

Trabajo final de SD

Grupo SD

June 7, 2016

Índice

1	Miembros del grupo.	3
2	Elección de proyecto.	4
2.1	¿Por qué esta idea y no otra?	4
2.2	Desarrollo de la idea.	4
3	Planificación y Reparto de tareas.	4
3.1	Reparto de trabajo.	4
3.2	Planificación del trabajo.	4
4	Tecnologías empleadas para el proyecto.	5
5	Código proporcionado.	6
6	Información técnica sobre el host.	15
7	Manual de Usuario	16
7.1	Requisitos previos para poder ejecutar el servicio.	16
7.2	Servicio de forma local.	16
7.3	Acceder de forma remota.	19

1 Miembros del grupo.

En esta sección encontramos los nombres y apellidos de los miembros que han trabajado en este proyecto

1. Iván Barbosa Gutiérrez.
2. Ángel Luis Bayón Romero.
3. Cristina Mateo Gil.
4. Antonio Sánchez Roldán.

2 Elección de proyecto.

2.1 ¿Por qué esta idea y no otra?

Hoy en día hemos llegado a descubrir nuevas generaciones como los *Hipsters*, *Youtubers* y, los que nos afectan en este caso, *Gamers*.

Estos jugadores emplean tanto tiempo a los videojuegos que de hecho, desde hace poco, se han instanciado como deportes, son los llamados ESports.

Teniendo en cuenta esta tendencia de los nuevos jugadores, y su curiosidad por descubrir nuevos mundos y opciones, nos ha hecho pensar que podríamos hacerles las búsquedas más fáciles.

Nuestra idea consiste en realizar un buscador selectivo, para hacer un filtrado previo relacionado con videojuegos, nuestra web recogerá información de una de las web de videojuegos más conocidas del mundo(3djuegos).

2.2 Desarrollo de la idea.

Vamos a encontrar distintas opciones de selección en nuestro buscador, dichas opciones van a ser del tipo plataforma, género y orden, estos filtros buscarán bajo los criterios solicitados, y nos mostrarán el nombre, la descripción del mismo, una puntuación de valoración estimada por las distintas reviews, y una fecha de publicación del mismo.

Por otro lado, también se permitirá realizar la búsqueda de las noticias relacionadas que van a ser mostradas en nuestro menú *Home*, mostrando las últimas más recientes.

De esta forma se ha proporcionado un entorno más visual y sencillo de utilizar para el usuario.

3 Planificación y Reparto de tareas.

3.1 Reparto de trabajo.

1. Página Web - Iván Barbosa Gutiérrez, Cristina Mateo Gil y Antonio Sánchez Roldán.
2. Documentación - Ángel Luis Bayón Romero.
3. Revisiones y correcciones - Todos.

3.2 Planificación del trabajo.

Se adjunta captura de Diagrama de Gantt REAL con los tiempos que hemos empleado para realizar dichos trabajos.

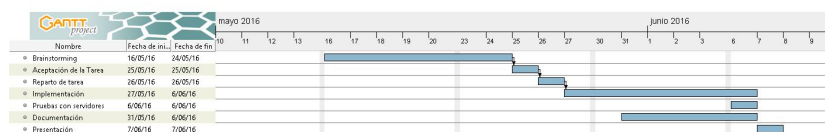


Figure 1: Diagrama de Gantt.

A continuación se desglosa en que consiste cada actividad:

1. Brainstorming: Durante esta etapa, todos los miembros del grupo propusieron distintas ideas sobre qué realizar como proyecto, dicha duración se debe a que la idea no fue elegida hasta el último momento.
2. Aceptación de la tarea: Dicha etapa hace referencia al visto bueno del profesor, para así poder empezar a realizar el proyecto cuanto antes.
3. Reparto de Tarea: Esta etapa consistió en un reparto previo de trabajo a los miembros citados anteriormente, aun que al final todos hemos hecho un poco de todo.
4. Implementación: En esta etapa se ha desarrollado todo el código fuente pertinente, tanto para hacer los servicios de captura de datos, como el propio entorno de la página web que lo sustentaba.
5. Pruebas con servidores: Esta tarea hace referencia a la subida de los ficheros a los distintos repositorios de internet, desde Github hasta el host.
6. Documentación: Relacionada con el mismo documento que se proporciona junto al proyecto, haciendo también el diagrama y descomponiendo en partes para añadir claridad al documento.
7. Presentación: En esta etapa, solo se refleja el tiempo que se va a tardar en realizar la presentación del prototipo final del proyecto.

