Grau de Tecnologies de la Telecomunicació

Treball Final de Grau

Descentralització d'Aplicacions amb Blockchain

Ángel Noriega Moliner
Aplicacions i sistemes distribuïts

Consultor: Félix Freitag
Data Lliurament: 09/06/2019

FITXA DEL TREBALL FINAL

Títol del treball:	*Descentralització d'Aplicacions amb Blockchain*
Nom de l'autor:	*Ángel Noriega Moliner*
Nom del consultor:	*Félix Freitag*
Data de lliurament (mm/aaaa):	*06/2019*
Àrea del Treball Final:	*Aplicacions i sistemes distribuïts*
Titulació:	*Grau en Tecnologies de la Telecomunicació*

Resum del Treball (màxim 250 paraules):

Aquest projecte final de grau marca l'objectiu de donar una perspectiva sobre les tecnologies de descentralització d'aplicacions, centrant-se en la seva major part en el *Blockchain*, i de reforçar aquest coneixement al final amb el desenvolupament d'una aplicació descentralitzada.

De forma inicial, explica i documenta quin és el significat d'un terme molt utilitzat en els últims anys com és el de *Tecnologia Blockchain*. També fa èmfasi sobre l'ús dels contractes intel·ligents, el projecte Ethereum, altres tecnologies descentralitzades com el protocol IPFS, i les aplicacions WEB d'execució en client.

La segona part del treball defineix els requeriments tant funcionals com tècnics d'una aplicació web descentralitzada, descrivint el procés d'inici a fi i relatant les necessitats bàsiques per al funcionament. Es realitzarà el desenvolupament d'una aplicació basada en les tecnologies mencionades per tal d'experimentar amb la matèria investigada a la part teòrica.

Abstract (in English, 250 words or less):

This final degree project marks the objective of giving a perspective on the tecnologies of decentralization of applications, focusing in the *Blockchain* among others, and reinforcing this knowledge at the end with the development of a decentralized application.

Initially, it explains what is the meaning of a term widely used in recent years, such as that of Blockchain Technology. It also emphasizes about the use of *smart-contracts*, the Ethereum project, other decentralized technologies such as the IPFS protocol, and the WEB applications that run on client.

The second part of the project defines the functional and technical requirements of a decentralized web application, describing the development process from the beginning to end and relating the basic needings. The development of an application based on the mentioned technologies will be carried out in order to make an experimentation with the subject investigated in the theoretical part.

Paraules clau (entre 4 i 8):
Blockchain, Ethereum, D-App, IPFS, WEB3, Vue.JS, Solidity

Índex

Llistat de figures

1. Introducció

1.1 Context i justificació del Treball

Aquest treball parteix de la tendència que està començant a sorgir en l'àmbit de la descentralització de xarxes. En l'actualitat, els proveïdors de serveis de xarxa i *hosting* sovint tenen els seus servidors dispersats arreu del món en funció dels seus interessos i conveniències. Factors econòmics, climàtics, tecnològics: són determinants a l'hora d'escollir la localització dels nodes servidors per part dels proveïdors de xarxa, pensant per tant en oferir els seus productes de forma singular o combinada sense necessitat de que es trobin a la mateixa màquina. Un clar exemple és la coexistència de servidors CDN, BBDD, Web, Clústers formant tot ells parts de la mateixa arquitectura, intercomunicats, i realitzant cadascun la seva tasca (per exemple, una plataforma d'*streaming*). Aquests dissenys descentralitzats obren la porta a l'accés i a la comunicació entre servidors i usuaris el qual provoca que sigui un procés completament transparent.

Just després d'aquest punt, és on s'inicia el relat d'aquest treball, consistent en la investigació sobre una solució per al desenvolupament d'aplicacions completament distribuïdes. Per tant, la rellevància d'aquesta obra recau en l'acció d'afegir un esglaó als serveis coneguts de forma comuna per tal de donar a conèixer la possibilitat de crear serveis amb independència en quant al control per part d'un servidor. Aprofitant el seguit de tecnologies descentralitzades que han sorgit en els darrers anys (de forma fonamental el *Blockchain* i el IPFS), aquesta redacció pretén tenir com a propòsit documentar els coneixements obtinguts en el desenvolupament d'una aplicació completament autònoma, de la qual la seva gestió i control és responsabilitat de la xarxa.

De forma resumida, el resultat final desitjat, és el de poder posar en marxa un servei web complet sense necessitat de servidor ni sistema de control central.

1.2 Objectius generals i específics del treball

Aquest treball final de grau pretén desenvolupar la tasca d'investigació sobre el concepte de base de dades descentralitzades, així com fer una síntesi de la història del *Blockchain*, i fer entendre les possibilitats d'explotacions mitjançant la integració d'aplicacions.

L'objectiu mes general, és el de reunir totes les pautes i coneixements necessaris per al desenvolupament i llançament d'una aplicació amb connexió a una base de dades descentralitzada. Contendrà la documentació necessària, així com un pla organitzatiu escalable a altres projectes.

De forma resumida, la fita principal és la generació d'un document capaç de fer entendre de forma clara el funcionament d'una aplicació basada en una base de dades descentralitzada, i fer-se partícip del gran ventall de possibilitats que obre en quant a la seva explotació per a projectes tecnològics.
Més específicament, es plantegen una sèrie de reptes que serviran per aconseguir complir objectius mes concrets.

Investigació sobre la lògica de funcionament.
Es pretén conèixer el funcionament i els actors que tenen lloc a la descentralització, així com de les garanties del seu funcionament.

Implantació de la BBDD.
El projecte també contempla la instal·lació i configuració d'una base de dades descentralitzada basada en *Ethereum*, documentant el procés d'instal·lació i configuració.

Desenvolupament aplicatiu descentralitzat.
Complementàriament, es té com a objectiu aconseguir desenvolupar una aplicació practica utilitzant el *framework* de Javascript Vue.JS i els corresponents complements necessaris, que interactuï amb la base de dades i en faci un ús correcte.

1.3 Enfocament i mètode seguit

El desenvolupament d'aquest projecte es basarà completament en la investigació. Consisteix en un projecte complet, que combinarà diferents tecnologies per aconseguir un únic objectiu final. La temàtica escollida i les eines amb les quals es pretén treballar, encara que tenen a disposició documentació d'ús a disposició dels usuaris, no han sigut creades per a ser utilitzades conjuntament de forma especifica. El punt de major dificultat i esforç recaurà en la part de connexió i interacció entre els diferents elements.

Per tant, l'estratègia a seguir serà la d'aprendre i formar-se en els conceptes bàsics de la descentralització, per tal de futurament iniciar la fase de desenvolupament que requerirà una investigació més profunda sobre cadascuna de les eines i de la seva interconnexió. Per les característiques del projecte, penso que aquesta estratègia és la única i adequada, especialment a causa de la manca de d'exemples d'us real que facin ús d'aquestes tecnologies.

1.4 Planificació del Treball

Brainstorming i presa de decisió. Consisteix en un plantejament general del projecte, incloent la elecció de la temàtica a implementar després de la recerca d'informació i documentació sobre les opcions inicials. Gràcies a l'ajuda del tutor es comença a definir l'abast del projecte.

Creació i planificació del pla de treball. Es realitza un esborrany inicial esmentant la consistència de la feina a desenvolupar, així com els objectius a assolir i la planificació.

PAC1. Punt de control amb l'entregable de la planificació. Presentació de la idea inicial a desenvolupar.

Introducció del TFG. Petita descripció del context històric del sorgiment del *Blockchain*, descripció de la lògica i funcionament, i evolució dels diferents projectes i companyies.

Ethereum. Descripció del projecte, les diferencies, i les seves característiques.

Contractes intel·ligents. Fa una descripció sobre el seu concepte.

Les D-App i exemples reals d'us. Detall sobre els diferents tipus d'aplicacions i els seus fonaments de funcionament.

PAC2. Punt de control amb l'entregable documentatiu. Està previst que contingui els avanços del gruix de la part teòrica i l'inici de la investigació tècnica de la part pràctica.

Disseny tècnic de la D-App a desenvolupar. Acotament de l'abast i temàtica del desenvolupament. Documentació sobre l'aplicació a desenvolupar, anàlisi funcional, descripció de les tecnologies emprades, i descripció tècnica dels objectes generats..

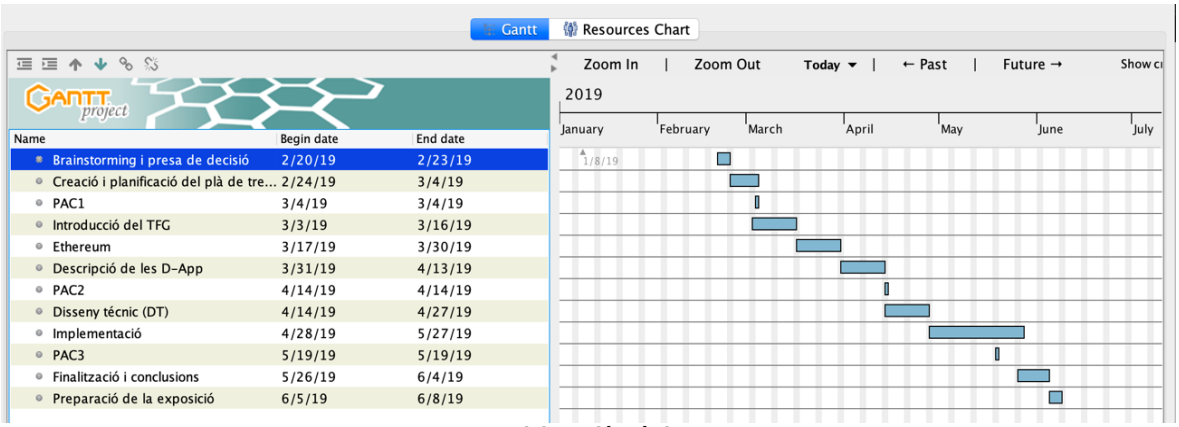
Instal·lació, configuració i desenvolupament.

PAC3. Punt de control amb l'entregable pràctic. Recavarà principalment un anàlisi de l'avenc de la part practica del projecte així com una organització de les tasques a enllestir. Marcarà l'inici de la finalització del projecte.

Enllestiment de la feina. Coixí de temps per a imprevistos, conclusions i maquetació.

Lliurament final. Preparació del suport gràfic per a l'exposició i vídeo.

Figura 1: Diagrama de Gantt



Font: Elaboració pròpia

En la Figura 1 es mostra el diagrama temporal creat inicialment. Distribueix la estimació aproximada de temps a dedicar a cadascun dels apartats. De forma inicial es centra en la part d'assoliment de conceptes per a passar posteriorment a la part practica.

1.5 Breu sumari de productes obtinguts

El resultat final del projecte constarà de 4 parts en total:

Memòria: Contindrà tota la documentació teòrica, el detall tècnic del desenvolupament i la informació sobre el projecte

PACs: Contindran informació sobre el plantejament del projecte (la primera PAC) i suposaran punts de control que enregistraran el progrés i les dificultats trobades durant les corresponents jornades que duri el projecte. Especialment es farà èmfasi en el detall d'aquelles jornades que han trobat major dificultat i les respectives solucions encetades.

Back-end: Consistent en el desenvolupament del contracte intel·ligent, el qual s'encarregarà de la comunicació entre l'aplicació i la base de dades.

D-App: és la part visual i pràctica de l'aplicació que donarà a l'usuari les funcionalitats per a la interacció entre l'usuari i els serveis.

1.6 Breu descripció dels altres capítols de la memòria

Introducció al *Blockchain*
En aquest apartat es realitza una síntesi sobre la tecnologia i dona els coneixements bàsics sobre la temàtica del treball.

Projecte Ethereum
Descriu les característiques principals del projecte i serveix d'introducció per a la futura part pràctica.

Contractes intel·ligents
Explica el concepte d'aplicació descentralitzada enfocant-se en la lògica de funcionament sense sistemes de control.

Tecnologies descentralitzades
En aquest punt es fa un enfoc general al concepte d'aplicació descentralitzada amb l'ús de diferents exemples. Té l'objectiu de donar al lector una visió general prèvia a la presa de possibles decisions tecnològiques.

Descripció funcional
En aquest apartat es defineixen les funcionalitats i l'abast de l'aplicació a desenvolupar per tal de garantir la usabilitat.

Detall tecnològic
S'enumeren i descriuen les eines utilitzades en la fase de desenvolupament i la motivació de l'ús de cadascuna d'elles.

Disseny tècnic
Descriu el procés de desenvolupament de l'aplicatiu descentralitzat, aportant informació tècnica sobre els diferents components utilitzat.

Conclusions
És un relat resumit de l'experiència viscuda durant l'elaboració del projecte, posant valor en els punts febles i virtualitzant les possibles millores a implantar en una pròxima fase.

Bibliografia
Recull la documentació i fonts d'informació que han format de les quals s'ha extret informació per a la materialització del projecte.

Annexos
Es un apartat complementari al treball que conté tota aquella informació que per qüestió d'extensió o temàtica no han tingut cabuda en la part principal del projecte.

2. El *Blockchain*

2.1 Introducció a la descentralització

Si s'hagués de definir el *Blockchain* en un sola frase, una aproximació adient, correcta i que s'acostaria bastant al significat real podria ésser: "base de dades distribuïda, formada per blocs de dades encadenats caracteritzada per la seva immutabilitat".
La principal de les seves característiques, gracies a la qual és protagonista d'aquest projecte, és la de descentralització. Per a definir aquest terme, començarem explicant la idea totalment contrària.

En molts (per no dir la gran majoria) dels serveis o aplicacions que utilitzem habitualment i que requereixen l'ús de la xarxa, la informació i l'intercanvi de dades es realitza de forma centralitzada. La qual cosa vol dir que les connexions i el flux de dades són controlades per un servidor central. Aquest fet, provoca que aquesta xarxa sigui potencialment feble davant d'un seguit de situacions o riscos, com poden ser per exemple casos de caiguda del sistema o d'atacs de suplantació d'identitat, que suposarien un perill respecte a la integritat de la informació. L'usuari no podria valorar el correcte intercanvi d'informació i no tindria cap control sobre les seves dades.

En canvi, en un sistema descentralitzat ideal, les dades són emmagatzemades i controlades per un número de nodes indeterminat dins de la xarxa, i la verificació d'una petició o de qualsevol transacció hauria d'esser validada per n número de punts. D'aquesta manera, la validació d'una operació no recaurà sobre un únic element de control de xarxa, la comprovació d'un moviment serà realitzat mitjançant el consens dels nodes que rebran el mateix registre. Per una altra banda, en cas de caiguda de servei d'un node, l'afectació del control de l'aplicació serà mínima gràcies a la característica de distribució i a la continua existència de la xarxa.

En els últims anys, s'ha posat molt de moda l'ús d'aquest tipus de tecnologia en diferents camps, on de forma destacable sobresurt el profit aplicat a l'àmbit de les criptodivises. És molt estrany que algú no hagi sentit algun cop la paraula "*Bitcoin*", i per una altra part, cada dia són més els individus que actualment tenen o han tingut algun "*token*" d'alguna criptomoneda en possessió.

