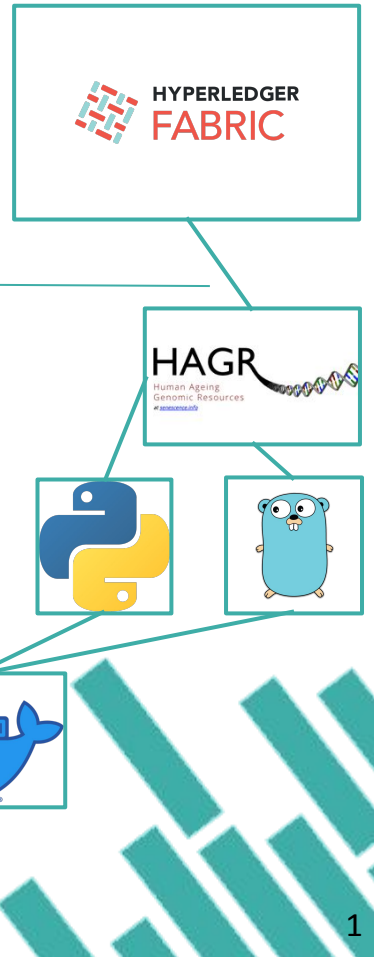
HyperLedger Fabric to store and share Human Ageing Genomics data



IBC Project - CSE 528 May 2021
Course Faculty - Dr. Donghoon Chang
Introduction to Blockchain and Cryptocurrency

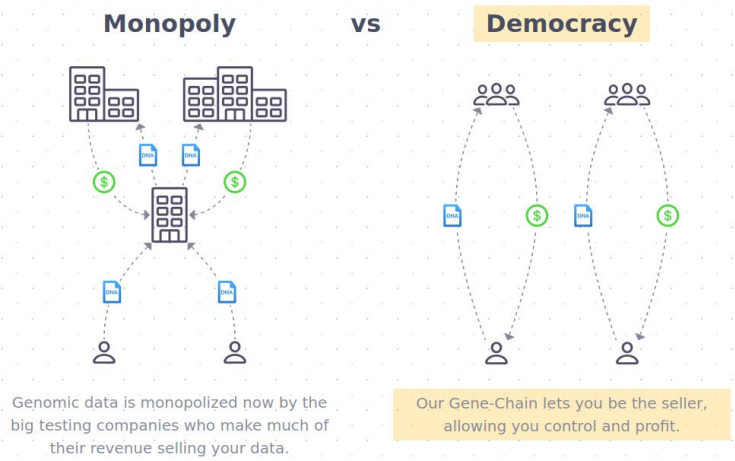
MT19213
Ghanendra Singh
MTech Computational Biology

INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

- Blockchain Startups in Genomics

Encrypgen

- Encrypgen develops software for genomic data, empowering patients and donors, facilitating health, business, and science.



Buy DNA tokens to buy DNA data

Bitcoin (BTC), DNA tokens, and soon other cryptocurrencies may be used to purchase genomic data, and other DNA products and health related services in the Gene-Chain Marketplace.

DNA tokens are an ERC-20 based cryptocurrency available on numerous crypto exchanges all over the world and exchangeable for Bitcoin (BTC), Ethereum (ETH) and several other altcoins.

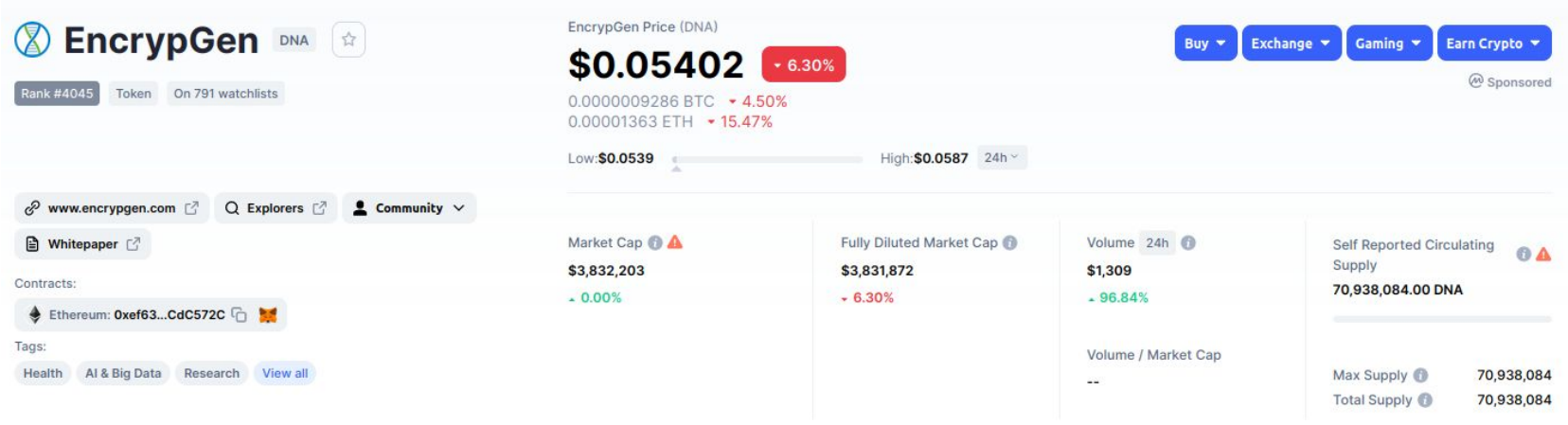
How it works

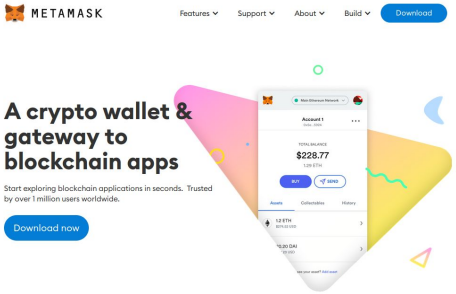
If you have had your DNA tested you may upload your raw DNA data file and create a Gene-Chain profile now.

EncrypGen de-identifies the raw DNA data file by stripping it away from name, email, and other sensitive information. DNA data buyers search Gene-Chain profiles suitable for their projects and purchase de-identified genomic data with DNA tokens.