Indicar que de forma paralela tanto a la implementación como a las pruebas con servidores (que vienen siendo las subidas a GitHub y un host concreto), se han ido realizando las distintas revisiones del código entre todos los miembros del grupo.

4 Tecnologías empleadas para el proyecto.

En esta sección vamos a indicar las tecnologías empleadas para el desarrollo del proyecto.

1. Python: Nos ha permitido hacer la extracción de información de los servicios, en concreto hemos usado *Beautifulsoup*, y creando servicios *flask*.
2. HTML: Que nos da un entorno web proporcionado por una plantilla con Bootstrap.
3. Javascript: Para las invocaciones del código y la gráfica.
4. CSS: Para darle un entorno más llamativo y visual.

5 Código proporcionado.

En esta sección añadiremos el código con breves explicaciones para ayudar a su comprensión.

```
# -*- coding: utf-8 -*-

from bs4 import BeautifulSoup
import requests
import os

maxPages = 20
codigo = ""

def plataforma(eleccion):
    global codigo
    plat = ''
    if (eleccion == '2'):
        plat = '-pc'
        codigo = "f1"
    elif (eleccion == '3'):
        plat = '-ps4'
        codigo = "f37"
    elif (eleccion == '4'):
        plat = '-ps3'
        codigo = "f2"
    elif (eleccion == '5'):
        plat = '-ps2'
        codigo = "f7"
    elif (eleccion == '6'):
        plat = '-xbox-one'
        codigo = "f38"
    elif (eleccion == '7'):
        plat = '-x360'
        codigo = "f4"
    elif (eleccion == '8'):
        plat = '-wiiu'
        codigo = "f35"
    elif (eleccion == '9'):
        plat = '-wii'
        codigo = "f3"
    elif (eleccion == '10'):
        plat = '-3ds'
        codigo = "f34"
    elif (eleccion == '11'):
        plat = '-ds'
        codigo = "f5"
    elif (eleccion == '12'):
        plat = '-psvita'
        codigo = "f36"
    elif (eleccion == '13'):
        plat = '-psp'
        codigo = "f6"
    elif (eleccion == '14'):
        plat = '-ios'
        codigo = "f9"
    elif (eleccion == '15'):
        plat = '-web'
        codigo = "f8"
    elif (eleccion == '16'):
        plat = '-android'
```

```

        codigo = "f32"
    else:
        codigo = "f0"

    return plat

def genero(eleccion):

    global codigo
    gen = ""

    if (eleccion == '2'):
        codigo += "f1"
        ger = "-accion"
    elif (eleccion == '3'):
        codigo += "f3"
        gen = "-aventura"
    elif (eleccion == '4'):
        codigo += "f9"
        gen = "-casual"
    elif (eleccion == '5'):
        codigo += "f10"
        gen = "-conduccion"
    elif (eleccion == '6'):
        codigo += "f6"
        gen = "-deportes"
    elif (eleccion == '7'):
        codigo += "f2"
        gen = "-estrategia"
    elif (eleccion == '8'):
        codigo += "f8"
        gen = "-mmo"
    elif (eleccion == '9'):
        codigo += "f7"
        gen = "-rol"
    elif (eleccion == '10'):
        codigo += "f5"
        gen = "-simulacion"
    elif (eleccion == '11'):
        codigo += "f4"
        gen = "-otros"
    else:
        codigo += "f0"

    return gen

def orden(eleccion):
    orde = ""

    if (eleccion == '1'):
        orde = "juegos-populares"
    elif (eleccion == '2'):
        orde = "fecha"
    elif (eleccion == '3'):
        orde = "lanzamientos"
    elif (eleccion == '4'):
        orde = "esperados"
    else:
        orde = "valoracion"

    return orde

