Treball de fi de grau
Trivial
Informe final

Guillem Reixach Almenara, *Estudiant, Enginyeria Informàtica, UAB 2021*

1 Introducció

Actualment la gent dedica cada cop més temps a l'entreteniment virtual, ja sigui en forma de videojocs o consumint i creant contingut multimèdia, entre molts altres. Aquest consum de serveis digitals interactius facilita a la gent noves formes de relacionar-se però, en molts casos, requereix de l'ús de comunicacions en temps real.
En aquest treball es pretén implementar un mecanisme de comunicació en temps real per a un entorn web. En particular, es vol crear una pàgina web que permeti fer partides de trivial amb altres persones i poder compartir així la diversió d'un clàssic joc de taula amb amics o amb gent desconeguda de la resta del món.

2 Objectius

Es planteja crear una plataforma web que permeti fer partides de trivial amb amics o desconeguts. Es vol implementar el joc de manera que permeti jugar a temps real i que es puguin crear partides privades per jugar només amb amics, compartint un codi de tauler, o unir-se a partides amb altres persones.
Es crearà una interfície visual senzilla per centrar-se en el desenvolupament funcional del joc. Per tant, en essència, amb aquest projecte es pretén implementar un aplicatiu web que permeti la comunicació en temps real entre dos o més clients per poder jugar a un joc per torns.
A més, es vol guardar tot el necessari en una base de dades per tal de poder garantir una correcta gestió dels usuaris i de les partides en curs.

3 Metodologia

La metodologia a seguir ha estat la d'implementar la solució per fases. S'ha emprat una gestió híbrida [1] del projecte, que ha combinat un disseny en cascada [2] amb un enfocament de desenvolupament àgil [3]. Això ha permès implementar els objectius en un ordre força seqüencial però buscant sempre la funcionalitat de la solució.
El desenvolupament del treball s'ha fet en tres fases, a més d'una primera part de disseny: (1) partida entre dos jugadors, (2) partides simultànies entre múltiples jugadors, (3) gestió de la informació amb base de dades.
En la primera fase s'ha implementat les funcionalitats necessàries per a permetre jugar una partida entre dos jugadors. L'aplicatiu permet a dos clients connectar-se i fer una partida de manera estàtica però amb comunicacions en temps real.
En la segona fase del desenvolupament s'ha fet les modificacions al codi i les abstraccions necessàries per equipar l'aplicatiu amb dinamisme i permetre a varis usuaris jugar partides diferents de manera simultània.
La tercera fase ha englobat una sèrie de tasques relacionades amb base de dades i gestió de la informació. En aquesta part del desenvolupament la base de dades ha estat la gran protagonista: s'ha implementat l'accés i un entramat de consultes per tractar els atributs dels usuaris i les partides per brindar funcionalitats extres en relació a l'experiència de joc i poder mantenir l'estat de les partides per reprendre-les després de desconnectar-se.

4 Disseny

Abans de la fase de desenvolupament s'ha realitzat una sèrie de tasques de disseny per a definir els objectius i funcionalitats del projecte.
Per a aquesta part de disseny de la web s'ha utilitzat un diagrama de flux per seguir quin hauria de ser el funcionament del software i poder definir de manera concreta els objectius tècnics del projecte. A més, també s'ha explotat la sencillesa i utilitat dels diagrames UML [4] per a definir els casos d'ús del programari. Finalment s'ha dissenyat una primera versió de la base de dades mitjançant un diagrama relacional ER [5].
Tot això en conjunt ha donat peu a poder omplir un Kanban amb tots els requisits i objectius a nivell tècnic i de manera força específica del que serien les tasques per completar el projecte.

DIAGRAMES UML CASOS D'ÚS

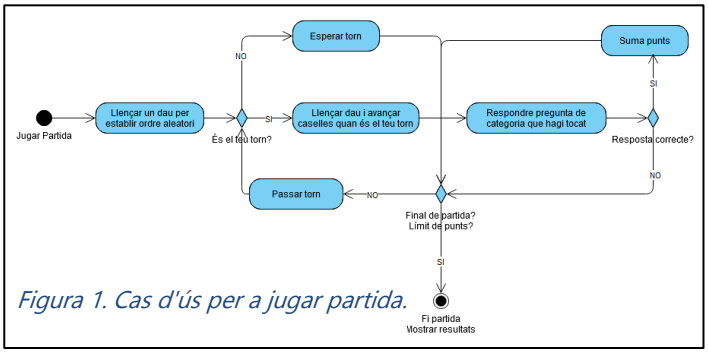

Figura 1. Cas d'ús per a jugar partida.

Aquesta primera imatge mostra el diagrama seguit per a implementar una partida: els jugadors llencen un dau i jugaran en ordre del número obtingut de manera descendent, és a dir, qui ha tret el nombre més alt comença; l'usuari a qui toqui jugar llençarà un dau de nou per determinar de quina categoria haurà de ser la seva pregunta; acte seguit, aquest veu la seva pregunta i la contesta (sobra dir que si l'encerta obtindrà punts); finalment es cicla el torn i juga el següent usuari, a no ser que sigui el final de la partida (exhaurit el nombre màxim de rondes).

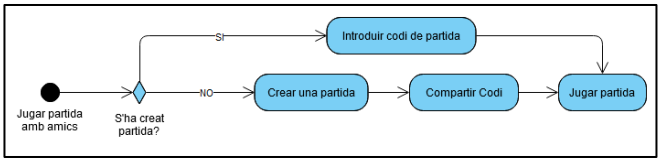

Figura 2. Cas d'ús per a jugar partida amb amics.

En la figua 2 es pot veure el diagrama UML utilitzat per a la implementació de partides privades. Un usuari haurà d'encarregar-se de crear una partida privada i compartir el codi de tauler amb els seus amics. La resta de jugadors poden introduir el codi per poder unir-se al joc.

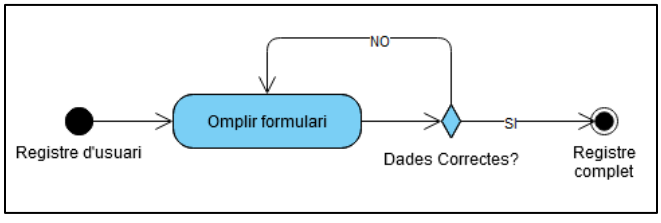

Figura 3. Cas d'ús per al registre d'un nou usuari.

