METAL PUNCH

Planning and analysis of the development of a video game VR for Android externally controlled by gloves designed with Arduino

Content

PHASE I: Project identification

Virtual reality (VR) allows the immersion of the user in simulated environments that can be used for the acquisition or improvement of new competences, since it allows to represent scenarios that in real life could not be used to practice the skills of the subject, Examples would be the use of virtual reality environments to represent a fire or complex operation. It is also an important leisure resource for users of all ages.

In this context, which shows the importance and necessity of the existence of programmers with knowledge of virtual reality development, the creation of a boxing video game using VR is proposed, in which the players' movements will be captured by gloves created specifically for the game, which in turn will communicate with the game to transmit such information.

In support of the game, a REST-based web server will be created for storing player data, user authentication and the transmission and reception of information.

In addition, to facilitate user registration, as well as to inform about the game and its objectives, a web application will be created. This will also have the ability to draw a table with the ranking of users based on their scores.

PHASE II: Project Design

1. Objective and technical and legal feasibility of the project

The goal is to be able to simultaneously develop both an RV video game for android and gloves that serve as an external controller.

After analyzing the different tools and solutions available to developers, it has been concluded that there are currently a large number of methods in place to carry out the tasks set for the project without entailing a technical difficulty that is excessive enough to make the project impractical. This is thanks to the fact that there are currently many open-source projects that provide a lot of resources to both novice and expert programmers, which enables faster and easier learning.

The purpose of this project is to acquire the ability to work with some of the technologies most in demand in the market, as well as to implement the skills acquired throughout the higher level and to increase the knowledge of them.

Legally there is no impediment to its development, since all the software resources used are open source and are not protected against use or modification, except for Unity, whose free license cannot be used if with the developed game you get income or financing greater than 120,000 USD

2. Methodology

2.1 Organization of the workload

To organize the different activities to develop to complete the project, we have chosen to apply the methodology Waterfall or 'cascade'.

The sequence following this method consists of the following phases:

- Capture and documentation of requirements
- Design
- Development
- Test
- UATs
- Bug fixes and final adjustments

• Putting into production

The reason for this choice is that, as it is a non collaborative project, and not oriented to the marketing of the product, it has been considered preferable to opt for a linear structure of the stages of development, which makes it possible to track the progress of the project in a simpler way, since the scope is known in advance, also, since two clearly differentiated components (the video game and the control gloves) must be designed, a parallel development will facilitate the early detection of errors and requirements that could not be recognised in the design phases, which will provide more space to make any necessary changes.

3. Resource planning

3.1 Software

A REST-based web server will be created to store player data, allow user registration and authentication, and handle information sent by the game to modify user scores. It will also allow access to player information and ranking.

This web server will be created using Python and the Django and Django REST frameworks.

For the creation of the Django project, it is recommended to create a virtual development environment, to isolate the settings and that, in case you are working on several projects at the same time, each with its different configurations, there is no conflict.

Anaconda has been used for this. Anaconda Distribution includes Conda, an open-source system for managing environments and packages, among its features are the rapid installation of packages and their dependencies, as well as the ease of creating and using new virtual environments, tool to be used in this project.

Javascript and React JS will be used to create the web application.

ReactJS is based on a paradigm called component-oriented programming in which each component is a piece with which the user can interact. These pieces are created using a syntax called JSX allowing you to write HTML (and optionally CSS) within JavaScript objects, which makes it easier for the average developer to create new projects. Some of the main advantages of React JS is that it is open source, has the ability to accelerate application development, employs Active Reloading function (Hot Reloading), which allows the developer to implement changes to a code and see in real time how they affect the application.

The web application will be developed using Visual Studio Code.Se creará un contenedor haciendo uso de Docker que permita correr el servidor en una máquina virtual en Azure que se creará para este proyecto. Esta hará uso de una imagen de Ubuntu Server y contará con una dirección IP reservada para acceder al servidor.

The connection to the virtual machine will be via SSH and the MobaXterm tool will be used for this purpose.

Unity will be used for the development of the game, given the free resources it offers (interactive tutorials, free license), the possibility of developing virtual reality games for Android without using additional platforms and the number of options for development, animation and modeling that we will have.

To program the Arduino plates that will allow us to use the gloves to manage the movements of the players, the C++ language will be used, and Arduino IDE will be used as a development environment.

3. 2 Hardware

As for the hardware resources to be used, they will be the following:

- **Virtual reality glasses for Android compatible with GoogleCardboard.** Google Cardboard works by placing the phone at an optimal distance from the lenses and, using compatible applications, the lenses create a 3D effect. Using them you can even move your head and the images will respond to the movement, giving the impression to the player that he is immersed in the game. The reason to decide to use Google Cardboard is its easy configuration, its economic price and its functionality. While Google removed them from the market, there are still stores that sell compatible VR glasses.

- **Arduino One.** Arduino has been chosen as a microcontroller with a lot of support from its community (there is a lot of documentation and example projects that help the developer when making decisions in the design of the project) as well as having an economic price and all the possibilities it offers.

Initially, it is proposed to use 1 Arduino One per glove, one **MPU-6050** module (this module allows high performance with low energy consumption and cost; combines a 3-axis gyroscope and a 3-axis accelerometer and will be used for motion capture) and the Bluetooth **HC-06 module** for connection to Unity.

4. Activities

4.1 Obtaining the resources

For software-related resources, all except free-to-use Azure (including Unity, provided a personal license is used, MobaXterm, as long as Home Edition and Anaconda are used, provided the Distribution Edition is used), simply go to their download page.

UNITY: https://unity.com/download

ARDUINO IDE: https://www.arduino.cc/en/software

PYTHON: https://www.python.org/getit/

MOBAXTERM: https://mobaxterm.mobatek.net/download.html

ANACONDA: https://www.anaconda.com/products/distribution

VISUAL STUDIO CODE: https://code.visualstudio.com/download

For the creation of the Azure virtual machine, steps will be indicated in the Creation of the Azure virtual machine section of this memory. It should be noted that, although it is not free, Azure Students offers $100 initial credit, which is what has been used for this project.

To obtain React, the steps to be followed will be indicated in the section Creation of the website .

In order to obtain the resources related to the hardware, various stores have been chosen, taking into account the quality offered and the prices of the products:

VR GLASSES: https://www.amazon.es/Gafas-realidad-virtual-Google-Cardboard

PLACA ARDUINO UNO: http://store.arduino.cc/products/arduino-uno-rev3

MÓDULO MPU-6050: https://es.aliexpress.com/item/1734534460

MÓDULO HC-06: https://www.amazon.es/Modulo_Bluetooth

PROTOBOARD: https://www.amazon.es/Breadboard-Arduino

JUMPER WIRE CABLES: https://www.amazon.es/Jumper-Wire-Cables

5. Sequencing

5.1 Steps

The steps to follow are determined by the chosen methodology, however with some modifications, since the project will not be put into production, and neither will the UAT's, since there is no end customer in this case, and, given the project deadline, the tests have been dispensed with; thus the final phases would be the compilation of the results obtained and the preparation for the defense of the project, so that the project would be developed in the following phases:

- Capture and documentation of technical requirements and materials
- Design of the project structure
- Development
- Development of the database
- Simultaneous development of the different components of the project
- Bug fixes and final adjustments
- Collection and evaluation of results
- Preparing the defense of the project

6. Evaluation

6.1 Tools and methods used to evaluate project performance

To evaluate the final outcome of the project, it will be taken into account if the objectives set have been met, these being the creation of the server, the creation of the website, the development of video game and the creation of gloves.

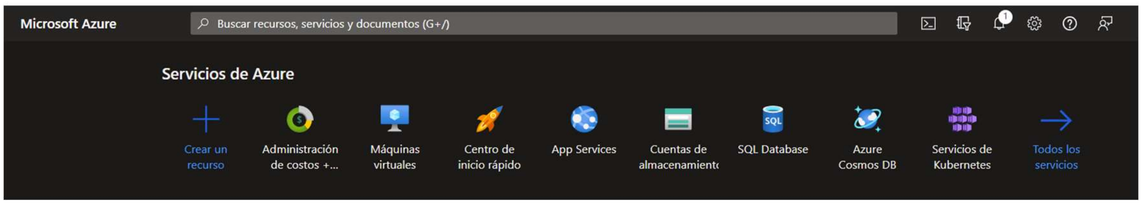
It has therefore been considered important, in view of the expected delivery dates of the project, to achieve the objectives mentioned above, to broaden knowledge about the programmes, tools and languages used and acquire the ability to perform the proposal, approach and analysis efficiently.

PHASE III: Development of the project

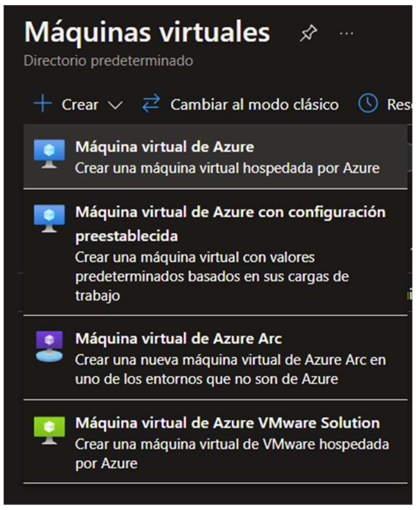
1. Web Server Development

1.1 Azure Virtual Machine creation

First you must access the homepage of the Azure portal https://portal.azure.com/#home. Once there, in the Azure services section, select the 'Virtual machines' option:



Once in the virtual machine directory, select the create option and, in the drop-down that appears, the 'Azure virtual machine' option:


Máquinas virtuales
Directorio predeterminado
Crear
Cambiar al modo clásico
Res
Máquina virtual de Azure
Crear una máquina virtual hospedada por Azure
Máquina virtual de Azure con configuración preestablecida
Crear una máquina virtual con valores predeterminados basados en sus cargas de trabajo
Máquina virtual de Azure Arc
Crear una nueva máquina virtual de Azure Arc en uno de los entornos que no son de Azure
Máquina virtual de Azure VMware Solution
Crear una máquina virtual de VMware hospedada por Azure

The basic data of the new virtual machine must be filled in on the virtual machine creation form. An image of Ubuntu Server 20.04 LTS has been chosen this time, but there are several valid options that could be selected.