2.1.1 Avantatges i inconvenients

El potencial el qual són capaces d'aportar les aplicacions descentralitzades és molt gran. Com s'anirà veient, ens brindaran un gran nombre d'avantatges en l'àmbit de la seguretat, eficiència i estabilitat en els serveis que en facin ús, però encara i així hi sorgiran un seguit d'inconvenients o problemes a tenir en compte per tal que no pugui fallar el seu funcionament.

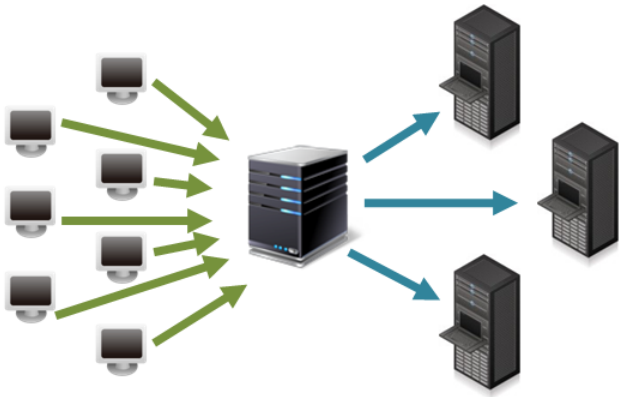
A continuació es mostren un seguit de punts que valoren els seus factors.

Avantatges

Balanceig de càrrega. Com indica la pròpia paraula, descentralització vol dir que quelcom deixi de dependre d'una entitat central de control homogènia, i passi a dependre d'organismes autònoms. Aquesta independència aporta una major estabilitat

a la connectivitat, evitant possibles colls d'ampolla i donant el control a la xarxa en comptes de a un servidor central.
Un exemple molt il·lustratiu de la importància del balanceig és el que es pot trobar en un sistema convencional, la caiguda del servidor provoca la caiguda del servei, en la descentralització, la caiguda d'un node no afectarà al funcionament de la xarxa i per tant es mitiga proporcionalment la probabilitat de caiguda del sistema com major sigui l'existència de nodes. Cal dir que per fortuna, avui dia els sistemes centrals estan preparats per aquest tipus de successos amb solucions de *mirroring* i balanceig, però també cal dir, que ho aconsegueixen amb la penalització de tenir uns costos elevats a base de replicació dels seus sistemes de serveis.

Figura 2: Esquema de balanceig de càrrega



Font: https://laredinfinita.wordpress.com/2014/05/11/balanceo-de-carga/

En la Figura 2 es mostra un exemple de balanceig, en el qual el tràfic de dades i peticions és rebut per part d'un node de càrrega, el qual s'encarregarà de redirigir el tràfic al servidor corresponent de la forma més òptima. Amb aquest sistema s'aconsegueix evitar saturacions de peticions que afectin directament als servidors i distribuir uniformement l'ús dels recursos.

Seguretat. L'existència de múltiples nodes amb la copia d'un mateix registre de la base de dades, a part de ser una garantia davant la possible pèrdua de dades, permetrà validar les transaccions de forma mes segura, rapida i amb simultaneïtat. En cas que un atacant aconseguís corrompre un node de la xarxa, la informació no estaria en risc gracies a les lògiques de consens de dades i a la replicació de la base de dades per tota la xarxa. Atacs del tipus "*man in the middle*" deixen de tenir èxit, ja que l'existència de tercers nodes impossibiliten la credibilitat d'agents modificadors externs. Per a que un atac tingues èxit[1], seria necessari corrompre el 51% dels nodes de forma simultània, trencant tota la bateria de tecnologies de seguretat que conformen la integritat de la xarxa.

Escalabilitat. Un sistema descentralitzat, gràcies a la seva propietat de diversificació, serà fàcilment escalable en funció de les necessitats de la xarxa. Donada una aplicació que viu i pateix un creixement molt ràpid i accelerat, és molt probable que arribi a un punt de saturació, on els nodes no siguin capaços de validar totes les transaccions que reben produint-se els temuts "colls d'ampolla". Al ocorrer casos com l'esmentat, pot arribar el moment del sorgiment de la necessitat d'integració de nous nodes que aportin entre d'altres, augment del poder de processament. Aquesta acció es realitzaria amb la simple replicació dels nodes existents, disposant això si, d'una copia del registre comú de la cadena de blocs per tal passar a formar de la xarxa. Queda vist doncs, que la facilitat de redimensionament de les aplicacions és molt senzilla d'abordar, no essent

igual de simple en un sistema dissenyat de forma central, en el qual de l'adhesió o ampliació de l'organisme de control seria problemàtica i de ben segur implicaria un canvi en l'arquitectura i una aturada o detriment del servei almenys, de forma temporal.

Confiança. Cada usuari és propietari de les seves dades, mitjançant la xarxa, sense dependència de terceres entitats com succeeix per exemple en altres casos, com els sistemes centrals. De forma complementaria, els sistemes distribuïts no requereixen conèixer dades personals per a identificar als seus usuaris. Únicament existiran adreces publiques que correspondran a identificadors de dades interpretables pel sistema.

Inconvenients

Lògica complexa. Per a poder fer realitat el funcionament d'aplicacions descentralitzades basades en la xarxa, és necessària l'existència i implementació d'algorismes de control robustos, que impliquen molta més complexitat que la que requereix un sistema de control central. Els nodes descentralitzats, en general, treballaran i compartiran les seves dades en mig de xarxes publiques com per exemple Internet. És cert que també hi haurà possibilitat de que siguin executades en entorns privats. Igualment, cal pensar que serà imprescindible que existeixin mecanismes que garanteixin la seguretat d'aquesta informació així com la seva integritat davant de manipulacions, i això suposarà un alt cost inicial de recursos i de temps per a la posada en marxa del sistema "automatitzat".

Saturació. Aquest sistema no podrà evitar la existència de situacions de saturacions de la xarxa mitjançant la previsió. En moments d'alt nombre de transaccions, els temps de processament de peticions inevitablement seran elevats al igual que en sistemes de control centrals. Com a exemple molt representatiu, aquesta situació va ésser viscuda per la xarxa *Bitcoin* a finals de 2017, on la alta quantitat de transaccions de compra i venda va provocar que únicament es gestionessin aquelles amb una comissió més elevada.
Aquestes situacions no seran ni fàcils ni rapides de resoldre, ja que serà necessari l'existència d'un consens majoritari entre nodes per a l'execució d'accions, la qual cosa penalitzarà temporalment totes aquelles solucions que no siguin assolibles únicament amb la inclusió de nous nodes.

2.2 Història i context

La historia i funcionament del *Blockchain* depèn directament de l'evolució de la criptografia i dels seus avenços. Els inicis de la criptografia[2], procedeixen de forma directa com moltes altres tecnologies (per exemple els sistemes de localització GPS) de les necessitats militars del segle XX. Des de sempre, la comunicació i l'intercanvi d'informació d'alt secret ha sigut una de les estratègies fonamentals per tal de disposar d'avantatges tàctiques, especialment en maniobres amb pretensió d'enxampar per sorpresa a l'enemic. La clau i l'èxit d'aquestes necessitats es fonamenta en la capacitat de poder realitzar comunicacions secretes i privades, impossibles d'interpretar per part de l'enemic, i que no requereixin una dificultat excessiva en quant a temps per a la seva descodificació al llenguatge entenible per part del receptor.

La segona guerra mundial va ésser el punt de partida de la necessitat de confidencialitat de la informació, principalment en la difusió de missatges en un territori entre unitats d'un mateix bàndol sabent que l'enemic era capaç de rebre els missatges. L'existència

de codis i de claus únicament conegudes per part de l'emissor i receptor feia que les comunicacions fossin inintel·ligibles per part de l'enemic. No obstant, i des d'aquell moment, la batalla entre la codificació i la descodificació de missatges confidencials no ha cessat, passant a convertir-se en una cursa en la qual dos gossos (que simbolitzen l'encriptació i desencriptació) intenten mossegar-li la cua a l'altre. L'afany de millorar de forma continua la secretització dels missatges propis, i la descodificació avançada de les claus enemigues representen el símil de la oració anterior.

Figura 3: Màquina "Enigma"

Font: Fundació Telefónica

Una de les àvies de l'encriptació que s'encarregaven d'aquesta tasca és la màquina "Enigma", que es pot visualitzar a la part esquerra de la Figura 3. Aquesta va ésser una de les màquines de xifratge més famoses de la nostra història i també una de les peces més dèbils que va tenir el regim Nazi. Des d'aleshores, l'ús de la criptografia ha anat avançant especialment forçat per les necessitats de confidencialitat.

Entrant en matèria amb el projecte, els inicis de la tecnologia de cadena de blocs daten del 1991, quan els científics Stuart Haber i W. Scott Stornetta[3] van introduir una solució computacional que permetia signar amb temporització documents digitals per tal que no poguessin ésser modificats o manipulats.

Aquest sistema consistia en una cadena de blocs criptogràficament segura que emmagatzemava documentació. No obstant al 1992 es va introduir al disseny de cadena de blocs els arbres de Merkle, cosa que va aportar major eficiència al permetre que diversos documents es poguessin guardar en un mateix bloc, i al facilitar la comprovació de la integritat mitjançant els Hash. Encara de l'efectiu funcionament del sistema, aquesta tecnologia no es va arribar a utilitzar i la patent va caducar al 2004.

Hal Finney, va introduir al 2004 un sistema anomenat RPoW (Proba de Treball reutilitzable). Aquest sistema va resoldre el problema del doble ús d'un mateix token, mantenint la seva propietat i permetent a la resta d'usuaris de tot el món verificar la seva exactitud i integritat en temps real.

A partir d'aquest moment, a finals de 2008, va néixer el que avui dia coneixem com el projecte pare de totes les aplicacions descentralitzades que es coneixen a dia d'avui: el *Bitcoin*.

2.3 Lògica de funcionament

Blockchain és la determinació que rep de la base de dades distribuïda utilitzada per la cripto-moneda *Bitcoin*. Mes endavant es descriurà en detall informació sobre aquest projecte, però per ara, és inevitable explicar el funcionament de la cadena de blocs sense definir molts dels aspectes principals del *Bitcoin* degut a la seva relació fonamental.

El següent apartat té com a objectiu aportar una visió general del funcionament de la cadena de blocs.

2.3.1 Fonaments

El cor del *Bitcoin* és format per una base de dades distribuïda, la ubicació de la qual és continguda en una còpia diversificada a tots els nodes de la xarxa, anomenada "llibre comú". Aquest llibre comú, és similar al que podríem conèixer de forma tradicional en el model econòmic FIAT, com a llibre major.

En els sistemes financers tradicionals, els valors estan representats mitjançant aquests llibres comptables (base de dades) administrades per institucions financeres. Els usuaris que volen formar part del sistema estan obligats a dipositar la seva confiança en aquestes administracions centrals i creure en la seva bona fé. Alguns dels aspectes susceptibles de compliment per a la confiança són la garantia de no alteració de les dades sota cap motiu i confiant en que la seva integritat enfront a fraus o atacs externs queda garantida.

En canvi, en el cas de la descentralització, aquest llibre contindrà tot el registre de transaccions des dels inicis de la moneda, amb tota la informació sobre generació de tokens, les transferències i els fons que disposa cada adreça de forma publica. Això vol dir, que és un sistema obert i transparent, en el que el registre és públic i qualsevol persona pot cercar i consultar informació sobre qualsevol transacció i conèixer els imports que conté qualsevol adreça.
Per una altra part, cada usuari és el propietari i responsable dels fons els quals és propietari, o dit d'una altra forma, dels quals disposa les corresponents adreces. Quan un usuari decideix gastar una part dels seus fons, ha d'utilitzar la seva clau privada per a signar un missatge amb l'adreça de destí i la quantitat de moneda a trametre.

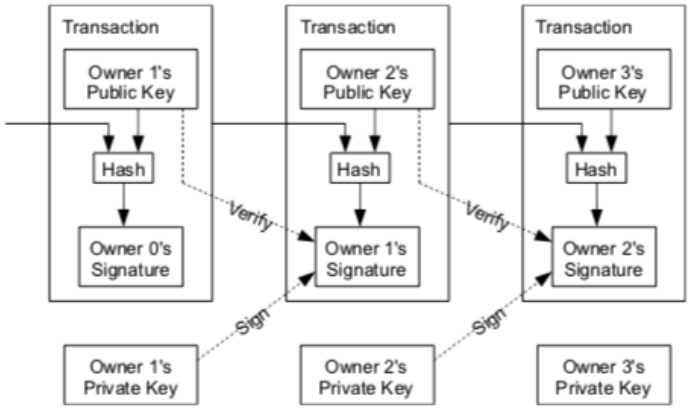
En l'execució d'una transacció[4], l'usuari transmet a la xarxa distribuïda el missatge signat, i es propaga a tots els nodes de la xarxa rebent cadascun d'ells una còpia d'aquesta transacció. Cada node valida de manera independent aquesta transacció i actualitza la copia del seu registre de la base de dades.

2.3.2 Transaccions

En el *Blockchain*, cada adreça és identificada amb una cadena de caràcters generada de forma aleatòria (p.e. 1F1tAaz5x1HUXrCNLbtMFqcw6o5GNn4xqX). Aquestes adreces, són

creades per claus digitals, les quals estan formades per dues parts: una part pública i una part privada. Tenir possessió de les dues atorga la propietat i control de l'adreça. La part pública, és equivalent a l'adreça sobre la qual podrem rebre dades, o en el cas del *Bitcoin*, moneda. La part privada, servirà per a signar les transaccions i realitzar transferències de dades cap a altres adreces.

Figura 4: Esquema de transaccions



Font: *Bitcoin*: A Peer-to-Peer Electronic Cash System

Com s'observa a la Figura 4. La clau privada permet realitzar la signatura de transaccions i la clau pública permet verificar la seva autenticitat mitjançant les funcions Hash. Aquesta transacció podrà esser accessible i visualitzable amb la clau pública.

Amb les claus digitals es podran generar adreces per a rebre dades o moneda, o per a enviar moneda a altres adreces. L'adreça podrà esser compartida públicament i servirà per a rebre moneda sense cap tipus de restricció i des de el complet anonimat. El procés que hi ha sota aquestes transaccions, serà el de la signa "d'acords" digitals entre altres individus i nosaltres. En l'acte de la recepció de tokens a la nostra adreça intervindran diferents nodes que validaran l'autenticitat de la transacció i confirmaran les dades intercanviades, com per exemple: validant que existeix crèdit per a la transferència per part de l'altre individu, que el nostre compte és correcte i que no s'està produint cap tipus de duplicitat en la transferència (amb això, es vol dir que es certificarà que aquest tokens enviats són reals i autèntics, no han sigut generats mitjançant manipulació ni corrupció, i que tampoc són una replicació o còpia d'altres tokens).

De la forma contraria, en el moment en que un individu decideix realitzar una transferència, es requerirà la seva clau publica i clau privada per a l'enviament. Un cop els corresponents nodes han aprovat la transferència i validat els contractes, el token arribarà a la direcció de destí i la quantitat haurà sigut restada de l'adreça de l'individu que envia.