Per al registre d'un nou usuari simplement cal omplir un formulari (nom d'usuari i contrasenya). Si aquestes dades tenen el format correcte, l'usuari queda registrat a la base de dades.

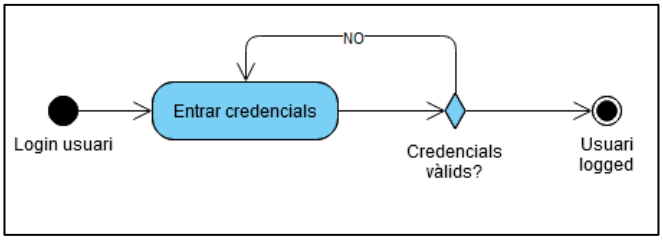

Figura 4. Cas d'ús d'inici de sessió d'usuari existent.

En el diagrama de la figura 4 es pot veure com per iniciar sessió, com sol ser habitual, només cal entrar uns credencials vàlids. Si nom d'usuari i contrasenya concorden, l'usuari accedeix al seu compte.

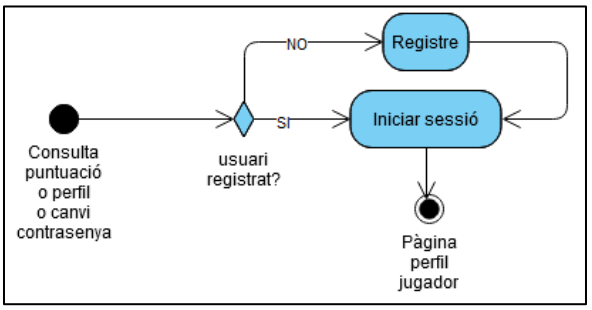

Figura 5. Cas d'ús per a consulta de pàgina de perfil d'usuari.

A la figura 5 es mostra el procés per a poder consultar la pàgina d'informació d'usuari. És requeriment necessari que l'usuari hagi iniciat sessió. Si el jugador no té perfil al portal web, es pot registrar.

DIAGRAMA ER

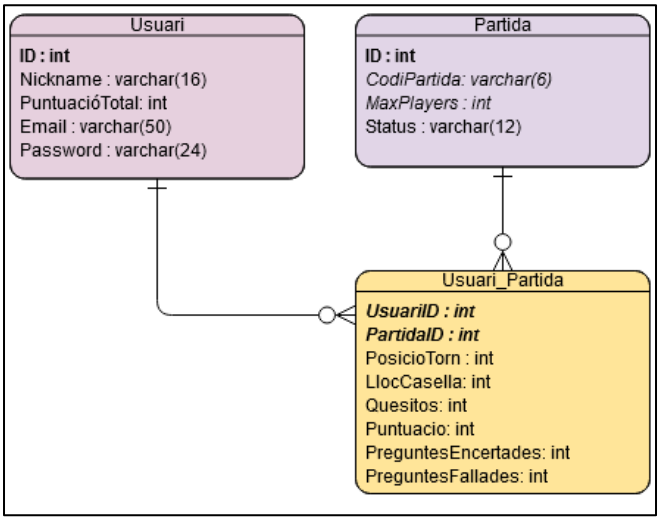

Figura 6. Diagrama Entitat Relació per al disseny de la base de dades.

La figura 6 mostra el diagrama ER (Entitat-Relació) utilitzat per al plantejament i disseny de la base de dades. Bàsicament existeixen 3 entitats en aquest model: usuari, partida i usuari_partida, que relacionarà les altres dues. L'usuari guardarà un identificador com a clau primària i tindrà el nom d'usuari, puntuació total del jugador, correu electrònic i un hash de la contrasenya (sota cap circumstància es guardarà una contrasenya en text pla). La partida també té un identificador com a clau primària i guarda el seu codi de tauler, el nombre màxim de jugadors i l'estat.
Finalment, l'entitat usuari_partida s'utilitza per a relacionar les dues anteriors. Les claus primàries són l'indentificador de partida i l'identificador d'usuari. A més es guarden dades d'un usuari en concret en una partida en concret, com ara la puntuació, les preguntes encertades i la posició en l'ordre de joc.

5 DESENVOLUPAMENT

Aquesta és, sens dubte, la secció més important del projecte. Com s'ha comentat en l'apartat de metodologia, per tal d'afrontar-lo correctament s'ha dividit en tres etapes per aconseguir una implementació ordenada i sempre enfocada a la funcionalitat de la solució.
A la resta d'aquest apartat del document es descriu: (1) la infraestructura utilitzada per la realització del projecte; (2) el desenvolupament d'una partida entre dos jugadors; (3) el desenvolupament per suportar partides simultànies entre múltiples usuaris; i (4) la implementació de gestió de la informació amb la base de dades.

5.1 Infraestructura

Primerament s'ha preparat un entorn sobre el que treballar i, per a tal efecte, s'ha creat una petita infrestructura amb l'ajuda d'una Raspberry Pi3.
Per tal de disposar d'un servidor web adient, en el qual es pugui executar codi fàcilment i es pugui gestionar temes de xarxa (poder garantir gestió de ports) s'ha optat per desplegar un Apache [6] en una Raspberry Pi3[7] amb Raspbian [8] (una distribució de Debian). Per aconseguir això, ja es disposava del hardware necessari i s'hi ha instal·lat el sistema operatiu. És un procés relativament senzill, donat que la Raspberry es comporta com un ordinador.
A continuació s'ha instal·lat Apache a la màquina i phpMyAdmin [9] per a la gestió de base de dades. També s'ha fixat una IP estàtica a la Raspberry i s'ha creat una regla NAT al router per garantir l'accés a la pàgina web des de qualsevol punt fora de la xarxa domèstica (aquest punt ha estat per comoditat a l'hora de treballar, donat que no cal ser en un lloc fixe).