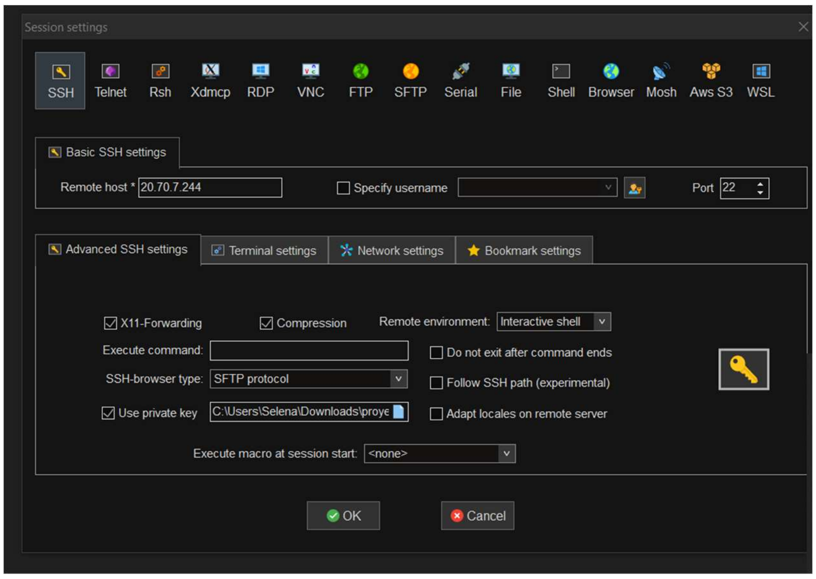
It is necessary, for the subsequent connection with the machine, that in 'Type of authentication' is chosen SSH Public Key.

Once the machine creation is finished, the private SSH key must be downloaded.

1.2 Connecting to the Azure virtual machine

To connect to the previously created virtual machine, you must access the MobaXterm application.

In the tools menu, select the 'Session' tab, a modal will appear in which the connection method must be chosen (in this case SSH). After selecting it, the advanced tools must be enabled, and the form will be filled in as follows:

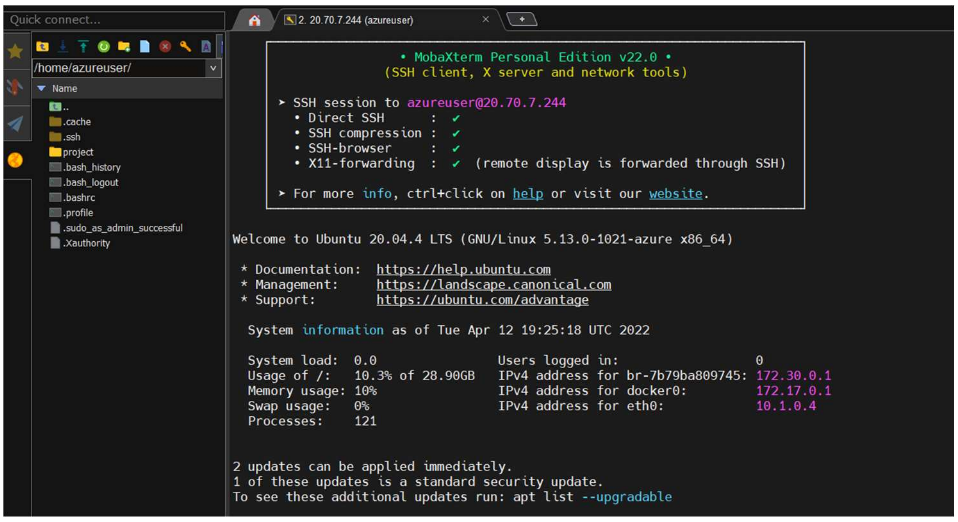


Remote host * 20.70.7.244
☐ Specify username
Port 22

☑ X11-Forwarding
☑ Compression
Remote environment: Interactive shell
Execute command:
☐ Do not exit after command ends
SSH-browser type: SFTP protocol
☐ Follow SSH path (experimental)
☑ Use private key C:\Users\Selena\Downloads\proye
☐ Adapt locales on remote server
Execute macro at session start: <none>

'Remote host' being the IP address of the azure machine and 'Port' the port number by which it will be accessed. Also, select the 'Use private key' option and select the file downloaded after the creation of the virtual machine, which contains the private access key.

Once this is done, a session will be created that will allow us to view, create, modify and delete items within the virtual machine. A window like the following will appear:



```
              • MobaXterm Personal Edition v22.0 •
            (SSH client, X server and network tools)

 ➤ SSH session to azureuser@20.70.7.244
   • Direct SSH      :  ✓
   • SSH compression :  ✓
   • SSH-browser     :  ✓
   • X11-forwarding  :  ✓  (remote display is forwarded through SSH)

 ➤ For more info, ctrl+click on help or visit our website.

Welcome to Ubuntu 20.04.4 LTS (GNU/Linux 5.13.0-1021-azure x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Tue Apr 12 19:25:18 UTC 2022

  System load:  0.0               Users logged in:                 0
  Usage of /:   10.3% of 28.90GB  IPv4 address for br-7b79ba809745: 172.30.0.1
  Memory usage: 10%               IPv4 address for docker0:        172.17.0.1
  Swap usage:   0%                IPv4 address for eth0:           10.1.0.4
  Processes:    121


2 updates can be applied immediately.
1 of these updates is a standard security update.
To see these additional updates run: apt list --upgradable
```

In which the screen will be split in two, on the left side you will find the tree view of the virtual machine elements and on the right side the terminal.

1.3 Creation of server and database

This can be done in two different ways. The first option would be directly on the Azure virtual machine, and the second would be on the local computer.

Since in MobaXterm files can be dragged from the local computer and placed directly in the Azure folders, both options will give the same result.

In this case, the second option has been chosen. First, it must be verified that python is installed. To do this, the terminal is accessed and the following command is executed:

python –version

In case it is not installed, it can be downloaded from the official website, whose link can be found in the Resources section of this memory.

Once installed, access the Anaconda Prompt, the command console on which this project will work.

The commands for the creation and activation of the environment will be the following:

--Create environment with name 'dev' (this can be replaced with the preferred name):

conda create --name dev

--Activate dev enviroment

conda activate dev

Once the second command is executed, it will enter the created environment. At this point, you will proceed to the creation of the Django project, to do this, in the console, you must be located in the folder in which the project will be created and execute the following commands:

--Installl django:

pip install django

-- Creation of the project structure:

django-admin startproject backend

-- Create application for models, views and url of players (named players):

python manage.py startapp players

The necessary modules for the development of the project are specified in the 'requirements.txt' file that is attached with the project code.

For the creation of the database the following aspects are highlighted:

• The User model of Django's default authentication system has been used.

• sqlite has been used for database creation (the database system provided by Django by default).

• A PlayerInfo model has also been created, which relates to User in relation to 1:1.

• The PlayerInfo model contains the user id, which it uses for relationship, player id, score and three methods, one to add score, one to subtract score, and one to convert it to String.

It has also used the Django REST framework that allows a simple and fast creation of REST APIs. This is what will allow us to get information from players.

The following URLs have been specified in the project:

--Create user:

```
players/create/<str:name>/<str:email>/<str:password>


        # ejemplo: players/create/selena/email@email.es/admin
```

You must first pass the string that will correspond to the username, second the email and finally the password.

-Log in:

```
players/login/<str:username>/<str:password>
```

```
        #ejemplo: players/login/myname/mypassword
```

In this case you are given the username and password of the same.

--Log out:

```
        #ejemplo: players/logout

players/logout/
```

In this case you should not pass any additional parameters, as you take it from the user's session.

--Add/Subtract score:

```
        players/score/<str:is_win>/<int:difficult_level>/

        # ejemplo 1: /players/score/win/1

        # ejemplo 2: /players/score/lost/2
```

To determine if you should add or subtract score, and how many points should be modified, first you identify the value of the first parameter (which can be win or lost), if the value is 'win', is that you have to add points, and if the value is 'lost', you have to subtract them.

As the second parameter you pass the difficulty level, which goes from 1 to 3. Being 1 the easy level, 2 the normal level and 3 the difficult level. The user determines the session.

The distribution of points is as follows:

If you win :

• Easy level: 10 points are added.

• Normal level: 20 points are added.

• Difficult level: 30 points are added.

If you lose :

• Easy level: 5 points are deducted.

• Normal level: 8 points are deducted.

• Difficult level: 10 points are deducted.

URL’s to access the API:

--View all the players:

```
players/api/players
```

--View all users:

```
players/api/users
```

--View a selected user by id:

```
players/api/players?user_id=1
```

In this case the user id is passed as parameter.