2.3.3 Base de dades

Un exemple molt il·lustratiu per ajudar a entendre el funcionament de la cadena de blocs és el del llibre d'un compte bancari. El registre de les comptes bancaries és el fruit resultant dels valors que el componen, sumant i restant imports durant en temps. Si manipuléssim del registre un número d'una donada data del passat, l'import final

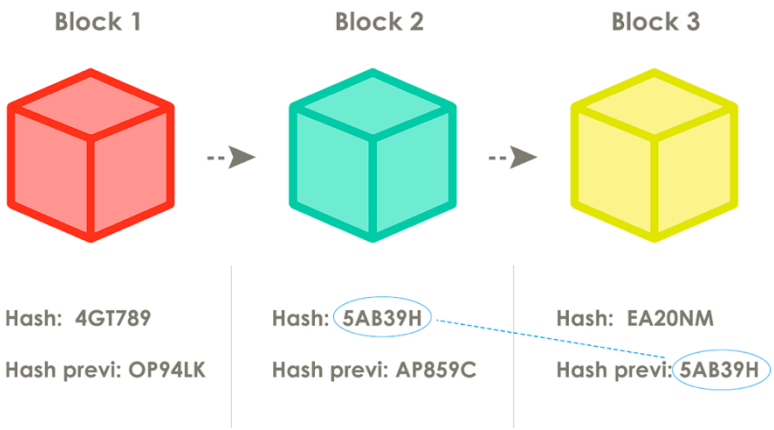
variaria i ja no correspondria amb la realitat, produint una corrupció de les dades que han estat introduïdes després de la manipulació.
La *Blockchain* es basa en aquest principi, però de forma molt mes complexa. De primeres, cada un dels blocs esmentats anteriorment, disposa de 3 parts:

- **Dades**: com la pròpia paraula indica, correspon a la informació registrada entre l'inici i final de la creació del bloc. Depenent de l'aplicació, contindrà un tipus de dades o un altre.

- **Hash**: és un codi unívoc generat de forma aleatòria amb una sèrie d'algorismes que actuen sobre les dades. S'executa una lògica matemàtica sobre les dades en el moment en que el bloc queda creat, que genera un codi únic el qual té un resultat distint en funció de les dades mitjançant les quals ha sigut processat. Per tant, si un cop calculat el hash, les seves dades fossin alterades, el codi hash del bloc deixaria d'esser coincident.

- **Hash del bloc anterior**: emmagatzema el codi resultant del bloc anterior, amb el doble efecte d'identificar quin és el bloc predecessor, i de tenir una copia del seu hash. En cas de manipulació de les dades dins un node del bloc predecessor, es podrà detectar fàcilment amb el bloc contigu.

Figura 5: Esquema cadena de blocs

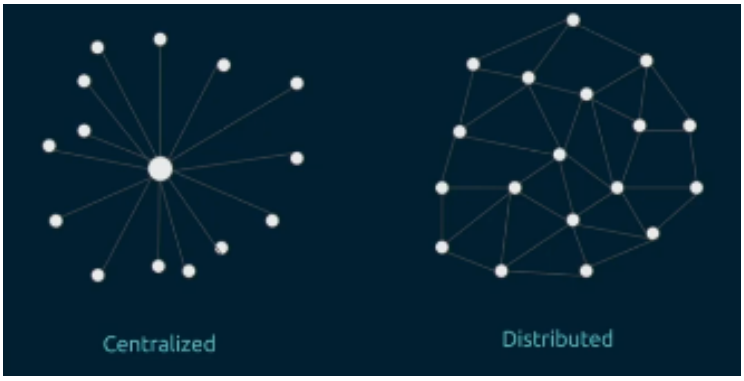


Font: Elaboració pròpia

A la Figura 5 es mostra un exemple del funcionament de la cadena de blocs, en el qual el registre de codis hash lliguen un bloc darrere de l'altre i fan que la seva unió estigui relacionada directament amb les dades que contenen.

Existeixen bases de dades *Blockchain* publiques o privades en funció de la configuració i necessitat del corresponent projecte o D-App. Totes dues, comuniquen els seus nodes mitjançant la tecnologia P2P(Peer-to-Peer)[4], el qual és un protocol de compartició i distribució de dades. Tenint la xarxa torrent com a referència, s'encarrega de la propagació de dades entre diferents nodes, on no existeix un únic servidor, sinó que tots els punts de la xarxa descarreguen i pugen dades de forma simultània. En qualsevol dels casos, l'adhesió d'un nou node a la xarxa implicarà la descarrega de tota la cadena de blocs existent entre tots els nodes i la seva constant verificació.

Figura 6: Xarxa Centralitzada i Xarxa Distribuïda



Font: Captura de InterPlanetary FileSystem (IPFS) Tutorial (Youtube)
https://www.youtube.com/watch?v=6kqgsGXpykM

La Figura 6 mostra dos models diferents en els quals s'aprecien visualment les diferencies entre ambdues xarxes, deixant clara la situació de no-centralització en la part distribuïda. El diagrama de l'esquerra podria representar un sistema client servidor, i el de la dreta una xarxa P2P.

La creació de cada bloc dura un temps aproximat de 10 minuts[4], independentment del nombre de transaccions que s'hi registrin. Un cop un node ha creat un bloc, procedeix a difondre'l a la resta de nodes mitjançant la tecnologia P2P. Cada node rebrà el corresponent bloc, i gràcies als potents processadors que existeixen a dia d'avui, seran capaços de verificar el seu hash i confirmar les seves dades en un període de temps molt curt. A més, al disposar el bloc del hash del bloc anterior, serà fàcilment ordenat i concatenat dins de la cadena de base de dades. No obstant, un cop verificat el hash, hi haurà una comunicació entre nodes anomenada "consens", que serà la qui validi la integritat d'aquest, que succeirà en el moment en que el 51% de la xarxa l'aprovi.

2.3.4 Seguretat

La característica distribuïda de la cadena de blocs, és un gran avencs que denota una gran potència, però al mateix temps deixa entreveure possibles vulnerabilitats o oportunitats d'infiltració per part d'agents externs.
La necessitat de connexió i intercanvi de dades entre nodes és imprescindible per al correcte funcionament del sistema distribuït. El fet que hi hagi comunicació entre dos o més punts, augmenta les possibilitats d'atacs del tipus suplantació d'identitat, d'intents de manipulació de missatges o de dades i d'exposició tant a la pèrdua o apoderament indegut de les claus que atorguen la propietat de les dades per part d'un individu.

Per a fer possible l'existència d'una xarxa segura i compartida, la *Blockchain* es nodreix d'una combinació d'algorismes i tecnologies que la fan segura i molt difícilment corrompible. A continuació s'enumeren un conjunt de mecanismes que fan possible les sever virtuts:

SHA2
Un algorisme Hash[5] consisteix en un seguit de càlculs matemàtics sobre un conjunt de dades. El resultat generat per part de l'algorisme matemàtic dona com a resultat una cadena codificada de longitud fixa.
Aquesta cadena resultant, varia en funció de les dades d'entrada a l'operació. L'operació és unidireccional, això implica que el hash obtingut és únic i depèn completament de les

dades i que, al no existir cap clau no és possible la descodificació de les dades ni tampoc cap tipus d'operació inversa.

Des de el 2001 que va esser creat, han anat sorgint millores i evolucions en les funcions per tal de poder seguir sent usables davant el descobriment de vulnerabilitats. En la cadena de blocs s'utilitza per a la validació la SHA256, amb la qual encara no s'han trobat col·lisions

Gràcies a aquesta tecnologia i l'aplicació al *Blockchain*, és possible validar l'autenticitat i integritat de qualsevol bloc de dades per part dels nodes de la xarxa. Disposant del valor de la cadena hash generada en el moment de la creació, amb la obtinguda en qualsevol moment, serà possible efectuar la verificació. En aquest cas, cada bloc registrarà el valor hash del bloc generat anteriorment.

Sistemes de claus
Per a garantir la seguretat de les dades així com la privadesa és necessari un sistema de claus.
De forma lògica, en el *Blockchain* el propietari de les dades ha de tenir la capacitat de poder visualitzar-les i modificar-les les mateixes. Gràcies al us de la criptografia serà possible signar les dades (clau privada) atorgant el seu control total i obtenir la seva informació (amb la clau pública).

El sistema esta pensat en que cada adreça estigui formada de dues parts, on només el propietari disposarà d'ambdues per a tenir el control total. En cas que la clau privada fos extraviada, serà possible la seva recuperació mitjançant la possessió de la clau privada. En el cas contrari, serà impossible la recuperació d'una clau privada ja que el seu us és unidireccional i la base de dades únicament les genera sense encarregar-se del seu emmagatzematge.

Aquesta ultima propietat, fa que el sistema sigui molt mes robust davant atacs ja que tot el control depèn de les claus privades, i aquestes no es trobaran en cap cas a la cadena de blocs.

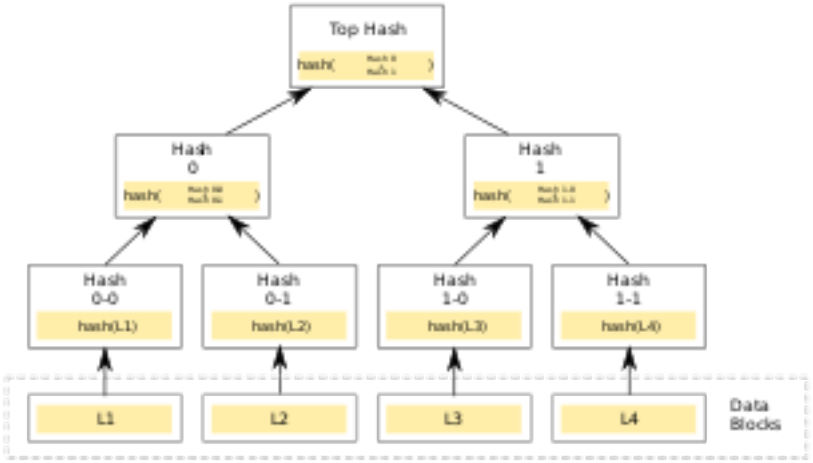
Amb el sistema de claus públiques i privades, és possible xifrar i signar les dades que són emmagatzemades a la *Blockchain* per part del emissor, com per exemple transaccions o *smart-contracts*. D'aquesta mateixa manera, els nodes podran validar la seva autenticitat amb l'ús de la clau pública per tal de poder d'escriure les corresponents operacions en el seu registre.
També cal anotar, que la clau pública permetrà la visibilitat de tot el registre de dades i actuarà en forma d'adreça.

Arbre de Merkle[6]
Consisteix en un sistema d'organització de blocs de dades en forma piramidal, amb l'objectiu de facilitar la verificació de les dades i reduir els recursos emprats en la validació de la integritat.

Està dissenyat de forma que de cada bloc hi penja una parella de blocs, on cadascun d'ells emmagatzema el codi hash del bloc predecessor. Tots ells penjaran d'un bloc gènesis, inicial o principal, que marcarà l'inici de la base de dades.

Amb aquest sistema, s'aconsegueix validar la informació de forma molt més eficient respecte a una organització lineal.

Figura 7: Arbre de Merkle



Font: https://es.wikipedia.org/wiki/Árbol_de_Merkle

Com es mostra a la Figura 7, aquesta distribució de blocs estalvia la quantitat de passos que cal recórrer per a validar la integritat d'un bloc concret, evitant haver de processar la integritat de tota la cadena en favor de validar únicament la seva branca corresponent fins arribar al bloc gènesis.

2.4 El bloc gènesis

Com s'ha vist a l'apartat anterior, la estructuració en blocs de la cadena en forma d'arbre obligarà a l'existència d'un bloc principal. Aquest, no tindrà predecessor i serà el primer bloc generat en la xarxa. Les peculiaritats[7] que té respecte a la resta faran que tingui unes característiques especials, com per exemple:

- L'anàlisi de la cadena de predecessors de qualsevol bloc de la base de dades sempre tindrà com a destí final el bloc gènesi.
- Garanteix la comunicació entre nodes, fent indispensable que qualsevol node de la xarxa tingui com a mínim una còpia del primer bloc.
- Cada bloc gènesis de cada cadena de blocs serà diferent. Podran existir dues cadenes de blocs en la mateixa xarxa disposant cadascun dels seus nodes amb el seu respectiu bloc gènesi.

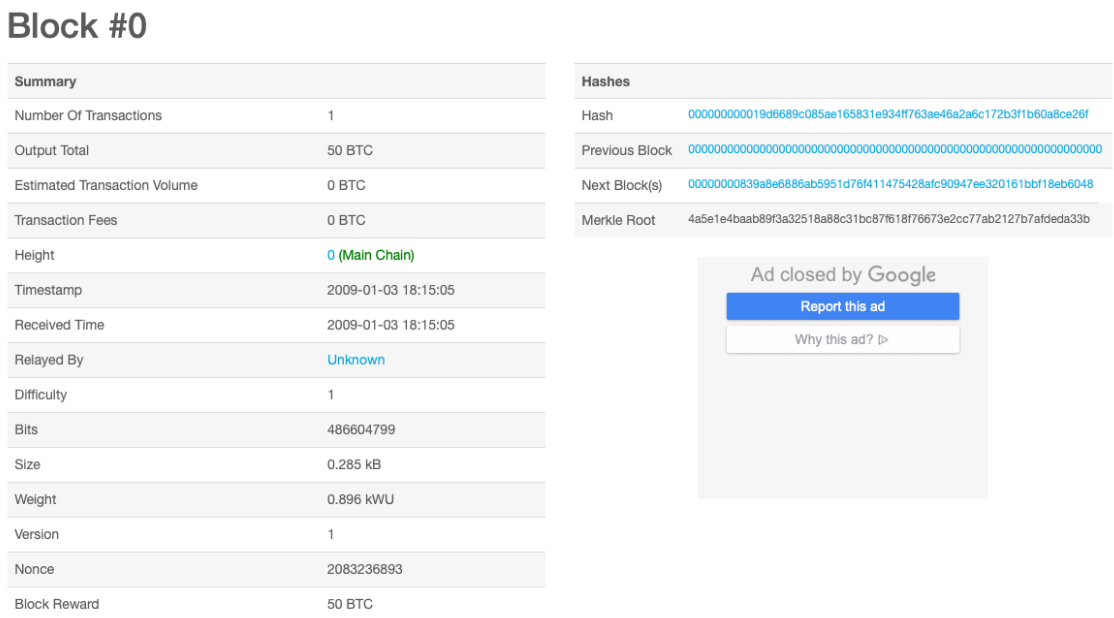
Respecte al projecte *Bitcoin*, en cara que es tractarà amb concreció en el següent apartat, hi haurà una sèrie de curiositats respecte al bloc inicial i el misteri que envolta al seu creador que és interessant mencionar.

En un primer lloc, es pot demostrar que va ésser generat en una data no abans del 3/01/2009, degut a que en el seu coinbase conté la frase "Chancellor on brink of second bailout for banks". Traduït significa que es va introduir el text d'un dels titulars de la portada del diari The Times d'aquell dia, es creu que com a reivindicació d'avant els moviments de rescat a la banca.

Una altra dada curiosa, és que el bloc conté la primera adreça creada, la qual posseeix la quantitat de 50 bitcoins (uns aproximadament 400.000 euros en el dia en que s'han

escrit aquestes línies), i que estranyament i sense motiu ni explicació, l'algorisme no permet el seu moviment.

