FeatureCloud

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare

Deliverable D6.4
Prototypical implementation of phase 2 and evaluation results

---

Work Package WP6
Blockchains and user right management

Disclaimer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078. Any dissemination of results reflects only the author's view and the European Commission is not responsible for any use that may be made of the information it contains.

Copyright message

Document information

Grant Agreement Number: 826078			**Acronym: FeatureCloud**	
Full title	Privacy preserving federated machine learning and blockchaining for reduced cyber risks in a world of distributed healthcare			
Topic	Toolkit for assessing and reducing cyber risks in hospitals and care centres to protect privacy/data/infrastructures			
Funding scheme	RIA - Research and Innovation action			
Start Date	1 January 2019	**Duration**	60 months	
Project URL	https://featurecloud.eu/			
EU Project Officer	Christos MARAMIS, Health and Digital Executive Agency (HaDEA) - Established by the European Commission, Unit HaDEA.A.3 – Health Research			
Project Coordinator	Jan BAUMBACH, UNIVERSITY OF HAMBURG (UHAM)			
Deliverable	D6.4 Prototypical implementation of phase 2 and evaluation results			
Work Package	WP6 Blockchains and user right management			
Date of Delivery	**Contractual**	30/06/2022 (M42)	**Actual**	30/06/2022 (M42)
Nature	Demonstrator	**Dissemination Level**	Public	
Lead Beneficiary	5 SBA			
Responsible Author(s)	Fenghong Zhang (SBA) Walid Fdhila (SBA) Nicholas Stifter (SBA) Aljosha Judemayer (SBA)			
Keywords	Federated Machine Learning, Blockchain Technology, Consent Management, Healthcare			

Table of Content

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

1 Objectives of the deliverable based on the Description of Action (DoA)

The objective of this deliverable is based on the description of action, which incorporates mostly aspects from Objective 1 and 2 and thus are part of the corresponding tasks 1 and 2 in work package 6. The corresponding tasks in this work package are partly described in Task 1:
“Based on this academic research SBA will construct a prototype that can be verified and tested with actual information. Required changes will thereby be fed back into the construction phase.”, and in Task 2:
“This will be done by analyzing the actual information and meta-information required (MUG, TUM) and designing mechanisms for introducing user-rights into smart contracts based on our blockchain mechanism (SBA).”

2 Executive Summary

Deliverable D6.4 builds upon the legal and technical requirement analysis conducted and described in deliverable D6.3, and the threat model developed in D6.2, to provide a proof-of-concept prototype for managing patient consents and securing audit processes in federated machine learning of healthcare data. The design relies on a permissioned blockchain solution, i.e., Hyperledger Fabric, where a test net has been locally deployed using a set of local nodes acting as the root of trust. The design also supports both tech-savvy (can use IT infrastructure) and non-tech-savvy patients (only use paper consents and wet signatures) to prevent discrimination. Multiple smart contracts (chain codes) have been deployed for the purpose of managing both consents and commitments of the necessary data related to the executed ML studies.
Additionally, and while not being the main focus of FeatureCloud, a discussion on how the project could benefit from Self-sovereign identity (SSI), and most particularly decentralized identifiers (DiDs) and verifiable credentials (VCs) is provided.

3 Introduction (Challenge)

When training machine learning (ML) models in the healthcare domain, it is often the case that the required data is distributed over various hospitals eventually located in different jurisdictions, and needs to be globally accessible for training, e.g., in a central cloud. This challenge is further compounded by the increasing legal (e.g., GDPR) and technical demands on data providers.

Unlike existing approaches, where the ML algorithm runs on data aggregated from different healthcare providers, federated learning could help address these legal and privacy issues by enabling an ML algorithm to be executed locally at each data provider's site, and the output trained models are subsequently collected and aggregated. However, because data is kept and processed locally, external parties cannot readily ascertain from the output results that the learning process was carried out correctly. Therefore, mechanisms should be put in place in order to minimize malicious behavior, where for example, a hospital utilizes non consented data, claims more data than they actually have, or creates fake identities and healthcare data to bias a study.
As such, some form of manual or automatic auditing process is required to verify the integrity of the results and that no data manipulation was carried out during the learning process. To address this problem, prior art has both, considered Byzantine resistant aggregation of ML outputs that rely on advanced cryptographic schemes such as MPC, e.g. (He et al., 2020), and proposed approaches that seek to render the process more accountable, e.g. through use of blockchain (Mugunthan et al., 2020; Passerat-Palmbach et al., 2019; Weng et al., 2019).

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

In the context of improving clinical research quality, in particular securing the auditing process, blockchain technology could offer desirable characteristics such as traceability, immutability, and integrity (Benchoufi et al., 2017) while retaining the decentralization afforded from federated learning as well as remaining largely compatible with legacy system designs. Thus, Blockchain technology may serve as an immutable audit trail used to secure federated machine learning processes, and help detect deviations, thereby creating reliable clinical studies (see deliverable D6.1). Additionally, the use of smart contracts helps enforce clinical trial protocols (such as consent management), with which participants must comply. Moreover, it becomes possible to prove the existence of study or data, while the latter remain confidential. The trusted, decentralized and chronological timestamping of healthcare data and process steps offered by blockchain also helps prevent posteriori reconstruction analysis.

4 Methodology

The development of the proof-of-concept prototype for securing the audit process and managing consents followed an iterative and incremental method. After iterating over requirements specification and analysis, identifying possible threats of the current system design, and providing mitigations, a first PoC was developed, which was continuously improved and adapted to meet design and architectural changes.

5 Results

5.1 Federated Machine learning: Roles and workflows

5.1.1 Roles

Coordinator	● Initiates a FeatureCloud study ● Selects/Defines the consortium of participants ● Provides the ML algorithm
Participant (e.g., hospital)	● Ensures local data security & privacy ● Performs data querying & preparation ● Executes & manages federated ML studies ● Manages patient identities & consents
Patient	● Gives revokes & updates consents
Trust Authorities (e.g., Health ministry)	● Define policies & onboarding rules ● Establish & governs the trust framework ● Registers/verifies/manages patient/participant Identities
Auditor	● Perform audits, checks integrity & compliance

5.1.2 Processes for Federated Machine Learning in Healthcare

The following figures **1-3** capture the different tasks that need to be executed by each of the aforementioned actors. In particular,

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

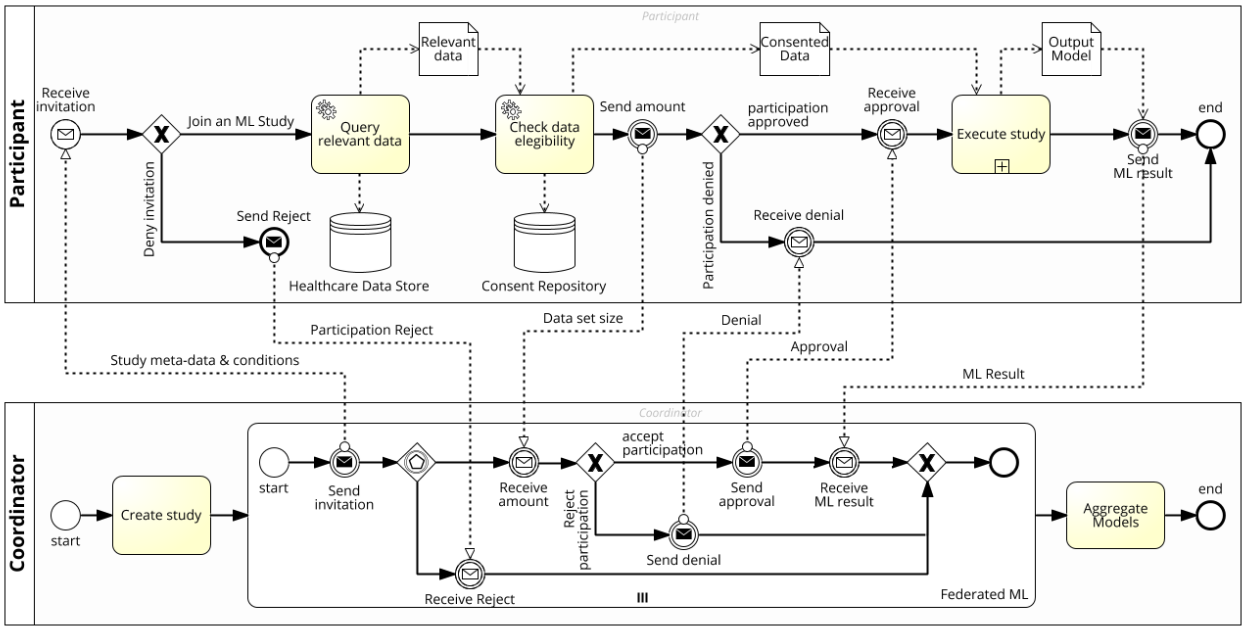

Figure 1. Collaboration diagram for Federated Machine Learning in healthcare

Figure **1** represents a collaboration diagram, which describes the interaction flow between the coordinator and a participant for a federated study, as well as the internal process logic of each of them. Upon receiving an invitation from the coordinator, a participant local project manager (LPM) will decide whether or not to join the federated study. The invitation includes meta-data about the scope of the study and participation requirements (e.g., conditions on data such as the minimum number of required records per participant). Upon acceptance to join the study, a participant should perform a data discovery check to compute the amount of consented data relevant to the study, upon which its participation in the study is either confirmed or denied. If the conditions are met, the project coordinator sends a participation approval that includes a data analysis (machine learning) application that can be executed locally by the participants (i.e., data providers). The latter are also responsible for ensuring the pre-processing pipeline (e.g., format conversion, data standardization). After collecting the ML output models from all participants, the project coordinator aggregates and eventually publishes the results.

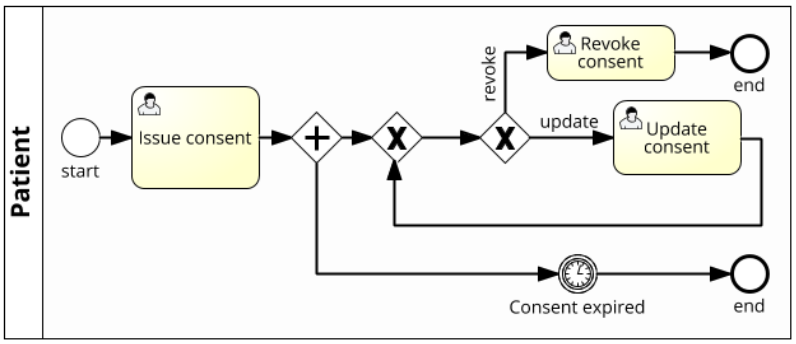

Figure 2. Consent Management process

Figure 2 depicts the patient process for specifically managing consents. A patient should be able to issue, update or revoke a consent at any time. A consent may include an expiry date, after which it

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

becomes revoked. As described in D6.2 and D6.3, a consent can be on paper, in a digital format or digitized by participants. The consent management process may either be fully automated, which requires patients to be able to use technology, or manual where a patient uses the traditional ways for giving or revoking consent (go in person to the hospital). Figure 3 describes a simplified version of the auditor process.