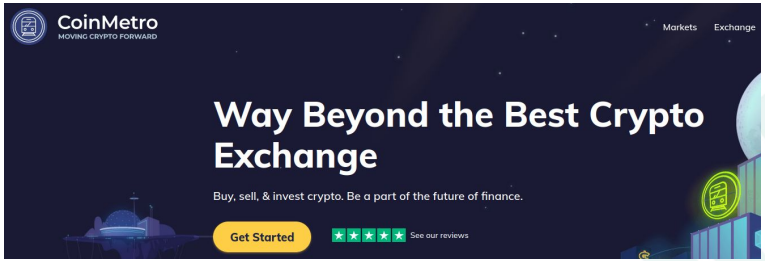
Ref: https://encrypgen.com/

Coinmarketcap and Crypto exchange





Ref: https://metamask.io/



Ref: https://coinmetro.com/

Ref:https://coinmarketcap.com/currencies/encrypgen/?utm_medium=widget&utm_campaign=cmcwidget&utm_source=encrypgen.com&utm_content=encrypgen

Zenome

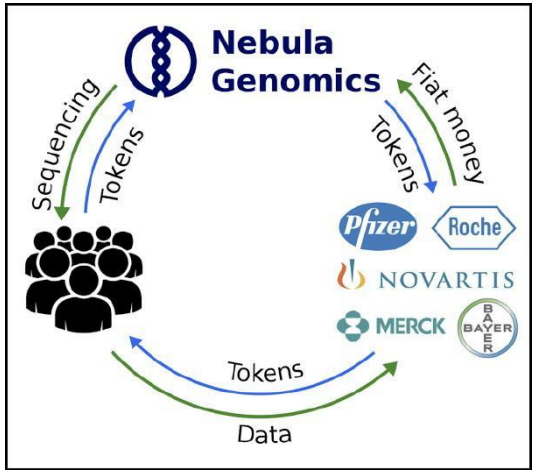
- Zenome plans to build a decentralized storage system for genomic data provided by network participants and supported financially with the help of internal cryptocurrency.

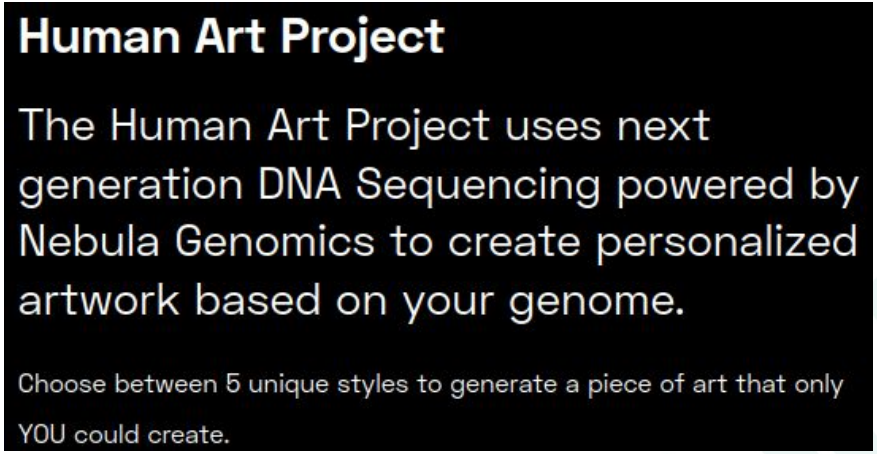
Ref: https://zenome.io/

Nebula Genomics

- We use Whole Genome Sequencing to decode 100% of your DNA and produce 10,000 times more data than other DNA tests like 23andMe and AncestryDNA.

Token flow

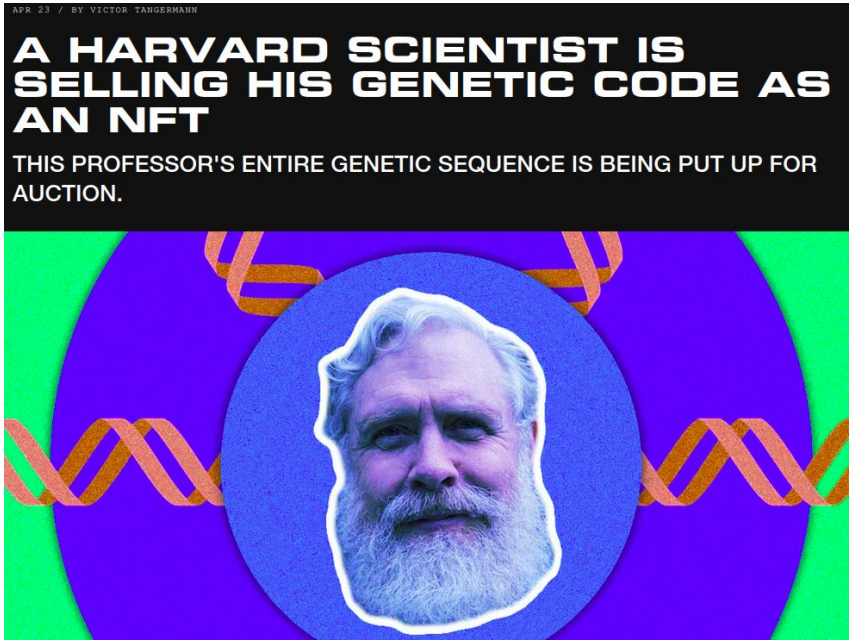




Ref: https://www.geneticsandsociety.org/biopolitical-times/personal-genomics-meets-blockchain