5.2 Implementació partida entre dos jugadors

Per a la implementació de la primera fase s'ha centrat els esforços en la funcionalitat de jugar i s'ha aconseguit crear i representar una partida entre dos jugadors.
Amb l'arquitectura MVC i en disposició del servidor de sockets, s'ha implementat les funcionalitats necessàries per aconseguir que dos usuaris (dues finestres de navegador diferents en l'entorn de probes) es connectin via socket al servidor i puguin jugar en una mateixa partida: es llença un dau per veure qui treu el número més alt i obté el primer torn, a continuació i per torns, es llença un dau per triar aleatòriament una categoria de la llegenda i es carrega una pregunta d'acord amb el resultat obtingut mitjançant una petició GET [10] a una API oberta de preguntes tipus Trivial (https://opentdb.com/)[11], es valida la resposta de l'usuari al servidor i es comunica el resultat al jugador en qüestió.
La primera fase d'implementació s'ha començat creant una arquitectura MVC [12] per tal d'oferir al client els recursos necessaris per visualitzar la pàgina web correctament. S'ha creat un arxiu index per permetre una navegació mitjançant REST uris [13], molt més agradable a la vista del client i molt més còmode per a la feina de programació. S'ha creat una vista principal que serveix com a punt d'accés a l'aplicatiu i des d'on es pot navegar a les diferents opcions disponibles: iniciar sessió, registrar-se, jugar, crear una partida privada i unir-se a una partida privada.
A continuació, per aconseguir comunicacions en temps real, s'ha implementat en el servidor una classe PHP [14] que actua com a servidor websocket [15], [16]. Aquesta s'ha creat fent ús de la llibreria websockets-PHP [17] - una de les més accessibles, més utilitzades, més versàtil i configurable (sense fer ús de cap framework) – i s'encarrega de gestionar les comunicacions via socket; és a dir, disposa d'una llista d'usuaris connectats (sockets oberts) i rep i envia missatges en format JSON[18] per dotar als clients i al servidor de comunicacions persistents i immediates (temps real).
En aquest punt, el servidor websocket disposa de les següents funcionalitats:

- Jugar: Quan un usuari vol jugar una partida el servidor s'encarrega de buscar si existeix algun tauler on faltin jugadors i l'afegeix; si no hi ha cap slot lliure, es crea una nova partida.
- Determinar torn (dau): Al principi de la partida, els usuaris llencen un dau per determinar en quin ordre jugaran. Aquesta funció rep el número que ha tret cada jugador i s'encarrega de determinar l'ordre i comunicar-lo als clients.
- Començar partida: Quan ja s'ha determinat en quin ordre jugaran els usuaris, aquesta funció s'encarrega de començar la partida i brinda al primer jugador l'opció de llençar un dau de nou per carregar una pregunta. Els usuaris que encara no tenen el torn, veuen a qui li toca jugar i han d'esperar fins que aquest contesti.
- Carregar pregunta: Quan la partida ha començat i és el teu torn, llences un dau per determinar aleatòriament de quina categoria és la pregunta. El servidor s'encarrega de fer una petició GET a *Open Trivia Database* amb els paràmetres necessaris i retorna al client la pregunta juntament amb les possibles respostes (mantenint sempre en secret, a la part del servidor, quina és la resposta correcta, per evitar conductes malicioses de la part del client).
- Corregir pregunta: En contestar la pregunta en qüestió, aquesta funció rep la resposta i valida si és correcte o no. Retorna el resultat al jugador i, si s'ha encertat, li suma punts.
- Avançar torn: Després de contestar la pregunta i corregir-la, es crida aquesta funció per ciclar el torn; és a dir, seguint l'ordre especificat anteriorment, es notifica al següent jugador a qui toca actuar. Si han jugat tots els usuaris de la partida, torna al primer i augmenta el número de ronda.

Mitjançant Javascript [19] el navegador del client és capaç de cridar aquestes funcionalitats i gestionar les respostes a través de missatges JSON amb un format consensuat:

FORMAT JSON PETICIÓ

- method: especifica la funció que es requereix del servidor.
- idPlayer: identificador del jugador (HardCoded, de moment).
- idPartida: identificador de partida (HardCoded, de moment).
- data: camp opcional que pot incloure una llista de dades necessàries per a la funcionalitat al servidor.

FORMAT JSON RESPOSTA

- status: conté l'estat resultant d'executar la funció del servidor. Si tot va bé, aquest camp contindrà un "ok"; en cas contrari, contindrà "error". Molt útil per al seguiment de les funcionalitats i control d'errors.
- method: especifica la funció javascript que ha de rebre i tractar el resultat a la part del client.
- res: conté el resultat de la funció. Aquí hi pot anar qualsevol tipus de dades (sempre en format json).

5.3 Implementació partides simultànies

La segona fase d'implemetació s'ha centrat en aplicar al codi l'abstracció necessària per aconseguir que el servidor sigui capaç de gestionar diferents partides amb múltiples usuaris. En aquesta etapa no es buscava implementar noves funcionalitats, sinó adaptar les ja aconseguides en l'apartat anterior per operar de manera molt més dinàmica i permetre al servidor atendre múltiples peticions simultànies de manera independent.

Per aconseguir això s'han creat clases PHP representant objectes [20] de la solució. Aquestes classes simulen els elements usuari, partida, pregunta i contenen tots els atributs i funcionalitats necessàries en respecte a la seva finalitat.

A la part del servidor s'ha utilitzat dues variables de tipus diccionari (array associatiu PHP) [21] que contenen: una equivalència entre l'identificador de socket i el d'usuari de l'aplicatiu; i una llista de partides indexades pel seu identificador. A més, l'objecte partida conté una llista de preguntes i una llista d'usuaris. Aquests identificadors tant d'usuari com de partida han estat molt importants en aquesta etapa del desenvolupament i, a partir d'aquest moment, comencen a ser presents en totes les comunicacions client-servidor per poder seguir les accions en tot moment.

D'aquesta manera, el servidor websocket és capaç, en aquest punt, de crear diferents partides amb un màxim de jugadors, garantint així que es puguin connectar i jugar simultàniament varis usuaris en partides independents. Per tant, per exemple, si el màxim de jugadors per partida és de dos i es connecten quatre persones, el servidor crearà dues partides de dos jugadors i les gestionarà de manera independent per garantir el correcte transcurs del joc per a tots els usuaris.