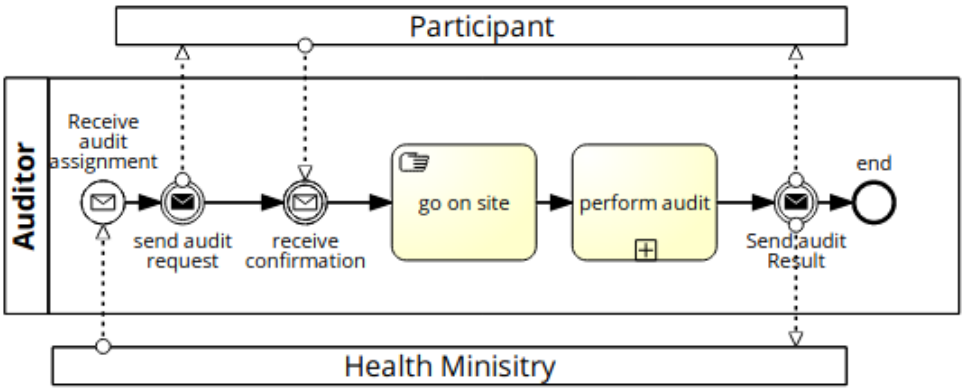

Figure 3. Simplified Audit Process

Finally, a network of trusted authorities (e.g., European actors from the healthcare sector, i.e., Health ministries) defines the governance rules for all stakeholders and enables legally binding relationships (e.g., onboarding policies, applicable regulations, governance rules, protocols).

5.2 System Design: Consent Management

In the following, we consider two types of patients in the system design:

1. Patients who are able to use electronic means/IT infrastructure for identification and consent management, e.g., patients can use applications or manage digital consents or signatures.
2. Patients who are not able or willing to use electronic means for identification or consent management, e.g., use of health insurance card and paper consents.

It is assumed that the software (e.g., a docker container) provided by the coordinator is trusted. This means that the software does not leak data and is designed to execute as reproducible as possible, i.e., ideally the execution is fully deterministic such that running it with the same inputs results in the exact same output. In the FeatureCloud AI store, the docker image of a federated algorithm (FeatureCloud app) is published by a developer, and then certified by FeatureCloud after checking its privacy and performance. Additionally, both paper and digital consents as well as digital and non-digital identities are supported to prevent discrimination.

5.2.1 Non-tech-savvy Patients

The following system design assumes that patients do not have the required knowledge to use IT infrastructure or digital signatures, or do not want to use electronic means.

Setup and assumptions

a) Each patient has a global identifier **ID_g** registered within a competent authority (e.g., insurance). **ID_g** can possibly be:

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

- The social security number SSN.

b) A consent is on paper and can be digitized and locally managed within the hospital (i.e., in a local secure database).
c) The hospital is responsible for digitizing paper consents.
d) The hospital is part of a consortium that runs a blockchain infrastructure / or has the permission to submit transactions.
e) A smart contract is deployed and governed by the consortium for registering/updating and revoking local identifiers (e.g., patient identifier within a hospital, hash of a consent).
f) The smart contract also defines the flow/rules for commitments to inputs/outputs of studies.

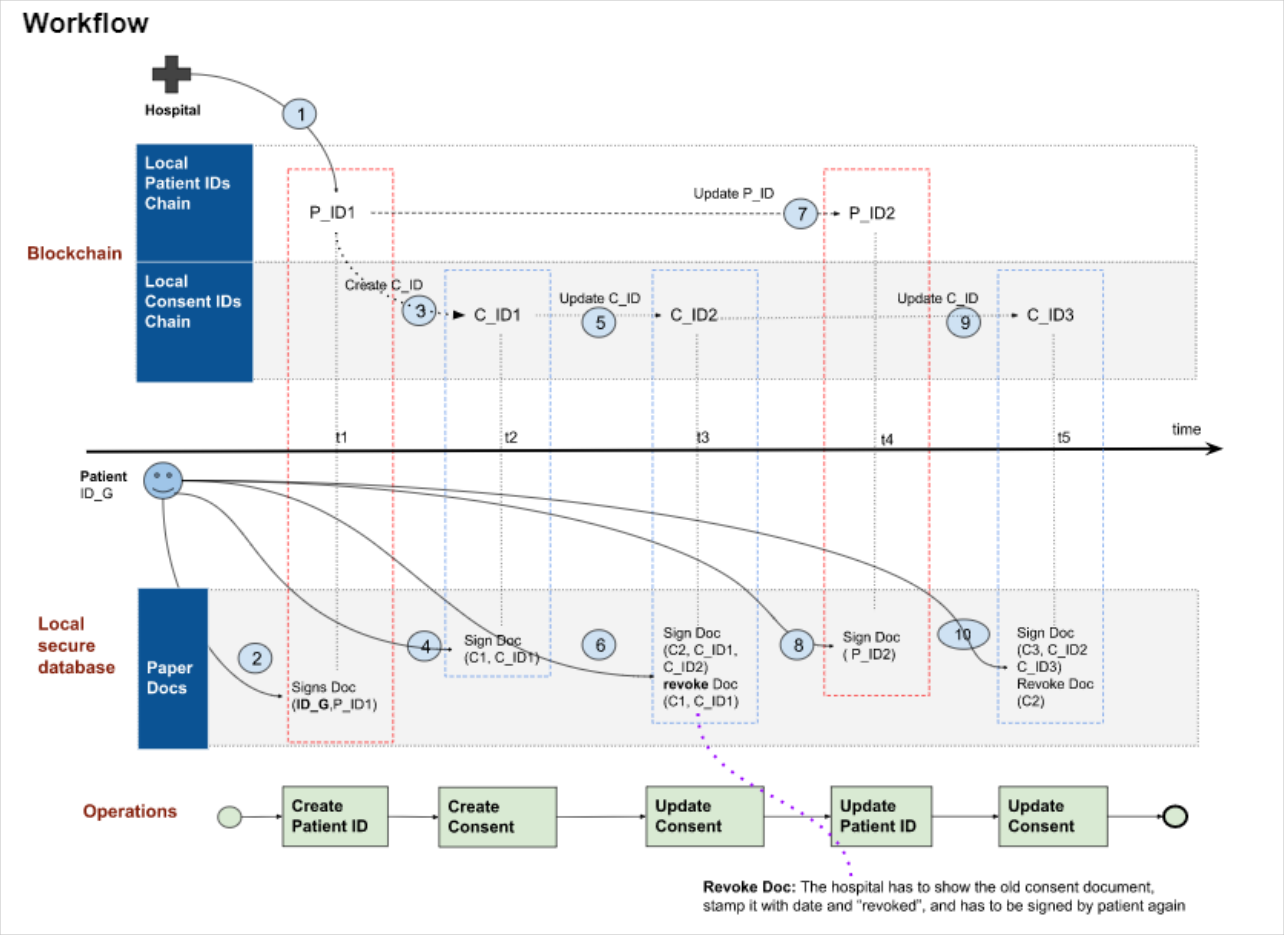

Figure 4. Consent management workflow for non-Tech-savvy Patients

Identity registration:

a) The hospital registers a new local patient identifier **PID** using the smart contract operation "Create". This identifier will be the root for all subsequent CRUD operations for that patient (Create, Register, Update, and Deactivate). As local identifiers are registered on blockchain, they must be different from ID_g (e.g., SSN) to avoid linkability across participants.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

b) The patient signs a **paper credential** containing this mapping (local ID, ID_G). This binds the PID to that actual patient. This is also important as it can serve as a proof for the auditor and prevents generating fake patient/consent identifiers. The paper signed binding is stored locally at the hospital and a copy is given to the patient.

Give Consent

a) The hospital informs the patient about eventual future research/studies and requests his consent to use their data if required. (upfront)
b) The patient agrees to give consent to use their data, e.g. for all subsequent studies on cancer research.
c) The hospital will create a consent identifier CID1, that is linked to the patient PID. This can be enforced through the smart contract logic. Link: PID--> CID1
d) The patient signs on paper the consent (includes scope, expiry date, etc), i.e., **must** include (**CID1, PID)**. The consent itself is stored locally at the hospital site.

Note: The consent identifier CID may be the hash of the consent plus a nonce.

Update Consent:

a) The patient informs the hospital that he wants to update their consent.
b) The hospital creates a new consent identifier CID2, which **must** be linked to CID1 (can be enforced through the smart contract or eventually checked by an auditor during an audit).
c) The patient signs the paper consent which **must** include (PID, CID2).
d) Using wet signatures, the hospital stamps the old paper consent (as revoked), adds CID2 to it, and has it signed by the patient. This prevents a hospital from using the old version of the consent.

Revoke Consent:

a) The hospital triggers a revoke operation on blockchain
b) using wet signatures, the hospital stamps the old paper consent (as revoked), timestamps it and has it signed by the patient.

Note: As the patient is not involved in the digital process (operations on blockchain), it is not possible for them to verify that the hospital indeed updated/revoked consent on blockchain.
However, the hospital will not be able to use the data associated with the patient for future studies, as the paper consent has to be shown to the auditor, which in turn must include the CID of the last operation on the consent before the study. The auditor can therefore check the conformance of that CID with what has been committed, by following the chain of transactions following the corresponding PID. Discrepancies between what has been committed (the chain with PID as root) and the paper consent will show that the updates were not committed.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

5.2.2 Patients who can use electronic means

In the following, we focus solely on the use case where consents are digital or digitized by participants, and patients can use applications to cryptographically sign and manage credentials.
We also assume that public representations of cryptographic keying material generated by the patients were signed by a competent authority (e.g., insurance), and that there exist one or multiple trusted registries (not necessarily a blockchain) for revoking identities and credentials.
Based on the legal and technical requirements identified in D6.3, a permissioned blockchain can serve as a distributed and immutable audit trail.
Access controls (read and write to the ledger) are also defined to restrict access to non-authorized subjects. A smart contract that is deployed and governed by a trust framework, will be used by authorized entities for registering/updating and revoking local identifiers (e.g., patient identifier within a hospital). Separate smart contracts are also deployed for managing ML studies and consents respectively (i.e., define the rules/logic for commitments).

