

WSC2018\_11in5\_Client\_Side

**TEST PROJECT**

**WEB DESIGN**

Client Side

WSC2018\_11in5\_Client\_Side\_EN

**Submitted by:**

# CONTENTS

This Test Project proposal consists of the following documentation/files:

1. WSC2018\_11in5\_Client\_Side\_EN.docx
2. WSC2018\_11in5\_Client\_Side\_MEDIA.ZIP – Media Files

# INTRODUCTION

In recent years, the Internet has become an integral part of our daily lives, enabling the dissemination of information in an inexhaustible source of content and interaction. Gaming plays a prominent role in our daily lives, allowing millions of people to get access to fun and entertainment quickly and free.

Reflecting on these concepts, you decided to develop a small game that works in the most common web browsers and showcases your talent in the skill of web design and development. The game will be called **Sky Angel**.

The development of the game will involve using HTML and CSS for the design of the layout and client-side programming (JavaScript and open source libraries) for the game functionalities. A few media files will be provided in a zip file. You can create new media and modify anything provided to you for the game. Your game will be developed in a tablet resolution with 1024x768 pixels (width x height). If game is opened in a larger screen, the game must be in the center of the screen (horizontally and vertically).

# DESCRIPTION OF PROJECT AND TASKS

This module is to be completed in 5 hours. The first 2.5 hours must be used to create the design of the game in three PNG images and the layout using HTML/CSS. Your HTML/CSS layout should follow the design (PNG) that you created. The final 2.5 hours you will create the functionality of game using JavaScript ensuring the game works correctly in different web browsers, following the requirements described below.

SKY Angel game uses elements described below:

1. Airplane: Element that is controlled by the user.
2. Clouds: Elements that move from right to left to give the impression of movement of the airplane in the sky.
3. Birds: Elements that airplane needs to dodge.
4. Stars: Elements that airplane needs to get to increment star’s counter.
5. Parachutes: Elements that airplane needs to get to increment the fuel level.
6. Fuel Counter: Elements that shows how much fuel is available.
7. Star Counter: Elements that shows how many stars the airplane got.
8. Timer: Elements that shows how much time the airplane has been flying.
9. Sound Button: Enable and Disable the sound in the game.
10. Font Size Buttons: Up and Down to increase and decrease the font size.
11. Pause/continue button: Button to pause/continue the game.
12. Logo: Add the provided logo in the game.

## FIRST 2.5 HOURS – DESIGN AND LAYOUT

1. Deliver at least 3 PNG image files that present:
   1. Game board layout: This design must present all elements described above: logo, airplane, clouds, birds, stars, parachutes, fuel counter, star counter, timer, sound button, font size buttons and pause/continue button.
   2. Ranking Table presentation: This design must present the logo of the game and ranking with the following columns: position, name, stars and time in this order, with the table is presented the “Start Game” button.
   3. Game instructions: screen of the game presents the instructions to the user and the “Start Game” button. The instructions for the game are included in the media files.

1. Develop the initial markup (HTML + CSS) of your game application. When the address is accessed (http://xxxxxxxxx/yy\_client\_side) the game is presented to the user.
2. instructions and the button “Start Game”. The instructions must be presented in an animated way.
3. xxxxxxxxx = Server Address. yy = Country Code
4. All buttons must have active hover effects. The background of the buttons in hover state must be:

#f39c12. The active state must follow the example called ripple which is provided in the media files.

