Degree Final Project on Bachelor Degree in Informatics Engineering, Information Technologies

Polytechnic University of Catalonia

New Innovations in eIDAS-compliant Trust Services: Blockchain

Author: Xiaolei Lin

Director: Francisco Jordan Fernández

In collaboration with the company

Entrust Datacard™

June 2020

Abstract

Technologies advance in leaps and bounds, they mark new trends that emerge to dominate the market, products that were previously novel and nowadays that must be adapted to remain competitive. For this reason, the team, that is made up of 3 students from the FIB - UPC (Arthur Bernal, Marc Méndez and Xiaolei Lin) and is led by the professor and director Francisco Jordan, proposes in this project new innovative technologies that will mark the technology of future and incorporate it into the TrustedX product.

This project will be divided into two parts. The first consists of the communal part, which is carried out by all team members and the second, is the individual part that is realized only by the author of this thesis.

The common part is based on expanding and incorporating all necessary components in the TrustedX on-premise product in order that it can function as TrustedX as a Service (TXaaS) and a multi-tenant system. This new product will have the ability to comply with eIDAS Regulation to offer digital signatures in the Cloud and have the same validity as the handwritten notarial signatures.

The individual part consists of creating a timestamp-based archiving prototype by using Blockchain technology and, integrating it into TXaaS. To fulfill with this, the operation of this technology and the different available options in the market are studied. In addition, all components which are required will be designed and implemented in order to rach with the objective.

Resumen

Los avances tecnológicos van a pasos agigantados, con ellos marcan nuevas tendencias que emergen para dominar el mercado, productos que antes era novedosos y que ahora deben adaptarse para seguir siendo competitivos. Por ello, el equipo compuesto por 3 estudiantes de la FIB – UPC (Arthur Bernal, Marc Méndez y Xiaolei Lin) y dirigido por el profesor y director Francisco Jordan proponen en este proyecto nuevas tecnologías innovadoras que marcará el futuro tecnológico e incorporarlo en el producto TrustedX.

Este proyecto se dividirá en dos partes, la primera que es la parte comuna es realizada por todos los integrantes del equipo y la segunda, la parte individual la realiza solo el autor de esta tesis.

La parte comuna se basa en expandir e incorporar los componentes necesarios en el producto TrustedX *on-premise* para que pueda funcionar como TrustedX *as a Service* (TXaaS) y un sistema *multi-tenant*. Este nuevo producto tendrá la capacidad de cumplir los Reglamentos de eIDAS para ofrecer firmas digitales en el Cloud y tener la misma validez que las firmas notariales manuscritas.

La parte individual consiste en crear un prototipo de archivado basado en *timestamp* utilizando la tecnología Blockchain e integrarlo en TXaaS. Para ello, se estudia el funcionamiento de esta tecnología y las diferentes opciones disponibles en el mercado. Además, se diseña e implementa todos los componentes requeridos para cumplir el objetivo.

Agradecimientos

Esta reflexión no solo va dirigido a este proyecto, sino a toda la etapa universitaria. Una vez alcanzado el final de un largo recorrido, en la recta final, quiero pararme y agradecer a todas las personas que han estado presentes.

Gracias Dr. Francisco Jordan por esta oportunidad de trabajar en Entrust Datacard y en una tecnología que personalmente considero interesante y por todo lo que he podido aprender, que en otro lugar no hubiera sido posible.

Gracias Arthur Bernal y Marc Méndez por ser compañeros de trabajo en la cual puedo confiar y cooperar.

Gracias Amigos por la compañía, sin vosotros la universidad no hubiera sido lo mismo.

Gracias Familia por apoyarme.

Gracias Yu por estar siempre a mi lado.

Index

List of Figures

List of Tables

Preface

TXaaS stands for TrustedX as a Service. TrustedX is an original product of the former Safelayer company, now part of the Entrust Datacard Corporation (EDC). As a leading manufacturer of Public Key Infrastructure (PKI) technology, the company created TrustedX to consume digital certificates in electronic signatures. At this time, the innovation was to provide a secure remote system for applications to consume the service centrally through web services. This was revolutionary at the time as recognized by the European Commission and other European associations (see https:/tinyurl.com/y7pk7joq and https://tinyurl.com/y6w8s598).

More than ten years after TrustedX, we launched the TXaaS Project. It is an "Innovation" project under the Office of the CTO at Entrust Datacard and sponsored by the CIO (Anudeep Parhar). The context strongly encourages innovation: a small autonomous team of 4 people, 3 final master/degree students at the UPC (Marc Mendez, Arthur Bernal and Xiaolei Lin) and myself, professor at the UPC and Director in the Office of the CTO at EDC, as the link between the academic and business worlds.

The common notable trait to mention about the team is that we are all passionate about technology. The whole team provides good informatics acknowledge and helped in different shared areas, as well as in new areas such as Artificial Intelligence and Machine/Deep Learning, Application Programming and Graphical User Interfaces, Cloud Computing, Distributed Ledger Technologies and Blockchain. What do all these technologies relate to TrustedX, PKI and Electronic Signatures? Only "Innovation" can answer this question comprehensively.

Innovation, and Research and Development, is now a must in all businesses, not to be a leader, but to survive. The art of adapting to the rapidly changing technology landscape cannot be improvised. A product or a business can improve a little by updating an existing technology, however, they can revolutionize and explode by embracing a completely new technology. Where a company ranks between the two extremes defines it in terms of long-term market leadership. Innovation is a long-term continuous race that is cleverly imprinted in one of my favorites A. Einstein quotes: "Don't expect different results by doing the same thing over and over again".

What is TXaaS? It can be seen as an Innovation project that seeks to improve TrustedX product, although what it certainly pursues is to revolutionize the way we can deliver value by adopting completely new technologies.

F.Jordan

Director Office of the CTO at EDC, PhD, Associate Professor at the UPC

June 2020

Introduction

Description

This project is proposed as a collaborative work of greater dimension than can be absorbed in a single academic work of Final Degree or Master Thesis. Not only from the point of view of volume but also from the point of view of the subjects and technologies to be dealt with. This is why a single project is proposed and this is divided into several parts that give rise to several Final Degree and Master's works, all of them carried out during the same period from mid-September 2019 to the end of June when the academic papers are defended.

The approach and genesis of this project has different plans, all of them interrelated. Topics namely:

- Specific Area/Technology. The project focuses on the digital signature, a cryptographic mechanism that make it possible to provide authenticity, integrity and non-repudiation to electronically generated data and their exchange.
- Normative and regulation. Currently, the digital signature enjoys legal recognition comparable to the handwritten signature, that is, a digitally signed document has the same validity as a handwritten one as long as certain technical and compliance requirements established in a law or regulation at the government level are met.
- Product. The project is located within the company Entrust Datacard Europe, part of the European-American group Entrust Datacard Corp (EDC), and focuses on a digital signature security product that is already on the market: the TrustedX Platform.
- Supporting technologies. This is an IT project about a complex distributed system with multiple components and diverse technologies that must be related through the network in a secure way. To the basic technologies of systems and web applications as well as security and cryptography, the project introduces new areas and technologies such as: Artificial Intelligence and Machine Learning, Application Programming and Graphical User Interfaces, Cloud Computing, Distributed Ledger Technologies and Blockchain.
- Innovation. The project not only aims to improve a product and some related processes but also to solve existing and new problems through emerging technologies by changing the way of doing things so far. Through innovation, global technological leadership is also pursued.

- Knowledge acquisition. The acquisition of knowledge and mastery of new emerging technologies within the company is also very important in the project.
- Validators and demonstrators. The work to be done with new technologies and the innovations that may arise in the project must be based on a rigorous theoretical and scientific substrate, but at the same time they must pursue the creation of proofs of concept, validators and practical demonstrators that allow the participants to interact and show the proposed improvements and advances.
- Academic plan. The development and results of this project must also be used to carry out the participants' Final Master and Degree Works.

Next points, we will introduce the multiple project plans, the formulation of the problem to be solved, as well as the general objectives and the stakeholders of the project.

Digital Signature and the eIDAS Regulation [1]

Electronic Identification, Authentication and Trust Services (eIDAS) is a European regulation, specifically, Regulation (EU) No. 910/2014. It allows electronic identification services and trust services to have secure and legal validity for all members of the European Union in electronic transactions within the Single Market. [2] [3]

The European Parliament and the European Council published eIDAS on 24 July 2014, and Member States were given a deadline of 29 September 2018 to migrate from the 1999 Directive to the eIDAS Regulation, include mutual recognition of electronic IDs among Member States. Currently, this process has been completed, and all Member States comply with the same high standards for providers of identity services and providers of trust services for authentication and signatures offering following benefits.

- A citizen can use electronic signatures and other trust services across the EU.
- National electronic IDs will be recognized equally in all Member States.
- Citizens can use their electronic IDs and electronic signatures to transact business across Europe.
- Electronic documents will be legally recognized in any EU Member State, regardless of the Member State in which it was written.
- Document seals and time stamps issued in any EU Member State will be considered valid in any other Member State.

The burden for eIDAS compliance falls on the organizations that provide trust services to the public, such as the so-called KYC (know your customer) services in backing.

Therefore, the individual consumer of these services does not have to worry about compliance.

The eIDAS Regulation contains multiple levels, but one of the interesting points is trust services, especially electronic seal and signing in the cloud.

The electronic seal is a new legal concept where electronic signature created by an individual can be a legal equivalent of a handwritten signature.

Signing in the cloud is an alternative approach to electronically signing documents, whereby signing keys are held on a trust service provider's Hardware Security Module (HSM). This approach eliminates the need for users to handle their own keys.

In essence, the TrustedX eIDAS platform is an implementation of signing in the cloud with HSM, which also allows signing with using mobile devices.

TrustedX eIDAS Platform [4]

TrustedX eIDAS Platform is a web service used to integrate the authentication and electronic signature, moreover it also provides multiple functionalities as multi-factor authentication, confidence level management, identity federation and remote signature with PKI keys stored in servers through the API web.

This product was designed to be able to provide secure identification of users in a mobility and cloud context by including authentication, SSO and identity federation. It uses remote signature functions following the technical normative CEN TS 419 241, certified as a Qualified Signature Creation Device for being used for the Trusted Service Providers (TSP).

The platform acts as an Identity Provider (IdP) and Signature Provider (eSigP) of users that want to use an application. The Identity provider is the one that validates the identity of the users, with authentication mechanisms based on PKI, SMS/Email, OTP, SafeLayer Mobile ID and other methods of authentication based on plugins to integrate to other services.

The final objective of TrustedX is the electronic signature, however this is not possible without an authentication method that has the legal requirements asked for this type of procedure. The platform provides these authentication methods that follow the normative which says that to legally be able to perform a signature the authentication method has to have a strong 2-factor verification, also it has signature methods.

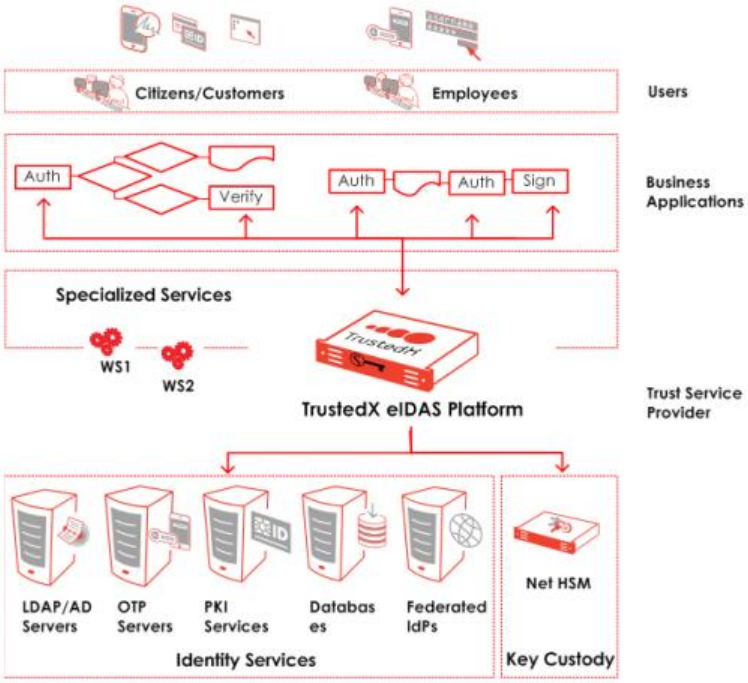


Figure 1 TrustedX eIDAS Platform

TrustedX eIDAS Platform was designed and implemented to be deployed on-premise and be controlled for the company to give them service, however, this Platform was also prepared for integration with other applications of the clients, that's why, for making a multitenant version we did not have to modify TrustedX but only its communication channels.

The procedure followed when new clients purchase it is a bit long and needs supervision of an expert as is needed to install and configure locally in the company. Just by analyzing the process mentioned, we can already ensure that there easier ways to do it and here is where the project starts to work. Solving these problems next to other ones making easier for the company all installation and maintenance process.

Problem Formulation

TrustedX as a Service (TXaaS)

From a product point of view, the project starts with the "TrustedX eIDAS Platform" product. It is a product originally designed for local deployment to be operated by an organization. However, currently, the predominant model for the deployment and operation of technological platforms and applications is based on the cloud and instead of product-oriented now tends to be as-a-Service. This model has demonstrated its

superiority in many facets and levels, which is why it already considers the technological model of the future and where everyone is going to.

The TrustedX eIDAS platform is a complex technological product not only from the application and technology support but also from the point of view of security implementation and the security approvals and certifications that has passed and continues to pass (for example, the Criteria certification common EAL4). From this perspective, we consider a good strategy to maintain the TrustedX eIDAS platform and transform it as a component that belongs to a broader platform that introduces and extends those capabilities and functionalities that do not exist or fall short for the transition of the product "TrustedX eIDAS Platform (TrustedX)" to the future service “TrustedX as a service (TXaaS) ".

Internal studies of the state of technological art (see State of the Art point for more detail), as well as the refinements of different architectural proposals, have led us to propose a general scheme of systems architecture that we consider the most appropriate for the current starting point and on all with a lot of vision for the future. Currently, the TrustedX platform can be represented architecturally, as Figure 2 indicates.

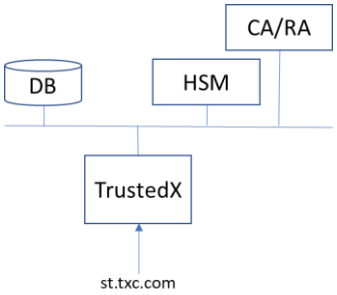


Figure 2 Current TrustedX Platform architecture

The current TrustedX Platform is distributed and runs on a dedicated computer (physical or virtual appliance) that connects to the network to:

1) Access data in a DB (Database), generally external for scalability and reliability (although it supports an internal DB with size restrictions). The stored data is from the users/signatories of the platform, the public signature material, configuration and management information, logs and activity events, etc.
2) Access an HSM (Hardware Security Module) or external security module for the generation, storage and encryption of keys and secure data.

3) Access a CA / RA (Certification and Registration Authority) which are trusted authorities for the certification of public signing keys and the issuance and management of trusted public digital certificates.
4) Provide trusted web services through an API (Application Programming Interface) and an administration GUI (Graphical User Interface) through an Internet address for example “st.txc.com” (representing an organization or tenant, consumer of the service in a domain, for example “txc”). The abbreviations "st" are the abbreviation of "single tenant".

This architecture is currently deployed in a multitude of clients. It offers services to organizations reliably and securely after passing the approvals and certifications required by current legislation. Each organization has its own TrustedX deployment which is independently operated and is also independently approved and certified.

The question that arises is how taking advantage of the capabilities (especially the security, approvals and certifications) of the TrustedX platform can extend the architecture to serve multiple organizations by sharing single computing, operation, approval and certification infrastructure. In the current vision, organizations (clients) seek to consume the service and reduce costs, so sharing all management resources and costs is undoubtedly the best strategy.

Additionally, at the level of an organization that deploys and operates a system, the material cost of the infrastructure must be taken into account, but above all, the cost of the necessary specialized personnel must be taken into account even more than in the case of security in general and PKI, and digital signature in particular requires a high specialization and is very rare.

The following general architecture is proposed as a solution to offer TrustedX as a service, that is, to be able to offer multiple organizations the original TrustedX service but owning the service provider, all the technical and operational infrastructure, so that client organizations’ only see access to the service through the Internet address "st-n.txc.com" where "txc" is the Internet domain of the service provider and "st-n" indicates the domain of an organization or single-tenant "n" of the multiples that are served by the TrustedX-as-a-Service or "TXaaS" platform.

Figure 3 we can see the current basic architecture of the TrustedX platform (with the DB, HSM and CA / RA components) which has been replicated and shared together with new components that are described below:

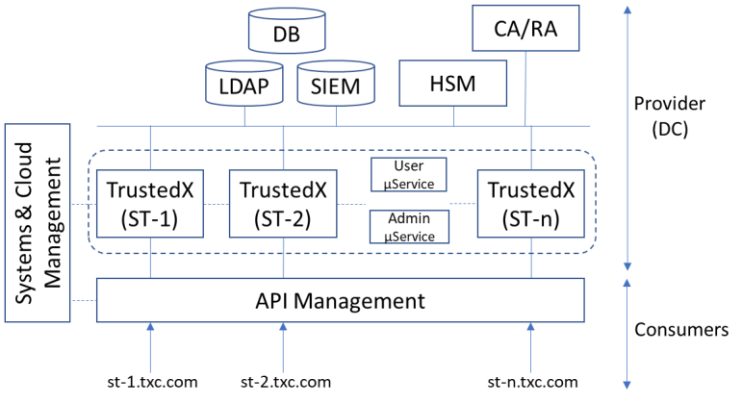


Figure 3 TXaaS general architecture

- **DB, HSM and CA/RA** components will now be shared among all TrustedX (ST-n) corresponding to different organizations (tenants). Properly configured existing technologies will be used to uniquely and securely share data across all the organizations.
- A **LDAP** component is introduced that will support the centralized management of users (both end-users and administrators) of the entire system. A single shared configured technology will be used in which organizations will exclusively and securely view their data. This technology must support millions of users.
- A **SIEM** component is introduced that will support the centralized management of information for logs and events in a general and secure way for the entire system. A single shared configured technology will be used in which organizations will exclusively and securely view their data. This technology will have to support thousands of millions of events.
- An **API Management** component is introduced that will support the management of the APIs of the different services in addition to helping to build a unified and uniform front-end of APIs through the integration and orchestration of different technologies and back-end services. This technology must support multiple APIs and absorb millions of service requests and responses from all consumer organizations (tenants).
- Technological elements **System & Cloud Management, User and Admin µServices** are introduced to support the configuration, deployment, integration, maintenance and management of the other components. For example, the µServices will support the provisioning of users, both end and administrators ones, and signature material (keys generated in the HSM, certificate

management, etc.) and also support for graphical administration (Graphical User Interface) of the entire system.

New technological components and elements will require additional capacities and requirements to those mentioned here. Each one will be developed in a specific point of the memories.

With this general architecture, the following points are intended:

- Take full advantage of the existing capabilities of the TrustedX platform. Developing, standardizing and certifying these capabilities on a new platform requires a significant amount of time and cost that now cannot / wants to be assumed, but later and in parallel when the new platform is in operation. Technically it is possible, and strategically it is much more suitable not only from the economic point of view but also from the technical point of view and the quality of the final service.
- Minimize the physical cost of the infrastructure and its management and administration by deploying the entire infrastructure in a data center (Provider DC) that will be responsible for providing the service to all organizations.
- Minimize the Total Cost of Ownership (TCO) for organizations and move to a service subscription model.
- Very significantly extend the intelligence management, reporting and analysis capabilities of the entire system, as well as individually for organizations.

So far, only the part of the proposal for the new TXaaS system has been described. Taking advantage of the new general architecture and the desire to introduce new technologies to solve certain problems, the following final architecture is proposed in the project:

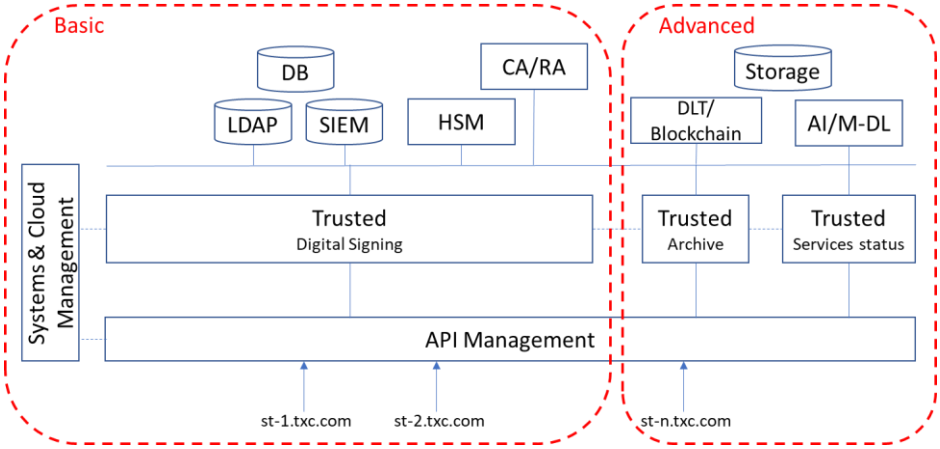


Figure 4 TXaaS Final architecture

Figure 4 we can see how the general architecture (or basic) that will support the digital signature service (Digital Signing) has been extended with new advanced services (Advanced part) to support the new services:

- **Trusted Archive** that will allow the long-term storage of digitally signed and notarized documents using Distributed Ledger (DLT) technologies in general and Blockchain in particular. This trusted advanced service is also regulated in eIDAS. However, it presupposes the use of technology other than DLT. However, there is evidence, and we will also demonstrate in this work that the use of DLT/Blockchain is more efficient and reliable.
- **Trusted Service Status (Anomaly Detection)** that aims to introduce analysis and prediction capabilities of the correct operation and use of the system in general, and of the digital signature service in particular. Using Artificial Intelligence (AI), Machine Learning and Deep Learning (M-DL) techniques, we want to infer information about the use of the service by consumers, in particular, we want to be able to define a “normal” behaviour of end users of the signature service through which we will be able to detect possible anomalies that could undermine trust in the system. For example, detecting possible impersonations of the signatories, thus providing a greater degree of confidence in the system and in the digital signatures generated.

Having arrived at the final formulation of what we want to build in this project, the following section proposes a division of the problem into different common and particular parts that will be developed by the team involved.

Trusted Digital Signing | Trusted Archive | Trusted Status Service

The architecture and final proposal of the project has been divided into 3 large blocks related to the basic and advanced services described:

- **Trusted Digital Signing** includes the new technological infrastructure made up of the TrustedX components (classic platform), and new LDAP and SIEM, as well as the new technologies introduced for Cloud Computing, API Management and general system administration through a single GUI. This block involves a common part of infrastructure and services for the rest of the blocks.
- **Trusted Archive** introduces an advanced service that is supported through the additional use of new DLT / Blockchain and Cloud Storage technologies.

- **Trusted Status Service (Anomaly Detection)** introduces an advanced service that is supported through the additional use of new Artificial Intelligence (AI), Machine Learning and Deep Learning (M-DL) technologies.

Digging deeper into the project division, block 1) Trusted Digital Signing is much larger than the rest of the blocks. For this reason and better organization, block 1) is divided into 2 parts:

a) **Common Infrastructure**. A common part that includes the basic infrastructure elements (LDAP, SIEM, CA / RA and HSM) and the basic TrustedX platform (existing product).
b) **API/GUI Management**. A part of new technologies and development focused on API management, and the design and creation of a unique and uniform graphical user interface (GUI) developed exclusively on generated APIs.

Finally, in addition to the study, design and development of different blocks, the proposal and development of proofs of concept, validators and demonstration applications are also included.

The distribution of all blocks, both common and private, has been carried out with criteria of specialization and profile of the participating team, as well as academics for the defense of the final works. See Scope point for more details.

Trusted Archive

New opportunities appear with the emergence of new technology, as Blockchain has. Entrust Datacard, as a leading company in the security and trust sector invests in R&D, promoting innovation and research on this technology to analyze use cases and incorporate it into its products. And, here is the question of the primary purpose of this project, the confidence based on Blockchain. Can Blockchain technology provide greater efficiency and confidence to documents signed?

Currently, because the validity period of digital certificates is limited and the cryptographic algorithms may become weak and/or break, digital signatures must be maintained fresh and valid over the time by incorporating new timestamps. But to archive this, the current required processes and the infrastructure are complex and expensive. In addition, the platform which is offered by timestamp has also an expiration, therefore, it requires renewal from time to time to continue complying with the strict measures of security to continue being “legal” and reliable, with all the cost that it entails. However, we have Blockchain (public) technology with its properties, such as immutability and a

synchronization system, to achieve a temporal order in its "blocks" that also uses the timestamp to prove the existence of an event in time. For this reason, we want to test and illustrate the idea of registering the existence of a digital document in a temporary stay for “eternity”, and offer an additional layer of trust to documents signed by using the timestamp of a public and reliable Blockchain network.

On the other hand, in the context of TXaaS and digital signature, we see a need to offer clients the possibility of archiving documents signed in a unique ordered repository, available 24/7 and accessible from any device and anywhere in the world.

Finally, with these ideas in mind, we want to innovate by offering a prototype of long-term archiving service to improve the current state of the art (see https://tinyurl.com/y89bosa6 and https://tinyurl.com/yaapqo2v).

Goals

General goals

These are the general high-level objectives:

- Proposal and implementation of service architecture and multi-tenant system.
- Study and proposal of technologies that allow multi-tenant architecture as well as its integration for the construction of the system.
- The final system must be scalable in all its aspects, both in the provision of services, volume of consumer users, integration of applications, support of an indefinite number of organizations, the volume of transactions, etc.
- The entire system must be able to be deployed in a private virtual infrastructure or in the public cloud (for example, AWS).
- Regardless of the technologies used, the system must include a unique and uniform API that allows consuming all the services, configuring the systems and subsystems, and accessing all the information that is generated.
- The system will offer high capacities for information management, as well as for analytics and business intelligence for its exploitation.
- The system will be able to obtain the safety approvals and certifications that allow compliance with the legislation of the current services associated in general and with the eIDAS Regulation in particular.
- The system proposal must minimize the cost of infrastructure and operation for the service provider.

- The proposed system must allow for a subscription-based business model and must be prepared to incorporate a pay-per-use model in the future.
- The system as a service (TXaaS) will offer or will be able to offer (because it is not limited by the new design) at least the same functionality that the existing on-prem system TrustedX already offers.
- The resulting system will have all the security protections regarding applications and web services, including a possible denial of service attacks.
- Knowledge of new technologies will be studied, analyzed and acquired from the point of view of the general innovation process of the company and not only from the point of view of its application in the specific systems and services proposed in this project.

Specific goals

Regarding the high-level specific goals of the Trusted Archive part, we proposed ourselves the following:

- Different storage services will be analyzed as service, and one will be chosen for integration into the system.
- Distributed Ledger technologies as well as different DLT and/or Blockchain networks will be studied and analyzed, and one will be chosen to integrate into the system.
- A trusted filing system will be designed and implemented.
- An API will be designed and implemented for the new service that will be integrated into the general API common framework.
- A web application will be designed and implemented to validate and demonstrate the new service.

Stakeholders

There are many stakeholders or interested parties in the results of this project, and we have divided them into four groups: 1) Consumers of the service and system, 2) Internal, that is, the Entrust Datacard company, 3) The University and the students, and 4) The market.

Service and system consumers

End user who will have a very convenient, safe and reliable system to practice the electronic signature as opposed to current smart card-based systems (such as the DNIe) or USBs that force them to carry an additional device that can only be used in a personal computer (PC) and not always compatible with the host system. With the proposed system you can use it on any device, be it a PC, a tablet, a mobile phone, a SmartTV, etc.

Application developers who introduce electronic signature and security based on signature identifiers in business, corporate or public use applications. These users will have a complete and uniform collection of APIs and support tools such as the developer portal. They will also have data analytics tools and business intelligence on all applications developed in operation.

Organizations (companies or public bodies) consuming the service that will allow them to practice a subscription model, reducing costs for outsourcing.

Administrator users / operators of the consumer organizations of the service that will have a single graphical administration console and powerful tools for operation and management, data analytics and business intelligence.

Administrators / operators of the TXaaS system will be able to perform their functions in a simpler and less expensive way, since it is a multi-tenant system and has graphical consoles and unified tools that give a unique and uniform view of the entire system through visualization, data analytics and business intelligence.

Business-oriented users in the Marketing, Commercial, Business Development, Finance, Management, etc. departments. They will have data on the exploitation and use/consumption of the services by end-users, developers, companies, organizations, etc. with the possibility of multiple segmentations such as temporal, geographic, certification authority, etc. Additionally, they may have a tool that automates the pay-per-use, that is, to account for the consumption of users and to make charges for consumption on a regular basis, for example against credit cards, etc.

Compliance and security personnel (eIDAS Regulations, and others) both at the service provider level as they have a uniform system to which the security and compliance audit is standardized for one organization and is replicated in multiple organizations. For the staff of consumer organizations, most of the compliance and accountability process is outsourced.

Entrust Datacard Company

The project allows you to **improve an existing product** (TrustedX) and create the bases of a new product (TXaaS) to be exploited.

The project allows you to **acquire new knowledge and mastery of new technologies** that you can apply horizontally throughout the company, whether in other existing products or new products or services.

The project allows you **to improve your internal innovation processes**, as well as use it as a tractor for new initiatives to continue the project or create new ones.

The project allows you to **expand and improve your image as a leader and innovator** in the market due to the direct relationship with the University, as well as the dissemination of the project itself and the results achieved (TFG and TFM reports, demonstrators, etc.).

Students and the University

The **University improves with business collaboration** both at the institutional level and at the level of the teachers and students who participate in this type of project. There has never been so much competition at the university offer level, nor has there been so much demand from society in terms of specialization and especially at the ICT level, nor has there been so much speed in technological advances, which makes university-company collaboration a factor determining in both directions.

The **students and university professors** who participate in this type of projects obtain from this collaboration a direct practical update of the business world that is essential to guide both groups in their initial career and in the focus of what society needs and demands.

The **university students** who participate in this type of projects not only amply fulfil their academic commitment but also obtain direct financing that helps them improve their quality of life, as well as promoting their professional project with many possibilities of joining the collaborating company.

The Market

The **market** benefits directly from the results for the progress made in the project and above all, for the creation of a superior competitive framework. The innovations and

results obtained in this type of project can be disruptive in that they are a perfect framework for adventure and free experimentation without ties or preconceptions.

The results of the project may affect the current competitive context, impacting **competing companies**. These projects are an incubator for new ideas and proposals that mainly seek differentiation (innovation) with the competition.

Contextualization

Entrust Datacard

Entrust Datacard[1] is a privately held company founded in 1969 with headquarters in Minneapolis, Minnesota, USA. Its origins are related to the financial sector and the creation and commercialization of hardware and software systems for the generation and personalization of credit, debit, etc. cards.

Starting in the 90s, the company diversified and entered the government and business market with the generation of identity documents, travel documents, passports, as well as identity and control cards for companies.

Until 2014 the company was called Datacard, and from this year the company changes its name to the current "Entrust Datacard" and expands its business strategy by adding the electronic and virtual world to the physical world of identity in which it operated until then. Datacard acquired in 2013 the company Entrust[2], a world leader in PKI technology and certificates for the electronic identification of people, services and machines.

In its digital business expansion strategy, in 2017 Entrust Datacard began its consolidation in Europe with the acquisition of several companies, one of them the Spanish Safelayer[3] at the end of 2018 Safelayer was a competitor to Entrust in the field of PKI and electronic identification, however, Safelayer had electronic signature technology, its own market niche in Europe and Latin America, and a group of experts specialized in PKI and eIDAS (European regulation) that complemented the Entrust Datacard group.