Setup and assumptions

1. The patient has a global identifier ID_g registered within a competent authority (e.g., ministry of health, insurance). **ID_g** can possibly be:
    - A qualified electronic signature.
    - A binding decentralized Identifier DiD
    - A verifiable credential issued by a legal competent authority, which includes the patient identity (public key)
    - An X.509 certificate or Idemix (privacy preserving identity method)
2. There is a mobile or web a application that can be used by patients to sign documents
3. A consent is digital and is managed locally within the hospital (i.e., in a local secure database). Again, a consent may be:
    - A verifiable credential
    - A digital doc signed qualified electronic signature
    - Any other digital format used within the hospital infrastructure
4. The hospital is part of a consortium that runs a blockchain infrastructure / or has the permission to submit transactions.
5. A smart contract is deployed and governed by the consortium for registering/updating and revoking local identifiers (e.g., patient identifier within a hospital, hash of a consent).
6. The smart contract also defines the flow/rules for commitments to inputs/outputs of studies.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

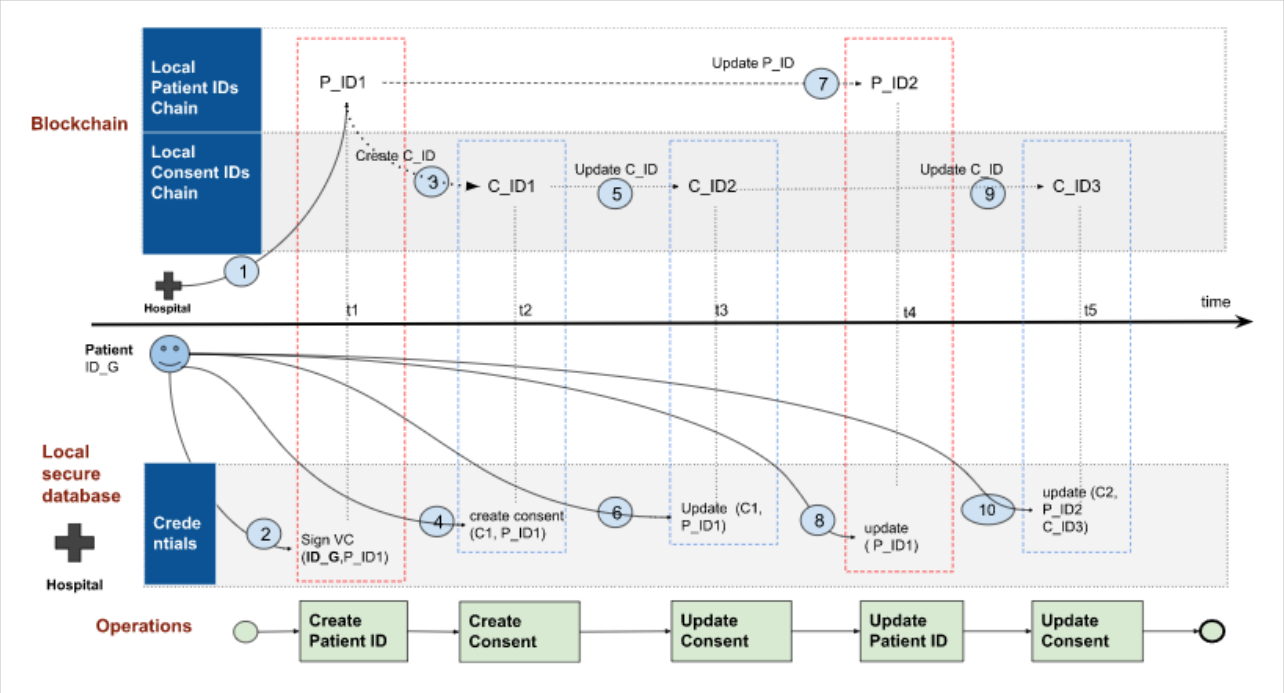

Figure 5. Consent Management Workflow for Tech-savvy Patients

Workflow

Identity registration:

1. The hospital registers a new local patient identifier **PID** using the smart contract operation "Create". This identifier will be the root for all subsequent CRUD operations for that patient.
2. The patient digitally signs this local ID using his global identifier ID_g (e.g., qualified electronic signature). This **binds** PID to that actual patient (it is possible to use a verifiable credential to store this binding). This is also important because it proves for an auditor that this PID corresponds to an actual patient and prevents generating fake patient/consent identifiers. The **signed binding** is stored **locally**.
   Note: a handwritten signature on a paper that binds that PID to the patient also works.
   Note: The patient can use a blockchain explorer to check the creation of the PID.

Give Consent:

3. The hospital informs the patient about eventual future research/studies and requests their consent to use their data if required. (upfront)
4. The patient agrees to give consent to use their data for all studies on cancer research.
5. The hospital will create a consent identifier, that is linked to that PID. This can be enforced through the smart contract logic. Link: PID--> CID1
   Note: The patient can use a blockchain explorer to check that CID1 and the link to PID.
6. The patient uses their app/wallet to sign the consent, which among other things (e.g., scope, expiry date), **must** include **CID1**. The consent itself is stored locally at the hospital site.
   Note: In case of a handwritten signature CID1 must be on the paper consent.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Update Consent:

7. The patient informs the hospital that they want to update or revoke his consent.
8. The hospital creates a new consent identifier CID2, which must be linked to CID1.
9. The patient first checks whether or not the link CID1->CID2 was validated on the blockchain.
10. The patient signs the digital consent update/revocation, which will be stored locally.
    Note: In case of a handwritten signature the CID2 must be on the paper consent

Audit Process

1. In the workflow above, it is assumed that the participant will act on behalf of the patient to update or revoke consents on the blockchain, and that patients are mainly involved in i) monitoring the correct execution of those operations on the blockchain (by having restricted read access), e.g., using a blockchain explorer, and ii) digitally signing documents, e.g., consents. However, in a more advanced setting, it is possible to envision a scenario where patients themselves have write access and can directly interact with the blockchain to manage their consents and identities, e.g., patients can themselves create and update their local identifiers and use them to sign the consents instead of using their ID_G. However, the appropriate access rules should be put in place to prevent patients from creating arbitrary non related identifiers. This may be handled using an authentication token issued by the hospital (using OPENID connect) or multi-signature scheme (patient and hospital). In both scenarios, patients are indirectly involved in the auditing process.

2. Because consents are bound to immutable local identifiers, which in turn, are anchored to a root identifier that is digitally signed by a real patient (ID_g), it becomes less possible for the hospital to create fake consents for fictitious patients. An auditor can check the data used for a study, and examine whether the corresponding consents are digitally signed by a patient registered within a competent authority. Note that this property is also ensured using the design described in D6.3 without having to create PID (also works for handwritten signatures). However, the advantage of using PID are:
    - The patient can actively check the commitments of their consents (indirectly useful for the audit)
    - In the case where they have restricted write access, patients can use wallets/apps to update, or revoke their PIDs (e.g., rotate the cryptographic material associated with the PID) by themselves. In this scenario, patients can manage consents by themselves by directly interacting with the blockchain. This gives more control to patients over their consents and identities however it also places a higher burden on them in regard to securely managing the associated keying material.

3. How to ensure that a participant (hospital) uses the most up-to-date consent for a study? When an auditor examines a data input used for a specific study, they will verify whether or not it was consented at the time of the study. Therefore, the auditor will first inspect the “consent identifier” CID in the consent that was presented by the hospital (which is embedded and signed by the patient within the consent). The auditor will verify the following things:
    - If the CID in the digital consent is somewhere on the blockchain.
    - If the CID is present, identify the root in that chain (PID)
    - Check the local signed binding as specified in step 2) of the workflow, which binds that PID to a global identifier ID_g (an actual person).
    - Compare the timestamp of the CID to that of when the study was conducted.
    -

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

5.3 Identities in the Healthcare Sector and FeatureCloud

Despite not being the main focus of FeatureCloud, identities play an essential role in ensuring authentication, authorization, non-repudiation and also integrity of learning data and consents. Indeed, without a trusted identity layer, a malicious hospital or a local project manager can always create fake identities linked to fake consents, which are in turn, linked to fake data that can be used to conduct a poison attack. The poisoned data set will therefore result in a biased output model that does not reflect the original data. Therefore, a well-established identity infrastructure will help reduce misuse of healthcare data and give patients more control over their records.

So far, different forms of identity have been used in the healthcare system (Soenens, 2009), which range from centralized to federated identities and from paper certificates (e.g., e-card) to more advanced digital formats (e.g., qualified electronic signatures or X.509 certificates). European initiatives such as eIDAS[1] already defined a regulatory framework for trusted electronic identification and authentication across European member states. However, in practice, it remains optimistic to assume that all participants will use the same identity scheme. In the following, we will specifically investigate how FeatureCloud could benefit from an emerging technology namely, self-sovereign identity and how decentralized identifiers (DiDs) and verifiable credentials (VCs) could be used in the context of FeatureCloud.

5.3.1 Self-sovereign Identity for FeatureCloud

So far, FeatureCloud relies on the assumption that identities of all actors involved in such collaborative settings are well defined (such as global identifier ID_g or patient local identifier PID), and that there exists an established public key infrastructure (PKI) that enables trustworthy communications and authentication. However, dealing with sensitive data such as in healthcare requires a particular level of protection against data misuse by providing mechanisms and means that improve their privacy and security. As such, an elaborate design of the identity layer seems crucial to establish a trusted infrastructure that secures and links public keys to real identities. Recent developments in this area such as certificate transparency already embrace authenticated data structures as a means of identifying manipulation attempts. For example, electronic health certificates have increasingly been relied upon in an effort to contain the spread of COVID-19 infection. Hereby, the difficulty lies in striking a balance between ensuring an individual's privacy and the ability to verify that the claims made in the certificate are authentic and tied to a particular identity.

Unlike previous identity systems, which used to rely on centralized or federated architectures that have already proven their inefficiency and lack of security and privacy, a new identity approach, i.e., namely self-sovereign identity (SSI), has emerged, which promises users control over their data, and ensures individuals are at the center of interactions. SSI often relies on blockchain technologies to record identity information or serve as the basis for decentralized public key infrastructures. Such Blockchain-based PKI infrastructure could, for instance, help provide more robust mechanisms for establishing (and revoking) digital identities that are used for aspects such as access control or rights management.

SSI initiatives resulted in two key concepts i) the verifiable credentials (VC) standard, and ii) the decentralized identifiers (DiDs) recommendation draft by the World Wide Web Consortium (W3C).