Figura 8: "*Bitcoin* Block 0"



Block #0

Summary	
Number Of Transactions	1
Output Total	50 BTC
Estimated Transaction Volume	0 BTC
Transaction Fees	0 BTC
Height	0 (Main Chain)
Timestamp	2009-01-03 18:15:05
Received Time	2009-01-03 18:15:05
Relayed By	Unknown
Difficulty	1
Bits	486604799
Size	0.285 kB
Weight	0.896 kWU
Version	1
Nonce	2083236893
Block Reward	50 BTC

Hashes	
Hash	000000000019d6689c085ae165831e934ff763ae46a2a6c172b3f1b60a8ce26f
Previous Block	0000000000000000000000000000000000000000000000000000000000000000
Next Block(s)	00000000839a8e6886ab5951d76f411475428afc90947ee320161bbf18eb6048
Merkle Root	4a5e1e4baab89f3a32518a88c31bc87f618f76673e2cc77ab2127b7afdeda33b

Font: https://www.blockchain.com/btc/block-index/14849

La figura número 8 mostra el contingut del primer bloc. La seva informació, encara i contenir dades que podrien considerar de caire privat, és completament pública i està a disposició per a la seva validació. La generació d'aquest bloc va ésser especial, ja que com es pot observar, el hash del seu bloc anterior és una cadena de zeros.

En el següent punt es tractaran els aspectes generals que afecten en general al projecte principal de la posada en marxa de la tecnologia *Blockchain.*

2.5 Projecte principal: *Bitcoin*

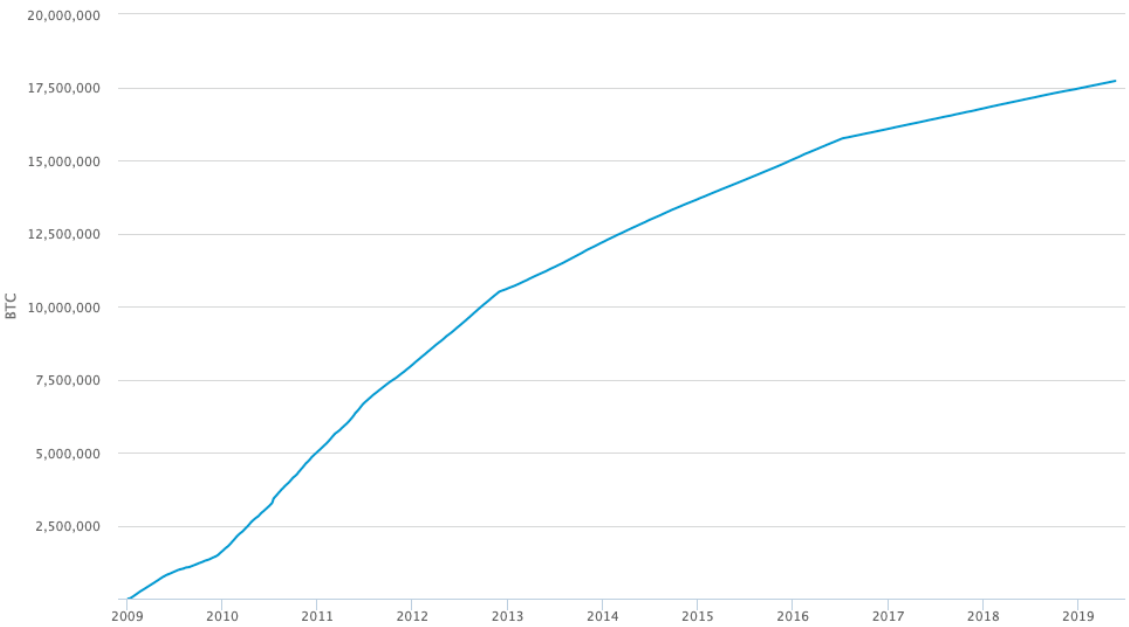
Aquest projecte, és considerat el pare i el punt inicial de la revolució generada per la tecnologia de cadena de blocs. Com el seu nom indica, podríem definir el *Bitcoin* com un llibre comptable i transparent distribuït mitjançant una xarxa P2P, basat en una idea o algorisme *open-source* i que fa ús de la seva pròpia divisa per a les transaccions. La moneda, el *bitcoin* (en minúscula), no és controlada per cap institució i únicament està respatllada pel seu algorisme i la seva robustesa.

El seu origen és avui dia una incògnita[3]. El poc que es sap és que el seu creador (o creadors) és amagat sota el pseudònim de "Satoshi Nakamoto", que en el 2009 va llençar el software *Bitcoin*, creant la xarxa que duu el mateix nom i generant les primeres unitats de moneda.

El *Bitcoin*, té la seva pròpia política monetària, que curiosament xoca de ple de forma contraria amb la filosofia FIAT. En el model capitalista, els bancs prenen decisions després de les avaluacions que van obtenint sobre l'estat de l'economia en casos de inflació, injeccions de capital, etc, en funció dels interessos del moment.

Per contra, el *Bitcoin* té establert que la base màxima de moneda serà de 21.000.000 de bitcoins. Aquesta divisa serà generada (minada) de manera planificada i previsible en el temps.

Figura 9: Històric bitcoins en circulació



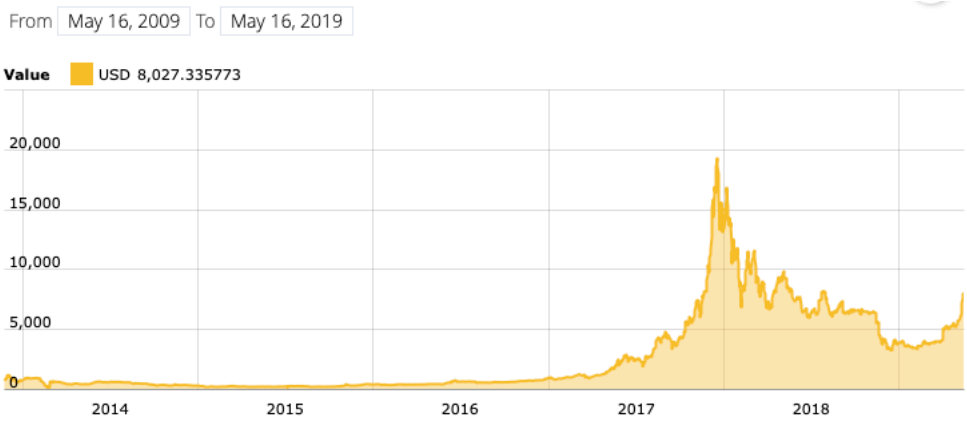
Font: https://www.blockchain.com/charts/total-bitcoins?timespan=all

En la Figura 9 es pot observar com la major part dels *tokens* ja han sigut produïts i que seguint la línia cap al límit, cada cop és més alta la dificultat d'aconseguir produir-ne de nous.
En la primera fase després del llançament del projecte, els *bitcoins* són generats pel que es denominen "miners", que en la practica consisteixen en aquells nodes que cedeixen la seva capacitat de processament com a recursos a la xarxa, per a la validació de transaccions i la realització de càlculs necessaris. Amb aquesta operació, s'aconsegueix donar un valor a la moneda que es rep com a premi (recompensa per l'esforç de processament), i genera un interès a què mes miners s'uneixin a la causa incrementant la seguretat i el poder computacional de la xarxa.

La quantitat de *bitcoins* és limitada, cada 210.000 es redueix a la meitat la recompensa rebuda per un miner al assegurar un bloc. Això significa que aproximadament cada 4 anys es decreix a la meitat el volum de *tokens* que es generen, no obstant, les transaccions entre usuaris poden opcionalment viatjar amb una comissió. Aquestes comissions de transferència seran repartides entre els miners i, en el moment en que es generin els 21.000.000 de bitcoins, passaran a ésser el seu sustent.

Com a punt de curiositat: Entre la meitat de 2017 i la meitat de 2018, aquesta criptomoneda va patir un gran creixement en quant al seu valor, i una forta inflació del seu preu generada per la gran expectació i interès. A dia d'avui, tal i com s'aprecia en el següent gràfic, el seu preu sembla estable després de la correcció, encara que en el moment de la redacció d'aquest document està començant a iniciar un altre cop un camí alcista.

Figura 10: Històric cotització *Bitcoin*



Font: https://es.cointelegraph.com/bitcoin-price-index

3. Tecnologies distribuïdes

3.1 Ethereum

3.1.1 Descripció del projecte

Ethereum, es pot considerar com el segon projecte mes important que fa ús de la tecnologia de cadena de blocs distribuïda. Està totalment basat en el *Bitcoin* i en el seu algorisme, i al igual que ell, també té la seva pròpia criptomoneda, que en aquest cas és anomenada *ether*. No obstant, encara que hi ha una gran similitud de primeres amb el *Bitcoin*, *Ethereum*[8] es considerada la seva evolució, ja que a més de la part de divisa, fa un pas més enllà fent possible la seva implementació com a plataforma d'execució de contractes intel·ligents i d'aplicacions descentralitzades.

En aquest cas, el seu creador té un nom real: Vitalik Buterin, que juntament amb altres cofundadors com són Gavin Wood i Joseph Lubin, van aconseguir tirar endavant el projecte. El sorgiment del projecte va ésser al Desembre de 2013, on el primer bloc minat data de Juliol de 2014, per tant, és considera molt recent.

En el cas de l'ether i en quant a les seves normes de control, admet la generació de 18.000.000 per any, és a dir no té límit de creació. Per tant, serà una moneda inflacionària que tampoc estarà gestionada ni administrada per cap sistema central. El seu sustent, i el seu minatge de token es basarà en l'esforç que realitzaran els miners per a mantenir la xarxa segura i fer possible les transaccions.

3.1.2 Diferències respecte a la resta de projectes

Fins aquí, el projecte *Ethereum* té força similitud amb el projecte *Bitcoin*. Però fora de les seves semblances, existeix una particularitat que suposa un pas endavant, i és el fet de posar a l'abast de la comunitat un nou ventall d'utilitats per a la implantació de projectes i aplicacions descentralitzades.

Continuant amb el concepte de transaccions que ofereix *Bitcoin*, on la xarxa validava transaccions purament monetàries del seu propi *token*, *Ethereum* implementa una variació fent que sigui possible la validació i intercanvi de qualsevol dada[10]. Per tant, adquireix la capacitat d'usar la cadena de blocs per a emmagatzemar dades completament customitzables gaudint de la mateixa seguretat que les cripto.

Amb el comentat en el paràgraf anterior, es crea la possibilitat de la firma de contractes intel·ligents i posa en mans dels desenvolupadors el poder de programar i implementar projectes distribuïts sense limitacions.
Els *smart-contracts*, seran desenvolupats en un llenguatge anomenat Solidity (posteriorment utilitzat en aquest projecte), amb una sintaxis senzilla, intuïtiva i semblant a la majoria de llenguatges. Solidity és la capa que permet la comunicació entre les D-App i la *Blockchain* mitjançant el resultat de la interpretació dels mètodes i lògiques implementats en forma de contracte.

Solidity, és un llenguatge basat en la orientació a objectes, que permet la creació de classes així com de mètodes al ús.

3.2 Contractes intel·ligents

El sentit del *Blockchain* no acaba només en els projectes que utilitzen la seva tecnologia: més bé al contrari. Els projectes que beuen del seu algorisme són el punt d'inici a un món de possibilitats d'adaptació i personalització digital distribuïda.

El lloc de partida de les *D-App*, és el poder explotar l'ús dels projectes existents, o de desenvolupar projectes descentralitzats privats. La cadena de blocs distribuïda i la tecnologia aplicada al consens, fa possible que amb una tecnologia open-source de baix cost d'implantació sigui possible l'adopció d'un nou ecosistema d'aplicacions i de possibilitats esperant a ésser creades.

El concepte de *smart-contracts* (contractes intel·ligents), és un mecanisme que utilitzen forçosament els projectes *Blockchain* descentralitzats amb la seva signatura d'acords. A continuació es mostren alguns exemples[9] on el desenvolupament de D-App podria ésser profitós i tenir una cabuda assegurada:

Implementació de serveis de registre per a notaries
El *Blockchain* per aquest cas, faria l'acció llibre de registre de les signatures de contractes per part dels corresponents implicats. El funcionament ideal seria que per cada documentació signada quedés total constància del fet/acció acordat, els actors, el lloc, dia i la hora. La descentralització faria possible la fàcil consulta de les dades per totes les parts i podria garantir de immutabilitat.

Sistemes electorals
En aquesta situació, el *Blockchain* seria l'autoritat que registra i valida el vot emès per cada individu, facilitant el sistema de recompte i garantint 100% la confidencialitat i la no duplicitat dels vots. Aquesta situació milloraria de lluny el sistema electoral actual que obliga a un recompte manual de les paperetes i a la presencialitat de la votació.

Sistemes d'emissió de certificats
En aquesta situació idíl·lica, el *Blockchain* tindria la tasca d'emmagatzemar els estatus d'emissió, visualització o execució d'accions concretes entre usuaris i les administracions. Per exemple: pagament de multes, emissió de certificats, tramitacions,...
Respecte al sistema actual milloraria en quant a agilitat els diferents tràmits i ajudaria a la simplificació de les gestions sense deixar de banda les garanties habituals.

Sistemes *cloud* distribuïts
Per aquets últim exemple, la cadena de blocs permetria disposar d'un registre de les accions realitzades amb fitxers i arxius per part dels usuaris mitjançant el registre de la seva direcció o adreça de descàrrega, i en integració amb altres sistemes de distribució.

3.3 IPFS

Es un protocol que permet la compartició d'arxius[12] mitjançant la tecnologia Peer-To-Peer. és molt similar al concepte WWW d'Internet, amb la diferencia de que no existeix la idea de web amb model client (realitza la petició i consumeix les dades) i servidor (disposa el servei) bidireccional. Genera una xarxa en la qual els diferents nodes poden compartir arxius i accedir a les dades posades a disposició per la resta d'integrants.

El funcionament és molt simple. Primerament qualsevol usuari o client pot realitzar la publicació dels arxius desitjats cap a la xarxa prèvia instal·lació del corresponent aplicatiu i l'execució de la instrucció indicada. A partir d'aquest punt, aquesta informació queda a disposició dels nodes que fan l'acció de processar el corresponent arxiu generant-li un identificador únic (una cadena de caràcters), posant-los a disposició des de aquest moment a la resta de la xarxa. Qualsevol integrant de la xarxa que disposi de l'identificador de l'arxiu podrà tenir accés a aquests arxius i descarregar-los realitzant una copia. En el cas que un altre node demani aquest arxiu, el començarà a descarregar des de tots aquells nodes de la xarxa que en disposin d'ell.

Com a la resta de protocols distribuïts, el poder d'emmagatzemament ara ja no es troba concentrat en un únic punt, si no que forma part i és responsabilitat de la xarxa, amb la qual cosa, no existeix dependència sobre un servidor com en els sistemes centralitzats.

Aquesta tecnologia, permet ser una possible alternativa a la centralització de la concepció web, mantenint una constant actualització de les dades entre els nodes i oferint com a virtut una major garantia davant caigudes de servei, que els sistemes centralitzats tenen difícil pal·liar.