This project began in September 2019 within the framework of Safelayer already within the Entrust Datacard group with the aim of innovating around PKI and electronic signature technologies by introducing new technological paradigms such as cloud

[1] https://www.entrustdatacard.com/about/overview
[2] https://en.wikipedia.org/wiki/Entrust
[3] https://en.wikipedia.org/wiki/Safelayer_Secure_Communications

computing, artificial intelligence and machine learning, Distributed Ledger and Blockchain, APIs and API management systems, responsive web applications and Rich Internet Applications (RIA) frameworks, etc.

eIDAS Regulation

Until the entry into force of the European Regulation eIDAS (Electronic Identification, Authentication and Trust Services[4]) in 2014 the identification and electronic signature "safe and trustworthy" was associated with use of smart cards with cryptographic chip as in the case of Spain the DNIe (electronic DNI[5]). Since 1999, the European Directive on Electronic Signature 1999/93 EC[6] was in force, regulating and recognizing the electronic signature, legally equating it with the handwritten signature.

The use of a physical smart card with a cryptographic chip and the technical complexity surrounding it and PKI technology manifested usability problems for users in applications, especially the Web. To this was added the boom of mobile devices and others such as WebTVs, etc. in which initially there was no possibility of using the physical card and today it is still a minority and not without usability problems.

The eIDAS Regulation is born as the answer to security and trust but, contrary to the previous Directive of e-Signature, taking into account the usability and accessibility by citizens. In this regard, Safelayer was innovative and an early precursor to this concept that was eventually regulated at the European level (see TrustedX European Innovation Award in 2007[7]).

This project wants to advance not only in the usability by the citizen of the security and trust of eIDAS systems, but also in the simplification of the integration and construction of eIDAS-compliant applications by developers, as well as in the facilitation of the provision of eIDAS-compliant services from both the consumer and provider point of view. Facilitating the deployment and way of offering applications and services using cloud computing, offering as-a-Service technologies and finding new paradigms for the improvement and construction of trusted eIDAS-compliant services is also a main objective of this project.

[4] https://en.wikipedia.org/wiki/EIDAS
[5] www.dnielectronico.es
[6] https://en.wikipedia.org/wiki/Electronic_Signatures_Directive
[7] https://tinyurl.com/y7pk7joq

State of the Art

In the previous sections, the general context in which the project is located has been described. As it is a technological project, the context always has to do with technology, so in this section we focus on the untreated topics of 1) competition and 2) the state of innovation at the beginning of this project.

Competitors

A technological and business project (that is, a business project) cannot be proposed without indicating the reference competitors for which the development of the project will represent an increase or advantage compared to these.

In this section we simply list very briefly those that we consider benchmark competitors. There are more, but we consider that they are perfectly covered with the selected group. Nor does this point intend to develop a detailed comparison of the competitors and conclusions of the competitive advantages to be obtained with this project. The simple enumeration of the competition already gives information on the current state of the art and where the project is located.

A significant competitor is DocuSign[8] currently the largest provider of electronic signature services although most are under other non-eIDAS laws / regulations. This competitor is a benchmark in e-Signature as a Service and in the way that it facilitates the integration and construction of applications that use the e-signature. However, DocuSign is not European nor is it a benchmark in services under eIDAS, at which point our project can obtain advantages. In this same group we can incorporate Adobe with its AdobeSign service[9].

There is a group of competitors that we could classify as Trust Service Providers (TSPs) whose main offer is that of security and trust services based on PKI technology and digital certificates. Companies like DigiCert[10], GlobalSign[11] and InfoCert[12] are our reference competence in this group. All of them have as their main service the generation and management of digital certificates, and all of them have joined the electronic signature service. However, the electronic signature service from a mentality or classic technological context, or in other words, modern technologies are not used and

[8] https://en.wikipedia.org/wiki/DocuSign
[9] https://en.wikipedia.org/wiki/Adobe_Sign
[10] https://en.wikipedia.org/wiki/DigiCert
[11] https://en.wikipedia.org/wiki/GlobalSign
[12] https://www.infocert.it/

comparatively, very different from the strategy of competitors DocuSign and Adobe/AdobeSign, which we consider much more successful and therefore reference in e-signature.

The world of technology and business is very competitive and must be viewed from a holistic and not a concrete point of view. Competitors can be seen as such or as allies depending on the perspective adopted. In our case, due to our characteristics and the current competitive context, the companies (and their associated technologies) DocuSign and Adobe can be perfectly allied, however, the TSPs companies are direct competition. With the former we find spaces where we can complement each other, but with the latter we do not.

This project aims to innovate in the creation of a proposal that unseats TSP competitors by introducing the competitive factor found in DocuSign or Adobe. Additionally, proposals are also made that represent competitive differences with DocuSign and Adobe.

Innovation

Innovation for the generation of competitive advantage is the vehicle with which this project and any business project or of any kind is approached. In the academic world, the objective of R&D and innovation also seeks competitive advantages, although the business and economic perspective is not always in mind, but the competitive nature is always and is also fundamental to the pursuit of objectives and achieving progress.

Innovation, and Research and Development, is now a must in all businesses, not to be a leader, but to survive. The art of adapting to the rapidly changing technology landscape cannot be improvised. A product or a business can improve a little by updating an existing technology, however, they can revolutionize and explode by embracing a completely new technology. Where a company ranks between the two extremes defines it in terms of long-term market leadership.

What is this Project? It can be seen as an Innovation project that seeks to improve an existing product, although what it certainly pursues is to revolutionize the way we can deliver value by adopting completely new technologies.

Scope

The challenges and barriers are quite common, although if any particular is identified, especially in new technologies and innovation, e.g. that the existing tools do not fulfill or do not function as expected, that the assumptions and theoretical proposals are not correct, etc.

Proposed Scope

General Scope:

- Propose, design and implement a complex “TXaaS System” made up of several parts that must be carried out jointly and separately (the 3 projects) that will result in several demonstrators, one general and others partial.

Specifics Scope:

The specifics scope is to develop a Trusted Archive. To fulfil the desired objective, it is also necessary to create a portal to interact with the documents and the Blockchain. The following tasks will be carried out:

- Study about the trust of Blockchain.
- Comparison of different Blockchain.
- Choose a storage service.
- Design and implement API storage.
- Implement the functionality of registering a hash in a Blockchain transaction.
- Design and implement the Graphical User Interface (GUI).
- Integration in TXaaS.

Challenges and Barriers

The following risks are the main barriers that we have to take into account:

- The technologies and tools selected ultimately do not meet the perspectives put on them. When working with very new technologies and tools, sometimes the choice is not appropriate and makes you delay or minimize the possible results or even not achieve the expected results.
- Attempting to comply with the Regulation imposes certain technical requirements that condition the design and development of systems that may suffer a major setback if the regulation changes or becomes more nuanced, especially in young

regulations such as that of eIDAS. In the case of eIDAS, there are still technical uncertainties that must be assumed and that could change.

- Time. The duration of the TFG is limited and therefore, due to poor planning, the service may not be finished for the presentation of the project.
- The volatility of the cryptocurrency. The main and almost all Blockchain public are very volatile, their value varies continuously, and in a matter of weeks or months, it can double or triple its value. It could have an impact on the final product value and costs.

There are other risks that are minor, but can influence the project:

- Bureaucracy. In a large company, when requesting resources, it may take a long time to be accepted.
- Data failure. Data loss in progress would be severe, but there are currently many resources that can be used to recover data in minutes.
- Hardware failure. A hardware failure can cause delays in completing tasks, but working with two computers mitigates this risk to almost zero, since, if one breaks, we can temporarily continue the other.

Project Technologies and Infrastructure start-up

Basic Technologies and Concepts

Cryptography, PKI and Digital Signature

Cryptography is associated with the process of converting ordinary plain text into unintelligible text and vice-versa. It is a technique that protects information from unwanted access and other uses. The two main cryptographic are symmetric and asymmetric, and each has different algorithms and purposes. The below concepts are basic to cryptography:

- Symmetric cryptography also is known as the private key that is the cryptographic method that uses the same key to encrypt and decrypt.
- Asymmetric cryptography also is known as a public key that is the method that uses two keys: the public one that can be disseminated and delivered to those interested and the private one that should not be shared with anyone. If the public key encodes a message, it can only be decrypted with the private key. The reverse use of keys is known as a digital signature, in which a message is signed

with the sender's private key and can be verified by anyone who has access to the sender's public key.

- Hash is a cryptographic mathematical operation that generates a unique and unrepeatable identifier from a piece of information. There are many types of hash functions, and one of the most important is the Secure Hash Algorithm, in its variant SHA-256 and SHA-512.

A public key infrastructure (PKI) is a set of roles, policies, hardware, software, and procedures needed to create, manage, distribute, use, store and revoke digital certificates and manage public-key encryption. This system is an arrangement that links public keys with the respective identities of entities. [5]

In this way, the parties involved trust an external organism that guarantees a public key belongs to an identity. PKI technology allows users to authenticate themselves against other users and use information from identity certificates (for example, the public keys of other users) to encrypt and decrypt messages, digitally sign information, guarantee non-repudiation of a shipment, and other uses.

A digital signature employs asymmetric cryptography and is an electronic, encrypted, stamp of authentication on digital information that guarantees information which is originated from the signer, and that the message was not altered in transit.

Web Services

There are many definitions of what web services are, henceforth the complexity to give a single definition that englobes everything they do. The one we will take is the following one. A Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format. Other systems interact with the Web service in a manner prescribed by its description using SOAP messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards. The most used web-services are: [6]

- **REST**(Representational state transfer): architecture that using the HTTP protocol provides an API (Application Programming Interface) that uses each of its methods (GET, POST, PUT, DELETE, etc.) to perform different operations between the application offered by the web service and the client (web browser).
- **HTTP** (Hypertext Transfer Protocol): communication protocol that allows the transfer of information on the World Wide Web.

- **SOAP** (Simple Object Access Protocol): standard protocol that defines how two objects in different processes can communicate via XML data exchange.
- **XML** (Extensible Markup Language): standard format for the data to be exchanged.

Web Applications

By web application we understand any application that is accessed through the internet or intranet. These applications are hosted on a server and are executed (partially) by the browser of the local machine. Therefore, it is a software application that is encoded in a language supported by web browsers in which execution is entrusted to the browser. Web applications are popular due to the practicality of the web browser as a thin client, the independence of the operating system, as well as the ease of updating and maintaining web applications without distributing and installing software to thousands of potential users. These applications usually follow a three-layer architecture (Domain - View - Controller). [7]

Cloud Computing

Cloud computing is the availability of computer resources on the cloud, as its name already explains. It consists in the offer through the internet of large number of services without having to worry about the memory space or CPU power to process some task. These services can go from data storage, virtual machines, programming environments or even monitor of procedures.

Cloud computing appears in the project as Amazon Web Services (AWS), which is a web platform that offers large number of services like the ones mentioned before and the most used everywhere.

Distributed Ledgers and Blockchain

A Distributed Ledgers Technology is a consensus of replicated, shared, and synchronized digital data geographically spread across multiple sites. There is no central administrator or centralized data storage. [8]

A Blockchain is a growing list of records, called blocks, that are linked using cryptography. Each block contains a cryptographic hash of the previous block, a timestamp, and transaction data. [9]

Blockchain is a type of Distributed Ledger Technology (DLT) which is a geographically distributed database that contains all transactions or occurred events and is replicated by all participants uniquely through a consensus algorithm. Cryptography is the heart of Blockchain that protects the integrity, guarantees consensus mechanisms and allows the network to function. Asymmetric cryptography is used in Blockchain, and a hash that corresponds to the address distributed is generated with the public key. On the other hand, we have the private key that is used to perform transactions.

There are three types of Blockchain:

- Public, it is decentralized and maintained by everyone willing to assign resource (E.g. computing power), called miners/validator who received a reward for the work. Therefore, everyone can participate and access it. E.g. Bitcoin.
- Private. It is generally centralized although it can be decentralized and maintenance is mostly managed by companies that they can have access control and be managed by administrators dictating who can see, write or participle to the chain. E.g. Hyperledger.
- Hybrid or Consortium. It is a merger between public and private Blockchain. In these Blockchain, participation or consensus in the network is private. Where access to network resources is controlled by one or more entities. However, the ledger is publicly accessible. E.g. XDC.

We think it is interesting to know the following networks:

- Bitcoin is an open source protocol that acts as a cryptocurrency created by Satoshi Nakamoto and is the first and the most important in the aspect of implementing Blockchain technology. Its principal function is the transfer of its own currency, bitcoin or abbreviation BTC.
- Ethereum is an open source platform started by Vitalik Buterin based on Blockchain with the objective of decentralizing the web. It introduces the concept of Smart Contract that allows creating distributed applications. It also has its own currency called ether.
- IOTA is an open source DLT that uses the Tangle protocol (DAG), not based on Blockchain. It is developed and maintained mainly by the IOTA Foundation, which aims to exchange information and value between machines, as its primary focus on the Internet of Things.

Artificial Intelligence, Machine and Deep Learning

The easiest way to explain the differences between those areas is by analyzing the Figure 5, which defines perfectly each one of the areas and the differences and common points they have.

Although in the Trusted Status Service was only used Deep Learning, it's worth to mention the areas where it comes from and what problems each of the areas tackle. This is a proof of concept version which can evolve and finally ending up applying some machine learning techniques or AI.

We would think of Artificial Intelligence as some programs that are able to act like humans. Those that have can take decisions or adapt to the circumstances, the concept is very big and it includes a lot of areas like could be robotic, logic or machine learning.

This last area just mentioned is a little part inside the AI, it includes the study of the computer algorithms that are able to improve automatically through the experience of training with similar data. It builds a mathematical model based on this data in a way that can be generalized and used on other datasets and still have good performance. There are different types of models that are included on this area but the ones we will care about are artificial neural networks.

ANN is a similar construction to the brain where each cell passes the information in a way that the other can understand, it starts from the input that is modified accordingly to obtain the desired output. Inside all the types of ANN, we find Deep Neural Networks which are ANN with large number of hidden layers.

Deep Neural Networks (Deep Learning) is a class of machine learning algorithm that uses multiple layers to extract higher level features from the input. This types of networks are used a lot on computer vision tasks, speech recognition, natural language processing and many more.

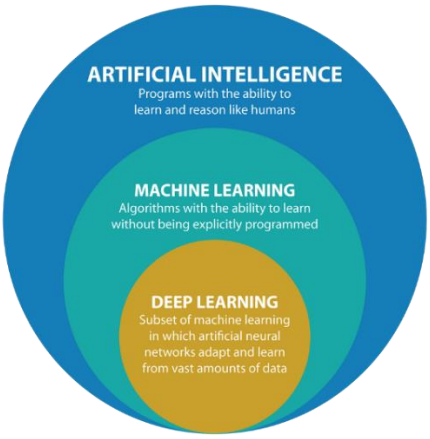


Figure 5 Comparison between AI, ML and DL

Proposed Technologies and Architecture

The proposed architecture is flexible enough to be fully deployed in a data center (private cloud) or completely distributed in the public cloud using shared computing. See the Figure 6 below in which all the proposed infrastructure technologies used in the project have been adjusted to the architecture.

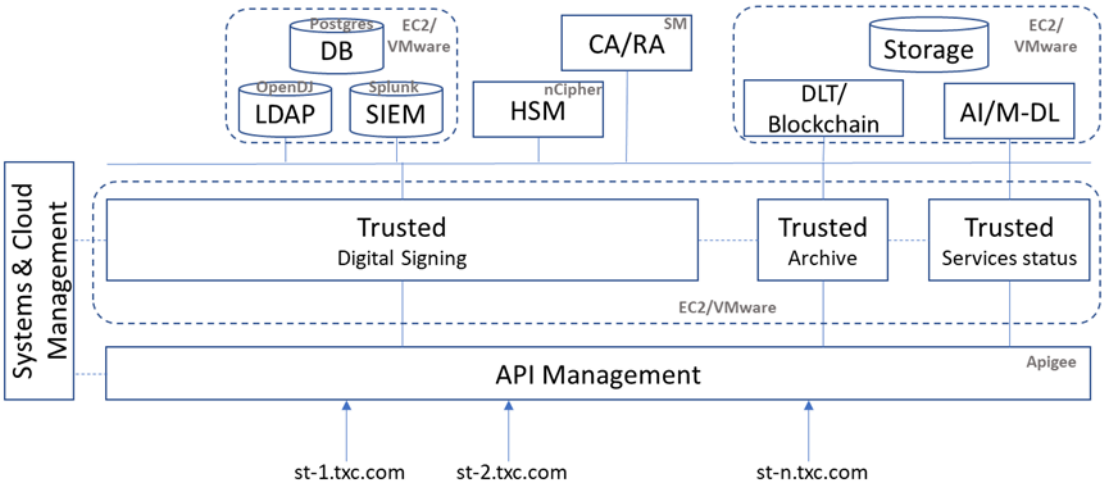


Figure 6 TXaaS proposed technologies and architecture

There are a few considerations to keep in mind:

- The “Storage” component is now implemented using cloud Amazon S3 services and there is no argument to change this.
- The “API Management” component is now implemented using Google Apigee in the cloud, however, Apigee allows to deploy hybrid architectures in which the component can be hosted in the data center.
- The “CA/RA” component will be always used as a service provided by a Trusted Service Provider, particularly, EDC is a TSP.

See “Specific Technologies Development” section in this thesis for DLT/Blockchain component details, “New Innovations in eIDAS-compliant Trust Services: Anomaly detection on Log data” of Marc Méndez’s thesis [10] for AI/M-DL component details, and “New Innovations in eIDAS-compliant Trust Services: Cloud and API Management” of Arthur Bernal’s thesis [11] for API Management.

Common Technologies and Infrastructure start-up

Virtualization Infrastructure: VMware

Both TrustedX and the technologies supporting the TXaaS project will be allocated in VMWare machines, and those machines will be in the cloud. For that reason we decided to create some scripts in order to manage the VMware instances in the cloud automatically. We want to automatically be able to start, stop, make a snapshot and start from a previous snapshot a TrustedX VM as well as personalize it to one tenant. We also wanted to be able to start and stop a CentOS machine where we will have the Database, LDAP and Splunk and start those technologies too. In order to achieve this we proposed the following scripts:

1. A script that launches the CentOS, creates the TX Database and an admin, starts OpenDJ and starts Splunk. With VMWare CLI we launch the CentOS and with SSH we execute scripts on the CentOS machine that creates the Database and starts the other technologies.
2. A script that launches the TX VM, and that given a tenant it configures the DB, the LDAP and the Splunk for that tenant. It has to configure the TX services in order for them to work with the new DB, LDAP and Splunk, this will be done with curl commands that already exist for TX. The idea is to launch with VMWare CLI a TX machine, connect via ssh to the CentOS machine that will have to be running and execute a script there that given a name of a tenant in the Database it generates its schema, tables and a user.
3. A script that given a list of the current VMs running it takes a Snapshot of a selected VM. For this script we only needed the VMWare CLI.
4. A script that given a list of the current VMs running, it stops a selected VM from the list. For this script we only needed the VMWare CLI.
5. A script that given a VM and a Snapshot from that VM it starts that VM from its snapshot. For this script we only needed the VMWare CLI.
6. A sixth script to shut the whole system properly. In order to achieve this we need to: Stop all TX services -> shutdown all TX VM -> stop DB/LDAP/Splunk -> Shutdown CentOS.
7. A seventh script that injects test users onto the OpenDJ (LDAP). There's 3 ldif that creates a different amount of users, the ldif generates 2 tenant users and their respective admins and passwords.

Cloud Infrastructure: AWS

As mentioned before, AWS was used as the cloud computing platform. When trying to offer a software as a service, the first problems that come to your mind are, where we are going to store the huge amount of information the application uses, how are they going to access this software, and how are we going to take care of the maintenance of it.

To do so, we basically used Elastic compute cloud. Services based on virtual machines where you deploy one instance that has been configured by them and adapt to your necessities or, you can use own instances.

The idea is very simple you deploy your instances there and AWS provides a public IP to access it. In this project we have 2 virtual machines, the first one being red hat operating system (Linux), this machine holds the TrustedX eIDAS Platform being this the base of the project, it was mainly used to configure the connections and some parameters to allow it work in cooperation with the other. The second machine, is a CentOS (Linux), this one has the Database, the LDAP and the Splunk. In our case, as it's a test version we only hold one TrustedX but, we will have as many TrustedX as number of clients we have. Each one of them will be connected to the centOS machine which will have direct connection to each one of the copies. Thanks to EC2 this connection is very easy to configure, it lets a set of instances be in the same private networks where the connections are very simple to configure and there are multiple parameters to filter such as number of ports we want to let them send information, type of communication (TCP, HTTP,etc.). It also lets to have a deployment configuration so every time a new instance is needed the process to deploy it is very simple.

Also another AWS services were used apart from EC2. One of them was Simple Email Service, as the name says, this service is the one that send emails when asking for a new password as a consequence of forgetting it. And also S3, which is a simple storage system on the cloud to store the different versions we are implementing, data related to the project or any other file that may be consulted by any developer.

Database Infrastructure: Postgres

Every TrustedX needs a database to store the information of users, logs, signature processes… As it has been commented before TrustedX has a small database included but since our objective is to make it multitenant we had to use a more powerful external database. We choose Postgres because is a powerful and tested technology as well as open source.

We want to use a single database to store the information of all different tenants. Normally every TrustedX has a dedicated database but in this case all the TrustedX instances will share the same one. This made that designing a multitenant database was very important since we did not want to have concurrency problems when accessing the database and we wanted to assure the privacy of the information of every tenant.

There are many different ways of achieving a multitenant database design but we had to find ourselves one that did not imply altering the behavior of TrustedX. The solutions we considered where as follows:

- We could create all the tables needed for our system only one time and then create a view for every single tenant that limited their view of those table to only the rows that belonged to them. While this solution seemed fine and scalable, we realized we could not get this to work with the TrustedX. This is due to the fact that in order for it to work we should have a new row in every table that acted as an identifier of the user, so at the end we discarded this option.
- The option we opted was to for every tenant creating a new schema inside our database and bind them to only that schema. On every schema we would then create all the tables that TrustedX needs. This solution is scalable, allows us to separate the information from every tenant and it was compatible with the structure that TrustedX follows nowadays.

Directory and Authentication Infrastructure: OpenDS and TrustedX

LDAP (OpenDJ)

The Lightweight Directory Access Protocol (LDAP) is an open and industry-standard application protocol for accessing and maintaining distributed directory information services. The OpenDJ is an open-source Java Directory Server that supports the LDAP, and it is the derivative of OpenDS with a new LDAPv3 compliant directory service that can manage hundreds of millions of entries.

As mentioned, TrustedX has its own directory for storing user information, but quite limited. For this reason, in this new TXaaS architecture, we incorporate OpenDJ for the centralized management of information for all users of the system and improve scalability.

The secure management of directory is of utmost importance so that only the respective organizations/users can access their own information. Therefore, we have configured the OpenDJ according to Figure 7.

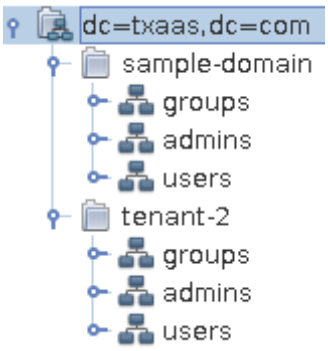


Figure 7 Structure for LDAP

This structure is made up of a hierarchy that begins with the root or Base DN (dc = txaas, dc = com), which is followed by the Organizations (E.g. Sample-domain and tenant-2) and within each tenant will have 3 Organization Units (groups, admins and users). Each of the OUs has the function of identifying and classifying the unique entries (uid). If they are administrator entries, they must be located in admins and be registered in their respective groups. If they are user entries, they must be located in users. E.g. A user entry will have the following format of the Distinguished Name (DN) format:

```
dn: uid=User_0@sample-domain.com,ou=users,o=sample-domain,dc=txaas,dc=com
```

One type of security that OpenDJ incorporates for access control is ACI (Access Control Instruction). Have to be aware of the default configuration of OpenDJ, and the logic of Deny takes precedence over Allow. We have configured 4 ACI.

The first two ACI are applied from Base DN (dc = txaas, dc = com) that have scope to all the entries of the base.

1. Allow the reading and search of all the proper attributes, except the password. It does not deny the comparison of the password (hash) required in the authentication.
2. Allow the change of password to itself.

The following two ACI are applied to tenants (E.g. Sample-domain) and have scope to all entries of the respective tenant.

3. Allow all access and operations to the group of administrators about their own tenant.
4. Deny access to everyone except the administrators and users of the organization itself

With these four ACI, we achieve that in the same directory (Base DN) can host different tenants with their own administrators and users who will only have access to their respective directory.

The user entries have different attributes, some mandatory for a logical relationship with TrustedX (E.g. Email attribute) and others optional such as the labeledURI attribute that we will use to store addresses of different DTL / Blockchain.

Authentication Delegated Process Servers

TrustedX offer SSO (Single Sign-On) is an authentication scheme that allows a user to log in with a credential to any of several related software systems.

On the one hand, TrustedX, as a product which is very flexible, and allows the incorporation of an external LDAP, even the application of embedded authentication adapts to the new situation. However, the objective of TXaaS (multi-tenant) is far from the functions of TrustedX (single-tenant). Therefore, we have created an application of independent authentication (run on TrustedX itself) to adjust to the new objectives of TXaaS.

On the other hand, this is possible because TrustedX has Authentication DPS (Delegated Process Servers) for delegating authentication to an external application.

The application (AuthNApp) must be modern, and customized for a multi-tenant environment and be compatible with TrustedX SSO, which will be one of the components of OAuth2. For this reason, we have implemented it in the Java Servlet in order to be compatible with TrustedX and following the API's guidelines for creating a DPS application.

The implementation consists of a delegated process of authentication that starts TrustedX. During the delegated process, AuthNApp will prompt the user to enter the email and the password. Subsequently, it will connect with the OpenDJ to verify the credentials and in their respective tenant (according to the identity attribute of the Identity Domains of TrustedX, from the delegation’s request). In the case of correct credentials, the delegation’s process will end and return the user attributes.

Password management and SSPR

With the application of authentication DPS implemented, we have a modern and independent application, but it lacks a necessary functionality, the password management.

An SSPR (Self Service Password Reset) is defined as any process or technology that allows users who have either forgotten their password to repair their own problem, without contacting the help desk or IT staff.

This section explains how SSPR is incorporated into AuthNApp since it is an extension and not an independent application. This functionality starts from the previous application, the use of an SMTP server (Amazon SES) and the concept of the symmetric key cryptography. And it works as follows:

1. The user requests the SSPR service for any reason in the AuthNApp of their tenant.
2. The application generates a URL with the username (email) and sends it to the user's email.
3. The user goes to the URL received in her email to reset the password.

As we can see, the Https service must be stateless. Therefore, we have created the following structure to save the information in the parameters of a URL, and to preserve the security to reset a user's password.

```
https://<tx>:8082/txaas-passwd/change-password?target=data1&state=data2
```

When the URL is generated, we must enter the fields target (data1) and state (date2) in the parameters:

- data2 are 8 characters which are generated randomly that are compatible with URL Encoding.
- data1 is the result in base 64 of the encryption of the time (Unix timestamp of the moment of creation), data2, username and tenant with a shared secret
- secret, it is a shared key between TrustedX and AuthNApp

```
data1 = ENC-B64(ENC(secret, “time\ndata2\nusername\ntenant\n”))
```

When a URL is received with the fields target (data1) and state (data2):

- data1 is decrypted with the shared secret. To get the values time, data2, username and tenant.

```
DEC(secret, DEC-B64(data1))
```

- It is verified that data2 is the same as the parameter state of the URL and that the time is within a range according to the configuration. We suppose that it is 10 minutes (time < time + 10min).
- If it complies the conditions, it shows the user a form in order to introduce the new password.

Event and Log Management Infrastructure: Splunk

Splunk is a software to monitor, search and analyse data generated by another software. In our case it is used to work with the logs generated by TrustedX. The Platform already had a log management tool inside the web interface but was not as powerful as the Splunk.

Splunk was designed to make easier to control the logs generated by the applications to detect patterns, provide measurement plots like amount of information sent a day in a simple way, and help the company to apply Business Intelligence. To be able to do that we had to send data in real-time to Splunk by deviating the logs to it via a TCP connection.

With it is easier to filter data depending the provider, once having multiple TrustedX instances, Splunk will help us organize every data apart from being able to receive information from multiple sources. Also, it includes modifiable configurations to the users restricting accesses to certain data, giving the permissions desired and filtering between data engineers and audit. We have an admin which has access to everything inside, he also has the ability to add and delete data and create the companies. Each company will have an admin that will only have access to their data, and having the possibility to create reports, and graphical samples. This admin of the company is also able to create new users under him with the same or less privileges, for example the previously mentioned audit user that the intention of it is to check some data and reports. Giving to much information to this users is not needed and sometimes is also self-defeating as they will last longer to find what they need.

As we can see in the Figure below, it provides a timeline to see the distribution of logs too see when the applications it has higher demand. In this screen we only have access to limited number of logs as the user audit-1 is from tenant txaas. It is configured in a

way that every certain period of time we make a report of the logs received so when asking for data the tool does not have to do it in real time as it is already done.

NOTE. We have used a “Splunk Enterprise Trial license” and a “Free license” (see https://docs.splunk.com/Documentation/Splunk/8.0.3/Admin/TypesofSplunklicenses) which are sufficient to probe the concepts and test the system. A non-trial license will be required if the system is deployed in production or non-trial environments. [12]

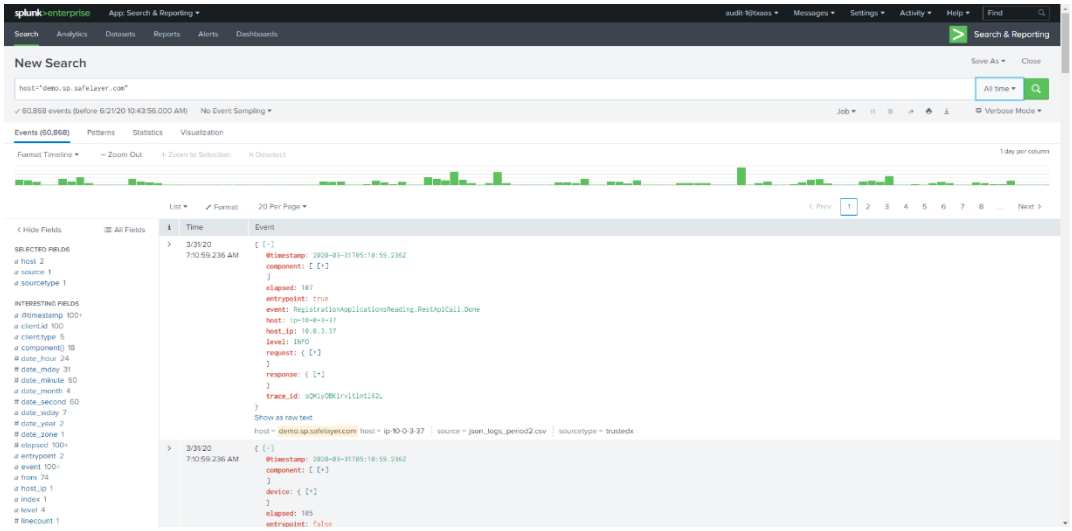


Figure 8 Splunk example of how data is presented

Digital Signature Infrastructure: TrustedX

The architecture of the TrustedX eIDAS platform is illustrated in Figure 9

Following the successful scheme of the major Internet service providers (Google, Amazon, etc.) and social networks (Facebook, Twitter, etc.), the identity architectural model implements a user-centric paradigm in which i) users can explicitly give consent/authorize the transmission of their identity data to the applications they access or ii) the identity provider (IdP) defines administrative authorization policies that the user knows and implicitly gives consent for.

