POLITECNICO
DI TORINO

System Design Project - Securing Robots and Exoskeletons

LaserBot Battle - User Manual

Cirici	Stefano	s243942
Panini	Francesco	s242547
Ledda	Luca	s237447
Spigarelli	Edoardo	s241486

Academic Year 2017-2018

June 2, 2018

Contents

1 Introduction

The project scenario consists in a laser battle involving robots remotely driven from a client browser, which have to move and shoot in order to survive. A robot is composed by a Raspberry connected to an Arduino, whose goal is to manage IR lasers, sensors and stepper motors attached to its chassis.

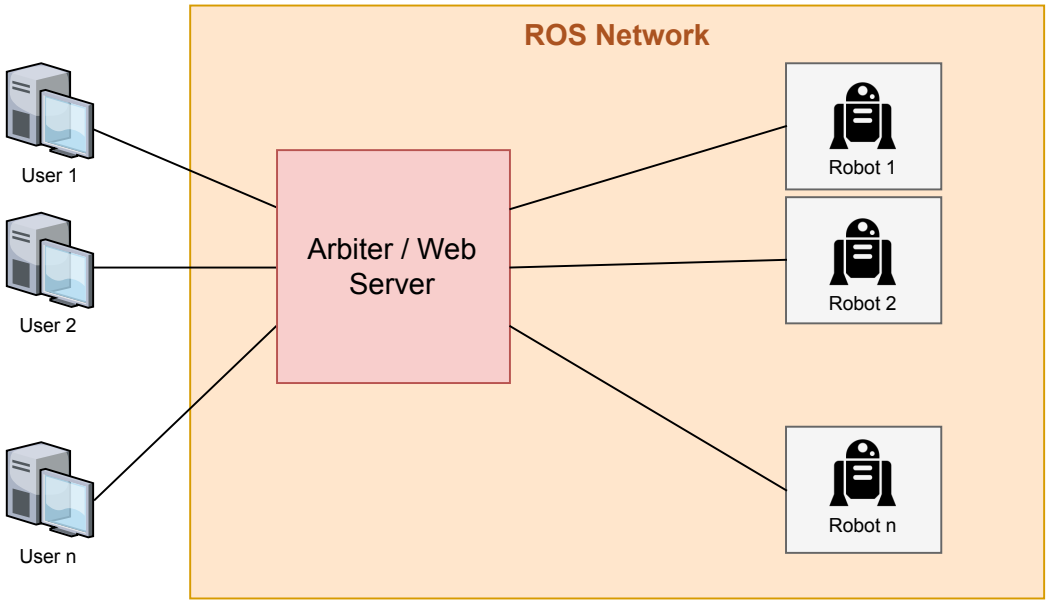

Figure 1: High Level Architecture - Whole scenario

This document contains all essential information for the user to make full use of the application, including a description of the system functions, capabilities and alternate modes of operation, and step-by-step procedures for system access and use.

2 Project files

This section reports a description related to all files needed for the environment to be installed and used.

2.1 Project files

The project directory could be downloaded from: pp-robot-2018. It contains the following folders:

- **arduino** : It contains all files needed for the Arduino board to operate in between Raspberry and Robot hardware. The sub-folder `libraries` contains the libraries to be included within the Arduino sketch to proper handle the robot HW connected to Arduino. Instead, the sub-folder `rosserial_node_arduino` is the Sketch to be flashed and run on Arduino for runtime operation.

- **documentation** : It contains the documentation of the project involving *"User manual"*, *"Tech manual"* and *"Presentation"*.

- **raspberry** : It contains the catkin `laser_bot_battle` ROS workspace to be installed into the raspberry in order to be operative in between server and Arduino. In particular, ROS services, ROS messages and python scripts are defined here.
  ROS services and ROS messages are needed from raspberry to communicate within the ROS environment with server and Arduino.
  The python script `scripts/ID_service_client.py`, launched at power-on, asks to the server to receive an ID. Once the ID is retrieved, a robot is associated to the raspberry and up to that moment raspberry works as a forwarder of ROS messages transiting between Arduino and Server. The python script `pingThread.py` is launched from `ID_service_client.py` script, once the raspberry has received an ID from the server. In practice, it is a thread that has the role to signal to the server that raspberry is alive.