Nebula Genomics

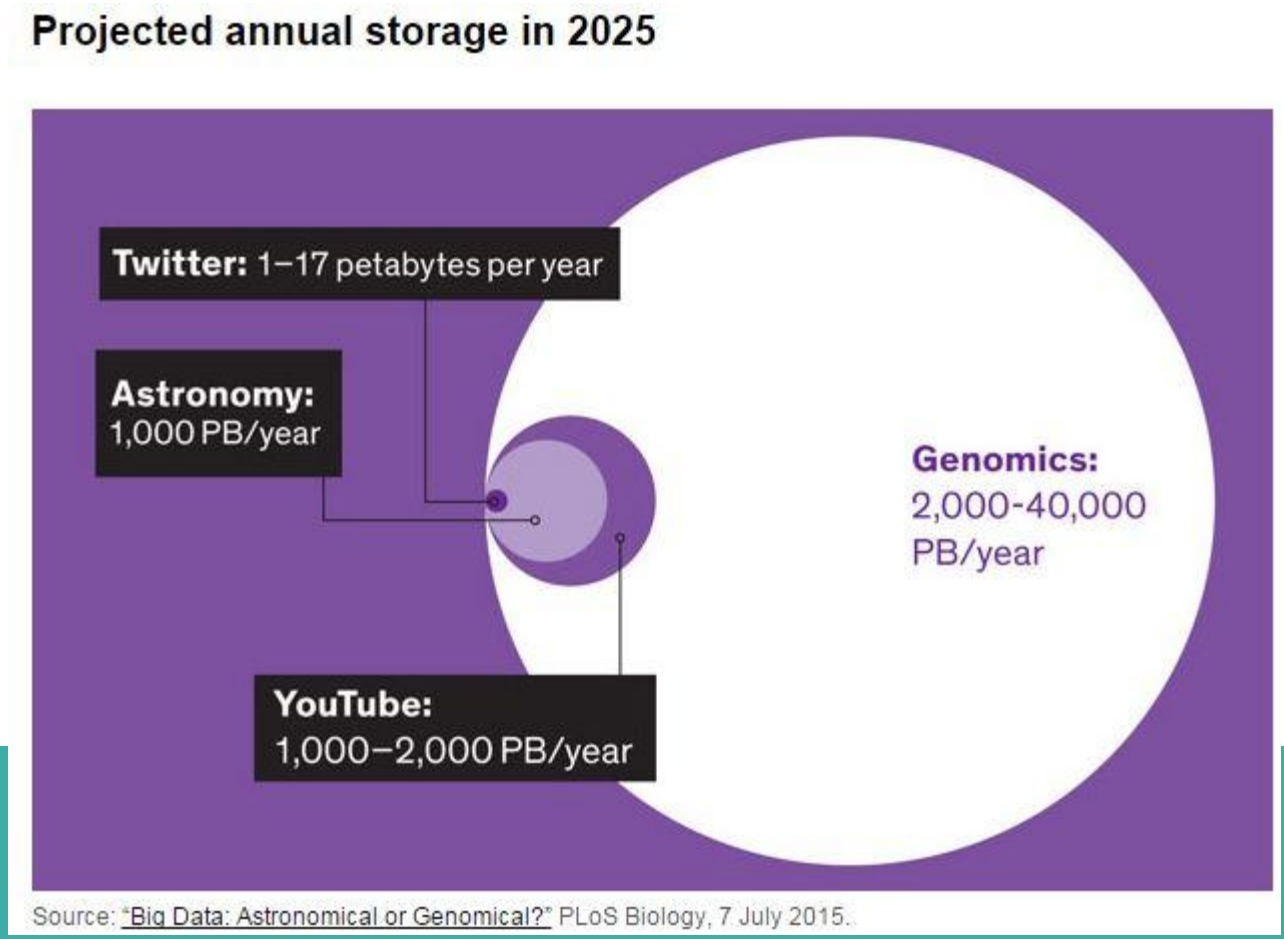
- 23rd Apr 2021



Ref: https://www.the-scientist.com/news-opinion/q-a-george-church-s-genome-up-for-auction-68682

Ref: https://futurism.com/neoscope/george-church-harvard-scientist-genetic-code-nft

- Why secure Genomics Data Storage Needed ?



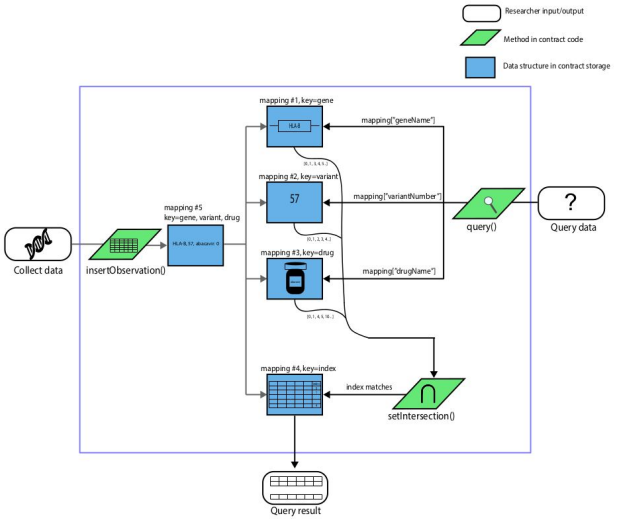
Research Paper

Technical Advance | Open Access | Published: 01 June 2020

Using Ethereum blockchain to store and query pharmacogenomics data via smart contracts