The model adheres to the OAuth 2.0 authorization framework [13] with OpenID Connect 1.0 support [14] and to the SAML 2.0 Web Browser SSO Profile [15] for authentication. These protocols are based upon a pure Web infrastructure according to a RESTful paradigm and are today the technical base of the systems supporting hundreds of millions of users that perform billions of interactions and service invocations on the Web. They form, in fact, the framework behind the success of the Social, Mobile and Cloud IT triad.

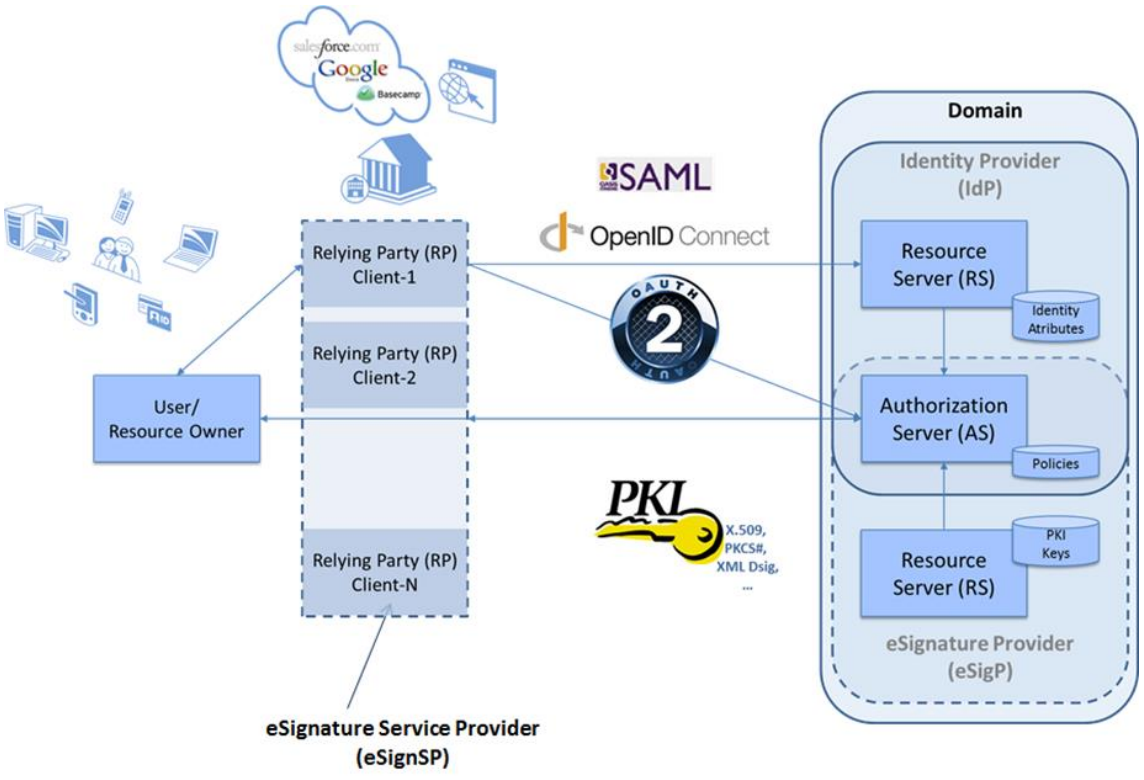


Figure 9 TrustedX eIDAS platform basic architecture

The architecture is a fusion of the classical 3-legged model already used in SAML, involving the user, the service provider (SP) and the identity provider (IdP), with OAuth's 3-legged model based around the user, the relying party (RP) and the authorization server (AS). The result is a combination of an IdP with an AS, with the SP and RP entities being one and the same with a different name and the user being the same in both cases. The advantage of the OAuth authorization framework is that it can integrate any authentication method, from the basic username and password to the high-security eID credentials and other authentication frameworks including SAML.

The eSignature Provider (eSigP) component of the framework is basically a resource server (RS) specialized in PKI identity attributes, specifically, PKI keys and certificates used in electronic signature. The RS delegates the authorization process to the AS, which in turn delegates the authentication process to the IdP. Thus, obtaining access authorization for a PKI attribute is equivalent to doing so with a normal attribute. However, access to the "PKI private key" attribute does not entail returning the value of the attribute but rather the use of the "private key", i.e., in a basic electronic signature process. For instance, in the case of RSA PKI keys for generating a PKCS #1 block [16].

The OAuth model allows applying in a standard manner a consent (authorization) process for obtaining access to the PKI key resource of the user for generating an eSignature, achieving in a natural way the explicit consent of the user who is signing.

NOTE. Text extracted from [https://link.springer.com/chapter/10.1007/978-3-658-06708-3_6] Author: F. Jordan, H. Pujol and D. Ruana. See [17] for more details.

Specific Technologies Development

This section we will explain in more detail the different technological components that we use to develop Trusted Archive, and the reason why we choose one technology over another.

However, we will mainly focus on cloud storage and DLT/Blockchain technology. Therefore, the API / GUI Management are explained in more detail in "New Innovations in eIDAS-compliant Trust Services: Cloud and API Management" of Arthur Bernal's thesis [11].

Cloud Storage and Long-term Archiving

We will start with the technology that gives name to the second word of our application, "Archive". The technology we use must have the ability to archive long-term files to last for decades, be robust and economically affordable.

The leading platforms that offer such conditions are AWS S3, Google Cloud Storage and Microsoft Azure. The decision has been easy, not because of the technological difference between the services mentioned above because all of them are similar and offer great features to comply our objective. However, because of the company's experience with Amazon services, and the better integration with the components of TXaaS by having the project services unified, we have decided to use AWS S3 for long-term archiving. See [11] for more details.

AWS S3

S3 is cloud storage that offers different low-cost storage options and is differentiated mainly by the latency in the recovery of an object and the fee to pay ($ 0.023 to $ 0.00099 per GB per month). However, each option offers a 99.99999999999% one-year durability guarantee, which means the possibility of losing one object for every 10,000,000 years of 10,000 objects stored in one year. Therefore, if Amazon meets such reliable protection, it perfectly complies the need for this project. [18]

After explaining the reliability of the S3, let us look at these important storage properties for the project:

- Amazon S3 is based on a simple key-value store and uses Bucket as a container of the object.

- A Bucket has different properties, such as the ability to configure object locking, to avoid deletion of objects once stored.
- An object is made up of a file and optionally, metadata associated with it.
- A key of an object is unique within a Bucket.
- The metadata of an object is made up of two types, system-defined and user-defined.
- Metadata is assigned at the time of uploading the file, and It cannot be modified later.
- The only way to change metadata is to make a copy or re-upload the file.

Another factor which need to consider is that S3 does not offer the service to search or index something. To get this, it must resort to a combination of other services such as S3 Inventory (generates daily or weekly index in CSV, ORC or Parquet files of the contents of a bucket) with Athena (query service with SQL in CSV, ORC or Parquet files) or Lambda (serverless computing service) with DynamoDB (relational database). However, it has a function that returns a list filtered according to the keys of the objects.

Distributed Ledgers and Blockchain

A DLT is a distributed database and does not require further explanation since it is a term that encompasses technologies that use a distribution of data through different means or mechanisms to achieve certain characteristics or objective, as is the case of Blockchain.

Blockchain is the most successful type of DTL, which has the unique feature of being made up of a sequential block to maintain the integrity of the data. We will explain in more detail in this section.

How Blockchain works and what security it provides so that we can be “Trusted” for the project's objective and the advantages compared to standard centralized technologies. On the other hand, we have discarded all networks that are not public since our objective is to offer a transparent, immutable, confidence, and evidence service, therefore, public networks comply strictly with these conditions, it is also the only one that has been proven for the longevity. Thanks to the Bitcoin that has been working for almost 11 years without interruption, and has never presented a technological problem. Because of this, we will explain Blockchain in the context of cryptocurrencies and Bitcoin.

Blockchain is a list of data records that are organized in blocks, which are arranged chronologically and secured by cryptography. This database is stored by nodes geographically distributed throughout the world with identical copies of the data.

Nodes

In a distributed system like the Blockchain, the nodes are devices or computers that are connected to each other through a P2P network, and that is responsible for acting as a communication point that can perform different functions. [19]

In the Blockchain context, different types of nodes vary according to the network, the protocol, the algorithm, and function. In this section, we will focus on the primary nodes that are essential to act directly and maintain the network, like the mining nodes in PoW or the validating nodes in PoS that are connected and synchronized by means of a consensus algorithm.

The nodes carry out the mining process, which consists of verifying transactions and added into the Blockchain public ledger. At the same time, they are rewarded with new cryptocurrencies and/or transaction fees (not all networks). During this process, the node generates a candidate block that will be validated when it meets certain conditions of the network protocols. [19]

Block

A block is made up of two parts, the header, and the body. The first contains information specific to the block, and the second stores data of transactions. A set of blocks are aligned sequentially in temporal order and are linked among them by hashes, which are called Blockchain. The composition of the head of a block can be seen in Figure 10.

A candidate block consists of a temporary block that is a set of parameters. Mainly includes the root hash of a Merkle Tree (the root hash is a hash that represents all the hashes of the tree) of the pending transactions with the maximum defined by the block size, a random number (nonce), the timestamp and the hash of the previous block. A mining node processes this set of elements to generate a hash that uniquely identifies the block.

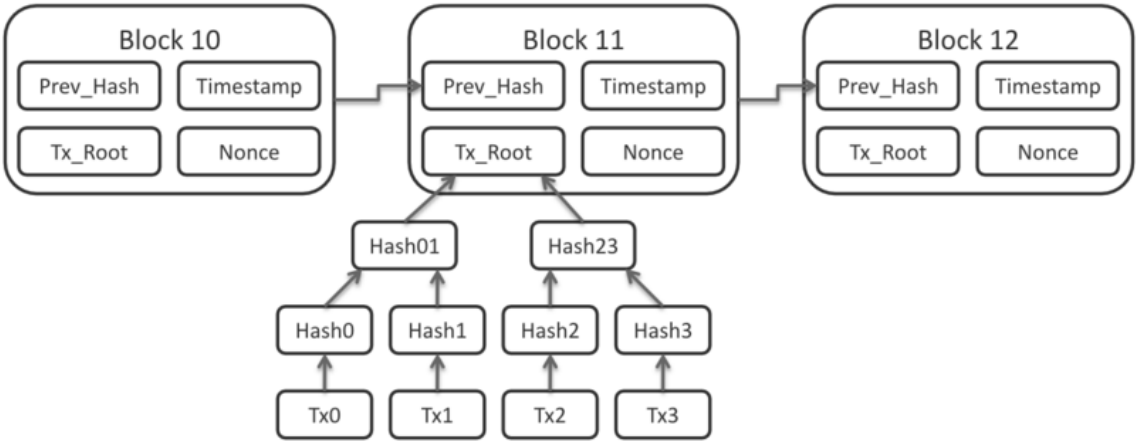


Figure 10 Block of Blockchain[13]

The hash of a candidate block is considered valid if it meets a specific value defined by the network protocol. In order that a valid block registers on the Blockchain, it must be distributed to the other nodes in the network in order to verify and accept it. [19]

Since each block contains the hash of the previous block, therefore, a new block must contain all previous blocks as a reference. To change an old block requires that to generate all successor blocks, which means that it must have the computational capacity to generate as many new blocks as successor blocks and faster than all other nodes. The older the block, the less possibility and the more cost it requires for its manipulation, therefore, protecting the network and offering immutability.

Timestamp

One of the biggest problems with a distributed system is synchronization, and time as a relative and imprecise value. The time of a Blockchain node is a local value in UTC in Unix Epoch Timestamps.

In Bitcoin, the timestamp of a block is accepted as valid if it is higher than the median timestamp of the previous 11 blocks and less than the median of the timestamp returned by all connected nodes plus two hours. As we can see, the time in Bitcoin is not exact, nor accurate since it exists up to a margin of two hours. [20]

Although the margin may seem significant, we must take into account that it is a mechanism that is designed to avoid Time Warp Attack (which allows reducing the difficulty of mining to produce more blocks in less time). However, the network itself has an internal controller by recalculating mining difficulties (2016 blocks in Bitcoin, almost

[13] https://commons.wikimedia.org/wiki/File:Bitcoin_Block_Data.png

for every block mined in Ethereum) in order that blocks are generated almost constantly (10 minutes Bitcoin, 15 seconds Ethereum). [21]

Therefore, the security of the network will be a decisive factor for the timestamp. When the network works properly and generates blocks constantly, the timestamp will have a good accuracy, but not as accurate about to UTC, because the miners will have to adjust their local time according to the previous blocks and their neighboring nodes.

Consensus Algorithm

As we commented, cryptography is the heart of the Blockchain system, and the consensus algorithm would be the mind that makes it possible.

The consensus algorithm is a mechanism that allows machines to be coordinated in a distributed environment, and it offers the capacity of a BFT system in order to ensure that all the agents which are involved in the system can agree to a unique source of truth. The following is a summary by Binance Academy about the Byzantine Generals' Problem[14]:

> *The Byzantine Generals' Problem was conceived in 1982 as a logical dilemma that illustrates how a group of Byzantine generals may have communication problems when trying to agree on their next move.*
>
> *The dilemma assumes that each general has its own army and that each group is situated in different locations around the city they intend to attack. The generals need to agree on either attacking or retreating. It does not matter whether they attack or retreat, as long as all generals reach consensus, i.e., agree on a common decision in order to execute it in coordination.*
>
> *Therefore, we may consider the following requirements:*
>
> - *Each general has to decide: attack or retreat (yes or no);*
> - *After the decision is made, it cannot be changed;*
> - *All generals have to agree on the same decision and execute it in a synchronized manner.*

[14] Byzantine Generals' Problem is a dilemma described by Leslie Lamport, Robert Shostak, and Marshall Pease. [38]

The aforementioned communication problems are related to the fact that one general is only able to communicate with another through messages, which are forwarded by a courier. Consequently, the central challenge of the Byzantine Generals' Problem is that the messages can get somehow delayed, destroyed or lost. In addition, even if a message is successfully delivered, one or more generals may choose (for whatever reason) to act maliciously and send a fraudulent message to confuse the other generals, leading to a total failure.

Binance Academy

Once we understand the problem of the generals, we will explain why the game theory[15] is important to achieve BFT. In Blockchain, especially the public one, that is highly linked to the economy of the cryptocurrency (Cryptoeconomics) that makes it possible to maintain the technology's whole infrastructure.

In the context of cryptocurrencies and Blockchain, the success is based on the result of the behavior of the nodes that are geographically distributed and must rely on mutual agreement for the validation of transactions and blocks. How are nodes made to make more rational and likely decisions in order that they can trust each other? Here is introduced the game theory and of the incentive, or the concept of Cryptoeconomics, which consists of a consensus algorithm to protect malicious activities, in order that the nodes act honestly, and at the same time, generate selfishly a competitive environment for the good common. In contrast, on account of any malicious activity of the nodes will give rise to a significant loss of money or expulsion from the network. Consequently, the most likely and rational decision is for the nodes to act honestly because they are selfish in nature who want a greater incentive for their own benefit.

Next, the Prisoner's dilemma, which is formalized by Albert W. Tucker.

Two members of a criminal gang are arrested and imprisoned. Each prisoner is in solitary confinement with no means of communicating with the other. The prosecutors lack sufficient evidence to convict the pair on the principal charge, but they have enough to convict both on a lesser charge. Simultaneously, the prosecutors offer each prisoner a bargain.

[15] It is a method by applying mathematics that is used to study human behavior about rational decision-making, which is formalized for the first time by John Von Neumann and Oskar Morgenstern. [39]

Each prisoner is given the opportunity either to betray the other by testifying that the other committed the crime, or to cooperate with the other by remaining silent. The possible outcomes are:

- *If A and B each betray the other, each of them serves two years in prison*
- *If A betrays B but B remains silent, A will be set free and B will serve three years in prison*
- *If A remains silent but B betrays A, A will serve three years in prison and B will be set free*
- *If A and B both remain silent, both of them will serve only one year in prison (on the lesser charge).*

(Wikipedia) [22]

Finally, we will explain PoW and PoS, two of the most important types of algorithms in Blockchain applying BFT at present, in which the nodes or validators represent the generals that try to reach a consensus in order to identify the unique, reliable source of a database, and besides, take into account the game theory.

Proof of Work

PoW is a system that discourages and makes unwanted behavior difficult for participants because it requires a certain workload, which is expensive enough to produce, but easy to verify. To complete this, Blockchain uses the hash function as the type of work that miners must be performed through the trial and error in generating a new candidate block. E.g. Hashcash PoW algorithm. [23]

Hashcash is designed by Adam Back to prevent spam email by hashing the content and later using it by Satoshi Nakamoto with Bitcoin and SHA256 in the Blockchain. [24]

In order for the miners to not act dishonestly, Blockchain or Nakamoto devised one of the most competent solutions for BFT by using incentives. Incentives consist of rewards for work done (valid block) with the minting of new coins (cryptocurrency) from the network and, in addition, in most networks, a transaction fee. Therefore, the nodes compete to get a valid block to claim the reward and create an environment of positive competition. On the other hand, the miners will not act dishonestly (because they run the risk of losing the resources that are invested), since they will not be able to claim the

reward, in addition to wasting a high cost of computation to try to modify new blocks that will be little useful or almost useless for old blocks, unless it is a 51% attack.

Before explaining the 51% attack, it should be noted that the PoW designed by Nakamoto is based on the idea the longest chain is the honest Blockchain since it represents the majority of the members with the most significant contribution to the network with computational power or CPU. [24]

What will happen if the majority of the network's participants (>51%) are dishonest? That would be the situation that represents the 51% attack, one of the main disadvantages of PoW, which is not totally tolerant to Byzantine failures, but, due to the high cost of the mining process and the incentive design. During these 11 years, Bitcoin has proven that PoW is one of the safest techniques and the most reliable for Blockchain.

The attack of 51 appears when a group that acts unanimously and owns more than 51% of the network's computational power and will be able to generate more blocks than the rest of the nodes. In Figure 11, the attackers will have enough power to monopolize the network by denying the extraction of new blocks to honest miners (denial attack) and being able to reverse transactions and cause the problem of double-spending (sending the same transaction more than once). Even though the attack is successful, the attacker would only be able to interrupt the network or reverse it transactions, but will never be able to carry out transactions from other users, since they will not have the private key. [25]

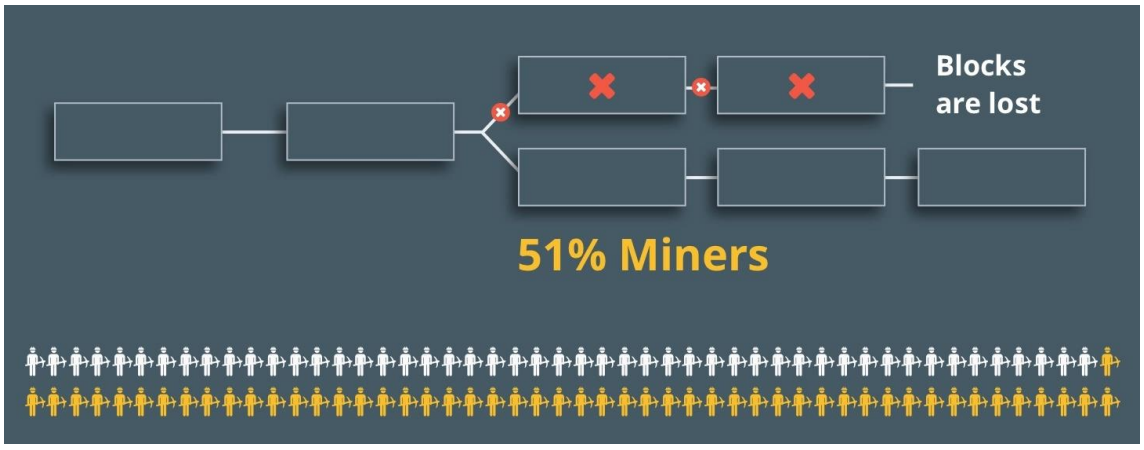


Figure 11 51% attack[16]

[16] https://cointelegraph.com/storage/uploads/view/5a39b177432a5ce6655ba1f5b3003ea9.jpg

On the other hand, Bitcoin has been able to survive for so many years is because attacking the network is not a profitable option, that requires an enormous amount of computing power. Once the network is compromised, it is value will vanish, and the attacker would not get any financial benefit unless their goal is to destroy the network, in which case the honest nodes could respond quickly and adapt via a fork, explained in Fork section.

This type of attack is unlikely in larger networks, such as Bitcoin or Ethereum. However, smaller networks can suffer from this type of attack because their hashing power and/or the value are relatively low, like Bitcoin Gold (a fork of Bitcoin), Monacoin or Ethereum Classic (a fork of Ethereum). [19]

Proof of Stake

PoS is the most plausible alternative of the PoW system, in which the mining term does not exist as providers of large external resources (CPU) to keep the security of the network, but rather as validators to provide internal resources (the network's own cryptocurrency) and a small computational resource. The most significant advantage of this algorithm consists of energy efficiency and security, in addition to offering a more efficient and faster system (TPS).

The system consists of a pseudorandom selection of a validator in order to grant the right to validate a block. Typically, the probability of being chosen is proportional to the amount of cryptocurrencies stacked, and validators are rewarded directly corresponding to their total stake, which encourages the nodes to validate the network based on a return on investment (ROI). [26]

Figure 12 Staking[17]

[17] https://www.ledger.com/wp-content/uploads/2019/10/What-is-proof-of-stake-2.jpg

The moment a validator assigns a number of coins for the stacking, they will not be able to move the coins throughout the process. During the period, if any dishonest action is taken (such as proposing invalid transactions), the system will penalize the total or partial loss of the participation according to severity. It induces nodes to act honestly because it is more profitable than dishonestly. [19]

The nodes that block more coins for the stacking are the more chance of being selected to be the next validator. Therefore, in order for the process not to favor only the wealthiest nodes in the network, methods are added ‘Randomized Block Selection’ or ‘Coin Age Selection’ for prevents large stake nodes from dominating the Blockchain’s network:

- The Randomized Block Selection method, the validators are selected by looking for nodes with a combination of the lowest hash value and the highest stake. [19]
- The Coin Age Selection method chooses nodes based on how long their tokens have been staked for. Coin age is based on the number of days the coins have been held as stake. Once a node has forged a block, their coin age is reset to zero, and they must wait a certain period of time to be able to forge another block. [19]

It is a system similar to PoW but applies greater security according to the game theory and the financial concepts. Since anyone to take advantage of dishonestly will do it if they benefit from such action. As a bet is realized with the network's own resources that must be acquired previously, if you perform any malicious activity on the network, it is equivalent to harming itself, which for rational reasons, will not do it.

On the one hand, 51% of attack still exists, but it is even more difficult to happen since it requires owning 51% of the issued coins.

On the other hand, it solves one of the centralization problems of PoW (concentration of hash power by groups of miners with great purchasing and financial powers), because coins which are distributed have better decentralization and require less financial power to participate in the network.

DPoS is a variant of PoS, which follows the same fundamentals but applies the concept of votes according to the number of coins stacked. These votes are used to select governors who will be a select group of validators who ensure the network (a governance system). This system requires fewer validators; therefore, it achieves a better network performance, but with a lower degree of decentralization. [19]

Finally, we can observe a summary of the differences between PoW and PoS in Figure 13. PoS could be an alternative with better performance, ecologically sustainable, and safe than a PoW system, but with the disadvantage of a complex system that has not been thoroughly tested in a larger environment (we'll see it in the Ethereum 2.0) that could be affected in future. Because it follows a financial trend, therefore, the rich will be richer (coins) and could threaten the decentralized system of the network.

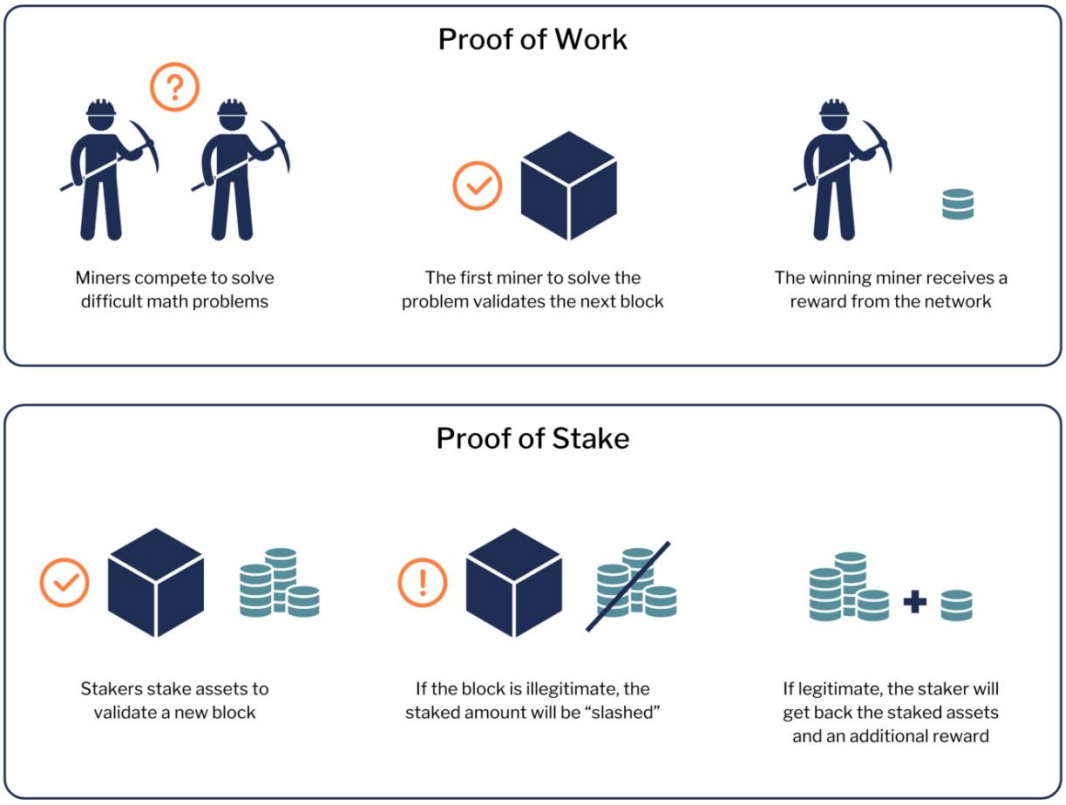


Figure 13 Proof of Work vs Proof of Stake[18]

Fork

Any software requires maintaining, either to correct errors or add new features. Blockchain is not the exception, but it has it is particularity, since, for any operation, it requires the consensus of the majority of the participants. When a change occurs, it is called Fork, and we can distinguish two types, Soft and Hard.

Soft Forks are slight changes in the network protocol that is backward compatible. Therefore, the nodes with the old version and the nodes with the new version work in parallel in the same network. [27]

[18] https://miro.medium.com/max/1400/1*PkT61WVy93x_RQWtOaxi0Q.png

However, the Hard Fork is the major change that renders the old versions incompatible, whether intentional or accidental, causing two networks to exist at one point, one with the old protocol and the other with the new one (see Figure 14). Therefore, nodes that are intended to operate according to the new protocols need to be updated. On the contrary, there will be a permanent division and the emergence of a new Blockchain. Like what happened with Ethereum and Ethereum Classic, Ethereum in 2014 updated the network to reverse exploitation in the badly encoded code of The DAO[19], in which there were detractors and supporters that create a division of the chain in Ethereum (updated) and Ethereum Classic (original), each has its own value and identity that shares a common origin, and includes all transactions and cryptocurrency that are duplicated before Hard Fork. [27]

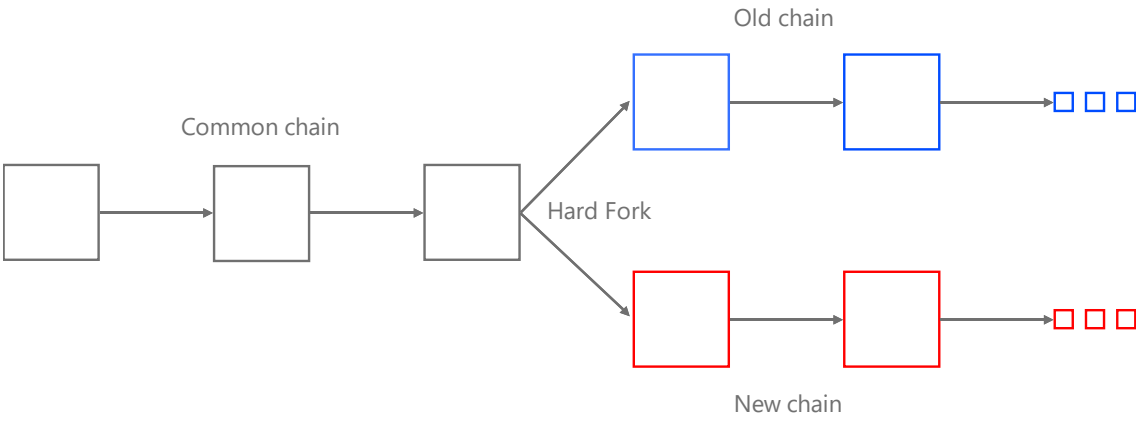


Figure 14 Hard Fork

It is also used to create new Blockchain from an existing one with different ideals, such as Bitcoin Cash, which is a Hard Fork of Bitcoin that has large network supporters and high market value.

Transactions

Once we have seen how Blockchain technology works internally, we will explain how a transaction is realized.

A transaction is an operation that is transmitted to the network in order that it performs some action, and is stored in a block as a record. This action is usually the transmission of the network's own cryptocurrency but may include other information according to the

[19] The DAO was a digital Decentralized Autonomous Organization.

protocol. But generally, it must include essential information such as the number of coins, the address of the recipient, and the signature.

This requires a wallet that has an accessible private key. This wallet will have an address that uniquely identifies it, which according to the network, uses a specific public-key cryptography algorithm. However, most follow the following pattern to generate an address from the private key.

Private Key → Public Key → Address

In which the nodes will be able to recognize, verify and accept a transaction if it is carried out from an address with the corresponding signature of the private key, and the process of a transaction follows several steps before being completed. We can see in Figure 15.

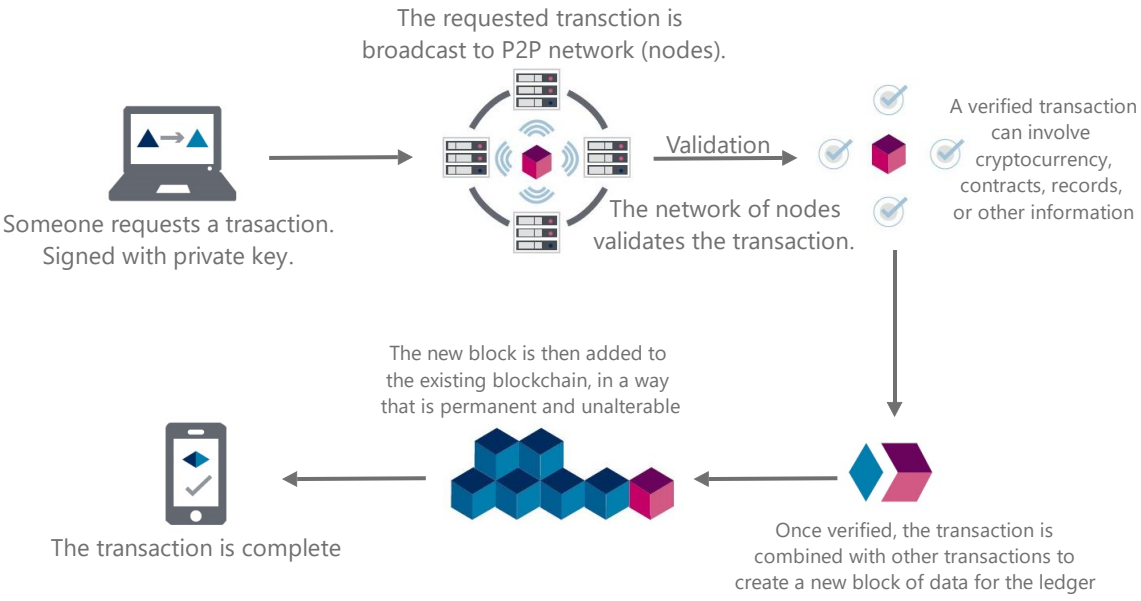


Figure 15 Transaction

Once completed, the transaction will be permanently stored in a block and available for consultation from any node that has a copy of the Blockchain. As the information is not encrypted, it will be visible and readable for any interested entity (according to the protocol and the purpose of the network).

