Objectives

In these trying times more than ever, being able to keep ourselves healthy and active without leaving the comfort of our homes is essential, although is not always easy. This project might be an additional handy tool for making us momentarily forget the urge to go outside by making our indoor life a little bit more pleasant. RemyTutor is a remote learning platform for anything people might want to teach and transmit to other learning enthusiasts. Ranging from school subjects, like math or history, to more practical things, like yoga or a musical instrument, just find the right teacher and book a lesson!

When it comes to teaching, finding the right person, or even finding someone, can be a challenging task and might often feel like a Russian roulette. Therefore, our goal is to provide an online safe space to bring supply and demand together in a structured way, to ease the flow of teaching and improve its overall quality.

Main functionalities

In order to meet the objectives, RemyTutor offers the following functionalities:

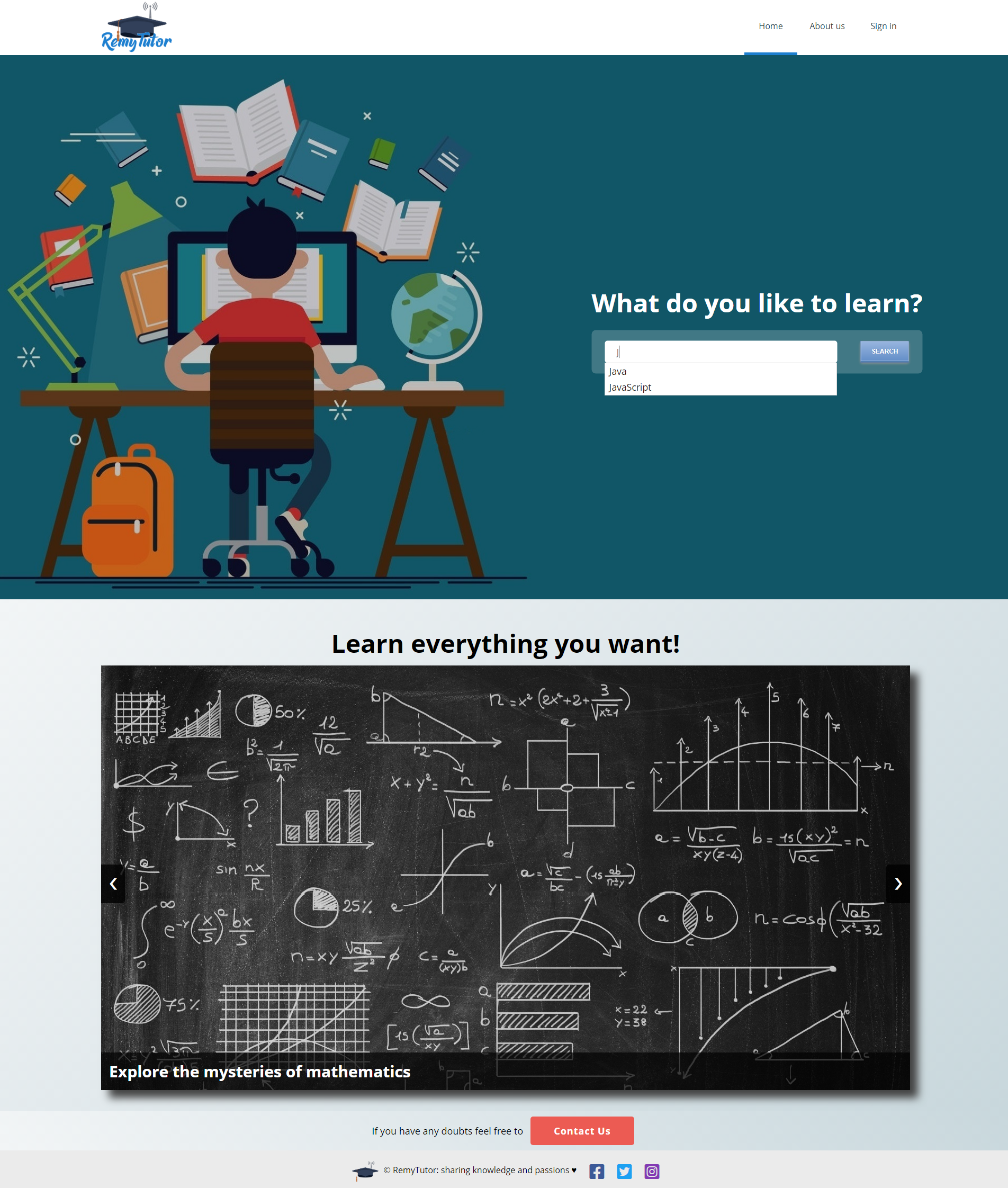
* Searching the subjects of interest;
* Choosing the teacher based on their rating and price;
* Managing personal profile;
* Asking/proposing the lessons with the chat included in the application;
* Rating a teacher after a lesson;
* Payment from the student to the teacher.