- Decentralized Identifiers (DiDs): A DID is a globally unique identifier, cryptographically generated and often registered on a distributed ledger, that points to a DID document (e.g. a JSON-LD object), which specifies cryptographic key material (e.g., public keys for authentication or authorization), verification methods essential for proving ownership of the

[1] https://digital-strategy.ec.europa.eu/en/policies/eidas-regulation

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

DID and eventually services endpoints for trustworthy and persistent communication channels. DIDs are strings whose formats should comply with the W3C specification[2] as follows:

did:<did_method>:<method_specific_identifier>.

For example, did:indy:sovrin:7Tqg6BwSSWapxgUDm9KKgg is a DiD created using the Hyperledger Indy DiD method and registered on the sovrin network [3]. The DID subject is the entity that is identified by the DID, and can be a person, an object or an organization. A DID method, on the other hand, is a specification that defines how a DID can be created, resolved, updated and revoked (i.e., CRUD operations). Currently there exist over a hundred different methods for creating and resolving DIDs, which rely on different blockchain solutions and technologies (e.g., IPFS). Although most of these methods comply with the W3C DID specification, each of them comes with specific properties and offers different guarantees depending on the underlying technology or governance framework (Fdhila et al., 2021; Ghesmati et al., 2021). While DIDs in conjunction with DID documents enable creating trustworthy communications (how to communicate with identity owners), verifiable credentials (VCs) represent information and claims about identity owners (e.g., name, age).

- Verifiable Credentials (VCs): Verifiable credentials are tamper-evident identity attributes and assertions about a specific subject issued by an identity provider. In contrast to other types of digital credentials, a relying party (third party service) can check the validity of a VC without having to interact with the issuer (i.e., preventing correlation). In the context of FC, a verifiable credential may represent a consent issued by an accredited patient to a hospital.

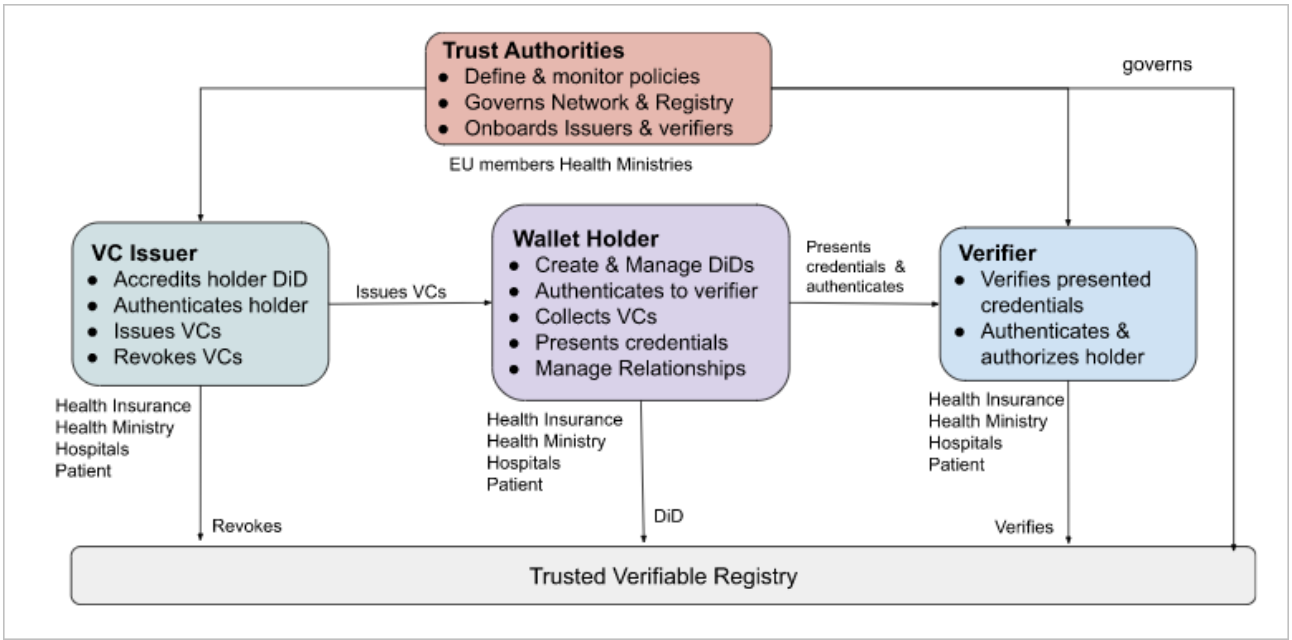

Figure 6. SSI actors and responsibilities

Figure 6 describes a possible identity infrastructure for FeatureCloud that relies on verifiable credentials and decentralized identifiers. The issuer is responsible for asserting claims about a

[2] https://www.w3.org/TR/did-core
[3] https://sovrin.org/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

subject in a verifiable credential that is issued to a holder. A VC is signed by the issuer and may be bound to a decentralized identifier controlled by the holder. In FeatureCloud, the issuer can be the patient, and the VC may represent a consent to use healthcare data for studies. Additionally, the issuer can be the health insurance, and the VC may represent the patient identity. Alternatively, a patient DiD accredit by the health insurance may be used instead. It is also possible to model local identifiers as DiDs created by hospitals directly and given to patients, or as DiDs created by patients themselves and accredited by the hospital (i.e., registration phase).
Using a DiD/VC wallet, a holder can manage DiDs and VCs, and can present credentials to verifiers, without having the latter interacting with the issuers (or centralized registries) to verify the authenticity of the VC, i.e., by relying on a decentralized verifiable registry (e.g., blockchain).
A verifier is any entity that needs to verify a credential presented by a holder (e.g., integrity, authenticity) in order to authorize the latter performing actions (e.g., access to a service). In the context of FC, a verifier may be a participant or an auditor that checks patient identities or consents, or a coordinator, which checks participants identities. It should be noted that the same actor may act as issuer and verifier at the same time. Finally, for all this to work, a trust framework should be established (e.g., European actors from the healthcare sector, i.e., Health ministries) to define what issuers will issue what identity or credentials under which policies.

In the context of FeatureCloud, a binding identity design is necessary to prevent malicious participants from biasing output ML models through poison attacks, i.e., using fake identities, consents or data. This means that identities need to be accredited by members of a trust framework. Furthermore, a privacy-preserving identity model would avoid cross-correlating patient data across multiple participants.
Unlike other alternatives such as X.509 certificates or qualified digital signatures, self-sovereign identities are i) decentralized, ii) more privacy-preserving (e.g., prevents linkability and supports zero-knowledge proofs), and iii) more secure (e.g., no single point of failure).
Additionally, in a report by the European Union Agency for Cybersecurity (ENISA, 2022), an assessment of the potential of SSI technologies and other eID solutions for ensuring secure electronic identification and authentication to access cross-border online services under the eIDAS Regulation was provided. It was acknowledged that SSI provides an effective basis for digital identities, which protects the privacy of personal data, but also needs to co-existent/operate with established technologies such as X.509 PKI, OpenID Connect and other identification schemes.

With respect to the previous workflow, in which patients can use IT infrastructure, the use of DiDs give patients more control over both their identities and consents. In particular, if the patients do use DiD/VC wallets and are able to interact with blockchain (i.e., hospitals do not act on behalf of patients). Thus, patients become responsible for the CRUD operations, and updates or revocations are handled in time (i.e., avoid the lengthy manual processing of update requests). Additionally, as the registry (e.g., blockchain) is not maintained by one single participant, but collectively by multiple nodes, the probability of the system failure, loss of data or tampering with the data is minimized.
Furthermore, consents can be formatted as verifiable credentials in a JSON-LD format. This has the advantage of:

- Using context in JSON-LD allows having an unambiguous definition or interpretation of the terms used within the consent.
- ZKP (e.g., BBS+ signatures) can be used for VC to generate proofs about only a selection of claims within the same credential to a verifier, i.e., selective disclosure.
- Prevent linkability: a verifiable credential contains the signature of the issuer. Thus, a presentation using bbs+ contains a derived proof which corresponds to the proof of signature (proof of knowledge of the signature). Otherwise, the signature would be a point of correlation.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

- Signature Blinding: this allows the issuer's signature, which is a unique value and therefore a correlating factor, to be randomized before it is shared with a verifier (e.g., auditor) [4].
- Private Holder Binding: this allows a credential to be bound to a holder without creating a correlating factor for the holder that needs to be revealed upon presentation.
- Predicates, which allow hidden values to be used in operations with a value provided by the verifier.

Using verifiable credentials in combination with BBS+ may help participants prove to auditors that consents were issued by a patient (accredited by a trusted authority, e.g., insurance company) for a specific data without showing the identity of the patient. Also, using DiDs, a patient can more easily rotate the keying material associated with the DiD. During DiD registrations, DiDs can be accredited by a trusted authority.

5.4 Prototype Architectural Design

5.4.1 Network Architecture

The FeatureCloud network is composed of the different actors involved in the federated machine learning but with different access rights and responsibilities. More importantly, the trust framework, i.e., the network of trusted authorities TA defines the governance rules for all stakeholders and enables legally binding relationships (e.g., onboarding policies, applicable regulations, governance rules, protocols). Members of the trust framework are also responsible for onboarding new members to FeatureCloud (e.g., hospitals, pharmaceutical companies, test labs, insurances) with specific access rights (e.g., writing to the ledger), and monitoring their compliance with the rules. All accredited members in this trust framework may act as both validator nodes and identity providers. Every node which joins the system keeps a white list of trusted nodes, which means that any participant that wants to join the FeatureCloud network must be approved by the FeatureCloud authority (Root CA). Additionally, each node within the FeatureCloud network keeps a copy of the ledger. Moreover, once a node propagates a new transaction to the FC network, the "majority endorsement" policy is used.

