
WEB DEVELOPMENT

FIRST PRACTICAL TASK – GRAPHICAL INTERFACE DESIGN – HTML AND CSS + BOOTSTRAP - AND CLIENT PROCESSING - JAVASCRIPT AND AJAX

This practice of the "Web development" (WD) course will have the objective of consolidating the knowledge acquired in class about the following points, all of them aimed at achieving an adequate construction of the "client" part which is necessary in web systems:

- ✓ Design of the view of the web system to be shown to the user, based on development with HTML, CSS and specific libraries such as Bootstrap.
- ✓ Programming, through Javascript and AJAX, as well as through the use of the new programming APIs that HTML5 provides, of the necessary processes so that the user view mentioned in the previous section behaves according to the established specifications.
- ✓ Use of a correct programming "methodology" in Javascript, as well as the possible use of specific development libraries that can facilitate the work.

The objective of the first practical task is to design the visual part of the client of the web system corresponding to the game site for a "ufo battle game", demonstrating his/her Javascript domain doing the processing of the game and the use of some asynchronous technologies used to connect to the server and process information that came from it.

Students will be free to design it in the format they want and with the appearance they prefer. The objective is to develop a game in which the user will try to get as high a score as possible by shooting a small missile at objects that move around the screen as a "battle game".

ATTENTION:

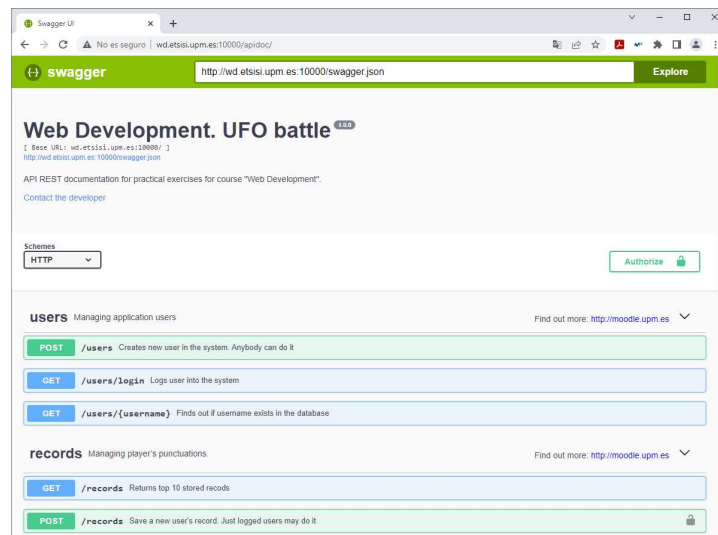
In order to implement this and the next task, a REST API has been developed that provides the "back-end" service to the application (management of users and scores). This REST API is available to students so that they can test and validate the generated code before it is submitted for evaluation. The basic features of the REST API are:

Base URL: <http://wd.etsisi.upm.es:10000> (Note the use of port 10000)

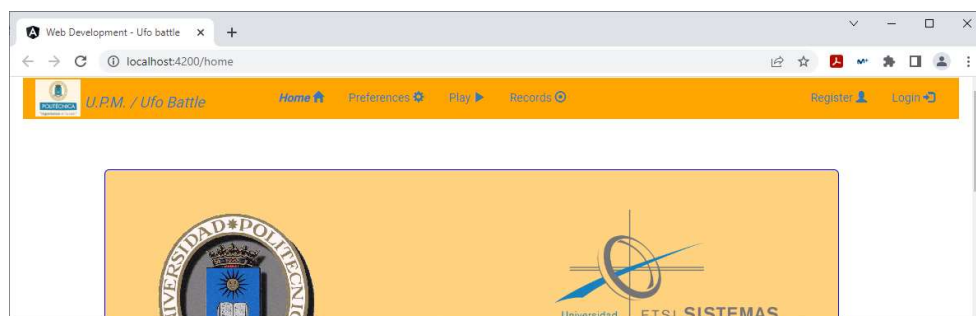
The API is sufficiently documented through the "swagger" tool at the url <http://wd.etsisi.upm.es:10000/apidoc>

This specification page allows, on the one hand, to check the accessible routes and methods; on the other, check the parameters necessary for its execution as well as the results that it provides as a response; and finally, test that its operation is understood through the execution of the methods available on the same page.