Design choices

We have decided to start our web application from scratch and not to use any template of libraries for the interface. This allows us to have full control and understanding about every parts of the web application. Instead, we have used font-awesome library to add icons on some input forms in order to meet the users’ habits. Since we don’t have massive use of JavaScript, we have limited the use of jQuery for the AJAX, so that our application is compatible among different browsers.

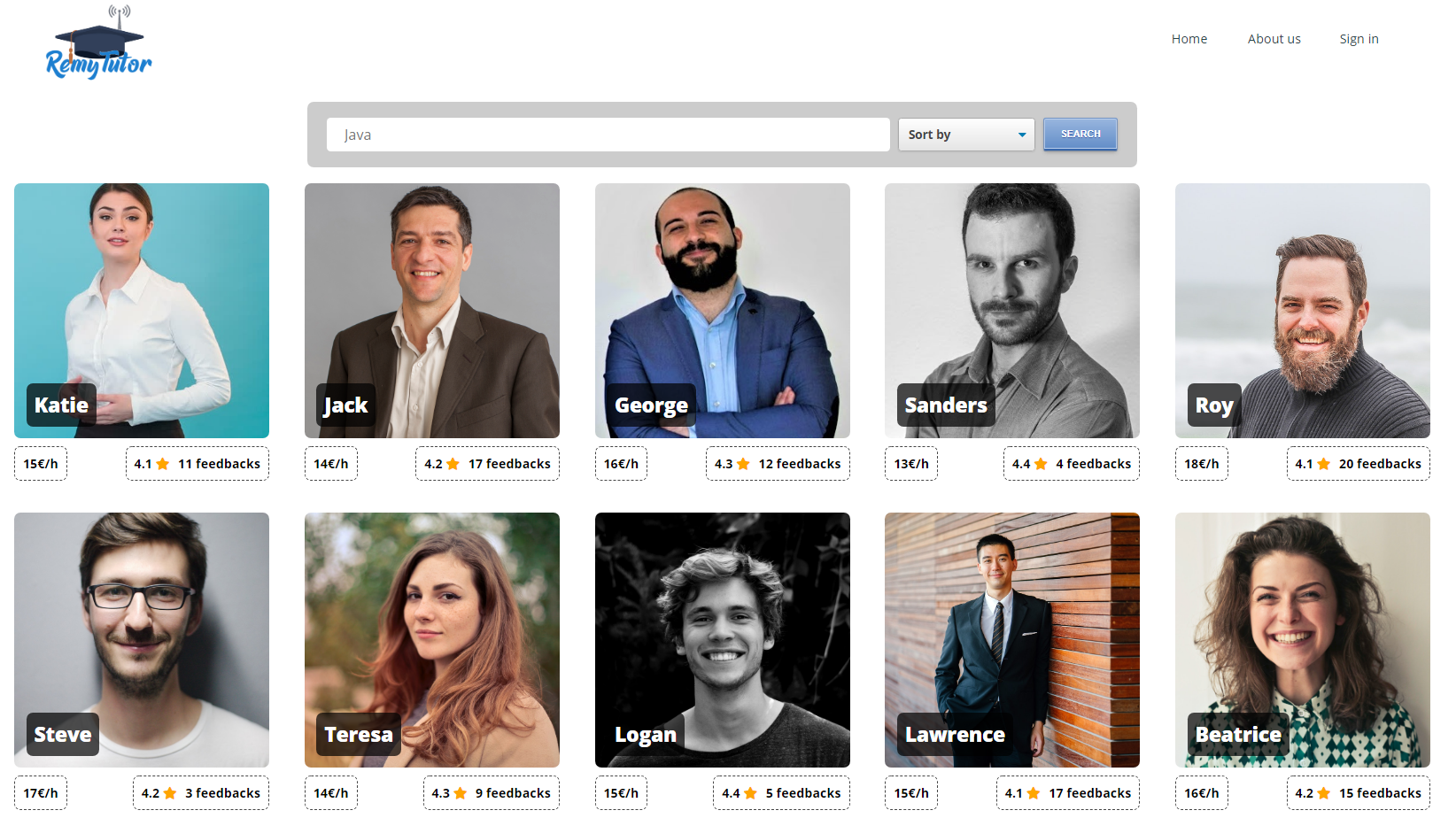
Homepage

This page main purpose is to channel the user through our service, and this is done by showing a search bar where he can immediately access the platform and browse available teachers, even without a registered account. This will be of course required in a second time when the user picks a teacher and decides to book a lesson. This page also hosts a slideshow that sums up all the opportunity our platform offers, to encourage users to join the team! The slideshow is implemented from scratch with pure Vanilla JS and it has a sliding animation effect powered by CSS3 transitions. It also scales responsively with the page and offers an infinite images loop. The search bar is equipped with an autocompletion/suggestions box powered by jQuery UI that lists all the offered lesson topics. From here it is possible to access the sign in or sign up interface described later.