```

```

def combinaciones(pla, gen, orde):
    pla = plataforma(pla)
    gen = genero(gen)
    orde = orden(orde)
    iniUrl = "http://www.3djuegos.com/novedades/juegos-generos/"
        juegos" + pla + gen + "/"
    finUrl = codigo + "f0/" + orde

    extraerDatos(iniUrl, finUrl)

def extraerDatos(iniUrl, finUrl):
    for i in range(1,maxPages):
        url = iniUrl + str(i-1) + finUrl
        req = requests.get(url)
        statusCode = req.status_code

        if statusCode == 200:
            html = BeautifulSoup(req.text, 'html.parser')
            entradas = html.find_all('table',{'class':'tb100
                fftit'})

            for i,entrada in enumerate(entradas):
                titulo = entrada.find('a', {'class' : 's18'}).
                    getText()
                descripcion = entrada.find('p', {'class' : 's13
                    c4 mar_t6 mar_t4 lhi7'}).getText()
                fecha = entrada.find('b')
                puntuacion = entrada.find('div', {'class' : '
                    val_cuadrado_gris fr mar_l16 mar_r2'})
                plat = entrada.find('span', {'class' : 'plats
                    bg_c4c mar_r4 dib mar_t8'})
                imagen = entrada.find('img')

                if (plat != None):
                    plat = plat.getText()
                else:
                    plat = ""

                if (puntuacion != None):
                    puntuacion = puntuacion.getText()

                if (fecha != None):
                    fecha = fecha.getText()

                print titulo.encode("utf-8") + " " + descripcion
                    .encode("utf-8") + " " + fecha.encode("utf-8"
                        ") + " " + puntuacion.encode("utf-8") + " "
                        + plat.encode("utf-8")

            else:
                break

def menu():
    os.system('cls')

    print "1. Todos"
    print "2. PC"
    print "3. PS4"
    print "4. PS3"
    print "5. PS2"
    print "6. XOne"

```



```

print "7. X360"
print "8. WiiU"
print "9. Wii"
print "10. 3DS"
print "11. DS"
print "12. Vita"
print "13. PSP"
print "14. iOS"
print "15. Web"
print "16. Android"
plataforma = raw_input("Plataforma: ")
os.system('cls')

print "1. Todos"
print "2. Acci\ 'on"
print "3. Aventura"
print "4. Casual"
print "5. Conducci\ 'on"
print "6. Deportes"
print "7. Estrategia"
print "8. MM0"
print "9. Rol"
print "10. Simulaci\ 'on"
print "11. Otros"
genero = raw_input("Genero: ")
os.system('cls')

print "1. Populares"
print "2. Nuevas altas"
print "3. A la venta"
print "4. M\ 'as esperados"
print "5. Valoraci\ 'on"
orden = raw_input("Orden: ")
os.system('cls')

combinaciones(plataforma, genero, orden)

menu()

```

Con este código ,ha sido con el que hemos ido realizando las pruebas de las capturas de datos y visto si eran correctos o no.

Basándonos en este sistema de captura, hemos realizado una web para poder realizar las búsquedas de forma más dinámica y vistosa, por lo que queda con tres secciones simples.

1. Fragmento que captura las búsquedas de los juegos.

```

# -*- coding: utf-8 -*-

from bs4 import BeautifulSoup
import requests
import os
import json

maxPages = 20
codigo = ""

def plataforma(eleccion):
    global codigo
    plat = ''

```

```

if (eleccion == 'pc'):
    plat = '-pc'
    codigo = "f1"
elif (eleccion == 'ps4'):
    plat = '-ps4'
    codigo = "f37"
elif (eleccion == 'ps3'):
    plat = '-ps3'
    codigo = "f2"
elif (eleccion == 'ps2'):
    plat = '-ps2'
    codigo = "f7"
elif (eleccion == 'xbox-one'):
    plat = '-xbox-one'
    codigo = "f38"
elif (eleccion == 'x360'):
    plat = '-x360'
    codigo = "f4"
elif (eleccion == 'wiiu'):
    plat = '-wiiu'
    codigo = "f35"
elif (eleccion == 'wii'):
    plat = '-wii'
    codigo = "f3"
elif (eleccion == '3ds'):
    plat = '-3ds'
    codigo = "f34"
elif (eleccion == 'ds'):
    plat = '-ds'
    codigo = "f5"
elif (eleccion == 'psvita'):
    plat = '-psvita'
    codigo = "f36"
elif (eleccion == 'psp'):
    plat = '-psp'
    codigo = "f6"
elif (eleccion == 'ios'):
    plat = '-ios'
    codigo = "f9"
elif (eleccion == 'web'):
    plat = '-web'
    codigo = "f8"
elif (eleccion == 'android'):
    plat = '-android'
    codigo = "f32"
else:
    codigo = "f0"

return plat

def genero(eleccion):

    global codigo
    gen = ""

    if (eleccion == 'accion'):
        codigo += "f1"
        ger = "-accion"
    elif (eleccion == 'aventura'):
        codigo += "f3"
        gen = "-aventura"
    elif (eleccion == 'casual'):

