UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU D'ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Wordle Crypto: Una aplicació Web3

Autor: Jordi Bonet Valiente

Director:	Dr. Eduardo Urruticoechea
Realitzat a:	Departament de Matemàtiques i Enginyeria informàtica

Barcelona, 10 de juny de 2022

Abstract

Wordle Crypto is a Web3 application, which means that it uses the blockchain technology, Ethereum, to perform economic transactions.

It adapts the popular Wordle game so that the user can use the obtaining of cryptocurrencies as a motivation to play, adding new functionalities and services, with a reward system to incentivize the player.

In this way, it is intended to demonstrate the value and usefulness that the Web3 world can bring to a website.

Resum

Wordle Crypto és una aplicació Web3, el que significa que utilitza la tecnologia de la *blockchain*, Ethereum, per realitzar transaccions econòmiques.

Aquesta adapta el popular joc Wordle perquè l'usuari utillitzi l'obtenció de criptomonedes com a motivació per jugar, afegint noves funcionalitats i serveis, amb un sistema de recompenses per incentivar al jugador.

D'aquesta manera es vol demostrar el valor i la utilitat que pot arribar a aportar el món Web3 a una pàgina web.

Resumen

Wordle Crypto es una aplicación Web3, lo que significa que utiliza la tecnología de la *blockchain*, Ethereum, para realizar transacciones económicas.

Ésta adapta el popular juego Wordle para que el usuario utilice la obtención de criptomonedas como motivación para jugar, añadiendo nuevas funcionalidades y servicios, con un sistema de recompensas para incentivar al jugador.

De esta forma se quiere demostrar el valor y la utilidad que puede llegar a aportar el mundo Web3 a una página web.

Agraïments

Vull agrair al meu tutor Eduardo per acceptar la meva petició del TFG, per la proposta del tema i pel seguiment setmanal, ajudant-me en els dubtes que m'han sorgit. També agrair als meus pares per la confiança i a algunes amistats per provar l'aplicació, ajudant-me a trobar errors més ràpidament.

Índex

1 Introducció

1.1 Context

Els últims anys s'ha popularitzat una tecnologia que promet ser el futur: la *blockchain*. Aquesta és coneguda per les populars criptomonedes i per les diferents aplicacions d'aquestes, com els NFTs. De manera que cada vegada sorgeixen més serveis, plataformes i aplicacions compatibles amb ella. Aquestes webs que treballen amb la *blockchain* es coneixen com a aplicacions Web3.

Per una altra banda, encara són les aplicacions Web2 les que es viralitzen a causa de la seva senzillesa i satisfacció. Un exemple recent és Wordle, una aplicació on cada dia s'ha d'esbrinar una paraula de cinc lletres diferent de la del dia anterior, la qual és comuna per tots els seus usuaris. Com se li afegeix la tecnologia de la *blockchain* a una aplicació de masses d'avui dia, com Wordle? Quin valor pot aportar a l'aplicació?

Dins del Grau d'Enginyeria Informàtica, les assignatures més relacionades amb aquest projecte són aquelles directament relacionades amb el disseny i desenvolupament de software. Les assignatures de Disseny de Software, Projecte Integrat de Software i Factors Humans i Computació donen les eines per planificar i dissenyar un projecte com aquest. A l'assignatura de Software Distribuït es va introduïr la *blockchain*, així com verificar l'autenticitat de l'usuari i principis de desenvolupament web (com els *endpoints* i la comunicació entre el *frontend* i el *backend*). L'assignatura d'Enginyeria de Software, afegeix eines i metodologies més eficaces per a la gestió de projectes. L'assignatura de Fonaments de Cirberseguretat també ha sigut útil a l'hora de protegir la web.

1.2 Motivació

Va sorgir la idea d'agafar una aplicació web d'èxit, coneguda com a Wordle, amb una implementació i una jugabilitat molt senzilla i accessible, la qual cosa fa que tingui un perfil d'usuari molt ampli i de totes de les edats, i afegir-li valor amb la Web3, amb les criptomonedes.

Per aconseguir això, és necessari fer canvis en la lògica i les característiques del joc de Wordle perquè la integració de les criptomonedes tingui sentit i aporti valor, així com definir un sistema de recompenses que sigui just i satisfactori per l'usuari, de manera que se senti motivat a jugar usant les criptomonedes.

1.3 Objectiu principal

L'objectiu principal d'aquest projecte és demostrar la utilitat i el valor que pot arribar a aportar el món Web3 a les aplicacions desenvolupant una aplicació Web3.

Concretament, es vol fer servir les criptomonedes per realitzar transaccions econòmiques dins l'aplicació a canvi de funcionalitats o serveis dins d'aquesta. D'aquesta

manera es veu les diferents aplicacions que poden tenir les criptomonedes dins d'una aplicació Web3.

També està la possibilitat d'autenticar-se amb una adreça pública d'un compte de la *blockchain*, trencant amb el tradicional mecanisme d'usuari i contrasenya de la majoria de webs.

S'ha decidit adaptar el popular joc Wordle per dur a terme transaccions amb criptomonedes, afegint noves funcionalitats i adaptant el joc perquè l'usuari se senti impulsat, motivat, a jugar a Wordle per aconseguir-les.

1.4 Objectius específics

A partir de l'objectiu principal podem definir els següents objectius específics:

- Adquirir els coneixements necessaris del món Web3 (*blockchain*, Ethereum, transaccions amb criptomondes, etc.) per desenvolupar una aplicació Web3.
- Definir l'arquitectura de l'aplicació Web3, així com les tecnologies que s'utilitzaran.
- Crear una pàgina web en producció amb un *frontend* i un *backend* funcionals i connectats entre si.
- Implementar la identificació d'usuaris usant la seva adreça pública, mitjançant MetaMask.
- Integrar els pagaments, transaccions de criptomonedes de l'usuari a l'aplicació, amb MetaMask.
- Definir i implementar el sistema de recompenses del mode principal de Wordle Crypto.
- Crear nous apartats dins la web amb noves funcionalitats i serveis que emprin les transaccions complementant al mode principal i a l'experiència d'usuari.
- Afegir més seguretat a l'aplicació. Encriptar la paraula del dia i controlar l'accés als endpoints del *backend*.
- Documentació i memòria.

1.5 Planificació temporal

A continuació es mostra el diagrama de Gantt amb el repartiment de tasques de cada sprint.

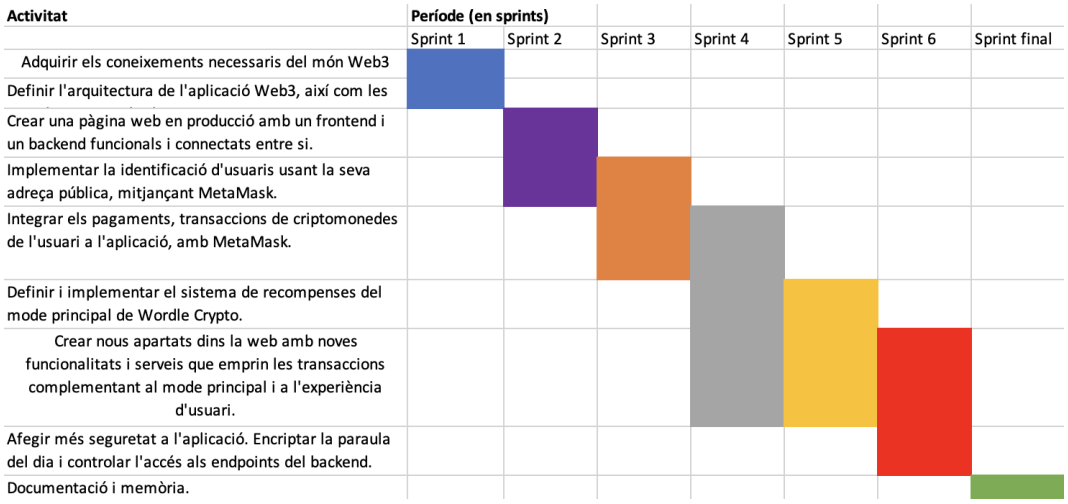

Figura 1: Diagrama de Gantt amb la planificació del projecte

El projecte s'ha estructurat en 7 sprints:

- Sprint 1: del 26 de gener al 22 de març.
- Sprint 2: del 22 al 29 de març.
- Sprint 3: del 29 de març al 15 d'abril.
- Sprint 4: del 15 d'abril al 27 d'abril.
- Sprint 5: del 27 d'abril al 6 de maig.
- Sprint 6: del 6 al 21 de maig.
- Últim sprint: del 21 de maig al 12 de juny.

1.6 Estructura del document

Breu descripció del contingut de cadascun dels capítols de la memòria.

- **Capítol 1, Introducció**: En aquest capítol s'introdueix el projecte, els objectius que es volen assolir i la planificació temporal per a realitzar-lo.
- **Capítol 2, Conceptes previs**: En aquest capítol s'expliquen els conceptes teòrics necessaris per comprendre el projecte.
- **Capítol 3, Anàlisi**: En aquest capítol s'analitza la funcionalitat, es defineix el sistema de recompenses, declaren les decisions de disseny i el disseny per identificar als usuaris amb Web3.
- **Capítol 4, Disseny**: En aquest capítol detalla el disseny de l'aplicació, la seva arquitectura, les seves transaccions i la base de dades utilitzada.

- **Capítol 5, Implementació**: En aquest capítol es fa èmfasis en les funcionalitats, la lògica i les característiques desenvolupades de l'aplicació.
- **Capítol 6, Conclusions i treball futur**: En aquest capítol es presenten les conclusions del treball i es comenten les línies de treball futur que plantegen els resultats del projecte.

2 Conceptes previs

2.1 Blockchain

Una *blockchain* és una base de dades pública. La seva nomenclatura prové dels següents conceptes: un "*Block*"és on es guarda dades i l'estat de la xarxa (per exemple, una transacció és exitosa quan s'afegeix les seves dades a un block) i "*Chain*"fa refèrencia al fet que cada "Block"referencia al seu pare (al block previ); és a dir, s'encadenen de manera que la data d'un block canviaria tota la subseqüència de blocks, és per això que es necessita definir un consens per tota la xarxa.

Tots els ordinadors de la xarxa, coneguts com a nodes, han d'estar d'acord a l'hora d'afegir un block i la cadena en el seu conjunt, d'aquesta manera s'assegura que tots els que interactuen amb la cadena de blocs tinguin les mateixes dades i per aconseguir-ho, les *blockchains* fan servir un mecanisme de consens. El mecanisme de consens defineix com s'afegeix un nou bloc a la cadena, com es mina un bloc. En un exemple pràctic, quan s'envia una criptomoneda a algú, la transacció s'ha de minar, depenent del tipus mecanisme de consens, i incloure en un bloc nou, actualitzant l'estat de la xarxa.

2.2 Ethereum

Al cas particular d'Ethereum, es fa servir una única màquina virtual global per guardar l'estat de la *blockchain*, del qual tots els participants de la xarxa Ethereum emmagatzemen i accepten: l'*Ethereum Virtual Machine* (EVM). Tots els ordinadors, els nodes, que participen en la xarxa Ethereum conserven una cópia de l'estat de l'EVM.

El funcionament és el següent: un node pot emetre una sol·licitud perquè aquest ordinador realitzi un càlcul arbitrari i els altres participants de la xarxa verifiquen, validen i duen a terme ("executen") el càlcul, el que provoca un canvi d'estat a l'EVM, que es confirma i es propaga per tota la xarxa. Aquestes sol·licituds de càlcul es coneixen com a peticions de transacció, de manera que el registre de totes les transaccions i l'estat actual de l'EVM s'emmagatzema a la cadena de blocs, que al mateix temps s'emmagatzema i és aprovada per tots els nodes.

L'ether (ETH) és la criptomoneda nativa d'Ethereum amb l'objectiu de permetre un mercat per la computació, on es proporciona un incentiu econòmic als participants per verificar i executar sol·licituds de transaccions i proporcionar recursos computacionals a la xarxa. Això es tradueix al següent funcionament: quan un participant emet una sol·licitud de transacció, també ha d'oferir una certa quantitat d'èter a la xarxa com a recompensa, la qual s'atorgarà a qui eventualment faci la feina de verificar la transacció, executar-la, comprometre-la amb la cadena de blocs i transmetre-la a la xarxa.

Un *Smart Contract*, o contracte intel·ligent, és un programa, un fragment de codi que es pot reutilitzar, carregat i executat a la xarxa d'Ethereum, emmagatzemat dins l'EVM, cridat amb determinats paràmetres, que realitza algunes accions o

càlcul en l'EVM si es compleixen determinades condicions. Els desenvolupadors poden publicar un contracte intel·ligent a la xarxa, fent servir la blockchain com a capa de dades, pagant Ether a la xarxa. Llavors, altre usuari pot cridar a aquest contracte intel·ligent per executar el seu codi, a canvi d'ether. Gràcies als contractes intel·ligents, els desenvolupadors poden crear i desplegar aplicacions i serveis més complexos.

2.3 Comptes d'Ethereum

Un compte d'Ethereum és una entitat on es guarda l'ether (ETH) que pot enviar transaccions a Ethereum. Els comptes i els saldos de comptes s'emmagatzemen a l'EVM, formant part de l'estat global d'EVM.

Els comptes es poden controlar per l'usuari, anomenats comptes de propietat externa, o desplegar-se com a contractes intel·ligents, també anomenats comptes de contracte. En tots dos es pot rebre, guardar i trametre ETH i *tokens*, com també interaccionar amb contractes intel·ligents ja desplegats.

Però és només en els comptes de propietat externa on donar-se d'alta és gratuït, pot començar les transaccions i es pot enviar exclusivament ETH o *tokens*. En canvi, crear un compte de contracte costa ether, ja que s'emmagatzema a la xarxa i ocupa un espai en un bloc que s'ha de minar (i els miners han de rebre una recompensa pel cost que suposa), només es poden enviar transaccions en resposta a la recepció d'una transacció (és a dir, no permet inicialitzar transacció en si) i permet executar codi quan es rep una transacció d'un compte extern, amb totes les diferents aplicacions que permet.

Un compte examinat té quatre camps:

1. *Nonce*. Un comptador del nombre de transaccions trameses des del compte, garantint que aquestes només es processin una vegada.
2. *Balance* o saldo. El nombre de WEI, una denominació d'ETH, en possessió d'aquesta adreça. La conversió de 1e+18 WEI equival a un ETH.
3. *Code hash*. Aquest *hash* fa referència al codi d'un compte de contracte a l'EVM, executant-se si el compte rep una trucada de missatge, i, per als de propietat externa, aquest camp és buit.
4. *Storage root*. És un *hash* on està codificat el contingut emmagatzemat del compte.

A la següent figura es pot apreciar com interactuen els camps:

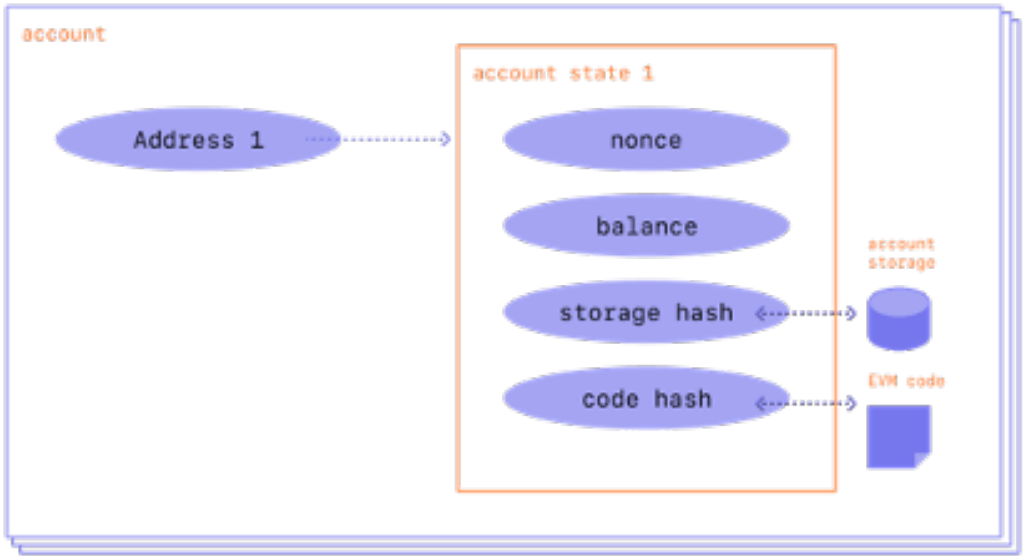

Figura 2: Diagrama dels camps d'un compte d'Ethereum

Cada compte està format per un parell de claus criptogràfiques, una pública i altra privada, per demostrar que una transacció va ser realment signada pel remitent i eviten falsificacions.

La clau privada es fa servir per signar transaccions; és a dir, atorga la custòdia dels fons associats al compte. Realment mai s'arriba a tenir les criptomonedes, sâobté les seves claus privades; ja que els fons sempre es troba en la xarxa d'Ethereum. Quan es crea un nou compte, es genera una clau privada aleatòria, formada per 64 caràcters hexadecimals, i es pot xifrar amb una contrasenya.

Llavors, la clau pública es genera a partir de la clau privada mitjançant l'algoritme de signatura digital de corba el·líptica (*Elliptic Curve Digital Signature Algorithm*). S'aconsegueix l'adreça pública del compte agafant els darrers 20 bytes del *hash* de 256 bits de la clau pública i afegint "0x"al principi. Es poden assolir noves claus públiques d'una clau privada, però no a l'inrevés (tanmateix, això és important mantenir-la privada). Signar transaccions produeix una signatura, que es pot fer servir per aconseguir la clau pública.

Finalment, cal destacar que no és mateix una cartera, un *wallet*, que un compte: el *wallet* és la interfície o l'aplicació que permet gestionar els comptes d'Ethereum.

2.4 Gas

Abans de parlar de les transaccions en Ethereum, és necessari introduir el concepte de "gas".

Ethereum és una plataforma de *software* que realitza càlculs senzills, els quals succeeixen simultàniament en un eixam d'ordinadors, els nodes, i hi ha un grup especial d'ells, anomenats miners, encarregats de protegir la xarxa dels atacs i fer

cálculs. Sense els miners, no hi hauria Ethereum i de manera que se'ls hi ha de pagar per aquesta tasca essencial que duen a terme.

Per pagar als miners, s'ha de quantificar el treball que fa Ethereum. Aquesta mesura s'anomena unitat de gas (*Gas Unit*). Ethereum només pot calcular un nombre limitat d'unitats de gas en un determinant moment, per la qual cosa els miners han de gestionar les sol·licituds. Els miners depenen del preu del gas (*Gas Price*) i del límit del gas (*Gas Limit*) per prioritzar-les.

Una unitat de gas mesura la feina feta, però no té un valor monetari. El preu del gas (Gas Price) es defineix amb unes denominacions d'ETH, anomenades WEI, assignades a cada unitat de gas. Es pot pagar més als miners perquè prioritzin una sol·licitud amb el preu del gas, utilitzant-lo com un "suborn"per saltar al capdavant de la fila. En canvi, si es posa aquest preu a 0, estarà encallat al final de la fila.