Gamze Gürsoy, Charlotte M. Brannon & Mark Gerstein

BMC Medical Genomics **13**, Article number: 74 (2020) | Cite this article



Ref: https://doi.org/10.1186/s12920-020-00732-x

Figure from Paper

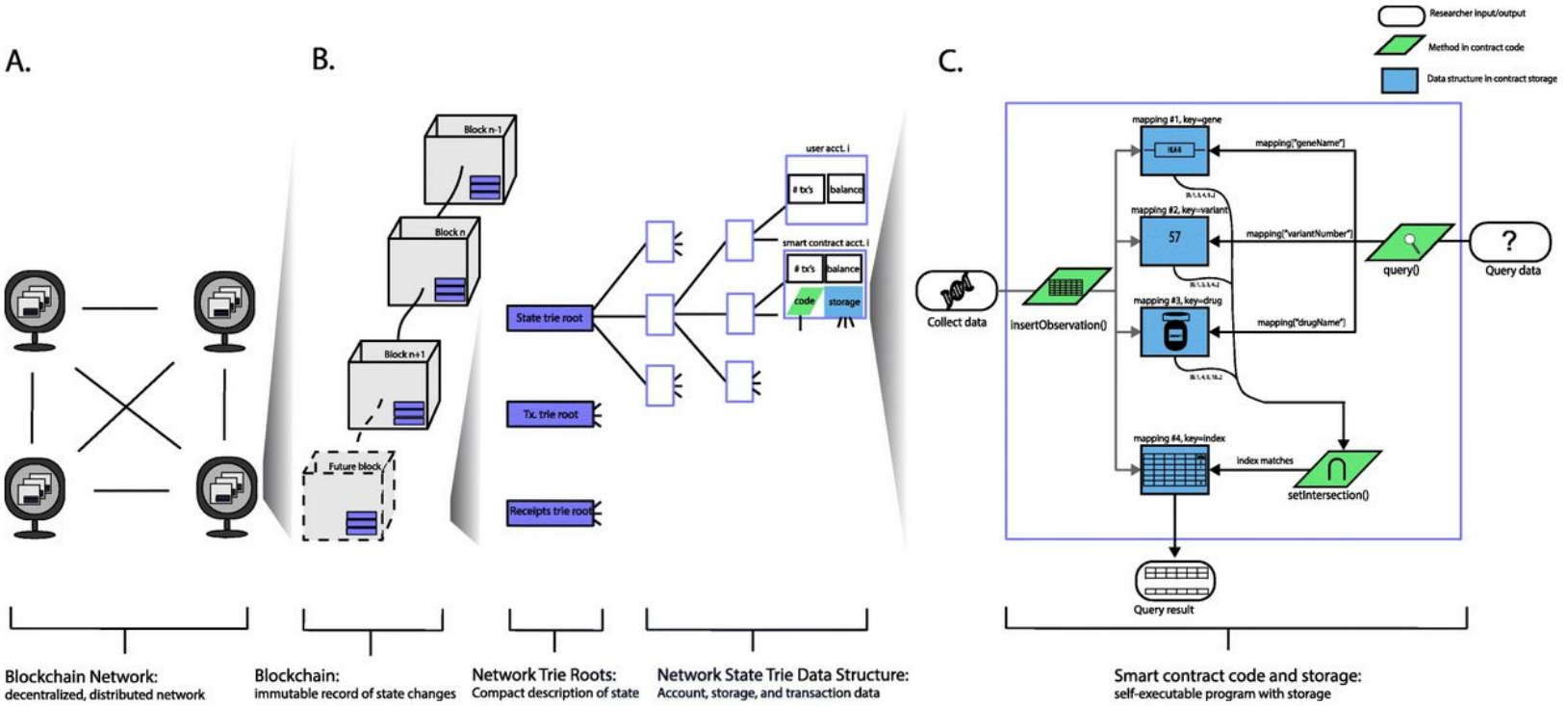
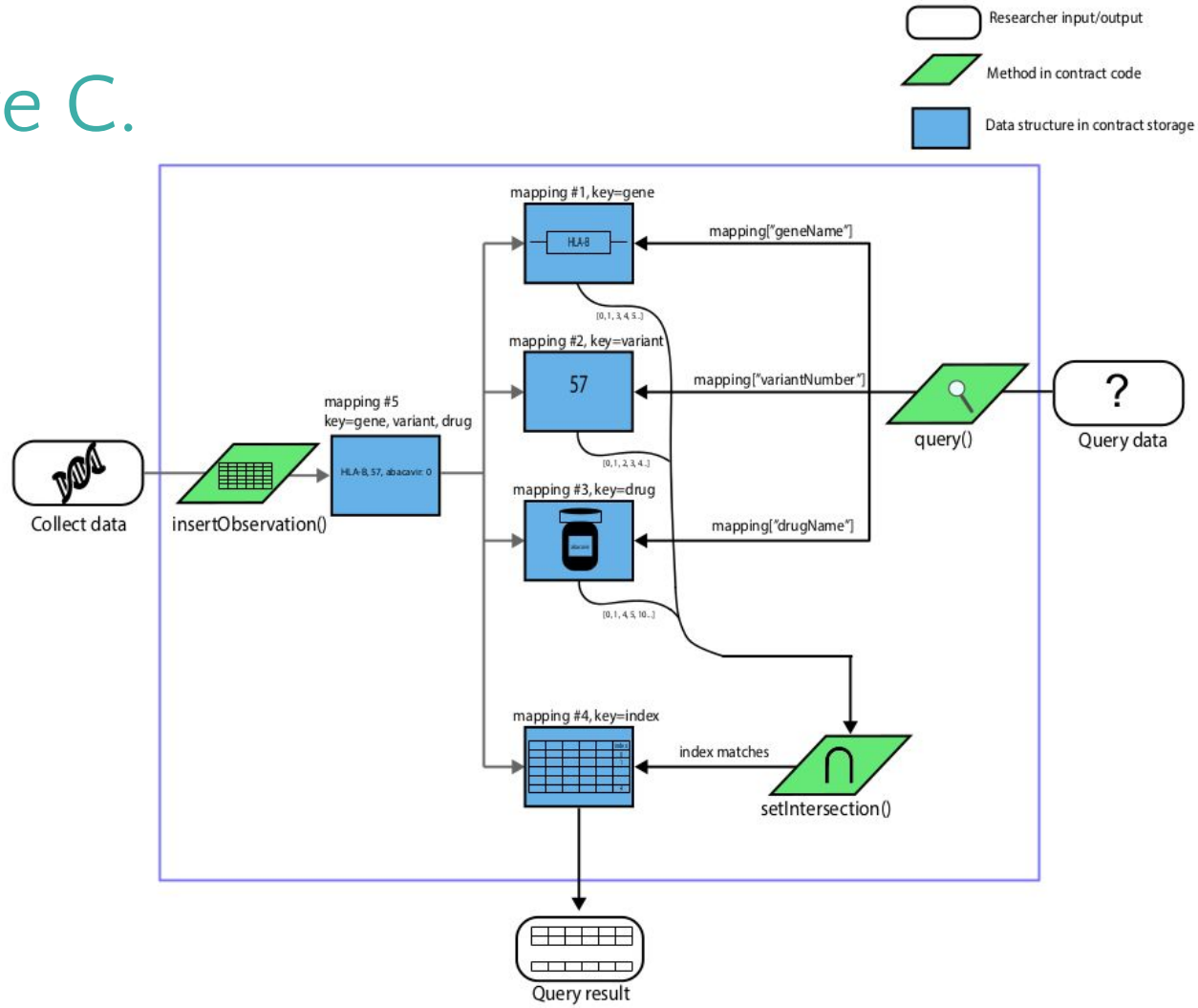


Figure C.