Furthermore, this will be the concept that we will apply to register a hash of a document in a Blockchain transaction for eternity.

Public networks

Next, we will see the proposals of the main public DLT/Blockchain networks that we consider interesting that allow embedding additional information in a transaction.

Bitcoin

Bitcoin is the first and the most famous Blockchain which is launched in 2009 by Satoshi Nakamoto. Its main function is the transfer of its BTC cryptocurrency, and it uses PoW Hashcash as a consensus algorithm. It generates a block every 10 minutes and has the ability to perform 4 transactions per second (tps).

The network protocol has a parameter in the transaction called OP_RETURN, which is an opcode that is used to mark a transaction output as invalid. But it can be used to embed arbitrary data to store it on the Blockchain. However, for the majority of the Bitcoin community, it is considered irresponsible, because it damages the main functionality of the network by occupying memory which is not intended for that purpose. [28]

Nowadays, Bitcoin is one of the oldest, the most secure and currently robust network of all existing public Blockchains, since it currently has more than 10,600[20] active and public nodes which are distributed throughout the world, and provides more than 105,000,000[21] TH/s of computational power to secure the network.

Ethereum

Ethereum is a decentralized platform that was launched in 2015 and is considered as the second most important Blockchain at least in the economic and/or developing factor, since it is cryptocurrency occupies the second place in market capitalization, behind Bitcoin. However, Ethereum has 7409[22] nodes that provide more than 180[23] TH/s and has the capacity to perform 20 tps (currently reaches 10-15[24] tps), which generates a block every 14 seconds.

On the other hand, the network is using KECCAK-256 algorithm and PoW (Ethash) as a consensus algorithm, but due to scalability, it is in the process of updating to a PoS system known as Ethereum 2.0, which can reach 3000 tps. The launch is expected to occur in 2021, after multiple delays.

[20] https://bitnodes.io/
[21] https://www.Blockchain.com/charts/hash-rate
[22] https://www.ethernodes.org/
[23] https://etherscan.io/chart/hashrate
[24] https://etherscan.io/chart/tx

One of the distinctive features of this network is the Ethereum Virtual Machine (EVM) that can execute Smart Contract (the first Blockchain to introduce this concept) and is responsible for processing transactions or any action on the Ethereum network. The EVM works through a stack-based architecture (last in, first out), it has RAM and also the storage, and they are all linked to Gas, which is used to pay for operations on the virtual machine. [29]

Gas is the unit used to measure the fees required for a particular computation. Gas price is the amount of Ether you are willing to spend on every unit of gas and is measured in gwei. Wei is the smallest unit of Ether, where 1×10^{18} Wei equals 1 Ether, and 1 gwei is 1×10^{9} Wei. E.g. Normal transaction requires 21000 Gas, and Gas price is 10 gwei = 0.00021 Ether or ETH. [29]

Although we do not go into detail how a Smart Contract works, it is interesting to know that they are programming codes or instructions which are stored in a Blockchain that is executed once a request is received and/or meets specified requirements. [30]

EOS

EOS is a Blockchain platform for decentralized applications (DApps) or Smart Contracts, which was launched in 2018. It works similarly to Ethereum and has fields for the data embedding in transactions. On the other hand, it uses DPoS as a consensus algorithm that gives greater speed and scalability with the ability to reach 4000 tps, but currently reaches around 50[25] tps. However, the resource management system is totally different, at the same time, is novel and risky.

The EOS network functions as a shared resource supercomputer that is classified into three components, CPU for processing, NET for data transfer, and RAM for the temporary storage of application logic. And any interaction requires the use of these resources, from highest to lowest intensity, depending on the application. The right that users have for the use of resources depends on the amount of the stagnant cryptocurrency EOS and the capacity of the network, therefore, the higher the amount of stagnant cryptocurrencies that a user has, the greater the right to use the resources of the network over a period of time (usage statistics are reset every day). [31]

A wallet could be considered a basic application that requires minimal resources for transactions. The higher the stagnant EOS, the higher the number of transactions can be carried out in a day.

[25] https://eosnetworkmonitor.io/

However, problems raised with network congestion that many wallets were unable to transactions (require thousands of dollars in stagnant EOS) that causes a new usage model or ecosystem of resource rentals.

IOTA [32]

IOTA is a DLT technology whose objective is to securely allow the exchange of information and value-focused on the IoT, which was launched in 2016. It is a network that does not use Blockchain, and it is transactions recorded in the form of the Directed Acyclic Graph (DAG) in its Tangle network, which enables commission-free transactions, low network latency and greater increased scalability.

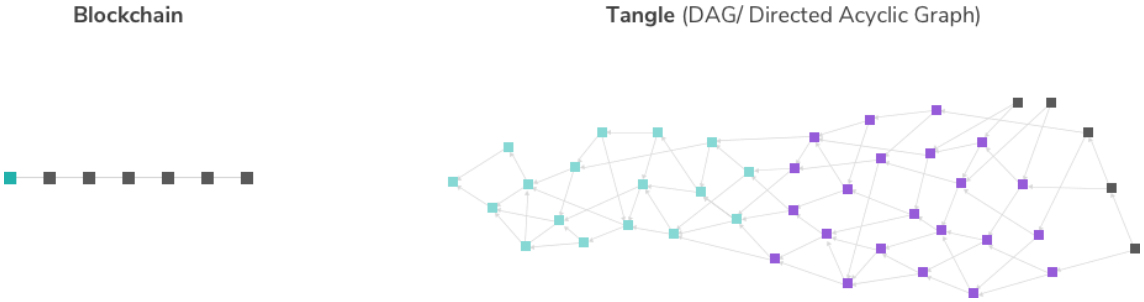


Figure 16 Blockchain vs Tangle (DAG)[26]

The main characteristics of this network consist of the consensus system, the operation of the transactions, and the nodes in a DAG system.

Since there no exists the concept of a block, transactions are unique units that are interconnected with each other, with behavior like the vertices of a DAG, and it is stored by the nodes.

On the one hand, since there are no miners or validators, the nodes assign a transaction its own weight and an accumulated weight (sum of all the weights of the related transactions). When a new transaction is added, it must be selected and validated two transactions pseudorandomly not that are no validated (cumulative weight is taken into account). Therefore, network participants are jointly responsible for the transaction validation, in addition to a small cryptographic calculation. [33]

On the other hand, in the current state of the network, the nodes do not confirm the transactions directly, for this, it requires a coordinator (centralized) because they do not use any consensus algorithm, such as PoW or PoS, therefore, they are unable to trust

[26] https://iota101.info/

each other and the consensus is obtained by a reference of transactions of a coordinator (milestones), which is sent from time to time.

Finally, nodes validate transactions only if they are related directly or indirectly by a milestone, and unrelated transactions are considered invalid and will never be confirmed. The logic consists of more people use the network, more transactions are carried out and more possibility that a transaction is related to a milestone. However, if there have little activity on the network, more the addition of malicious participants (requires little resource), there is a high possibility of producing unconfirmed transactions.

The objective of IOTA is to exchange information. Therefore, it has fields in the transaction to store arbitrary data. But we must take into account other characteristics and concepts, such as permanode and snapshots.

A snapshot is a moment in which a node decides to delete all the old transactions to realize a snapshot and preserve only the valuable information.

Permanode is a node that never snapshots; therefore, it preserves all transactions which are made on the network. Since the storage has a high cost, currently there only exists one (Tangle official node) public permanode.

Ethereum and IOTA

The factors in choosing one network or another have been, the confidence, the cost of a transaction, and technological features.

On the one hand, EOS was directly ruled out because it is a technology too young and risky that has shown problems of congestion and because of the lack of clarity and problem in its ecosystem of use.

On the other hand, we consider that the Ethereum network is more suitable and is the one that we will use for our project, because, both Ethereum and Bitcoin have a large community that supports development and maintenance. But Bitcoin has a high cost in the embedding of the arbitrary information because it is not designed for such function. However, Ethereum is built on these ideas, in addition to being more economically affordable.

Therefore, the difference of cost has been one of the decisive factors, as we can see in the calculations which were made in December 2019, during the analysis of the networks, that Ethereum has a cost of up to 10 times less than Bitcoin for register a 64 byte of hash 512.

Bitcoin at $7152, December 2019:

$$Tx_{feeBTC} = \left(226^{27} + 64\right) bytes \times 18^{28} \frac{satoshis}{bytes} = 5220\ satoshis = 0.00005220\ BTC$$

$$Tx_{feeBTC} \times \$7152 = \$0.37$$

Ethereum at $127, December 2019:

$$Tx_{feeETH} = \left(21000^{29} + 68^{30} \times 64\right) Gas \times 10^{31} \frac{gwei}{Gas} = 253520\ gwei = 0.00025352\ ETH$$

$$Tx_{feeETH} \times \$127 = \$0.037$$

Finally, as an R&D project, we want to include IOTA in our project, because we consider it to be an interesting DLT technology that does not use Blockchain, in addition to offering transactions without fees.

The explanation of different networks only consists of an introduction to the technology, since each one has different and unique characteristics that we do not cover in this thesis. However, the networks that have been chosen (Ethereum and IOTA) explain in more detail how a transaction is composed in the Implementation section.

Requirements

This section explains the functional and non-functional requirements that the Trusted Archive system must meet in order to guarantee good quality criteria.

Functional requirements

The following list describes the functional requirements of the system.

- Being able to upload files to cloud storage. This functionality provides a means for users to upload files to the cloud through direct communication to avoid that the file travels through different layers of the system. And it also includes the

[27] Average bytes required for a normal transaction.
[28] Average price expenses needed to be processed before 2 blocks, 20 min.
[29] Gas required for a normal transaction.
[30] Gas required for data non zero (bytes non zero)
[31] Average Gas price expenses needed to be processed before 2 blocks, about 22 seconds.

mechanisms to generate or obtain the important information of a file, such as the hash of the sha512.

- Being able to download files from cloud storage. This functionality provides a means for users to download files from the cloud through direct communication avoid that the file travel through different layers of the system.
- View information of those files uploaded to cloud storage. This functionality allows users to view the relevant information of the files stored in the cloud. Such as hash of transactions in the DLT.
- Being able to notarize files on DLT or Blockchain networks. This functionality provides the ability to record a hash of a file in different Blockchain or DLT networks, in addition to actualizing the information of a file in cloud storage consistently.
- Being able to authenticate through the TXaaS OAuth2 ACG system. This functionality provides the ability for the user to authenticate through the system that has already been implemented of the TXaaS OAuth2 service. This function allows with a single sign-on authentication to access all applications of TXaaS without re-authenticating.

Non-functional requirements

The following list describes the non-functional requirements that the system demands to complete certain criteria in quality.

- Usability. The web application must provide the user with the control of which information will be received and which information will be displayed. In addition, the application must be intuitive to avoid the existence of a long learning curve.
- Design. The web application must have a modern appearance and a design that realizes current standards.
- Security. In addition to complying with security standards in web communication, the system must be secure to prevent unwanted access, unauthorized data manipulation.
- Efficiency. The system must perform competently to return responses with the lowest possible latency.
- Scalability. The system must be able to adapt to new conditions without impacting the system.
- Availability. The system must be available practically 24 hours a day.

Specification

Once seen the functional and non-functional requirements, this section will show and explain how we design the Trusted Archive system.

Architecture

Figure 17 exposes the scheme of the system architecture. As you can see, the scheme seems complex, but once the different components have been explained, it will be easier to understand.

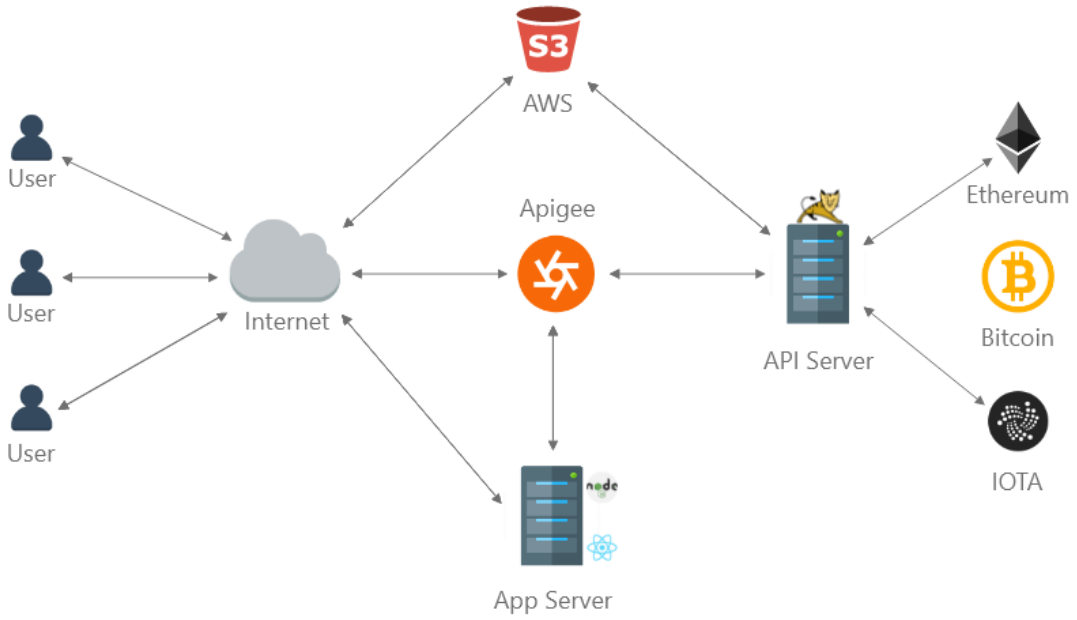


Figure 17 System architecture

Users

Users connect to the App Server in order to log in and obtain the web application. Once the application is obtained, you can contact Apigee to consume the system's services.

On the other hand, it also has direct communication with S3, but this will only be able to upload or download files through a previous request to Apigee. In this way, we avoid that the files travel through different layers, which would greatly reduce performance, and we provide security to our storage.

App Server

This server uses Nodejs to offer two primary functions. The first is offers to users to log in to the system, performing a communication flow with Apigee to obtain the tokens of

the OAuth2 ACG. The second is to host the web application and only send it to authenticated users.

On the other hand, the web application is implemented with ReactJS, mainly using the Material UI framework.

Apigee [34]

It is the main manager that provides an abstraction for the APIs of the backend services and greater security to the system.

Apigee offers all the necessary services to the application and, in turn, communicates with the API Server in order to obtain or carry out all user requests, providing security so that only authenticated users can access its resources.

NOTE. We have used a "Apigee Evaluation/Trial License" (see https://cloud.google.com/apigee/pricing and https://apigee.com/about/cp/apigee-edge-free-trial) which is sufficient to probe the concepts and test the system. A non-trial license will be required if the system is deployed in production or non-trial environments.

S3 Bucket

S3 is the service we use to archive long-term files. All files of the users of different tenants are stored in a single Bucket.

On the one hand, as mentioned, the S3 has direct communication with the user's application to upload and download authorized files.

On the other hand, the Bucket is managed by the API Server, who is the one that authorizes the access controls and the storage logic.

API Server

This server provides a RESTful API to Apigee for system functionality and is hosted on a Tomcat on AWS EC2.

The API is implemented with Java by using different libraries: JAX-RS for RESTful API, AWS SDK S3 for interactions with the S3 Bucket, Web3j for implementation and communication with the Ethereum's network, Iota-ledger-java for implementation and communication with the IOTA's network and finally, iText5 to analyze PDF files.

Ethereum and IOTA

They are nodes of the respective networks that allow interaction to consult or send transactions. On the other hand, it can be seen that the Bitcoin network is not connected to our API Server. However, the purpose of this representation is to indicate that any

Blockchain network that has the same transaction system as Bitcoin is compatible with our system. Therefore, adding a new network to our system would be simple to implement, and it will be enough to program the requirements of the new network by using its respective library and enabling calls in Apigee.

Interaction Diagram

Next, a high-level overview use cases will be presented by using interaction diagrams. First, we want to outstand the importance of Apigee, since it is an important point in which we provide security in the separation of the contents of the storage system. As we can see in the following diagrams, it consists of obtaining the tenant and the UUID of the user by using its Access Token. Therefore, a user will not be able to access another user's storage resources without their previous authorization.

Before starting to use our application, the service must be activated to obtain and configure the user's DLT address following the diagram in Figure 18. This interaction will not be present in the application but must be done by the TXaaS services, or by other means of provisioning or administration.

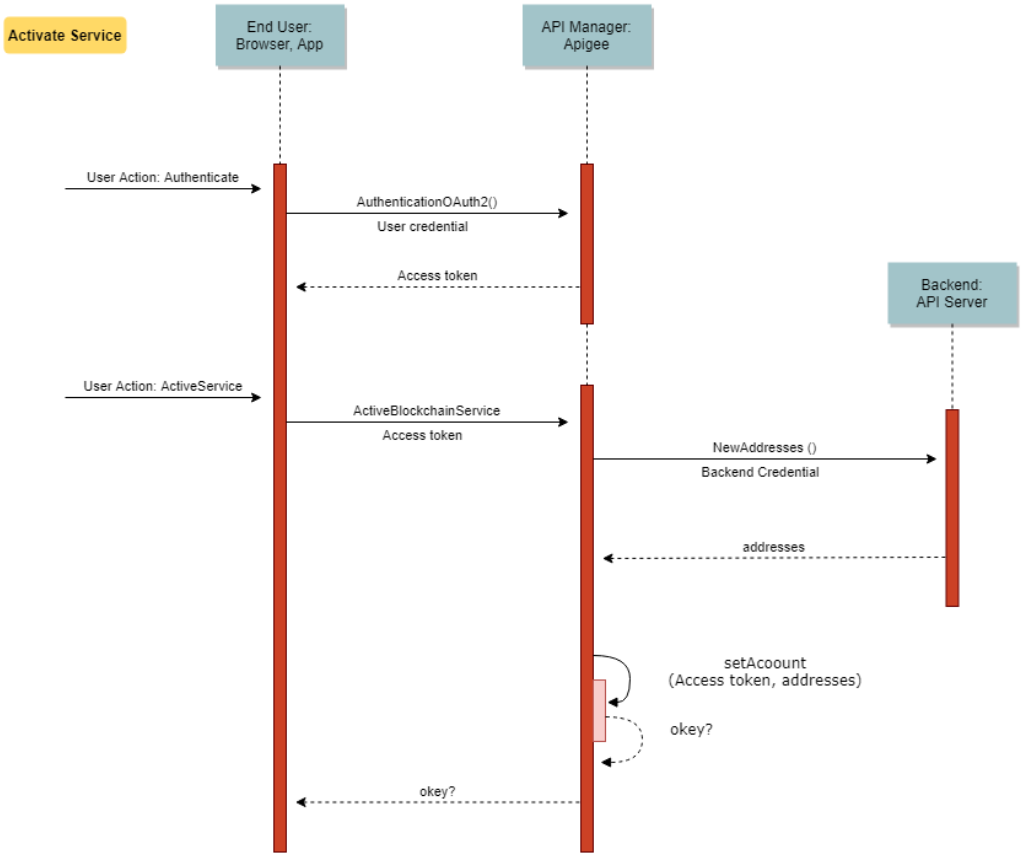


Figure 18 Diagram. Activate Service

File upload

This use case consists of uploading a file by the client to our cloud system. For more detail, it can be seen in the diagram of Figure 19.

1. The client requests a pre-signed URL to upload to Apigee, which in turn communicates with the API Server to generate the URL. Including the tenant and the UUID of the user according to their Access Token.
2. The application uploads the file to S3 by using the URL.
3. In the case of PDF documents, once the upload is completed, it will notify Apigee to perform a signature check.

Object List, Object Information and Notarize

The objective of this use case is notarization. Still, during the process, it involves other uses such as the listing of files or detailed information of a file that requires a sequential order of actions. Therefore, we expose it together, as shown in Figure 20 of these use cases.

1. The user initiates a request to Apigee to obtain a list of objects with a time interval, which will return a list with the essential information, such as the name, partial hash and existence of signature or notarization. In case of sending a request without parameters, it returns a list of dates that contains some object.
2. When the user selects an object from the list, he initiates another request to Apigee to get all the detailed information of the object. Includes a pre-signed URL to download the file.
3. The user's last action consists of requesting Apigee to notarize the file on DLT/Blockchain.

View / download file

This use case consists of viewing or downloading a file previously that is uploaded to our cloud system. These actions can be performed from the status of the file information. We do not have a sequence diagram because it consists of a simple HTTP request to S3 that uses the URL for view or download.

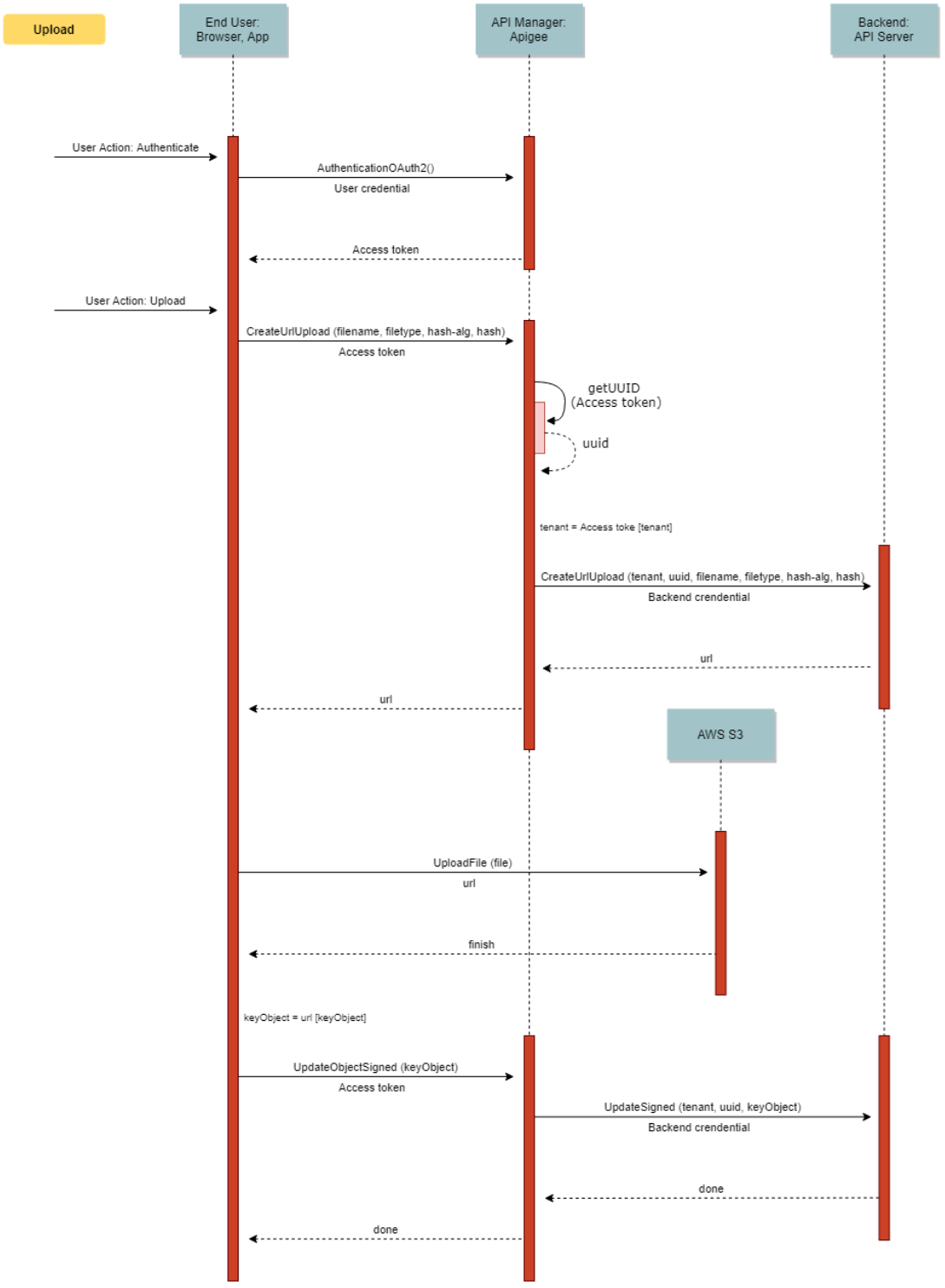


Figure 19 Diagram. Upload

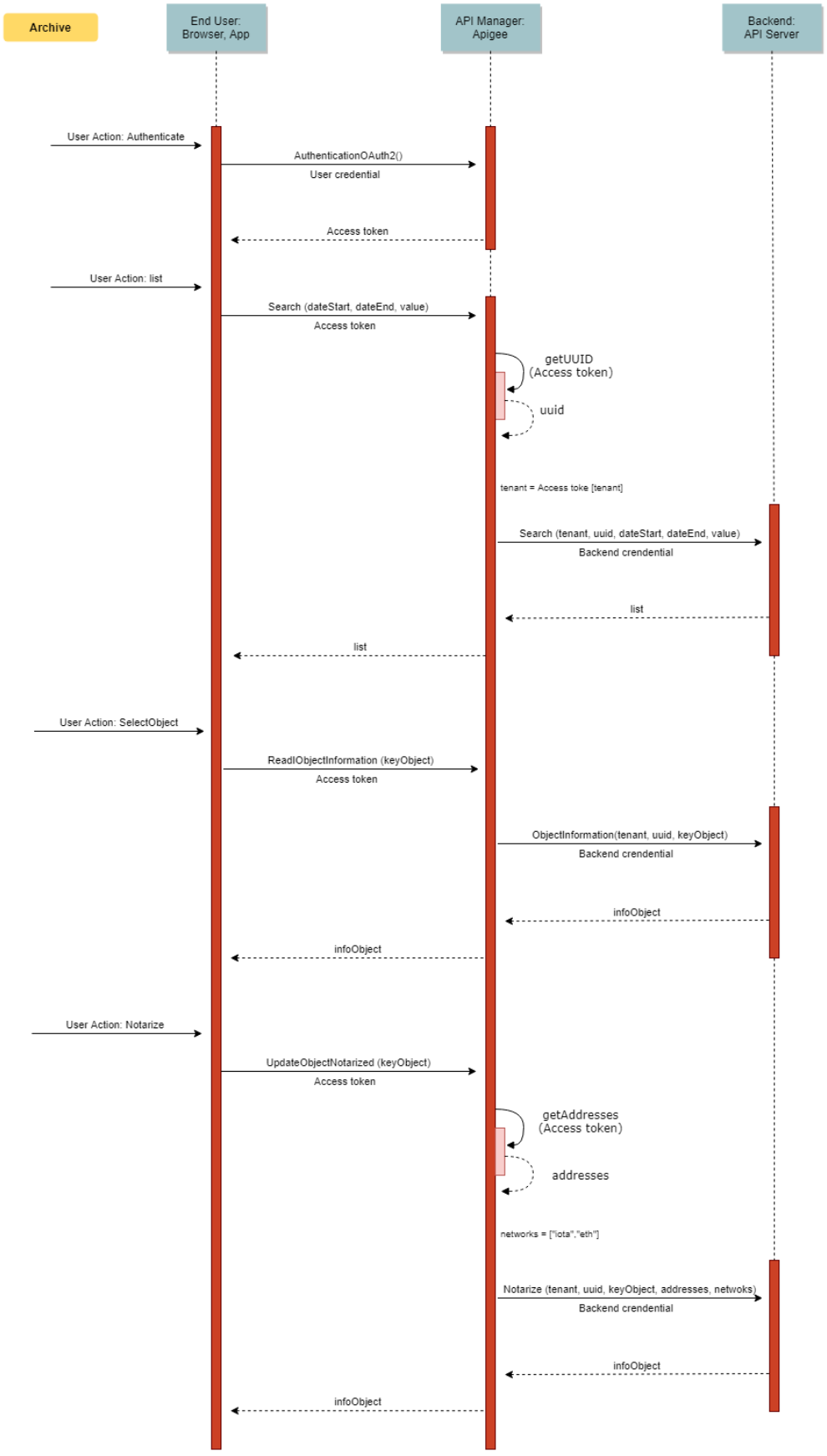


Figure 20 Diagram. Object List, Object Information and Notarize

Graphical User Interface Design

The GUI focuses mainly on the web application, although it is not the most important part of this project, it still has great weight, since it is the visual part in which the user will be able to interact with different technologies.

Therefore, the mockups have been designed using Adobe XD through some modern web templates and components. The objective is to have a visual idea of the functions to establish pattern or guides during the implementation to achieve the goals. Therefore, once the implementation is complete, the application may be different from the illustrations presented in this.

Figure 21 shows a list of user files stored on our system, filtered by a time interval.

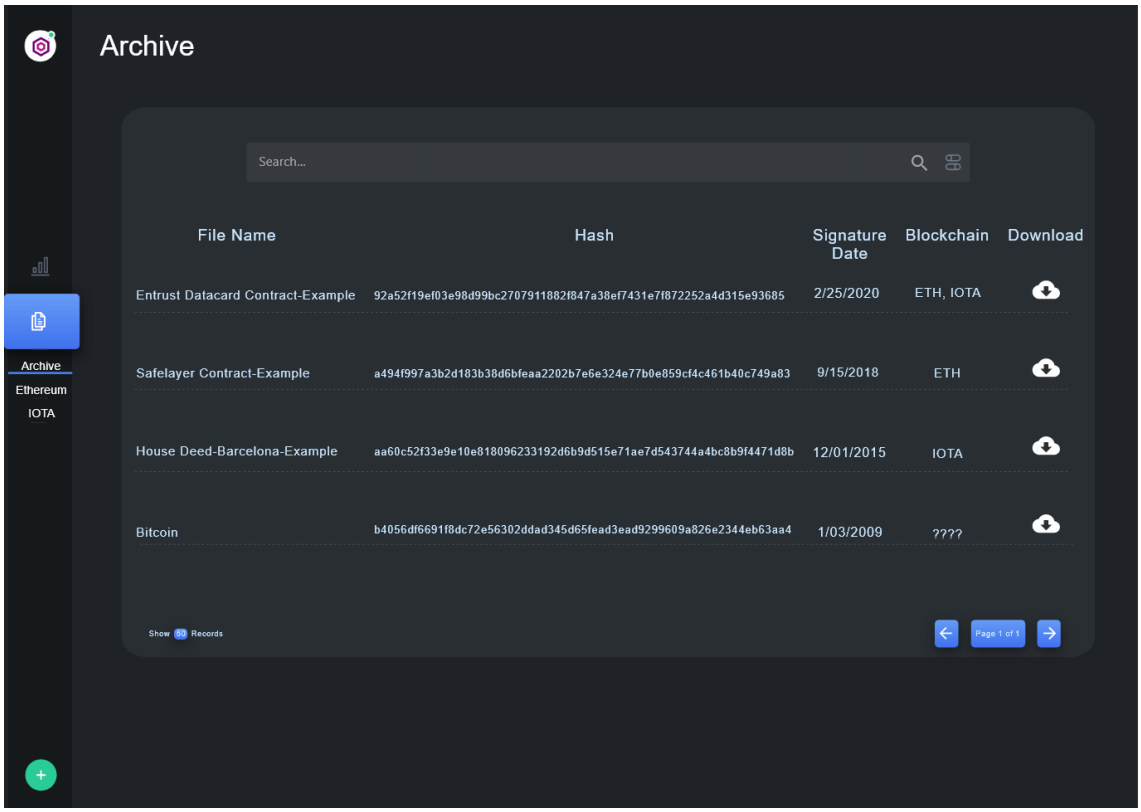


Figure 21 Mockups. List archives

When the user selects a file from the file list, he goes to a new screen to show more details and depends on the state of the file, exists two cases may be found.