El límit de gas (*Gas Limit*) és una aproximació de la quantitat total de treball que se sol·licita, protegint de gastar ETH il·limitats garantint un punt de parada a la feina a computar a l'EVM. Si el límit és massa baix, el teu treball de computació no s'arriba a acabar; de manera que la transacció falla i es perd l'ETH. Si el treball finalitza abans d'arribar al límit de gas, es recupera l'ETH no emprat.

El preu del gas (*Gas Price*) multiplicat pel límit del gas (*Gas Limit*) és el cost total per realitzar transacció.

2.5 Transaccions en Ethereum

Les transaccions són instruccions de comptes signades criptogràficament, iniciades per un compte de propietat externa (és a dir, un compte gestionat per un humà) per actualitzar l'estat de la xarxa Ethereum, l'estat de l'EVM, i, per tant, s'han de difondre a tota la xarxa.

Qualsevol node pot emetre una sol·licitud d'execució d'una transacció a l'EVM; després que això passi, un miner executarà la transacció i propagarà el canvi d'estat resultant a la resta de la xarxa.

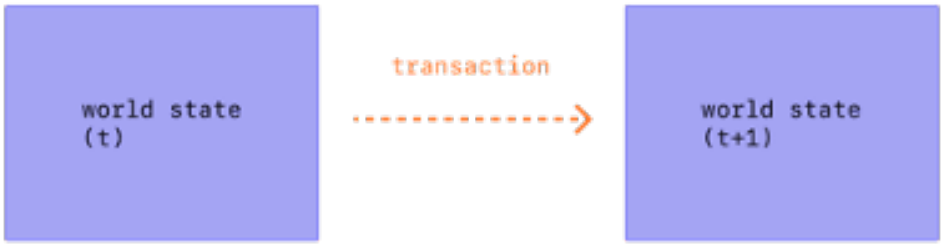

Figura 3: Diagrama complementari al concepte de transacció

Les transaccions requereixen una taxa (d'ether) i s'han de minar per ser vàlides. Cada transacció conté els següents camps:

- El *nonce*. Correspon al nombre de transaccions que porta el compte emissor.
- El preu de gas (*gasPrice*). Representa la quantitat d'ETH pagat pel signant per gas.
- El límit de gas (*gasLimit*). Normalment, amb 21.000 de gas es completen la majoria de les transaccions.
- L'adreça pública destí (representat amb *to* o *r*).
- L'adreça pública de l'emissor (anomenat *from* o *s*).
- Valor (*value*). Es refereix a la quantitat d'ETH a transferir del remitent al destinatari (en WEI, una denominació d'ETH).

2.6 Testnet o xarxa de proves

La cadena principal de la *blockchain* d'Ethereum, coneguda com a Mainnet, té un inconvenient a l'hora de desenvolupar aplicacions: té un cost econòmic significant escriure un bloc en aquesta i, com a conseqüència, fer transaccions també hi ha taxa d'ether a pagar perquè altres usuaris minin el bloc de la transacció. El que s'ha de pagar és el gas, el cost computacional necessari per dur a terme la transacció. És per aquest motiu que els desenvolupadors Web3 no fan servir directament la *blockchain* principal d'Ethereum, sinó que fan servir el que es coneix com una *testnet*.

Una *testnet* (o xarxa de proves) és una instància d'un altre *blockchain* que s'utilitzarà per testejar, per provar, i experimentar sense risc de perdre criptomonedes de la cadena principal de la blockchain en qüestió, Ethereum en el cas d'aquest projecte, amb un cost econòmic real. Les seves criptomonedes no tenen un valor real, no es poden intercanviar per diners, però es basen en la tecnologia de la cadena principal de la blockchain a la qual pertanyen, és un entorn similar al de la cadena principal.

Es fan servir per als desenvolupadors per provar el funcionament de les seves aplicacions Web3, dels seus *Smart Contracts* i DApps, abans de publicar-les a la cadena principal de la blockchain, on les transaccions tenen un valor real.

2.7 Proveïdor

Un proveïdor és una abstracció d'una connexió a la xarxa Ethereum, proporcionant una interfície concisa i consistent a la funcionalitat estàndard dels nodes Ethereum.

2.8 Signant

Un signant és una abstracció d'un compte d'Ethereum, utilitzada per signar transaccions i, un cop signades, enviar-les a la xarxa Ethereum per executar operacions de canvi d'estat a l'EVM.

Es pot distingir dos tipus de signants:

- Un *wallet* (o una cartera). És una classe que coneix la seva clau privada i, per tant, pot executar qualsevol transacció amb ella.
- Implementant un protocol anomenat JSON-RPC. Es fa servir per fer la connexió amb un node, per fer possible la comunicació amb la *blockchain* d'Ethereum. Un cop s'estableix connexió, el programa pot llegir informació i trametre transaccions a la xarxa.

3 Análisi

3.1 Análisi de la funcionalitat

Per complir l'objectiu principal s'han implementat funcionalitats addicionals sobre el concepte de Wordle perquè utilitzin la tecnologia Web3 per afegir un valor addicional.

Les dues tecnologies Web3 principals a implementar per satisfer l'objectiu són:

- La identificació d'usuaris a partir de l'adreça pública del compte d'Ethereum. D'aquesta manera es pot autentificar als usuaris d'una manera molt més amena que la popular alternativa d'usuari i contrasenya.
- Les transaccions de criptomonedes. Per implementar les transaccions amb criptomonedes, s'ha de definir un sistema de recompenses sobre l'aplicació base del Wordle. De manera que, l'usuari farà un abonament i, guanyant el joc, rebrà un recompensa que l'inciti a continuar jugant. A més del sistema de recompenses, les transaccions tenen altres aplicacions com intercanviar criptomonedes per objectes estètics pel joc.

3.2 Sistema de recompenses i el mecanisme de joc

Com s'ha plantejat a la motivació, un dels reptes principals d'aquest projecte és integrar un sistema de recompenses que incentivi al jugador a jugar per tal d'aconseguir ETH. Però abans de definir el sistema de recompenses, és necessari descriure el mecanisme de joc que es farà servir.

Trobem dos mecanismes o modes de joc diferents:

- *Wordle Crypto*. Aquest és el mode principal. Es basa en el popular concepte Wordle: hi ha una paraula del dia, el Wordle, i el primer que l'endevina obté una recompensa més gran que la resta. Per jugar a aquesta modalitat de joc cal apostar el dia anterior i es podrà jugar al Wordle a partir de l'endemà (a les 00:00).
- El mode multijugador. El mode multijugador està dissenyat perquè es pugui gaudir jugant a Wordle amb els amics en qualsevol moment, sense necessitat d'haver d'esperar a l'endemà, com al mode principal. En aquesta modalitat pots crear una sala, on es genera una clau que es pot compartir a altres jugadors perquè s'incorporin a aquesta sala. Un cop tots els usuaris estiguin preparats a la sala, l'amfitrió pot començar el joc i tothom juga a un Wordle comú: els jugadors han d'endevinar la mateixa paraula i el primer que l'endevini guanya una recompensa molt superior a la resta.

Els dos mecanismes de joc són similars entre si, de manera que el sistema de recompenses és comú als dos, tant a Wordle Crypto com al multijugador. Abans

de començar el joc, es fa una recollida en la qual tots els participants paguen 0.001 ETH; és a dir, la recaptació total és de 0.001 ETH multiplicat pel nombre de jugadors d'aquesta partida de Wordle. El primer jugador que endevini el Wordle guanya el 70% de la col·lecció total d'Ethereum, el següent guanyador es recompensa amb el 70% de l'ETH restant i així de manera successiva.

L'objectiu d'aquest sistema de recompenses és que més jugadors siguin recompensats per jugar a Wordle Crypto, de manera que, quan el primer jugador guanyi, la resta de jugadors encara se sentin incentivats a continuar jugant per tal d'aconseguir l'Ether restant.

3.3 Decisions de disseny

3.3.1 Ethereum

Avui dia, hi ha disponibles moltes *blockchains* diferents on poder realitzar les transaccions d'una aplicació Web3. A pesar d'això, l'elecció no resulta molt complicada, ja que, gràcies a les seves tecnologies que hem vist a l'apartat de *Conceptes previs*, la *blockchain* d'Ethereum és la més utilitzada pels desenvolupadors de la Web3 i, per tant, és la que té una comunitat més gran, amb molta més documentació i suport que la resta de *blockchains*.

És essencial que hi hagi una bona documentació i una gran comunitat per poder implementar la connexió amb la *blockchain* i fer servir les seves tecnologies per fer transaccions de la manera més ràpida possible. Un altre aspecte important que s'ha tingut en compte és que la xarxa d'Ethereum té disponibles diverses testnets amb les quals dur a terme transaccions gratuïtament, sense haver d'emprar ether real de la *Mainnet* d'Ethereum.

3.3.2 Rinkeby

Un cop escollit que la *blockchain* d'Ethereum és la *blockchain* amb la qual establir la connexió per realitzar les transaccions, s'ha d'escollir quina la xarxa es farà servir per a les transaccions. La xarxa principal d'Ethereum, la *Mainnet*, està descartada, ja que per desenvolupar la *blockchain* interessa utilitzar una *testnet* per no hi hagi un cost econòmic real a l'hora de dur a terme proves en l'aplicació.

La xarxa d'Ethereum disposa d'una gran varietat de xarxes de proves. Entre les *testnets* més populars dels darrers anys es troben Ropsten, Rinkeby i Kovan.

Al principi d'aquest projecte, es contemplava la idea d'implementar un *Smart Contract* (o contracte intel·ligent) que s'encarregaria de la gestió financera de la web. L'aplicació Wordle interactuaria amb el contracte intel·ligent quan l'usuari hagués de rebre o pagar ether. Un *Smart Contract* es programa en un llenguatge de programació poc comú, anomenat Solidity, i normalment interactuen amb una testnet anomenada RSK. Però es va descartar la idea, ja que no era necessari i resultava massa complex, i, llevat el comportament l'aplicació, només es necessita enviar transaccions des de diferents comptes d'Ethereum, la qual cosa es pot aconseguir a

través de llibreries de JavaScript i amb els proveïdors adequats.

Finalment, es fa servir una testnet de la *blockchain* d'Ethereum anomenada Rinkeby. Aquesta és la més popular i, com a conseqüència, té més suport, la qual cosa és important de cara a ser compatible amb els proveïdors i els signants que s'utilitzaran. També l'aspecte decisiu que es va decantar a favor d'aquesta xarxa de prova, és que és la *testnet* on és més fàcil es pot obtenir ether (hi ha moltes pàgines webs que en subministren diàriament), el qual és necessari per realitzar les proves de les transaccions.

3.3.3 MetaMask

Perquè l'usuari interactuï amb l'aplicació Web3 és necessari que tingui una cartera de criptomonedes, més conegut com a *wallet*, per tal de poder gestionar els seus comptes d'Ethereum, així com l'ether associats amb aquests. També és necessari definir un proveïdor a través del qual poder realitzar i confirmar les transaccions que es fan des del compte d'Ethereum, així com altres operacions com obtenir l'adreça pública del compte.

Aquestes tasques són les que s'encarrega l'extensió de navegador de MetaMask, o APP en el cas d'utilitzar un telèfon mòbil com a plataforma de l'aplicació. Aquesta extensió és una cartera de criptomonedes que es pot comunicar amb les aplicacions Web3; és a dir, abstreu la connexió amb la *blockchain* per fer operacions en aquesta, com consultar dades o fer transaccions, complint el rol de cartera de criptomonedes i de proveïdor Web3. A més, està implementat per connectar-se amb la xarxa d'Ethereum i es pot fer servir amb les diferents *testnets* d'aquesta, entre les quals es troba Rinkeby. Això és perquè és una eina molt orientada al desenvolupament Web3, oferint moltes eines pels programadors.

Llavors, la primera tasca que compleix MetaMask dins l'aplicacié és la identificació de l'usuari, proporcionant la seva adreça pública del compte d'Ethereum a l'aplicació Web3. D'aquesta manera, es podrà identificar l'usuari sense necessitat que el client proporcioni un nom d'usuari, o un correu, i una contrasenya, només cal el seu consentiment per compartir la seva adreça póblica del compte d'Ethereum amb l'aplicació Web3. D'aquesta manera es verifica l'autenticitat de l'usuari sense perdre seguretat, ja que l'usuari es verifica a través de MetaMask i és l'únic que té accés, que té la clau privada, del seu compte d'Ethereum.

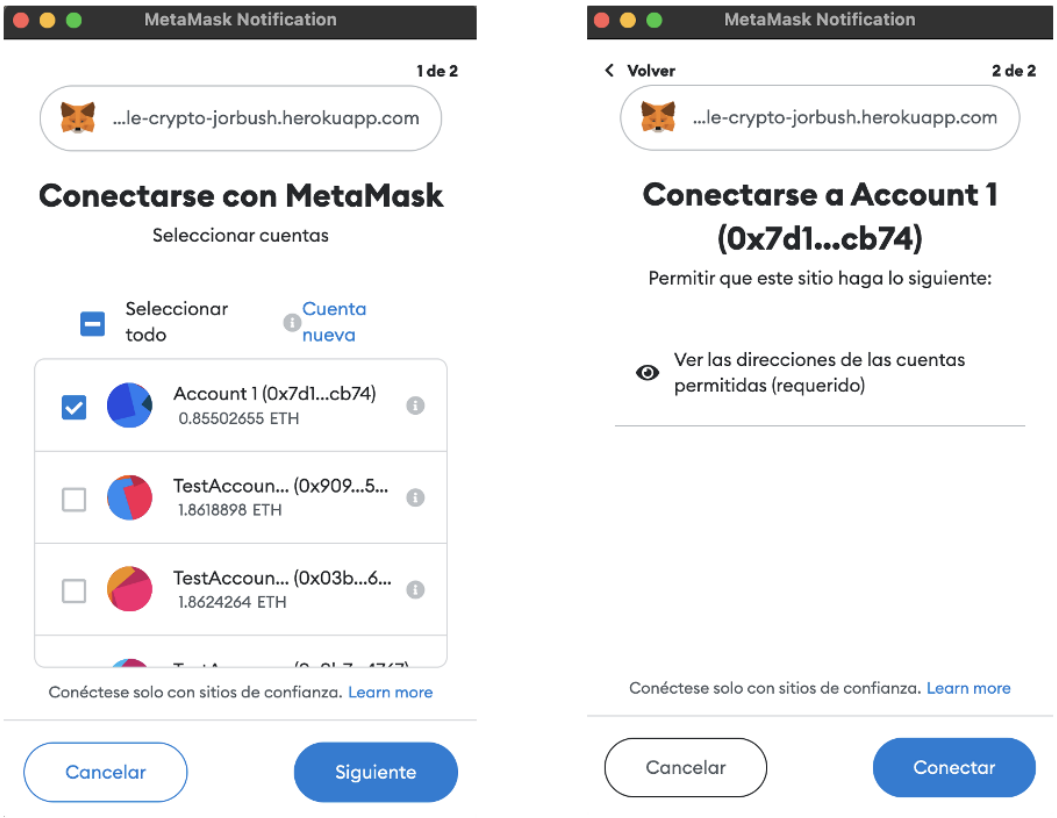

Figura 4: Captura de MetaMask amb la sol·licitud de connexió

L'altra funció que té com proveïdor Web3, és fer possibles les transaccions d'ether del client a un altre compte d'Ethereum, actuant també com a signant. Quan sigui necessari segons la funcionalitat dins l'aplicació Web3, se li sol·licitarà a l'usuari una confirmació de la transacció, amb una finestra emergent utilitzant la interfÃcie de MetaMask, ja que és l'usuari qui té el total control del seu compte d'Ethereum qui té la clau privada amb la qual fer possibles les transaccions d'ether a un altre compte i aquesta s'utilitza a través de MetaMask (fins i tot, des de la seva interfície es pot obtenir aquesta clau privada), funcionant com a signant.

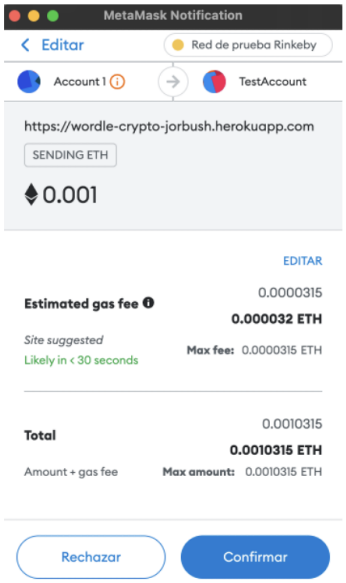

Figura 5: Captura de MetaMask amb la sol·licitud de transacció

Un cop ja es confirma la transacció, MetaMask actuarà com a proveïdor realitzant la connexió amb la blockchain i fent servir les seves tecnologies per portar a terme la transacció d'ether a l'altre compte.

Llavors, s'empra MetaMask com a proveïdor per assegurar la connexió amb el compte d'Ethereum, per proporcionar l'adreça pública d'aquest compte i, com a signant i proveïdor, per dur a terme transaccions del client al compte d'Ethereum de l'aplicació; és a dir, pels pagaments dins l'aplicació a canvi de serveis.

3.3.4 Infura

Encara falta un únic problema del sistema de recompenses amb Web3. Quan el client guanya una partida de Wordle, se li ha de retornar la recompensa d'ether corresponent.

Aquesta transacció es realitza des del compte d'Ethereum de la mateixa aplicació al compte de l'usuari. Llavors, ha de ser asíncrona i no pot requerir una interfície wallet per signar la transacció, ja que és el mateix servidor qui l'haurà de dur a terme de manera automàtica. Llevat aquests requeriments, MetaMask no és una opció viable.

Com s'ha vist a l'apartat de *Conceptes previs* sobre els comptes d'Ethereum, aquest tenen una adreça pública (poden tenir-ne més) i una clau privada. Aquesta clau privada serveix per signar i autoritzar les transaccions d'ether procedents d'aquest compte a un altre.

És per aquest motiu que es fa servir un altre proveïdor Web3, compatible amb Rinkeby, anomenat Infura.

Infura és una API en línia que proporciona accés a la xarxa d'Ethereum utilit-

zant HTTPS i WebSockets. Es pot accedir a aquest servei a través d'un *endpoint*, configurat específicament per la xarxa de Rinkeby. Aquesta s'encarrega de fer les transaccions de l'aplicació Web3 al client, fent servir la clau privada del compte d'Ethereum del servidor.

3.4 Identificació dels usuaris amb MetaMask

Una part de l'aplicació que fa servir la tecnologia Web3 aportant valor a l'aplicació innovant respecte a l'inici de sessió tradicional Web2 (és a dir, utilitzant usuari i contrasenya) és d'identificació d'usuari mitjançant la interfície de MetaMask.