Es podria afegir que en aquesta fase també s'ha fixat el nombre màxim de rondes per a la partida i s'ha fet la gestió necessària per mostrar els punts de cada jugador i determinar, quan acaba la partida, qui ha estat el guanyador. En aquesta tasca ha ajudat el fet de crear una entitat *TornManager* adjunta a l'objecte partida que ha facilitat el tractament de l'ordre dels jugadors, la gestió de les rondes i la detecció del final de partida.

5.4 Implementació gestió de la informació

Aquesta última fase d'implemetació ha posat l'atenció en la gestió de la informació. S'ha creat les funcionalitats d'inici de sessió, registre d'usuaris, creació de partides privades i llistat de partides pendents. Mitjançant l'ús de la base de dades i, tot plegat, enllaçant amb el dinamisme buscat en l'apartat anterior, s'ha aconseguit mantenir l'estat de totes les partides de cada jugador per deslligar-se de l'estat de la connexió dels sockets dels clients.

Amb aquests elements disponibles, s'ha pogut extendre les funcionalitats del servidor per validar i crear usuaris i per fer passar tota la informació i atributs d'objectes per la base de dades. D'aquesta manera s'ha aconseguit tenir un registre d'usuaris i poder conservar l'estat de les partides i els jugadors presents a cadascuna d'elles. Finalment s'ha aconseguit llistar les partides en curs d'un usuari validat i assenyalar les que estan pendents de la seva acció.

Primerament s'ha configurat l'accés a base de dades des del codi mitjançant una funcionalitat de connexió en el model. A partir d'aquí s'han escrit les consultes SQL [22] pertinents conforme han estat de menester, per tal d'escriure i obtenir totes les dades necessàries per al correcte desenvolupament de la partida.

S'ha creat noves vistes per a l'inici de sessió i registre d'usuaris i per carregar inputs de paràmetres quan es vol crear una partida privada (nombre de jugadors i dificultat) o unirse a una (codi de tauler). Totes aquestes vistes es carreguen fent us d'Ajax[23], donat que el refresc de pàgina provoca una desconnexió del servidor websocket. Aquest punt ha estat crític, també, pel que fa al seguiment d'identificadors de jugador i partida, perquè inicialment es plantejava guardar aquests en una variable de sessió que, a mig desenvolupament es va descobrir que era inaccessible per al servidor websocket.

Tots els camps d'entrada disponibles han estat dotats de validació tant a la part del client -mitjançant configuració dels tags HTML per a convertir els camps en obligatoris i forçant un format d'entrada concret- com a la part del servidor, comprobant que el format és l'esperat i aplicant una sèrie de funcions natives PHP per evitar errors i atacs de Cross Site Scripting (XSS) [24].

També ha calgut seguir modificant les funcionalitats ja existents per tal d'adaptar-se al nou sentit i direcció de la pàgina web. Per tant, per exemple, les funcions del servidor per jugar, començar una partida o determinar l'ordre dels jugadors han hagut d'incloure interaccions amb la base de dades per guardar tota la informació necessària i recuperar-la quan ha fet falta.

En aquest punt, el servidor websocket disposa de les següents funcionalitats afegides :

- Iniciar sessió: valida els credencials de l'usuari per iniciar sessió mitjançant una consulta SQL del model i retorna l'identificador de jugador si s'ha validat amb èxit. Els camps d'entrada són el nom d'usuari i contrasenya i tot i que passen una validació de la part del client, s'escapen i es tracten, també, a la part del servidor. Per comprovar que la contrasenya és correcta s'utilitza el mètode *password_verify* de PHP donat que a la base de dades es guarda el seu hash.
- Registre usuari: insereix els credencials i informació del jugador a la base de dades. Com en la funcionalitat de login, els camps d'entrada nom d'usuari i contrasenya es sanegen a la part de servidor i es guarda el hash generat de la contrasenya mitjançant la funció *password_hash* a la base de dades.
- Crear partida privada: rep paràmetres d'entrada (màxim de jugadors i dificultat) i crea una partida accessible només amb el codi de tauler aleatori generat que es retorna. Tots els atributs de partida es guarden a la base de dades i el camp *private* es posa en 1 (true).
- Unir-se a partida privada: permet a un jugador accedir a una partida privada mitjançant un codi de tauler. Aquest codi de tauler entrat es saneja a la part del servidor per evitar atacs XSS i es comprova si realment existeix a la base de dades una partida amb aquest codi i atribut *private* igual a 1.
- Llistar partides en curs: aquesta funció consulta a la base de dades les partides amb estat "actiu" que té un usuari i les retorna en format llista. Es retornen tant els jocs privats com els públics i es permet accedir als que requereixen de l'acció de l'usuari.
- Recuperar partida: recupera tots els atributs d'una partida en concret de la base de dades. Aquesta funció es crida quan el jugador vol interactuar amb una de les partides de la seva llista de partides pendents i la carrega igual que si la partida s'hagués creat en el moment, permetent llençar el dau per triar categoria i carregar la pregunta a contestar.

També s'ha aplicat a la web un mínim d'estils CSS [25] per brindar una experiència de joc més intuïtiva i agradable.

5.5 Canvis i contratemps

En aquest apartat es podria assenyalar com a contratemps una problemàtica obtinguda amb l'ús que s'esperava fer de la variable de sessió. Amb les comunicacions websocket, quan el client refresca la pàgina, es perd la connexió persistent i aquesta és reiniciada; de manera que l'identificador del socket canvia. Per evitar això, s'ha desenvolupat totes les funcionalitats de la part del client amb Ajax, cosa que està força bé perquè aporta molta fluïdesa a l'experiència d'usuari. Per altra banda, ha calgut replantejar el tema de l'identificador d'usuari, donat que es volia guardar en una variable de sessió que era inaccessible per al seridor websocket. El que s'ha fet ha estat obligar a iniciar sessió en obrir la web, per tal de mantenir l'id de jugador en les comunicacions següents.