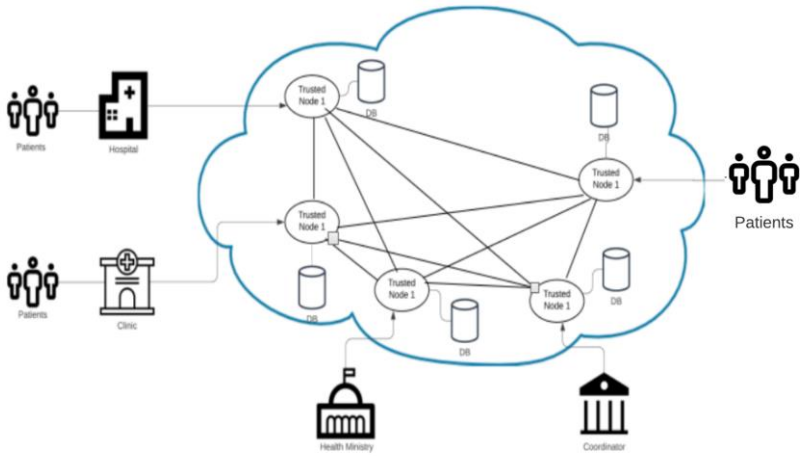

Figure 7. FeatureCloud Network

[4] https://www.evernym.com/blog/bbs-verifiable-credentials/

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

5.4.2 FeatureCloud Entities

A. Consent

A consent represents an agreement to use healthcare data records of a patient for a specific or all studies. The consent is issued by the patient (digitally or using a handwritten signature) that provides their medical data. It is also possible that a trusted-third-party issues a digitally signed consent on behalf of the patient.
Consents are not stored on blockchain but locally at each participant storage. However, they are committed to blockchain in order to ensure traceability. Consent commitments are managed by participants on behalf of patients, and by patients in case they have to use IT infrastructure. Patients may also monitor operations on their consents directly on blockchain. Consents are also tied to patient identities. the information that is stored on blockchain comprise:

1. a unique ID (CID)
2. patient ID (PID)
3. date of creation
4. data hash of the consent' content

B. ML Study

A machine learning study is a software provided by a coordinator, and which needs to be executed locally by each participant using their respective healthcare data. Each ML study has a unique identifier ID, and meta data that describes it. Only the hash of the ML study and its metadata is committed to blockchain.

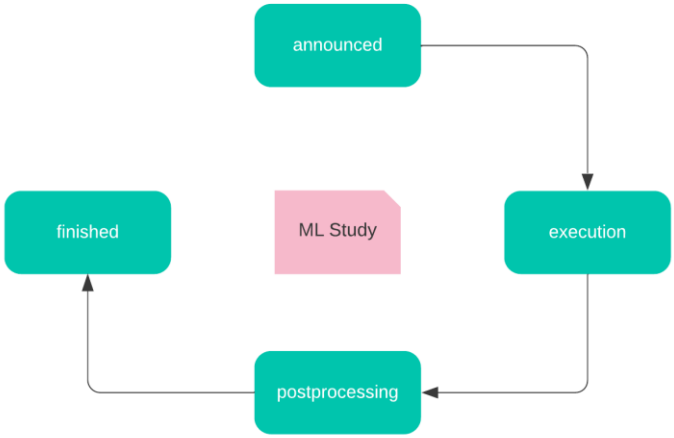

Figure 8. ML Study Lifecycle

C. ML Study Result

After collecting eligible healthcare data, (i.e., responds to the study requirements and has patient consent), a participant will prepare the data (data transformations may be required), then execute the study. The result will be the ML output model, which will need to be aggregated with other participant models.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

The information that needs to be committed to the blockchain includes all data inputs, patient consents employed for the study, possible data transformation functions, the ML study ID, and the output model. Only the hash of such information will be committed to blockchain.

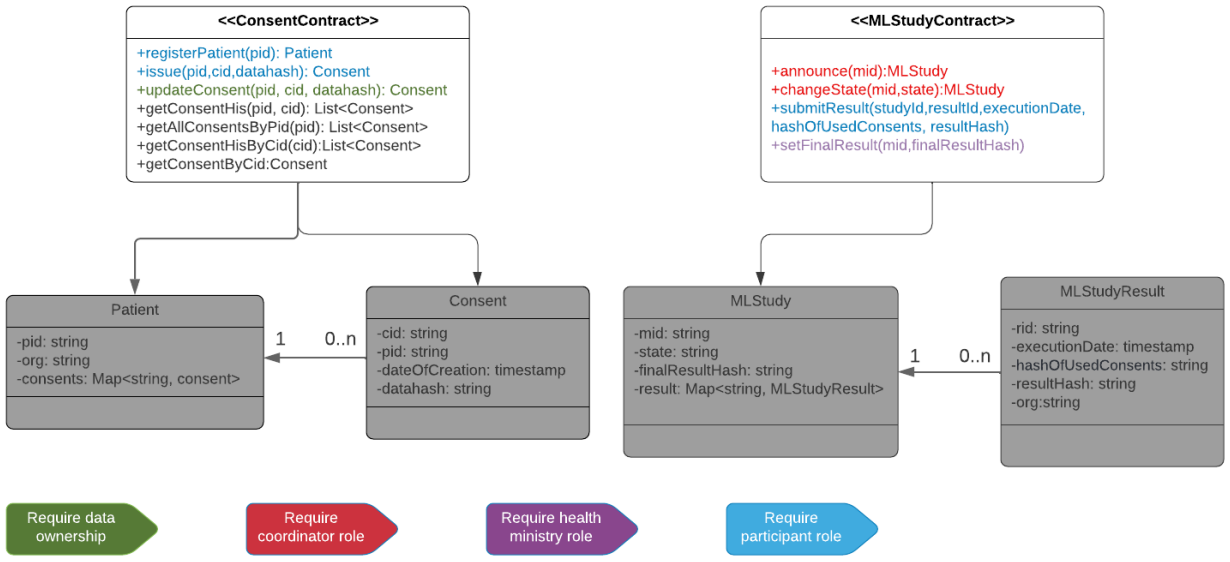

Figure 9. Contracts and Objects of FC Network

5.4.3 Smart Contracts

Each node in the FC blockchain network has its own database, which holds the current and historical state of the entities Consent, MLStudy and MLStudyResult. A smart contract therefore, defines the executable rules and logic for changing their states.
The smart contracts are executed by every node in the network. The execution of a method in a smart contract will either trigger a transaction or return a historical or current state of a system entity. Only if the majority of the system nodes produce the identical transaction result or return the same state of an entity makes the execution valid.

A. ConsentContract

A ConsentContract is a smart contract, which manages patient consents for using their healthcare data. Figure 9 describes the functions defined in this smart contract which implement the workflows presented in the previous sections. For example, Function "registerPatient" creates a new local identifier for the patient (PID), while function "issue" enables a participant (or eventually a patient) creates a new consent commitment and propagates it to the FC network. The latter can be updated by calling the function "updateConsent". Access writes for invoking these operations are defined, which means only actors with the appropriate rights can call a function.
Most functions require data ownership, which means only the provider of the data is allowed to call them. In the case where patients can use IT infrastructure, also patients (data owners) may acquire the rights to call those functions. The functions that require ownership are marked in green. The functions marked in blue require the participant role.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Furthermore, "getConsentHis", "getAllConsentsByPid", “getConsentHisByCid”, "getLastConsentBeforeExecution" and "getConsentByCid" help the competent authority, e.g., for example, function "getLastConsentBeforeExecution" in ConsentContract returns the last data hash of this consent before study execution. By locally comparing the hash of the actual consent used for a study with the last consent commitment before the study (using the timestamp), the auditor can detect a misuse, e.gThe wrong consent version was used.

B. MLStudyContract

Whereas the ConsentContract handles consent management, MLStudyContract manages the machine learning study result submitted by participants. This smart contract provides two functions that trigger the write action in the node's database, and three read functions that deliver the ML Study Results. "submitResult" is called by machine learning study participants. By calling it, a participant submits the execution result of a study, the following parameters must be submitted at the same time:

1. execution time
2. machine-learning study id
3. a hash of consents' ID
4. a hash of the execution result

The hash of consents' ID related to consent of the medical data and the execution time facilitate the audit process. Both hashes can be combined in one single hash.
Furthermore, this MLStudyContract provides the functions to quickly search all submitted ML Study Results by ML study ID. Auditors can submit the final result by calling "setFinalResult".

5.4.4 User Activities

Figure 10 illustrates the user activities as an activity diagram, which associates the tasks with the corresponding function calls of the smart contracts. Details about the processes are described in Section 5.1 and also in deliverable D6.2 and D6.3.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

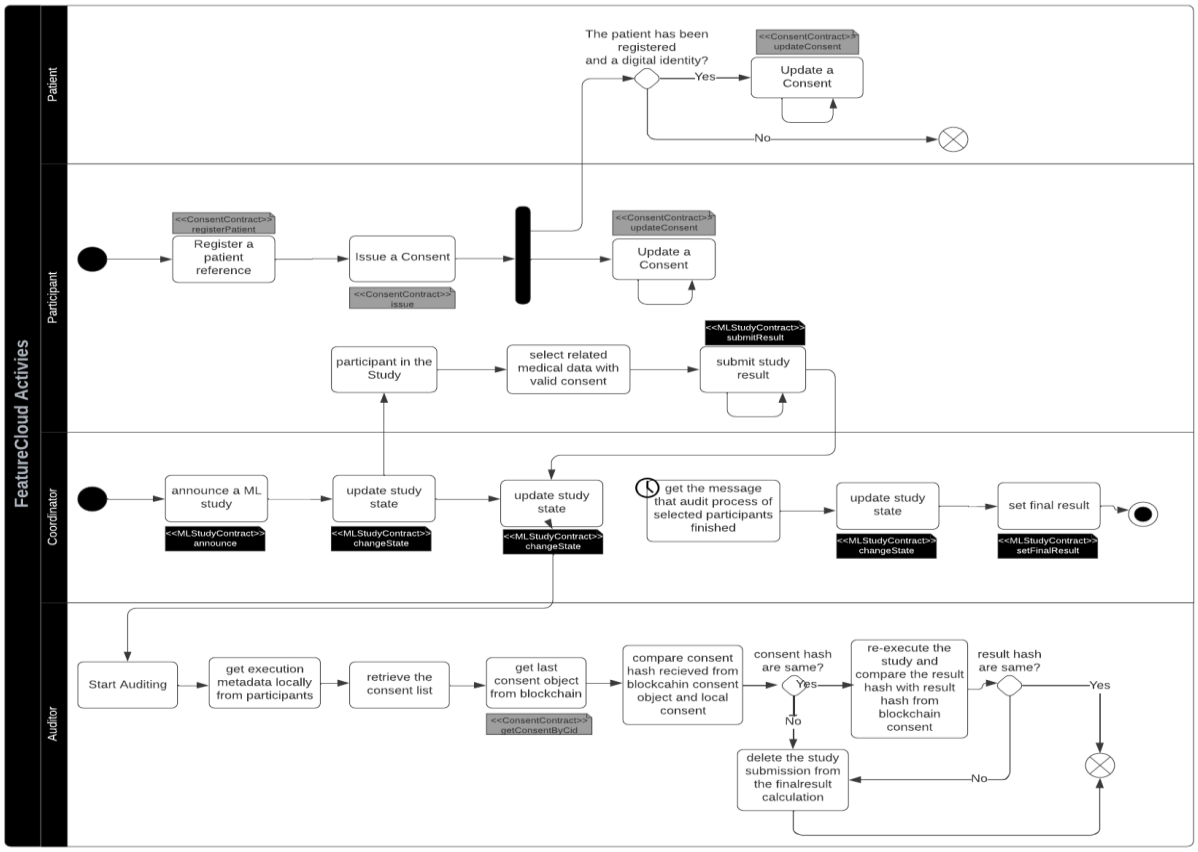

Figure 10. FeatureCloud Activities

5.5 Hyperledger Fabric-based solution

Based on both legal and technical requirements gathered and analyzed in deliverable D6.3, and given the sensitivity of the information handled in the context of Healthcare in general, and FeatureCloud in particular, we opted for a permissioned blockchain setting to handle commit operations (also see D6.1 and D6.2) and consent management. Besides better scalability and lesser costs compared to a public blockchain such as Bitcoin or Ethereum, a private permissioned blockchain offers more privacy and better access control parametrization.