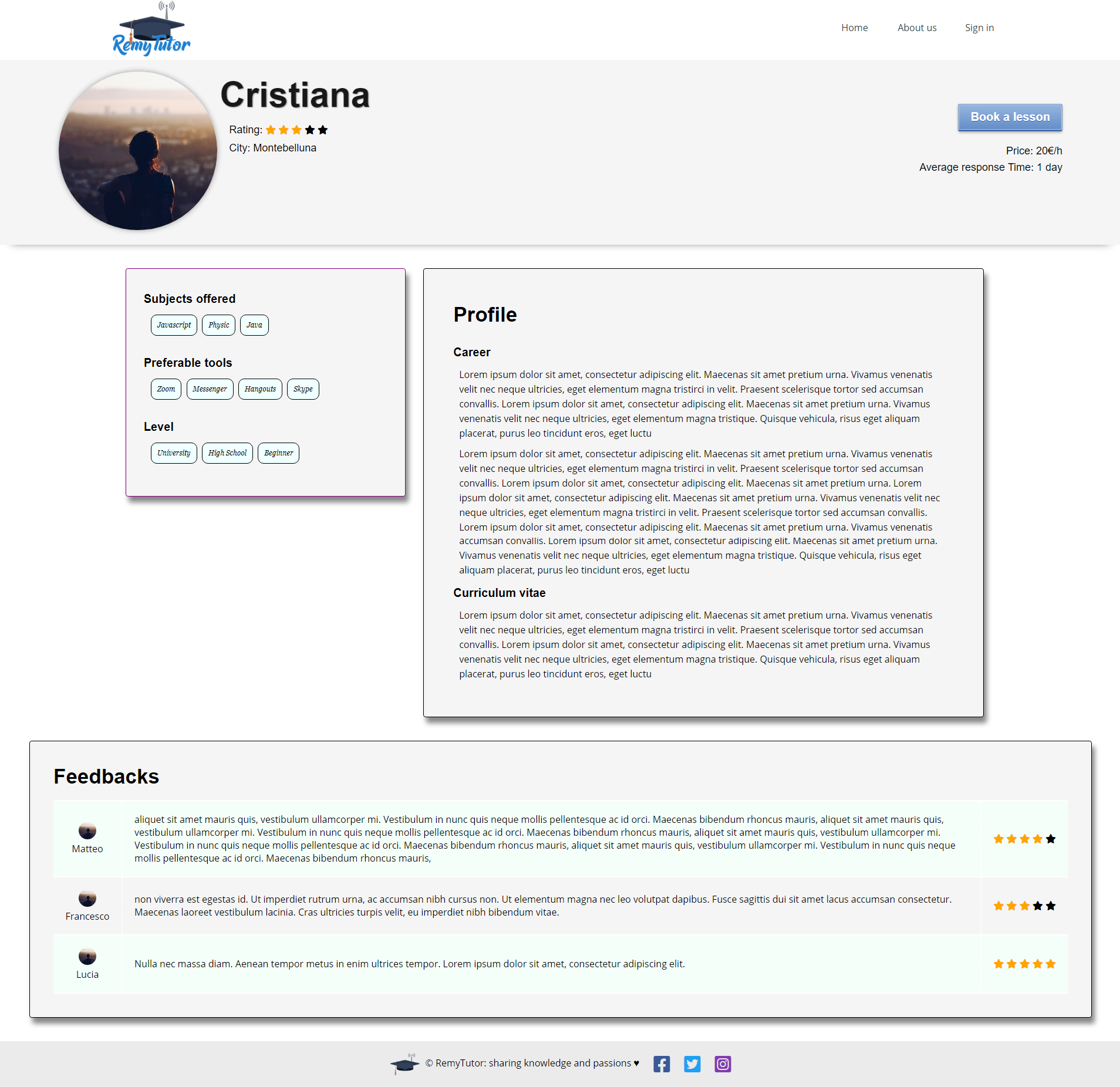
Search Interface

The name itself pretty much explain the purpose of this page. It is a common search page that offers the possibility of sorting results by average feedback rate and teacher’s tariff. Again, also from here is possible to access the sign up/in interface.

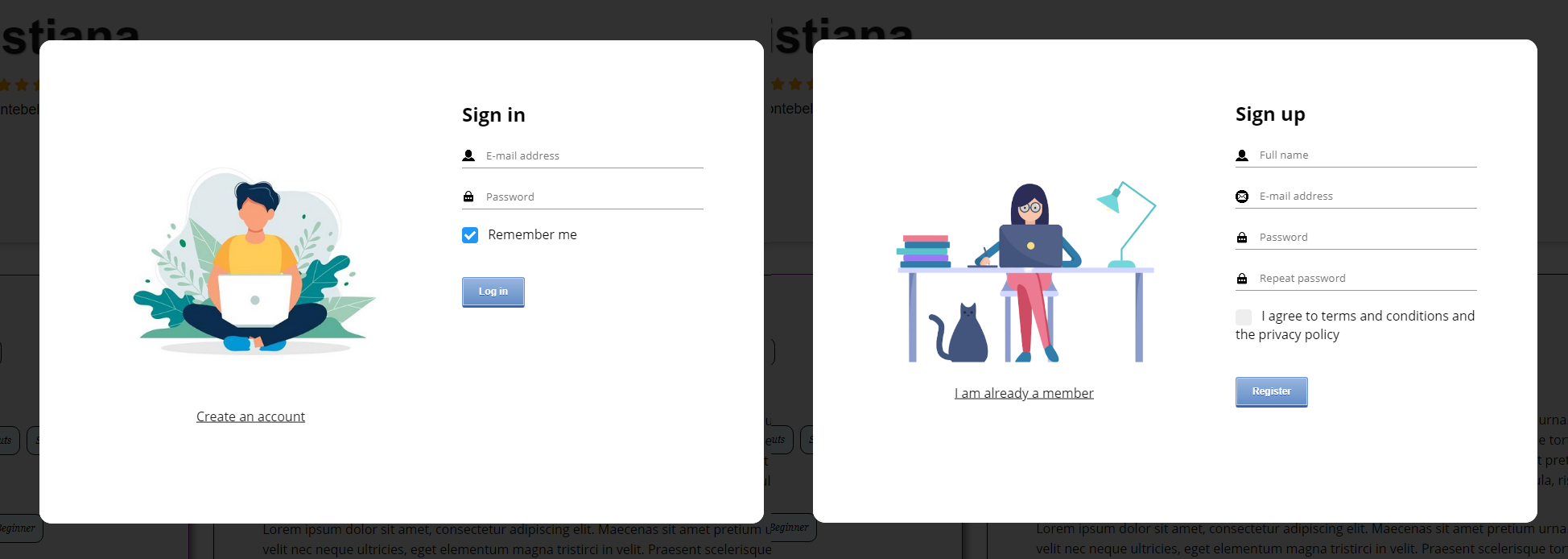


Teacher page

This is the page the user is presented when he picks a teacher in the search page. It contains the teacher’s presentation that he writes himself on the profile dashboard page. He can write whatever he wants but of course it should be aimed at winning the student’s trust and convincing him that he is the right one. At the bottom, the teacher previous student’s feedbacks are shown to prove teacher’s reliability, or lack thereof. In the upper part, a button is prepared for sending a request to the teacher that he can either accept or reject. It is at this point that, to perform this operation, an account is required.



Sign up/in Interface

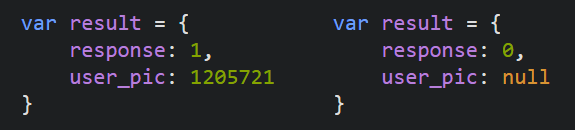


This interface appears as a modal on top of the page we are browsing, and it consists of a two-faced card with a flipping animation effect between the two, triggered by the link on the left side of each face.