Pel que fa als canvis, el projecte ha evolucionat força respecte al que es va plantejar en un començament. El joc ha deixat de ser un Trivial clàssic per convertir-se en un joc de preguntes per torns; es manté l'essència, amb preguntes de diferents categories i respostes tipus "test", però per simplificar la part gràfica de la web, s'ha abolit el tauler que en principi es volia implementar i s'ha canviat per una llegenda de categories numerades del 1 al 6 per correspondre a cadascun dels possibles resultats del dau llençat. D'aquesta manera s'evita haver de gestionar la posició de cada jugador en la partida i haver de representar-la, a més, en el tauler del client.

Gràcies a això s'ha pogut centrar el desenvolupament en objectius més ambiciosos en la part del servidor sense haver d'absorbir retràs en la implementació.

En resum es podria dir que s'ha sacrificat part gràfica i de la lògica del joc a canvi d'una millor gestió de les dades i de la informació.

6 Resultats

En aquest apartat s'inclouen una sèrie d'imatges que permeten veure quin és l'estat final del projecte en el moment de l'entrega. L'aplicació desenvolupada permet jugar una "partida ràpida" pública (el servidor emparella persones fins a un màxim de jugadors preestablerts – que seran 2 per a la presa d'imatges – i comença una partida de nivell mig), crear una partida privada mitjançant l'entrada dels paràmetres màxim de jugadors i dificultat i unir-se a una partida privada mitjançant l'entrada del codi de tauler.

Abans de mostrar aquestes opcions, però, s'obliga a iniciar sessió o registrar-se com a usuari si un jugador encara no disposa de compte.

Com es pot veure a les figures, en iniciar sessió, un jugador pot visualitzar les partides que té en actiu i pot interactuar amb les que esperen el seu torn. També es pot veure la informació de jugador, que inclou el nom d'usuari i puntuació total, calculada com la suma de puntuacions de partides acabades.

Figura 7. Pàgina inicial, punt d'accés a la pàgina web.

La imatge anterior mostra el primer que es veu quan s'accedeix al portal web. De manera obligatòria l'usuari ha d'iniciar sessió o donar-se d'alta mitjançant el formulari de registre.

Figura 8. Pàgina de registre d'usuari.

A la figura 8 es pot veure el formulari de registre. És molt senzill; finalment s'ha decidit no demanar cap dada més, donat que no n'hi ha cap més que sigui estrictament necessària per identificar el jugador en el punt del projecte actual.

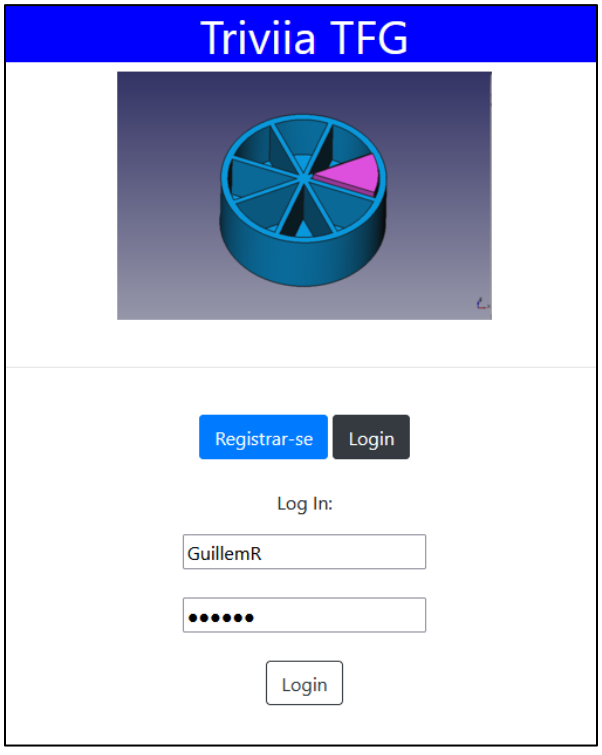

Figura 9. Pàgina d'inici de sessió d'usuari.

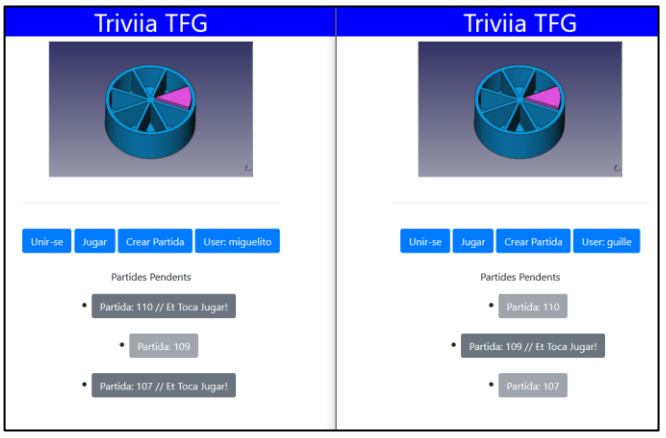

Figura 10. Llistat de partides pendents de dos usuaris diferents.

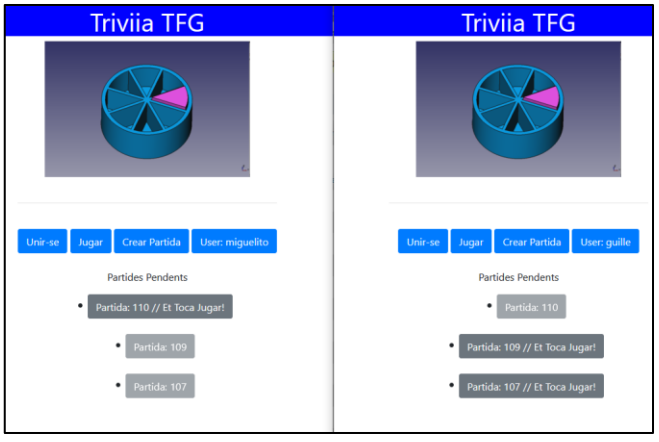

Figura 11. Llistat de partides pendents actualitzat, després que el jugador de l'esquerra ("miguelito") hagi jugat el seu torn en una d'elles.

A les figures 10 i 11 es poden veure els llistats de partides pendents de dos usuaris diferents. Per a fer que la imatge fos representativa, s'ha creat tres partides entre ells. A la figura 10, el jugador de l'esquerra té dues partides esperant el seu torn i, el de la dreta, en té una. A la figua 11 es veu el que passa quan el jugador de l'esquerra juga el seu torn: aquest passa a tenir només una partida pendent de la seva acció i el de la dreta passa a tenir-ne dues. Quan alguna d'aquestes partides acaba, desapareix de la llista de partides pendents.