```

```

        codigo += "f9"
        gen = "-casual"
    elif (eleccion == 'conduccion'):
        codigo += "f10"
        gen = "-conduccion"
    elif (eleccion == 'deportes'):
        codigo += "f6"
        gen = "-deportes"
    elif (eleccion == 'estrategia'):
        codigo += "f2"
        gen = "-estrategia"
    elif (eleccion == 'mmo'):
        codigo += "f8"
        gen = "-mmo"
    elif (eleccion == 'rol'):
        codigo += "f7"
        gen = "-rol"
    elif (eleccion == 'simulacion'):
        codigo += "f5"
        gen = "-simulacion"
    elif (eleccion == 'otros'):
        codigo += "f4"
        gen = "-otros"
    else:
        codigo += "f0"

    return gen

def combinaciones(pla, gen, orden):
    pla = plataforma(pla)
    gen = genero(gen)
    iniUrl = "http://www.3djuegos.com/novedades/juegos-generos"
    /juegos" + pla + gen + "/"
    finUrl = codigo + "f0/" + orden

    mensaje = extraerDatos(iniUrl, finUrl)
    return mensaje

def extraerDatos(iniUrl, finUrl):
    mensaje = []

    url = iniUrl + "0" + finUrl
    req = requests.get(url)

    statusCode = req.status_code
    if statusCode == 200:

        html = BeautifulSoup(req.text)
        entradas = html.find_all('tr')

        for i, entrada in enumerate(entradas):
            if(entrada.find('a', {'class' : 's18'}) == None):
                continue
            titulo = entrada.find('a', {'class' : 's18'}).
                getText()

            descripcion = entrada.find('p', {'class' : 's13 c4
                mar_t6 mar_t4 lh17'}).getText()
            fecha = entrada.find('b')
            puntuacion = entrada.find('div', {'class' : '
                val_cuadrado_gris fr mar_l16 mar_r2'})
            plat = entrada.find('span', {'class' : 'plats

```

```

        bg_c4c mar_r4 dib mar_t8'})
imagen = entrada.find('img')
imagen = format(imagen['src'])

if (plat != None):
    plat = plat.getText()
else:
    plat = "PC"

if (puntuacion != None):
    puntuacion = puntuacion.getText()
else:
    puntuacion="-"
if (fecha != None):
    fecha = fecha.getText()
else:
    fecha = "Sin fecha"

mensaje.append({'titulo': titulo.encode("utf-8"),
'descripcion' : descripcion.encode("utf-8"), '
fecha': fecha.encode("utf-8"), 'puntuacion':
puntuacion.encode("utf-8"), 'plataforma':plat.
encode("utf-8"), 'imagen':imagen})

else:
    print "Status Code %d" %statusCode

return mensaje

```

Ya en este caso hemos eliminado el menú ya que estará completo como código html para la web.