3.4 Concepte de D-App

Podríem definir-les com aquell conjunt d'aplicacions capaces d'explotar una base de dades distribuïda.
Utilitzen un principi semblant al de les aplicacions centralitzades comunes, amb la diferencia principal en la lògica de comunicació amb la xarxa.

Les aplicacions client/servidor, implementen una sèrie de protocols i fan ús de la xarxa per al transport de dades i la comunicació. Estan pensades per a treballar de punt a punt mitjançant missatges.
En el cas de les D-App, estan pensades per a fer ús de la xarxa per al processament, per a la seguretat, per a la comunicació, per a l'emmagatzematge, etc. Estan basades en xarxa. la xarxa són els nodes, i els nodes són tot el conjunt de clients, servidors i actors que contribueixen a la seva garantia i seguretat treballant alhora.

A l'hora de dissenyar i pensar una D-App, cal tenir en compte una sèrie de factors per tal de fer òptima la seva eficiència. No totes les aplicacions tindran el mateix comportament de forma final, realitzar la mateixa tasca, ni tampoc produir el mateix benefici. Hi ha aplicacions que contribueixen a l'emmagatzemament, n'hi ha que contribueixen al registre, n'hi ha que el seu punt fort és el càlcul i unes altres es basen únicament en la comunicació.

En funció de la funció prevista a desenvolupar, ens interessarà fer ús d'un tipus de tecnologia distribuïda o d'un altre.
Per exemple, l'ús de la tecnologia distribuïda *Blockchain* pot ésser molt favorable per al manteniment i emmagatzematge de dades de control d'una D-App, com per exemple una base de dades que emmagatzema instruccions o un registre d'operacions. En canvi per una altra part, serà una mala elecció en el cas de voler utilitzar-la per a l'emmagatzematge d'arxius, ja que l'elevat pes de dades d'aquest tipus, endarrerirà enormement l'encriptació i augmentarà les necessitats de processament de la xarxa provocant una exagerada lentitud.

Una possible solució per aquest exemple, serà l'ús del *Blockchain* conjuntament amb una altra eina d'emmagatzematge especifica, que doni com a resultat una D-App fruit d'una simbiosi de tecnologies.

A continuació es descriuen els usos freqüents de tres tecnologies que unides conformen per separat les principals necessitats d'un servei computacional: processament, seguretat i emmagatzematge. La intenció, és mostrar com la unió de serveis descentralitzats poden realitzar la mateixa tasca que realitzaria un servidor únic

MPI Message Passing Interface
En el cas de necessitar implementar una aplicació de processament distribuïda capaç d'aprofitar conjuntament el potencial de càlcul de diferents nodes, podria ésser interessant l'ús del software MPI[11].

MPI és un protocol de comunicació entre maquines, que entre altres està suportat en llenguatge C i C++, que permet l'execució de processos en paral·lel realitzant un clúster de processament entre les maquines desitjades.

És un projecte molt interessant que data del 1993, i que el seu principal punt fort és la robusta comunicació entre màquines mitjançant l'inici de sessions on imperen l'intercanvi de missatges amb les dades del processament en temps real.

Blockchain
Es tracta clarament de la tecnologia més de moda i sobre la qual s'ha parlat extensament al llarg d'aquest treball. Resumidament, el seu punt fort és la seguretat i la robustesa de les dades, amb la seva invariabilitat garantida.

IPFS InterPlanetary File System
A part d'aportar la funcionalitat de distribució d'arxius i de facilitar l'accés a la aplicació realitzant la seva pròpia distribució, també podrà ser una tecnologia útil per a la distribució i gestió de continguts des de ella mateixa. Un exemple seria la possibilitat de crear una xarxa distribuïda de vídeo, on els usuaris poguessin accedir a l'aplicació mitjançant el propi protocol IPFS, i també realitzar la pujada de les seves filmacions per tal que fossin accessibles i distribuïdes a qualsevol demandant de la xarxa.

D'aquesta forma, aquesta tecnologia és aprofitable per a obtenir ús per partida doble, de forma similar a l'actuació que tindria en una xarxa centralitzada un servidor que ens carregués un lloc web i també permetés allotjar els nostres arxius, però sense brindar les virtuts que ofereixen les xarxes distribuïdes.

4. Part tècnica

4.1 *Kick off*

Iniciant la part experimental del projecte, s'ha decidir realitzar el desenvolupament d'una aplicació per tal de posar a prova els conceptes coneguts a la part teòrica. Després de la valoració de la decisió de la temàtica, s'ha determinat que l'opció més completa es la de desenvolupar una D-App de tipus Web que requerís de l'ús de les tecnologies Blockchain, IPFS i Javascript. Les tres són tecnologies que principalment s'executen de forma nodal o en client, la qual cosa permet la independència en l'oferiment de serveis enfront l'ús de servidors.

Per aquest cas, s'ha pensat en un desenvolupament que requerís la interacció entre l'usuari i la base de dades, i també la de l'usuari amb la xarxa IPFS. Per tant, la elecció més directa, és la confecció d'un sistema gestor de continguts que, de pas sigui dit i gràcies a les seves característiques, pogués fer també les accions de servei *cloud*.

Un cop aclarida la temàtica, s'ha optat per la implementació d'una solució informàtica capaç d'aportar un servei de registre i emmagatzematge de documents enfocada en un principi en món empresarial. El propòsit, és que el nou aplicatiu pugui esser utilitzat per a la compartició de dades per part d'empreses amb seus i plantes de fabricació d'escala mundial.

Per al cas, no totes les seus disposen un accés a la xarxa d'alta velocitat i amb garantia davant talls en la connexió, per tant, la necessitat de que l'aplicació sigui descentralitzada és un requisit, fent inviable l'ús de qualsevol sistema centralitzat.

La funcionalitat de l'aplicació, permetrà el registre d'usuaris, la pujada de fitxers a la xarxa generant un registre històric de la biblioteca virtual, i una petita secció per a la gestió de permisos.

4.2 Anàlisi funcional

De principi, és necessari que l'aplicació sigui accessible mitjançant qualsevol navegador web per tal de garantir la compatibilitat amb qualsevol sistema.

Per a la descripció de la lògica de funcionament, s'iniciarà amb l'explicació de les diferents tipologies d'usuaris necessaris per a la gestió. La D-App serà d'us exclusiu per part dels empleats de l'empresa. Les regles inicials de funcionament marquen necessària l'existència de 3 tipus diferents de rols d'us:

Perfil Administrador
L'administrador serà l'usuari amb majors privilegis de l'aplicació. Tindrà la capacitat de visualitzar tots els documents del sistema. També serà l'encarregat d'atorgar o revocar privilegis a la resta d'usuaris en funció de la necessitat.

En el *smart-contract* quedarà definit com *Admin*.

Perfil Usuari
Serà el perfil d'usuari convencional, amb la capacitat de visualitzar, descarregar, pujar arxius al sistema.

En el *smart-contract* quedarà definit com *Writter*.

Perfil de visualització
Aquest rol, únicament tindrà permisos per a veure el registre de documents, amb la opcionalitat d'efectuar la seva descàrrega. Al contrari del perfil usuari, no podrà ni realitzar carregues d'arxius ni tampoc esborrar.

En el *smart-contract* quedarà definit com *Viewer*.

Els rols quedaran enregistrats al *Blockchain* i hi existirà el corresponent bloc de dades que guardarà la relació entre usuari i rol. Aquesta classificació serà la que farà servir el *front-end* per a la gestió dels components i vistes a mostrar depenent del permís.
Per a evitar un ús indegut per part de possibles atacants, en el moment del registre els usuaris rebran automàticament el rol amb menors privilegis. Haurà d'esser l'administrador posteriorment, qui de forma manual atorgui el rang correcte a cada usuari.

Tant la informació sobre els rols (la denominació), com dels usuaris i la dels fitxers quedaran registrades en arrays d'estructures al *smart-contract*. Aquesta definició permetrà organitzar la informació i accedir-hi de forma simple amb l'adreça d'assignació. Cada estructura contindrà les següents dades:

Rols: Nom del rol
Usuaris: Nom de l'usuari, adreça, data de creació.
Arxius: Nom del fitxer, hash IPFS de l'arxiu, adreça de l'usuari que l'agrega i data d'enregistrament.

En quant a la part d'enregistrament de fitxers, s'opta per la implementació de la càrrega d'arxius sobre xarxa IPFS. Com s'ha comentat en el punt 3.4 de la memòria, encara que és possible emmagatzemar de forma binaria el contingut d'un fitxer, la càrrega de dades d'elevat pes suposen un gran cost per al processament de l'emmagatzemament, per tant s'encomana la tasca a IPFS atorgant-li la funcionalitat de difusió del lloc web i de distribuïdor de matèria carregada.

Finalment, per a l'administració de rols es requerirà d'una vista senzilla que llisti tots els usuaris del sistema i permeti el canvi de modalitat de forma simple.

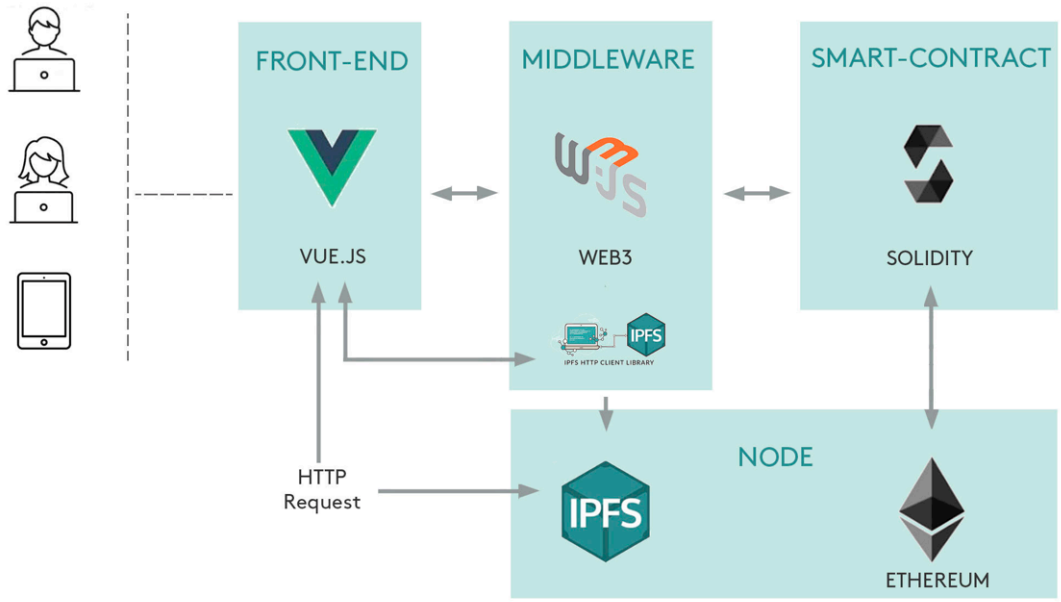
4.3 Tecnologies i eines aplicades

Després de l'anàlisi, i de la definició sobre l'aplicació a desenvolupar, s'ha realitzat una recerca sobre les eines viables per a dur a efecte l'execució del projecte.

Es veu amb claredat, que l'aplicació constarà d'una combinació d'eines que faran possible l'aplicació distribuïda.
En la següent taula és resumeixen les diferents capes de l'aplicació i l'eina encarregada:

Requeriment	Tecnologia/App
Base de dades	*Blockchain*
Sistema gestor de la base de dades	Ethereum
Base de dades per al entorn de desenvolupament	Ganache
Framework *smart-contracts*	Ethereum Solidity
Compilador de contractes	Truffle
Enllaç entre *back-end* i *front-end*	Web3
Framework web *front-end*	Vue.JS
Desplegament	IPFS

Figura 11: Gràfic de la Arquitectura de la D-App



Font: Elaboració pròpia

Com es veu en el diagrama de la Figura 11, es mostra la interacció entre les tecnologies i eines aplicades. En primer lloc els usuaris interactuen amb la capa *front-end*, que de primeres serà proporcionada per el node IPFS. Aquesta serà la que està en comunicació de forma directa amb el ventall de serveis i redirigirà el transit en funció de la necessitat d'interacció segons escaigui amb la base de dades, o amb la gestió d'arxius.
Per tant, en funció del requisit de la petició dirigirà el tràfic cap al corresponent *middleware* que s'encarregarà de connectar amb el corresponent node, on en el cas de *Ethereum* caldrà realitzar la peticions amb les consignes programades al *smart-contract* de la forma que es veurà més endavant.

En els punts que es troben a continuació es realitza una breu explicació amb informació sobre algunes de les eines que seran necessàries.

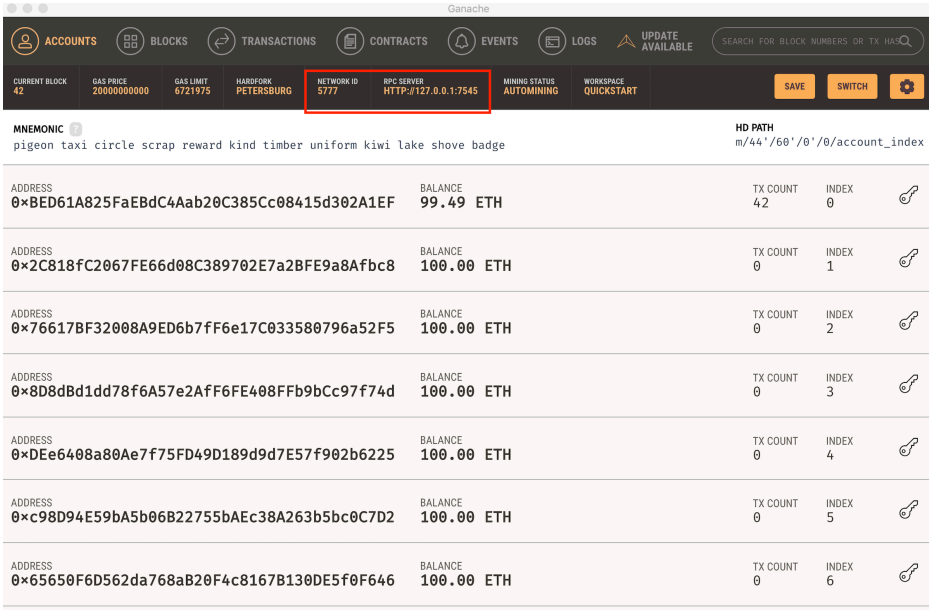
4.3.1 Ganache

Ganache[13] és una aplicació multiplataforma que proveeix d'un node d'Ethereum. Les avantatges que ens aporta entre d'altres són la seva senzillesa i que resulta molt

intuïtiva d'executar, de forma que preconfigura un node privat des de cero, preparat per a connectar amb la nostra aplicació.
A part, la seva GUI ens permet visualitzar i analitzar les transferències que han corregut sobre la *Blockchain*.

El seu ús està destinat bàsicament a donar suport durant el desenvolupament facilitant la vida al programador i evitant la tasca de gestionar un node convencional mitjançant les comandes Geth estalviant temps i recursos durant les proves.