The first state that is not notarized, as seen in Figure 22, in which it shows relevant information of the file and offers the option of notarizing.

Figure 23 shows the second state that is notarized, where the file information is shown in detail and also, details of transactions of DLT in which the file has been notarized.

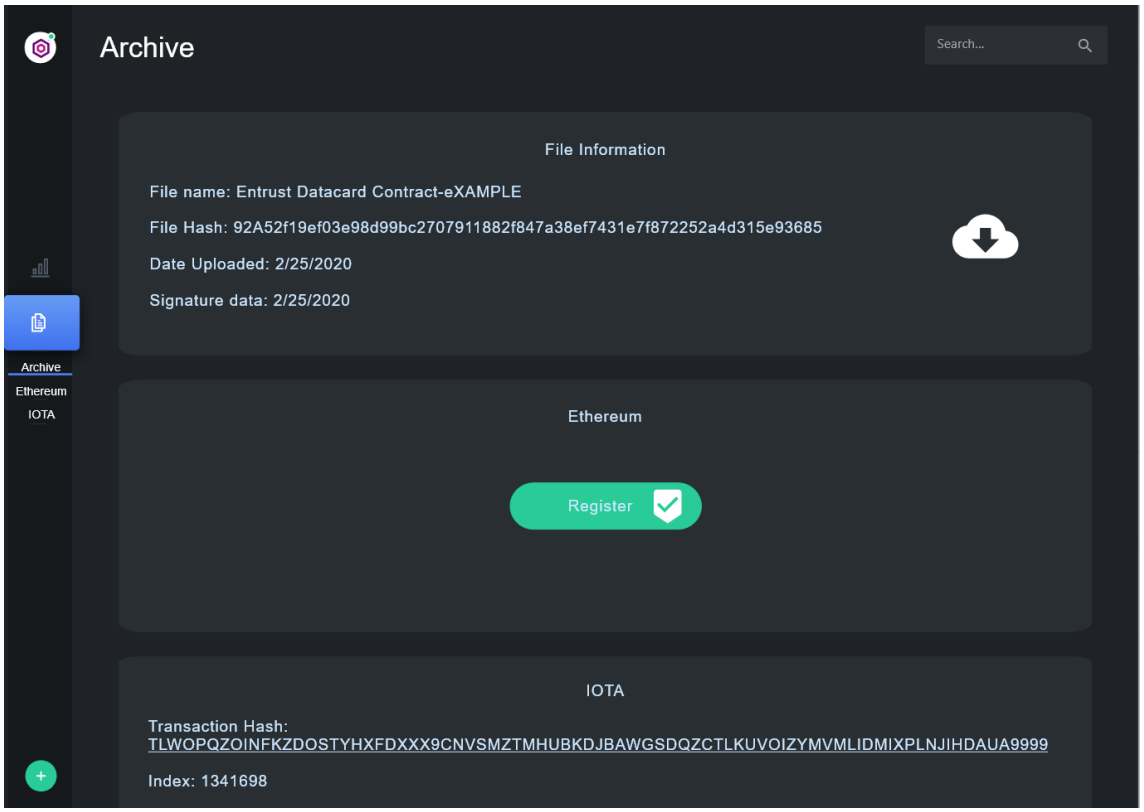


Figure 22 Mockups. More information 1

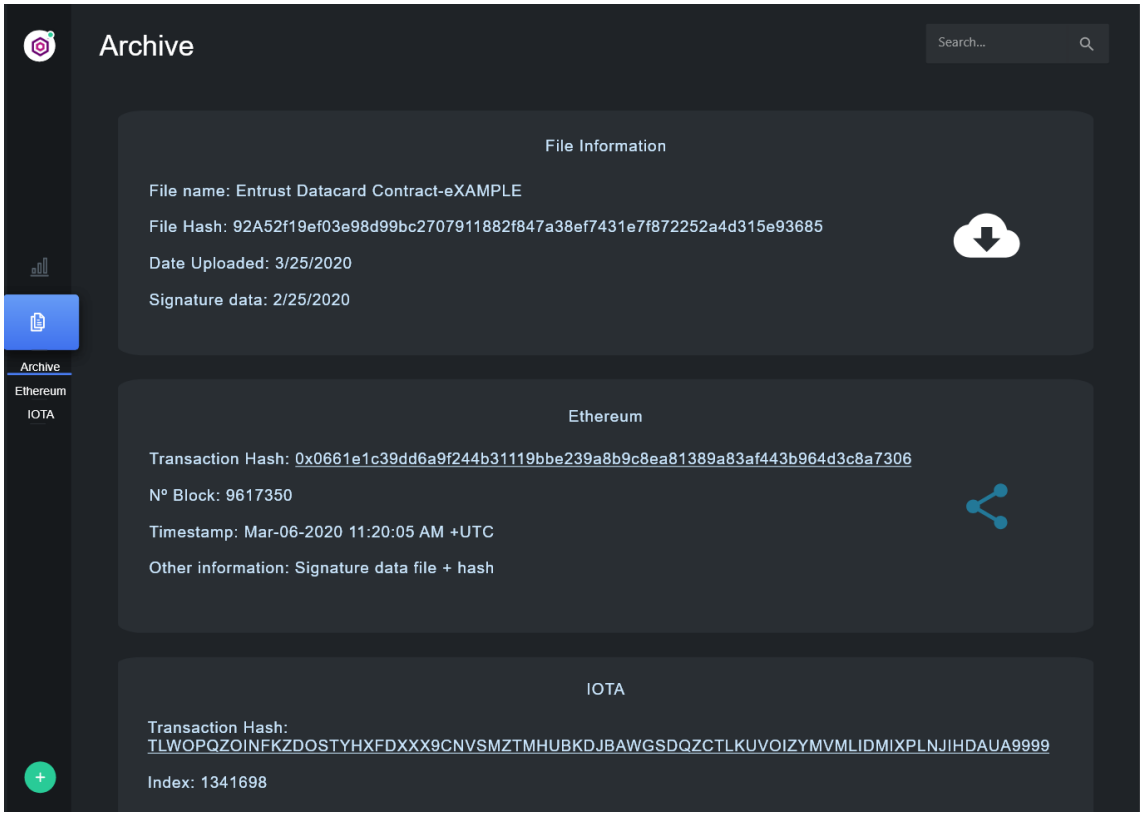


Figure 23 Mockups. More information 2

Figure 24 shows the screen in which the user can upload files to our system by drag & drop or by manual selection.

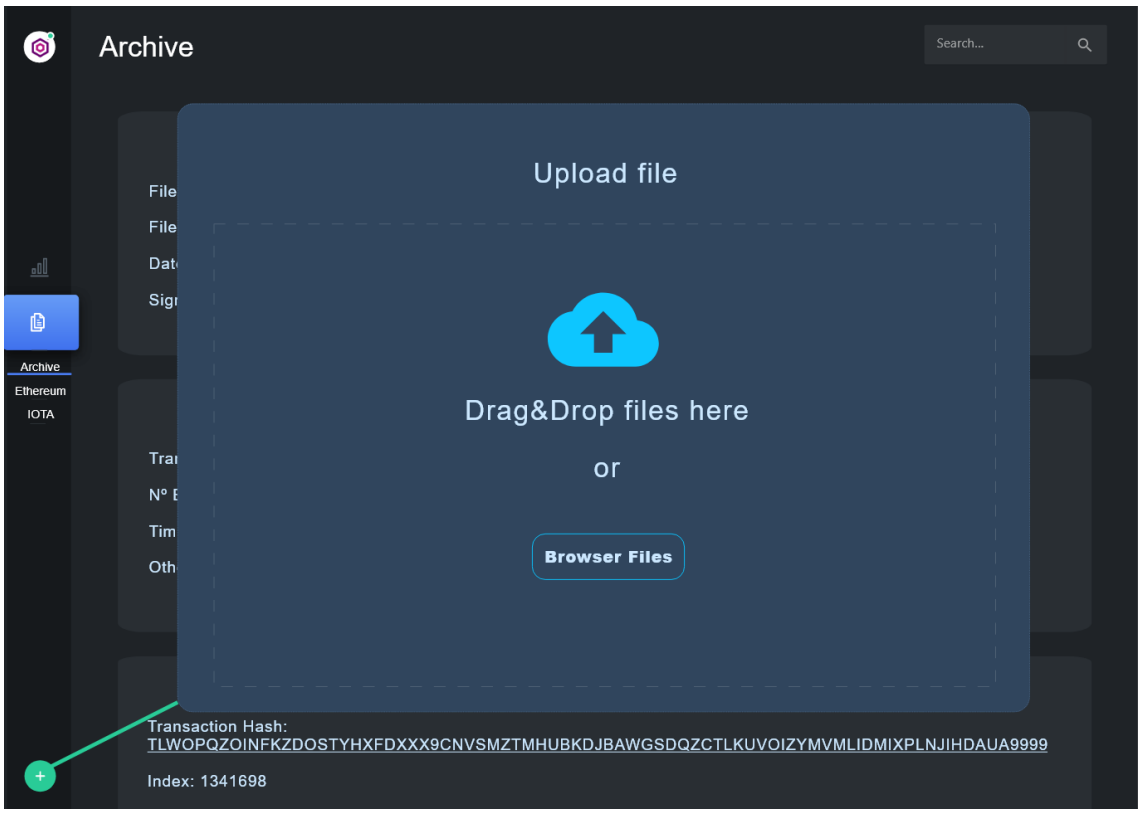


Figure 24 Mockups. Upload

On the other hand, the next mockup in Figure 25 is about functionality that was proposed from the beginning. However, it will not be carried out since it does not provide much use to the system or the purpose of this project. However, we will offer a shortcut confidently to third-party explorers to view all transactions of one account of a user.

Finally, Figure 26 presents the prototype of the stored files' statistics, in addition to other information that could be interesting to the user. We will only be implemented if sufficient time is available.

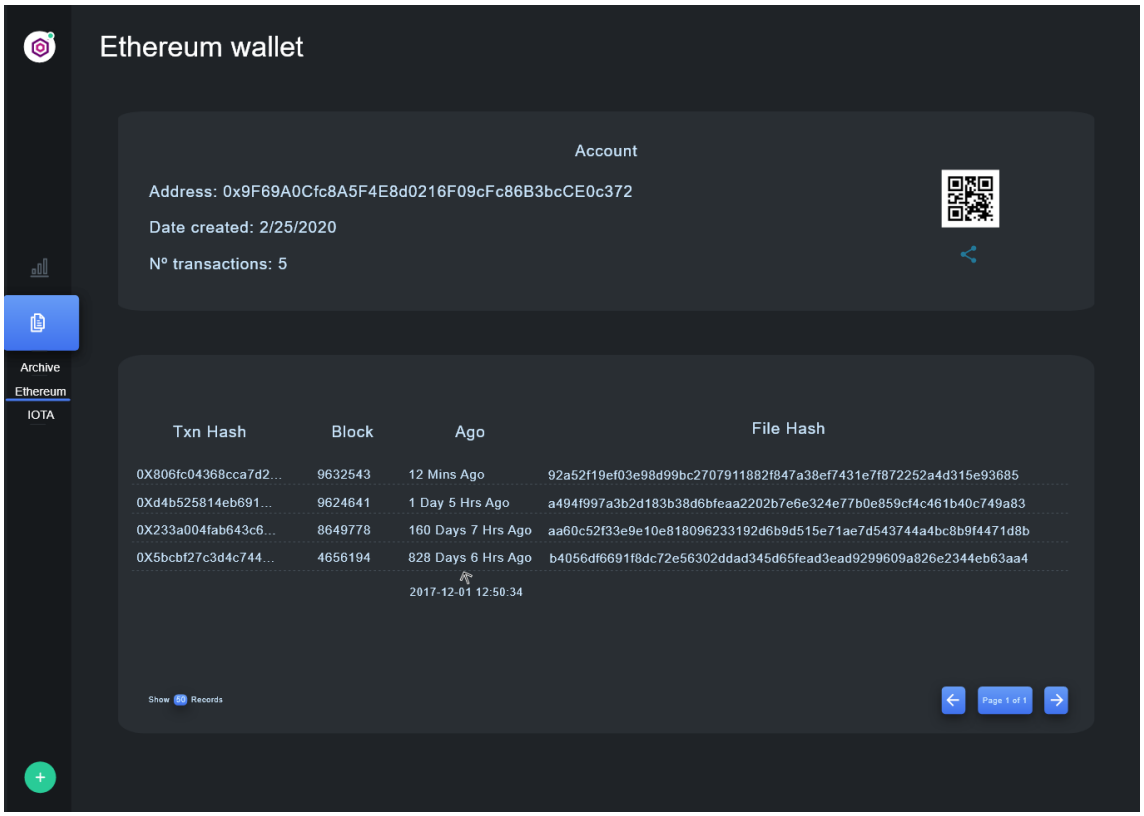


Figure 25 Mockups. Wallets

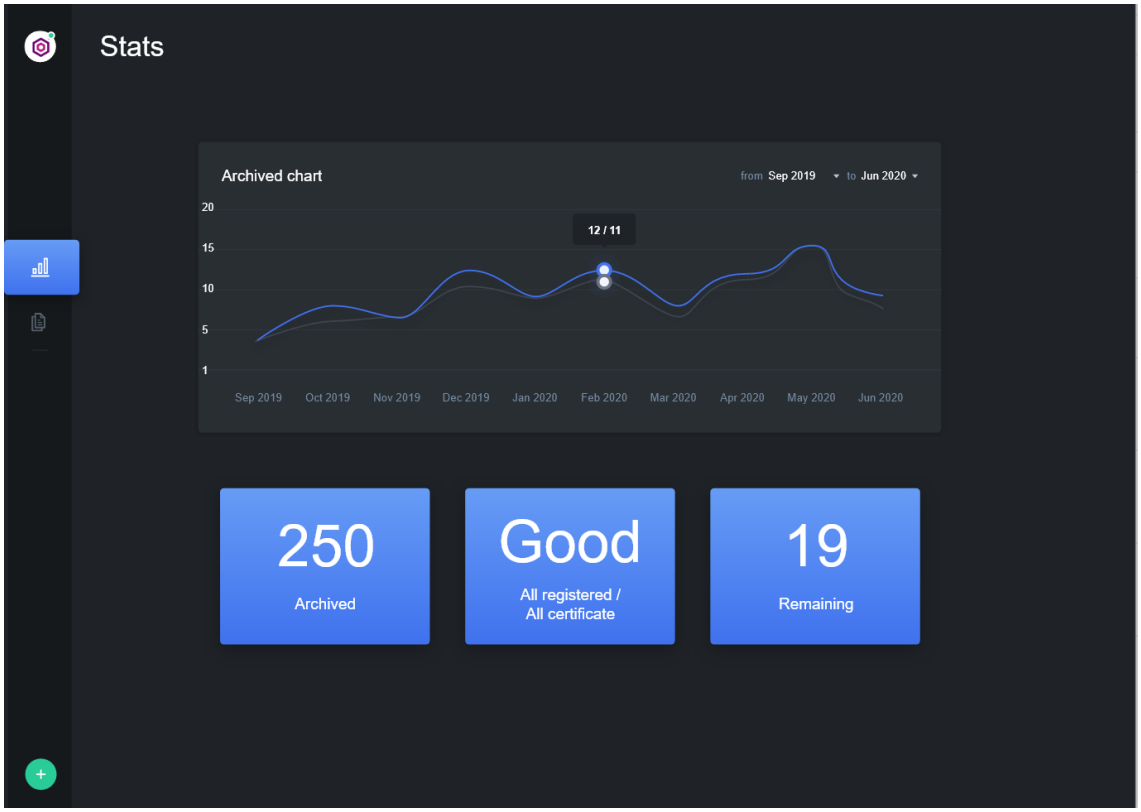


Figure 26 Mockups. Stats

Implementation

In this section, we are going to explain how to implement the backend services and how to build the application that consumes the system services.

Backend

In the first place, we will explain how the services have been implemented of the system, which corresponds to all services offered in the API Server with Tomcat. Moreover, for the project Java, the tool Maven for repository management has been used.

The project has the structure of Figure 27, where the main folder "com.txaas.ws.rest.apis" is located the RESTful API services that we want to implement, followed by some directories as Ethereum, IOTA and S3 with respective implementations of the technologies that we will explain bellow.

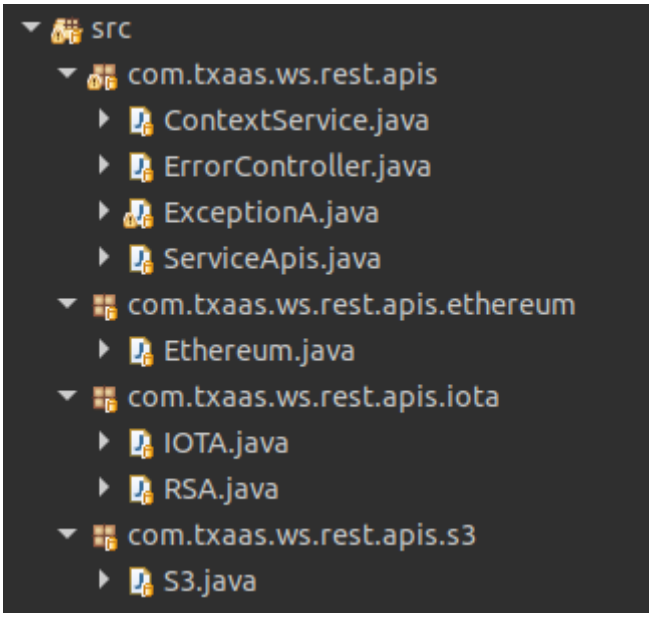


Figure 27 Java Project

Amazon S3 Bucket

The structure of storage is of the utmost importance since good organization depends on it to facilitate the subsequent search and the access to the resources. Since we do not want to the use unnecessary resources, we have chosen the simplest and most effective solution to achieve our objective, use a single Bucket to store all the files of all the users of different tenants.

On one side, these objects will use the metadata to store the file information, the object's own key for the indexing and el system of the filter by the prefix of S3 for search.

On the other hand, it could be interesting to apply an adaptive policy according to the object's longevity and apply a type of storage to save cost. But for the primary purpose of this project, we will use the standard class that has an acceptable cost and will leave the configuration of an adaptable plan for future work.

Key Object

The following structure allows us to separate content from different tenants and users, as well as to allow us to obtain a list of objects temporarily grouped by year and month using the filter by the prefix. A unique key to an object in S3 has the following format:

```
Tenant/UUID/Year/Month/Object
```

Identifier	**Description**
Tenant	Unique identifier that represents an entity.
UUID	Unique identifier of a user.
Year	Identifier of one year that the file has been uploaded.
Month	Identifier of the one month that the file was has been uploaded.
Object	Identifier of a coded object in a way to allow filtering and realizing the search.

The following format of the key-value is used for the coding of **Object** to identify the relevant information in a file without extracting the metadata. Where a key is made up of the initial characters with a double underscore (__), followed by the identifier of the key and a hyphen (-) is used to indicate the end of the identifier and the beginning of the value (<v>). E.g. "*__Filename-"*

```
__filename-<v>__hash-<v>__sig-<v>__eth-<v>__miota-<v>.<filetype>
```

Identifier	Description
__filename-<v>	The original filename
__hash-<v>	The first 9 characters of the *hash*
__sig-<v>	Representative value that identifies whether the file has a signature. The value can be "yes" or "no" to represent their respective meanings.
__eth-<v>	Representative value to identify if the file is notarized in the Ethereum network. The value can be "yes" or "no" to represent their respective meanings.
__miota-<v>	Representative value to identify if the file is notarized in the IOTA network. The value can be "yes" or "no" to represent their respective meanings.
.<filetype>	The format of the extension of file, "pdf" for PDF documents and "gif" for GIF images.

Metadata Object

The metadata used to store the information together with the files are the following.

Key	Description
x-amz-meta-filename	The original filename.
x-amz-meta-filetype	The format of the file.
x-amz-meta-hash_alg	The algorithm used to generate the hash of the file. Currently, only "sha512" is allowed
x-amz-meta-hash_value	The value of the hash
x-amz-meta-eth	The hash of the transaction of the Ethereum's network. The value can be the hash of the transaction or “none”.
x-amz-meta-miota	The hash of the transaction of the IOTA’s network. The value can be the hash of the transaction or “none”.
x-amz-meta-signature	Indicates the timestamp of the signature if it exists or “none” to indicate the opposite. Currently "0" indicates that exists a signature.

As we can see, this system's main feature is separate detailed information (metadata) and basic information (key) that allows an acceptable search. Therefore, we achieve a design capable of fulfilling the important functionalities of this storage technology (saving and obtaining a file) and overcoming the lack of search without resorting to other services.

On the other hand, it is essential from the beginning to design a system adaptable to future technologies. Therefore, we have implemented the storage system so that it has the ability to add new networks or revoke existing ones without restructuring the entire system or causing incompatibility between versions. Therefore, we have two possible scenarios:

1. To add a new DLT network. We would have to create a new identifier for the network and for all new files, and an *x-amz-meta-X* key would be added in the metadata, where "X" would be the identifier of the new network. On the other hand, the key of the *Object* must match the metadata values. Therefore, the new network and its respective value must be included in the coding of Object, before the *filetype*. E.g. If we want to add Bitcoin, the identifier would be *btc*, the metadata would contain the field *x-amz-meta-btc*, and the key of the *Object* would have the following format:

```
__filename-<v>__hash-<v>__sig-<v>__eth-<v>__miota-<v>__btc-<v>.<filetype>
```

   The new coding would be backward compatible with the previous codings. New files would use the new coding, and old files will be able to continue working with the old coding without a problem.
2. Revoke an existing DLT network. It would have the same effect as scenario 1, but inversely. A direct explanation using the previous example to revoke IOTA, which means that in the metadata it would stop adding the key *x-amz-meta-miota* for the new files, and the coding of the *Object* would have the following format:

```
__filename-<v>__hash-<v>__sig-<v>__eth-<v>__btc-<v>.<filetype>
```

Through this design, we acquire that different codings of the object are compatible with each other, and the information displayed is implicit with the supported technologies.

Once you know the structure of the storage, the implementations with AWS SDK JAVA will be explained in a simple way. The main functions are to get a list of dates, a list of keys object, and information about an object, generate a pre-signed URL to download/upload, copy an object to update the metadata and update signed.

1. **List of dates**

```
ArrayList<String> listDates (String context)
```

This function returns a list of dates (Year/Month) that contains at least one object. As a parameter, it requires the *context* (*Tenant/UUID*).

2. **List of keys object**

```
ArrayList<String> listKeyObject (String context, String date)
```

This function returns a list of keys that begins with a prefix (Tenant/UUID/Year/Month) that corresponds to the concatenation of the *context* (Tenant/UUID) and *date* (Year/Month).

3. **Information about an object**

```
Map <String,String> metadataUserObject (String key)
```

This function returns all the information stored in the user-metadata of an object. As a parameter, it requires the *key* (Tenant/UUID/Year/Month/Object).

4. **Generate a pre-signed URL to download**

```
String generatePresignedUrlDownload (String key)
```

This function returns a URL to access a file stored in S3 that has a temporary expiration according to the values of configuration. As a parameter, it requires the *key* of an object (Tenant/UUID/Year/Month/Object).

5. **Generate a pre-signed URL to upload**

```
String generatePresignedUrlUpload (String key, JSONObject body)
```

This function returns a URL to upload a file to S3 that has a temporary expiration according to the values of configurational. As a parameter, it requires the *key* (Tenant/UUID/Year/Month/Object) and *body* (metadata information).

6. **Copy an object to update the metadata**

```
String copyWithMetadata (String sourceKey, Map<String,String> metaUser)
```

This function copies an object in S3 to another located with new metadata and returns the key of the new object. As a parameter, it requires the *sourceKey* (Tenant/UUID/Year/Month/Object) and *metaUser* (the new key will be generated with this information).

7. **Update Signed**

```
boolean updateSigned (String key)
```

This function downloads the document PDF from S3 to check whether it has a signature, in case affirmative, the object metadata is updated with the corresponding information. As a parameter, it requires the key (Tenant/UUID/Year/Month/Object). This function uses the iText5 to search for possible signatures.

Ethereum

Ethereum's transaction system is simple, we only require that our system has a single wallet that has the ability to send simultaneous transactions to multiple user's addresses.

On the one hand, a transaction is made up of multiple fields, which the following are essential for the design of a notarial transaction:

- TxHash, is the hash that uniquely identifies a transaction within the Ethereum's network. This value will be recorded in the metadata of the notarized file in the *x-amz-meta-eth* field. On the other hand, detailed information on the transaction can be obtained from any network explorer with this value. This hash is generated at the time of sending the transaction.
- Timestamp, is the field that indicates when the txHash was registered in a block. Therefore, it is the timestamp we use to verify that a file exists at any given time.
- From, this field identifies the address of the issuer and we use it to verify that the transaction has been carried out through our application and to ensure that the file is stored in our system. This value can be extracted from the private key of the wallet, which our system will have direct access to it.

- To, this field identifies the recipient's address and is the parameter we use to send a transaction to the user. The client must provide this value through the addresses that will have stored in OpenDJ.
- Input Data, is a field of varied utility, but in our case, we can interpret it as the field of a message we use to record the hash of a file. This field, like the hash of a transaction or address, always starts with “0x”. Therefore, the message will start after this value. The content must be provided by the client that corresponds to the field *x-amz-meta-hash_value* of the metadata of a stored file.

On the other hand, there are other essential fields to finish a transaction successfully:

- Nonce, is the number of transactions that sent from a given address or wallet. This value is the decisive factor to obtain concurrent and simultaneous transaction shipments. Therefore, it must be obtained through the network to account for pending transactions and avoid cancellations of transactions or possible double expenses.
- Value, in our case, the value of this field will always be 0 since we are not interested in transferring ether.
- Gas Used, this value is not configured and represents the actual expense of the gas used by a transaction. A normal transaction always spends a defined value of 21000, but in our case, we also want to attach a hash with certain bytes. Therefore, the cost also increases, but it will never be higher than the Gas Limit which is configured previously.
- Gas Limit, is the maximum number of gas units that we are willing to pay for a transaction. If the limit is exceeded, the transaction will not end, and the used gas will be lost, instead, if a transaction ends successfully, but not all the gas that we were willing to pay has been used, it will return the unused gas to us. Based on our calculation for a hash with 64 bytes of an algorithm sha512, we will at most spend 25352[32]. Therefore, we will allocate a total of 25500 to apply a higher security margin.
- Gas Price, is the value that we are willing to pay for a unit of gas. The higher this value, the more cost the transaction will be, but the faster it will be confirmed. It varies according to the overload of the network.
- Transaction fee, in ETH that we pay at the end for a transaction will depend directly on the Gas Price and the Gas Used.

[32] Gas Limit = $G_{transaction}$ + ($G_{txdatanonzero}$ x $Bytes_{inputData}$) = 21000 + (68 x 64) [40]

Communication with the Ethereum's network is realized through the nodes. Currently, launching our own full node is not profitable, since it requires 180GB of storage and a long-time synchronization and processing. On the other hand, launching our own light node would not be very useful either, since our objective is not to consult or send cryptocurrencies, but to archive and obtain information on those old transactions. Therefore, the best option is to use private third-party nodes to communicate with the Ethereum's network. That's why we have used Infura[33] as a service platform that provides connections to Ethereum using the most popular tools of development of APIs around JSON RPC.

Finally, we will expose a simplified explanation about the main functions implemented with the Web3j.

1. **Create Address**

```
String createAddress ()
```

This function returns a new address for a wallet of Ethereum. It consists of obtaining the address from a public key, which in turn is generated from a private key of Elliptic Curve Cryptography.

2. **Send Transaction**

```
String sendTransaction (String addressTo, String message)
```

This function returns the hash of the transaction, which is sent to an Ethereum's node. According to the values of the system configuration and the values of parameter values (addressTo and message), the function configures all the necessary fields for a transaction to send the signed transaction with the private key.

3. **Find Transaction**

```
JSONObject findTransaction (String txHash)
```

This function returns the basic information of an Ethereum's transaction, which is indicated in the txHash parameter. During the obtainment of the transaction's information, it performs the pertinent checks to verify that it is a legitimate transaction in our system.

[33] Infura is a provider that offers Ethereum nodes

IOTA

In this case, the concept of notarizing a file is the same, which is to register a hash in a transaction. However, this network is full of spam by offering commission-free transactions. This behavior is not bad for the operation of the network, even it helps in a certain way in the confirmations, but it is inconvenient for our proposed system that every user has an address since everyone could sent unwanted transactions to an address of a user.

On the other hand, in this network, we cannot take advantage of the sender's address to verify if a transaction from our application is legitimate, because we cannot reuse a wallet that the private key is compromised after sending the first transaction and it ceases to be safe. The following design a solution for the mentioned problems.

A transaction in IOTA has several fields, but we are interested in the following for our solution:

- TxHash, is the hash that uniquely identifies a transaction within the IOTA's network. This value will be recorded in the metadata of the notarized file in the field *x-amz-meta-miota*. On the other hand, with this value, detailed information of the transaction can be obtained from each permanodes or the official explorer Tangle.
- Address, is the receiver's address. In our case, it corresponds to the address of the user that is stored in the OpenDJ.
- Timestamp, in this case, it indicates the time in which the transaction was sent, but we also have another timestamp that shows the time in which the transaction was confirmed. In this design, we will use the first timestamp to verify that a file existed at a given time, and the second only to check the confirmation of the same transaction.
- Tag, this field allows the network to classify transactions and allow a more efficient search, for this, we will use it as a first identifier of our application.
- Message, contains the information that we want to store on the network (equivalent to the field *Input Data* of Ethereum), on the other hand, the content will be an object JSON, which will contain the hash of a file to be notarized and a second parameter that we will call it *certificate*. The latter is used to verify the authenticity of the transaction and avoid spam.

The *Tag* would be the substitute of the field "*To"* of a transaction in Ethereum but without the property of uniquely identifying the issuer of a transaction or guaranteeing that the

transaction was carried out by our system because any individual could place the same tag as us and impersonate our identity. However, it serves as a first filter to dismiss the validity of transactions Spams.

To guarantee the validity of the transaction, we have the parameter *certificate* of the field *Message* that consists of a cryptographically encrypted value with a private key of type RSA of the content to be notarized, in this case, it is the hash of a file. With the use of this method, we can guarantee that a transaction is valid if we verify that at the moment when decrypting the content of the *certificate* with the public key, it returns the same value as the parameter hash of the field *Message*, therefore, the content has not been modified, and We can assure that It has been sent by our system and not by a third party.