More specifically, we have selected Hyperledger Fabric[5] as a distributed and immutable audit trail. The blockchain is also not public, which means that only authorized entities will have read access to restricted views on data. Note that Identities of all entities of FeatureCloud are legally binding. Other key features of Hyperledger fabric are [6]:

- Highly modular
- Support of privacy features using channels, private data collections and data isolation
- support of EVM and solidity as well as other programming languages, e.g., Go, java
- support of multiple consensus mechanisms
- low latency of finality/confirmation

[5] https://www.hyperledger.org/use/fabric
[6] https://www.hyperledger.org/wp-content/uploads/2020/03/hyperledger_fabric_whitepaper.pdf

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

5.5.1 Key Configurations

Digital Certificates

Hyperledger Fabric currently inherently supports two types of identity: i) digital certificates compliant with the X.509 standard and ii) idemix, e.g., privacy-preserving identities.
In the following, we use X.509 certificates, but the prototype can easily be extended to any other type of identity such as decentralized identifiers DiDs (e.g., did:indy) or verifiable credentials Vcs.
A digital certificate is a document that contains identifying details about an entity, e.g., keying material, issuer, subject and other attributes. A digital certificate can be used to prove ownership of an identity as long as the issuer is trusted (a certificate authority). It is also used to establish secure communication with other parties. The cryptographic material defined in the certificates is also used to digitally sign and verify the authenticity of transactions. Hyperledger Fabric provides a built-in CA component, namely Fabric CA, which is a private root CA provider capable of managing X.509 certificates.

Certificate Authority (CA)

A certificate authority (CA) is a trusted authority that issues digital certificates to different actors. In the context of FeatureCloud, the Root CAs correspond to the legal authorities responsible for accrediting and onboarding new members to the network, e.g., health ministries. Other certificate authorities can be onboarded (e.g., insurance companies), forming a chain of trust (trust framework). The trust framework (e.g., European actors from the healthcare sector, i.e., Health ministries) defines what issuers will issue what identity or credentials under which policies. This network of trusted authorities TA defines the governance rules for all stakeholders and enables legally binding relationships (e.g., onboarding policies, applicable regulations, governance rules, protocols). Members of the trust framework are also responsible for onboarding new members to FeatureCloud (e.g., hospitals, pharma companies, test labs, insurances) with specific access rights (e.g., writing to the ledger), and monitoring their compliance with the rules. All accredited members in this trust framework may act as both validator nodes and identity providers.

Access Control List (ACL)

Access control lists (ACLs) can be used to manage access to resources by associating a policy with a resource, e.g., chaincode. Different types of policies can be defined for different purposes such as channel configuration or administration. In the context of FeatureCloud, ACLs specify access controls for each of the actors, e.g., Hospitals, patients (in case they can use IT infrastructure) or insurance companies. This also enables certain roles to access predefined functions in a smart contract.

Chaincode endorsement policies

Hyperledger Fabric uses chaincode to define an executable business logic and manages the ledger state through transactions. Chaincode endorsement policies define the rules on the number of peers required to execute and validate a transaction. By default, the "Majority Endorsement" is used. It means that a majority of the peers need to execute and validate a transaction. We use the default setting. Thus, it allows a newly joined organization in the channel to become automatically added to the chaincode endorsement policy. Fabric also gives the possibility to define customized endorsement policies.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

Orderer nodes

Unlike many other blockchains, which rely on probabilistic consensus algorithms, Hyperledger Fabric uses ordering nodes to guarantee deterministic consensus algorithms. The transaction order is organized by ordering nodes and is added deterministically as a block to the chain. Fabric provides Raft, Kafka, and Sole for the ordering service implementation. We decided to use Raft as our ordering service.

Ledger

According to the design described above, our database saves simple key-value pairs. By default, Hyperledger Fabric has LevelDB, which is indeed quite suitable for the simple key-value data structure design. Besides, Hyperledger Fabric also provides the possibility to use CouchDB. CouchDB is particularly appropriate if ledger states should be structured as JSON documents. So we decided to use CouchDB to keep the data as JSON documents.

Channel

A channel represents a separate ledger that involves a specific set of organizations, and which is not visible to any other organization outside of this network. Multiple channels can be defined within the same network.

Chaincode Implementation

One advantage of Hyperledger Fabric is that it supports different languages of chaincode implementation, e.g., Java, Go, or Node JS. Java has many advantages, like platform-independent and object-oriented. For this PoC we use java to implement the chaincodes.

5.5.2 Overall Architecture

After adapting the design with the inherent characteristics of Hyperledger Fabric we construct our prototype design as shown in the following Figure 11.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

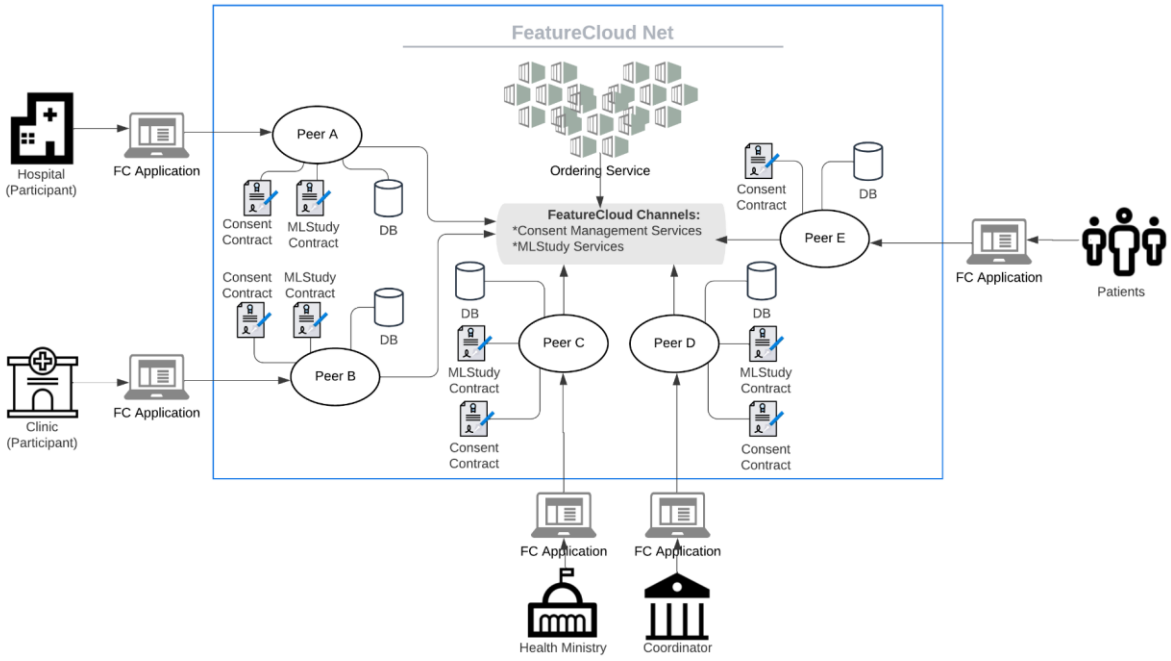

Figure 11. Architectural Design using Hyperledger Fabric

5.5.3 Testing and Evaluation

In this section, we demonstrate a test network based on our implementation of chain code and the customized configurations. We verified if the test network basically catches the necessary functional requirements of FeatureCloud.

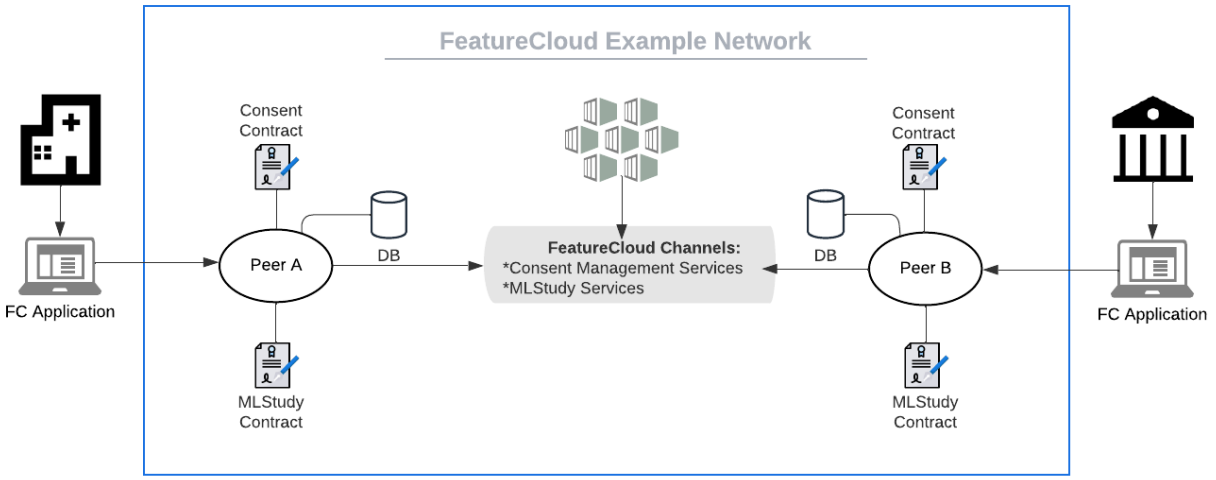

Figure 12. Example including two hospitals

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

In the test network, we deploy two different nodes and one orderer node using the docker container. Every node except the orderer node acts as one organization like a hospital or pharmaceutical company. Also, every organization node has two chain codes deployed, i.e., “ConsentContract” and “MLStudyResultContract”.