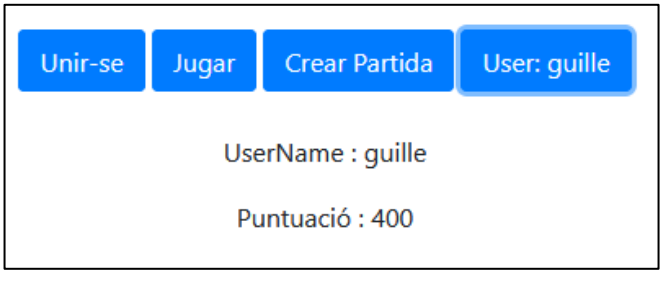

Figura 12. Pàgina d'informació del jugador.

Aquesta fotografia mostra la pàgina de perfil del jugador. És molt senzilla, però es pot veure el seu nom d'usuari i la puntuació total aconseguida.

Figura 13. Pàgina de creació de partida privada.

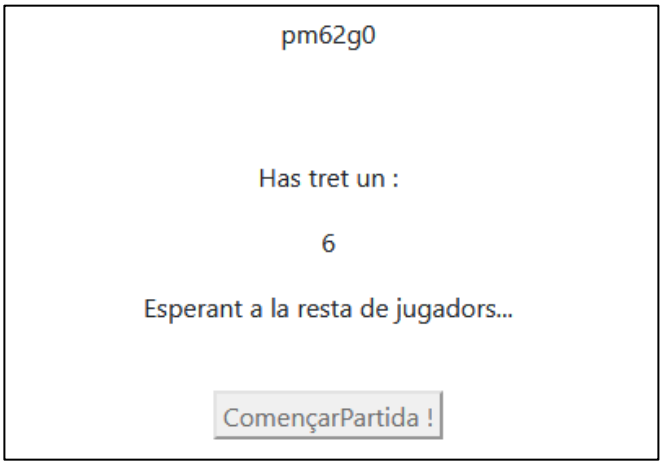

Figura 14. Partida privada creada i començant. A la part superior es pot veure el codi de tauler que serveix per entrar a la partida.

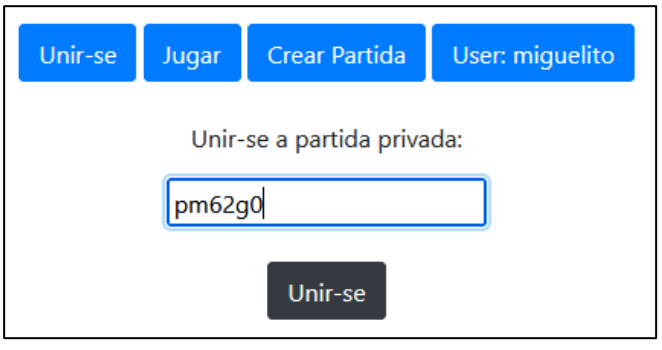

Figura 15. Pàgina per a unir-se a una partida privada.

Les figures 13 i 14 són el que veuria un usuari creant una partida privada: es pot seleccionar el màxim de jugadors i la dificultat de les preguntes i, un cop creada la partida, es pot veure el codi de tauler necessari per unir-se. El servidor espera que hi hagi tots els jugadors (màxim establert) i aquests llencin un dau per establir l'ordre i començar la partida. La figura 15 mostra el que hauria de fer un usuari per unir-se a la partida creada a la figura 14; en aquest cas concret, "miguelito" s'uneix a la partida de "guille".

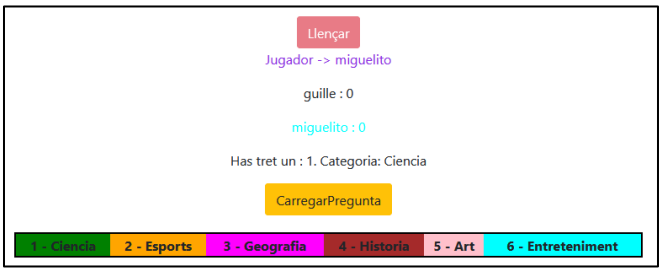

Figura 16. El jugador ha llençat un dau per triar categoria aleatòriament.

Quan un jugador entra en partida veu el que hi ha plasmat a la figura 16. Si és el seu torn se li permet llençar un dau per triar la categoria de la seva pregunta. Si no fos el seu torn, hauria d'esperar i, mentrestant, es mostraria el nom d'usuari del jugador a qui toca llençar.

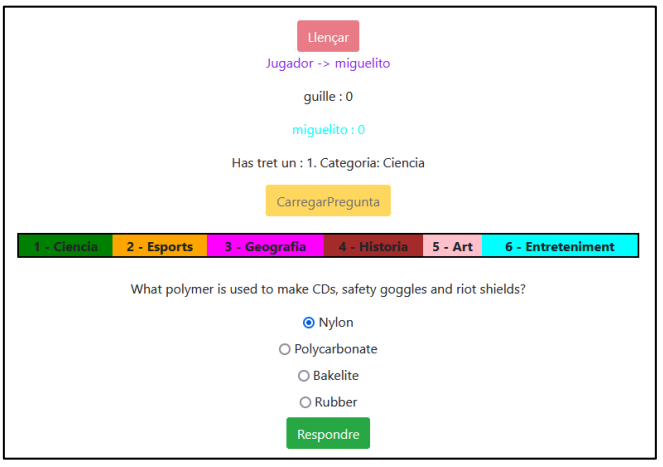

Figura 17. S'ha carregat la pregunta mitjançant l'API d'Open Trivia DB i s'espera la resposta del jugador.