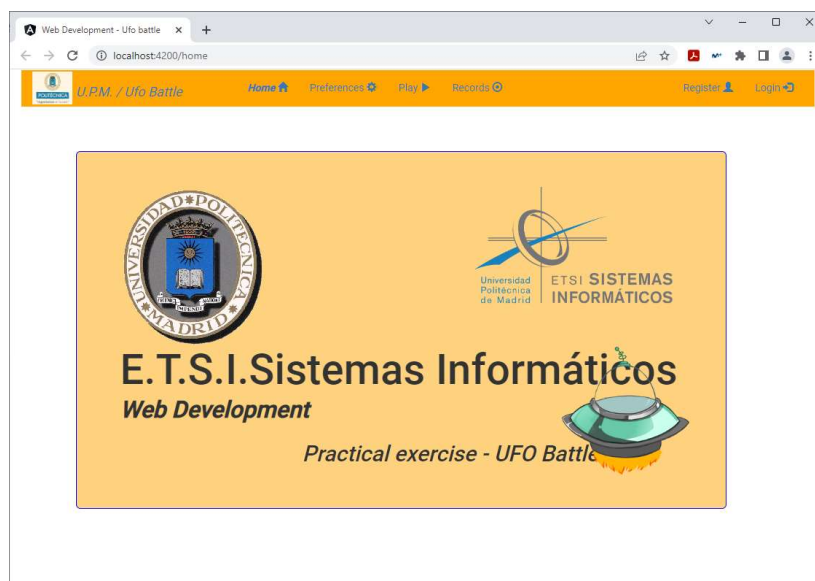
It is **important** to note that all the parameters passed to the API, whether in the query or in the form, are passed as independent parameters, with the name indicated in the documentation. However, all responses received from the REST API will be in **json** format (even if it's just a simple string or number). Obviously, the token received in the Authorization header will be an exception, as detailed below.



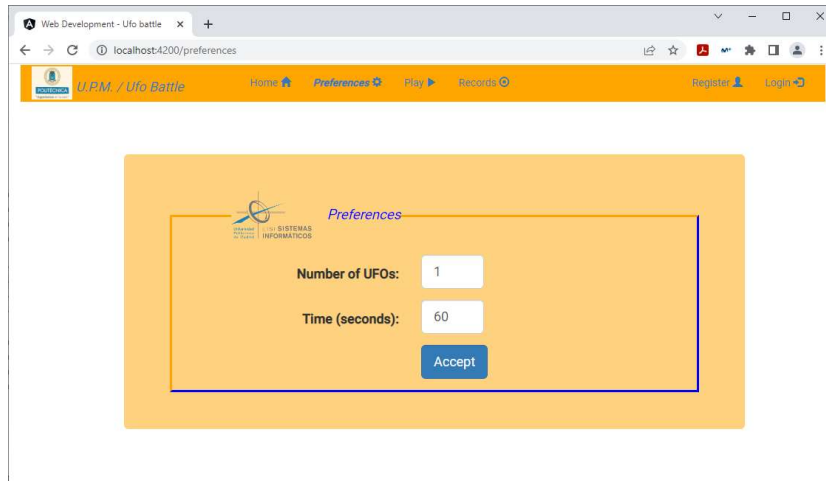
The main page of the website will show a presentation to the designer's liking, knowing that its elaboration will be taken into account when assessing the work. In this first practice, although the functionality of the system is described, it must be taken into account that the “Registration” option that is going to be presented does NOT have any type of process beyond showing the form. This page must necessarily show some type of menu with, at a minimum, the following options:



Home. – When pressed, the system will always show the presentation/welcome page. An example can be seen in the figure in which the menu appears in the upper part with the options divided into two groups (left: functions and right: registration and login) and in the lower part a presentation appears that the student must prepare to his liking.