Figura 12: Aplicació Ganache



Font: Elaboració pròpia

Com es mostra a la Figura 12, el software ofereix la possibilitat de visualitzar les diferents adreces de la xarxa generada, i les dades per a la connexió des de aplicatius externs per a la interacció. També permet des de la pestanya "transactions" visualitzar les diferents connexions rebudes per al registre de dades.

4.3.2 Truffle

Truffle[14] és una aplicació que proveeix d'un entorn de treball per al desenvolupament de codi *back-end* basat en el llenguatge Solidity.
Després de la seva instal·lació i del desenvolupament del corresponent contracte, farà possible la compilació i migració a codi JSON per a la interpretació de les instruccions per part del *front-end*.

En el punt 9.1 dels annexos es detalla el procés seguit per a la instal·lació i la creació del corresponent projecte.

Geth: és l'aplicatiu genuí d'Ethereum que permet configurar i executar un nou node. Els seus comandaments seran del tipus "geth version"

4.3.3 Web3

Es una llibreria[15] basada en Javascript que fa d'enllaç entre el node Ethereum i el front end. és la capa que proporciona un seguit de mètodes per a fer possible tant la connexió com la comunicació i interacció entre les dues parts.

4.3.4 Vue.JS

Es un framework[16] web de tipus *front-end* basat en Javascript que s'encarrega d'aportar i renderitzar a codi Javascript les seves instruccions. Es basa en Angular JS i el seu objectiu és el de simplificar la connexió entre les vistes i els components programables estalviant temps i línies de codi i fer-les reactives al màxim.

Accepta la instal·lació de dependències que augmenten la seva potència i la interpretació del codi.

4.3.5 IPFS

Es de forma probable la part mes fonamental de l'aplicació i encara que ha sigut explicada al apartat 3.3, és important esmentar que en el nostre cas s'encarregarà de la difusió de la D-App i de la distribució dels fitxers carregats per part dels usuaris.

4.4 Preparació de l'entorn i desenvolupament

En aquets apartat es detallaran els passos seguits per al desenvolupament del ecosistema de l'aplicació. En primer lloc es farà un detall sobre la part del *back-end*, a continuació del *front-end* desenvolupat, i finalment un detall sobre el node IPFS.

4.4.1 *Back-end*

Aquesta part fonamental ha sigut desenvolupada mitjançant el llenguatge Solidity[17]. Com ja s'ha introduït anteriorment, l'eina utilitzada per a la realització d'aquesta part del projecte ha sigut Truffle, per la qual la seva part d'instal·lació i configuració ha quedat recollida en el punt 9.1 de l'annex.

A partir d'aquí, s'ha desenvolupat el corresponent arxiu anomenat DFileCloud.sol. Aquesta peça de codi conté una sèrie de variables i mètodes que fan possible la interacció amb la base de dades. La definició dels elements queda de la següent manera:

- **Variables i tipus de dades**

Primerament es declaren sobre variables constants els corresponents rols que utilitzarà l'aplicatiu. Per aquesta, s'utilitzarà un tipus *bytes32 constant*, capaç d'emmagatzemar text alhora que atorguen valors fixos a les variables:

```
bytes32 constant ac_admin = "Admin";
bytes32 constant ac_writter = "Writter";
bytes32 constant ac_viewer = "Viewer";
```

En quant als requisits per a l'emmagatzemament dels registres de dades, es realitza la següent declaració de tipus *struct* que conformaran les estructures de dades:

```
// User structure
struct User {
    string name;
    address wAddr;
    uint createdAt;
    uint updatedAt;
}

// Role structure
struct Role {
    bytes32 role;
}

// File structure
struct File {
    string name;
    string ipfs_hash;
    address addedBy;
    uint addedAt;
}
```

Per la part de les variables, serà necessària la declaració de dos elements del tipus mapping, que s'encarregaran de relacionar els Id d'usuari amb l'adreça, i l'adreça de cada usuari amb el corresponent rol. Per una altra part, es declararan els arrays per a emmagatzemar la informació de les estructures i finalment una variable que servirà per a controlar la creació del primer usuari:

```
mapping (address => uint)    public gt_usersId;
mapping (address => bytes32) public gt_usersRole;

User[] public gt_users;
Role[] public gt_roles;
File[] public gt_files;

bool private gv_isFirst;
```

- **Mètodes**

En primer lloc, es desenvoluparà el mètode constructor, encarregat de posar en marxa la base de dades per al funcionament. Es crearà un primer usuari i es registraran els 3 rols a la corresponent taula per tal de fer accessible les dades per part dels futurs mètodes desenvolupats:

```
constructor() public {

    // NOTE: the first user MUST be empty
    addUser(address(0x0), "", ac_admin);
    gt_usersRole[msg.sender] = ac_admin;

    gv_isFirst = false;

    // Storing the roles
    uint lv_roleId;
    // NOTE: the first role is MANDATORY
    lv_roleId = gt_roles.length++;
    gt_roles[lv_roleId] = Role({ role: ac_admin });

    lv_roleId = gt_roles.length++;
    gt_roles[lv_roleId] = Role({ role: ac_admin });    // Add Admin role

    lv_roleId = gt_roles.length++;
    gt_roles[lv_roleId] = Role({ role: ac_writter });  // Add Writter role

    lv_roleId = gt_roles.length++;
    gt_roles[lv_roleId] = Role({ role: ac_viewer });   // Add Viewer role

}
```

Un cop inicialitzada la instància de l'objecte, quedaran a disposició per a ser cridats des de el *front-end* el següent conjunt de mètodes per a la interacció i control de la *Blockchain*:

registerUser() i addUser(): Aquest mètode s'encarrega de registrar nous usuaris al sistema. S'executa en el moment en que l'usuari accedeix per primer cop a l'aplicació des de una adreça no registrada i emmagatzema el seu nom d'usuari.

```
function registerUser(
    string memory uv_name )
public returns(uint)

function addUser(
    address uv_wAddr,
    string  memory uv_name,
    bytes32 uv_role)
private returns(uint)
```

isRegistered(): Aquesta funció retorna *true* si l'adreça d'accés està assignada a un usuari, i *false* en el cas contrari. Utilitzada per a saber si una adreça ha de tenir control de l'aplicació.

```
function isRegistered() public view returns (bool)
```

totalUsers(): es retorna el nombre total d'usuaris registrats, necessari per a muntar el llistat que mostra la informació sobre els usuaris i permet la modificació dels rols.

```
function totalUsers() public view returns (uint)
```

getUserById(): es retorna l'usuari registrat al *Blockchain* que correspon amb el Id passat per paràmetre utilitzat en el moment de mostrar les dades dels usuaris des de el *front-end*.

```
function getUserById(uint uv_id) public view
returns(
    uint,
    string memory,
    bytes32,
    address,
    uint,
    uint
)
```

getUserNameById(): es retorna el nom d'usuari que te assignat un Id, utilitzada per a mostrar-lo en el llistat de fitxers.

```
function getUserNameById(uint uv_id) public view
returns( string memory)
```

totalRoles(): retorna el nombre total de rols registrat. En aquest cas serà utilitzat per al mostreig del desplegable de canvi de permisos d'usuari.

```
function totalRoles() public view returns (uint)
```

getRoleById(): retorna el rol corresponent al Id de rol rebut per paràmetre. Per donar la capacitat al *front-end* de rebre els noms dels rols.

```
function getRoleById(uint uv_id) public view
returns(
    uint,
    bytes32
)
```

isAdmin(): funció que retorna *true* en el cas que l'adreça que accedeix estigui registrada amb aquest rol, i *false* en el cas contrari. És utilitzada en aquelles vistes en les quals és necessari disposar rols d'administrador per al seu accés.

```
function isAdmin() public view returns (bool)
```

isFileLoader(): funció que verifica si l'adreça que accedeix a la vista de fitxers està registrada al sistema, i a més té permisos d'administració o escriptura. Amb aquesta s'aconsegueix habilitar o deshabilitar l'element de la vista per a la pujada de fitxers.

```
function isFileLoader() public view returns (bool)
```

updateRole(): funció que rep una adreça i un node, i en el cas d'existir a la base de dades un usuari registrat coincident actualitza el seu rol. És utilitzada per a administrar els rols dels usuaris i gestionar els seus permisos.

```
function updateRole(address uv_wAddr, uint _roleId) public
returns( uint )
```

saveFile() i addFile(): funció que s'encarrega d'enregistrar un fitxer en la *Blockchain* en el moment en que ha sigut pujat a IPFS.

```
function saveFile(
    string memory uv_name,
    string memory uv_ipfsHash )
public returns(uint)

function addFile(
    address uv_wAddr,
    string memory uv_name,
    string memory uv_ipfsHash
)
private returns(uint)
```

totalFiles(): es retorna el nombre total de fitxers, necessari per a muntar el llistat que mostra la informació sobre els arxius registrat en el sistema.

```
function totalFiles() public view returns (uint)
```

getFileById(): rebent la variable d'identificador d'arxiu, és capaç de recuperar les seves dades i retornar-les. Internament s'encarregarà de recuperar també informació sobre l'usuari que va realitzar la seva pujada.

```
function getFileById(uint uv_id) public view
returns(
    uint,
    string memory,  // User Name
    string memory,  // File Name
    uint,
    string memory  // IPFS Hash
)
```

4.4.2 *Front-end*

Per al desenvolupament d'aquesta part de l'aplicació, és necessari implementar una interfície visual capaç de comunicar-se amb l'smart-contract i compatible amb IPFS (mitjançant el protocol HTTP). La necessitat d'execució sobre la maquina client, i la complexitat dels mètodes a desenvolupar fan que Vue.Js sigui un framework Javascript que s'utilitzarà en el desenvolupament.

La seva instal·lació i especialment la configuració de dependències és una de les parts més importants de cara al correcte funcionament dels components necessaris. La generació del projecte de treball serà documentada al punt 9.2 de l'annex.

El disseny de l'aplicació serà molt simple, i gràcies al component Vue Router serà possible l'execució asíncrona de mètodes mitjançant les peticions HTTP, és a dir, serà un *website* dinàmic amb peticions modularitzades.

Figura 13: Plantilla *Front-End* DFileCloud de la D-App DFilecloud

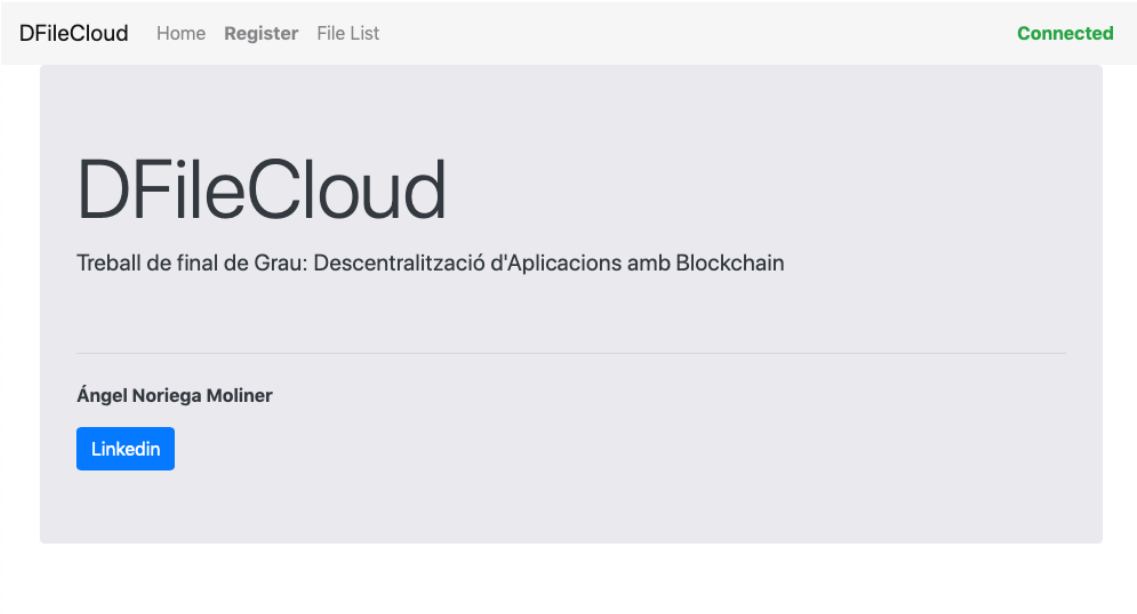
Font: Elaboració pròpia

Com es mostra a la Figura 13, la disposició inicial constarà d'una barra superior que contindrà un menú que enllaçarà amb les corresponents vistes, i una variable visual que informarà de l'estat de la connexió amb la cadena de blocs mostrant Connected en el cas satisfactori. Aquest component serà emmagatzemat a l'arxiu **NavBar.vue**.

El bloc principal de l'aplicatiu, contindrà les 4 vistes que donen les funcionalitats principals: pàgina de benvinguda, registre d'usuaris, la gestió d'usuaris i la gestió de fitxers.

Home.vue
Aquesta vista serà la pàgina d'inici de la D-App. Mostrarà una petita descripció i les dades sobre el projecte i l'autor.

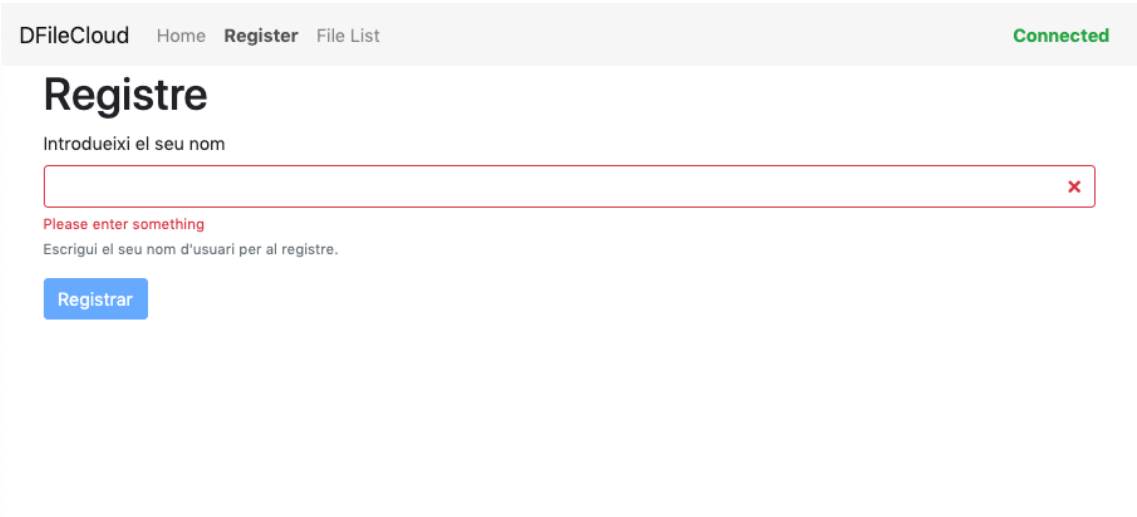
Figura 14: Vista de la pantalla de Benvinguda de la D-App DFilecloud



Font: Elaboració pròpia

Register.vue
Aquesta vista es mostrarà en el menú únicament si l'adreça amb la qual s'accedeix no té registrat cap usuari. Per al cas, es mostrarà un *input* per a introduir el nom, i un botó per a registrar l'usuari al sistema. Un cop fet, es redireccionarà a l'usuari a la vista de gestió de fitxers.