--View a selected user by name:

```
players/api/user?username=selena
```

In this case the username is passed as parameter

--View the players ranking:

```
players/api/ranking
```

1.4 Creation of Docker Container

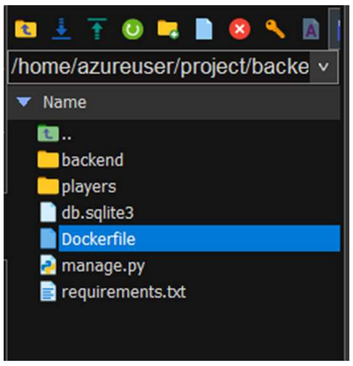
For this phase of the project you must be located in the terminal of the virtual machine created in Azure, as indicated in the section Connection to the Azure virtual machine .

First, it is necessary to locate in the folder that will store the project. In this case the structure is as follows:

-Project

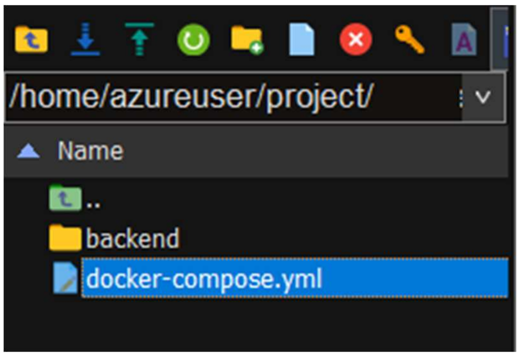
backend

A file called dockerfile must be created in the backend folder. It will be blank for now.



```
/home/azureuser/project/backe
Name
..
backend
players
db.sqlite3
Dockerfile
manage.py
requirements.txt
```

In the Project folder you will create a file called docker-compose.yml, also blank.



```
/home/azureuser/project/
Name
..
backend
docker-compose.yml
```

After that, you must install docker, to do so will make use of the following commands:

First, update existing packages.

```
sudo apt update
```

Then, install the dependencies

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

The next command will add the GPG key

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

Add the required repository::

```
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
```

Install docker:

```
sudo apt install docker-ce
```

Docker Compose will also be used for this project. This is a tool that allows you to run multi-container application environments based on definitions set in a YAML file. It uses service definitions to create fully customizable environments with multiple containers that can share networks and data volumes.

First, we will perform the installation of Docker Compose:

Command to download the current stable version:

```
curl -SL https://github.com/docker/compose/releases/download/v2.2.3/docker-compose-linux-x86_64 -o ~/.docker/cli-plugins/docker-compose
```

Apply the necessary permissions to make the docker compose command executable

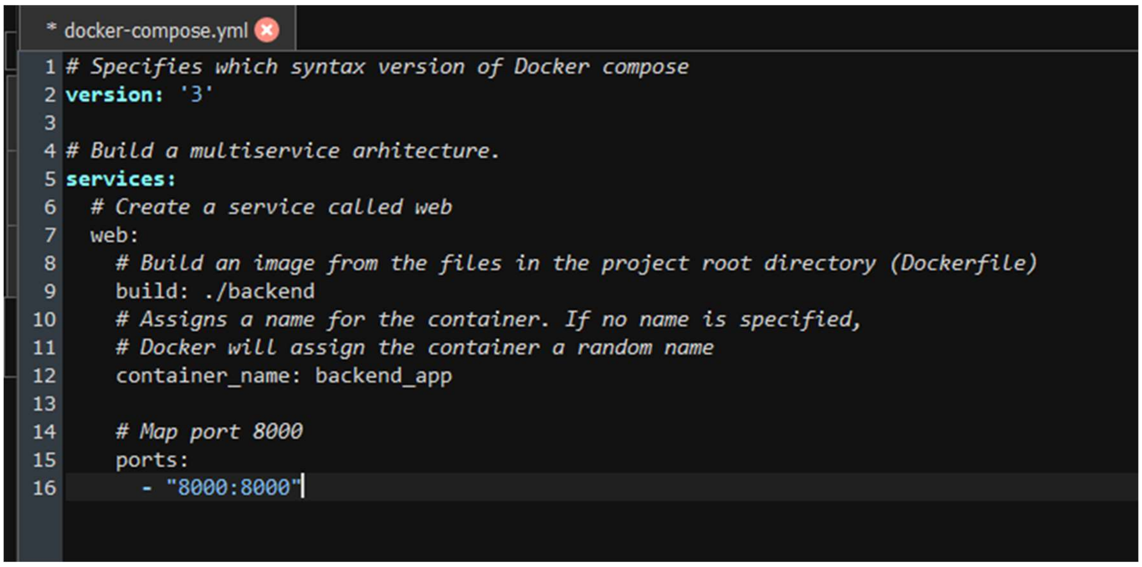
```
chmod +x ~/.docker/cli-plugins/docker-compose
```

The docker-compose.yml file will then be edited as shown in the screenshot below.

First specify the version of Docker Compose to be used.

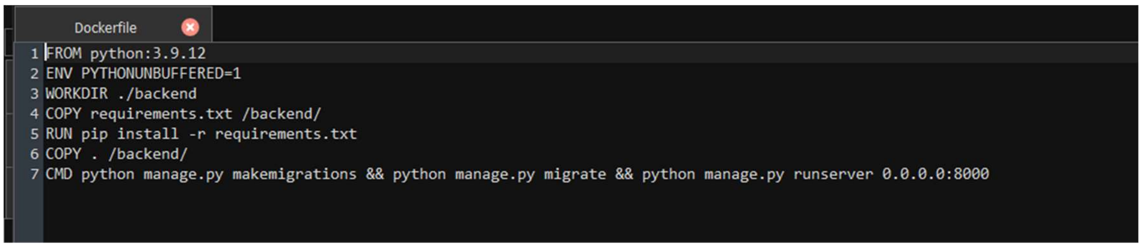
After that, a service will be created which will be called 'web', which will use the image that will be created later in the aforementioned Dockerfile file.

You will assign a container name, which can be any, in this case backend_app. After that you will indicate that port 8000 will be used.



```yaml
# Specifies which syntax version of Docker compose
version: '3'

# Build a multiservice arhitecture.
services:
  # Create a service called web
  web:
    # Build an image from the files in the project root directory (Dockerfile)
    build: ./backend
    # Assigns a name for the container. If no name is specified,
    # Docker will assign the container a random name
    container_name: backend_app

    # Map port 8000
    ports:
      - "8000:8000"
```

Now, in the Dockerfile, write the following:



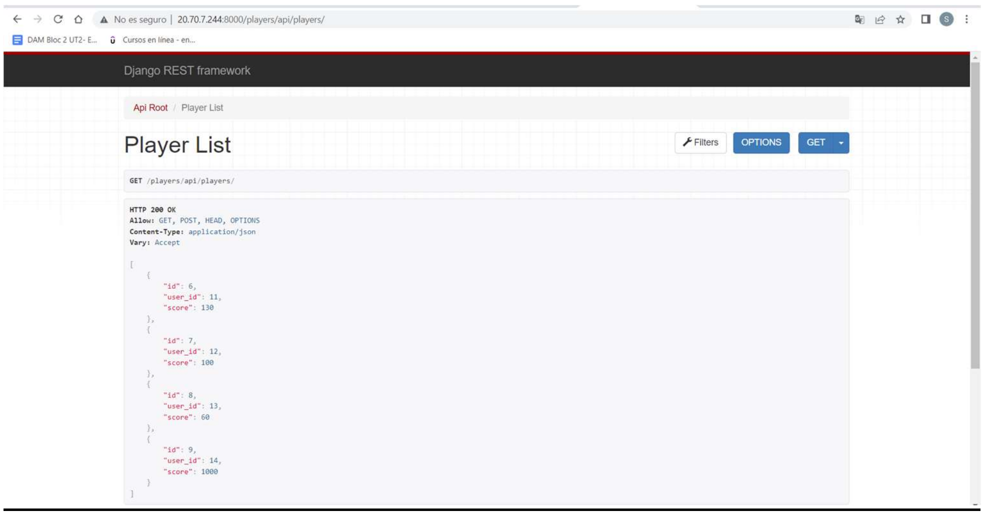
```
FROM python:3.9.12
ENV PYTHONUNBUFFERED=1
WORKDIR ./backend
COPY requirements.txt /backend/
RUN pip install -r requirements.txt
COPY . /backend/
CMD python manage.py makemigrations && python manage.py migrate && python manage.py runserver 0.0.0.0:8000
```

This indicates the version of Python that will be used and the directory in which it will work. You will be prompted to copy and install the contents of the requirements.txt file, copy to the project you are working on, and launch commands to perform database migrations and run the server on port 8000.

Finally, you have to go to the Project folder and run the following command:

sudo docker-compose up –d

From now on the container will be created and active. From the local computer, you can check by accessing the IP of the Azure machine, along with one of the URLs specified on the server.



```
GET /players/api/players/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 6,
        "user_id": 11,
        "score": 130
    },
    {
        "id": 7,
        "user_id": 12,
        "score": 100
    },
    {
        "id": 8,
        "user_id": 13,
        "score": 60
    },
    {
        "id": 9,
        "user_id": 14,
        "score": 1000
    }
]
```

2. Creation of the website

We have chosen to create a basic website that allows users to view information about the game, the ranking of players, with their respective scores, and the registration of new users.

Javascript has been used, along with its React framework, given its simplicity and ease of use for new users.

React JS also facilitates the collection of information in real time, and since version 16.8, Hooks were introduced.

A Hook is a special function that allows you to "connect" to React features.

This allows us to facilitate obtaining data through the API, determine whether a user should display a screen or another, etc.

It has also used Material UI, a library of components for React, which helps build web pages more intuitively and quickly. The components created with Material UI offer the developer the possibility to create more attractive designs using less code and time.

The steps are as follows:

2.1 Configuration and start of the project

To create the web, we will use Create React App, this configures your development environment to use the latest javascript features and optimizes your application for production environments.