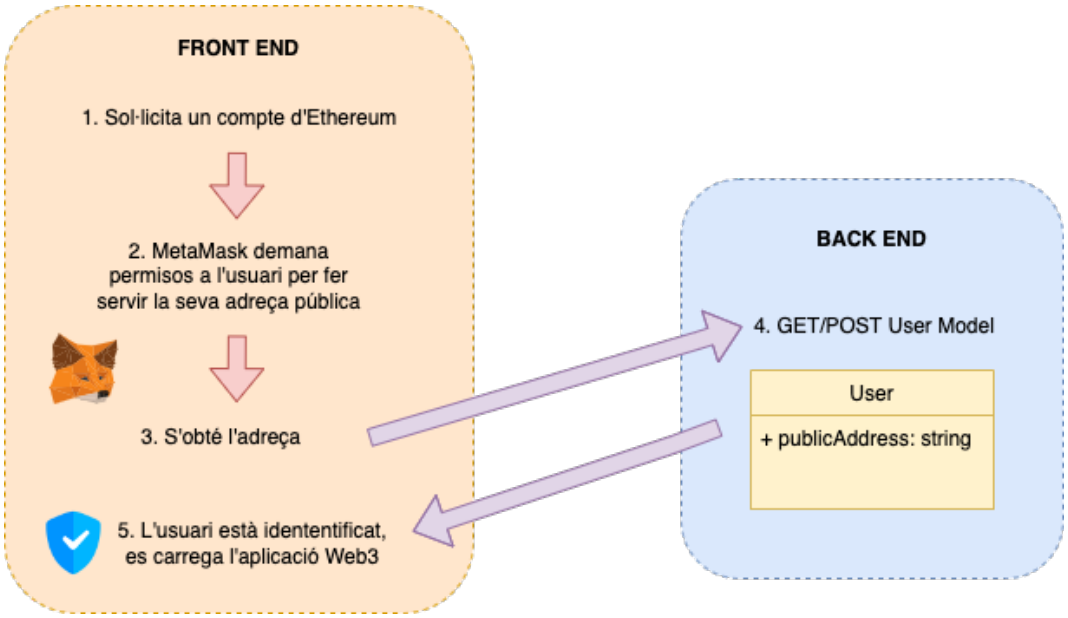

Figura 6: Diagrama de la identificació d'usuaris amb Web3

Els passos a seguir per identificar l'usuari són els següents:

1. El frontend sol·licita un compte d'Ethereum quan l'usuari selecciona l'opció d'iniciar sessió amb MetaMask.

2. L'extensió de MetaMask detecta aquesta sol·licitud d'un compte d'Ethereum. Obre una finestra emergent amb la seva interfície sol·licitant a l'usuari permisos per connectar-se a l'aplicació Web3 i compartir l'adreça pública del compte, donant-li l'opció de seleccionar un dels seus compte.

3. Un cop l'usuari selecciona un compte d'Ethereum i accepta les peticions, l'aplicació obté l'adreça pública amb la qual poder gestionar el compte de l'usuari dins l'aplicació.

4. Es comprova si l'usuari està ja donat d'alta a la base de dades. Si no és el cas, es dona d'alta fent una petició POST al backend i es recuperen les noves dades

generades. Si ja està donat d'alta, es realitza una petició GET al backend i es recupera la informació de l'usuari per l'aplicació.

5. Un cop s'han recuperat la informació de l'usuari de la base de dades, la identitat de l'usuari ja està comprovada a través de la seva adreça pública del seu compte d'Ethereum i carrega l'aplicació Web3.

4 Disseny

4.1 Arquitectura de la web

Al següent diagrama es pot apreciar l'arquitectura que fa servir la web, així com les tecnologies emprades per cada component.

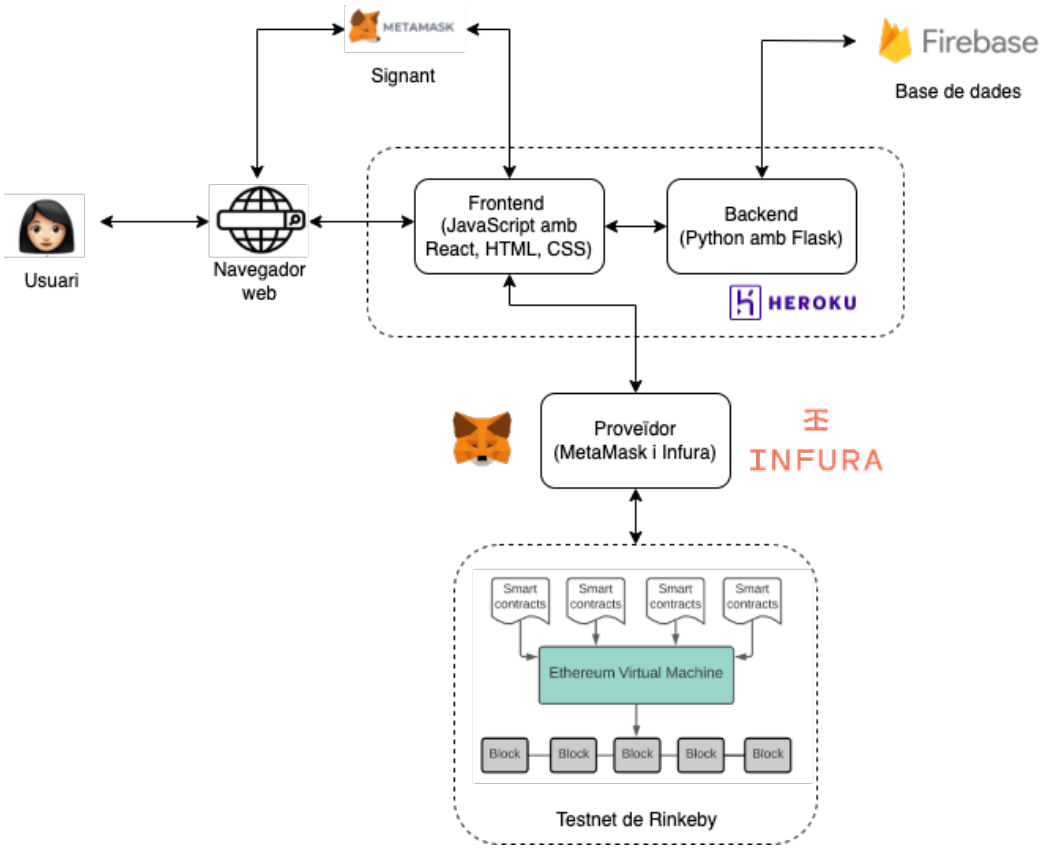

Figura 7: Diagrama de l'arquitectura de la web

L'usuari es comunica amb la web mitjançant el navegador web, pot ser Google Chrome, Firefox, Brave o qualsevol que sigui compatible amb l'extensió de MetaMask. Si es vol fer servir des de telèfons mòbils, s'ha de fer servir l'aplicació oficial de MetaMask. L'usuari principalment interactuarà amb el *frontend* de web, com una web corrent que no està relacionada amb el món de les criptomonedes. Però hi ha l'extensió MetaMask, és qui fa d'intermediari entre el frontend i l'usuari quan hi ha una transacció Web3 a signar, un abonament de criptomonedes per part de l'usuari a la web. Això és el que es coneix en Web3 com signant.

Tant el *frontend* com el *backend* es troben allotjats a un servidor web, concretament es troba publicat a Heroku. El *backend* s'encarrega de gestionar els endpoints que es criden des del *frontend* per accedir o modificar la base de dades, de definir les diferents taules o models de la base de dades i d'accedir i modificar la base de dades d'una manera segura. Com a base de dades es fa servir Firebase de Google.

Allà es guarden totes les dades que fa servir la web per oferir el seu servei d'una manera òptima.

Finalment, està els proveïdors Web3, que són els encarregats de gestionar les transaccions entre el frontend i la blockchain on s'escriuen les transaccions en blocs, que aquest cas es tracta de la *testnet* de Rinkeby basada en Ethereum. Els proveïdors utilitzats són MetaMask i Infura. MetaMask, a més de servir com signant pel client, té el paper de proveïdor quan es fa un pagament del client al servidor. En canvi, per realitzar transaccions del servidor al client s'ha d'usar-ne un altre, ja que és ha de ser una funcionalitat asíncrona que no pot requerir una signatura a l'instant de fer-la. És per aquest motiu que per fer les transaccions de la web al client, s'empra Infura.

4.2 Diagrama de transaccions

4.2.1 Pagament

Quan es fa un abonament per part de l'usuari a l'aplicació, per qualsevol servei de la pàgina web, ja sigui un pagament per jugar una partida de Wordle o per adquirir un tema de la botiga, el proveïdor encarregat de gestionar la transacció és MetaMask.

Al següent diagrama es pot observar quin és el flux d'ether entre carteres:

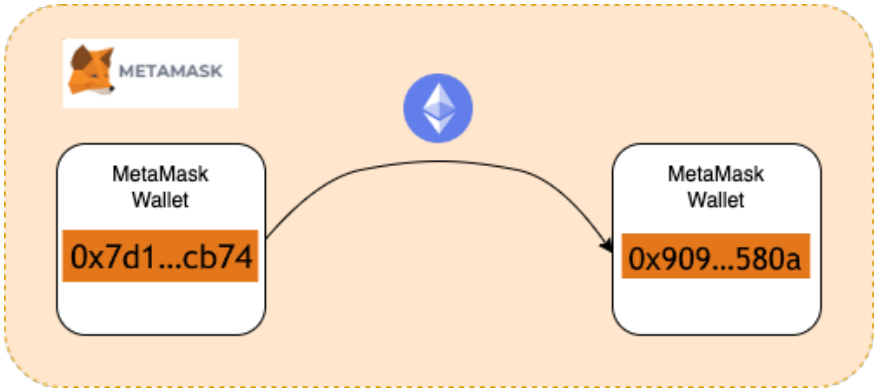

Figura 8: Diagrama d'una transacció de l'usuari a l'aplicació

Bàsicament, el que es fa a alt nivell és agafar l'ether del moneder del client, en aquest cas l'adreça de la cartera de l'usuari en qüestió és "0x7d1...cb74", i s'envia a la cartera de criptomonedes de l'aplicació, que correspon a "0x909...580a".

4.2.2 Recompensa

Quan es fa un abonament per part de l'aplicació a l'usuari, perquè aquest usuari ha guanyat una partida de Wordle i ha de rebre una recompensa d'ether, el proveïdor encarregat de gestionar la transacció és Infura.

Al següent diagrama es pot observar quin és el flux d'ether entre carteres:

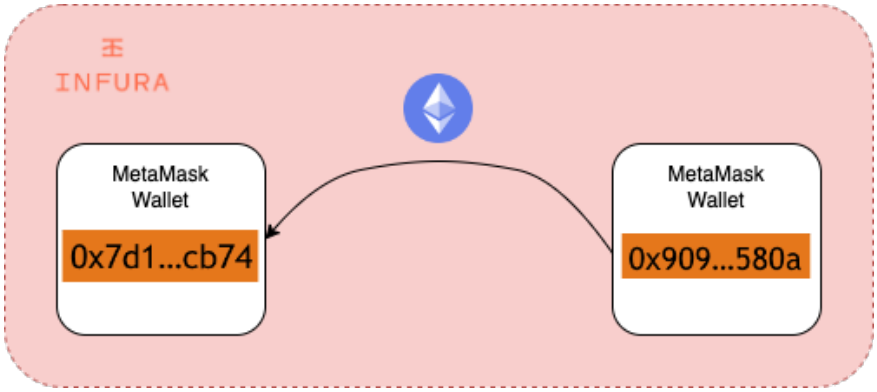

Figura 9: Diagrama d'una transacció de l'aplicació a l'usuari

Amb un flux contrari al del cas anterior, al cas del pagament, s'envia ether de la cartera de l'aplicació, la cartera amb l'adreça "0x909...580a", i s'envia a la cartera de criptomonedes de l'usuari, amb l'adreça "0x7d1...cb74".

També destacar que a altres serveis de la web s'utilitzen altres carteres de criptomonedes diferents per garantir el correcte funcionament dels diferents de manera que si una cartera es queda sense, la qual cosa és bastant complicada, només deixarà d'estar disponible un d'aquests serveis i no totes les funcions de la web.

Concretament, és l'apartat *Obtain Crypto* el que fa servir una adreça diferent de la resta. El motiu és que en aquest apartat es regalen ether per tal que l'usuari primerenc pugui jugar directament a Wordle Crypto sense haver de recórrer a pàgines externes. Llavors, si hi ha molts nous usuaris que demanen aquest ether, es pot donar el cas de la cartera es quedi buida i, en el cas que s'usés una única cartera digital de criptomonedes, el problema no només és que ja no es podria regala ether als nous usuaris, sinó que probablement tampoc funcionaria les recompenses que hem mencionat anteriorment, ja que, a més de la recompensa dels usuaris en si (pel fet que al final això és una col·lecta grupal dels usuaris que juguen la partida del Wordle), també s'ha de tenir en compte les taxes de les transaccions (són molt petites, pràcticament insignificants perquè es mou molt poc ether).

És per això que el flux d'ether entre carteres en el cas del servei *Obtain Crypto* es fa servir una adreça diferent que a les recompenses. L'esquema en qüestió és el següent:

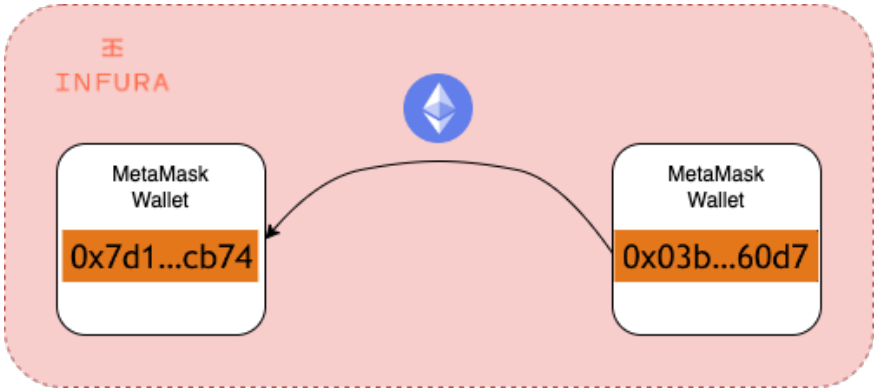

Figura 10: Diagrama d'una transacció en el mode *Obtain Crypto*

Com es pot observar l'adreça que es fa servir per gestionar l'ether a regalar als usuaris primerencs és "0x03b...60d7".

4.3 Base de dades

4.3.1 Introducció

Firebase va ser seleccionada des del començament, ja que es necessitava una base de dades senzilla, còmode i coneguda, i gràcies a l'experiència adquirida a l'assignatura de *Projecte Integrat de Software* i *Enginyeria del Software*, era l'opció més adequada.

També un altre factor decisiu va ser el fet de tenir la web en producció, publicada i funcional en Heroku des del primer dia de desenvolupament de l'aplicació, i la base de dades de Google és l'opció més ràpida en aquest cas.

Firebase és una base de dades NoSQL i suposa que s'han de definir al *backend* els diferents models que es necessiten pel funcionament de la web.

Els models són: *users*, *wordle-word*, *bets*, *wordle-spanish*, *transactions*, *rooms*, *obtain-crypto* i *themes*.

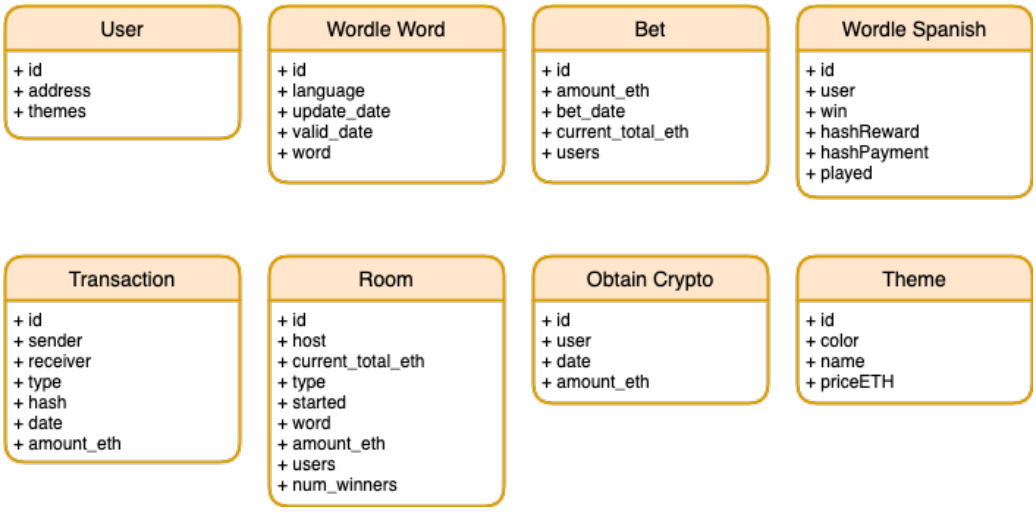

Figura 11: Diagrama dels models de la base de dades

Tots els models tenen un identificador únic (ID) per referenciar cadascun dels elements en el cas que fos necessari. Els models *wordle-word*, *bets* i *wordle-spanish* són els encarregats d'assegurar el funcionament del principal mode de joc de Wordle Crypto, de manera que estan relacionats entre si.

4.3.2 User

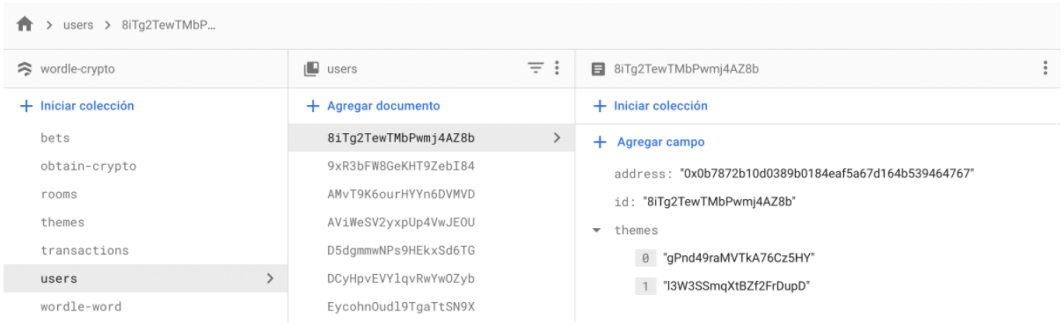

Figura 12: Captura del model *User* a Firebase

El primer model que es va implementar va ser la col·lecció de "*users*", ja que el primer que es va implementar va ser la identificació d'usuari amb MetaMask. El model "*user*"s'encarrega de guardar la informació important sobre l'usuari. Aquesta informació és l'adreça del cartera digital i la llista de temes que posseeix en aquest moment.