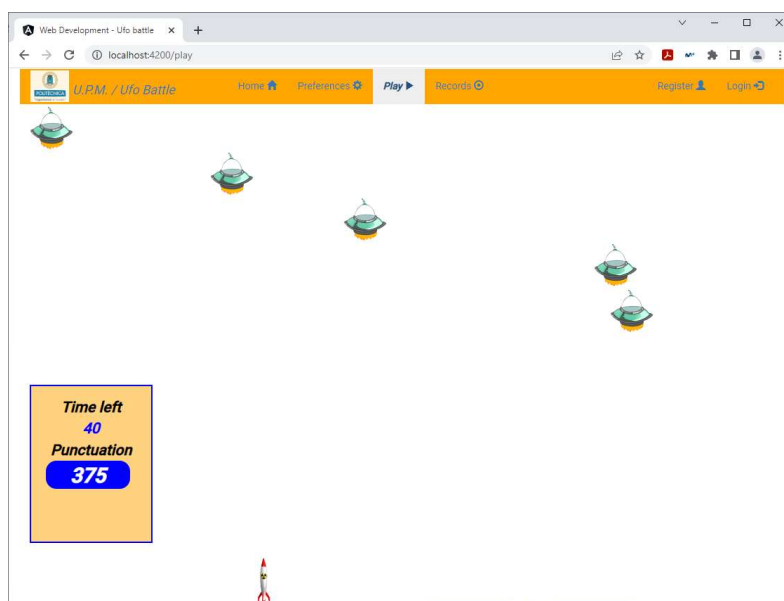


Preferences. – This option will give way to a form in which the player can choose both the number of ufos with which he is going to play and the time he is going to have. The number of ufos can be chosen **between 1 (by default) and 5**. Time can be **60 (by default), 120 or 180 seconds** (which will be the number of seconds that the player will have to play the game before the system prevents him from continuing playing). These preferences should be stored locally so that they can be consulted when needed.



Optionally, once the player presses "**Accept**", the screen corresponding to the "**Play**" option can be presented directly

Play. – This option presents a screen where the user may play. Initially, the number of ufo images will correspond with the one indicated by the preferences parameter and there will be just one missile to fire. At the beginning, each ufo image will be located at a different height and will go in a random direction (left or right). Also there will be an area where remaining time is shown (in seconds), starting with the one corresponding to the one selected in the preferences. At each moment, remaining time shall be updated until it reaches zero, moment in which the system will prevent the user from continuing to play.



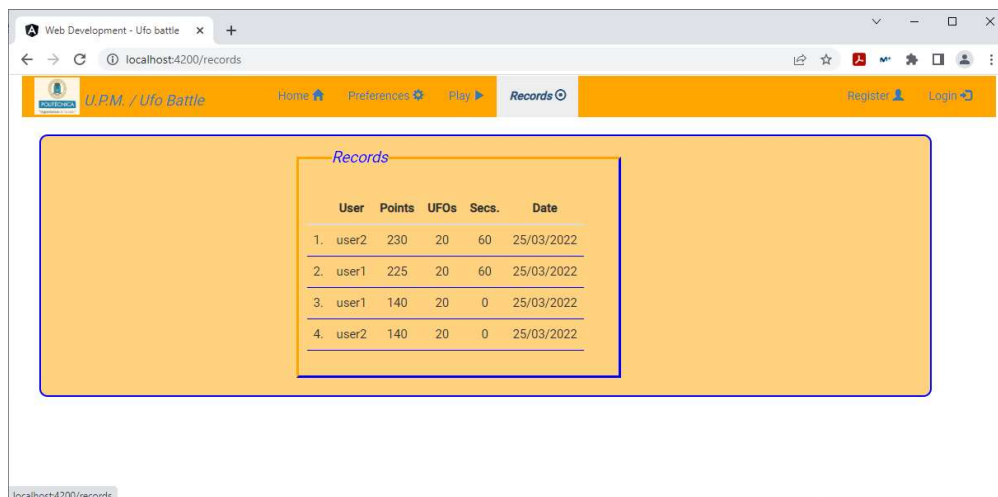
Game: The user will try to hit any of the displayed ufos launching the missile. At any moment there will be just one missile at a time (if it is not launched, it may move over the horizontal line at the base

of the screen, following the keystrokes on the user's keyboard). If launched, the user may NOT launch another one until it hits an ufo or it disappears over the upper limit of the screen. When launched, the user will not be able to move the missile to the left or right. If the missile, in its way up, hits one of the ufos, then the player will get **100 points** (that should be reflected in his score panel) and the missile will appear again at the baseline of the screen. The ufo won't disappear and will keep its way (left or right) normally (an image to indicate the hit may be used for a while). Each time that the missile misses all the ufos and it disappears over the top of the screen, then the punctuation shall be **reduced in 25 points**.