2. Fragmento que captura y recoge las noticias.

```

# -*- coding: utf-8 -*-

from bs4 import BeautifulSoup
import requests
import os
import json

def getTitulo(url):
    req = requests.get(url)
    statusCode = req.status_code

    if statusCode == 200:
        html = BeautifulSoup(req.text, 'html.parser')
        titulo = html.find('h1',{'class':'tit_arti_1 ffitit3
c3b'})

        if(titulo != None):
            #Texto
            titulo = titulo.getText()

        else:
            #Video
            titulo = html.find('h1',{'class':'b fl '}).getText
            ()

    return titulo

```

```

def extraerDatos():
    mensaje = {}
    url = "http://www.3djuegos.com/"
    req = requests.get(url)
    statusCode = req.status_code

    if statusCode == 200:
        html = BeautifulSoup(req.text, 'html.parser')
        entradas = html.find_all('div',{ 'id': 'zona_chapas_gfx'
        })
        referencia = entradas[0].find_all('div')
        imagen = entradas[0].find_all('img')

        for i in [0,1,2]:
            ref = format(referencia[i]['data-url'])
            img = format(imagen[i]['src'])
            titulo = getTitulo(ref)
            mensaje[i]=({'titulo': titulo, 'imagen':img, '
                referencia':ref})

    return mensaje

def extraerContenido(url):
    mensaje = {}
    req = requests.get(url)
    statusCode = req.status_code

    if statusCode == 200:
        html = BeautifulSoup(req.text, 'html.parser')
        titulo = html.find('h1',{ 'class': 'tit_arti_1 fftit3
        c3b'})
        texto = html.find('p',{ 'class': 's16 b fftext c2 lh27'
        })
        video = html.find('meta',{ 'itemprop': 'contentURL'})

        if(titulo != None):
            #Texto
            titulo = titulo.getText()
            texto = texto.getText()

        else:
            #Video
            titulo = html.find('h1',{ 'class': 'b fl '}).getText
            ()
            texto = html.find('div',{ 'class': 'pr oh c3 lh16
            a_n a_c7 a_c0h mar_rl52 s13 fftit'}).getText()
            video=format(video['content'])

        mensaje=({'texto': texto.encode("utf-8"), 'video':video
        })

    return mensaje

```

Con este tramo recogemos las últimas noticias de la web y las mostramos con un carrusel, y como algunas noticias incluyen vídeos hemos añadido otro capturador para ellos.

3. Fragmento que gestiona los servicios proporcionados por el sistema.

```

from flask import Flask
from flask import request
import json
import patrones
import extrae_noticias

from datetime import timedelta
from flask import make_response, request, current_app
from functools import update_wrapper

def crossdomain(origin=None, methods=None, headers=None,
                max_age=21600, attach_to_all=True,
                automatic_options=True):
    if methods is not None:
        methods = ', '.join(sorted(x.upper() for x in methods)
                             )
    if headers is not None and not isinstance(headers,
        basestring):
        headers = ', '.join(x.upper() for x in headers)
    if not isinstance(origin, basestring):
        origin = ', '.join(origin)
    if isinstance(max_age, timedelta):
        max_age = max_age.total_seconds()

    def get_methods():
        if methods is not None:
            return methods

    options_resp = current_app.
        make_default_options_response()
    return options_resp.headers['allow']

    def decorator(f):
        def wrapped_function(*args, **kwargs):
            if automatic_options and request.method == '
                OPTIONS':
                resp = current_app.
                    make_default_options_response()
            else:
                resp = make_response(f(*args, **kwargs))
            if not attach_to_all and request.method != '
                OPTIONS':
                return resp

            h = resp.headers

            h['Access-Control-Allow-Origin'] = origin
            h['Access-Control-Allow-Methods'] = get_methods()
            h['Access-Control-Max-Age'] = str(max_age)
            if headers is not None:
                h['Access-Control-Allow-Headers'] = headers
            return resp

        f.provide_automatic_options = False
        return update_wrapper(wrapped_function, f)
    return decorator

app = Flask(__name__)

@app.route("/prueba")
def hello():

```

```

        data = {"data": "Soy la mega prueba, salvadora de
                    personitas fragiles y de tostadas que caen al suelo
                    !!!!!!"}

        return json.dumps(data)

@app.route("/devolverjuegos/<string:plataforma>/<string:genero>/<string:orden>", methods=['GET', 'OPTIONS'])
@crossdomain(origin='*')
def devolverJuegos(plataforma, genero, orden):
    mensaje={}
    mensaje = patrones.combinaciones(plataforma, genero, orden)
    return json.dumps(mensaje)

@app.route("/devolvernoticias", methods=['GET', 'OPTIONS'])
@crossdomain(origin='*')
def devolverNoticias():
    mensaje={}
    mensaje = extrae_noticias.extraerDatos()
    return json.dumps(mensaje)

@app.route("/devolvercontenido/<string:url>", methods=['GET', 'OPTIONS'])
@crossdomain(origin='*')
def devolverContenido(url):
    mensaje={}
    url=url.replace(';','/')
    mensaje = extrae_noticias.extraerContenido(url)
    return json.dumps(mensaje)

if __name__ == "__main__":
    app.run(debug=True, port=5000)