When a sign in request is issued, an AJAX call to the server will be performed with the following very simple JSON data format:



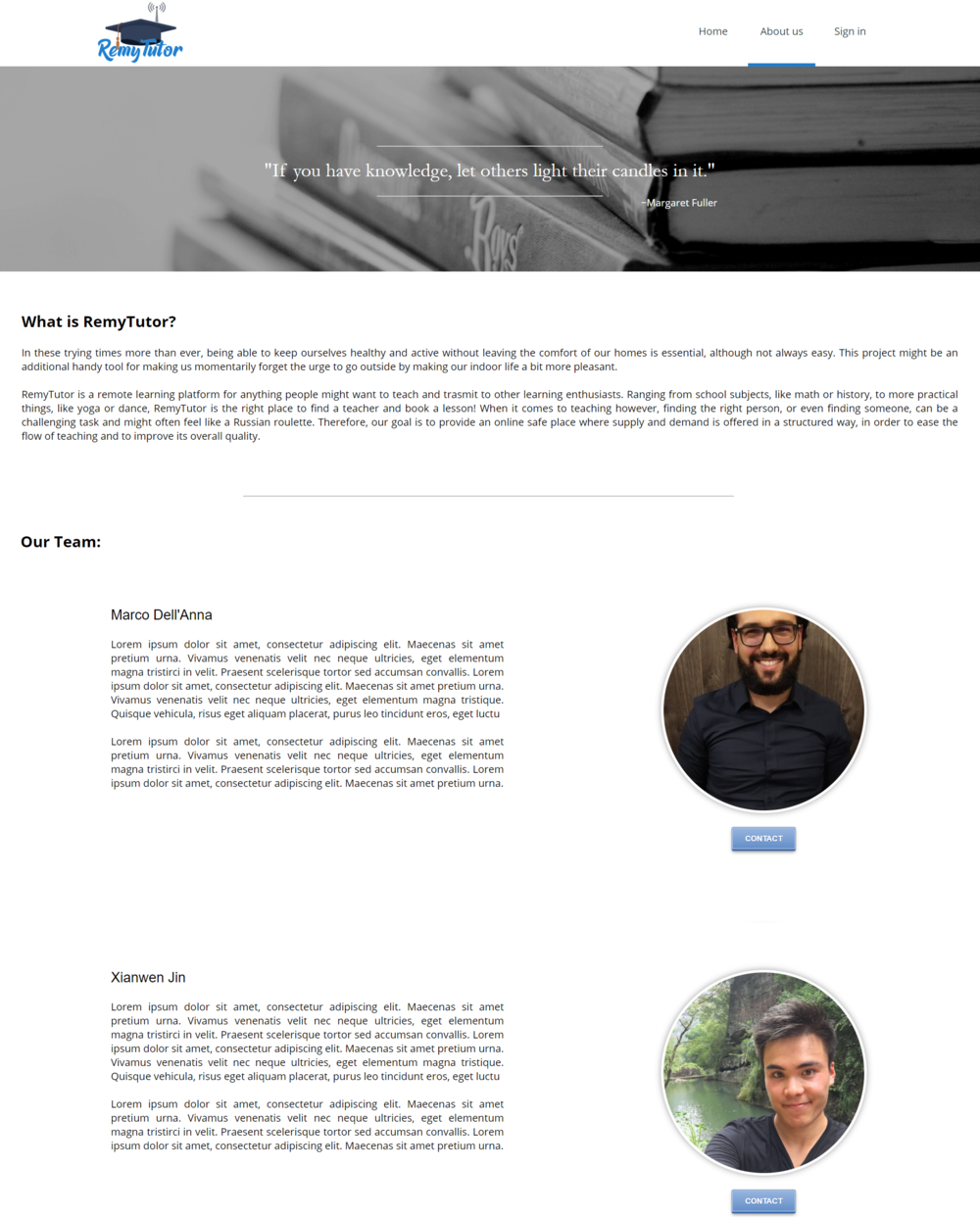
The security of this operation is not a problem at all, since the server can easily be equipped with a certificate, therefore it is possible to switch to the HTTPS protocol so the data will be automatically encrypted. If the credentials are correct, the server will answer positively (1) otherwise it will return a negative response (0). The answer will also contain the user ID or null for invalid credentials.