Researcher input/output
Method in contract code
Data structure in contract storage
mapping #1, key=gene
HLA-B
mapping["geneName"]
[0, 1, 3, 4, 5..]
mapping #2, key=variant
57
mapping["variantNumber"]
[0, 1, 2, 3, 4..]
mapping #5
key=gene, variant, drug
HLA-B, 57, abacavir: 0
mapping #3, key=drug
mapping["drugName"]
[0, 1, 4, 5, 10..]
mapping #4, key=index
index matches
setIntersection()
query()
?
Query data
Collect data
insertObservation()
Query result

Algorithms

Algorithm 1 Challenge Solution - Insertion

```
1: procedure INSERT(STRING gene, UINT variant, STRING drug, STRING outcome, BOOL relation, BOOL sideEffect)
2:     entry.push(gene, variant, drug, outcome, relation, sideEffect)
3:     if entry exists in database then
4:         uniqueEntries[i].push(gene, variant, drug)
5:     geneMapping[gene].push(ID)
6:     variantMapping[variant].push(ID)
7:     drugMapping[drug].push(ID)
8:     database[ID].push(entry)
9:     ID++
```

Algorithm 2 Challenge Solution - Query

```
1: procedure QUERY(STRING gene, STRING variant, STRING drug)
2:     idList = []
3:     results = []
4:     genes = []
5:     variants = []
6:     drugs = []
7:     if database is empty then
8:         return []
9:     else
10:        len ← number of fields queried (0, 1, 2, or 3)
11:        idList.push(IDs matching the query)
12:        if len is 0 then
13:            idList.push(all IDs in database)
14:        else
15:            genes.push(geneMapping[gene])
16:            variants.push(variantMapping[variant])
17:            drugs.push(drugMapping[drug])
18:            idList.push(intersection of IDs in genes,
19:            variants, and drugs)
20:        for i ≤ length(uniqueEntries in contract storage) do
21:            for j ≤ length(idList) do
22:                if idList[j] is ID of ith uniqueEntry then
23:                    results.push(database[j])
24:                    convert results counts to percentages
25:                j++
26:            i++
27:        return results
```

Algorithms

Algorithm 3 fastQuery Solution - Insertion

```
1:  procedure INSERT(STRING gene, UINT variant, STRING drug, STRING outcome, BOOL relation, BOOL sideEffect)
2:      entryIdentity.push(gene, variant, drug)
3:      entryData.push(outcome, relation, sideEffect)
4:      ID ← idKeeper[gene][variant][drug]
5:      if database[ID] exists then
6:          idKeeper[gene][variant][drug] ← counter
7:          ID ← counter
8:          database[ID].push(entryIdentity)
9:          genes[gene] ← counter
10:         variants[variant] ← counter
11:         drugs[drug] ← counter
12:         counter++
13:     update database[ID] counts based on entryData
```

Algorithm 4 fastQuery Solution - Query

```
1:  procedure QUERY(STRING gene, STRING variant, STRING drug)
2:      idList = []
3:      results = []
4:      genes = []
5:      variants = []
6:      drugs = []
7:      if database is empty then
8:          return []
9:      else
10:         len ← number of fields queried (0, 1, 2, or 3)
11:         idList.push(IDs matching the query)
12:         if len is 0 then
13:             idList.push(all IDs in database)
14:         else
15:             genes.push(geneMapping[gene])
16:             variants.push(variantMapping[variant])
17:             drugs.push(drugMapping[drug])
18:             idList.push(intersection of IDs in genes,
19:             variants, and drugs)
20:         for i ≤ length(idList) do
21:             results.push(database[i])
22:             convert results counts to percentages
23:             i++
24:         return results
```

- Project Design and Implementation

In detail, implementation is explained in IBC project reports.

Objective

- **To develop a robust method for storing, sharing and updating human aging genomics data among different entities like pharm companies, academics, researchers and healthcare professionals so that the data can be selectively shared while maintaining privacy.**

- Users own the data and can decide to share their genomics data anonymously, researchers get credit for their contributions, pharmaceutical companies get paid for their research and development.

Plan in Brief

March

- Ideation
- Fabric Fundamentals
- Project Framework
- CouchDB database creation

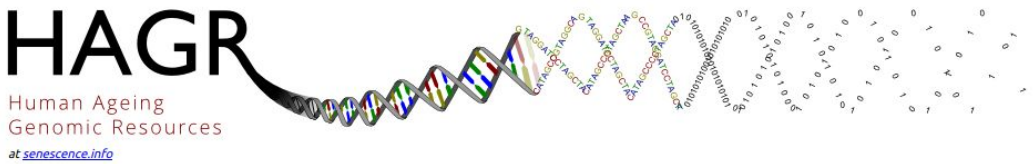
April

- Creation of Fabric Network
- Development of Smart Contract
- Fabric Security
- Deployment of Fabric Network

For plan described in detail please visit below reference link.

Ref: https://github.com/Ghanendra19213/IBC/blob/main/IBC%20Project%20Plan.pdf

Genomic Data



Databases

- GenAge Database of Ageing-Related Genes
- LongevityMap: Genetic association studies of longevity
- GenDR Database of Dietary Restriction-Related Genes
- DrugAge Database of Anti-Ageing Drugs

Why Aging data ?

In future nearly after 15 -20 years, research work in the field of ageing and longevity will mature enough to understand and help us to fight diseases and live healthier and longer lives as we age.

Ref: https://genomics.senescence.info/

GenAge: The Ageing Genes

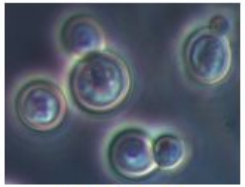
Benchmark database of genes related to ageing.

- GenAge is divided into genes related to longevity and/or ageing in model organisms (yeast, worms, flies, mice, etc.) and ageing related human genes.
- The section on human ageing-related genes includes the few genes directly related to ageing in humans plus the best candidate genes obtained from model organisms. Human genes are thus considerably better annotated and include more information.

Mouse
Download mouse genes
Download homologs from other model organisms

Fruit fly
Download fruit fly genes
Download homologs from other model organisms

Roundworm
Download roundworm genes
Download homologs from other model organisms



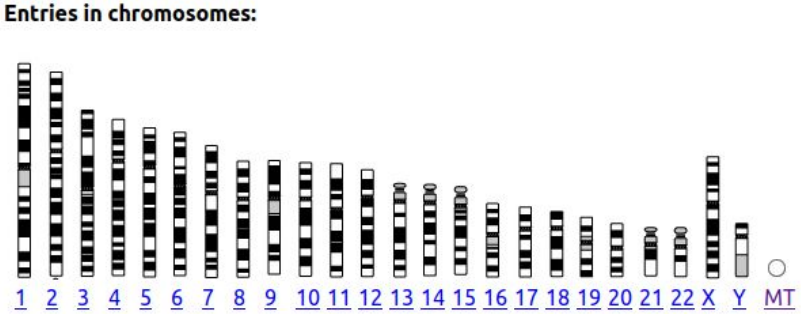
Baker's yeast
Download baker's yeast genes
Download homologs from other model organisms

Ref: Models Organisms

LongevityMap: Human Longevity Genes

A database of human genetic variants associated with **longevity**.