Com es pot observar no és necessari emmagatzemar cap mena de contrasenya de l'usuari, ja que la identificació es fa mitjançant MetaMask, llavors només és necessari obtenir la seva adreça per recuperar la seva informació dins l'aplicació Web3.

La llista de temes conté els identificadors únics dels temes estètics que posseeix l'usuari, de manera que, quan està dins una partida de Wordle, pot canviar el tema, els colors, de la interfície gràfica del joc. A mesura que l'usuari compri temes a la botiga de la pàgina web, s'afegiran els seus identificadors a aquesta llista.

4.3.3 La paraula del dia

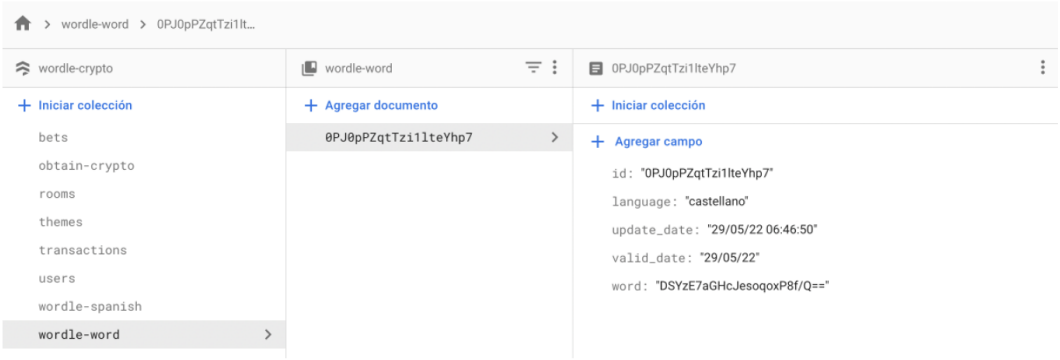

Figura 13: Captura del model *Wordle Word* a Firebase

El següent pas és un model per definir la paraula del dia: *wordle-word*. Aquest model guarda la paraula del dia que es fa servir en el mode de joc principal de l'aplicació.

La paraula s'agafa d'una llista de paraules de cinc de lletres (com exigeix el Wordle original) i comprova que aquesta paraula existeix en una API d'un diccionari en línia. Això es fa perquè no totes les paraules de la llista estan ben escrites, perquè o bé tenen accent o no tenen accent, però ho haurien de tenir; llavors, d'aquesta manera es pot assegurar que la paraula escollida està ortogràficament correcte. En el cas que la paraula no sigui correcte, es torna a agafar altra paraula aleatòria de la llista i es torna a comprovar. Així successivament fins que es troba una de vàlida (normalment triga com a molt 4 intents).

La llista d'on s'agafa la paraula depèn del valor de l'atribut *language* del model (de moment, només està implementada l'opció en castellà, però des d'un principi es volia implementar més idiomes). Un cop s'ha comprovat que la paraula que s'ha agafat aleatòriament de la llista de paraules és correcta, s'encripta la paraula i s'emmagatzema en l'atribut *word*. Cal destacar que abans de guardar la paraula encriptada s'ha de posar en format UTF-8 perquè si no Firebase no deixa guardar-la.

Ja s'ha generat i guardat aquesta paraula, però encara és necessari actualitzar-la diàriament. És per aquest motiu que és necessari un nou camp en aquest model per complir el seu objectiu: *valid_date*. Aquest guarda la data del dia que aquesta paraula és vàlida, de manera que, quan es faci una petició per obtenir la paraula del dia i el *backend* detecti que la paraula té aquest camp desactualitzat (amb una data menor que el dia en el qual es fa la petició), es tornarà a generar una nova paraula just com s'ha explicat abans.

Finalment, hi ha un atribut *update_date* que guarda la data exacta (amb hores, minuts i segons) quan s'ha actualitzat per últim cop la paraula.

4.3.4 Abonament

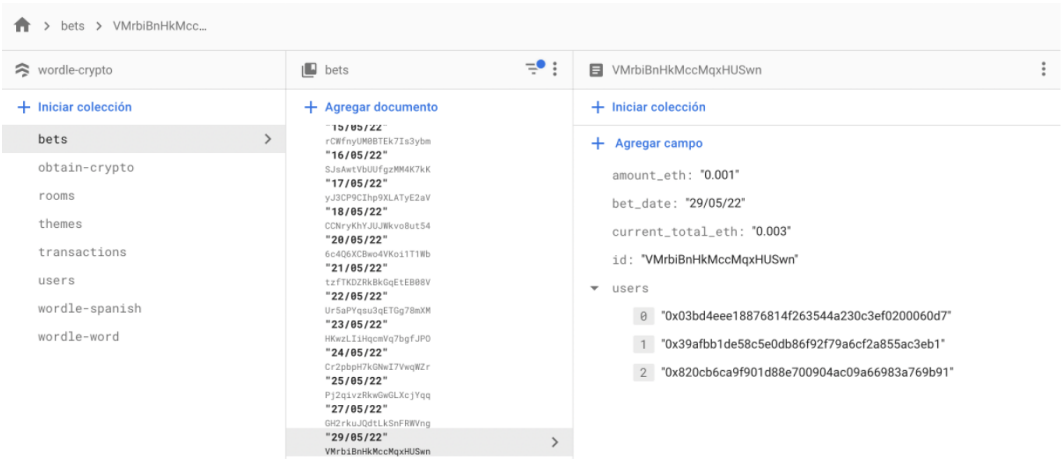

Figura 14: Captura del model *Bet* a Firebase

Com s'ha mencionat a la motivació del projecte, un dels reptes d'aquest projecte és la modificació del Wordle convencional implementant-li un sistema de recompenses fent servir ethers. Aquesta és la finalitat del model "*bet*" que s'encarrega de gestionar els abonaments dels usuaris: guarda els usuaris que han pagat per jugar la partida, la quantitat d'ether abonada per cadascú i la col·lecta d'ether total.

L'usuari, a través del *frontend*, pagarà la quantitat d'ether definida en el "*bet*" (corresponent a l'atribut *amount_eth*), s'afegirà la seva adreça a la llista d'adreces dels usuaris que participen en aquest dia i s'actualitzarà la col·lecta d'ether total (*current_total_eth*).

Si encara no hi ha un "*bet*" d'aquest dia; és a dir, no hi ha cap "*bet*" amb un *bet_date* igual al de la data del dia que l'usuari va pagar per jugar al Wordle de l'endemà, es crea automàticament un de nou de manera que cada dia hi ha un únic "*bet*" on fer l'abonament.

La col·lecta total, *current_total_eth*, es calcula multiplicant la quantitat d'ether (*amount_eth*) a pagar pel nombre d'usuaris que hi ha, *len(users)*. La quantitat d'ether que definida en cada bet (*amount_eth*) està definida en una constant del codi del *frontend*; és a dir, no és un valor que tingui per defecte totes les bets quan es creen des del *backend*, de manera que es pot canviar fàcilment en qualsevol moment i no seria necessari actualitzar el codi en el cas que, per exemple, en un futur fossin els mateixos usuaris qui decideixin la quantitat d'ether a apostar.

L'atribut *bet_date*, a més de tenir la funció de saber si és o no necessari crear un nou "*bet*", és el valor que relaciona aquest model amb la paraula del dia, amb la

valid_date del model *wordle-word*, de manera que cada *bet* té associat un *wordle-word*.

4.3.5 L'estat de la partida

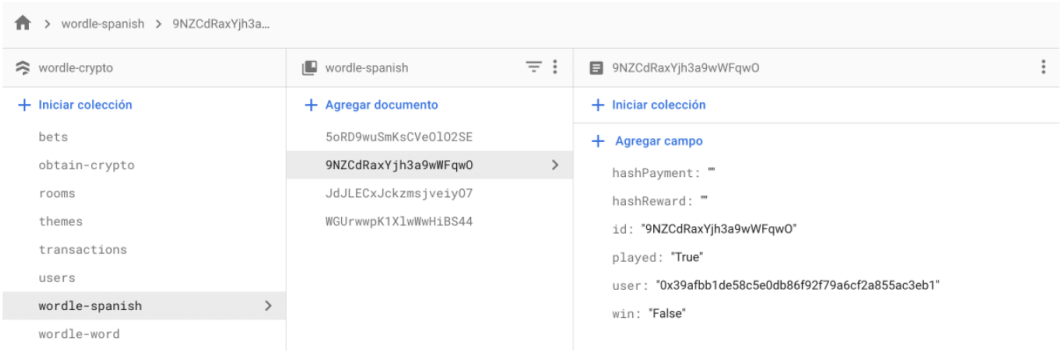

Figura 15: Captura del model *Wordle Spanish* a Firebase

Ara que ja hi ha un model que s'encarrega de la gestió de la paraula a esbrinar del dia i altre per la gestió econòmica, fa falta un model per gestionar cada partida del mode de joc principal i actualitzar el seu estat. Aquesta és la finalitat del model *wordle-spanish* encarregat de la gestió de l'estat de cada partida Wordle del dia de cadascun dels jugadors (els quals es troben definits a l'atribut *users* del model *bet* d'aquest dia).

Aquest model guarda si l'usuari ha acabat de jugar la partida (*played*), si ha guanyat (*win*) i el *hash* de la transacció en la *testnet* de Rinkeby del seu pagament (*hashPayment*) i del de la seva recompensa (*hashReward*), si és que n'hi ha.

L'objectiu de guardar el *hash* de la transacció de la seva recompensa és per poder comprovar l'estat de la transacció en el cas que l'usuari no rebi l'ether que li correspon un cop ha guanyat la partida, a pesar que és una situació que mai ha passat, ja que les transaccions es fan, es minen, gairebé al moment.

En funció de l'atribut *played*, el *frontend* carrega el Wordle perquè l'usuari el jugui (en el cas que encara no l'hagi jugat) o li carrega la interfície per fer l'abonament del següent Wordle (on es fa servir l'anterior model explicat, *bet*, del següent dia a la data d'avui).

La idea principal era que hi hagués un model per la gestió de la partida per cada idioma, que hi hagués també un *wordle-english* i un *wordle-catalan*.

En el *backend* del model *wordle-word*, està imposat que quan es canvia de paraula del dia; és a dir, *valid_date* és menor que la data actual, el model *wordle-spanish* (en el cas que l'atribut *language* de *wordle-word* sigui "*castellano*", que com no s'ha arribat a implementar més idiomes és l'únic cas de moment) s'esborra sencer perquè significa que ha començat una nova partida i totes les dades anteriors ja no són necessàries.

4.3.6 Transaccions

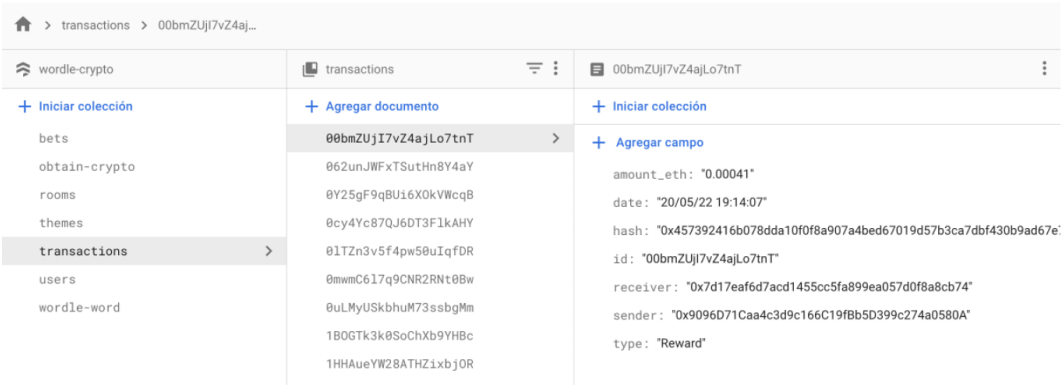

Figura 16: Captura del model *Transaction* a Firebase

Totes les transaccions que es fan a l'aplicació Web3 es poden consular a la *testnet* de Rinkeby, fent servir, per exemple, una web com Etherscan. I per tenir un registre de les transaccions que es fan a la web i d'aquesta manera no dependre d'una web externa, s'ha implementat un model propi transaction per guardar la informació important sobre la transacció en Rinkeby, a més d'informació útil pel seu filtratge.

Els atributs crucials referents a la transacció en si, són:

1. *amount_eth*. Amb una funció semblant a la vista en models anteriors, s'encarrega de definir la quantitat d'ether que s'ha enviat a la transacció.
2. *date*. La data exacta quan s'ha realitzat la transacció.
3. *hash*. El *hash* corresponent a la transacció dins la *testnet* de Rinkeby, amb aquest hash pots comprovar l'estat de la transacció dins la testnet fent servir una pàgina com Etherscan.
4. *receiver*. L'adreça de la cartera del receptor, a qui va destinat la transacció.
5. *sender*. Conté l'adreça de la cartera d'on precedeix l'ether enviat en la transacció.

Un camp rellevant i propi d'aquest model és l'atribut *type* que conté el tipus de transacció que és dins Wordle Crypto, d'aquesta manera podem distingir la finalitat que té aquesta transacció dins l'aplicació.

Hi ha definits tres tipus de transacció:

- *Payment*. Quan un client fa un abonament per jugar una partida de Wordle Crypto, del multijugador o adquireix un tema estètic a la botiga de temes.
- *Reward*. Correspon a quan es fa una transacció del servidor al client perquè aquest ha guanyat una partida de Wordle i ha de rebre l'ether que li correspon.

- *Obtain Crypto.* Quan un usuari reclama l'ether inicial que ofereix la web als usuaris primerencs per poder jugar les primeres partides sense haver de demanar ethers de Rinkeby a pàgines externes.

4.3.7 Sala multijugador

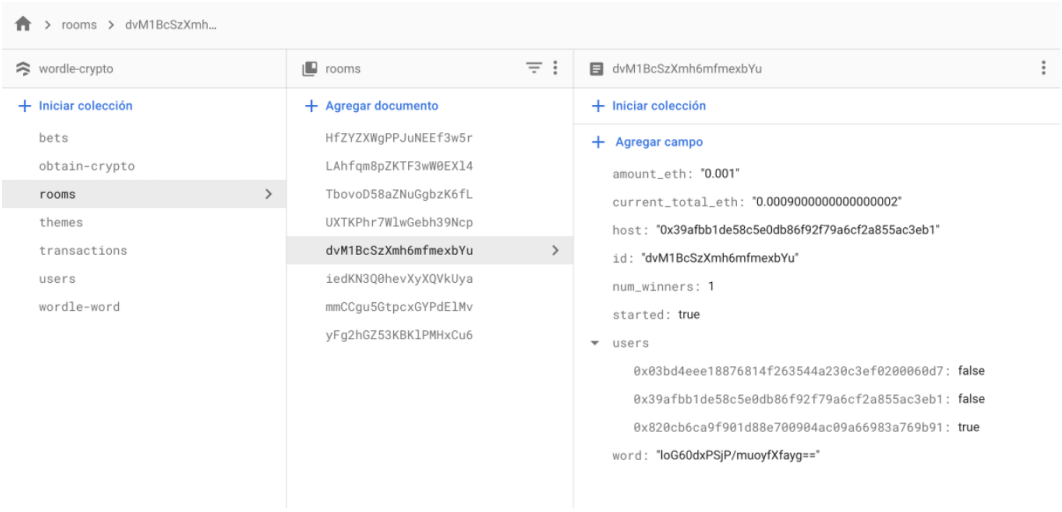

Figura 17: Captura del model *Room* a Firebase

Pel mode multijugador s'ha fet un únic model en comptes de tres models diferents com al mode principal Wordle Crypto. Aquest model és *room*, aquest model emmagatzema la gestió econòmica de l'ether (com s'encarrega el model *bet*), de la gestió de la partida (com s'ocupa *wordle-spanish*) i gestió de la paraula a esbrinar (com es fa a *wordle-word*).

El concepte del mode jugador és crear una sala a la qual accedeixen la resta de jugadors a través de l'identificador únic (correspon al camp *id*) i aquest juguen a un Wordle comú de manera similar al mode principal però sense haver d'esperar un dia sencer. És per aquest motiu que la paraula encriptada s'emmagatzema dins el model de *room* (a l'atribut *word*), ja que només es fa servir en aquesta partida en concret. La paraula es genera i s'encripta igual que al model de *wordle-word*.

Una diferència apreciable respecte als anteriors models mencionats l'atribut *user* que, a més de tindre la mateixa funció que a *bet* de guardar els jugadors que poden participar en la partida, també guarda si han acabat de jugar al Wordle o no, en un format de diccionari on la clau és l'adreça de l'usuari en qüestió i el valor és si ha jugat la partida o no. Amb aquest atribut es controla l'estat de l'usuari de manera similar a com es feia a *wordle-spanish*.

Com es pot observar, com té també la col·lecció de *bets*, tenim un atribut *amount_eth*, amb la quantitat d'ether que ha d'apostar cada usuari, i un *current_total_eth*, amb la col·lecta total d'ether (com es pot observar a la captura, aquesta ha disminuït un 70%, ja que un jugador ja ha jugat, ha guanyat la partida,

i, per tant, ha rebut la seva recompensa).

Hi ha dos atributs necessaris per a la gestió dins la sala d'espera (quan els jugadors estan esperants que els jugadors que falten s'uneixin a la sala):

- *host*. Guarda l'adreça de l'amfitrió, de l'usuari que ha creat la sala, ja que és l'únic jugador que pot començar la partida.
- *started*. S'encarrega de definir si la partida de Wordle ha començat o no.

D'aquesta manera quan l'amfitrió, o *host*, comenci la partida a la resta de jugadors que estan esperant a la sala d'espera els hi carregarà el Wordle per jugar.

Finalment, hi ha un atribut que guarda nombre de guanyadors (*n_winners*) utilitzat per mostrar en quina posició has quedat quan has guanyat el Wordle.