A node in the network can:

- register a patient
- issue a consent
- retrieve all consent by patient ID
- update an existing consent, which they generated
- retrieve the history of a consent by its cid
- announce a machine learning study
- change the state of a machine learning study, which they announced
- submit a result to an announced machine learning study, which has the state of execution

Prerequisites and Deployment

Prerequisites

- Hyperledger Fabric V2.x[7]
- Docker version 17.06.2-ce or greater
- Java 11 or greater

Deployment

To deploy the example network, run the script with the following command:

```
./example_start.sh
```

To start as a first participant in the network, run the script with the command below:

```
./participant_A.sh
```

To start as a second participant in the network, run the script with the command below:

```
./participant_B.sh
```

> ***NOTE:*** You can switch between participant A and participant B anytime by stopping the shell script and re-executing it or running two instances simultaneously.

Once the terminal is ready for command typing, you can get the command menu by typing

```
help
```

To bring the network down, run the following command:

```
./example_clean.sh
```

[7] https://hyperledger-fabric.readthedocs.io/en/release-2.4/install.html

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

Usage/Examples

1. Register a new patient

> Command: registerPatient [pid]

```
#example
#Organization A
registerPatient p0001
registerPatient p0002

#Organization B
registerPatient p0003
registerPatient p0004
registerPatient p0005
```

2. Issue a new consent

> Command: issue [pid] [cid] [dataHash]

```
#example
#Organization A
issue p0001 c0001 60C88968F789C29610DDC920F13D7FCAAAFDD4277AD3D14F6A08504C5FB2B491
issue p0001 c0002 BC7D57BF95AE63B359E71AEDB4897EA9EC68B6DBC62019B165A64F5BAB10BB73
issue p0002 c0003 1C833132B0BF869F823D0852DFABA2FFA55E1A8C8BB4BBB2407AEFD0B76E2DFA
issue p0002 c0004 6081CD4C19025BF436743B644F49B515A7CAABAF587CBDDFC7A66C9ED7FC8C57

#Organization B
issue p0003 c0005 C43C590964EC9F53885E8809538529F084D41BD8A45392530AC9D25864FF468E
issue p0004 c0006 95E2554A3C62F31EAF8DAA9FAAE94AA5F951A0901BF62B7D322FC581C951E55E
issue p0005 c0007 1BD178D5634FF21F6F93F5E7379CE7F4E283CE9E9607BDA4008C25B416C59BF0
issue p0005 c0008 D59C2CE54CEBBDF3BE14BD7C146146B8C69B0BBD47616D85CEFD70E75E898981
```

3. Update an existing consent

> Command: updateConsent [pid] [cid] [dataHash]

```
#example
#Organization A
updateConsent p0001 c0001 DCABB67E7A50F8BD6FADAB9B64255614AACA08C85D7D5D7C1C3F4F621C7BE0E9
```

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

4. Retrieve all consents by patient's ID

> Command: getAllConsentsByPid [pid]

```
#example
getAllConsentsByPid p0001
```

5. Retrieve the history of a consent by given its pid and cid

> Command: getConsentHis [pid] [cid]

```
#example
getConsentHis p0001 c0001
```

6. Announce a new machine learning study

> Command: announce [mid]

```
#example
#Organization A
announce m0001
```

7. Change the state of a existing machine learning study

> Command: changeState [mid] [state]

```
#example
#Organization A
changeState m0001 execution
changeState m0001 postprocessing
```

8. Submit a machine learning study result

> Command: submitResult [mid] [rid] [executionDate] [hash of consent list] [result hash]

```
#example
submitResult m0001 r0002 14.03.2022,11:28 0EC9113091A3F7B91DEA5DAAFBBC12DE8BCE39D7E4D55AA71A555D62639FB799 B30B8
```

9. Set final result of a published machine learning study

> Command: setFinalResult [mid] [result]

```
#example
#Organization A
setFinalResult m0001 9C2C46A1DB0424CDB660C3E67FE46505D6E8A02A20A00FC161A9838D1602FFC5
```

After deployment, we obtained 8 containers in the network as shown in Figure 13. Container b818d3745067 and d2238d8fb470 present “MLStudyContract” in organization 2 and organization 1’s network node, 3741567574b7 and a5e095e4b75a are “ConsentContract” on nodes. 4203cfa7893e is organization 1’s node, 83dbadd03525 is organization 2’s node and 5d32e90cf269 is orderer node.

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.

```
feng@fengi:~$ docker ps --format "table {{.ID}}\t{{.Names}}\t{{.Ports}}\t{{.Command}}"
CONTAINER ID   NAMES                                                                                  PORTS                                            COMMAND
b818d3745067   dev-peer0.org2.example.com-cp_1-a84300dc0cd9144983122fdf657ce91ce7dc3f9b043206ff884c65befbdce3ad                                            "/root/chaincode-jav…"
d2238d8fb470   dev-peer0.org1.example.com-cp_1-92e45738d8f896ad7534f9b0bc92897bb55effdf94a51f92fdf4a44fafc97edd                                            "/root/chaincode-jav…"
3741567574b7   dev-peer0.org2.example.com-cp_0-3e45721943fc40fefd5e20719ac3b3ba8f49faa4346020a3d6516f156eac0441                                            "/root/chaincode-jav…"
a5e095e4b75a   dev-peer0.org1.example.com-cp_0-5be835255af06d7d9518e7dd174125d51eea31c2a9af8e424ee2afc81c11cd06                                            "/root/chaincode-jav…"
da7d57996b14   cli                                                                                                                                         "/bin/bash"
4203cfa7893e   peer0.org1.example.com                                                                 0.0.0.0:7051->7051/tcp                           "peer node start"
83dbadd03525   peer0.org2.example.com                                                                 7051/tcp, 0.0.0.0:9051->9051/tcp                 "peer node start"
5d32e90cf269   orderer.example.com                                                                    0.0.0.0:7050->7050/tcp, 0.0.0.0:7053->7053/tcp   "orderer"
```

Figure 13 FeatureCloud Containers

Note. A cli interface for network participants to communicate with the network is provided.

The communication between the organization's nodes and the orderer node is shown in Figure 14. We installed the chain codes in both nodes of two organizations. As an example, the admin user of organization 1 calls the "issue" function in chain code "ConsentContract" through Cli App. This "Transaction A" first is sent to organization 1's and organization 2's node in the FC network and calculated by installed chain code in each node. The results calculated by nodes' chain code are sent to the orderer node for comparison. If results are identical, then the orderer will return the final verified result back to organization nodes. In the end, the result is kept consistent in both nodes' databases. If several transactions come simultaneously, the orderer node takes the responsibility to chronologically order them in blocks. To avoid the single point of failure, the orderer nodes are grouped with more than 3 members. They communicate with each other and arrange the final transaction sequence of a block.

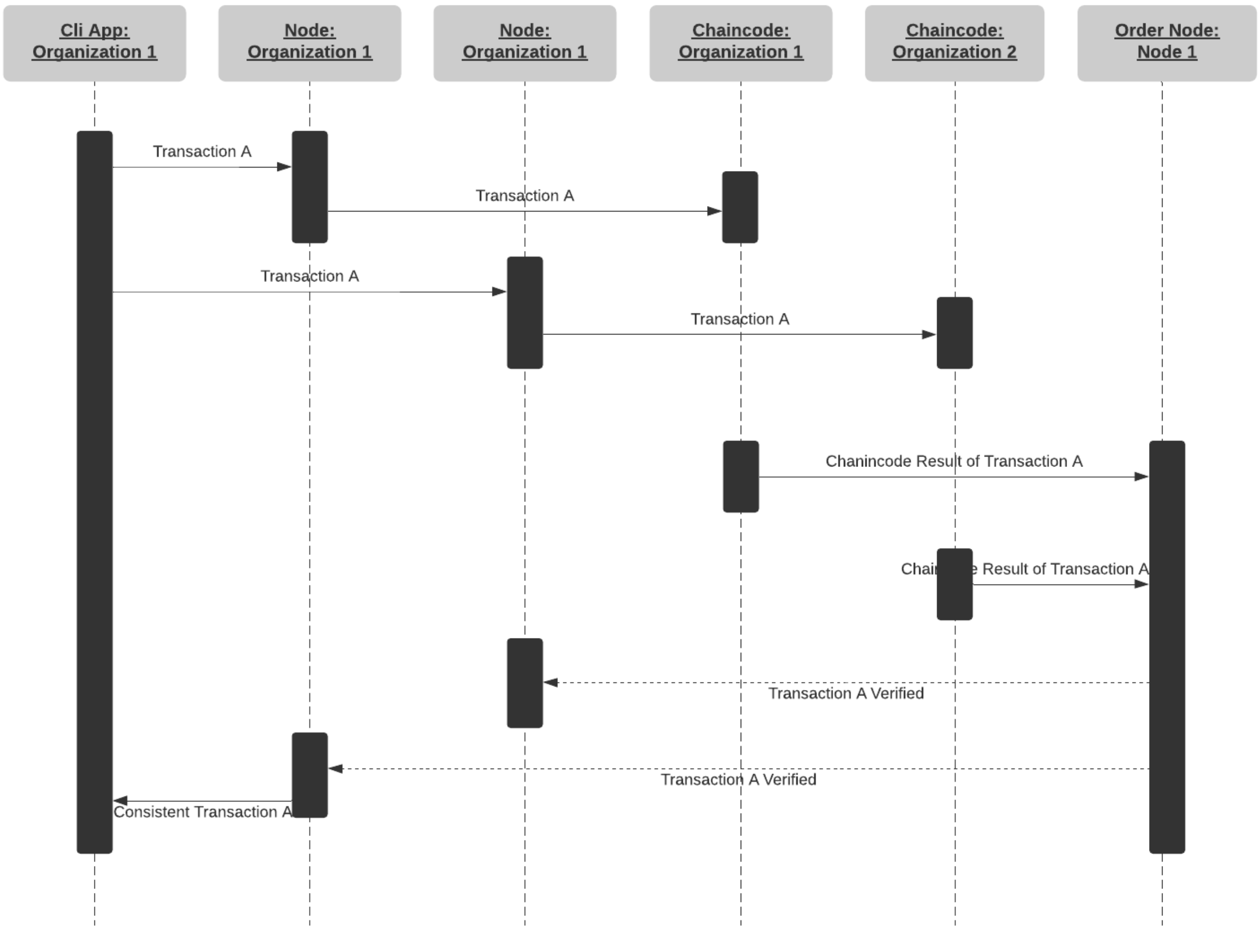

Figure 14. Communication Interaction between Nodes in FC Test Network

The FeatureCloud ledger consists of two parts, i.e., the world state and the blockchain. World state is a database, where all the current states of the business object are saved to enable quick access to the last value of business objects (e.g., consents) without having to parse all transactions in the blockchain. Furthermore, it becomes easy to for example query all consents by patient ID, or to recover a failed node based on the distributed blockchain's transactions. Furthermore, a consistent world state of newly joined nodes is also possible.
Note that only the valid transaction can trigger the changes to the world state database. If we take a closer look at the blockchain structure (Figure 14), we can see a block in blockchain consists of the block header, block data, and block metadata. The block header contains a block number, a hash of the current block, and a hash of the previous block header. Besides, a transaction should include a header, signature, proposal, response, and endorsement. The transaction header contains some metadata about this transaction. A digital signature ensures that the transaction hasn't been tampered with. The proposal encodes the parameters that trigger a function in the smart contract. The response saves the final verification result from the orderer node. Endorsement contains all responses from other nodes in the network.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

