What is the need to implement prescription verification at all?

In some countries, like in Italy, the way medical prescriptions are managed by the national health care system (NHS) poses a lot of issues. The following are the main identified limits:

**Anonymity and privacy of patient data**: medical data are managed in cleartext;

**Security of communications**: medical data are sent by email exposing it to phishing, malware or similar attacks;

**Information certification**: there isn’t a protocol in place to guarantee certification of the prescription, in terms of authenticity of the content and involvement of actors;

**Availability and resilience of the service**: current platforms are based on centralized applications and databases that are subjected to failures, attacks and routine maintenance unavailability;

**Counterfeiting of the medical prescriptions**: NHS deals with a lot of scams and false prescriptions uses, often facilitated by the weakness of the protocol;

**Paper based processes**: current processes are often cumbersome and not fully digital.

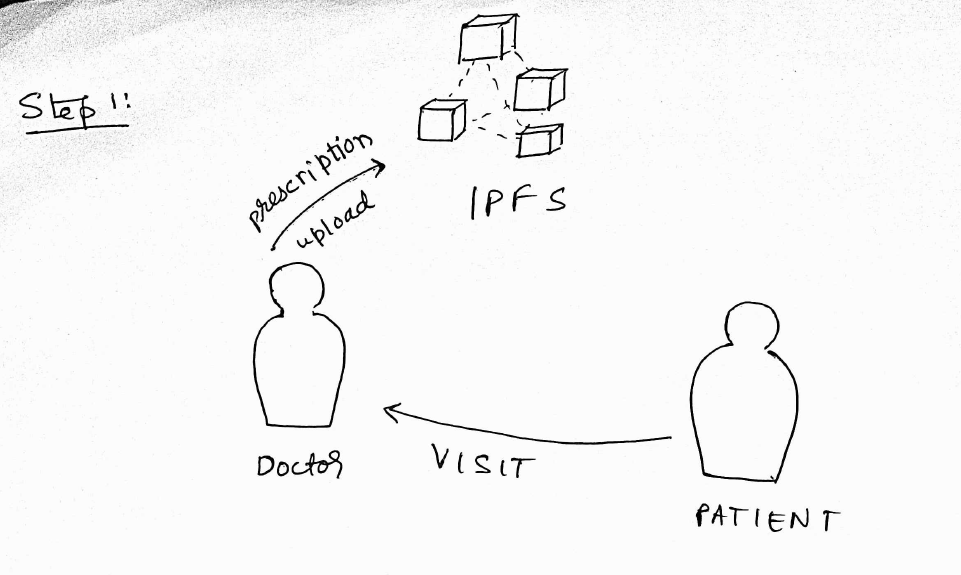
**Working:**

The Platform has 2 components:

1. Doctor
2. Patient
3. E-commerce pharmacy website

**Process of events**:

The patient who has some medical issue visits the doctor. The doctor examines the patient and prescribes them some medicines. This prescription is stored on IPFS with the patient name redacted, in the form of a JSON object. The corresponding content ID (cid) is held and stored in an ethereum smart contract. In what manner the cid is stored will be discussed later in this document.



For each prescription given out by the doctor we will generate a private key and its corresponding public key for both the doctor and the patient. We will then sign the cid with the private key of the doctor as well as the private key of the patient (we will refer to this pair of public and private keys as a layer 2 (L2) key pair). Note that after signing the cid a hash is generated. Since we are signing it twice, two hashes will be generated, one corresponding to the doctors L2 private key signing the cid and the other corresponds to the patient signing it with his L2 private key. Now we take the cid, hash corresponding to the doctors’ sign, hash corresponding to the patients sign as well as the L2 public key of both the doctor and the patient and store these 5 values on the blockchain. We will outline the exact structure of the smart contract further in this document.

Below we will describe the structure of the smart contract:

1.

struct prescription\_on\_chain {

        address publicKeyDoctor;

        address publicKeyPatient;

        string signedByDoctor;

        string signedByPatient;

        string cid;

    }

This is a struct to store the 5 values on the chain.

2. mapping(address => prescription\_on\_chain[]) patient\_to\_prescriptions;

This mapping maps the Patients L1 public key to an array of all his prescriptions.

3. function IssuePrescription(

        string memory \_cid,

        address \_publicKeyDoctor,

        string memory DoctorSign,

        address \_publicKeyPatient

        )

This functions is called by the doctor only and it assigns the parameters to their corresponding values in the struct prescription\_on\_chain

This struct is then appending to the array in the mapping patient\_to\_prescriptions

patient\_to\_prescriptions[\_publicKeyPatient].push(prescription\_local);

Note that in this function we aren’t adding the ‘signedByPatient’ variable in the struct because that is called by the patient where as, this function is called by the doctor.