- **server_ws** : It contains the catkin `laser_bot_battle` ROS workspace and web environment used by the server in order to be operative in between user and raspberry. In particular, ROS services, ROS messages and python scripts are defined here.
  ROS services and ROS messages are needed from server to communicate within the ROS environment with raspberry.
  The folder `scripts`, instead, contains the web environment (allowing the user to access to the application services through HTTP requests) and python scripts that permit the server to communicate with raspberry. User intents through HTTP requests are interpreted and forwarded to the associated raspberry robot. At the same time robot information coming from Arduino are sent by Raspberry to the server, which interprets them making available to the user.

- **LICENSE** : It specifies the LICENSE

- **README.me** : A brief tutorial that introduces main project topics.

For further technical details, please consult the "Tech manual".

3 Docker

This section explains step-by-step how Docker could be installed.
Docker installation is needed for sections section 4 (user environment configuration) and section 5 (server environment configuration). The complete guide can be found at the Docker website.

3.1 Install

All dependencies needed to make the robot and server working are provided in two separate Docker images.

To automatically install Docker CE:

```
$ curl -fsSL get.docker.com | sudo sh
```

In order to use Docker as a non-root user, it is necessary to add the user to the "docker" group issuing:

```
$ sudo usermod -aG docker $USER
```

Finally, to make operative this change, a log-out and log-in is needed.

To check if the installation has terminated without errors, we can issue:

```
$ docker --version
```

this will return the current installed version of Docker.

For uninstalling the Docker CE package:

```
$ sudo apt-get purge docker-ce
```

4 User Environment

In this section all the steps needed to configure the robot are covered.
Each robot is composed of three main parts connected together: Raspberry, Arduino and sensors-actuators. Detailed information for each part will be provided in the related subsection.

4.1 Raspberry

4.1.1 Install

In order to make Raspberry properly operate, following steps must be performed :

1. **Rasbian OS** : It is required Rasbian operating system up and running on the Raspberry. Use the link provided to download and install it before proceeding with the following configuration.
2. The dependencies needed to make the robot working are provided in a Docker image. Follow Docker installation guide provided in section 3.

4.1.2 Launch

Now, the image can be launched issuing the command:

```
$ sudo docker-compose run robot
```

This command is just needed to automatically launch Docker with all needed configurations. **Note** that this command should be run when both Arduino and sensors-actuators have been properly connected together.

4.2 Arduino

It is required to have Arduino IDE installed on the working PC to be able to properly flash Arduino with the proper sketch. Use the link provided to download and install it before proceeding with the following configuration.

4.2.1 Upload

In order to correctly flash Arduino the following steps must be performed:

1. With Arduino connected to the working PC (through the USB cable) open the IDE
2. Go to Tools → Board and select the correct model of the Arduino in use
3. Go to Tools → Port and check if `/dev/ttyACM0` or `/dev/ttyUSB0` is present in the available ports. If not, there could be a problem with installed USB driver [1]
4. Open the provided sketch "rosserial_node_arduino.ino"
5. Use the Upload button to deploy the sketch

[1] Check the Arduino guide for further information

4.2.2 Connections

Once the Arduino has been properly programmed, it is necessary to make all the necessary connection between the involved components.

Connection to Raspberry
Connecting Arduino to Raspberry is done by mean of USB cable. This will enable serial communication between them, such that all command messages from the master will reach Arduino, but also sensor data messages will arrive to the master.