4.3.8 Obtain Crypto

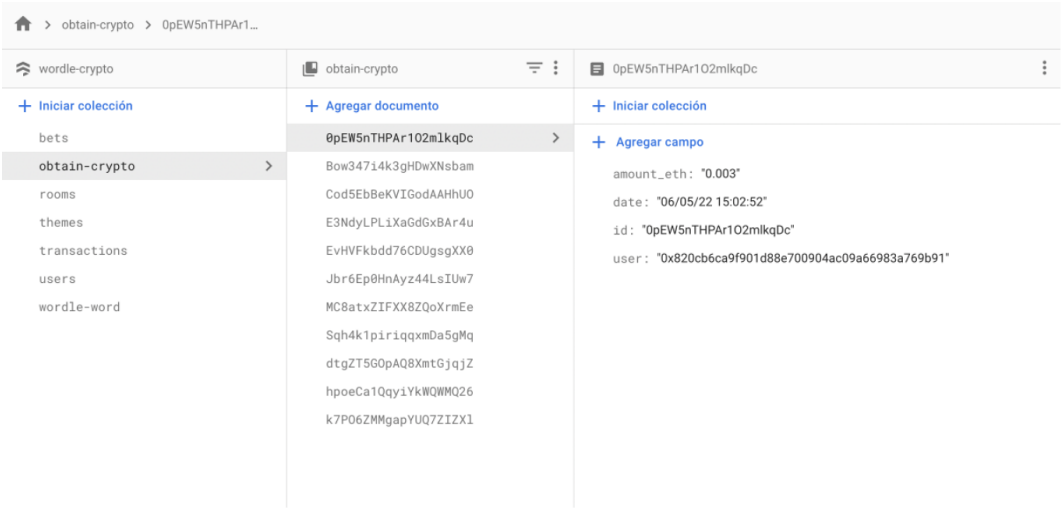

Figura 18: Captura del model *Obtain Crypto* a Firebase

Altre apartat de la web on és necessari definir un model per la seva gestió és "*Obtain Crypto*". En aquest apartat es regalen ethers als usuaris primerencs, concretament es regalen 0.003 ETH, perquè aquests usuaris puguin gaudir de les funcionalitats de Wordle Crypto sense haver de sol·licitar els ethers en una pàgina externa. Però només es regala un únic cop, de manera que és necessari porta un registre dels usuaris que han rebut aquest ether gratuït. És per aquest motiu que sorgeix la necessitat d'un nou model: *obtain-crypto*.

Aquest model guarda la quantitat d'ether regelada en l'atribut *amount_eth* (el qual és molt familiar, ja que té una funcionalitat similar en altres models), guarda la data exacta quan s'ha sol·licitat l'ether en el camp *date* i l'adreça de l'usuari que l'ha sol·licitat en l'atribut *user*. D'aquesta manera, al *frontend*, un cop es detecti

que l'usuari està registrat en aquesta col·lecció ja no podrà tornar a sol·licitar aquest ether gratuït.

4.3.9 Temes

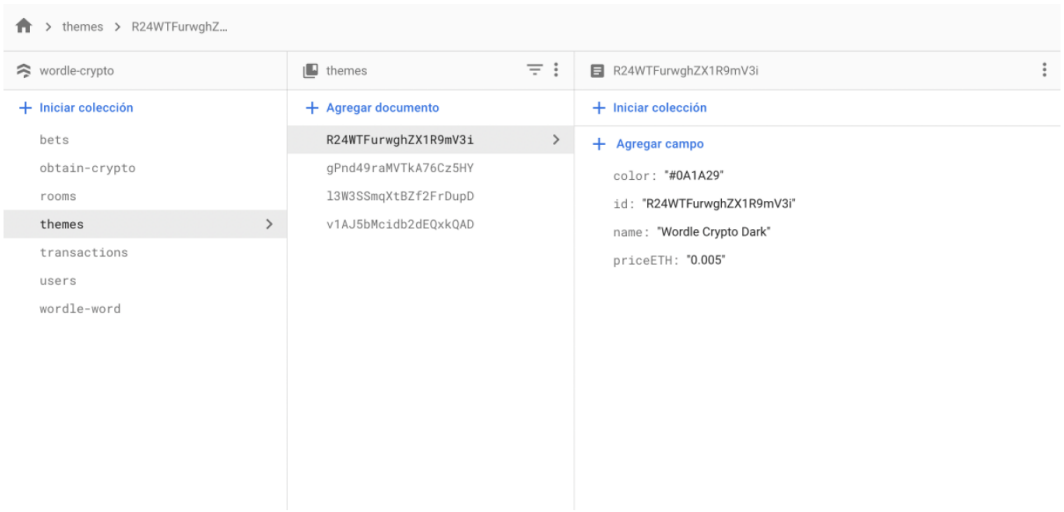

Figura 19: Captura del model *Theme* a Firebase

Finalment, tenim un últim model per gestionar la informació referent als temes estètics: *theme*. Aquesta informació es fa servir tant a la botiga de Wordle Crypto, com a les partides de Wordle, ja que a partir de l'identificador únic de cada tema (*id*), es recupera els valors necessaris per establir el tema a la interfície gràfica del Wordle, que es tracta bàsicament del camp *color*, que conté el codi RGB del color que s'estableix com color de fons al Wordle.

No són necessaris més camps, ja que al *frontend* es disposa de funcions que detecten si el color de fons és més fosc o clar i en funció d'aquest valor, estableix la resta de colors de la resta d'elements DOM de la pàgina de joc del Wordle. Per exemple, si el color de fons és negre (el codi RGB serà "121213"), al frontend es detecta això i a la resta d'elements, com pot ser el color del text, es posen de color blanc.

Els camps *priceETH* i *name*, són atributs necessaris per a la botiga. L'atribut *name*, com el seu nom indica, conté el nom comercial, més familiar per qualsevol usuari, i l'atribut *priceETH* guarda el preu en ether a pagar pel tema a la botiga perquè passi a ser de la seva propietat.

Un cop l'usuari compra un tema, s'afegirà el seu identificador únic a l'atribut *themes* del model *user* d'aquest usuari. D'aquesta manera li apareixerà aquest nou tema com una opció a l'apartat de configuració dins d'una partida Wordle.

5 Implementació

5.1 Pagament amb MetaMask

S'ha implementat en una única funció, anomenada *startPayment()*, la qual permet realitzar qualsevol transacció a l'adreça especificada per paràmetre a la funció, de la quantitat d'ether especificat, també per paràmetre. Es fa servir la llibreria *Ethers* de JavaScript per comunicar-se amb MetaMask i la *blockchain*.

El pseudocodi de la funció en qüestió és el següent:

```
1: try
2:     // Comprova que està MetaMask instal·lat
3:     if not getProveidorWeb3(window) then
4:
5:         // Si no, se suggereix la seva instal·lació;
6:         throw Error("Instal·la MetaMask");
7:
8:     // S'aconsegueix el proveïdor que té instal·lat l'usuari: MetaMask
9:     proveidor = getProveidorWeb3(window)
10:
11:    // S'obté el signant d'aquest proveïdor
12:    signant = proveidor.getSignant()
13:
14:    // Es firma la transacció i s'envia amb MetaMask
15:    tx = signant.enviaTransaccio(adreça, quantitat_ether)
16:
17:    // Es guarden els camps importants de la transacció a la base de dades
18:    postTransaccio(tx)
19:
20: catch error
21:     print error
```

El primer que fa l'aplicació és comprovar si té instal·lat algun wallet, en aquest cas MetaMask. Es mira l'objecte *window*, un objecte comú a tots els navegadors web. Es mira l'atribut *ethereum* i, si no te un valor nul, significa que hi ha un *wallet* instal·lat. Es fa servir aquest atribut per la comunicació amb MetaMask, de manera que l'extensió respon a les sol·licituds a aquest objecte.

Si està instal·lat, es sol·licita un proveïdor i contesta MetaMask. A partir del proveïdor s'obté el signant.

Es comprova que l'adreça pública destí, la passada per paràmetre, és una adreça d'Ethereum i es pot enviar ether a aquesta.

Es fa una petició de firma, per confirmar la transacció, al signant, i s'espera fins que el signant, l'usuari, respongui. És en aquest pas que a la interfície de l'usuari li surt la finestra emergent de MetaMask demanant una confirmació de la transacció.

Quan l'usuari confirma la transacció, es realitza a la *testnet* de Rinkeby, es mostra un missatge de confirmació per pantalla i es guarda a la base de dades.

També aquesta funció és l'encarregada de fer el control d'errors; és a dir, de mostrar agafar els errors que es puguin produir durant la transacció i de canviar el missatge perquè sigui més amigable i comprensible per l'usuari. Aquests errors són no tindre prou ether per dur a terme la transacció, que no tingui MetaMask instal·lat o que l'usuari rebutgi la transacció quan se li demana la confirmació per signar-la.

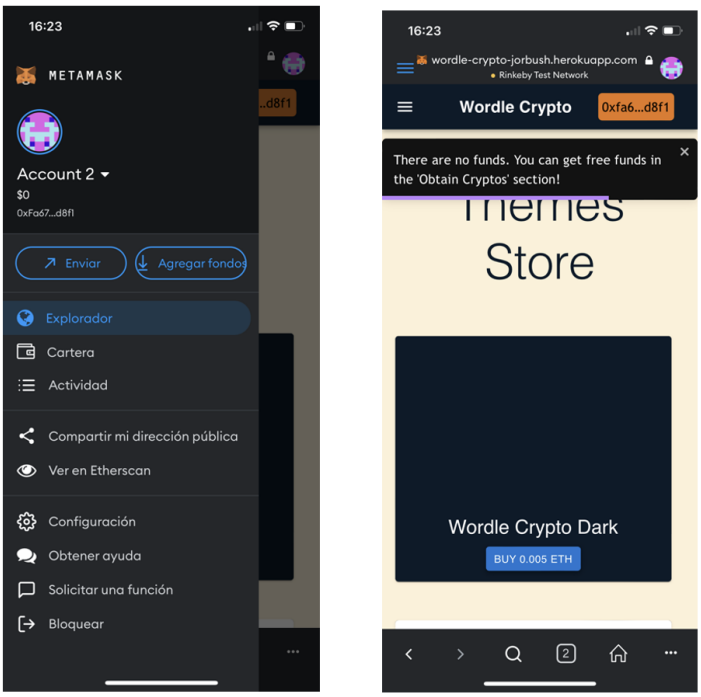

Figura 20: Captura del control d'errors a Wordle Crypto

Aquesta funció s'utilitza tant l'abonament inicial per participar en els dos modes de joc de Wordle Crypto, com a la botiga de temes estètics.

5.2 Recompensa amb Infura

Per fer transaccions a la inversa, del servidor al client, es fa servir altre proveïdor, i, per tant, altra implementació. S'ha implementat en una funció anomenada *receiveRewardWeb3()* dins dels components Wordle dels dos modes de joc. També està definida com a *startRewardWeb3()* per l'apartat *Obtain Crypto*, ja que fa servir una adreça diferent.

El motiu pel qual no hi ha una única funció com al cas anterior és que es criden des d'una classe *Component* de React (els Wordles són components no funcions) i no funcionen correctament les funcions asíncrones si no es defineixen dins el mateix component. A pesar d'això, totes tenen un comportament similar i la mateixa funció.

Aquesta funció és l'encarregada d'enviar ether a l'adreça pública passada per paràmetre, de la quantitat d'ether passat també per paràmetre. Fa servir les llibreries "Web3" i "EthereumJS-TX".

El pseudocodi és el següent:

```
1: try
2:     // S'ajusta la quantitat d'ether
3:     ether = regula(ether)
4:
5:     // S'aconsegueix el proveïdor Infura a partir de l'ennlaç de l'API
6:     proveidor = getProveidorWeb3(EndpointInfuraRinkeby)
7:
8:     // S'obté el nombre de transaccions del compte de l'aplicació
9:     nombreTx = proveidor.getTransactionCount(adreçaWordle)
10:
11:    // Es crea la transacció
12:    tx = {adreçaUsuari, ether, nombreTx, preuGas, limitGas}
13:
14:    // Es firma la transacció amb la clau privada del compte de l'aplicació
15:    tx.signa(WordleClauPrivada)
16:
17:    // S'envia la transacció firmada amb Infura
18:    hash = proveidor.sendSignedTransaction(tx)
19:
20:    // Es guarden els camps importants de la transacció a la base de dades
21:    postTransaccio(tx, hash)
22:
23: catch error
24:     print error
```

Per la recompensa del Wordle, es regula el valor d'ether a trametre. De manera que, si l'ether té més de 8 xifres, s'aproxima el valor la transacció. També es comprova que aquest valor no és menor que 0.00001, ja que hi ha un mínim d'ether requerit per enviar-lo a altre compte.

Un cop s'ha comprovat que el valor d'ether és correcte per trametre-ho, es realitza la transacció al compte de l'usuari, l'adreça passada per paràmetre. Per aconseguir-ho és configura el proveïdor Web3 com Infura (fent servir l'endpoint que proporciona la seva API). A més, és necessari configurar-lo perquè utilitzi la xarxa de Rinkeby.

Altre requisit és obtenir la clau privada del compte des d'on es farà la transacció, ja que d'aquesta manera se li està donant el control per manipular el contingut, l'ether, d'aquest compte, del compte de Wordle Crypto. Amb aquesta clau privada, Infura podrà enviar ether del compte d'Ethereum del Wordle Crypto a l'adreça destí, l'adreça de l'usuari. Aquesta clau s'ha assolit prèviament fent servir la interfície de MetaMask.

Per dur a terme la transacció es fa servir una funció, *getTransactionCount()*,

que retorna el nombre de transaccions que porta el compte, la qual cosa és molt útil a l'hora de definir el *nonce* que ha de tenir. A diferència del proveïdor anterior és necessari configurar els camps de la transacció abans de realitzar-la. Els valors d'aquests camps són els següents:

```
const transaction_Object = {
   to: addr,
   gasPrice: w3.utils.toHex(20000000000),
   gasLimit: w3.utils.toHex(21000),
   nonce: w3.utils.toHex(transactionCount),
   value: w3.utils.toHex(w3.utils.toWei(ether, "ether"))
};
```

Figura 21: Captura del valors dels camps d'una transacció

- El preu del gas (*gasPrice*) són 20000000000 WEI. El valor d'aquest paràmetre és essencial a l'hora de fer la transacció, ja que si és baix mai arriba a minar la transacció i, per tant, no s'arribarà a executar. Per aconseguir aquest valor es va anar fent proves incrementant el preu del gas fins que es fan totes les transaccions i es minen en pocs segons.
- El límit de gas (*gasLimit*) té un valor de 21000 de gas, el valor recomanat perquè es dugui a terme la transacció.
- El camp *nonce* té el valor d'una variable que proporciona la mateixa funció de la llibreria.
- El valor d'ether (*value*) es converteix a WEI per enviar-ho.
- El paràmetre *to* fa referència a l'adreça pública on s'envia l'ether.

Aquest objecte transacció es firma amb la clau privada del compte des d'on s'envia, el compte d'Ethereum de l'aplicació. Mitjançant el proveïdor Infura, es tramet firmat a la *blockchain* de Rinkeby, on es mina i es computa.

Un cop es processa s'obté el *hash* de la transacció dins de Rinkeby i es guarda una còpia dels camps importants a la base de dades de l'aplicació. També es mostra un missatge informant de l'estat de la transacció.

Com a la funció de l'anterior apartat, hi ha control d'errors. Per exemple, avisa si la transacció no s'ha pogut realitzar perquè hi ha moltes transaccions pendents d'aquest compte.

A més, Infura proporciona una interfície, a través de la seva web, on es pot observar les funcions executades pel proveïdor i el nombre de vegades que s'han executat. A la següent figura es pot veure les estadístiques del mes de maig.

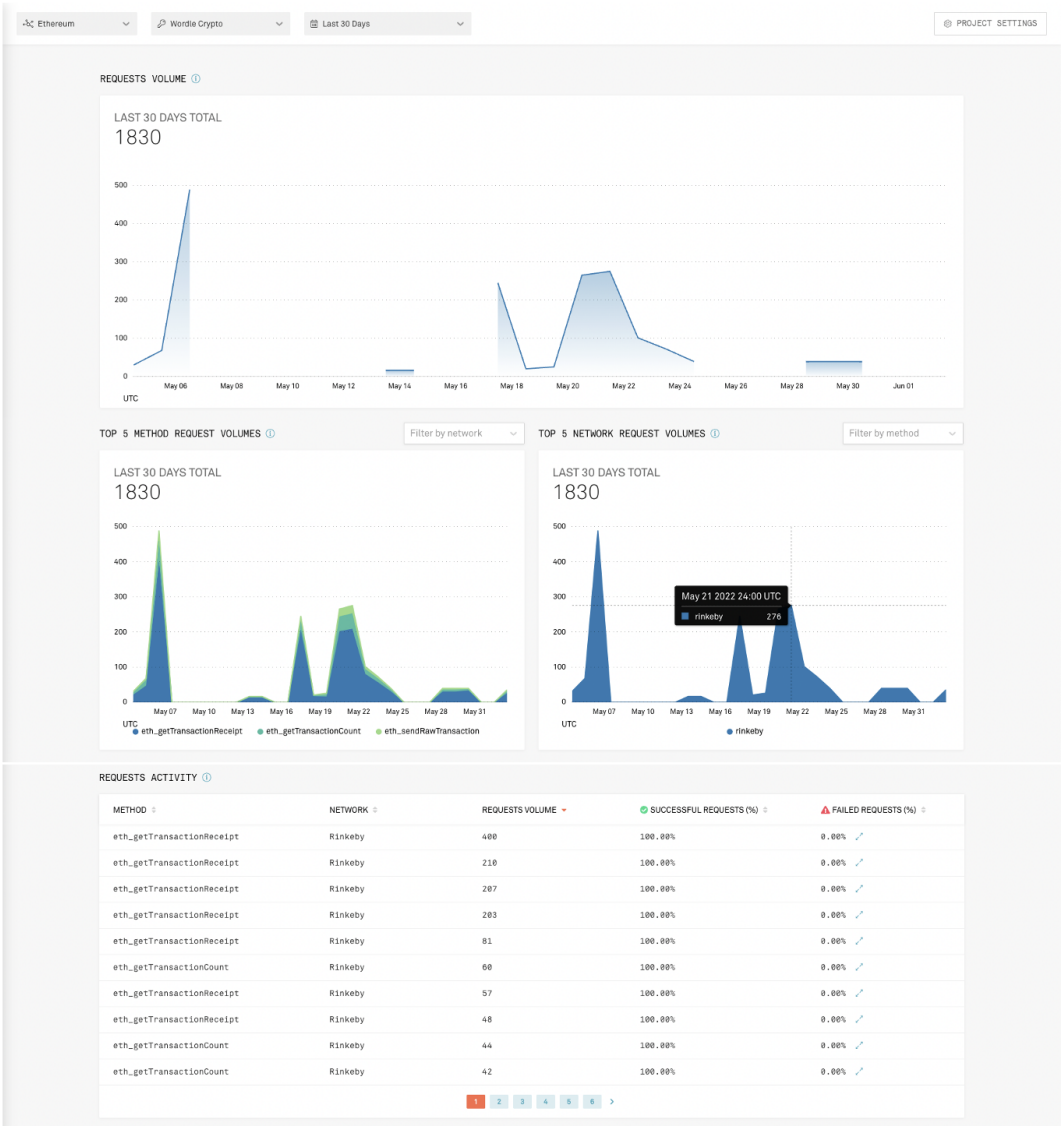

Figura 22: Captura de la interfície web d'Infura

5.3 La lògica i les funcionalitats de la web

En aquest apartat s'explicarà la lògica i les funcionalitats més importants de cada apartat de la web. L'aplicació Web3 la podem dividir en els següents apartats: *Login amb MetaMask*, *HowToPlay* i *About*, *Obtain Crypto*, *Send Ether*, *Wordle Crypto*, "Multijugador" i *Themes Store*.