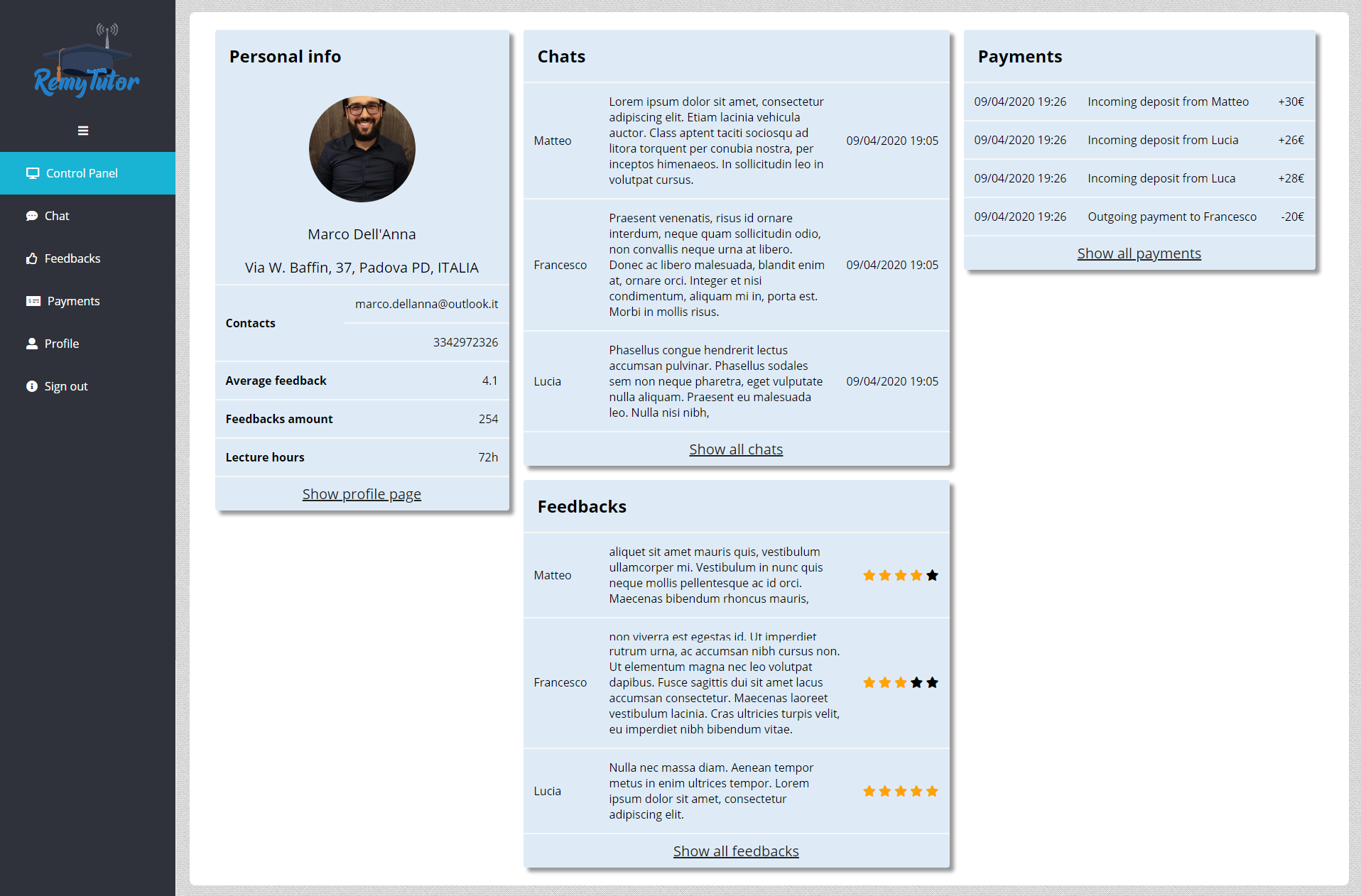
On the sign-up face instead, the form is validated and only in case everything is fine the form is sent through a standard POST HTTP request to the server, which will re-validate the data and create the new user account.