Connection of motors
Each stepper motor (right and left) has four Arduino pins reserved, as shown in Figure 2. The proper connection involves these steps:

1. Connect, following increasing order, the four reserved pins of Arduino to the IN1 to IN4, by mean of four wires. Respectively for the left and right motors, the connections would be: 8-9-10-11 to IN1 IN2 IN3 IN4 and 4-5-6-7 to IN1 IN2 IN3 IN4
2. Connect the flat cable of the motor to the respective socket in the driver
3. Choose the intended power supply voltage (5V - 12V), move the jumper in the respective position and connect the power supply, following positive and negative polarity. Refer to Figure 3, 1 and 2 highlight the stepper dedicated pins

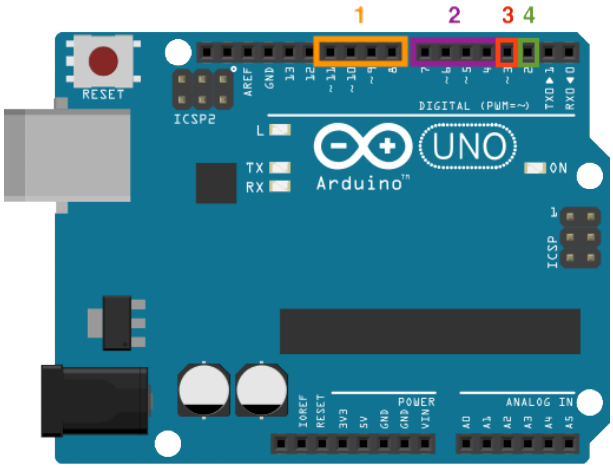

Figure 2: Arduino pins

Figure 3: Stepper motor and driver

Connection of IR Diode

5 Server

In this section description of how to install and how to launch the server are reported.

5.1 Install

The dependencies needed to make the server working are provided in a Docker image. Follow Docker installation guide provided in section 3, before proceeding.

5.2 Launch

Now that Docker is installed, the Docker image to launch the Server can be run issuing:

```
$ sudo docker-compose run server
```

If no error arises Server is now up.

6 Application

This section provides a description of system functions.
It is important to notice that previous setup steps as section 5.2 (for server to be up), section 4.1.2 (for raspberry to be up), section 4.2.2 (for arduino to be connected to raspberry) and section 4.2.1 (for arduino to proper handle robot hardware) should be performed in order for the user to access to a working environment.

6.1 Access

User can access to the application services through the Home web page.
From a browser connect to the following address: `laser_bot_master.local:5000`. The following page should be displayed.

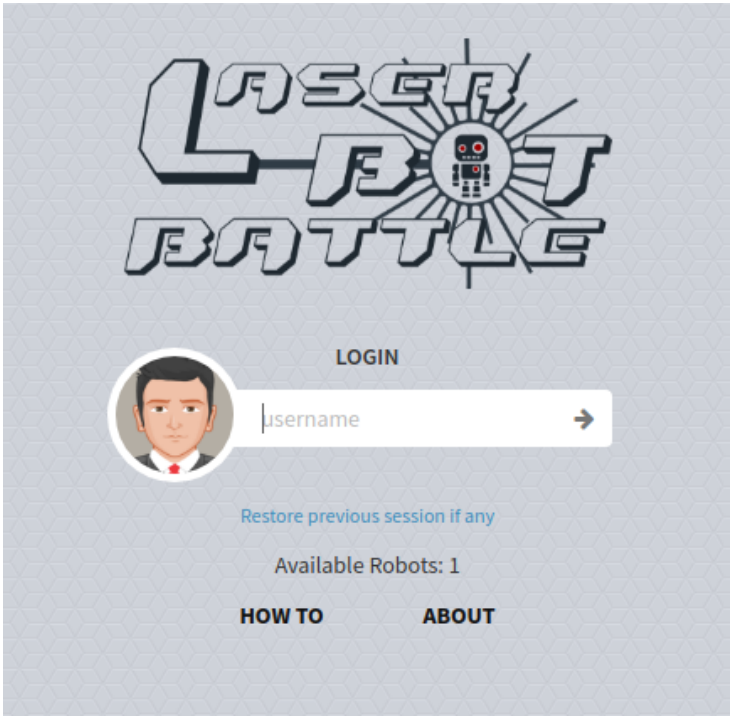

Figure 4: Index page - View

From this page, "HOW TO" alert and "ABOUT" page can be consulted.
The "HOW TO" alert reports information related to how to play the game, as the following image shows :

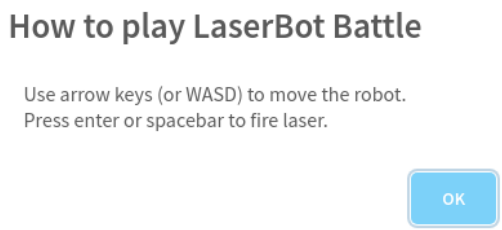

Figure 5: "How to" alert - View

The "ABOUT" page involves the team description, aims and motivation of the project.