First, it is essential to have installed Node >= 14.0.0 and npm >= 5.6.

Once this requirement is met, the following commands must be executed to create a new project:

```
npx create-react-app app

cd app

npm start
```

With this the project would be created and we could start with development.

2.2 Development of application components

In React, components are stand-alone elements that can be reused on the page.

In this way, we could use the same FormComponent component to create several different forms, thus using fewer lines of code and resources, also favoring subsequent changes. If you wanted to add one more form, there would be very few lines of code to add, and it would not affect forms used elsewhere on the website.

In the case of this application it has been decided to create a Table component, which allows the construction of tables in a simple and fast way, as well as the ordering of the same according to the indicated columns.

To do this, a folder has been created inside the app/src folder, called Components, and inside a file called TableComponents.

It is important at this point to note that, to export a function or component to another part of the application, it can be indicated in two ways:

1. Creating the function and, at the end of the file, add the export default MyFunctionName formula

2. When creating the function, add the words export default above, for example, export default function MyFunctionName(){}

In this case, the function has been created as follows:

```
export default function TableComponent(data)
```

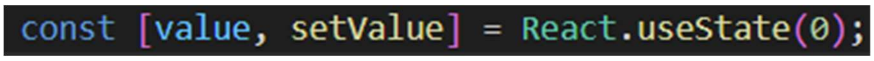
Where 'data' is the parameter that will be passed when using the component in the application.

1.3 Creation of the application views

In the application there will be 3 different views, as already mentioned before, the first one will show a small summary of the game and its objective, the second one a table with the ranking of players and the third one a registration form.

The app/src/app.js.

In order for the change between functions views, a hook must be created using the React.useState() function, passing it as parameter 0, which would be the initial value of the tab in which the user is located.



```
const [value, setValue] = React.useState(0);
```

useState is a Hook that allows you to add React status to a component of function.

The only argument for the Hook useState() is the initial state. Unlike classes, the state does not have to be an object.

UseState returns a pair of values: the current state and a function that upgrades.Several state hooks are used in this code, but we will only need this to determine the view displayed.

We will also add a function, HandleChange, which will be called when the user change view. The new value will be passed as a parameter, and will use the setValue() status to modify the view value, thus allowing the content change.

The contents of each view will remain hidden until the user selects the for this purpose, a class will be created in the app/src/App.css file, called 'hidden', which has a single property, display, whose value is 'none'.

An example of this would be in the player ranking view:

```
        <div style={{ marginTop: '3%', marginBottom: '5%', backgroundColor: 'white', width:
'80%', marginLeft: '10%', height: 300 }}

          className={value == 1 ? '' : 'hidden'}>



          <TableComponent rows={rowsData} users={usersData} headCells={headCells} />



        </div>
```

To allow the user to select each view, a bar will be displayed

navigation that will toggle between the three views, and that will call the above

HandleChange function mentioned to update the new view value.

This would then be the code for alternating views:

```
          <Tabs value={value} onChange={handleChange} sx={{ marginTop: 3 }}>

            <Tab label="Inicio" />

            <Tab label="Ranking" />

            <Tab label="Registro" />

          </Tabs>
```

1.4 API Related Functions

In order for the application to comply with the proposed functionality, it must be able to access the API.

The functions that obtain and/or modify the necessary data are the following:

1.4.1 Function to get the players ranking

This function will return a collection in json format of the player ranking. , making use of the URL created during the development of the Rest API.

The function would be the following:

```
//get the ranking data

const getRankingData = async () => {


  var api_url = 'http://20.70.7.244:8000/players/api/ranking/'

  const dataResponse =

    await fetch(api_url);

  const data = await dataResponse.json();

  //the result will be our rows for the ranking table

  setRowsData(data)


}
```

As can be seen in the last line, it uses the function called setRowsData() by passing the player data as a parameter. This is a function to update the value of the Hook state rows which, as can be seen in the application code, is passed as a parameter when invoking the <TableComponent/> component in the application ranking view.

1.4.2 Function to get users

```
      //users data

      const getUsers = async () => {



        var api_url = 'http://20.70.7.244:8000/players/api/users/'

        const dataResponse =

          await fetch(api_url);

        const data = await dataResponse.json();

     //we will use the users in order to determinate the player's  username and to make the
player registration

       setUsersData(data)

      }
```

This function is used to obtain registered users, as in the previous function, it updates a status Hook, in this case users, which is passed as a parameter to the <TableComponent> and which allows to determine the username of the player according to the userid of the same.

An example of the code is attached to clarify this last point:

```



     {stableSort(rows, getComparator(order, orderBy))

          .slice(page * rowsPerPage, page * rowsPerPage + rowsPerPage)

            .map((row, index) => {

              const labelId = `enhanced-table-checkbox-${index}`;
```

```
                if (users != undefined && rows !=undefined) {
                  if (users != [""] && rows !=[""]) {
                  if (users.length > 2 && rows.length>2) {
                          stableSort(users, getComparator(order, "id"))
                       .map((user) => {
                           if (row.user_id == user.id) {
                            row.username = user.username
                                  }
                                })
                              }
                            }
                          }
```

Hook users are also used when registering a new player to ensure that neither the username nor password is being used.

2.4.3 Function to register new users

This function is passed as a parameter the username, password and email of the new user to register.

```
      //function to register a new user, we will pass the user, email and password
      async function registerUser(username, password, email) {


        var api_url = 'http://20.70.7.244:8000/players/create/' + username + '/' + email + '/' +
password
         const dataResponse =
            await fetch(api_url);

        }
```

Prior to the feature call, checks must be made to ensure that the username and email have not been previously used by another player.

Two functions are used.

The first, HandleSubmit, obtains the form data, verifies that it contains the necessary data and, if so, uses the second formula, isInList, which receives a value and a key as a parameter, to validate the uniqueness of the data.

In case the form data is valid, the function will be called to register new users, and it will be indicated that the user has been registered to show a confirmation image. Otherwise, the error shall be indicated.

HandleSubmit function :

```
        //function to handle the submited form

          const handleSubmit = (event) => {

            //first we get the data from the inputs

            var username = document.getElementById("username-field").value

            var password = document.getElementById("password-field").value

            var email = document.getElementById("email-field").value

            //validate the data

            if (username != null && username != "" && password != null && password != "" && email
!= null && password != "") {

              //we will use our predefined function in order to know it the user or the email are already
in use,

              //if is the case, we will show the error in the corresponding field

              if (isInList(username, 'username')) {
```

```
          setUsedUser(true)

        } else {

          setUsedUser(false)

          if (isInList(email, 'email')) {

            setUsedEmail(true)

          }

            //if everything is correct we send the data to our register function and set the user registered state at true in order

            //to show our welcome message and let the user know that they registered without problems

          else {

            setUsedEmail(false)

            registerUser(username, password, email)

            setRegistered(true)

          }

        }



      }

    }
```

Function isInList:

```
  //function to know if an element is in a list, we will pass the key and the value to find

  function isInList(valueToFind, key) {

      //since the functions to get the data could return a promise, first we need to verify that there has been no errors before

     //mapping the array or it will throw us an error
```

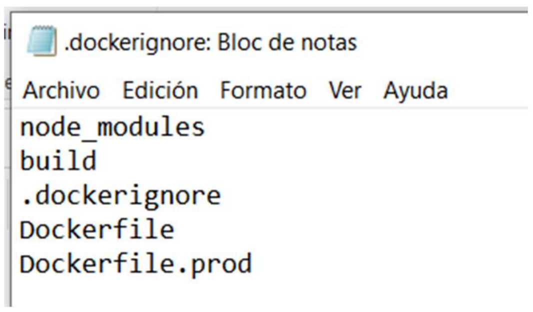
```
    if (usersData != null && usersData != [""] && usersData.length > 2) {
      var isInList = false
      usersData.map((user) => {
        if (user[key] == valueToFind) {
          isInList = true
        }
      })
    }
    return isInList
  }
```

1.5 Creating Docker container to store the application

First, before you start creating the other files needed to launch the application container, you must configure a .dockerignore.

The. dockerignore file works similar to the archive. gitignore that is usually created when working with GIT, and indicates that certain files or folders must be ignored.

In this case, the contents of the file should be as follows:


.dockerignore: Bloc de notas
Archivo Edición Formato Ver Ayuda

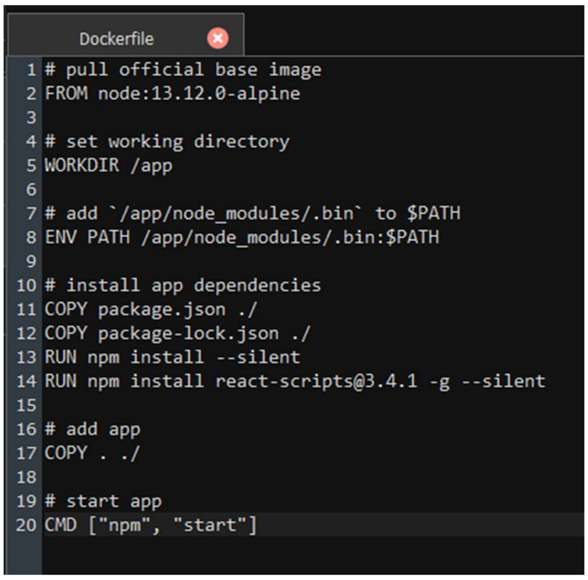
```
node_modules
build
.dockerignore
Dockerfile
Dockerfile.prod
```

The node_modules folder contains all modules used in the React application. This folder does not need to be attached when you send or upload a project with React, as it takes up a lot of storage space and slows the project upload or upload. Also, running the npm install command installs the required modules defined in the package.json and package-lock.json files.