1. The HTML and CSS code must be valid in the W3C standards for HTML 5 and CSS 3 rules.

## FINAL 2.5 HOURS – GAME FUNCTIONALITIES

1. Pressing the “Start Game” button in the initial screen, the game starts, the aircraft starts to fly in an animated way and clouds should move from right to left to give impression of movement of the aircraft in the sky. The timer is started and displays the time that the aircraft is in movement. The fuel counter, starts to decrease, one point per second. When the game starts, there are 10 points of fuel giving the player 10 seconds to fly. The max capacity of fuel is 30 points (30 seconds of flight).
2. The user can use the arrow keys to move the aircraft. The left arrow moves the aircraft to the left. The right arrow moves the aircraft to the right. The up arrow moves the aircraft up. The down aircraft moves the aircraft down. It is not possible fly the aircraft unless it is visible on the screen. The aircraft cannot fly off the screen in any direction.
3. During flight, the aircraft needs to avoid birds that are presented in the sky. If the aircraft hits a bird, the fuel must decrease by 10 points and the birds will fall down with animation. The birds need to be presented in random position and in an animated way, flying from right to left. It should always be possible for the aircraft dodge the birds (for example, there is no solid wall of birds which could not be flown through).
4. During the flight, the aircraft needs to collect parachutes and stars dropping from the sky. For each parachute icon collected, the fuel counter is increased in 10 points (10 seconds). Each collected star increase the star’s counter in 1 point. The star’s counter starts with 0 points.
5. The parachutes and stars are presented in an animated way dropping (from the top of the screen) in the sky randomly.
6. During the flight the user can pause the game clicking the pause button, or by pressing the space bar. When the game is paused, all interactions and sounds must be stopped. If the user clicks the pause button again, or presses the space bar again, the game continues from the moment that it was paused.
7. The clouds in background need to present an animation to give the sensation of movement.
8. If the fuel counter reaches zero, the game is over.
9. When the game is over, the animations stop and the timer counter stops, the game will collect the name of the user in a form field. The user fills the field and click the button “Continue”. The button will be disabled until the user fills that field.
10. The user name, the time of flight and the number of collected stars need to be registered in the server through an AJAX request. Send this information to the server address and using these variables:
    1. <http://xxxxxxxxx/register.php>
    2. Method: post
    3. name: name of the user
    4. time: number (integer) time in seconds
    5. stars: number (integer) of collected stars
11. The register.php will register your information in one database shared with all competitors and will return a JSON information with a ranking of users. An example of the JSON structure is presented below:

[

{"id":"1","name":"Player 1", "time":"20", "stars":"10"},

{"id":"2","name":"Player 2","time":"14","stars":"8"}

]

1. The game presents the ranking to the user with the button “Start Game” that permits restarting the game.
2. The ranking needs to be ordered by the number of collected stars and for time elapsed in decreasing order. If more than one user has the same quantity of collected stars and the same time, they receive the same position in the ranking. The JSON data returned by the server is not ordered, it is your task order correctly the data to present to the user.
3. To increase the game interaction, some sound effects need to be used:
4. background.mp3: used during the flight;
5. hit.mp3: played when the flight hit a bird (game is over);
6. star.mp3: played when the aircraft get a star;
7. finish.mp3: played when the fuel is out (game is over).
8. Use your talent to increase the usability of the game as much as possible to permit a better experience for the user.
9. To improve the accessibility your game must have options to increase/decrease the size of font in the screen (timer, star’s counter, fuel and ranking).
10. There must be an option to disable/enable game sounds. If the sound is disabled, no sounds should be played. If sound is enabled all sounds must be played.
11. Your game should work without JavaScript errors or messages shown in the browser console.
12. Maintain your HTML/CSS and JavaScript code organized and clean to facility the future maintenance, use correct indentation and comments.
13. The game needs to works correctly in both browsers (Google Chrome and Mozilla Firefox). The game requirements will be checked in Google Chrome and compatibility will be checked in Mozilla Firefox.

# INSTRUCTIONS TO THE COMPETITOR

* The media files are available in the ZIP file. You can modify the supplied files and create new media files to ensure the correct functionality and improve the application. You can use jQuery and jQuery UI if you find it necessary.
* Save your design files in a folder call "YY\_Client\_Side/YY\_design" where YY is your workstation number.
* Files names:
* Game board: YY\_game\_board.png
* Ranking: YY\_ranking.png
* Game instructions: YY\_ game\_instructions.png
* You should create additional images for each of the requested resolutions to highlight hidden elements, animations, interactions, or any additional information that will assist in the presentation of the game design.
* Additional file names
* Game board: YY\_game\_board\_2.png, YY\_game\_board\_3.png …
* Ranking: YY\_ranking\_2.png, YY\_ranking\_3.png …
* Game instructions: YY\_ game\_instructions\_2.png, YY\_ game\_instructions\_3.png …
* Save any image source files to a folder named "**YY\_fonts**" inside the "**YY\_Client\_Side/YY\_Design**" folder. The source files are the files that contain the layers, development files, ie .psd, .ai, .svg.
* Save the working game to the directory on the server named "**YY\_Client\_Side**". Be sure that your main file is called index.html.
* Use meaningful variable names and document your code as much as possible so another would be able to modify your work in the future.
* The registration of game results will be handled through a server service for all competitors. Be sure that the address is correct and the variables and formats are correct to permit a correct registration in the database. In case of error the JSON error messages returned to request.

[

{"error":"Error Message"}

]

* You are responsible for managing your time. If you finalize a task you can continue to another task. If you complete the required tasks to be submitted at the end of the 2.5 hours, you can start the tasks for the final 2.5 hours.
* The module will be assessed in Chrome and FireFox.