When time is over, the system should **avoid** the player to **continue playing** and it will **update** the **final score** (showing it in the panel) according to the next rules:

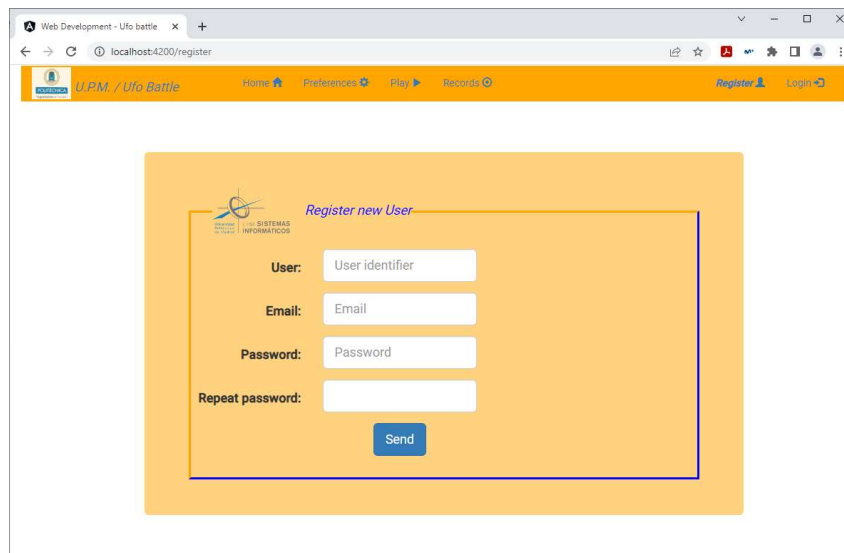
- The punctuation **will be divided by the number of minutes** that the user has selected: if 60 seconds was selected as playing time, punctuation will be divided by 1 (so stays equal), if 120 was selected, then punctuation will be divided by 2, if 180 was selected, punctuation will be divided by 3.
- After the prior division, the punctuation will be reduced in **50 points per each of the selected number of ufos that exceeds from one**. That is, if there was just one ufo in the game, the final punctuation won't be reduced, ..., if the number of ufos was 5, then punctuation would be reduced by $4 \cdot 50 = 200$ points.

Records. – This option presents a list with the 10 highest records in the server (if there are not 10 registered, the ones that exist will be shown). To do this, this list will be requested from the server as indicated in the REST API documentation (get method at <http://wd.etsisi.upm.es:10000/records>).