On the other hand, there are other essential fields for a transaction:

- Level, is the level of security when generating a private key for a new address. In our case, it is not important, since our purpose is not to send cryptocurrencies but to send a zero-value transaction.
- Depth, is an argument, which defines how many milestones in the past that the node starts to a selection algorithm. That the optimal configuration will depend on the node and the network, in our case, it is set to of 1.
- Minimum Weight Magnitude, is the specification of the level of proof of work for a transaction. Currently, the network requires an MVM of 14.

Therefore, the design of transactions of IOTA is made up of three layers or filters to check the validity of a transaction:

- First, the field *Address* checks if the received transaction belongs to the user.
- Second, the field *Tag* allows us to verify if the transaction is sent by our system and filter the results and avoid spam.
- Third, the parameter *certificate* within the field *Message* allows us to guarantee that the content has not been modified, and the transaction is sent by our system.

It should be noted that this design was initially intended to offer the functionality of listing all the transactions of a user in our application, without the explorer of third-party. But with a subsequent redesign of the client's application, we realized that this level of security was not so necessary.

The communication with IOTA is realized through a direct connection to the public nodes of The Tangle. Although the cost of launching our own node is low and we could obtain

great benefits in confirmations, we have decided that the ideal option is to use public nodes for this first conceptual project.

Before explaining the main functions of IOTA, I will introduce the signing and verification functions with RSA public-key cryptography. To generate the public and private keys, we use the OpenSSL, which has an open source tool in Linux.

With this first command, we generate a private key 512-bit RSA encoded with PEM.

```
openssl genrsa -out private_key.pem 512
```

With the following command, we export the previous private key to a format PKCS8 / DER to make it compatible with Java.

```
openssl pkcs8 -topk8 -inform PEM -outform DER -in private_key.pem
-out private_key.der -nocrypt
```

With this last command, we generate the public key in format DER, to make it is also compatible with Java.

```
openssl rsa -in private_key.pem -pubout -outform DER -out public_key.der
```

Of three generated files, we will keep the files public_key.der and private_key.der in our system, which will be the public and private keys, respectively.

The following two functions use the RSA keys generated by the library Java Security.

a. Sign

```
String sign (String message)
```

This function uses the previously generated private key to encrypt the content of the parameter *message* and returns the result.

b. Verify

```
Boolean verify (String message, String certificate)
```

This function returns true or false, after checking if decrypting *certificate* with the public key gets the same values as the *message*.

Finally, a simplified explanation of the main functions implemented with the library iota-ledger-java will be exposed.

1. **Create Address**

```
String createAddress ()
```

This function returns an address of IOTA. This address is generated from a signature Winternitz, Hash-based cryptography.

2. **Send Transaction**

```
String sendTransaction (String addressTo, String message)
```

This function returns the hash of the transaction sent to a node of IOTA in 4 steps. First, generate a random wallet to send the transaction. Second, it generates the value *certificate* with the encryption of the content of the *message* using the private key RSA of our system (a. Sign) and including the two values (*certificate* and *message*) in the field *Message*. Third, configure all the necessary fields of a transaction according to the configuration values. Fourth and last, it sends the transaction to the node of IOTA at address addressTo.

3. **Find Transaction**

```
JSONObject findTransaction (String txHash)
```

This function returns the basic information of a transaction of IOTA indicated in the parameter *txHash*. Once the information of the transaction is obtained, the integrity and authenticity verification are performed through the field Tag and to decrypt the content of the *certificate* of the field Message (b. Verify) through the use of the public key of our system.

API RESTful

In this section, we will explain how the APIs RESTful services have been implemented using the functions of the different technologies explained above.

As mentioned in the specifications section, the Tenant and UUID values are extremely important for the security of all the APIs in this section. Therefore, Apigee as the only reliable source must guarantee the correct return of these values according to the user since all users of different tenants share a single Bucket in our storage system. On the other hand, we have configured the SSL/TLS certificates of the server Tomcat in order to be sure that only Apigee has access to the APIs, and we will prevent unauthorized access.

The implementation of the APIs has used the standards of the library JAX-RS that simplify the creation of RESTful Web Services, which we have mainly used methods like GET, POST and PUT, as well as the formats accepted by the request and response is JSON.

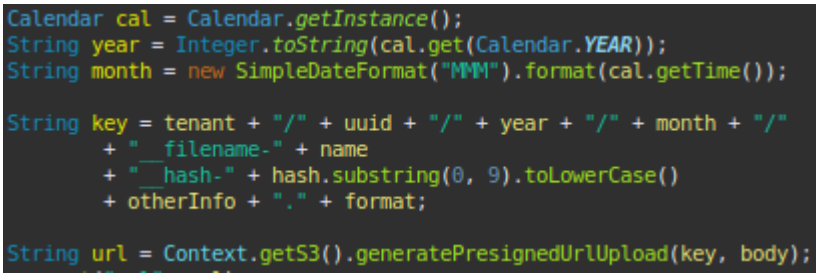
For more detail and examples of the implemented APIs, check the documentation generated by Postman, here[34].

Method	POST
Path	/addresses/new
Params / Body	-
Description	Generate and return an address for each configured DLT / Blockchain

Implementation	`String addressETH = Context.getETH().createAddress();`
	`res.put("eth", addressETH);`
	
	`String addressIOTA = Context.getIOTA().createAddress();`
	`res.put("miota", addressIOTA);`

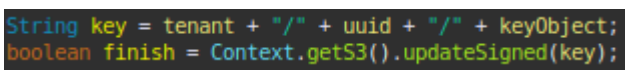
Table 1 API New Addresses

[34] https://documenter.getpostman.com/view/9340760/Szmb5eqW

Method	POST
Path	/objects/url-presigned/upload
Params / Body	tenant | uuid | filename | filetype | hash_alg | hash_value
Description	Generate and return a pre-signed URL to upload based on the information provided
Implementation	

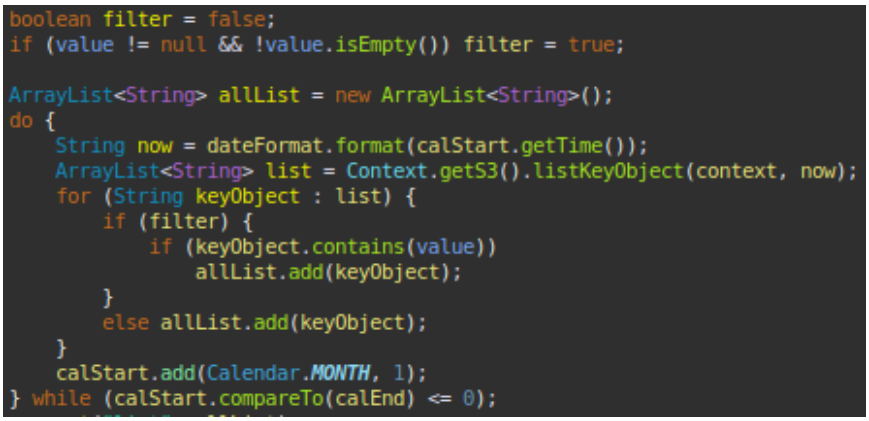
```
Calendar cal = Calendar.getInstance();
String year = Integer.toString(cal.get(Calendar.YEAR));
String month = new SimpleDateFormat("MMM").format(cal.getTime());

String key = tenant + "/" + uuid + "/" + year + "/" + month + "/"
        + "__filename-" + name
        + "__hash-" + hash.substring(0, 9).toLowerCase()
        + otherInfo + "." + format;

String url = Context.getS3().generatePresignedUrlUpload(key, body);
```

Table 2 API Pre-Signed URL to Upload

Method	PUT
Path	/objects/updateSigned
Params / Body	tenant | uuid | keyObject
Description	Check if the object *keyObject* is a PDF, that contains a signature. If the answer is affirmative, update the information associated with the object
Implementation	

```
String key = tenant + "/" + uuid + "/" + keyObject;
boolean finish = Context.getS3().updateSigned(key);
```

Table 3 API Update Signed

Method	GET
Path	/objects/search
Params / Body	tenant | uuid | dateStart | dateEnd | value
Description	Returns a list of objects in the time interval between *dateStart* and *dateEnd*, optionally *value* to filter the search. Returns a list of dates if the parameters dateStart, dateEnd and Value are null at the same time.
Implementation	

```
boolean filter = false;
if (value != null && !value.isEmpty()) filter = true;

ArrayList<String> allList = new ArrayList<String>();
do {
    String now = dateFormat.format(calStart.getTime());
    ArrayList<String> list = Context.getS3().listKeyObject(context, now);
    for (String keyObject : list) {
        if (filter) {
            if (keyObject.contains(value))
                allList.add(keyObject);
        }
        else allList.add(keyObject);
    }
    calStart.add(Calendar.MONTH, 1);
} while (calStart.compareTo(calEnd) <= 0);
```

Table 4 API Search

Method	GET
Path	/objects/information
Params / Body	tenant | uuid | keyObject
Description	Returns all the information related to the object, includes the information of transactions of the network DLT/Blockchain and a pre-signed URL to download the file
Implementation	

```
//Get Metadata
String key = tenant + "/" + uuid + "/" + keyObject;
Map <String,String> metadata = Context.getS3().metadataUserObject(key);
JSONObject object = new JSONObject();
for (Map.Entry<String, String> entry : metadata.entrySet()) {
    object.put(entry.getKey(), entry.getValue());
}
String url = Context.getS3().generatePresignedUrlDownload(key);
object.put("url", url);

res.put("object", object);

//Get Info Tx IOTA and ETH
String txETH = metadata.get("eth");
String txIOTA = metadata.get("miota");

JSONObject eth = new JSONObject();
JSONObject iota = new JSONObject();
if (!txETH.equals("none"))
    eth = Context.getETH().findTransaction(txETH);

if (!txIOTA.equals("none"))
    iota = Context.getIOTA().findTransaction(txIOTA);
```

Table 5 API Information Object

Method	PUT
Path	/objects/notarize
Params / Body	tenant | uuid | keyObject | addresses | networks
Description	Notarizes the object *keyObject* in the DLT/Blockchain specified in *networks* and at the addresses indicated in *addresses* and returns the updated information of the object
Implementation	

```
//Get hash512
String key = tenant + "/" + uuid + "/" + keyObject;
Map <String,String> metaUser = Context.getS3().metadataUserObject(key);

String hash512 = metaUser.get("hash_value");

//notarize
for (int i = 0; i < networks.length(); ++i) {
    String net = networks.getString(i);
    String address = addresses.getString(net);
    if (metaUser.get(net).equals("none")) {
        switch (net) {
        case "eth":
            String txETH = Context.getETH().sendTransaction(address, hash512);
            metaUser.replace(net, txETH);
            modified = true;
            break;
        case "miota":
            String txIOTA = Context.getIOTA().sendTransaction(address, hash512);
            metaUser.replace(net, txIOTA);
            modified = true;
            break;
        default:
            break;
        }
    }
}
//copy with new Metadata
String newKey = key;
if(modified) {
    newKey = Context.getS3().copyWithMetadata(key, metaUser);
    if (!newKey.equals(key)) {
        Context.getS3().deleteObject(key);
    }
}
```

Table 6 API Notarize

Frontend

Once the implementation of the backend is finished, we will explain how the App Server has been implemented together with the web application so in order that users can interact with our system.

ReactJS SPA

In this chapter, we will explain the process of implementing the web application without going into great detail. Due to the restructuring and orientation of the subsequent design follow the pattern of Material UI, we will be able to see that there are notable differences compared to mockups, but the essence and original functionalities remain. As the dimension of the application is large to explain it all together, we have divided it into the following pages to simplify it. On the other hand, all the components used in the application come from the Material-UI Framework and use the essential component "Grid" for the relative positions of all the components of the screen in order that ReactJS can manage the content according to the size of the device.

Main Layout

This design is the main and minimum layer of the application that defines how the elements that appear on the screen are composed. In addition to defining the essential elements in order that the application is responsive and adaptable on all platforms, both on Desktop, Tablet and Smartphone.

The screen in Figure 28 is made up of four components:

1. TopBar. Shows the title of the application, logout and action to show/hide the navigation. The latter only on devices such as tablets or smartphones (see Annex). It has been implemented with the main components *AppBar* and *Toolbar*.
2. SideBar. There are the options of listing of date and the button Upload. This panel is hidden when the device is small enough, like the Smartphone or a mini Tablet. It has been implemented with the main components Drawer, List, Collapse and Button.
3. Main. It is not a component of Material-UI, but an HTML element that displays the main contents of the page. In our case, it will be the only element that will change according to the application's different pages.

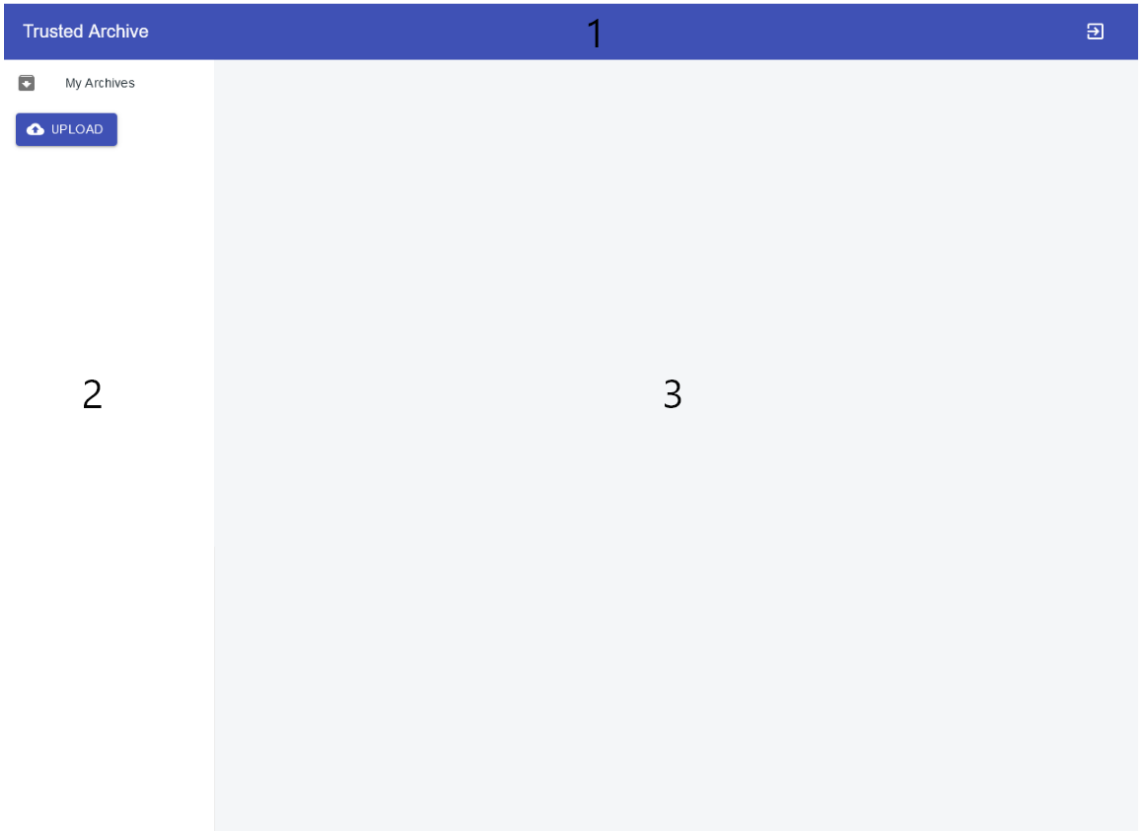


Figure 28 Web App. Main Layout

Upload

The page of Figure 29, we present the process to upload a file and the components that we use in the following order

1. When it interacts with the button Upload, it shows component 2.
2. DropZone. Files can be attached through the use of methods drag and drop or manual selection with explorer of file. This component comes from *material-ui-dropzone* framework.
3. Once Submit is pressed. Start the flow of the diagram in Figure 19 for each present file in the DropZone-Files.
4. Shows the process of upload

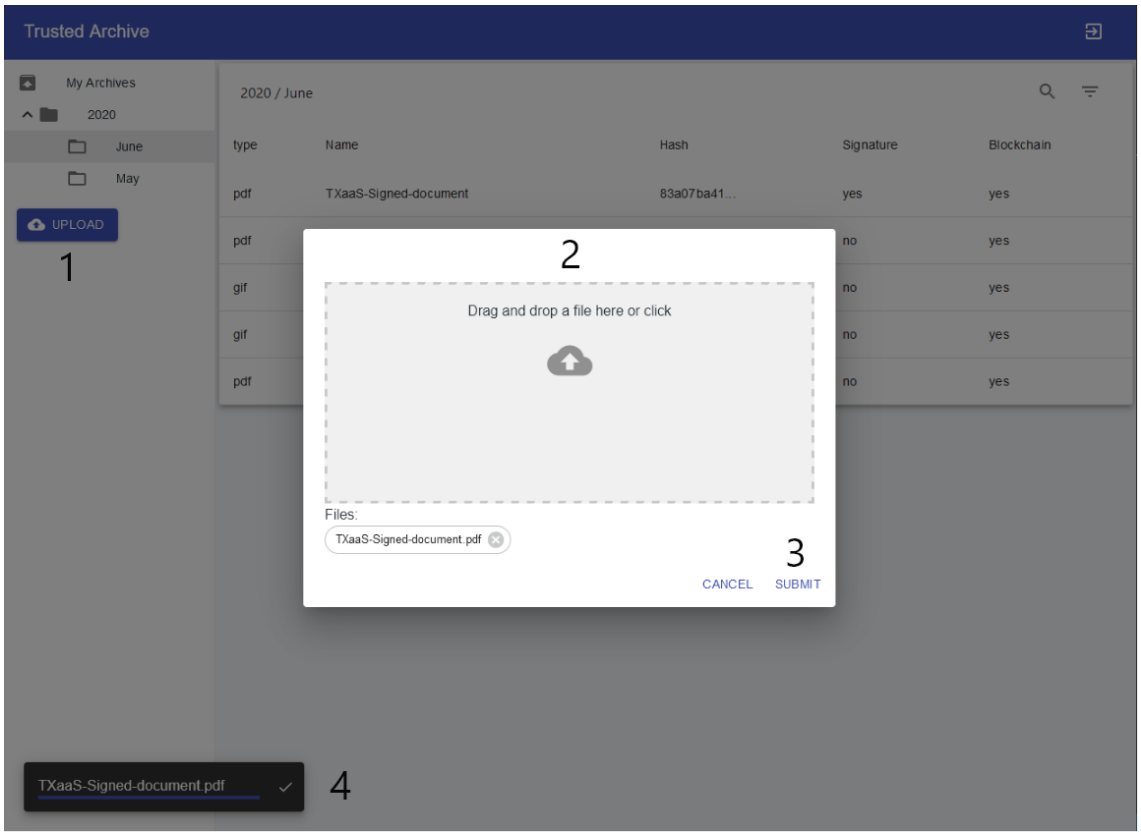


Figure 29 Web App. Upload

File List

Figure 30 shows the screen of the search and listing of the file and uses mainly mui-datatables framework.

1. When interacting with the element *My Archives*, it initiates the request of the *user action list* in the diagram of Figure 20 without parameters. It shows the result by using a self-created component from a multi-level list to indicate the years and months.
2. When it interacts with the element year, it expands to show the months. In case of the month, send the selected date to component 3.
3. When it receives a new date, it initiates the request of the *user action list* of the diagram of Figure 20 with the new date's parameters and shows the results in the table. The content of the table can be filtered by column or textual search using available options.
4. When you select an element from the table, it initiates the request of the user action *informationObject* of the diagram of Figure 20 with the key of the object.

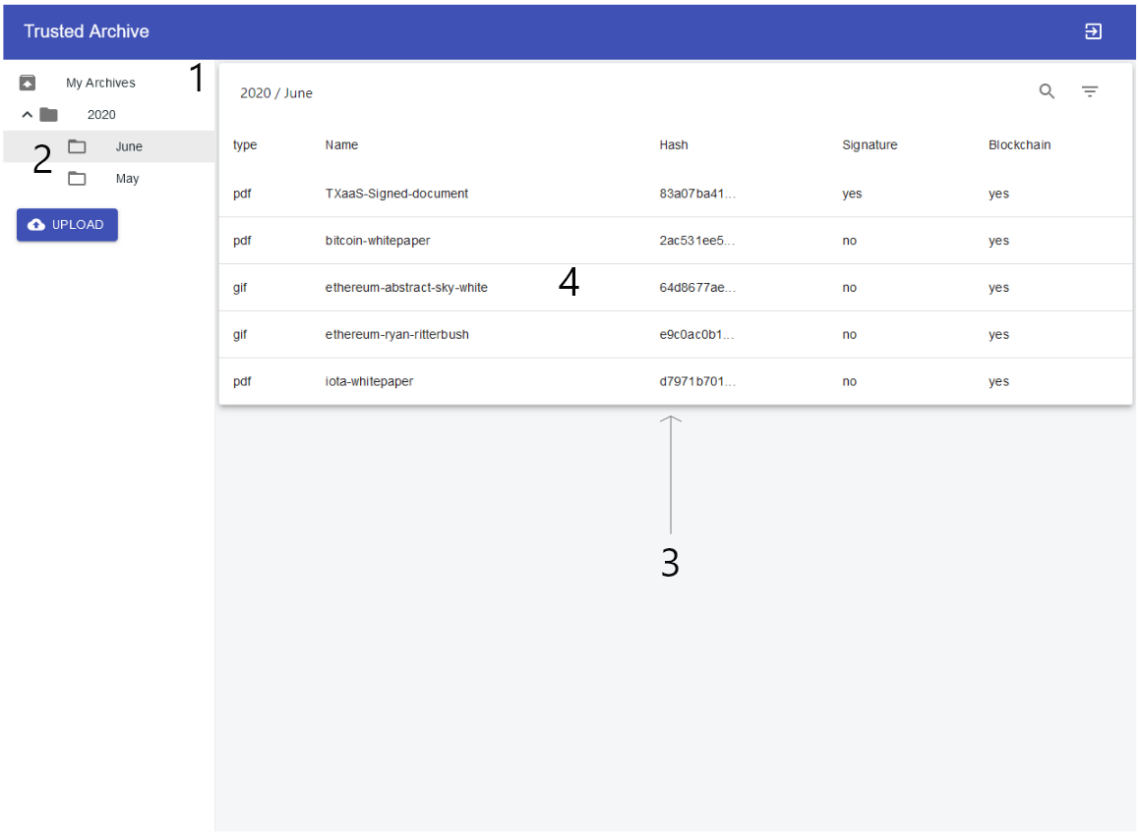


Figure 30 Web App. File List

File Information

This page is the main one of the application, it shows the information of a file and allows to notarize, download or view. We arrive at this screen when we receive information of a file of the request *informationObject*, started on the previous page. The main components used are *Card* and *ExpansionPanel*.

1. The card shows information of file that can be two types. Figure 31 shows a PDF document and Figure 32, a GIF image. On the other hand, in this card, we have the option to download, view or sign. The functionality of the latter is not implemented.
2. The card shows information or allows action to notarize. We can find four states: Figure 31 allows to notarize and initiates the request of the *user action Notarize* Figure 20 if it is not notarized, Figure 33 notarized without confirmation, Figure 34 notarized with partial confirmation, Figure 35 notarized completed.

In the last one, Figure 35, we can see that once the panel has been expanded, we have a button in the respective networks DLT/Blockchain to allow viewing the information in more detail in an external explorer. In the case of this figure, the file is notarized in

Ethereum and IOTA. Check the transactions on the links, Etherscan[35] and Tangle Explorer[36].

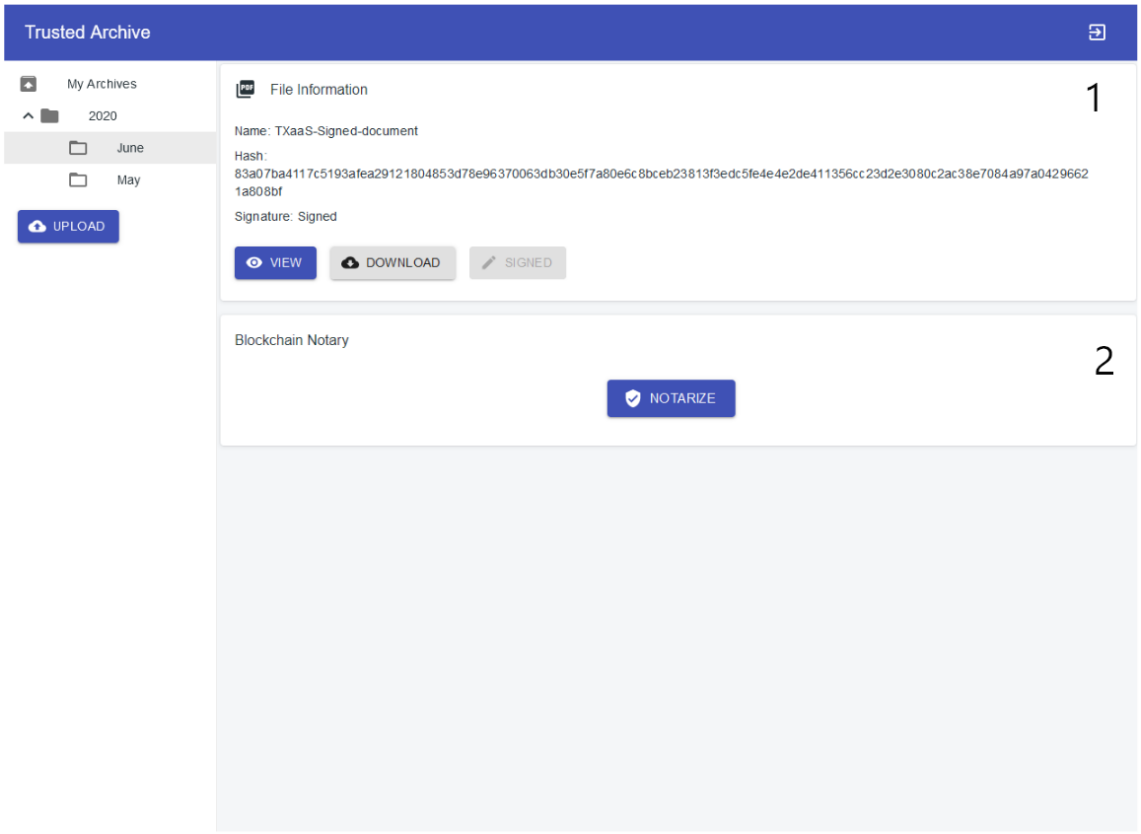


Figure 31 Web App. File Information – No Notarized

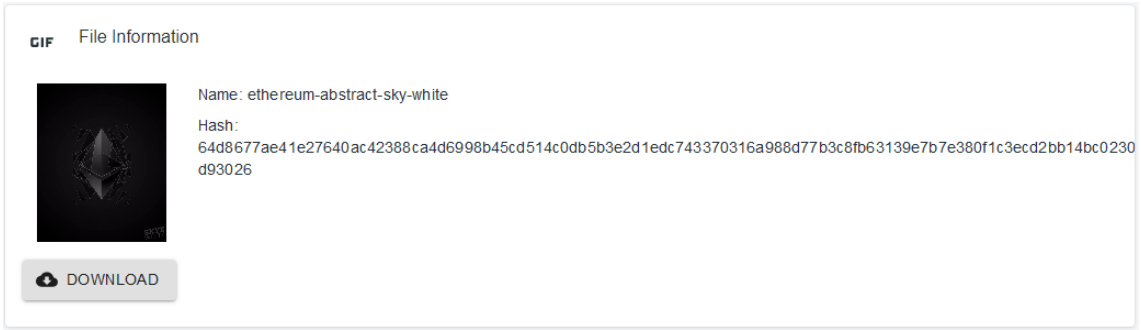


Figure 32 Web App. File Information – GIF

35 https://ropsten.etherscan.io/tx/0x9c6064089cdafd794642de92832908fb6bcc5318d64c7d5de88d5f7e4f009ea6

36 https://utils.iota.org/transaction/RRXTYDUBFDDLWCYCJZONMRAQAE9IDVTPKCNLHMXNIULOBWBKWZZIJVLGWAPDAOLHXJSBMBGGXVNXZ9999

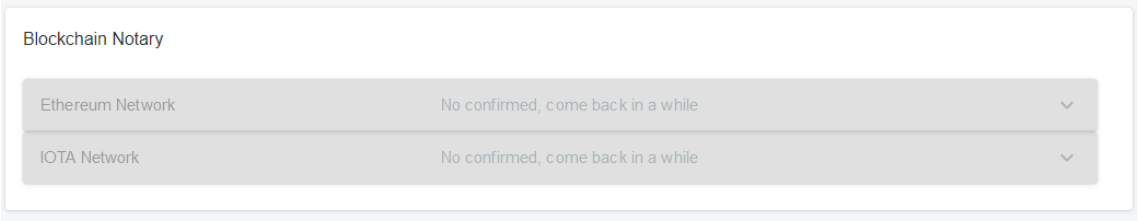


Figure 33 Web App. File Information – No Confirmed

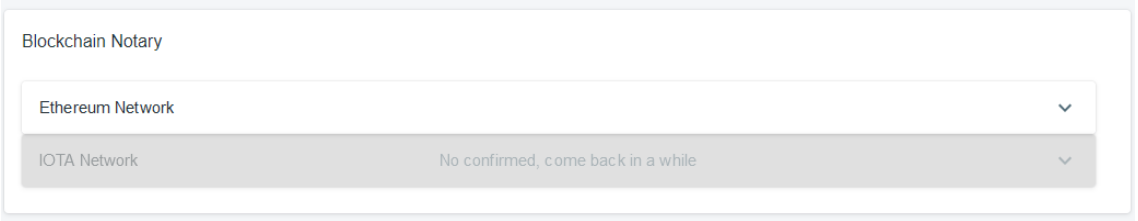


Figure 34 Web App. File Information – Partial Confirmation

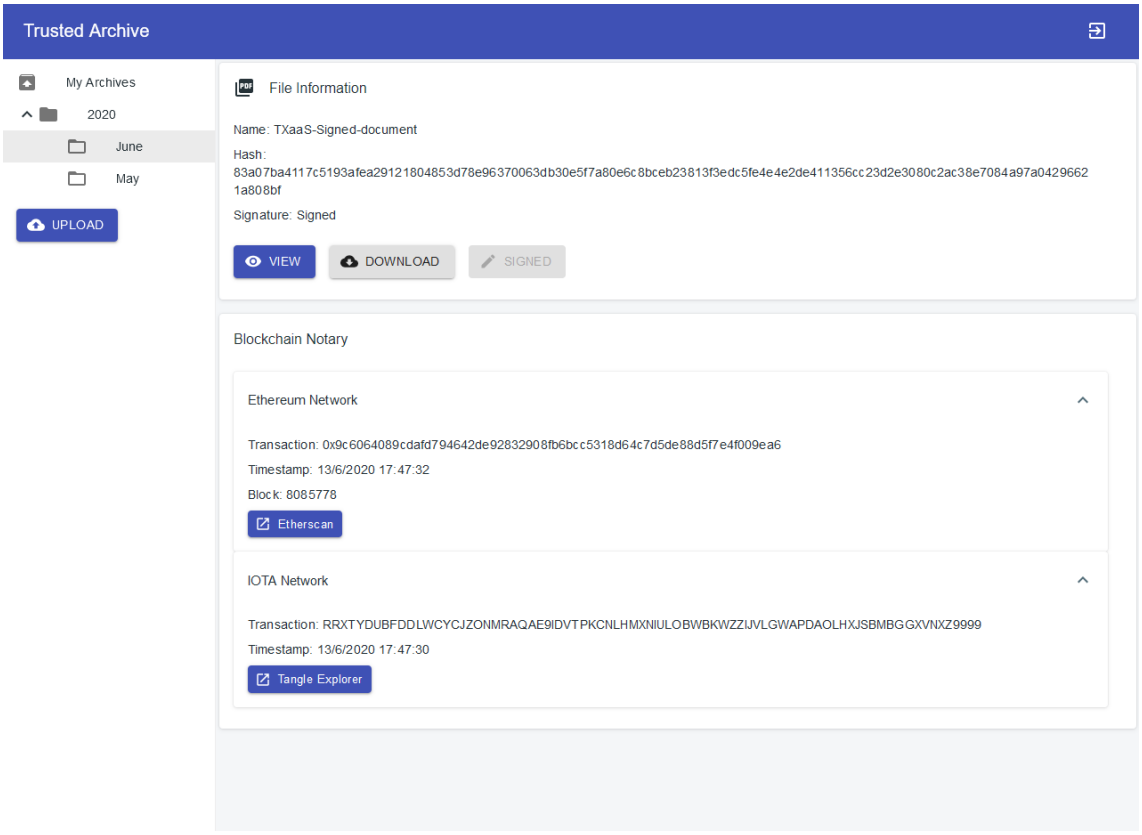


Figure 35 Web App. File Information – Complete

App Server

A web application in a browser does not have the means to manage the credentials of application securely. Since the flow of OAuth2 ACG consists of multiple parts, some of which require the use of keys securely, we have been forced to use a server. For this reason, we have used Nodejs to deploy the application and handle authentication.