1.5.1 Creating the Dockerfile

As during the creation of the web server, we must create a Dockerfile file to create a Docker container. This container will be built and uploaded along with the backend container, as will be seen later in the modification of the docker-compose file.

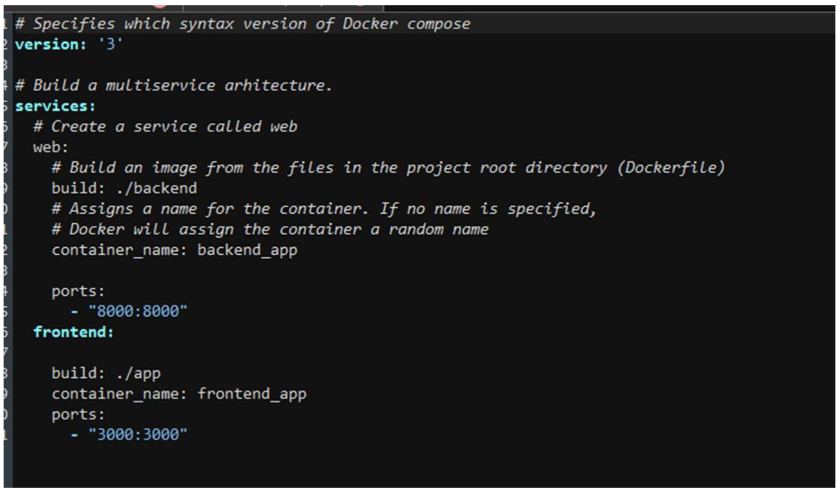
The Dockerfile file, located in the project/app/ path, will have the following content:



```
# pull official base image
FROM node:13.12.0-alpine

# set working directory
WORKDIR /app

# add `/app/node_modules/.bin` to $PATH
ENV PATH /app/node_modules/.bin:$PATH

# install app dependencies
COPY package.json ./
COPY package-lock.json ./
RUN npm install --silent
RUN npm install react-scripts@3.4.1 -g --silent

# add app
COPY . ./

# start app
CMD ["npm", "start"]
```

1.5.2 Modification of the docker-compose file

Finally, the contents of the docker-compose.yml file under the project/ folder should be modified, updating it as follows:



```
# Specifies which syntax version of Docker compose
version: '3'

# Build a multiservice arhitecture.
services:
  # Create a service called web
  web:
    # Build an image from the files in the project root directory (Dockerfile)
    build: ./backend
    # Assigns a name for the container. If no name is specified,
    # Docker will assign the container a random name
    container_name: backend_app

    ports:
      - "8000:8000"
  frontend:

    build: ./app
    container_name: frontend_app
    ports:
      - "3000:3000"
```

1.6 Final result

Once we have done the above steps, we will be able to verify that the website will work and be accessible from anywhere using the http://20.70 route. 7.244:3000/ .

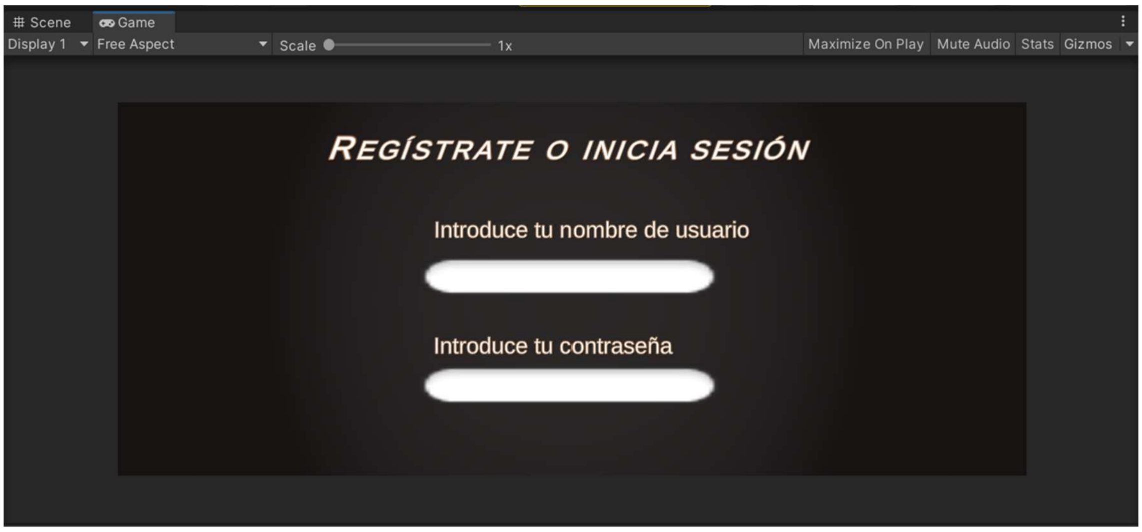
Needless to say, the route will be different depending on the IP of the Azure machine running as a server

2. Videogame development

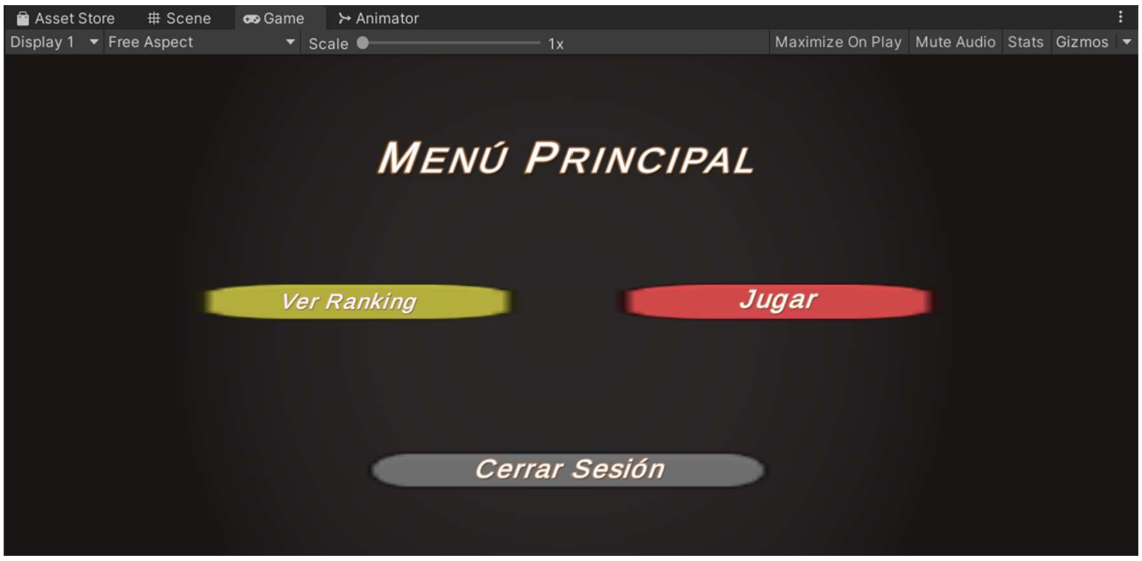
For the development of the game has chosen to use Unity, as explained above, provides a number of facilities to the user to create virtual reality video games compatible with Google Cardboard that have been considered optimal for being simple, free and varied.

In a video game developed with Unity, various scenes must be provided, each of which serves a specific purpose, so the first step has been to realize the creation of such scenes, which, taking into account the requirements and objectives of the video game, are as follows:

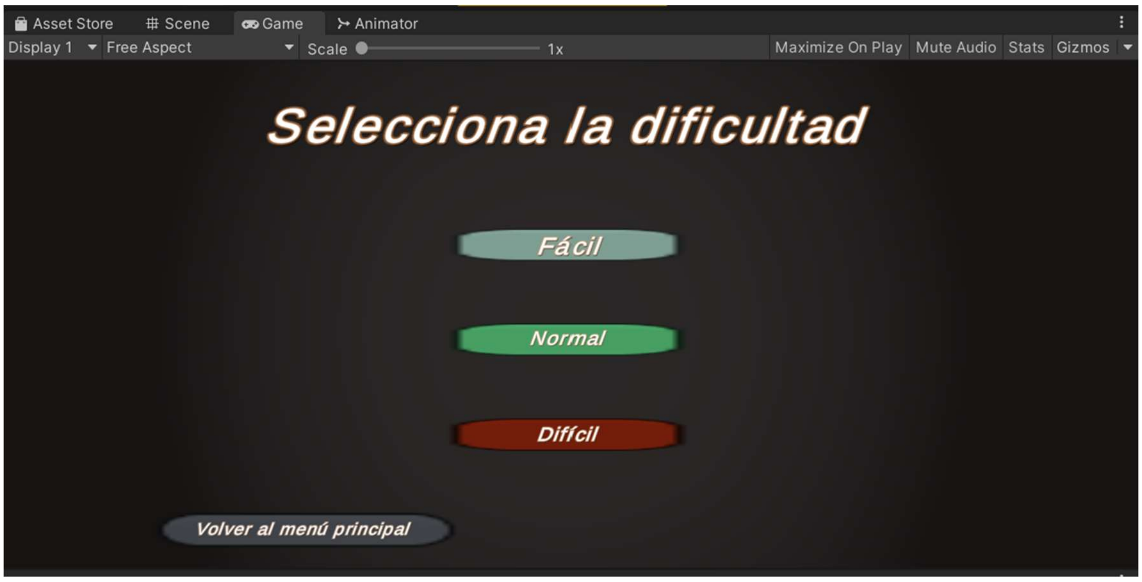
1. Initial screen, where the user's login is performed if the user is already registered. Otherwise, you will be redirected to the game website to perform it.