- Negative results are also included in the LongevityMap to provide visitors with as much information as possible regarding each gene and variant previously studied in context of longevity.
- LongevityMap serves as a repository of genetic association studies of longevity and reflects our current knowledge of the genetics of human longevity.



Longevity is the length of the life span independent of the biological aging process.

Ref: https://genomics.senescence.info/longevity/

GenDR: Dietary Restriction Genes

- Dietary restriction (DR), limiting nutrient intake from diet without causing malnutrition, retards age-related degeneration and extends lifespan in multiple organisms.

- DR induces multiple changes, yet its underlying mechanisms remain poorly understood. To facilitate research on the genetic and molecular mechanisms of DR-induced life-extension, we developed GenDR, a database of genes associated with DR.

Ref: https://genomics.senescence.info/diet/

DrugAge: Anti-Ageing Drugs

- DrugAge database contains an extensive compilation of drugs, compounds and supplements (including natural products and nutraceuticals) with anti-ageing properties that extend longevity in model organisms.

- Our focus is on drugs/compounds potentially impacting on ageing, and therefore drugs/compounds extending lifespan in disease-prone animals (e.g., cancer models) are excluded.

Ref: https://genomics.senescence.info/drugs/

HyperLedger Fabric

It is an enterprise-grade permissioned distributed ledger framework.

- Depending on the problem statement and type of interaction between parties involved in business, we define few key features.

Participants, Assets and Transactions.

Participants (Organisations like Pharma company, Healthcare).

Assets (Genomic data) which gets transferred between entities.

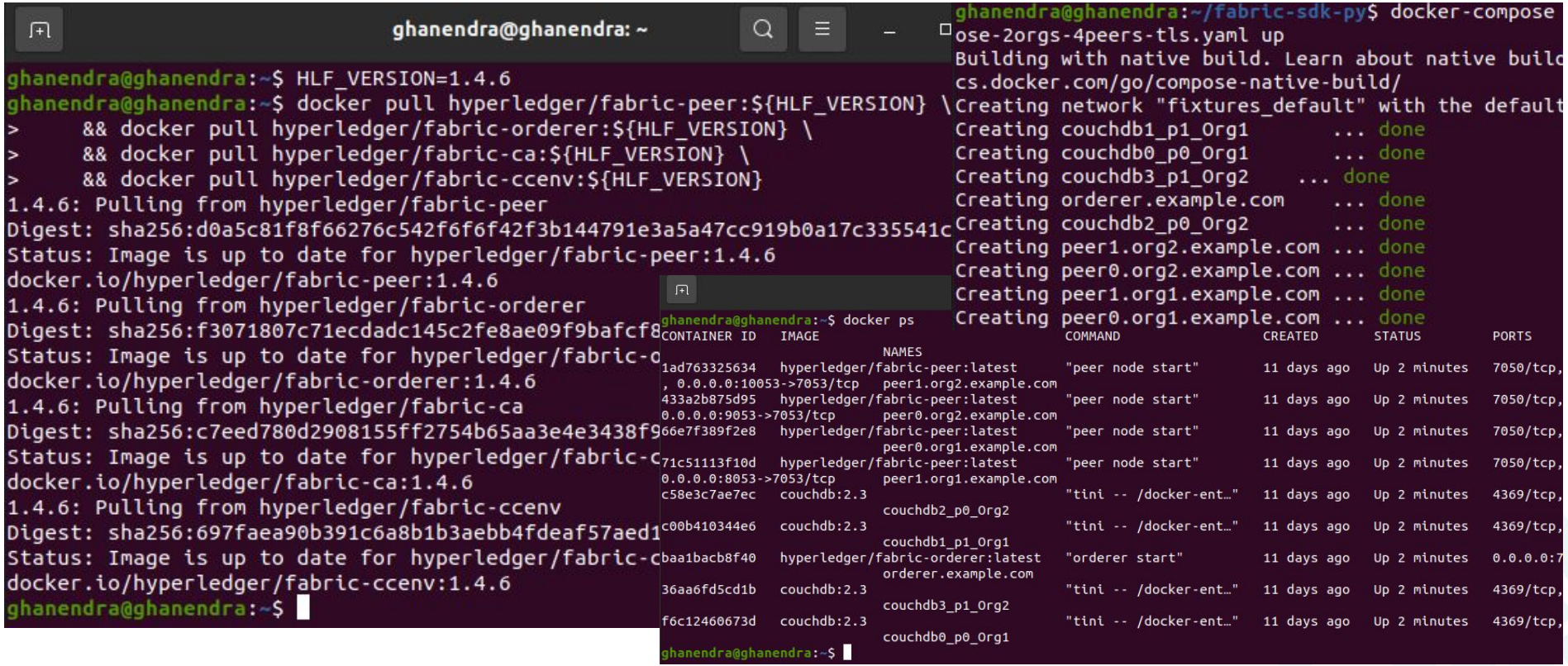
Transactions (Reading/ Writing, Querying genomic data) which change the state of the ledger.

Why Hyperledger Fabric ?

In the case of genomics data storage, it provides off the shelf selective private data access between participants using CouchDB to store rich data queries.