About us

All the information about our website and our team is displayed on this page. Here we explain the importance of having a website such as RemyTutor through an image and a quote by Margaret Fuller (which is clickable) displayed at the top of the page. The overall structure of the page was done using a Flexbox container in order to make it responsive. In the first section we explain what RemyTutor is about and our overall goals with it. After giving a brief explanation of the website, in the following sections we present our team with each member having a small description, an image of themselves and a contact button on the right side with which users can interact.



Dashboard



|  |  |
| --- | --- |
|  | The dashboard main page is simply a condensed version of all the other detailed pages. Here only the most recent items of each other page are shown, together with a box containing the user’s main personal information. The user can be redirected to the specific page through the navigation bar or a link at the bottom of each box. The navigation bar is shared among all the dashboard pages and it has been made collapsible in order to offer more space as needed. |

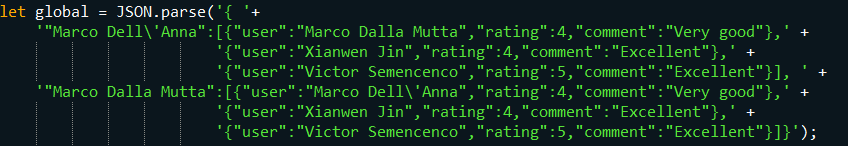
Chat page

…

Feedbacks page

The feedbacks page shows the user general evaluation, the feedbacks that he/she received from other users, and a box for giving feedbacks.

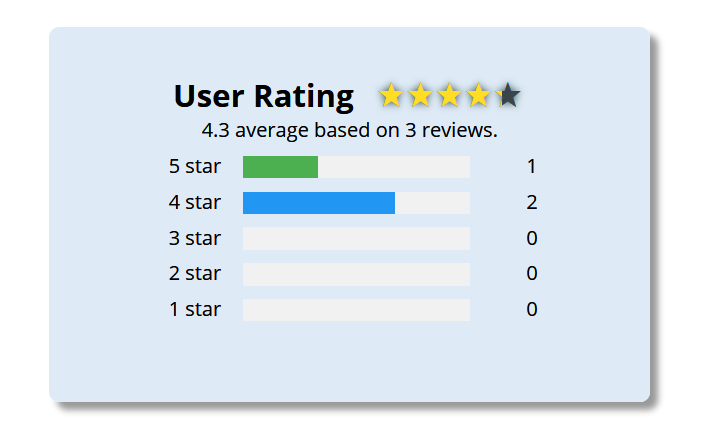
As of now, the data for computing the user evaluation and showing feedbacks is taken from a JSON object parsed by JavaScript (we are unsure if this will become an AJAX + REST call or if everything will be generated by the server with JSP, the first option is the most probable right now). We imagine that we can retrieve from the server a list of all feedbacks given to all the users:

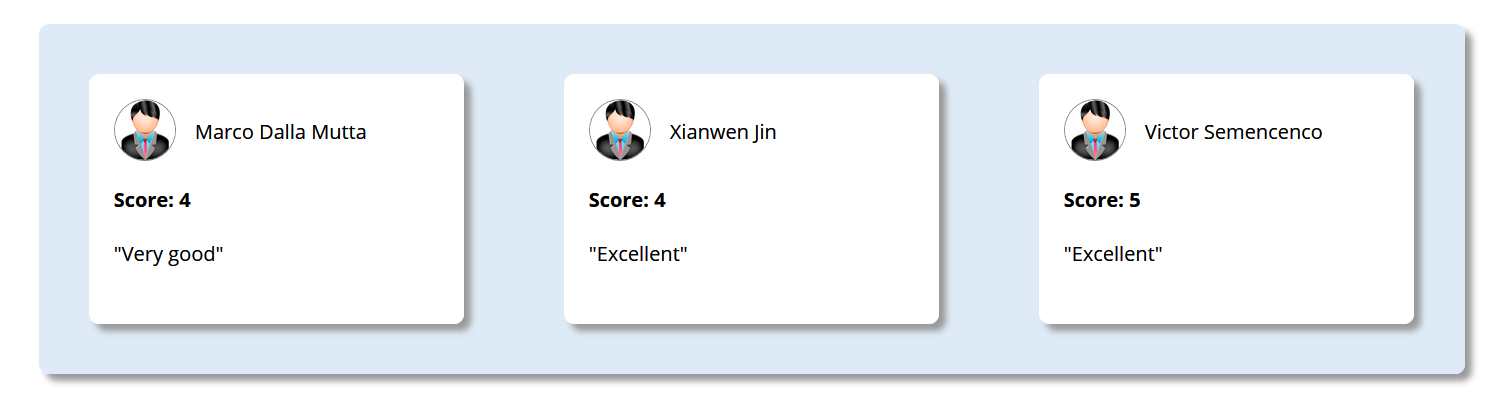


and we imagine to ask for only the feedbacks given to the current user:



and the resulting object is used for both showing the feedbacks, that are showed in a horizontal array of boxes (scrollable if needed), and for showing the generic evaluation.