```

Si llegas a leer esto Mota, se que te va a gustar el servicio de prueba para ver si funcionaba.

Por otro lado hemos realizado también código HTML para poder acoplarlo con nuestros servicios y tener el sistema completo.

Dicho código se encuentra alojado en un repositorio indicado más tarde.

Es posible que las versiones de código encontradas en la documentación, no sean las últimas que se han utilizado, por ello es recomendable acceder al repositorio donde se encuentra incluso con las versiones anteriores.

6 Información técnica sobre el host.

El host que hemos utilizado en este caso es *Openshift*, ya que nos proporcionaba tanto una sección para ejecutar las partes de código python, como un host de forma gratuita.

El enlace a nuestro proyecto por tanto sería el siguiente:

<http://chachijuegos-esisd.rhcloud.com/>

De esta forma se nos va a permitir acceder teniendo solamente acceso a internet.

Los enlaces para acceder al proyecto de GitHub son los siguientes:

1. <https://github.com/IvanBarbosa/chachijuegos>
2. <https://github.com/AngelBayRo/WebSD>

7 Manual de Usuario

7.1 Requisitos previos para poder ejecutar el servicio.

En primer lugar debemos tener instalado el servicio de BeautifulSoup, que como todos los anteriores se instala con un:

```
|| pip install beautifulsoup4
```

7.2 Servicio de forma local.

Una vez instalado se debe en primer lugar ejecutar con python el fichero *servicios.py* ya que nos va a activar las funcionalidades de la web, y una vez activo, se ejecuta el fichero *index.html* que nos va a mostrar la página web.

Dentro de la web encontramos distintas secciones, una de ellas que será la que se muestra como principal en la web que nos va a mostrar las últimas noticias, y otra que va a realizar la captura de los datos de forma sencilla (con selectores).

A continuación se añaden capturas de imágenes con un ejemplo, para ver como se debe ejecutar el servicio.

Una vez accedemos a la web, encontramos la parte de *Home*, vemos las noticias como hemos dicho antes.

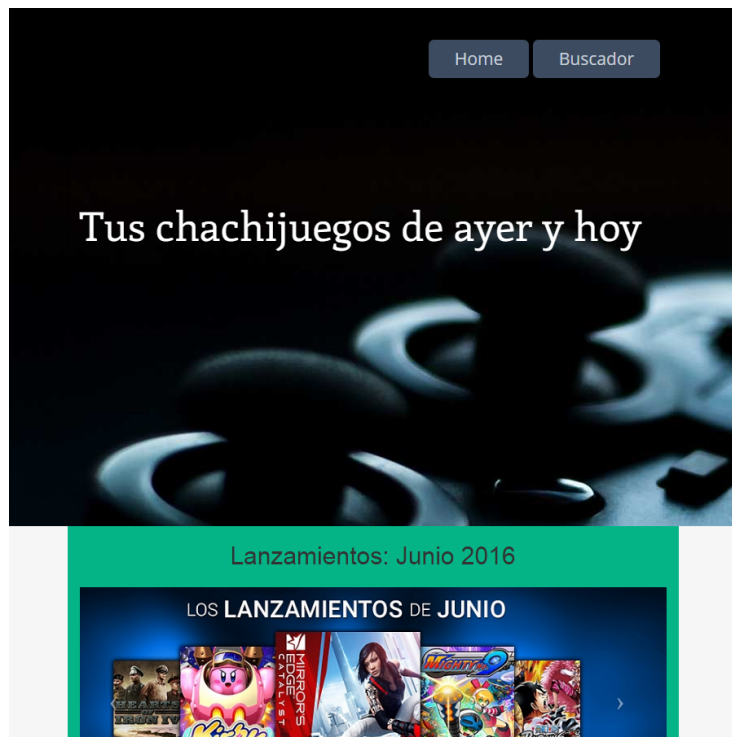


Figure 2: Home.