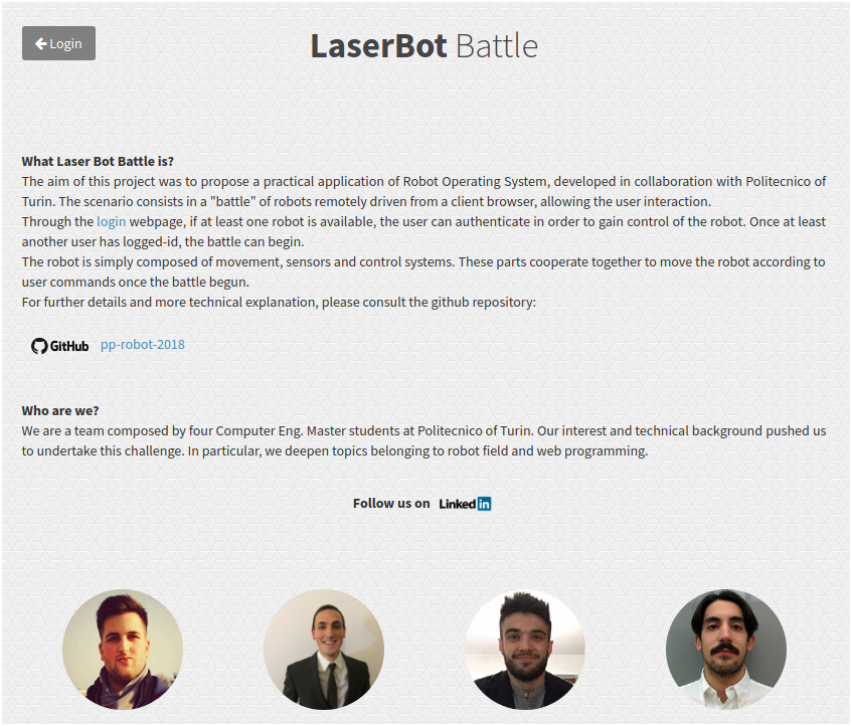

Figure 6: About page - View

In addition, the number of "Available Robots" signals to the user if there is the possibility to login to the application (section 6.2).

6.2 Login

User can login to the application after accessing the index web page (section 6.1).
If the number of "Available Robots" is greater than 0, the user can login to the page after having chosen a username and an avatar. Username has to be filled in the "username" form while the avatar is chosen by clicking on the "avatar" image. Respective errors as validation or uniqueness of the username are displayed as a pop-up message on the page, as for example the following ones :

