**iDASH Privacy & Security Workshop 2023 – Secure Genome Analysis Competition**

**Track 2: Dynamic Patient Consent Management for Healthcare and**

**Genomic Research Data Sharing using Smart Contracts**

**README**

## Rules of Solution

1. A person can only be a member of one team.
2. Please include as comments in the submission file: team name, name and email of each team member.
3. Team membership cannot be changed after the submission due date.
4. If there is solution-wise communication/collaboration across teams, it needs to be acknowledged as comments in each team's submission file.
5. The solution should be able to store the data on one node of the blockchain network and retrieve on any other node, without any other method of communication between machines.
6. The solution should be symmetric (i.e., identical software for each node).
7. The single Solidity file should be standalone, such that upon being copied it should be able to read and write to an existing blockchain without any external setup or configuration.
8. Any external resources used (code snippets, 3rd party libraries, etc.) needs to be disclosed as comments in the single Solidity submission file, and the team must be compliant with the licensing requirements from both the source code providers and iDASH organizers.
9. No modification of the underlying Go-Ethereum source code.
10. Any changes to the function declarations in the Skeleton Code will disqualify the submission.
11. Multiple submissions are allowed before the due date. However, only the last submission will be evaluated.

## Evaluation

The efficiency of each solution will be evaluated as follows. First, we will insert patient consents from all data-contributing nodes for a certain time (e.g., 30 minutes). Then, we will query all nodes to verify that the insertions were done correctly with 100% accuracy. We will compute the average number of records inserted per second **(40%)**, the average time to run the Researcher Query (**40%**), and the average time to run the Patient Query (**20%**) as our final evaluation metric.

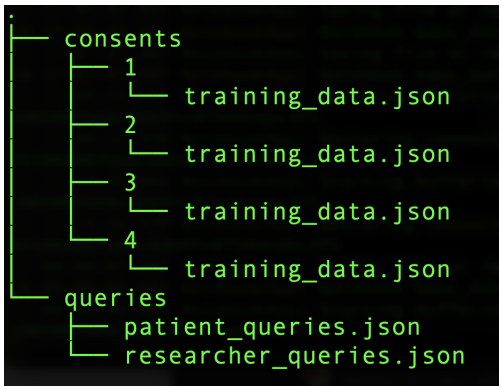
## Network Setting

We will be testing the solutions using 5 mining nodes (with 4 data-contributing nodes and 1 moderator node). See below for other details on network configurations.

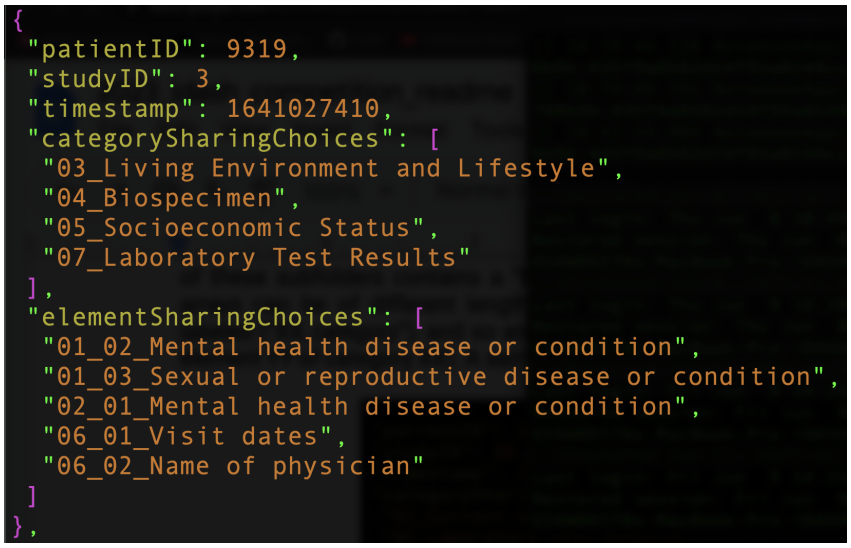
- Platform: Solc 0.8.12, Geth 1.10.7, Ubuntu 18.04.5
- Command to compile code: solc -o . --bin --abi --overwrite --evm-version byzantium DynamicConsent.sol (submitted solution must be compliable and runnable using this command)
- Gas limit = 0xFFFFFFF, gas price = 1, cliche period = 1 (subjected to change, yet the submitted solution must work with this setting)

## Data Format

The training dataset **Release** contains 2 folders, 1 for "**consents**" and 1 for "**queries**." See tree structure.




In the "**consents**" folder, there are 4 subfolders for 4 data-contributing nodes from 1 to 4. Each of these subfolders contains a "**training_data.json**" array of the consent JSON objects. The arrays can be of different length. The categories of shared health data will be numerically labeled ("*AA_category*"), and so are their corresponding elements ("*AA_BB_element*", referring to element *BB* of category *AA*). *AA* and *BB* are 2-digit numbers (e.g., "03"). A sample consent object is included.




In the "**queries**" folder, the "**researcher_queries.json**" file includes some sample query combinations for the Query for Researcher function. Similarly, the "**patient_queries.json**" file includes some sample query combinations for the Patient Query function. These queries may or may not be used for solution evaluation, and are only for example purposes.

Details on the Query for Researcher and Query for Patient functions are provided under **Skeleton Code**.

## Skeleton Code

We have provided a skeleton contract named **DynamicConsent.sol** which contains **3 functions** which must be completed. All testing will be through interaction with these functions. Participants may create other helper functions, however, they will not be tested, scored, or debugged by the organizer.

**Each of the core functions (storeRecord, queryForResearcher, queryForPatient) must maintain the provided function declarations (parameters may be renamed)**.