Figura 15: Vista del Registre de nous usuaris de la D-App DFilecloud

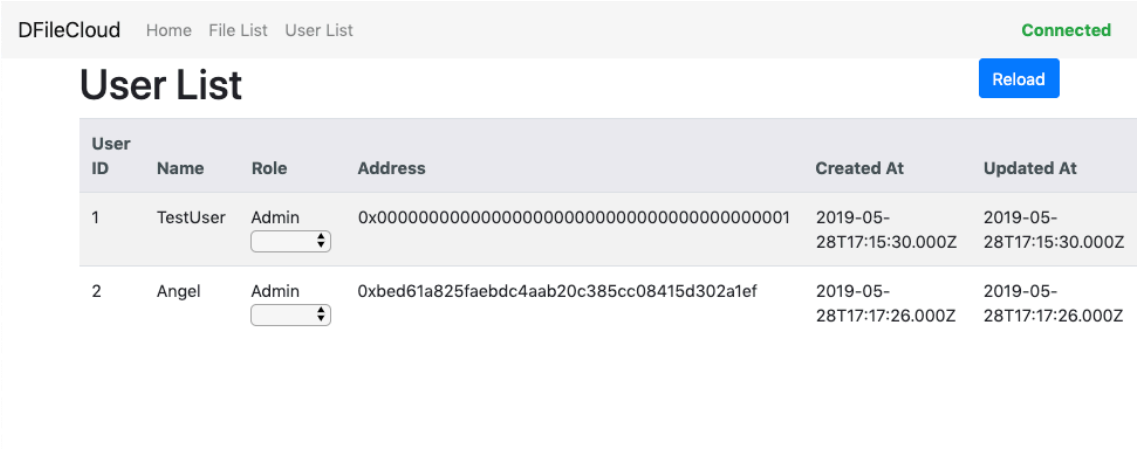


Font: Elaboració pròpia

UserList.vue
Té la funció de visualitzar els usuaris que constin a la base de dades i de modificar el seu rol. El llistat es mostrarà en forma de taula i únicament serà visible per aquells que

estiguin enregistrats amb rol d'administrador. En cas contrari es mostrarà un missatge que informarà sobre la falta de permisos.

Figura 16: Vista de gestió d'usuaris de la D-App DFilecloud

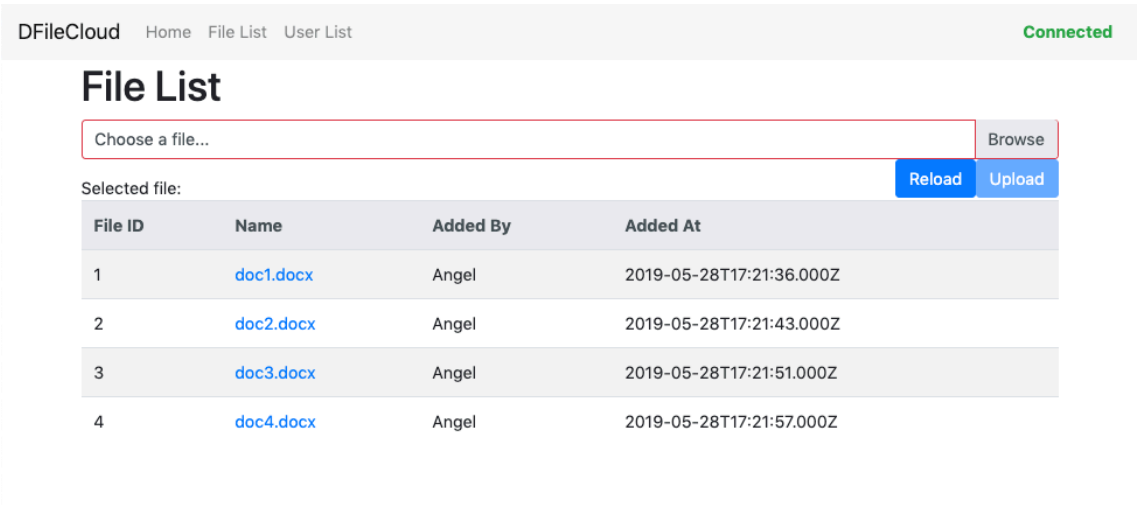


Font: Elaboració pròpia

FileList.vue

Dona la possibilitat de gestionar el fitxers, tant per la part de càrrega com la de visualització del registre i descàrrega. La part que conformarà la taula de fitxers serà visible per tots els rols d'usuari, incloent aquells que no estiguin enregistrats. En quant al input per a la pujada d'arxius, únicament serà visible per als usuaris amb rols "admin" o "writter".

Figura 17: Vista de gestió de documents de la D-App DFilecloud



Font: Elaboració pròpia

4.4.3 *Middleware*

Tal i com mostra l'esquema de la D-App a l'apartat 4.3, serà necessària la comunicació entre la part del *front-end* i el *back-end*. Per a interconnectar-los, els *middleware* seran la part fonamental que portaran a terme aquesta tasca. A continuació es detallen els passos seguits per al funcionament dels 2 components utilitzats per a la unió de la part de Javascript amb els corresponents nodes.

Web3
Ampliant la informació donada a l'apartat 4.3.4[16], aquest component serà afegit a l'aplicació per tal de comunicar-la amb l'*smart-contract*. Es tracta d'una llibreria de funcions que faciliten la interacció amb la base de dades. A continuació es detallen les consideracions fonamentals

S'implementa la dependència amb **versió "0.19.1"**: de principi es va intentar afegir la dependència amb la última versió "1.0", però no ha sigut possible degut a problemes amb el desenvolupament de les seves funcions en mètodes asíncrons de Vue.JS.

Serà necessari que el contracte de Solidity, hagi sigut exportat en format ABI (Aplication Binary Interface) per tal que pugui esser interpretat. A continuació, web3 obtindrà de l'arxiu l'adreça del contracte i iniciarà la seva instanciació en un objecte de Javascript. Aquest component web3, un cop carregat l'arxiu JSON del contracte serà capaç de comunicar-se amb el node d'Ethereum i executar els mètodes desenvolupats a Solidity.

De forma particular, un exemple de la crida que es realitza per a obtenir el total d'usuaris es:

```
window.ethconn.contract().totalUsers.call((err, cv_total) => {
    // err = retorna l'error
    // cv_total = retorna el valor que envia el mètode
})
```

Ipfs-http-client
Es l'altre component[18] utilitzat per a la comunicació nodal. En aquest cas permet interactuar amb un node IPFS amb les mateixes instruccions que possibilita la consola de comandes. Per aquest cas s'ha utilitzat la versió "29.1.0". Les dificultats trobades en el moment d'aquesta implementació s'han focalitzat alhora de l'execució de mètodes asíncrons que en fessin crida.

Per aquest cas finalment després d'una profunda investigació s'ha arribat a la solució d'instal·lar a la D-App la dependència "babel-polyfill" que dona la capacitat d'interpretar mètodes asíncrons encadenats. (s'encadenen mètodes en el cas de la pujada de fitxers, ja que es realitza una pujada a IPFS, i seguidament es registra la línia al node Ethereum.

En quant a la implantació, primerament s'ha configurat un fitxer anomenat ipfs.js el qual instanciarà l'objecte IPFS passant-li per paràmetre la IP i el port on es troba el node.

```
const ipfs = new IPFS('127.0.0.1', 5001 );
```

Per a la comunicació entre el *front-end* i el IPFS per a la pujada d'un arxiu, primerament serà necessària la seva forma binària (emmagatzemant l'arxiu en una variable Buffer), i posteriorment la crida al corresponent mètode "add". A continuació es mostra un exemple:

```
ipfs.add(this.buffer)
    .then((hashedFile) => {
    fileHash = hashedFile[0].hash;
    console.log("fileHash: " + fileHash);
    });
 //la variable fileHash contindrà l'string identificador IPFS de l'arxiu
```

Un cop pujat l'arxiu al corresponent node, aquest podrà visualitzar-se o descarregar-se accedint a un navegador des de qualsevol de les següents adreces:

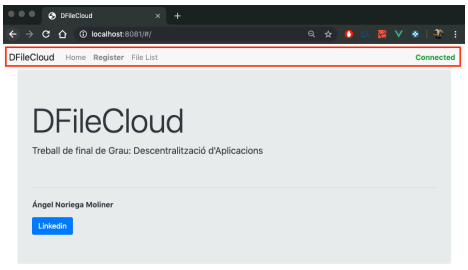
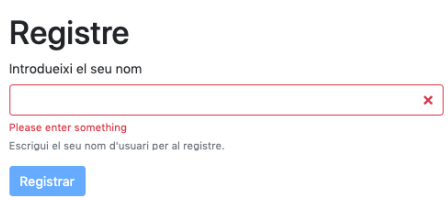
http://localhost:8080/ + fileHash
https://ipfs.io/ipfs/ + fileHash
https://gateway.ipfs.io/ipfs/ + fileHash

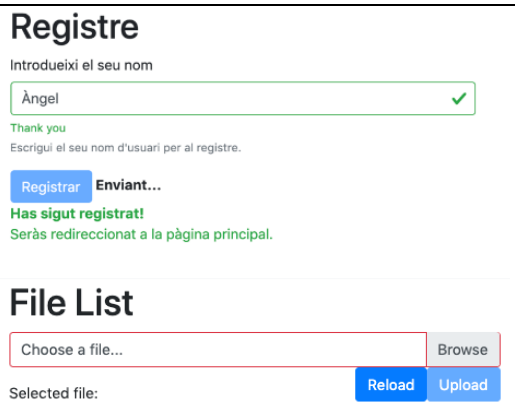
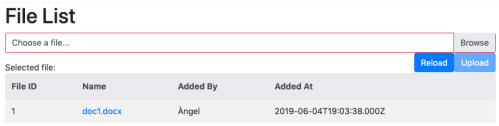
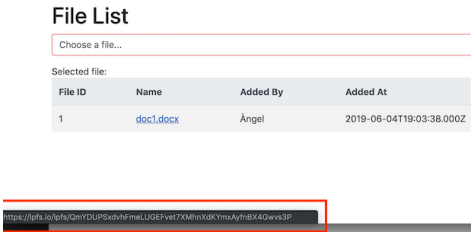
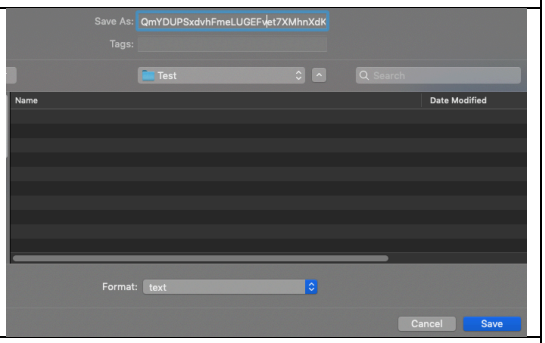
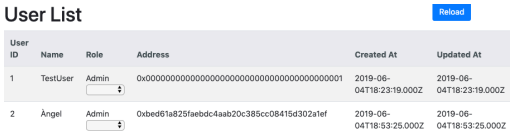
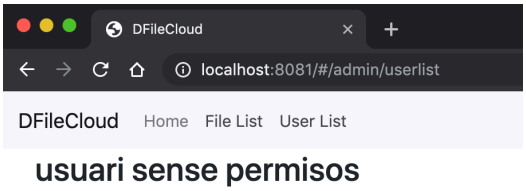
4.5 Proves unitàries de funcionament

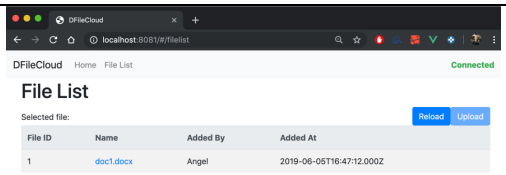
Per a la posada en marxa de l'aplicació en l'entorn local de desenvolupament, primerament és necessari que tant el node IPFS com el node d'Ethereum es trobin iniciats. Per a tal acció s'executa la següent comanda (explicada amb detall a l'annex 7.3) i s'inicia l'aplicatiu de Ganache.

```
anoriega$ ipfs daemon
```

A continuació es mostra un joc de proves unitàries que validen el funcionament de la D-App:

Acció	Sortida desitjada	Resultat
S'executa la comanda "npm run dev" des de l'entorn de comandes, i dins de la ubicació del fitxer del projecte dfilecloud.	Després d'uns segons, s'obre el navegador per defecte del sistema i es mostra la pagina principal de la D-App sobre la URL http://localhost:8081/#/. En aquesta apareixerà el menú propi d'usuari no registrat i el semàfor de connexió en verd.	Passed: OK 
Es prem el link "Register" de la barra de menú	Es visualitza la finestra de registre d'usuaris	Passed: OK 
S'introdueix un nom de més de 4 caràcters i es prem el botó "Registrar"	L'input del "nom" passarà a ser de color verd i es mostrarà un missatge confirmant el registre durant un segon. A continuació es	Passed: OK

	redireccionarà a la vista de gestió d'arxius:	
Des de la vista de "File List" es realitza la pujada d'un arxiu i es pren el botó Upload	Primerament el botó Upload es mostra desactivat fins que no es carrega l'arxiu. Un cop premut el botó, es mostra una taula amb l'arxiu en qüestió registrat	Passed OK 
Es passa el punter del mouse per sobre del link amb el nom de l'arxiu.	Es valida que apareix correctament la seva URL de descarrega	Passed OK 
Es fa click sobre el link	Es valida que s'obre el formulari propi del navegador web per a realitzar la descarrega arxius en local.	
S'accedeix a la vista de gestió d'usuaris "User List" des de el menú superior amb ´'usuari recentment registrat.	Es valida que es mostra un llistat amb dos usuaris, un de test i l'usuari recentment registrat.	Passed: OK 
En el desplegable de la columna Role de l'user creat, es procedeix a seleccionar l'opció Viewer.	Es verifica que es canvia el rol de l'usuari quedant sense permís per a gestionar usuaris. Al recarregar pàgina es mostrarà un missatge informatiu.	Passed:OK 
S'accedeix amb l'usuari amb rol	Es verifica que ara no es mostra el input de pujada de fitxers, però si que es	Passed: OK

"viewer" a la vista de gestió de fitxers.	pot visualitzar la taula d'arxius.	

4.6 Valoració del resultat obtingut

El treball ha donat com a resultat una SPA (Simple Page Application) que ha ajudat a posar en pràctica els coneixements assolits a la part teòrica del projecte.
El propòsit inicial era el de realitzar el llançament d'una D-App i relatar la metodologia seguida. Encara i totes les millores futurament potencials que podrien implementar-se, l'aplicació ha sigut materialitzada i funcional en quant a la seva part didàctica.

En aquesta experiència, s'han estudiat les diferents eines i una de les diverses maneres de posar en marxa l'entorn de funcionament (Ganache i node IPFS en aquest cas).
El desenvolupament del codi per a *l'smart-contract*, m'ha donat una nova perspectiva en quant a l'ús de la programació orientada a objectes per a la definició de les taules que emmagatzemen la informació a la cadena de blocs. Penso que l'estructuració mitjançant classes, en lloc de l'ús de llenguatges del tipus SQL, dona una major polivalència gràcies al fet de poder tenir la definició lògica de les dades i el codi *back-end* en el mateix arxiu.