A la figura 17 es pot veure la pàgina amb la pregunta carregada a l'espera de la resposta de l'usuari.
A més, a l'annex hi ha adjuntes les figures 18, 19 i 20 on es poden apreciar les comunicacions en temps real mostrant com es veuria una partida entre 3 jugadors. Les tres imatges s'han pres de la manera més representativa possible per mostrar la simultaneïtat de la situació en cadascun dels clients. A la figura 20 es mostra el que veuen els jugadors quan acaba la partida: el resum de puntuacions i qui ha estat el guanyador.
Val a dir, però, que en essència, s'ha complert amb tots els objectius tècnics proposats en un inici, tot i que el joc hagi experimentat algun canvi: s'ha aconseguit crear una web implementant un joc de preguntes per torns, s'ha aconseguit comunicacions websockets en temps real i gestionar la informació d'usuaris i partides mitjançant l'ús de la base de dades.

7 Conclusions

La implementació d'una solució amb websockets per permetre la comunicació a temps real ha estat més complexa del que en un primer moment s'esperava, però és una tecnologia molt potent i que s'utilitza en moltes plataformes actualment. El paradigma web-API com a eina de programació web és molt versàtil, donat que permet, entre altres coses, un complet desacoblament del desenvolupament de la part de servidor i de la part de client compartint entre elles només el format de les comunicacions.

També val a dir que un projecte com aquest es pot escalar fins a convertir-se en un autèntic joc online amb una experiència d'usuari molt més complerta del que es disposa ara i, també, es podria convertir en una aplicació mòbil per a arribar a molta més gent (és potser la plataforma més utilitzada actualment per a jocs d'aquest caire).

Com a treball futur, es podria millorar l'estat de les partides per mantenir l'identificador de jugador en la part del client. Això facilitaria una millor solució per simular l'ús d'una variable de sessió permetent el refresc de pàgina a la part del client.

També es podria millorar la part visual de la web, que en general ha quedat fora dels objectius principals d'aquest projecte. Es podria, per exemple, afegir un tauler i moure-hi per sobre una fitxa per a cada jugador o es podria afegir una animació per simular el llançament del dau d'una manera més visual. També es podria implementar un timeout per a respondre les preguntes i que les puntuacions fossin en funció del temps emprat per a respondre; és a dir, quant més ràpid es respon, més punts es reben en encertar (així també s'evitaria l'empat, un cas que ara mateix es pot donar). Finalment, es podria afegir un sistema de nivells per als jugadors, de manera que la puntuació de partida es convertís en punts d'experiència i permetés als jugadors augmentar de nivell conforme més i millor juguin. Això aportaria al joc un extra de competitivitat i fidelitzaria als usuaris.

En definitiva, s'ha construït una molt bona base en un projecte que podria créixer fàcilment i s'ha aconseguit complir amb tots els objectius proposats a l'inici d'aquest treball.

Referències

[1] "¿Qué es la gestión híbrida de proyectos?" wrike.com, 2021. [Online]. Available: https://www.wrike.com/es/blog/que-es-la-gestion-hibrida-de-proyectos/ [Accessed 05-March-2021]

[2] "Desarrollo en cascada", wikipedia.org, 2021. [Online]. Available: https://es.wikipedia.org/wiki/Desarrollo_en_cascada [Accessed 05-March-2021]

[3] "Tipos de metodologías ágiles: adáptate al cambio continuo en tus proyectos", sinnaps.com, 2021. [Online]. Available: https://www.sinnaps.com/blog-gestion-proyectos/tipos-de-metodologias-agiles [Accessed 05-March-2021]

[4] "Llenguatge de modelització unificat", wikipedia.org, 2021. [Online]. Available: https://ca.wikipedia.org/wiki/Llenguatge_de_modelitzaci%C3%B3_unificat [Accessed 02-Apr-2021]

[5] "Qué es un diagrama entidad-relación", LucidChart.com, 2021. [Online]. Available: https://www.lucidchart.com/pages/es/que-es-un-diagrama-entidad-relacion [Accessed 02-Apr-2021]

[6] "Install and configure Apache", ubuntu.com, 2021. [Online]. Available: https://ubuntu.com/tutorials/install-and-configure-apache#1-overview [Accessed 05-March-2021]

[7] "Raspberry Pi 3 model B", raspberrypi.org, 2021. [Online]. Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[8] "How to install Raspbian on the Raspberry Pi", thepi.io, 2017. [Online]. Available: https://thepi.io/how-to-install-raspbian-on-the-raspberry-pi/ [Accessed 05-March-2021]

[9] "Bringing MySQL to the web", phpmyadmin.net, 2021. [Online]. Available: https://www.phpmyadmin.net/ [Accessed 16-March-2021]

[10] "HTTP Request Methods", w3schools.com, 2021. [Online]. Available: https://www.w3schools.com/tags/ref_httpmethods.asp [Accessed 10-May-2021]

[11] "Free to use, user-contributed trivia question database." OpenTrivia Database, 2021. [Online]. Available: https://opentdb.com/ [Accessed 10-May-2021]

[12] Chris Pitt, "Pro PHP MVC", p. 1-27, 2012. [Accessed 09-March-2021]

[13] "Rest resource naming guide", restfulapi.com, 2021. [Online]. Available: https://restfulapi.net/resource-naming/ [Accessed 20-Apr-2021]

[14] Manual PHP, php.net, 2021. [Online]. Available: https://www.php.net/manual/es/ [Recurrent accesses]

[15] "WebSockets con PHP + Javascript (Vanilla)", www.natapuntes.es, 2018 [Online]. Available: https://www.natapuntes.es/websockets-con-php-javascript/ [Accessed 06-Apr-2021]

[16] "The WebSocket Protocol", tools.ietf.org, 2011. [Online]. Available: https://tools.ietf.org/html/rfc6455 [Accessed 06-Apr-2021]

[17] "PHP-Websockets", github.com, 2018. [Online]. Available: https://github.com/ghedipunk/PHP-Websockets [Accessed 20-Apr-2021]

[18] "JSON", wikipedia.org, 2021. [Online]. Available: https://es.wikipedia.org/wiki/JSON [Accessed 06-Apr-2021]

[19] JavaScript Tutorials, mozilla.org, 2021. [Online]. Available: https://developer.mozilla.org/ca/docs/Web/JavaScript [Recurrent accesses]

[20] "Qué es la programacion orientada a objetos", desarrolloweb.com, 2019. [Online]. Available: https://desarrolloweb.com/articulos/499.php [Accessed 06-Apr-2021]

