWeb.git

3. Logic architecture

3.1.Structure



I have tried to respect a view-functions structure of presenting my code.

Fron-end :

I chose to realise this part using HTML , CSS, javascript and jquery in order to have a site more interactive and easy to manipulate. I also use some designs Materialize CSS and some functionalities BOOTSTRAP.

Back-end:

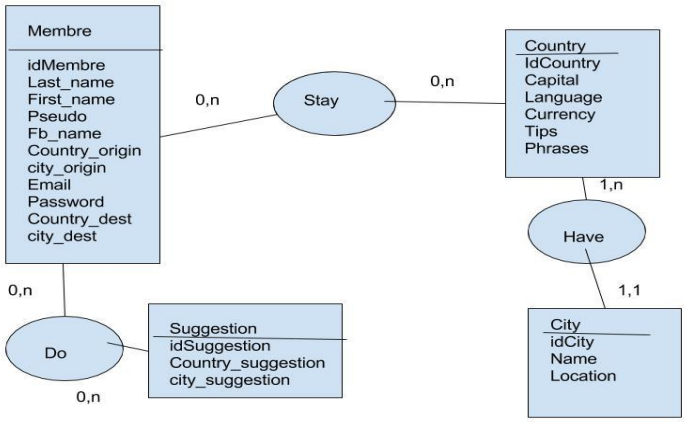
I chose PHP because i found it more easy to understand and use, especially for a starter.

3.2. Database

3.2.1 SGBD

I chose as a database MySql and as a tool i make all my test via phpMyAdmin.

3.2.2 MCD



3.2.3 Triggers

Country

- **Delete_city_on_country()**
  Make sure that once you have deleted the country, the city will be deleted also

Suggestion

- **One_display()**
  Check if the city suggested by the member exists or not already in the database

4. Deployment architecture

I chose as a way of deployement OpenShift, once because it is more easy to explore and also because of his option MySql for the database and second because I could not realize the connection with PostgreSql.

URL of my site: http://lifechanging-repertoire.rhcloud.com/modules/index.php

5. At the end of the day

Why i did not succeed and want to do the in the future?

- I did not chose to right way to start the project, personally I start with the design part and think about how should look instead of thinking about its functionalities and what I want the site to do;
- Should have started first testing all the options, looking for some reviews and then chosing the database + deployement place that best fits me, not letting this the second step and losing precious time just for trying to connect at sites;
- Lack of time;
- Problems with my computer to connect for both Heroku and Openshift and I arrived to commit all my files online from another computer, which made me lose more time.