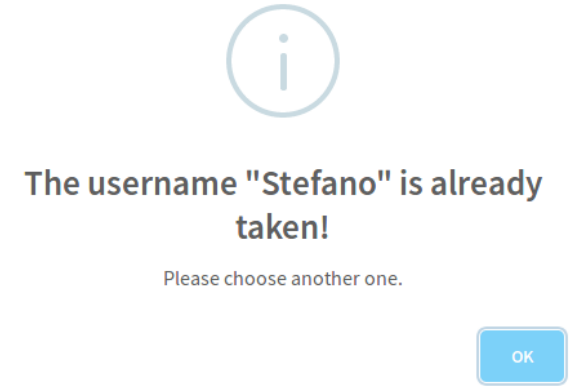

Figure 7: Login - Uniqueness error

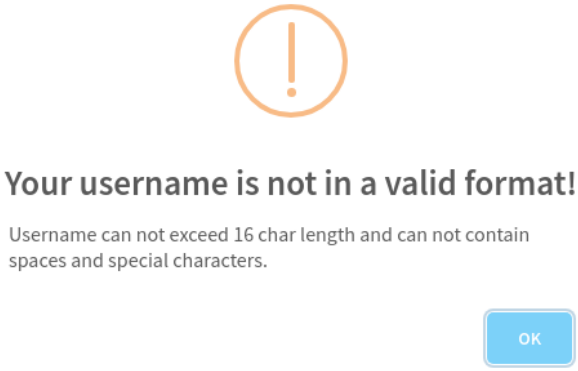

Figure 8: Login - Validation error

If the login is performed without any errors, a robot is associated to the user and the Home webpage, from which the user can be involved in a battle, should be get (section 6.3).
Moreover it is possible from this page to restore previous sessions by clicking on the respective link. If the user has already logged in without making any sign out, a session restoring is performed. It consists in a redirection to the Home page with all previous login information restored (as username and avatar).

6.3 Home

The Home is the page which is get after login.

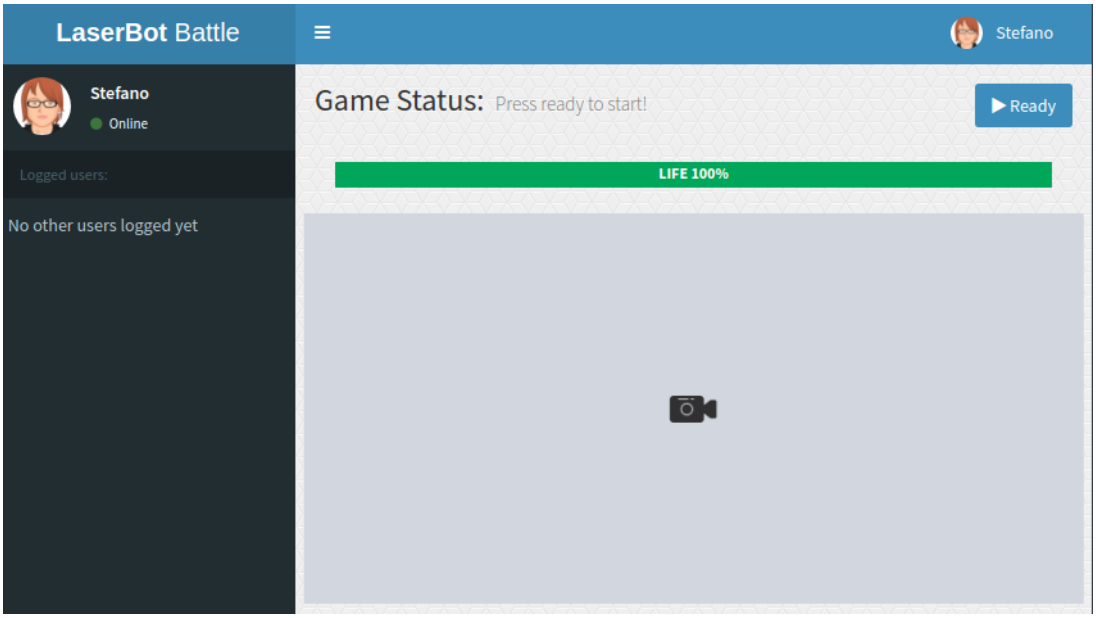

Figure 9: Home page - View

This is a personal page from which the user can logout, initiate a battle, consult users information and watch the streaming video coming from the associated robot camera.

6.4 Sign Out

By clicking on the right-top icon, a drop down menu appears, as the following one :