Ref: https://hyperledger-fabric.readthedocs.io/en/release-2.2/

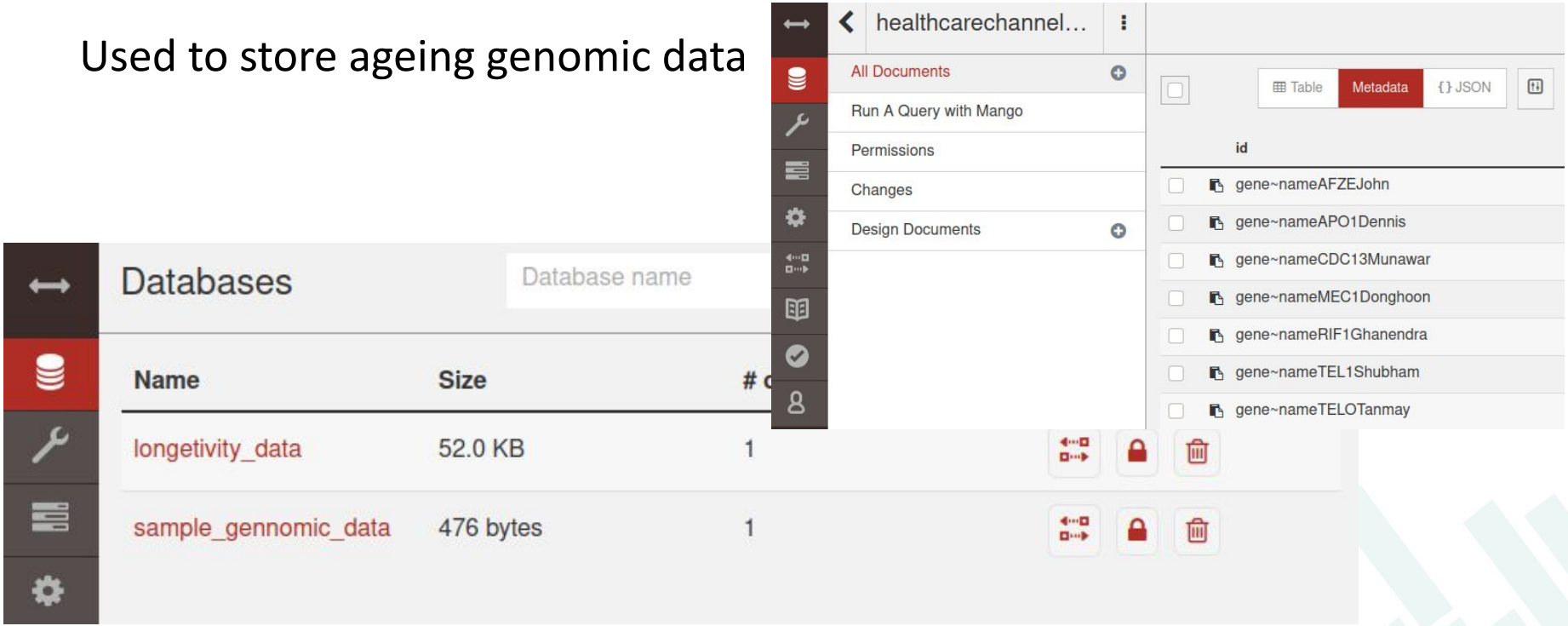
Docker and docker-compose



Ref: https://docs.docker.com/get-started/

CouchDB Database

Used to store ageing genomic data



Ref: https://couchdb.apache.org/

Ref: http://127.0.0.1:5984/_utils/

Smart Contract

```go
package main

import (
	"bytes"
	"encoding/json"
	"fmt"
	"strings"

	"github.com/hyperledger/fabric/core/chaincode/shim"
	pb "github.com/hyperledger/fabric/protos/peer"
)

// SimpleChaincode example simple Chaincode implementation
type SimpleChaincode struct {
}

type gene struct {
	ObjectType string `json:"docType"` //docType is used
	Id         int    `json:"id"`      //the fieldtags ar
	Name       string `json:name`
	Population string `json:"population"`
	Gene       string `json:"gene"`
	Size       int    `json:"size"`
}

type genePrivateDetails struct {
	ObjectType string `json:"docType"` //docType is used
	Name       string `json:"name"`    //the fieldtags ar
	Age        int    `json:"age"`
	Varient    string `json:"varient"`
	Price      int    `json:"price"`
}
```

Ref: [smartContract_genomic_cc.go]

```go
// Invoke - Our entry point for Invocations
// ========================================
func (t *SimpleChaincode) Invoke(stub shim.ChaincodeStubInterface) pb.Response {
	function, args := stub.GetFunctionAndParameters()
	fmt.Println("invoke is running " + function)

	// Handle different functions
	switch function {
	case "initGene":
		//create a new gene
		return t.initGene(stub, args)
	case "readGene":
		//read a gene data
		return t.readGene(stub, args)
	case "readGenePrivateDetails":
		//read a gene private details
		return t.readGenePrivateDetails(stub, args)
	case "transferGene":
		//change owner of a specific gene information
		return t.transferGene(stub, args)
	case "delete":
		//delete a gene
		return t.delete(stub, args)
	case "queryLongetivityMapByGene":
		//find genes from LongetivityMap for user X using rich query
		return t.queryLongetivityMapByGene(stub, args)
	case "queryAgeingDrugs":
		//find Ageing Drugs based on Compound name using rich query
		return t.queryAgeingDrugs(stub, args)
	case "getGenesByRange":
		//get genes based on range query
		return t.getGenesByRange(stub, args)
	default:
		//error
		fmt.Println("invoke did not find func: " + function)
		return shim.Error("Received unknown function invocation")
	}
}
```

Fabric Security

Used to store users enrolment credentials and certificates.

Use couchdbwalletStore to store credentials

```
In [7]:
# start first ca service before storing enrolement data.
from hfc.fabric_ca.caservice import ca_service
from hfc.fabric_network import couchdbwalletstore
```