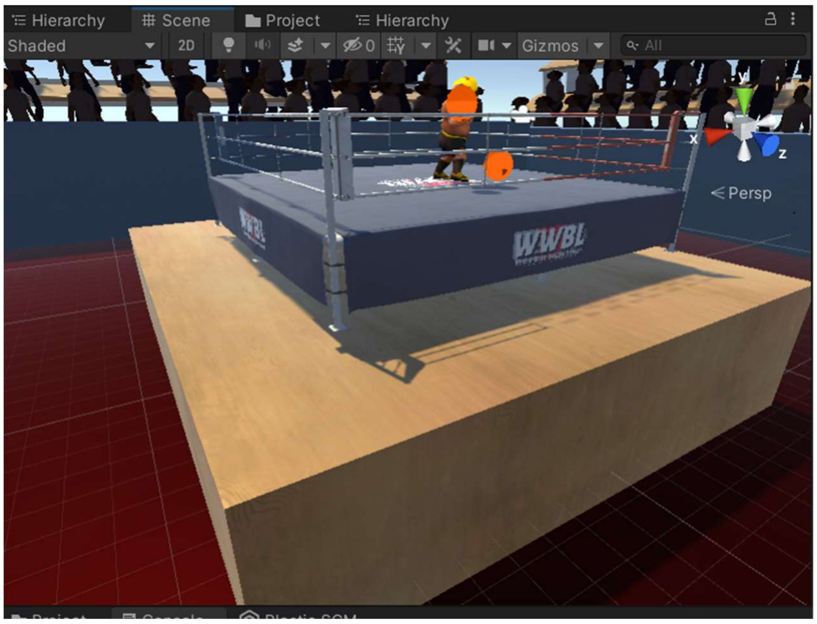
2. Main menu screen. In this the user will choose between playing, logging out or viewing ranking. In case the latter option is selected, the player will be redirected to the registration website

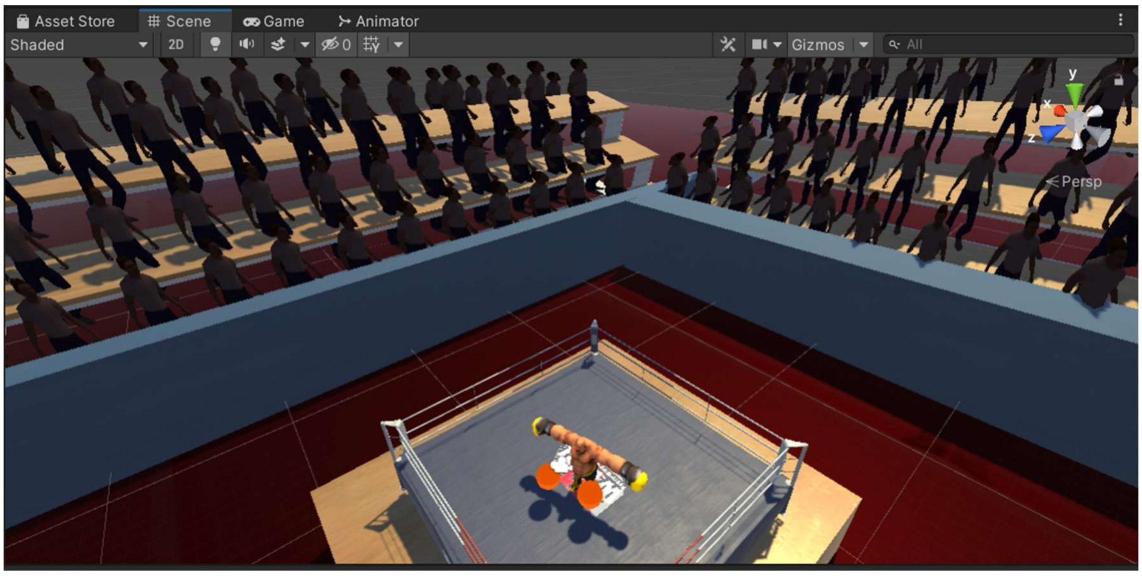


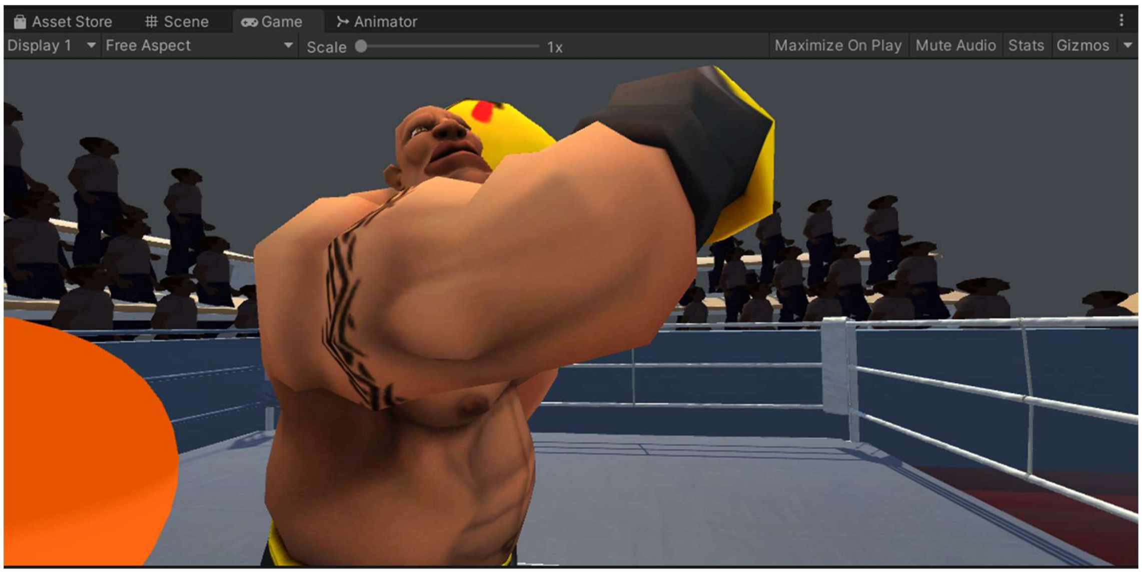
3. Difficulty selection screen. If the user chooses to play, they will be redirected to this screen, in which they will be able to select between the three available difficulties: Easy, Normal and Difficult.



4. Game screen. This screen will be where the user can play. It has sought to simulate a stadium, incorporating physical elements such as stands, people, a boxing ring and walls. Also, an enemy has been created before which the player must face, and whose speed of movement will be determined by the difficulty previously chosen, as well as the time between changes of movements.


Hierarchy
Scene
Project
Hierarchy
Shaded
2D
Gizmos
All
Persp
WWBL


Asset Store
Scene
Game
Animator
Shaded
2D
Gizmos
All
Persp


Asset Store
Scene
Game
Animator
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos

3. Gloves design

4.1 Physical components

Finally, only the design of the gloves would be needed to control the position of the player and send the data to Unity.

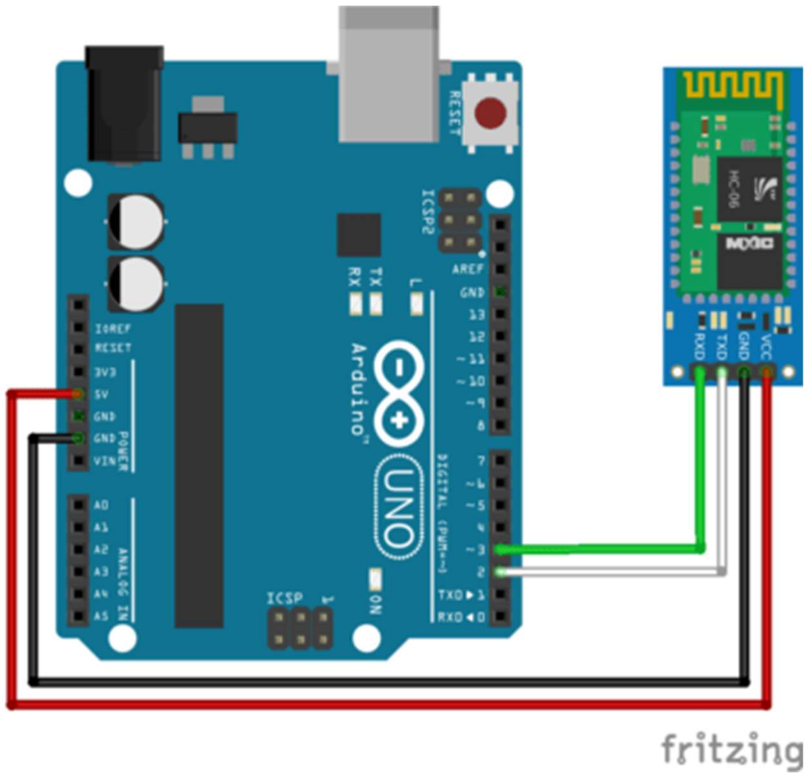
Initially different alternatives were considered, given the budget initially available, an ESP01 microcontroller was chosen.

Finally, it was decided to use a Bluetooth module to make communications with Unity. Compared to the ESP01 microcontroller, it is easier to program, as indicated below connections are simple and fast. As an additional advantage, its small size allows a better fit to the gloves.

Other advantages of using this module would be that it has good characteristics of transmission and reception of information, as well as the low current consumption it has both in operation and in standby mode.

Module HC-06 has 4 pins, two of them for power and the other two for connections.