5.3.1 Login amb MetaMask

El primer cop que s'accedeix a l'enllaç de Wordle Crypto (*https://wordle-crypto-jorbush.herokuapp.com/*), s'ha d'identificar l'usuari utilitzant MetaMask.

En el cas que no estigui instal·lat l'extensió de MetaMask al navegador, es pot

comprovar consultat l'estat de l'objecte *window.ethereum*, sortirà una alerta amb un missatge suggerint descarregar l'extensió (en el cas de dispositius mòbils intel·ligents, obre la botiga del software amb l'opció de descarregar l'APP de MetaMask). Quan l'usuari ha descarregat l'extensió i ha configurat un compte de MetaMask, realitzarà el flux d'accions següent:

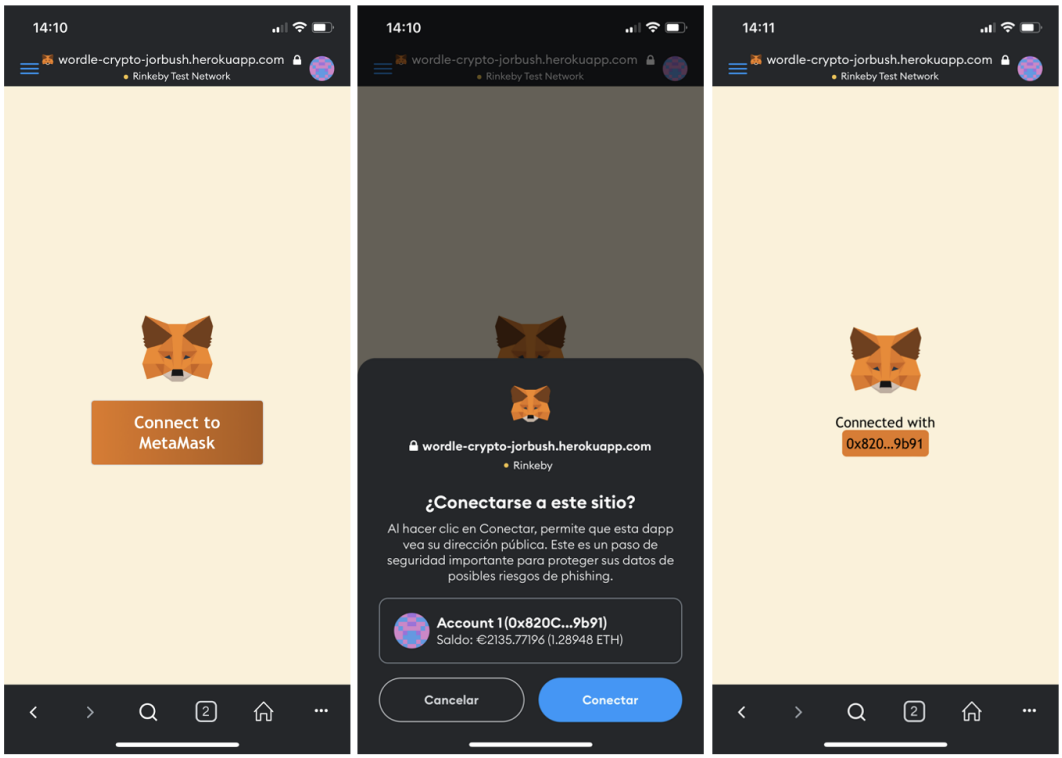

Figura 23: Captures de l'inici de sessió de Wordle Crypto

- L'usuari pren el botó de "*Connect to MetaMask*", MetaMask sol·licitarà permisos a l'usuari per fer servir l'adreça de la seva cartera amb la web de Wordle Crypto.
- L'usuari accepta la petició.
- Es donarà d'alta l'usuari a la base de dades (es farà una petició POST) en el cas que no ho estigui, si no només es farà un GET de la seva informació.

Si en algun moment dins l'aplicació Web3 l'usuari desconnecta el seu compte de MetaMask, l'aplicació redirigirà l'usuari altre cop a aquesta pantalla, on haurà de tornar a autoritzar a l'aplicació Web3, Wordle Crypto, per fer servir el seu compte de MetaMask.

Això ho comprova concretament el component header (la capçalera), que és present en totes les pàgines de l'aplicació, consultant l'estat de l'objecte *window.ethereum .accounts* present a qualsevol navegador.

5.3.2 Wordle Crypto

Aquesta és la secció principal de l'aplicació Web3. La finalitat d'aquesta secció, anomenada com la mateixa web, *Wordle Crypto*, és que l'usuari guanyi ethers amb les seves habilitats jugant a Wordle.

Aquest mode de joc fa servir el sistema de recompenses definit amb anterioritat, on l'usuari fa un abonament inicial, de 0.001 ETH, i quan guanyi rebrà el 70% de l'ether que hi hagi col·lectat en el moment que guanya; és a dir, si ha guanyat una persona abans serà el 70% del 30% restant. Per aconseguir aquest sistema de recompenses, els passos ha seguir per jugar són les següents:

- L'usuari ha de fer l'abonament inicial necessari per poder participar en el Wordle de l'endemà.
- L'endemà, quan entri en aquesta secció, podrà jugar directament al Wordle que ha abonat.

Quan l'usuari entra a l'apartat de Wordle Crypto i no havia apostat al dia anterior o si ja ha jugat al Wordle d'avui, li apareixerà aquesta interfície:

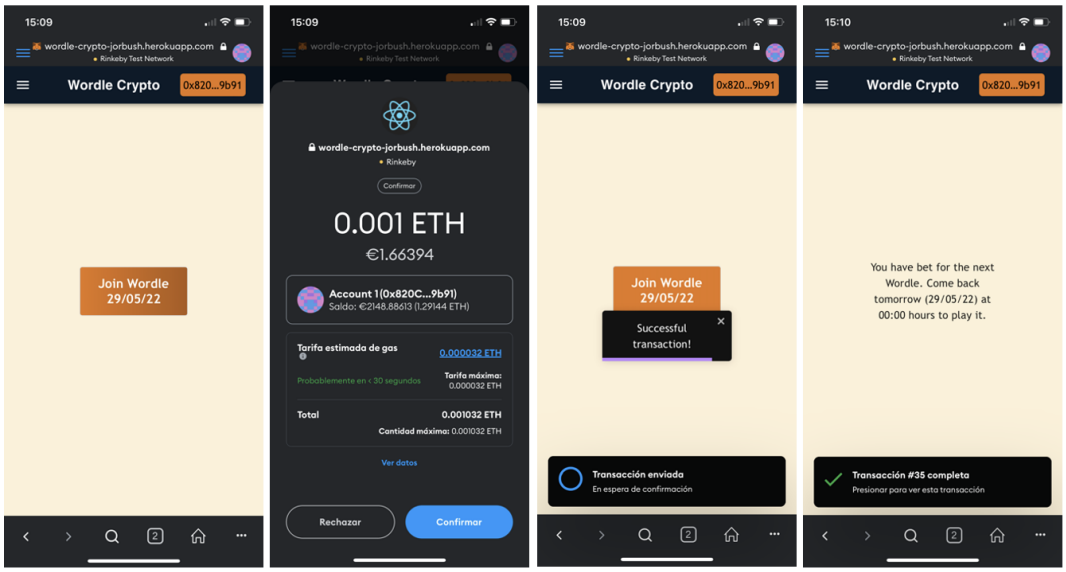

Figura 24: Captures de l'apartat *Wordle Crypto* fent l'abonament

El primer que fa l'aplicació web en aquest apartat és mirar si hi ha un *bet* creat pel dia d'avui i si l'usuari hi és dins. Si és el cas carregarà directament el Wordle, si no carregarà aquesta interfície.

Les accions que es realitzen en carregae aquest *layout* són les següents:

1. Comprova si s'ha creat el *bet* de l'endemà i, si no és el cas, es crea.

2. Mira si l'usuari ha pagat per jugar al Wordle de l'endemà (mirant el *bet*). Si no és el cas, apareix un botó amb el text "*Join Wordle*" més la data del Wordle al qual aposta (l'endemà).

3. Si l'usuari prem el botó per participar en la següent partida de Wordle Crypto, li sortirà una sol·licitud de confirmació de transacció per fer l'abonament de 0.001 ETH al compte de Wordle Crypto.

4. Un cop s'ha fet l'abonament, li sortirà el missatge que pot jugar l'endemà a partir de les dotze de la nit.

Cal destacar que si l'usuari no té prou fons d'ethers, li sortirà un missatge d'error controlat on l'informa que no té prou fons d'ether i li suggereix reclamar els ethers gratuïts d'*Obtain Crypto*.

Quan l'usuari entra a l'apartat Wordle Crypto, havia fet l'abonament pel Wordle d'avui i encara no l'ha jugat (amb això s'entén que no l'ha acabat en cap moment, sigui guanyar o perdre), carregarà el joc de Wordle.

La interfície del joc és la següent:

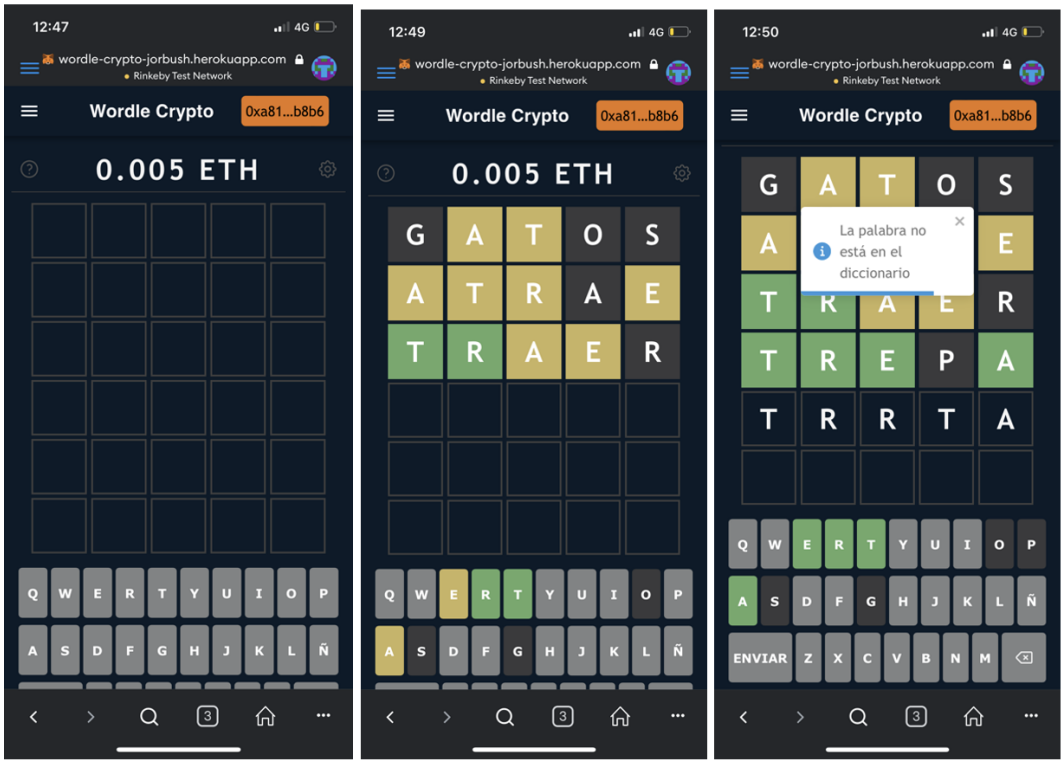

Figura 25: Captures de l'apartat *Wordle Crypto* jugant al Wordle

El joc de Wordle consisteix a esbrinar la paraula del dia (definida al model *wordle-word* del *backend*) en sis intents o menys. Es pot escriure tant amb el teclat físic com el teclat dibuixat a la web.

Quan s'escriu una paraula sencera de cinc lletres i es polsa la tecla "Enviar", o es pren l'enter, aquesta paraula es comprova en una API de diccionari des del backend si existeix i, si és el cas (si no ho és, t'ho notifica), es compara amb la paraula a endevinar

Se li dona *feedback* a l'usuari de la següent forma:

- Si la casella de la lletra es pinta de color verd, significa que aquesta lletra està en aquesta posició i és correcte.
- Si es pinta de color groc, vol dir que la paraula conté aquesta lletra però no en aquesta posició.
- Si la casella es pinta de color gris, significa que la paraula a esbrinar, el Wordle, no conté aquesta lletra.

Aquestes normes es poden consultar en qualsevol moment polsant la icona en forma d'interrogant situat a la cantonada superior esquerra de la interfície.

A la part superior cèntrica de la interfície de joc, es mostra la quantitat d'ether de la col·lecta que hi ha en aquest moment. Abans d'enviar la recompensa aquesta s'actualitza.

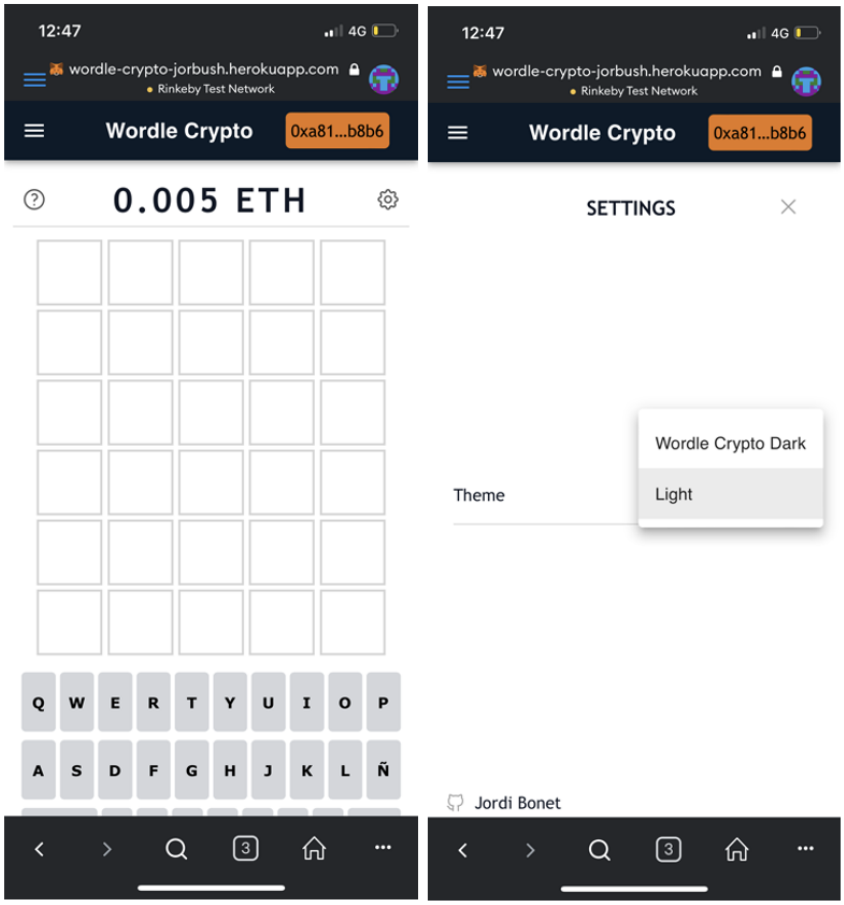

Figura 26: Captures de l'apartat *Wordle Crypto* canviant de tema

A la seva dreta, es troba la icona de configuració on podem canviar el tema estètic de la interfície del Wordle per un dels temes que ha adquirit l'usuari (els pot adquirir a la botiga de temes).

Cal destacar que en tot moment es guarda l'estat de la partida a les *cookies* de manera que, pel que sigui, l'usuari surt de la web, podrà continuar la partida exactament per on la deixada.

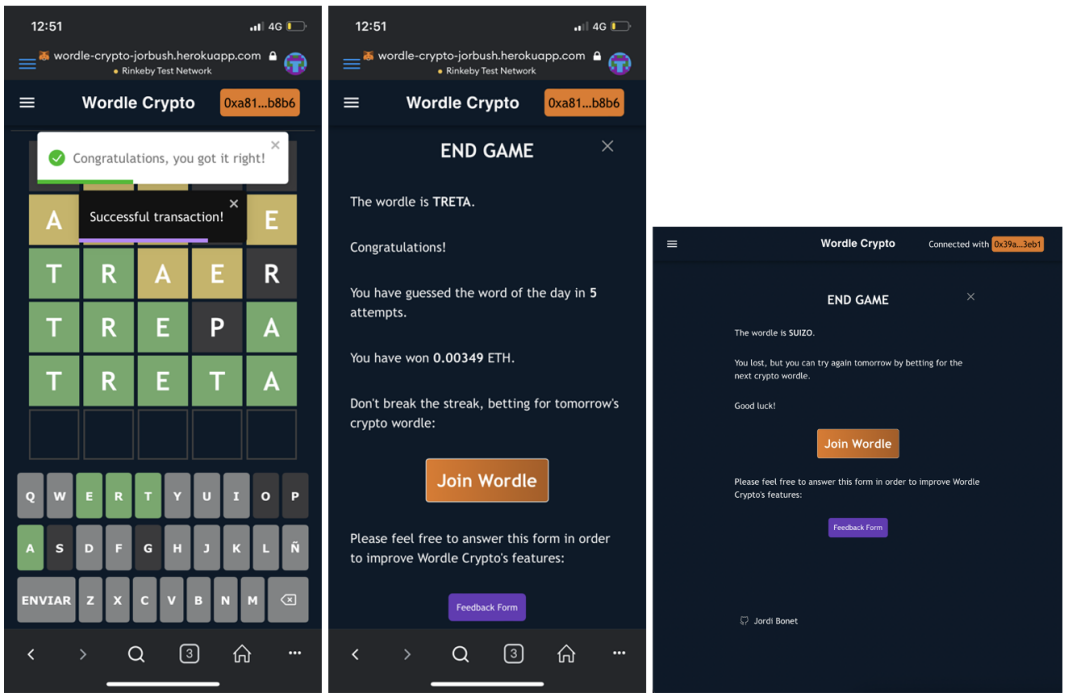

Figura 27: Captures de l'apartat *Wordle Crypto* acabant la partida Wordle

La partida pot acabar de dues formes:

- L'usuari guanya. Encerta la paraula, les cinc caselles de la fila actual es posen de color verd, i obté la recompensa. Es realitza la transacció de recompensa del compte de Wordle Crypto a l'adreça de l'usuari.
- L'usuari perd. Esgota els sis intents i no ha encertat totes les lletres.

Sis segons després de guanyar o perdre, apareix la pantalla de "fi de joc", o End Game, conté estadístiques sobre la partida que ha fet l'usuari. Depenent de si l'usuari guanya o perd la partida, li sortirà un missatge diferent d'aquesta pantalla.

Tant si ha guanyat o perdut, li diu la paraula del dia. Si ha guanyat, l'informa del nombre d'intents que has fet per esbrinar-la i la quantitat d'ether guanyat. A més, incita a l'usuari a jugar una altra partida, amb un botó que li porta a la fer l'abonament pel següent Wordle, i també ofereix l'usuari l'oportunitat de contestar un qüestionari per donar *feedback* sobre l'aplicació Web3.