Ahora vamos con la parte del buscador, en ella se ha implementado un selector sencillo, a continuación vemos un ejemplo.

Donde simplemente, paso a paso, se selecciona y se presiona *Buscar*, una vez realizado veremos los resultados.

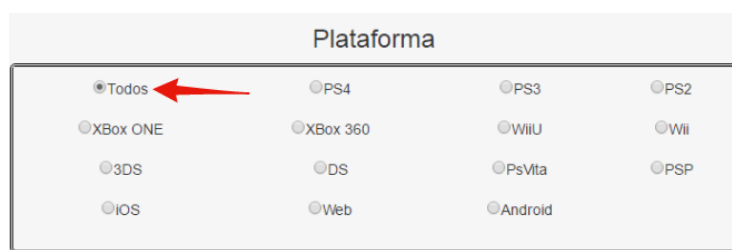


Figure 3: Primer selector.

De forma que al marcar todas las opciones quedaría:

Buscador de juegos

Plataforma

☒ Todos ☐ PS4 ☐ PS3 ☐ PS2
☐ Xbox ONE ☐ Xbox 360 ☐ WiiU ☐ Wii
☐ 3DS ☐ DS ☐ PsVita ☐ PSP
☐ IOS ☐ Web ☐ Android

Genero


☐ Todos ☐ Aventura ☐ Casual ☐ Conducción
☐ Deportes ☒ Estrategia ☐ MMO ☐ Rol
☐ Simulación ☐ Acción ☐ Otros

Orden

☒ Populares ☐ Fecha ☐ Lanzamientos ☐ Esperados
☐ Valoración

Figure 4: Buscador completo.

De cada videojuego vamos a encontrar una foto, su artículo relacionado y la puntuación proporcionada por prensa, dichas puntuaciones se verán de forma comparativa en un gráfico de barras abajo de la búsqueda.



Dishonored 2

Segunda entrega del videojuego Dishonored creado por Arkane Studios, que apuesta por dos protagonistas y una ambientación en Karnaca que tiene lugar 15 años después del original. Su fórmula de juego combina acción en primera persona, asesinatos, sigilo, movilidad y el brutal sistema de combate que vimos en Dishonored, pero mejorado. El videojuego estrena nueva co-protagonista para acompañar a Corvo Attano, la emperatriz Emily Kaldwin, y marcar una experiencia jugable muy diferente en cuanto a poderes y posibilidades. A nivel de armas regresan la pistola, la ballesta y la mítica espada, pero también se suman otras nuevas.

Figure 5: Ejemplo de juego encontrado.

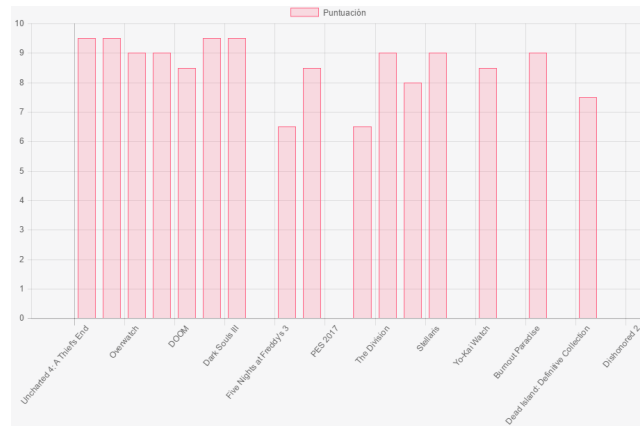


Figure 6: Gráfico de barras comparativo.

Una vez deje de interesarnos la búsqueda podemos simplemente, repetir el proceso con búsquedas nuevas tantas veces como se desee.

7.3 Acceder de forma remota.

Empleando el enlace proporcionado en el apartado anterior de la documentación simplemente debemos acceder a él y seguir el manual de uso del paso previo ya que el entorno gráfico seguirá siendo el mismo.