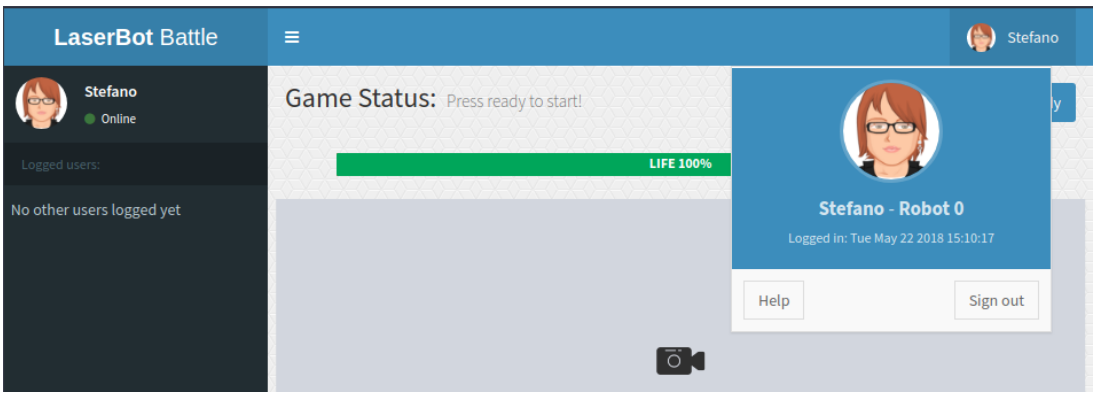

Figure 10: Personal menu - Home

From this menu the signout action can be performed. After signout, the user is logged out and redirected to the index page (Figure 4). Moreover, the "HOW TO" alert can be opened (Figure 5) to consult game instructions.

6.5 Users information

Users information can be retrieved on the left dark column.

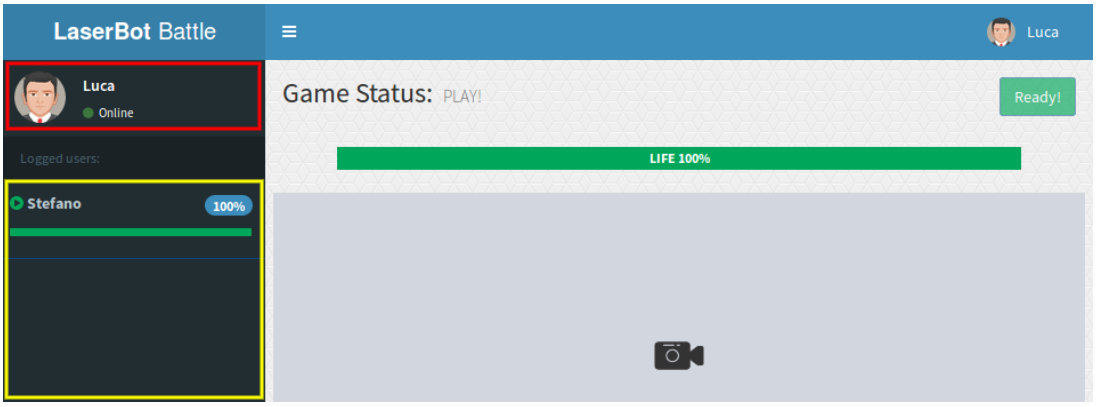

Figure 11: Users information - Home

Information reported are :

- **Personal info** (red shape on Figure 11) : At the top, personal user info as "username", avatar and status are reported.
- **Users list** (yellow shape on Figure 11) : Other users are listed here. For any of them the following attributes are reported :
  - **username** : Username of the user.
  - **status** : Status of the user
    - `Ready` : waiting for a battle to start.
    - `Online` : just logged in.
    - `Fight` : involved in a battle.
  - **life percentage** : percentage expressing the remaining life.
  - **lifebar** : bar expressing the remaining life.

6.6 Game

Game information can be retrieved on the right side column.

The top space is reserved to show the game status (red outline in Figure 12).

On the right side the user can press the Ready button (green outline in Figure 12) to set his status to ready (or from ready to unready).

Under it there is the user lifebar. Each user will start with 100% of health and each shoot received from another robot will decrease the hit robot health. When a user robot is dead (its life is at at 0%) the user has lost the game.

At the end of the game, when all robot but one are dead, a pop-up will appear showing the current user position and global rank. Dismissing this message, an user can begin another game setting again his status to ready.

6.6.1 Starting the game

If at least another user is logged (and ready), a 15 second countdown will start (see Figure 12). In this time other users can join the game but currently ready users can't change their status anymore.