4. function PatientSignCertificate(string memory \_cid,

                                    string memory PatientSign,

                                    address \_publicKeyPatient)

This function is called by the patient to update the ‘signedByPatient’ variable in the struct. To do we have to iterate through the prescription list of this patient until we find the right prescription. This is done by comparing ‘cid’. Once found a simple update is performed.

5. function getPrescriptionForVerification(address \_publicKeyPatient, string memory \_cid) view public returns(address publicKeyDoctor,

        address publicKeyPatient,

        string memory signedByDoctor,

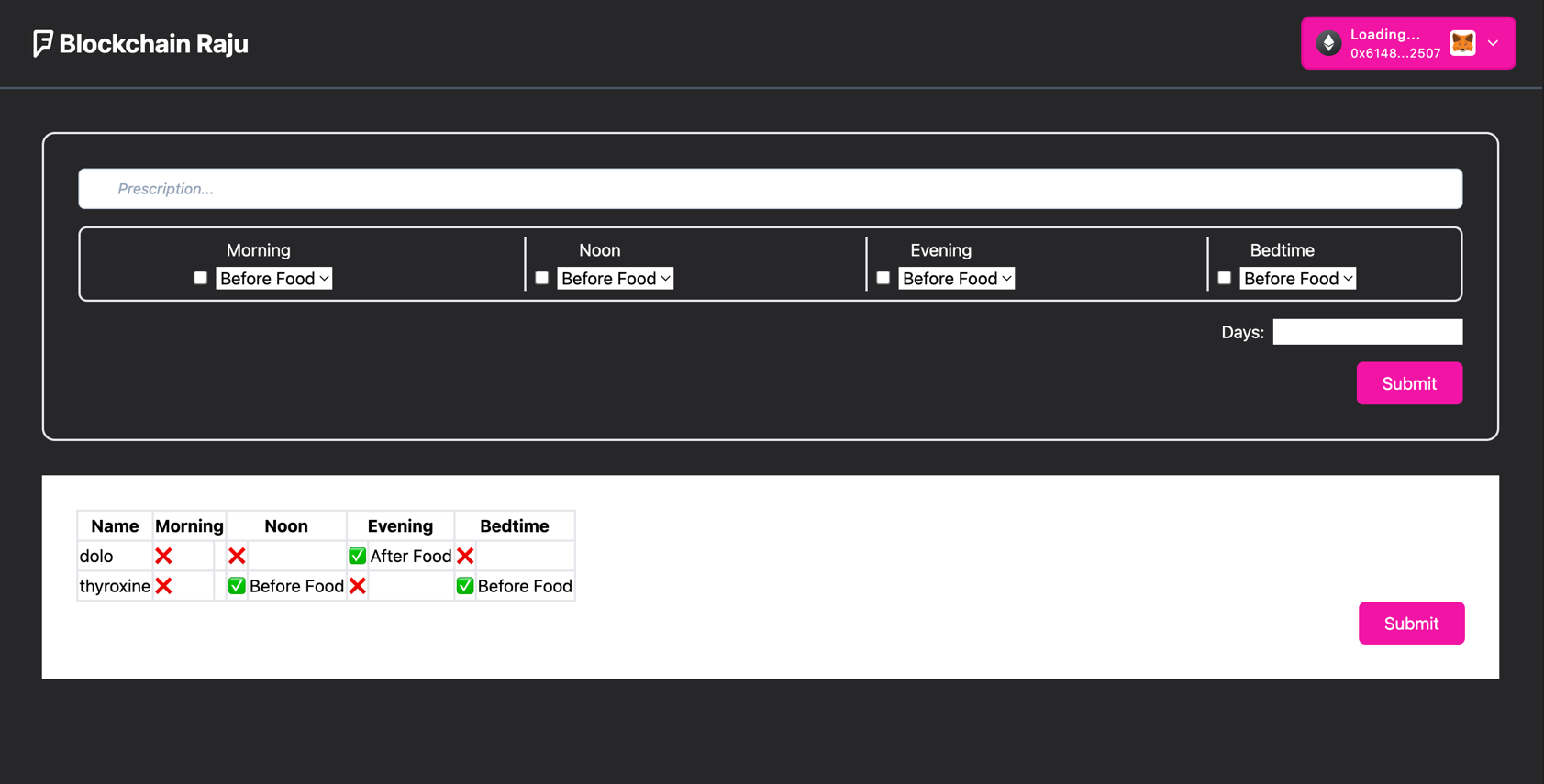
        string memory signedByPatient,

        string memory cid)

This function just returns the struct struct prescription\_on\_chain

The searching for the right prescription is done the same way as it was done in 4.