Si l'usuari pren el botó de "*Join Wordle*"a la pantalla d'*End Game*, sortirà la interfície vista anteriorment per fer el pagament pel Wordle de l'endemà (realment es refresca la pàgina, no es fa una redirecció, ja que està definit que, si l'usuari ja ha jugat al Wordle d'avui, mostra la interfície per unir-se a la següent partida).

5.3.3 Multijugador

L'inconvenient de l'apartat principal, de Wordle Crypto, és que només es pot jugar un cop al dia i s'ha d'esperar l'endemà per jugar en una hora molt concreta (a les dotze de la nit), la qual cosa pot ser una dificultat per alguns usuaris. Per aquest motiu sorgeix la necessitat d'implementar un nou mode de joc, fent servir la mateixa tecnologia però amb un concepte diferent.

Aquest nou mode és el mode multijugador, en el que els jugadors poden desafiar a altres usuaris en qualsevol moment, sense aquesta limitació diària.

En aquesta modalitat hi han dues opcions:

- *Create Room*. Crea una sala multijugador. Genera una clau que es pot passar a altres jugadors.
- *Join Room*. Permet a l'usuari unir-se a una sala multijugador prèviament creada.

Els passos a seguir per crear una sala són els següents:

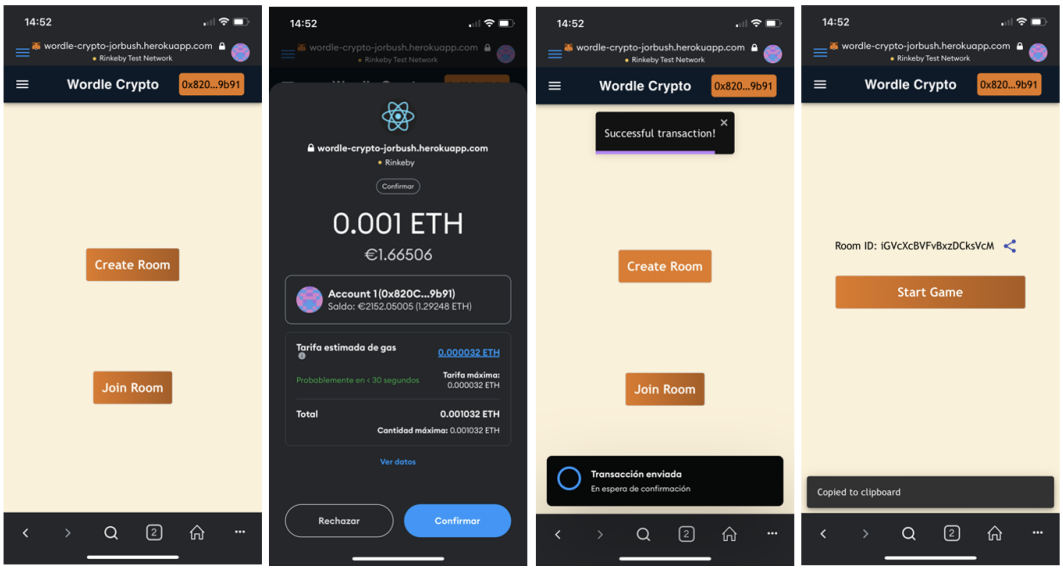

Figura 28: Captures de l'apartat *Multiplayer* creant una sala

- L'usuari paga 0.001 ETH per crear una sala.
- Confirma la transacció amb la cartera de MetaMask.
- Carrega la sala d'espera.

En aquesta sala d'espera l'usuari pot compartir l'identificador únic de la sala fent servir, per exemple, qualsevol xarxa social de confiança, pot copiar-lo al portaretalls prement el botó amb la icona de compartir, amb la resta de jugadors perquè s'uneixin amb l'opció de "*Join Room*" i poder jugar junts.

Un cop tots els usuaris estiguin preparats a la sala, l'amfitrió és l'únic jugador de la sala que pot començar el joc.

Llavors, tothom jugarà un Wordle comú: els jugadors han d'endevinar la mateixa paraula i el primer que l'endevini guanya una recompensa molt superior a la resta.

Cal destacar que un cop l'usuari comenci la partida, no es podrà unir cap altre jugador.

Un usuari es pot unir a una sala amb l'opció de "*Join Room*". En aquest component es realitzen les següents accions:

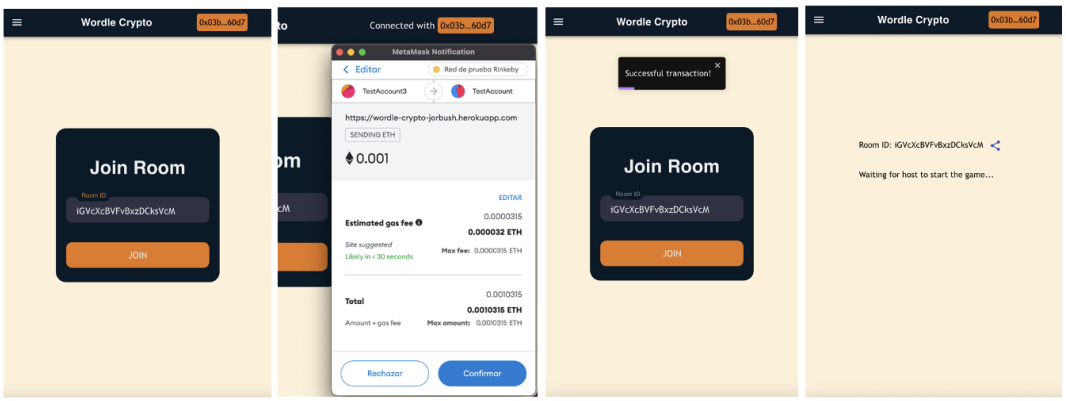

Figura 29: Captures de l'apartat *Multiplayer* unint-se a una sala

1. Carrega un formulari senzill per unir-se a la sala. En aquest, l'usuari ha d'escriure l'identificador únic de la sala multijugador. Cal destacar que si en el camp ID del formulari ha d'escriure exactament l'identificador que li ha d'haver passat l'amfitrió, en el cas que la sala no existeixi, hagi començat la partida o el camp estigui buit, es notificarà per pantalla.

2. L'usuari prem el botó de "*Join*".

3. Surt una petició d'abonament de MetaMask.L'usuari ha de pagar la mateixa quantitat que ha pagat l'usuari que l'ha creat, ja que, com està definit al sistema de recompenses, tothom paga el mateix.

4. L'usuari confirma la transacció.

5. Carrega la sala d'espera. Al cap de pocs segons, li carrega la sala d'espera, on, com al cas de l'amfitrió pot compartir la clau per unir-se, però, com informa el *frontend*, no pot començar la partida, ha d'esperar que l'amfitrió comenci.

Si un usuari se surt del Wordle es pot tornar a unir en qualsevol moment fent servir l'opció de "*Join Room*".

Hi ha una gran diferència respecte a la sala d'espera de l'amfitrió, el client d'aquest usuari fa peticions cada pocs segons al servidor, al *backend*, per comprovar si l'amfitrió ha començat la partida.

De manera que si un usuari no amfitrió passa més de dos minuts a la sala d'espera, l'expulsarà i haurà de tornar a unir-se fent servir el "*Join Room*", òbviament sense cap cost addicional, ja que havia pagat amb anterioritat. Firebase té un nombre de peticions, consultes, mensuals limitades, i d'aquesta forma s'evita arribar a aquest límit.

També hi ha un motiu de ciberseguretat per la implementació d'aquesta característica. Si no estigués això implementat, un usuari amb males intencions, que vol deixar la pàgina web fora de servei, només hauria de tenir diversos usuaris esperant a una sala d'espera sense fer res per deixar la web sense servei.

Quan, l'amfitrió li dona a començar la partida, a tots els usuaris que estan a la sala d'espera, els hi carregarà en Wordle, sense necessitat de carregar de nou la pàgina (ja que s'està comprovant cada pocs segons si l'amfitrió ha començat la partida).

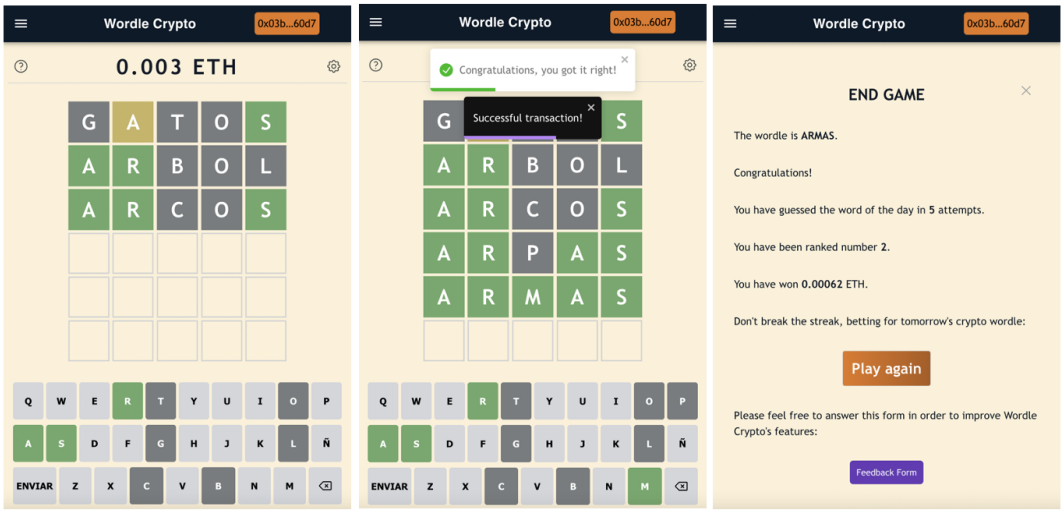

Figura 30: Captures de l'apartat *Multiplayer* jugant al Wordle

Les funcionalitats dins el joc de Wordle són les mateixes que al mode principal però adaptades al caràcter del nou mode de joc. A la pantalla fi de joc, a diferència del mode Wordle Crypto, diu la posició en la qual ha quedat l'usuari.

5.3.4 Themes Store

Aquesta secció és una botiga on intercanviar elements estètics, temes, per criptomonedes, pels ethers de Rinkeby.

Aquests temes estètics es poden utilitzar per la interfície de joc del Wordle dels dos modes de joc. Es poden aplicar, canviar, en qualsevol moment dins d'una partida de Wordle.

Va sorgir per dos motius:

- Mostrar altra aplicació de les criptomonedes. Un dels objectius d'aquest projecte és demostrar la utilitat i el valor que pot arribar a aportar el món Web3 a les aplicacions.

- Revalorar l'ether de Rinkeby donant-li un propòsit dins l'aplicació. L'ether de Rinkeby no té un valor real, no té un valor econòmic real com el té la *Mainnet* d'Ethereum, almenys que tingui un valor dins l'aplicació Wordle Crypto.

A mesura que l'usuari millora en el joc i aconsegueix més ethers, pot adquirir més temes estètics, sentint un progrés real dins la web.

Aquest apartat compleix és un dels objectius del sistema de recompenses: aconseguir que l'usuari se senti impulsat, motivat, a jugar a Wordle per aconseguir criptomonedes. Si aquestes, no tenen cap utilitat, l'usuari perd aquesta motivació. Oferint aquest servei, aquest ether té una finalitat i no queda inservible.

Es mostra els temes de la base de dades mapejats en *cards*. El color mostrat en aquestes és el color que tindrà de fons el Wordle si s'aplica el tema.

Segueix el flux d'accions característic de Wordle Crypto:

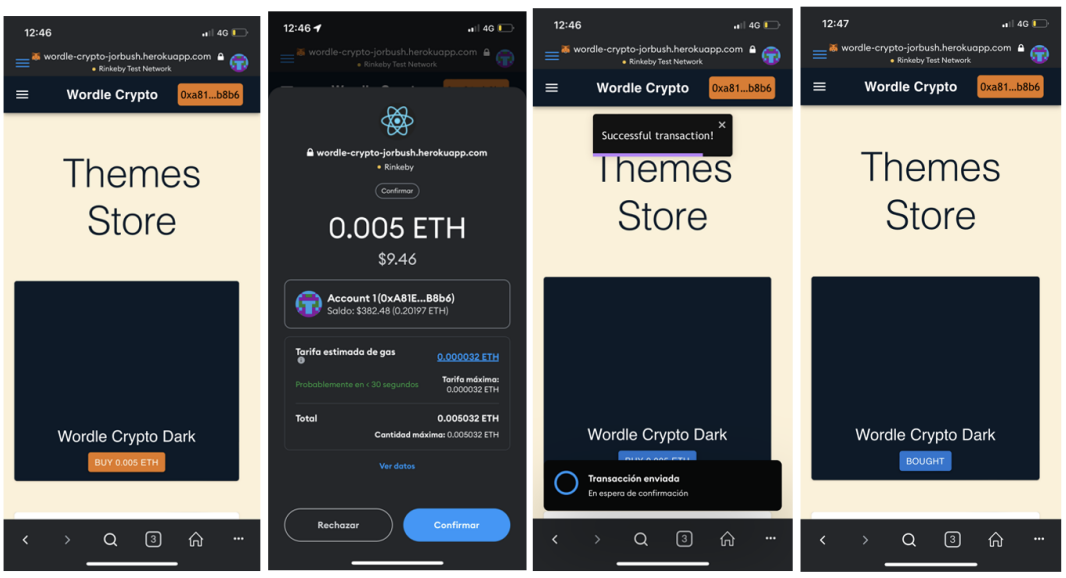

Figura 31: Captures de l'apartat *Themes Store* comprant un tema

1. L'usuari polsa el botó de compra d'un tema.

2. MetaMask fa una sol·licitud de transacció, del compte de l'usuari al del Wordle Crypto, de la quantitat d'ethers especificada en el botó de compra.

3. L'usuari accepta l'abonament.

4. La interfície visual s'actualitzarà. Es notifica a l'usuari i canvia el botó a color blau amb un text de "comprat".

5.3.5 Obtain Crypto

Abans de gaudir dels modes de joc de Wordle Crypto, l'usuari necessita ether per fer l'abonament inicial que requereixen aquests modes de joc. És per aquest motiu que existeix aquesta secció: *Obtain Crypto*.

Aquí l'usuari primerenc pot reclamar uns ethers gratuïts que ofereix Wordle Crypto, concretament són 0.003 ETH. Amb aquest ether, l'usuari podrà jugar dues partides (0.001 ETH cadascuna més la taxa del gas).

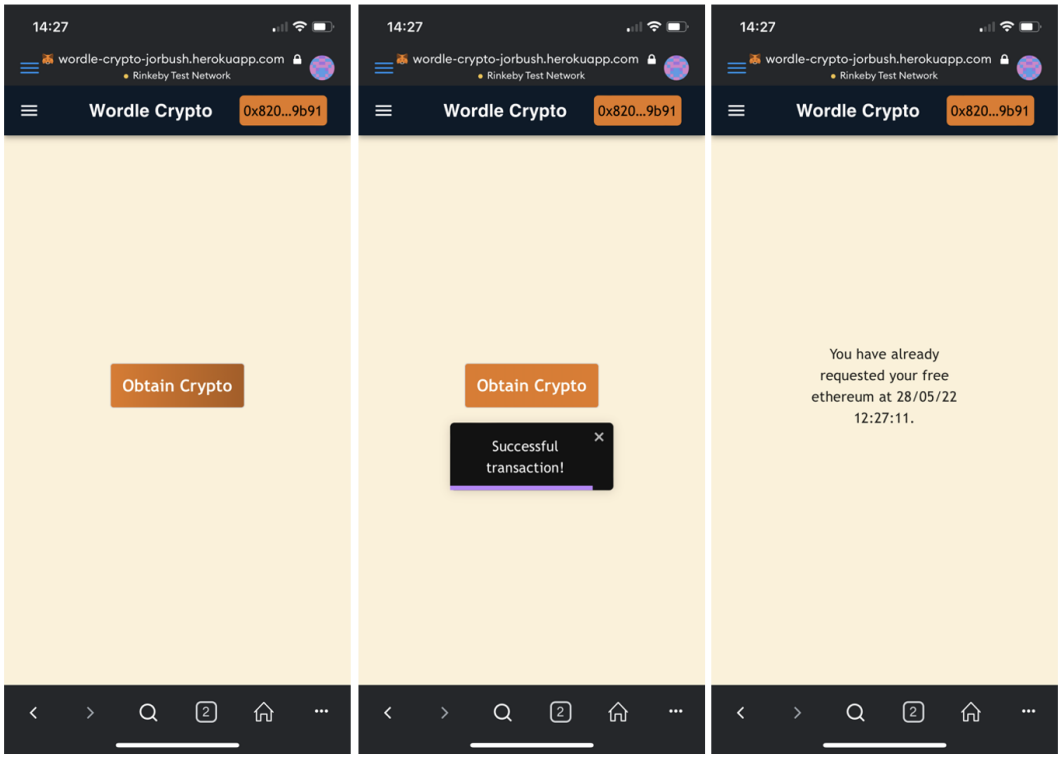

Figura 32: Captures de l'apartat *Obtain Crypto* de Wordle Crypto

L'usuari ha de prémer el botó i obtindrà l'ether. Cada usuari de Wordle Crypto només pot aconseguir aquest ether un sol cop, és per això que un cop rep les criptomonedes s'imprimeix per pantalla la data exacta de quan es va sol·licitar, perquè no hi hagi pas a la confusió.

5.3.6 Send Ether

En el cas que un jugador vulgui enviar ethers a un altre jugador perquè aquest últim no tens suficients fons per jugar o pel motiu que sigui, pot fer sevir la secció *Send Ether*. En aquesta secció completant un breu formulari es pot trametre ethers entre usuaris de Wordle Crypto.