The submission should contain only a single file of uncompiled Solidity code. We will compile and deploy it ourselves. For initial development we recommend beginning by using Remix (remix.ethereum.org).

### WILDCARD Value

If a function parameter receives a value of **-1**, that means any value is accepted.

### Consent Insertion Function

*function storeRecord(uint256 _patientID, uint256 _studyID, uint256 _recordTime, string[] calldata _patientCategoryChoices, string[] calldata _patientElementChoices) public*

The *storeRecord* function pushes a single patient consent onto the blockchain.

*If a patient shares category X, they will share all elements of category X*.

See sample consent.

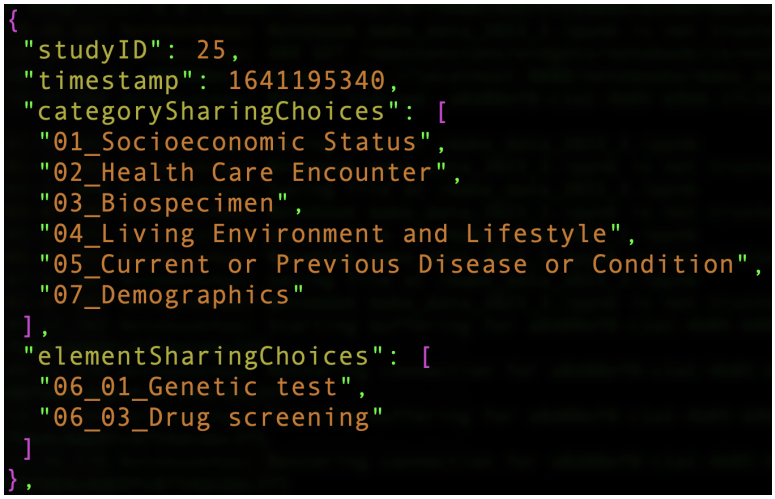

```
{
 "patientID": 2970,
 "studyID": 60,
 "timestamp": 1641026957,
 "categorySharingChoices": [
  "03_Current or Previous Disease or Condition",
  "05_Laboratory Test Results"
 ],
 "elementSharingChoices": [
  "01_04_Marital Status",
  "04_01_Substance abuse related disease or condition"
 ]
},
```

**Query for Researcher**

*function queryForResearcher(uint256 _studyID, int256 _endTime, string[] calldata _requestedCategoryChoices, string[] calldata _requestedElementChoices) public view returns(uint256[] memory)*

The *queryForResearcher* function inputs and output:

- Inputs: *studyID*, *timestamp*, array of requested health data categories *_requestedCategoryChoices*, and array of requested health data elements not already covered by the categories *_requestedElementChoices*.
    - Example



```
{
 "studyID": 25,
 "timestamp": 1641195340,
 "categorySharingChoices": [
  "01_Socioeconomic Status",
  "02_Health Care Encounter",
  "03_Biospecimen",
  "04_Living Environment and Lifestyle",
  "05_Current or Previous Disease or Condition",
  "07_Demographics"
 ],
 "elementSharingChoices": [
  "06_01_Genetic test",
  "06_03_Drug screening"
 ]
},
```

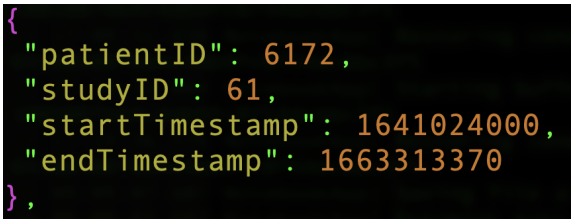
- Output: an array of all *patientIDs* who have consent to share ***at least*** the requested categories and elements with the specific *studyID*, and who have made such decisions ***at or before*** the requested *timestamp*.

**Query for Patient**

*queryForPatient(uint256 _patientID, int256 _studyID, int256 _startTime, int256 _endTime) public view returns(string memory)*

The *queryForPatient* function inputs and output:

- Inputs: *patientID*, *studyID*, *startTimestamp*, *endTimestamp*
  - Example inputs



```
{
 "patientID": 6172,
 "studyID": 61,
 "startTimestamp": 1641024000,
 "endTimestamp": 1663313370
},
```

- Output: a string that entails all patient consents for that specific *patientID* and *studyID*, which are made ***on or after*** *startTimestamp* and ***before or at*** *endTimestamp*. Fields within the same consent are separated by commas, while multiple consents are separated by line breaks.
  - Format of one single consent: *'studyID,recordTime,[patientCategoryChoices],[patientElementChoices]*\n*'*
    - For eg: '36,1641101813,[01_Contact Information],[02_04_Ethnicity]\n'

```
"29,1641625627,[01_Family Health History,02_Current or Previous Disease or Condition,03_Living Environment and Lifestyle,04_Biospecimen,05_Socioeconomic Status,06_Health Care Encounter,07_Laboratory Test Results],[]\n"
```

  - Format for multiple consents: *'studyID,recordTime,[patientCategoryChoices],[patientElementChoices]*\n*studyID,recordTime,[patientCategoryChoices],[patientElementChoices]*\n*'*
    - For eg: '36,1641101813,[01_Contact Information],[02_04_Ethnicity]\n36,1641101813,[01_Contact Information],[02_04_Ethnicity]\n'

```
"29,1641397465,[01_Family Health History],[]\n29,1641443790,[01_Family Health History,02_Current or Previous Disease or Condition,03_Living Environment and Lifestyle,04_Biospecimen,05_Socioeconomic Status,06_Health Care Encounter,07_Laboratory Test Results],[]\n29,1641625627,[01_Family Health History,02_Current or Previous Disease or Condition,03_Living Environment and Lifestyle,04_Biospecimen,05_Socioeconomic Status,06_Health Care Encounter,07_Laboratory Test Results],[]\n"
```