The operation is the same as a conventional server of web applications, but more simply since we only offer the flow of the authentication OAuth2 ACG and the return of the web application. For this, we have used the Express framework that allows us to streamline the creation of a Web API. Finally, our little server consists of two APIs of GET method, and the variables CLIENT_ID and APIKEY are the credentials stored on the server.

The first API has the function of login to redirect to our service of User Authentication DPS of the TXaaS, according which in turn is configured in order that once the authentication is completed successfully, it redirects to our second API. We can see in Figure 36.

```
app.get('/:tenant/login', function (req, res) {
    let url = ACG_login_uri + req.params.tenant + ACG_login_param
        + "&client_id=" + process.env.CLIENT_ID
        + ACG_login_redirect + Uri_login_redirect + req.params.tenant + "/authorize"
        + ACG_login_scope_state;
    res.redirect(url)
});
```

Figure 36 App Server. Login

The second API has the function of obtaining the token and the return of the web application. A successful user authentication on the first API returns a set of parameters, which is the most important is the code that we will use together with our APIKEY stored on the server to request the Access Token to the Authentication Server (AS). As we can see in Figure 37, we return the web application to the user together with the Access Token, the time of expiration and the tenant.

Finally, through OAuth2 ACG, we ensure the security of our web application is robust and fully integrated with TXaaS, and allows greater flexibility in changing applications of the system. Both the web application and this mini server are hosted on Heroku with a direct link to the repository of GitHub.

```
app.get('/:tenant/authorize', function (req, res) {
    let headers = {
        "Authorization": 'Basic '+ process.env.APIKEY,
        "Content-Type": 'application/x-www-form-urlencoded'
    };
    let dataString = "grant_type=authorization_code&redirect_uri=" + Uri_login_redirect +
        req.params.tenant + "/authorize&code=" + req.query.code + "&state=" + req.query.state
    let options = {
        url: ACG_token_uri,
        method: 'POST',
        headers: headers,
        body: dataString
    };

    function callback(error, response, body) {
        if (!error && response.statusCode == 200) {
            let time = new Date().getTime() + JSON.parse(body).expires_in  * 1000
            res.sendFile(path.join(__dirname, 'client/build', 'index.html'))
            res.redirect(URL_APP +
                "?access_token=" + JSON.parse(body).access_token +
                "&expires_token=" + time +
                "&tenant=" + req.params.tenant
            )
        }
    }
    request(options, callback);
```

Figure 37 App Server. Authorize

Testing

In this section, we will proceed to explain the proofs of testing that are carried out in two groups, which correspond to the section backend and frontend.

Backend

Postman is the tool that is used to check the correct and expected operation of the APIs of backend during development as well as after the implementation.

The tool is complete and powerful, but for our use, we have used the basics. For this, we have configured an environment to configure the parameters that are requirements between them or shared, and for each API, we have created small Test scripts that are configured in the corresponding section of the Postman.

During the development, API calls were made, and the result checked manually for greater flexibility to display or modify possible variables.

Finally, once all the APIs have been implemented and the correct individual operation has been verified, we have used Postman's Collection Runner to automate the order of execution and the iterations (5 iterations) to check the Test scripts that we have

implemented for each API and use an environment shared among them for the required sequential variables. As we can see in Figure 38, we have successfully passed the Basic Test without errors. On the other hand, it should be noted that the response time is not included, in which we could see that all the APIs took less than 1s, except Notarize which obtains a very varied result that goes from 4s to more than 10s and NewAddresses that in certain cases exceeds 1s, but never more than 1.5s.

Figure 38 Testing. Postman

Frontend

Instead, for testing of the web application we have not used a sophisticated tool such as the proper tool that integrates ReactJS, but has been more trivial, through tools of the Firefox browser and our visual intuition of the content displayed. In addition to checking through the metrics offered as the “network” (see Figure 39) to compare with the expected and visual responses of our application.

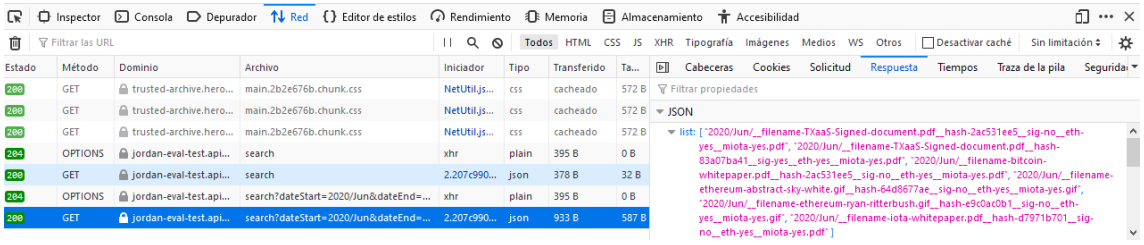


Figure 39 Testing. Firefox

Methodology and rigour

Methodology

This project uses the scrum methodology, a type of agile method which consists of divide and conquers. Complex tasks are divided into more straightforward tasks that are assigned an estimate of time to complete, and over of several weeks, a set of tasks could be selected. When it is completed, another round of tasks is chosen again in a self-regulated way according to individual capacity.

Process monitoring

Since most of the work is accomplished at home, the members of the team decided to have a meeting face to face or a video conference usually once a week to track the work, resolve questions related to the project and discuss the next steps for the development of the service.

The team realizes daily communication with the director by e-mail and between teammates use WhatsApp for messages and use Microsoft Teams or WebEx for voice calls or video calls.

A version control system can be very versatile and useful for analyzing and monitoring progress because it makes management much easier. Besides, we can save from some trouble causing by mistake or loss of code because Git has been used as a version control tool and GitHub as a platform for the convenience. Moreover, for the exchange of documents and files, OneDrive is used through shared folders between team members.

Validation

The validation of the tasks, weekly meetings are used to confirm the correct progress of the tasks. On the other hand, at any time, we can communicate electronically to carry out the verification.

The project manager usually does final validations, but before being given instructions and resources, an individual task check is performed by the other team members. In this way, we ensure the correct operations of the implementations.

Validation management is performed in the first step in local testing by using VMware. Once the local test has been completed, it proceeds to join an instance in AWS EC2 to

carry out checks and has a fully functional version in the Cloud, and this last part is in charge of the team manager, who will return feedback of the chores.

Initial Planning

The project, according to the agreement with the company, had an expected duration of 9 months with the start date on September 16, 2019, and the end on June 16, 2020. The time involved is 25 hours a week, with approximately 900 hours of work during the project.

Apart from the hours of the working day in the company, the extra personal time dedicated to reference research on this project is included. Such as the writing of the documents and the management of the project. So, the total duration is approximately 1132 hours, including all aspects. On the other hand, the hours which are dedicated in the specific phase correspond approximately to the spring semester of 2020 (February - June), more parts of the management project or other non-specific tasks.

The intention is to be able to present the oral defense, which has a planned period between June 29, 2020, and July 3, 2020.

Definition of tasks

This section describes the tasks that will be performed in this project. These tasks are divided into three different groups. Each task has an identifier, a brief description, and a temporary estimate.

[A] Project management

This group consists of tasks that direct and maintain the correct performance for the realization of the project. All the following activities have an approximate allocation total of 282 hours:

- [A1] Support tool: Study on a set of tools that will facilitate the performance of the work. For example, it can be a reference manager. It has an estimate of 5 hours.
- [A2] Contextualization and scope: Write an introduction and define the scope of the project. It has an estimate of 20 hours.

- [A3] Temporary plan: Manage and expose temporary planning to execute the activities efficiently and finish the project at the expected time. It has an estimate of 15 hours.
- [A4] Budget and sustainability: Document the cost of the project and the sustainable management involved. It has an estimate of 10 hours.
- [A5] Initial Milestone: This involves reviewing and improving activities A2, A3, and A4 and integrating it into the preparation of the final document of the initial Milestone. It has an estimate of 15 hours.
- [A6] Report: There are monthly reports of the company about the activities completed. But, at the same time, it is used for project documentation. It has an estimate of 50 hours.
- [A7] Thesis: The preparation of the thesis is based on the documentation referring to the activities. That is being completed throughout the project in parallel. Although a review of all the contents will be made and be perfected in the last instance before delivery, because of these reasons, the time spent will be significant. It has an estimate of 50 hours.
- [A8] Presentation: It consists of the previous preparation for the defense of this project in the day assigned. It consists of the development of the presentation, such as we could rehearse or demonstrate a demo of the project. It has an estimate of 35 hours.
- [A9] Meetings: These are weekly meetings that we complete the project with our manager. These meetings have the purpose of monitoring the work and exposing problems and solutions throughout the entire project. It has an estimate of 82 hours.

[B] Generic phase

This group consists of the first phase of the project that started before the registration of the TFG. It has a significant dedication, and it is fundamental and indispensable since it corresponds to the standard part of the team and is the basis of this project. It has an expected duration of 505 hours for the following tasks:

- [B1] Study TrustedX: Study what is the product TrustedX and what services does it offers, such as the digital signature. It has an estimate of 15 hours.

- [B2] Configure TrustedX: It consists of mounting the operating environment and configuration of TrustedX on the company laptop. It has an estimate of 25 hours
- [B3] Test TrustedX: Test the correct operation of TrustedX in the local environment. It has an estimate of 15 hours.
- [B4] TXaaS Structure: Design with the team the structure and definition of the project TXaaS. It has an estimate of 25 hours. (In Groups)
- [B5] TXaaS development environment: This activity is based on creating the resources necessary for the development of TXaaS. By emulation with VMware. It has an estimate of 5 hours.
- [B6] Study OpenDJ: Study the operation and options offered by the directory server that implements a wide range of Lightweight Directory Access Protocol (LDAPv3). It has an estimate of 20 hours.
- [B7] Configure OpenDJ: It consists of linking TrustedX with an external LDAP to create the first step of the configuration of TXaaS to make the directory server independent. It has an estimate of 15 hours.
- [B8] OpenDJ Structure: Plan, design, and implement the structure of LDAP for multi-tenant service. It has an estimate of 35 hours.
- [B9] AuthNApp: Design and create an authentication portal using a username and password and integrate it into a TrustedX Delegated Process Service (DPS) for multi-tenant authentication. It has an estimate of 55 hours.
- [B10] Other AuthNApp: Add and improve the AuthNApp application. Such as it can be the implementation of a Self-Service Password Reset (SSPR) to reset the passwords of users of OpenDJ multi-tenant through a One-Time Password (OTP). It has an estimate of 45 hours.
- [B11] TXaaS Basic Integration: Develop and unite all the essential components of TXaaS for the proper function of local a PC. It has an estimate of 50 hours.
- [B12] TXaaS Cloud: Know the basic AWS options to configure and support the migration of the TXaaS service to the Cloud. It has an estimate of 25 hours. (In Groups)
- [B13] TXaaS TBD: it is a set of undefined tasks that are not included in the initial planning of the manager. It serves to improve or fix problems that arise throughout the project. It has an estimate of 175 hours. The hours will be distributed as new tasks arise or other tasks that require it.

[C] Specific phase

This phase is focused on the specific branch and primary objective of this project, Trusted Archive with Blockchain. It has an expected duration of 345 hours for the following tasks:

- [C1] Study about the trust of Blockchain: Analyze the trust involved in Blockchain and its technology. It has an estimate of 35 hours.
- [C2] Study on eIDAS: Observe and know a basic notion of the function of the regulations that directly affect this service. It has an estimate of 5 hours.
- [C3] Compare with traditional service: The purpose of this task is to analyze and compare Blockchain with a traditional centralized archiving service. It has an estimate of 15 hours.
- [C4] Study different Blockchain: It consists of analyzing different Blockchain of today and which are the most suitable for this objective. It has an estimate of 35 hours.
- [C5] Register hash in a transaction: This task consists in implementing the functionality of the register in a transaction of Blockchain. It has an estimate of 30 hours.
- [C6] Trusted Archive GUI Design: Design the Graphical User Interface for interaction, perform transactions, and display all the functionalities necessary for the Trusted Archive service. It has an estimate of 35 hours.
- [C7] Trusted Archive GUI DEV FrontEnd: Implement points C5 and C6 FrontEnd. It has an estimate of 55 hours.
- [C8] Trusted Archive GUI DEV BackEnd: Implement points C5 and C6 BackEnd. It has an estimate of 65 hours.
- [C9] Trusted Archive integration in TXaaS: Add the functionality of Trusted Archive in the TXaaS service. It has an estimate of 35 hours.
- [C10] Trusted Archive Testing: Perform appropriate tests for the proper functionality of this service. It has an estimate of 10 hours.
- [C11] Innovation: It consists of studying, analyzing, and thinking about other types of innovations that Blockchain technology can be applied in this technological area. It has an estimate of 25 hours.

Temporary estimation

Dependencies

Table 7 shows a summary of all the tasks that will be performed during the project. Where each row is a specific task, and the columns respectively represent the identifier, task name, the estimate of the assigned time, and the dependencies of each task.

The task that depends on others can be started when it has a necessary minimum of operating requirements, close to the completion of the predecessor tasks.

Gantt estimated

Figure 40 observes the Gantt chart about the temporal planning of this project. As we can see, the timeline of the first 5-6 months is grouped monthly, which coincides with a report of the activities of the development of the generic phase and the theoretical part of the specific phase. Finally, the last few months, we focus on the development of the Trusted Archive and the documentation of the project.

Deviations and final Gantt

Regarding the temporal planning, from the beginning of the project with the acquisition of knowledge on Blockchain of the specific phase and with the implementation of the generic phase, they have been followed and fulfilled as planned. In other words, until week the 25 of Gantt estimated (see Figure 40), in the middle of march.

However, from week 25, there were changes that we can see in Figure 41 of the final Gantt. The time which is dedicated to the TXaaS TBD was used for the implementation of the specific phase, mostly in tasks C7 and C8, which has been prolonged more than expected.

Resources

Human resources

The TXaaS project, as indicated above, is composed of a team of four members, including the manager. This project is a specific part of the project TXaaS, and the author of this document, he is responsible for preparing all the tasks as mentioned earlier.

Material resources

The materials needed to complete this project are grouped into two groups:

- Hardware: This group includes the company's laptop, personal computer, and AWS services (S3, EC2, and SES).

- Software: These resources are necessary programs which are used on the company's laptop and on the personal computer to perform the work, which consists of VMware, Visual Studio, Eclipse IDE, Atom and the complete Microsoft 365 Enterprise package (Teams, Outlook, OneDrive, Office ...) and web browsers. Besides, OpenDJ and TrustedX are resources of the service of TXaaS.

ID	Task name	Time (h)	Dependence
A	Project management	282	—
A1	Support tool	5	—
A2	Contextualization and scope	20	—
A3	Temporary plan	15	—
A4	Budget and sustainability	10	—
A5	Initial Milestone	15	A2;A3;A4
A6	Report	50	—
A7	Thesis	50	—
A8	Presentation	35	A7
A9	Meetings	82	—
B	Generic phase	505	—
B1	Study TrustedX	15	—
B2	Configure TrustedX	25	B1
B3	Test TrustedX	15	B2
B4	TXaaS Structure	25	—
B5	TXaaS development environment	5	B2
B6	Study OpenDJ	20	—
B7	Configure OpenDJ	15	B6
B8	OpenDJ Structure	35	B6
B9	AuthNApp	55	B2;B4;B5;B7;B8
B10	Other AuthNApp	45	B9
B11	TXaaS Basic Integration	50	B5;B7;B9
B12	TXaaS Cloud	25	B11
B13	TXaaS TBD	175	—
C	Specific phase	345	—
C1	Study about the trust of Blockchain	35	—
C2	Study on eIDAS	5	—
C3	Compare with traditional service	15	C1
C4	Study different Blockchain	35	—
C5	Register hash in a transaction	30	C4
C6	Trusted Archive GUI Design	35	C2;C3;C4
C7	Trusted Archive GUI DEV FrondEnd	55	C6
C8	Trusted Archive GUI DEV BackEnd	65	C6
C9	Trusted Archive integration	35	C7;C8
C10	Trusted Archive Testing	10	C9
C11	Innovation	25	—
	Total	1132	

Table 7: Summary of the tasks to be performed

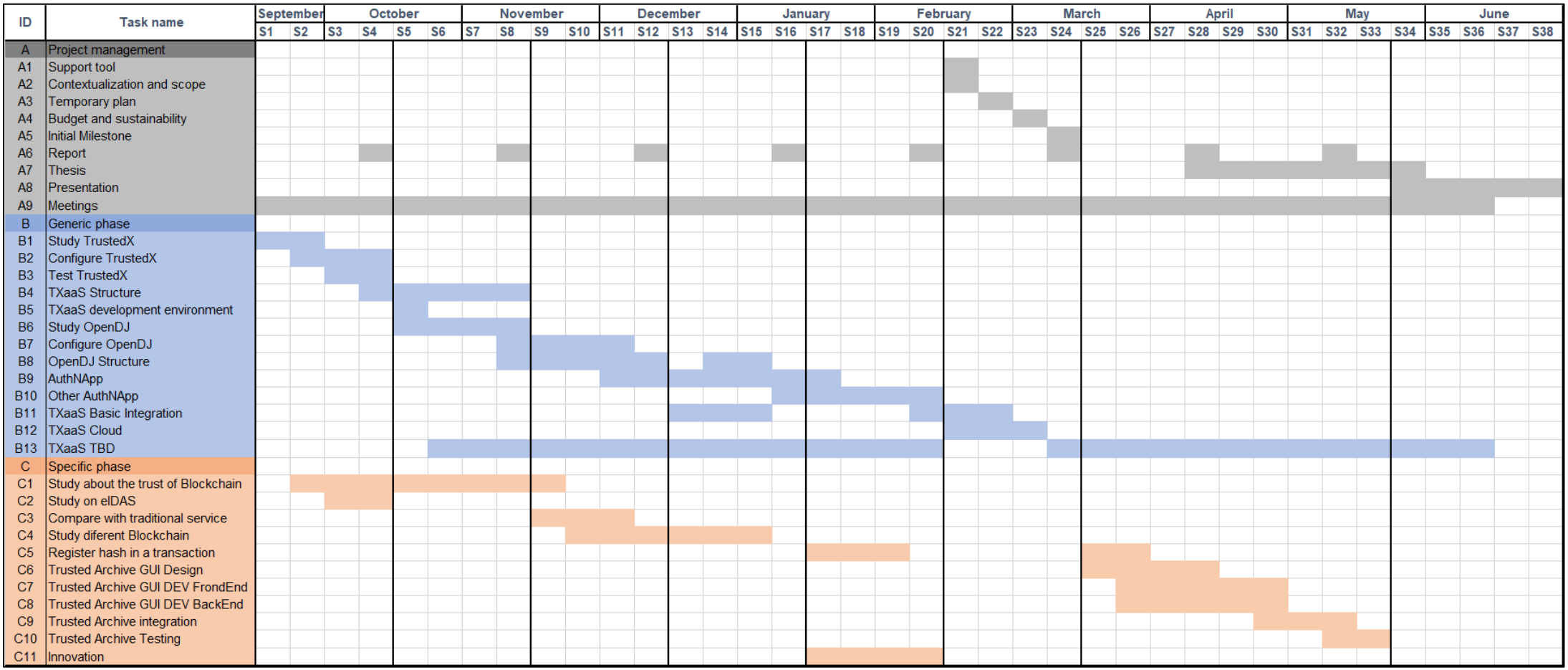


Figure 40 Gantt estimated

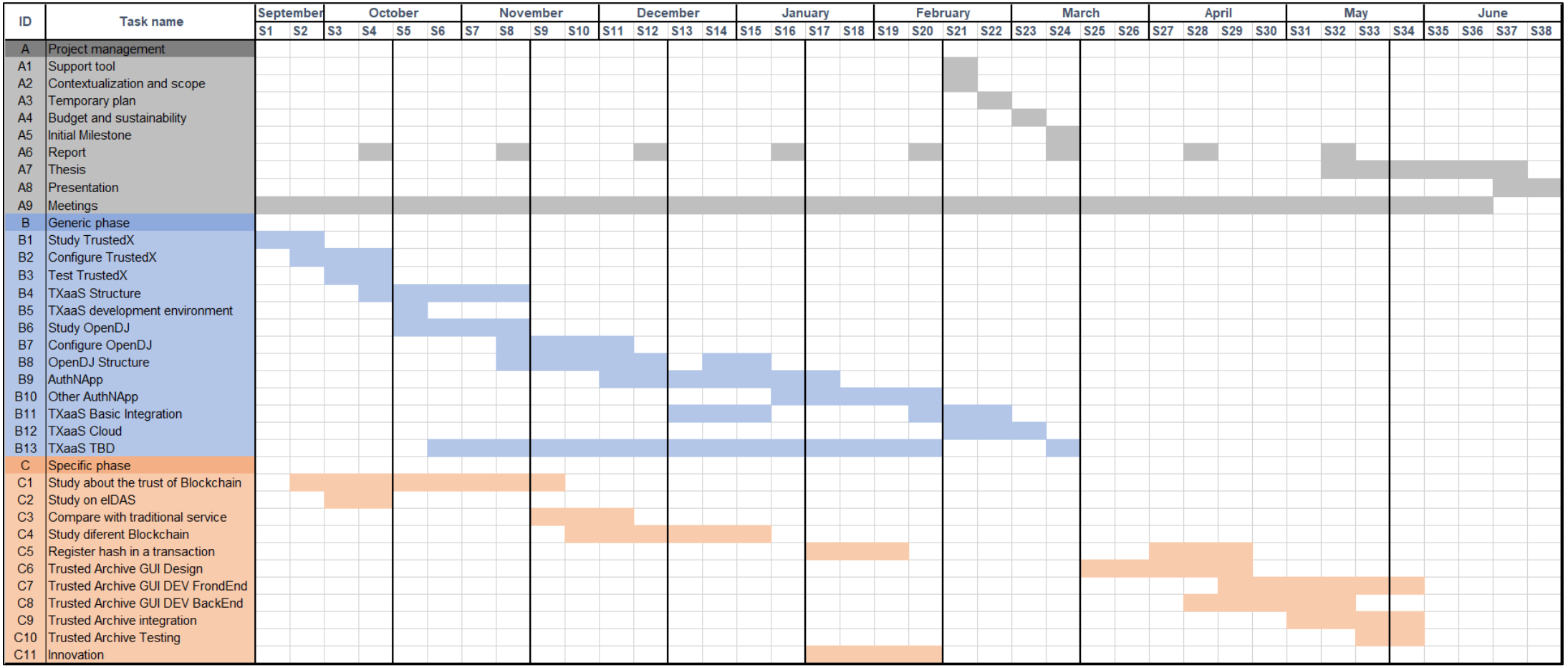


Figure 41 Final Gantt

Risk management

As indicated in the Challenges and Barriers section, the three main problems are:

- Time. In case of bad planning of the time or duration of the tasks, a deviation arises. In extreme cases, it is not possible to extend weekly hours; the tasks would have to be adjusted, avoiding the less important characteristics and keep the tasks less critical for the last instance. With this alternative, we get the fulfillment of the fundamental functions to prevent the critical problem of the project if we do not have more time.
- The volatility of the Blockchain. This problem is foreign to our actions since volatility cannot be controlled; it depends entirely on the market. However, to mitigate as much as possible the adverse effects that it causes us, it would be created a good plan for forecasting the use of the network that we need and make the initial purchase of assets necessary for a time interval. Through this plan, we ensure that the price per constant transaction, but in case of extreme volatility and the cost of the transaction is high, the decision will be made by the company and is beyond our reach. That would only be in productivity. In the development, we have a Testnet that the cost is null.
- New technologies. This problem is related to the first problem that has commented on before, and the solution would be to take more hours. Alternatively, only in the early phase of the project, change to another environment or programming language. Although, in this case, ReactJS is very flexible and facilitates interpolation between different frameworks.

Although there are other minor risks, it is also important to mention it that its impact is almost nil:

- Bureaucracy. When we request a resource from the company, there has a waiting time, in this case, the only solution is to wait. However, we can also mitigate the effects of delays using other temporary resources if it exists. Otherwise, there is no other choice but to proceed with a different task that does not depend on a requested resource. This case also includes the use of other resources, especially those for payment. Therefore, it is very important to decide what resources will be needed in the first months.
- Data loss. The best solution is to use applications or services that facilitate data recovery in case of loss. To do that, there must follow useful guidelines and have a backup copy. Therefore, Git and GitHub is the ideal solution.

- Hardware failure. As we mentioned, the possibility is very low since we have two computers. The alternative plan, in case of simultaneous failure, it would be to use a computer that is available in the offices or to use the resources of the FIB.

Finally, we also have a task (B13) that has many hours that does not have a definition of specific jobs, which is used to solve problems or other unforeseen tasks. Only if necessary, this time can be used for other more critical tasks for the functional. In this way, we can ensure success in the completion of the project by having a plan B in case of emergency and focus on the main functionalities.

Budget

Human resources budget

The human resources budget is one of the essential sources in almost all projects. Because of this, we have analyzed the occupations and the number of hours spent in each profile.

The author of this TFG occupies the following roles within this project: project manager, DevOps, designer, and developer.

DevOps is a set of practices that combines development and operations with the idea of streamlining the process from development to implementation [35]. Therefore, the designer and developer profiles could be part of the DevOps process, but the decision to separate it could benefit the project economically.

According to the statistics report of the website *Tecnoempleos* in March 2020, we obtained the average gross wages per year in Spain [36]. According to Spanish regulations, one person works eight hours a day; in other words, including holidays, 1691 hours per year [37]. There have calculations to obtain an approximation of the hourly prices of each profile in Table 8. Instead, the author of this project has a labor contract of 25 hours a week with a salary of 9 €/h and will occupy all the positions as mentioned earlier. The TAX of 35% is an approximate value of all taxes (VAT, personal income tax, social security, etc.) is included.

Profile	Annual salary	Hourly Wage
Project manager	35.200,00 €	20,82 €
DevOps	31.500,00 €	18,63 €
Web designer	25.400,00 €	15,02 €
Software development	26.100,00 €	15,43 €

Table 8: Approximate salary of the profiles

Task name	Time	PM	DevOps	Design	Dev	Cost
Project management	282 h	282 h	0 h	0 h	0 h	5.870,14 €
Support tool	5 h	5 h	—	—	—	104,08 €
Contextualization and scope	20 h	20 h	—	—	—	416,32 €
Temporary plan	15 h	15 h	—	—	—	312,24 €
Budget and sustainability	10 h	10 h	—	—	—	208,16 €
Initial Milestone	15 h	15 h	—	—	—	312,24 €
Report	50 h	50 h	—	—	—	1.040,80 €
Thesis	50 h	50 h	—	—	—	1.040,80 €
Presentation	35 h	35 h	—	—	—	728,56 €
Meetings	82 h	82 h	—	—	—	1.706,92 €
Generic phase	505 h	50 h	295 h	25 h	135 h	8.995,27 €
Study TrustedX	15 h	—	15 h	—	—	279,42 €
Configure TrustedX	25 h	—	25 h	—	—	465,70 €
Test TrustedX	15 h	—	15 h	—	—	279,42 €
TXaaS Structure	25 h	10 h	15 h	—	—	487,58 €
TXaaS development environment	5 h	—	—	—	5 h	77,17 €
Study OpenDJ	20 h	—	20 h	—	—	372,56 €
Configure OpenDJ	15 h	—	15 h	—	—	279,42 €
OpenDJ Structure	35 h	20 h	15 h	—	—	695,74 €
AuthNApp	55 h	—	—	15 h	40 h	842,70 €
Other AuthNApp	45 h	—	—	10 h	35 h	690,42 €
TXaaS Basic Integration	50 h	—	50 h	—	—	931,40 €
TXaaS Cloud	25 h	—	25 h	—	—	465,70 €
TXaaS TBD	175 h	20 h	100 h	—	55 h	3.128,03 €
Specific phase	345 h	25 h	135 h	35 h	150 h	5.876,11 €
Study about the trust of Blockchain	35 h	—	35 h	—	—	651,98 €
Study on eIDAS	5 h	—	5 h	—	—	93,14 €
Compare with traditional service	15 h	—	15 h	—	—	279,42 €
Study different Blockchain	35 h	—	35 h	—	—	651,98 €
Register hash in a transaction	30 h	—	—	—	30 h	463,04 €
Trusted Archive GUI Design	35 h	—	—	35 h	—	525,72 €
Trusted Archive GUI DEV FrondEnd	55 h	—	—	—	55 h	848,91 €
Trusted Archive GUI DEV BackEnd	65 h	—	—	—	65 h	1.003,25 €
Trusted Archive integration	35 h	—	35 h	—	—	651,98 €
Trusted Archive Testing	10 h	—	10 h	—	—	186,28 €
Innovation	25 h	25 h	—	—	—	520,40 €
Total	1132 h	357 h	430 h	60 h	285 h	20.741,51 €

Table 9: Breakdown of expenses by task and role

The occupations of the roles are assigned to each of the temporary planning activities to calculate the individual cost and the approximate total cost, see Table 9. Similarly, Table 10 shows the cost associated with each profile.

Profile	Time	Hourly Wage	Cost
Project manager	357 h	20,82 €	7.431,34 €
DevOps	430 h	18,63 €	8.010,05 €
Web designer	60 h	15,02 €	901,24 €
Software development	285 h	15,43 €	4.398,88 €
Total			20.741,51 €

Table 10: Budget profiles

General budget

In Table 11, it can be seen that the costs of material resources and amortization. Since the personal computer with all its peripherals is used for convenience that is expendable and is already amortized. Additionally, AWS S3, SES, and EC2 services have free versions up to a certain amount of processing that is sufficient for the development of this project, except two EC2 instances used for the TXaaS Cloud that requires more computational calculation. However, all the software or services mentioned are free in the section of material resources that are not specified in the table. In addition to other services or software that is not mentioned, but is used in the project, such as Apigee or Splunk. All of them use trial licenses, evaluation or are opensource.