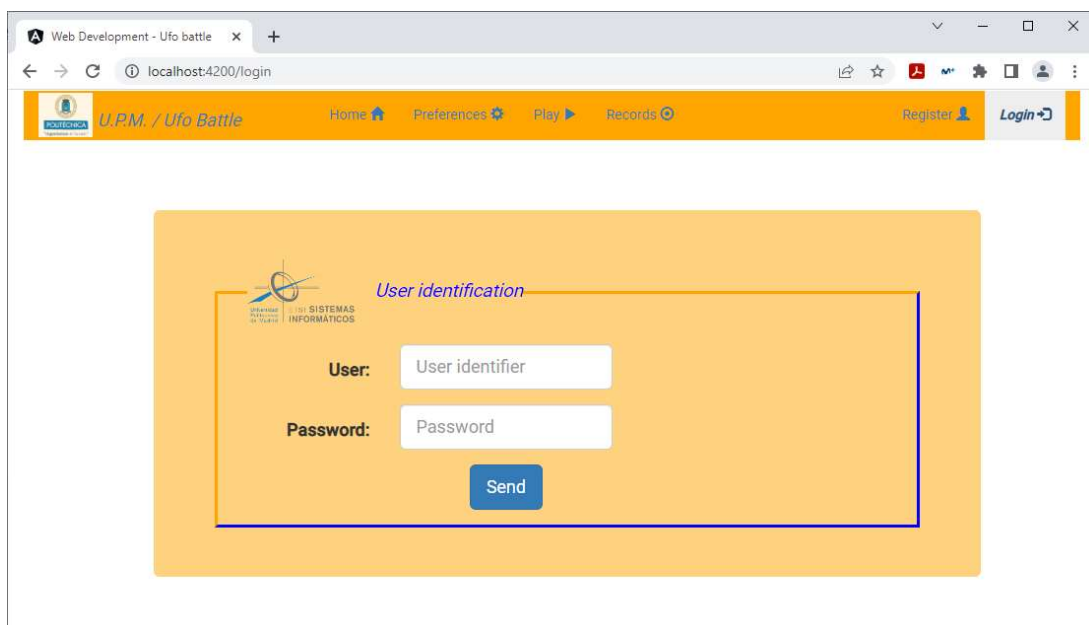
	User	Points	UFOs	Secs.	Date
1.	user2	230	20	60	25/03/2022
2.	user1	225	20	60	25/03/2022
3.	user1	140	20	0	25/03/2022
4.	user2	140	20	0	25/03/2022

Register. – It allows a user to register in the system, which will give him the ability to save his scores. A user who wants to register in the system must enter a **username**, an **email address** and a **password** to be able to identify himself (all mandatory). In this practice the process associated with this option in the REST API will **NOT** be developed, although the form will be implemented in order to prepare it for the next practical task.



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/register'. The page has an orange header with the text 'U.P.M. / Ufo Battle' and navigation links: 'Home', 'Preferences', 'Play', 'Records', 'Register', and 'Login'. The main content area features a yellow box with the title 'Register new User'. Inside this box, there is a form with the following fields: 'User:' with a placeholder 'User identifier', 'Email:' with a placeholder 'Email', 'Password:' with a placeholder 'Password', and 'Repeat password:' with a placeholder. A blue 'Send' button is located at the bottom right of the form.

Login: This option will allow a registered user to access the system to proceed to save his/her scores. In order to do "login", the user must identify himself by means of a **username** accompanied by the **password** that he also registered:



The screenshot shows a web browser window with the address bar displaying 'localhost:4200/login'. The page has an orange header with the text 'U.P.M. / Ufo Battle' and navigation links: 'Home', 'Preferences', 'Play', 'Records', 'Register', and 'Login'. The main content area features a yellow box with the title 'User identification'. Inside this box, there is a form with the following fields: 'User:' with a placeholder 'User identifier' and 'Password:' with a placeholder 'Password'. A blue 'Send' button is located at the bottom right of the form.

In this task, it is not necessary to complete the whole login process. Whenever a valid user sends this data (username and password) to the server (check "*user login*" in the API documentation) causes that the server responds with a message in which header ("**Authorization**" header) there is a security token. This token should be obtained from the header (there is another copy on the body of the server response just in case the student does not know how to get it from the header). Students must check that they receive this token, and they have to store it locally (webstorage). Once the token is stored, nothing else about this option will be implemented in this practical task. If no token is received (an error message will be received by the server), then a "**login error**" should be displayed. The whole process will be completed in task number two.

In order to check that token is being received, students may use the username: "**user1**" or "**user2**" with passwords "**user1**" and "**user2**" respectively. These are already registered users.

It is requested:

A zip file must be uploaded to the moodle platform containing all those elements so that, once decompressed, the practice can be viewed and executed without problems. The decompression of the file must give rise to the hierarchy of directories necessary to place each file where it belongs, including those that may belong to the libraries (bootstrap, jquery, etc) if CDN is not used for their incorporation.

Library elements that provide non-standard HTML or CSS elements (including Bootstrap) can be freely used but process may be just in plain Javascript (or jQuery)

EVALUATION

- ✓ The evaluation of the practice will be carried out by assessing the aspects indicated in the corresponding rubric published on the platform.