[21] "PHP Associative Arrays", w3schools.com, 2021. [Online]. Available: https://www.w3schools.com/php/php_arrays_associative.asp

[22] "SQL Tutorial", w3schools.com, 2021. [Online]. Available: https://www.w3schools.com/sql/ [Recurrent accesses]

[23] "JQuery Ajax", Jquery, 2021. [Online]. Available: https://api.jquery.com/jquery.ajax/ [Accessed 06-Apr-2021]

[24] "Cross-site scripting", wikipedia.org, 2021. [Online]. Available: https://es.wikipedia.org/wiki/Cross-site_scripting [Accessed 08-May-2021]

[25] CSS Tutorial, w3schools.com, 2021. [Online]. Available: https://www.w3schools.com/css/ [Recurrent accesses]

ANNEX

Material addicional

A continuació s'adjunta tres imatges mostrant el comportament del software en tres suposats clients diferents jugant una partida.
La última és una imatge amb el diagrama de Gantt generat amb el Microsoft Project d'acord amb el Planning preparat per a la primera entrega de seguiment.
Aquestes imatges s'adjunten aquí donat que el seu format complica integrar-les en el document amb un tamany que permeti interpretar-les correctament.

Figura 18. Demostració de joc en temps real en una partida entre 3 jugadors.

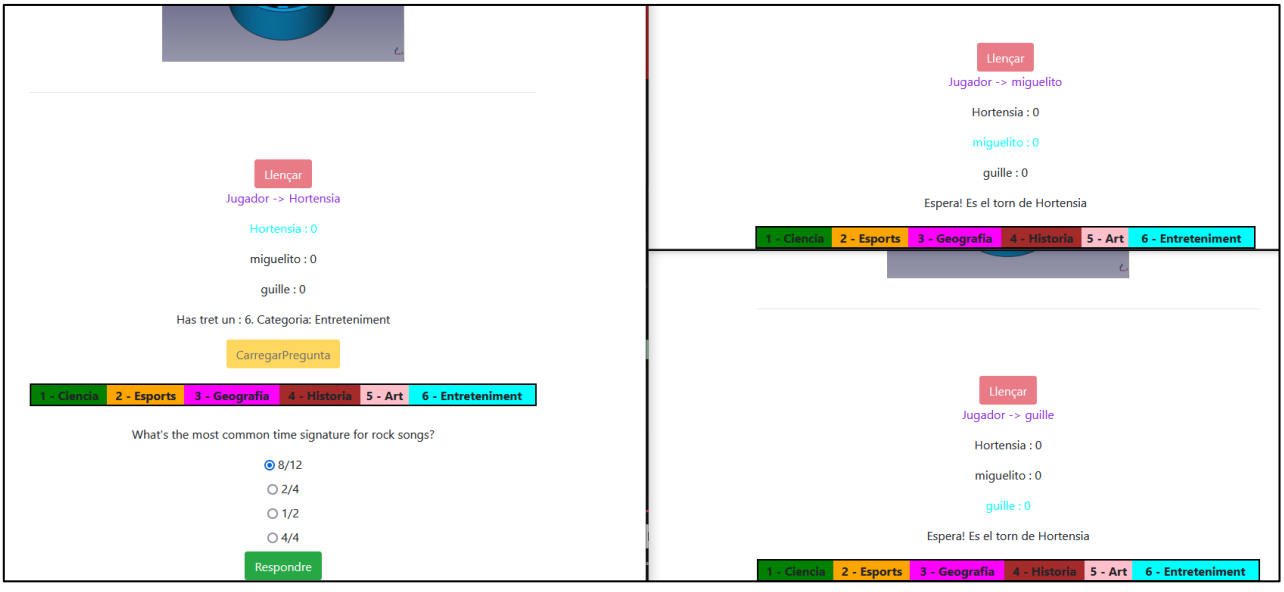

Figura 19. Demostració de joc en temps real: L'Hortensia es disposa a respondre la seva primera pregunta.

Figura 20. Pantalla de fi de partida mostrada als 3 jugadors a l'hora. En miguelito ha guanyat!

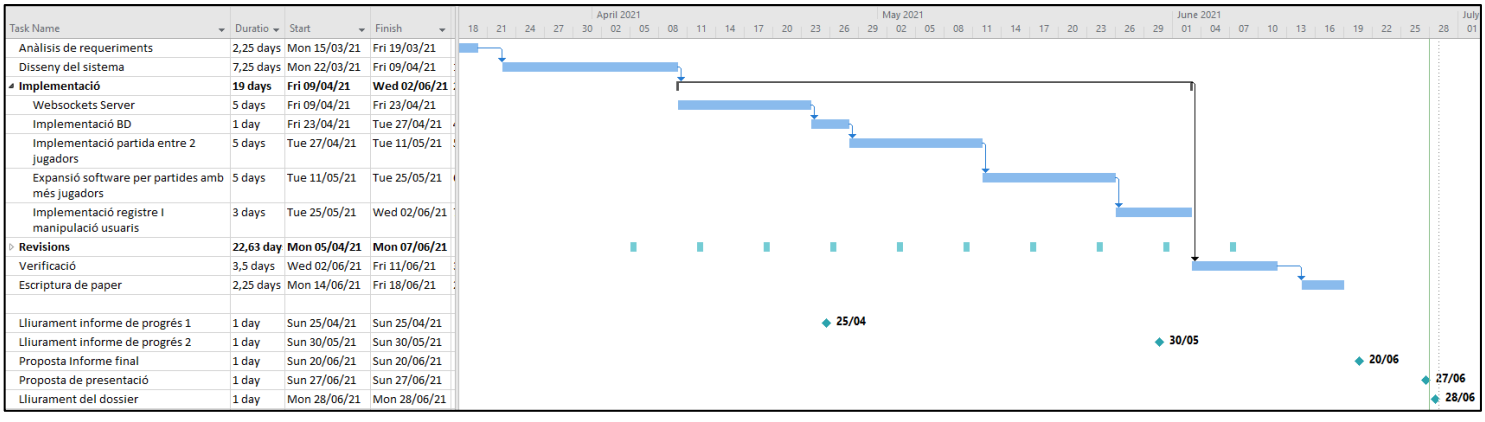

Figura 21. Diagrama de Gantt del projecte.