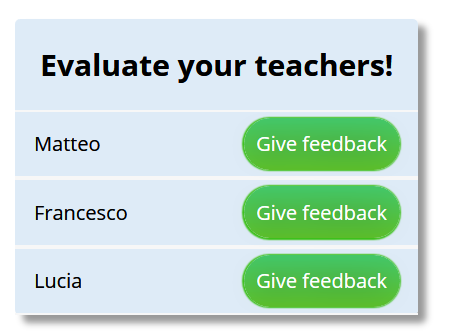




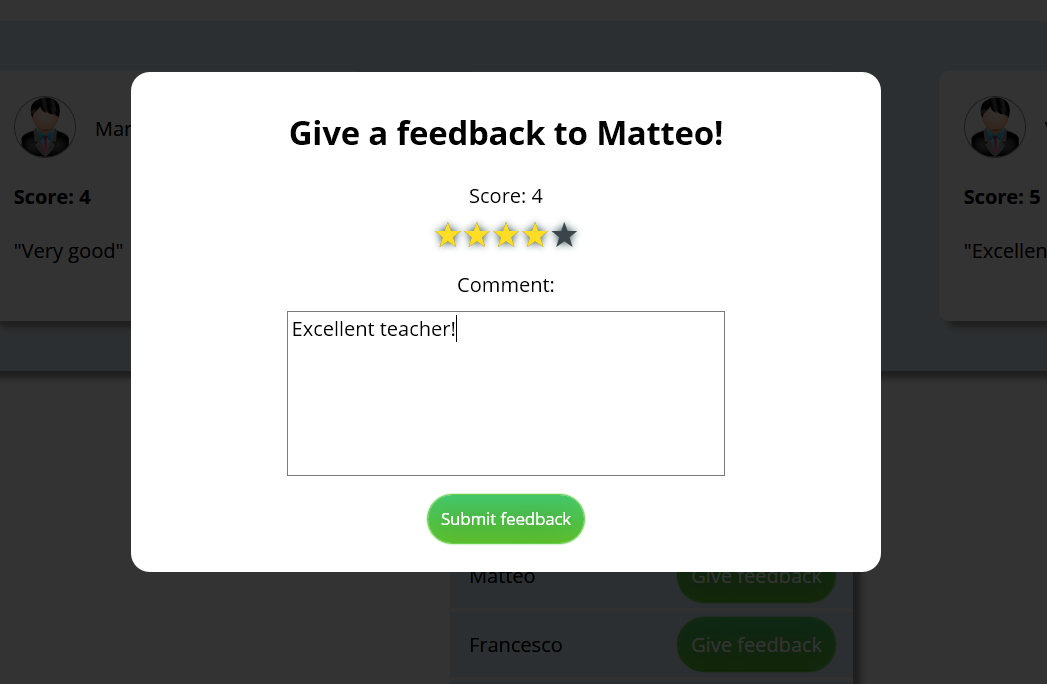
In this page is also present the possibility to give a feedback to a user that lectured us. Again, let’s imagine the server has a list of users we can give a feedback to:



In this case, the page obtains a box where it’s possible to select an user to give the feedback to:



Every button opens a dedicated form inside a modal that has the same base structure as the Sign up/in one. In this form, the user can choose the score clicking on the stars and can insert a short comment.



Payments page

…

Personal Profile

The profile page provides a complete editable view about user’s own main personal information. It includes also other useful information such as qualification and identity card. These are used as an optional identification mechanism since we base the main “reliability rate” on feedbacks. In this way there are not big entrance barriers, thus attracting a larger number of users, who can later base their choice on average feedbacks also by giving them the opportunity of sorting results by this value. Nevertheless, teachers might want to add this information to the profile in order to appear trustworthy towards students. Furthermore, all aforementioned information could be updated by the user by filling simple forms or uploading new files. In order to ensure the password strength, the new password should be at least six characters including upper/lowercases and numbers.