**Project Interface: - Doctor Platform**



***IPFS:***

* A blockchain which allows the user to store data in a decentralized manner. This is accomplished by breaking the data set into small chunks and storing them on different nodes of the network.
* Each chunk will have its own hash, and these hashes are linked to form of blockchain (a chain of links).
* This data can be accessed by using a unique hash called as “CID”, which is returned on storing the file in an IPFS network

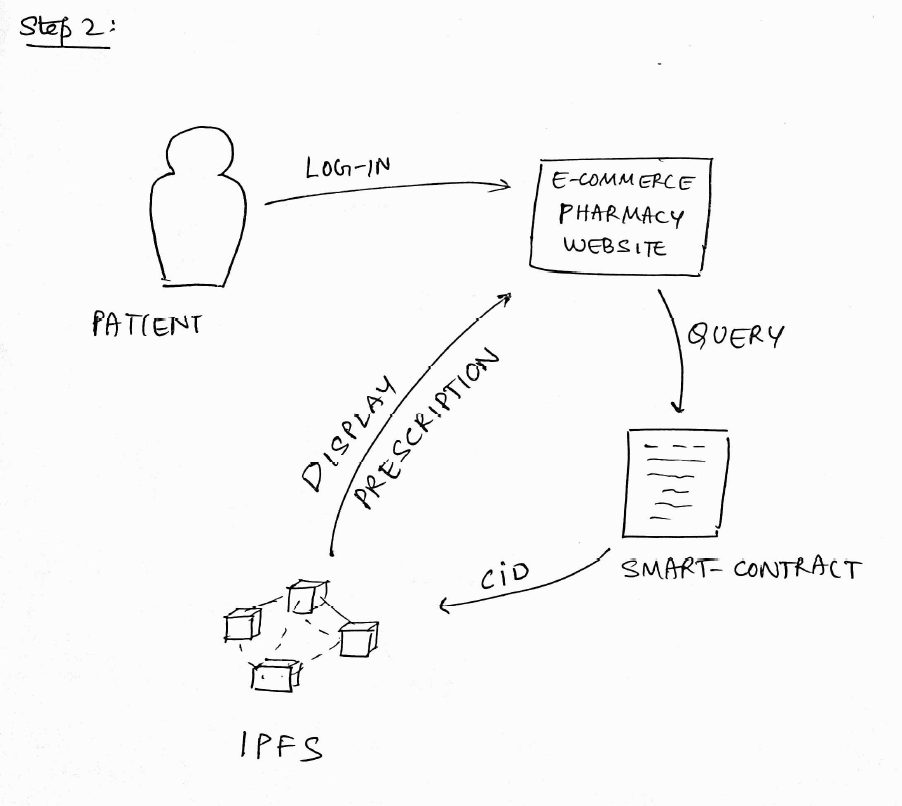
For each prescription created, a level two private and public key would be generated at both the Doctor and the Patient side. Whose use will be later specified in the document.

**QR-Scan:**

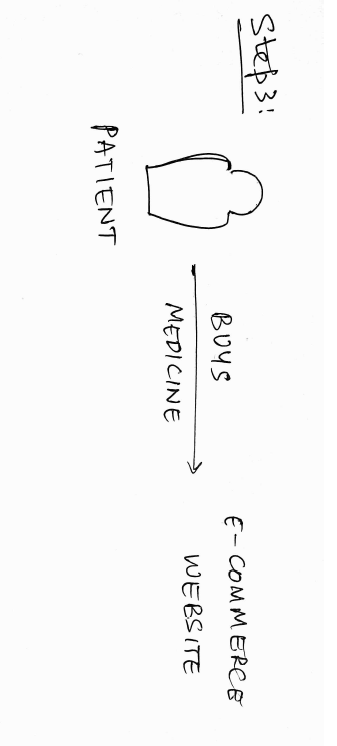
* As of now we have a set of public and private key on the patient side as well as on the doctor side.
* We have implemented a double verification: Where both the Patient and the Doctor must be physically present together, to scan the QR of each other.
* In one of the QR scan, the public key (level2) of the User would be transferred to the doctor. The doctor has its private key, he would use this private key to hash the “CID”. In the end we are trying to make
  + Public key of patient (L2)
  + Public key of doctor (L2)
  + CID value
  + Signed value of CID by private key of Doctor: Qualified doctor has verified the prescription
  + signed Value of CID by private key of patient
* **Note:** The 5th tuple value can only be signed at the patient end, so by doing this he/she is able to verify that they are the right person who has been prescribed the medicine in the prescription. Therefore, using the second QR scan, we are able to manipulate the 5th tuple