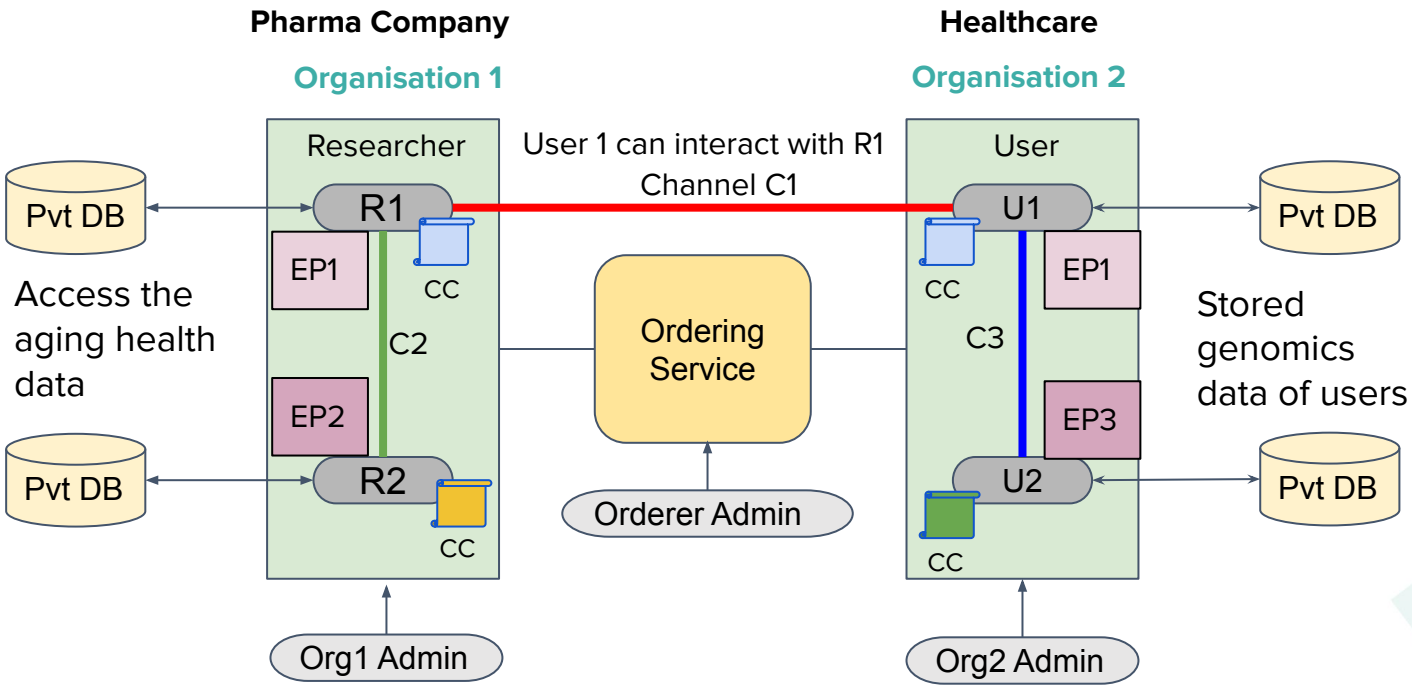
ghanendra@ghanendra: ~/fabric-sdk-py

```
fabric-ca | 2021/03/27 20:45:01 [INFO]    public key file location: /etc/hyperledger/fabric-ca-server/IssuerPublicKey
fabric-ca | 2021/03/27 20:45:01 [INFO] The Idemix issuer revocation public and secret key files already exist
fabric-ca | 2021/03/27 20:45:01 [INFO]    private key file location: /etc/hyperledger/fabric-ca-server/msp/keystore/IssuerRevocationPrivateKey
fabric-ca | 2021/03/27 20:45:01 [INFO]    public key file location: /etc/hyperledger/fabric-ca-server/IssuerRevocationPublicKey
fabric-ca | 2021/03/27 20:45:01 [INFO] Home directory for default CA: /etc/hyperledger/fabric-ca-server
In
fabric-ca | 2021/03/27 20:45:01 [INFO] Operation Server Listening on [::]:40581
fabric-ca | 2021/03/27 20:45:01 [INFO] Listening on http://0.0.0.0:7054
Out
fabric-ca | 2021/03/27 20:45:26 [INFO] signed certificate with serial number 197815981832100971458519712554531420015842158824
fabric-ca | 2021/03/27 20:45:26 [INFO] 172.17.0.1:59624 POST /api/v1/enroll 201 0 "OK"
In
fabric-ca | 2021/03/27 20:45:26 [INFO] 172.17.0.1:59628 POST /api/v1/register 500 0 "Registration of 'user2' failed: Identity 'user2' is already registered"
fabric-ca | 2021/03/27 20:45:37 [INFO] signed certificate with serial number 536037026501892266510563587510044423870602046952
Out
fabric-ca | 2021/03/27 20:45:38 [INFO] 172.17.0.1:59652 POST /api/v1/enroll 201 0 "OK"
fabric-ca | 2021/03/27 20:45:38 [INFO] 172.17.0.1:59656 POST /api/v1/register 500 0 "Registration of 'user3' failed: Identity 'user3' is already registered"
fabric-ca | 2021/03/27 20:45:50 [INFO] signed certificate with serial number 285845115234896762149419711568105381260281755 2
In
fabric-ca | 2021/03/27 20:45:51 [INFO] 172.17.0.1:59662 POST /api/v1/enroll 201 0 "OK"
Ou
fabric-ca | 2021/03/27 20:45:51 [INFO] 172.17.0.1:59666 POST /api/v1/register 201 0 "OK"
fabric-ca | 2021/03/27 20:45:51 [INFO] signed certificate with serial number 540175120214996901240207063803836597474968054102
fabric-ca | 2021/03/27 20:45:51 [INFO] 172.17.0.1:59670 POST /api/v1/enroll 201 0 "OK"
In
fabric-ca | 2021/03/27 20:45:51 [INFO] signed certificate with serial number 173962058786617856077937163269748590989128420764
fabric-ca | 2021/03/27 20:45:51 [INFO] 172.17.0.1:59674 POST /api/v1/reenroll 201 0 "OK"
Out
fabric-ca | 2021/03/27 20:45:51 [INFO] 172.17.0.1:59678 POST /api/v1/revoke 200 0 "OK"
```

```
In [10]:
cdb_ws.exists('Molly')
```

```
Out[10]: True
```

Fabric Network Architecture



Invoke and Instantiate transactions

Perform read/write and query data.

- Project Demonstration

Demo of fabric network components - Docker containers, chaincodes, transactions and couchDB data storage over zoom or google meet video.

References

Project Link: https://github.com/Ghanendra19213/IBC

- Hyperledger Fabric A Blockchain Platform for the Enterprise
- SideDB pptx file Privacy Enabled Ledger
- Tacutu, R., *et al.* (2018) "Human Ageing Genomic Resources: new and updated databases." *Nucleic Acids Research* **46**(D1):D1083-D1090
- Gürsoy, G., Brannon, C.M. & Gerstein, M. Using Ethereum blockchain to store and query pharmacogenomics data via smart contracts. *BMC Med Genomics* **13,** 74 (2020). https://doi.org/10.1186/s12920-020-00732-x
- Blockchain in Healthcare Today Blockchain in Healthcare Today.

Thanks

Block creation time ~ 1s.

Q?