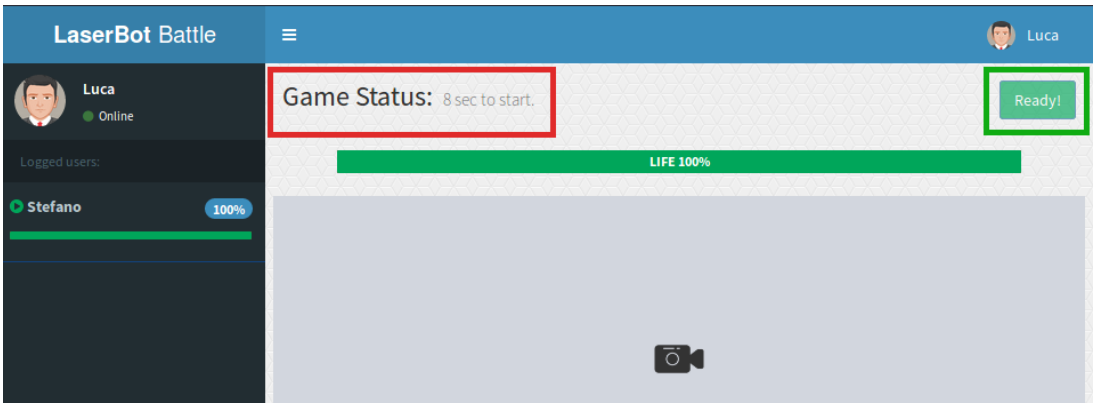

Figure 12: Countdown - Home

Once the game is started, the "Game Status" will change to "PLAY!" as the Figure 13 and the users will be able to control and move their respective robots.

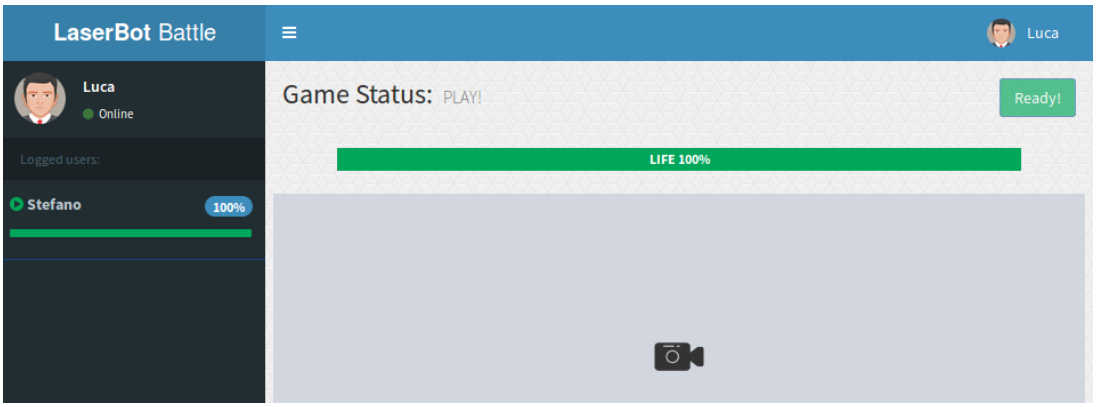

Figure 13: Play - Home

6.6.2 Controlling the robot

The user associated robot can be moved only during the PLAY status of the game.

To control the robot movements the user can use both the directional arrow on his keyboard or the `WASD` keys to respectively move the robot forward, left, backwards and right. Combination of keys (e.g. `W+A/D` or `S+A/D`) are allowed, permitting the robot to move in the resulting direction. Conflicting combinations of keys (e.g. `W+S` or `A+D`) result in no movement of the robot.

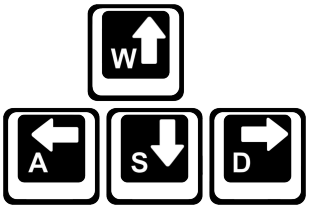

Figure 14: Robot controls - Home

To shoot, simply press the SPACEBAR or the ENTER keys.

6.6.3 Viewing the robot

The robot webcam stream will start automatically on the user login and will be displayed under the user lifebar.