The connection scheme would be as follows:



HC-06	**ARDUINO**
VCC	5V
GND	GND
RX	TX(3)
TX	RX(2)

On the other hand, we must also consider connections with the MPU-6050 module.

This is a sensor containing a MEMS accelerometer and a MEMS gyroscope embedded in the same chip. A MEMS (MicroElectroMechanical Systems) works in a similar way to a spring-loaded mass system, so that acceleration can be measured.

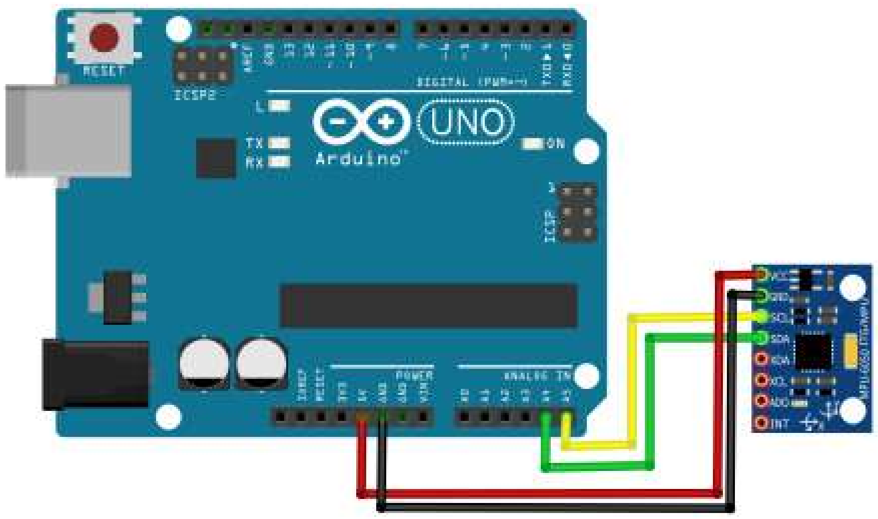
It also contains 16-bit analog-to-digital conversion hardware for each channel, allowing you to capture 'X', 'Y', and 'Z' channels at the same time.

The communication of this module is by I2C, this allows you to work with most microcontrollers.

The reason for choosing this module is that, by incorporating at the same time a three-axis gyroscope and a triaxial accelerometer, it is a great choice when developing applications and video games that incorporate virtual reality.

In this case, the connection with Arduino should be made as follows:

MPU-6050	**ARDUINO**
SCL	A5
SDA	A4
GND	GND
VCC	3.3V



At this point it is important to mention the protoboarding.

A protoboard is a test plate, containing numerous holes in which cables and other electronic elements can be inserted for the realization of provisional circuits. The advantage of this device is that it does not require welding its components to have an operational circuit.

A protoboard is built by the following essential parts.

• Central channel, located in the middle of the board and where integrated circuits are connected to achieve the isolation of the pins on both sides.

• Buses: located on the sides of the protoboard, are identified by black or blue stripes indicating the ground bus; but also, by red stripes denoting the positive voltage bus.

Protoboards specific to Arduino usually come already assembled, so no welding is necessary.

The auxiliary circuits are mounted on the plate and tested.

In this case, as several tests have been carried out, it has been considered necessary to acquire one. However, account should be taken of the various disadvantages arising from their use, particularly in more complex circuits:

• They are only useful at low frequencies, not greater than 20MHz.

• Cannot be used with currents greater than 5A.

• It is dangerous when working with high voltages.

• False contacts may be established.

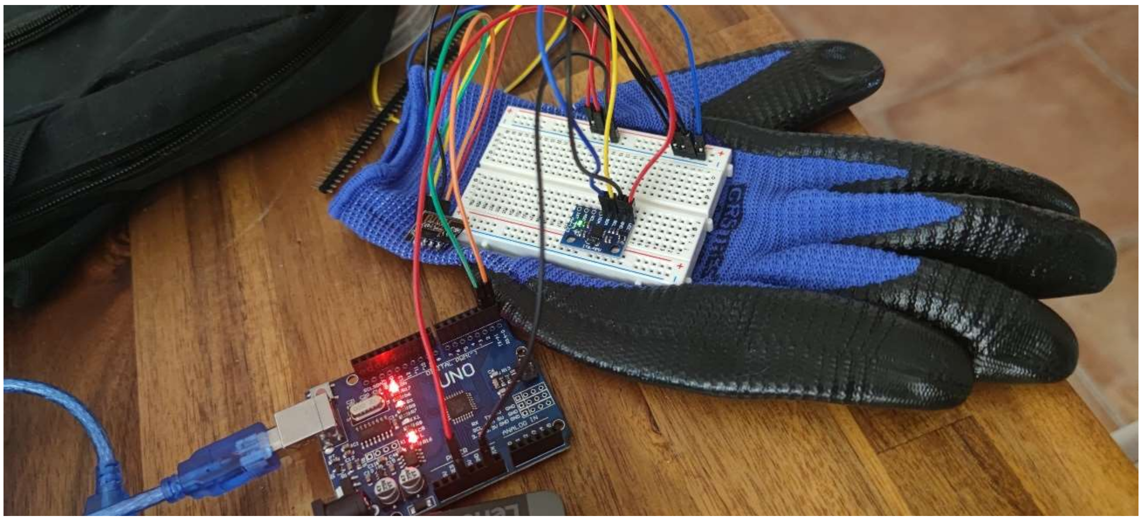
• Sometimes wiring can be complicated, especially when there are many interconnected components.

4.2 Software

When developing the software that allows the communication of the gloves with the video game, you must have, on the one hand, the code that will be uploaded to Arduino and, on the other, the code that we will use in the Unity scripts to obtain the information.

Firstly, to indicate that, for Arduino code, it is advisable to use Arduino IDE, since it provides useful and varied functionalities when working with these components.

In order to upload the code to Arduino, a B011 cable has been used to connect the board to the computer.



1. Imagen tomada durante la prueba de conexiones.

In this way, in addition, it is provided with food, since the project does not have external power, since it has tried to obtain the best performance with the least possible use of resources.

Once you open Arduino IDE, you must choose the type of board you are using from the tool tab, and select the serial port to which it is connected.

For communication with the Bluetooth module, we must make sure to include the SoftwareSerial library, which allows serial communication on other digital pins of the Arduino board, using the software to replicate the functionality.

For the connection with Arduino, you must define the Bluetooth port, making use of the aforementioned library, and pass as arguments the transmission and reception pins, being in this case:

SoftwareSerial BlueSerial (2,3);

Pin 2 and 3 respectively indicate the RX pin and the TX pin on the microcontroller.

To start the communication we must open the serial port and indicate the transmission speed (baud rate) to which we want to work. It is of the utmost importance that all devices that interfere with communication are at the same speed.

To do this, we would use the following function:

BlueSerial.begin(9600); //We set the transmission speed to 9600

To send data via serial port, use the Serial.print() function, for example:

BlueSerial.print("Communicating via Bluetooth...");

It should be noted that C++ is being programmed, so the code will be subject to the functions and characteristics of that programming language.

For receiving the data obtained with the MPU-6050 module, we must take into account that, with the accelerometer that incorporates, we can perform indirect measurements.

For example, if the acceleration is integrated in time the speed is obtained, and if it is integrated again, the displacement will be obtained, requiring in both cases the speed and the initial position respectively.

Xf = 1/4 (Af + Ao) t ^ 2 + Vo t + Xo

Where Xf is the final distance in meters, Af is the final CURRENT acceleration in m/s 2, Ao is the previous acceleration of the last data set in m/s 2, t is the CHANGE in time BETWEEN the data sets Af and Ao in SECONDS, Vo is the instantaneous speed of the last data set in m/s, and Xo is the final distance of the last data set or the sum of all previous distances in meters.

Vo should be calculated using the previous acceleration and acceleration of two previous data sets or Ao-1 using the following kinematic equation:

*Vo = 1/2 (Ao + Ao-1) * t + Vo-1*

Where Vo is the previous instantaneous speed in m/s, Ao is the previous acceleration in m/s 2, Ao-1 is the acceleration of two datasets in m/s 2, t is the change in time between Ao and Ao-datasets1 in SECONDS, and Vo-1 is the instantaneous speed in m/s of the Ao-1 dataset or two datasets.

On the other hand, with the gyroscope, you can measure the angular velocity, and, if you integrate the angular velocity with respect to time, get the angular displacement (angular position if you know where the rotation started).

$$\omega = \frac{\theta}{t}$$

The following libraries must be included to work on this project, in addition to the aforementioned:

```
#include "I2Cdev.h"

#include "MPU6050.h"

#include "Wire.h"
```

The ADDR pin internally in the module has a resistance to GND, so if it is not connected, the default address will be 0x68. This will be important when creating the variable or object for the MPU6050, because if you do not connect this pint, it could be created as follows:

```
MPU6050 sensor;
```

But if you want to work in another direction, it should be modified as follows:

```
MPU6050 sensor(0x69); //The MPU6050 only works on 0x68 and 0x69
```

In the void loop() function, both the I2C communication and the sensor and the serial bluetooth communication as described above must be started:

```
BlueSerial.begin(9600); // Initiating serial bluetooth communication

Wire.begin();   //Starting I2C

sensor.initialize();   //Starting the sensor
```

When initializing the sensor, the default ranges will be:

- accelerometer -2g to +2g

- gyroscope: -250°/sec to +250°/sec

Given that the resolution of the readings is 16 bits, the reading range is -32768 to 32767.

Following the loop function, the readings should be performed as follows:

sensor.getAcceleration(&ax, &ay, &az);

sensor.getRotation(&gx, &gy, &gz);

This will store the data in the variables passed as a parameter.