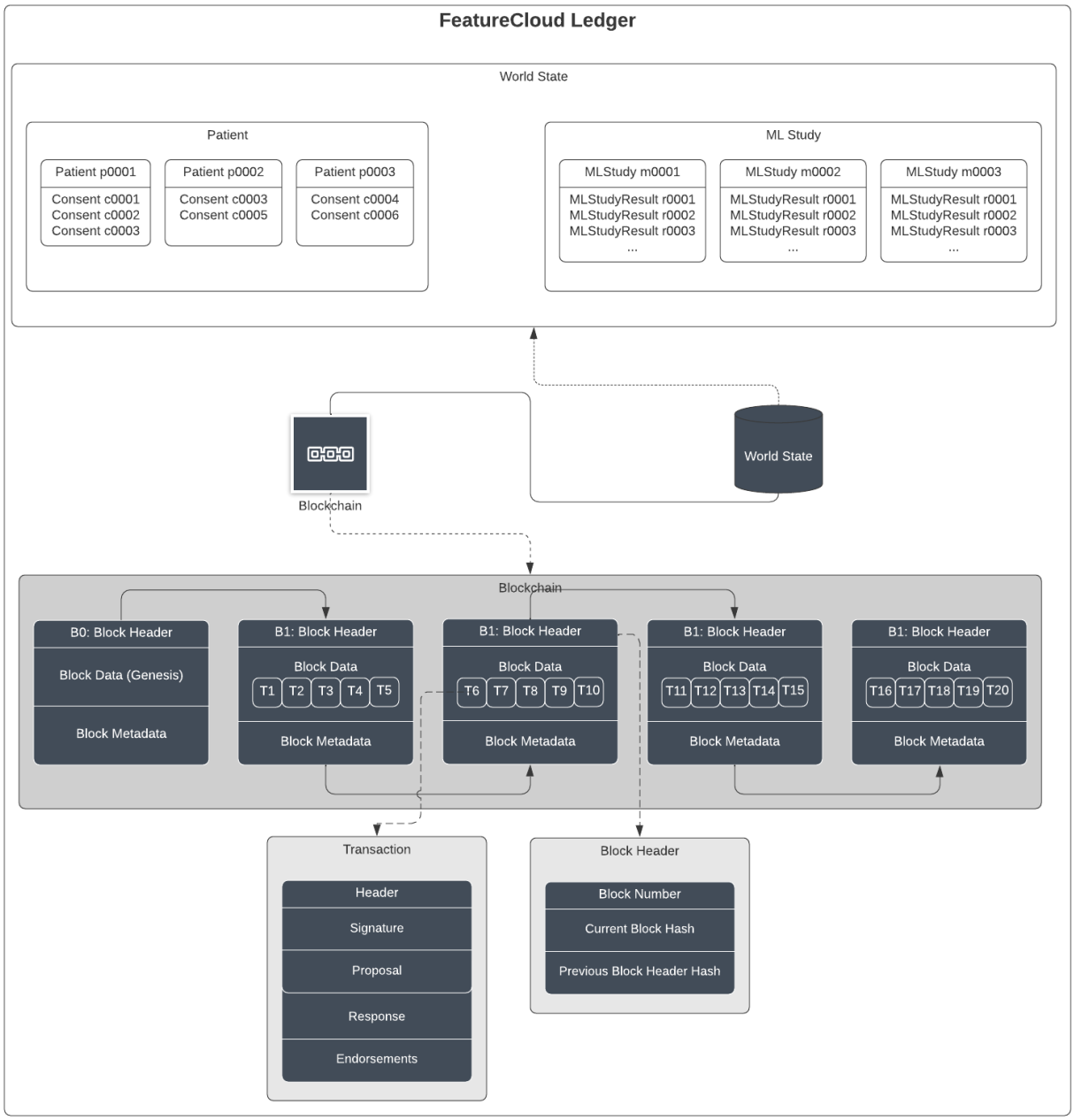

Figure 15. FeatureCloud Ledger

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

6 Open issues

The current PoC version has a limited functionality and implements the basic components, which will serve as basis for the further developments of a secure auditing process. In particular, the current implementation does not yet support access controls for patients, and currently assumes that participants act on behalf of patients to interact with the distributed ledger. Integrating Hyperledger Explorer[8] with the PoC, using well defined access controls for patients, will at least enable patients, auditors and also participants to monitor the transactions and consents.
As a first prototype, it still requires several internal improvements such as investigating i) how private data collections in Hyperledger fabric could help improving privacy in the context of FeatureCloud, ii) running tests with more nodes and on actual anonymized data, and iii) conduct usability studies with typical participants. Other functionalities may also be needed when integrating the PoC with the other components of FeatureCloud.
The current document serves as a base for delivering the future iterations of the PoC prototype, and thus should be seen as a living document that evolves together with the prototype.

7 Conclusion

Deliverable D6.4 proposes a proof-of-concept prototype for securing federated machine learning processes using blockchain technology, which serves as an immutable audit trail. The prototype uses Hyperledger Fabric where a test net has been deployed with few nodes acting as the root of trust. In the current version, only x.509 certificate-based identities are supported for authentication as they are inherently supported by Fabric. Two smart contracts were implemented and deployed for managing patient consents and securing ML study processes. The design relies on the compelling properties of blockchain to secure commitments to the execution of these private processes that can later be audited. In this regard, the utilization of BT does not prevent misbehavior, however it offers detectability and non-repudiation as a strong deterrent.
Note that this is a first prototype, with limited functionalities, and that more iterations are still required to improve the current version until it becomes ready for integration with the other components of FeatureCloud.

[8] https://www.hyperledger.org/use/explorer

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

8 References

Benchoufi, M., Porcher, R., Ravaud, P., 2017. Blockchain protocols in clinical trials: Transparency and traceability of consent. F1000Research 6.

ENISA, 2022. Digital Identity: Leveraging the SSI Concept to Build Trust [WWW Document]. ENISA. URL https://www.enisa.europa.eu/publications/digital-identity-leveraging-the-ssi-concept-to-build-trust (accessed 6.20.22).

Fdhila, W., Stifter, N., Kostal, K., Saglam, C., Sabadello, M., 2021. Methods for Decentralized Identities: Evaluation and Insights, in: González Enríquez, J., Debois, S., Fettke, P., Plebani, P., van de Weerd, I., Weber, I. (Eds.), Business Process Management: Blockchain and Robotic Process Automation Forum, Lecture Notes in Business Information Processing. Springer International Publishing, Cham, pp. 119–135. https://doi.org/10.1007/978-3-030-85867-4_9

Ghesmati, S., Fdhila, W., Weippl, E., 2021. Studying Bitcoin Privacy Attacks and Their Impact on Bitcoin-Based Identity Methods, in: González Enríquez, J., Debois, S., Fettke, P., Plebani, P., van de Weerd, I., Weber, I. (Eds.), Business Process Management: Blockchain and Robotic Process Automation Forum, Lecture Notes in Business Information Processing. Springer International Publishing, Cham, pp. 85–101. https://doi.org/10.1007/978-3-030-85867-4_7

He, L., Karimireddy, S.P., Jaggi, M., 2020. Secure byzantine-robust machine learning. ArXiv Prepr. ArXiv200604747.

Mugunthan, V., Rahman, R., Kagal, L., 2020. BlockFLow: An Accountable and Privacy-Preserving Solution for Federated Learning. ArXiv200703856 Cs Stat.

Passerat-Palmbach, J., Farnan, T., Miller, R., Gross, M.S., Flannery, H.L., Gleim, B., 2019. A blockchain-orchestrated federated learning architecture for healthcare consortia. ArXiv Prepr. ArXiv191012603.

Soenens, E., 2009. Identity Management Systems in Healthcare: The Issue of Patient Identifiers, in: Matyáš, V., Fischer-Hübner, S., Cvrček, D., Švenda, P. (Eds.), The Future of Identity in the Information Society, IFIP Advances in Information and Communication Technology. Springer, Berlin, Heidelberg, pp. 56–66. https://doi.org/10.1007/978-3-642-03315-5_4

Weng, Jiasi, Weng, Jian, Zhang, J., Li, M., Zhang, Y., Luo, W., 2019. Deepchain: Auditable and privacy-preserving deep learning with blockchain-based incentive. IEEE Trans. Dependable Secure Comput.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 826078.

9 Table of acronyms and definitions

ACL	Access Control List
CA	Certificate Authority
CID	Consent Identifier
concentris	concentris research management GmbH
CRUD	Create, Register, Update, Delete
DiD	Decentralized Identifier
IPFS	InterPlanetary File System
GDPR	General Data Protection Regulation
GND	Gnome Design SRL
LPM	Local Project Manager
ML	Machine Learning
MPC	Multi-Party Computation
MS	Milestone
MUG	Medizinische Universitaet Graz
Patients	In this deliverable, we use the term “patients” for all research subjects. In FeatureCloud, we will focus on patients, as this is already the most vulnerable case scenario and this is where most primary data is available to us. Admittedly, some research subjects participate in clinical trials but not as patients but as healthy individuals, usually on a voluntary basis and are therefore not dependent on the physicians who care for them. Thus, to increase readability, we simply refer to them as “patients”.
PID	Patient Identifier
PKI	Public Key Infrastructure
PoC	Proof of Concept
RI	Research Institute AG & Co. KG
SBA	SBA Research Gemeinnutzige GmbH
SDU	Syddansk Universitet
SSI	Self-Sovereign Identity
TA	Trusted Authority
TUM	Technische Universitaet Muenchen
UHAM	University of Hamburg
UM	Universiteit Maastricht
UMR	Philipps Universitaet Marburg
VC	Verifiable credential
W3C	World Wide Web Consortium
WP	Work package

This project has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No 826078.