The calculation of the amortization is based on the following formula that applies to each product, except for services. Considering that the laptop has a residual value of 500 euros after a lifetime.

```
Amortization = (Initial cost – Residual value) / Lifetime
```

Resources	Price	Lifetime	Amortization	Cost
ThinkPad T480 i7-8650U vPro 32GB/512GB	2.000,00 €/unit[37]	4 years	375,00 €	2.000,00 €
Microsoft 365 Enterprise	19,70 €/month[38]	9 months	-	177,30 €
AWS Services EC2 t2-medium	4,64 €/month[39]	9 months	-	41,76 €
Total			375,00 €	2.177,30 €

Table 11: Budget material resource and amortization

[37] https://www.lenovo.com/es/es/laptops/thinkpad/t-series/ThinkPad-T480/p/22TP2TT4800
[38] https://www.microsoft.com/es-es/microsoft-365/business/compare-more-office-365-for-business-plans
[39] https://calculator.s3.amazonaws.com/index.html

On the other hand, Table 12 shows the expenses that it indirectly involves, such as electricity and the Internet, and the workplace. The home is the main work area but has been simulated based on the approximate price of 380 euros per month for a dedicated-desk in Barcelona according to *WeWork*[40]. To perform the calculations has been based on the price of *MásMóvil*, 30 euro a month[41], to obtain the hourly cost of the Internet and the average daily electricity prices indicated by the website *Tarifaluzhora* on the day of the consultation[42].

Resources	Price	Lifetime	Cost
WeWork	380 €/month	9 months	3.420,00 €
Internet	0,04 €/h	1132 h	47,17 €
Electricity	0,09 €/h	1132 h	98,48 €
	Total		3.565,65 €

Table 12: Indirect costs

Total budget

Before obtaining the total cost, has been created a contingency of 15% of the subtotal in human, material and indirect resources as a precaution. Equivalent to 3.972,67 Euros (26.484,46 Euros * 0.15).

On the other hand, as each project may arise the unexpected, specifically in this project has a high probability of a deviation from the time allocated to human resources. Therefore, the cost of an increase of 15% has been reserved, which represents 3.111,23 Euros (20.741,51 Euros * 0.15).

As a summary, we can see in Table 13 the sum of all the expenses indicated above. The table would represent the approximate total cost of this project, including contingency and contingencies.

Resources	Cost
Human resource	20.741,51 €
Material resource	2.177,30 €
Indirect cost	3.565,65 €
Subtotal	26.484,46 €
Contingency	3.972,67 €
Incidentals	3.111,23 €
Total	33.568,36 €

Table 13: Total budget

[40] https://www.wework.com/es-ES/workspace/dedicated-desk
[41] https://www.masmovil.es/
[42] https://tarifaluzhora.es/?tarifa=normal&fecha=2020-03-07

Management control

For the trace of the budget, an update of the budget based on actual consumption, including hours and other expenses, will be carried out for each finished activity.

The budget related to human resources would be the most critical point in this project since the planning of the duration of the activities is approximate and due to unforeseen circumstances or other difficulties, it cannot be known precisely the period until the activity is finished. Because of this, the calculation of the deviation of each activity will be carried out according to the following formula:

```
Deviation = (Estimated cost per hour – Real cost per hour) * Total hours
```

On the other hand, the general costs will not be very altered, since only serious accidents could adversely affect. However, the probabilities that occur are almost nonexistent, or the impact of the repercussions is low compared to the entire project. Similarly, the following formula can be used to obtain deviation information:

```
Deviation = Estimated Costs – Real Costs
```

Through this information, we can compare the real cost of the project with the estimate and verify that the contingency and the unexpected of margins were adequate.

The primary deviations would be centered in the critical zone of the project, which resides in human resources in the accomplishment of the tasks. We can locate 2 critical points:

- The first is located in the course of the completion of the general phase and the beginning of the development of the specific phase. In the middle of February and March, we have begun the implementations of the tasks TXaaS integration and Cloud of the general phase.
- The second. Consists of completion of the specific phase and integration with TXaaS in the first weeks of May.

In summary, in these two time zones, we should have more possibilities of causing a deviation from the human resources budget due to incompatibility or unexpected operation of the implementations of these tasks performed. On the other hand, as I mentioned, the deviation in overall cost is very unlikely to occur.

Sustainability

Self-assessment

About sustainability, I have already known the three dimensions involved a project TIC. After conducting the survey, I realized that I have good knowledge of the economic dimensions, aspects of improving in the social dimension and much to advance in the environmental dimension.

Although I do not have experience previously, when it comes to a project of information technology, I have a habit of prioritizing economically, since it is essential for me to know the feasibility. At the same time, for the good future of the project, it is important to know the impact that it can represent in society.

On the one hand, I have minimal knowledge about the impact on sustainability because my proactive form has never reflected on this issue and the repercussions that I could imply, especially the environmental one. Therefore, to find the point of balance in sustainability, I have to assess and improve my shortcomings, since all my current knowledge about this section comes from my university education.

On the other hand, my reflection on a project with Blockchain technology. I think that it can conceptually positively influence sustainability and achieve a holistic solution that can affect the economic, social and environmental dimensions because of its own characteristic's technology.

Finally, as I have said, I must enrich my knowledge to make better decisions that have a good social commitment, economically suitable and especially respectful of the environment.

Economic dimension

Regarding PPP: Reflection on the cost you have estimated for the completion of the project

In the economic context, the estimated cost is consistent and reasonable considering that it is a more extended project than usual and having used the necessary minimum resources, since the main price is from the human resource that cannot be limited. Example, the company's laptop is the only element of a massive budget of the material resource. Still, it is also justified with the intensive use of virtualization with VMware that requires a high-power computer.

Regarding Useful Life: How are currently solved economic issues (costs...) related to the problem that you want to address (state of the art)? And, how will your solution improve economic issues (costs ...) with respect other existing solutions?

The use of shared resources and on-demand Cloud services, it considerably reduces the expenses in material resources in favor of obtaining a service on demand. Therefore, since it is a digital service, it has a lower cost compared to other non-digital solutions.

On the other hand, the solution cannot be improved much because it is necessary for proper operation. Without the mentioned resources or the decrease in the characteristics of the general resources, it could imply an increase in the principal cost.

Environmental dimension

Regarding PPP: Have you estimated the environmental impact of the project?

In the first instance, no take into account the environmental impact, but into the economy, which in this aspect is related to the use of the minimum resources needed. Furthermore, with the use of shared digital resources and implementation through Cloud services, we almost completely avoid the use of physical resources. Therefore, minimizing the cost has also influenced the reasonable use of resources, and that positively affects the environment.

Regarding PPP: Did you plan to minimize its impact, for example, by reusing resources?

In a first approach was to reuse a laptop of the company, if it existed. Still, at that time, if not have a laptop with the necessary specifications and with the recent acquisition of the company that involves changes corporate materials, the company has chosen a new one. Thanks to the fact that it is a high-power laptop, it can be used for other projects that require a not-so-modern computer of the time and get the maximum possible performance. On the other hand, work from home, almost all the personal components that help the reuse are reused, such as the personal computer monitor.

Another plan to minimize the environmental impact, it was a good idea to use shared resources as a work tool, since it facilitates our work, in addition to spending fewer resources.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)? And, how will your solution improve the environment with respect other existing solutions?

One of the most significant environmental problems is the excessive use of material resources. However, through this service, we favor the digitization and the dematerialization. Thanks to this digital service of Trusted Archive, we can reduce the environmental cost, for example, the printing document of traditional signature in favor of keeping the document digitally signed witch is long-term.

Could you develop better and with less resources and more effective in a potential second generation or if you would do a second new project like this one?

Blockchain technology still has a long journey of improvement and innovation to be able to unleash its full potential. Therefore, if, in the future, the Blockchain technology improves its performance, consumption and adaptation weaknesses, in addition to providing the capability to store documents efficiently. No doubt, a second project will have considerable environmental and economic improvements.

Social dimension

Regarding PPP: What do you think you will achieve -in terms of personal growth- from doing this project?

Personally, this project gives me the technical knowledge and unique experience in the workplace, as well as working on a new emerging technology that I am passionate about it.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)? And, how will your solution improve the environment with respect other existing solutions?

This system implies a direct improvement in society because it facilitates the management and speeds up the use of digital signatures, therefore, it helps to reduce the digital divide in society. On the other hand, offer greater security, confidence, and legal recognition with signed documents thanks to the new Blockchain technology.

Regarding Useful Life: Is there a real need for the project?

Certainly, it exists a real need, because it must adapt to new technologies, but not stagnate, in addition to complying with European eIDAS regulations. Also, we improve the cost efficiency for society to obtain legal recognition in this information age. Furthermore, it offers the company the ability to continue innovating and be at the forefront by providing long-term Trusted Archive services.

Sustainability Matrix

The analysis of a project's sustainability is divided into three parts, identified by the matrix columns of Table 14.

	PPP	Exploitation	Risks
Environmental	Reasonable electricity consumption and use only when needed. Improve resource management and reuse of materials	It favors digitization and dematerialization	Although it is low, it depends on the environmental policy of external services. For example, Amazon or maintenance of a Blockchain, perhaps the environmental cost in the future will increase more than the expectation.
Economic	The budget is appropriately adjusted to the type of project	Not have many things to improve, perhaps in new implementation or look for other cheaper Blockchain networks	Very low, since it is a multinational company and on the part of the FIB, perhaps the limitation of the hours of the contract
Social	Technical knowledge and unique experience of a new technology	Improve confidence and security while streamlining the management of digital signatures and responsible consumptions	Reject Blockchain technology, because of the bad news about Blockchain-related hacking

Table 14: Sustainability Matrix

Conclusions

Technical competences

Throughout the degree, knowledge has been obtained in multiple areas of computing and, with the development of this project, we have been able to apply it to the following technical skills:

CTI2.3 To demonstrate comprehension, apply and manage the reliability and security of the computer systems (CEI C6). [Enough]

Cryptographic knowledge was applied throughout the project, such as symmetric or hash cryptography in the implementation of the AuthNApp authentication application, or the understanding of the Blockchain and the asymmetric cryptography application in the specific part (RSA in IOTA), in addition to applying security in the web application, such as avoiding the top vulnerabilities according to OWASP.

CTI3.1 To conceive systems, applications and services based on network technologies, taking into account Internet, web, electronic commerce, multimedia, interactive services and ubiquitous computation. [In depth]

This competition has been presented throughout the development of the project, both in the design until the implementation of the backend and the frontend. As it can be seen, the system starts from cero (except TrustedX), therefore, all factors have been taken into account to design and implement TXaaS and Trusted Archive.

CTI3.2 To implement and manage ubiquitous systems (mobile computing systems). [Enough]

It can be seen that the web application is responsive and adaptable to all types of devices and sizes, like desktop, tablets or mobile.

CTI3.3 To design, establish and configure networks and services. [Enough]

It is an important competence that we have had to apply to understand, configure, design and adapt the TrustedX (on-premises) product to TXaaS (as a services), in addition to connecting all the technologies which are used, such as the Cloud (S3), VMware, servers (Tomcat, NodeJS), etc.

CTI4 To use methodologies centred on the user and the organization to develop, evaluate and manage applications and systems based on the information technologies which ensure the accessibility, ergonomics and usability of the systems. [Enough]

The application of this competition is mainly focused on authentication for safe, ergonomic access and ease of use with the AuhtNApp authentication application, including SSPR (reset password) and its use with OAuth2 and SSO at Trusted Archive.

Conclusions

Once the development of this project is completed, it is observed that the main objectives have complied satisfactorily.

On the one hand, TXaaS is deployed and operated as an as-a-service service with all the new technologies incorporated and Trusted Archive is also deployed. As we can see in the figures Web App and Annex, which represents the last completed phase of the project, in which it consumes the implemented APIs. In addition to allowing the archiving of PDF documents, it also accepts GIFs, both for viewing and notarizing.

On the other hand, since cryptocurrencies are volatile, we consider that it is interesting to recalculate the cost of a transaction after 6 months:

Bitcoin at $9310, June 2020:

$$Tx_{feeBTC} = (226 + 64)\ bytes \times 22 \frac{satoshis}{bytes} = 6380\ satoshis = 0.0000638\ BTC$$

$$Tx_{feeBTC} \times \$9310 = \$0.59$$

Ethereum at $230, June 2020:

$$Tx_{feeETH} = (21000 + 68 \times 64)\ Gas \times 15\ \frac{gwei}{Gas} = 380280\ gwei = 0.00038028\ ETH$$

$$Tx_{feeETH} \times \$230 = \$0.087$$

As we can see, the cost of Ethereum transactions has increased considerably, but it is still lower than the cost of Bitcoin, which has increased slightly. However, the initial December gas can still be applied (10 gwei / gas, although it takes longer to confirm). Therefore, the volatility problem should not be an impediment if a good forecast and plan is realized in the initial purchase of the cryptocurrency to achieve stability in the budget.

On the other hand, IOTA has not worked as we expected, and we consider that it is not a very suitable red for our purpose, for two reasons:

- Unconfirmed transactions. The operation of the Tangle network is unstable, that requires a large number of activities in the network to offer a good percentage of confirmed transactions. However, when the network has little activity, it is unable to confirm transactions.
- Persistence of transactions or permanodes. The possibility that IOTA offers to eliminate old transactions (snapshot), and the little existence of permanode causes a dependency on a few nodes (only 1 permanode public has been found), therefore it is equivalent to a centralized database.

There have solutions, but we think that they are not very consistent with usability (sending a new transaction), or require more infrastructure (database). Therefore, we consider that IOTA does not comply the objectives of this project, because it cannot guarantee that the transactions will be confirmed or that the confirmed transactions will be long-lived (reduced permanode and centralized). However, as justification, IOTA, we use it as a secondary network that was initially selected to investigate a different technology than Blockchain.

Finally, can Blockchain technology provide greater efficiency and confidence to documents signed? The answer is yes since through a transaction on the Ethereum network, and we can verify and confirm that a document signed which existed in a specific temporary instance with a current cost of $ 0.087. However, we should realize a more exact investigation about the accuracy of the timestamp of the Ethereum, which we will put for future work.

Problems and obstacles

The main difficulties of this project have been at a technical level in certain tasks of the phase of implementation:

- Java versions. The initial idea was to implement the APIs for the use of the different Blockchain in the Tomcat of TrustedX, but due to the incompatibility of the Java version, we had to use an independent server. However, the main problem was not the incompatibility, but the time wasted in vain in looking for compatible versions and the rewriting of the code.
- ReactJS. One of the main difficulties consists of the learning of this Framework and the language of JavaScript programming, in which, personally, I was not

used to this type of syntax or to asynchronous logic. But once I had acquired enough knowledge, it was easy to implement the basic elements.

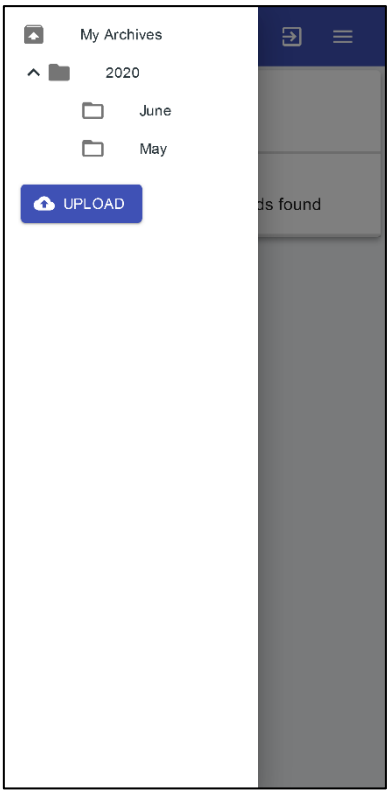
Another factor that has been a problem consists of the design of the initial architecture, which was intended to be used by Apigee to handle all RESTful API calls in order to perform any action and communication between DLT and S3. But we did not take into account that the authentication method required to invoke S3 APIs is not a standard nor trivial to implement in Apigee (competing company). Therefore, we had to restructure the design in order that the operations of actions and communications between the DLT and AWS S3 were managed by a server offering an API service to Apigee, and in order that this latest take charge of the secure communication between the client and the server.

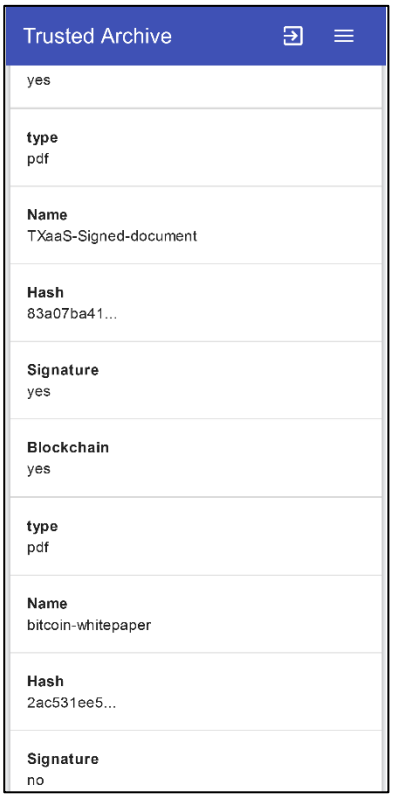
Future work

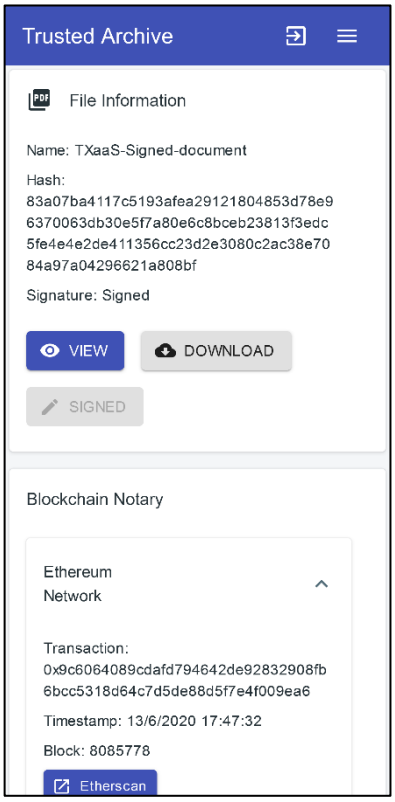
Next, I will explain more detail about some proposal of future works that will be carried out of this project:

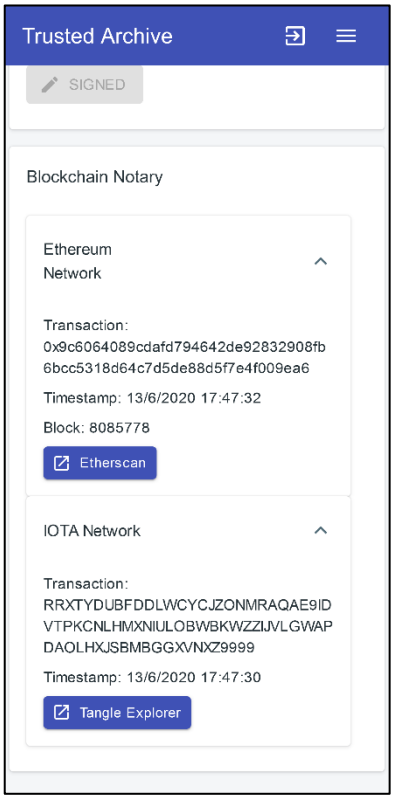
- Continue expanding functionalities, and improving services and the web application, such as the implementation of Figure 26 to obtain statistics that could be interesting for the user.
- As explained, an adaptive plan of the classes of S3 storage could be applied to reduce cost. However, it would only be interesting if it exists in production and mass use, which uses large quotas of storage.
- Implement a system for automatic archiving. Currently, the documents that are signed by TXaaS must be manually saved. The idea is to offer the option of an automatic save in Trusted Archive after signing a document.
- Complete the function of the button "sign" to sign a document.

Annex Smartphone

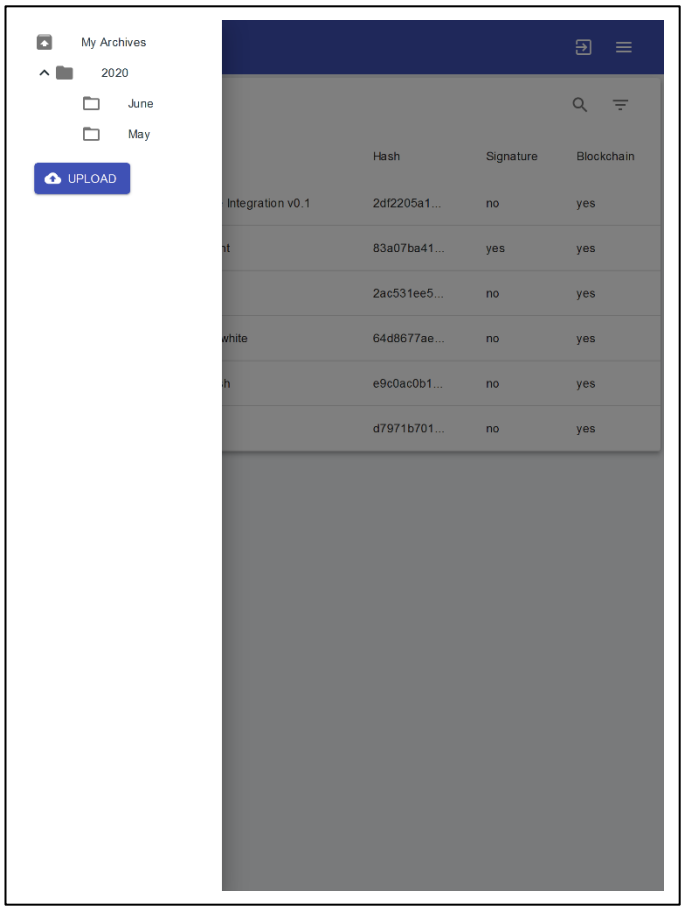


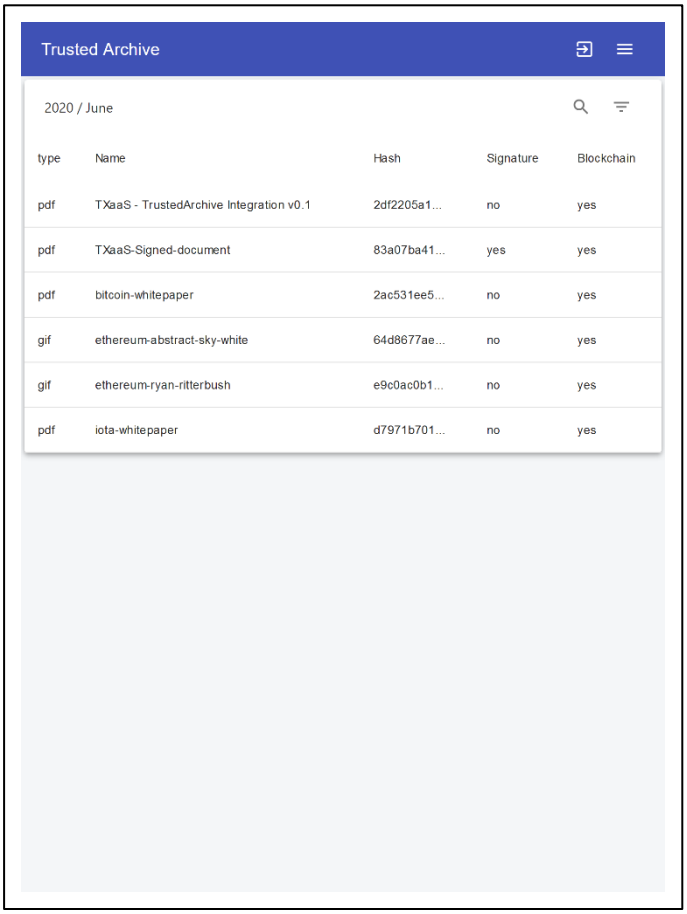


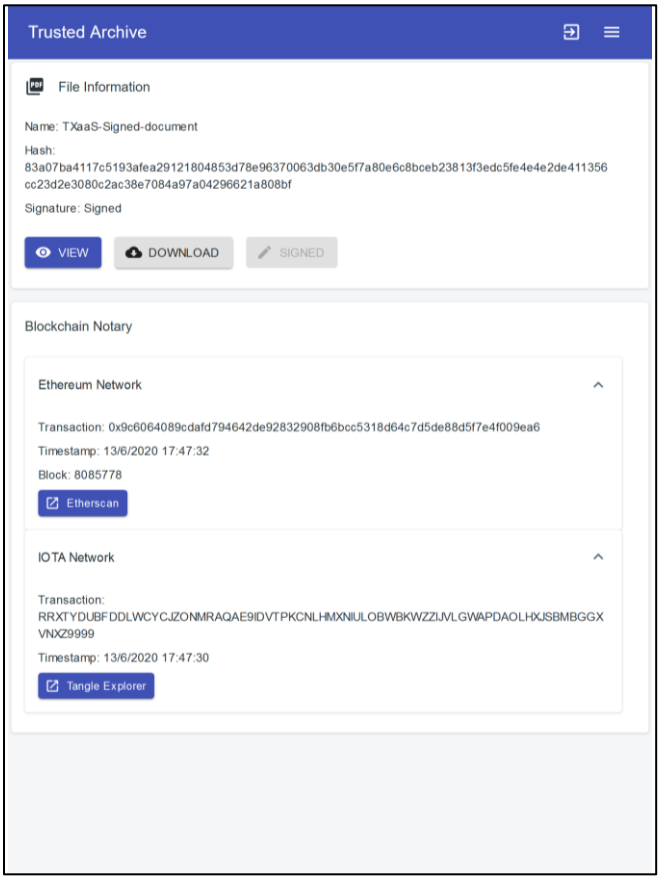




Annex iPad/Tablet







Bibliography

[1] J. Allin and N. Pope, "eIDAS Regulation," 2017.

[2] E-government Portal, "eIDAS," [Online]. Available: https://administracionelectronica.gob.es/ctt/eidas. [Accessed 19 February 2020].

[3] Wikipedia, "EIDAS," [Online]. Available: https://en.wikipedia.org/wiki/Electronic_identification. [Accessed 3 June 2020].

[4] Safelayer, "TrustedX eIDAS Platform," [Online]. Available: https://www.safelayer.com/es/productos/trustedx-eidas-platform. [Accessed 19 February 2020].

[5] Wikipedia, "PKI," [Online]. Available: https://en.wikipedia.org/wiki/Public_key_infrastructure. [Accessed 20 June 2020].

[6] World Wide Web Consortium (W3C), "Web services architecture," [Online]. Available: https://www.w3.org/TR/ws-arch/#whatis.

[7] Lifewire, "What exactly is a web application?," [Online]. Available: https://www.lifewire.com/what-is-a-web-application-3486637.

[8] Wikipedia, "Distributed Ledger Technology," [Online]. Available: https://en.wikipedia.org/wiki/Distributed_ledger. [Accessed 16 June 2020].

[9] Wikipedia, "Blockchain," [Online]. Available: https://en.wikipedia.org/wiki/Blockchain. [Accessed 16 June 2020].

[10] M. Méndez, New Innovations in eIDAS-compliant Trust Services: Anomaly detection on Log data, June 2020.

[11] A. Bernal, New Innovations in eIDAS-compliant Trust Services: Cloud and API Management, June 2020.

[12] Splunk, "Documentation - Splunk documentation," [Online]. Available: https://docs.splunk.com/Documentation. [Accessed 12 February 2020].

[13] D. Hardt, "The OAuth 2.0 Authorization Framework," RFC 6749, IETF, October 2012.

[14] N. Sakimura, J. Bradley, M. Jones, B. de Medeiros and C. Mortimore, "OpenID Connect Core 1.0," Febrary 2014.

[15] S. Cantor, "Profiles for the OASIS Security Assertion Markup Language (SAML) V2.0," OASIS SSTC, March 2005.

[16] J. Jonsson and B. Kaliski, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography. Specifications Version 2.1," RFC 3447, IETF, February 2003.

[17] F. Jordan, H. Pujol and D. Ruana, "Achieving the eIDAS Vision Through the Mobile, Social and Cloud Triad," Springer Vieweg, Wiesbaden, October 2014.

[18] Amazon, "AWS S3," [Online]. Available: https://aws.amazon.com/es/s3/. [Accessed 3 June 2020].

[19] Binance Academy, "Blockchain," [Online]. Available: https://academy.binance.com/blockchain/. [Accessed 5 June 2020].

[20] Bitcoin Wiki, "Timestamp," [Online]. Available: https://en.bitcoin.it/wiki/Block_timestamp. [Accessed 5 June 2020].

[21] J. Maldonado, "¿Qué es timestamp? Cronometrando el funcionamiento de la blockchain," [Online]. Available: https://es.cointelegraph.com/explained/what-is-timestamp-timing-the-operation-of-the-blockchain. [Accessed 5 June 2020].

[22] Wikipedia, "Prisoner dilemma," [Online]. Available: https://en.wikipedia.org/wiki/Prisoner%27s_dilemma. [Accessed 5 June 2020].

[23] Bitcoin Wiki, "Proof of work," [Online]. Available: https://en.bitcoin.it/wiki/Proof_of_work. [Accessed 5 June 2020].

[24] S. Nakamoto, "Bitcoin: A Peer-to-Peer Eletronic Cash System," [Online]. Available: https://bitcoin.org/bitcoin.pdf. [Accessed 5 June 2020].

[25] A. Tar, "¿Qué es Prueba de trabajo o Proof of Work (PoW)?," [Online]. Available: https://es.cointelegraph.com/explained/proof-of-work-explained. [Accessed 5 June 2020].

[26] Ledger, "Proof of stake," [Online]. Available: https://www.ledger.com/academy/blockchain/what-is-proof-of-stake. [Accessed 5 June 2020].

[27] Wikipedia, "Fork Blockchain," [Online]. Available: https://en.wikipedia.org/wiki/Fork_(blockchain). [Accessed 5 June 2020].

[28] Bitcoin Wiki, "OP RETURN," [Online]. Available: https://en.bitcoin.it/wiki/OP_RETURN. [Accessed 5 June 2020].

[29] P. Kasireddy, "How does Ethereum work, anyway?," [Online]. Available: https://medium.com/@preethikasireddy/how-does-ethereum-work-anyway-22d1df506369. [Accessed 5 June 2020].

[30] Bit2me, "Smart Contracts," [Online]. Available: https://academy.bit2me.com/que-son-los-smart-contracts/. [Accessed 5 June 2020].

[31] Bit2me, "EOS (EOS)," [Online]. Available: https://academy.bit2me.com/que-es-eos-criptomoneda/. [Accessed 5 June 2020].

[32] IOTA foundation, "IOTA," [Online]. Available: https://www.iota.org/. [Accessed 10 June 2020].

[33] Wikipedia, "IOTA tecnhology," [Online]. Available: https://en.wikipedia.org/wiki/IOTA_(technology). [Accessed 5 June 2020].

[34] Google Apigee, "API Management Platform," [Online]. Available: https://cloud.google.com/apigee and documentation https://docs.apigee.com/ .

[35] Redhat, "DevOp," [Online]. Available: https://www.redhat.com/es/topics/devops. [Accessed 7 Marc 2020].

[36] Tecnoempleo, "Informe Empleo Informática," [Online]. Available: https://www.tecnoempleo.com/informe-empleo-informatica.php. [Accessed 6 March 2020].

[37] El País, "¿Cuántas horas se trabaja en España?," [Online]. Available: https://elpais.com/economia/2016/12/12/actualidad/1481571706_252130.html. [Accessed 15 March 2020].

[38] L. Lamport, R. Shostak and M. Pease, "The Byzantine Generals Problem," [Online]. Available: https://www.microsoft.com/en-us/research/uploads/prod/2016/12/The-Byzantine-Generals-Problem.pdf. [Accessed 5 June 2020].

[39] Wikipedia, "Game Theory," [Online]. Available: https://en.wikipedia.org/wiki/Game_theory. [Accessed 5 June 2020].

[40] G. Wood, "Ethereum : A Secure Decentralised Generalised Transaction Ledger Petersburg Version," [Online]. Available: https://ethereum.github.io/yellowpaper/paper.pdf. [Accessed 10 June 2020].