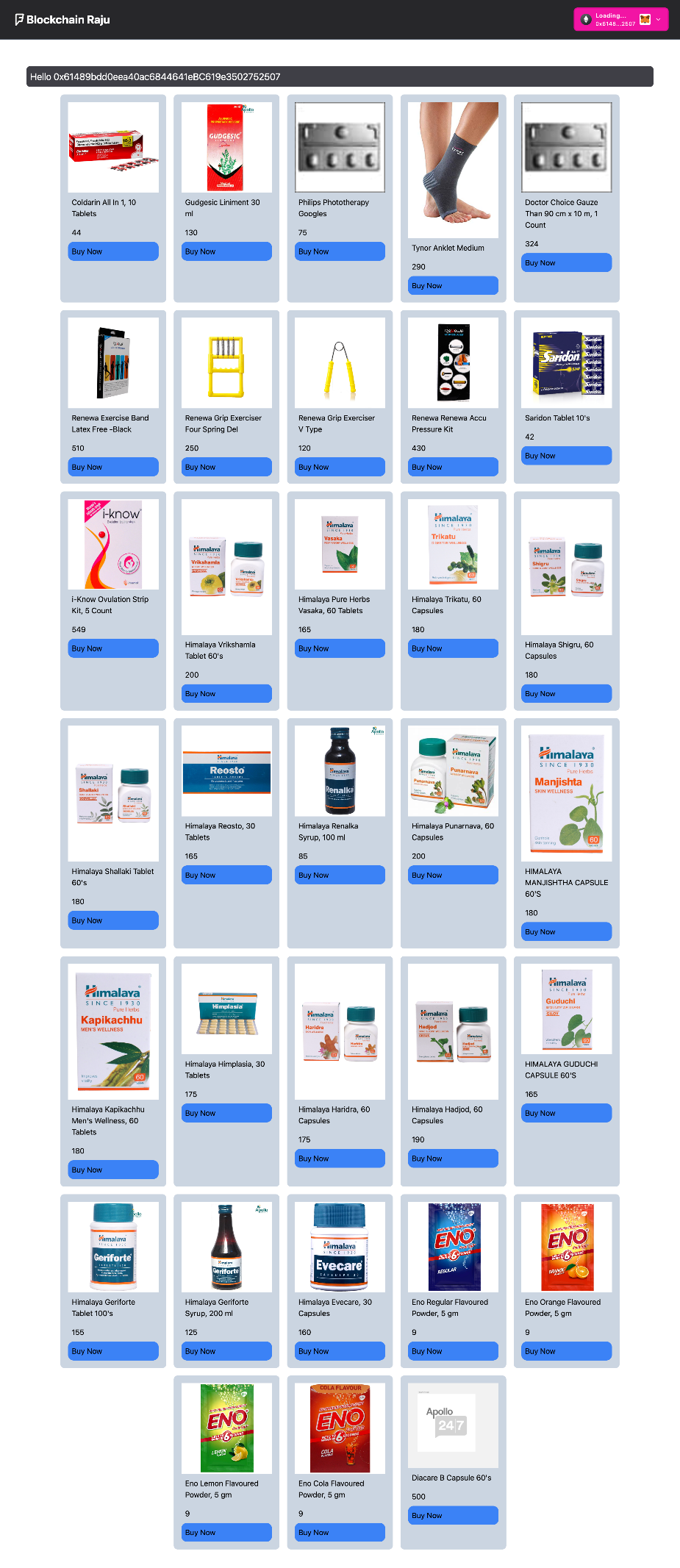
After the doctor has 4 elements in the tuple (everything excluding the signed value of CID by the private key of the User) the doctor is going to append the prescription corresponding to the public key of the user. Now, with the 2nd QR scan, the responsibility of user side verification falls on him where he appends the 5th element as mentioned above. This completes one cycle of prescription-addition to the database. [Relate this information to the smart contract details given above]

**Project Interface: - Patient Platform**

****

The prescription will contain the medicinal details like dosage etc. In addition to this it will also contain duration. Which means that this prescription is only valid for that many days after the prescription has been given (timestamped on blockchain).





**Conclusions:**

This project aims to be a tangible proof of concept, a prototype allowing to manage the entire life cycle of a medical prescription in a fully digital process, with its implementation on the Ethereum Blockchain, in which all involved parties (doctor, patient and pharmacist) interact with the Blockchain through their wallets, through a specific interface.

**Future Developments**:

* With an increasing use of our system, it could be possible to have a large dataset that could be analysed - through AI techniques for example – the definition of predictive models of drug consumption and mapping of related pathologies.