To scale the readings and convert the read value to an acceleration or angular velocity value, the following equations should be used:

*float ax_m_s2 = ax * (9.81/16384.0); //Repeat with az and ay*

*float gx_deg_s = gx * (250.0/32768.0); //Repeat with gz and gy*

To calculate the angle of inclination, if one considers that the only force acting on the sensor is the force of gravity, then the values obtained in the accelerometer components correspond to gravity and the resulting angles will be the inclination of the sensor plane, since gravity is always vertical.

To calculate the angles of inclination in a 3D space, in both X and Y, the following formulas should be used:

$$\theta_x = \tan^{-1}\left(\frac{a_x}{\sqrt[2]{a_Y{}^2 + a_z{}^2}}\right)$$

$$\theta_y = \tan^{-1}\left(\frac{a_y}{\sqrt[2]{a_x{}^2 + a_z{}^2}}\right)$$

So, to calculate the angles of inclination, we would use the following:

```
float accel_ang_x=atan(ax/sqrt(pow(ay,2) + pow(az,2)))*(180.0/3.14);

float accel_ang_y=atan(ay/sqrt(pow(ax,2) + pow(az,2)))*(180.0/3.14);
```

To calculate the current angle you must integrate the speed and know the current angle, which is done with the following formula:

$$\theta_x = \theta_{x_0} + \omega_x \Delta t$$

$$\theta_y = \theta_{y_0} + \omega_y \Delta t$$

So, in the code of Arduino would be as follows:

```
dt = millis()-tiempo_prev;

tiempo_prev=millis();



girosc_ang_x = (gx/131)*dt/1000.0 + girosc_ang_x_prev;

girosc_ang_y = (gy/131)*dt/1000.0 + girosc_ang_y_prev;



girosc_ang_x_prev=girosc_ang_x;

girosc_ang_y_prev=girosc_ang_y;
```

PHASE IV: Evaluation and final conclusions

1. Complications encountered during project development

Throughout the development of the project various complications have appeared.

In the first place, the lack of budget has not allowed the acquisition of optimal resources, so we proceeded to look for economic alternatives that would allow the proper functioning of the project.

On the other hand, many of the knowledge used has been acquired during the process, thus limiting the time spent on development.

One of the first problems that arose was the initial approach. The ESP01 microcontroller was used to allow communication between the gloves and the video game.

The ESP8266 ESP01 microcontroller is a low cost chip, which has the ability to connect via WIFI and can be used as a motherboard or as a module. The main problem when using these devices is given by its difficult configuration. It can be programmed via a serial/USB adapter or with the appropriate wiring, via Arduino. The connectors that come by default, do not allow to connect it to the protoboard, and taking into account, moreover, the objective to be achieved with the project, it would be difficult to maintain a stable connection with a phone, not to mention that it would be impractical, since various modifications should be made each time you want to switch devices.

After testing it and concluding that it was not suitable for the objective that the project aims to reach, it was decided to use a Bluetooth module. However, this slowed down the project development process.

At a technical level, most problems have occurred during the development of gloves. One of the main drawbacks was the use of the MPU-6050 module.

Although it meets the requirements of the project, and other similar sensors present the same problem, it is necessary to note that, since it is highly unlikely that the sensor is in a 100% horizontal position, leading to possible reading errors, it is necessary to calibrate it, configuring OffSets and thus compensate for any errors that may be received.

To solve this problem, changes were made in the code to introduce the calibration, so that they are constantly modifying the offset, trying to eliminate the error with the actual measure we want, in this case ax=0.ay=0.az=1g and gx=0.gy=0.gz=0. During calibration, it is necessary to position the sensor horizontally and avoid moving it, as this position will be the level for future positions.

During the calibration the program performs the readings of both the accelerometer and the gyroscope, using a filter we stabilize the readings a little and every 100 readings we check if the values are close to the values we want to read, depending on this increases or decreases offsets. This will cause the filtered readings to converge to:

•Acceleration: p_ax=0 , p_ay=0 , p_az=+16384

•Angular velocity: p_gx=0 , p_gy=0 , p_gz=0

Values should be expected to approach: 0 0 +16384 0 0 0.

This step between calibration and obtaining the actual data is done from Unity.

Another drawback related to the use of sensors such as the MPU-6050 is that the measurement is not accurate.

Even when you do not move the angle varies, or if you rotate a certain angle and then you return to the original position the angle you measure is not the initial one, this is because, by integrating the angular velocity and adding the initial angle, there is an error due to the poor measurement of time or noise in the reading of the MPU, the error, however small, accumulates in each iteration and grows, this error is known as DRIFT.

To decrease drift there are several methods, most apply filters to eliminate noise from sensor readings. Other sensors such as magnetometers or accelerometers can also be used and with angles calculated with these improve the gyroscope calculation.

For this purpose, the complement filter has been chosen. This combines the angle calculated by the gyroscope and the angle calculated by the accelerometer.

In this way the angle of the accelerometer is passing through a low pass filter, cushioning the sudden variations of acceleration, and the angle calculated by the gyroscope has a high pass filter, having great influence when there are quick rotations.

```
//Calcular ángulo de rotación con giroscopio y filtro complemento

  ang_x = 0.98*(ang_x_prev+(gx/131)*dt) + 0.02*accel_ang_x;

  ang_y = 0.98*(ang_y_prev+(gx/131)*dt) + 0.02*accel_ang_y;
```

2. Evaluation and conclusions

As can be seen, the objective of the project has been achieved.

The REST-based web server has been created, which communicates with the web page created for the registration and view of the player ranking, as well as with the Unity virtual reality video game that has been developed as planned.

In turn, the gloves that have been created work correctly as predicted.

While it is true that, given the resource constraints, no optimal result has been achieved, as, for example, the tests have been carried out from the computer, using the B011 cable to connect the Arduino board to the USB port and provide it with power.

However, despite possible improvements, the development of this project has demonstrated the large number of tools, programs and services that can be created even with minimal resources.

Virtual reality is not only limited to video games, there are many areas in which their application could contribute to improvements in them. This would be the case, for example, of the development of environments in which fire or accident simulations can be carried out which, due to their characteristics, might not be suitable for regular use in real environments.

It could also be used to simulate operations in which people engaged in surgery can realistically practice prior to a critical operation that does not allow errors.

And as far as Arduino is concerned, although over the last few years it has only demonstrated its versatility and its many applications, it is important to continue highlighting it, since one of the reasons why it has been able to develop up to that point, and for which it is not expected to stop doing so, it is precisely because of its open source.

In general, both during the research and development phase it has been observed that those programs or projects that have open source have a wider and stronger community of contributors that allows both facilitating learning for people without prior knowledge of how to make further progress more quickly, of which not only developers but society as a whole end up benefiting, since many of these advances can be in health, disaster prevention... in conclusion, it has been possible to observe how important open source is even beyond the scope of programming.

References

AranaCorp. (s.f.). *Comunicación con arduino y módulo hc-06*. Obtenido de https://www.aranacorp.com/es/comunicacion-con-arduino-y-el-modulo-hc-06/

Arduino. (s.f.). *Software-serial documentation*. Obtenido de https://docs.arduino.cc/learn/built-in-libraries/software-serial

Ardunity. (s.f.). *Getting started with MPU series*. Obtenido de https://sites.google.com/site/ardunitydoc/getting-started/run-examples/mpuseries

Areaciencias. (s.f.). *Desplazamiento angular*. Obtenido de https://www.areaciencias.com/fisica/desplazamiento-angular/

DigitalOcean. (s.f.). *How to install and use docker on ubuntu 20-04*. Obtenido de https://www.digitalocean.com/community/tutorials/how-to-install-and-use-docker-on-ubuntu-20-04

Django Software Foundation. (s.f.). *Django documentation*. Obtenido de https://docs.djangoproject.com/en/4.0/

Docker. (s.f.). *Docker documentation*. Obtenido de

https://docs.docker.com/

ElectroCrea. (s.f.). *Módulo Bluetooth HC-06*. Obtenido de https://electrocrea.com/blogs/tutoriales/33307075-modulo-bluetooth-hc-06#:~:text=Algunas%20de%20las%20ventajas%20del,como%20en%20modo%20de%20espera.

ElectroWings. (s.f.). *MPU-6050 Interfacing with Arduino Uno*. Obtenido de https://www.electronicwings.com/arduino/mpu6050-interfacing-with-arduino-uno

M Olmo, R. N. (s.f.). *Descripción del movimiento en una dimensión*. Obtenido de http://hyperphysics.phy-astr.gsu.edu/hbasees/mot.html

Microsoft. (s.f.). *Create Windows Virtual Machine in Azure*. Obtenido de https://docs.microsoft.com/es-es/learn/modules/create-windows-virtual-machine-in-azure/

Mobatek. (s.f.). *MobaXterm documentation*. Obtenido de https://mobaxterm.mobatek.net/documentation.html

Rice University. (s.f.). *Aceleración media e instantánea*. Obtenido de https://openstax.org/books/f%C3%ADsica-universitaria-volumen-1/pages/3-3-aceleracion-media-e-instantanea

Rice University. (s.f.). *Calcular la velocidad y el desplazamiento a partir de la aceleración*. Obtenido de

https://openstax.org/books/f%C3%ADsica-universitaria-volumen-1/pages/3-6-calcular-la-velocidad-y-el-desplazamiento-a-partir-de-la-aceleracion

Naylamp Mechatronics (s.f.). Tutorial mpu6050, acelerómetro y giroscopio.

Obtenido de

https://naylampmechatronics.com/blog/45_tutorial-mpu6050-acelerometro-y-giroscopio.html