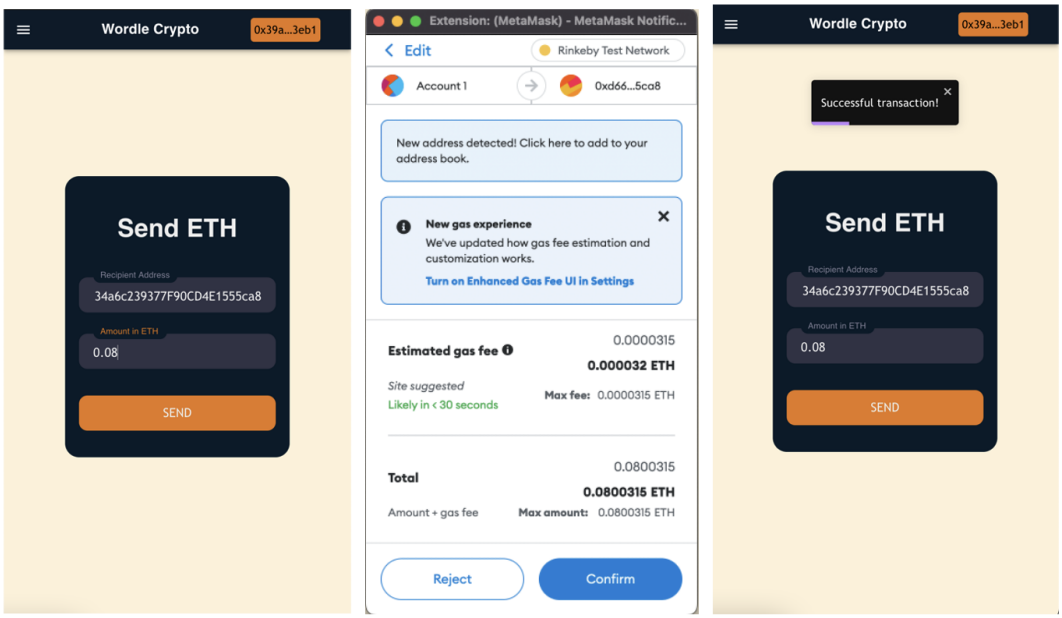

Figura 33: Captures de l'apartat *Send Ether* de Wordle Crypto

Quan es completen els camps del formulari, els camps d'adreça i quantitat d'ether, i es polsa el botó d'enviar, sortirà una confirmació de MetaMask amb el cost total de la transacció; és a dir, l'ether que es vol enviar més la taxa del gas. Un cop s'ha tramès, surt una notificació per pantalla.

5.3.7 HowToPlay i About

Si és el primer cop que l'usuari accedeix a la web, la qual cosa es detecta fent servir les *cookies* del navegador (l'emmagatzematge local de la web), carregarà la secció *HowToPlay*, però a través del menú de la capçalera es pot accedir en qualsevol moment.

En aquesta secció, es fa una breu guia global a l'usuari sobre els diferents apartats i funcionalitats disponibles en aquesta aplicació Web3. Es pot veure el contingut de la pàgina en les següents captures:

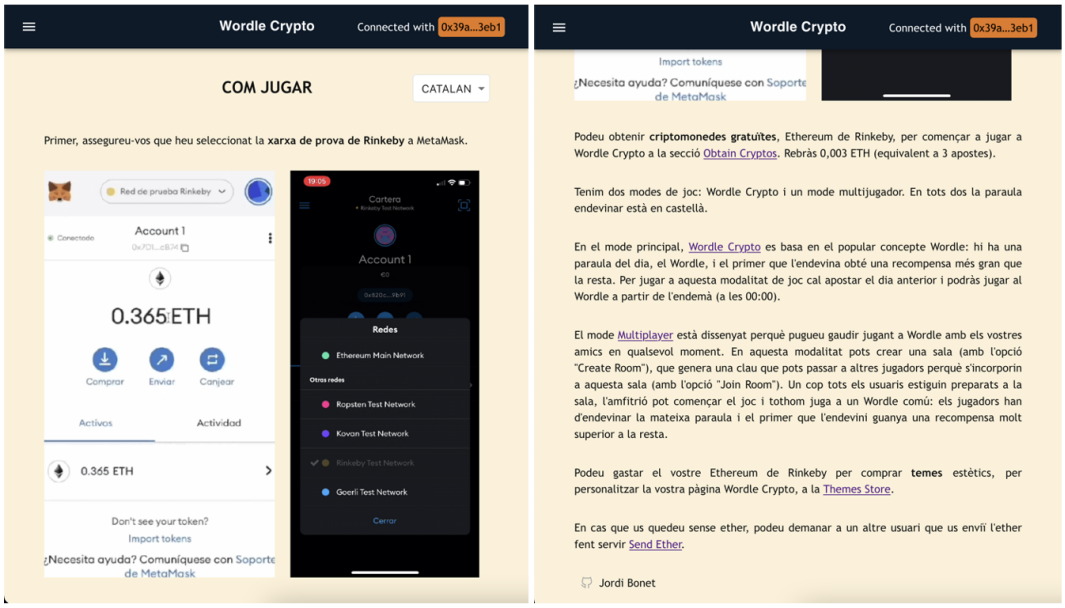

Figura 34: Captures de l'apartat *HowToPlay* de Wordle Crypto

Abans d'explicar les seccions de l'aplicació, hi ha dos GIFs mostrant com seleccionar la xarxa de Rinkeby, ja que és necessària per fer servir les funcionalitats de la web. Es pot canviar l'idioma del text utilitzant el selector d'idioma situat a la dreta del títol i es guarden les preferències de l'usuari a les *cookies* perquè no hagi de canviar d'idioma sempre que vulgui consultar aquesta informació.

Fent servir el menú de la capçalera podem anar fins a una secció que té un comportament similar: *About*.

Figura 35: Captura de l'apartat *About* de Wordle Crypto

Aquí es pot trobar informació més detallada de les tecnologies que fa servir la

web així com una breu explicació de com funciona el sistema de recompenses de Wordle Crypto.

Per finalitzar, hi ha un formulari de Google, on l'usuari pot donar la seva opinió sobre Wordle Crypto, reportar bugs i sobretot donar un *feedback* sobre el sistema de recompenses i l'aplicació amb l'objectiu de millorar-la.

5.4 Seguretat

5.4.1 Autentificació amb tokens

Un apartat essencial és la seguretat de la pàgina web, assegurar que la privacitat dels usuaris es manté, així com assegurar que la integritat de la base de dades es manté; és a dir, que ningú pugui manipular les dades. És per aquest motiu que s'ha d'implementar una manera d'autentificar als usuaris quan accedeixin als endpoints del *backend*, concretament s'utilitzarà l'autentificació mitjançant *tokens* amb la tecnologia *JSON Web Token*, més conegut com a *JWT*.

El *JSON Web Token* és una forma segura de transferir *tokens* aleatoris entre dues parts o entitats, que en el nostre cas és entre l'usuari i el servidor.

Si un usuari vol accedir a un *endpoint* de la base de dades que tingui implementat aquest requisit d'autentificació, li haurà de passar un usuari i contrasenya que el servidor pugui verificar com autoritzat. És per això, que quan es criden als *endpoints* del *backend* des del *frontend*, fent servir la llibreria Axios, s'ha de passar un paràmetre addicional de configuració amb un usuari i contrasenya autoritzats.

Si un usuari intenta accedir a un *endpoint* del *backend* des de fora de la interfície de la web, ja sigui accedint directament a l'URL en el navegador web o utilitzant Postman, li demanarà usuari i contrasenya i en el cas que no estigui autoritzat, no podrà accedir a l'endpoint.

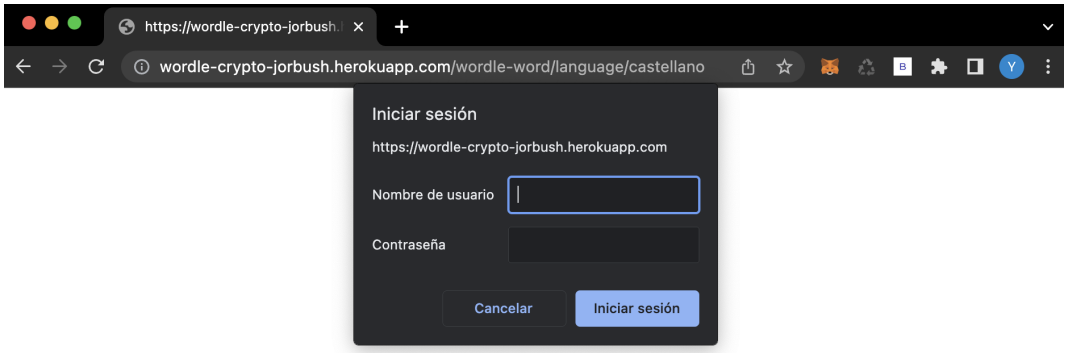

Figura 36: Captura intentant accedir a un *endpoint* protegit

Aquesta tecnologia està implementada a tots els *endpoints* que manipulen la base de dades; és a dir, els mètodes PUT, POST i DELETE. També s'ha implementat en alguns mètodes GET que retornen dades que poden vulnerar la privacitat de l'usuari o que contenen informació que pot afectar a la integritat del joc, concretament la

paraula a esbrinar, ja que, si algú arriba a obtenir-la de manera fraudulenta, podria guanyar la majoria d'ether de la col·lecta, el 70%, la qual cosa serà injusta per la resta de jugadors de la partida.

5.4.2 Encriptació

A més de l'autentificació amb *token* per part de l'usuari, amb l'objectiu d'assegurar que ningú pugui fer trampes i accedir de manera fraudulenta a la paraula a esbrinar quan estigui jugant a una partida de Wordle Crypto.

En el següent diagrama es pot observar com s'encripta la paraula a esbrinar (en aquest exemple es "*solaz*").

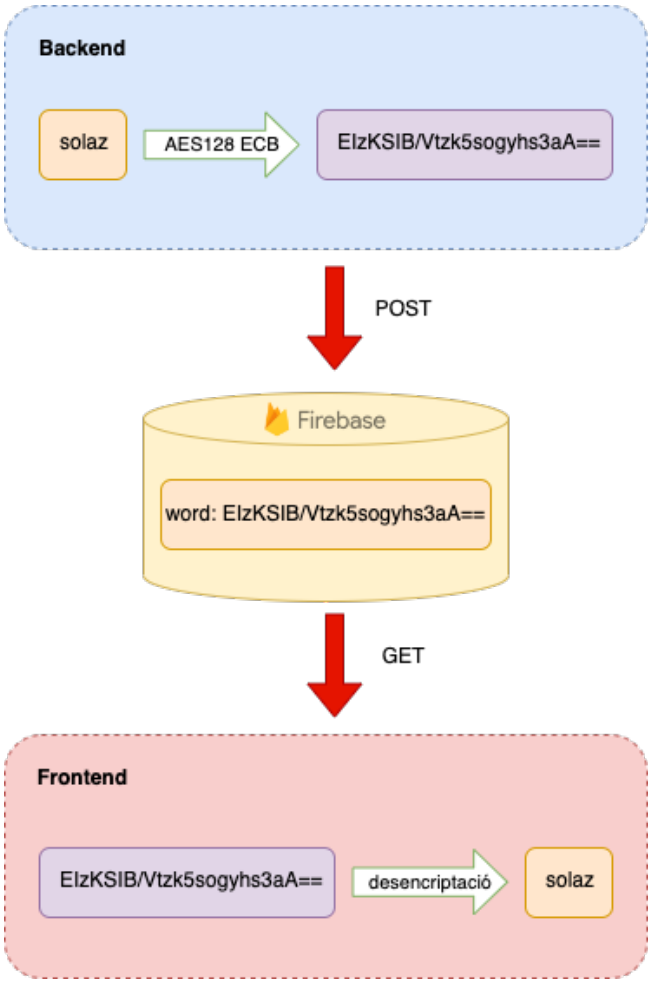

Figura 37: Diagrama d'encriptació de la paraula del dia

L'encriptació segueix els següents passos:

1. S'encripta la paraula en el *backend*. Un cop s'ha generat aleatòriament la paraula de cinc lletres i s'ha comprovat que aquesta és una paraula que existeix a l'API del diccionari, es codifica fent servir una encriptació AES128 ECB.

2. Es guarda la paraula encriptada en la base de dades, fent una petició POST.

3. S'obté la paraula encriptada, guardada a la base de dades, en el *frontend*, realitzant una petició GET.

4. La paraula es desencripta en el *frontend*. Es desencripta cada vegada que s'hagi de fer servir la paraula dins del codi perquè si només es desencripta un únic cop en una única variable, aquesta és més senzilla de vulnerar i aconseguir el seu valor, que no pas si no es guarda en cap moment el valor desencriptat.

L'encriptació AES és un procés que oculta dades electròniques amb un algorisme de xifratge simètric del *Advanced Encryption Standard* (o AES), en aquest cas de 128 bits com indica la nomenclatura, la qual cosa vol dir que la clau que es fa servir per encriptar i desencriptar és de 128 bits.

Aquesta encriptació està implementada tant al mode Wordle Crypto com al mode multijugador.

6 Conclusions i treball futur

6.1 Conclusions

L'objectiu principal d'aquest projecte era crear una aplicació Web3, concretament agafar una aplicació popular per tots els públics, com és el Wordle, i aplicar-li les criptomonedes de manera que aquestes li aportin valor.

Una característica que compleix aquest requisit és la identificació d'usuaris utilitzant l'adreça pública del compte d'Ethereum de l'usuari mitjançant el *wallet* de MetaMask. D'aquesta manera s'identifica l'usuari amb un parell de *clicks*, la qual cosa suposa una innovació respecte a la tradicional identificació d'usuari de la Web2, on era necessari introduir un usuari, o correu, i contrasenya per identificar-lo.

Aquest propòsit s'ha complert, ja que, gràcies al sistema de recompenses, Wordle Crypto és prou satisfactori per continuar jugant per aconseguir ether per desbloquejar nous temes estètics pel joc.

El mode multijugador fa que jugar amb altres jugadors a Wordle sigui més interessant i hi hagi una motivació extra per guanyar. Llavors, és per aquest motiu que el requeriment d'afegir el món Web3 i aportar valor a la pàgina web, s'ha satisfet.

Complint la hipòtesi plantejada, és possible que si el món Web3, les criptomonedes, es tornen cada vegada més acceptades per la nostra societat, com està passant, cada vegada hi haurà més serveis que integrin aquesta tecnologia en aplicacions i pàgines web ja existents, i que n'apareguin de noves, perquè emportant-se una comissió per cada transacció dins la seva aplicació podria arribar a ser molt beneficiós econòmicament per aquesta, la qual cosa sol ser l'objectiu principal de tota empresa.

6.2 Treball futur

- *Mainnet* d'Ethereum. Com s'ha mencionat en diverses ocasions, la finalitat d'una xarxa de proves, o una testnet, és per provar el funcionament de les aplicacions Web3 per desplegar-les fent utilitzant criptomonedes reals en un futur. En aquest cas, en comptes de fer servir la xarxa de Rinkeby, es faria servir la *Mainnet* d'Ethereum, on l'ether té un valor econòmic real. Per fer això possible, s'hauria de fer petits canvis en el codi i en els proveïdors de serveis, per seleccionar que es fes servir la xarxa principal d'Ethereum. A més, seria convenient, eliminar la secció *Obtain Crypto*, ja que no té cap benefici rendible regalar ethers reals a qualsevol usuari que es doni d'alta, sobretot amb el valor econòmic que tenen avui dia.

- *Smart Contract*. En un futur es podria implementar la lògica del sistema de recompenses dins un *Smart Contract*, perquè d'aquesta manera seria més robust i sòlid, sense necessitar una base de dades externa per la seva gestió. D'aquesta manera també es podria publicar a la pròpia *blockchain* d'Ethereum,

passant de ser una aplicació Web3 a una DApp (una aplicació descentralitzada), ja que s'ubicaria directament dins d'un *block* en la blockchain, com si fos una transacció.

- Sistema de recompenses. A partir del *feedback* rebut a les enquestes disponibles dins l'aplicació Web3, es pot arribar a fer canvis afegint nous modes de joc, arreglant futurs bugs i errors, i, sobretot el que es fa èmfasi en l'enquesta, el sistema de recompenses. Potser la majoria dels usuaris prefereixen altre factor a tindre en compte per guanyar el Wordle (que no sigui el temps) o troben més just actualitzar el percentatge de la recompensa d'ether que s'emporta el guanyador, el qual ara mateix es troba al 70%.

- Preu del gas (*Gas Price*). Un altre factor a tindre en compte, que potser s'ha d'actualitzar en un futur, és el preu de gas assignat a les transaccions que es fan del servidor a l'usuari (utilitzant el proveïdor d'Infura), ja que podria ser que en un futur aquest fos més elevat i s'hagués d'actualitzar, però també és possible que no sigui necessari, depèn de com evolucioni la xarxa de Rinkeby.

- Base de dades. En el referent a millores en l'àmbit d'aplicació, seria millor fer servir un altre servidor i base de dades, amb l'objectiu d'incrementar la seguretat i la qualitat del servei (com el temps de resposta i el nombre de peticions). A causa del límit de peticions de Firebase, la base de dades deixa de donar servei a les 55000 peticions al dia, la qual cosa s'ha tingut en compte a l'hora de fer la implementació de la web perquè això no passi. A pesar d'això, si l'aplicació es tornés popular sí que seria un canvi necessari, o com a mínim pagar una versió *Premium* del servei de Google.

- Seguretat. També altra vulnerabilitat són les claus privades del compte d'Ethereum de Wordle Crypto, la clau privada per encriptar i l'usuari i contrasenya per fer modificacions a la base de dades, es troben al codi, i, a pesar que no es poden aconseguir a priori, es pot trobar un sistema millor on guardar-les.

- Contingut. Una altra millora és crear més contingut dins l'aplicació, incrementar el nombre de temes estètics a la botiga i afegir més idiomes per jugar al Wordle, com l'anglès i el català (la qual cosa es volia implementar en un principi).

- Implementació de *bots* amb AI. Una altra característica interessant que es pot afegir en un futur a la web són *bots* que, mitjançant intel·ligència artificial, juguin al Wordle com un jugador més perquè sempre hi hagi els suficients jugadors al mode principal de joc i la recompensa sigui més generosa. La dificultat seria ajustable perquè no sigui frustrant o injust per la resta de jugadors, de manera que jugarien com un jugador mitjà de l'aplicació.

- Un mode de joc més just i accessible. A l'etapa final del desenvolupament, es va plantejar la implementació d'un tercer mode de joc on la paraula fos aleatòria, una paraula diferent per cada jugador, i que es tingués en compte altre factor, en lloc de ser el primer que resol la paraula, a l'hora de premiar al

guanyador: el nombre d'intents. D'aquesta manera, els jugadors no haurien de jugar a una hora exacta, ja que no tothom disposa de la mateixa disponibilitat, i seria impossible que es filtrés la paraula del dia, pel fet que tots en tenen una de diferent.

Referències

[1] Ethereum Foundation. *Ethereum Development Documentation*, `https://ethereum.org/en/developers/docs/`, 2022.

[2] Radar Relay. *Ethereum Gas Explained*, `https://ethgas.io/`, 2022.

[3] Max Thake. *What is Proof of Stake? (PoS)*, `https://maxthake.medium.com/what-is-proof-of-stake-pos-479a04581f3a`, 2022.

[4] Aaron Davis. *MetaMask's Developer Documentation*, `https://docs.metamask.io/`, 2022.

[5] Infura Inc. *Infura*, `https://infura.io/`, 2022.

[6] Creative Commons. *Documentation*, `https://docs.ethers.io/`, 2022.

[7] Michael Aboagye. *How to Secure a Flask REST API with JSON Web Token?*, `https://geekflare.com/securing-flask-api-with-jwt/`, 2020.

[8] Sacha Dehe. *Encrypt/Decrypt Data between Python 3 and JavaScript (AES algorithm)*, `https://medium.com/@sachadehe/encrypt-decrypt-data-between-python-3-and-javascript-true-aes-algorithm-7c4e2fa3a9ff`, 2021.