Respecte a la part *front-end*, he pogut experimentar en l'ús del *framework* Vue.JS, un llenguatge nou per a mi. Les majors dificultats ocorregudes en aquest desenvolupament han estat sobre aquesta part, concretament a l'hora de trobar la versió adient de les dependències dels *middleware* per tal de fer funcionar l'execució dels mètodes de les llibreries dels diferents components.

En quant a la part IPFS, s'ha pogut instal·lar des de cero un node i "jugar" amb el potencial d'aquesta tecnologia, fent ús de les seves funcionalitats de publicació d'arxius a la xarxa IPFS des de l'entorn de comandes, i la interacció del *front-end* amb el node. També s'ha vist a part d'accessibilitat als arxius pujats des de la D-App, com la possibilitat de publicar la D-App sencera per tal que pugui esser visitada sense la necessitat de cap tipus de servidor.
L'estudi d'aquesta ultima tecnologia ha servit per a veure un nou concepte de distribució d'arxius, amb el qual és possible publicar material a la xarxa de forma molt senzilla i de la mateixa manera, fàcilment accessible.

5. Conclusions

Amb aquest treball he pogut comprovar que una bona planificació és una de les parts més importants del projecte que no es pot deixar de banda en el desig de produir un bon resultat. Aquest document tracta d'un projecte pràctic que ha requerit força investigació, i amb la dificultat afegida de tractar una sèrie de tecnologies que són relativament recents. L'estudi de l'abast del projecte, del temps a dedicar i haver tingut un llistat de tasques ben definides ha sigut fonamental per a la correcta finalització del treball.

Per la meva part, opino que s'han complert els objectius establerts inicialment. Cal dir que en la primera fase del projecte, degut al desconeixement sobre la tecnologia de la proposta, vaig tenir clara la necessitat i el valor que donaria al treball la realització una aplicació que poses en practica els coneixements, però em va ésser dificultós decidir en aquell moment quin tipus de D-App desenvoluparia. Al cap d'unes setmanes, i gràcies en bona part a l'ajut d'en Fèlix, vam canviar la proposta de desenvolupament pràctic plantejada inicialment (aplicació per al control de gastos), i vam decidir implantar un servei d'emmagatzemament d'arxius descentralitzat i poder agregar al projecte la tecnologia IPFS.

En quant a l'elaboració del tasca en el dia a dia, s'ha seguit la planificació sense endarreriments i s'han executat totes les entregues (PACs) en les dates indicades. Com a punt positiu, es va poder avançar la part d'investigació de la tecnologia a desenvolupar en una setmana, amb la qual cosa s'ha pogut tenir una part central i final del projecte controlada. Sense una bona planificació, possiblement no hagués sigut possible acabar, ja que per exemple hi ha hagut punts del treball en els quals s'ha hagut de dedicar alguns dies a investigar abans de poder continuar (per exemple les dificultats per a la pujada de fitxers des de *front-end* a IPFS).

Personalment, si hagués de tornar a repetir un projecte, o aquest document hagués de servir d'ajuda a un altre alumne o lector, aconsellaria esperar a prendre les decisions sobre les eines amb les quals s'implementarà l'aplicació a tenir un mínim coneixement sobre les mateixes per tal d'assegurar al màxim que serà exitós. Al tractar-se d'un tipus de treball on és necessària simbiosi entre components, és molt important tenir la seguretat de que els elements escollits tindran total compatibilitat per tal d'evitar un possible fracàs.

De cara a estudiar possibles línies de futur del projecte, hi ha varies vies en les quals realitzar ampliacions o millores. En el primer cas seria interessant de millorar les dades registrades de cada arxiu, de forma que s'emmagatzemi un major nombre de metadades. Aquesta informació extra seria molt interessant en els casos de voler gestionar la base de dades des d'altres entorns i sistemes i realitzar càlculs essencials estalviant la descarrega i processament de cada arxiu (per exemple la lectura des d'un sistema SAP).
Per una altra part, penso que podria tenir un bon recorregut l'estudi de la part de la arquitectura de nodes que requeriria la posada en marxa de DFileCloud sobre la xarxa d'una multinacional. Des de l'estudi de l'ús de la xarxa, fins a l'eficiència tant de la cadena de blocs com dels nodes IPFS.

Tancant l'apartat de conclusions, opino que gràcies al treball d'aquests últims mesos, m'emporto un aprenentatge que estic convençut que algun dia podré aprofitar en el

meu futur professional. Haver desenvolupat un projecte d'aquest caire des de cero, l'estudi de la tecnologia *Blockchain* i el nou domini adquirit sobre la tecnologia IPFS, són fites que és molt probable que siguin utilitzades dins de la trajectòria professional que encamino ara mateix.

6. Bibliografia

[1] *BitcoinWiki - 51% Attack*. Web: https://en.bitcoinwiki.org/wiki/51%25_attack. [Últim accés: 03/04/2019].

[2] *Historia i orígen del Blockchain*. Web: https://www.bbva.com/es/historia-origen-blockchain-bitcoin/ [Últim accès: 03/04/2019].

[3] *Wikipedia - Blockchain*. Web: https://en.wikipedia.org/wiki/Blockchain. [Últim accés: 03/04/2019].

[4] *Satoshi Nakamoto - Bitcoin: A Peer-to-Peer Electronic Cash System*. Web: https://bitcoin.org/bitcoin.pdf. [Últim accés: 03/04/2019].

[5] *Wikipedia - SHA2.* Web: https://es.wikipedia.org/wiki/SHA-2. [Últim accès: 03/04/2019].

[6] *Wikipedia - Arbre de Merkle*. Web: https://es.wikipedia.org/wiki/Árbol_de_Merkle. [Últim accès: 03/04/2019].

[7] Bit2me – Que es el bloque gènesis. Web: https://academy.bit2me.com/que-es-bloque-genesis/. [Últim accès: 03/04/2019].

[8] Wikipedia - Ethereum. Web: https://es.wikipedia.org/wiki/Ethereum. [Últim accès: 03/04/2019].

[9] Bit2me – Que es la cadena de bloques blockchain. Web: https://academy.bit2me.com/que-es-bloque-genesis/. [Últim accès: 03/04/2019].

[10] Youtube Simply Explained - Smart contracts. Web: https://www.youtube.com/watch?v=ZE2HxTmxfrI. [Últim accès: 03/04/2019].

[11] The Message Passing Interface (MPI) Standard. Web: https://www.mcs.anl.gov/research/projects/mpi/. [Últim accès: 03/04/2019].

[12] IPFS. Web: https://ipfs.io/#why. [Últim accès: 03/04/2019].

[13] Ganache. Web: https://truffleframework.com/ganache. [Últim accés: 05/05/2019]

[14] Truffle. Web: https://truffleframework.com/truffle. [Últim accés: 05/05/2019].

[15] Web3 Documentation – Web: https://web3js.readthedocs.io/en/1.0/. [Últim accés: 05/05/2019].

[16] Vue.JS. Web: https://vuejs.org/. [Últim accés: 05/05/2019].

[17] Solidity Documentation. Web: https://solidity-es.readthedocs.io/es/latest/. [Últim accés: 05/05/2019].

[18] Github - ipfs-http-client. Web: https://github.com/ipfs/js-ipfs-http-client. [Últim accés: 05/05/2019].

[19] Team Tree House - How to install Node.js and NPM on a Mac. Web: https://blog.teamtreehouse.com/install-node-js-npm-mac. [Últim accés: 05/05/2019].

[20] NPMJS - Creating a Package json file. Web: https://docs.npmjs.com/creating-a-package-json-file. [Últim accés: 05/05/2019].

[21] Youtube - IPFS Alpha Demo. Web: https://www.youtube.com/watch?v=8CMxDNuuAiQ. [Últim accés: 05/05/2019].

7. Annexos

A continuació es mostra el passos per a l'habilitació de les eines útils per al desenvolupament.

En cas de disposar d'una màquina amb el sistema operatiu OSX. És requisit tenir instal·lada l'eina brew (https://brew.sh/index_es).
Es tracta un gestor de paquets mitjançant comandes, similar al "apt" de Linux que facilita la instal·lació de software.

Per a gestionar els paquets d'aquest projecte, també serà necessària la instal·lació de NPM.
S'executa la següent comanda per a la instal·lació[19] de NPM, el gestor de paquets utilitzar:

```
anoriega$ brew install node

anoriega$ npm -v
6.9.0
```

7.1. Configuració del compilador d'smart-contracts Truffle

El codi desenvolupat en llenguatge Solidity, serà necessari que sigui compilat per a la seva interpretació per part del *front-end*. Per a dur a terme aquesta tasca s'ha utilitzat el software Truffle[15]. Molt útil per a l'organització de projectes amb *smart-contracts* i un all-in-one en quant a funcionalitats referides.
S'executen les següents comandes per a la seva instal·lació, incloent la creació de l'arxiu DFileCloud.sol que contindrà tot el codi del nostre *back-end*.

```
anoriega$ npm install -g truffle

anoriega$ mkdir solidity

anoriega$ cd solidity

anoriega$ truffle init

anoriega $ ls -la
total 24
drwxr-xr-x   7 anoriega  staff   224 May 25 13:17 .
drwxr-xr-x  18 anoriega  staff   576 May 25 12:18 ..
drwxr-xr-x   3 anoriega  staff    96 May 25 13:17 contracts
drwxr-xr-x   3 anoriega  staff    96 May 25 13:17 migrations
-rw-r--r--   1 anoriega  staff   206 May 25 12:11 package.json
-rw-r--r--   1 anoriega  staff  4233 May 25 13:17 truffle-config.js

anoriega $ cat truffle.js
module.exports = {
         networks: {
                   development: {
                             host: "127.0.0.1",
                             port: 7545,
                             network_id: "*"
                   }
         }

anoriega$ cat contracts/DFileCloud.sol  //Mostra el contingut del contracte de la D-App

pragma solidity >=0.4.21 <0.6.0;

contract DFileCloud {
```

```
    //Aquí es contindrà tot el codi del contracte que ha sigut explicat al punt 6.3.1 de la
memoria.

}
```

Per a compilar el codi, serà necessari tenir executat un node d'Ethereum. En aquest cas s'ha executat la eina Ganache.
Un cop realitzat, és possible executar la següent comanda per a obtenir l'arxiu JSON final:

```
anoriega$ rm -rf build && truffle compile --all && truffle migrate

   //Compilarà el codi font generant un arxiu Json a la carpeta builds/contracts del
projecte
```

7.2. Generació del *front-end* Vue.JS

Un cop situats per comandes a la carpeta on es desitja instal·lar el projecte, es procedeix a generar l'arxiu package.json, que contindrà el llistat de dependències del projecte, amb la següent comanda[20]. Seguidament es mostra el llistat de dependències necessàries per al projecte:

```
anoriega$ mkdir dfilecloud

anoriega$ cd dfilecloud

anoriega$ npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.

See `npm help json` for definitive documentation on these fields
and exactly what they do.

Use `npm install <pkg>` afterwards to install a package and
save it as a dependency in the package.json file.

{
  "name": "DFileCloud",
  "description": "",
  "version": "1.0.0",
  "author": "Angel Noriega <eanoriega20@gmail.com>",
  "private": true,
  "scripts": {
    "dev": "webpack-dev-server --mode development --open --hot",
    "build": "cross-env NODE_ENV=production webpack --progress --hide-modules"
  },
  "dependencies": {
    "bootstrap": "^4.3.1",
    "bootstrap-vue": "^2.0.0-rc.20",
    "buffer": "^5.2.1",
     "font-awesome": "^4.7.0",
    "ipfs-http-client": "^29.1.0",
    "popper.js": "^1.12.9",
    "vue": "^2.5.3",
     "vue-router": "^3.0.3",
    "web3": "^0.19.1"
  } ,
  "devDependencies": {
    "@vue/cli-plugin-babel": "^3.6.0",
    "@vue/cli-plugin-eslint": "^3.6.0",
    "@vue/cli-service": "^3.6.0",
    "autodll-webpack-plugin": "^0.4.2",
    "babel-core": "^6.26.0",
    "babel-eslint": "^10.0.1",
    "babel-loader": "^7.0.0",
    "babel-polyfill": "^6.8.0",
```

```
    "babel-preset-env": "^1.6.1",
    "cross-env": "^5.1.1",
    "css-loader": "^0.28.7",
    "eslint": "^5.16.0",
    "eslint-plugin-vue": "^5.0.0",
    "extract-text-webpack-plugin": "^3.0.2",
    "file-loader": "^1.1.5",
    "style-loader": "^0.19.0",
    "vue-loader": "^14.2.2",
    "vue-template-compiler": "^2.5.21",
    "webpack": "^4.20.2",
    "webpack-cli": "^3.1.1",
    "webpack-dev-server": "^3.1.14"
  }
}

anoriega$ npm install
//S'instal·laran els paquets de dependencies del projecte. Trigarà uns minuts.


anoriega$ npm run dev
//Executa una instància del projecte i la obre al navegador web a la IP
http://localhost:8081
```

Arribats a aquest punt, ja està tot llest per a iniciar el desenvolupament de la part *front-end*.

7.3. Instal·lació del node IPFS

A continuació es relata breument la descarrega instal·lació i execució[21] del node IPFS, necessari per a la nostra D-App:

```
anoriega$ tar xvfz go-ipfs.tar.gz

anoriega$ cd go-ipfs

anoriega$ ./install.sh

anoriega$ ipfs version
ipfs version 0.4.20

anoriega$ ipfs init

    //Confirmació de la inicialització del node

anoriega$ ipfs daemon
Initializing daemon...
go-ipfs version: 0.4.20-
Repo version: 7
System version: amd64/darwin
Golang version: go1.12.4
Swarm listening on /ip4/127.0.0.1/tcp/4001
Swarm listening on /ip4/192.168.1.128/tcp/4001
Swarm listening on /ip6/::1/tcp/4001
Swarm listening on /p2p-circuit
Swarm announcing /ip4/127.0.0.1/tcp/4001
Swarm announcing /ip4/192.168.1.128/tcp/4001
Swarm announcing /ip6/::1/tcp/4001
API server listening on /ip4/0.0.0.0/tcp/5001
WebUI: http://0.0.0.0:5001/webui
Gateway (readonly) server listening on /ip4/0.0.0.0/tcp/8080
Daemon is ready
```

Un cop executat el dimoni, el node queda iniciat i escoltant les crides per part del middleware ipfs-http-client.

Si es desitgés realitzar la pujada d'algun arxiu, únicament seria necessari executar la següent comanda

```
anoriega@dfilecloud$ ipfs add arxiu.txt
```

Des d'aquest instant, un cop sincronitzat amb la xarxa IPFS aquest seria accessible des de les següents ubicacions:

http://localhost:8080/ + &cadena retornada pel comandament&
https://ipfs.io/ipfs/ + &cadena retornada pel comandament&
https://gateway.ipfs.io/ipfs/ + &cadena retornada pel comandament&

En quant al desplegament de l'aplicació, únicament és necessària l'execució de la següent comanda sobre el projecte per a fer-la publica i accessible:

```
anoriega@dfilecloud$ ipfs add -r .
```